sinclair

QL

Manuale

LEGGERE QUESTE RIGHE PRIMA DI APRIRE L'INVOLUCRO

Il manuale del QL non è stato rilegato, per evitare danni durante il trasporto, e per permettere rapidi aggiornamenti. Oltre all'involucro contenente le pagine del manuale, vengono forniti un rilegatore con anelli e cartine divisorie.

Le cartine divisorie devono essere inserite nel rilegatore. L'ordine consigliato è il seguente:

Posizione	Cartina divisoria
Inizio	Introduzione
	Primo approccio
	Manuale
	Parole chiave
	Concetti
	QL Quill
	QL Abacus
	QL Archive
	QL Easel
Fine	Software Psion

In questo modo le etichette divisorie seguono un ordine logico, ma è possibile seguire un ordine differente, eliminando le parti che si pensa di non utilizzare.

Esaminando le varie sezioni, è possibile notare che ognuna comincia con un titolo posto sotto il logo Sinclair. Le pagine all'interno di ogni sezione sono state inserite nell'ordine corretto. Attenzione quindi a non mescolarle tra loro.

La SINCLAIR RESEARCH LIMITED è lieta di presentare la versione italiana del QL, del software e del manuale d'uso.

Abbiamo voluto realizzare non soltanto una guida tecnica al QL, ma anche introdurre i concetti fondamentali della microinformatica.

La SINCLAIR RESEARCH LIMITED è disponibile a ricevere suggerimenti per migliorare le successive edizioni.

DIREZIONE SOFTWARE

Per nessun motivo né la Sinclair Research Limited né la Psion Limited sono responsabili di eventuali danni diretti o indiretti, incidentali o consequenziali, o di eventuali perdite, derivanti da un qualsiasi errore, difetto o mancato funzionamento dell'Hardware e del Software del QI.

La Sinclair Research adotta un sistema di continuo sviluppo dei suoi prodotti. Si riserva quindi, tutti i diritti di modifica di manuali, hardware, software e firmware in qualsiasi momento e senza l'obbligo di darne notizia.

Manuale del QL. Prima edizione
Pubblicato dalla Sinclair Research Limited 1985
25 Willis Road, Cambridge

©Sinclair Research Limited
©Psion Limited

Nessuna parte di questo manuale può essere riprodotta in qualsiasi forma
senza un'autorizzazione scritta della Sinclair Research Limited

QL, QLUB, QLnet, Qdos e Microdrive sono marchi registrati
della Sinclair Research Limited

Quill, Archive, Easel e Abacus sono marchi registrati
della Psion Limited

sinclair

QL

Manuale

Introduzione
Primo approccio
Manuale
Parole chiave
Concetti
Software Psion
QL Quill
QL Abacus
QL Archive
QL Easel
Applicazioni avanzate

sinclair

QL

Introduzione

©1985 SINCLAIR RESEARCH LIMITED

Introduzione ai personal/home computer

Architettura di un PC

Più che una macchina il computer è un sistema. Il nucleo del sistema è l'unità centrale di elaborazione (Central Processing Unit, o CPU). Qui sono situati i circuiti che assicurano il suo funzionamento, gli forniscono la capacità di ricordare ed elaborano in modo logico i dati ricevuti, risolvendo i problemi posti dall'utente.

Attorno all'unità centrale gravitano diverse apparecchiature, che vengono chiamate unità periferiche. Cominciamo con la tastiera, che è simile a quella di una macchina per scrivere: serve infatti per scrivere le istruzioni, i dati, le operazioni che la macchina deve eseguire.

Tutto quello che l'operatore batte sui tasti appare su un'altra periferica, lo schemo video: questo consente di controllare i dati appena introdotti e le operazioni eseguite: ciò che appare sul video può essere corretto, cancellato, spostato, modificato. Inoltre lo schemo è utile anche quando si desidera gettare un'occhiata rapida su qualche informazione.

Tra le altre periferiche, quella più vicina, al punto che spesso si trova incorporata nell'unità centrale, è l'unità di memoria di massa, quella che gestisce gli "archivi": nastri o dischi magnetici in cui sono conservate sia le procedure di elaborazione con le quali il computer può compiere diverse operazioni sia le registrazioni del lavoro effettuato e dei risultati ottenuti.

Altre periferiche sono le stampanti: servono per trascrivere su carta quello che è stato elaborato e memorizzato. Altre ancora permettono di introdurre nell'unità centrale dati diversi dalle lettere e dai numeri (per esempio disegni). E ci sono periferiche con le quali questi possono essere riprodotti dopo l'elaborazione (plotter).

I collegamenti tra le varie parti del sistema vengono chiamati interfacce. Non si tratta di semplici cavi elettrici come quelli che, per esempio, in un impianto ad alta fedeltà collegano il giradischi all'amplificatore. Nel computer le apparecchiature sono di tipo completamente diverso le une alle altre: lavorano, cioè, con velocità e con codici differenti. L'interfaccia ha il compito di mantenere le diverse unità allo stesso ritmo e di tradurre il codice dell'una nel codice dell'altra.

Per poter funzionare, un elaboratore deve disporre, oltre che di una parte fisica, anche di una logica: il programma.

Il programma contiene una serie di ordini e li impartisce per mezzo di un codice. Al momento attuale, infatti, gli elaboratori non sono ancora tanto perfezionati da poter comprendere una lingua corrente.

Unità centrale di elaborazione, unità di memoria di massa, unità periferiche, interfacce, programmi e linguaggi costituiscono dunque il sistema del computer.

CPU

Il computer è diventato "personale" grazie al progresso della tecnologia elettronica dell'ultimo decennio e in particolare grazie all'invenzione del microprocessore (o microelaboratore): un complicatissimo circuito impresso con tecniche microscopiche su un quadratino di silicio di pochi millimetri quadrati. Il microprocessore assicura il funzionamento del computer ed esegue le operazioni logiche ed aritmetiche.

Accanto al microprocessore troviamo una memoria permanente (ROM = Read Only Memory, o memoria a sola lettura), in cui sono contenute le istruzioni per svolgere alcuni compiti specifici: controllare il sistema, interpretare le indicazioni in codice della tastiera e prepararsi a leggere i programmi di lavoro (programma di boot, cioè di autoavviamento).

L'unità centrale di elaborazione dispone di un'altra memoria, questa volta di tipo "labile": è la memoria operativa (RAM = Random-Access Memory, o memoria di accesso casuale). Possiamo descriverla grossolanamente come lo "spazio", la lavagna in cui si scrive, in cui vengono disposti provvisoriamente i termini del problema, le domande e le soluzioni.

ROM e RAM costituiscono dunque la memoria centrale del calcolatore. Quando leggete un opuscolo pubblicitario o parlate con un venditore di computer, scoprirete che RAM e ROM hanno "dimensioni" diverse a seconda dei diversi modelli e prodotti.

Possiamo immaginare la memoria come un casellario. Ogni casella contiene un solo carattere (cifra o lettera che l'operatore batte alla tastiera), sotto forma di simbolo numerico (sistema binario), chiamato byte. La capacità della memoria è quindi data dal numero di byte disponibili. Una RAM di 1 kbyte (1 chilobyte = 1,024 bytes) può maneggiare, tra programma e dati da elaborare, 1.024 lettere e/o numeri (l'equivalente di mezza pagina dattiloscritta). La RAM del QL di 128 kbyte (l'equivalente di 65 pagine dattiloscritte).

Si deve decidere quanto deve essere ampia la memoria centrale in base al tipo di lavoro che si intende far fare al QL: se deve maneggiare piccole quantità di dati, o eseguire programmi brevi, basterà una memoria modesta, mentre se il QL è chiamato a svolgere funzioni complesse, con programmi molto lunghi, ci vorrà una memoria di grandi dimensioni (256-512K).

I personal computer, del resto, si possono dividere per classi di prezzo anche in relazione alla capacità della memoria centrale ed il QL è Leader. Al momento dell'accensione il computer ha bisogno di un piccolo programma che gli consenta di autoavviarsi. È questo il programma di boot.

SISTEMA OPERATIVO

Per cominciare a funzionare veramente il computer ha bisogno anche del **sistema operativo** e di un programma che "traduce" le istruzioni date dall'utente alla tastiera. Il sistema operativo è un insieme di programmi che provvedono alla gestione delle operazioni standard dell'elaboratore. Poiché si tratta di servizi comuni a qualsiasi lavoro svolto dall'elaboratore alcuni computer li registrano permanentemente nella ROM (come il QL).

In altri, invece, vengono ripresi da memorie periferiche e riversati ogni volta nella RAM: questo significa che una parte dello "spazio-lavagna" verrà occupato da istruzioni e non sarà perciò disponibile per il lavoro da fare, ciò non è solo uno svantaggio: in questi computer c'è la possibilità di cambiare sistema operativo. Tutto ciò permette una flessibilità molto più estesa, perché la macchina può essere adattata per elaborazioni diverse.

Bisogna anche dire che i personal computer hanno in genere una RAM "espandibile": si possono cioè aggiungere quando si desidera una o più unità supplementare di memoria, secondo le esigenze dell'utente.

TASTIERA

Come una qualsiasi macchina ha i suoi comandi per essere manovrata, così il computer ha una tastiera: attraverso la tastiera si impartiscono ordini, si chiedono informazioni, si forniscono cifre, parole e segni da memorizzare. A volte la tastiera è incorporata nel blocco dell'unità centrale, ma spesso è staccata.

MONITOR/TV

Le indicazioni relative al programma in corso, così come tutto quello che l'operatore batte alla tastiera, appaiono istantaneamente sul monitor. Il monitor può essere in bianco e nero o a colori (bassa, media, alta risoluzione grafica), con caratteristiche speciali per rendere agevole la lettura e non affaticare la vista all'utente.

Sul video le lettere, i numeri e i diversi simboli grafici appaiono formati da un insieme di punti luminosi: maggiore è il numero dei punti, migliore è la leggibilità (si dice che, per essere leggibili chiaramente, i caratteri devono essere generati da una matrice di 5×7 punti in grado di produrre lettere con elementi "discendenti", come la g, la p, la q). Un altro fattore di leggibilità è la stabilità dell'immagine e la sua luminosità, che nei monitor di un certo livello è regolabile in funzione della luce ambiente.

Quando l'utente richiede al computer lavori di grafica deve poter disporre, oltre che del programma adatto alla generazione di immagini, di un **monitor ad alta risoluzione**, cioè un video capace di visualizzare punti luminosi molto piccoli: è questa la condizione essenziale per ottenere linee continue e non spezzettate, particolari nitidi e colori nella sfumatura desiderata.

In pratica questo è determinato dal numero di punti luminosi che possono essere visualizzati sul monitor. Generalmente si ritiene che uno schermo sia dotato di una sufficiente risoluzione quando può visualizzare almeno 250 punti in orizzontale e 200 in verticale.

MEMORIA DI MASSA

Come abbiamo visto nella prima parte della nostra presentazione del QL, i dati del lavoro da eseguire occupano la RAM, la memoria ad eccesso casuale, che abbiamo definito come una "lavagna" sempre disponibile a nuovi calcoli: a patto di essere cancellata ogni volta per lasciare lo spazio utilizzabile.

Il sistema di memoria di massa più diffuso sui personal computer, è l'unità floppy, una specie di "mangiadischi" in cui vengono inseriti dei dischetti flessibili chiamati floppy-disk e realizzati in materiale plastico molto resistente. Rivestiti di uno strato magnetico, i dischi ricevono la registrazione su piste concentriche: ma la testina di lettura può disporsi su una qualsiasi pista.

Il QL è espandibile con unità a floppy-disk della SINCLAIR, ma possiede un'altra memoria di massa: i cartridge, che si inseriscono nelle unità a microdrive. I floppy-disk sono unità più veloci ed affidabili dei microdrive e sono consigliabili per impieghi professionali del QL. I floppy-disk SINCLAIR sono perfettamente compatibili con qualsiasi programma realizzato su cartridge.

PERIFERICHE

Tra le periferiche, la stampante è la più impiegata. È una macchina che scrive sotto la dettatura elettronica dell'elaboratore, al quale è collegata mediante un'opportuna interfaccia.

Le stampanti comunemente disponibili sono di tre tipi: quella termica a matrice di punti, quella a impatto a matrice di punti e quella a margherita. La scelta tra questi diversi sistemi è importante per più motivi: ne dipende infatti la velocità, la qualità della scrittura e la versatilità.

La scelta deve essere oculata perché il prezzo d'acquisto è in alcuni casi abbastanza elevato. A differenza delle altre parti di un sistema, la stampante è infatti un'apparecchiatura "più meccanica" e quindi meno soggetta a diminuire il prezzo con i perfezionamènti introdotti dalla tecnologia elettronica. Oltre alla rapidità, la **stampante termica** presenta il vantaggio di essere assolutamente silenziosa. Viene quindi consigliata per operare in ambienti di lavoro dove macchine rumorose creerebbero disturbo. Il rovescio della medaglia è costituito dal prezzo della carta termosensibile, che è circa 10 volte maggiore di quello della carta comune.

Sempre con matrici ad aghi funzionano anche le **stampanti a impatto**: qui abbiamo una sola testina, che si sposta attraverso il foglio mentre gli aghi martellano la carta e imprimono i caratteri. Questa percussione permette di ottenere più di una copia per volta, cosa impossibile alle stampanti termiche. La velocità di esecuzione è ancora notevole: 100 caratteri al secondo, che sale a 140 e oltre nelle stampanti bi-direzionali. In queste ultime la testina non compie ritorni a vuoto ma stampa le righe alternativamente da sinistra a destra e da destra a sinistra, gestendo le aree vuote in modo ottimale.

E veniamo alla stampante margherita, che non ha bisogno di descrizioni data la sua diffusione: questo insieme di bacchette in forma circolare, come un fiore, porta in cima ai suoi petali flessibili un carattere tipografico che un dispositivo fa battere su un nastro inchiostrato e quindi sulla carta. Il modello margherita ha rivoluzionato le macchine per scrivere elettriche prima ed elettroniche poi.

Alcune di queste possono essere "interfacciate" cioè collegate al QL, realizzando quindi la scrittura automatica di stampati di ottima qualità. Staccando l'interfaccia, possono essere adoperate normalmente come macchine da scrivere.

Mentre la qualità della stampa è nettamente superiore con le macchine a margherita, la loro velocità è un po' inferiore: fino a 90 caratteri al secondo nei modelli bi-direzionali.

Modem

Quando il QL è collegato ad una certa distanza (non ha importanza quale, può variare da centinaia di metri a migliaia di chilometri) ad un altro computer (detto host-computer) il link avviene attraverso linee telefoniche. In qusto caso si utilizza una unità periferica, chiamata modem, che converte i segnali digitali provenienti dalla tastiera in segnali analogici che vengono trasmessi all'host-computer tramite il telefono. A sua volta l'host-computer è dotato di modem per tradurre inversamente i segnali da analogici in segnali digitali, interpretati poi dalla sua CPU e visualizzati sullo schermo.



Va ricordato che quando si utilizza un modem è necessario comporre il numero telefonico del computer che si vuole chiamare.

Plotter

Il plotter è un dispositivo per le stampe di grafici o testi ad alta qualità. Al contrario di una stampante, il suo utilizzo presuppone l'esistenza di un software ad hoc che guidi nella maniera più opportuna le penne. È possibile produrre anche lucidi tramite il plotter.

LINGUAGGI DI PROGRAMMAZIONE

Un elaboratore richiede istruzioni specifiche non ambigue per svolgere una qualsiasi funzione. La programmazione non è altro che la progettazione di una sequenza di istruzioni che il computer deve eseguire per svolgere una funzione.

I computer non possono però capire istruzioni in lingue naturali come inglese, italiano, tedesco e francese. Al contrario, le loro istruzioni devono essere espres-

se in uno dei linguaggi per computer. Questi differiscono dalle lingue naturali principalmente per il rigore della loro grammatica e per la povertà del loro vocabolario.

Il linguaggio di base dell'elaboratore è il linguaggio della macchina stessa, il microprocessore. Le istruzioni del linguaggio macchina sono costituite da insiemi di zeri e uno che in realtà rappresentano gruppi di impulsi elettrici (sistema binario).

Nessuno usa il linguaggio macchina direttamente; lavorare con insiemi di zeri e uno è troppo gravoso. Al contrario si usano altri linguaggi, che si avvicinano alle lingue naturali (solitamente Inglese) per alcuni gradi, e si traducono questi linguaggi in linguaggio macchina.

Il linguaggio del computer più vicino al linguaggio macchina è l'Assembler. Vi permette di usare parole di senso compiuto e lettere invece che numeri solamente, ma le istruzioni sono ancora orientate verso la logica dell'elaboratore piuttosto che secondo la logica corrente. Perciò per scrivere un programma in linguaggio assembler si deve conoscere come il computer lavora all'interno.

I linguaggi più semplici per scrivere programmi sono orientati verso problemi o procedure e comprendono il BASIC, il COBOL, il PASCAL, il FORTRAN, il C. ed altri. Ciascuna singola istruzione in questi linguaggi ad alto livello corrisponde a una sequenza di istruzioni nel linguaggio macchina, così i programmi possono essere più brevi e meno complessi.

Sintassi e logica

Tutti i linguaggi di programmazione hanno regole rigide di ortografia, di punteggiatura, di ordine di parole e simbologia in una istruizione. Prese insieme tali regole sono dette "sintassi". Alcune regole di sintassi sono naturali, ma altre richiedono di essere imparate a memoria. Per esempio il segno di più (+) rappresenta l'addizione, ma l'asterisco (*) rappresenta la moltiplicazione e la barra (/) rappresenta la divisione. La tastiera non ha i soliti simboli di × o di · per la moltiplicazione né il simbolo : per la divisione, cosicché sono stati scelti altri simboli.

Imparare tante regole di sintassi, però, non vi renderà un programmatore. Dovete piuttosto imparare ad analizzare il problema scindendolo in passi logici e scegliere le istruzioni appropriate che dicano all'elaboratore come affrontare ogni passo. Questa tecnica si impara con la pratica e ciò risulterà per alcuni più facile che non per altri.

BASIC

Il BASIC è il linguaggio ad alto livello più popolare tra quelli usati sui piccoli elaboratori. È un linguaggio generale che può consentire di risolvere ugualmente bene applicazioni negli affari e nella vita industriale, professionale e privata.

Il nucleo del BASIC fu sviluppato nel 1964 dal Darmouth College per le limitate richieste di programmazione di utenti che facevano parte di un unico grande sistema di computer. Il BASIC in origine usava telescriventi per trasferire dati, non tastiere o schermi ed era scarsamente predisposto all'uso di drive di dischi. Col passare degli anni dozzine di produttori di computer allargarono e arricchirono il linguaggio, cosicché ora può disegnare grafici, usare drive di dischi ed altri dispositivi periferici. Sfortunatamente non si è sviluppato nessuno standard per questi ampliamenti; ciascun fabbricante ha cambiato il BASIC in modo diverso. Il risultato è che il BASIC, come la maggior parte dei linguaggi naturali, ha numerosi dialetti talora incompatibili. Ecco perché un programma in BASIC per un calcolatore Apple non può girare sul QL. Comunque una volta imparata una versione del BASIC è facile imparare nuovi dettagli di un'altra versione.

APPLICAZIONI

Scrivere un programma per il QL non è un processo semplice come molti credono. Infatti il linguaggio di programmazione, molto spesso il BASIC interpretato, non è assolutamente destinato all'utente finale anche se è il più accessibile a chi voglia incominciare a risolvere in proprio i problemi più semplici. Richiede un appren-

dimento delle istruzioni di linguaggio che non è insuperabile come difficoltà ma che non è trascurabile in termini di tempo. Ciò non risolve tuttavia la questione in quanto il grosso ostacolo, insormontabile in tanti casi, consiste nella difficoltà logica di trasposizione del problema da risolvere in una struttura procedurale di comandi imperativi. Per costruire un programma devono essere usati comandi volti alla descrizione della sequenza di azioni necessarie a raggiungere il risultato (occorre specificare come si vuole ottenere qualcosa) e non, semplicemente alla descrizione di cosa si vuol avere. Questo approccio non è "naturale" e rende difficile apprendere. Occorre ancora attendere che vengano diffusi linguaggi naturali o, più concretamente, linguaggi descrittivi più vicini alla mentalità dell'utente finale .

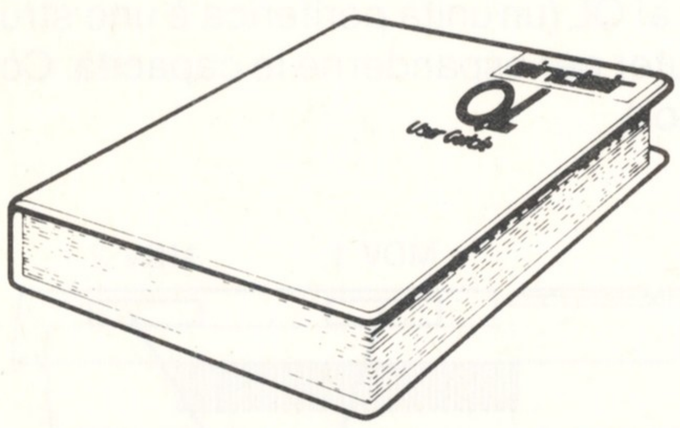
L'unico caso di "personal computing" veramente degno di questo nome, per cui la tecnologia software ha già predisposto strumenti realmente alla portata di tutti, è la gestione tabellare di dati. È un tipico problema d'ufficio, la gestione estemporanea di prospetti di dati numerici in forma di righe e colonne, caratterizzata da variabilità dei dati o delle relazioni tra dati e risultati o infine delle applicazioni. In particolare essa è caratteristica dell'attività lavorativa in ambienti finanziari, commerciali e di marketing dove sono estremamente frequenti le esigenze di stilare previsioni di incassi/esborsi o di vendite, definire piani di attività commerciali, elaborare consuntivi, preventivi, budget, stime di costi, effettuare valutazioni di investimenti, analisi di alternative di gestione, analisi statistiche, piani di ammortamento, listici, etc., etc.

Qui l'impiegato, il professionista, il manager, è abituato a ragionare, fare previsioni, prendere decisioni, eseguire calcoli su dati in forma di matrice o tabella su un foglio di carta; spessissimo però si trova di fronte all'esigenza di risolvere problemi di questa natura soprattutto senza poter richiedere l'aiuto mai tempestivo del servizio EDP oppure con la prospettiva di dover effettuare da solo (o con la collaborazione di una segretaria, spesso impegnata in altre attività) una gran mole di calcoli a mano o con la calcolatrice per provare varie alternative fino a raggiungere il risultato desiderato.

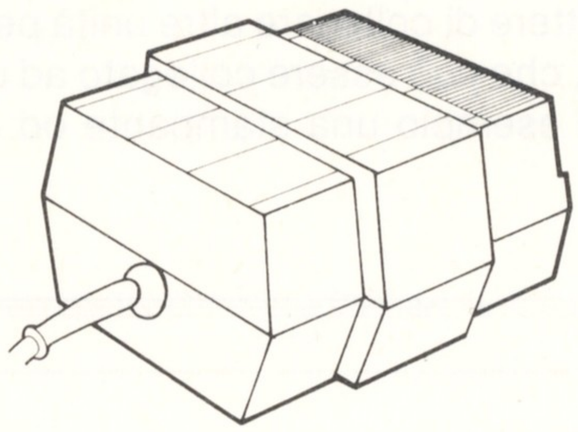
Esistono sul QL 4 programmi in grado di soddisfare tutte queste necessità. Per utilizzare i 4 prodotti, che alcuni chiamano di supporto alle decisioni o DSS (Decision Support Systems), bastano poche ore di addestramento autonomo (al personal computer con manuale a fianco) per sfruttarli già al meglio. Non necessitano lunghi e costosi corsi di addestramento presso il fornitore. L'apprendimento può essere programmato ed effettuato secondo i ritmi e le esigenze individuali. Anche per chi non faccia uso continuo di questi strumenti non c'è pericolo di dimenticare l'operatività, salvo forse in qualche dettaglio comunque recuperabile sul manuale o sul video. Infine essi sono disponibili direttamente a chi ha il QL quasi nello stesso tempo che impiega per mettere mano alla penna e alla calcolatrice. Il tempo di accendere la macchina, inserire un dischetto, battere un comando ed il QL si trasforma in un potente strumento previsionale, decisionale e di calcolo.

INTRODUZIONE AL QL

La scatola del QL contiene:



IL MANUALE DEL QL



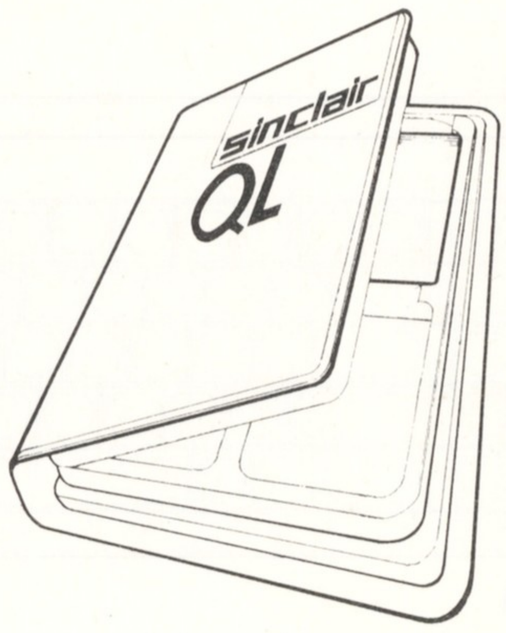
UN ALIMENTATORE

DUE PICCOLI ASTUCCI

nel cui interno si trovano:

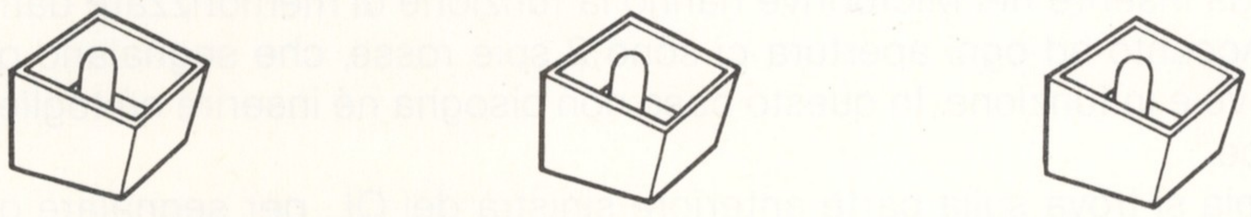
- QL Abacus
- QL Archive
- QL Easel
- QL Quill





e quattro cartucce vuote per i Microdrive del QL.

TRE PIEDINI IN PLASTICA



I piedini possono essere inseriti negli appositi fori, che si trovano sulla base posteriore del QL, per inclinare la tastiera e rendere più agevole la scrittura.

UN CAVO

lungo circa due metri, con differenti morsetti alle due estremità, per permettere di collegare il QL ad un televisore.

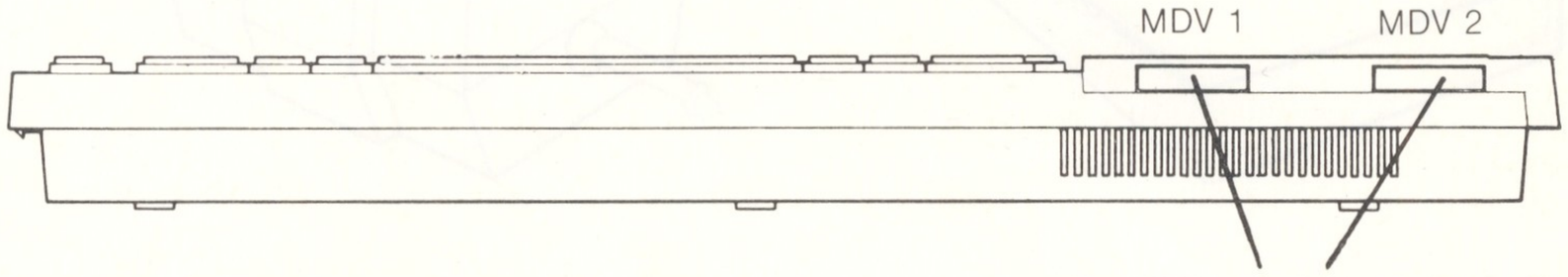
UN CAVO PER RETE LOCALE

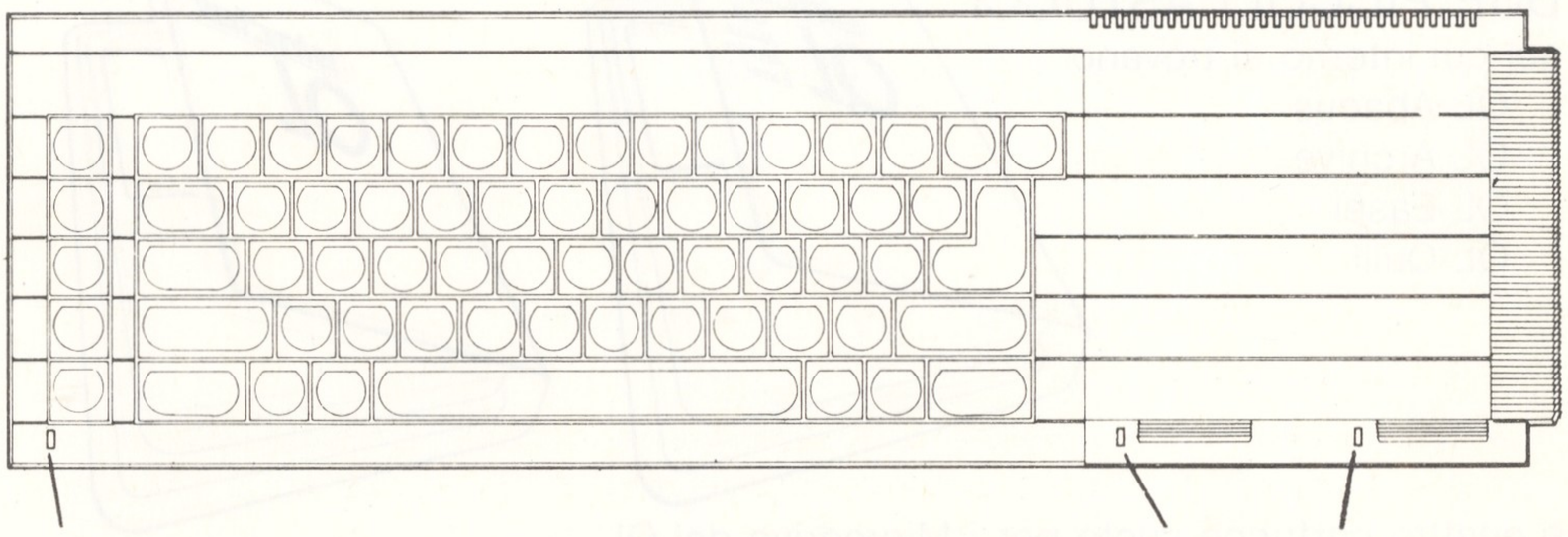
lungo circa due metri, con attacchi uguali alle estremità per permettere di collegare il QL ad altri QL o Spectrum in modo da formare una rete locale.

Sul retro e ai lati del computer ci sono numerose interfacce. Le due fessure nella parte destra anteriore sono i Microdrive. Le cartucce che si inseriscono nei Microdrive sono usate per memorizzare dati e programmi. L'accensione delle piccole spie, poste a sinistra dei Microdrive, ne segnala il funzionamento. Non estrarre o inserire le cartucce a spia accesa. La spia gialla posta vicino ai tasti funzione, indica che il QL è acceso.

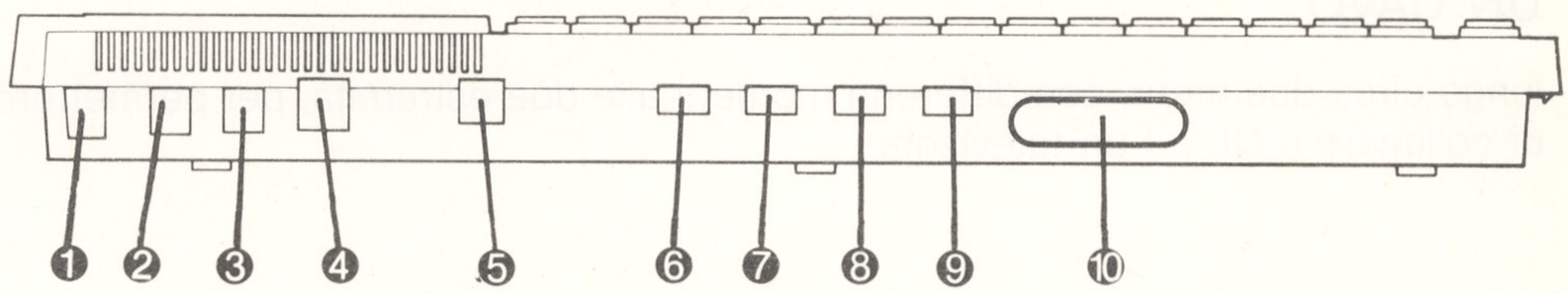
DESCRIZIONE DEL COMPUTER

Sul retro e sui lati del computer ci sono una serie di connettori. Sono usati per permettere di collegare altre unità periferiche al QL (un'unità periferica è uno strumento, che può essere collegato ad un computer per espanderne le capacità. Come ad esempio una stampante od un monitor).





Sulla parte anteriore destra del QL ci sono due aperture. Sono i 2 Microdrive. Le cartucce da inserire nei Microdrive hanno la funzione di memorizzare dati e programmi. Accanto ad ogni apertura ci sono 2 spie rosse, che segnalano quando il Microdrive è in funzione. In questo caso non bisogna né inserire né togliere delle cartucce.
Un'altra spia si trova sulla parte anteriore sinistra del QL, per segnalare quando il QL è funzionante.

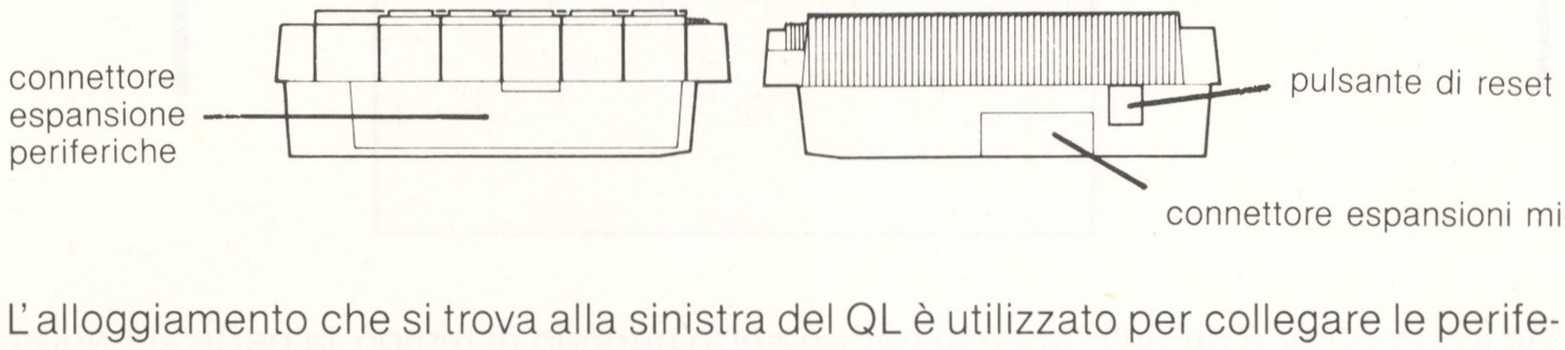


I connettori che si trovano sul retro del QL sono utilizzati per i vari collegamenti:

1	**NET**	collegamento in rete locale
2	**NET**	collegamento in rete locale
3	**POWER**	alimentatore
4	**RGB**	uscita monitor RGB
5	**UHF**	uscita TV UHF
6	**SER1**	prima porta seriale RS-232-C
7	**SER2**	seconda porta seriale RS-232-C
8	**CTL1**	primo controllo joystick
9	**CTL2**	secondo controllo joystick
10	**ROM**	connettore per cartucce ROM

LE CARTUCCE ROM DELLO ZX NON SONO COMPATIBILI CON LE CARTUCCE DEL QL E NON POSSONO ESSERE USATE DAL QL.

Sulla parte destra del QL vi è un'apertura ricoperta da una striscia di plastica. Può essere utilizzata per collegare altri Microdrive fino ad un massimo di sei. I Microdrive dello Spectrum non sono utilizzabili con il QL, mentre cartucce non ancora formattate possono essere usate in entrambi i computer.



L'alloggiamento che si trova alla sinistra del QL è utilizzato per collegare le periferiche e permettere espansioni di memoria.

Il pulsante di reset che si trova sulla parte destra del QL è utilizzato per spegnere e per accendere il computer. Quando il pulsante viene premuto, tutti i programmi in memoria sono cancellati. Prima di premere il tasto, rimuovere le cartucce dai Microdrive.

INSTALLAZIONE

Per permettere al computer di funzionare, sono necessarie alcune operazioni preliminari.

ALIMENTATORE

L'alimentatore del QL è munito di due cavi. Uno termina con un connettore rettangolare con tre fori. L'altro è il cavo principale, senza connettore, da inserire nella presa di alimentazione elettrica, dopo aver collegato una spina adatta.

Non collegare il cavo di alimentazione al QL prima che tutti gli altri cavi e periferiche siano collegati.

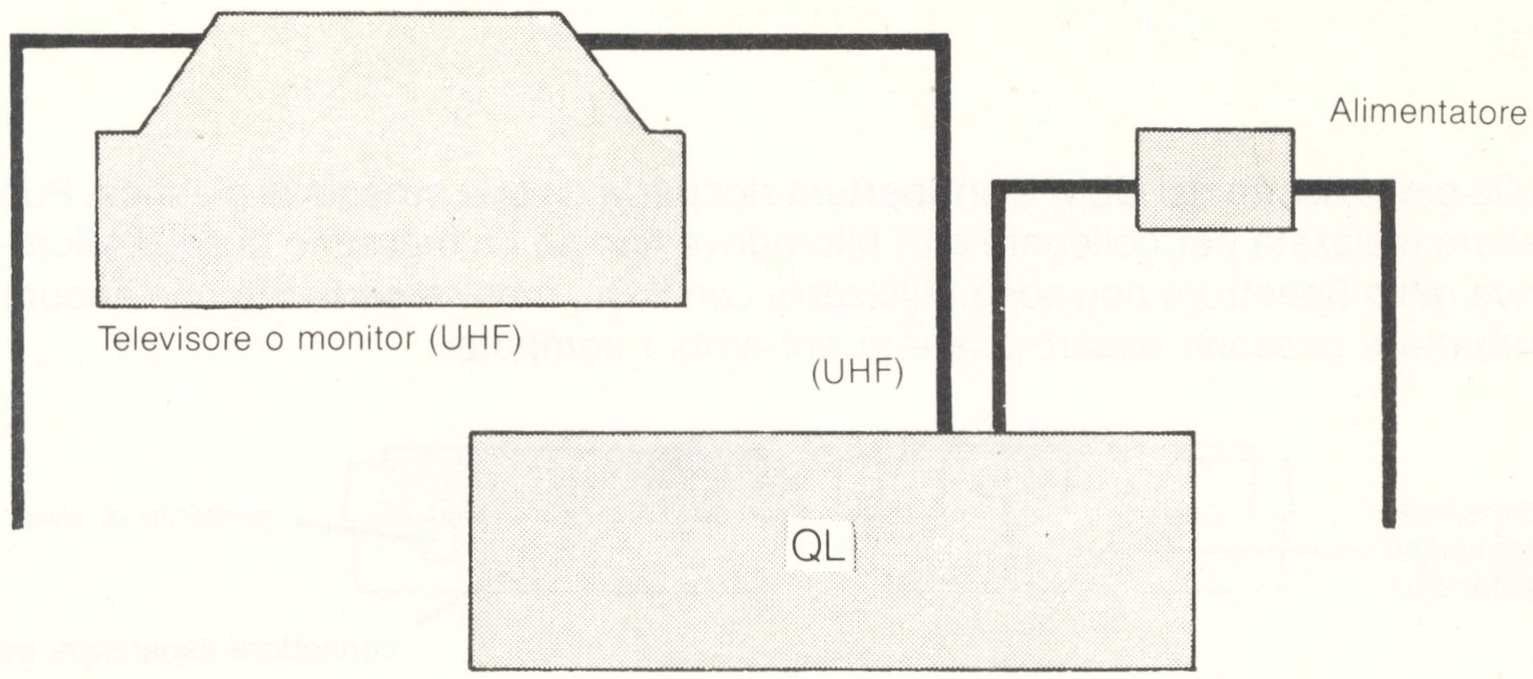
VIDEO

Anche se il QL funziona dopo aver inserito l'alimentatore, non è possibile visualizzare i programmi fino a che non si è collegato il computer a un apparecchio televisivo o a un monitor.

Un monitor ha lo schermo simile alla televisione, ma non può riceverne i segnali. Presenta tuttavia una migliore risoluzione rispetto agli apparecchi televisivi, ma è più costoso.

Naturalmente, per visualizzare i colori del QL, è necessario avere uno schermo a colori, ma il QL funziona benissimo anche in bianco e nero, rappresentando i colori con ombre di grigio.

Tutti i televisori sono adattabili al QL. Dopo aver sfilato il cavo dell'antenna, inserire il cavo del QL nella frequenza UHF.



Se il computer è rimasto acceso per un certo periodo di tempo, la parte superiore dei Microdrive appare calda. Tutto ciò è normale. Il QL non ha un tasto di accensione, ma può essere acceso o spento rimuovendo il connettore dell'alimentatore. **Ricordare che tutti i programmi e tutti i dati inseriti nel computer non sono recuperabili se, prima di spegnere il QL, non vengono memorizzati.**

SINTONIA TV

La frequenza del segnale, per ricevere le immagini del QL su un apparecchio televisivo, è vicina al canale 36.



Immagine di Copyright

Il QL non utilizza il suono del televisore perché possiede generatori di suono propri.

Al momento dell'accensione, lo schermo si riempie di strani segni. In questo periodo, il QL esegue un programma di controllo interno e, dopo pochi secondi, lo schermo diventa nero ed appare l'immagine di COPYRIGHT.

Il QL deve sapere, ora, se si utilizza un monitor o un televisore. Il tasto:

F1 predispone il calcolatore per essere usato con un monitor e

F2 predispone il calcolatore per essere usato con un apparecchio televisivo

Il Microdrive 1 gira per qualche secondo e la spia rossa accanto al Microdrive è accesa. Il QL sta cercando i programmi da caricare ed eseguire. Se non è stata inserita alcuna cartuccia contenente programmi eseguibili, il QL è pronto per accettare i comandi.

TASTIERA

SHIFT

La tastiera del QL possiede due tasti di **SHIFT**. Premendo il tasto **SHIFT** ed allo stesso tempo i tasti corrispondenti a caratteri alfabetici, si ottengono le lettere maiuscole. Premendo **SHIFT** ed altri tasti, si ottengono i simboli situati nella parte superiore dei tasti.

CAPSLOCK

CAPS LOCK ha una funzione analoga a SHIFT, ma fornisce solo lettere maiuscole e non i simboli posti nella parte superiore dei tasti. Dopo aver premuto **CAPS LOCK** si continuano a scrivere lettere maiuscole, finché non lo si preme nuovamente.

CANCELLAZIONE

Premendo il tasto **CTRL** contemporaneamente al tasto del cursore sinistro, è cancellato il carattere alla sinistra del cursore e il cursore continua a muoversi verso sinistra. Premendo **CTRL** ed il tasto del cursore destro, è cancellato il carattere sulla posizione del cursore e il testo si sposta verso destra per riempire il vuoto.

premere

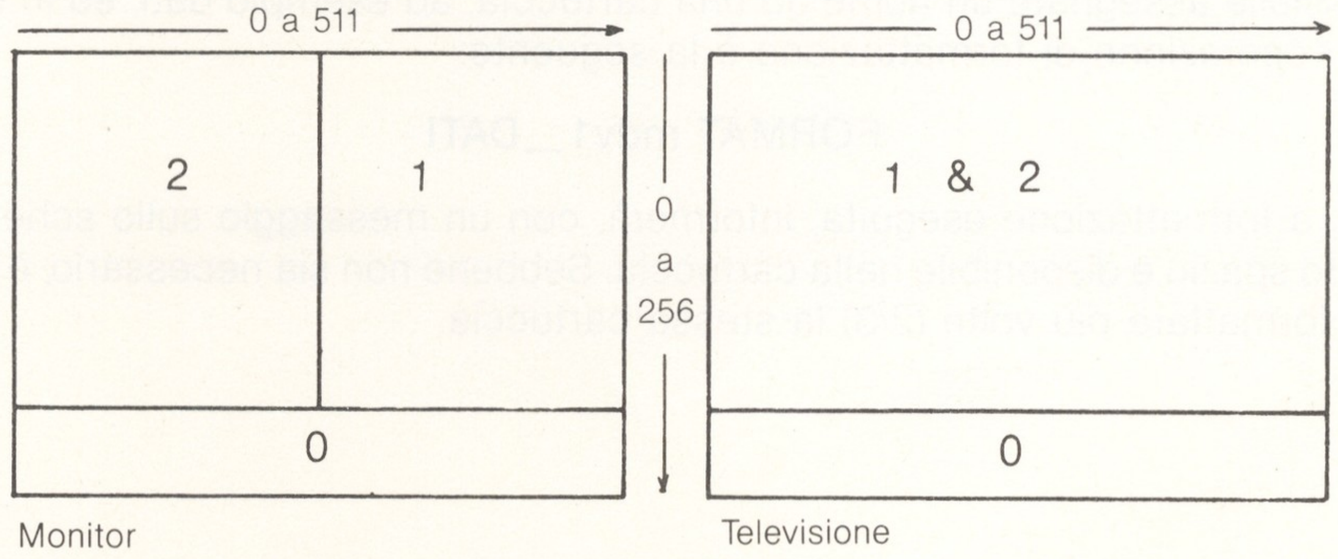
CTRL

e

← **o** →

VIDEO

Lo schermo del QL è diviso in differenti aree. Una volta acceso, e dopo aver premuto F1 o F2, lo schermo si presenta nel modo seguente:



Monitor Televisione

La finestra più sottile, nella parte inferiore dello schermo, è utilizzata per visualizzare i comandi inseriti nel computer, e inizialmente in questa finestra appare il cursore lampeggiante.

Se il cursore è visibile, il QL è pronto ad accettare eventuali istruzioni. Il cursore scompare durante l'esecuzione delle istruzioni.

Per fermare l'esecuzione di un programma Superbasic, premere il tasto **CTRL** e **SPAZIO**.

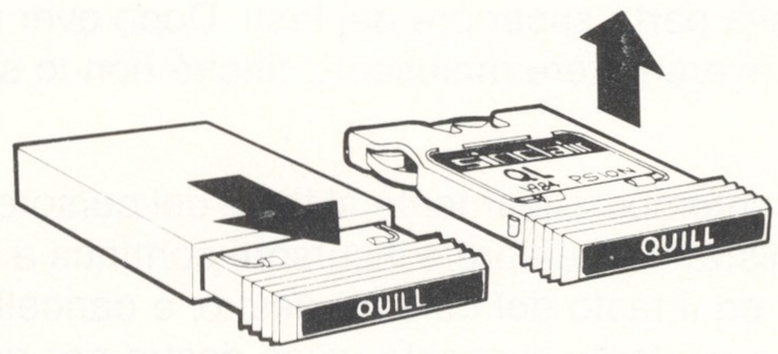
CTRL & SPAZIO

Un eventuale messaggio di errore **ISTRUZIONE SCORRETTA** indica che il computer non ha capito l'istruzione inserita. Cancellare o correggere la riga utilizzando il tasto del cursore.

MICRODRIVES

I due Microdrive del QL sono chiamati **mdv1_** e **mdv2_**.

Le cartucce devono essere inserite in modo corretto nei Microdrive.



È necessario prestare molta attenzione alle cartucce. Non bisogna mai spegnere o accendere il QL mentre una cartuccia è inserita nel Microdrive. Aspettare sempre che le spie rosse, che segnalano il funzionamento dei microdrive, siano spente prima di inserire o rimuovere una cartuccia. Riporre, dopo l'uso, la cartuccia nella sua custodia. Prima di utilizzare una nuova cartuccia è necessario eseguire un'operazione chiamata formattazione. La formattazione cancella tutti i dati ed i programmi eventualmente contenuti sulla cartuccia.

È buona abitudine tenere almeno una copia dei programmi importanti, in caso si verifichino danni alla cartuccia in uso. Per formaţtare una cartuccia è necessario inserirla in un microdrive, l'uno per esempio, e scrivere

FORMAT mdv1_

È possibile assegnare un nome ad una cartuccia, ad esempio dati, ed in questo caso l'operazione di formattazione è la seguente:

FORMAT mdv1_DATI

Il QL, a formattazione eseguita, informerà, con un messaggio sullo schermo di quanto spazio è disponibile nella cartuccia. Sebbene non sia necessario, è opportuno formattare più volte (2/3) la stessa cartuccia.

COME INIZIARE

Esistono diversi modi, per utilizzare questo manuale. Se si vogliono scrivere dei programmi, ma non si ha alcuna conoscenza della programmazione, è necessario iniziare la lettura del manuale dalle prime pagine. Se invece si conosce il Basic, è possibile tralasciare i primi sette capitoli.

IN CASO DI PROBLEMI

Se dovessero sorgere dei problemi utilizzando il QL e i suoi programmi è possibile:

1. Leggere le sezioni appropriate del manuale.
2. Leggere altre pubblicazioni sul QL.
3. Servirsi degli aiuti offerti dal QLUB.

Se nonostante ciò, non fosse possibile utilizzare il QL, consultate le istruzioni inserite nel certificato di garanzia.

4 PROGRAMMI DEL QL

Questa parte del capitolo delinea i quattro programmi consegnati insieme al QL e descrive le caratteristiche comuni.

I quattro programmi sono:

QL Quill - elaboratore di testi
QL Abacus - tabella elettronica
QL Archive - archivio elettronico
QL Easel - programma di grafica gestionale

Più avanti nel Manuale, ad ognuno di questi programmi è dedicato un capitolo.

CARICAMENTO

È buona abitudine, prima di utilizzare un programma del QL, fare almeno una copia di back-up (di sicurezza) e usare la copia. Conservare il programma originale e utilizzarlo solo per fare altre copie.

Per ottenere una copia di un programma è sufficiente:

1. Inserire la cartuccia originale nel Microdrive 2
2. Inserire la cartuccia vuota, o una contenente dati che non interessano, nel Microdrive 1 e scrivere:

lrun mdv2_clone

Dopo aver premuto ENTER, sullo schermo appare il messaggio:

FORMAT mdv1_ premere spazio per continuare

Premere la barra spaziatrice solo quando si è certi che i dati contenuti sulla cartuccia non interessano, perché la formattazione opera una cancellazione completa dei dati. Il computer formatta la cartuccia e copia il programma.

Attendere che la spia del Microdrive sia spenta prima di rimuovere la cartuccia.

Esistono due sistemi per caricare un programma:

Se nei Microdrive non è inserita alcuna cartuccia, premere RESET. Inserire la cartuccia contenente la copia del programma nel Microdrive 1, e successivamente premere F1 o F2. Dopo qualche secondo appare sullo schermo un messaggio che conferma che il programma è stato caricato.

Oppure inserire la cartuccia con il programma nel Microdrive 1 e scrivere:

lrun mdv1_boot

Dopo aver premuto ENTER il programma è caricato in modo analogo al precedente.

Se si vuole sfruttare il tasto di aiuto disponibile per ognuno dei quattro programmi, non rimuovere le cartucce dal Microdrive.

La zona di controllo, nella parte superiore dello schermo, permette di scegliere i comandi propri di ogni programma e le opzioni proprie di ogni comando. I programmi suggeriscono delle risposte (default) alle domande rivolte. Premere il tasto ENTER se si accettano i default, diversamente è necessario eseguire la propria scelta.
Premendo il tasto F2 scompare la zona di controllo e riappare solo se lo stesso tasto è nuovamente premuto.

L'area centrale dello schermo visualizza i dati che si stanno utilizzando, ad esempio il testo di un documento, un grafico, ecc...

Nella parte inferiore dello schermo sono visualizzate le istruzioni o i comandi assegnati.

Più in basso esiste un'"area di status" che indica lo stato attuale del lavoro. Visualizza ad esempio il nome del documento, lo spazio di memoria ancora disponibile, ecc...

TASTI FUNZIONE

Tre dei cinque tasti di funzione hanno lo stesso significato nei quattro programmi.

Tasto	Funzione
F1	Richiesta livelli di aiuto
F2	Cancella e ridisegna l'area di controllo
F3	Richiama i comandi

I due rimanenti tasti di funzione sono utilizzati per eseguire particolari azioni in ogni programma.

Aiuto

La prima opzione, che appare sulla parte superiore sinistra dell'area di controllo, indica che è possibile scegliere il livello di aiuto premendo F1.

L'aiuto suggerisce altre voci con le quali può essere richiamato uno stesso argomento. Scrivere il nome dell'argomento e premere poi il tasto ENTER . Non è necessario scrivere il nome completo, ma un numero di caratteri sufficienti a distinguerlo da altri. È possibile eseguire questa operazione quante volte è necessario.

Premendo ENTER senza aver selezionato un argomento, il programma torna al precedente livello di aiuto.

Escape

Il tasto **ESC** permette di abbandonare i livelli di aiuto e di rientrare nel programma.

EDITOR DI RIGA

È possibile utilizzare un editor di riga per modificare o correggere una riga di comandi che si sta scrivendo. Tutti i programmi del QL utilizzano un editor di riga, ma ognuno con diverse funzioni. QUILL utilizza un editor di riga per modificare il testo, ARCHIVE per modificare programmi.

L'editor di riga utilizza i quattro tasti del cursore, in combinazione con i tasti **CTRL** e **SHIFT**.

Tasti	Azione
←	Sposta il cursore di un carattere a sinistra
→	Sposta il cursore di un carattere a destra
SHIFT & ←	Sposta il cursore di una parola a sinistra
SHIFT & →	Sposta il cursore di una parola a destra
CTRL & ←	Cancella il carattere alla sinistra del cursore
CTRL & ←	Cancella il carattere nella posizione del cursore
CTRL & ←	Cancella la riga alla sinistra del cursore
CTRL & →	Cancella la riga alla destra del cursore
SHIFT & CTRL & ←	Cancella la parola alla sinistra del cursore
SHIFT & CTRL & →	Cancella la parola alla destra del cursore

Il simbolo **&** indica che il primo tasto deve essere premuto contemporaneamente al secondo.

MICRODRIVES

Prima di utilizzare i livelli di aiuto o un comando di stampa, assicurarsi che la cartuccia del programma sia inserita nel Microdrive 1.

Il Microdrive 2 e i Microdrive addizionali possono essere utilizzati per memorizzare informazioni, ad esempio documenti scritti con Quill, ecc.

Nomi dei File

Le informazioni sono memorizzate su una cartuccia in vari "file". Ogni file deve avere un nome che lo distingua dagli altri della stessa cartuccia. Il nome di un file non può essere più lungo di otto caratteri, senza spazi. È buona abitudine utilizzare dei nomi che descrivano il contenuto dei file, ad esempio : "vendite".

Per richiamare un file è sufficiente scrivere il nome del file stesso.

Esiste una terza componente nel nome di file che non è visualizzata, perché è assegnata automaticamente dal programma. È un'estensione di tre lettere del nome che identifica il file. Le estensioni usate sono:

QL Quill	__doc
QL Abacus	__aba
QL Easel	__grf
QL Archive (file di dati)	__dbf
QL Archive (file di programma)	__prg o __pro
QL Archive (file di schermo)	__scn

Esiste poi un'estensione riconosciuta da tutti i quattro programmi, __exp, che consente di trasferire informazioni fra i quattro programmi.

Elenco dei file di una cartuccia

In tutti i programmi, ad eccezione dell'Archive, è possibile richiedere l'elenco dei nomi dei file residenti sulla cartuccia. Ogni volta che con uno dei quattro programmi si vuole memorizzare un file, è possibile scegliere tra diverse opzioni.

Premere ENTER se si accetta il nome suggerito dal programma.

Scrivere il nome del file seguito da ENTER quando si vuole assegnare un nuovo nome ad un file.

Premere il simbolo "?" seguito da ENTER per richiedere l'elenco dei file del Microdrive 2.

Se viene scritto il punto di domanda (e ENTER), il programma visualizza:

mdv2__

suggerendo di esaminare l'elenco dei file del Microdrive 2. È possibile accettare il default o modificare il numero del drive per riferirsi ad un altro Microdrive (mdv1__), premere poi ENTER per ottenere l'elenco dei file. Quando l'elenco è completo, il programma chiede di scrivere il nome del file.

L'Archive non utilizza questo metodo, ma dispone di un comando (**Dir**) che elenca i file.

Escape

In generale, il comando **ESC** cancella l'azione del comando corrente e ritorna il controllo ad un punto immediatamente precedente alla scelta del comando. È possibile utilizzare il comando **ESC** per cancellare qualsiasi numero o testo che si è scritto in una riga di istruzioni o per annullare l'effetto di un comando prima che sia eseguito.

I dati possono essere memorizzati e caricati su altre unità oltre che su Microdrive.

Ad esempio, caricare e salvare dati in rete locale:

Prima di caricare un programma, è necessario assegnare un numero di stazione ad ogni computer collegato alla rete locale. Accendere il computer, ma non inserire la cartuccia. Premere F1 o F2.

RETE LOCALE

Per fissare il numero di stazione, usare il comando **NET** seguito dal numero scelto. Ad esempio:

NET 5 ENTER

Inserire la cartuccia contenente il programma nel Microdrive 1 e caricare il programma scrivendo:

lrun mdv1_boot ENTER

Una volta che il programma è in fase di esecuzione, è possibile ricevere i dati trasmessi dalla rete locale utilizzando il comando **load** nel modo usuale. Se i dati sono trasmessi dalla stazione 12, ad esempio, scrivere:

LOAD__neti__12

L'operazione deve essere eseguita prima che la stazione cominci la trasmissione.

Per trasmettere dei dati, utilizzare il comando **Save**. Supponendo di trasmettere alla stazione 23, scrivere:

SAVE_neto_23

La stazione 23 deve essere pronta a ricevere i dati prima che venga premuto il tasto ENTER

sinclair

QL

Primo approccio

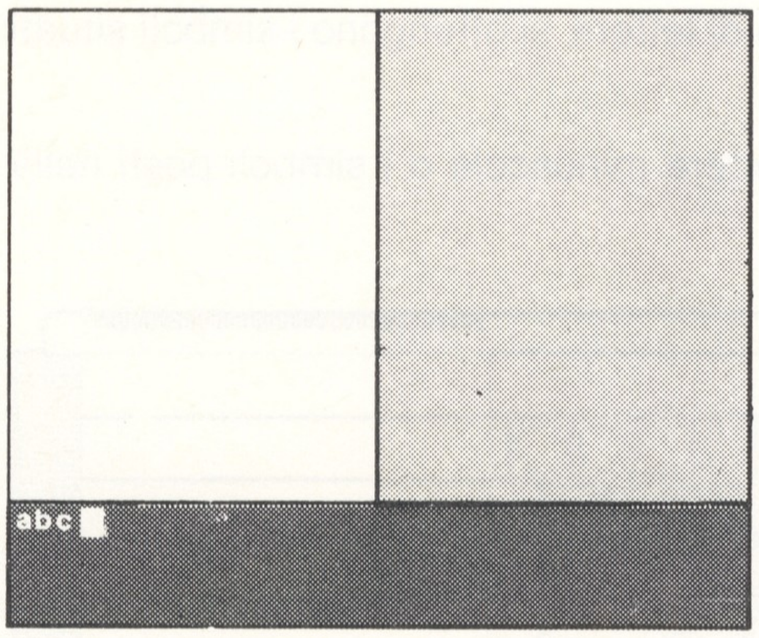
©1985 SINCLAIR RESEACH LIMITED

Было обыскано
ОТ
ЕДИНЕН

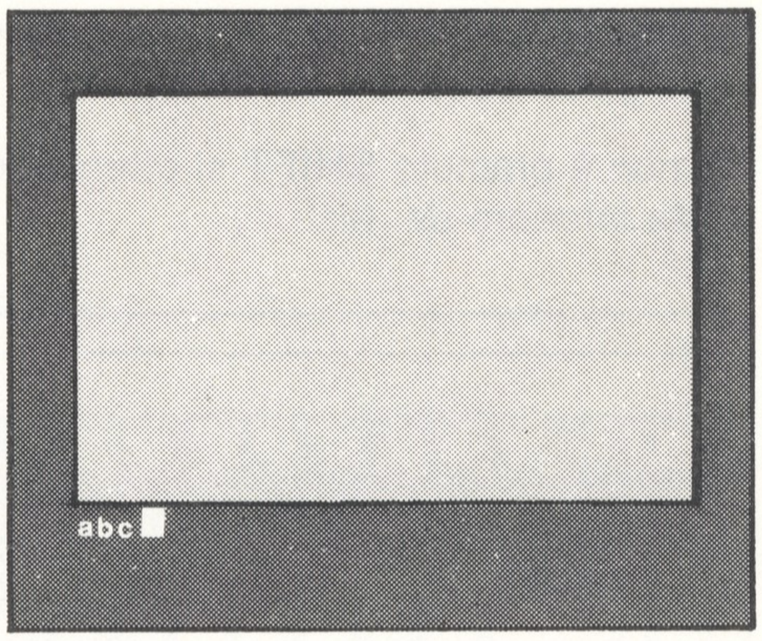
CAPITOLO 1
IL COMPUTER

LO SCHERMO

Il QL deve essere collegato ad una televisione o ad un monitor prima dell'accensione. Premendo alcuni tasti, ad esempio abc, lo schermo appare come nelle figure sottostanti.



Monitor



Televisione

Se lo schermo non dovesse presentarsi nel modo ora descritto, consultate il capitolo **INTRODUZIONE** per avere informazioni più precise.

LA TASTIERA

Il QL è un computer versatile e potente. Perciò non tutte le funzioni della tastiera saranno presentate in questo capitolo. Per il momento, chiariremo solo quelle di immediato utilizzo.

BREAK

BREAK interrompe il lavoro del computer. Il suo uso è consigliabile quando si è deciso di sospendere l'azione di un comando, o l'esecuzione di un programma. Data la potenza dell'istruzione, è stato reso volutamente difficile attivare casualmente il BREAK. Per farlo è necessario premere contemporaneamente i tasti **CTRL** e **SPAZIO**.

L'esecuzione di un programma interrotta con BREAK, può essere ripresa scrivendo il comando **CONTINUE**, se nel frattempo non è stata fatta alcuna modifica al programma.

RESET

RESET non è un comando, ma un piccolo pulsante che si trova sulla parte destra del QL. È stato costruito in una posizione di difficile accesso, perché i suoi effetti sono più critici di quelli del BREAK. Se non si ottiene alcun risultato con il comando BREAK, premere il pulsante **RESET**. RESET ripristina il sistema. In altre parole ciò corrisponde a spegnere e riaccendere il computer.

SHIFT

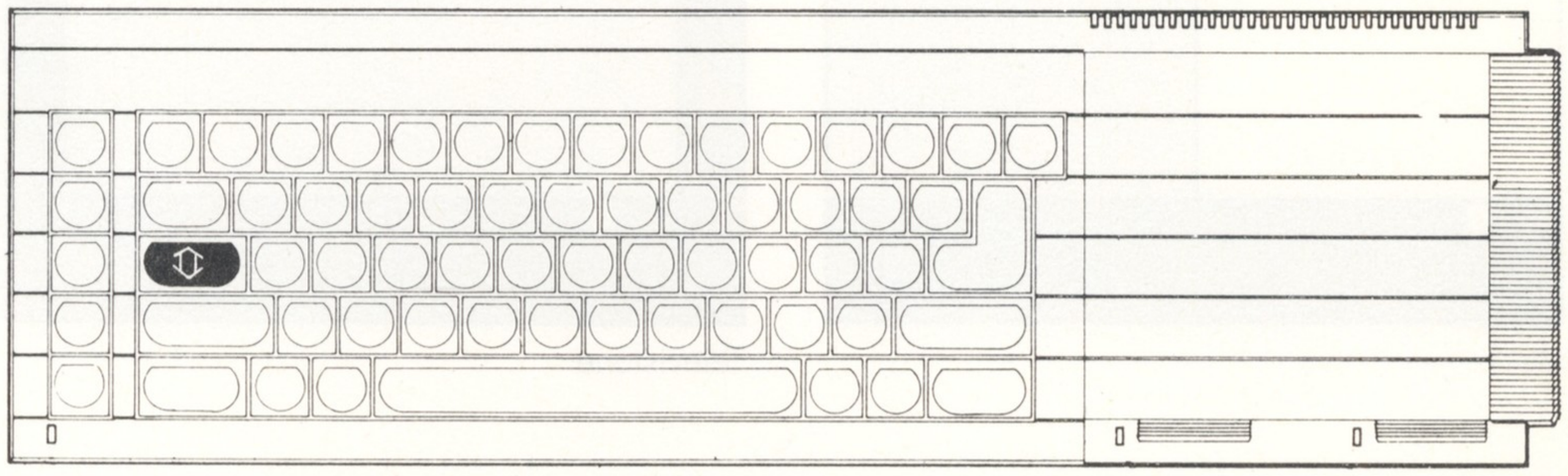
Il QL ha due tasti "**SHIFT**". Sono usati molto spesso e quindi devono essere utilizzabili con entrambe le mani.

Premendo **SHIFT** ed allo stesso tempo i tasti corrispondenti a caratteri alfabetici, si ottengono le lettere maiuscole.

Tenendo premuto **SHIFT** ed altri tasti, non di lettere, si ottengono i simboli situati nella parte superiore dei tasti.

Se non è premuto **SHIFT** si ottengono lettere minuscole o i simboli posti nella parte inferiore dei tasti.

CAPS LOCK

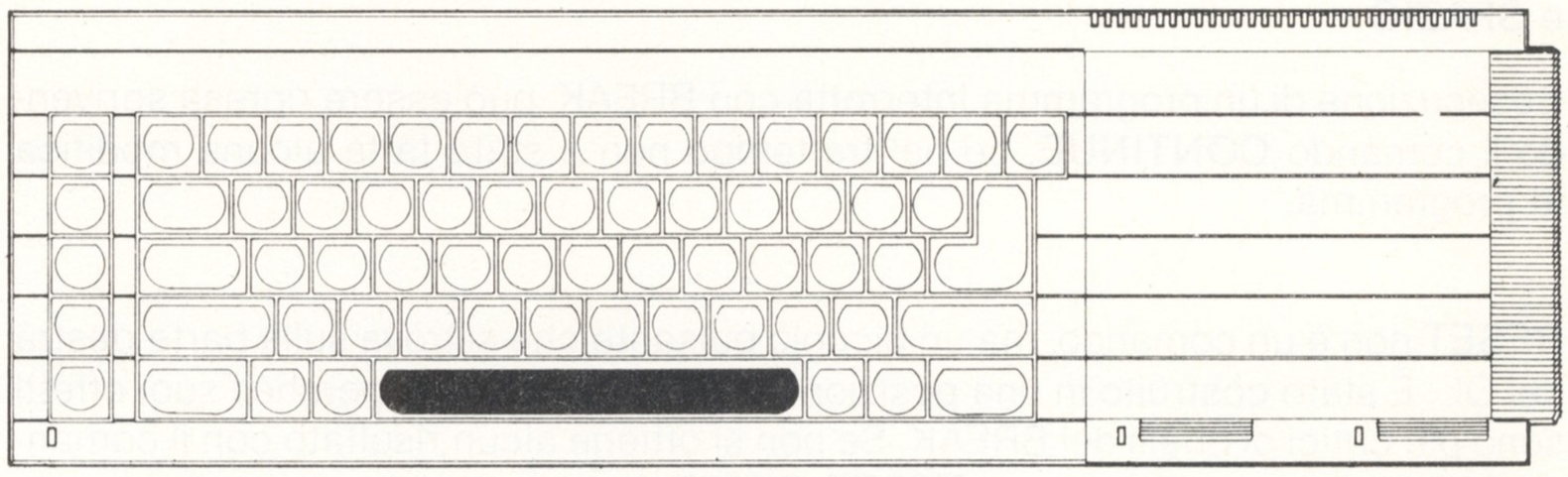


CAPS LOCK ha una funzione analoga a SHIFT, ma fornisce solo le lettere maiuscole e non i simboli posti nella parte superiore dei tasti.

Premere alcuni tasti.
Premere **CAPS LOCK** una volta.
Scrivere alcuni caratteri alfabetici.

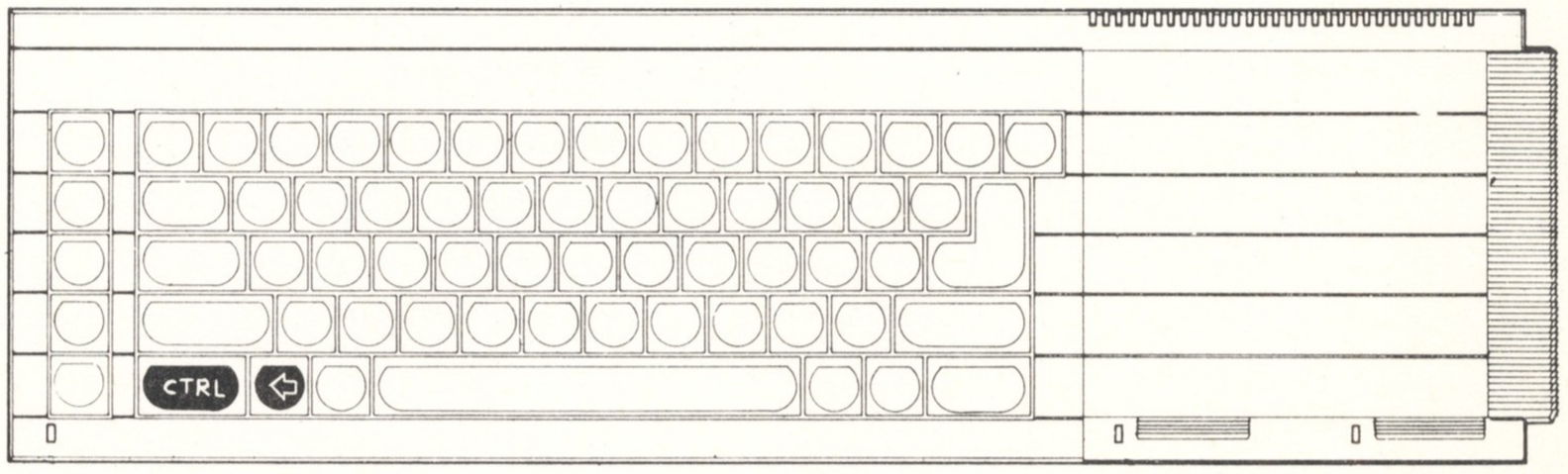
Come si può osservare, dopo aver premuto **CAPS LOCK** si continuano a scrivere lettere maiuscole finché non lo si preme nuovamente.

SPAZIO

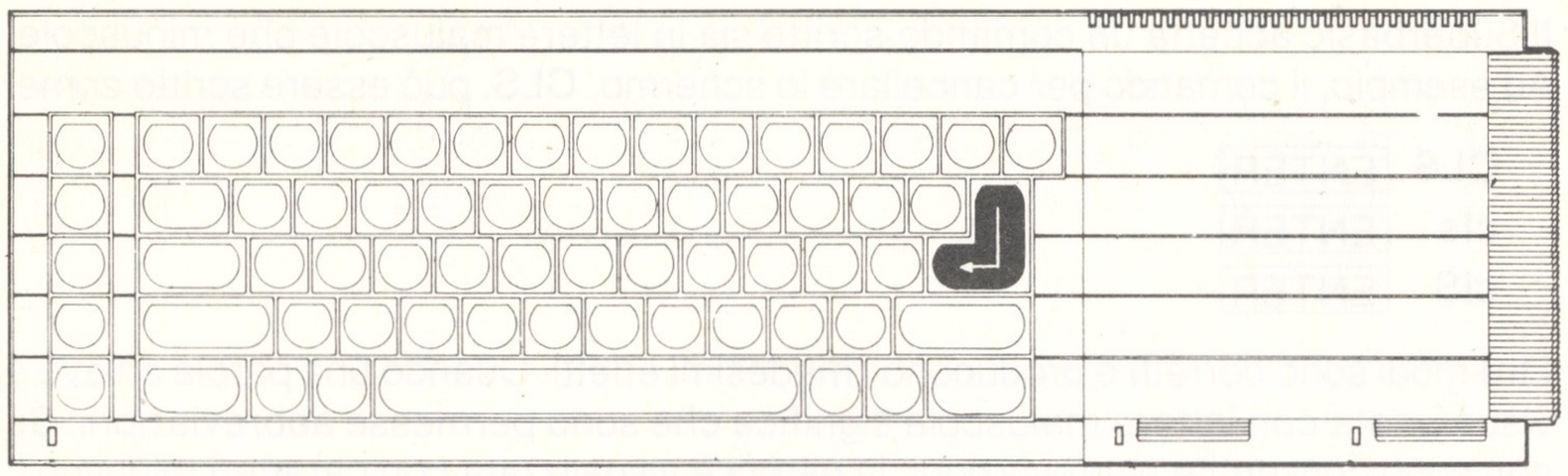


Il tasto più lungo, posto nella parte inferiore della tastiera, dà gli spazi. La sua funzione è molto importante nel Superbasic.

CANCELLAZIONE



La combinazione del tasto **CTRL** con il tasto del cursore sinistro cancella l'ultimo carattere scritto.



ENTER

Dopo aver scritto un messaggio o un'istruzione completa, come ad esempio **RUN**, è necessario che il computer esegua l'istruzione. Ciò si ottiene premendo il tasto ENTER .

ENTER è usato molto frequentemente.

Provare ENTER scrivendo:

PRINT "Corretto" ENTER

Se l'istruzione è stata scritta in modo corretto il computer risponde:

Corretto

ALTRI TASTI DI USO IMMEDIATO

*	moltiplicazione	+	addizione
—	sottolineatura	=	uguale
„	virgolette	'	apostrofo
,	virgola	!	punto esclamativo
;	punto e virgola	&	e commerciale
:	due punti	.	punto decimale o punto
/	divisione	$	dollaro
(	parentesi aperta	)	parentesi chiusa

MAIUSCOLE E MINUSCOLE

Il Superbasic accetta un comando scritto sia in lettere maiuscole che minuscole. Ad esempio, il comando per cancellare lo schermo, **CLS**, può essere scritto come:

CLS ENTER
cls ENTER
cIS ENTER

I tre modi sono corretti e producono i medesimi effetti. Quando una parola chiave è visualizzata con lettere maiuscole significa che sono permesse abbreviazioni. Se non sono consentite abbreviazioni, la parola è visualizzata con caratteri minuscoli.

LE VIRGOLETTE

La normale funzione delle virgolette è quella di delimitare una parola, una frase o una stringa di caratteri. Ad esempio:

PRINT ''L'istruzione funziona'' ENTER

Il computer risponde:

L'istruzione funziona.

Le virgolette non sono visualizzate, ma indicano che sullo shermo deve apparire tutto ciò che si trova al loro interno. È possibile usare il simbolo '' ' '' (apostrofo) con la stessa funzione delle virgolette.

PRINT 'Il simbolo delle virgolette è '' '

sul video appare:

Il simbolo delle virgolette è ''

ERRORI DI SCRITTURA

Il tasto con il numero zero si trova, assieme agli altri tasti numerici, nella parte più alta della tastiera, ed è leggermente più sottile. Il tasto della lettera 'O' si trova tra le altre lettere. Fare attenzione a non confondere i due tasti. Allo stesso modo prestare attenzione a non confondere il tasto della lettera ''l'' con il numero 1.

COMBINAZIONI DI TASTI

Il tasto **SHIFT** è utilizzato in combinazione con altri tasti per ottenere particolari risultati. Lo stesso vale per i tasti **CTRL** e **ALT**, la cui funzione sarà chiarita in seguito. Scrivere le due istruzioni:

CLS ENTER
PRINT ''Ciao'' ENTER

Questo è un semplice programma. L'operazione più importante è la memorizzazione del programma. Le istruzioni appena date sono eseguite immediatamente premendo ENTER . Un programma è scritto in modo appropriato quando ad ogni istruzione è assegnato un numero di riga.

10 CLS ENTER
20 PRINT ''Ciao'' ENTER

Ora il programma appare nella parte superiore dello schermo. Ciò significa che è stato accettato, perché corretto sia da un punto di vista grammaticale che di sintassi. Rispetta le regole del linguaggio Superbasic, ma non è ancora stato eseguito. È residente in memoria. Per farlo funzionare scrivere:

RUN ENTER

La differenza tra un comando diretto e una sequenza di istruzioni in memoria è chiarita nel secondo capitolo. Per il momento è possibile realizzare delle semplici prove sfruttando le istruzioni già descritte, più altre due di immediata utilizzazione.

LIST ENTER

Provoca la visualizzazione sullo schermo del programma in memoria.

NEW ENTER

Provoca la cancellazione del programma in memoria, in modo da poterne scrivere un altro.

TEST SUL CAPITOLO 1

In questo test si possono ottenere al massimo sedici punti. Controllare il punteggio con le risposte della pagina seguente.

1. In quali casi è consigliabile usare la sequenza BREAK? (3 punti)
2. Dove si trova il tasto di RESET?
3. Quali sono gli effetti prodotti dal tasto di RESET?
4. Indicare almeno due differenze tra i tasti SHIFT e CAPS LOCK. (2 punti)
5. Come si può cancellare un carattere che si è erroneamente scritto?
6. Quale è lo scopo della funzione ENTER ?
7. Quale simbolo viene usato per indicare la funzione ENTER?

Quali sono gli effetti prodotti dai comandi indicati dal numero 8 al numero 11?

8. CLS ENTER
9. RUN ENTER
10. LIST ENTER
11. NEW ENTER
12. Gli effetti delle parole chiave sono gli stessi se queste sono scritte con lettere minuscole?
13. Perché alcune parti di parole chiave appaiono scritte solo con lettere maiuscole?

RISPOSTE AL TEST SUL CAPITOLO 1

1 La sequenza **CTRL SPAZIO** (**BREAK**) è utilizzata per interrompere l'esecuzione di un programma quando: 1 — esiste qualche errore

2 — il programma non interessa più

3 — qualsiasi altro problema (3 punti)

2. Il tasto **RESET** si trova sulla parte destra del Ql.

3. Il tasto di **RESET** spegne e accende il computer

4a). Il tasto **SHIFT** è attivo solo mentre è premuto. Il tasto **CAPS LOCK** rimane attivo dopo che è stato premuto. (1 punto)

4b) **SHIFT** ritorna le lettere maiuscole i numeri e tutti gli altri simboli posti nella parte superiore dei tasti, mentre **CAPS LOCK** ritorna solo le lettere. (1 punto)

5. La combinazione dei tasti **CTRL** e cursore sinistro cancella il primo carattere alla sinistra del cursore.

6. Il tasto ENTER consente di memorizzare un'istruzione.

7. Per indicare ENTER è utilizzato il simbolo: ENTER .

8. **CLS** ENTER pulisce una parte dello schermo.

9. **RUN** ENTER provoca l'esecuzione di un programma residente in memoria.

10. **LIST** ENTER lista sullo schermo un programma residente in memoria.

11. **NEW** ENTER predispone la memoria principale per ricevere un nuovo programma.

12. Le parole chiave in Superbasic possono essere scritte sia in lettere maiuscole che minuscole.

13. La parte di parola chiave che appare sullo schermo in lettere maiuscole è l'abbreviazione permessa per quella parola chiave.

CONTROLLO DEL PUNTEGGIO

Da 14 a 16. Ottimo. Proseguire la lettura.

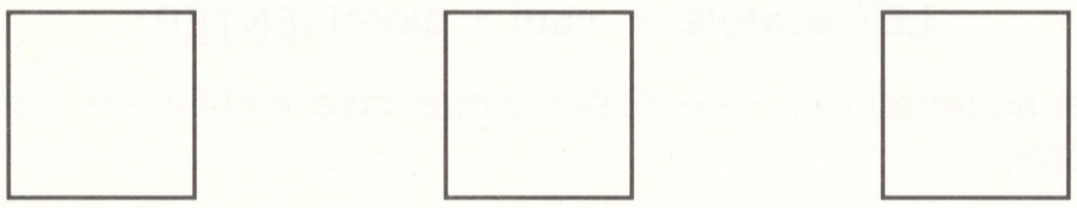
Da 12 a 13. Buono. Rileggere alcune parti del capitolo.

Da 10 a 11. Sufficiente. Rileggere alcune parti del capitolo e ripetere il test.

Meno di 10. Rileggere con attenzione il capitolo e ripetere il test.

CAPITOLO 2
ISTRUZIONI DEL COMPUTER
NUMERI, NOMI E CASELLE
esempio

Un computer ha bisogno di memorizzare istruzioni come sequenze di numeri. La memoria può essere immaginata come un insieme di caselle, alle quali è necessario assegnare un nome per poterle distinguere.



Supponiamo di voler risolvere un semplice problema.

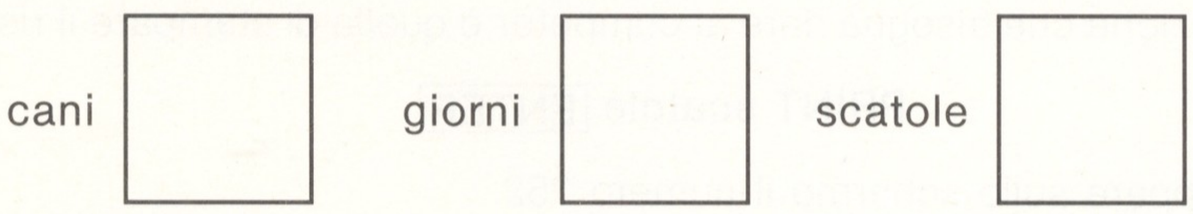
Un allevatore di cani deve nutrire 9 cani per 28 giorni.

Ognuno di essi mangia una scatola di "Beefo" al giorno.

Calcolare e stampare il numero di scatole necessarie. Un modo di risolvere il problema richiede tre "caselle", per:

numero di cani
numero di giorni
totale numero di scatole

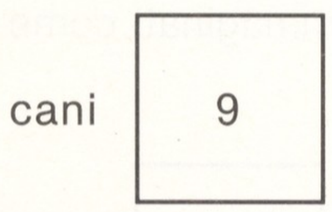
Il Superbasic permette di attribuire ad ogni casella un nome. Per esempio:



Il computer può, con una sola istruzione, creare una casella, assegnarle un nome e inserire un numero al suo interno:

```
LET cani = 9 ENTER
```

Con questa istruzione si crea una casella, chiamata cani, contenente il numero 9, cioè:



La parola **LET** ha un significato speciale nel Superbasic. È una parola chiave. Il Superbasic ha molte altre parole chiave, che saranno descritte in seguito. È necessario prestare attenzione allo spazio dopo LET e dopo le altre parole chiave. Poiché in Superbasic è possibile scegliere il nome delle caselle con la massima libertà, una eventuale parola *LETcani* sarebbe assunta come nome di una casella. L'uso della parola chiave LET è facoltativo nel Superbasic e, per questo, istruzioni come:

```
LET cani = 9 ENTER
```

sono valide, ma il comando crea una casella chiamata LETcani. In Superbasic i nomi, i numeri e le parole chiave devono essere separati uno dall'altro da spazi, a meno che non siano separati da caratteri speciali.

È possibile controllare che una casella esista scrivendo:

```
PRINT cani ENTER
```

Sullo schermo apparirà il contenuto della casella:

```
9
```

Fare attenzione ad inserire uno spazio dopo **PRINT**.

Per risolvere il problema è possibile scrivere un programma composto da una serie di istruzioni. Le prime due sono:

```
LET cani  = 9 ENTER
LET giorni = 28 ENTER
```

Attraverso queste due istruzioni si creano due caselle, a cui sono attribuiti i valori 9 e 28. L'istruzione successiva deve operare una moltiplicazione, il cui simbolo per il computer è "*" e inserire il risultato in una nuova casella chiamata "scatole".

LET scatole = cani * giorni ENTER

1. Il computer accetta i valori 9 e 28 per le due caselle chiamate rispettivamente cani e giorni.
2. Il numero 9 viene moltiplicato per 28.
3. Si crea una nuova casella chiamata scatole.
4. Il risultato della moltiplicazione è il valore della casella chiamata scatole.

Tutto ciò può sembrare inutilmente complicato, ma la cosa più importante è comprendere l'idea. L'effetto può essere immaginato molto semplicemente, come rappresentato nelle caselle:

Cani	9	Giorni	28	Scatole	252

L'ultima istruzione che bisogna dare al computer è quella di stampare il risultato:

PRINT scatole ENTER

Di seguito appare sullo schermo il numero 252.

In conclusione il programma:

```
LET cani = 9 ENTER
LET giorni = 28 ENTER
LET scatole = cani * giorni ENTER
PRINT scatole ENTER
```

produce gli effetti che sono stati immaginati come tre caselle, con i loro nomi, contenenti ciascuna un numero:

Cani	9	Giorni	28	Scatole	252

sullo schermo appare:

252

Naturalmente si sarebbe ottenuto lo stesso risultato, con maggiore velocità, usando una calcolatrice o lo stesso QL, scrivendo semplicemente:

PRINT 9 * 28 ENTER

I concetti ora espressi sono un punto di partenza essenziale per la programmazione in Superbasic e sono talmente importanti da essere comuni a tutti i linguaggi di programmazione.

1. Nomi come cani, giorni e scatole sono detti **Variabili**.
2. Una riga del tipo:

   LET cani = 9 ENTER

   è detta **istruzione.**
3. L'associazione di un nome ad una casella è chiamata **variabile**. L'esecuzione dell'istruzione assegna il valore 9 alla variabile *cani*.
   Un'istruzione come:

   LET cani = 9 ENTER

rappresenta all'interno del computer un processo dinamico, ma il testo stampato è statico e il simbolo "=" non assume il significato matematico di equivalenza. Sarebbe meglio pensare o dire (ma non scrivere):

LET cani assume il valore 9

e pensare al processo come avente una direzione da destra verso sinistra.

cani ⟵ 9

Il simbolo "=", in un'istruzione LET, assume un significato diverso da quello che ha usualmente in matematica. Ad esempio l'istruzione:

LET cani = cani + 1 ENTER

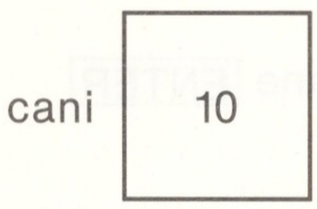
da un punto di vista matematico non ha senso, ma in termini di operazioni col computer tutto è molto semplice. Se il numero di cani prima di **LET cani = cani + 1** era 9, dopo la istruzione è 10. Provare scrivendo:

```
LET cani = 9 ENTER
PRINT cani ENTER
LET cani = cani + 1 ENTER
PRINT cani ENTER
```

Il risultato è:

```
 9
 10,
```

dimostrando che il valore finale della variabile è 10:



Un buon sistema per capire come si comportano le variabili, è eseguire un'operazione chiamata "simulazione". Una simulazione consiste nell'esaminare semplicemente un'istruzione alla volta, annotare i valori delle variabili risultanti e verificare come le variabili mutino il loro valore durante l'esecuzione del programma.

	cani	giorni	scatole
LET cani = 9 ENTER	9		
LET giorni = 28 ENTER	9	28	
LET scatole = cani * giorni ENTER	9	28	252
PRINT scatole ENTER	9	28	252

Il risultato è:

```
 252
```

Si può notare che, la prima volta che compare, il nome di una variabile è scritto nella parte sinistra di un'istruzione LET. Dopo che si è creata la casella corrispondente e che alla stessa è stato attribuito un valore, il nome può anche essere usato nella parte destra di un'istruzione LET. Per esempio si supponga di incoraggiare un bambino a risparmiare dei soldi e gli si promettono due pezzi di cioccolata per ogni moneta risparmiata. Il calcolo è il seguente:

```
LET pezzi = moneta * 2 ENTER
PRINT pezzi ENTER
```

La simulazione di questo esempio non è fattibile, perché non si conosce il numero di monete risparmiate. È stato volontariamente commesso un errore nell'usare la variabile **moneta** alla destra di un'istruzione LET, senza averla prima "tradotta" in casella e senza averle assegnato un valore. Il QL ricercherà la variabile **moneta** e non trovandola darà un messaggio di errore.

Se si prova a stampare il valore di "moneta", il QL risponde con un "*" sul video, per indicare che la variabile non è stata definita. Si dice che alla variabile non è stato attribuito un valore iniziale. Il programma funziona con questa piccola variazione:

```
LET moneta = 7 ENTER
LET pezzi = moneta * 2 ENTER
```

pezzi	monete
7	
7	14

e dà come risultato

PROGRAMMA IN MEMORIA

Scrivere un programma senza indicarne i numeri di riga, può produrre il risultato voluto. Tuttavia ci sono due ragioni ben precise perché questo metodo non risulti soddisfacente.

1. Il programma funziona alla velocità con cui si scrive. Non è molto, per una macchina in grado di effettuare milioni di operazioni al secondo.
2. Le singole istruzioni non sono memorizzate dopo la esecuzione, ed in tal modo il programma non può funzionare più di una volta e non è possibile correggere un errore senza riscriverlo completamente,

Charles Babbage, un pioniere del computer del diciannovesimo secolo, affermò che un computer deve essere in grado di memorizzare istruzioni, allo stesso modo di dati all'interno di caselle. Le istruzioni memorizzate sono poi eseguite in rapida successione senza ulteriore intervento dell'uomo. Per esempio:

```
10 LET prezzo = 15 ENTER
20 LET penne =  7 ENTER
30 LET costo = prezzo * penne ENTER
40 PRINT costo ENTER
```

Esternamente non accade nulla, ma l'intero programma è memorizzato internamente. Perché funzioni, è sufficiente scrivere:

```
RUN ENTER
```

e il risultato

```
105
```

appare sullo schermo. In questo modo è possibile scrivere nuove istruzioni o correggere un programma con poco sforzo.

CORREGGERE UN PROGRAMMA

Più avanti sarà chiarito come correggere un programma in Superbasic, ma già a questo punto è possibile eseguire con facilità tre operazioni:

sostituire una riga
inserire una nuova riga
cancellare una riga

Sostituzione di una riga

Si supponga di voler correggere il programma precedente, perché il prezzo di ogni penna è ora 20. In questo caso è sufficiente riscrivere la riga 10.

```
10 LET prezzo = 20 ENTER
```

Questa riga sostituisce la precedente riga 10 e le altre righe restano immutate. Per eseguire il programma scrivere:

```
RUN ENTER
```

e la nuova soluzione, 140, è visualizzata sullo schermo.

Inserimento di una riga

Si supponga di voler inserire una nuova istruzione prima dell'ultima, per stampare le parole "Costo Totale". Situazioni di questo tipo si verificano molto spesso, ed è per permettere l'inserimento di altre righe che solitamente sono scelti numeri

di riga come 10, 20, 30... Per inserire un'ulteriore riga è sufficiente scrivere:

35 PRINT "Costo Totale" ENTER

e la nuova istruzione è inserita prima della riga 40. Il computer permette di scegliere i numeri di riga tra 1 e 32768. Scrivere ora:

RUN ENTER

e la nuova risposta è:

Costo Totale
140

Cancellazione di una riga

È possibile cancellare la riga 35 scrivendo:

35 ENTER

È come se una riga vuota avesse sostituito la precedente.

STAMPA

È essenziale sottolineare l'importanza dell'istruzione **PRINT**. Si può ottenere la stampa di una parola o di una frase, usando sia le virgolette che l'apostrofo; per esempio:

PRINT "pezzo di cioccolata" ENTER

È possibile stampare il valore delle variabili scrivendo istruzioni quali:

PRINT pezzo ENTER

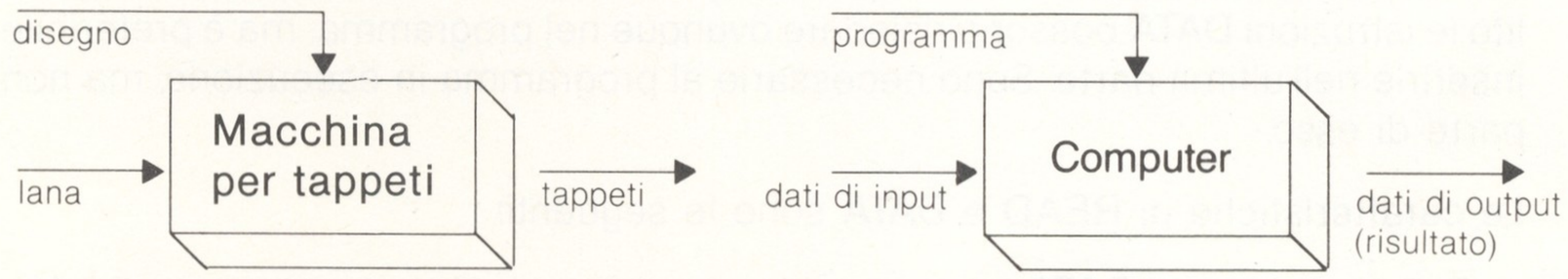
senza usare le virgolette.

In seguito si vedrà la versatilità dell'istruzione PRINT nel Superbasic, per stampare un testo o un qualsiasi altro dato esattamente nel punto dello schermo desiderato. Per il momento le due possibilità ora innanzi esaminate sono sufficientemente utili:

stampa di un testo,
stampa dei valori delle variabili.

ISTRUZIONI: INPUT, READ E DATA

Una macchina che produce tappeti ha bisogno della lana come elemento fondamentale. Produrrà poi il tappeto a seconda del disegno, indipendentemente dal tipo di lana.



In un computer è realizzato un processo analogo. Se i dati sono inseriti nel computer con l'istruzione **LET** si incontrano due svantaggi:

scrivere un'istruzione **LET** è laborioso
modificare i dati è altrettanto laborioso

Scrivere:

NEW ENTER

in tal modo il programma memorizzato in precedenza è cancellato, ed il QL è pronto a riceverne un altro.

Scrivere ora:

```
10 LET prezzo = 15 ENTER
20 PRINT "quante penne?" ENTER
30 INPUT penne ENTER
40 LET costo = prezzo * penne ENTER
50 PRINT costo ENTER
RUN ENTER
```

Il programma si interrompe alla riga 30, chiedendo il numero di penne. Si può scrivere per esempio:

```
4 ENTER
```

Non dimenticare di premere ENTER . Il risultato è:

```
60
```

Le istruzioni **LET** e **INPUT** sono utili solo per inserire nel computer quantità limitate di dati. Il Superbasic, come la maggior parte dei Basic, utilizza un altro sistema di lettura di dati noto come **READing** dalle istruzioni **DATA**.

È possibile, quindi, riscrivere il programma precedente in una nuova forma, ottenendo gli stessi risultati senza però alcuna interruzione. Scrivere:

```
NEW ENTER
10 READ prezzo, penne ENTER
20 LET costo = prezzo * penne ENTER
30 PRINT costo ENTER
40 DATA 15, 4 ENTER
RUN ENTER
```

Il risultato è:

```
60
```

esattamente come prima.

Ogni volta che il programma è eseguito, è necessario specificare al Superbasic dove cominciare a leggere l'istruzione **DATA**. Il comando **RESTORE** seguito dai numeri di riga di **DATA**, o l'istruzione **CLEAR** hanno questa funzione.

Entrambe le istruzioni possono essere inserite all'inizio del programma.

Successivamente il programma assegna i valori specificati nell'istruzione **DATA** alle variabili definite nell'istruzione **READ**, esattamente nello stesso ordine. Di solito le istruzioni **DATA** possono risiedere ovunque nel programma, ma è preferibile inserirle nell'ultima parte. Sono necessarie al programma in esecuzione, ma non parte di esso.

Le caratteristiche di **READ** e **DATA** sono le seguenti:

1. Tutte le istruzioni **DATA** sono considerate come una lunga sequenza di informazioni. Queste informazioni sono numeri, ma potrebbero essere anche parole o lettere.
2. Ogni volta che viene eseguita un'istruzione **READ**, le informazioni necessarie sono trasferite dall'istruzione **DATA** nelle variabili dell'istruzione **READ**.
3. Il sistema conserva le tracce delle informazioni che sono state lette (**READ**) per mezzo di un registro interno. Se il programma cerca di leggere più informazioni di quante ne esistano in tutte le istruzioni **DATA**, segnala un errore.

NOMI DI VARIABILI

Le variabili sono state chiamate con nomi (tipo: cane, pezzi...) per motivi ben precisi:

Un nome non può comprendere alcuno spazio;
Un nome deve iniziare con una lettera;
Un nome deve essere composto da lettere, cifre, dai simboli $, %, __ (sottoli-

neatura). I simboli $ e % hanno funzione particolari, spiegate più avanti, mentre la sottolineatura può rendere più leggibili i nomi delle variabili.

Il Superbasic non distingue le lettere maiuscole dalle minuscole. Così nomi come SCATOLE o scatole hanno lo stesso valore.

Il numero massimo di caratteri che possono essere contenuti in un nome di variabile è 255.

Le variabili sono usate nel Superbasic anche per altre funzioni. Vi è grande libertà nella scelta dei nomi, per rendere i programmi più facili da interpretare. Nomi come totale, conto, penne sono più utili di nomi come Z, Q, P.

TEST SUL CAPITOLO 2

In questo test si possono ottenere al massimo 21 punti. Controllare le risposte a pag. 14.

1. Come può essere immaginato un numero in memoria?
2. Indicare i due sistemi per memorizzare un numero e creare una casella. (2 punti)
3. Come è possibile stampare il valore di una casella?
4. Come vengono solitamente definite nel linguaggio tecnico le caselle?
5. Quando una casella acquista un valore per la prima volta?
6. Una variabile è così chiamata perché è possibile modificarne il valore non appena il programma è eseguito. Quale è il metodo per ottenere tale cambiamento?
7. Il simbolo = in un'istruzione **LET** non ha lo stesso significato del simbolo matematico. Quale è il suo significato?
8. Cosa accade se si inserisce un'istruzione non numerata?
9. Cosa accade se si inserisce un'istruzione numerata?
10. Quale è la funzione delle virgolette in un'istruzione **PRINT**?
11. Che cosa succede se non vengono usate le virgolette in un'istruzione **PRINT**?
12. Quali sono le differenze fra le istruzioni **LET** e **INPUT**?
13. Quale istruzione in un programma non è mai eseguita?
14. Quale è la funzione di un'istruzione **DATA**?
15. Come può essere chiamata una casella?
16. Scrivere tre nomi correnti di variabili utilizzando lettere, lettere e numeri, lettere e sottolineature. (3 punti)
17. Perché nel Superbasic è così importante la barra spaziatrice?
18. Perché nella programmazione è importante la scelta dei nomi per le variabili?

RISPOSTE AL TEST SUL CAPITOLO 2

1. Un numero memorizzato può essere immaginato come una casella con un nome e al cui interno possono essere inseriti dei numeri.

2. Un'istruzione **LET** che usa un nome per la prima volta creerà una casella, ad esempio:

   LET conto = 1 (1 punto)

   Un'istruzione **READ** che usa un nome per la prima volta produce i medesimi effetti, ad esempio:

   READ conto (1 punto)

3. È possibile visualizzare il valore di una casella con un'istruzione **PRINT**.

4. Il nome tecnico per indicare una casella è "variabile" perché il suo valore può mutare durante l'esecuzione del programma.

5. Una variabile acquista il suo primo valore quando è usata prima in un'istruzione **LET**, **INPUT** o **READ**.

6. Una modifica del valore di una variabile è provocata dall'esecuzione di un'istruzione **LET**.

7. Il simbolo = in un'istruzione **LET** indica un'operazione:

   "Valutazione di tutto ciò che è contenuto nella parte destra e inserimento in una casella nella parte sinistra".

8. Un'istruzione non numerata è eseguita immediatamente.

9. Un'istruzione numerata è memorizzata.

10. Le virgolette in un'istruzione **PRINT** indicano che il testo che si trova al loro interno deve essere stampato.

11. Se non sono usate le virgolette, è stampato il valore della variabile.

12. Un'istruzione **INPUT** provoca una pausa nel programma.

13. Le istruzioni **DATA** non sono mai eseguite.

14. Le istruzioni DATA sono utili per assegnare i valori alle variabili di istruzioni **READ**.

15. La parola che indica il nome di una variabile è l'identificatore o nome della variabile .

16. Esempi:

    i. giorno
    ii. giorno__23
    iii. giorno__del__mese

17. La barra spaziatrice è importante perché inserisce spazi prima e dopo le parole chiave, per non confonderle con nomi di variabili.

18. La libera scelta dei nomi è importante perché rende i programmi più comprensibili.

CONTROLLA IL PUNTEGGIO

Da 18 a 21. Ottimo. Proseguire la lettura.
Da 16 a 17 Buono. Rileggere alcune parti del capitolo.
Da 14 a 15 Sufficiente. Rileggere alcune parti del capitolo e ripetere il test.
Meno di 14. Rileggere con attenzione il capitolo e ripetere il test.

PROBLEMI SUL CAPITOLO 2

1. Eseguire una simulazione per determinare il valore di tutte le variabili non appena ogni riga del seguente programma è eseguita:

```
10 LET ore = 40 ENTER
20 LET tariffa = 3 ENTER
30 LET salario = ore * tariffa ENTER
40 PRINT ore, tariffa, salario ENTER
```

2. Scrivere e verificare un programma, simile al problema di cui al punto 1, per calcolare l'area di un tappeto di 3 metri di larghezza e 4 metri di lunghezza. Utilizzare le variabili lunghezza, larghezza e area.

3. Riscrivere il programma del problema di cui al punto 1, utilizzando due istruzioni **INPUT** al posto delle istruzioni **LET**.

4. Riscrivere il programma del problema di cui al punto 1, in modo che i dati di input (40 e 3) compaiano in un'istruzione **DATA** e non in un'istruzione **LET**.

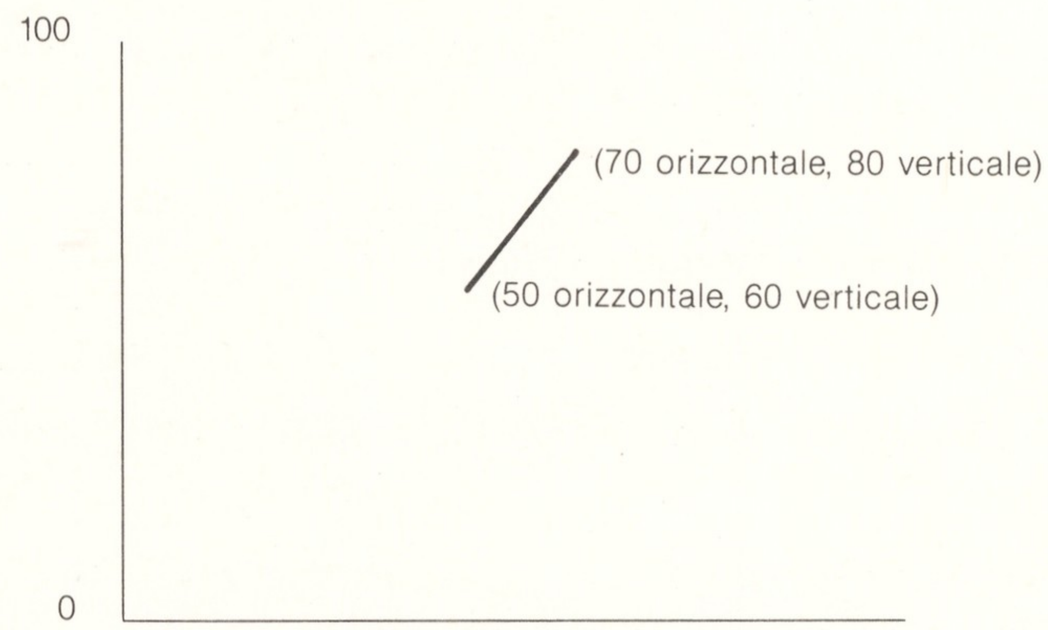
5. Riscrivere il programma del problema di cui al punto 2, usando un differente metodo di inserimento dati. Utilizzare **READ** e **DATA**, se inizialmente si è utilizzato **LET** e viceversa.

CAPITOLO 3
DISEGNARE SULLO SCHERMO

LO SCHERMO

Il QL, collegato ad un televisore o ad un monitor, dispone di due modalità di schermo. Il **MODE 8** ha una risoluzione grafica di 256 per 256 pixel con otto colori. Il **MODE 4** ha una risoluzione di 512 per 256 pixel con quattro colori e un massimo di 85 caratteri per riga, se il QL è collegato ad un monitor. È stato ideato un sistema di scala grafica per evitare difficoltà nel caso si usino, in uno stesso programma, le due modalità di schermo.

Si supponga, ad esempio, di scegliere una scala verticale di 100 e di voler disegnare una linea dal punto (50, 60) al punto (70, 80).



Scala Grafica

UNA LINEA COLORATA

È necessario prestare attenzione a tre elementi:

PAPER (colore del fondo)
INK (colore dei caratteri sullo schermo)
LINE (coordinate iniziali e finali)

Il seguente programma disegna una linea rossa su un fondo bianco

```
NEW ENTER
10 PAPER 7: CLS ENTER
20 INK 2 ENTER
30 LINE 50, 60 TO 70, 80 ENTER
RUN ENTER
```

Nella riga 10 la scelta del colore di fondo è la prima istruzione, ma perché sia possibile vederne gli effetti è necessario un ulteriore comando, **CLS**, la cui funzione è quella di pulire lo schermo.

COLORI E MODALITÀ DI SCHERMO

Il QL ha due gradi di risoluzione grafica. L'attivazione di una modalità determina non solo la risoluzione, ma anche il numero dei colori ottenibili.

Il **MODE 8** dispone di otto colori fondamentali.
Il **MODE 4** dispone di quattro colori fondamentali.

Ogni colore è codificato nella seguente tabella:

Codice	Effetti	
	MODE 8	MODE 4
0	Nero	Nero
1	Blue	Nero
2	Rosso	Rosso
3	Magenta	Rosso
4	Verde	Verde
5	Ciano	Verde
6	Giallo	Bianco
7	Bianco	Bianco

Ad esempio, l'istruzione **INK 3** seleziona il colore magenta nel MODE 8 e il colore rosso nel MODE 4.

In seguito sarà spiegato come mescolare i colori di base per creare ombre e combinazioni speciali.

EFFETTI CASUALI

È anche possibile ottenere particolari effetti usando sequenze di numeri casuali, che possono essere generati con la funzione **RND**.

Ad esempio:

PRINT RND (1 TO 6) ENTER

stampa un numero intero compreso tra 1 e 6, proprio come se si lanciasse un dado:

Il programma seguente mostra come fare:

```
NEW ENTER
10 LET Dado = RND (1 a 6) ENTER
20 PRINT Dado ENTER
RUN ENTER
```

Ogni volta che il programma è eseguito, si ottengono diverse serie di numeri. È possibile ottenere numeri casuali generati da un insieme qualsiasi di numeri, ad esempio:

```
PRINT RND (0 TO 100)
```

stampa un numero fra 0 e 100. Quando l'insieme dei numeri casuali ha come limite inferiore 0 è possibile omettere il primo numero nell'istruzione e scrivere:
RND (100)

```
NEW ENTER
10 PAPER 7 : CLS ENTER
20 INK RND (5) ENTER
30 LINE 50, 60 TO RND (100), RND (100) ENTER
RUN ENTER
```

Il programma genera una linea retta che, partendo da un punto vicino al centro dello schermo (50, 60) termina in un punto casuale (RND (100), RND (100)). I colori utilizzabili dipendono dalla modalità di schermo scelta.

BORDI

La parte di schermo sulla quale sono apparse le linee è chiamata **finestra**. È possibile cambiare la grandezza e la posizione di una finestra o crearne una a piacere e disegnare un bordo attorno alla finestra. Lo schermo può essere immaginato come un reticolo di minuscoli rettangoli, ognuno dei quali è comandabile dal computer. Nel Mode 8, detto a bassa risoluzione, ci sono 256 pixel sullo schermo in orizzontale e 256 in verticale. Nel Mode 4, detto ad alta risoluzione grafica, ci sono 512 pixel in orizzontale e 256 in verticale. La grandezza del pixel dipende dalla scelta del Mode. È possibile ottenere un bordo all'interno dell'altro, scrivendo:

```
BORDER 4, 2 ENTER
```

SEMPLICI CICLI

I computer sono in grado di compiere operazioni in tempi molto brevi, ma non sarebbe possibile sfruttare le loro capacità se ogni azione non fosse tradotta in un'istruzione. Un costruttore ha dei problemi simili. Se vuole che un operaio posi un centinaio di mattonelle, deve dirgli esattamente ciò che vuole. Non può dargli cento diverse istruzioni. Nel Basic un sistema per ottenere la ripetizione di sequenze di istruzioni è usare le istruzioni **GO TO** (o **GOTO**), ad esempio:

```
NEW ENTER
10 PAPER 6 : CLS ENTER
20 BORDER 1, 2 ENTER
30 INK RND (5) ENTER
40 LINE 50, 60 TO RND (100), RND (100) ENTER
50 GOTO 30 ENTER
RUN ENTER
```

Tuttavia non è questo il modo migliore per ottenere la ripetizione di sequenze di istruzioni.

Scomponiamo il programma precedente in "blocchi" logici:

```
10 parti fisse - non ripetibili
20
30 parti mutabili - ripetibili
40
50 programma di controllo
```

È così possibile riscrivere il programma omettendo l'istruzione **GOTO** ed usando le istruzioni **REPeat** e **END REPeat**

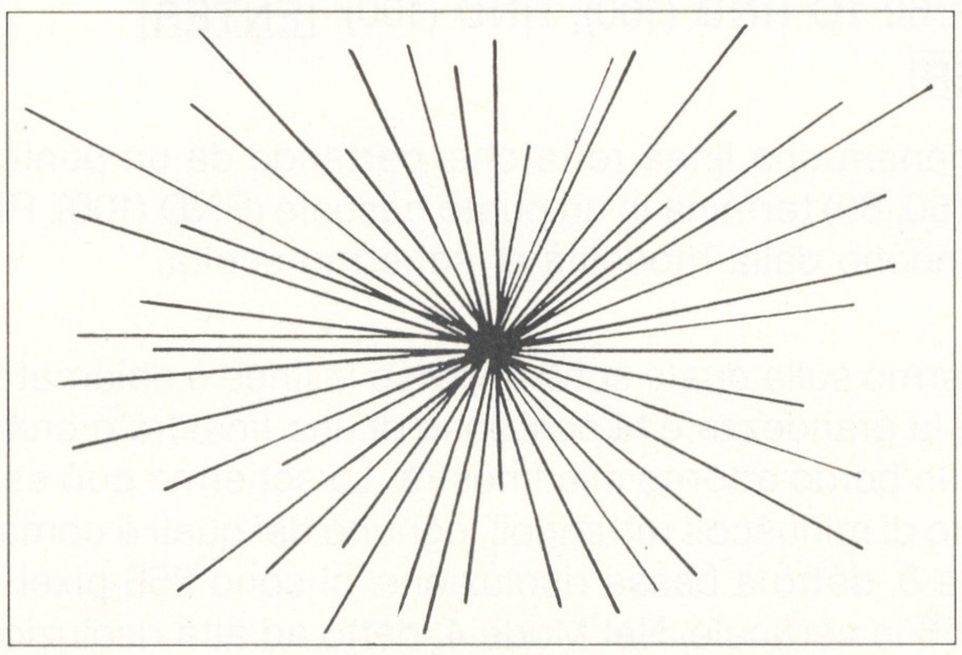
```
NEW ENTER
10 PAPER 6 : CLS ENTER
20 BORDER 1, 2 ENTER
30 REPEAT stella ENTER
40      INK RND (5) ENTER
50      LINE 50, 60, TO RND (100), RND (100) ENTER
60 END REPEAT stella ENTER
RUN ENTER
```

Alla struttura REPeat è stato assegnato un nome: stella. La struttura è composta da due istruzioni:

```
REPeat stella
END REPeat stella
```

Tutto ciò che è racchiuso fra le due istruzioni è il contenuto della struttura. L'uso di lettere maiuscole indica che **REP** è un'abbreviazione corretta di **REPEAT**.

Il programma disegna infinite linee colorate che formano una stella come nella figura seguente.



Il programma STELLA

È possibile interrompere l'esecuzione del programma, premendo contemporaneamente i tasti **CTRL** e **SPAZIO** (**BREAK**).

Il Superbasic è dotato di un metodo per interrompere processi ripetitivi, che altrimenti sarebbero infiniti. L'istruzione EXIT ha questa funzione, ma deve essere usata in modo appropriato.

Per esempio, è possibile ampliare la scelta dei colori della stella disegnata dal programma precedente, con una piccola variazione.

```
40 INK RND (0 to 6) ENTER
```

Quando l'istruzione genera il numero 6, il colore delle linee della stella e del fondo sono uguali. Questa può essere una buona ragione per terminare la ripetizione delle istruzioni. Si può modificare il programma nel modo seguente.

```
NEW ENTER
10 PAPER 6 : CLS ENTER
20 BORDER 1, 2 ENTER
30 REPeat stella ENTER
40      LET colore = RND (6) ENTER
50      IF colore = 6 THEN EXIT stella ENTER
60      INK colore ENTER
70      LINE 50, 60, TO RND (100), RND (100) ENTER
80 END REPeat stella ENTER
```

Il programma è eseguito fino a che il colore non diventa 6.

A questo punto è interrotta la ripetizione delle istruzioni interne alla struttura REPEAT e l'esecuzione raggiunge la riga 80. Se non vi sono altre righe di programma dopo la 80, il programma termina. È stato così introdotto un altro concetto molto importante: come prendere una decisione.

```
IF colore = 6 THEN EXIT stella
```

Questa è un'altra struttura propria del Superbasic. La scelta fra il fare o no qualche operazione è definita decisione binaria. La sua forma può essere espressa:

```
IF condizione THEN istruzione (i)
```

Le ripetizioni (iterazioni) e le decisioni (scelte) sono le strutture fondamentali di un qualsiasi programma.

TEST SUL CAPITOLO 3

Nel test seguente è possibile ottenere al massimo 13 punti. Controllare il punteggio con le risposte a pag. 20.

1. Che cosa è un pixel?
2. Quanti pixel compaiono sullo schermo del Mode a bassa risoluzione?
3. Quanti pixel ci sono in verticale nel Mode a bassa risoluzione?
4. Quali sono i due numeri che determinano l'"indirizzo" o la posizione di un punto grafico sullo schermo?
5. Quanti colori sono utilizzabili nel Mode a bassa risoluzione?
6. Indicare le parole chiave che permettono di: (5 punti)
   i. disegnare una linea
   ii. selezionare il colore del testo stampato
   iii. selezionare il colore dello sfondo
   iv. disegnare una cornice (5 punti)
7. Quali sono le istruzioni che aprono e chiudono una struttura REPeat?
8. Come è possibile interrompere un'istruzione REPeat?
9. Perché nel Superbasic le strutture REPeat END REPeat hanno dei nomi?

RISPOSTE AL TEST SUL CAPITOLO 3

1. Un pixel è l'area di schermo più piccola che può essere visualizzata sullo schermo.
2. Nella modalità di schermo a bassa risoluzione ci sono 256 pixel orizzontali
3. Nella modalità di schermo a bassa risoluzione ci sono 256 pixel verticali.
4. I valori che determinano la posizione di un punto sono:
   il valore della coordinata verticale, da 0 a 100.
   il valore della coordinata orizzontale, da 0 ad un numero calcolato da sistema.
5. Nelle modalità di schermo a bassa risoluzione sono disponibili otto colori compresi il bianco ed il nero.
6. i. LINE (5 punti)
   ii. INK
   iii. PAPER
   iv. BORDER
7. Le istruzioni che aprono e chiudono una struttura REPeat sono: REPeat nome... end REPeat nome.
8. Una struttura REPeat termina quando è eseguita un'istruzione EXIT.
9. In Superbasic le iterazioni hanno dei nomi, in modo che sia possibile fare riferimento ad esse facilmente.

CONTROLLO DEL PUNTEGGIO

Da 11 a 13. Ottimo. Proseguire la lettura.

Da 8 a 10. Buono. Rileggere alcune parti del capitolo.

Da 6 a 7. Sufficiente. Rileggere alcune parti del capitolo e ripetere il test.

Meno di 6. Rileggere con attenzione il capitolo e ripetere il test.

PROBLEMI SUL CAPITOLO 3

1. Scrivere un programma per disegnare linee rette su tutto lo schermo. Le linee dovranno avere lunghezza, direzione e colore casuali e ciascuna dovrà partire dal punto finale della precedente.

2. Scrivere un programma per disegnare linee casuali con la sola condizione che ogni linea abbia un punto d'inizio casuale sul margine sinistro dello schermo.

3. Scrivere un programma per disegnare linee casuali con la sola condizione che le linee abbiano lo stesso punto d'inizio sul margine inferiore dello schermo.

4. Scrivere un programma per disegnare linee di lunghezza, punto iniziale e colore casuali. Tutte le linee devono essere orizzontali.

5. Come il problema di cui al punto 4, ma le linee devono essere verticali.

6. Scrivere un programma per disegnare un quadrato a spirale in modo tale che ogni linea generi un colore casuale.

   NOTA BENE: determinare le coordinate di alcuni angoli, poi inserirle in gruppi di quattro. Si scoprirà un modello.

CAPITOLO 4
VARIABILI ALFABETICHE O STRINGHE

In questo capitolo sono introdotte alcune idee sul trattamento delle parole e delle stringhe di caratteri, proponendo semplici approcci per determinare la lunghezza media delle parole.

Con la locuzione "stringa di caratteri" (chiamata semplicemente stringa) s'intende una qualsiasi sequenza di caratteri (lettere, numeri, simboli), che compaiono sulla tastiera. Le stringhe sono utilizzate in modo simile ai numeri, ma naturalmente non si possono eseguire le stesse operazioni. Non è possibile moltiplicare o sottrarre due stringhe. Si possono tuttavia unire, separare o trasformare nel modo desiderato.

NOMI E CASELLE PER STRINGHE

Come esistono le variabili per memorizzare numeri, è possibile definire variabili per memorizzare stringhe. Ad esempio, se si vogliono memorizzare parole come:

PRIMO SECONDO TERZO
E
GENNAIO FEBBRAIO MARZO

scegliamo dei nomi di variabile:

giorno$ ☐ mese$ ☐

Notare il simbolo del dollaro. Le variabili di stringa sono internamente diverse da quelle dei numeri, e il Superbasic ha bisogno di riconoscere le une dalle altre. Tutti i nomi delle variabili di stringa devono terminare con il simbolo $, altrimenti sarebbero confuse con le variabili numeriche.

L'istruzione LET funziona come quando si riferisce a numeri.

Scrivendo:

LET giorno$ = "PRIMO" ENTER

è creata la variabile giorno$ che contiene il valore "PRIMO" al suo interno:

giorno$ | PRIMO |

Le virgolette non sono memorizzate, ma vengono utilizzate per indicare ciò che deve essere memorizzato nella variabile. Controllare scrivendo:

PRINT giorno$ ENTER

sullo schermo appare il contenuto della variabile:

PRIMO

È corretto utilizzare il simbolo di apostrofo al posto delle virgolette.

LUNGHEZZA DELLE STRINGHE

Con il Superbasic è possibile calcolare facilmente la lunghezza o il numero dei caratteri che compongono una stringa. Ad esempio:

PRINT LEN (giorno$) ENTER

Se la variabile giorno$ contiene PRIMO sullo schermo è stampato il valore 5. Infatti:

```
NEW ENTER
10 LET giorno$ = "PRIMO" ENTER
20 PRINT LEN (giorno$)ENTER
RUN ENTER
```

Sullo schermo appare il numero

5

LEN è una parola chiave nel Superbasic.

Un diverso metodo per ottenere lo stesso risultato usando una variabile di stringa e una variabile numerica, è il seguente:

```
NEW ENTER
10 LET giorno $ = "PRIMO" ENTER
20 LET lungh = LEN (giorno$) ENTER
30 PRINT lungh = ENTER
RUN ENTER
```

Sullo schermo ancora una volta è stampato il valore:

come nell'esempio precedente, e le due variabili contengono i valori:

giorno$	PRIMO	lungh	5

Si supponga, ad esempio, di voler calcolare la lunghezza media delle parole:

PRIMO, DI, FEBBRAIO

Quando un problema appare più difficile del previsto, è buona abitudine schematizzare la sequenza logica delle operazioni prima di scrivere il programma.

COME SCRIVERE UN PROGRAMMA

1. Memorizzare le tre parole in variabili.
2. Calcolare la lunghezza e memorizzarla.
3. Calcolare la media.
4. Stampare il risultato.

PROGRAMMA

```
NEW ENTER
10 LET giorno$ = "PRIMO" ENTER
20 LET parola$ = "DI" ENTER
30 LET mese$ = "FEBBRAIO" ENTER
40 LET lungh1 = LEN (giorno$) ENTER
50 LET lungh2 = LEN (parola$) ENTER
60 LET lungh3 = LEN (mese$) ENTER
70 LET somma = lungh1 + lungh2 + lungh3 ENTER
80 LET media = somma/3 ENTER
90 PRINT media ENTER
RUN ENTER
```

Il simbolo "/": è il simbolo della operazione di divisione. Il risultato del programma è

5

Si sono create otto variabili:

giorno$	PRIMO	lungh1	5
parola$	DI	lungh2	2
mese$	FEBBRAIO	lungh3	8
		somma	15
		media	5

Il programma può essere sintetizzato in una sola riga di istruzioni, ma sarebbe meno facile da leggere. Un ragionevole compromesso è ottenuto usando il simbolo & che implica l'operazione di unione di due stringhe.

Scrivere ora:

```
NEW ENTER
10 LET giorno$ = "PRIMO" ENTER
20 LET parola$ = "DI" ENTER
30 LET mese$ = "FEBBRAIO" ENTER
40 LET frase$ = giorno$ & parola$ & mese$ ENTER
50 LET lungh = LEN (frase$) ENTER
60 PRINT lungh/3 ENTER
```

RUN ENTER

Il risultato è 5 come in precedenza, ma ci sono alcune differenze.

giorno$	PRIMO	lungh	15
parola$	DI		
mese$	FEBBRAIO		
frase$	PRIMODIFEBBRAIO		

È possibile una ulteriore semplificazione utilizzando le istruzioni **READ** e **DATA** in luogo delle prime tre istruzioni **LET**. Scrivere:

```
NEW ENTER
10 READ giorno$, parola$, mese$ ENTER
20 LET frase$ = giorno$ & parola$ & mese$ ENTER
30 LET lungh = LEN (frase$) ENTER
40 PRINT lungh/3 ENTER
50 DATA "PRIMO", "DI", "FEBBRAIO" ENTER
RUN ENTER
```

L'istruzione READ crea le variabili in modo analogo all'istruzione LET.

Il simbolo del dollaro identifica la variabile come stringa. $ deve essere sempre posto al termine del nome della variabile. Parole quali "PRIMO", "DI" sono i valori delle variabili di stringa chiamate: giorno$ e parola$.

EFFETTI CASUALI

È possibile utilizzare i codici di carattere per generare sequenze di lettere casuali. Le lettere maiuscole dalla A alla Z hanno i codici corrispondenti dal 65 al 90. La funzione **CHR$** trasforma questi codici in lettere. Il seguente programma stampa ad esempio la lettera B,

```
NEW ENTER
10 LET codice = 66 ENTER
20 PRINT CHR$ (codice) ENTER
RUN ENTER
```

Il successivo genera sequenze delle tre lettere A, B, C, in ordine casuale, finché non è formata la parola CAB.

```
NEW [ENTER]
10 REPeat caso [ENTER]
20      LET primo$ = CHR$ (RND(65 to 67)) [ENTER]
30      LET secondo$ = CHR$ (RND(65 to 67)) [ENTER]
40      LET terzo$ = CHR$ (RND(65 to 67)) [ENTER]
50      LET parola$ = primo$ & secondo$ & terzo$ [ENTER]
60      PRINT ! parola$ ! [ENTER]
70      IF parola$ = "CAB" THEN EXIT caso [ENTER]
80 END REPeat caso [ENTER]
```

TEST SUL CAPITOLO 4

In questo test è possibile ottenere al massimo 10 punti. Controllare il punteggio con le risposte a pag. 26.

1. Che cosa è una stringa di caratteri?
2. Quale è l'abbreviazione del termine stringa di caratteri?
3. Cosa si intende con la locuzione "variabile di stringa"?
4. Come si pronuncia una parola come: parola$?
5. Che parola chiave è usata per calcolare il numero di caratteri da cui è composta una stringa?
6. Quale simbolo è usato per unire due stringhe?
7. Gli spazi possono essere parti di una stringa. Come sono definiti i limiti di una stringa?
8. Quando un'istruzione come:

   LET carne$ = "bistecca"

   è eseguita, sono memorizzate le virgolette?
9. Quale funzione trasforma un numero di codice in una lettera?
10. Come si possono generare sequenze casuali di lettere maiuscole?

RISPOSTE AL TEST SUL CAPITOLO 4

1. Una stringa è una sequenza di caratteri (come lettere, numeri o altri simboli).
2. Il termine "stringa di carattere" è abbreviato con stringa.
3. Il nome di una variabile stringa termina sempre con il simbolo "$".
4. Nomi come "parola$" sono pronunciati "paroladollaro".
5. La parola chiave LEN calcola la lunghezza o il numero di caratteri da cui è composta una stringa.
6. Il simbolo che indica l'operazione di unione fra due stringhe è "&".
7. I limiti di una stringa possono essere definiti dalle virgolette o con l'apostrofo.
8. Le virgolette non sono parte della stringa e non sono memorizzate.
9. La funzione è **CHR$**. Bisogna usarla con parentesi, ad esempio **CHR$ (66)** o **CHR$ (RND(65 to 67))**.
10. Le lettere casuali sono generate con le istruzioni:

```
codice lettera = RND (65 to 90)
PRINT CHR$ (codice lettera)
```

CONTROLLO DEL PUNTEGGIO

Da 9 a 10. Ottimo. Proseguire la lettura.

Da 7 a 8. Buono. Rileggere alcune parti del capitolo.

Da 5 a 6. Sufficiente. Rileggere alcune parti del capitolo e ripetere il test.

Meno di 5. Rileggere con attenzione il capitolo e ripetere il test.

PROBLEMI SUL CAPITOLO 4

1. Memorizzare le parole "Buono" e "giorno" in due variabili separate. Utilizzare un'istruzione LET per unire i valori delle due variabili in una terza variabile. Stampare il risultato.
2. Memorizzare le seguenti parole in quattro diverse caselle: luce, lascia, essere, là.

   Unire le parole per formare una frase aggiungendo spazi e un punto. Memorizzare l'intera frase in una variabile, sent$, e stampare la frase e il numero totale dei caratteri da cui è composta.
3. Scrivere un programma che utilizzi la parola chiave:

   CHR$RND (69 to 90)

   per creare un centinaio di parole di tre lettere. Controllare se vi sono parole di senso compiuto. Provare gli effetti di:

   a) ";" al termine di un'istruzione **PRINT**
   b) "!" prima delle variabili o di informazioni da stampare.

CAPITOLO 5
CONSIGLI UTILI

Ora che si sono esaminati alcuni semplici programmi, si è potuto constatare quanto siano utili i seguenti accorgimenti.

1. Usare lettere minuscole per indicare nomi di variabili, o strutture di ripetizione.
2. Incolonnare spostando verso destra le istruzioni contenute fra REPeat e END REPeat.
3. Utilizzare nomi di variabile che ne richiamino il contenuto.
4. Modificare un programma:

   sostituzione di una riga
   inserimento di una nuova riga
   cancellazione di una riga

ESEMPIO DI PROGRAMMA

A questo punto è necessario imparare a studiare i programmi e capirne il funzionamento. Il loro meccanismo deve essere ben assimilato, e nei prossimi capitoli è evitata la costante ripetizione di:

NEW prima di ogni programma
ENTER al termine di ogni riga
RUN per eseguire un programma

La loro omissione nel testo permette di fissare l'attenzione sugli aspetti più importanti di un programma.

Per esempio, il programma seguente crea delle serie di lettere maiuscole casuali fino a che non compare una Z. Non vengono scritte le parole **NEW, RUN** o il simbolo di ENTER , ma è necessario usarle.

```
10 REPeat lettere
20     LET codice_lettera=RND (65 TO 90)
30     cap$=CHR$ (codice_lettera)
40     PRINT cap$
50     IF cap$="Z" THEN EXIT lettere
60 END REPeat lettere
```

In questo e nei capitoli seguenti, le istruzioni dei programmi sono riportate senza il simbolo di ENTER . Anche i comandi diretti sono scritti senza il simbolo ENTER . È tuttavia necessario utilizzare il tasto ENTER .

NUMERAZIONE AUTOMATICA DI RIGA

È noioso inserire manualmente i numeri di riga. È più facile scrivere:

```
AUTO
```

prima di cominciare a programmare, per consentire al QL di generare automaticamente il numero di riga:

```
100
```

Continuando a scrivere nelle strutture del programma, sullo schermo appaiono le istruzioni:

```
100 PRINT "PRIMO"
110 PRINT "SECONDO"
120 PRINT "FINE"
```

Per eliminare la numerazione automatica usare la funzione **BREAK**: premere contemporaneamente i tasti **CTRL** e **SPAZIO**. Sullo schermo è visualizzato il seguente messaggio:

```
130 operazione interrotta
```

e la riga 130 non è parte del programma.

Se si commette un errore e non si desidera interrompere la numerazione automatica delle righe, è possibile continuare la scrittura delle istruzioni e modificare la riga più avanti.

Se si vuole cominciare da un determinato numero di riga, ad esempio 600, con un passo diverso da 10, e cioè 5, scrivere:

AUTO 600, 5

Le righe sono così numerate: 600, 605, 610 ecc.

Per eliminare **AUTO**, premere contemporaneamente i tasti **CTRL** e **SPAZIO**.

MODIFICA DI UNA RIGA

Per modificare una riga è sufficiente scrivere **EDIT**, seguito dal numero di riga, ad esempio:

EDIT 110

La riga è visualizzata sullo schermo con il cursore alla fine della stessa:

110 PRINT "Secondo"

Il cursore può essere spostato con i tasti usuali.

[←] verso sinistra
[→] verso destra

Per cancellare un carattere alla sinistra del cursore premere:

CTRL e [←]

Per cancellare un carattere nella posizione del cursore, premere:

CTRL e [→]

Il carattere alla destra del cursore è spostato per chiudere lo spazio.

UTILIZZO DELLE CARTUCCE

Prima di utilzzare una nuova cartuccia è necessario formattarla ed assegnarle un nome. Il nome di una cartuccia deve rispettare le regole esaminate a proposito dei nomi di variabile nel Superbasic, ma il nome non può essere lungo più di 10 caratteri. È una buona idea scrivere il nome della cartuccia su di essa con un'etichetta adesiva.

È opportuno tenere almeno una copia di back up (di sicurezza) di qualsiasi programma. Seguire le istruzioni contenute nella parte Informazioni del Manuale.

ATTENZIONE

SE UNA CARTUCCIA CONTENENTE PROGRAMMI E/O DATI VIENE FORMATTATA, TUTTI I PROGRAMMI E/O I DATI VENGONO CANCELLATI

COME MEMORIZZARE UN PROGRAMMA

Il seguente programma disegna dei bordi, larghi 8 pixel, in rosso (codice 2), in tre finestre chiamate: #0, #1, #2.

```
100 REMark Bordi
110 FOR K=0 TO 2: BORDER #K, 8, 2
```

Il programma può essere memorizzato scrivendo:

SAVE mdv1_bord

Il programma è memorizzato in un file chiamato **BORD** su una cartuccia.

VERIFICA DI UNA CARTUCCIA

È possibile sapere quali programmi o quali dati sono contenuti su una cartuccia, inserendola nel Microdrive 1 e scrivendo:

DIR mdv1_

Sullo schermo è visualizzato l'elenco dei file contenuti su quella cartuccia. Se la cartuccia si trova nel Microdrive 2 scrivere:

DIR mdv2_

COPIARE FILE E PROGRAMMI

Una volta che il programma è memorizzato in un file su una cartuccia, può essere copiato. È questo uno dei modi di fare una copia di back up (di sicurezza) di un programma o file. È possibile copiare il programma precedente, su un'altra cartuccia inserita nel Microdrive 2 scrivendo:

```
COPY mdv1_bord TO mdv2_bord
```

CANCELLARE UN PROGRAMMA

Per cancellare il programma chiamato prog. da una cartuccia, scrivere:

```
DELETE mdv1_prog
```

CARICARE UN PROGRAMMA

Un programma può essere trasferito da una cartuccia in memoria scrivendo:

```
LOAD mdv2_bord
```

È possibile provare il programma scrivendo:

LIST per ottenere l'elenco delle istruzioni
RUN per eseguirlo

Anziché scrivere **LOAD** seguito da RUN è possibile unire le due operazioni in un unico comando:

```
LRUN mdv2_bord
```

Il programma è caricato ed eseguito immediatamente.

FUSIONE DI PROGRAMMI

Supponiamo di aver salvato due programmi nel Microdrive 1 e cioè: prog1 e prog2:

```
100 PRINT "Primo"
```

```
110 PRINT "Secondo"
```

se si scrive:

```
LOAD mdv1_prog1
```

seguito da:

```
MERGE mdv1_prog2
```

i due programmi sono fusi in un unico programma.
Per verificarlo è sufficiente scrivere LIST e sullo schermo si vedrà:

```
100 PRINT "Primo"
110 PRINT "Secondo"
```

Se è utilizzata un'istruzione **MERGE** per concatenare due programmi, assicurarsi che tutti i numeri di riga del programma residente su cartuccia siano diversi da quelli del programma già in memoria. Altrimenti alcune righe del programma già in memoria sono sostituite con le corrispondenti righe del nuovo programma.

Prestare molta attenzione nell'uso delle cartucce. Fare sempre una copia di back up e, per evitare spiacevoli sorprese, sarebbe meglio averne due. I professionisti raramente perdono dei dati. Essi sanno, però, che anche le macchine migliori possono occasionalmente provocare dei problemi e si regolano di conseguenza.

Se si vuol chiamare un programma con un nome particolare, come ad esempio, quadrato, è una buona abitudine usare nomi come: qu1, qu2... per le versioni preliminari. Quando il programma è nella sua forma definitiva, fare almeno due copie chiamate quadrato, mentre le altre possono essere cancellate riformattando la cartuccia.

TEST SUL CAPITOLO 5

In questo test è possibile ottenere un punteggio massimo di 14 punti. Controllare il punteggio con le risposte a pag. 32.

1. Perché è preferibile utilizzare le lettere maiuscole per scrivere i nomi delle variabili nel programma?
2. Quale è lo scopo di incolonnare spostando verso destra le istruzioni all'interno di una struttura REPeat END REPeat?
3. Cosa influenza la scelta dei nomi di variabili di cicli?
4. Indicare tre sistemi per modificare un programma residente nella memoria del computer. (3 punti)
5. Quale tasto è necessario premere al termine di ogni riga di programma o dopo ogni comando?
6. Cosa è necessario fare prima di scrivere un programma?
7. Cosa è necessario inserire all'inizio di ogni riga per memorizzarla come una parte del programma?
8. Quale è l'istruzione che consente di eseguire un programma?
9. Con quale istruzione è possibile inserire un'informazione in un programma senza che questa abbia alcuna esecuzione?
10. Quali comandi memorizzano i programmi e li rintracciano nelle cartucce? (2 punti)

RISPOSTE AL TEST SUL CAPITOLO 5

1. Le lettere minuscole sono preferite per i nomi di variabili e cicli iterativi per differenziarli dai nomi delle parole chiave, che sono scritti in lettere maiuscole.
2. Lo scopo di spostare verso destra le istruzioni contenute in iterazioni e altre strutture, è quello di evidenziare l'estensione ed il contenuto.
3. I nomi di variabile devono avere un significato chiaro.

   Ad esempio scegliere conto o parola$' e non C o P$.
4. È possibile modificare un programma: (3 punti)

   sostituendo una riga
   inserendo una riga
   cancellando una riga
5. Il tasto ENTER è usato per inserire un comando o una riga di istruzioni.
6. La parola **NEW** è prima dell'inizio di un nuovo programma.
7. Per memorizzare una riga, come parte di un programma. è necessario utilizzare un numero di riga.
8. La parola **RUN** seguita da ENTER provoca l'esecuzione del programma.
9. La parola chiave **REMark** permette di inserire nel programma delle informazioni che sono ignorate durante l'esecuzione dello stesso programma.
10. Le parole chiave **SAVE** e **LOAD** permettono di memorizzare e caricare un programma. (2 punti)

CONTROLLO DEL PUNTEGGIO

Da 12 a 14. Ottimo. Proseguire la lettura.

Da 10 a 11. Buono. Rileggere alcune parti del capitolo.

Da 8 a 9. Sufficiente. Rileggere alcune parti del capitolo e ripetere il test.

Meno di 8. Rileggere con attenzione il capitolo e ripetere il test.

PROBLEMI SUL CAPITOLO 5

1. Riscrivere il seguente programma con lettere minuscole per ottenere una miglior presentazione. Aggiungere le parole NEW e RUN. Usare i numeri di riga e il simbolo ENTER solo quando si vuole memorizzare e eseguire un programma. Utilizzare REMark per dare un nome al programma.

```
LET DUE$="DUE"
LET QUATTRO$="QUATTRO"
LET SEI$=DUE$ & QUATTRO$
PRINT LEN (SEI$)
```

   dimostrare come due e quattro possono dare 7.

2. Utilizzando separatori, lettere minuscole, NEW, RUN, numeri di riga e il simbolo di ENTER, indicare come si dovrebbe scrivere ed eseguire il seguente programma:

```
REPEAT CICLO
CODICE LETTERA=RND (65 TO 90)
LET LETTERE$=CHR$ (CODICE-LETTERA)
PRINT LETTERA$
IF LETTERA$='Z' THEN EXIT CICLO
END REPEAT CICLO
```

3. Scrivere il seguente programma in forma migliore utilizzando nomi sensati per identificare le variabili.
   Riscriverlo poi in forma definitiva:

```
LET S=0
REPeat TOTAL
LET N=RND (1 TO 6)
PRINT ! N !
LET S=S+N
IF n=6 THEN EXIT TOTAL
END REPeat TOTAL
PRINT S
```

CAPITOLO 6
VETTORI
COS'È UN VETTORE

Come si è potuto osservare nei capitoli precedenti, i numeri o le stringhe di caratteri possono diventare valori di variabili. È opportuno, a questo proposito, un esempio: si supponga che quattro operai siano inviati in un paese, dove è stato trovato il petrolio. Il paese è uno dei pochi posti in cui le case hanno un nome, e quattro possono essere affittate. Tutti i nomi delle case terminano con il simbolo del dollaro.

Ovest$ Lago$ Rosa$ Quercia$

I quattro operai si chiamano:

VAL	HAL	MEL	DEL

e a ciascuno può essere assegnata una casa con uno di questi due metodi:

PROGRAMMA 1

```
100 LET ovest$="VAL"
110 LET lago$="HAL"
120 LET rosa$="MEL"
130 LET quercia$="DEL"
140 PRINT! ovest$! lago$! rosa$! quercia$
```

PROGRAMMA 2

```
100 READ ovest$, lago$, rosa$, quercia$
110 PRINT! ovest$! lago$! rosa$! quercia$!
120 DATA "VAL", "HAL", "MEL", "DEL"
```

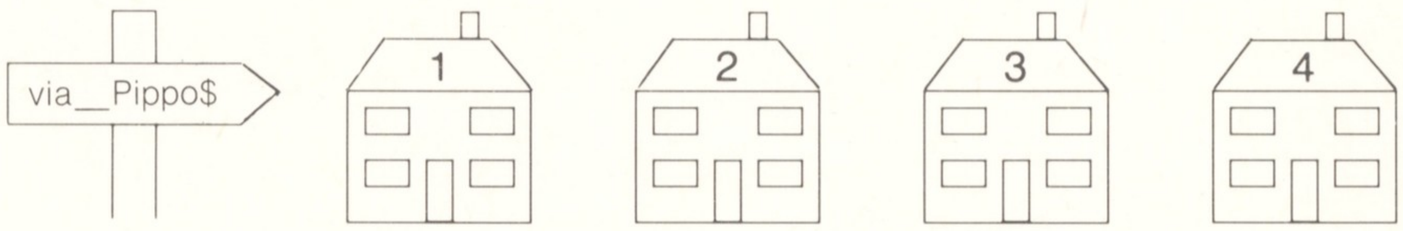
Quanto più numerosi sono i dati tanto più grandi sono i vantaggi di READ e DATA rispetto a LET. Tuttavia, quando la numerosità dei dati supera un certo limite, anche le istruzioni **READ e DATA** sono noiose da usare.

La soluzione di questo e di molti altri problemi di gestione dati, è rappresentata da un nuovo tipo di variabile, nella quale più elementi possono condividere uno stesso nome. Devono essere distinti uno dall'altro, in modo che ogni variabile abbia un numero, proprio come i numeri delle case di una stessa via. Si supponga di aver bisogno di quattro case libere in Via Pippo, numerate da 1 a 4. Nel Superbasic si dice che esiste un **vettore** di quattro case. Il nome del vettore è Via__Pippo$ e le quattro case sono numerate da 1 a 4.

Tuttavia non è possibile utilizzare queste nuove variabili come semplici variabili. È necessario indicare la dimensione del vettore. Il computer assegna uno spazio interno e deve sapere quante variabili di stringa ci sono nel vettore, e la lunghezza massima di ogni elemento del vettore. L'istruzione **DIM** ha questa funzione.

```
DIM via__Pippo$ (4,3)
                 |  |
                 |  |___ lunghezza massima di ogni elemento
                 |
                 |______ numero di elementi (stringhe)
```

Solo dopo che l'istruzione **DIM** è stata eseguita, i vettori possono essere utilizzati. Le case sono costruite, ma restano ancora vuote.



I cinque programmi di seguito riportati hanno la stessa funzione: fanno occupare le quattro case.

L'ultimo è formato da quattro righe soltanto, ma i precedenti conducono gradatamente ad esso. Si ricerca anche la massima economia.

PROGRAMMA 1

```
100 DIM via_Pippo$ (4, 3)
110 LET via_Pippo$ (1)="VAL"
120 LET via_Pippo$ (2)="HAL"
130 LET via_Pippo$ (3)="MEL"
140 LET via_Pippo$ (4)="DEL"
150 PRINT ! via_Pippo$ (1) ! via_Pippo$ (2) !
160 PRINT ! via_Pippo$ (3) ! via_Pippo$ (4)!
```

PROGRAMMA 2

```
100 DIM via_Pippo$ (4, 3)
110 READ via_Pippo$ (1), via_Pippo$ (2), via_Pippo (3), via_Pippo (4)
120 PRINT ! via_Pippo$ (1) ! via_Pippo$ (2) !
130 PRINT ! via_Pippo$ (3) ! via_Pippo$ (4)!
140 DATA "VAL", "HAL", "MEL", "DEL"
```

La costante ripetizione di via_Pippo$ è noiosa e rende il programma apparentemente disordinato. È possibile, attraverso una tecnica già conosciuta, la struttura **REPeat**, dare una nuova forma al programma.

È necessario utilizzare un "contatore", cioè un numero che è incrementato di una unità ogni volta che la struttura **REPeat** è eseguita.

PROGRAMMA 3

```
100 RESTORE 190
110 DIM via_Pippo$ (4, 3)
120 LET numero=0
130 REPeat caso
140       LET numero=numero+1
150       READ via_Pippo$ (numero)
160       IF numero=4 THEN EXIT caso
170 END REPeat caso
180 PRINT via_Pippo$(1)! via_Pippo$(2) ! via_Pippo$ (3)! via_Pippo$(4)!
190 DATA "VAL", "HAL", "MEL", "DEL"
```

Il tipo di iterazione, in cui una sequenza di istruzioni deve essere ripetuta più volte, è già ben conosciuta. Per questo tipo di operazione si utilizza una struttura chiamata ciclo **FOR**, che comprende al suo interno una funzione di contatore. Il programma diventa così:

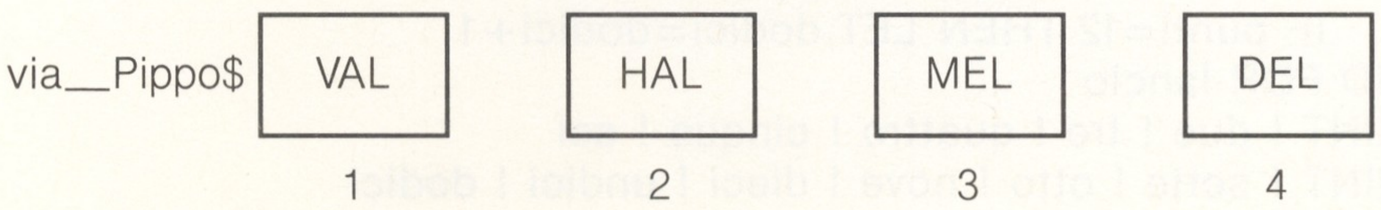
PROGRAMMA 4

```
100 RESTORE 160
110 DIM via_Pippo$ (4, 3)
120 FOR numero=1 to 4
130       READ via_Pippo$ (numero)
140       PRINT ! via_Pippo$ (numero)
150 END FOR numero
160 DATA "VAL", "HAL", "MEL", "DEL"
```

Il risultato di tutti i programmi è lo stesso:

VAL HAL MEL DEL

È quindi dimostrato che i dati sono stati memorizzati internamente nei quattro elementi di un vettore.



Il quarto programma è senza dubbio il migliore. Infatti può funzionare con semplici variazioni indipendentemente dal numero di informazioni.

Il ciclo **FOR** è abbastanza simile alla forma più semplice della struttura **REPeat**.

Le due strutture possono essere confrontate:

```
100 REPeat saluti                100 FOR saluti=1 to 40
110       PRINT "ciao"           110       PRINT "ciao"
120 END REPeat saluti            120 END FOR saluti
```

La struttura **REPeat** stampa "ciao" infinite volte (per fermarla usare la funzione **BREAK**), e la struttura **FOR** stampa "ciao" per 40 volte.

Osservare che il nome del ciclo **FOR** è anche una variabile, "saluti", il cui valore muta da 1 a 40 durante l'esecuzione del programma. Questa variabile è chiamata **variabile di ciclo** o **variabile di controllo** del ciclo.

La struttura di entrambi i cicli assume la forma di:

Istruzioni d'apertura
Contenuto
Istruzioni di chiusura

Si possono usare forme più sintetiche per il ciclo **FOR**. In questo modo il programma diventa:

PROGRAMMA 5

```
100 RESTORE 140: CLS
110 DIM via__Pippo$ (4, 3)
120 FOR numero= 1 to 4 : READ via__Pippo$ (numero): END FOR numero
130 FOR numero= 1 to 4 : PRINT! via__Pippo$ (numero): END FOR numero
140 DATA "VAL", "HAL", "MEL", "DEL"
```

I due punti ":" indicano la fine di un'istruzione e sostituiscono il simbolo di ENTER . Il programma può essere scritto in una forma ancora più sintetica. Per visualizzare il contenuto del vettore via__Pippo$, è sufficiente sostituire l'istruzione 130 con

```
130 PRINT! via__Pippo$ !
```

È stato introdotto il concetto di vettore alfabetico, in cui i soli numeri coinvolti sono i numeri d'ordine degli elementi di ogni variabile.

I vettori possono essere alfabetici o numerici. Vettori numerici compaiono negli esempi riportati nel seguito.

Fingere di lanciare due dadi per 400 volte. Calcolare la frequenza di ogni punteggio (ovviamente compreso fra 2 e 12).

PROGRAMMA 1

```
100 REMark DAD01
110 LET due=0: tre=0: quattro= 0: cinque=0: sei=0
120 LET sette=0: otto=0: nove=0: dieci=0: undici=0: dodici=0:
130 FOR lancio=1 TO 400
140       LET dado1=RND (1 TO 6)
150       LET dado2=RND (1 TO 6)
160       LET punti=dado1+dado2
170       IF punti=2 THEN LET due=due+1
180       IF punti=3 THEN LET tre=tre+1
190       IF punti=4 THEN LET quattro=quattro+1
200       IF punti=5 THEN LET cinque=cinque+1
210       IF punti=6 THEN LET sei=sei+1
220       IF punti=7 THEN LET sette=sette+1
230       IF punti=8 THEN LET otto=otto+1
240       IF punti=9 THEN LET nove=nove+1
250       IF punti=10 THEN LET dieci=dieci+1
260       IF punti=11 THEN LET undici=undici+1
270       IF punti=12 THEN LET dodici=dodici+1
280 END FOR lancio
290 PRINT ! due ! tre ! quattro ! cinque ! sei
300 PRINT ! sette ! otto ! nove ! dieci ! undici ! dodici
```

Nel programma precedente sono state introdotte undici variabili per memorizzare l'insieme dei punteggi. Disegnando un istogramma dei punteggi, si nota che il diagramma risultante è quasi triangolare.

Le frequenze più alte sono quelle dei punteggi sei, sette, otto, e le più basse sono due e dodici.

PROGRAMMA 2

```
100 REMark dado2
110 DIM freq(12)
120 FOR lancio=1 to 400
130    LET dado_1=RND (1 to 6)
140    LET dado_2=RND (1 to 6)
150    LET punti=dado_1 + dado_2
160    LET freq(punti)=freq(punti) + 1
170 END FOR lancio
180 FOR numero=2 to 12: PRINT ! freq (numero) !
```

Nel primo ciclo FOR, gli elementi del vettore sono le frequenze dei punteggi. Ciò significa che il valore dell'opportuno elemento del vettore è aumentato di una unità dopo ogni lancio.

Nel secondo ciclo FOR, quando il valore della variabile muta da 2 a 12, tutti i valori del vettore sono stampati.

In un'istruzione DIM, per un vettore numerico è sufficiente dichiarare il numero di elementi richiesti. Non ci sono problemi di limiti di lunghezza dei singoli elementi, come nei vettori alfabetici.

TEST SUL CAPITOLO 6

In questo test è possibile ottenere un punteggio massimo di 16 punti. Controllare il punteggio con le risposte a pag. 38.

1. Indicare le difficoltà che si incontrano quando i dati di un programma sono molto numerosi e non sono usati vettori. (2 punti)
2. Se in un vettore dieci variabili hanno lo stesso nome, come è possibile riconoscerle?
3. Cosa bisogna fare in un programma, prima di utilizzare un vettore?
4. Come è chiamato il numero che indica la singola variabile di un vettore?
5. Immaginare due situazioni di vita reale corrispondenti al concetto di vettore nella programmazione. (2 punti)
6. In una struttura **REPeat** il processo termina quando per determinate condizioni è eseguita un'istruzione **EXIT**. Quale è la causa che determina la fine del processo **FOR**?
7. Quale funzione ha il nome, in un ciclo **FOR**, oltre a quelle di indicare l'inizio e la fine del ciclo stesso?
8. Quali sono i 2 nomi con cui è chiamata la variabile utilizzata per identificare un ciclo **FOR**? (2 punti)
9. Indicare almeno una delle funzioni dei valori di una variabile di un ciclo.
10. Indicare quale delle seguenti caratteristiche hanno in comune una struttura **REPeat** e un ciclo **FOR**. (4 punti)

    a. Una parola chiave o un'istruzione di apertura
    b. Una parola chiave o un'istruzione di chiusura
    c. Un nome di ciclo
    d. Una variabile ciclo o una variabile di controllo.

RISPOSTE AL TEST SUL CAPITOLO 6

1. Non è facile trovare nomi di variabile per memorizzare dei dati. Diversi nomi devono essere scritti, in un'istruzione **LET** o **READ**, se non sono usati i vettori.
2. Un numero, chiamato indice, è parte del nome di una variabile di un vettore. Tutte le variabili di un vettore utilizzano uno stesso nome, ma ognuna ha un indice diverso.
3. È necessario dichiarare un vettore indicando le sue dimensioni in un'istruzione **DIM**, inserita all'inizio del programma, prima di utilizzare il vettore stesso.
4. Il numero che distingue un elemento del vettore è chiamato indice.
5. Le case di una stessa via appartengono tutte alla via, ma ognuna ha un numero diverso.

   I letti di un reparto in ospedale appartengono al reparto, ma ciascuno ha un diverso numero.

   Le celle di una prigione possono appartenere ad uno stesso blocco, ma hanno un diverso numero .
6. Un'iterazione **FOR** termina quando il processo corrispondente all'ultimo valore della variaqbile del ciclo è stato completato.
7. Il nome di un'iterazione **FOR** è anche il nome della variabile di controllo dell'iterazione.
8. Le due frasi sono: variabile di ciclo e variabile di controllo.
9. I valori di una variabile di controllo possono essere usati come indici per i nomi di variabili di un vettore.
10. Entrambe le strutture hanno:
    a. una parola chiave d'apertura

       REPeat, FOR

    b. un'istruzione di chiusura

       END REPeat nome, END FOR nome

    c. un nome

    Solo la struttura **FOR** ha:

    d. una variabile di ciclo o una variabile di controllo.

CONTROLLO DEL PUNTEGGIO

Questo test è più difficile rispetto ai precedenti.

Da 15 a 16. Eccellente. Proseguire la lettura.

Da 13 a 14. Ottimo. Riguardare i programmi per capire come funzionano.

Da 11 a 12. Buono. Rileggere alcune parti del capitolo.

Da 8 a 10. Sufficiente. Rileggere alcune parti del capitolo e ripetere il test.

Meno di 8. Rileggere con attenzione il capitolo e ripetere il test.

PROBLEMI SUL CAPITOLO 6

1. Inserire un numero tra 1 a 4 casualmente in cinque variabili di vettore utilizzando un ciclo **FOR**:

   carta(1), carta(2), carta(3), carta(4), carta(5).

   Non ha alcuna importanza se un numero viene ripetuto più volte. Utilizzare un secondo ciclo **FOR** per scoprire i valori delle cinque variabili "carta".
2. Immaginare che i quattro numeri rappresentino i quattro segni: cuori, quadri, picche e fiori. Quali righe di programma sarebbero necessarie per inserire queste parole al posto dei numeri?

3. Inserire cinque numeri casuali compresi tra 1 e 13 in un vettore di cinque variabili utilizzando un ciclo **FOR**:

   carta(1), carta(2), carta(3), carta(4), carta(5).

Utilizzare un secondo ciclo **FOR** per scoprire i valori delle cinque variabili carta.

CAPITOLO 7
SEMPLICI PROCEDURE

Risolvere problemi particolarmente complessi, utilizzando un computer, non è facile. È consigliabile scomporre un problema in piccole parti, chiamate compiti, e successivamente dividere questi compiti in altri più piccoli, fino a che ogni parte del problema principale possa essere risolta con facilità. Lo stesso fenomeno si verifica nelle relazioni umane. Ad esempio, il successo di un governo dipende da una buona distribuzione delle responsabilità. Il Primo Ministro affida alcuni compiti ai vari Ministri, che a loro volta li distribuiscono attraverso la pubblica amministrazione, fino a che i compiti diventano individuali.

Anche un buon programmatore deve lavorare in modo analogo. Un linguaggio moderno come il Superbasic offre maggiori possibilità rispetto alle versioni precedenti del Basic.

primo aggettivo	secondo aggettivo	nomi
Pieno	quinta generazione	sistemi
Sistematico	conoscitivo	macchine
Intelligente	compatibile	computer
Controllato	cibernetica	feedback
Automatizzato	facile	transputer
Sincronizzato	parallelo	micro-chip
Funzionale	istruttivo	capacità
Facoltativo	adattabile	programmazione
Positivo	modulare	package
Bilanciato	strutturato	database
Integrato	orietato alla logica	spread
Coordinato	file-orientato	word-processor
Sofisticato	standardizzato	obiettivo

ANALISI

Scrivere un programma che componga dieci frasi casuali utilizzando i vocaboli precedenti.

Le varie fasi del programma sono:

1. Memorizzare le parole in tre vettori stringa
2. Scegliere tre numeri casuali che sono assunti come indice delle variabili del vettore.
3. Stampare la frase
4. Ripetere i punti 2 e 3 dieci volte.

PROCEDURE

Si hanno tre vettori, dei quali i primi due contengono aggettivi o parole usate come aggettivi. Il terzo vettore contiene invece i sostantivi. Ogni vettore è composto da 13 elementi (parole) e la parola più lunga ha 19 caratteri.

vettore	scopo
agg_1$ (13,12)	**primo aggettivo**
agg_2$ (13,16)	**secondo aggettivo**
nome$ (13,19)	**nomi o sostantivi**

Sono utilizzate tre procedure per realizzare l'esercizio.

memoria memorizza i tre blocchi di tredici parole
caso genera tre numeri casuali tra 1 e 13
frase stampa una frase.

È un metodo semplice, ma efficace, in cui il lavoro principale è svolto dalle procedure.

```
100 REMark ******************
110 REMark *  frasi casuali  *
120 REMark ******************
130 DIM agg_1$ (13,12), agg_2$ (13,12), nome$ (13,19)
140 memoria
```

```
150 FOR volte 1 TO 10
160    caso
170    frase
180 END FOR volte
190 REMark **********************************
200 REMark *  Definizioni della procedura  *
210 REMark  **********************************
220 DEFine PROCedure memoria
230       REMark *  procedure per memorizzare le frasi  *
240       RESTORE 420
250       FOR parole = 1 TO 13
260           READ agg__1$, agg__2$, nome$
270       END FOR parole
280 END DEFine
290 DEFine PROCedure caso
300    REMark *  procedura per scegliere la frase  *
310    LET agg 1 = RND (1 TO 13)
320    LET agg 2 = RND (1 TO 13)
330    LET n = RND (1 TO 13)
340 END DEFine
350 DEFine PROCedure frase
360             REMark *** Procedura per stampare la frase ***
370             PRINT! agg__1$ (ag1) ! agg__2$ (ag2) ! nome$ (n)
380 END DEFine
390 REMark ************************
400 REMark *  Dati del programma  *
410 REMark ************************
420 DATA "Pieno", "quinta generazione", "sistemi"
430 DATA "Sistematico", "conoscitivo", "macchine"
440 DATA "Intelligente", "compatibile", "computer"
450 DATA "Controllato", "cibernetica", "feedback"
460 DATA "Automatizzato", "facile", "transputer"
470 DATA "Sincronizzato", "parallelo", "micro-chip"
480 DATA "Funzionale", "istruttivo", "capacità"
490 DATA "Facoltativo", "adattabile", "programmazione"
500 DATA "Positivo", "modulare", "package"
510 DATA "Bilanciato", "strutturato", "database"
520 DATA "Integrato", "orientato alla logica", "spread"
530 DATA "Coordinato", "file-orientato", "word-processor"
540 DATA "Sofisticato", "standardizzato", "obiettivo"
```

Automatico quinta generazione capacità
Funzionale istruttivo package
Pieno parallelo obiettivo
Positivo facile spreadsheet
Intelligente file-orientato capacità
Sincronizzato cibernetico transputer
Funzionale orientato alla logica micro-chip
Positivo parallelo feedback
Bilanciato istruttivo database
Controllato cibernetico obiettivo

COME PASSARE INFORMAZIONI ALLE PROCEDURE

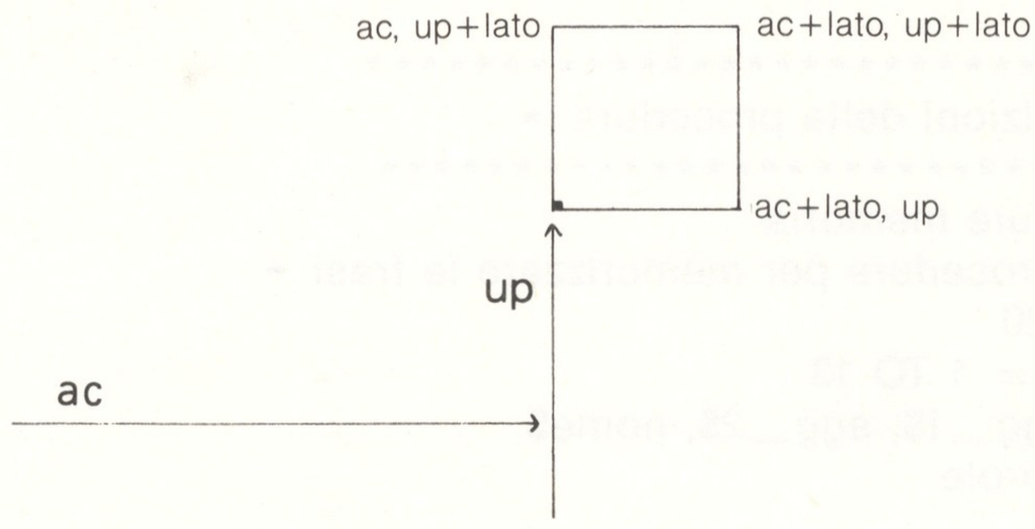
Supponiamo di voler disegnare quadrati di varie dimensioni, in diversi colori, in differenti punti dello schermo.

Se vogliamo, per realizzare l'esercizio, definire una procedura, quadrato, sono necessarie quattro informazioni:

lunghezza di un lato
colore (codice colore)
coordinate (orizzontali ac e verticali up)

La posizione del quadrato è determinata dall'attribuzione di due valori alle coordi-

nate, orizzontali e verticali, che indicano l'angolo inferiore sinistro del quadrato, come raffigura il disegno seguente:



Il colore del quadrato è facilmente attribuibile.

```
200 DEFine PROCedure quadrato (lato, ac, up)
210         LINE ac, up TO ac+lato, up
220         LINE TO ac+lato, up+lato
230         LINE TO ac, up+lato TO ac, up
240 END DEFine
```

Perché questa procedura possa essere eseguita, i valori di lato e delle coordinate ac e up devono essere fissati.

Questi valori sono stabiliti quando la procedura è eseguita. Ad esempio è possibile aggiungere al programma principale un'istruzione per ottenere un quadrato verde con un lato di 20.

```
100 PAPER 7 : CLS
110 INK 4
120 quadrato 20, 50, 50
```

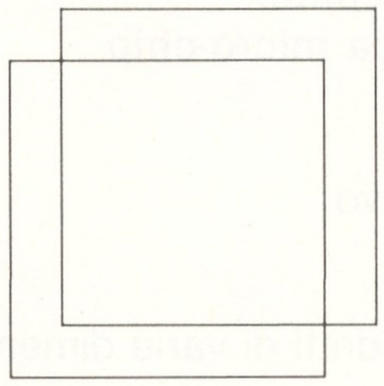
I numeri 20, 50, 50 sono chiamati parametri e sono i valori assegnati alle variabili presenti nella definizione della procedura:

```
quadrato 20, 50, 50

DEFine PROCedure quadrato (lato, ac, up)
```

I numeri 20, 50, 50 sono i parametri reali. In questo caso sono dei numeri, ma potrebbero essere anche variabili o espressioni. Le variabili "lato", "ac", "up" sono chiamati parametri formali.

Un programma più "complicato" genera uno schema casuale di due quadrati colorati. Ognuno dei due quadrati è ottenuto spostando di 1/5 entrambe le coordinate del vertice.



Supponendo che la procedura quadrato sia ancora definita nella riga 200, il seguente programma realizza quanto voluto.

```
100 REMark Quadrati
110 PAPER 7 : CLS
120 FOR paio = 1 TO 20
130    INK RND (5)
```

```
140    LET lato = RND (10 TO 20)
150    LET ac = RND (50) : up = RND (70)
160    quadrato, lato, ac, up
170    LET ac = ac+lato/5 : up = up+lato/5
180    quadrato, lato, ac, up
190 END FOR paio
```

I vantaggi di questa procedura sono:

1. La stessa serie di istruzioni è usata più volte nello stesso programma.
2. È possibile suddividere un compito in compiti più semplici e scrivere procedure per ognuno di questi, rendendo l'analisi e il disegno del programma più facili.
3. Le procedure possono essere provate separatamente. Questo favorisce la prova e il debugging.
4. Un programma è più leggibile se il nome, l'inizio e il termine di una procedura sono indicati chiaramente.

TEST SUL CAPITOLO 7

In questo test è possibile ottenere al massimo 14 punti.
Controllare il punteggio con le risposte a pag. 44.

1. Come è affrontato un problema molto complesso nelle relazioni umane?
2. Come può essere applicato lo stesso principio alla programmazione?
3. Quali sono le due più ovvie caratteristiche di una semplice procedura? (2 punti)
4. Quali sono gli effetti principali che si ottengono usando le procedure? (2 punti)
5. Quale vantaggio si ottiene utilizzando nomi di procedure in un programma prima che siano scritte le definizioni delle procedure?
6. Quale vantaggio si ottiene scrivendo una definizione di procedura prima di usarla in un programma?
7. Come può l'uso di una procedura aiutare un programmatore a scrivere programmmi più complessi?
8. Alcuni programmi utilizzano parte della memoria per definire le procedure, ma in quali casi le procedure non utilizzano lo spazio della memoria?
9. Indicare due modi per trasferire un informazione da un programma ad una procedura. (2 punti)
10. Cos'è un parametro reale?
11. Cos'è un parametro formale?

RISPOSTE AL TEST SUL CAPITOLO 7

1. Normalmente un lavoro complesso è diviso in compiti più facili.
2. Lo stesso principio può essere applicato nella programmazione frazionando un problema complesso e scrivendo una procedura per singolo compito.
3. Una procedura è composta da:

   una serie di istruzioni
   un nome opportuno (2 punti)

4. Con la chiamata di una procedura si ottiene:

   l'attivazione della procedura
   il ritorno del controllo del programma all'istruzione successiva alla chiamata. (2 punti)

5. I nomi di procedura possono essere usati in un programma prima di scrivere le procedure. Questo permette di avere una visione completa del lavoro.
6. È possibile controllare il buon funzionamento di una procedura, scrivendone la definizione prima di utilizzarla nel programma.
7. Un programmatore può dividere il lavoro in procedure per rendere più agevole il controllo.
8. L'uso di una procedura permette di risparmiare spazio di memoria, se è necessario ripetere più volte una medesima sequenza di istruzioni nel programma.
9. In un programma possono essere definite delle variabili con le istruzioni **LET** o **READ**. Le variabili sono accessibili alla procedura.

   Un secondo siostema è di usare dei parametri nella chiamata di procedura. Questi valori sono trasmessi alla variabile della definizione di procedura che li utilizza quando necessario. (2 punti)

10. Un parametro è il valore trasmesso da un'istruzione di chiamata di una procedura alla procedura stessa.
11. Un parametro formale è una variabile dichiarata in una definizione di procedura che riceve il valore trasmesso alla procedura del programma.

CONTROLLO DEL PUNTEGGIO

Da 12 a 14. Eccellente. Proseguire la lettura.

Da 10 a 11. Ottimo. Rivedere solo alcuni punti del capitolo.

Da 8 a 9. Buono. Rileggere alcune parti del capitolo.

Da 6 a 7. Sufficiente. Rileggere alcune parti del capitolo e ripetere il test.

Meno di 6. Rileggere con attenzione il capitolo e ripetere il test.

PROBLEMI SUL CAPITOLO 7

1. Scrivere una procedura per generare uno di questi quattro semi: cuori, quadri, fiori e picche. Richiamare la procedura cinque volte per ottenere cinque output casuali.

2. Scrivere un altro programma per risolvere il problema 1, utilizzando un numero, compreso tra 1 e 4, come parametro per determinare la parola di output. Se l'operazione è già stata eseguita, cercare di scrivere il programma senza parametri.

3. Scrivere una procedura che dia il valore di una carta, rappresentata come un numero compreso tra 2 e 10, o di una parola tra: "asso", "jack", "regina", "re".

4. Scrivere un programma che richiami cinque volte la procedura del punto 3 in modo da ottenere 5 valori casuali.

5. Scrivere il programma del problema 3 utilizzando un numero compreso tra 1 e 13 come parametro da inserire nella procedura. Se questo metodo è già stato usato, riscrivere il programma senza parametri.

6. Scrivere un programma, nella miglior forma possibile, utilizzando procedure, per creare quattro mani di cinque carte ciascuna. Non preoccuparsi delle carte doppie. Un programma, per essere definito ottimo, deve possedere tre qualità: leggibilità, sintesi ed efficienza.

CAPITOLO 8
DAL BASIC AL SUPERBASIC

INTRODUZIONE

Se si ha già familiarità con qualche versione di Basic, è possibile trascurare i primi sette capitoli.

Se, tuttavia, nell'affrontare i successivi si incontrassero delle difficoltà, può essere utile una veloce lettura delle pagine precedenti.

Passare dal Basic al Superbasic, non significa semplicemente disporre di un linguaggio più potente, ma vuol dire poter sfruttare un ambiente di programmazione considerevolmente più avanzato.

CONFRONTI ALFABETICI

In Superbasic, come negli altri Basic, è possibile eseguire semplici confronti.

Ad esempio:

```
LET pet1$ = "CANE"
LET pet2$ = "GATTO"
IF pet1$ < pet2$ THEN PRINT "Miao"
```

L'output è "Miao", perché nel confronto il simbolo "<" significa:

prima (più vicino ad A nell'alfabeto)

Il Superbasic può realizzare confronti più complessi. Ad esempio:

"cane" viene prima di "GATTO"

e

"ERD98L" viene prima di "ERD&$L"

Il sistema molto superficiale di confronto fra codici interni di caratteri, dà un risultato scorretto in entrambi i casi. Il programma seguente determina la prima, in ordine alfabetico, di due stringhe di caratteri.

```
100 INPUT var1$, var2$
110 IF var1$ < var2$ THEN PRINT var1$
120 IF var1$ = var2$ THEN PRINT "uguali"
130 IF var1$ > var2$ THEN PRINT var2$
```

INPUT		OUTPUT
VAR1$	VAR2$	
cane	gatto	cane
cane	GATTO	cane
ERD98L	ERD746L	ERD98L
ABC	abc	ABC

Il tema del confronto fra due stringhe nel Superbasic è trattato più in dettaglio nel capitolo "Regole di programmazione".

VARIABILI E NOMI DI VARIABILI

In Superbasic, il simbolo che caratterizza una variabile stringa è quello del dollaro, come ultimo carattere del nome della variabile. Ad esempio:

variabili numeriche:	variabili stringa:
conto, somma, totale	parola$, Via_Pippo$, giorno$

Il ruolo svolto dal nome della variabile in Superbasic è spiegato nel capitolo "Regole di programmazione". La lunghezza massima consentita per un nome di variabile è 255 caratteri.

È preferibile usare parole brevi, che abbiano un senso compiuto. Solo raramente è consigliabile utilizzare singoli caratteri come nomi di variabili.

Nomi di variabili come X, Z, P3, Q2 introducono un livello di astrazione che spesso è inutile.

VARIABILI INTERE

Il Superbasic dispone anche di variabili **intere**, il cui valore è un numero intero. Sono contraddistinte dal simbolo della percentuale:

```
conto%
numero%
```

Esistono quindi due tipi di variabili numeriche.

Il secondo tipo, quello che ammette l'uso di numeri interi e decimali, è chiamato **floating point** o **segno decimale mobile.**

```
LET prezzo = 9
LET costo = 7.31
LET conto% = 13
```

Scrivendo:

```
LET conto% = 5.43
```

il valore della variabile conto% è 5. Scrivendo invece:

```
LET conto% = 5.73
```

il valore di conto% è 6.

COERCIZIONE O CONVERSIONE

Il Superbasic segue il principio di correggere, fin dove è possibile, eventuali errori. Ad esempio, se la variabile stringa marco$ ha come valore

```
64
```

ed esiste l'istruzione

```
LET punti = marco$
```

la variabile punti assume il valore numerico di 64, mentre in altre versioni del Basic sarebbe stato segnalato l'errore:

```
Type mis-match
```

Solo se la stringa non può essere convertita è dato un messaggio di errore.

VARIABILI LOGICHE

Esiste nel Superbasic un altro tipo di variabile, chiamata variabile logica. L'istruzione Superbasic:

```
IF vento THEN vola_aereo
```

può essere anche scritta come:

```
IF v=1 THEN GOSUB 300
```

In questo caso v=1 è una condizione o un'espressione logica, che può essere vera o falsa. Se è vera, è eseguito il sottoprogramma che ha come punto d'inizio la riga 300. Un sottoprogramma è la parte di programma compresa tra la linea di controllo, il cui numero è quello che compare nell'istruzione GOSUB, e RETURN.

Un programmatore esperto userà l'istruzione:

```
IF v=1 THEN GOSUB 300 : REM vola_aereo
```

per rendere il programma più leggibile.

La variabile "vento" può avere solo due valori: vero (1), o falso (0), benché sia una variabile numerica a segno decimale mobile. Il valore 1, o comunque un qualsiasi altro valore diverso da zero, è considerato vero. Zero è considerato falso. Così una singola parola, come vento, è valutata come condizione di un'espressione logica.

L'altra parola, vola__aereo, è il nome di una procedura. Nel seguente programma sono utilizzate variabili logiche e procedure.

```
100 INPUT pioggia
110 IF pioggia THEN apri_ombrello
120 IF NOT pioggia THEN fai_gita
130 DEFine PROCedure apri_ombrello
140       PRINT "prendi l'ombrello"
150 END DEFine
160 DEFine PROCedure fai_gita
170       PRINT "vai al mare"
180 END DEFine
```

INPUT	OUTPUT
0	vai al mare
1	prendi l'ombrello
2	prendi l'ombrello
-2	prendi l'ombrello

Come è possibile verificare, solo il valore zero è considerato falso.

L'ISTRUZIONE LET

Nel Superbasic, l'istruzione **LET** è facoltativa, ma è utilizzata nel manuale per evitare di confondere le istruzioni di assegnazione. Il significato di = in:

```
LET conto = 3
```

e in:

```
IF conto = 3 THEN EXIT
```

è diverso, e l'istruzione **LET** aiuta a capire la differenza.

Quando si assegna un valore o un'espressione ad un nome di variabile, è sufficiente usare il simbolo "=". Tuttavia esplicitare l'istruzione **LET** consente una miglior comprensione del programma. Ad esempio:

```
100 LET primo = 0
110 LET secondo = 0
120 LET terzo = 0
```

Ma è possibile ottenere lo stesso risultato scrivendo:

```
100 LET primo = 0 : secondo = 0 : terzo = 0
```

Il simbolo ":" segna il termine di una istruzione.

IL VIDEO

Nel capitolo successivo sono illustrate le caratteristiche grafiche più avanzate. Ad esempio come disegnare cerchi, o altre figure più complicate, ma per il momento è preferibile capire cosa sono e come funzionano i pixel. Esistono sul QL 2 modalità grafiche:

bassa risoluzione 8 colori 256 pixel orizzontali e 256 pixel verticali	MODE 256 MODE 8
alta risoluzione 4 colori 512 pixel orizzontali e 256 verticali	MODE 512 MODE 4

In entrambe le modalità grafiche il numero di pixel è compreso tra:

0-511 in orizzontale
0-255 in verticale

Poiché il MODE 8 possiede in orizzontale la metà del numero di pixel rispetto al MODE 4, i pixel del MODE 8 hanno una larghezza doppia rispetto a quelli del MODE 4.

Ciò significa che la scelta delle coordinate è indipendente dal MODE utilizzato.

Se il QL è collegato con un televisore, non tutti i pixel sono visibili.

I COLORI

I colori disponibili sono:

MODE 256	CODICE	MODE 512
Nero	0	Nero
Blue	1	
Rosso	2	Rosso
Magenta	3	
Verde	4	Verde
Ciano	5	
Giallo	6	Bianco
Bianco	7	

Codice del colore

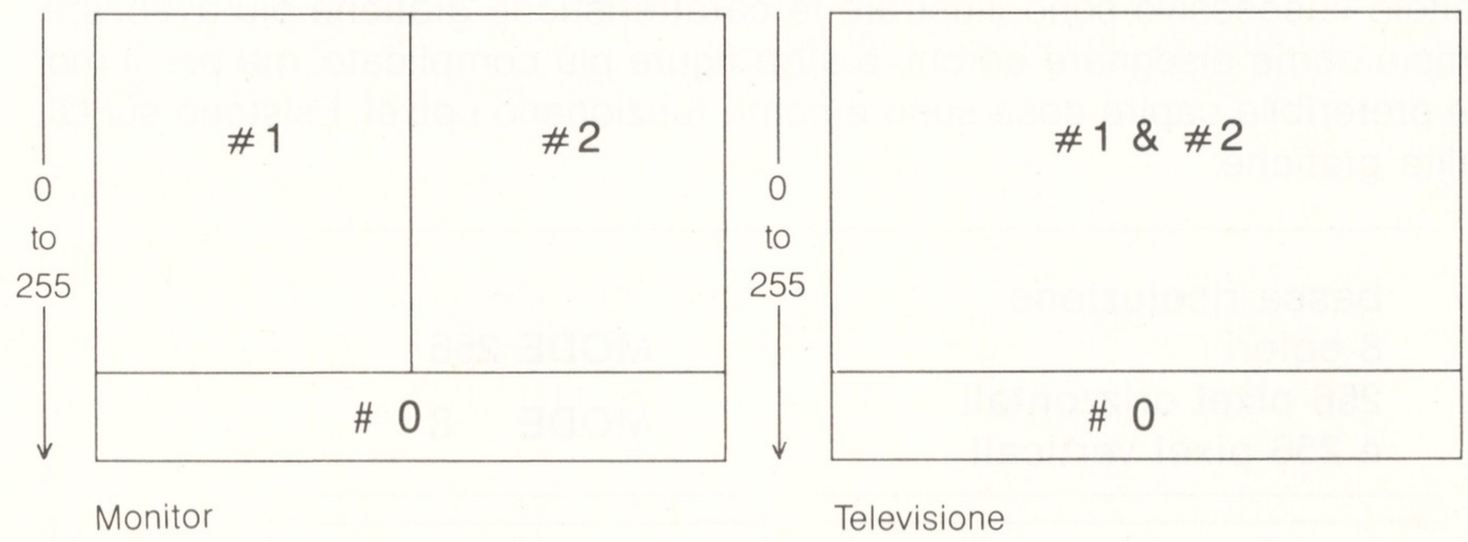
Nel Mode ad alta risoluzione grafica ogni colore è selezionato da uno dei due codici.

Di seguito sono riportate alcune parole chiave che occorre conoscere per gestire il video:

INK *colore*	colore della scrittura
BORDER *larghezza, colore*	disegna un bordo sulla parte superiore dello schermo
PAPER *colore*	colore dello sfondo.
BLOCK *larghezza, altezza, ac, up, colore*	colora il rettangolo definito dalle coordinate ac e up.

GESTIONE VIDEO

Se al momento dell'accensione il QL è collegato ad un monitor, il video appare diviso in tre aree, chiamate finestre, come nella figura seguente. Se il QL è collegato ad un televisore, lo schermo è interamente bianco.



Le finestre sono indicate con i simboli #0, #1, #2, in modo da poter ottenere determinati effetti in ognuna delle tre aree. Ad esempio, l'istruzione:

```
CLS
```

pulisce la finestra #1. Per pulire l'area sinistra bisogna scrivere:

```
CLS #2
```

Se si vuole un diverso colore di fondo, ad esempio il verde, scrivere:

```
PAPER 4 : CLS
```

oppure

```
PAPER #2, 4 : CLS #2
```

se si vuole cancellare il contenuto della finestra #2.

I numeri #0, #1, #2 sono chiamati **numeri di canale**. Non tutte le istruzioni elencate nella tabella sottostante possono avere un numero di canale. La terza colonna della tabella indica i canali di default, in altre parole quelli che sono scelti automaticamente dal sistema, se non sono stati specificati dall'operatore.

Parole chiave	Effetti	Default
AT	Posizione carattere	#1
BLOCK	Disegna un rettangolo	#1
BORDER	Disegna un bordo	#1
CLS	Pulisce lo schermo	#1
CSIZE	Grandezza carattere	#1
CURSOR	Posizione cursore	#1
FLASH	Inserisce/elimina lampeggio	#1
INK	Colore scrittura	#1
OVER	Effettua la stampa	#1
PAN	Sposta il video oriz.	#1
PAPER	Colore di fondo	#1
RECOL	Cambia il colore	#1
SCROLL	Sposta il video vert.	#1
STRIP	Colore per le stampe	#1
UNDER	Sottolinea	#1
WINDOW	Crea le finestre	#1
LIST	Lista il programma	#2
DIR	Lista il file	#1
PRINT	Stampa i caratteri	#1
INPUT	Riceve l'immissione della tastiera	#1

Le istruzioni appaiono nella finestra #0.

RETTANGOLI E LINEE

Il seguente programma disegna un rettangolo verde, nel Mode 256, su fondo rosso con bordo giallo e larghezza di un pixel. Il rettangolo ha il suo vertice sinistro superiore nel punto di coordinate 100, 100. La sua larghezza è di 80 unità orizzontali (40 pixel) e la sua altezza è di 20 unità verticali (20 pixel).

```
100 REMark Rettangolo
110 MODE 256
120 BORDER 1, 6
130 PAPER 2 : CLS
140 BLOCK 80, 20, 100, 100, 4
```

Nel Mode 256 i valori delle coordinate orizzontali devono essere compresi tra 0 e 511, anche se i pixel sono 256.

INPUT E OUTPUT

Il Superbasic utilizza le istruzioni **LET, INPUT, READ e DATA** per l'immissione dei dati.
È possibile stampare le informazioni secondo un formato particolare usando simboli chiamati separatori.

I separatori del Superbasic sono quattro:

, Virgola ha la funzione di tabulatore

; Punto e virgola permette di stampare il successivo elemento di seguito al precedente, senza passare alla riga successiva

/ Barra rovesciata provoca un ritorno a capo

! Punto esclamativo è utilizzato come spazio non all'inizio di riga, oppure effettua un ritorno a capo se ciò che deve essere stampato non può essere contenuto nella riga corrente.

TO Permette di stampare caratteri o valori in una posizione determinata del video.

CICLI O ITERAZIONI

Saranno chiariti altri aspetti di una struttura di programmazione fondamentale: i cicli iterativi, attraverso due esempi:

a) Simulare 6 lanci di un dado

```
100 FOR lancio = 1 TO 6
110       PRINT RND (1 TO 6)
120 NEXT lancio
```

b) Simulare il lancio di un dado fino ad ottenere il punteggio di 6

```
100 dado = RND (1 TO 6)
110 PRINT dado
120 IF dado <> 6 THEN GOTO 100
```

Entrambi i programmi sono corretti, ma le versioni seguenti sono preferibili.

a)
```
100 FOR lancio = 1 TO 6
110       PRINT RND (1 TO 6)
120 END FOR lancio
```

b)
```
100 REPeat lanci
110       dado = RND (1 TO 6)
120       PRINT dado
130       IF dado = 6 THEN EXIT lanci
140 END REPeat lanci
```

Lo schema della struttura **REPeat** è il seguente:

REPeat nome variabile: istruzioni
END REPeat nome variabile

È possibile inserire l'istruzione **EXIT** in un punto qualunque della struttura, ma deve essere seguita dal nome della variabile di identificazione del ciclo per indicare al Superbasic da quale struttura uscire. Ad esempio:

EXIT lanci

trasferisce il controllo all'istruzione successiva a:

END REPeat lanci

Tutto ciò può sembrare inutilmente complicato. Tuttavia la struttura **REPeat** è molto potente e consente di risolvere con semplici istruzioni problemi complessi.

È necessario assegnare un nome ad ogni interazione **REPeat**, che si vuole introdurre in un programma. In questo modo è più facile controllare il termine del ciclo iterativo, semplicemente verificando che esiste un'istruzione **END REPeat** seguita dal nome assegnato nella corrispondente istruzione **REPeat**. Analoga situazione accade con il ciclo iterativo **FOR... END FOR**.

Esistono altre proprietà tipiche di un ciclo iterativo, che i programmatori esperti conoscono.

L'incremento del valore della variabile di controllo di un ciclo **FOR** è normalmente di una unità. È possibile modificare a piacere l'incremento, usando l'istruzione **STEP**, come negli esempi seguenti:

```
100 FOR pari = 2 TO 10 STEP 2
110    PRINT! pari!
120 END FOR pari
```

L'output è: **2 4 6 8 10**

```
100 FOR indietro = 9 TO 1 STEP — 1
110    PRINT! indietro!
120 END FOR indietro
```

L'output è: **9 8 7 6 5 4 3 2 1**

È possibile anche "nidificare" due cicli iterativi: inserirli, cioè, uno all'interno dell'altro. Il seguente programma, per esempio, stampa 4 righe di 10 croci (×) ciascuna

```
100 REMark croci
110 FOR riga = 1 TO 4
120    PRINT "riga numero"! riga
130    FOR croci = 1 TO 10
140       PRINT! "×" !
150    END FOR croci
160    PRINT
170 END FOR riga
```

L'output è:

```
Riga numero 1
××××××××××
Riga numero 2
××××××××××
Riga numero 3
××××××××××
Riga numero 4
××××××××××
```

Un vantaggio del Superbasic è la possibilità di nidificare non solo 2 cicli **FOR**, ad esempio, ma 2 o più cicli iterativi di tipo diverso, come un **REPeat** all'interno di un **FOR** o viceversa

Il seguente programma simula e ripete il lancio di 2 dadi, finché non è ottenuto il punteggio 7 in ognuna delle 4 righe.

```
100 REMark lanci di dadi
110 FOR riga = 1 TO 4
120    PRINT "riga numero"! riga
130    REPeat lanci
140       LET dado1 = RND (1 TO 6)
150       LET dado2 = RND (1 TO 6)
160       LET punti = dado1+dado2
170       PRINT punti!
180       IF punti = 7 THEN EXIT lanci
190    END REPeat lanci
200 END FOR riga
```

Un esempio di output è il seguente:

```
Riga numero 1
8 11 6 3 7

Riga numero 2
4 6 2 9 4 5 12 7

Riga numero 3
7

Riga numero 4
6 2 4 9 9 7
```

Un'altra proprietà importante di un ciclo iterativo **FOR** è quella di permettere estrema flessibilità nella scelta dei valori della variabile di controllo. Il programma seguente stampa tutti i numeri divisibili per 2 (eccetto il 2) da 1 a 20

```
100 REMark numeri divisibili per 2
110 FOR numero = 4, 6, 8 TO 10, 12, 14 TO 16, 18, 20
120    PRINT! numero!
130 END FOR numero
```

DECISIONI E SCELTE

In uno dei programmi precedenti è stata effettuata una decisione con l'istruzione

```
IF dado = 6 THEN EXIT lanci
```

Questo tipo di istruzione è disponibile in tutti i Basic, ma il Superbasic offre la possibilità di operare delle scelte quando le alternative sono più di due .

I seguenti programmi si spiegano da soli.

a)
```
100 REMark IF... END IF
110 LET sole = RND (0 TO 1)
120 IF sole THEN
130    PRINT "metti gli occhiali da sole"
140    PRINT "vai a passeggiare"
150 END IF
```

b)
```
100 REMark IF... ELSE... END IF
110 LET sole = RND (0 TO 1)
120 IF sole THEN
130    PRINT "metti gli occhiali da sole"
140    PRINT "vai a passeggiare"
150 ELSE
160    PRINT "metti l'impermeabile"
170    PRINT "vai al cinema"
180 END IF
```

Le strutture decisionali del tipo IF... ELSE... END IF hanno la stessa forma dei cicli iterativi. È possibile nidificarli e combinarli all'interno dei cicli **FOR... END FOR** o **REPeat... END REPeat**.

SUBROUTINES E PROCEDURE

La maggior parte dei Basic dispongono di istruzioni come **GOSUB**, che possono essere usate per richiamare blocchi di programma chiamati subroutines. Per varie ragioni l'istruzione **GOSUB** può dare luogo ad inconvenienti, soprattutto a programmatori non esperti.

Il Superbasic permette di sfruttare altri tipi di strutture chiamate **PROCEDURE**, che rendono inutile utilizzare istruzioni come **GOSUB**.

Entrambi i seguenti programmi disegnano un quadrato verde su sfondo rosso con il lato lungo 50 pixel partendo dal punto di coordinate (in pixel) 200, 100.

a) usando **GOSUB**

```
100 LET colore = 4 : sfondo = 2
110 LET x = 200: y = 100: lato = 50
120 GOSUB 800
130 PRINT "FINE"
140 STOP
800 REMark disegno quadrato
810 BLOCK lato, lato, x, y, colore
820 RETURN
```

b) usando una procedura con parametri

```
100 quadrato 4, 50, 200, 100, 2
110 PRINT "FINE"
120 DEFine PROCedure quadrato (colore, lato, x, y, sfondo)
130    PAPER sfondo: CLS
140    BLOCK lato, lato, x, y, colore
150 END DEFine
```

Nel primo programma i valori di *colore, x, y, lato* sono fissati da un'istruzione **LET**, prima che l'istruzione **GOSUB** richiami le linee 800, 810. Il controllo del programma è rimandato dall'istruzione **RETURN** alla riga successiva al **GOSUB**.

Nel secondo programma i valori sono passati, dalla riga 100, come parametri alla procedura *quadrato*.

Nella forma più semplice, le procedure non hanno parametri. Isolano semplicemente una parte del programma, ma anche in questo caso il loro uso è preferibile a **GOSUB**.

La potenza e la semplicità delle procedure sono tanto più evidenti quanto più grande è il programma. In questo caso utilizzare le procedure rende molto più semplice anche il controllo e la correzione di eventuali errori.

ESEMPIO

Il seguente programma accetta in input un numero compreso fra 1 e 3999 e lo converte nell'equivalente numero romano, non generando, tuttavia, la forma più elegante (ad esempio il numero 4 sarà scritto come IIII e non come IV).

```
100 REMark numeri romani
110 LET numero = 3289
120 FOR tipo = 1 TO 7
130    READ lettera$, valore
140    REPeat output
150       IF numero < valore: EXIT output
160       PRINT lettera$;
170       numero = numero-valore
180    END REPeat output
190 END FOR tipo
200 DATA "M", 1000, "D", 500, "C", 100
210 DATA "L", 50, "X", 10, "V", 5, "I", 1
```

CONCLUSIONI

Il Superbasic è un linguaggio complementare strutturato che consente di modificare in modo semplice blocchi di programma.

Tutte le strutture devono essere identificate e deve essere loro assegnato un nome.

Spiegazioni più dettagliate si trovano nelle parti "Parole chiave" e "Concetti".

CAPITOLO 9
LE VARIABILI

È chiaro che un programma (sequenza di istruzioni), ha bisogno di ricevere dei dati per ottenere un determinato risultato. È altrettanto chiaro che i dati devono essere memorizzati internamente al computer e disponibili al programma. Le variabili hanno questa funzione.

Ci sono quattro diversi tipi di variabili chiamate, rispettivamente: a segno decimale mobile (floating point), intere, stringhe e logiche.

NOMI DI VARIABILI

I nomi di variabile sono soggetti a delle semplici regole:

1. I caratteri consentiti in un nome di variabile sono le lettere maiuscole o minuscole, i numeri e la sottolineatura. Il primo carattere deve essere una lettera.
2. Il nome di una variabile può avere una lunghezza massima di 255 caratteri. In pratica non esiste alcun limite.
3. Il nome di una variabile non può essere una parola chiave del Superbasic.
4. Il nome di una variabile intera deve terminare con il simbolo %.
5. Il nome di una variabile a stringa deve terminare con il simbolo $.
6. Nessun altro tipo di variabile può usare i simboli % e $.
7. Il nome di una variabile deve essere scelto in modo da avere un significato preciso per il lettore, anche se ciò non ha alcuna importanza per il Superbasic.

VARIABILI A SEGNO DECIMALE MOBILE

Esempi di variabile a segno decimale mobile sono:

```
100 LET giorni = 24
110 LET vendite = 3649.84
120 LET vendite_per_giorno = vendite/giorni
130 PRINT vendite_per_giorno
```

Una variabile a segno decimale mobile può assumere tutti i valori compresi nell'intervallo:

da -10^{+615} a $+10^{+615}$ con 8 cifre significative

Si supponga che nel precedente programma il valore della variabile "vendite" sia 0,03. La riga 110 diventa:

```
110 LET vendite = 0.03
```

Il programma cambia in questo modo:

```
110 LET vendite = 3E-2
```

Per capire l'equivalenza è sufficiente pensare al valore 3 come 3.0 e spostare il punto decimale di -2, posizioni, cioè due spazi verso sinistra:

3E-2 è uguale a 0.03

Dopo l'esecuzione del programma la media giornaliera di vendite è:

1.25E-3 o 0.00125

I numeri seguiti da una E sono rappresentati in forma esponenziale.

(mantissa) E (esponente) = (mantissa) X 10 elevato alla potenza (esponente)

VARIABILI INTERE

Le variabili intere possono assumere tutti i valori compresi nell'intervallo da -32768 a 32767. I seguenti sono esempi di variabili intere. (Devono terminare con il simbolo %)

```
LET conto% = 10
LET numero_3% = 3
```

Ricordiamo che il simbolo % non ha alcun rapporto (in questo caso) con l'idea di percentuale.

FUNZIONI NUMERICHE

La funzione **INT** calcola il valore intero di un numero o espressione numerica. Ad esempio l'output di:

```
PRINT INT (5.6)
```

è 5.

I seguenti esempi illustrano il normale uso della funzione **INT**.

```
100 REMark rotondo
110 INPUT decimale
120 PRINT INT (decimale + 0.5)
```

Nell'esempio si inserisce un valore decimale e il risultato è arrotondato. Così 4.7 diventa 5 ma 4.3 diventa 4.

Le funzioni trigonometriche sono trattate più avanti, mentre le funzioni numeriche più comuni sono illustrate nella seguente tabella.

Funzione	Effetti	Esempi	Valori
ABS	Valore assoluto	ABS(7) ABS (-4.3)	7 4.3
INT	Parte intera di un numero decimale	INT(2.4) INT(0.4) INT(-2.7)	2 0 -3
SQRT	Radice quadrata	SQRT(2) SQRT(16) SQRT(2.6)	1.414214 4 1.612452

OPERAZIONI MATEMATICHE

Il Superbasic consente di eseguire le operazioni matematiche fondamentali. Alcune volte l'operazione è indicata da simboli familiari come "+" o "*", altre è indicata da una parola chiave come **DIV o MOD.** Le operazioni numeriche hanno un ordine di precedenza. Ad esempio il risultato di:

```
PRINT 7 + 3*2
```

è 13, perché la moltiplicazione è eseguita prima dell'addizione.

```
PRINT (7 + 3)*2
```

dà come risultato 20, perché le parentesi possono sovvertire l'ordine di priorità.

Le operazioni, che sono di seguito descritte, hanno il seguente ordine di precedenza:

alto - **moltiplicazione e divisione (comprese DIV e MOD), esponente, potenza**
basso - **addizione e sottrazione**

I simboli "+" e "—" sono anche usati semplicemente per definire un numero positivo o negativo.

Se due simboli hanno lo stesso ordine di priorità, le operazioni sono eseguite in ordine da sinistra a destra. Ad esempio in:

```
PRINT 7-2+5
```

la sottrazione è eseguita prima dell'addizione.

Operazione	Simbolo	Esempio	Risultato	Note
Addizione	+	7+6.6	13.6	
Sottrazione	—	7-6.6	0.4	
Moltiplicazione	*	3*2.1	6.3	
		2.1*(-3)	-6.3	
Divisione	/	7/2	3.5	Non dividere per 0
		-17/5	-3.4	
Potenza	^	4^1.5	8	
Divisione intera	DIV	-8 DIV 2	-4	Solo interi
		7 DIV 2	3	Non dividere per 0
Modulo	MOD	13 MOD 5	3	
		21 MOD 7	0	
		-17 MOD 8	7	

L'istruzione **MOD** visualizza il resto di una divisione. Qualsiasi tentativo di dividere per 0 provoca un errore e blocca l'esecuzione del programma.

ESPRESSIONI NUMERICHE

Un'espressione numerica può essere formata da costanti numeriche o variabili. Le espressioni numeriche, in Superbasic, sono simili alle espressioni matematiche, ma devono essere scritte in sequenza.

$$\frac{5 + 3}{6 - 4}$$

diventa in Superbasic (e in Basic):

```
(5 + 3)/(6 - 4)
```

ESEMPIO 1

Utilizzando un'espressione algebrica, è possibile risolvere, ad esempio, un'equazione di secondo grado:

$$ax^2 + bx + c = 0$$

una soluzione algebrica è data dalla formula:

$$x = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

Ad esempio, risolvere l'equazione:

$$2x^2 - 3x + 1 = 0$$

ESEMPIO 2

Il seguente programma ottiene il risultato cercato.

```
100 READ a, b, c
110 PRINT "Radice è" ! (-b+SQRT (b ^ 2 - 4*a*c))/(2*a)
120 DATA 2, -3, 1
```

In un problema in cui bisogna simulare la distribuzione delle carte, conviene indicare le carte con i numeri da 1 a 52 nel seguente modo:

```
 1 a 13     Asso, due... re di cuori
14 a 26     Asso, due... re di quadri
27 a 39     Asso, due... re di picche
40 a 52     Asso, due... re di fiori
```

Una carta particolare può essere identificata dal programma:

```
100 REM identificazione carta
110 LET carta = 23
120 LET seme = (carta-1) DIV 13
130 LET valore = carta MOD 13
140 IF valore = O THEN LET valore = 13
150 IF valore = 1 THEN PRINT "Asso di";
160 IF valore > = 2 AND valore < = 10 THEN PRINT valore ! "di";
170 IF valore = 11 THEN PRINT "Fante di";
180 IF valore = 12 THEN PRINT "regina di";
190 IF valore = 13 THEN PRINT "re di";
200 IF seme = 0 THEN PRINT "cuori";
210 IF seme = 1 THEN PRINT "quadri";
220 IF seme = 2 THEN PRINT "picche";
230 IF seme = 3 THEN PRINT "fiori";
```

Nella riga 160 è esposta una nuova idea. Il significato è chiaro: il numero (valore) è stampato solo se le due condizioni logiche sono vere, e cioè:

Se valore è maggiore o uguale a 2
e valore è minore o uguale a 10.

Notare l'uso del simbolo "!" nell'istruzione PRINT per inserire uno spazio e del simbolo ";" per proseguire la stampa sulla stessa riga.

VARIABILI LOGICHE

Il Superbasic non utilizza esclusivamente variabili di tipo logico, ma ottiene gli stessi risultati anche con variabili numeriche.

Infatti:

```
100 REMark variabile logica
110 LET affamato = 1
120 IF affamato THEN PRINT "prendi un dolce"
```

Il programma interpreta il valore 1, della variabile *affamato*, come vero. Il risultato è:

```
prendi un dolce
```

Se la riga 110 fosse:

```
LET affamato = 0
```

non ci sarebbe alcun output, perché il programma interpreta il valore 0 come falso e tutti gli altri valori come veri. È possibile anche scrivere:

```
100 REMark variabile logica
110 LET vero = 1 : falso = 0
120 LET affamato = vero
130 IF affamato THEN PRINT "prendi un dolce"
```

per "nascondere" il contenuto numerico della variabile *affamato*

PROBLEMI SUL CAPITOLO 9

1. Un ricco petroliere gioca d'azzardo lanciando una moneta. Ogni volta che esce croce guadagna 1, se esce testa lancia ancora la moneta e la possibile vincita è raddoppiata.

   Tutto è ripetuto finché le vincite non sono:

   LANCI 1 2 3 4 5 6 7
   VINCITE 1 2 4 8 16 32 64

   Simulare il gioco per stabilire quale dovrebbe essere la posta iniziale delle seguenti partite:

   a) il giocatore ha un massimo di 7 lanci per partita
   b) non vi è alcun limite ai lanci.

2. Bill e Ben fanno un gioco. Ad un segnale ciascuno divide il proprio denaro in due parti e passa una parte all'altro. Ciascuno divide poi la nuova somma e passa la metà all'altro. Stabilire l'evoluzione del gioco sapendo che Bill aveva inizialmente 16 lire e Ben 64 lire.
3. Cosa succede se il gioco cambia in modo che entrambi consegnano una somma uguale alla metà dell'altro?
4. Scrivere un programma che generi parole casuali di tre lettere scelte fra E, P, A, R e stamparle finchè non è composta la parola PER.
5. Modificare il precedente programma in modo che termini quando è composta una qualsiasi parola esistente di tre lettere.

CAPITOLO 10
LOGICA
PERCHÉ LA LOGICA

Dopo la lettura dei precedenti capitoli è possibile affermare che le ripetizioni (o iterazioni), le scelte e la suddivisione dei programmi in sottoprogrammi, sono i concetti più importanti nell'affrontare l'analisi di un problema, e nel disegnare lo schema di un programma. Due di questi concetti, le ripetizioni e le scelte, utilizzano espressioni logiche, come nelle seguenti istruzioni:

```
IF punti = 7 THEN EXIT lanci
IF seme = 3 THEN PRINT fiori
```

La prima utilizza l'istruzione **EXIT**, parte di una struttura **REPeat**. La seconda esprime semplicemente la scelta di fare o non fare qualcosa. Un'espressione matematica può avere da uno a milioni di possibili valori. Allo stesso modo, una stringa può contenere caratteri in milioni di combinazioni diverse. Sembra strano che un'espressione logica, alla quale è stata attribuita grande importanza abbia solo due possibili valori: vero o falso.

È necessaria maggiore attenzione alle espressioni che contengono le parole **OR, AND** e **NOT**. Tuttavia **OR, AND, NOT** sono operatori fondamentali nella programmazione.

AND

La parola **AND**, in Superbasic, ha lo stesso significato della congiunzione nel linguaggio ordinario.

Osservare il programma:

```
100 REMark AND
110 PRINT "Inserire due valori"/ "1 per vero, 0 per falso"
120 INPUT piove, buco_nel_tetto
130 IF piove AND buco_nel_tetto THEN PRINT "ti bagni"
```

Come nella realtà, "ti bagni" solo se *piove* e se c'è un *buco_nel_tetto*. Se una (o entrambe) di queste condizioni logiche:

piove
buco_nel_tetto

è falsa, l'intera espressione logica:

piove **AND** *buco_nel_tetto*

è falsa. Sono necessari due valori di verità perché l'espressione sia vera, come è riportato nella seguente tabella:

piove	buco_nel_tetto	piove AND buco_nel_tetto	effetti
FALSO	FALSO	FALSO	NESSUNO
FALSO	VERO	FALSO	NESSUNO
VERO	FALSO	FALSO	NESSUNO
VERO	VERO	VERO	BAGNATO

OR

Nella vita di tutti i giorni la parola "o" assume due significati. È possibile spiegare l'uso di **OR** pensando ad un allenatore di una squadra di baseball, che sceglie i giocatori. Egli chiede: "vuoi battere o lanciare?" È contento se un giocatore fa bene una delle due azioni, ma è ancora più felice se qualcuno le fa bene entrambe. Tutto questo avviene anche nella programmazione: un'espressione composta usando **OR** è vera se una o entrambe le condizioni sono vere.

Eseguire il seguente programma:

```
100 REMark OR test
110 PRINT "inserire due valori"/ "1 per vero o 0 per falso"
120 INPUT "vuoi_battere?", battitore
130 INPUT "vuoi_lanciare?", lanciatore
140 IF battitore OR lanciatore THEN PRINT "nella squadra"
```

Gli effetti delle differenti combinazioni sono riportati nella seguente tabella.

battitore	lanciatore	battitore OR lanciatore	effetto
FALSO	FALSO	FALSO	no in squadra
FALSO	VERO	VERO	in squadra
VERO	FALSO	VERO	in squadra
VERO	VERO	VERO	in squadra

NOT

La parola **NOT** ha un significato più ovvio.

NOT vero è uguale a falso
NOT falso è uguale a vero

È necessario prestare attenzione. Si supponga, per esempio, di avere un triangolo rosso e di dire:

NOT *rosso* **AND** *quadrato*

L'espressione è ambigua. Infatti se con questa affermazione si intende:

(**NOT** *rosso*) **AND** *quadrato*

l'espressione è falsa.

Se invece si intende:

NOT (*rosso* **AND** *quadrato*)

l'espressione è vera.

Una regola della programmazione consente di capire questo meccanismo. La regola è che **NOT** ha la precedenza rispetto a **AND**. Così l'interpretazione di:

(**NOT** *rosso*) **AND** *quadrato*

è corretta, ed è uguale a:

NOT (*rosso* **AND** *quadrato*)

PARENTESI

Per ottenere diverse interpretazioni devono essere usate le parentesi. È possibile non utilizzare le parentesi se si rispettano le seguenti regole (attribuire a Augustus De Morgan).

NOT (a AND b) è uguale a: **NOT a OR NOT b**
NOT (a OR b) è uguale a: **NOT a AND NOT b**

Ad esempio:

NOT (*alto* e *magro*) è uguale a
NOT *alto* **OR NOT** *magro*

NOT (*affamato* **OR** *assetato*) è uguale a
NOT *affamato* **AND NOT** *assetato*

Verificare scrivendo:

```
100 REMark NOT e parentesi
110 PRINT "inserire due valori"/ "1 per VERO o 0 per FALSO"
120 INPUT "alto"; alto
130 INPUT "magro"; magro
140 IF NOT (alto AND magro) THEN PRINT "PRIMO"
150 IF NOT alto OR NOT magro THEN PRINT "SECONDO"
```

Le due espressioni logiche sono equivalenti.

XOR O OR ESCLUSIVO

Si supponga che un giocatore di golf voglia assumere un assistente, che possa gestire il negozio e dare lezioni di golf. Tuttavia, se chi si presenta per ricevere il lavoro, è in grado di fare entrambe le cose, non ottiene il posto, perché il maestro può aver paura di essere superato dall'allievo. Sarà accettato un buon giocatore che non sappia gestire il negozio o un giocatore mediocre che sappia amministrare il negozio. Questa è una tipica situazione **XOR**: sono accettabili uno o l'altro, ma non entrambi.

```
100 REMark XOR test
110 PRINT "inserire 1 per si o 0 per no."
120 INPUT "sai gestire il negozio?", negozio
130 INPUT "sai insegnare golf?", golf
140 IF negozio XOR golf THEN PRINT "adatto"
```

Le sole combinazioni che generano la risposta "adatto" sono (0 e 1) o (1 e 0). Le regole di **XOR** sono riportate di seguito.

capace di gestire il negozio	capace di insegnare	negozio XOR insegnare	effetti
FALSO	FALSO	FALSO	no lavoro
FALSO	VERO	VERO	si lavoro
VERO	FALSO	VERO	si lavoro
VERO	VERO	FALSO	no lavoro

PRECEDENZE

L'ordine di precedenza dell'esecuzione degli operatori logici è:

NOT
AND
OR, XOR

Ad esempio l'espressione:

ricco **OR** *alto* **AND** *magro*

è uguale a:

ricco **OR** (*alto* **AND** *magro*)

L'operazione **AND** è eseguita per prima. Per verificare che le due espressioni logiche danno un risultato analogo eseguire il programma:

```
100 REMark Precedenza
110 PRINT "inserire tre valori"/ "1 per si e 0 per no"
120 INPUT ricco, alto, magro
130 IF ricco OR alto AND magro THEN PRINT "SI"
140 IF ricco OR (alto AND magro) THEN PRINT "CIAO"
```

Qualsiasi combinazione di valori, composta da tre zero o da tre uno, darà come ouput:

```
SI
CIAO
```

Provare inserendo queste serie di numeri:

```
000 001 010 011 100 101 110 111
```

PROBLEMI SUL CAPITOLO 10

1. Inserire dieci numeri in un'istruzione **DATA**. Con l'istruzione **READ** stampare tutti i numeri maggiori di 20.

2. Scrivere tutti i numeri da uno a cento e stampare solo i quadrati perfetti o quelli divisibili per 7.

3. I giocattoli sono indicati come utili (U), o inutili (I), costosi (C) o economici (E), e anche per ragazze (R) o per ragazzi (B) o per tutti (T). Un terno di lettere indicherà le qualità di ciascun giocattolo. Inserire 5 serie di tre lettere di un'istruzione **DATA** e stampare i giochi che sono utili e adatti solo alle bambine.

4. Modificare il programma 3 stampando solo i giocattoli costosi e inutili.

5. Modificare il programma 3 stampando solo i giocattoli che sono utili, economici e adatti a tutti.

CAPITOLO 11
STRINGHE

Le variabili alfabetiche sono già state utilizzate per memorizzare stringhe di caratteri. Le regole per la manipolazione delle stringhe non hanno alcun valore per le variabili numeriche. Il Superbasic offre una serie di facilitazioni per la manipolazione delle stringhe di caratteri.

ASSEGNAZIONE DELLE STRINGHE

Non appena una stringa è stata definita da un programma, le è assegnata una parte di memoria. Ad esempio le righe:

```
100 LET parole$ = "lunghe"
110 LET parole$ = "più lunghe"
120 PRINT parole$
```

visualizzano le parole "più lunghe". La prima definizione richiede uno spazio per sei lettere, ma è sovrapposta dalla seconda che richiede spazio per undici caratteri.

È comunque possibile dimensionare una stringa, nel qual caso è definita la lunghezza massima, e la variabile si trasforma in vettore.

UNIONE DI STRINGHE

Si supponga di essere un insegnante e di voler memorizzare tre voti per ogni studente nelle materie: letteratura, storia e geografia. I voti sono contenuti nelle variabili:

lett$	62

st$	56

geog$	71

È possibile trasformare i valori di queste tre stringhe in una stringa di sei caratteri chiamata voto$. È sufficiente scrivere:

```
LET voto$ = lett$ & st$ & geog$
```

per creare la variabile:

voto$	625671

Notare che il simbolo "**&**" nel Superbasic è utilizzato per unire più stringhe. In altre versioni dei Basic è usato il simbolo "+" con lo stesso significato.

Tutte le variabili stringa sono inizialmente vuote e hanno lunghezza zero. Se si cerca di trascrivere una stringa in un'altra stringa con una lunghezza insufficiente, la trascrizione non è riconosciuta dal Superbasic.

Prima di effettuare una simile operazione, è opportuno assicurarsi che la stringa di destinazione sia abbastanza ampia da contenere la prima stringa, compresi gli eventuali spazi.

```
100 LET soggetto$ = "Calcolo Inglese matematica"
110 LET studente$ = ""
120 LET studente$ (9 TO 15) = soggetto$ (9 TO 15)
```

L'istruzione 120 significa che i caratteri dal nono fino al quindicesimo della stringa *soggetto$* sono riportati nella stringa *studente$*, a partire dalla posizione 9 fino alla 15

COERCIZIONE O CONVERSIONE

Supponiamo di avere una variabile chiamata *voto$*, che contiene un insieme di voti di esami. La parte di stringa contenente il voto di storia può essere estratta e sostituita da un altro punteggio. Il seguente programma estrae il voto di storia:

```
100 LET voto$ = "625671"
110 LET st$ = voto$ (3 TO 4)
```

Ora il valore "56" della variabile *st$*, è una stringa di caratteri e non un valore numerico. Se l'insegnante volesse aumentare il voto, moltiplicandolo per esempio per 1.125, il valore di *st$* deve essere convertito in numerico. Il Superbasic attua automaticamente questa conversione, scrivendo semplicemente:

```
120 LET numero = 1.125 * st$
```

La riga 120 converte la stringa "56" nel numero 56, lo moltiplica per 1.125 ottenendo come risultato 63.

Ora è possibile sostituire il voto precedente con quello nuovo, cioè 63, ma prima di essere inserito nella stringa originale, deve essere convertito nei caratteri alfabetici "63". Ancora una volta il Superbasic attua la conversione in modo automatico.

```
130 LET voto$ (3 TO 4) = numero
140 PRINT voto$
```

Il risultato dell'intero programma è:

```
626371
```

che indica come il voto di storia sia cambiato.

Non è corretto tentare di assegnare un valore alfabetico che non può essere convertito in numerico ad una variabile numerica. È sbagliato scrivere:

```
LET numero = "LEONE"
```

In questo caso è visualizzato un messaggio di errore, ma se si scrive:

```
LET numero = "65"
```

il programma assegna il valore 65 alla variabile *numero*. Il programma è quindi:

```
100 LET voto$ = "625671"
110 LET st$ = voto$ (3 TO 4)
120 LET num = 1.125 * st$
130 LET voto$ (3 TO 4) = num
140 PRINT voto$
```

Il risultato è ancora lo stesso.

Nella riga 120 il valore della stringa è stato convertito in forma numerica per poter essere moltiplicato: nella riga 130 un numero è stato convertito in stringa. Queste conversioni sono chiamate **coercizioni**.

Il programma può essere scritto in forma più sintetica:

```
100 LET voto$ = "625671"
110 LET voto$ (3 TO 4) = 1.125 * voto$ (3 TO 4)
120 PRINT voto$
```

RICERCA DI UNA STRINGA

Se si conoscono altre versioni di Basic, non si può non apprezzare la semplicità e la grande utilità delle coercizioni.

È possibile anche un'operazione inversa: risalire ad una stringa da una sottostringa con l'istruzione **INSTR**.

Ad esempio:

```
100 LET giungla$ = "ABLEONE CT"
110 INPUT "animale$"
120 IF animale$ INSTR giungla$ THEN PRINT "corretto"
130 ELSE
140       PRINT "sbagliato"
150 END IF
```

L'operatore **INSTR** dà come valore zero se il contenuto della variabile *animale$* non è trovato nella stringa *giungla$*.

L'espressione:

animale$ INSTR giungla$

può essere considerata come un'espressione logica.

ALTRE FUNZIONI

L'istruzione **LEN**, già descritta in precedenza, calcola il numero di caratteri di una stringa.

Supponiamo di voler ripetere un carattere più volte. Se ad esempio vogliamo stampare una riga piena di asterisco, invece di creare un'iterazione, è possibile scrivere:

```
PRINT FILL$ ("*", 40)
```

Si può utilizzare, infine, la funzione **CHR$** per convertire i codici interni in caratteri di stringa.

```
PRINT CHR$ (65)
```

stampa la lettera A.

CONFRONTI FRA STRINGHE

Uno degli aspetti più significativi e utili dei computer è l'organizzazione dei dati e la loro ricerca. Talvolta è necessario classificare i dati secondo il loro ordine alfabetico. La base di ogni processo di classificazione è il confronto fra stringhe per determinare la precedente. Poiché le lettera A, B, C.. hanno rispettivamente i codici 65, 66, 67.. è naturale considerare corrette le seguenti istruzioni:

A è minore di B
B è minore di C

e poiché il confronto interno, carattere per carattere, è eseguito automaticamente allora:

CANE è minore di GATTO

Si può scrivere ad esempio:

```
IF "CANE" < "GATTO" THEN PRINT "BAU"
```

ed è visualizzato:

```
BAU
```

Allo stesso modo:

```
IF "GATTO" > "CANE" THEN PRINT "MIAO"
```

visualizza:

```
"MIAO"
```

Anche nei confronti tra stringhe sono utilizzati i simboli matematici di confronto.
Tutte le seguenti espressioni logiche sono possibili e vere.

```
"ALF"  <   "BEN"
"KIT"  >   "BEN"
"KIT"  <=  "LEN"
"KIT"  >=  "KIT"
"PAT"  >=  "LEN"
"LEN"  <=  "LEN"
"PAT"  <>  "PET"
```

Una stringa è minore di un'altra se il primo carattere della prima stringa precede, nell'ordine alfabetico, il primo carattere della seconda stringa. Le lettere minuscole sono maggiori delle maiuscole. I numeri sono minori delle lettere.

Se, durante il confronto fra due stringhe, si raggiunge la fine di una stringa, quella più corta è considerata la più piccola.

> **Maggiore di**
Confronto alfabetico dipendente dai caratteri, maiuscoli o minuscoli, i numeri sono confrontati nell'ordine usuale.

< **Minore di**
Confronto alfabetico dipendente dai caratteri, maiuscoli o minuscoli, i numeri sono confrontati nell'ordine usuale.

= **Uguale a**

== **Equivalente a**
Le stringhe devono essere "quasi" uguali. Non dipende dai caratteri maiuscoli o minuscoli. I numeri sono confrontati nell'ordine usuale.

>= **Maggiore di, o uguale**
Confronto indipendente dai caratteri, maiuscoli o minuscoli. I numeri sono confrontati nell'ordine usuale.

<= **Minore di, o uguale**
Confronto indipendente dai caratteri, maiuscoli o minuscoli. I numeri sono confrontati nell'ordine usuale.

PROBLEMI SUL CAPITOLO 11

1. Inserire 12 lettere, tutte diverse, in una stringa, e altre sei in una seconda stringa. Cercare nella prima stringa ogni lettera della seconda e dire se è stata trovata o no.

2. Ripetere il primo esercizio sostituendo le stringhe con due vettori. Inserire venti lettere maiuscole casuali in una stringa ed elencare quelle ripetute.
3. Scrivere un programma per calcolare il numero di parole del testo riportato di seguito. La parola è riconoscibile perché inizia con una lettera, e seguita da uno spazio o da un punto o da altri caratteri di punteggiatura.

   "LE VOCI SULLA MIA MORTE SONO NOTEVOLMENTE ESAGERATE. CABLOGRAMMA DI MARK TWAIN ALL'ASSOCIAZIONE STAMPA, LONDRA 1896"
4. Riscrivere l'ultimo programma illustrando l'uso delle variabili logiche e delle procedure.

CAPITOLO 12
IL VIDEO

Il Superbasic dispone di una gestione molto sofisticata del video. Si è preferito suddividere la descrizione delle istruzioni di gestione del video in due parti: le istruzioni di semplice stampa e le istruzioni di gestione delle caratteristiche grafiche.

La prima parte del capitolo descrive come ottenere sul video la stampa di un testo. In questa sezione sono illustrate le regole di base per scrivere messaggi, testi o dati numerici.

La seconda parte è decisamente più vasta, ma permette di eseguire determinate operazioni in modo abbastanza semplice. Ad esempio è possibile disegnare un cerchio scrivendo semplicemente la parola **CIRCLE** seguita da altri elementi che indicano le coordinate del centro e la grendezza del raggio.

Ogni parola chiave è stata scelta in modo che il nome in Inglese rifletta gli effetti che provoca. **WINDOW** definisce un'area dello schermo; **BORDER** crea un bordo intorno ad essa; **PAPER** definisce il colore del fondo; **INK** determina il colore dei caratteri.

PRINT

La parola chiave **PRINT** deve essere seguita da un elenco di ciò che si vuole stampare:

un testo come: "questo è un testo"
una variabile: num, parola$
espressioni come: 3 * num, giorno$ & mese$

È possibile stampare più di un dato con la stessa istruzione **PRINT**, ognuno dei quali deve essere separato dagli altri da opportuni simboli. Separatori di stampa sono:

; permette di stampare il successivo elemento di seguito al precedente senza spazi

! inserisce uno spazio tra due elementi da stampare, o effettua un ritorno a capo alla riga successiva, se ciò che deve essere stampato non è contenuto per intero sulla riga corrente. Lo spazio all'inizio di una riga è eliminato

, funziona da tabulatore spostando la posizione di stampa di otto colonne per volta

/ effettua un ritorno a capo

TO permette l'incolonnamento.

ESEMPI

Supponiamo di voler stampare i numeri 1, 2, 3. Osserviamo nella seguente tabella gli effetti dei separatori.

Istruzione	Effetto
100 PRINT 1, 2, 3	1/2/3
100 PRINT 1! 2! 3!	1 2 3
100 PRINT 1/2/3	1 2 3
100 PRINT 1; 2; 3	123
100 PRINT "questo è un testo"	questo è un testo
100 LET parola$ = " " 110 PRINT parola$	sposta la posizione stampa
100 LET num = 13 110 PRINT num	13
100 LET an$ = "si" 110 PRINT "ho detto" ! an$	ho detto sì
110 PRINT "la somma" ! 4 + 2	La somma è 6

SEMPLICI STAMPE

È possibile stampare un dato in una qualsiasi posizione dello schermo con l'istruzione **AT**.

Ad esempio:

AT 10, 15 : PRINT "riga 10 colonna 15"

Il comando **CURSOR** permette di posizionare il dato da stampare in un qualsiasi punto della scala grafica dello schermo. Ad esempio:

CURSOR 100, 150: PRINT "100 pixel in orizzontale e 150 in verticale

In un'istruzione **PRINT** sono considerate espressioni:

un testo all'interno delle virgolette, le variabili ed i numeri.

I separatori sono classificati come informazioni di stampa.

LO SCHERMO

Questa parte del capitolo tratta l'utilizzo dello schermo e delle modalità di risoluzione grafica. L'istruzione:

MODE 8 o MODE 256

seleziona il MODE 8 composto da:

256 pixel orizzontali numerati da 0 a 511 (due valori per pixel)
256 pixel verticali numerati da 0 a 255
8 colori

Il pixel è l'area di colore minima che è possibile visualizzare. Con otto colori è possibile ottenere ombre ed altri effetti speciali. Se il QL è collegato ad un televisore questi effetti non sono visibili.

L'istruzione:

MODE 4 o MODE 512

seleziona il MODE 4 composto da:

512 pixel orizzontali numerati da 0 a 511
256 pixel verticali numerati da 0 a 255
4 colori

I COLORI

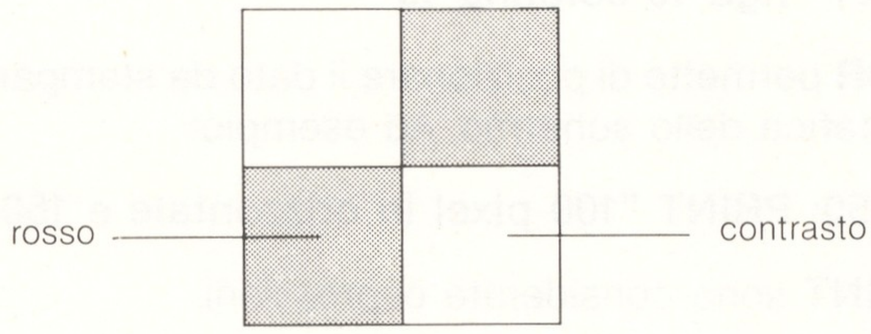
È possibile scegliere un colore usando i codici riportati nella seguente tabella, in combinazione con le parole chiave: **PAPER, INK**, ecc. È importante sottolineare che i numeri in se stessi non hanno alcun significato. Sono interpretati solo se usati con parole chiave.

Mode 8	codice	Mode 4
nero	0	nero
blue	1	nero
rosso	2	rosso
magenta	3	rosso
verde	4	verde
ciano	5	verde
giallo	6	bianco
bianco	7	bianco

Ad esempio **INK 3** seleziona il colore magenta nel Mode 8.

GLI STIPPLE

È possibile inserire due colori in un'istruzione. Ad esempio INK 2, 4 visualizza una "scacchiera" di quattro pixel. Due di questi sono rossi (codice 2) e corrispondono al colore selezionato per primo, gli altri due pixel sono nel colore di contrasto. Non è possibile riprodurre questo effetto su un televisore.



INK 2, 4

il colore è dato dai due codici 2 e 4. Queste scelte sono chiamate colore e contrasto.

INK colore, contrasto

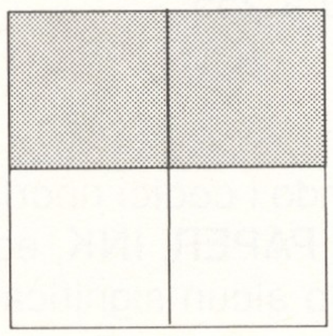
Scriviamo ora il seguente programma:

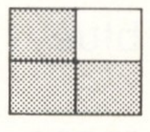
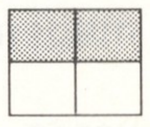
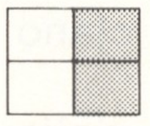
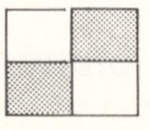
```
100 REMark colore/contrasto
110 FOR colore=0 TO 7 STEP 2
120         PAPER colore : CLS
140         FOR contrasto=0 TO 7 STEP 2
150             BLOCK 100, 50, 40, 50, colore, contrasto
160             PAUSE 50
170         END FOR contrasto
180 END FOR colore
```

Gli "Stipple" sono quadratini di 4 pixel, dei quali alcuni sono nel colore principale ed altri in quello di contrasto. Se si vogliono ottenere nuovi 'stipple' è sufficiente aggiungere un terzo codice. Ad esempio con:

INK 2,4,1

si ottiene un effetto a strisce orizzontali verdi e rosse. Un blocco di quattro pixel può essere il seguente:



Codice	Nome	Effetto
0	Singolo pixel di contrasto	
1	Strisce orizzontali	
2	Strisce verticali	
3	Scacchiera	

PARAMETRI DI COLORE

È possibile assegnare i colori di un effetto 'stipple' servendosi di tre parametri. Ad esempio:

INK *colore, contrasto, stipple*

Esistono tuttavia dei limiti:

colore: *da 0 a 7*
contrasto: *da 0 a 7*
stipple: *da 0 a 3*

Il seguente programma visualizza tutti i possibili effetti di colore:

```
100 REMark Effetti di colore
110 FOR num=0 TO 255
120         BLOCK 100,50,40,50,num
130         PAUSE 50
140 END FOR num
```

PAPER

PAPER seguito da uno, due o tre numeri determina il colore del fondo. Ad esempio:

```
PAPER 2          rosso
PAPER 2, 4       scacchiera rosso/verde
PAPER 2, 4, 1    strisce orizzontali rosse/verdi
```

Il colore non è visibile fino a che non è eseguita un'operazione di pulizia dello schermo con l'istruzione CLS.

INK

INK seguito da uno, due o tre numeri determina il colore dei caratteri stampati, delle linee o di altri grafici. I colori e gli effetti sono gli stessi dell'istruzione PAPER.

CLS

L'istruzione **CLS** cancella dallo schermo il colore attuale.

FLASHING

È possibile rendere lampeggianti le emissioni sullo schermo solo nel MODE 8. Il valore 1 attiva l'istruzione **FLASH**, mentre il valore 0 la annulla.

I FILE

I Microdrive sono stati fino ad ora utilizzati per memorizzare programmi, attraverso le istruzioni **LOAD** e **SAVE**. Le cartucce possono, tuttavia, essere utilizzate per memorizzare sia programmi che dati. La parola "file" indica una sequenza di record di dati. Un record è formato da una serie di informazioni correlate come nome, indirizzo e numero telefonico.

I tipi di file più usati sono i file sequenziali e i file ad accesso diretto. I dati scritti su un file sequenziale sono memorizzati sequenzialmente, uno dopo l'altro, nell'ordine in cui sono inviati. Ogni elemento è letto nello stesso modo, dal primo nel file fino all'ultimo. Così, se si desidera leggere il cinquantesimo record, è necessario leggere i precedenti quarantanove. Al contrario, nei file ad accesso diretto, è possibile trovare il record voluto in modo molto più rapido. Non è infatti necessario leggere i precedenti record. Uno degli esempi più semplici di file è quello composto da una sequenza di numeri. Per chiarire l'idea si vogliono scrivere in un file chiamato "numeri", i numeri da 1 a 100. Il nome completo del file si compone di due parti:

nome dell'unità
informazioni aggiuntive

Supponiamo di voler creare il file numeri, su una cartuccia inserita nel microdrive 1. Il nome dell'unità è:

mdv1_

e l'informazione aggiuntiva è il nome del file:

numeri

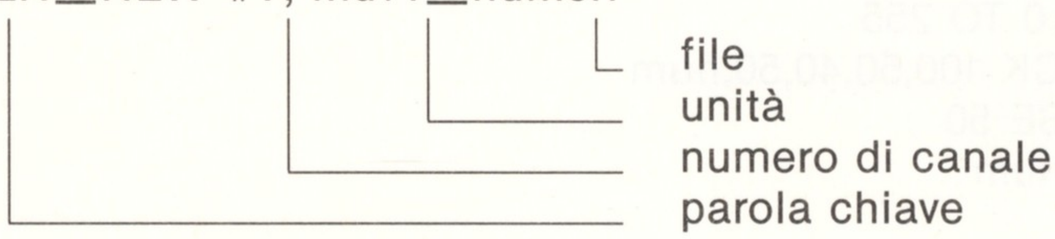
Così il nome completo del file è:

mdv1_numeri

CANALI

È possibile utilizzare più file all'interno di un unico programma. Ad ogni file è associato un numero di canale compreso fra 0 e 15. L'associazione è ottenuta con l'istruzione **OPEN** o, se si tratta di un nuovo file con **OPEN_NEW**. Ad esempio, si può scegliere il canale 7 per il file "numeri" e scrivere:

```
OPEN_NEW #7, mdv1_numeri
```



È possibile fare riferimento al file "numeri" semplicemente citando il simbolo #7. Il programma completo per creare il file ed inserire i valori è:

```
100 REMark file
110 OPEN_NEW #7, mdv1_numeri
120 FOR numero=1 TO 100
130         PRINT #7, numero
140 END FOR numero
150 CLOSE #7
```

L'istruzione **PRINT** permette la scrittura dei numeri nel file contenuto nella cartuccia, perché #7 è associato al file "numeri". L'istruzione CLOSE #7 è necessaria perché il sistema sappia che l'associazione fra il file numeri ed il canale #7 è conclusa. Il canale 7 è quindi utilizzabile per altre funzioni. Dopo l'esecuzione del programma scrivere:

```
DIR mdv_1
```

e si ottiene la conferma che il file "numeri" è stato memorizzato sulla cartuccia inserita nel Microdrive 1.

È necessario anche verificare che il file sia stato scritto in modo corretto. Per eseguire questa operazione è sufficiente leggerlo e controllarlo. La parola chiave **OPEN_IN** ha questa funzione. Il programma che legge i dati del file è simile al precedente.

```
100 REMark leggere un file
110 OPEN_IN #6, mdv1_numeri
120 FOR note=1 TO 100
130       INPUT #6, numero
140       PRINT ! numero !
150 END FOR note
160 CLOSE #6
```

Il programma visualizza i numeri da 1 a 100, ma solo se la cartuccia contenente il file numeri è ancora inserita nel Microdrive 1.

UNITÀ E CANALI

Nell'esempio precedente l'unità mdv_1 è stata associata ad un particolare canale. Alcune unità sono associate dal sistema stesso in modo definitivo a canali.

canale		uso
#0	OUTPUT	comando finestra
	INPUT	parola chiave
#1	OUTPUT	stampa finestra
#2	LIST	lista di un programma

WINDOW

È possibile costruire una finestra di qualsiasi grandezza in una posizione qualunque dello schermo. Il nome dell'unità associata ad una finestra è:

```
scr_
```

e le informazioni aggiuntive richieste sono:

```
scr_360x150a80x40
```

coordinata verticale
coordinata orizzontale
altezza
larghezza

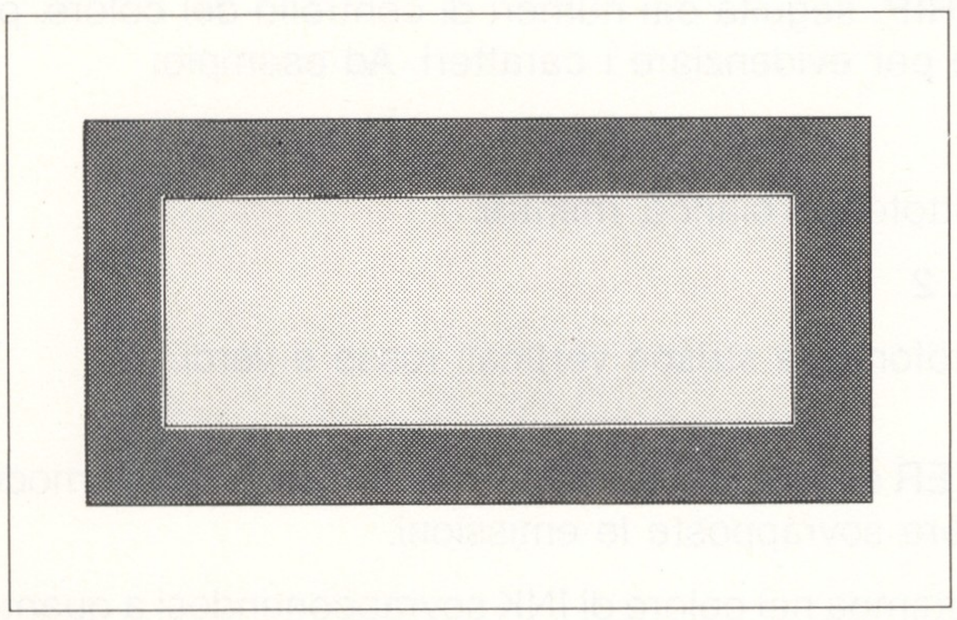
Il seguente programma genera una finestra associata al canale numero 5 e la riempie di verde (codice 4)

```
100 REMark crea una finestra
110 OPEN #5, scr_ 400x200a29x50
120 PAPER #5, 4 : CLS #5
130 CLOSE #5
```

È possibile variare la posizione o la dimensione di una finestra senza doverla ridefinire. Aggiungere due righe al precedente progamma:

```
124 WINDOW #5, 300, 100, 110, 65
126 PAPER #5, 2 : CLS #5
```

Rieseguendo il programma, una finestra rossa appare all'interno di quella originale verde. La finestra rossa è ora associata al canale 5.



BORDER

È possibile creare un bordo attorno allo schermo o ad una finestra. Ad esempio:

```
BORDER #5, 6
```

genera un bordo attorno alla finestra associata al canale 5. Il bordo ha uno spessore di 6 pixel e le dimensioni della finestra sono ridotte corrispondentemente. Il colore del bordo è indicato nel modo usuale.

```
BORDER #5, 6, 2
```

disegna un bordo rosso. È naturalmente possibile ottenere bordi di altri colori.

```
BORDER 10
```

aggiunge un bordo di 10 pixel, trasparente, alla finestra. (Il bordo è trasparente perché non è stato specificato alcun colore).

```
BORDER 2, 0, 7, 0
```

aggiunge bordi a strisce di due pixel di colore nero e bianco.

BLOCK

È possibile indicare le dimensioni, la posizione e il colore di un blocco con una singola istruzione.

```
BLOCK #5, 10, 20, 50, 100, 2
```

produce un rettangolo nella finestra associata al canale 5 con un vertice nel pun-

to di coordinate 50 orizzontale e 100 verticale. È largo 10 pixel e alto 20. Il colore dell'interno è rosso.

È importante sottolineare che le istruzioni WINDOW e BLOCK hanno gli stessi effetti sia nel Mode 4 che nel Mode 8, perché i valori delle coordinate orizzontali variano sempre da 0 a 511 e quelle verticali sono sempre 256.

CSIZE

Il Superbasic possiede un'istruzione che permette di modificare le dimensioni dei caratteri da stampare sullo schermo. Ad esempio:

```
CSIZE 3, 1
```

assegna al carattere l'ampiezza massima, e:

```
CSIZE 0, 0
```

assegna al carattere l'ampiezza minima. Vi sono, in tutto, quattro dimensioni possibili in larghezza, da 0 a 3;e due dimensioni in altezza da 0 a 1. All'accensione la dimensione dei caratteri è definita come 2,0 se si utilizza il MODE 256 e come 0,0 se si lavora con il MODE 512.

I numeri di riga e le colonne utilizzabili per ogni carattere dipendono dallo schermo.

In bassa risoluzione il QL non consente di scegliere una dimensione più piccola di quella di default.

MODE 4 CSIZE 0,0 25 linee × 84 colonne
MODE 8 CSIZE 1,0 25 linee × 42 colonne

STRIP

L'istruzione **STRIP**, seguita dai numeri di controllo del colore, produce un sottofondo speciale per evidenziare i caratteri. Ad esempio:

```
STRIP 7
```

produce un sottofondo bianco, mentre:

```
STRIP 2, 4, 2
```

genera un sottofondo a strisce verticali rosse e verdi.

OVER

L'istruzione **OVER** seguita dai numeri —1, 0 e 1 determina il modo secondo il quale devono essere sovrapposte le emissioni:

OVER— 1 stampa nel colore di INK sovrapponendosi a quanto è scritto, senza cancellarlo

OVER 0 Disattiva la sovrapposizione. I nuovi caratteri cancellano i precedenti e il colore di fondo è quello di STRIP.

OVER 1 Effettua la stampa nel colore attuale di INK sovrapponendola, senza cancellare, al colore di fondo.

UNDER

Con il comando **UNDER** il QL sottolinea, con il colore di INK, tutto ciò che deve essere stampato. Il comando seguito dal numéro uno è attivato, mentre seguito da 0 è disattivato.

SCALA GRAFICA

Se si vogliono disegnare sul monitor o su un apparecchio televisivo figure geometriche non è facile utilizzare come sistema di riferimento i pixel. È meglio servirsi del sistema di coordinate grafiche.

La scala del sistema di coordinate grafiche, assunta automaticamente dal computer è di 100 in verticale, mentre in orizzontale la scala è determinata di volta in volta.

L'origine del sistema grafico è nell'angolo in basso a sinistra della finestra video.

PUNTI E LINEE

È facile disegnare punti e righe usando una scala grafica. Con una scala verticale di 100, un punto vicino al centro può essere segnato con l'istruzione:

```
POINT 60, 50
```

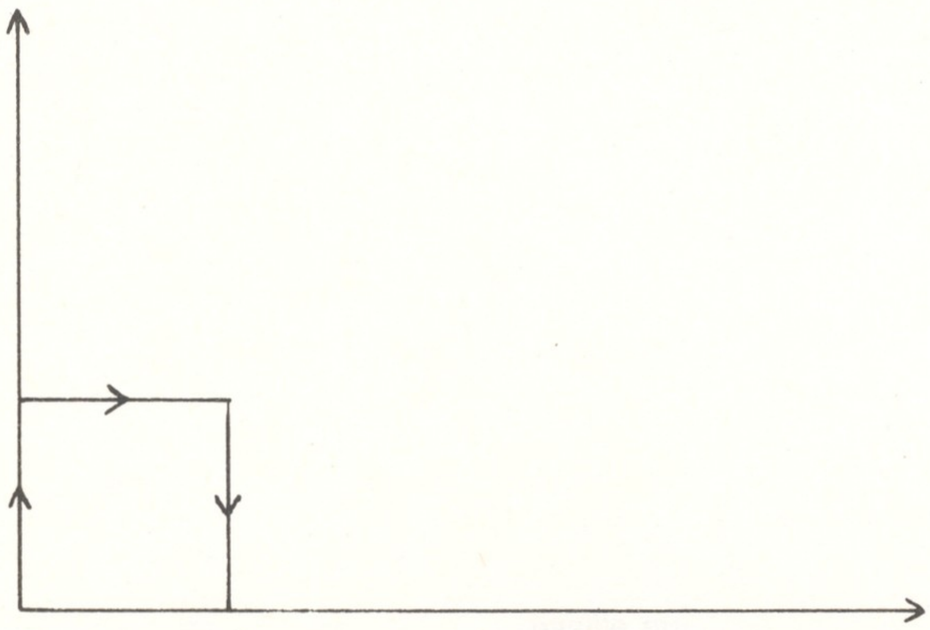
Il punto compare nel colore di INK corrente.

Allo stesso modo una linea può essere disegnata con l'istruzione:

```
LINE 60, 50 TO 80, 90
```

Possono essere aggiunti altri elementi. Ad esempio, per disegnare un quadrato è sufficiente scrivere:

```
LINE 60, 50 TO 70, 50 TO 70, 60 TO 60, 60 TO 60, 50
```



MODO RELATIVO

Le coordinate:

orizzontali e verticali

definiscono normalmente un punto relativo all'origine 0, 0 che si trova nell'angolo in basso a sinistra di una finestra. È talvolta più utile definire i punti in base alla posizione corrente del cursore. Ad esempio, il quadrato precedente può essere tracciato usando l'istruzione **LINE_R** che significa:

"usare le coordinate relative alla posizione corrente del cursore"

```
POINT 60,50
LINE_R 0, 0 TO 10, 0 TO 0, 10 TO —10, 0 TO 0, —10
```

Il punto 60,50 diventa l'origine, poi, quando le linee sono tracciate, la fine di ogni riga è l'origine della successiva.

Il seguente programma traccia una serie di quadrati in posizioni casuali.

```
100 REMark quadrati colorati
110 PAPER 7 : CLS
120 FOR qu=1 TO 100
130         INK RND (1 TO 6)
140         POINT RND (90), RND (90)
150         LINE_R 0, 0 TO 10, 0 TO 0, 10 TO —10, 0 TO 0, —10
160 END FOR qu
```

CERCHI ED ELLISSI

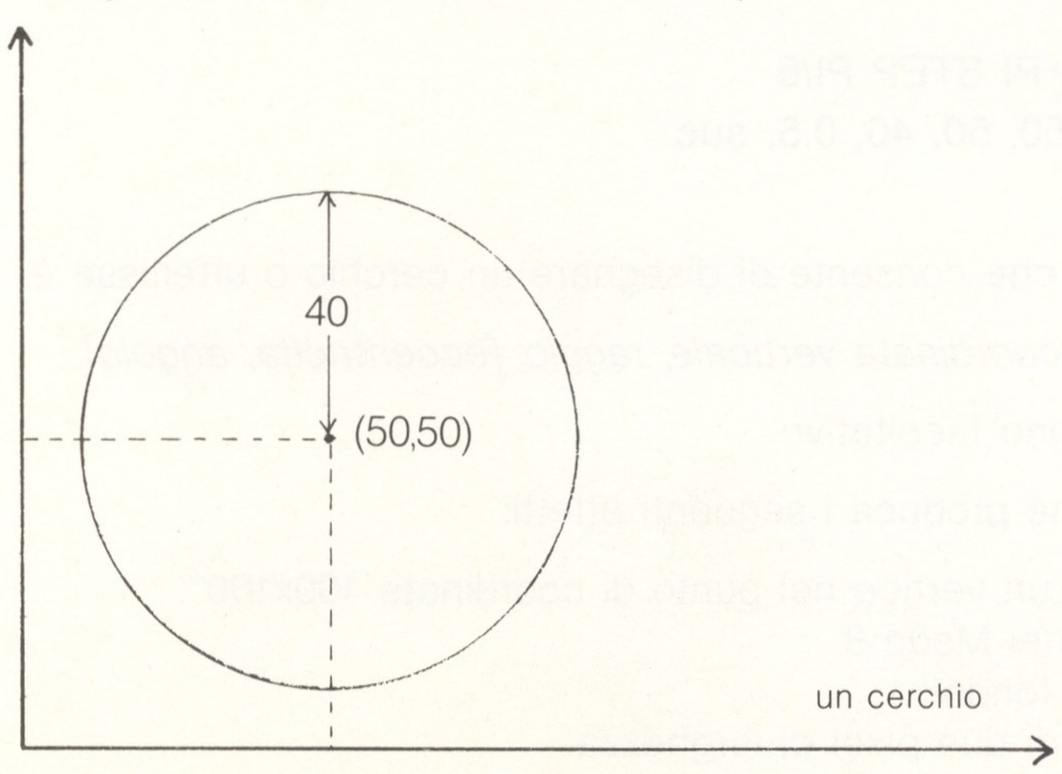
Se si vuole disegnare un cerchio è necessario indicare:

le coordinate del centro, ad esempio 50, 50
il raggio, ad esempio 40

L'istruzione:

```
CIRCLE 50, 50, 40
```

disegna un cerchio con il centro nel punto di coordinate 50, 50 e il raggio di 40.

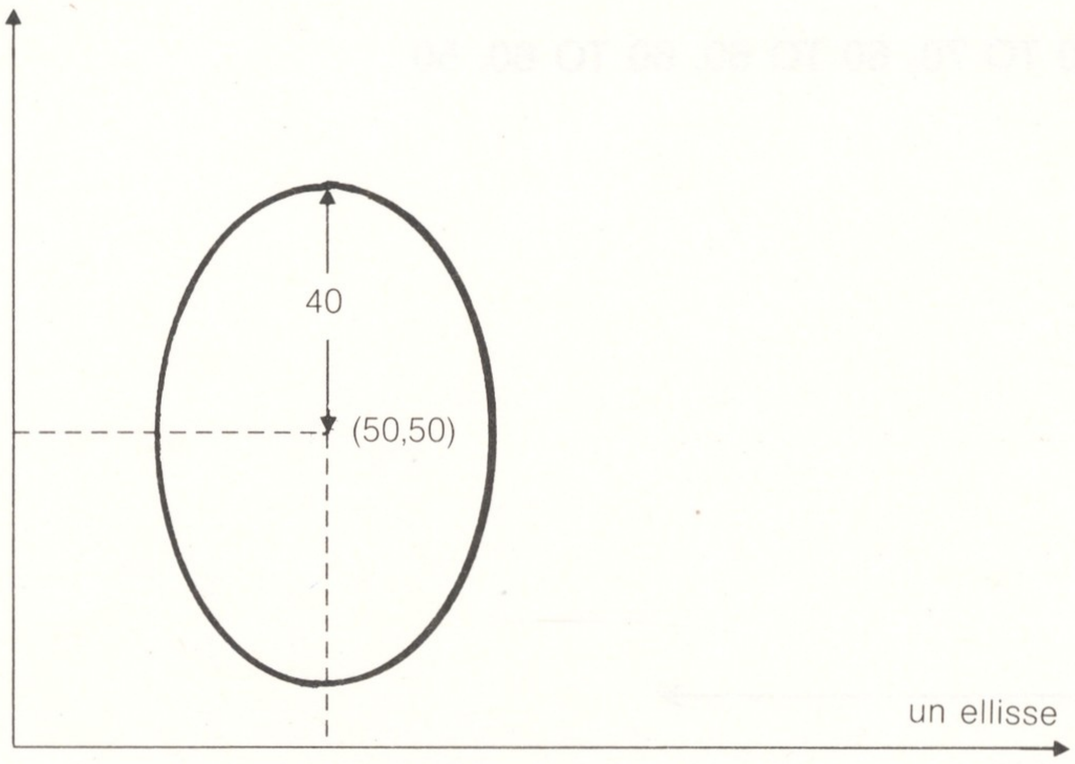


un cerchio

Se si aggiunge un altro parametro:

```
CIRCLE 50, 50, 40, .5
```

si ottiene un'ellisse. Le parole chiave CIRCLE e ELLIPSE sono equivalenti.

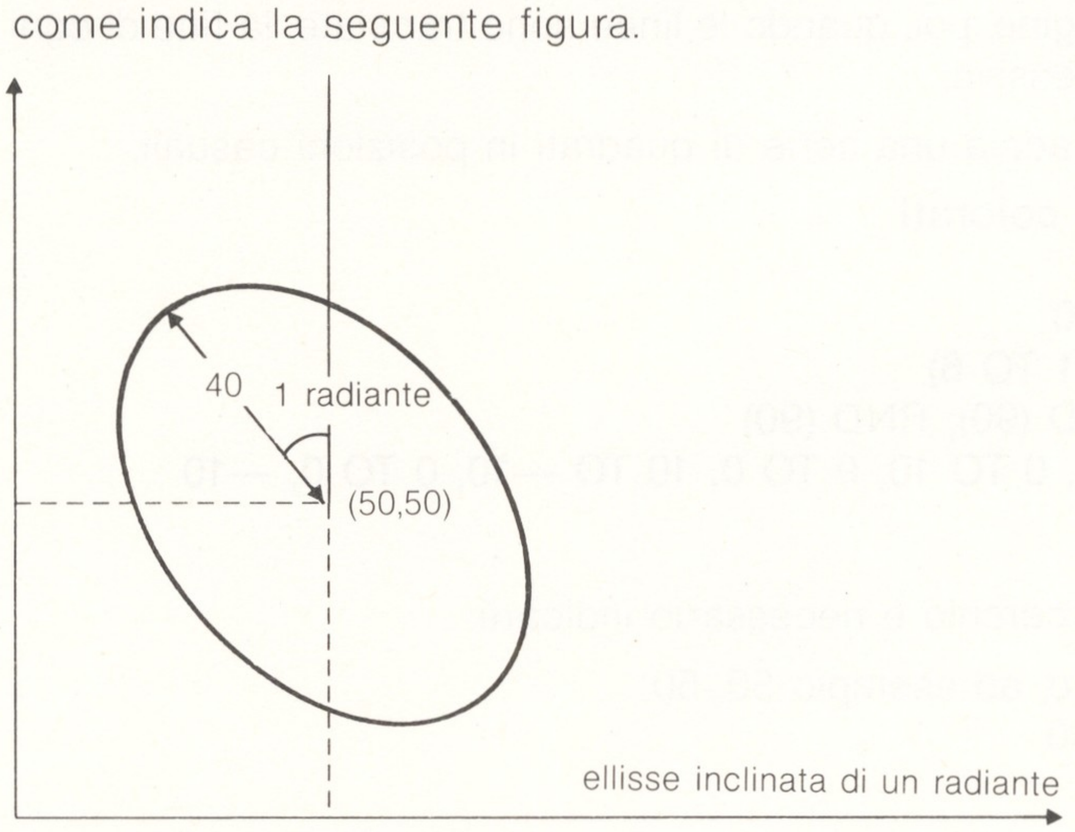


L'altezza dell'ellisse è 40 come prima, ma il raggio è ora 0.5 dell'altezza. Il numero 0.5 è chiamato eccentricità. Se l'eccentricità è uguale a 1 si ha un cerchio; se è minore di 1, ma maggiore di 0 si ha un'ellisse. Se si vuole inclinare un'ellisse, rispetto all'asse verticale dello schermo è sufficiente introdurre un quinto parametro.

L'istruzione:

```
CIRCLE 50, 50, 40, .5, 1
```

provoca l'inclinatura dell'ellisse in senso antiorario di un radiante, circa 57 gradi, come indica la seguente figura.



Un angolo piatto è di 180 gradi o PI-GRECO radianti. Il programma disegna una serie di ellissi:

```
100 FOR suc=0 TO 2*PI STEP PI/6
110           CIRCLE 50, 50, 40, 0.5, suc
120 END FOR suc
```

La sequenza di parametri che consente di disegnare un cerchio o un'ellisse è:

coordinata orizzontale, coordinata verticale, raggio, [eccentricità, angolo]

Gli ultimi due parametri sono facoltativi.

ESEMPIO Scrivere un programma che produca i seguenti effetti:

1. Aprire una finestra con un vertice nel punto di coordinate 100x100
2. Utilizzare la scala 100 nel Mode 8
3. Selezionare il colore di fondo
4. Creare un bordo verde di due pixel di larghezza

5. Disegnare un insieme di sei ellissi colorate
6. Chiudere la finestra.

```
100 REMark insieme
110 MODE 8
120 OPEN #7, scr_100x100a 100x50
130 SCALE #7, 100, 0, 0
140 PAPER #7, 0 : CLS #7
150 BORDER #7, 2, 4
160 FOR colore=1 TO 6
170         INK #7, colore
180         LET suc=2*PI/colore
190         CIRCLE #7, 50, 50, 30, 0.5, suc
200 END FOR colore
210 CLOSE #7
```

È possibile ottenere effetti molto interessanti apportando piccole modifiche al programma:

```
160 FOR colore=1 TO 100
180 LET suc=colore*PI/50
```

ARCHI

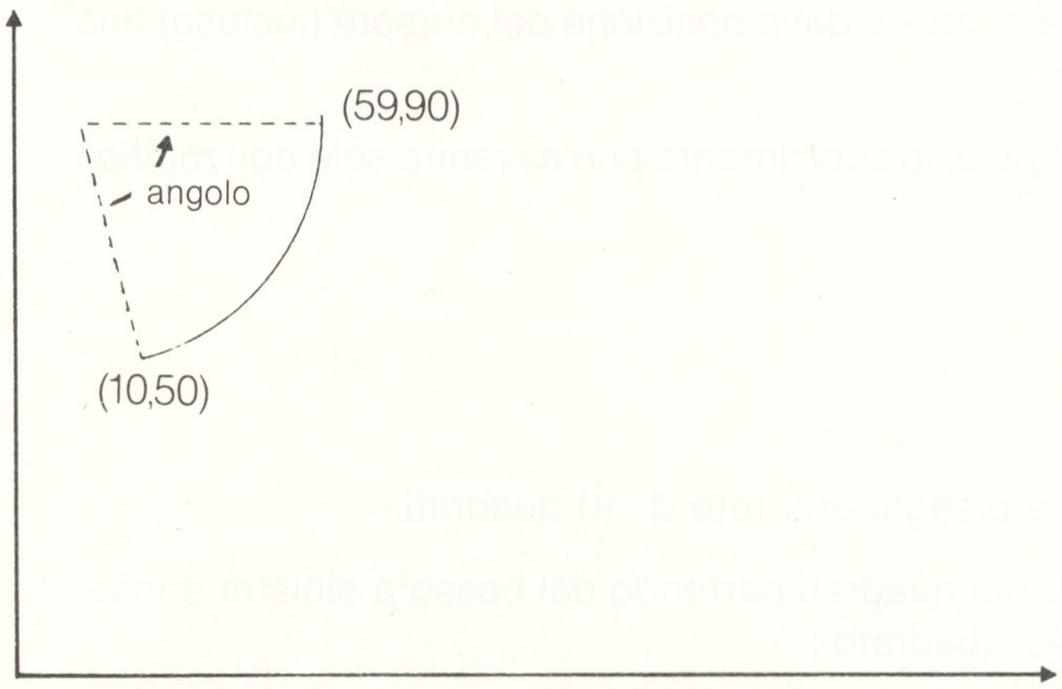
Per disegnare un arco di cerchio è necessario indicare:

il punto iniziale
il punto finale
il grado di curvatura

Le prime due informazioni sono semplici, ma il grado di curvatura non lo è. È necessario decidere l'angolo che l'arco deve sottendere e indicare l'angolo in radianti. Un angolo di 1.5 radianti dà una curvatura acuta mentre un angolo piccolo dà una curvatura leggera. Ad esempio:

```
ARC 10, 50 TO 50, 90, 1
```

disegna una curvatura leggera nel colore corrente di INK.



FILL

L'istruzione **FILL** consente di 'riempire' figure chiuse con il colore corrente di INK scrivendo semplicemente:

```
FILL 1
```

L'istruzione FILL deve precedere le istruzioni necessarie al disegno delle figure.
Il seguente programma disegna un cerchio verde:

```
INK 4
FILL 1
CIRCLE 50, 50, 30
```

L'istruzione:

```
FILL 0
```

disattiva il riempimento.

SCROLL E PAN

Con il QL è possibile ottenere lo scorrimento del contenuto di una finestra, come se si stesse azionando una telecamera.

L'istruzione **SCROLL** sposta il contenuto della finestra in senso verticale. È possibile specificare la direzione ed il numero di pixel dello scorrimento. Ad esempio:

```
SCROLL 10
```

sposta il contenuto della finestra di 10 pixel verso l'alto; mentre:

```
SCROLL —8
```

sposta il contenuto della finestra di 8 pixel verso il basso.

Se invece si vuole far scorrere solo una parte della finestra, è sufficiente aggiungere un secondo parametro, separandolo dal primo con una virgola. Così il comando:

```
SCROLL —8, 1
```

fa scorrere solo la parte di finestra fino alla linea precedente il cursore; mentre:

```
SCROLL —8, 2
```

effettua lo scorrimento della parte inferiore della finestra a partire dalla linea successiva a quella contenente il cursore.

Gli spazi rimasti vuoti dopo lo scorrimento sono riempiti dal colore corrente di PAPER.

Con l'istruzione **PAN** si ottiene lo scorrimento in senso orizzontale del contenuto della finestra corrente.

PAN 40 provoca lo scorrimento verso destra di 40 pixel
PAN —40 provoca lo scorrimento verso sinistra di 40 pixel

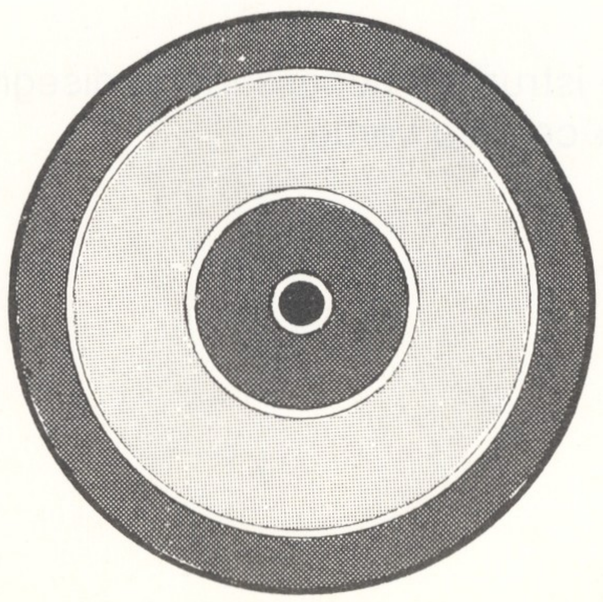
L'istruzione PAN può essere seguita da due parametri. Il primo controlla il movimento dei pixel, il secondo può essere un valore fra 0, 3 o 4.

0 — provoca lo scorrimento dell'intero schermo
3 — provoca lo scorrimento solo della linea contenente il cursore
4 — la linea del cursore è fatta scorrere dalla posizione del cursore (incluso) fino al termine.

Nel Mode 8 o in presenza di 'stipple', lo scorrimento può avvenire solo con multipli di 2 pixel.

PROBLEMI SUL CAPITOLO 12

1. Scrivere un programma che disegni una rete di 10 quadrati.
2. Inserire i numeri da 1 a 100 nei quadrati partendo dal basso a sinistra e inserendo la lettera F nell'ultimo quadrato.
3. Disegnare un bersaglio sullo schermo, che comprende un anello esterno contenente dei numeri. Un doppio e un triplo anello e un centro, che è un anello attorno ad esso.



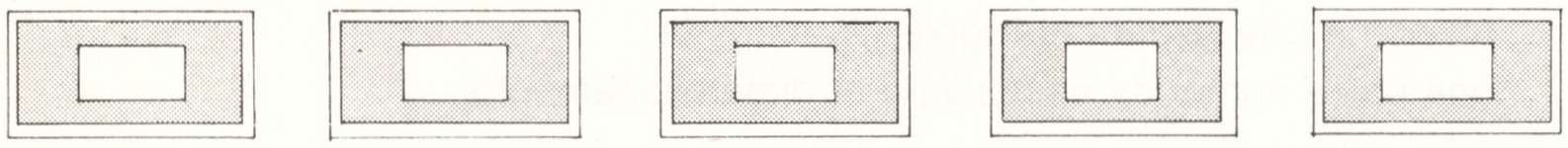
CAPITOLO 13
I VETTORI
PERCHÉ I VETTORI

Si supponga di essere il direttore di una prigione e di avere un nuovo blocco chiamato Ovest. Il blocco è pronto per ospitare 50 nuovi prigionieri. È necessario assegnare una cella ad ogni prigioniero. Non è necessario attribuire un nome ad ogni cella, ma è più semplice assegnare ad ognuna un numero da 1 a 50.

Per simulare il problema con il computer, è sufficiente considerare solo 5 prigionieri numerati, inseriti nel computer tramite un'istruzione DATA:

```
DATA 50, 37, 86, 41, 32
```

Costruiamo un vettore, chiamato Ovest, i cui elementi sono:



ovest (1) ovest (2) ovest (3) ovest (4) ovest (5)

È necessario dichiarare il vettore ed indicarne le dimensioni con un'istruzione **DIM**:

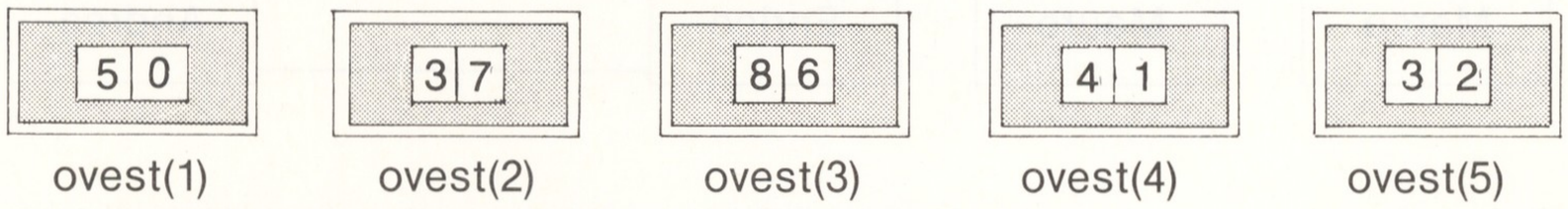
```
DIM ovest (5)
```

Dopo che l'istruzione DIM è stata eseguita, le cinque variabili sono utilizzabili.

I numeri dei prigionieri possono essere letti con l'istruzione **READ** e assegnati ai cinque elementi che compongono il vettore.

```
FOR cella=1 TO 5 : READ ovest(cella): END FOR cella
```

È possibile aggiungere un'iterazione **FOR** con un'istruzione **PRINT** per verificare che ogni prigioniero sia stato assegnato ad una cella:



ovest(1) ovest(2) ovest(3) ovest(4) ovest(5)

Il seguente programma è quindi completo:

```
100 REMark prigionieri
110 DIM ovest(5)
120 FOR cella=1 TO 5 : READ ovest(cella): END FOR cella
130 FOR cella=1 TO 5 : PRINT cella ! ovest(cella): END FOR cella
140 DATA 50, 37, 86, 41, 32
```

L'output del programma è:

```
1 50
2 37
3 86
4 41
5 32
```

I numeri da 1 a 5 sono chiamati elementi del vettore numerico ovest.

Se si sostituisce la riga 130 con:

```
130 PRINT ovest
```

sono visualizzati solo i valori:

```
0  86
50 41
37 32
```

Il numero 0 appare perché gli indici di un vettore sono numerati da zero fino alla dimensione dichiarata.

Quando un vettore numerico è DIMensionato, a tutti i suoi elementi è attribuito un valore nullo (0)

VETTORI ALFABETICI

I vettori alfabetici sono simili ai vettori numerici, ma sono DIMensionati in modo leggermente diverso. È necessario inserire nell'istruzione DIM un ulteriore indice, che determina la lunghezza delle stringhe componenti il vettore.

Si supponga che dieci dei migliori giocatori del campionato italiano di calcio siano indicati con il loro nome, e inseriti in istruzioni DATA.

```
DATA "Marco", "Mauro", "Enrico", "Filippo", "Piero"
DATA "Paolo", "Roberto", "Sergio", "Luigi", "Andrea"
```

In questo modo sono necessarie dieci variabili, ma se ci fossero cento o mille giocatori, il lavoro diventerebbe impossibile. Un vettore ha la funzione di risolvere problemi simili. Ogni nome di vettore è composto da due parti:

un nome che rispetti le regole già studiate
una parte numerica necessaria al dimensionamento.

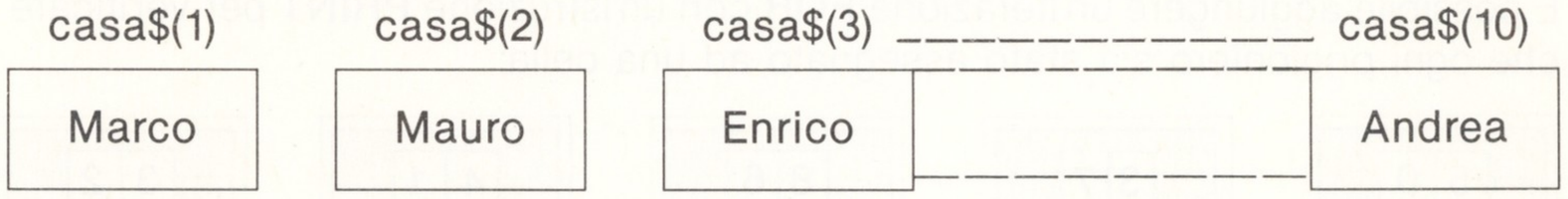
Prima di utilizzare gli elementi del vettore è necessario indicarne le dimensioni:

```
DIM casa$(10, 8)
```

L'operazione mette a disposizione 11 variabili (da 0 a 10) utilizzabili nel programma. Ogni elemento del vettore può contenere fino a otto caratteri. È consigliabile inserire le istruzioni DIM all'inizio del programma. Scrivere:

```
FOR numero=1 TO 10 : READ casa$(numero): END FOR numero
```

Le istruzioni assegnano i nomi dei giocatori agli elementi del vettore casa$.



Supponiamo di voler invertire i posti di Enrico e Marco nel vettore. Uno di loro, ad esempio, Marco, deve essere memorizzato in una variabile temporanea per permettere ad Enrico di occupare il suo posto, senza che alcuna informazione sia perduta.

Quindi:

```
LET temp$=casa$(1): REMark Marco nella variabile temporanea
LET casa$(1)=casa$(3): REMark Enrico nella casa$(1)
LET casa$(3)=temp$: REMark Marco nella casa$(3)
```

Il seguente programma riassume quanto descritto:

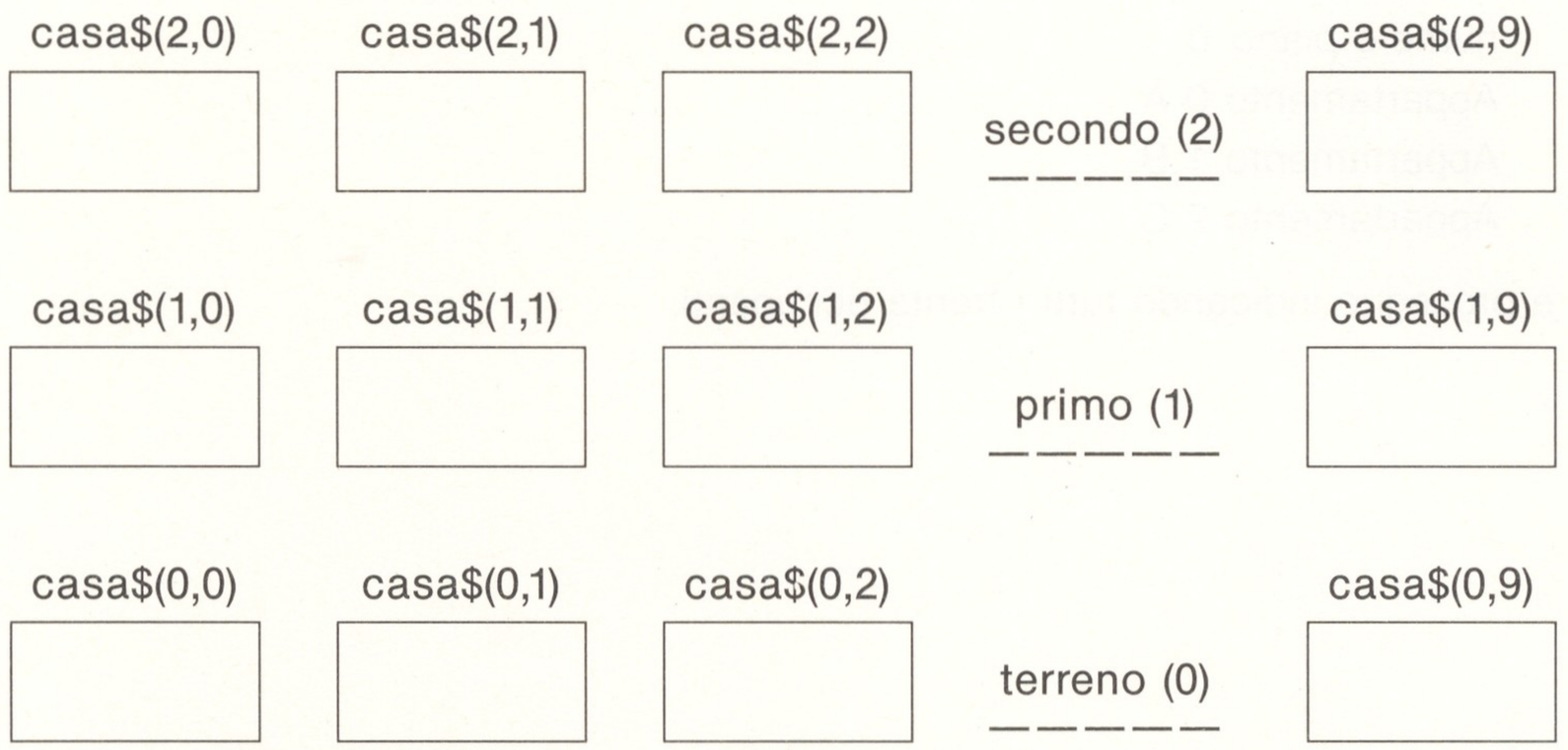
```
100 REMark casa dei giocatori
110 DIM casa$(10, 8)
120 FOR numero=1 TO 10 : READ casa$(numero): END FOR numero
130 stampa_lista
140 scambio
150 stampa_lista
160 REMark fine programma principale
170 DEFine PROCedure stampa_lista
180       FOR num=10 TO 10: PRINT num, casa$(num): END FOR num
190 END DEFine
200 DEFine PROCedure scambio
210       LET temp$=casa$(1)
220       LET casa$(1)=casa$(3)
230       LET casa$(3)=temp$
240 END DEFine
250 DATA "Marco", "Mauro", "Enrico", "Filippo", "Piero"
260 DATA "Paolo", "Roberto", "Sergio", "Luigi", "Andrea"
```

output riga 130	output riga 150
1 Marco	1 Enrico
2 Mauro	2 Mauro
3 Enrico	3 Marco
4 Filippo	4 Filippo
5 Piero	5 Piero
6 Paolo	6 Paolo
7 Roberto	7 Roberto
8 Sergio	8 Sergio
9 Luigi	9 Luigi
10 Andrea	10 Andrea

VETTORI BIDIMENSIONALI

Talvolta la natura di un problema richiede l'utilizzo di vettori bidimensionali.

Supponiamo di voler distribuire 30 giocatori in un palazzo di 30 appartamenti su tre piani.



Il sistema più corretto per inserire i nomi negli appartamenti è:

```
120 FOR piano=0 TO 2
130       FOR num=0 TO 9
140             READ casa$ (piano, num)
150       END FOR num
160 END FOR piano
```

È inoltre necessaria un'istruzione DIM:

```
110 DIM casa$ (2, 9, 8)
```

per indicare che il primo indice del vettore può variare da 0 a 2 e il secondo da 0 a 9. Il terzo indice determina il numero massimo di caratteri di ogni elemento del vettore.

Di seguito è riportato un programma di stampa per dimostrare che ai giocatori sono stati assegnati gli appartamenti. (Sono state usate le iniziali dei nomi dei giocatori per risparmiare spazio).

```
100 REMark 30 giocatori
110 DIM casa$ (2, 9, 8)
120 FOR piano=0 TO 2
130         FOR num=0 TO 9
140               READ casa$(piano, num): REMark i giocatori entrano
150         END FOR num
160 END FOR piano
170 REMark fine di input
180 FOR piano=0 TO 2
190         PRINT "numero piano" ! piano
200         FOR num=0 TO 9
210             PRINT 'casa' ! num ! casa$(piano, num)
220         END FOR num
230 END FOR piano
240 DATA "A", "B", "C", "D", "E", "F", "G", "H", "I", "J"
250 DATA "K", "L", "M", "N", "O", "P", "Q", "R", "S", "T"
260 DATA "U", "V", "W", "X", "Y", "Z", "A", "B", "C", "D"
```

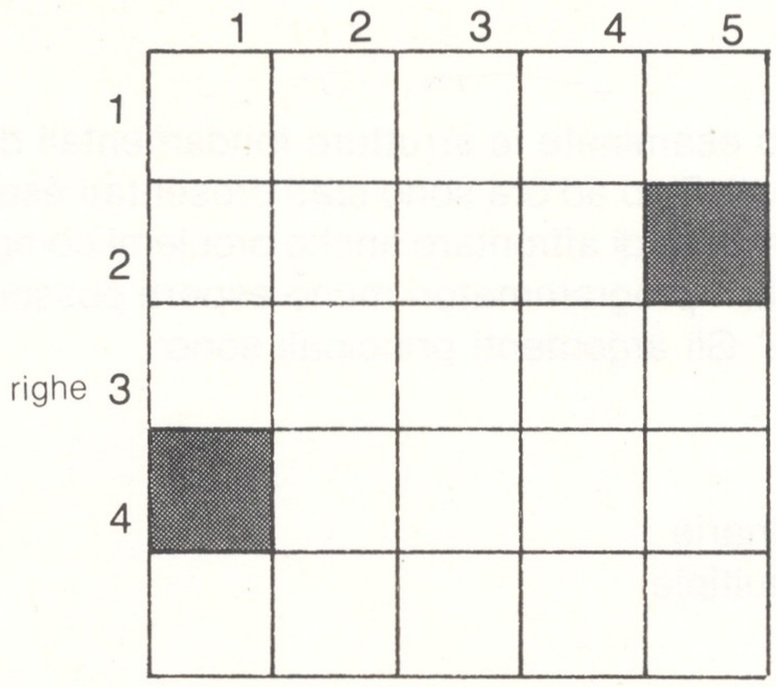
L'output inizia con:

```
numero piano 0
Appartamento 0 A
Appartamento 1 B
Appartamento 2 C
```

e prosegue indicando tutti i trenta occupanti.

PROBLEMI SUL CAPITOLO 13

1.



Le parole crociate hanno di solito un numero dispari di righe o colonne, nelle quali le caselle nere hanno una disposizione simmetrica. La disposizione è detta simmetrica perché la figura non cambia in una rotazione di 180 gradi.

Notare che dopo una rotazione di 180 gradi la casella nella riga 4, colonna 1 diventa la casella nella riga 2 colonna 5. Cioè, la riga 4, colonna 1 diventa la riga 2, colonna 5 in uno schema di parole crociate 5x5.

Scrivere un programma che crei e visualizzi la simmetria.

2. Modificare lo schema delle parole crociate in modo che non esista nessuna sequenza, né orizzontale né verticale, di quattro caselle bianche.

3. Le carte di un mazzo da gioco sono contrassegnate dai numeri da 1 a 52, memorizzati in un vettore. Le carte devono essere mescolate nel modo seguente:

   Scegliere una posizione tra 1 e 51, ad esempio 17.

   Inserire una carta in questa posizione in una variabile temporanea.

   Inserire le carte dalla 52 alla 18 in posizione da 51 a 17

   Inserire la carta scelta nella posizione 52.

   Fare le stesse operazioni con le carte da 1 a 50, 1 a 49 fino da 1 a 2, in modo che il mazzo sia ben mescolato. Visualizzare il risultato.

4. Scrivere sei istruzioni DATA contenente ognuna un cognome, iniziali e numero di telefono. Creare un insieme di vettori per memorizzare le informazioni. Stampare i dati usando una iterazione **FOR**.

CAPITOLO 14

LA STRUTTURA DEI PROGRAMMI

In questo capitolo sono esaminate le strutture fondamentali di un programma: i cicli iterativi e decisionali. Fino ad ora sono stati presentati esempi molto semplici, ma il Superbasic consente di affrontare anche problemi complessi. Alcune parti del capitolo sono difficili. I programmatori meno esperti possono omettere la lettura di questi paragrafi. Gli argomenti principali sono:

Iterazioni
Nidificazioni
Decisioni (scelte) binarie
Decisioni (scelte) multiple

ITERAZIONI

Nella prima parte, dedicata alle Iterazioni, sono illustrati i problemi che si incontrano nell'uso di cicli iterativi, attraverso le simulazioni di alcune scene. Il contesto può essere noioso e a volte sembrare insignificante, ma è la base per una discussione più approfondita ed è indicativo delle difficoltà che si incontrano nella programmazione.

ESEMPIO 1

Un bandito è bloccato in un edificio chiamato Vecchia Scuola e lo sceriffo ha sei colpi nella sua pistola. Simulare la sequenza degli spari.

PROGRAMMA 1

```
100 REMark Western
110 FOR colpi=1 TO 6
120    PRINT "prendi la mira"
130    PRINT "spara"
140 END FOR colpi
```

PROGRAMMA 2

```
100 REMark Western REPeat
110 LET colpi=6
120 REPeat bandito
130    PRINT "prendi la mira"
140    PRINT "spara"
150    LET colpi=colpi —1
160    IF colpi=0 THEN EXIT
170 END REPeat bandito
```

Entrambi i programmi visualizzano lo stesso output. Infatti le frasi:

prendi la mira
spara

sono stampate sei volte.

Se, in entrambi i programmi, il numero 6 fosse cambiato con un altro numero, il programma funzionerebbe come nel caso esaminato.

ESEMPIO 2

Ma supponiamo, che la pistola dello sceriffo sia scarica. Questa situazione, nei programmi, corrisponde alla sostituzione di 6 con 0.

PROGRAMMA 1

```
100 REMark Western Caso Zero
110 FOR colpi=1 TO 0
120    PRINT "prendi la mira"
130    PRINT "spara"
140 END FOR colpi
```

Il programma è eseguito senza che il Superbasic segnali alcun errore, ma non produce alcun output.

PROGRAMMA 2

```
100 REMark Western REPeat Caso Zero
110 LET colpi=0
120 REPeat bandito
130    PRINT "prendi la mira"
140    PRINT "spara"
150    LET colpi=colpi —1
160    IF colpi=0 THEN EXIT bandito
170 END REPeat bandito
```

Il programma non è corretto per due motivi.

1. Le frasi:
   prendi la mira
   spara

   sono stampate nonostante non vi sia alcun colpo nella pistola.

2. Nella riga 160 la variabile *colpi* ha un valore di —1 e non potrà mai assumere il valore zero. È necessario riscrivere la riga 160:

   160 IF colpi < 1 THEN EXIT bandito

L'errore è nella logica del programma, che non ha previsto come il valore iniziale della variabile *colpi* potesse essere zero. Per correggere questo errore, è necessario inserire un'istruzione condizionale **EXIT** prima delle istruzioni **PRINT**.

PROGRAMMA 3

```
100 REMark Western REPeat Caso Zero
110 LET colpi=0
120 REPeat Bandito
130       IF colpi=0 THEN EXIT bandito
140       PRINT "prendi la mira"
150       PRINT "spara"
160       LET colpi=colpi —1
170 END REPeat bandito
```

Ora il programma è corretto in tutti i casi.

Come indica la seguente, tabella i cicli iterativi **FOR** e **REPeat** hanno una struttura simile:

FOR *nome* =	(parola chiave d'apertura)	**REPeat** *nome*
(istruzioni)	(contenuto)	(istruzioni)
END FOR *nome*	(parola chiave di chiusura)	**END REPeat** *nome*

La struttura **REPeat**, tuttavia, deve contenere al suo interno un'istruzione **EXIT**, perché la ripetizione non sia eseguita all'infinito.

L'istruzione **EXIT** cede il controllo del programma all'istruzione immediatamente successiva all'istruzione **END REPeat**.

Un'istruzione **NEXT**, inserita in un'iterazione, riporta il controllo del programma all'istruzione che si trova subito dopo la parola chiave di apertura **FOR** o **REPeat**.

ESEMPIO 3

La situazione è la stessa del primo esempio. Lo sceriffo ha una pistola caricata con sei colpi e vuole sparare al bandito, ma esistono altre due condizioni:

1. Se lo sceriffo colpisce il bandito smette di sparare e torna a Dodge City.

2. Se lo sceriffo finisce i colpi prima di aver colpito il bandito, ordina ai suoi aiutanti di controllare il bandito e ritorna a Dodge City.

PROGRAMMA 1

```
100 REMark Western FOR
110 FOR colpi=1 TO 6
120       PRINT "prendi la mira"
130       PRINT "spara"
140       LET colpito=RND (9)
150       IF colpito=7 THEN EXIT colpi
160 NEXT colpi
170       PRINT "guarda il bandito"
180 END FOR colpi
190 PRINT "ritorna a Dodge City"
```

Le istruzioni contenute tra **NEXT** e **END FOR**, sono una specie di epilogo, eseguito quando l'iterazione **FOR** compie l'intero ciclo. Nel caso sia eseguita l'istruzione **EXIT**, il controllo del programma passa alla riga 190 ed è stampata la frase "ritorna a Dodge City".

È possibile ottenere lo stesso risultato utilizzando la struttura **REPeat**, anche se la precedente è la soluzione migliore.

PROGRAMMA 2

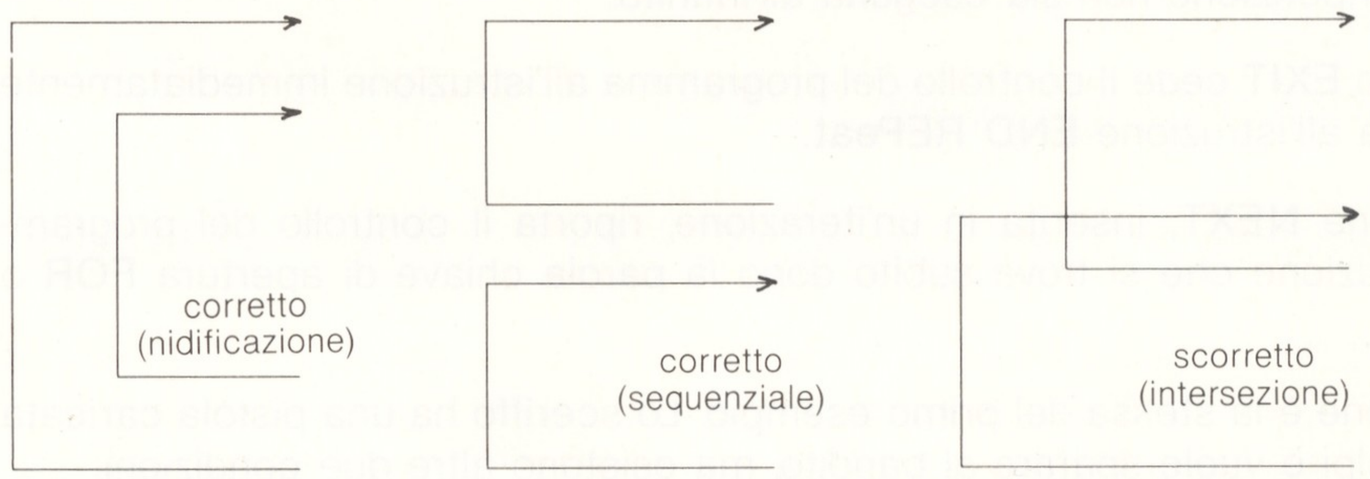
```
100 REMark Western REPeat
110 LET colpi=6
120 REPeat bandito
130       PRINT "prendi la mira"
140       PRINT "spara"
150       LET colpito=RND(9)
160       IF colpito=7 THEN EXIT bandito
170       LET colpi=colpi -1
180       IF colpi <> ø THEN NEXT bandito
190       PRINT "guarda bandito"
200 END REPeat bandito
210 PRINT "ritorna a Dodge City"
```

CICLI NIDIFICATI

Consideriamo la seguente iterazione **FOR** che traccia (**PLOT**) una serie di punti in diverso colore, scelti in modo casuale (non neri).

```
100 REMark Riga di pixel
110 PAPER 0 : CLS
120 LET verticale=50
130 FOR orizzontale=20 TO 60
140       INK RND (2 TO 7)
150       POINT orizzontale, verticale
160 END FOR orizzontale
```

Se si volessero ottenere 51 righe di punti, sarebbe sufficiente ripetere il programma precedente in modo che i valori della variabile verticale cambino da 30 a 80. È importante sottolineare che una struttura può essere contenuta all'interno di un'altra. È anche possibile ottenere una sequenza di due strutture, ma non è possibile intersecarle.



Lo schema del nuovo programma è quindi:

```
FOR verticale=30 TO 80
  FOR orizzontale=20 TO 60
    INK RND (2 TO 7)
    POINT orizzontale, verticale
  END FOR orizzontale
END FOR verticale
```

Il programma è:

```
100 REMark Righe di pixel
110 PAPER 0 : CLS
120 FOR verticale=30 TO 80
130   FOR orizzontale=20 TO 60
140     INK RND (2 TO 7)
150     POINT orizzontale, verticale
160   END FOR orizzontale
170 END FOR verticale
```

È possibile nidificare strutture di diverso tipo. Supponiamo, ad esempio, di sostituire l'iterazione **FOR** del programma precedente con una struttura **REPeat**. Il ciclo **REPeat** ha termine quando è selezionato il codice di colore 0.

```
100 REMark REPeat in FOR
110 PAPER 0 : CLS
120 FOR verticale=30 TO 80
130   LET orizzontale=19
140   REPeat punti
150     LET colore=RND (7)
160     INK colore
170     LET orizzontale=orizzontale+1
180     POINT orizzontale, verticale
190     IF colore=0 THEN EXIT punti
200   END REPeat punti
210 END FOR verticale
```

Le regole da rispettare sono due:

1. Usare sempre la struttura più adatta al caso in esame.
2. Ogni struttura può essere seguita da un'altra o può essere inserita completamente nel suo interno, ma non è possibile che due cicli iterativi si intersechino.

DECISIONI BINARIE

I tre tipi di decisione binaria sono facilmente illustrabili con un esempio: cosa fare quando piove.

a)
```
100 REMark forma breve di IF
110 LET piove=RND (0 TO 1)
120 IF piove THEN PRINT "apri l'ombrello"
```

b)
```
100 REMark forma lunga IF...END IF
110 LET piove=RND (0 TO 1)
120 IF piove THEN
130   PRINT "metti il cappotto"
140   PRINT "apri l'ombrello"
150   PRINT "cammina veloce"
160 END IF
```

c)
```
100 REMark forma lunga IF...ELSE...END IF
110 LET piove=RND (0 TO 1)
120 IF piove THEN
130   PRINT "prendi il bus"
140 ELSE
150   PRINT "cammina"
160 END IF
```

I primi due esempi sono semplici: accade o non accade un evento. La terza è una decisione binaria con due diversi possibili tipi di azione, ma entrambi devono essere definiti.

Nella forma più lunga è possibile omettere **THEN**. Nella forma breve è possibile sostituire **THEN** con il simbolo **:**

ESEMPIO

Consideriamo un esempio più complesso, nel quale sono nidificate decisioni binarie: calcoliamo le vocali, le consonanti e altri caratteri presenti nel testo riportato di seguito. Ignoriamo gli spazi. Per rendere l'operazione più semplice il testo è scritto in lettere maiuscole.

"LA STORIA DEL COMPUTER È STATA SCRITTA NEL 1984"

```
leggere i dati
FOR ogni carattere:
  IF lettera THEN
    IF vocale
      aumenta il numero delle vocali
    ELSE
      aumenta il numero delle consonanti
    END IF
    ELSE
    IF no spazio THEN aumenta il numero di altri caratteri
  END IF
END FOR
PRINT risultato
```

```
100 REMark conto dei caratteri
110 RESTORE 290
120 READ testo$
130 LET vocali=0 : cons=0 : altri=0
140 FOR num=1 TO LEN (testo$)
150   LET ca$=testo$(num)
160   IF ca$> ="A" AND ca$< ="Z"
170     IF ca$ INSTR "AEIOU"
180       LET vocali=vocali+1
190     ELSE
200       LET cons=cons+1
210     END IF
220   ELSE
230     IF ca$ <> " " THEN altri=altri+1
240   END IF
250 END FOR num
260 PRINT "il numero delle vocali è" ! vocali
270 PRINT "il numero delle consonanti è" ! cons
280 PRINT "il numero di altri caratteri è" ! altri
290 DATA "LA STORIA DEL COMPUTER È STATA SCRITTA NEL 1984"
```

```
Il numero delle vocali è 14
Il numero delle consonanti è 21
Il numero degli altri caratteri è 4
```

DECISIONI MULTIPLE

SELect

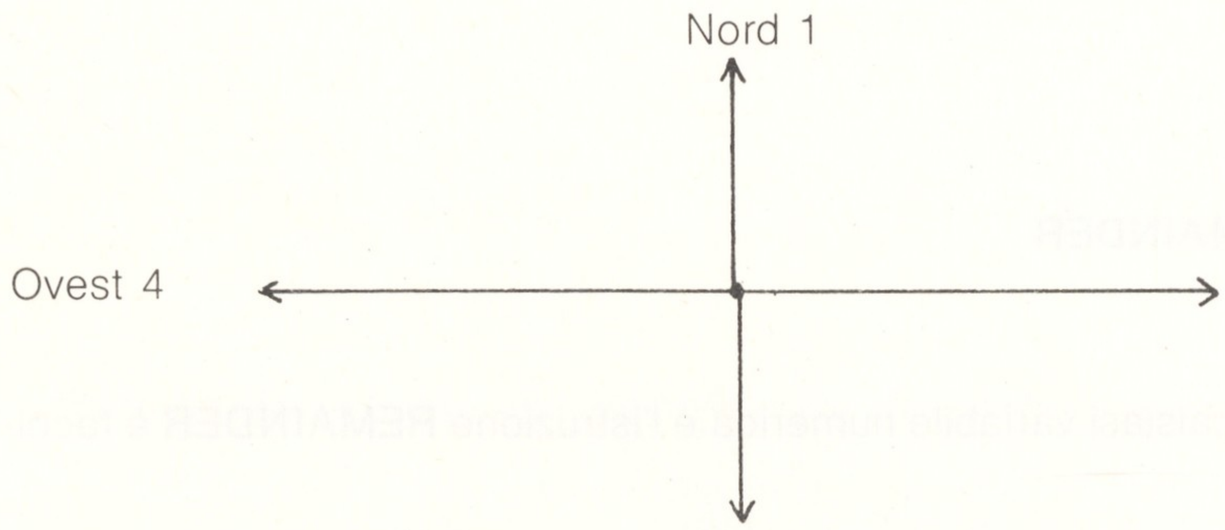
Quando esistono due o tre possibili azioni e nessuna dipende da una scelta precedente, la struttura più appropriata è **SELEect**.

ESEMPIO

Un serpente magico cresce senza limiti. La crescita avviene aggiungendo alla testa segmenti lunghi venti unità. Ogni segmento deve essere rivolto in una di queste direzioni: Nord, Sud, Est o Ovest. Il serpente è al centro della finestra.

METODO

Mentre il serprente è ancora sullo schermo, scegliere una lunghezza casuale e un colore di INK. La direzione della crescita è determinata con i numeri 1, 2, 3 e 4 come indica la seguente figura:



DISEGNO DEL PROGRAMMA

Selezionare **PAPER**
Costruire il serpente al centro della finestra
REPeat
  Scegliere: direzione, colore, lunghezza di crescita
  FOR unità=1 a fine
    Fare crescere il serpente a: nord, sud, est o ovest
    IF serpente esce dalla finestra **THEN EXIT**
  END FOR
END REPeat
PRINT fine

PROGRAMMA

```
100 REMark serpente magico
110 PAPER 0 : CLS
120 LET orizzontale=50 : verticale=50
130 REPeat serpente
140   LET direzione=RND (1 TO 4) : colore=RND (2 TO 7)
150   LET crescita=RND (2 TO 20)
160   INK colore
170   FOR unit=1 TO crescita
180     SELect ON direzione
190       ON direzione=1
200         LET verticale=verticale+1
210       ON direzione=2
220         LET orizzontale=orizzontale+1
230       ON direzione=3
240         LET verticale=verticale—1
250       ON direzione 4
260         LET orizzontale=orizzontale—1
270     END SELect
280     IF orizzontale<1 ORorizzontale>99 OR verticale<1 OR
        verticale>99 THEN EXIT serpente
290     POINT orizzontale, verticale
300   END FOR unita
310 END REPeat serpente
320 PRINT "serpente fuori dallo schermo"
```

La struttura **SELect ON** è così formata:

```
SELect ON num
ON num=elenco di valori
   Istruzioni
ON num=lista di valori
   Istruzioni
   —
   —
   —
   —
ON num = REMAINDER
   Istruzioni
   END SELect
```

dove num è una qualsiasi variabile numerica e l'istruzione **REMAINDER** è facoltativa.

Esiste una forma più breve della struttura **SELect**. Ad esempio:

```
100 INPUT num
110 SELect ON num=0 TO 9 : PRINT "numero"
```

PROBLEMI SUL CAPITOLO 14

1. Memorizzare 10 numeri in un vettore e creare uno schema di ordinamento chiamato "bubble sort". L'algoritmo consiste nel confrontare il primo numero con i successivi, scambiando le posizioni, se necessario, fino al nono numero. La prima serie di nove confronti inserisce i numeri più alti nelle loro corrette posizioni. Con altre otto serie di confronti si correggono altre otto posizioni. In tal modo solo il numero più basso deve essere inserito nella unica posizione (corretta) rimasta libera.

2. Esaminare come è possibile accelerare il bubble sort.

3. Un banditore d'asta vuole vendere un vecchio orologio, e chiede come prima offerta 50 dollari. Se nessuno accetta, può scendere a 40, 30, 20, ma non più in basso. Se nessuno accetta, l'orologio è ritirato dall'asta. Quando l'asta inizia, accetta rialzi di 5 dollari per volta, finchè l'orologio non è aggiudicato. Se l'offerta finale è almeno di 35 dollari o più, l'orologio è venduto, altrimenti è ritirato dall'asta.

   Simulare l'asta lanciando un dado. Il punteggio 'sei' provoca la prosecuzione dell'asta indipendentemente dal prezzo iniziale.

CAPITOLO 15

PROCEDURE E FUNZIONI

Nella prima parte di questo capitolo sono illustrate due fra le più importanti caratteristiche del Superbasic: le procedure e le funzioni.

Il Superbasic permette operazioni semplici in modo semplice, ma può offrire molto di più.

PARAMETRI

Nei precedenti capitoli è stato esaminato come un valore può essere trasferito all'interno di una procedura. Ecco un altro esempio:

ESEMPIO

Nel menù di Chan, un self service cinese, ci sono solo sei piatti.

Riso	Dolce
1 gamberi	4 gelato
2 pollo	5 frittelle
3 speciale	6 lyches

Chan calcola i prezzi in modo molto semplice:

Un piatto di riso costa: 300+10 moltiplicato il numero di ordine che compare sul menù.

Un gelato costa: 12 moltiplicato quattro che è il numero di ordine che compare sul menù.

Chi mangia un piatto di riso speciale e un gelato paga quindi:

```
300 + 10 * 3 + 12 * 4=378
```

Una procedura, chiamata "conto", accetta un numero di menù come parametro e stampa il conto del cliente.

PROGRAMMA

```
100 REMark costo del piatto
110 conto 3
120 conto 4
130 DEFine PROCedure conto(num)
140    IF num < =3 THEN LET prezzo=300 +10 * num
150    IF num > =4 THEN LET prezzo=12 * num
160    PRINT ! prezzo !
170 END DEFine
```

l'output è: **330 48**

Nel programma principale sono utilizzati i parametri 3 e 4. La definizione di procedura ha un parametro formale, *num*, che assume il valore trasmessogli dal programma principale. I parametri formali devono essere racchiusi tra parentesi.

ESEMPIO

Ora supponiamo che la variabile *prezzo* sia usata nel programma principale con un significato diverso, ad esempio per identificare il prezzo di una birra (70 nell'esempio).

PROGRAMMA

```
100 REMark prezzo globale
110 LET prezzo=70
120 conto 3
130 conto 4
140 PRINT ! prezzo !
150 DEFine PROCedure conto(num)
160    IF num < =3 THEN LET prezzo=300 + 10 * num
170    IF num > =4 THEN LET prezzo=12 * num
180    PRINT ! prezzo !
190 END DEFine
```

l'output è: **330 48 48**

Il prezzo della birra è stato alterato dalla procedura. Infatti la variabile *prezzo* è **globale** perchè può essere inserita in una qualsiasi parte del programma.

ESEMPIO

Rendere la variabile, *prezzo* **LOCALE** nella procedura, significa che il Superbasic la tratta come una variabile speciale, accessibile solo dall'interno della procedura. La variabile *prezzo* del programma principale è quindi diversa dalla variabile **LOCALE** *prezzo*, benchè abbia lo stesso nome.

PROGRAMMA

```
100 REMark LOCAL prezzo
110 LET prezzo=70
120 conto 3
130 conto 4
140 PRINT ! prezzo !
150 DEFine PROCedure conto(num)
160    LOCAL prezzo
170    IF num < =3 THEN LET prezzo=300 + 10 * num
180    IF num > =4 THEN LET prezzo=12 * num
190    PRINT ! PREZZO !
200 END DEFine
```

L'output è: **330 48 70**

L'istruzione contenuta nella riga 160 fa in modo che la variabile *prezzo* sia definita solo all'interno della procedura conto. Il valore dell'altra variabile *prezzo* non è cambiato dalla procedura.

ESEMPIO

I parametri come *num*, non interferiscono con le variabili del programma principale. Per dimostrarlo cancelliamo l'istruzione **LOCAL** e utilizziamo *num* per identificare il prezzo della birra.

PROGRAMMA

```
100 REMark LOCAL parametro
110 LET num=70
120 conto 3
130 conto 4
140 PRINT ! num !
150 DEFine PROCedure conto(num)
160    IF num < =3 THEN LET prezzo=300 + 10 * num
170    IF num > =4 THEN LET prezzo=12 * num
180    PRINT ! prezzo !
190 END DEFine
```

l'output è: **330 48 70**

PARAMETRI VARIABILI

Fino ad ora sono stati usati solo parametri di procedura per passare i valori alla procedura. Ma supponiamo che il programma principale richieda di utilizzare il valore del costo di ogni piatto per calcolare il costo totale. L'operazione è eseguita facilmente inserendo un altro parametro nell'istruzione che chiama la procedura, detto parametro variabile, che deve essere dello stesso tipo del corrispondente parametro dichiarato nella definizione di procedura.

ESEMPIO

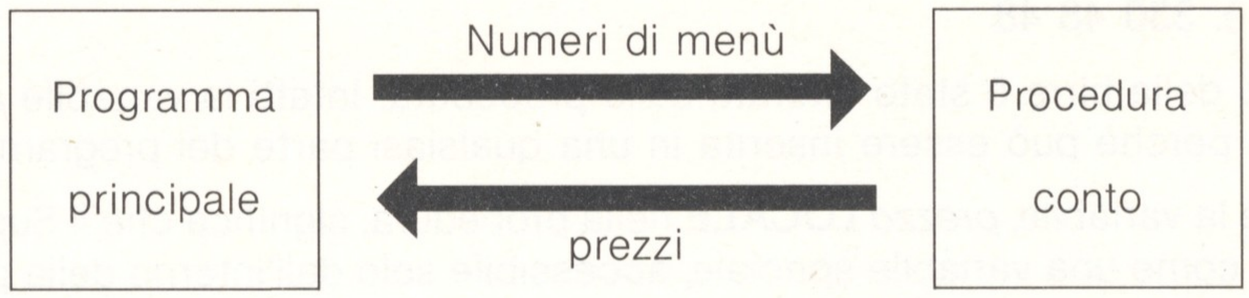
Usare i parametri variabili, *costo__1* e *costo__2* per ricevere i valori della variabile prezzo dalla procedura. Calcolare il conto totale e stamparlo.

PROGRAMMA

```
100 REMark parametro variabile
110 LET num = 70
120 conto 3, costo__1
130 conto 4, costo__2
140 LET conto = num + costo__1 + costo__2
150 PRINT conto
160 DEFine PROCedure conto (num, prezzo)
170    IF num < = 3 THEN LET prezzo = 300 + 10 * num
180    IF num > = 4 THEN LET prezzo = 12 * num
190 END DEFine
```

l'output è: **448**

I parametri *num e prezzo* sono entrambi locali. La seguente figura mostra come le informazioni passano dal programma principale alla procedura e viceversa.



FUNZIONI

È già stato esaminato il compito di una funzione di sistema. Ad esempio la funzione:

```
SQRT (9)
```

calcola il valore 3, che è la radice quadrata di 9. Si dice che la funzione ritorna il valore 3. Una funzione, come una procedura, può avere uno o più parametri, ma a differenza di una procedura, la funzione ritorna un solo valore. Scrivere:

```
PRINT 2 * SQRT (9)
```

I parametri non devono essere necessariamente numerici.

```
LEN ("stringa")
```

ritorna il valore numerico 7.

ESEMPIO

Riscrivere il programma precedente sostituendo la procedura *conto* con la funzione *prezzo*.

Il valore di ritorno della funzione è definito dall'istruzione **RETURN**:

PROGRAMMA

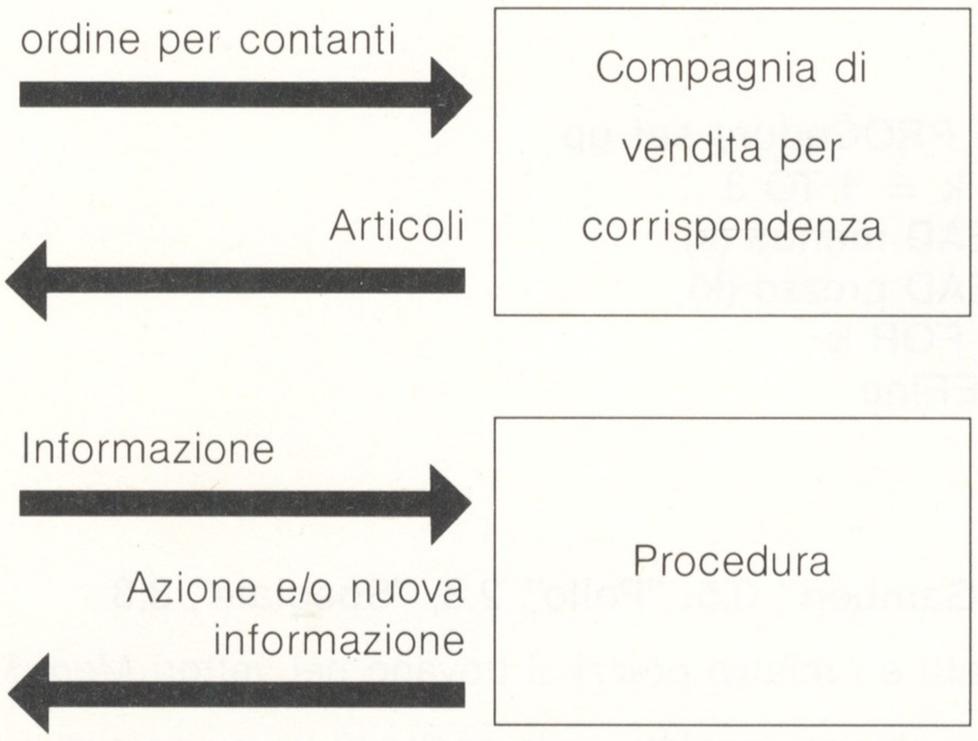
```
100 REMark funzione con RETurn
110 LET num = 70
120 LET conto = num + prezzo (3) + prezzo (4)
130 PRINT conto
140 DEFine FUNction prezzo (num)
150    IF num < = 3 THEN RETurn 300 + 10 * num
160    IF num > = 4 THEN RETurn 12 * num
170 END DEFine
```

l'output è **448**

PROCEDURE

Una procedura può essere paragonata ad una "scatola nera" che riceve informazioni dall'esterno e compie determinate operazioni come, ad esempio, restituire le informazioni all'esterno. L'esterno, in questo caso, è il programma principale.

Il termine "scatola nera" implica che i meccanismi interni non hanno alcuna importanza. È necessario pensare solo a ciò che entra e a ciò che esce. Quando una procedura utilizza una variabile, e ne modifica il valore nel corso delle operazioni, può anche modificare una variabile del programma principale con lo stesso nome. Pensiamo ad una ditta di vendita per corrispondenza. Se la ditta riceve un ordine ed il denaro corrispondente spedisce i vari articoli. Allo stesso modo un'informazione è mandata ad una procedura e la procedura ritorna un'azione e/o una nuova informazione.



Non si vuole che la ditta di vendita si serva dei nomi e degli indirizzi per altri scopi. Allo stesso modo, non si vuole che una procedura provochi cambiamenti di valore alle variabili usate nel programma principale.

La seconda ragione per usare le procedure è di rendere un programma modulare. Anziché avere una lunga sequenza di istruzioni, è possibile dividere il programma in parti più piccole. Queste sono le procedure, ognuna abbastanza ridotta da essere capita e controllata facilmente.

Il terzo scopo è di evitare di scrivere la stessa sequenza di istruzioni più volte. È più facile scrivere una procedura e se necessario richiamarla tutte le volte che si vuole. Le funzioni e le procedure scritte per un programma possono essere usate direttamente in altri programmi. È anzi possibile creare una biblioteca di procedure e funzioni.

Un altro esempio di come le procedure rendono un programma modulare è il seguente:

ESEMPIO

Al self service di Chan è stato fatto un ordine per sei piatti, e il menù è:

numero	piatto	prezzo
1	Gamberi	3.50
2	Pollo	2.80
3	Speciale	3.30

Scrivere le procedure che eseguono le seguenti operazioni:

1. Formare due vettori di tre elementi che indicano il menù, piatti e prezzi. Utilizzare un'istruzione DATA.
2. Simulare un ordine di sei piatti, in modo casuale, usando una procedura, *Scelta*, e stilare un elenco del numero di volte in cui ogni piatto è scelto.
3. Passare i tre numeri alla procedura, *barman*, che ritorna il costo dell'ordine al programma principale usando il parametro *costo*. La procedura *barman* chiama due altre procedure, *calcolo* e *cuoco*, che calcolano il costo e simulano la cottura di un piatto.
4. La procedura, *cuoco*, stampa semplicemente il numero richiesto e il nome di ogni piatto.

Il programma principale chiama le procedure quando necessario, riceve il costo totale dalla procedura *barman*, aggiunge il 10% di mancia, e stampa l'ammontare del conto totale.

DISEGNO

Il seguente programma illustra il passaggio dei parametri fra procedura e programma principale.

```
100 REMark PROCedures
110 RESTORE 490
120 DIM menu$ (3, 7), prezzo (3), piatto (3)
130 REMark *** PROGRAMMA ***
140 LET mancia = 0.1
150 set_up
—
—
210 DEFine PROCedure set up
220    FOR k = 1 TO 3
230       READ menu$ (k)
240       READ prezzo (k)
250    END FOR k
260 END DEFine
—
—
—
490 DATA "Gamberi", 3.5, "Pollo", 2.8, "Speciale", 3.3
```

L'elenco dei piatti e i relativi prezzi si trovano nei vettori *Menu$ e Prezzo*.

La prossima operazione consiste nella scelta di un numero di piatti per ognuno dei sei clienti. I numeri di ogni piatto richiesto sono memorizzati nel vettore *piatto*.

```
160 scegli piatto
—
—
270 DEFine PROCedure scegli (piatto)
280    FOR i = 1 TO 6
290       LET numero = RND (1 TO 3)
300       LET piatto (numero) = piatto (numero) + 1
310    END FOR i
320 END DEFine
```

Il parametro formale piatto è locale nella procedura.

I tre valori sono restituiti al vettore *Piatto* che ha carattere **GLOBALE**. Questi valori sono quindi passati alla procedura *barman*.

```
170 barman piatto, conto
—
—
—
330 DEFine PROCedure barman (piatto, costo)
340    calcolo piatto, costo
350    cuoco piatto
360 END DEFine
```

Barman passa il numero di ogni piatto alla procedura *calcolo*, la quale calcola il costo e lo restituisce.

```
370 DEFine PROCedure calcolo (piatto, totale)
380    LET totale = 0
390    FOR k = 1 TO 3
400       LET totale = totale + piatto (k) * prezzo (k)
410    END FOR k
420 END DEFine
```

Barman passa ancora le informazioni a *cuoco* che stampa semplicemente il numero richiesto per ogni piatto del menù.

```
430 DEFine PROCedure cuoco (piatto)
440    FOR c = 1 TO 3
450       PRINT ! piatto (c) ! nota$ (c) !
460    END FOR c
470 END DEFine
```

Ancora una volta, il vettore, *piatto*, nella procedura *cuoco*, è locale. Riceve l'informazione che la procedura utilizza nella sua istruzione **PRINT**

Il programma completo è così:

```
100 REMark Procedure
110 RESTORE 490
120 DIM menu$ (3, 7), prezzo (3), piatto (3)
130 REMark *** PROGRAMMA ***
140 LET mancia = 0.1
150 set_up
160 scelta piatto
170 barman piatto, conto
180 LET conto = conto + mancia * conto
190 PRINT "il costo totale è"; conto
200 REMark *** DEFINIZIONI DI PROCEDURA ***
210 DEFine PROCedure set_up
220    FOR k = 1 TO 3
230       READ menu$ (k)
240       READ prezzo (k)
250    END FOR k
260 END DEFine
270 DEFine PROCedure scelta (piatto)
280    FOR i = 1 TO 6
290       LET numero = RND (1 TO 3)
300       LET piatto (numero) = piatto (numero) + 1
310    END FOR i
320 END DEFine
330 DEFine PROCedure barman (piatto, costo)
340    calcolo, piatto, costo
350    cuoco, piatto
360 END DEFine
370 DEFine PROCedure calcolo (piatto, totale)
380    LET totale = 0
390    FOR k = 1 TO 3
400       LET totale = totale + piatto (k) * prezzo (k)
410    END FOR k
420 END DEFine
430 DEFine PROCedure cuoco (piatto)
440    FOR c = 1 TO 3
450       PRINT ! piatto (c) ! menu$ (c)
460    END FOR c
470 END DEFine
480 REMark *** DATI DEL PROGRAMMA ***
490 DATA "Gamberi", 3.5, "Pollo", 2.8, "Speciale", 3.3
```

Output

L'output dipende dalla scelta casuale dei piatti ed il seguente è un esempio.

```
3 gamberi
1 pollo
2 speciali

Il Costo totale è: £ 20.40
```

COMMENTO

Naturalmente l'uso di procedure e parametri in un programma così semplice non è strettamente necessario, ma se il programma fosse molto più complesso le procedure permetterebbero una costruzione modulare con verifiche ad ogni livello.

Il seguente esempio illustra l'uso delle funzioni.

Nell'esempio precedente le procedure *"barman"* e *"calcolo"* restituiscono un solo valore. Riscrivere le procedure come funzioni e indicare i cambiamenti necessari.

```
DEFine FUNction barman (piatto)
   cuoco piatto
   RETurn calcolo (piatto)
END DEFine

DEFine FUNction calcolo (piatto)
LET totale = 0
FOR k = 1 TO 3
LET totale = totale + piatto (k) * prezzo (k)
END FOR k
RETurn totale
END DEFine
```

Il programma funziona come il precedente. Tuttavia i parametri sono in numero minore. Questo perché i nomi delle funzioni sono utilizzati come parametri che restituiscono l'informazione alla funzione chiamata.

ESEMPIO

Tutte le variabili usate come parametri formali nelle procedure o nelle funzioni sono automaticamente assunte come variabili locali. Tuttavia vi sono delle variabili usate in alcune procedure che non solo locali. È quindi necessario inserire istruzioni per renderle tali. Le istruzioni sono:

LOCAL k
LOCAL 1, num

PARAMETRI SENZA TIPO

Non è necessario specificare che tipo di variabile deve essere un parametro formale. Tuttavia è preferibile che un parametro che deve sostituire dei numeri sia di tipo numerico ed uno che deve sostituire caratteri alfabetici sia alfabetico. Comunque è sempre possibile, in un programma, ignorare questo problema.

Il seguente programma ne è la dimostrazione.

PROGRAMMA

```
100 REMark numeri o parole
110 cameriere 2
120 cameriere "polli"
130 DEFine PROCedure cameriere (ordine)
140    PRINT! ordine!
150 END DEFine
```

L'output è: **2 polli**

Il tipo di parametro è determinato solo quando è chiamata la procedura ed è passato il valore reale del parametro.

FUNZIONI DELLE VARIABILI

Esaminando il seguente programma la domanda che si pone, è: quale sarà l'output?

```
110 REMark funzione delle variabili
120 LET numero = 1
130 prova
140 DEFine PROCedure prova
150       LOCAL numero
160       LET numero = 2
170       PRINT numero
180       indovina
190 END DEFine prova
200 DEFine PROCedure indovina
210       PRINT numero
220 END DEFine indovina
```

Naturalmente il primo numero che comparirà sul video sarà 2, ma la variabile *numero* della riga 120 è globale?

La risposta è che il valore 2, attribuito alla variabile *numero* dell'istruzione 160, è passato alla procedura *indovina*.

Una variabile, che è definita locale in una procedura, è locale anche per tutte le procedure chiamate dalla prima. Se la procedura *indovina* fosse chiamata dal programma principale, la variabile *numero* avrebbe lo stesso valore che nel programma principale. Le implicazioni possono sembrare strane, ma sono logiche:

1 — La variabile *numero* nella riga 120 è *globale*.

2 — La variabile *numero* nella procedura *prova* è locale.

3 — La variabile *numero* nella procedura *indovina* appartiene alla parte di programma che ha richiamato la procedura per ultimo.

Le idee ora esposte sono i concetti basilari per usare funzioni e procedure.

Le funzioni e procedure sono strumenti molto potenti del Superbasic ed, anche se il loro utilizzo non è di comprensione immediata, è consigliabile sfruttarle non appena possibile.

PROBLEMI SUL CAPITOLO 15

1. Sei impiegati indicati con il loro cognome devono pagare una tassa particolare sul fondo pensione, espressa come percentuale dello stipendio. I seguenti dati rappresentano gli stipendi e la tassa sul fondo pensione dei sei impiegati.

Rossi	13800	6.25
Bianchi	8700	6.00
Verdi	10300	6.25
Neri	15000	7.00
Biondi	6200	6.00
Bruni	5100	5.75

   Scrivere delle procedure per:

   inserire i dati in vettori
   calcolare il contributo attuale del fondo pensioni
   visualizzare l'elenco dei nomi e calcolare i contributi.

   Unire le procedure in un unico programma chiamandole in sequenza.

2. Scrivere una funzione con due argomenti *"limite"* e *"colpi"*. La funzione deve restituire un numero casuale compreso fra i limiti che non sia il valore di colpi.

   Utilizzare la funzione all'interno di un programma per scegliere un colore casuale di **PAPER** e disegnare cerchi in colori casuali di **INK**, in modo che nessuno abbia il colore di **PAPER**.

3. Riscrivere la soluzione del problema 1 utilizzando una funzione, pensione, che abbia come argomenti stipendio e tassa, e ritorni il valore del contributo al fondo pensioni. Utilizzare due procedure: una per inserire i dati e l'altra per stampare i risultati.

4. Scrivere:

   una procedura che crei un mazzo di carte.
   una procedura che mescoli le carte.
   una funzione che abbia come argomento un numero e ritorni un valore di stringa descrivendo le carte.
   una procedura che crei e visualizzi quattro mani di poker di cinque carte ciascuna.
   un programma principale che chiami le precedenti procedure.

CAPITOLO 16
ALCUNE TECNICHE

Nel presente capitolo sono riportate alcune applicazioni delle idee esaminate in precedenza.

SIMULAZIONE DI CARTE DA GIOCO

È facile "memorizzare" e "manipolare" un mazzo di carte da gioco, rappresentandole con i numeri da 1 a 52. È così possibile convertire un numero nella carta equivalente. Supponiamo che sia visualizzato il numero 29, e supponiamo che:

le carte da 1 a 13 rappresentano il seme cuori
le carte da 14 a 26 rappresentano il seme quadri
le carte da 27 a 39 rappresentano il seme picche
le carte da 40 a 52 rappresentano il seme fiori

il numero 29 è quindi una carta di picche. Esiste un'istruzione che permette, assegnato un numero fra 1 e 52, di determinare il seme che il numero rappresenta.

L'istruzione:

```
LET seme = (carta - 1) DIV 13
```

crea un valore compreso tra 0 e 3 che può essere usato per identificare il seme corrispondente. Il valore può essere ridotto nell'intervallo da 1 a 13 scrivendo:

```
LET valore = carta MOD 13
IF valore = 0 THEN LET valore = 13
```

PROGRAMMA

I numeri dall'uno al tredici possono essere chiamati come: Asso, 2, 3, Fante, Donna, Re, o se si preferisce con frasi tipo: "due di cuori"... ecc. Il seguente programma stampa il nome della carta corrispondente al numero richiesto.

```
100 REMark carte
110 DIM nome_seme$ (4, 8), val_carta (13, 5)
120 LET f$ = "di"
130 set_up
140 REPeat carte
150   INPUT "inserire un numero di carta 1-52:" ! carta
160   IF carta < 1 OR carta > 13 THEN EXIT carte
170   LET seme = (carta-1) DIV 13
180   LET valore = carta MOD 13
190   IF valore = 0 THEN LET valore = 13
200   PRINT val_carta$ (valore) ! f$ ! nome_seme$ (seme)
210 END REPeat carte
220 DEFine PROCedure set_up
230   FOR s = 1 TO 4 : READ nome_seme$ (s): END FOR s
240   FOR v = 1 TO 13 : READ val_carta$ (v): END FOR v
250 END DEFine
260 DATA "cuori", "quadri", "picche", "fiori"
270 DATA "asso", "due", "tre", "quattro", "cinque", "sei", "sette"
280 DATA "otto", "nove", "dieci", "fante", "donna", "re"
```

INPUT E OUTPUT

```
13
re di cuori
49
dieci di fiori
27
asso di picche
```

COMMENTO

Notare l'uso delle istruzioni **DATA** per creare un file permanente di dati utilizzabili dal programma. I dati variabili dopo ogni esecuzione del programma sono inseriti con istruzioni **INPUT**. Se i dati di input sono noti prima dell'esecuzione del programma è consigliabile usare l'istruzione **READ** e più istruzioni **DATA**.

FILE SEQUENZIALI

File di numeri

Il seguente programma crea un file contenente cento numeri.

```
100 REMark file di numeri
110 OPEN_NEW #6, mdv1_numeri
120 FOR num = 1 TO 100
130    PRINT #6, num
140 END FOR num
150 CLOSE #6
```

Dopo l'esecuzione è possibile controllare l'esistenza del file "numeri", con questa istruzione:

```
DIR mdv1_numeri
```

È anche possibile visualizzare sullo schermo il contenuto del file:

```
COPY mdv_1 numeri TO scr
```

Il seguente programma legge il file e stampa i record sullo schermo:

```
100 REMark leggi file
110 OPEN_IN #6, mdv1_numeri
120 FOR num = 1 TO 100
130    INPUT #6, nota
140    PRINT ! nota !
150 END FOR num
160 CLOSE #6
```

File di caratteri

In modo analogo, i seguenti programmi creano un file di cento lettere casuali e lo rileggono.

```
100 REMark file di lettere
110 OPEN_NEW #6, mdv1_chfile
120 FOR num = 1 TO 100
130    LET ch$ = CHR$ (RND(65 TO 90))
140    PRINT #6, ch$
150 END FOR num
160 CLOSE #6
```

```
100 REMark lettere
110 OPEN_IN #6, mdv1_chfile
120 FOR num = 1 TO 100
130    INPUT #6, nota$
140    PRINT ! nota$ !
150 END FOR num
160 CLOSE #6
```

CREARE UN FILE DI DATI

Supponiamo di voler creare un semplice archivio contenente numeri di telefono.

Paolo	678462
Giorgio	896487
Carlo	249386
Andrea	584621
Mauro	482349
Gigi	438975
Cristina	392938

Il programma che crea il file richiesto è:

```
100 REMark numeri telefono
110 OPEN_NEW #6, mdv1_telefono
120 FOR record = 1 TO 7
130    INPUT nome$, num$
140    PRINT #6, nome$, num$
150 END FOR record
160 CLOSE #6
```

Scrivere l'istruzione **RUN** ed inserire un nome ed un numero, premere il tasto **ENTER** e ripetere l'operazione sette volte.

I dati sono memorizzati internamente fino a quando il sistema non è pronto a trasferirli nel Microdrive. Il programma accede al Microdrive una sola volta.

COPIARE UN FILE

Quando un file è stato creato, è consigliabile tenere almeno una copia di back up (sicurezza). Per eseguire l'operazione è sufficiente scrivere:

```
COPY mdv1_ telefono TO mdv2_telefono
```

LEGGERE UN FILE

Per essere certi che il file esista e per visualizzarlo sullo schermo è necessario il seguente comando:

```
COPY mdv2_telefono TO scr
```

Sullo schermo non è inserito alcuno spazio tra i nomi ed i numeri. L'output è:

```
Paolo 678462
Giorgio 896487
Carlo 249386
Andrea 584621
Mauro 482349
Gigi 438975
Cristina 392938
```

Con il seguente programma si ottiene una miglior presentazione dei dati:

```
100 REMark leggi numeri telefono
110 OPEN_IN #5, mdv1_telefono
120 FOR record = 1 TO 7
130    INPUT #5, rec$
140    PRINT, rec$
150 END FOR record
160 CLOSE #5
```

Le informazioni sono stampate come prima, ma questa volta sono inserite nella variabile rec$ prima di essere visualizzate. Si ha quindi la possibilità di stamparle nella forma desiderata.

ORDINAMENTI

Nella modalità di schermo a bassa risoluzione sono disponibili i seguenti colori. (ordinati secondo il numero di codice 0-7).

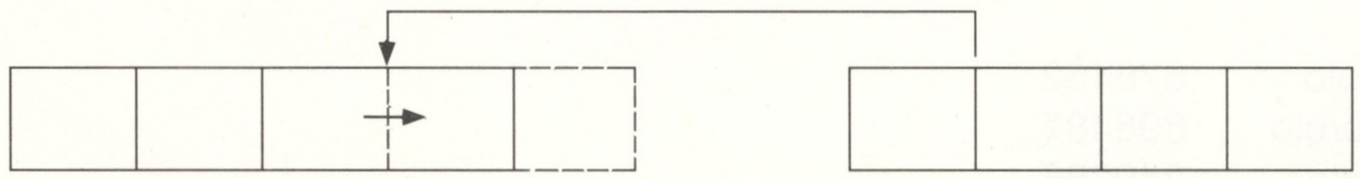
nero blue rosso magenta verde ciano giallo bianco

ESEMPIO

Scrivere un programma per ordinare i nomi dei colori alfabeticamente.

METODO

Inseriamo gli otto nomi in un vettore, colore$, che dividiamo in due parti:



PARTE ORDINATA — PARTE NON ORDINATA

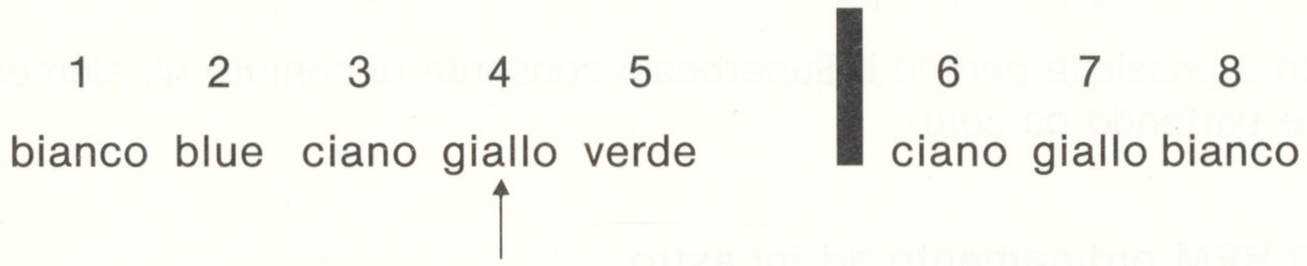
Il metodo consiste nel prendere il primo elemento a sinistra della parte non ordinata e confrontarlo con tutti gli elementi successivi della parte ordinata, fino a trovare la posizione corretta.

Supponiamo di avere ordinato quattro elementi, e esaminiamo ora il verde, l'elemento più a sinistra della parte non ordinata.

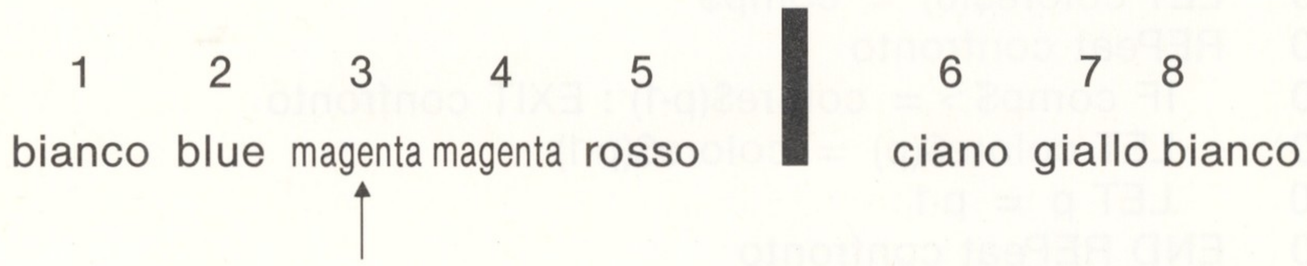
1	2	3	4		5	6	7	8
bianco	blue	ciano	giallo		verde	rosso	nero	magenta
PARTE ORDINATA					PARTE NON ORDINATA			

1. Inseriamo il verde nella variabile *comp$*, e creiamo una variabile *p* con valore 5.
2. La variabile *p* indica la posizione in cui va inserito il verde. Quando il verde si muove verso sinistra, il valore di *p* è diminuito di una unità.
3. Confrontiamo il verde con il giallo. Se il verde è maggiore o uguale al giallo, allora il verde rimane nella posizione attuale.

   Diversamente trascriviamo il giallo nella posizione 5 e diminuiamo il valore di *p*.



4. Ripetiamo l'operazione confrontando il ciano con il verde



I punti 1,2,3,4 devono essere ripetuti finché non è stato esaminato l'ultimo colore della parte sinistra.

ANALISI DEI PROBLEMI

1. Memorizzare i nomi dei colori in un vettore *colore$ (8)* e utilizzare le variabili:

   comp$ per memorizzare il colore corrente da confrontare
   p come indice di posizione

2. Un'iterazione **FOR** esamina le posizioni 2 e 8 alternativamente.
3. Una struttura **REPeat** permette i confronti fino a quando è indicato dove deve inserirsi il valore di *comp$*.

   REPeat *confronto*
   IF *comp$* non deve andare più a sinistra **EXIT**
   Trascrivere un colore nella posizione alla sua destra e diminuire *p*
   END REPeat *confronto*

4. Dopo essere uscito **EXIT** dal ciclo **REPeat** il nome memorizzato in *comp$* è inserito nella posizione *p* e il ciclo continua.

DISEGNO DEL PROGRAMMA

1. Dichiarare il vettore *colore$*
2. Leggere i colori nel vettore
3. **FOR** num = 2 **TO** 8
   LET p = num
   LET comp$ = colore$ (p)
   REPeat confronto
   IF comp$ [2] = colore$ (p-1) : **EXIT** confronto
   LET colore$(p) = colore$ (p-1)
   LET p = p-1
   END REPeat confronto
   LET colore$ (p) = comp$
   END FOR num
4. **PRINT** vettore colore$ ordinato
5. **DATA**

Ulteriori verifiche segnalano un errore. Questo avviene quando nei dati il primo elemento non è nella sua posizione corretta. Un semplice cambio di dati rivela il problema:

rosso nero blue magenta verde ciano giallo bianco

Confrontiamo il nero con il rosso e diminuiamo p di uno. Ora confrontiamo il nero con l'elemento colore$(p-1) che è colore$(0), il quale non esiste.

Questo tipo di problema si presenta spesso. Una possibile soluzione è di inserire un controllo alla fine del vettore. Appena prima della struttura REPeat è necessario scrivere:

LET colore$(0) = comp$

Questo è possibile perché il Superbasic consente di contare gli elementi di un vettore partendo da zero.

MODIFICA DI UN PROGRAMMA

```
100 REM ordinamento ad incastro
110 DIM colore$ (8, 7)
120 FOR num = 1 TO 8 : READ colore$(num) : END FOR num
130 FOR num = 2 TO 8
140    LET p = num
150    LET comp$ = colore$(p)
160    LET colore$(0) = comp$
170    REPeat confronto
180       IF comp$>= colore$(p-1) : EXIT confronto
190       LET colore$(p) = colore$(p-1)
200       LET p = p-1
210    END REPeat confronto
220    LET colore$(p) = conp$
230 END FOR num
240 PRINT "ORDINATO...." ! colore$
250 DATA "nero", "blue", "magenta", "rosso"
260 DATA "verde", "ciano", "giallo", "bianco"
```

COMMENTO

1. Il programma è eseguito regolarmente. È possibile verificarlo con dati più complessi.

   AAAAAAA
   BABABAB
   ABABABA
   BCDEFGH
   GFEDCBA

2. Un ordinamento del tipo precedente non è molto veloce, ma può essere utilizzato nel caso si aggiungano elementi ad una serie già ordinata.

ORDINARE UN FILE SU CARTUCCIA

Per ordinare, ad esempio, alfabeticamente il file "telefono" è necessario leggerlo in un vettore, ordinarlo e quindi creare il nuovo file in cui i nomi sono in ordine alfabetico.

Non è buona abitudine cancellare un file prima che la sua nuova versione sia verificata. È utile copiare, per sicurezza, il file precedentemente utilizzato in un altro file con nome differente. La sequenza logica è la seguente:

1. Copiare il file "telefono", in "telefono__temp"
2. Leggere il file "telefono", in un vettore
3. Ordinare il vettore
4. Pausa per verificare che tutto sia in ordine
5. Cancellare il file "telefono"
6. Creare il nuovo file "telefono"

Il nuovo file può essere controllato utilizzando l'istruzione:

COPY mdv1__telefono TO scr

Ogni ulteriore controllo può essere eseguito in uno dei seguenti modi:

```
DELETE mdv2_telefono
COPY mdv1_telefono TO mdv2_telefono
COPY mdv1_telefono To scr
DELETE mdv1_telefono_temp
```

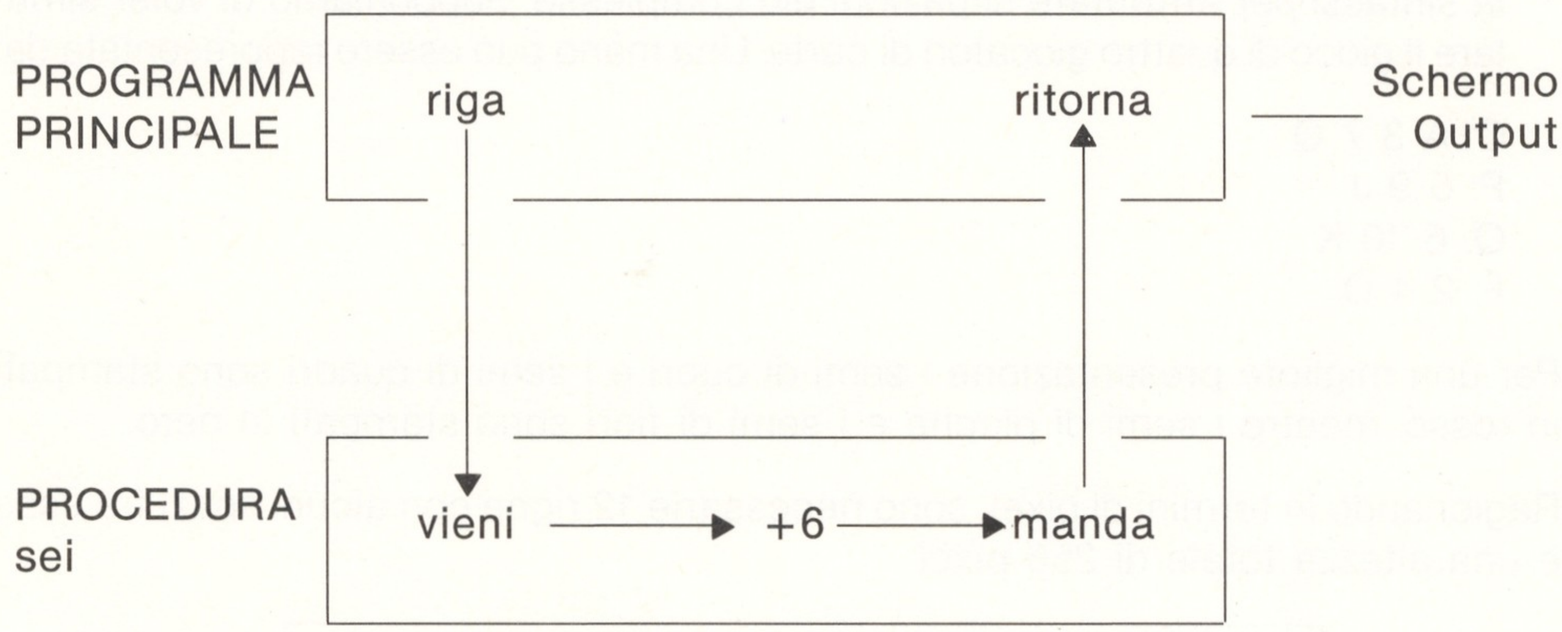
Le precedenti operazioni completano il processo di sostituzione di un file ordinato con uno originariamente non ordinato.

PARAMETRI VETTORE

Nel seguente programma è illustrato lo scambio di vettori tra il programma principale e le procedure. Il flusso è in entrambe le direzioni.

Nella riga 130 il vettore, *riga*, contenente i numeri 1,2,3, è passato alla procedura, *sei*. Il parametro *vieni*, riceve il valore entrante e la procedura aggiunge il valore sei ad ogni elemento. Il parametro vettore *manda* contiene ora i numeri 7,8,9.

Questi numeri sono restituiti al programma principale e diventano valori del vettore *ritorna*.



PROGRAMMA

```
100 REMark Cambio di vettori
110 DIM riga (3), ritorna (3)
120 FOR k = 1 TO 3 : LET riga (k) = k : NEXT k
130 sei riga, ritorna
140 FOR k = 1 TO 3 : PRINT ! ritorna (k) ! : NEXT k
150 DEFine PROCedure sei (ritorna manda)
160    FOR k = 1 TO 3 : LET manda (k) = vieni (k)+6 : NEXT k
170 END DEFine
```

L'output è:

```
7 8 9
```

La seguente procedura riceve un vettore contenente i dati da ordinare. L'elemento zero contiene il numero di iterazioni. Non ha importanza se il vettore è numerico o alfabetico.

È interessante notare che l'elemento del vettore *vieni* (0), ha due funzioni:

1. contiene il numero di informazioni da ordinare
2. contiene l'indice dell'elemento in esame

```
100 DEFine PROCedure "ordina" (vieni, manda)
110    LET num = vieni (0)
120    FOR volte = 2 TO num
130       LET p = volte
140       LET vieni = 0 = vieni (p)
150       REPeat confronto
160          IF vieni (0)> = vieni(p-1) : EXIT confronto
170          LET vieni (p) = vieni (p-1)
180          LET p = p-1
190       END REPeat confronto
200       LET vieni (p) = vieni (p-1)
210    END FOR volte
220    FOR k = 1 TO 7 : manda(k)=vieni(k) : NEXT k
230 END DEFine
```

Le seguenti righe aggiuntive verificano il buon funzionamento della procedura *ordina* .

```
10 REMark Verifica "ordina"
20 DIM riga$ (7, 3), ritorna$ (7, 3)
30 LET riga$ (0) = 7
40 FOR k = 1 TO 7 : READ riga$(k)
50 Ordina riga$, ritorna$
60 PRINT ! ritorna$ !
70 DATA "EEL", "DOG", "ANT", "GNU", "CAT", "BUG", "FOX"
```

L'output è:

```
ANT BUG CAT DOG EEL FOX GNU
```

COMMENTO

Il programma dimostra come si possono manipolare vettori in Superbasic. Dopo che la procedura è eseguita è possibile utilizzare l'istruzione **MERGE mdv1_ordina** per aggiungerla ad un programma residente nella memoria principale.

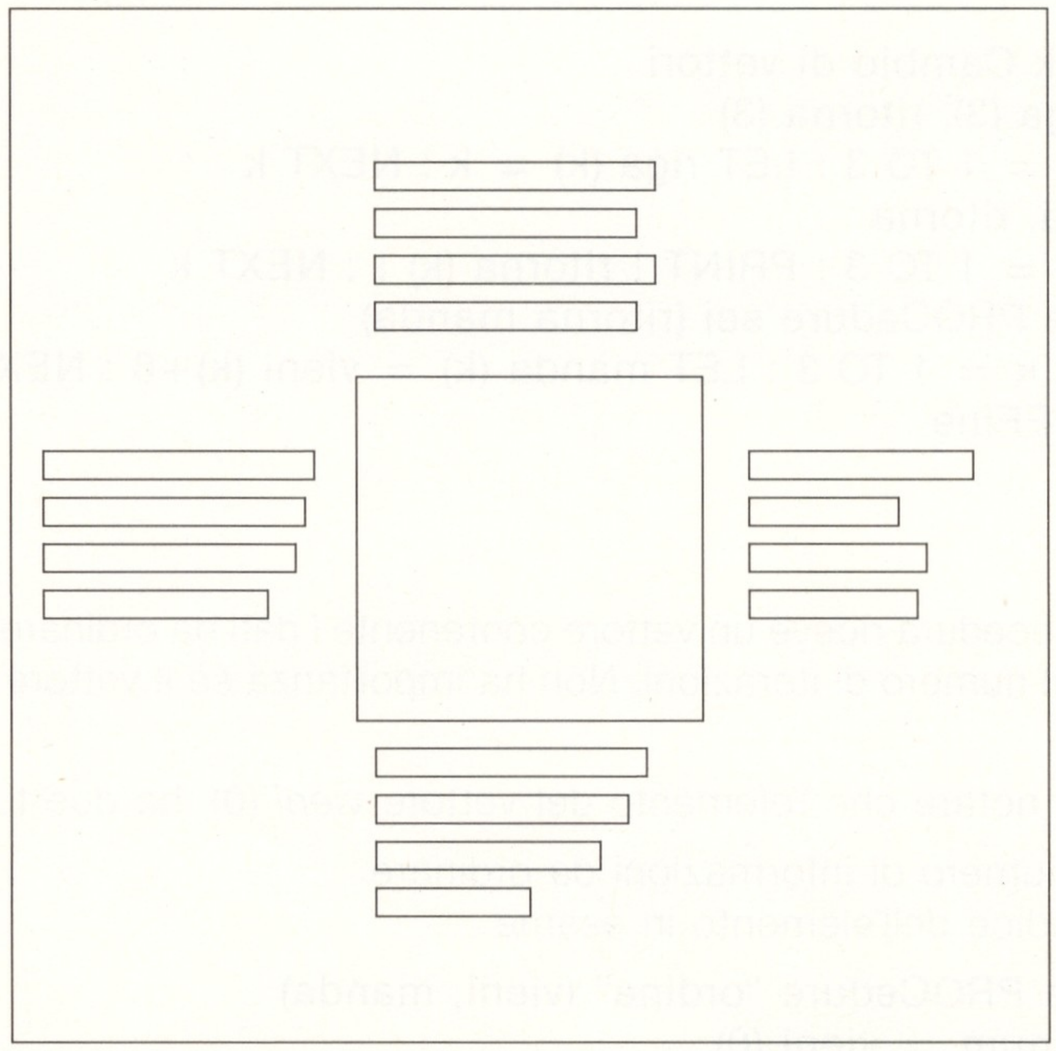
Si ha ora una conoscenza sufficiente delle tecniche di programmazione e della sintassi per affrontare situazioni più complesse. Supponiamo di voler simulare il gioco di quattro giocatori di carte. Una mano può essere rappresentata da:

```
C: A 3 7 Q
P: 5 9 J
Q: 6 10 K
F: 2 4 Q
```

Per una migliore presentazione i semi di cuori e i semi di quadri sono stampati in rosso, mentre i semi di picche e i semi di fiori sono stampati in nero.

METODO

Ragionando in termini di pixel, sono necessarie 12 righe con alcuni spazi fra esse e una altezza totale di 256 pixel.



È utile ricordare che l'altezza possibile per i caratteri è 10 o 20 pixel.

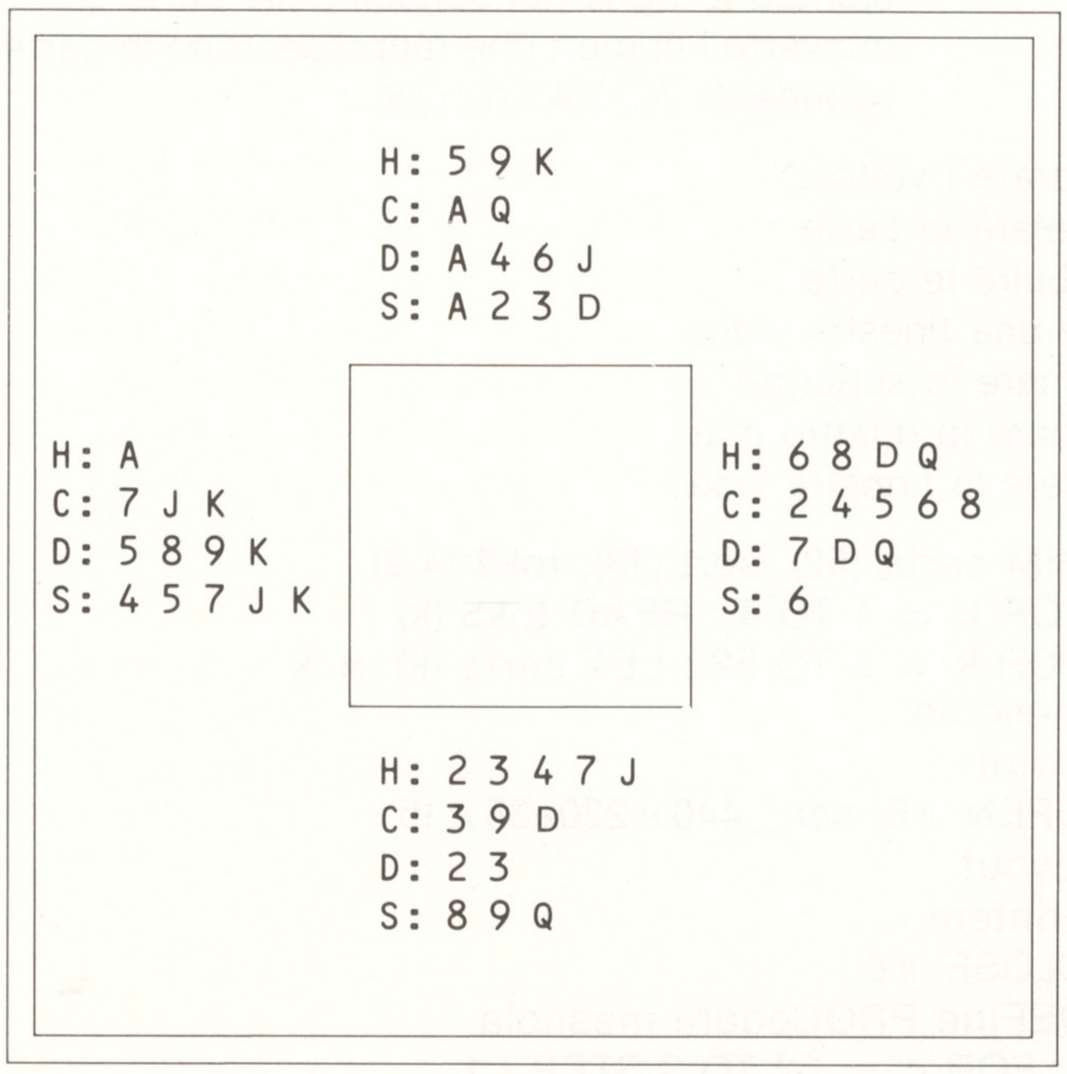
Deve essere calcolato il numero di caratteri occupati sullo schermo. Se è usato il simbolo "D" per indicare il numero 10, tutti i valori delle carte possono essere rappresentati come singoli caratteri. Si supponga per semplicità, che siano permesse al massimo 8 carte dello stesso seme. Il numero di caratteri in orizzontale richiesti sullo schermo è:

mano ovest + grandezza tavolo + mano est

e lasciando uno spazio fra i caratteri diventa:

20 + grandezza tavolo + 20

La decisione ora dipende dalla modalità di schermo utilizzata. Nel Mode 256 il problema è risolto facilmente, ma per il momento utilizziamo il Mode 512. La dimensione più piccola per i caratteri è di sei pixel, che danno come totale 240 pixel + la grandezza del tavolo (circa metà di 240).



FINESTRE 440 × 220 a 35,15
verde con bordi neri di 10 unità

TAVOLO 100 × 60 a 150,60
scacchiera con stipple rossi e verdi

MANI posizione iniziale del cursore:

nord	150,60
est	260,60
sud	150,130
ovest	30,60

DIMENSIONI CARATTERE standard nel Mode 512
NUMERO DI PIXEL tra le linee è 12

COLORE CARATTERE bianco
COLORI CARTE rosso per cuori e quadri
nero per picche e fiori

VARIABILI

carta(52)	memorizza i numeri delle carte
sorte(13)	distribuisce le carte
semi$(4,2)	memorizza i semi C. Q. P. F.
k, m, c, m, h	variabili di lavoro
ran	posizione casuale per cambio carte
temp	variabile temporanea
item	carta da inserire nel vettore sorte$
dart	indice per determinare la posizione nel vettore sorte
comp	variabile per memorizzare il numero di carte del vettore sorte
inc	incremento nel numero di pixel
seat	posizione corrente
ac, dn	coordinate posizione cursore
row	riga corrente
lin$	variabile per memorizzare una riga di caratteri
max	numero di carta più alto
p	posizione della carta
n	numero corrente di carta

PROCEDURE

mescola mescola le carte
split distribuisce le carte e richiama la procedura sortem
sortem ordina 13 carte in ordine ascendente
layout disegna il colore di fondo, i bordi ed il tavolo da gioco
printem stampa le righe dei simboli delle carte
getline converte i numeri che rappresentano le carte nei simboli corrispondenti: A,2,3,4,5,6,J,Q,K

SCHEMA DEL PROGRAMMA

1. Dichiarare i vettori,
2. Mescolare le carte
3. Distribuire le carte
4. Aprire una finestra video
5. Disegnare lo schermo
6. Stampare le quattro mani
7. Chiudere la finestra video

PROGRAMMA

```
100 DIM carta (52), sort (13), tok$ (4,2)
110 FOR k = 1 TO 4 : READ tok$ (k)
120 FOR k = 1 TO 52 : LET carta (k) = k
130 mescola
140 dividi
150 OPEN #6, scr_440x220a35x15
160 layout
170 printem
180 CLOSE #6
190 DEFine PROCedure mescola
200   FOR c = 52 TO 3 STEP - 1
210     LET ran = RND (1 TO c-1)
220     LET temp = card(c)
230     LET card(c) = card(ran)
240     LET card(ran) = temp
250   END FOR c
260 END DEFine
270 DEfine PROCedure split
280   FOR h = 1 TOP 4
290     FOR c = 1 TO 13
300       LET sort(c) = card(h-1)*13+c)
310     END FOR c
320     sortem
330     FOR c = 1 TO 13
340       LET card(h-1)*13+c) = sort(c)
350     END FOR c
360   END FOR h
370 END DEFine
380 DEFine PROCedure sortem
390   FOR item = 2 TO 13
400     LET dart = item
410     LET comp = sort (dart)
420     LET sort(0) = comp
430     REPeat compare
440       IF comp > = sort(dart-1) : EXIT compare
450       LET sort(dart) = sort(dart-1)
460       LET dart = dart-1
470     END REPeat compare
480     LET sort (dart) = comp
490   END FOR item
500 END DEFine
510 DEFine PROCedure layout
520   PAPER #6,4 : CLS #6
530   BORDER #6, 10,0
540   BLOCK #6, 100, 60, 150, 60, 2, 4
550 END DEFine
560 DEFine PROCedure printem
570   LET inc = 12 : INK #6,7
580   LET p = 0
```

```
590    FOR seat = 1 TO 4
600       READ ac, dn
610       FOR riga = 1 TO 4
620          getline
630          CURSOR #6, ac, dn
640          PRINT #6, lin$
650          LET dn = dn + inc
660       END FOR riga
670    END FOR seat
680 END DEFine
690 DEFine PROCedure getline
700    IF riga MOD 2 = 0 THEN STRIP #6,0
710    IF riga MOD 2 = 1 THEN STRIP #6,2
720    LET lin$ = tok$ (riga)
730    LET max = riga$13
740    REPeat one_suit
750       LET p = p+1
760       LET n = card (p)
770       IF n>max THEN p = p-1 : EXIT one_suit
780       LET n = n MOD 13
790       IF n = 0 THEN n = 13
800       IF n = 1 : LET ch$ = "A"
810       IF n> =2 AND n< =9 : LET ch$ = n
820       IF n = 10 : LET ch$ = "T"
830       IF n = 11 : LET ch$ = "J"
840       IF n = 12 : LET ch$ = "g"
850       IF n = 13 : LET ch$ = "K"
860       LET lin$ = lin$ & " " & ch$
870       IF p = 52 : EXIT one_suit
880       IF card(p) > card(p+1) : EXIT one_suit
890    END REPeat one_suit
900 END DEFine
910 DATA "H:", "C:", "D:", "S:"
920 DATA 150, 10, 260, 60, 150, 130, 30, 60
```

COMMENTO

Il programma utilizza il Mode 256, ma le varie linee dei simboli delle carte possono coprire il *tavolo* o uscire dalla finestra. Un semplice cambiamento nella procedura *getline* evita incovenienti.

da **860 LET lin$ = lin$ & " " & ch$**

a

860 LET lin$ = lin$ & ch$

Lo spazio fra i caratteri scompare, ma la maggior dimensione dei caratteri permette una migliore lettura. Il programma è eseguito anche usando il sistema grafico.

CONCLUSIONE

Sono state illustrate le principali caratteristiche del Superbasic e la facilità con la quale è possibile risolvere problemi complessi.

Le operazioni eseguibili in Superbasic sono tante, che non è possibile riportarle in un manuale. L'obiettivo del manuale è stato quello di fornire le idee basilari della programmazione per consentire a chiunque di utilizzare il QL ed il Superbasic.

Keywords

sinclair

QL

parole chiave

Il capitolo PAROLE CHIAVE elenca in ordine alfabetico tutte le parole chiave del Superbasic.

Di ogni comando è data una breve descrizione seguita dalla definizione della sintassi e da esempi. Consultare il capitolo GUIDA ALLA PROGRAMMAZIONE per informazioni riguardo alle convenzioni sintattiche.

Di ogni parola chiave è indicato, se esiste, il tipo di operazione a cui si riferisce. Ad esempio l'istruzione **DRAW** è una operazione grafica e ulteriori informazioni possono essere ottenute consultando la voce "grafica" nel capitolo CONCETTI.

A volte è necessario trattare più di un'istruzione per volta. Ad esempio, i comandi **IF, THEN, ELSE, END IF** fanno tutti parte della parola chiave **IF**.

L'indice finale del manuale tenta di contenere tutte le possibili descrizioni di una parola chiave. Ad esempio il comando **CLS** si trova nell'indice sotto le voci **CLS** CANCELLA VIDEO e VIDEO.

©1985 SINCLAIR RESEARCH LIMITED
by Stephen Berry (Sinclair Research Limited)

ABS
funzioni matematiche

ABS calcola il valore assoluto del parametro specificato. Il valore assoluto è sempre un numero positivo o nullo.

sintassi: **ABS** *(espressione__numerica)*

esempi: i. **PRINT ABS (0.5)**
ii. **PRINT ABS (a-b)**

ACOS, ACOT, ASIN, ATAN
funzioni matematiche

ACOS e **ASIN** calcolano rispettivamento l'arcocoseno e l'arcoseno del parametro specificato. **ACOT** e **ATAN** calcolano l'arcocotangente e l'arcotangente del parametro specificato. Non è richiesto nessun limite al parametro. Il valore calcolato da tutte le funzioni è un angolo espresso in radianti. Per convertire i radianti in gradi è sufficiente moltiplicare il valore dell'angolo in radianti per 180/pi.

sintassi: **ACOS** *(espressione__numerica)*
esempi: i. **PRINT ASIN (1)**
ii. **PRINT ACOT (3.675)**
iii. **PRINT ATAN (a-b)**

ADATE

orologio

ADATE permette di regolare l'orologio interno del QL.

sintassi: *secondi = espressione numerica*
ADATE *secondi*

esempi: i. **ADATE 3600** {sposta l'orologio in avanti di un'ora}
ii. **ADATE -60** {sposta l'orologio indietro di un minuto}

ARC
ARC_R

grafica

ARC disegna un arco di cerchio compreso fra due punti specificati, nella finestra associata al canale assegnato o di default. Gli estremi dell'arco sono assegnati usando il sistema di coordinate grafiche.

Con una sola istruzione **ARC** possono essere disegnati più archi.

Gli estremi dell'arco possono essere specificati in coordinate assolute (relative all'origine del sistema di coordinate grafiche) o in coordinate relative (aventi come origine la posizione del cursore). Se non è assegnato alcun valore al punto iniziale, l'arco è disegnato partendo dalla posizione del cursore fino all'estremo finale, con curvatura determinata dall'angolo assegnato.

ARC usa sempre coordinate assolute, mentre **ARC_R** usa coordinate relative. L'origine delle coordinate è la posizione iniziale del cursore.

sintassi: *x = espressione_numerica*
y = espressione_numerica
angolo = espressione_numerica (in radianti)
punto = x, y
parametro_2 = **TO** *punto, angolo* 1
punto **TO** *punto, angolo* 2
parametro_1 = *punto* **TO** *punto, angolo* 1
TO *punto, angolo* 2

ARC *[canale,] parametro_1 *[parametro_2]**

ARC_R *[canale,] parametro_1 *[parametro_2]**

dove: 1 traccia l'arco dal punto iniziale fino al secondo punto con angolo assegnato.

2 traccia l'arco dall'ultimo punto disegnato al punto specificato con angolo assegnato

esempi i. **ARC 15, 10 TO 40, 40, 40 pi/2**
(traccia un arco dal punto 15, 15 a 40, 40 con un angolo di 90 gradi)

ii. **ARC_R 10, 10 TO 55, 45, 0.5**
(traccia un arco cominciando dal punto di coordinate relative (rispetto all'ultimo punto disegnato) 10, 10 fino al punto di coordinate relative (rispetto al punto iniziale dell'arco) 55, 45, con un angolo di 0.5 radianti.

AT
finestre

AT permette di definire la riga e la colonna di stampa in dipendenza dalla dimensione del carattere scelto. AT usa un sistema di coordinate in cui l'origine (riga 0, colonna 0) è il vertice superiore sinistro della finestra associata al canale di stampa.

sintassi: *riga = espressione__numerica*
colonna = espressione__numerica

AT *[canale,] riga, colonna*

esempi: i. **AT 10, 20 : PRINT "riga 10 colonna 20"**

AUTO

AUTO genera automaticamente i numeri di riga di un programma. **AUTO** crea il numero di riga successivo nella sequenza. Se un numero di riga è già esistente, il suo contenuto è visualizzato sulla linea di immissione del programma ENTER verifica la sintassi della riga e permette il suo inserimento nel programma.

AUTO è interrotto premendo:

CTRL	SPAZIO

sintassi: *prima__riga = numero__riga*
incremento = espressione__numerica

AUTO *[prima__riga], [incremento]*

esempi:
i. **AUTO** {numero prima__riga 100, incremento 10}
ii. **AUTO 10, 5** {numero prima__riga 10, incremento 5}
iii. **AUTO, 7** {numero prima__riga 100, incremento 7}

BAUD

comunicazioni

BAUD seleziona la velocità di trasmissione delle porte seriali. La velocità delle porte non può essere assegnata indipendentemente.

sintassi: *velocità = espressione__numerica*

BAUD *velocità*

Il valore dell'espressione numerica deve essere uno dei seguenti:

75
300
600
1200
2400
4800
9600
19200 {solo trasmissione}

Se la velocità di trasmissione selezionata non è tra queste, è segnalato un errore.

esempi: i. **BAUD 9600**
ii. **BAUD velocità**

BEEP

suono

BEEP causa l'emissione di suono dal QL. **BEEP** accetta un numero di parametri variabili, per ottenere vari livelli di controllo del suono prodotto. Occorre quanto meno specificare la durata e il tono. **BEEP** usato senza alcun parametro, sospende l'esecuzione di qualsiasi suono.

sintassi:
durata = espressione__numerica {da - 32768 a 32768}
tono = espressione__numerica {da 0 a 255}
grad__x = espressione__numerica {da - 32768 a 32768}
grad__y = espressione__numerica {da -8 a 7}
replica = espressione__numerica {da 0 a 15}
rumore = espressione__numerica {da 0 a 15}
caso = espressione__numerica {da 0 a 15}

BEEP [*durata, tono*
[, *tono__2, grad__x, grad__y*
[, *replica*
[, *rumore*
[, *caso*]]]]]

Durata definisce la durata del suono in unità di 72 microsecondi. Un valore 0 provoca l'emissione di suono, interrotta soltanto da un altro comando **BEEP**

Tono Determina la tonalità del suono. Un valore di 1 è alto e 255 è basso.

Tono__2
Determina un secondo livello di tonalità.

Grad__x Determina il tempo fra gli intervalli dei toni.

Grad__y Definisce la dimensione di ogni intervallo.
Replica Forza la ripetizione del suono per il numero di volte specificato.
Rumore Determina la quantità di "rumore" aggiunto al suono originale.
Caso Aggiunge casualità all'emissione del suono.

BEEPING

suono

BEEPING è una funzione che dà un valore 0 (falso) se il suono del QL non è attivo, e dà valore 1 (vero) in caso contrario.

sintassi: **BEEPING**

esempi:
```
100 DEFine PROCedure silenzio
110       BEEP
120 END DEFine
130 IF BEEPING THEN silenzio
```

BLOCK

finestre

BLOCK colora un'area di dimensioni e forma assegnate nella finestra associata al canale specificato o di default.

BLOCK usa il sistema di coordinate a pixel.

sintassi: *larghezza = espressione_numerica*
altezza = espressione_numerica
x = espressione_numerica
y = espressione_numerica

BLOCK *[canale,] larghezza, altezza, x, y, colore*

esempi: i. **BLOCK 10, 10, 5, 5, 7**

ii.
```
100 REMark istogramma
110 CSIZE 3, 1
120 PRINT istogramma
130 fondo=100 : dimen=20 : sinistra=10
140 FOR isto=1 TO 10
150     LET colore=RND(O TO 255)
160     LET altezza=RND(2 TO 20)
170     BLOCK dimen, altezza, sinistra+isto*dimen, fondo-altezza
180     BLOCK dimen-2, altezza-2, sinistra+isto* dimen+1, colore
190 END FOR isto
```

Se si usa un apparecchio televisivo l'istruzione 150 deve essere:

```
150 LET colore = RND (0 TO 7)
```

BORDER

finestre

BORDER traccia un bordo attorno alla finestra associata al canale specificato o di default.

Per tutte le operazioni successive, eccetto un'altra istruzione **BORDER**, le dimensioni della finestra sono ridotte per lasciare spazio al bordo. Se è usato un altro comando **BORDER**, la finestra è riportata alle dimensioni originali prima che il nuovo bordo sia aggiunto. In questo modo una sequenza di comandi **BORDER** ha l'effetto di cambiare la dimensione e il colore di un singolo bordo. Non sono tracciati bordi multipli senza un'istruzione specifica.

Se l'istruzione **BORDER** è usata senza specificare il colore, il bordo è trasparente.

sintassi: *larghezza = espressione_numerica*

BORDER *[canale,] dimensione [,colore]*

esempi: i. **BORDER 10, 0, 7**

ii.
```
100 REMark Bordi indistinti
110 FOR spessore = 50 TO 2 STEP -2
120       spessore, RND (O TO 255)
130 END FOR spessore
140 BORDER 50
```

Se si usa un apparecchio televisivo l'istruzione 120 deve essere:

```
120 spessore, RND (1 TO 10)
```

CALL

qdos

È possibile accedere al Superbasic dal linguaggio macchina, usando l'istruzione **CALL**. L'istruzione **CALL** accetta fino a 13 parametri, che sono memorizzati nei registri da D1 a D7 e da A0 a A5.

L'istruzione **CALL** non ritorna alcun dato ad un programma Superbasic.

sintassi: *indirizzo = espressione_numerica*
dati = espressione_numerica
CALL *indirizzo *[dati]**

esempi: i. **CALL 262144, 0, 0, 0**

ii. **CALL 262500, 12, 3, 4, 1212, 6**

attenzione

Il registro A6 non deve essere usato in routine richiamate dal comando **CALL**. Per tornare a Superbasic usare le istruzioni:

```
MOVEQ #0, D0
RTS
```

CHR$

basic

CHR$ è una funzione che ritorna il carattere il cui valore è stato specificato come parametro.
CHR$ è la funzione inversa di **CODE**.

sintassi: **CHR$** *(espressione numerica)*

esempi: i. **PRINT CHR$(27)** {stampa il carattere ESCAPE}
ii. **PRINT CHR$(65)** {stampa la A}

CIRCLE
CIRCLE_R

grafica

CIRCLE traccia un cerchio (o un'ellisse) di raggio e centro assegnati. Il cerchio è disegnato nella finestra associata al canale specificato o di default.

CIRCLE usa il sistema di coordinate grafiche assolute.

CIRCLE_R usa il sistema di coordinate grafiche relative.

Con una sola istruzione **CIRCLE** possono essere disegnati più cerchi o ellissi. Ogni insieme di parametri deve essere separato dagli altri dal punto e virgola (;).

È possibile usare l'istruzione **ELLIPSE** in luogo del comando **CIRCLE**

sintassi: *x = espressione_numerica*
y = espressione_numerica
raggio = espressione_numerica
ecc = espressione_numerica
angolo = espressione_numerica (da 0 a 360 gradi)
parametri = x, y 1
raggio, ecc, angolo 2
dove 1 traccia un cerchio
2 traccia un'ellisse di eccentricità e inclinazione desiderate

CIRCLE *[canale,] parametri *[; parametri]**

x	coordinata orizzontale del centro
y	coordinata verticale del centro
raggio	raggio del cerchio
ecc	eccentricità (rapporto fra raggio maggiore e minore dell'ellisse)
angolo	angolo fra l'asse maggiore dell'ellisse e l'asse verticale dello schermo. L'angolo deve essere espresso in radianti.

esempi: i. **CIRCLE 50, 50, 20**
ii. **CIRCLE 50, 50, 20, 0.5, 0** {ellisse}

CLEAR

CLEAR cancella l'area di memoria usata dalle variabili del programma attivo.

sintassi: **CLEAR**

esempio: **CLEAR**

commento È consigliabile utilizzare il comando **CLEAR** dopo che è stata interrotta l'esecuzione di un programma, a causa di un errore, prima di riprendere l'esecuzione.

CLOSE

unità e canali

CLOSE chiude il canale specificato. Eventuali finestre associate al canale sono disattivate.

sintassi: *canale = #espressione_numerica*

CLOSE *canale*

esempi: i. **CLOSE #4**
ii. **CLOSE #canale_input**

CLS
finestre

CLS cancella il contenuto della finestra associata al canale specificato o di default e attiva il colore assegnato con una istruzione **PAPER**. Il bordo della finestra non subisce modifica. È possibile specificare una serie di parametri per cancellare parzialmente il contenuto di una finestra.

sintassi: *parte = espressione_numerica*

CLS *[canale,] [parte]*

dove: *parte = 0 intero schermo (default)*
parte = 1 schermo sopra linea cursore
parte = 2 schermo sotto linea cursore
parte = 3 intera linea cursore
parte = 4 linea cursore a destra

esempi:
i. **CLS**
ii. **CLS 3** {cancella la linea del cursore}
iii. **CLS #2, 4** {cancella nella finestra associata al canale 2, la linea del cursore a destra del cursore}

CODE

CODE è una funzione che ritorna il codice **ASCII** del carattere specificato come parametro. Se il parametro è un'intera stringa, **CODE** ritorna il codice **ASCII** del primo carattere della stringa

CODE è la funzione inversa di **CHR$**

sintassi: **CODE** *(espressione_numerica)*

esempi:
i. **PRINT CODE ("A")** {stampa 65}
ii. **PRINT CODE ("Superbasic")** {stampa 83}

CONTINUE
RETRY

gestione errori

CONTINUE riprende l'esecuzione di un programma dopo un'interruzione. **RETRY** permette ad un'istruzione che ha causato un errore di essere rieseguita.

sintassi: **CONTINUE**
RETRY

esempi: **CONTINUE**
RETRY

attenzione

È possibile riprendere l'esecuzione di un programma se:

1 - non sono state aggiunte nuove istruzioni
2 - non sono state aggiunte nuove variabili
3 - non sono state modificate le istruzioni

COPY
COPY_N

unità e canali

COPY copia un file da una unità di input ad una di output. **COPY_N** ha la stessa funzione di COPY, ma non copia, se esiste, l'identificatore del file. L'identificatore permette ad un file di essere copiato da un Microdrive ad un'altra unità in modo corretto.

Gli identificatori devono essere sempre rimossi quando si copia un file ad una unità che non permette di strutturare una directory. Ad esempio i Microdrive o i floppy disk sono dispositivi che consentono di creare una directory, mentre non ha alcun significato definire una directory per le porte seriali.

sintassi: **COPY** *unità* **TO** *unità*
COPY_N *unità* **TO** *unità*

esempi: i. **COPY mdv1_file TO con_** {copia il file nella finestra di default}
COPY mdv1_file TO con_
ii. **COPY_N mdv1_dati TO ser1** {copia il file dati sulla porta seriale eliminando l'identificatore}

COS
funzioni matematiche

COS calcola il coseno del parametro specificato.

sintassi: *angolo = espressione__numerica* {da -51.500 a 10.000 in radianti}

COS *(angolo)*

esempi: i. **PRINT COS(beta)**
ii. **PRINT COS(3.141592/2)**

COT
funzioni matematiche

COT calcola la cotangente del parametro specificato.

sintassi: *angolo = espressione__numerica* {da -30000 a 30000 in radianti}

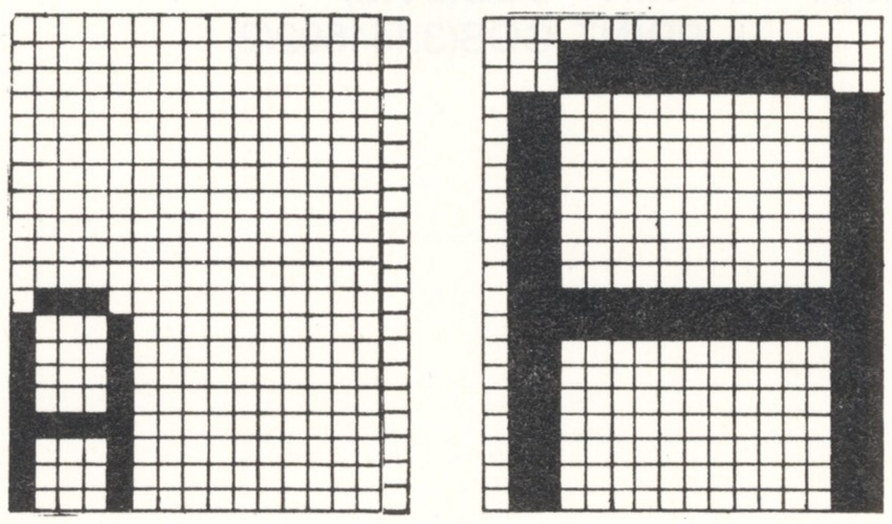
COT*(angolo)*

esempi: i. **PRINT COT(3)**
ii. **PRINT COT(3.141592654/2)**

CSIZE
finestre

CSIZE assegna una nuova dimensione ai caratteri della finestra associata al canale specificato o di default. La dimensione standard è di 0,0 nel Mode 512 e di 2,0 nel Mode 256.

La grandezza dei caratteri è dimensionata in modo da riempire proporzionalmente lo spazio disponibile.



larghezza	dimensione	altezza	dimensione
0	6 pixel	0	10 pixel
1	8 pixel	1	20 pixel
2	12 pixel		
3	16 pixel		

sintassi: *larghezza = espressione__numerica* {da 0 a 3}
altezza = espressione__numerica {da 0 a 1}

CSIZE *[canale,] larghezza, altezza*

esempi: i. **CSIZE 3,0**
ii. **CSIZE 3,1**

CURSOR
finestre

CURSOR permette di posizionare il cursore in un qualsiasi punto della finestra associata al canale specificato o di default.

CURSOR usa il sistema di coordinate a pixel con origine relativa alla finestra corrente. Le dimensioni del cursore dipendono dalla dimensione del carattere scelto.

Se l'istruzione **CURSOR** è usata con quattro parametri, la prima coppia è interpretata come un punto nel sistema di coordinate grafiche, la seconda coppia definisce la posizione del cursore (nel sistema di coordinate a pixel) relativa al primo punto.

sintassi: *x = espressione__numerica*
y = espressione__numerica

CURSOR *[canale,] x, y, [,x,y]*

esempi: i. **CURSOR 0,0**
ii. CURSOR 20, 30
iii. CURSOR 50, 50, 10, 10

DATA
READ
RESTORE

basic

READ, DATA e **RESTORE** consentono di assegnare dati contenuti in un programma Superbasic a delle variabili.

L'istruzione **DATA** è usata per definire i dati, **READ** per assegnare i dati alle corrispondenti variabili, e **RESTORE** per scegliere i dati.

DATA

L'istruzione **DATA** non è eseguita dal programma e può essere ovunque nel programma.

sintassi: **DATA** **[espressione,]**

READ

L'istruzione **READ** legge le informazioni contenute nelle istruzioni **DATA** e le assegna ad una serie di variabili. Inizialmente il puntatore è in corrispondenza della prima istruzione **DATA** del programma ed è incrementato dopo ogni comando **READ**. La riesecuzione del programma non azzera il puntatore.

sintassi: **READ** **[nomi di variabile]**

RESTORE

L'istruzione **RESTORE** consente di rileggere le istruzioni DATA di una determinata riga. Se **RESTORE** è seguito da un numero di riga, il valore del puntatore è modificato in corrispondenza di quella riga. Se non è specificato alcun parametro, il puntatore è ripristinato al valore iniziale.

sintassi: **RESTORE** *[numero di riga]*

esempi: i.
```
100 REMark esempio
110 DIM giorno$(7,4)
120 RESTORE
130 FOR num = 1 TO 7
135       READ giorno$(num)
140 END FOR num
150 PRINT giorno$
160 DATA "lun", "mar", "mer", "gio"
170 DATA "ven", "sab", "dom"
```

ii.
```
100 REMark esempio
110 DIM mese$(12,9)
120 RESTORE
130 FOR num = 1 TO 12
135       READ mese$(num)
140 END FOR num
150 PRINT mese$
160 DATA "gen", "feb", "mar", "apr"
170 DATA "mag", "giu", "lug"
180 DATA "ago", "set", "ott", "nov"
190 DATA "dic"
```

attenzione

L'istruzione **RESTORE** non è eseguita automaticamente prima dell'esecuzione di un programma.

DATE$
DATE

orologio

DATE$ è una funzione che ritorna la data e l'ora contenute nell'orologio interno del QL.

Il formato della stringa ritornata dalla funzioine è:
"aaaa mmm dd hh:mm:ss"
dove

aaaa è l'anno 1984, 1985, ecc.

mmm è il mese in inglese {jan, feb, ecc}
dd è il giorno **hh** è l'ora
mm sono i minuti
ss sono i secondi

DATE è una funzione che torna la data come valore numerico e può essere usata per memorizzare le date in forma compatta.

sintassi: **DATE$** {legge la data dall'orologio}
DATE$ *(espressione_numerica)* {interpreta l'espressione numerica come data}

esempi: i. **PRINT DATE$**
ii. **PRINT DATE$(234567)**

DAY$

orologio

DAY$** è una funzione che ritorna il nome del giorno della settimana corrispondente alla data letta dall'orologio. Quando è caso sia specificato un parametro, la funzione **DAY$ interpreta il parametro come data e ritorna il nome del giorno della settimana corrispondente.

sintassi: **DAY$**
DAY$ *(espressione_numerica)*

esempi: i. **PRINT DAY$**
ii. **PRINT DAY$ (234567)**

DEFine
FUNction
END DEFine

funzioni e procedure

DEFine FUNction permette di costruire la definizione di una funzione. La sequenza di istruzioni compresa fra **DEFine** e **END DEFine** determina il tipo di funzione. Nella definizione possono essere dichiarati alcuni parametri formali, che trasferiscono valori esterni all'interno della funzione. Sia i parametri formali che reali devono essere racchiusi fra parentesi.

I parametri formali sono dello stesso tipo dei corrispondenti parametri reali.

È possibile visualizzare i valori delle variabili attuali con l'istruzione **RETURN** nella sequenza di istruzioni che definiscono la funzione.

In Superbasic una funzione è richiamata semplicemente attraverso il suo nome.

In Superbasic le funzioni possono richiamare se stesse direttamente o indirettamente attraverso una serie di altre chiamate.

Il tipo di funzione, alfabetica o numerica, è determinato dal tipo di identificatore che segue l'istruzione **DEFine FUNction**.

sintassi: *parametri_formali = (espressione *[,espressione]*)*
*parametri_reali = (espressione *[,espressione]*)*
identificatore = $, %

DEF FUNction *identificatore [parametri formali]*
*[**LOCAL** identificatore *[,identificatore]*]*
istruzioni
RETURN *espressione*
END DEFine

RETURN può essere in una qualsiasi posizione fra le istruzioni che definiscono la funzione.

LOCAL deve precedere la prima istruzione eseguibile nella serie di istruzioni che definiscono la funzione.

esempi: i.
```
10 DEFine FUNCtion media (a, b, c)
20    LOCAL risultato
30    LET risultato = (a + b + c)/3
40    RETURN risultato
50 END DEFine
60 PRINT media (10, 20, 30)
```

commenti

Per migliorare la leggibilità di un programma è possibile ripetere il nome della funzione nell'istruzione **END DEFine.**

DEFine
PROCedure
END DEFine

funzioni e procedure

DEFine PROCedure definisce una procedura Superbasic. Il corpo della procedura è definito dalle istruzioni comprese fra **DEFine PROCedure** e **END DEFine**. La definizione di procedura può includere una serie di parametri formali, che trasferiscono dati esterni all'interno della procedura. I parametri formali devono essere racchiusi fra parentesi quando sono dichiarati nella definizione, ma le parentesi non sono necessarie quando la procedura è richiamata. Se la procedura non richiede parametri, non è necessario specificare un insieme vuoto fra parentesi nella definizione della procedura.

I parametri formali sono dello stesso tipo dei parametri reali.

È possibile definire variabili locali all'interno di una procedura. Le variabili locali possono avere lo stesso nome di variabili già esistenti al di fuori della procedura, perché il loro valore è modificato solo da istruzioni interne alla procedura. Se necessario, anche vettori possono essere variabili locali.

Una procedura è richiamata scrivendo il suo nome come prima parola di un'istruzione Superbasic seguito da una serie di parametri reali. In Superbasic una procedura può richiamare se stessa direttamente, o indirettamente tramite una serie di altre chiamate.

sintassi: *parametri__formali = (espressione *[,espressione]*)*
*parametri__reali = (espressione *[,espressione]*)*

DEF PROCedure *identificatore [parametri__formali]*
[**LOCAL** *identificatore *[,identificatore]**]
istruzioni
RETURN *espressione*
END DEFine

RETURN può essere in una qualsiasi posizione nella serie di istruzioni che definiscono la procedura

LOCAL deve precedere la prima istruzione eseguibile nella serie di istruzioni che definiscono la procedura.

esempi: i.
```
100 DEFine PROCedure video
110    WINDOW 100, 100, 10, 10
120    PAPER 7 : INK 0 : CLS
130    BORDER 4,255
140    PRINT "ciao a tutti!"
150 END DEFine
160    video
```

ii.
```
100 DEFine PROCedure video__lento (limite)
110    LOCAL num
120    FOR num = 1 TO limite
130       SCROLL 2
140    END FOR num
150 END DEFine
160 video__lento 20
```

commento

Per migliorare la leggibilità di un programma è possibile ripetere il nome della procedura nell'istruzione **END DEFine.**

DEG

funzioni matematiche

DEG è una funzione che converte un angolo espresso in radianti nel corrispondente valore in gradi.

sintassi: **DEG** *(espressione__numerica)*

esempi: i. **PRINT DEG (pi/2)** {stampa 90}

DELETE
Microdrive e floppy

DELETE cancella un file dal Microdrive o dal floppy disk specificato.

sintassi: **DELETE** *unità*

esempi: i. **DELETE mdv1_dati_doc**
ii. **DELETE mdv2_grafico_mio**

DIM
basic

DIM definisce le dimensioni ed il tipo di vettore in un programma Superbasic. Sono permessi vettori di numeri interi, o a virgola mobile, e vettori alfabetici. I vettori alfabetici devono contenere elementi di lunghezza fissa, che è rappresentata dal valore dichiarato nell'istruzione **DIM**.

Gli indici del vettore variano da 0 fino al valore specificato nell'istruzione **DIM**. Il numero di elementi di un vettore è dato dal valore espresso nell'istruzione **DIM** più uno.

L'istruzione **DIM** ha anche la funzione di azzerare i valori degli elementi del vettore.

sintassi: *indice = espressione_numerica*
*vettore = identificatore ([indice *[,indice]*)*

DIM *vettore * [,vettore]**

esempi: i. **DIM vettore_parola$(10, 10, 50)**
ii. **DIM matrice(100, 100)**

L'ultimo indice (50 nell'esempio i), che compare in una istruzione **DIM** di un vettore alfabetico, è la lunghezza di ogni elemento.

DIMN

vettori

DIMN è una funzione che ritorna il valore della dimensione richiesta di un vettore. Se non è specificata alcuna dimensione, l'istruzione **DIMN** ritorna il valore della prima dimensione del vettore. Se la dimensione richiesta non esiste la funzione **DIMN** dà un valore 0.

sintassi: dimensione = espressione_numerica

DIMN *(vettore[,dimensione])* {1 per il primo indice, ecc.}

esempi: Supponiamo di aver definito il seguente vettore: **DIM A(2,3,4)**

i. **PRINT DIMN(A,1)** {stampa 2}
ii. **PRINT DIMN(A,2)** {stampa 3}
iii. **PRINT DIMN(A,3)** {stampa 4}
iv. **PRINT DIMN(A)** {stampa 2}
v. **PRINT DIMN(A,4)** {stampa 0}

DIR

Microdrive/Floppy

DIR visualizza, nella finestra associata al canale specificato o di default, l'elenco dei file contenuti nel Microdrive indicato.

sintassi: **DIR** *unità*

Le informazioni appaiono sul video nel seguente formato.

settori_vuoti=numero di settori ancora vuoti
settori_disponibili=numero totale di settori disponibili sulla cartuccia
nome_file=nome del file
formato video: nome volume
settori_vuoti/settori_disponibili
nome_file
nome_file

esempi:
i. **DIR_mdv1**
ii. **DIR_mdv2**
iii. **DIR "mdv" &numero_Microdrive$ &"_"**

DIV
funzioni matematiche

DIV è una funzione che esegue una divisione a quoziente intero.

sintassi: *espressione__numerica*

DIV *espressione__numerica*

esempi: i. **PRINT 5 DIV 2** {stampa 2}
ii. **PRINT —5 DIV 2** {stampa —3}

DLINE
basic

DLINE cancella una riga o un gruppo di righe da un programma Superbasic.

sintassi: *limite = numero__riga* ***TO*** *numero__riga* 1
numero__riga ***TO*** 2
TO *numero__riga* 3
numero__riga 4

DLINE limite *[,limite]*

dove 1 cancella un gruppo di istruzioni
2 cancella le istruzioni da numero__riga fino al termine dl programma
3 cancella le istruzioni dall'inizio del programma fino a numero__riga
4 cancella l'istruzione specificata

esempi: i. **DLINE 10 TO 70,80,200 TO 400**
{cancella le righe dalla 10 alla 70 incluse, la 80, dalla 200 alla 400 incluse}

ii. **DLINE** {non cancella niente}

EDIT

EDIT consente l'accesso all'editor del Superbasic. L'istruzione **EDIT** è simile all'istruzione **AUTO**. Il comando **EDIT** permette la correzione di una sola riga per volta, a meno che non sia specificato un secondo parametro, che è assunto come incremento.

Se uno dei numeri di riga specificati non esiste, il Superbasic accetta l'inserimento della nuova riga.

Il cursore può essere spostato sulla riga di editing premendo i tasti usuali.

[→] cursore a destra	[↑] cursore in alto	ha lo stesso effetto di [ENTER] ma predispone la riga precedente all'**EDIT**.
[←] cursore a sinistra	[↓] cursore in basso	ha lo stesso effetto di [ENTER] ma predispone la riga successiva all'**EDIT**.

[CTRL] [→] cancella carattere a destra

[CTRL] [←] cancella carattere a destra

Quando le correzioni sono terminate, [ENTER] permette la memorizzazione della nuova istruzione.

Se è stato specificato un passo, è predisposta all'**EDIT** la riga successiva nella sequenza.

sintassi: *passo=espressione_numerica*

EDIT *numero_riga[,passo]*

esempi:
i. **EDIT 10**
ii. **EDIT 20,10** {predispone all'**EDIT** le righe 20,30 ecc}.

EOF
file o canali

EOF indica che è stata raggiunta la fine del file associato ad uno specificato canale. Se non è stato indicato alcun canale, **EOF** determina la fine dei dati (inseriti con le istruzioni DATA) di un programma.

sintassi: **EOF** *[(canale)]*

esempi:
i. **IF EOF (#6) THEN STOP**
ii. **IF EOF THEN PRINT "fine dati"**

EXEC
EXEC_W
qdos

EXEC e **EXEC_W** caricano in memoria una sequenza di programmi e li eseguono in parallelo.

EXEC ritorna il controllo al sistema operativo dopo l'inizio dell'esecuzione di tutti i programmi.

EXEC_W ritorna il controllo al sistema operativo solo al termine dell'esecuzione di tutti i programmi.

sintassi: *programma=dispositivo*
EXEC *programma*

esempi: i. **EXEC mdv1_gestione_prg**
ii. **EXEC_W mdv1_stampa_prg**

EXIT
cicli o
iterazioni

EXIT consente di terminare un ciclo **FOR** o una struttura **REPeat** e di riprendere l'esecuzione di un programma.

sintassi: **EXIT** *identificatore*

esempi: i.
```
100 REM inizio loop
110 LET num=0
120 REPeat loop
130       LET num=num+1
140       PRINT num
150       IF num=20 THEN EXIT loop
160 END REPeat loop
```
{il programma esce dalla struttura loop quando num=20}

ii.
```
100 FOR n=1 TO 100
110       IF RND >0.5 THEN EXIT n
120 END FOR n
```
{il programma esce dal ciclo **FOR** quando è generato un numero casuale maggiore di 0.5}

EXP

funzioni matematiche

EXP calcola il valore di e (base dei logaritmi naturali), elevata alla potenza specificata.

sintassi: **EXP** *(espressione_numerica)* {da —500 a 500}

esempi: i. **PRINT EXP(3)**

FILL

grafica

FILL attiva o disattiva il riempimento di figure chiuse e convesse (senza rientranze). Figure che non hanno queste caratteristiche devono essere scomposte in parti più piccole, ognuna chiusa e convessa.

FILL 0 disattiva la funzione di riempimento
FILL 1 attiva la funzione di riempimento

sintassi: *funzione=espressione_numerica* {0,1}

FILL*[canale,]funzione*

esempi: i. **FILL 1: LINE 10,10 TO 50,50 TO 30,90 TO 10,10**
disegna un triangolo pieno

ii. **FILL 1: CIRCLE 50,50,20 : FILL 0**
{disegna un cerchio pieno}

FILL$

stringhe

FILL$ crea una stringa di lunghezza specificata contenente la ripetizione di uno o due caratteri.

sintassi: **FILL$***(caratteri, espressione_numerica)*

il numero di caratteri può essere uno o due

esempi:
i. `PRINT FILL$("A",5)` {stampa AAAAA}
ii. `PRINT FILL$("oO",7)` {stampa oOoOoOo}
iii. `LET A$=A$ & FILL$(" ",10)`

FLASH

finestre

FLASH rende lampeggianti i caratteri sullo schermo. **FLASH** produce effetti visibili soltanto nel modo a bassa risoluzione.

FLASH 0 disattiva la funzione di lampeggiamento
FLASH 1 attiva la funzione di lampeggiamento

sintassi: *funzione=espressione_numerica* {0,1}

FLASH*[canale,]funzione*

esempi: i.
```
100 PRINT "A";
110 FLASH 1
120 PRINT "lampeggiamento"
130 FLASH 0
140 PRINT "ciao"
```

attenzione

È consigliabile evitare di stampare su una parte di schermo lampeggiante, perché questo può produrre risultati inaspettati.

FOR
END FOR

iterazioni **FOR** consente la ripetizione di un gruppo di istruzioni il numero di volte dichiarato.

NEXT e FOR possono essere usate in uno stesso ciclo **FOR**, se si vuole che una sequenza di istruzioni non sia eseguita, quando l'uscita dal ciclo **FOR** avviene tramite l'istruzione **EXIT**.

variabile=espressione__numerica
*esp__numerica **TO** esp__numerica*
*esp__numerica **TO** esp__numerica **STEP** esp__numerica*

*elenco=variabile *[,variabile]**

forma breve Quando il ciclo **FOR** è concluso, il controllo del programma passa all'istruzione successiva. Non è necessario concludere un ciclo **FOR** con le istruzioni **NEXT o END FOR**.

In questo caso non è possibile annidare più cicli **FOR**.

sintassi:
```
FOR variabile=elenco
istruzione
istruzione
.....
END FOR
```

esempio:
```
100 INPUT "scrivi un numero" ! x
110 LET fattoriale=1
120 FOR valore=x TO 1 STEP —1
130       fattoriale=fattoriale * valore
140       PRINT x !!!! fattoriale
150       IF fattoriale > E20 THEN
160          PRINT "numero troppo grande"
170          EXIT valore
180       END IF
190 END FOR valore
```

attenzione La variabile di controllo del ciclo **FOR** deve essere un numero in virgola mobile.

FORMAT

Microdrive **FORMAT** formatta e rende pronta all'uso una cartuccia inserita nel Microdrive specificato.

sintassi: **FORMAT***[canale,] unità*

Sullo schermo è riportato il numero totale di settori della cartuccia ed il numero totale di settori disponibili per memorizzare i dati.

Prima di usare una cartuccia è consigliabile formattarla tre o quattro volte.

esempi: i. **FORMAT mdv1__cartuccia__dati**
{alla cartuccia formattata è assegnato il nome cartuccia__dati}

attenzione L'istruzione **FORMAT** può essere usata per riutilizzare una cartuccia già usata. In questo caso tutti i dati memorizzati sulla cartuccia sono cancellati.

GOSUB

basic

Per ragioni di compatibilità con gli altri Basic, il Superbasic ammette l'istruzione **GOSUB.**

GOSUB trasferisce il controllo del programma al numero di riga specificato. **RETURN** è necessario per riportare l'esecuzione del programma all'istruzione immediatamente successiva a **GOSUB**.

sintassi: **GOSUB** *numero_riga*

esempi: i. **GOSUB 100**
ii. **GOSUB 4*variabile**

I tipi di struttura disponibili in Superbasic rendono inutile l'istruzione **GOSUB.**

commento

GOTO

basic

Per ragioni di compatibilità con gli altri Basic, il Superbasic ammette l'istruzione **GOTO.**

GOTO trasferisce incondizionatamente l'esecuzione del programma al numero di riga specificato.

sintassi: **GOTO** *numero_riga*

esempi: i. **GOTO inizio**
ii. **GOTO 9999**

I tipi di struttura disponibili in Superbasic rendono inutile l'istruzione **GOTO**

commento

IF
THEN
ELSE
END IF

L'istruzione **IF** consente di controllare il flusso di un programma per mezzo di condizioni logiche.

L'istruzione **IF** ammette due forme: breve e lunga.

FORMA BREVE

Nella forma breve, le istruzioni **THEN** e **ELSE** devono essere seguite immediatamente da altri comandi.

Se la condizione espressa nell'istruzione **IF** è vera, sono eseguite tutte le istruzioni comprese fra **THEN** e **ELSE**.

Se la condizione è falsa, sono eseguite le istruzioni successive ad **ELSE**.

Se la sequenza di istruzioni non contiene **ELSE** e la condizine **IF** è vera, sono eseguite le istruzioni successive a **THEN**. Se la condizione è falsa, l'esecuzione del programma continua normalmente.

sintassi: **IF** *espressione* **THEN** *istruzioni* [**ELSE** *istruzioni*]

esempi:

i.
```
IF a=32 THEN PRINT "limite": ELSE PRINT "OK"
```

ii.
```
IF test > max THEN LET max=test
```

Se l'istruzione **IF** ha come ultima parola della riga **THEN** ed è seguita da una serie di istruzioni che terminano con **END IF**, allora il programma esegue le righe contenute fra **IF** e **END IF**. L'istruzione **ELSE** seguita da un'altra serie di istruzioni è facoltativa.

sintassi:
```
IF espressione THEN
   istruzione
.....
[ELSE
   istruzione
.....]
ENDIF
```

esempi: i.
```
100 LET limite=10
110 INPUT "scrivi un numero" ! numero
120 IF numero > limite THEN
130    PRINT "fuori limite"
140 ELSE
150    PRINT "ok"
160 END IF
```

commento

Le istruzioni **IF** possono essere "nidificate" a piacere, compatibilmente con la disponibilità di spazio in memoria.

esempi: i.
```
100 IF a=b THEN ————————————————————
110    IF c=d THEN ————————————————  |
120       PRINT "errore"           |  |
130    ELSE                        |  |
140       PRINT "corretto"         |  |
150    END IF <———————————————————  |
160 ELSE <————————————————————————————
170 PRINT "non controllato"
180 END IF
```

INK
finestre

INK seleziona il colore con il quale sono stampati i caratteri sul video, nella finestra associata al canale specificato o di default.

sintassi: **INK***[canale,]colore*

esempi:
i. **INK 5**
ii. **INK 6,2**
iii. **INK #2,255**

INKEY$
basic

INKEY$ legge un carattere dal canale specificato o di default.

Può essere assegnato un tempo di attesa prima della lettura. Se non è specificato alcun parametro, **INKEY$** legge immediatamente il carattere.

sintassi: **INKEY$***[(canale (,tempo))]*

dove:

tempo=1....32767	{attesa di n cinquantesimi di secondo}
tempo=-1	{attesa a tempo indeterminato}
tempo=0	{nessuna attesa}

esempi:
i. **PRINT INKEY$** {input dal canale di deault}
ii. **PRINT INKEY$(#4)** {input dal canale 4}
iii. **PRINT INKEY$(#3,100)** {attesa per 100 cinquantesimi di secondo di un input dal canale 3.}

INPUT

basic

INPUT consente l'immissione di dati, durante l'esecuzione di un programma, direttamente dalla tastiera. Quando un programma incontra un'istruzione **INPUT** si arresta ed aspetta che la quantità di dati richiesti da **INPUT** sia inserita. L'immissione di ogni dato deve terminare con ENTER .

Il cursore della finestra associata al canale di **INPUT**, lampeggia fino a quando l'immissione è esaurita.

sintassi: *separatori=!*
,
/
;
TO

richiesta=[canale,] separatore

INPUT *[richiesta][canale] variabile *[,variabile]**

esempi:

i. **INPUT** "indovina un numero"; numero

ii.
```
100 INPUT"dimensioni vettore?" ! limite
110 DIM vettore(limire —1)
120 FOR elemento=0 TO limite —1
130       INPUT ("dato per l'elemento nr" & elemento) ! vettore
          (elemento)
140 END FOR elemento
150 PRINT vettore
```

INSTR

funzioni

INSTR ricerca il contenuto di una stringa in un'altra stringa e ritorna il valore della posizione in cui è stata trovata la corrispondenza. Se la stringa non è stata trovata **INSTR** ritorna il valore 0.

sintassi: *espressione_alfabetica* **INSTR** *espressione_alfabetica*

esempi:

i. **PRINT "a" INSTR "mela"** {stampa 4}

ii. **PRINT "x" INSTR "uova"** {stampa 0}

iii. **PRINT "atto" INSTR "gatto"** {stampa 2}

INT
funzioni matematiche

INT ritorna l'intero più grande che è minore o uguale al valore specificato.

sintassi: **INT** *(espressione__numerica)*

esempi: i. **PRINT INT(x)**
ii. **PRINT INT(3,141592)**

KEYROW
funzioni

KEYROW è una funzione che verifica lo stato della fila di tasti specificata (la tabella di seguito riportata illustra la disposizione della tastiera in una matrice di otto righe per otto colonne).

KEYROW richiede un parametro, che deve essere un intero fra 0 e 7. Il parametro definisce quale è la riga da verificare.

Il valore ritornato da **KEYROW** è un intero fra 0 e 255, che rappresenta, in forma binaria, il tasto premuto nella riga scelta.

Poiché **KEYROW** è usata in alternativa alle istruzioni di input **INKEY$** e **INPUT**, il buffer della tastiera è azzerato. I tasti premuti prima di una istruzione **KEYROW** non sono letti da un'istruzione **INPUT** o **INKEY$** successive.

sintassi: *riga=espressione__numerica* {da 0 a 7}

KEYROW*(riga)*

esempi: i.
```
100 REMark KEYROW
110 REPeat loop
120     CURSOR 0,0
130     FOR riga=0 TO 7
140         PRINT riga !!! KEYROW(riga) ;" "
150     END FOR riga
160 END REPeat loop
```

MATRICE DELLA TASTIERA

RIGA	COLONNA 1	2	4	8	16	32	64	128
7	7 SHIFT 1	7 CTRL 2	7 ALT 4	7 X 8	7 V 16	7 Ò 32	7 N 64	7 ; 128
6	6 ^ . 1	6 é 2	6 — 4	6 Q 8	6 E 16	6 à 32	6 T 64	6 U 128
5	5 Ç 1	5 Z 2	5 I 4	5 TAB 8	5 R 16	5) 32	5 Y 64	5 O 128
4	4 L 1	4 " 2	4 H 4	4 # 8	4 A 16	4 P 32	4 D 64	4 J 128
3	3 í 1	3 CAPS LOCK 2	3 K 4	3 S 8	3 F 16	3 - 32	3 G 64	3 M 128
2	2 $ 1	2 W 2	2 : 4	2 C 8	2 B 16	2 * 32	2 , 64	2 Ù 128
1	1 ENTER 1	1 ← 2	1 ↑ 4	1 ESC 8	1 → 16	1 < 32	1 SPAZIO 64	1 ↓ 128
0	0 F4 1	0 F1 2	0 (4	0 F2 8	0 F3 16	0 F5 32	0 / 64	0 è 128

valore passato alla funzione **KEYROW** (numero di riga)

TASTO — 0 F1 8 — legenda

valore ritornato da **KEYROW** (numero di colonna)

LBYTES
file e memoria

LBYTES trasferisce in memoria un file di dati iniziando da un indirizzo assegnato.

sintassi: *indirizzo=espressione_numerica*

LBYTES *dispositivo_nome file, indirizzo*

esempi:
i. **LBYTES mdv1_figura, 131072**
⟨trasferisce in memoria uno schermo⟩

ii. **LBYTES mdv1_programma, indirizzo**
⟨trasferisce in memoria un programma ad un indirizzo specifico⟩

LEN
stringhe

LEN ritorna il numero di caratteri di una variabile alfabetica.

sintassi: **LEN***(espressione_alfabetica)*

esempi:
i. **PRINT LEN("ciao a tutti")**
ii. **PRINT LEN(stringa$)**

LET

LET assegna il valore di un'espressione ad una variabile. L'istruzione **LET** è facoltativa. Il Superbasic converte automaticamente dati e variabili incompatibili, quando possibile.

sintassi: **[LET]** *variabile=espressione__numerica*

esempi:
i. **LET a=1 + "1"**
ii. **LET a$="12345"**
iii. **LET a$=6789**

LINE
LINE_R

grafica

LINE permette di tracciare un segmento nella finestra associata al canale specificato o di default. Gli estremi del segmento sono assegnati usando il sistema di coordinate grafiche.

Con una sola istruzione **LINE** è possibile tracciare più di un segmento.

Gli estremi del segmento possono essere specificati in coordinate assolute o relative. Se il primo punto non è assegnato, è disegnato un segmeno dalla posizione del cursore al punto specificato. Se è omesso il secondo punto, non è tracciato alcun segmento, ma il cursore è spostato sulla posizione del primo punto.

LINE traccia un segmento usando coordinate assolute.

LINE__R traccia un segmento usando coordinate relative.

sintassi: *x=espressione__numerica*
y=espressione__numerica
punto=x,y

parametro__2=	**TO** *punto*	1
	,punto **TO** *punto*	2
parametro__1	**TO** *punto, angolo*	1
	TO *punto*	2
	punto	3

LINE *[canale,] parametro__1 * [, parametro__2]**

LINE__R *[canale,] parametro__1 * [,parametro__2]**

dove: 1- traccia un segmento fra i due punti specificati
2- traccia un segmento dal punto assegnato all'ultimo punto disegnato
3- non è tracciato alcun segmento e il cursore è spostato sulla posizione del punto assegnato

esempi: i. **LINE 0,0 TO 0,50 TO 50,0 TO 50,50 TO 0,0**
{disegna un quadrato}

LIST
basic

LIST lista il programma in memoria sul video o sulla stampante.

È possibile interrompere l'azione di **LIST** premendo **CTRL SPAZIO (BREAK).**

sintassi: *riga=* *numero_riga* **TO** *numero_riga* 1
numero_riga **TO** 2
TO *numero_riga* 3
numero_riga 4
5

LIST *[canale,] riga * [,riga]**

dove:

1 lista tutte le istruzioni fra gli estremi specificati
2 lista tutte le istruzioni fra il numero di riga specificato e l'ultima
3 lista tutte le istruzioni fra la prima ed il numero di riga specificato
4 lista il numero di riga specificato
5 lista l'intero programma

esempi: i. **LIST** {lista tutte le righe}
ii. **LIST 10 TO 300** {lista dalla riga 10 alla 300}
iii. **LIST 12,20,50** {lista le righe 12,20 e 50}

commento

.Se l'output dell'istruzione **LIST** è diretto ad una porta seriale, il programma è listato sulla stampante.

È possibile interrompere il comando con

CTRL	SPAZIO

LOAD
unità è
microdrive

LOAD carica un programma Superbasic da una qualsiasi unità specificata. l'istruzione **LOAD** prima di caricare un programma, cancella il contenuto della memoria.

Se, durante il caricamento, il Superbasic riscontra un errore di sintassi in qualche istruzione, inserisce la parola **MISTAKE** fra il numero di riga e il primo carattere dell'istruzione.

sintassi: **LOAD** *unità_nome programma*

esempi: i. **LOAD mdv1_gioco**
ii. **LOAD ser1_e**
iii. **LOAD neti_3**

LN
LOG10

funzioni matematiche

LN calcola il logaritmo naturale del valore specificato.

LOG10 calcola il logaritmo in base dieci del valore assegnato.

sintassi: **LOG10***(espressione_numerica)* {maggiore di 0}
LN*(espressione_numerica)* {maggiore di 0}

esempi:
i. **PRINT LOG10(20)**
ii. **PRINT LN(3141592)**

LOCAL

funzioni e procedure

LOCAL permette di definire delle variabili come locali ad una funzione o procedura. Le variabili definite localmente, sono cancellate al termine della esecuzione della funzione o procedura. Le variabili locali possono avere lo stesso nome di variabili già esistenti nel programma. Anche i vettori possono essere definiti in modo locale, purchè siano dimensionati nell'istruzione **LOCAL**.

L'istruzione **LOCAL** deve precedere la prima istruzione eseguibile all'interno della definizione di una procedura o funzione.

sintassi: **LOCAL** *variabili *[,variabili]**

esempi:
i. **LOCAL a,b,c(10,10)**
ii. **LOCAL var1**

LRUN
unità e
microdrive

LRUN permette di caricare ed eseguire un programma Superbasic da una unità specificata. **LRUN**, cancella il contenuto della memoria, prima di caricare il programma.

Se, durante il caricamento, il Superbasic riscontra un errore di sintassi in qualche istruzione, inserisce la parola **MISTAKE** fra il numero di riga e il primo carattere dell'istruzione.

sintassi: **LRUN** *unità_nome programma*

esempi: i. **LRUN mdv1_gioco**

MERGE
unità e
microdrive

MERGE carica un file da una unità specificata e lo interpreta come programma Superbasic. L'istruzione **MERGE** fonde le righe del file del programma caricato con quelle del programma residente in memoria.

I numeri di riga del programma in memoria, esistenti anche nel file caricato, restano invariati.

Se, durante il caricamento, il Superbasic riscontra un errore di sintassi in qualche istruzione, inserisce la parola **MISTAKE** fra il numero di riga e il primo carattere dell'istruzione.

sintassi: **MERGE** *unità_nome programma*

esempi: i. **MERGE mdv1_nuovo_programma**

MOD

funzioni matematiche

MOD calcola il modulo, o resto, della divisione fra due numeri interi.

sintassi: *espressione_numerica* **MOD** *espressione_numerica*

esempi:
i. **PRINT 5 MOD 2** {stampa 1}
ii. **PRINT 5 MOD 3** {stampa 2}

MODE

schermo

MODE seleziona il tipo di risoluzione dello schermo. L'**MODE** cancella il contenuto di tutte le finestre sul video, ma conserva le loro posizioni e dimensioni. Il modo a bassa risoluzione (otto colori) consente come grandezza minima dei caratteri il valore 2,0 in un'istruzione **CSIZE**.

sintassi: **MODE** *espressione_numerica*

dove: 8 o 256=bassa risoluzione
4 o 512=alta risoluzione

esempi:
i. **MODE 256**
ii. **MODE 4**

MOVE
grafica

MOVE è un'istruzione tipica di altri linguaggi, quali il LOGO. **MOVE** consente di muovere sul video la cosiddetta tartaruga grafica. È possibile variare la direzione del movimento della tartaruga con le istruzioni **TURN e TURN TO**. Se nell'istruzione **MOVE** sono specificati valori negativi, la tartaruga si muove all'indietro. La tartaruga appare nella finestra associata al canale specificato o di default.

sintassi: *distanza=espressione_numerica*

MOVE *[canale,] distanza*

esempi:
i. **MOVE #2,20** {muove la tartaruga nel canale 2 di 20 passi avanti}
ii. **MOVE -50** {muove la tartaruga nel canale di default 50 passi indietro}

MRUN
Unità e microdrive

MRUN interpreta un file come un programma Superbasic e lo fonde con il programma in memoria. Se usato come comando diretto, **MRUN** carica in memoria il nuovo programma e lo esegue.

Se, durante il caricamento, il Superbasic riscontra un errore di sintassi in qualche istruzione, inserisce la parola **MISTAKE** fra il numero di riga e il primo carattere dell'istruzione.

sintassi: **MRUN** *unità_nome programma*

esempi: i. **MRUN mdv1_nuovo_programma**

NET
rete locale

NET consente di assegnare ad un QL, collegato in network, il numero di identificazione di stazione.

Il numero assunto automaticamente dal QL è 1.

È consigliabile evitare di avere due stazioni con uno stesso numero di identificazione in una rete locale.

sintassi: *stazione=espressione_numerica* {da 1 a 127}

NET *stazione*

esempi: i. **NET 63**

NEW

NEW cancella le variabili ed i programmi in memoria. Chiude, inoltre, tutti i canali ad eccezione di **#0, #1, #2.**

sintassi: **NEW**

esempio: **i. new**

NEXT
iterazioni

NEXT permette di chiudere i cicli iterativi **FOR** e **REPeat**.

sintassi: **NEXT** *identificatore*

esempi: i.
```
10 REMark ciclo_infinito
11 REPeat ciclo
12       PRINT "per sempre"
13 NEXT ciclo
```

ii.
```
10 REMark 20 ripetizioni
11 LET limite=20
12 FOR indice=1 TO limite
13       PRINT indice
14 NEXT indice
```

iii.
```
10 REMark il ciclo quando numero=15
11 REPeat ciclo
12       LET numero=RND(1 TO 100)
13       IF numero <> 30 THEN NEXT ciclo
14       PRINT numero; "è 30"
15 EXIT ciclo
16 END REPeat ciclo
```

REPEAT

Se l'istruzione **NEXT** è usata all'interno di una struttura **REPeat** forza la continuazione dell'iterazione, senza che siano eseguite le istruzioni comprese fra **NEXT** e **END REPeat**.

FOR

Se l'istruzione **NEXT** è usata all'interno di un ciclo **FOR** chiude il processo di iterazione, e assegna alla variabile di controllo del ciclo il valore successivo.

ON...GOTO
ON...GOSUB
basic

Per ragioni di compatibilità con gli altri Basic, il Superbasic ammette le istruzioni **ON GOTO e ON GOSUB**. Queste istruzioni trasferiscono l'esecuzione del programma ai numeri di riga determinati in base al valore di una variabile.

sintassi: **ON** *variabile* **GOTO** *espressione *[,espressione]**
ON *variabile* **GOSUB** *espressione *[,espressione]**

esempi; i. **ON x GOTO 10,20,30**
ii. **ON variabile GOSUB 1000,2000**

commento

In Superbasic è preferibile utilizzare la struttura **SELECT**

OPEN
OPEN_IN
OPEN_NEW

unità e microdrive

OPEN consente di associare un canale logico ad una unità fisica per operazioni di input e output.

OPEN_IN apre un file già esistente sul Microdrive e **OPEN_NEW** crea un nuovo file sul Microdrive.

sintassi: *canale= #espressione_numerica*
OPEN *canale, unità_nome file*

esempi:
i. **OPEN #5, nome$**
ii. **OPEN_IN #9, "mdv1_nome"**
iii. **OPEN_NEW #7, mdv1_file**
iv. **OPEN #6, con_10x20a20x20_32**
{associa il canale #6 al video, creando una finestra di 10x20 pixel nella posizione 20x20, con un buffer di 32 caratteri}

OVER

finestre

OVER determina la sovrapposizione della stampa nella finestra associata al canale specificato o di default. La scelta effettuata è attiva fino alla successiva istruzione **OVER**.

sintassi: *valore=espressione_numerica* {—1, 0, 1}
OVER*[canale,] valore*

dove valore=0 disattiva la sovrapposizione
valore=1 stampa nel colore di INK sovrapponendosi al colore di fondo che è trasparente
valore=-1 stampa nel colore di INK, sovrapponendosi al colore di fondo, senza cancellare

esempi: i.
```
10 REMark ombreggiatura
11 PAPER 7 : INK 0 : OVER 1 : CLS
12 CSIZE 3,1
13 FOR i=0 TO 10
14    CURSOR i,i
15    IF i=10 THEN INK 2
16    PRINT "ombra"
17 END FOR i
```

PAN
finestre

PAN permette alla finestra specificata di scorrere lungo l'asse orizzontale in entrambi i sensi. È possibile specificare un secondo parametro per consentire lo scorrimento di una parte soltanto del video.

sintassi: *distanza=espressione_numerica*
parte=espressione_numerica

PAN *[canale,] distanza [,parte]*

dove: *parte*=0 intero schermo
parte=3 l'intera riga del cursore
parte=4 caratteri a destra del cursore

esempi:
i. **PAN #2,50** {scorrimento a sinistra di 50 pixel}
ii. **PAN -100** {scorrimento a destra di 100 pixel}
iii. **PAN 50,3** {scorrimento dell'intera riga del cursore a sinistra di 50 pixel}

attenzione

Se è usata un'istruzione **STIPPLE** o il video è in bassa risoluzione è necessario specificare il valore dello scorrimento con multipli di 2 pixel.

PAPER
finestre

PAPER determina il colore di fondo del video (colore usato da **CLS, PAN, SCROLL**). Il colore assegnato con **PAPER** determina anche il colore di **STRIP**.

PAPER cambia il colore di fondo nella finestra associata al canale specificato o di default.

sintassi: **PAPER** *[canale,] colore*

esempi:
i. **PAPER #3,7** {fondo bianco sul canale 3}
ii. **PAPER 7,2** {fondo bianco a strisce rosse}
iii.
```
10 REMark colori e strisce
20 FOR colore=0 TO 7
30         FOR contrasto=0 TO 7
40             FOR striscia=0 TO 3
50                 PAPER colore,contrasto, striscia
60                 SCROLL 6
70             END FOR striscia
80         END FOR contrasto
90 END FOR colore
```

{non funziona con apparecchio televisivo}

PAUSE

PAUSE sospende l'esecuzione di un programma per un periodo di tempo specificato. Il valore del tempo di interruzione è specificato in unità di 16.67 ms. Se non è assegnato alcun parametro, l'esecuzione del programma è sospesa indefinitamente. In questo caso, un qualsiasi input da tastiera consente la ripresa del programma.

sintassi: *tempo=espressione_numerica*

PAUSE [*tempo*]

esempio: i. **PAUSE 50**

PEEK
PEEK_L
PEEK_W

basic

PEEK legge il contenuto dell'indirizzo di memoria specificato.

L'istruzione **PEEK** ha tre forme, che consentono di leggere un byte (8 bit), una parola (16 bit), una parola lunga (32 bit).

sintassi: *indirizzo=espressione_numerica*

PEEK *(indirizzo)* {legge un byte}
PEEK_W *(indirizzo)* {legge una parola}
PEEK_L *(indirizzo)* {legge una parola}

attenzione

Per le istruzioni **PEEK_W e PEEK_L**, l'indirizzo specificato deve essere un numero pari.

PENUP
PENDOWN
grafica

PEN è eseguito solo quando è attiva la "tartaruga grafica".

PENUP non produce alcun effetto sul video.

PENDOWN traccia sul video le linee corrispondenti al cammino della "tartaruga grafica".

Le linee sono tracciate nella finestra associata al canale specificato o di default. Le linee sono disegnate nel colore assegnato con **INK**.

sintassi: **PENUP** [*canale*]
PENDOWN [*canale*]

esempi:

i. **PENUP**
ii. **PENDOWN #2**

PI
funzioni matematiche

PI ritorna il valore numerico della costante matematica "pi-greco" (3.141592654)

sintassi: **PI**

POINT
POINT_R

grafica

POINT disegna un punto nella posizione assegnata, nella finestra associata al canale specificato o di default.

POINT usa il sistema di coordinate grafiche assolute.

POINT_R usa il sistema di coordinate grafiche relative.

Con una sola istruzione **POINT** è possibile disegnare più di un punto.

sintassi: *x = espressione_numerica*
y = espressione_numerica

POINT [*canale*], *parametri* *[,*parametri*]*

esempi:
i. **POINT 246, 128**
ii. **POINT x, x*x**

POKE
POKE_L
POKE_W

basic

POKE consente di cambiare il contenuto dell'indirizzo di memoria specificato.

L'istruzione **POKE** ha tre forme, che consentono di accedere ad un byte (8 bit), ad una parola (16 bit), ad una parola lunga (32 bit).

sintassi: *indirizzo = espressione_numerica*
valore = espressione_numerica

POKE *indirizzo, valore*
POKE_W *indirizzo, valore*
POKE_L *indirizzo, valore*

esempi:
i. **POKE 12245,0** {scrive il valore 0 nell'indirizzo di memoria 12235}
ii. **POKE_L 131072, 12345**

attenzione

Per le istruzioni **POKE_W e POKE_L**, l'indirizzo specificato deve essere un numero pari.

Non è consigliabile usare l'istruzione **POKE** per modificare indirizzi di memoria usati dal Qdos.

PRINT
unità periferiche

PRINT stampa i dati specificati su una qualsiasi unità di output del QL.

sintassi: separatori= | !
| ,
| /
| ;
| **TO** *espressione_numerica*

Elemento= | *espressione*
| *canale*
| *separatore*

PRINT *[*elemento*]*

È permesso l'uso di più separatori. Almeno un separatore deve dividere il valore del canale assegnato dalla lista delle espressioni da stampare.

esempi: i. **PRINT "ciao"** {stampa la parola ciao sulla schermo}
ii. **PRINT #5, "dati",1,2,3,4** {scrive sul file "dati", associato al canale 5, i valori 1,2,3,4,}
iii. **PRINT TO 20; "Questa è la colonna 20"**

separatori

! è un separatore che consente di inserire uno spazio fra le variabili da stampare. Se ciò che deve essere stampato non può essere contenuto per intero sulla riga corrente, il ! effettua un ritorno a capo. Se lo spazio capita all'inizio della riga, è ignorato. ! agisce sull'elemento successivo da stampare, per cui deve essere inserito prima della variabile da stampare.

, è un separatore che consente di spostare la posizione di stampa di otto in otto colonne (partendo dall'inizio della riga).

/ effettua un ritorno a capo.

; è un separatore che permette di stampare il successivo elemento di seguito al precedente.

TO esegue un incolonnamento. **TO** seguito da un'espressione numerica consente di stampare l'elemento specificato nella colonna corrispondente al valore dell'espressione numerica. Se il valore dell'espressione numerica è al di fuori dei limiti accettati dall'unità specificata, **TO** non produce alcun effetto.

RAD
funzioni matematiche

RAD converte un angolo in gradi nel corrispondente valore in radianti.

sintassi: **RAD** *espressione_numerica*

esempi: i. **PRINT RAD(180)** {stampa 3,141593}

RANDOMISE

funzioni matematiche

RANDOMISE ripristina il generatore di numeri casuali. È possibile specificare un parametro con la funzione di nuovo generatore di numeri casuali.

sintassi: **RANDOMISE** [*espressione_numerica*]

esempi:
i. **RANDOMISE**
ii. **RANDOMISE 3.3255**

RECOL

finestre

RECOL ricolora i singoli pixel della finestra associata al canale specificato o di default.
I parametri specificati sono, nell'ordine, i codici di colore con cui sono ricolorati i singoli pixel.
Ad esempio, il primo parametro specifica il nuovo colore dei pixel neri, il secondo il nuovo colore dei pixel blue, ecc.

sintassi:
c0 = nuovo colore per pixel neri
c1 = nuovo colore per pixel blue
c2 = nuovo colore per pixel rossi
c3 = nuovo colore per pixel magenta
c4 = nuovo colore per pixel verdi
c5 = nuovo colore per pixel ciano
c6 = nuovo colore per pixel gialli
c7 = nuovo colore per pixel bianchi

RECOL [*canale,*]*c0, c1, c2, c3, c4, c5, c6, c7*

esempi:
i. **RECOL 2,3,4,5,6,7,1,0**
{ricolora blue con magenta, rosso con verde, magenta con ciano, ecc.}

REMark
basic

REMark consente di inserire note esplicative all'interno di un programma. Le istruzioni REMark non sono eseguite.

sintassi: **REMark** *testo*

RENUM
basic

RENUM consente di rinumerare una serie di istruzioni di un programma Superbasic. Se non è specificato alcun parametro, **RENUM** rinumera le istruzioni dell'intero programma che iniziano con il numero di riga 100 con passo 10.
Se è specificato il valore della prima istruzione da rinumerare, tutti i numeri di riga precedenti non sono cambiati.
Se sono specificati sia numero iniziale che un numero finale, sono numerate le istruzioni comprese fra i due valori specificati, con il passo assegnato.

sintassi: *inizio = espressione_numerica*
fine = espressione_numerica
uovo inizio = espressione_numerica
passo = espressione_numerica

RENUM [*inizio* **TO**[*fine*];[*nuovo inizio*][,*passo*]

esempi:
i. **RENUM** {rinumera l'intero programma}
ii. **RENUM 100 TO 200** {rinumera le istruzioni dalla 100 alla 200 con passo 10}.

attenzione

RENUM cambia anche eventuali numeri di riga contenuti in istruzioni **GOSUB**, **GOTO**, **IF... THEN**. L'istruzione **RENUM** non deve essere usata all'interno di un programma.

REPeat
END REPeat

iterazioni

REPeat permette di definire cicli iterativi. Esistono due forme di **REPeat** e **END REPeat**: la forma breve e quella lunga.

FORMA BREVE

Nella forma breve **REPeat** e il suo identificatore sono seguiti da una sequenza di istruzioni Superbasic separate dai **:**. **EXIT** riporta il controllo del programma alla successiva riga del programma.

sintassi: **REPeat** *identificatore: istruzioni*

esempi: i. **REPeat attesa: IF INKEY$ <> "" THEN EXIT attesa**

FORMA LUNGA

Nella forma lunga **REPeat** ed il suo identificatore sono le uniche istruzioni contenute sulla riga. Le righe successive contengono una serie di istruzioni concluse da un **END REPeat**.
In questo caso le istruzioni comprese fra **REPeat** e **END REPeat** sono eseguite ripetutamente.

commento

Almeno una delle istruzioni fra **REPeat** e **END REPeat** è l'istruzione **EXIT**.
Nella forma breve non è richiesta l'istruzione **END REPeat**.

RESPR

qdos

RESPR riserva spazio in memoria per ulteriori procedure.

sintassi: *spazio = espressione_numerica*

RESPR *(spazio)*

esempi: i. **PRINT RESPR (1024)**
{stampa l'indirizzo di un blocco di 1024 bytes}

RETurn
funzioni e procedure

RETURN forza la conclusione di una funzione o di una procedura e passa il controllo del programma all'istruzione successiva a quella del richiamo della procedura.

Usata all'interno di una sequenza di istruzioni che definiscono una funzione, **RETURN** restituisce il valore della funzione.

sintassi: **RETURN**[*espressione*]

esempi: i.
```
100 PRINT a(3,3)
110 DEFine FUNCtion a(m,n)
120       IF m = 0 THEN RETURN n + 1
130       IF n = 0 THEN RETURN a(m-1,1)
140       RETURN a(m-1,a(m,n-1)
150 END DEFine
```

ii.
```
10 LET flag = 1
11 LET numero_errore = RND (0 TO 10)
12 test numero_errore
13 DEFine PROCedure test(n)
14    IF flag THEN
15       PRINT "attenzione";
16       SELECT ON n
17          ON n = 1
18             PRINT "microdrive pieno"
19          ON n = 2
20             PRINT "area dati insuf."
21          ON n = REMAINDER
22             PRINT "errore nel programma"
23       END SELECT
24    ELSE
25       RETURN
26    END IF
27 END DEFine
```

commento

Non è obbligatorio avere un'istruzione **RETURN** all'interno di una procedura, perché l'istruzione **END DEFine** chiude in ogni caso la procedura.

L'istruzione **RETURN** è usata anche per riportare il controllo del programma ad un'istruzione successiva ad un **GOSUB**.

RND
funzioni matematiche

RND genera un numero casuale. Possono essere specificati due parametri, che definiscono l'intervallo numerico fra i cui estremi **RND** genera un numero casuale intero.

Se non è specificato alcun parametro, **RND** genera un numero casuale in virgola mobile nell'intervallo 0-1.

sintassi: **RND** ([*espr_numerica*] **TO**[*espr_numerica*])

esempi:
```
i.    PRINT RND (10 TO 20)      {intero fra 10 e 20}
ii.   PRINT RND (1 TO 6)
iii.  PRINT RND (10)            {intero fra 0 e 10}
```

commento

Se non è specificato il primo estremo, **RND** genera un numero casuale fra 0 ed il secondo estremo.

RUN

basic

RUN inizia l'esecuzione di un programma. Se specificato un numero, l'esecuzione del programma comincia da quel numero di riga in avanti.

Sebbene l'istruzione **RUN** possa essere usata all'interno di un programma, è consigliabile utilizzarla come comando diretto.

sintassi: **RUN** [*espressione_numerica*]

esempi:
i. **RUN**
ii. **RUN 10**
iii. **RUN 2*20**

SAVE

unità e microdrive

SAVE salva un programma Superbasic su una qualsiasi unità del QL.

sintassi: riga = | *numero_riga* **TO** *numero_riga* 1
| *numero_riga* **TO** 2
| **TO** *numero_riga* 3
| *numero_riga* 4
| 5

SAVE *unità* *[,*riga*]*

dove: 1 salva tutte le istruzioni fra gli estremi specificati
2 salva tutte le istruzioni fra il numero di riga specificato e l'ultima
3 salva tutte le istruzioni fra la prima ed il numero di riga specificato
4 salva il numero di riga specificato
5 salva l'intero programma

esempi:
i. **SAVE mdv1_programma, 20 TO 70**
ii. **SAVE mdv2_programma, 10, 20, 40**
iii. **SAVE ser1**
{stampa il programma in memoria sulla stampante collegata alla porta seriale.}

SBYTES
unità e microdrive

SBYTES consente di salvare un'area di memoria su una unità del QL.

sintassi: *indirizzo = espressione_numerica*
lunghezza = espressione_numerica

SBYTES *unità, indirizzo, lunghezza*

esempi: i. **SBYTES mdv1_programma, 50000, 10000**
{salva sul file programma il contenuto della memoria dall'indirizzo 50000 per 10000 bytes}.

SCALE
grafica

SCALE determina il fattore di scala usato dalle procedure grafiche. Il valore assunto automaticamente è 100. **SCALE** consente anche di determinare l'origine del sistema di coordinate grafiche nella finestra associata al canale specificato o di default.

sintassi: *x = espressione_numerica*
y = espressione_numerica

origine = x,y
scala = espressione_numerica

SCALE *[canale,] scala, origine*

esempi: i. **SCALE 0.5, 0.1, 0.1** {assegna l'origine delle coordinate al punto 0.1, 0.1 con un fattore di scala 0.5}

SCROLL
finestre

SCROLL permette alla finestra specifica di scorrere lungo l'asse orizzontale in entrambi i sensi. È possibile specificare un secondo parametro, per consentire lo scorrimento di una parte soltanto del video.

sintassi: *distanza = espressione_numerica*
parte = espressione_numerica

SCROLL [*canale*,] *distanza* [,*parte*]

dove: *parte* = 0 intero schermo
parte = 1 dalla riga del cursore in alto
parte = 2 dalla riga del cursore in basso

esempi:
i. **SCROLL #2,50** {scorrimento in basso di 50 pixel}
ii. **SCROLL -100** {scorrimento in alto di 100 pixel}
iii. **SCROLL 10,2** {scorrimento verso il basso della parte inferiore del video di 10 pixel}

SDATE
orologio

SDATE consente di regolare l'orologio del QL.

sintassi: *anno = espressione_numerica*
mese = espressione_numerica
giorno = espressione_numerica
ora = espressione_numerica
minuto = espressione_numerica
secondo = espressione_numerica

SDATE *anno, mese, giorno, ore, minuti, secondi*

esempi:
i. **SDATE 1984,4,2,0,0,0**
ii. **SDATE 1984,1,12,0,0,0**
iii. **SDATE 1984,3,21,0,0,0**

SELect
END SELect

basic

SELECT consente di scegliere fra un certo numero di azioni possibili, dipendenti tutte dal valore di una variabile, desiderata.

definizione: *variabile__select = variabile__numerica*

variabile =		*espressione*
		espressione **TO** *espressione*
elenco =		*variabile* *[,*variabile*]*

forma lunga

SELECT consente di scegliere tra un numero di condizioni a piacere. Al termine delle varie scelte possibili, ognuna delle quali è contraddistinta da un **ON** iniziale, è possibile aggiungere l'istruzione **ON REMAINDER** che consente di effettuare l'ultima scelta per i restanti valori della variabile in esame.

sintassi:
```
SELect ON variabile__select
   *[[ON variabile__select] = select__elenco
        istruzioni]*
   [ON variabile__select] = REMAINDER
        istruzioni
END SELect
```

esempi:
```
100 LET numero__errore = RND (1 TO 10)
110 SELect numero__errore
120    ON numero__errore = 1
130         PRINT "divisione per 0"
140       LET numero__errore = 0
150    ON numero__errore = 2
160         PRINT "file non trovato"
170       LET numero__errore = 0
180    ON numero__errore = 3 TO 5
190         PRINT "file non trovato sul Microdrive"
200       LET numero__errore = 0
210    ON numero__errore = 0
220       PRINT "errore non documentato"
230 END SELect
```

forma breve

Esiste anche una forma breve della struttura **SELect**, che può sostituire in alcuni casi l'istruzione **IF THEN**.

Se la condizione definita nell'istruzione **SELect** è vera, sono eseguite le istruzioni che seguono **SELect**.

sintassi: **SELect ON** *variabile__select=elenco : istruzioni**[:*istruzioni*]**

esempi: i. **SELect ON test = 1 TO 10 : PRINT "risposta entro i limiti ammessi"**

SEXEC

qdos

SEXEC salva il contenuto di una parte di memoria in forma opportuna per essere, in seguito, caricata ed eseguita con un comando **EXEC**.

Il contenuto dell'area di memoria deve essere un programma in codice macchina.

sintassi: *indirizzo = espressione_numerica*
lunghezza = espressione_numerica
spazio = espressione_numerica
{lunghezza dell'area dati che sarà richiesta dal programma}

SEXEC *unità_nome file, indirizzo, lunghezza, spazio.*

esempi: i. **SEXEC mdv1_programma, 262164,3000,500**

SIN

funzioni matematiche

SIN calcola il seno dell'angolo assegnato, espresso in radianti.

sintassi: *angolo = espressione_numerica* {da -10000 a 10000 in radianti}

esempi: i. **PRINT SIN(3)**

SQRT
funzioni matematiche

SQRT calcola la radice quadrata dell'argomento specificato. L'argomento deve essere maggiore o uguale a zero.

sintassi: **SQRT** *(espressione_numerica)* $\{=>0\}$

esempi: i. **PRINT SQRT(3)**

STOP
basic

STOP termina l'esecuzione di un programma e ritorna il controllo al sistema operativo.

sintassi: **STOP**

esempi: i. **IF n = 100 THEN STOP**

È possibile riprendere l'esecuzione del programma con l'istruzione **CONTINUE**.

L'ultima riga eseguibile di un programma funziona come uno **STOP**.

STRIP

finestre

STRIP assegna un colore alla finestra associata al canale specificato o di default.

Il colore di **STRIP** è usato dopo un'istruzione **OVER 1**

sintassi: **STRIP** [*canale,*] *colore*

esempi:
i. **STRIP 7** {fondo bianco}
ii. **STRIP 0,4,2,** {fondo nero e verde}

TAN

funzioni matematiche

TAN calcola la tangente dell'angolo specificato, che deve essere espresso in radianti ed avere un valore compreso fra -30000 e 30000.

sintassi: **TAN** *espressione_numerica* {da -30000 a 30000}

esempi:
i. **PRINT TAN(3)**

TURN
TURNTO
grafica

TURN consente di ruotare la "tartaruga grafica" di un angolo specificato.

TURNTO consente di indirizzare la "tartaruga grafica" nella direzione assegnata.

L'angolo di rotazione è espresso in gradi. Un valore positivo dell'angolo ruota la "tartaruga grafica" in senso antiorario.

sintassi: *angolo = espressione_numerica*

TURN [*canale,*] *angolo*
TURNTO [*canale,*] *angolo*

esempi: i. **TURN 90**
TURNTO 0

UNDER
finestre

UNDER consente di sottolineare con il colore corrente di **INK** tutto ciò che è stampato nella finestra associata al canale specificato o di default.

sintassi: *valore = espressione_numerica* {0 oppure 1}

esempi: i. **UNDER 1** {sottolinea}
ii. **UNDER 0** {non sottolinea}

WIDTH
unità e microdrive

WIDTH permette di assegnare una ampiezza ad una unità che non sia la tastiera (ad esempio stampante o file).

sintassi: *ampiezza = espressione_numerica*

WIDTH [*canale,*] *ampiezza*

esempi: i. **WIDTH #6,72** {assegna una lunghezza di 72 caratteri all'unità (file) associata al canale 6}

WINDOW
finestre

WINDOW consente di definire la posizione e le dimensioni della finestra associata al canale specificato o di default.

Ogni bordo preesistente è cancellato quando è ridefinita una finestra.

Le coordinate sono assegnate usando i pixel come unità di riferimento.

sintassi: *larghezza = espressione_numerica*
lunghezza = espressione_numerica
x = espressione_numerica
y = espressione_numerica

WINDOW [*canale,*] *larghezza, lunghezza, x, y*

esempi: i. **WINDOW 30,40,10,10** {definisce una finestra di 30 per 40 pixel alla posizione 10,10}

INDICE DELLE PAROLE CHIAVE

A

ABS 1
ACOS/ACOT/ASIN/ATAN 1
ADATE 2
Assegnazioni 32
ARC 2
ARCO__cotangente 1
ARCO__tangente 1
ARC__R 2
AT 3
AUTO 3

B

BAUD 4
BEEP 4
BEEPING 5
BLOCK 5
BORDER 6

C

CALL 6
Canali
- CHR$ 7
- CLOSE 8
- CODE 9

CIRCLE 7
CIRCLE__R 7
CLEAR 8
CLOSE 8
CLS 9
CODE 9
Codice macchina
- CALL 6
- EXEC 21
- EXEC__W 21
- SEXEC 54
- Caricamento 21
- RESPR 48

Colori
- INK 27
- MODE 36
- PAPER 41
- RECOL 46

Commenti 47
- REMARK 47

Comunicazioni
- velocità trasmissione 4
- rete locale 38

Condizioni
- IF 26
- SELect 53

CONTINUE 10
COPY 10
COPY__N 10
COS 11
coseno 11
COT 11
cotangente 11
CSIZE 12
CURSOR 12

D

DATA 13
DATE 14
DATE$ 14
DAY$ 14
DEFine 15, 16
DEFine
- FuNction 15
- PROCedure 16

DEG 16
DELETE 17
- file 17
- righe 20

DIM 17
DIMN 18
DIR 18
Directory 18
Disegno
- ARC 2
- CIRCLE 7
- ELLIPSE 7
- FILL 22
- LINE 32
- LINE__R 32

DIV 19
Divisione intera 19
DLINE 19

E

EDIT 20
END
- DEFINE 15, 16
- FOR 24
- IF 26
- REPeat 48
- SELect 53

EOF 20
Errori
- CONTINUE 10
- RETRY 10

EXEC 21
EXEC__W 21
EXIT 21
- FOR 24
- REPEAT 48

EXP 22

F

Files
- COPY 10
- COPY__N 10
- DELETE 17
- DIR 18
- directory 18
- LBYTES 31
- LOAD 33
- Load e run 33
- LRUN 35

MERGE 35
merge e run 37
MRUN 37
OPEN 40
OPEN__IN 40
OPEN__NEW 40
PRINT 45
RUN 50
SAVE 50
FILL 23
FILL$ 22
Finestre
AT 3
BLOCK 5
BORDER 6
CSIZE 12
Dimensione carattere 12
cancellazione 9
controllo cursore 3, 12
FILL 22
FLASH 23
INK 27
MODE 36
OVER 40
PAN 41
PAPER 41
posizione di stampa 3
SCROLL 52
STRIP 56
UNDER 57
WINDOW 58
FLASH 23
FOR 24
EXIT 24
NEXT 24
FUNCTION 15
DEFine 15
RETurn 49
Funzioni matematiche
ABS 1
valore assoluto 1
ACOT 1
arcotangente 1
ATAN 1
arcotangente 1
COS 11
coseno 11
COT 11
EXP 22
esponenziale 22
INT 29
parte intera 29
LN 34
logaritmo naturale 34
LOG 34
logaritmo 34
PI 43
pi-greco 43
RAD 45
conversione in radianti 45
SIN 54
seno 54
SQRT 55
radice quadrata 55
TAN 56
tangente 56

G

GOSUB 25
GOTO 25
Gradi 16
Grafica
ARC 2
ARC__R 2
CIRCLE 7
CIRCLE__R 7
ELLIPSE 7
ELLIPSE__ R 7
FILL 23
LINE 32
LINE__R 32
POINT 44
POINT__R 44
SCALE 51
Tartaruga
FILL 23
MOVE 37
SCALE 51
PENDOWN 43
PENUP 43
TURN 57
TURNTO 57

I

i/e
INKEY$ 27
KEYROW 29
IF 26
nidificazioni 27
INK 27
INKEY$ 27
INPUT 28
INSTR 28
INT 29

K

KEYROW 29

L

LBYTES 31
LEN 31
LET 32
LINE 32
cancellazione 19
editor 20
numerazione 3
re-numerazione 47
LINE__R 32
LIST 33
LN 34
LOAD 33
load e run 35
LOCAL 34
logaritmo 34

LRUN 35
LN 34
LOG10 34

M

Merge e run 37
Microdrives
COPY 10
DELETE 17
Cancellazione files 17
FORMAT 24
formattazione cartucce 24
LOAD 33
SAVE 50
MOD 36
MODE 36
modulo 36
MOVE 37
MRUN 37
Multitasking
PAUSE 42
SEXEC 54

N

NET 38
NEW 38
NEXT 39
FOR 24
REPeat 48
Numeri casuali 49

O

ON GOSUB 39
ON GOTO 39
OPEN 40
canali 40
finestre 40
porte seriali 40
OPEN_IN 40
OPEN_NEW 40
Operatori
INSTR 29
MOD 36
Orologio
ADATE 2
DATE 14
DATE$ 14
DAY$ 14
SDATE 52
OVER 40

P

PAN 41
PAPER 41
Parametri 15, 16
PAUSE 42
PEEK 42
PEEK_L 42
PEEK_W 42
PENDOWN 43
PENUP 43
PI 43
POINT 44
POINT_R 44
POKE 44
POKE_L 44
POKE_W 44
PRINT 45
OVER 40
UNDER 57
Procedure
DEFine 15, 16
LOCal 34
RETurn 49
Programmi
CONTINUE 10
RETRY 10
RUN 50
SAVE 50

R

RAD 45
Radice quadrata 55
RANDOMISE 46
READ 13
RECOL 46
REM 47
REMark 47
RENUM 47
REPeat 48
EXIT 22
NEXT 39
Ripetizioni
FOR 24
NEXT 39
Risoluzione 36
RESPR 48
RESTORE 13
RETE LOCALE 38
RETRY 10
RETurn 49
FUNCTION 15
PROCEDURE 16
RND 49
Routines 16
RS-232-C 4
RUN 50
LRUN 35
load e run 35

S

SAVE 50
SBYTES 51
SCALE 51
SCROLL 52
SDATE 52
SELect 53
SIN 54
seno 54

Suono
BEEP 4
BEEPING 5
SQRT 55
STOP 55
Stringhe
CHR$ 7
FILL$ 23
INSTR 29
LEN 31
lunghezza 31
STRIP 56
subroutines 15, 16

T

TAN 56
Tangente 56
THEN 26
Tempo
regolazione orologio 12
data 42
PAUSE 42
tartaruga grafica
FILL 23
MOVE 37
PENUP 43
PENDOWN 43
TURN 57
TURNTO 57
TURN 57
TURNTO 57

U

UNDER 57
Unità
CLOSE 8
Directory 18
LBYTES 31
LOAD 33
LRUN 35
MERGE 35
MRUN 37
NET 38
OPEN 40
OPEN__IN 40
OPEN__NEW 40
RUN 50
SAVE 50
SBYTES 51

V

Valore assoluto 1
Variabili locali 15, 16
Vettori
DIM 17
DIMN 18
Velocità di trasmissione 4
Video
BLOCK 5
BORDER 6
dimensione caratteri 12
FLASH 23
INK 27
MODE 36
output 45
OVER 40
PAN 41
PAPER 41
PRINT 45
RECOL 46
SCROLL 52
STRIP 56
UNDER 57
WINDOW 58

W

WIDTH 58
WINDOW 58

sinclair

QL

concetti

Il capitolo "CONCETTI" contiene informazioni relative al Superbasic e all'Hardware del QL. Le informazioni sono elencate in ordine alfabetico, e compaiono con il termine più probabile per una loro ricerca.

Quando un esempio contiene un intero programma, è possibile scriverlo ed eseguirlo sul QL.

Un indice al termine del capitolo è di aiuto per trovare il numero di pagina corrispondente all'informazione richiesta.

©1985 SINCLAIR RESEARCH LIMITED
by Stephen Berry *(Sinclair Research Limited)*

accensione

Subito dopo l'accensione (o il RESET), il QL effettua una verifica della RAM, che provoca un'immagine confusa sul video. Se il test non ha segnalato alcun problema, dopo pochi secondi il video del QL appare come nell'immagine riportata di seguito.

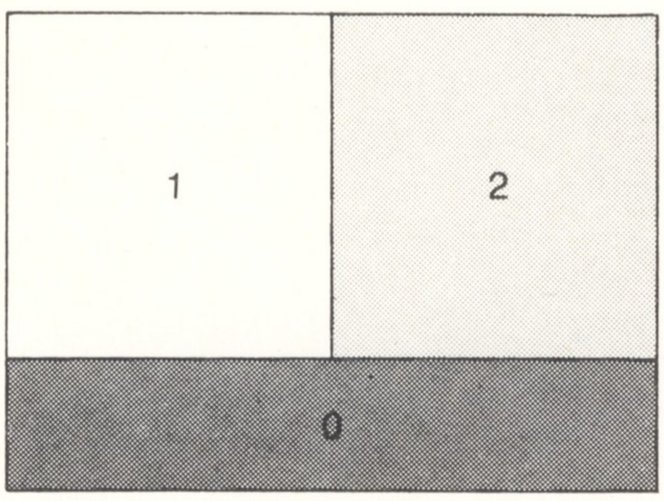


Il QL usa differenti modalità di schermo e di finestre in dipendenza del tasto premuto inizialmente: il tasto **F1** (disponendo di un monitor), oppure il tasto **F2** che predispone il QL ad essere usato con un apparecchio televisivo.

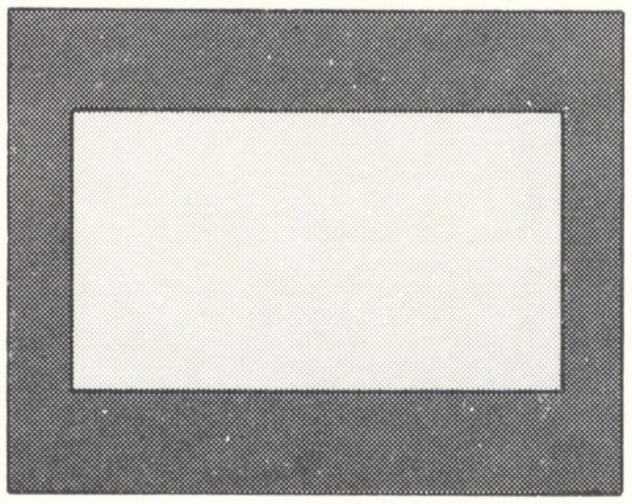
Il QL ha la possibilità di caricare in memoria ed eseguire automaticamente un programma contenuto nell'espansione ROM o sul Microdrive 2.

Se né la ROM né il Microdrive 2 contengono alcun programma, il QL verifica se nel Microdrive 1 è stata inserita una cartuccia. In questo caso, se sulla cartuccia esiste un file chiamato **BOOT**, il QL carica in memoria ed esegue il programma **BOOT**.

Il QL ha tre canali di default che corrispondono alle tre finestre di default.



Monitor


Televisione

Il canale 0 è usato per inserire comandi e visualizzare messaggi di errore. Il canale 1 è usato per l'output di programmi e sul canale 2 compare il listato del programma in memoria.

I canali possono essere modificati e l'output inviato ad un canale particolare con opportuni comandi.

attenzione

È importante non accendere e non eseguire il RESET quando una cartuccia è inserita in un Microdrive. Inserire le cartucce nel Microdrive dopo l'accensione o il RESET, ma prima di premere **F1** o **F2**.

basic

Il Superbasic possiede la maggior parte di funzioni e di istruzioni comuni ad altri Basic. Tuttavia alcune istruzioni sono state incluse nel Superbasic solo per ragioni di compatibilità con gli altri Basic.

GOTO	usare IF, REPeat, ecc.
GOSUB	usare DEFine PROCedure
ON GOTO	usare SELect
ON GOSUB	usare SELect

Spesso può sembrare che il Superbasic non abbia alcuni comandi propri degli altri Basic. Tuttavia questi possono essere sempre ottenuti utilizzando funzioni di carattere più generale. Ad esempio, in Superbasic non esistono le istruzioni **LPRINT** o **LLIST**, ma possono essere facilmente ottenute indirizzando l'output sulla stampante (aprendo il canale corrispondente) ed utilizzando i comandi **PRINT** o **LIST**.

LPRINT	usare PRINT #
LLIST	usare LIST #
VAL	non richiesta dal Superbasic
STR$	non richiesta dal Superbasic
IN	non utilizzabile dal 68008
OUT	non utilizzabile dal 68008

Tutti i Basic usano le funzioni VAL(x$) e STR$(x) per convertire stringhe alfabetiche in numeri e viceversa.

Il Superbasic, dove possibile, attua automaticamente questa conversione per cui le funzioni VAL e STR$ non esistono.

break

È possibile interrompere l'esecuzione di un programma Superbasic o di un comando diretto con la funzione **BREAK**.

BREAK è attivato premendo i tasti:

CTRL

e

SPAZIO

L'esecuzione di un programma interrotta con **BREAK**, può essere ripresa con il comando **CONTINUE**.

canali

Un canale è un mezzo per ricevere o mandare dati e programmi ad una qualsiasi unità del QL. Prima di usare un canale, è necessario aprirlo con il comando **OPEN**. Alcuni canali (#0, #1, #2), sono sempre aperti, perché sono quelli corrispondenti alla tastiera e al video. Un canale è chiuso con l'istruzione CLOSE.

Un canale è identificato da un numero, chiamato numero di canale. Il numero di canale deve essere preceduto dal simbolo #.
Quando è aperto un canale, è possibile associare una unità fisica a quel numero di canale. Un canale può essere identificato semplicemente dal suo numero. Ad esempio:

```
OPEN #5, SER1
```

Questa istruzione associa al canale #5, la porta seriale. Per chiudere il canale #5 è sufficiente scrivere:

```
CLOSE #5
```

I dati possono essere mandati su un determinato canale con un'istruzione **PRINT**. L'istruzione **PRINT** scrive i dati sul canale specificato. Ad esempio:

```
10 OPEN #5, mdv1-test_file
20 PRINT #5, "scrivi sul file test_file"
30 CLOSE #5
```

Questo semplice programma scrive sul file "test_file" il testo riportato fra virgolette nell'istruzione PRINT #5. È importante chiudere un file al termine della fase di scrittura, per avere la sicurezza che i dati siano stati memorizzati.

Allo stesso modo è possibile eseguire l'input di una serie di dati con l'istruzione **INPUT**.

L'istruzione **INKEY$** consente l'input di un carattere alla volta.

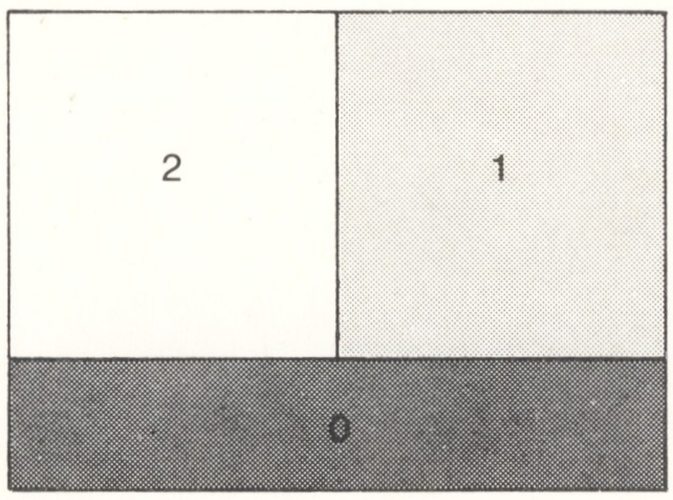
Quando è aperto un canale associato al video, è aperta anche una finestra di dimensioni e forma assegnabili a piacere.

Se diversi canali video sono attivi, è possibile che da più di un canale si richieda contemporaneamente un input.

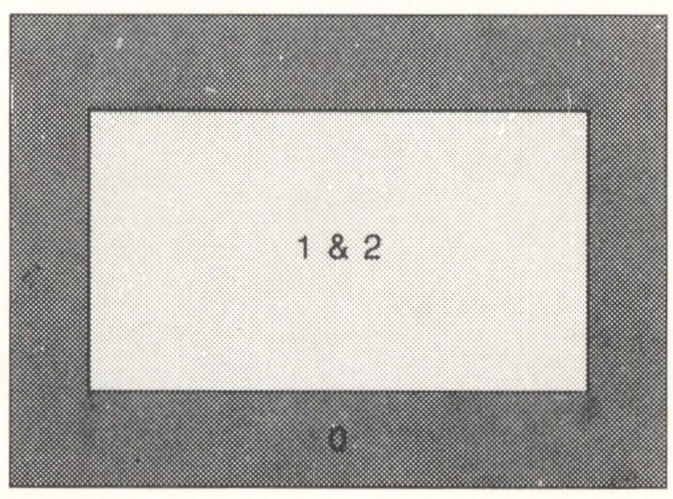
In questo caso è possibile, premendo **CTRL C** passare da un canale all'altro ed indirizzare l'input al canale desiderato. Il cursore nella finestra associata al canale che riceve l'input lampeggia per segnalare la finestra attiva.

Il QL ha tre canali di default, che sono aperti automaticamente. Ad ognuno di questi è associata una finestra sul video:

canale #0 - canale immissione comandi e segnalazione messaggi di errore
canale #1 - canale di default per l'output di programmi
canale #2 - canale in cui compare il listato del programma



Monitor



Televisione

Comandi	Funzione
OPEN	apre il canale per I/E
CLOSE	chiude un canale aperto
PRINT	output ad un canale
INPUT	input da un canale
INKEY$	input da un canale di un carattere

caratteri e tasti

I tasti che controllano il cursore non sono parte del sistema operativo: tuttavia, se un programma vuole controllare il cursore, deve usare i tasti di controllo del cursore e questi tasti non possono essere usati per nessun'altra funzione.

Decimale	Hex	Tasti	Carattere/Funzione
0	00	CTRL ESC	NULL
1	01	CTRL A	
2	02	CTRL B	
3	03	CTRL C	Cambio canale di input
4	04	CTRL D	
5	05	CTRL E	
6	06	CTRL F	
7	07	CTRL G	
8	08	CTRL H	
9	09	TAB (CTRL I)	Campo successivo
10	0A	ENTER (CTRL J)	Nuova istruzione
11	0B	CTRL K	
12	0C	CTRL L	
13	0D	CTRL M	Enter
14	0E	CTRL N	
15	0F	CTRL 0	
16	10	CTRL P	
17	11	CTRL Q	
18	12	CTRL R	
19	13	CTRL S	
20	14	CTRL T	
21	15	CTRL U	
22	16	CTRL V	
23	17	CTRL W	
24	18	CTRL X	
25	19	CTRL Y	
26	1A	CTRL Z	
27	1B	ESC	
28	1C	CTRL SHIFT u grave	
29	1D	CTRL SHIFT e grave	
30	1E	CTRL SHIFT .	
31	1F	CTRL SHIFT ESC	
32	20	Space	Spazio
33	21	SHIFT o grave	!
34	22	SHIFT o grave	!
35	23	#	#
36	24	$	$
37	25	SHIFT u grave	%
38	26	SHIFT $	&
39	27	'	'
40	28	(	(
41	29	)	)
42	2A	*	*
43	2B	SHIFT -	+
44	2C	,	,
45	2D	—	—
46	2E	SHIFT;	.
47	2F	SHIFT:	/
48	30	SHIFT a grave	0
49	31	SHIFT #	1
50	32	SHIFT e acuta	2
51	33	SHIFT "	3
52	34	SHIFT '	4
53	35	SHIFT (	5
54	36	SHIFT_	6
55	37	SHIFT e grave	7
56	38	SHIFT ^	8
57	39	SHIFT c cediglia	9
58	3A	:	:
59	3B	;	;
60	3C	<	<
61	3D	SHIFT i grave	=
62	3E	SHIFT <	>
63	3F	SHIFT ,	?

I codici fino a 20 hex (esadecimale) sono caratteri di controllo. I codici CO-DF controllano i tasti cursore.

Decimale	Hex	Tasti	Carattere/Funzione
64	40	CTRL__	
65	41	SHIFT A	A
66	42	SHIFT B	B
67	43	SHIFT C	C
68	44	SHIFT D	D
69	45	SHIFT E	E
70	46	SHIFT F	F
71	47	SHIFT G	G
72	48	SHIFT H	H
73	49	SHIFT I	I
74	4A	SHIFT J	J
75	4B	SHIFT K	K
76	4C	SHIFT L	L
77	4D	SHIFT M	M
78	4E	SHIFT N	N
79	4F	SHIFT O	O
80	50	SHIFT P	P
81	51	SHIFT Q	Q
82	52	SHIFT R	R
83	53	SHIFT S	S
84	54	SHIFT T	T
85	55	SHIFT U	U
86	56	SHIFT V	V
87	57	SHIFT W	W
88	58	SHIFT X	X
89	59	SHIFT Y	Y
90	5A	SHIFT Z	Z
91	5B	CTRL c cediglia	[
92	5C	SHIFT)	/
93	5D	CTRL a grave	]
94	5E	^	↑
95	5F	_	←
96	60	CTRL e grave	£
97	61	A	a
98	62	B	b
99	63	C	c
100	64	D	d
101	65	E	e
102	66	F	f
103	67	G	g
104	68	H	h
105	69	I	i
106	6A	J	j
107	6B	K	k
108	6C	L	l
109	6D	M	m
110	6E	N	n
111	6F	O	o
112	70	P	p
113	71	Q	q
114	72	R	r
115	73	S	s
116	74	T	t
117	75	U	u
118	76	V	v
119	77	W	w
120	78	X	x
121	79	Y	y
122	7A	Z	z
123	7B	CTRL)	{
124	7C	CTRL .	|
125	7D	CTRL —	}
126	7E	CTRL <	~
127	7F	SHIFT ESC	Copyright
128	80	CTRL $	a dieresi (ä)
129	81	CTRL SHIFT #	ã
130	82	CTRL SHIFT '	a circolare
131	83	e acuta	e acuta
132	84	CTRL SHIFT "	o dieresi (ö)
133	85	CTRL SHIFT (	õ
134	86	CTRL SHIFT__	o barrata
135	87	CTRL '	u dieresi (ü)
136	88	c cediglia	c cediglia (ç)
137	89	CTRL SHIFT a grave	ñ

Decimale	Hex	Tasti	Carattere/Funzione
138	8A	CTRL SHIFT c cedilla	ae (dittongo)
139	8B	CTRL SHIFT i grave	oe (dittongo)
140	8C	CTRL *	a acuta
141	8D	a grave	a grave
142	8E	CTRL SHIFT o grave	a circonflessa
143	8F	CTRL :	e dieresi (ë)
144	90	e grave	e grave
145	91	CTRL #	e circonflessa
146	92	CTRL SHIFT e acuta	i dieresi (ï)
147	93	CTRL "	i acuta
148	94	i grave	i grave
149	95	CTRL SHIFT ;	i circonflessa
150	96	CTRL SHIFT :	o acuta
151	97	o grave	o grave
152	98	CTRL SHIFT Z	o circonflessa
153	99	CTRL SHIFT V	u acuta
154	9A	u grave	u grave
155	9B	CTRL ;	u circonflessa
156	9C	CTRL SHIFT ,	doppia S tedesca
157	9D	CTRL i grave	simbolo percentuale
158	9E	CTRL ,	simbolo Yen
159	9F	CTRL SHIFT *	apostrofo rovesciato
160	A0	CTRL SHIFT <	Ä
161	A1	CTRL SHIFT A	Ã
162	A2	CTRL SHIFT B	A circolare
163	A3	CTRL SHIFT C	E acuta
164	A4	CTRL SHIFT D	O dieresi (Ö)
165	A5	CTRL SHIFT E	Õ
166	A6	CTRL SHIFT F	O barrata
167	A7	CTRL SHIFT G	U dieresi (Ü)
168	A8	CTRL SHIFT H	C cedilla
169	A9	CTRL SHIFT I	Ñ
170	AA	CTRL SHIFT J	AE dittongo
171	AB	CTRL SHIFT K	OE dittongo
172	AC	CTRL SHIFT L	alpha
173	AD	CTRL SHIFT M	delta
174	AE	CTRL SHIFT N	theta
175	AF	CTRL SHIFT O	lambda
176	B0	CTRL SHIFT P	mu
177	B1	CTRL SHIFT Q	pi
178	B2	CTRL SHIFT R	phi
179	B3	CTRL SHIFT S	¡
180	B4	CTRL SHIFT T	¿
181	B5	CTRL SHIFT U	?
182	B6	SHIFT *	Simbolo di sezione
183	B7	CTRL SHIFT W	¤
184	B8	CTRL SHIFT X	«
185	B9	CTRL SHIFT Y	»
186	BA	CTRL o grave	grado
187	BB	CTRL u grave	simbolo divisione
188	BC	CTRL e acuta	freccia sinistra
189	BD	CTRL (	freccia destra
190	BE	CTRL SHIFT)	freccia in su
191	BF	CTRL SHIFT -	freccia in giù
192	C0	cursore sinistro	
193	C1	ALT cursore sinistro	
194	C2	CTRL cursore sinistro	
195	C3	CTRL ALT cursore sinistro	
196	C4	SHIFT cursore sinistro	
197	C5	SHIFT ALT cursore sinistro	
198	C6	SHIFT CTRL cursore sinistro	
199	C7	SHIFT CTRL ALT cursore sinistro	
200	C8	cursore destro	
201	C9	ALT cursore destro	
202	CA	CTRL cursore destro	
203	CB	CTRL ALT cursore destro	
204	CC	SHIFT cursore destro	
205	CD	SHIFT ALT cursore destro	
206	CE	SHIFT CTRL cursore destro	
207	CF	SHIFT CTRL ALT cursore destro	
208	D0	cursore su	
209	D1	ALT cursore su	
210	D2	CTRL cursore su	
211	D3	CTRL ALT cursore su	

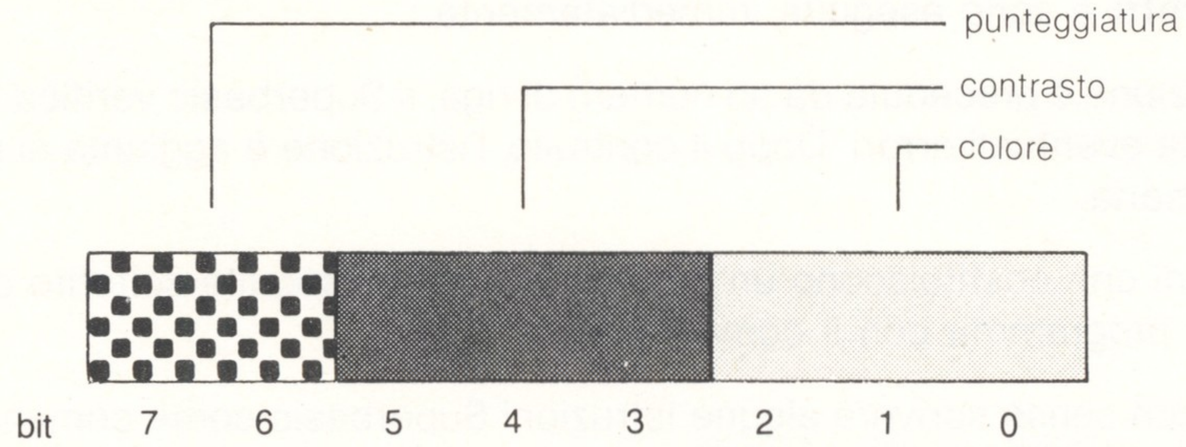
Decimale	Hex	Tasti	Carattere/Funzione
212	D4	SHIFT cursore su	
213	D5	SHIFT ALT cursore su	
214	D6	SHIFT CTRL cursore su	
215	D6	SHIFT CTRL cursore su	
215	D7	SHIFT CTRL ALT cursore su	
216	D8	cursore giù	
217	D9	ALT cursore giù	
218	DA	CTRL cursore giù	
219	DB	CTRL ALT cursore giù	
220	DC	SHIFT cursore giú	
221	DD	SHIFT ALT cursore giù	
222	DE	SHIFT CTRL cursore giù	
223	DF	SHIFT CTRL ALT cursore giù	
224	E0	CAPSLOCK	
225	E1	ALT CAPSLOCK	
226	E2	CTRL CAPSLOCK	
227	E3	ALT CTRL CAPSLOCK	
228	E4	SHIFT CAPSLOCK	
229	E5	SHIFT ALT CAPSLOCK	
230	E6	SHIFT CTRL CAPSLOCK	
231	E7	SHIFT CTRL A CAPSLOCK	
232	E8	F1	
233	E9	CTRL F1	
234	EA	SHIFT F1	
235	EB	CTRL SHIFT F1	
236	EC	F2	
237	ED	CTRL F2	
238	EE	SHIFT F2	
239	EF	CTRL F2	
240	F0	F3	
241	F1	CTRL F3	
241	F2	SHIFT F3	
242	F3	CTRL SHIFT F3	
243	F4	F4	
245	F5	CTRL F4	
246	F6	SHIFT F4	
247	F7	CTRL SHIFT F4	
248	F8	F5	
249	F9	CTRL F5	
250	FA	SHIFT F5	
251	FB	CTRL SHIFT F5	
252	FC	SHIFT spazio	
253	FD	SHIFT tab	
254	FE	SHIFT ENTER	
255	FF	Leggere nota successiva	

Il tasto **ALT**, usato in combinazione con tasti diversi dai tasti cursore o **CAPSLOCK**, genera il codice FF, seguito da un byte che indica quale sarebbe stato il codice se non fosse stato premuto **ALT**.

Non è possibile individuare un CTRL C senza modificare le variabili del sistema.

colore

I colori del QL possono essere sia pieni che punteggiati. È possibile definire un colore con tre parametri: il colore, il colore di contrasto e la punteggiatura.



Se è specificato un solo parametro, questo determina anche gli altri due. Il colore principale è contenuto negli ultimi tre bit del byte che definisce il colore. I successivi tre bit contengono il colore di contrasto e gli ultimi due bit determinano la punteggiatura.

Se non è assegnato alcun valore al parametro che determina la punteggiatura, il QL usa il colore specificato in modo pieno.

colori

I codici dei colori dipendono dalla modalità di schermo in uso:

codice	stato dei bit	composizione	colori	
			8 colori	4 colori
0	0 0 0		nero	nero
1	0 0 1	blue	blue	nero
2	0 1 0	rosso	rosso	rosso
3	0 1 1	rosso+blue	magenta	rosso
4	1 0 0	verde	verde	verde
5	1 0 1	verde+blue	ciano	verde
6	1 1 0	verde+rosso	giallo	bianco
7	1 1 1	verde+rosso+blue	bianco	bianco

punteggiatura

La punteggiatura (**STIPPLE**) dei colori può essere usata allo stesso modo di un colore pieno, ma non produce alcun effetto su un apparecchio televisivo.
I quattro tipi di punteggiatura (stipple) sono i seguenti:



Stipple 0 Stipple 1 Stipple 2 Stipple 3

Il default è stipple 3.

comandi diretti

Il Superbasic distingue le istruzioni precedute da un numero di riga da quelle scritte senza alcun numero di riga. Le istruzioni senza numero di riga sono chiamate **Comandi Diretti** e sono eseguite immediatamente.

Se un'istruzione è preceduta da un numero di riga, il Superbasic verifica la sintassi e segnala eventuali errori. Dopo il controllo, l'istruzione è aggiunta al programma in memoria.

Le istruzioni che costituiscono un programma sono eseguite soltanto quando è eseguito il programma con il comando **RUN**.

Non ha alcun senso scrivere alcune istruzioni Superbasic come comandi diretti, ad esempio **END FOR, END DEFine**, ecc.

Comandi	Funzione
RUN	esegue un programma
NEW	cancella la memoria
CLEAR	cancella le variabili
FORMAT	formatta una cartuccia Microdrive
SAVE	memorizza un programma
LOAD	carica un programma
MERGE	unisce 2 programmi o file
LRUN	carica ed esegue un programma
LIST	lista un programma
RENUM	rinumera un programma
DLINE	cancella linee del programma
PRINT	stampa
EXEC	esegue un lavoro
EDIT	modifica un programma
AUTO	numera le istruzioni
COPY	copia file fra unità

comunicazioni RS-232-C

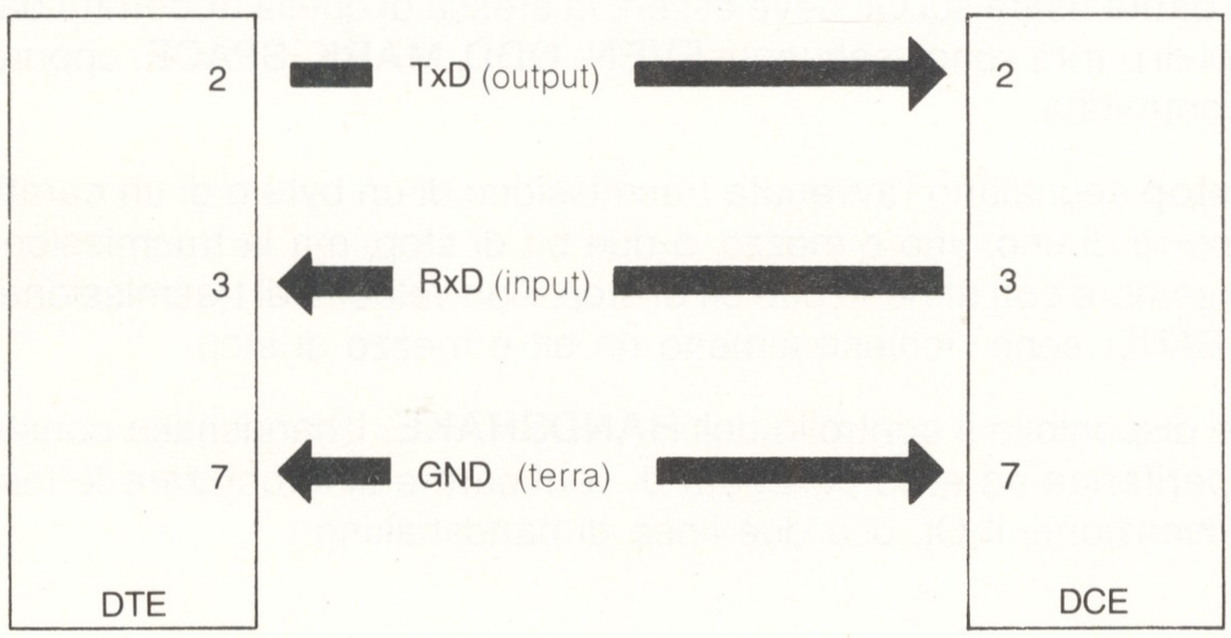
Il QL ha due porte seriali (chiamate SER1 e SER2) per consentire collegamenti con periferiche che usino uno standard **RS-232-C**.

Lo standard **RS-232-C** originariamente aveva la funzione di permettere ai computer di scambiare dati lungo linee telefoniche, usando un modem. Tuttavia lo standard è largamente diffuso anche per collegare periferiche come stampanti, plotter, floppy disk ai computer.

Spesso, anche per un esperto, è molto difficile collegare due unità attraverso interfacce RS-232-C.

Lo standard RS-232-C raggruppa due tipi di interfacce:

1. **DTE** (DATA TERMINAL EQUIPMENT)
2. **DCE** (DATA COMMUNICATION EQUIPMENT)



La figura precedente mostra come l'interfaccia DTE trasmetta dati attraverso il PIN 2, mentre la DCE riceve dati sul PIN 2. Allo stesso modo l'interfaccia DTE riceve dati sul PIN 3, mentre la DCE trasmette dati dal PIN 3.

In alcuni computer l'interfaccia RS-232-C è configurata come una DCE, mentre in altri come una DTE.

Per ovviare a questa difficoltà, il QL ha due porte seriali. La SER1 è configurata come una DCE, mentre la SER2 come una DTE. In questo modo è sempre possibile collegare il QL ad un'altra periferica che usi uno standard RS-232-C, attraverso l'opportuna porta seriale. La configurazione delle due porte è di seguito riportata.

SER1

PIN	NOME	FUNZIONE
1	GND	segnale di terra
2	TxD	segnale di input
3	RxD	segnale di output
4	DTR	segnale di terminale pronto alla ricezione
5	CTS	segnale di terminale pronto alla trasmissione
6	—	+12 V

SER2

PIN	NOME	FUNZIONE
1	GND	segnale di terra
2	TxD	segnale di output
3	RxD	segnale di input
4	DTR	segnale di terminale pronto alla trasmissione
5	CTS	segnale di terminale pronto alla ricezione
6	—	+12V

Dopo che una periferica è stata collegata all'opportuna porta, è necessario che la velocità di trasmissione sia la stessa per entrambe le apparecchiature. Le velocità di trasmissione selezionabili sul QL sono le seguenti:

75
300
600
1200
2400
4800
9600
19200 (solo in trasmissione) BAUD (1 bit/sec)

La velocità di trasmissione sul QL è determinabile con l'istruzione **BAUD. BAUD** assegna la velocità di trasmissione ad entrambe le porte, perché queste non sono indipendenti.

Il tipo di parità usata sul QL deve essere la stessa di quella accettata dalla periferica. I tipi di parità sono i seguenti: **EVEN, ODD, MARK, SPACE**, oppure la parità non è controllata.

I **bit di stop** segnalano l'avvenuta trasmissione di un byte o di un carattere. Il QL può disporre di uno, uno e mezzo, o due bit di stop, ma la trasmissione dei dati avviene sempre con almeno due bit di stop. Con velocità di trasmissione superiori a 9600 BAUD, sono richiesti almeno un bit e mezzo di stop.

Sul QL è disponibile il controllo dell'**HANDSHAKE**. L'handshake consente al QL ed alla periferica ad esso collegata di verificare e sincronizzare le loro velocità di comunicazione. Il QL usa due linee di handshaking:

CTS
DTR

Se il DTE non può sostenere la velocità di trasmissione dei dati chiude la linea DTR ed impedisce al DCE di trasmettere dati. Naturalmente quando il DTE ha sincronizzato la sua velocità, il DCE è riattivato attraverso la linea DTR e la trasmissione riprende. Allo stesso modo il DCE può fermare, negando la linea CTS, la ricezione dei dati del DTE.

Sebbene trasmettere e ricevere dati col QL sia sempre possibile senza utilizzare un protocollo handshaking, il QL non riceve dati in modo corretto se non è stata utilizzata la linea CTS sulla porta SER1 e la linea DTR sulla porta SER2

La trasmissione dati sul QL avviene sempre in **"full duplex"**. Questo significa che il QL può ricevere e trasmettere dati allo stesso tempo.

Il tipo di parità e di handshaking sono determinati quando è aperto il canale seriale.

Comandi	Funzione
BAUD	seleziona la velocità di trasmissione
OPEN	apre le porte seriali
CLOSE	chiude le porte seriali

confronto fra stringhe

Di seguito è riportata la successione ordinata dei caratteri numerici e alfabetici del QL:

spazio
!"'()* +,/:;< = >?∂[/]î—(:)
lettere o numeri in ordine numerico
AaBbCcDdEeFfGgHhIiJjKkLlMmNnOoPpQqRrSsTtUuVvWwXxYyZz

e altri caratteri non stampabili.

Le relazioni logiche fra due stringhe possono essere le seguenti:

uguale: tutti i caratteri o i numeri sono gli stessi

minore: il primo carattere della prima stringa è precedente al primo carattere della seconda stringa nell'ordine sopra definito

maggiore: il primo carattere della prima stringa è successivo al primo carattere della seconda stringa nell'ordine sopra definito.

I confronti fra stringhe contenenti caratteri non stampabili possono dare risultati imprevisti.

conversione o coercizione

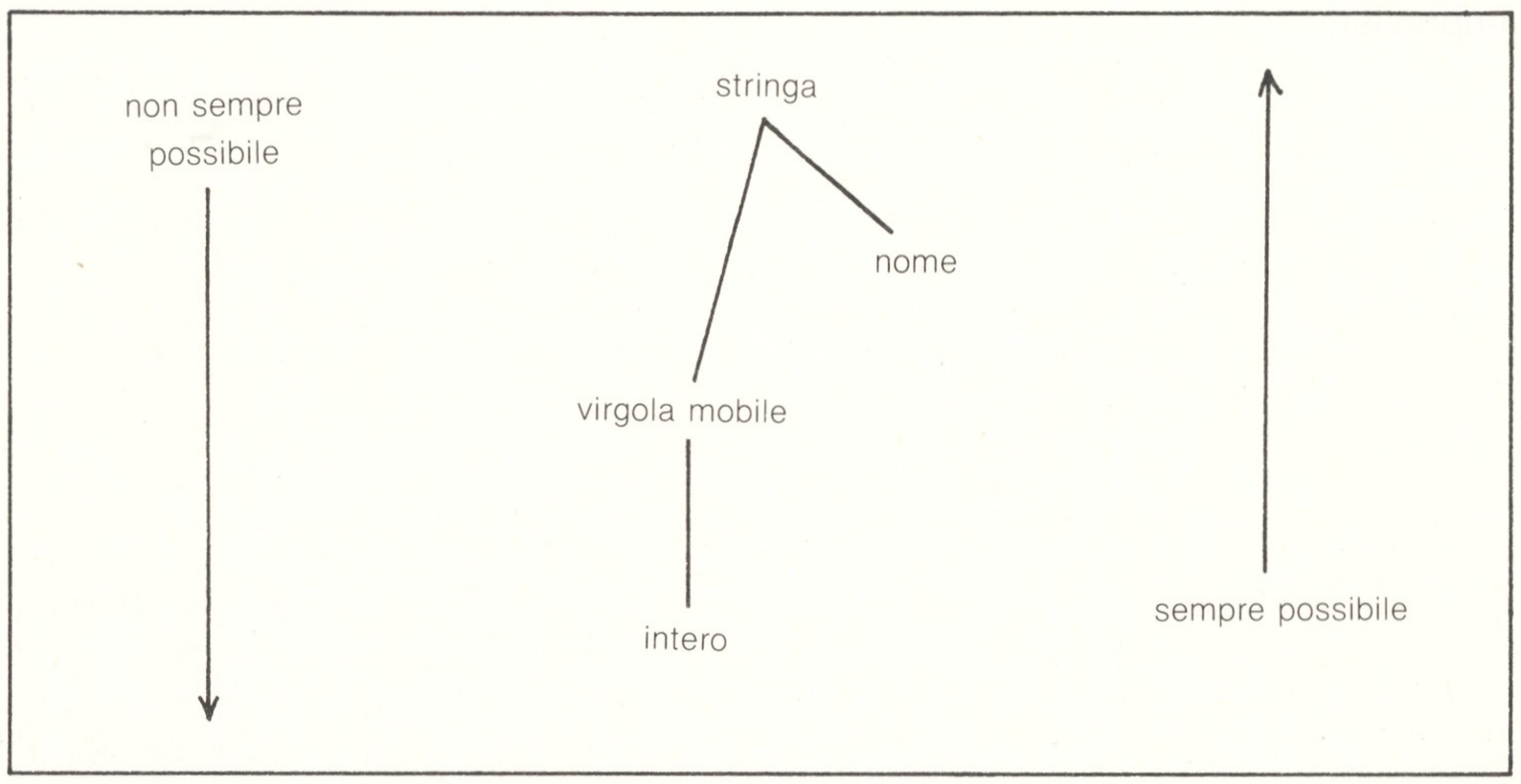
Quando è necessario il Superbasic converte tipi di dati differenti fra loro.

L'operatore usato determina il tipo di conversione richiesta. Per esempio se un'operazione richiede una stringa ed invece è stata specificata una variabile numerica, il Superbasic converte la variabile numerica in stringa. Se la conversione non è possibile, il Superbasic segnala un errore.

Anche i parametri di funzioni o procedure possono subire lo stesso tipo di conversione.
Ad esempio, il comando Superbasic **LOAD** richiede come parametro un nome, ma può essere anche accettata una stringa che è convertita in nome.

Esiste un ordinamento naturale dei tipi di dati, riportato nella figura.

Una stringa è il tipo più generale di dato, perché può rappresentare nomi, numeri interi e numeri in virgola mobile. I numeri in virgola mobile non sono tipi di dati così generali come le stringhe, ma sono più generale dei numeri interi, perché un numero intero può essere sempre rappresentato come un numero in virgola mobile. La conversione dei dati può sempre avvenire dall'alto verso il basso, come mostrato nella figura, ma non è sempre possibile il cammino contrario.



esempio

a=b + c	non è richiesta alcuna conversione prima di eseguire l'addizione
a%=b + c	non è richiesta alcuna conversione prima di eseguire l'operazione, ma il risultato è convertito in numero intero prima che il valore sia assegnato alla variabile a%
a$=b$ + c$	il contenuto di b$ e c$ è convertito in numero a virgola mobile, se possibile, prima di eseguire l'addizione. Il risultato è poi convertito in stringa prima che il valore sia assegnato alla variabile a$
LOAD "mdv1_data"	la stringa **"mdv1_data"** è convertita in nome prima che sia eseguito il comando **LOAD**

convenzioni sintattiche

La sintassi del Superbasic è stata definita usando i seguenti tipi di notazioni:

il simbolo | [indica di scegliere uno degli elementi contenuti all'interno

il simbolo [ ] indica che gli elementi contenuti all'interno sono facoltativi

il simbolo * * indica che è possibile una ripetizione degli elementi contenuti all'interno

il simbolo { } indica un commento

esempi: | A | B | indica di scegliere A oppure B

[A] indica che A è facoltativo

A indica che è possibile una ripetizione di elementi come A

errori

Gli errori in Superbasic sono segnalati nel seguente formato standard:

alla linea numero__riga numero__errore testo,

dove il numero di linea è il numero dell'istruzione che ha provocato l'errore e il testo dell'errore è uno fra quelli riportati di seguito.

1. **Operazione interrotta**
   È stata interrotta un'operazione con **BREAK**

2. **Lavoro non valido**
   È stato trovato un errore relativo a funzioni di sistema che controllano il multitasking o l'I/E

3. **Superata la capacità di memoria**
   Lo spazio disponibile in memoria è insufficiente

4. **Fuori dimensioni**
   È stato utilizzato un indice di vettore scorretto

5. **Buffer pieno**
   Un'operazione di I/E ha riempito il buffer prima che sia stato trovato un indicatore di fine record

6. **Canale non aperto**
   Si è cercato di utilizzare un canale non ancora aperto

7. **File non trovato**
   È stato fatto riferimento ad un file o ad una unità non presenti, oppure il Superbasic non ha trovato un identificatore. In questo caso l'errore è stato provocato da una scorretta nidificazione di cicli o procedure

8. **File già esistente**
   Si vuole assegnare ad un file o ad una unità un nome già usato

9. **File ancora aperto**
   Si è cercato di riaprire un file o un canale aperti in precedenza

10. **Fine del file**
    È stato intercettato un indicatore di fine file durante un'operazione di input

11. **Disco pieno**
    Tutto lo spazio su Microdrive o su disco è occupato

12. **Comando scorretto**
    È stato trovato un errore nel valore di un parametro. È anche possibile che sia stato usato un nome al di fuori di un contesto corretto. Ad esempio una variabile è stata usata come una procedura

13. **Errore di trasmissione**
    È stato riscontrato un errore di parità nell'interfaccia RS-232-C

14. **Formattazione non riuscita**
    L'operazione di formattazione di una cartuccia o floppy disk non è riuscita. La cartuccia o il floppy sono irrimediabilmente danneggiati

15. **Parametro scorretto**
    È stato individuato un errore nell'elenco dei parametri di una procedura o di una funzione
    È stato fatto un tentativo di leggere i dati da una unità di sola scrittura.

16. **Disco danneggiato**
    È stata inserita una cartuccia danneggiata

17. **Espressione scorretta**
    È stato riscontrato un errore in un'espressione numerica o logica

18. **Eccedenza valori limite**
    È stata tentata una divisione per zero, o una radice quadrata di un numero negativo, ecc.

19. **Non disponibile**

20. **Protetto da scrittura**
Si è tentato di scrivere su un file aperto solamente in lettura

21. **Errore di sintassi**
L'istruzione contiene un errore di sintassi

22. **PROC/FN non definita**
Questo non è un messaggio di errore, ma informa che il programma è stato interrotto e modificato in seguito costringendo il Superbasic a cancellare dalla memoria le procedure o funzioni definite in precedenza.

ripresa programma

Dopo che è stato segnalato un errore, l'esecuzione del programma può essere ripresa dall'istruzione successiva scrivendo:

CONTINUE

Se è possibile la correzione della condizione di errore senza modificare il programma, l'esecuzione può essere ripresa dalla stessa istruzione che ha generato l'errore scrivendo:

RETRY

espansioni periferiche

Il connettore di espansione del QL consente di collegare altre unità.

Di seguito è riportata la configurazione del connettore.

Il connettore ha 64 poli (maschio) ed è del tipo DIN-41612

GND	a	1	b	GND
D3	a	2	b	D2
D4	a	3	b	D1
D5	a	4	b	D0
D6	a	5	b	ASL
D7	a	6	b	DSL
A19	a	7	b	RDWL
A18	a	8	b	DTACKL
A17	a	9	b	BGL
A16	a	10	b	BRL
CLKCPU	a	11	b	A15
RED	a	12	b	RESETCPUL
A14	a	13	b	CSYNCL
A13	a	14	b	E
A12	a	15	b	VSYNCH
A11	a	16	b	VPAL
A10	a	17	b	GREEN
A9	a	18	b	BLUE
A8	a	19	b	FC2
A7	a	20	b	FC1
A6	a	21	b	FC0
A5	a	22	b	A0
A4	a	23	b	ROMOEH
A3	a	24	b	A1
DBGL	a	25	b	A2
SP2	a	26	b	SP3
DSMCL	a	27	b	IPLOL
SP1	a	28	b	BERRL
SP0	a	29	b	IPL1L
VP12	a	30	b	EXTINTL
VM12	a	31	b	VIN
VIN	a	32	b	VIN

La 'L' che segue il nome di un segnale indica che è un segnale a bassa intensità.

segnale	funzione
A0..A19	linee indirizzo 68008
RDWL	lettura/scrittura
ASL	address strobe
DSL	data strobe
BGL	richiesta permesso bus
DSMCL	chip principale
CLKCPU	orologio di CPU
E	orologio periferiche
RED	rosso
BLUE	blue
GREEN	verde
CSYNCL	sincronismo composito
VSYNCH	sincronismo verticale
ROMOEH	abilitazione output ROM
FC0	stato del processore
FC1	stato del processore
FC2	stato del processore
RESETCPUL	reset di CPU

Segnali di output periferici

segnale	funzione
DTACKL	asegnazione bus
BRL	richiesta bus
VPAL	indirizzo periferico corretto
IPLOL	interruttore priorità livello 5
IPL1L	interruttore priorità livello 2
BERRL	errore bus
EXTINTL	interruttore esterno
DBGL	bus garbage collector

segnali di input periferici

segnale	funzione
D0..D7	linee dati

segnali bi-direzionali periferici

segnale	funzione
SP0..SP3	selezione periferiche da 0 a 3
VIN	9 V DC(nominale)__ 500mA al massimo
VM12	-12 V
VP12	+12 V
GND	terra

La descrizione sopra riportata ha la sola funzione di rendere più comprensibile il meccanismo di espansione.

Non è sufficiente per realizzare una unità di espansione.

Al QL possono essere collegate fino a 16 unità periferiche. Una singola unità periferica può essere collegata direttamente allo slot di espansione, mentre per collegare più unità è necessario un modulo di espansione.

In questo contesto con il termine "unità periferica" si intende anche una espansione di memoria. Sebbene le aree della mappa di memoria assegnate all'espansione di memoria siano differenti da quelle assegnate ad unità periferiche, il meccanismo di base è lo stesso. È possibile espandere il QL con un'espansione di memoria per volta. Lo spazio disponibile per espansioni periferiche nella mappa di memoria fisica del QL è di 16 Kbytes per unità.
I 16 Kbytes devono contenere la mappa di memoria di I/E richiesta da ogni unità e il programma di controllo della stessa unità.

La posizione di ogni unità periferica nella mappa globale di memoria del QL è determinata dalla scelta delle linee periferiche SP0, SP1, SP2, SP3. La scelta dell'indirizzo di queste linee genera un segnale corrispondente alla posizione dello slot nel modulo di espansione del QL. Per ogni unità periferica è quindi necessario scegliere un indirizzo di input fra A14, A15, A16, A17, che deve essere lo stesso del segnale proveniente da SP0, SP1, SP2, SP3.

espressioni

Una espressione Superbasic può essere alfabetica, numerica, logica, o una combinazione di queste. Dati incompatibili fra loro sono convertiti dal sistema, quando è possibile.

mono-operatori + — NOT

espressione = [mono-operatori] esp. operatore esp.
(espressione)
atomo

atomo = variabile
costante
funzione [(espressione *[,espressione]*)]
elemento di vettore

variabile = nome variabile
nome variabile %
nome variabile $

funzione = nome funzione
nome funzione %
nome funzione $

costante = cifre *[cifre]*

[cifre] . *[cifre]*

[cifre] [.] *[cifre]* E *[cifre]*

Il valore di una espressione può essere un numero intero, un numero a virgola mobile, o una stringa alfabetica.

All'interno di una espressione possono essere inclusi anche operatori logici. Se l'operazione logica è vera, l'espressione assume il valore 1. In caso contrario assume il valore 0.

file

Tutte le operazioni di input e output avvengono per mezzo di file logici. Esistono vari tipi di file.

I programmi Superbasic, i file di dati o di testo, creati con le istruzioni **PRINT** e **SAVE** sono dello stesso tipo.
È possibile accedere a questi file usando i comandi: **LOAD, INKEY$, INPUT**.

DATI

Esistono poi i file di programmi salvati con un comando **SEXEC** e richiamati con i comandi **EXEC** e **EXEC_W**

EXEC

Ultimo tipo di file, sono quelli creati usando il comando **SBYTES** e richiamabili con il comando **LBYTES**.

CODE

finestre

Le finestre sono aree del video che si comportano come se ognuna di esse fosse uno schermo indipendente. Questo significa che in ogni finestra si possono eseguire operazioni come cancellazione, scrittura, ecc., indipendentemente dalle altre finestre.

Le finestre possono essere associate ad un canale quando il canale è aperto. La forma e le dimensioni delle finestre sono assegnabili a piacere con il comando **WINDOW**.

Ogni finestra può essere circondata da un bordo con il comando **BORDER**.

Operazioni di output possono essere indirizzate ad una qualsiasi finestra specificando il canale ad essa associato.

Anche le operazioni di input possono avvenire da una finestra specificando il canale ad essa associato. Se più di un canale è pronto per l'input, è possibile spostare le operazioni di input da un canale all'altro premendo

CTRL C.

Il cursore lampeggia nella finestra associata al canale che riceve l'input.

Il cursore può essere spostato in una qualsiasi posizione di una finestra con comandi **CURSOR** e **AT**.

Alcuni comandi (**CLS, PAN**, ecc.) accettano dei parametri che permettono di delimitare le loro operazioni ad una parte della finestra. I parametri sono definiti di seguito.

parte	descrizione
0	intero schermo
1	parte superiore ed esclusa la linea del cursore
2	parte inferiore ed esclusa la linea del cursore
3	l'intera linea del cursore
4	linea del cursore destra del cursore

comando	funzione
WINDOW	ridefinisce una finestra
BORDER	disegna un bordo attorno alla finestra
PAPER	definisce il colore di fondo
INK	definisce il colore dei caratteri
STRIP	definisce linee colorate
PAN	sposta in orizzontale una finestra
SCROLL	sposta in verticale una finestra
AT	sposta il cursore in una posizione assegnata
CLS	cancella il contenuto di una finestra
CSIZE	determina la grandezza dei caratteri
FLASH	determina il lampeggiamento dei caratteri
RECOL	ricolora una finestra.

funzioni e procedure

Le funzioni e le procedure sono definite in Superbasic con le istruzioni **DEFine FUNction** e **DEFine PROCedure**.

Una funzione è richiamata semplicemente scrivendo il suo nome in una opportuna espressione Superbasic. Il richiamo alla funzione deve essere incluso in un'espressione, perché ritorna un valore che deve essere usato nell'espressione. Una procedura è richiamata scrivendo il suo nome come prima parola di un'istruzione Superbasic.

I dati possono essere passati all'interno di una funzione o procedura facendo seguire il nome della funzione o della procedura da un elenco di variabili dette "parametri reali".
L'elenco dei parametri è confrontato con le variabili specificate al momento della definizione della funzione o della procedura, chiamate "parametri formali". I parametri formali devono avere la forma di variabili del Superbasic, mentre quelli reali possono essere vettori o espressioni del Superbasic, di cui la variabile è la forma più semplice.

I parametri formali sono utilizzati semplicemente per indicare come devono essere trattati i parametri reali. Non hanno la necessità di avere un particolare tipo di variabile associato ad essi. Gli elementi delle due liste di parametri sono accoppiati in ordine quando la funzione o la procedura sono richiamate e i parametri formali diventano equivalenti a quelli reali. Ci sono tre modi diversi di usare i parametri.

Se il parametro reale è una sola variabile e se è assegnato un valore al parametro formale nella funzione o nella procedura, lo stesso valore è anche assegnato al parametro attuale corrispondente.

Se il parametro reale è un'espressione, l'assegnazione di dati al corrispondente parametro formale non produce alcun effetto all'esterno della procedura.

Se il parametro reale è una variabile che non è stata precedentemente definita, l'assegnazione di un valore al corrispondente parametro formale definisce anche il tipo di parametro reale.

È possibile definire variabili in modo locale all'interno di una funzione o procedura con l'istruzione **LOCal**. Le variabili locali non alterano variabili con lo stesso nome che esistono al di fuori della funzione o procedura. Una variabile locale è disponibile per qualsiasi funzione o procedura richiamata dalla procedura o funzione in cui essa è stata dichiarata , purché la stessa variabile non sia definita più di una volta nella sequenza di richiami.
Le funzioni e le procedure in Superbasic possono essere recursive, cioè possono richiamare se stesse direttamente o indirettamente.

Comandi	Funzione
DEFine FUNction	Definisce una funzione
DEFine PROCedure	Definisce una procedura
RETURN	Lascia una funzione o una procedura oppure ritorna il valore di una variabile da una funzione
LOCal	Definisce una variabile locale in una funzione o procedura

funzioni matematiche

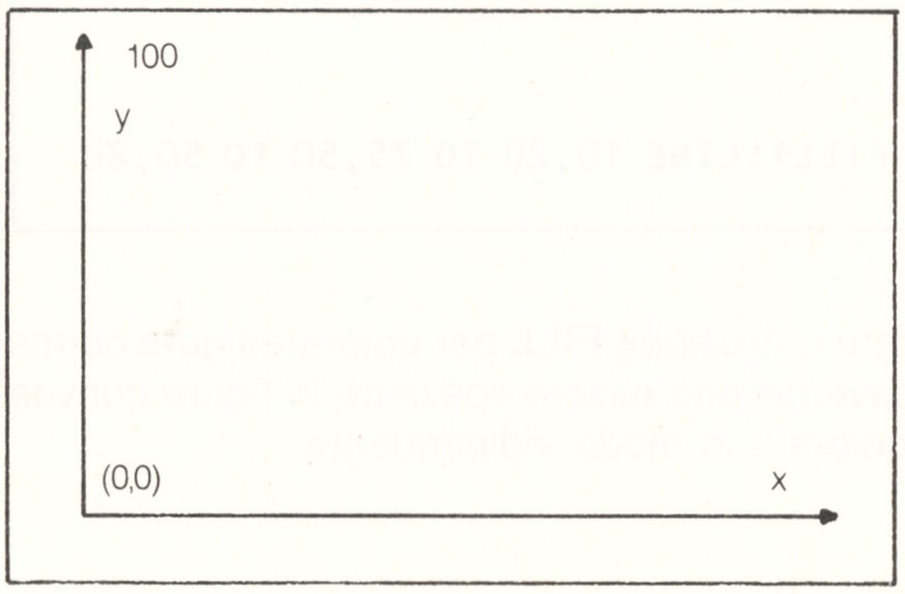
Il Superbasic consente di utilizzare le funzioni matematiche e trigonometriche standard.

Funzione	Nome
COS	coseno
SIN	seno
TAN	tangente
ATAN	arcotangente
ACOT	arcocotangente
ACOS	arcocoseno
ASIN	arcoseno
COT	cotangente
EXP	esponenziale
LN	logaritmo naturale
LOG10	logaritmo in base 10
INT	intero
ABS	valore assoluto
RAD	converte in radianti
DEG	converte in gradi
PI	ritorna il valore di pi-greco
RND	genera un numero casuale
RANDOMISE	ripristina il generatore di numeri casuali

grafica

È importante sottolineare che lo schermo del QL non ha pixel quadrati ma rettangolari e che cambiare la modalità di schermo provoca un cambiamento della forma dei pixel. Di conseguenza se le procedure grafiche fossero state basate semplicemente sull'idea di pixel, gli stessi disegni tracciati nei due modi avrebbero forme diverse. Ad esempio un cerchio in una modalità grafica sarebbe stato un'ellisse nell'altra modalità.

Le procedure grafiche assicurano l'indipendenza dalla modalità grafica scelta di quello che è disegnato sullo schermo. Non è possibile usare il pixel come unità di misura per indicare le dimensioni delle figure. Così le istruzioni grafiche usano una scala ed un sistema di coordinate arbitrarie per determinare la posizione e le dimensioni delle figure. Le istruzioni grafiche usano il sistema di coordinate grafiche, che ha come origine l'angolo inferiore sinistro della finestra specificata o di default. Questa non è la stessa origine di quella usata dal sistema di coordinate a pixel. L'origine grafica consente di utilizzare un sistema di coordinate cartesiane. Le coordinate della posizione del cursore dopo ogni operazione grafica e le operazioni successive possono essere relative alla nuova posizione del cursore o assolute, cioè relative alla origine del sistema di coordinate.



Il Sistema di Coordinate Grafiche

Il fattore di scala è determinato in modo che la dimensione verticale della finestra specificata abbia lunghezza 100. Il fattore di scala può essere variato con il comando **SCALE**. La scala lungo la direzione orizzontale (asse x) è la stessa di quella in verticale. Tuttavia la lunghezza di una linea tracciata lungo la direzione orizzontale dipende dalla forma della finestra. Ingrandire il fattore di scala corrisponde ad ingrandire la dimensione massima della figura che può essere disegnata all'interno di una finestra. Se l'output è indirizzato ad una finestra di dimensioni diverse, tutto il contenuto della finestra è ridimensionato in modo da sfruttare le dimensioni della nuova finestra. Se una figura supera le dimensioni della finestra, è tagliata.

Il comando **SCALE** consente di sceglier una nuova origine.

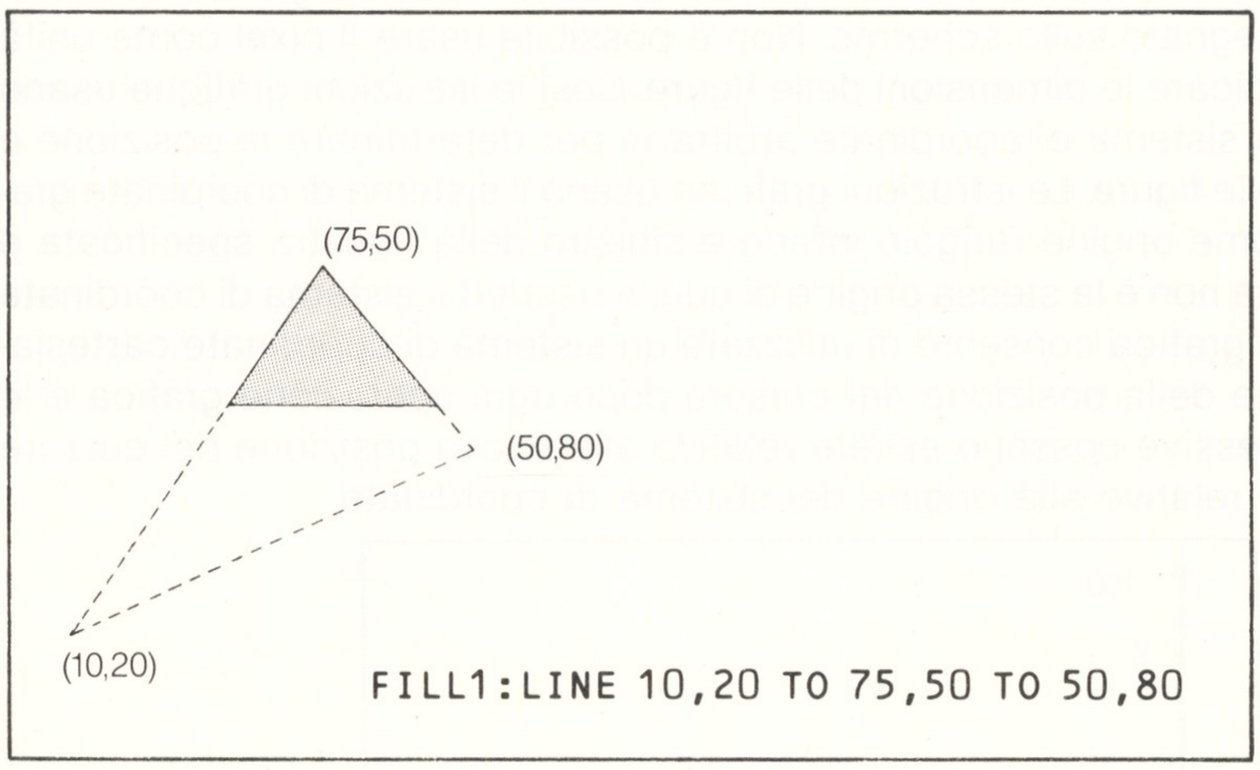
Le istruzioni grafiche tracciano figure nella finestra associata al canale specificato o di default, nel colore selezionato con INK.

Comando	Funzione	
CIRCLE	Disegna un'ellisse o un cerchio	assoluti
LINE	Disegna una linea	
ARC	Disegna un arco di cerchio	
POINT	Disegna un punto	
CIRCLE_R	Disegna un'ellisse o un cerchio	relativi
LINE_R	Disegna una linea	
ARC_R	Disegna l'arco di un cerchio	
POINT_R	Disegna un punto	
SCALE	Determina la scala e l'origine	
FILL	Riempie una figura	
CURSOR	Posiziona il cursore	

riempimento

Anche le figure tracciate con la "tartaruga grafica" possono essere riempite con un colore. Se l'opzione **FILL** è attiva, le figure vengono colorate mentre sono tracciate.

L'istruzione **FILL** memorizza un elenco di punti. Quando una figura si chiude, ci sono due punti su una stessa linea. Questi due punti sono congiunti da un segmento tracciato nel colore corrente di INK e l'operazione si ripete finchè tutta la figura non è colorata. L'istruzione **FILL** deve essere sempre specificata prima di tracciare un nuovo disegno, perché il 'buffer' usato per memorizzare l'elenco di punti sia azzerato e reso nuovamente disponibile. La seguente figura dimostra il funzionamento dell'istruzione **FILL**.



attenzione Non è possibile usare l'istruzione **FILL** per colorare figure concave (con rientranze). Le figure concave devono essere spezzate in figure convesse, ognuna delle quali può essere colorata in modo indipendente.

istruzioni

Un'istruzione Superbasic è un comando che permette al QL di eseguire particolari operazioni. Ad esempio:

```
LET a=2
```

assegna alla variabile a il valore 2.

Su una stessa riga è possibile scrivere più istruzioni, separate dai due punti (:).

Ad esempio:

```
LET a=a + 2 : PRINT a
```

Se un'istruzione non è preceduta da un numero di riga, è chiamata **comando diretto** ed è eseguita immediatamente dal Superbasic.

Se un'istruzione è preceduta da un numero di riga, allora è parte di un programma ed è eseguita quando è eseguito il progamma.

Non ha senso utilizzare alcune istruzioni Superbasic come comandi diretti. Ad esempio: **REM, IF, REPeat**.

iterazioni

Le iterazioni in Superbasic sono controllate da due tipi di costruzioni sintattiche:

REPeat *nome_variabile*
istruzioni
END REPeat *nome_variabile*

FOR *nome_variabile=limiti*
istruzioni
END FOR *nome_variabile*

Altre due istruzioni possono comparire all'interno delle due precedenti costruzioni:

NEXT *nome_variabile*

EXIT *nome_variabile*

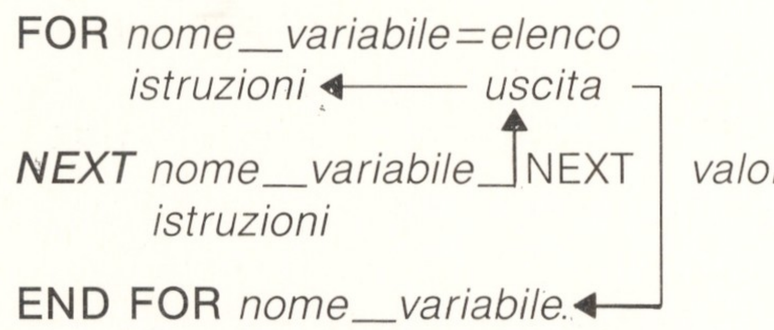
Quando un programma incontra un'istruzione **NEXT**, ritorna il controllo all'istruzione successiva al corrispondente **FOR** o **REPeat**, oppure, se sono stati superati i limiti assegnati nell'istruzione **FOR**, il controllo del programma ritorna all'istruzione successiva a **NEXT**.

Quando il programma incontra un'istruzione **EXIT**, il controllo ritorna all'istruzione successiva all'**END FOR** o all'**END REPeat** corrispondenti.

È sempre necessario avere almeno una istruzione **EXIT** in un ciclo **REPeat** per concludere l'iterazione.

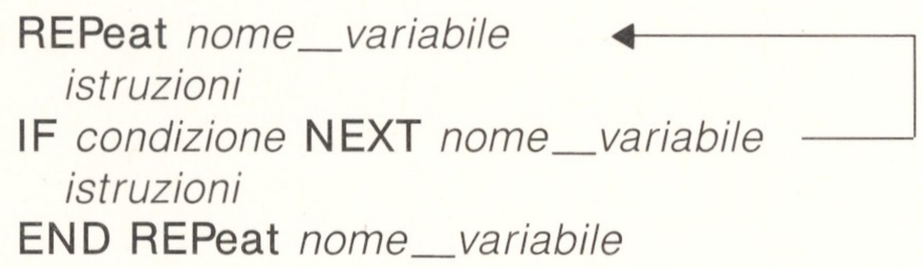
È possibile utilizzare una combinazione di istruzioni **NEXT e EXIT** per forzare il termine di un ciclo iterativo prima dell'istruzione **END**.

Questo significa che, in dipendenza da particolari condizioni logiche, un ciclo iterativo può essere condizionato nella sua esecuzione.



Le istruzioni comprese fra **NEXT e END FOR** sono eseguite soltanto quando il ciclo iterativo è eseguito normalmente.

È possibile avere una costruzione simile anche nel ciclo **REPeat.**



joystick

Le porte dei joystick, **CTL1** e **CTL2** consentono di collegare due joystick al QL. Il movimento dei joystick corrisponde alla pressione di cinque tasti e qualsiasi programma che usa un joystick può anche usare i cinque tasti elencati di seguito:

	CTL1	CTL2
modo	tasto	tasto
su	cursore su	F4
giù	cursore giù	F2
sinistra	cursore sinistro	F1
destra	cursore destro	F3
fuoco	spazio	F5

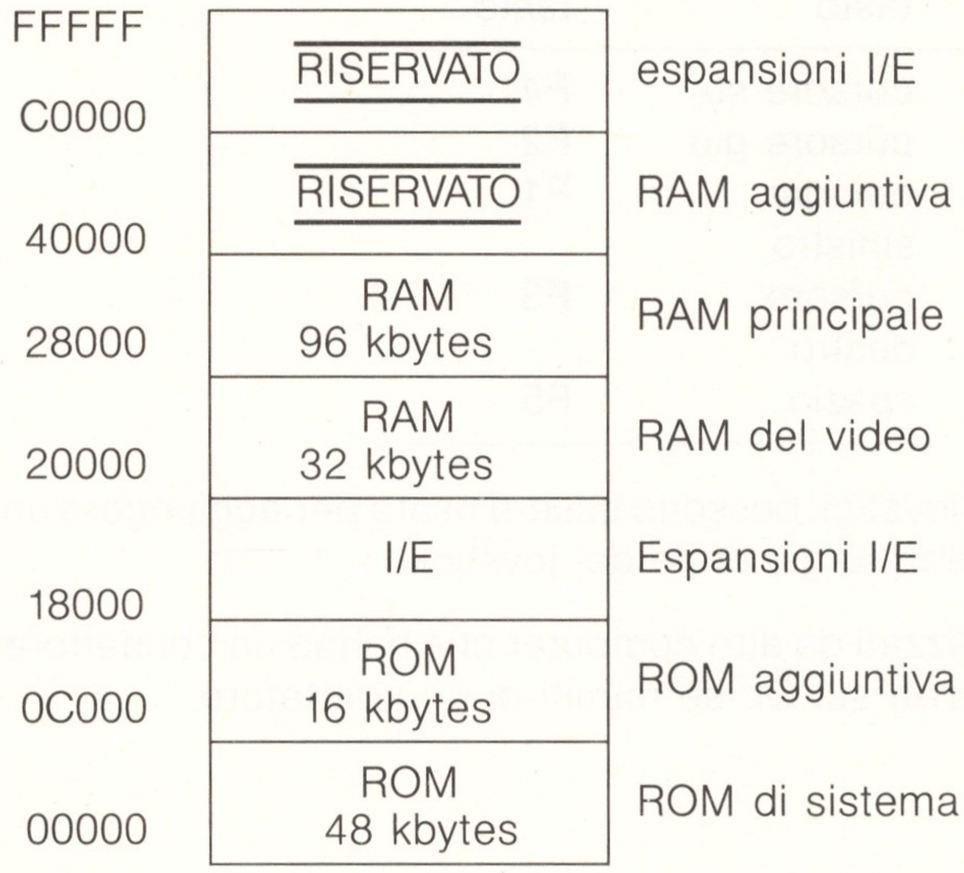
commento

Le porte dei joystick possono essere usate per aggiungere unità periferiche diverse e di natura più generale dei joystick.

I joystick utilizzati da altri computer che hanno un connettore a 9-poli 'D' possono essere utilizzati sul QL se muniti di un adattatore.

mappa di memoria

Il QL usa un processore Motorola 68008, che può indirizzare fino a un megabyte di memoria (da 00000 a FFFFF in esadecimale). Gli indirizzi all'interno di questi valori sono definiti nella figura seguente.



Mappa di memoria

La RAM del video è organizzata in una serie di parole di 16 bit, cominciando dall'indirizzo esadecimale 20000. I bit all'interno di ogni parola sono organizzati in modo che i pixel più significativi siano da sinistra verso destra (allo stesso modo di un numero binario).

Tuttavia l'organizzazione delle informazioni relative ai colori è diversa nelle due modalità di schermo:

byte alti A0=0	byte bassi A0=1	modo
VVVVVVVV	RRRRRRRR	512 (alta risoluz.)
VLVLVLVL	RBRBRBRB	256 (bassa risoluz.)

V-verde B-blue R-rosso L-lampeggio

Mappa dei colori

Nel modo ad alta risoluzione i colori Rosso e Verde usati assieme sono interpretati dal QL come bianchi.

attenzione Si raccomanda di non scrivere dati nelle aree di memoria riservate. Scrivere dati nei 'buffer' dei Microdrive può danneggiare i dati e causare una perdita di informazioni.

Microdrive

Le cartucce che si inseriscono nel Microdrive sono la memoria di massa del QL. Ogni cartuccia ha una capacità di 100 kbyte. Quando esiste spazio in memoria, il Qdos assegna questo spazio ad un buffer di Microdrive per migliorare le prestazioni.

Ogni cartuccia deve essere formattata prima di essere usata e può contenere fino a 255 settori di 512 bytes ciascuno.

Il Qdos mantiene un elenco (directory) dei file esistenti su cartuccia.

Una cartuccia può essere protetta dalla scrittura togliendo la piccola protezione posta sul lato destro.

È consigliabile formattare ripetutamente una cartuccia prima di usarla.

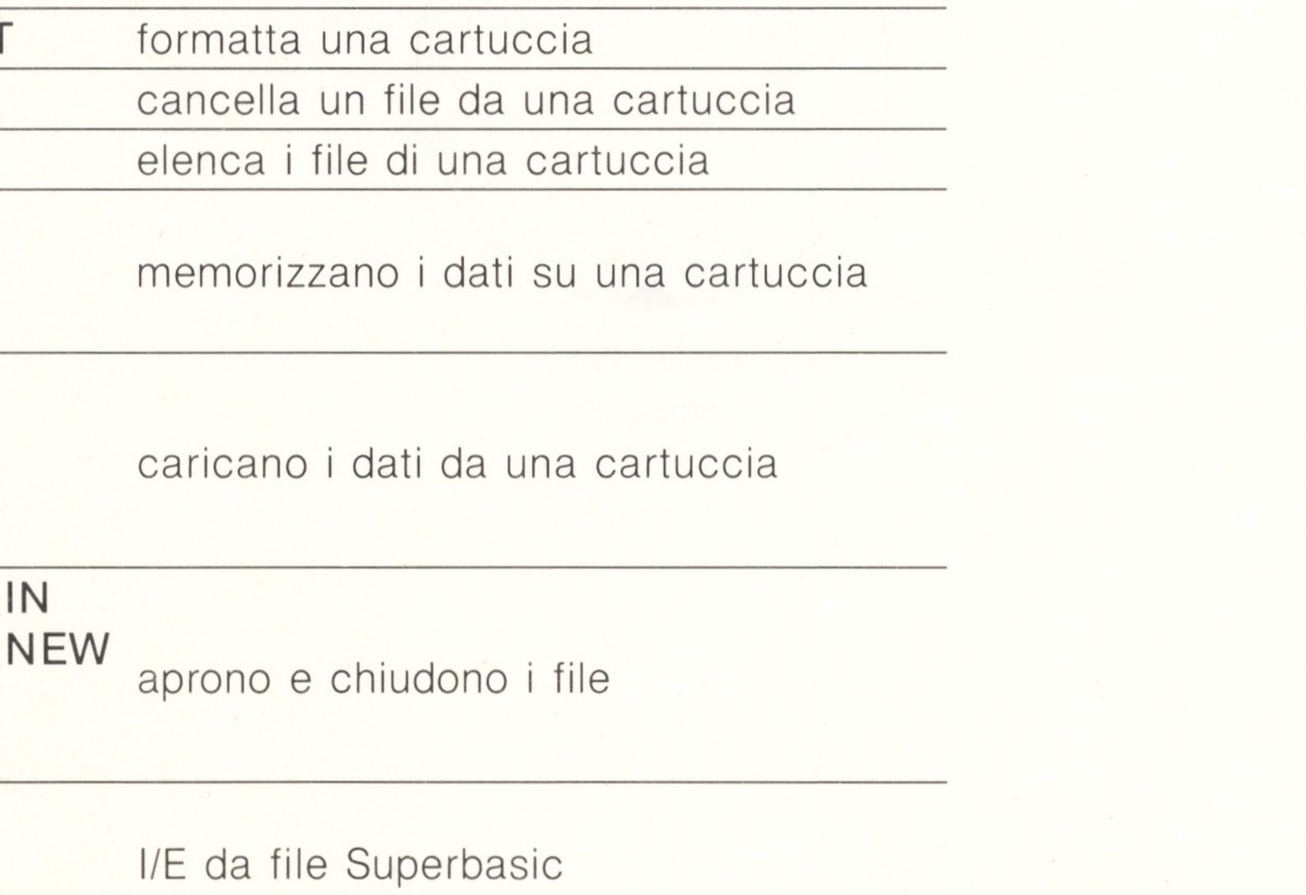
attenzioni generali

Fisicamente una cartuccia contiene circa 500 centimetri di nastro ad alta qualità che si muove alla velocità di 70 centimetri al secondo. Il nastro completa un riavvolgimento in circa 7 secondi e mezzo.

1. Non toccare il nastro con le dita.
2. Non accendere il computer con una cartuccia inserita nel Microdrive.
3. Inserire e levare le cartucce con attenzione.
4. Non spostare il QL (anche se spento) con cartucce inserite nel Microdrive.
5. Non toccare le cartucce quando il Microdrive è attivo.
6. Riporre le cartucce nelle proprie custodie quando non sono usate.
7. Non inserire le cartucce quando il Microdrive è in funzione.

nastro

Se il nastro dovesse uscire dalla cartuccia, riavvolgerlo con attenzione all'interno usando una penna a sfera. In ogni caso non toccare il nastro con le mani.



cartuccia di microdrive

Comando	Funzione
FORMAT	formatta una cartuccia
DELETE	cancella un file da una cartuccia
DIR	elenca i file di una cartuccia
SAVE SBYTES SEXEC	memorizzano i dati su una cartuccia
LOAD LBYTES EXEC MERGE	caricano i dati da una cartuccia
OPEN_IN OPEN_NEW OPEN CLOSE	aprono e chiudono i file
PRINT INPUT INKEY$	I/E da file Superbasic

attenzione

Se si tenta di scrivere su una cartuccia che è stata protetta dalla scrittura il QL segnala l'errore: "PROTETTO DA SCRITTURA".

monitor

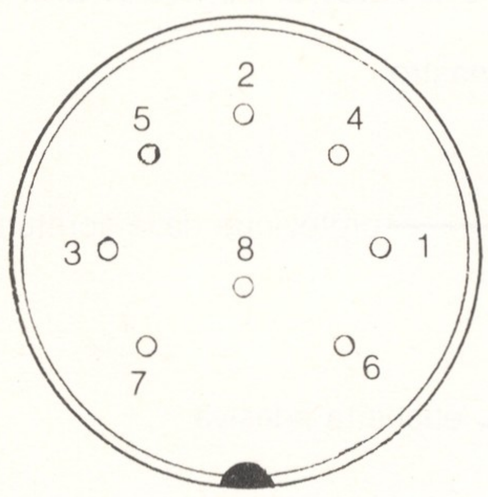
Un monitor a colori può essere connesso al QL attraverso l'uscita **RGB** posta sul retro. Per un monitor a colori il cavo è un DIN a otto poli, mentre per un monitor in bianco e nero è un DIN a tre poli. I segnali che escono dai poli sono indicati in tabella, dove sono anche riportati i colori dei cavi.

polo	funzione	segnale	colore cavo
1	PAL	PAL composito	arancio
2	GND	Terra	verde
3	VIDEO	Video composito monocromo	marrone
4	CSYNC	Sincronismo composito	giallo
5	VSYNC	Sincornismo verticale	blue
6	VERDE	Verde	rosso
7	ROSSO	Rosso	bianco
8	BLUE	Blue	viola

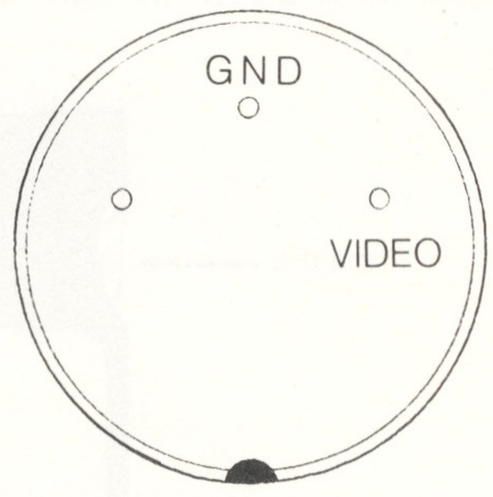
Al QL può essere collegato un monitor in bianco e nero usando un cavo a tre o a otto poli. In questo caso solo i poli 2 (terra) e 3 (video composito) devono essere attivi.

Al QL può essere collegato un monitor a colori (RGB) con un cavo a otto poli. La disposizione dei poli sul monitor varia a seconda del tipo di monitor usato.

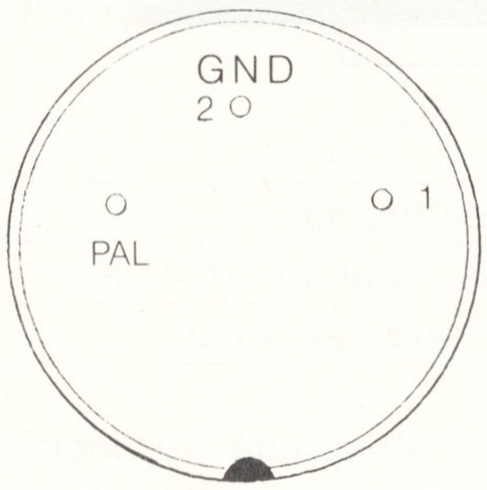
È possibile anche utilizzare un monitor composito PAL. In questo caso solo i segnali 2 (terra) e 1 (PAL composito) devono essere attivi.



monitor
a colori



monitor
monocromatico



monitor PAL
composito

nomi di variabile

Il nome di una variabile in Superbasic è costituito da una sequenza di lettere o numeri e sottolineature.

definizione: lettera= da A a Z
da a a z

numero= | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 |

nome variabile=lettera *[lletteralnumerol__l]*

Un nome di variabile deve avere come primo carattere una lettera seguita da una serie di lettere o numeri o sottolineature e può essere lungo fino a 255 caratteri. I caratteri minuscoli e maiuscoli sono equivalenti.

I nomi di variabile sono utilizzati in Superbasic per identificare variabili, procedure, funzioni, ecc.

operatori

Operatore	Tipo	Funzioni
=	numerico stringa	confronto
= =	numerico stringa	quasi uguale **'
+	numerico	addizone
—	numerico	sottrazione
/	numerico	divisione
*	numerico	moltiplicazione
<	numerico stringa	minore
>	numerico stringa	maggiore
< =	numerico stringa	minore uguale
> =	numerico stringa	maggiore uguale
< >	stringa	diverso
&	stringa	concatenazione
&&	bit	AND
| |	bit	OR
^^	bit	XOR
~ ~	bit	NOT
OR	logico	OR
AND	logico	AND
XOR	logico	XOR
NOT	logico	NOT
MOD	intero	modulo
DIV	intero	divisione
INSTR	stringa	confronto
^	numerico	potenza
—	numerico	numero negativo
+	numerico	numero positivo

** quasi uguale = uguale con approssimazione di 10^{-7}.

Quando un'espressione logica assume valore di verità, è ritornato un valore numerico diverso da zero.

Se l'espressione è falsa, il valore numerico ritornato è zero.

precedenze

Le precedenze degli operatori sono definite nella seguente tabella. Se l'ordine di esecuzione di una espressione non può essere dedotto dalla tabella, le operazioni avvengono da sinistra verso destra.

precedenza maggiore
segni negativo o positivo
concatenazione fra stringhe
instr
potenze
moltiplicazione, divisione, modulo
addizione, sottrazione
not (bit e logico)
and (bit e logico)
precedenza inferiore or o xor (bit e logico)

Le precedenze possono essere mutate a piacere con l'uso di parentesi.

operazioni fra stringhe

In un vettore è sempre possibile far riferimento ad un suo sottoinsieme chiamato sottovettore. Ogni sottovettore può essere definito come una serie di elementi appartenenti al vettore originale. In questo contesto, con il termine vettore si fa riferimento ad un vettore numerico, alfabetico o semplicemente ad una stringa di caratteri.

sintassi

sintassi: indice=espressione__numerica
espressione__numerica **TO** espressione__numerica
espressione__numerica **TO**
TO espressione__numerica

rif__vettore=variabile
variabile (variabile [indice*[,indice]*])

esempi

esempi: i. **PRINT lettera$(12 TO 15)**
ii. **PRINT vettore(3)(2 TO 4)**

L'operazione di estrazione di sottostringhe da stringhe alfabetiche è uguale a quella per i vettori numerici.

a$(n)	seleziona il carattere n-simo
a$(n TO m)	seleziona tutti i caratteri dalla posizione m alla n inclusa
a$(n TO)	seleziona tutti i caratteri dalla posizione n inclusa alla fine
a$(1 TO m)	seleziona tutti i caratteri dall'inizio alla posizione n inclusa
a$	seleziona l'intero contenuto di a$

Alcune versioni di Basic utilizzano delle funzioni chiamate **LEFT$, MID$, RIGHT$**, che non sono necessarie nel Superbasic. La tabella di seguito riporta l'equivalenza delle istruzioni.

Superbasic	Altri Basic
a$(n)	MID$(a$,n,1)
a$(n TO m)	MID$(a$,n,m+1-n)
a$(1 TO n)	LEFT$(a$,n)
a$(n TO)	RIGHT$(a$,LEN(a$)+1-n)

orologio

Il QL è dotato di un orologio, in funzione mentre il computer è acceso.

Il formato utilizzato per indicare la data segue lo stanard ISO.

1985 apr 15 12:09:10

È possibile ottenere i valori singoli dell'anno, mese, giorno, ora, estraendoli dalla stringa ritornata dall'istruzione **DATE**

Comandi	Funzione
SDATE	fissa l'orologio
ADATE	regola l'orologio
DATE	ritorna la data sotto forma di numero
DATE$	ritorna la data sotto forma di stringa
DAY$	ritorna il giorno della settimana

parole chiave

In Superbasic esistono una serie di parole chiave che possono essere scritte sia in lettere maiuscole che minuscole. Ognuna di esse esegue una funzione ben precisa. L'insieme delle parole chiave può essere esteso aggiungendo delle procedure, create dall'utente. Le parole chiave non possono essere usate come nomi di variabili o procedure.

La parte di parola scritta in maiuscolo indica il numero minimo di caratteri che deve essere scritto in un programma Superbasic, perché il QL riconosca il nome come parola chiave.

ELENCO DELLE PAROLE CHIAVE

ABS	DEF FN,	LIST	PRINT
ADATE	END DEF	LOAD	RAD
ARC	DEF PROC,	LOC	RANDOMISE
ARC_R	END DEF	LN	RND
AT	DELETE	LOG10	RECOL
ATAN,ACOS	DIM	LRUN	REM
ACOT,ASIN	DIMN	MRUN	RENUM
AUTO	DIV	MERGE	REP, END REP
BAUD	DIR	MOD	RESPR
BEEP	DLINE	MODE	RET
BEEPING	EDIT	MOVE	RETRY
BLOCK	ELLIPSE	NET	RUN
BORDER	EOF	NEW	SAVE
CALL	EXEC	NEXT	SBYTES
CHR$	EXEC_W	ON...GOTO	SDATE
CIRCLE	EXIT	ON...GOSUB	SDATE$
CIRCLE_R	EXP	OPEN	SIN
CLEAR	FILL	OPEN_IN	SCALE
CLOSE	FILL$	OPEN_NEW	SCROLL
CLS	FLASH	OVER	SEL, END SEL
CODE	FOR, END FOR	PAN	SEXEC
CONTINUE	FORMAT	PAPER	SIN
COPY	GOSUB	PAUSE	SQRT
COPY_N	GOTO	PEEK	STOP
COS	IF, THEN, ELSE	PEEK_W	STRIP
COT	END IF	PEEK_L	TAN
CSIZE	INK	PENUP	TO
CURSOR	INKEY$	PENDOWN	TURN
DATA,READ,	INPUT	PI	TURN TO
RESTORE	INSTR	POINT	UNDER
DATE$	INT	POINT_R	VER
DATE	LET	POKE	WIDTH
DAY$	LINE	POKE_W	WINDOW
DEG	LINE_R	POKE_L	

Per maggiori informazioni consultare il capitolo "Parole chiave" dello stesso manuale.

programmi

Un programma Superbasic è formato da una sequenza di istruzioni, ognuna delle quali è preceduta da un numero di riga. È possibile scegliere i numeri di riga da 1 a 32767.

sintassi: numero__riga= *[numero]* {da 1 a 32767}

*[numero__riga istruzione *[:istruzione]*]*

Comando	Funzione
RUN	esegue un programma in memoria
LRUN	carica un programma da una unità e lo esegue
CTRL SPAZIO	interrompe l'esecuzione di un programma

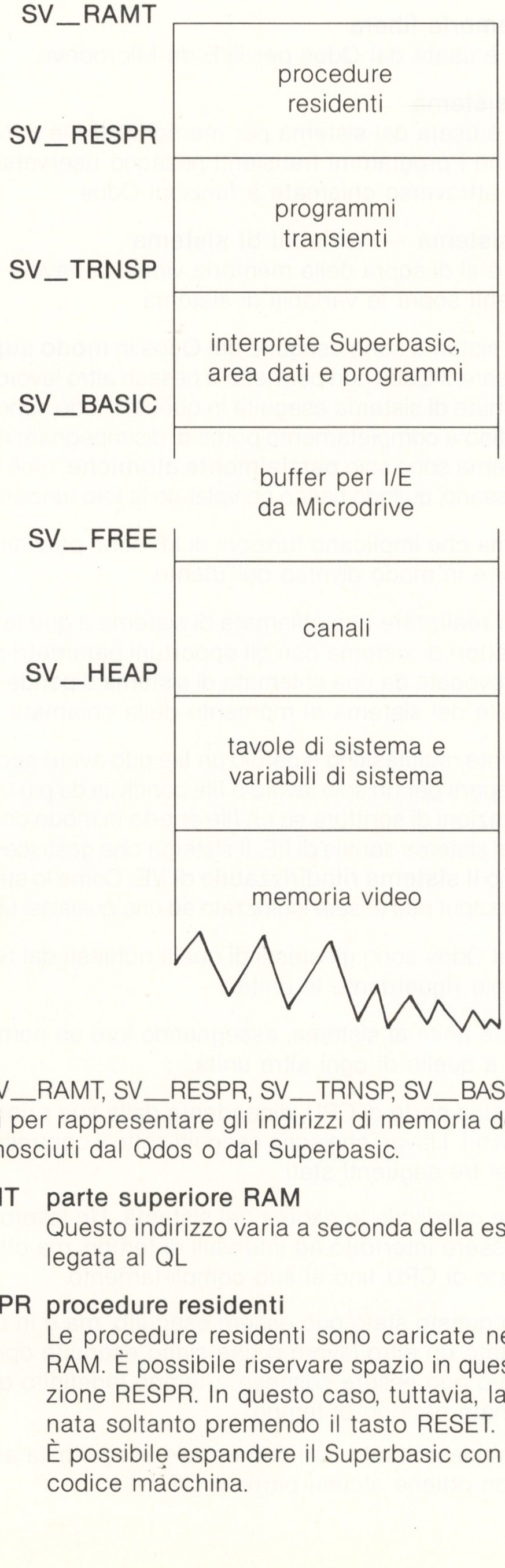
qdos

Il Qdos è il sistema operativo del QL e controlla:

— l'ordine di esecuzione dei lavori
— l'allocazione delle risorse del computer
— il video
— l'I/E da Microdrive
— le porte seriali e le uscite in rete locale
— la tastiera
— la gestione della memoria

Una completa descrizione del Qdos è al di fuori degli scopi di questo manuale, tuttavia nel seguito è riportata una mappa schematica della RAM e del sistema operativo.

Mappa della memoria



Mappa memoria Qdos

I termini SV_RAMT, SV_RESPR, SV_TRNSP, SV_BASIC, SV_FREE, SV_HEAP sono usati per rappresentare gli indirizzi di memoria del QL. Questi termini non sono riconosciuti dal Qdos o dal Superbasic.

SV_RAMT parte superiore RAM
Questo indirizzo varia a seconda della espansione di memoria collegata al QL

SV_RESPR procedure residenti
Le procedure residenti sono caricate nella parte superiore della RAM. È possibile riservare spazio in quest'area di RAM con la funzione RESPR. In questo caso, tuttavia, la RAM può essere ripristinata soltanto premendo il tasto RESET.
È possibile espandere il Superbasic con altre procedure scritte in codice macchina.

SV_TRNSP **programmi transienti**
I programmi transienti sono caricati nella parte di RAM immediatamente inferiore a quella usata dalle procedure residenti. Ogni programma deve contenere spazio per i suoi dati e per il suo puntatore. Un programma transiente è eseguito dal Superbasic con il comando **EXEC** o da Qdos come lavoro indipendente. Questa area può anche essere usata per memorizzare dati, ma anche in questo caso il Qdos tratta i dati come lavoro attivo.

SV_BASIC **area Superbasic.**
Quest'area contiene tutti i programmi Superbasic e/o dati relativi. Quest'area si espande e si contrae usando la RAM disponibile a seconda dell'impegno richiesto da ogni programma.

SV_FREE **area di memoria libera**
Quest'area è usata dal Qdos per l'I/E da Microdrive.

SV_HEAP **spazio di sistema**
Quest'area è usata dal sistema per memorizzare le definizioni dei canali. Anche i programmi transienti possono riservarsi spazio in quest'area, attraverso chiamate a funzioni Qdos

Tavole di sistema — Variabili di sistema
Quest'area è al di sopra della memoria video. Le tavole di sistema sono residenti sopra le variabili di sistema.

chiamate di sistema

Le chiamate a funzione di sistema sono eseguite dal Qdos in **modo supervisore**. Quando è in modo supervisore, il Qdos non permette a nessun altro lavoro di controllare il processore. Le chiamate di sistema eseguite in questo modo sono chiamate **atomiche**. Sono eseguite fino a completamento prima di disimpegnare il processore. Alcune chiamate di sistema sono solo **parzialmente atomiche**, cioè lasciano libero il processore, se necessario, quando hanno completato la loro funzione primaria.

Tutte le chiamate di sistema che implicano funzioni di I/E sono parzialmente atomiche, se non sono definite in modo diverso dall'utente.

La procedura standard per realizzare una chiamata di sistema è quella di inserire un indicatore in uno dei vettori di sistema con gli opportuni parametri nei registri del processore. L'azione provocata da una chiamata di sistema dipende dalla chiamata e dallo stato generale del sistema al momento della chiamata.

input/output (I/E)

Il Qdos permette un ambiente multitasking e perciò un file può avere accessi multipli. Il Qdos può gestire file aperti per un solo lavoro o file condivisi da più lavori. Tuttavia non sono possibili operazioni di scrittura su un file aperto in modo condiviso. Le unità del Ql sono gestite dal sistema seriale di I/E. Il sistema che gestisce i file e il sistema seriale di I/E formano **il sistema riindirizzabile di I/E**. Come lo stesso nome suggerisce, qualsiasi tipo di output può essere indirizzato ad una qualsiasi unità del QL.

I nomi di unità richiesti dal Qdos sono gli stessi di quelli richiesti dal Superbasic e sono discussi nel capitolo riguardante le unità.

È possibile aggiungere altre unità al sistema, assegnando loro un nome e realizzando un accesso simile a quello di ogni altra unità.

multitasking

Ad ogni lavoro è riservata una parte di CPU, dipendente dalla sua priorità e dalla sua interazione con altri lavori. I lavori che sono eseguiti sotto il controllo del Qdos possono essere in uno dei tre seguenti stati:

attivo è eseguito e condivide le risorse del sistema. Un lavoro in questo stato può essere interrotto ad intervalli di tempo, ma ottiene sempre una parte di CPU fino al suo completamento.

sospeso un lavoro in questo stato può essere eseguito, ma è in attesa che sia completato un altro lavoro o che siano eseguite operazioni di I/E. Un lavoro può essere sospeso a tempo indefinito oppure per un determinato periodo di tempo.

inattivo un lavoro in questo stato non può essere eseguito, la sua priorità è zero, e non ottiene alcuna parte di CPU.

esempio

Questo programma provoca l'emissione sul video dell'orologio di sistema ed è eseguito come un lavoro indipendente.

Memorizzare il programma su una cartuccia inserita nel Microdrive 2 con il nome di orologio. Prima di eseguire il programma regolare l'orologio con il comando **SDATE**. Eseguire il programma con l'istruzione:

```
EXEC mdv2_orologio
```

```
100  c=RESPR(100)
110  FOR i=0 TO 68 STEP 2
120     READ x : POKE_W i+c,x
130  END FOR i
140  SEXEC mdv2_orologio,c,100,256
1000 DATA 29439, 29697, 28683, 20033, 17402
1010 DATA 48,13944,200,20115,12040
1020 DATA 28691,20033,17402,74;27698
1030 DATA 13944,236,20115,8279;11314
1040 DATA 13944,208,20115,16961,16962
1050 DATA 30463,28688,20035,24794
1060 DATA 0,7,240,10,272,200
```

La linea 1060 controlla la posizione ed il colore della finestra in cui appare l'orologio.

I dati contenuti nelle istruzioni DATA rappresentano nell'ordine:

— colore e dimensione bordo
— colore fondo e caratteri
— larghezza finestra
— origine x
— origine y.

I dati sono coppie di bytes, introdotti come parole con l'istruzione **POKE_W**.

L'origine dell'asse x e y (gli ultimi valori delle istruzioni DATA) devono essere 272 e 202 se si utilizza un monitor, mentre 240 e 216 se si usa un apparecchio televisivo.

I bytes relativi al colore di fondo dei caratteri sono generati, ad esempio, come 256 * PAPER + INK. Così uno sfondo bianco con caratteri rossi è:

```
256*7+2=1794
```

rete locale

Il QL può essere collegato in rete locale con altri QL o Spectrum, fino ad un massimo di 64. Se più di due computer sono collegati fra loro, ad ognuno deve essere assegnato un numero che identifichi univocamente la stazione. L'operazione è realizzata sul QL con il comando **NET**.

Le informazioni sono trasmesse lungo la rete a blocchi. Per una corretta comunicazione tra due stazioni, il computer ricevente deve riconoscere il blocco trasmessogli. Se un blocco di informazioni è danneggiato, la stazione ricevente chiede che sia trasmesso nuovamente.

Usare un numero di stazione 0 ha un significato speciale. Infatti mandare un messaggio alla stazione **NET__0**, significa che il messaggio è letto da tutte le unità della rete (Broadcasting).

Comando	Funzione
NET	assegna un numero di stazione
OPEN	apre un canale della rete
CLOSE	chiude un canale della rete
PRINT **INPUT** **INKEY$**	I/E di rete
LOAD **SAVE** **LBYTES** **SBYTES** **EXEC** **SEXEC** **LRUN** **MRUN** **MERGE**	caricano e salvano file sulla rete

commento

Quando si collegano parecchi QL in rete locale è consigliabile utilizzare un doppino telefonico.

Sebbene il software possa controllare fino a 64 stazioni, l'hardware può al massimo controllare un cavo lungo 100 metri.

ROM

Lo slot di espansione ROM consente di usare il software residente su cartuccia ROM. Una cartuccia ROM può anche contenere del software per espandere il Superbasic. Le cartucce ROM dello Spectrum non sono utilizzabili sul QL.

Segnale	Funzione
AO....A15	linee indirizzi
DO....D8	linee dati
ROMOEH	segnale per l'output della ROM
VDD	5 V
GND	terra

attenzione

Attenzione a non inserire o sconnettere una cartuccia ROM quando il QL è acceso.

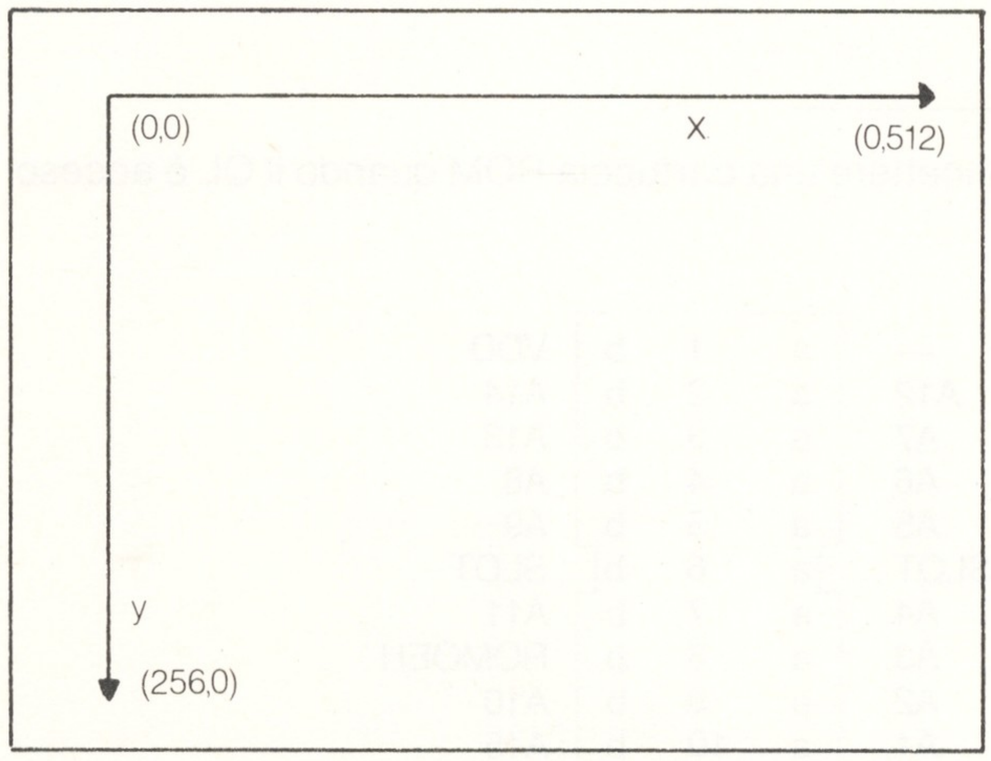
—	a	1	b	VDD
A12	a	2	b	A14
A7	a	3	b	A13
A6	a	4	b	A8
A5	a	5	b	A9
SLOT	a	6	b	SLOT
A4	a	7	b	A11
A3	a	8	b	ROMOEH
A2	a	9	b	A10
A1	a	10	b	A15
A0	a	11	b	D7
D0	a	12	b	D6
D1	a	13	b	D5
D2	a	14	b	D4
GND	a	15	b	D3

sistema di coordinate pixel

Il sistema di coordinate a pixel è usato per definire la posizione e la dimensione di finestre, blocchi, e del cursore. Il sistema di coordinate ha la sua origine nell'angolo superiore sinistro del video e usa i pixel come se lo schermo fosse sempre in alta risoluzione (512).

Il sistema usa il pixel disponibile più vicino per rendere le coordinate indipendenti dalla modalità di schermo in uso.

Alcuni comandi si riferiscono sempre all'origine della finestra di default (**WINDOW**), mentre altri si riferiscono all'origine della finestra corrente (**BLOCK**).



Il sistema di coordinate pixel

suono

Il suono sul QL è generato da un secondo processore (8049) ed è controllato specificando:

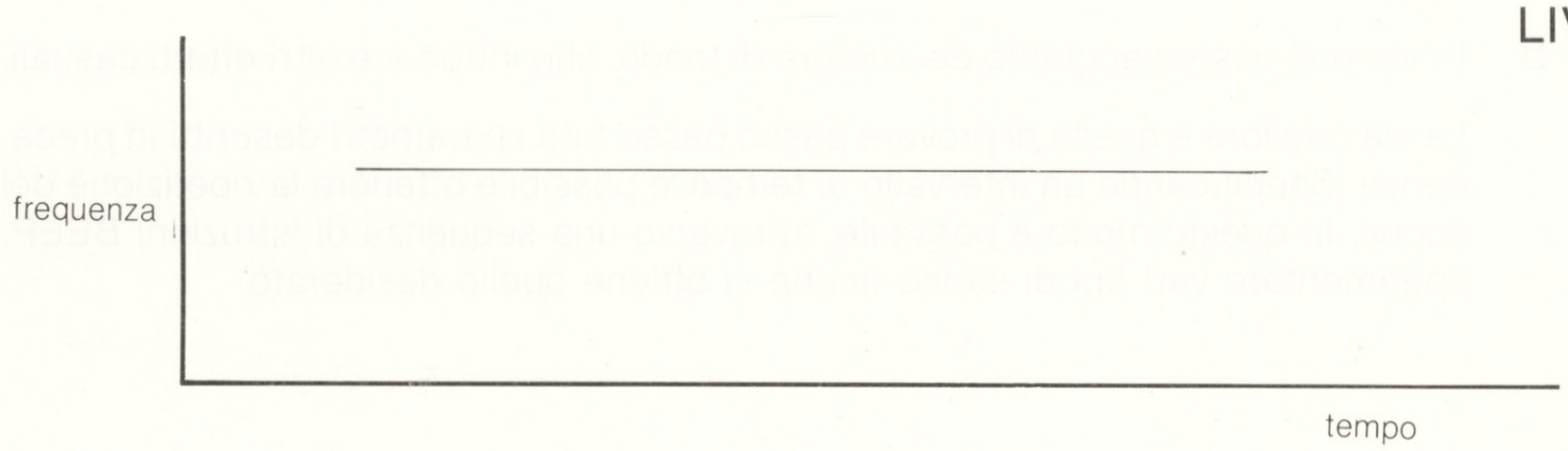
— due frequenze
— La velocità con la quale il suono si deve muovere fra le due frequenze
— comportamento del suono quando ha raggiunto una delle due frequenze
— la casualità con cui il suono può muoversi attorno alle due frequenze
— il rumore che può essere costruito attorno alle due frequenze prinipali.

Il rumore dà come risultato un ronzio attorno al suono principale, mentre la casualità determina un rumore melodico di fondo.

La complessità e l'articolazione del suono può essere costruita poco per volta. È questo il modo migliore per capire come si comporta il suono sul QL.

Specificando la durata ed una singola frequenza, l'onda sonora può essere così rappresentata:

LIVELLO 1

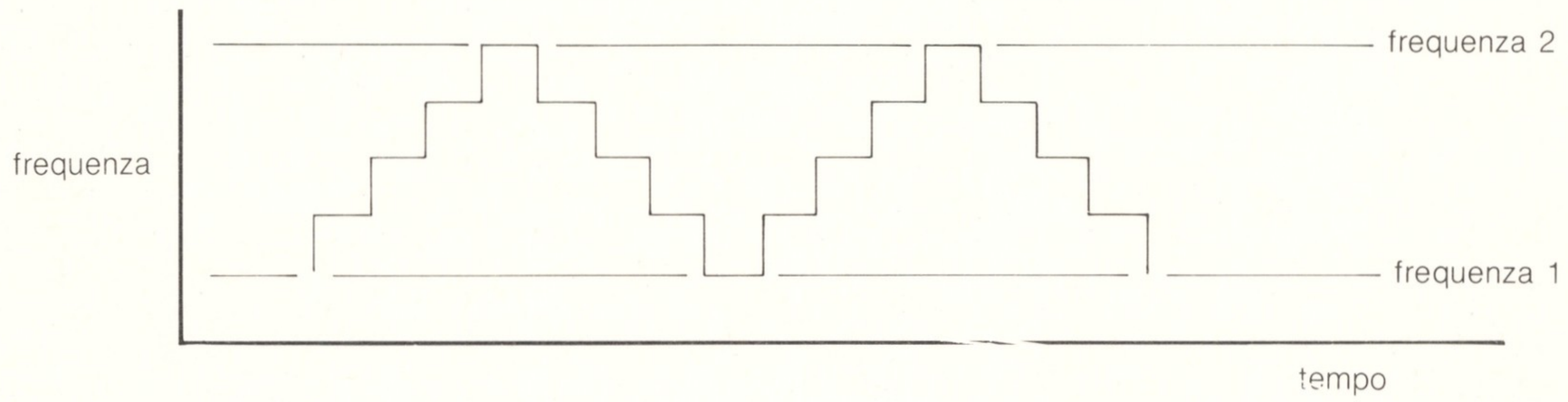


Questa è la forma più semplice del comando **BEEP**.

LIVELLO 2

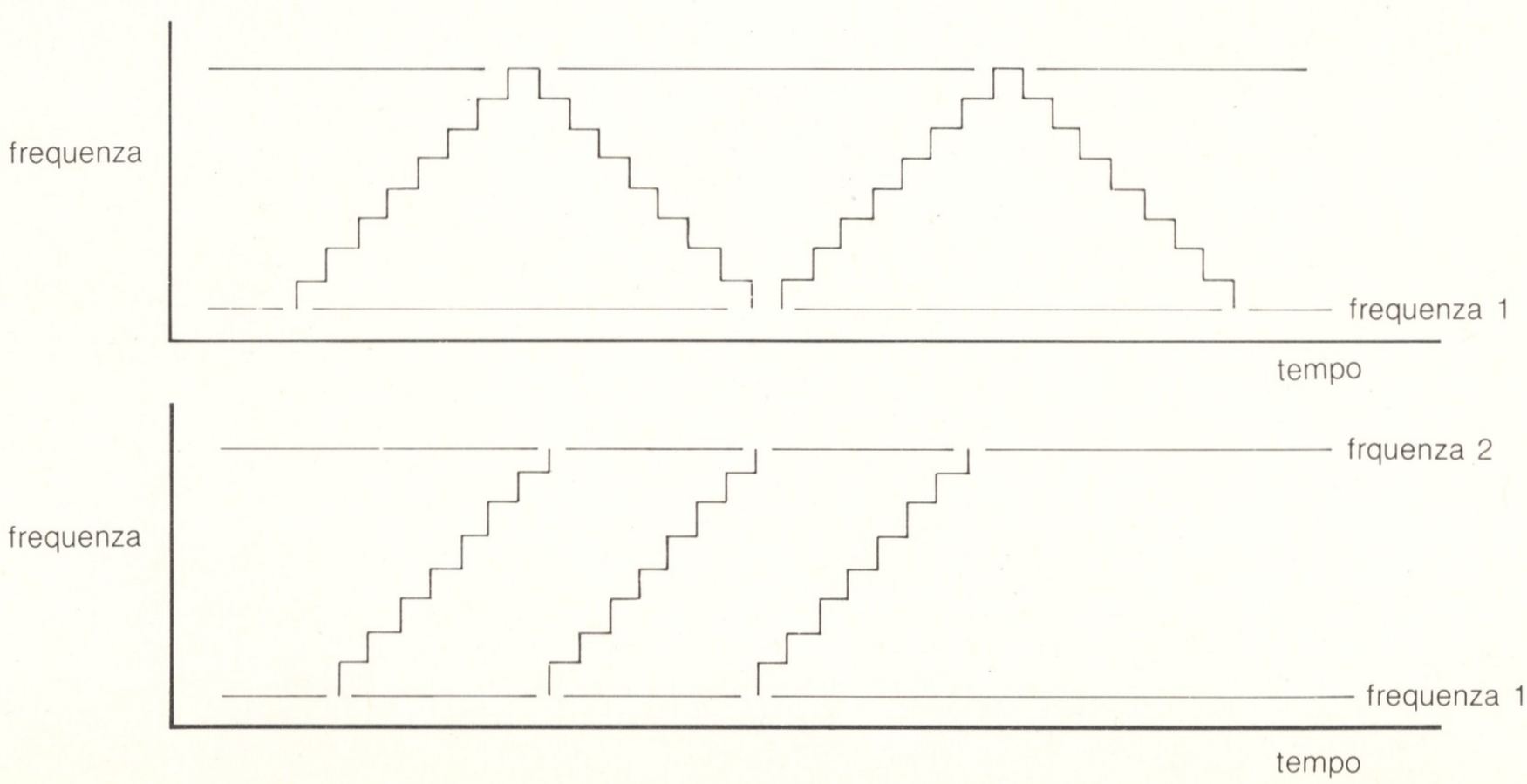
Come secondo gradino, può essere poi aggiunta un'ulteriore frequenza e un gradiente. Il suono "rimbalza" così fra le due frequenze alla velocità determinata dal gradiente.

I suoni prodotti a questo livello possono variare da: tonalità semi-musicali, brontolii e gemiti.



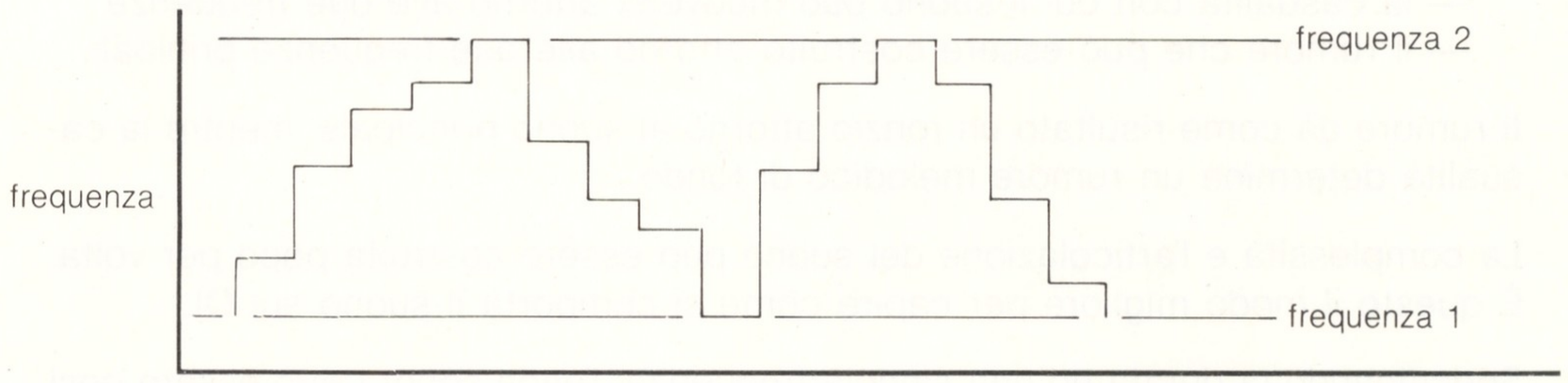
LIVELLO 3

Può essere poi aggiunto un altro parametro che controlla il comportamento del suono quando questo diventa uguale ad una delle 2 frequenze specificate. Il suono può essere fatto rimbalzare fra i due toni oppure è possibile ottenere delle repliche dei toni (anche infinite).



LIVELLO 4 Inoltre può essere aggiunta una determinata dose di casualità. La casualità è una deviazione delle frequenze dai gradienti specificati.

Mescolando i valori dei diversi parametri è possibile ottenere degli effetti sonori sorprendenti.



LIVELLO 5 Infine può essere aggiunto del rumore di fondo, che introduce altri effetti casuali.

La via migliore è quella di provare passo passo tutti i parametri descritti in precedenza. Specificando un intervallo di tempo, è possibile ottenere la ripetizione del suono. In questo modo è possibile, attraverso una sequenza di istruzioni **BEEP**, sperimentare vari tipi di suono finché si ottiene quello desiderato.

tartaruga grafica

Il Superbasic ha un insieme di comandi simili a quelli esistenti in altri linguaggi, tipo LOGO, che consentono di guidare la **tartaruga grafica**.

Comando	Funzione
PENUP	ferma il disegno
PENDOWN	inizia il disegno
MOVE	muove la tartaruga
TURN	gira la tartaruga
TURNTO	gira la tartaruga verso una direzione specificata

Questo è un insieme di comandi minimo. Tuttavia è possibile creare delle piccole procedure per integrare il movimento della tartaruga grafica. Ad esempio:

```
100 DEFine PROCedure avanti (distanza)
110    MOVE distanza
120 END DEFine
130 DEFine PROCedure dietro (distanza)
140    MOVE -distanza
150 END DEFine
160 DEFine PROCedure sinistra (angolo)
170    TURN angolo
180 END DEFine
190 DEFine PROCedure destra (angolo)
200    TURN -angolo
210 END DEFine
```

Questo insieme di procedure definisce i più noti comandi della tartaruga, esistenti anche in altri linguaggi.

Inizialmente la penna che disegna la tartaruga non scrive, e la tartaruga è sulla parte destra dello schermo rivolta nella stessa direzione dell'asse verticale dello schermo.

Anche le figure tracciate con la tartaruga grafica possono essere riempite usando l'istruzione **FILL**. Inoltre tutte le istruzioni grafiche possono essere ugualmente utilizzate quando la tartaruga grafica è attiva. La direzione della tartaruga non può essere modificata dalle istruzioni grafiche del Superbasic.

tipo di variabili

interi

I numeri interi possono essere compresi nell'intervallo -32768 e 32767. Una variabile è definita intera quando il suo nome termina con il simbolo di percentuale (%). Le costanti in Superbasic sono trattate come numeri in virgola mobile.

sintassi: **nome_variabile%**

esempi:
i. **contatore%**
ii. **limite%**
iii. **questa_è_una_variabile_intera%**

virgola mobile

I numeri in virgola mobile sono compresi nell'intervallo -10^{615} e 10^{615}, con otto cifre decimali significative. In Superbasic, per default, una variabile numerica è in virgola mobile.

sintassi: **nome_variabile|costante**

esempi:
i. **valore**
ii. **76.2536**
iii. **354E25**

stringa

Una stringa è una sequenza di 32766 caratteri al massimo. Il nome di una variabile stringa o alfabetica deve terminare con il simbolo del dollaro ($). I caratteri assegnati ad una stringa sono racchiusi fra virgolette o apostrofo.

sintassi: **nome_variabile $**

esempi:
i. **stringa$**
ii. **"questa_è_una_variabile_alfabetica."**

unità

Una **unità** è un dispositivo dal quale possono essere letti e scritti dei dati.

Poiché il sistema non richiede di specificare il tipo di unità usato, è molto facile variare di volta in volta il tipo di dispositivo utilizzato. Ad esempio un programma che indirizzi l'output su stampante, può, se la stampante per qualche ragione non fosse disponibile, indirizzare lo stesso output su Microdrive. Il file può essere stampato quando la stampante torna ad essere disponibile. È possibile pensare di trattare le unità del QL come file logici. Questo significa che il sistema operativo Qdos è indipendente dal tipo di unità utilizzata.

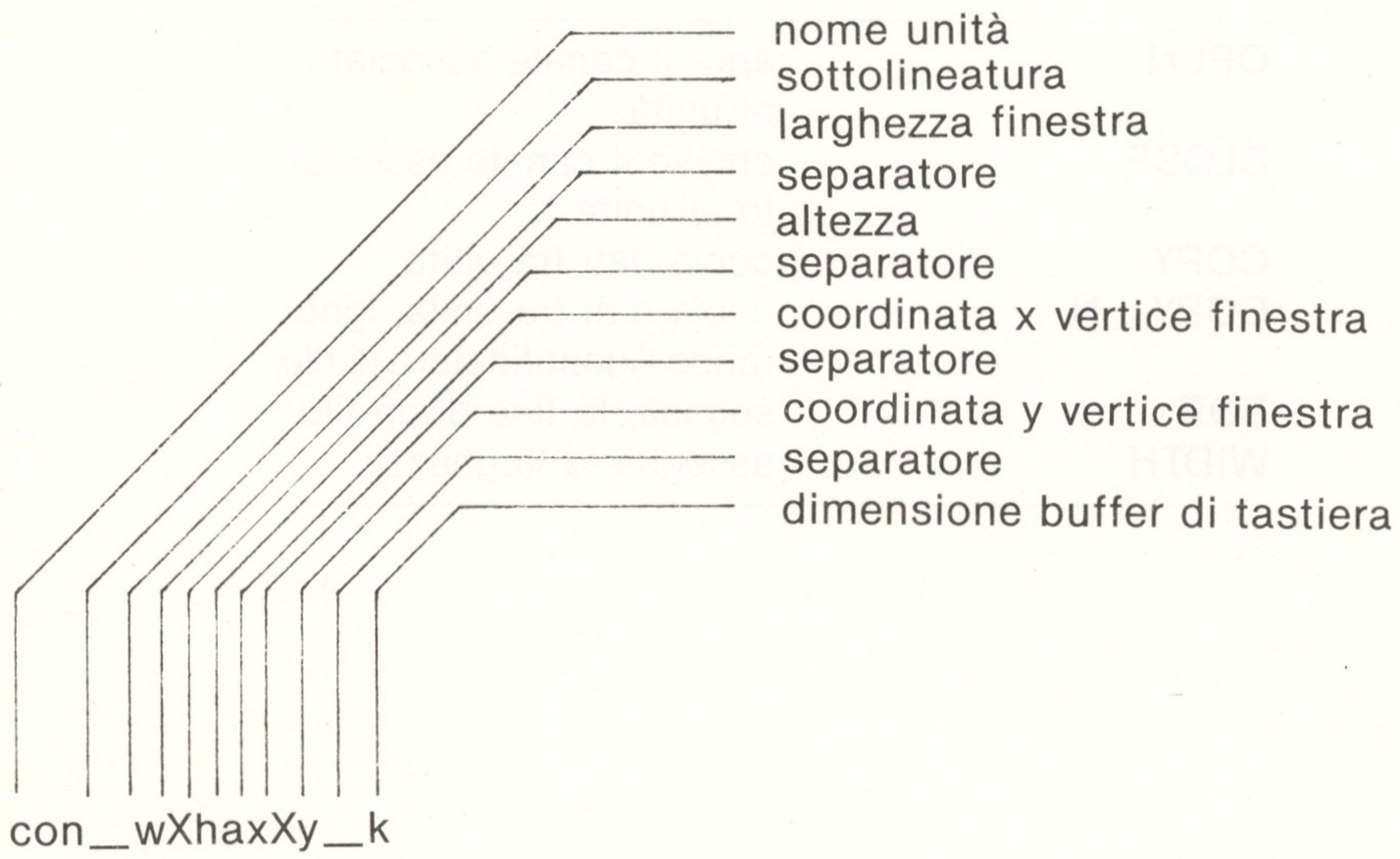
Tutte le operazioni relative ad unità, sono effettuate da particolari file di comandi, chiamati **device drivers**. Il Qdos ricerca automaticamente il file che contiene i comandi dell'unità interessata ad operazioni di I/E.

Quando è attivata una unità, è aperto un canale, che è assegnato a quella unità. A volte, per aprire un canale in modo corretto, è necessario specificare ulteriori informazioni dopo il nome dell'unità.

Ogni unità logica sul QL richiede una serie di informazioni aggiuntive, sebbene, dove possibile, il sistema assegni di parametri default.

apertura di un canale di console

esempio



[vXh] larghezza e altezza finestra

con__wXhaxXy__k

[AxXy] coordinate del vertice in alto e sinistra della finestra

[k] lunghezza buffer di tastiera in bytes

default: con__448x180a32x16__128

canale video

scr__wXhaxXy

[vXh] larghezza e altezza finestra
[AxXy] coordinate del vertice in alto a sinistra della finestra

default: scr__448x180a32x16

SERnphz Porte seriali RS-232-C apertura di un canale seriale

[p] parità	[h] handshaking	[z] protocollo
e = even	**i** = no handshake	**r** = no EOF
o = odd	**h** = handshake	**z** = CTRL Z è EOF
m = mark		**c** = come z, ma converte ASCII 10 (il carattere Qdos di fine riga) in ASCII 13 (CR).
s = space		

default **ser1RH (8 bit, nessuna parità, handshake)**

NETd_s I/E da rete locale.

[d] indica la direzione	**s** = numero di stazione
i = input	**0** = per broadcast
o = output	

default **nessun default**

MDVn_nome Accesso ai file su Microdrive.

n = numero di Microdrive
nome = nome del file su Microdrive
default = nessun default

Parola chiave	Funzione
OPEN	apre il canale associato all'unità
CLOSE	chiuse il canale associato all'unità
COPY	copia dati fra unità
COPY_N	copia dati fra unità, ignorando l'identificatore di file
EOF	segnala la fine di un file
WIDTH	assegna la larghezza

vettori alfabetici

I vettori alfabetici e numerici hanno la stessa struttura ma sono trattati in modo leggermente diverso dal Superbasic. L'ultima dimensione di un vettore alfabetico definisce la lunghezza di ogni elemento del vettore. Le stringhe alfabetiche possono essere lunghe al massimo 32766 caratteri.

In ogni caso non occorre che tutti gli elementi di un vettore alfabetico abbiano la stessa lunghezza, ma nell'istruzione DIM è necessario dichiarare la lunghezza massima prevedibile, altrimenti i caratteri in più sono troncati.

commento

A differenza di molti altri Basic, il Superbasic non tratta i vettori alfabetici come stringhe di lunghezza fissa.

Se i dati memorizzati in un vettore alfabetico non occupano la dimensione massima del vettore, questa è ridotta.

attenzione

Assegnare dati ad una sottostringa o ad un sottovettore alfabetico può non produrre l'effetto desiderato. Infatti queste assegnazioni non aggiornano la lunghezza della stringa, che è variata solo quando l'assegnazione è fatta sull'intera stringa.

Comando	Funzione
FILL$	crea una stringa
LEN$	calcola la lunghezza di una stringa

vettori numerici

I vettori devono essere **dimensionati** con l'istruzione **DIM** prima di poter essere utilizzati. Quando un vettore è dimensionato, tutti i suoi elementi sono posti a zero.

La dimensione di un vettore può variare da 0 fino al valore specificato. Non c'è alcun limite, tranne la capacità di memoria al numero di dimensioni che possono essere assegnate ad un vettore. Riferirsi ad un elemento di un vettore A(a,b,c) è equivalente che riferirsi allo stesso elemento come A(a)(b)(c).

Comando	Funzione
DIM	dimensiona un vettore
DIMN	ritorna il valore della dimensione specificata.

video

mode 512

Le dimensioni del video in pixel sono 512 in orizzontale e 256 in verticale. I colori che possono essere visualizzati in questa modalità di schermo sono:

nero
rosso
verde
bianco

È possibile nella modalità a bassa risoluzione ottenere anche il lampeggiamento.

mode 256

Nella modalità di schermo che ha 256 pixel in orizzontale e 256 in verticale, possono essere visualizzati i seguenti colori:

blue
rosso
magenta
verde
ciano
giallo
bianco

attenzione

Un' apparecchio televisivo non è in grado di visualizzare l'intero schermo del QL. Le parti laterali e la parte superiore dello schermo non sono riprodotte sul video di una televisione. La dimensione iniziale di default delle finestre tengono conto di questo problema e riducono quindi la dimensione originale dell'immagine, che può essere ripristinata con l'istruzione **WINDOW**.

Comando	Funzione
MODE	seleziona la modalità di schermo

INDICE DEI CONCETTI

A

ACCENSIONE 1

B

BASIC 2
BAUD 12
BEEP 45
BOOTING 1
BREAK 3

C

CANALI 4
CARTUCCE
  microdrive 31
  rom 43
CARATTERI 5
CODICI
  carattere 5
  colore 9
  errori 16
  esadecimali 5
COERCIZIONE 14
COLORE 9
COMANDI
  diretti 10
  finestre 22
  parole chiave 37
  tartaruga grafica 47
  VIDEO 53
COMUNICAZIONI
  canali 4
  unità 49
  RETE LOCALE 42
CONFRONTO FRA STRINGHE 13
CONVENZIONI SINTATTICHE 15
CONVERSIONE 14
CONSOLE 49
COORDINATE
  grafiche 25
  pixel 44
CTS 11, 12
CURSORE 25, 14

D

DATI
  file 21
  memorizzazione 31
  strutture 51, 52, 33
  tipi 48
DCE 11
DEFine FUNCTION 23
DEFine PROCEDURE 23
DIMENSIONAMENTO 52
DTE 11

E

ELEMENTI 51, 52
ERRORI 16
ESPANSIONI
  rom 43
  periferiche 18
ESPRESSIONI 20

F

FILE 21
FILL 26
FINESTRE 22
FOR 28
FUNZIONI E PROCEDURE 23
FUNZIONI MATEMATICHE 24

G

GRAFICA
  sistema grafico 25
  tartaruga grafica 47

H

HANDSHAKE 12, 49

I

I/E
  finestre 22
  monitor 32
  periferiche 18
  qdos 39
INPUT
  canali 4
  finestre 22
  unità 49
INTERI 48
ISTRUZIONI 27
ITERAZIONI 28

J

JOYSTICK 29

L

LINEE 25

M

MAPPA DI MEMORIA 30
MICRODRIVE 31
MODE 53
MONITOR 32

MULTITASKING 39

N

NETWORK 42
NEXT 28
NOMI DI VARIABILE 33

O

OPEN 4
OPERATORI 34
OPERAZIONI FRA STRINGHE 35
ORDINAMENTO 13
OROLOGIO 36

P

PARAMETRI 23
PAROLE CHIAVE 37
PERIFERICHE 18
PIXEL 44
PORTE SERIALI 11
PROCEDURE 23
PROGRAMMI 38
PUNTI 25

Q

QDOS 39

R

RAM 39
RETE LOCALE 42
RGB 32
ROM 43
RS-232-C 11
RTS 11
RXD 11

S

SCALA 25
SEGNALI 11
SER1 11
SER2 11
SINTASSI 15
SISTEMA COORDINATE PIXEL 44
SISTEMA OPERATIVO 39
STIPPLE 9
STRINGHE
  confronti 13
  variabili 48
  vettori 51
SUONO 45

T

TARTARUGA GRAFICA 47
TEMPO 36
TIPO DI VARIABILI 48
TXD 11

U

UNITÀ
  colori 9
  con 49
  finestre 22
  modi 53
  scr 49

V

VARIABILI
  locali 23
  numeriche 48
  stringa 48
VETTORI ALFABETICI 51
VETTORI NUMERICI 52
VIDEO 53

QL Quill

CAPITOLO 1
IL QL QUILL

Il QL Quill un sofisticato programma di elaborazione di testi. E' stato ideato per darti la massima potenza e flessibilità, ed essere allo stesso tempo facile da imparare ed usare. Come vedrai più avanti, vieni costantemente informato di quello che puoi fare nella fase successiva e come farlo.

Puoi usare un elaboratore di testi in qualsiasi circostanza in cui useresti una macchina da scrivere. Le due macchine funzionano in un modo molto simile, anche se l'elaboratore di testi presenta molti vantaggi che non si riscontrano in una normale macchina da scrivere. La differenza più ovvia la facilità con cui possono essere corretti gli errori. Poiché il testo non viene stampato nel momento in cui lo scrivi, puoi fare tutte le correzioni che vuoi. Stamperai poi il testo quando sarai sicuro che corrisponde esattamente ai tuoi desideri e in questo modo avrai la garanzia di ottenere sempre risultati perfetti.

Nel leggere questo manuale ti renderai conto che ci sono molti altri vantaggi. Per esempio, nell'usare una macchina da scrivere bisogna battere il tasto di "a capo" alla fine di ogni riga. In Quill, questa funzione viene eseguita automaticamente. Quando il testo stampato raggiunge la fine della riga ne comincia una nuova. Premi **ENTER** quando vuoi iniziare un nuovo paragrafo. All'inizio di ogni nuova riga, ti accorgerai che la spaziatura del testo dell'ultima riga viene variata in modo che i margini destro e sinistro vengano allineati in tutto il testo. Questo processo, che viene chiamato *giustificazione*, dà un aspetto molto professionale al risultato finale, senza alcuno sforzo da parte tua. Come la maggior parte delle caratteristiche di Quill, la forma di giustificazione usata può essere modificata, a seconda di quello di cui hai bisogno.

Se in qualsiasi momento non sei sicuro di cosa fare, ricorda che puoi chiedere Aiuto premendo il tasto **F1**. Inoltre ricorda che puoi annullare qualsiasi operazione parzialmente completata (per esempio quando stai eseguendo un comando) premendo **ESC**.

CAPITOLO 2
AVVIAMENTO

CARICAMENTO DEL QL QUILL

Il QL Quill viene caricato come descritto nella Presentazione del programma. Al caricamento verrà visualizzato il seguente messaggio:

```
QL QUILL
elaboratore di testi
versione x.xx
Copyright ©1984 PSION Ltd
tutti i diritti riservati
```

in cui x.xx il numero della versione, per es. 2.23

Quill ha bisogno di utilizzare la cartuccia nel Microdrive 1 solo quando stai stampando un documento o chiedi Aiuto.

Mentre stai scrivendo un testo, Quill ha bisogno di una cartuccia nel Microdrive 2 solo se il testo più lungo di tre pagine. Quando necessario, Quill ti chiederà una cartuccia. Una volta inserita, la cartuccia non deve essere tolta finché il documento non sia memorizzato o abbandonato.

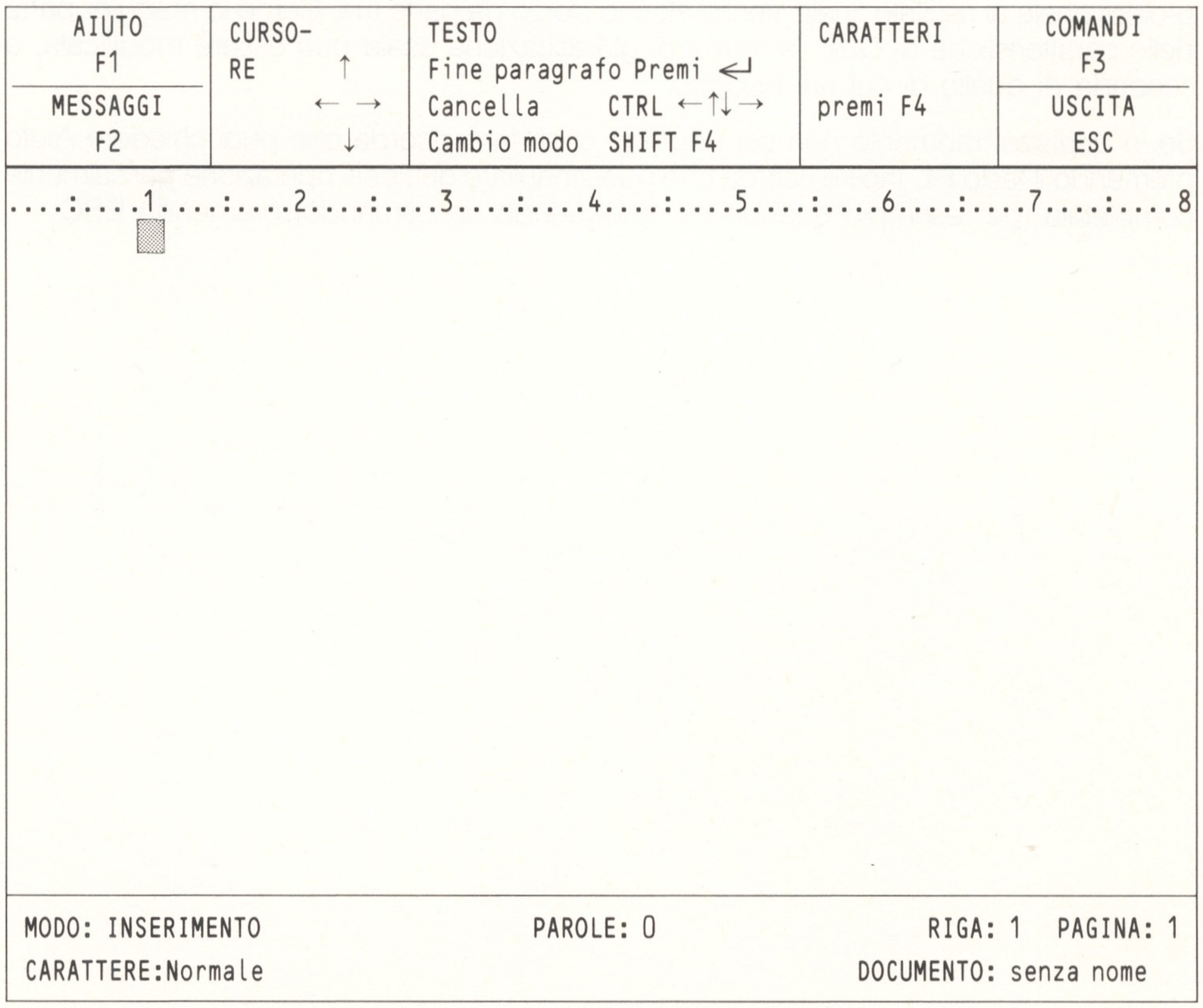


Figura 2.1. Lo schermo principale nel caso di impiego di un monitor (80 caratteri).

ASPETTO GENERALE

All'inizio lo schermo di Quill deve avere un aspetto simile alla figura 2.1 o alla figura 2.2. Esso viene indicato come lo *schermo principale*.

Con Quill puoi avere una visualizzazione di 80, 64 o 40 caratteri per riga di schermo. Se usi un apparecchio televisivo, forse la visione non sarà abbastanza chiara per leggere 80 caratteri per riga. In questo caso dovrai usare 64 o 40 caratteri. Lo schermo da 64 caratteri molto simile a quello da 80 caratteri. Lo schermo da 40 caratteri disposto differentemente e lo schermo principale sarà simile alla figura 2.2.

All'inizio Quill sceglie una visualizzazione da 80 o 64 caratteri – a seconda se hai premuto **F1** o **F2** all'accensione dell'elaboratore. In qualsiasi momento puoi cambiare le forme di visualizzazione usando il comando **Quadro**, che viene descritto più avanti.

Indipendentemente dalle differenze di aspetto Quill funziona esattamente nello stesso modo con tutte e tre le forme di visualizzazione. La maggior parte delle figure di questo manuale riportano lo schermo da 80 caratteri.

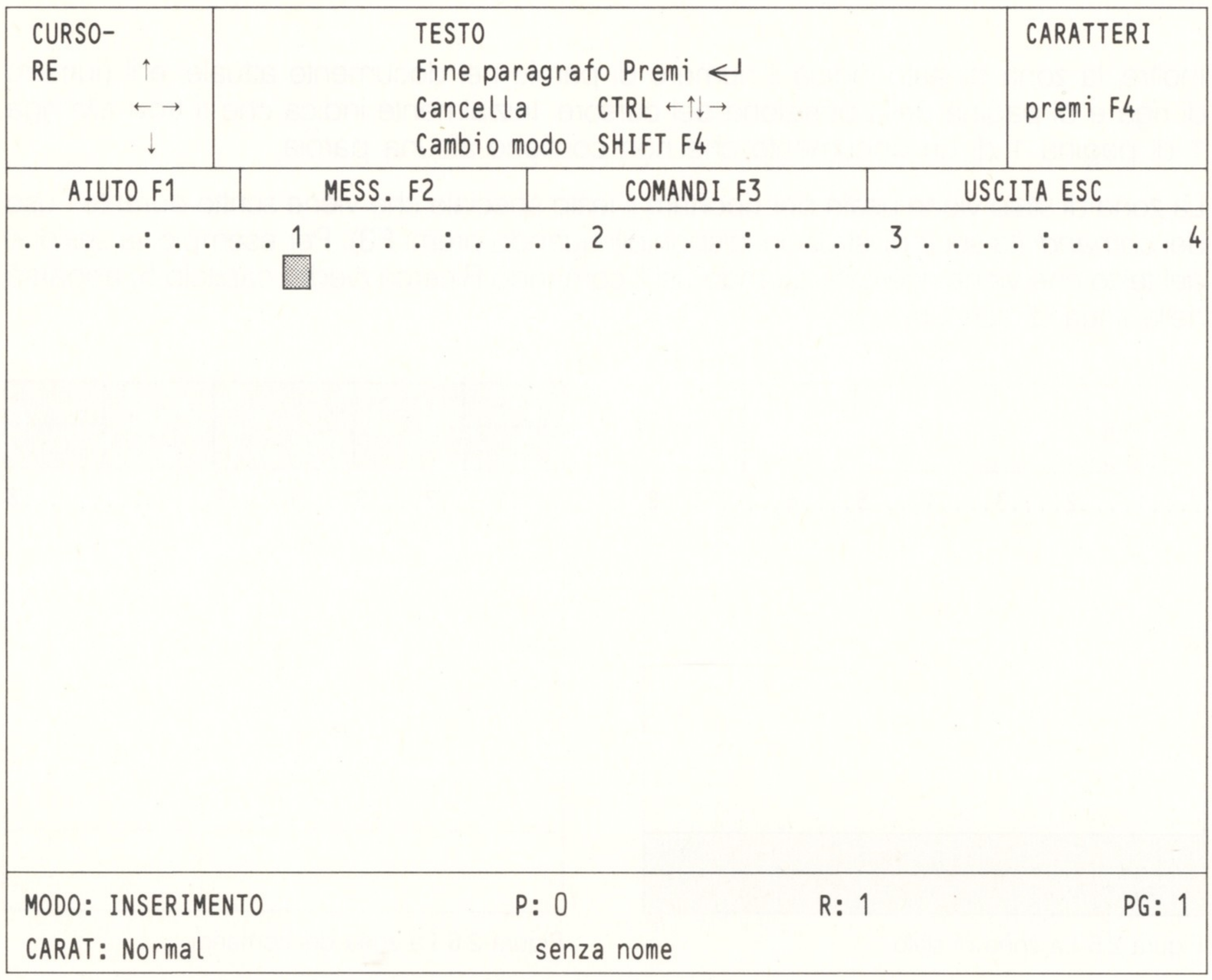


Figura 2.2. Lo schermo principale con 40 caratteri

Lo schermo diviso in tre sezioni principali: la *zona di visualizzazione*, la *zona di stato* e la *zona dei comandi*.

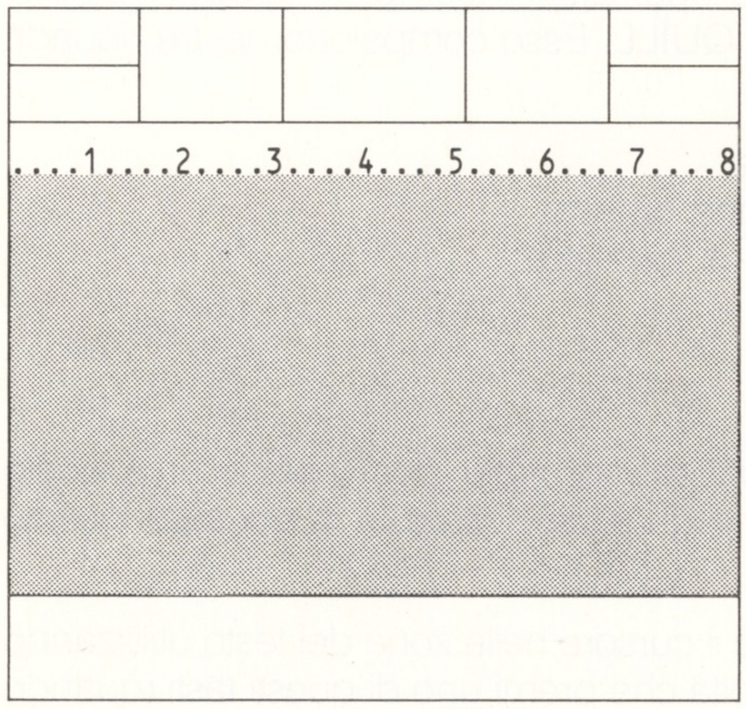


Figura 2.3 La zona di visualizzazione.

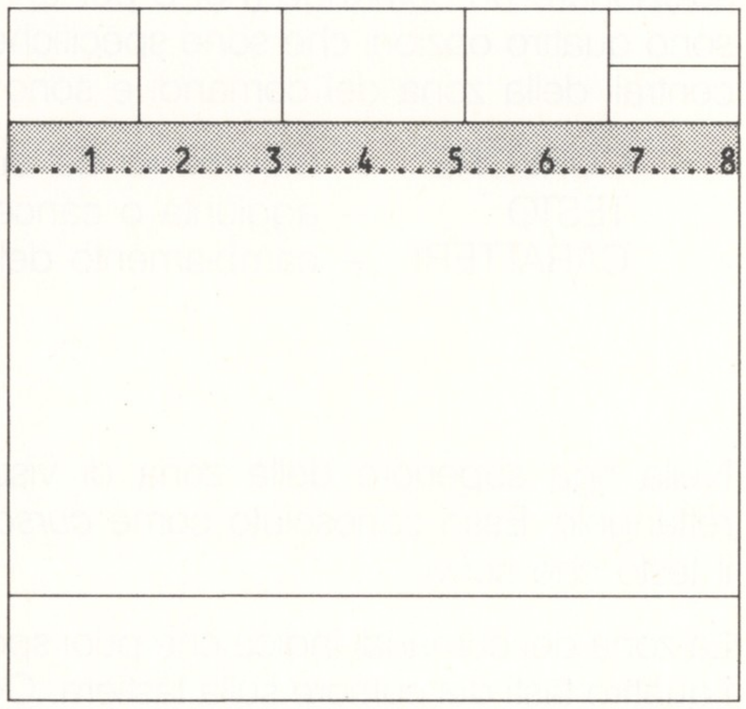


Figura 2.4 La riga indicatrice

La zona di visualizzazione

La zona maggiore, al centro dello schermo, riservata al testo del tuo documento. In questa zona comparirà quasi tutto quello che scrivi.

Lungo la zona di visualizzazione superiore si trova la *riga indicatrice*. Essa consiste in una fila di punti che marcano lo spazio di un carattere per tutta la larghezza dello schermo. Ogni cinque spazi c'èun duepunti (:) ed ogni dieci spazi c'èun numero progressivo.

La zona di stato.

La *zona di stato*, che utilizza le ultime tre righe dello schermo, riporta le informazioni relative al documento corrente. Per esempio, normalmente riporta il suo nome. All'inizio non avrai ancora dato un nome al documento, e Quill riporta le parole "senza nome". Quill riporterà questa scritta per qualsiasi testo tu scriva, fino a che non darai un nome al documento.

La zona di stato indica anche che Quill attualmente in *modo inserimento*, Questo significa che tutto quello che scrivi nel tuo documento verrà inserito (invece che scritto sopra il testo precedente). Indica anche che non c'ènessun *carattere* speciale, cioè che stai usando caratteri normali. Sono disponibili anche caratteri in grassetto (neretto), sottolineati, bassi rispetto al rigo (pedici), alti rispetto al rigo (apici) e italici e più avanti vedremo come usarli.

Inoltre, la zona di stato indica il numero di parole nel documento attuale, e il numero di riga e di pagina della posizione del cursore. Inizialmente indica che ti trovi alla riga 1 di pagina 1 di un documento che non contiene alcuna parola.

La zona di stato viene usata per riportare il testo speciale che viene scritto durante l'uso dei *comandi* (la serie di istruzioni disponibili quando premi **F3**). Per esempio, la sezione del testo che viene ricercata quando usi il comando Ricerca (vedi il capitolo 5) apparirà nella linea di scrittura.

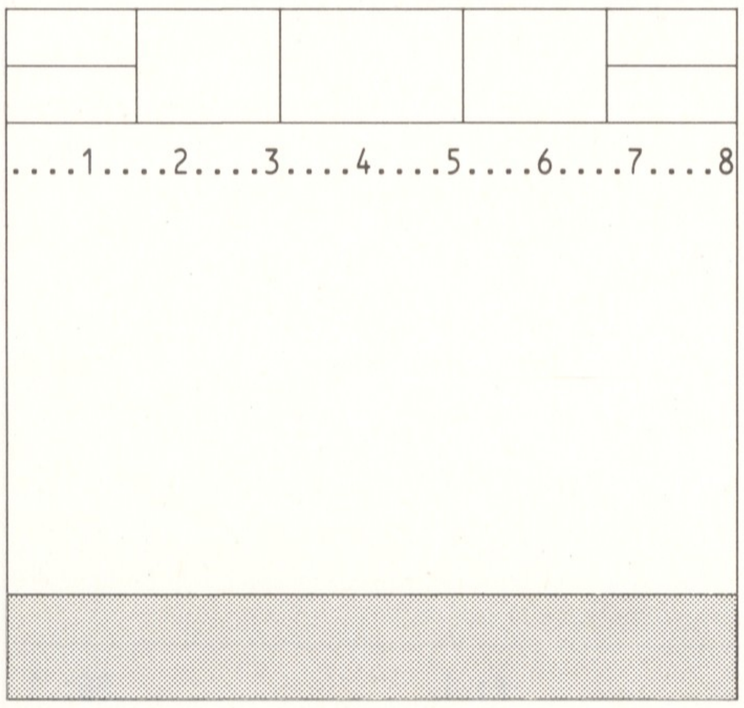


Figura 2.5 La zona di stato

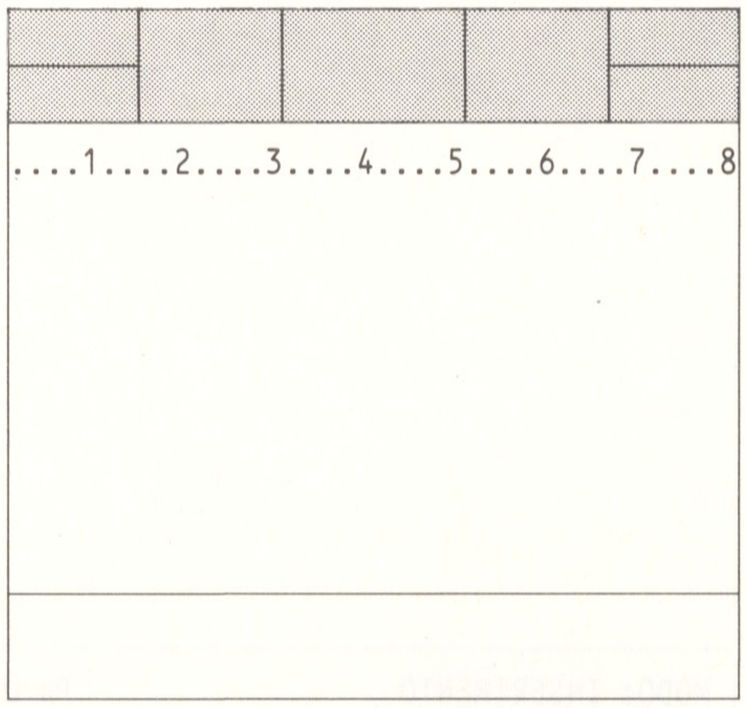


Figura 2.6 La zona dei comandi

La zona dei comandi

La zona dei comandi occupa alcune righe nella parte superiore dello schermo. Riporta le opzioni normali per ottenere Aiuto (**F1**), per attivare e disattivare messaggi (**F2**), per selezionare un comando (**F3**, e per annullare operazioni incomplete (**ESC**). Inoltre ci sono quattro opzioni che sono specifiche per QUILL. Esso compaiono nei tre riquadri centrali della zona dei comandi e sono:

CURSORE – movimento del cursore
TESTO – aggiunta o cancellazione di testo
CARATTERI – cambiamento del tipo di carattere

IL CURSORE

Nella riga superiore della zona di visualizzazione centrale puoi notare un piccolo rettangolo. Esso conosciuto come *cursore* e segnala la posizione in cui sarà inserito il testo che scrivi.

La zona dei comandi indica che puoi spostare il cursore nella zona del testo utilizzando i quattro tasti del cursore sulla tastiera. Ogni volta che premi uno di questi tasti (quando hai del testo nel tuo documento) il cursore si sposterà di uno spazio nella direzione indicata dalla freccia. Il cursore non oltrepasserà la fine del testo. Se non c'ètesto nel documento non puoi spostare il cursore dalla sua posizione originale.

Puoi anche spostare il cursore lungo il testo a passi maggiori. Se tieni premuto **SHIFT** e premi i tasti verso sinistra o destra del cursore, esso si sposterà a sinistra o a destra di una parola. Quando premi **SHIFT** assieme ai tasti verso l'alto o il basso del cursore, esso si sposterà indietro o in avanti di un paragrafo.

TESTO

L'opzione riportata al centro della zona dei comandi indica i vari modi in cui puoi cambiare il testo del documento. Il semplice scrivere sulla tastiera inserirà il testo alla posizione del cursore.

La seconda linea dell'opzione del testo indica che **ENTER** viene usato per segnare l'inizio di un nuovo paragrafo. Non è necessario premere **ENTER** quando arrivi alla fine di una riga del testo. Continua a scrivere finché non raggiungi la fine di una riga di testo. Continuando a scrivere, le nuove parole appariranno automaticamente sulla seconda linea e la spaziatura delle parole sulla prima riga verrà aggiustata. Questa la *giustificazione*, che controlla il modo in cui il testo viene allineato rispetto ai margini destro e sinistro.

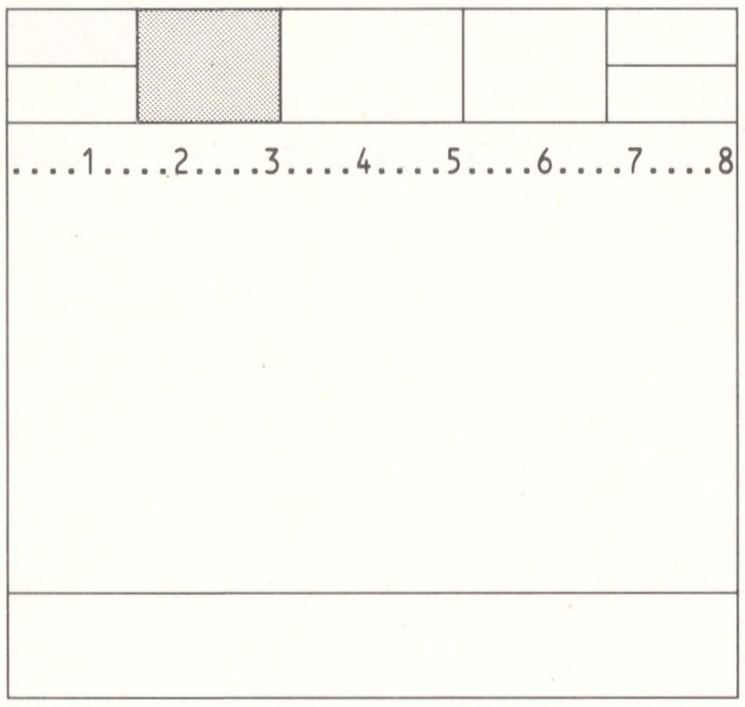


Figura 2.7 Movimento del cursore

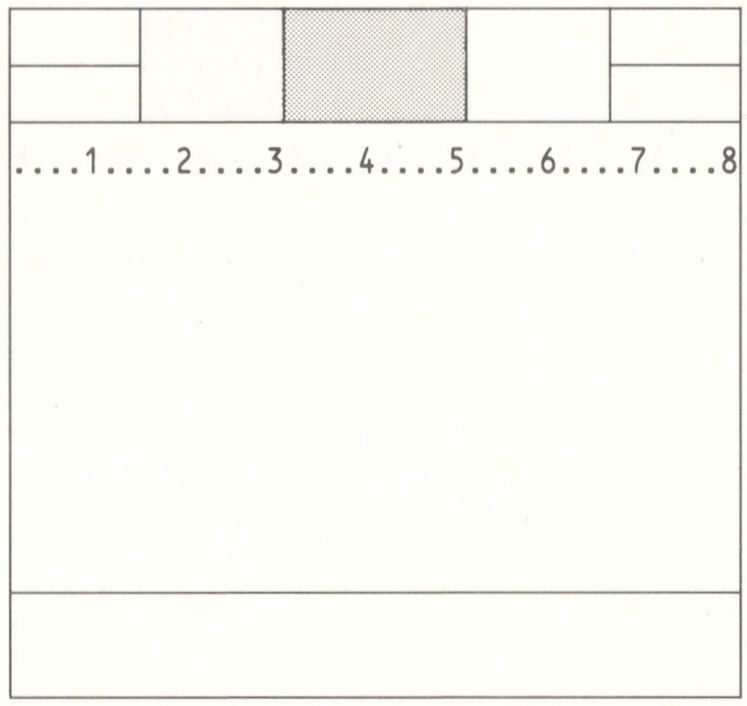


Figura 2.8 Le opzioni del testo

Prova a premere **ENTER** e a scrivere dell'altro testo per vedere cosa succede quando inizi un nuovo paragrafo. Non preoccuparti se il rientro del nuovo paragrafo non è quello che desideri – troverai come cambiarlo nel Capitolo 4.

Nel tuo documento puoi scrivere caratteri che non sono indicati sulla tastiera. Essi vengono ottenuti premendo **CTRL** oppure **CTRL** e **SHIFT** ed un altro tasto. La guida alla programmazione contiene un elenco completo dei caratteri ottenibili, assieme ai tasti da utilizzare.

Puoi cancellare testo, un carattere alla volta, sia a sinisctra che a destra della posizione del cursore. Tieni premuto **CTRL** e premi il tasto di movimento verso sinistra o verso destra del cursore.

Mentre scrivi del testo noterai dei cambiamenti nella zona di stato nella parte inferiore dello schermo. I conteggi di parola e riga concorderanno sempre con il contenuto del documento. Il resto della zona di stato non sarà cambiato. In particolare, il documento sarà ancora senza nome. Il nome al documento viene dato al momento di memorizzarlo su una cartuccia di Microdrive (come descritto nel capitolo 7).

Ora che hai del testo nel tuo documento, puoi provare a spostare il cursore lungo il testo usando tutti e quattro i tasti del cursore. Quando hai finito, porta il cursore alla fine del testo.

CARATTERI

Un'ulteriore opzione nella zona dei comandi è chiamata **CARATTERI** ed è usata per modificare l'aspetto del testo nel tuo documento.

Premi **F4** ed avrai a disposizione cinque scelte:

usare caratteri in grassetto (neretto)
visualizzare caratteri sopra il rigo (apici)
visualizzare caratteri sotto il rigo (indici)
produrre testo sottolineato
"evidenziare" il testo esistente.

Queste scelte vengono attuate premendo **F4** e quindi un singolo tasto dall'elenco riportato nella zona dei comandi. Come esempio utilizziamo questa opzione per produrre del testo sottolineato. Premi **F4**, e poi il tasto **S**.

Lo schermo ritorna normale e sembra che non sia cambiato nulla, eccetto che il carattere indicato come "Sottolineato" nella zona di stato. Se batti dell'altro testo noterai che viene scritto sullo schermo già sottolineato.

In QUILL vedi esattamente ciò che apparirà nella versione finale stampata. Le sole cose che non sono sempre visibili sullo schermo sono i margini di pagina inferiore e superiore, le intestazioni e i piè di pagina, e la spaziatura tra le linee (quando scegli una spaziatura doppia o tripla). QUILL non le mostra poiché ridurrebbero la quantità di testo visibile sullo schermo.

Per disattivare la sottolineatura, premi nuovamente i tasti **F4** e **S**. Se ora scrivi altre parole vedrai che non sono sottolineate – l'opzione di sottolineatura funziona come un interruttore di acceso/spento.

Nel Capitolo 4 troverai una descrizione più completa della sottolineatura e delle altre tre opzioni di carattere.

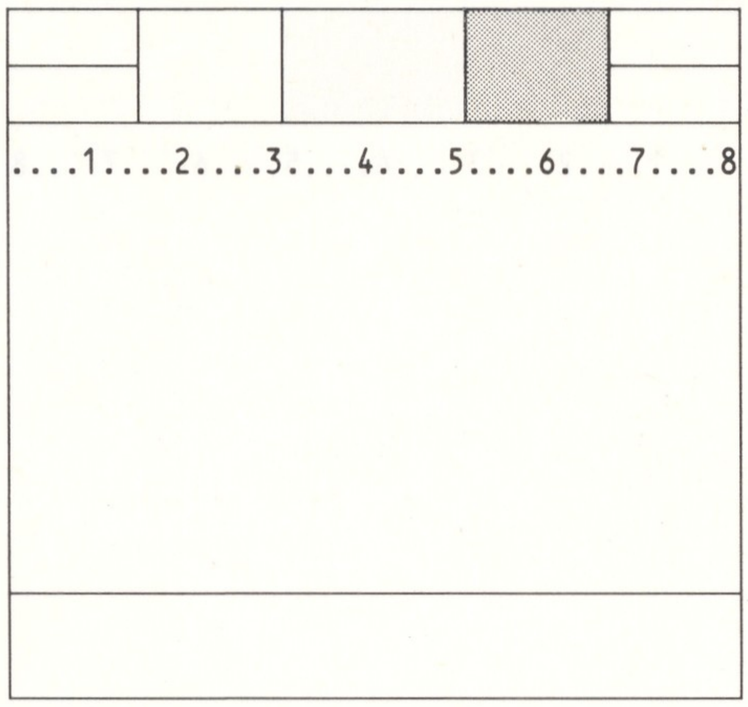


Figura 2.9 Caratteri

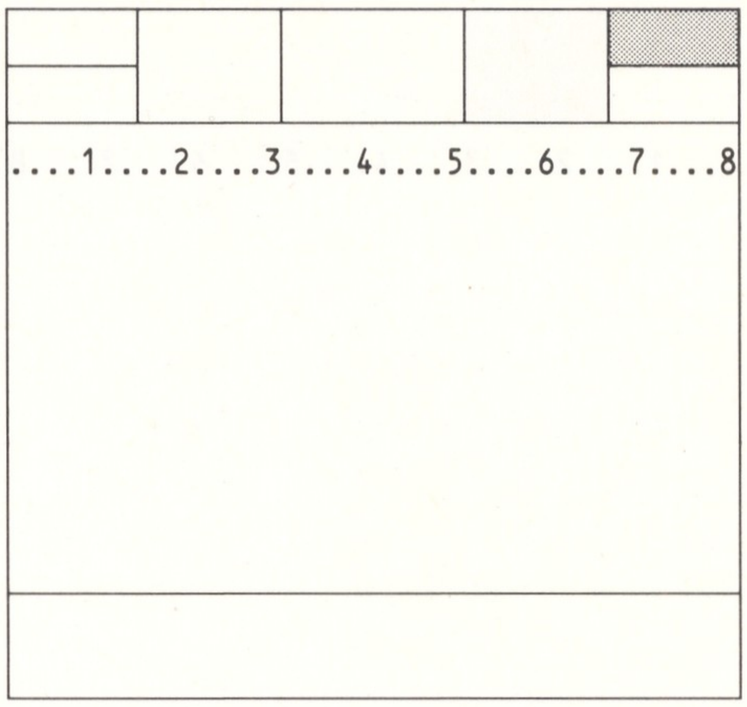


Figura 2.10 I comandi

COMANDI

Scegli un comando premendo **F3**. L'elenco nella zona dei comandi viene indicato col nome di *menù dei comandi*.

Puoi scegliere qualsiasi comando riportato nel menù (elenco) al centro della zona dei comandi premendo il tasto corrispondente alla sua prima lettera.

QUILL ha più comandi di quelli che possono essere visualizzati nel menù dei comandi. Quindi hai a disposizione due elenchi e puoi passare dall'uno all'altro con il comando Nuovo elenco.

Poiché alcuni comandi iniziano con la stessa lettera, importante accertarsi che il comando voluto sia visualizzato nella zona di comando prima di selezionarlo.

La descrizione dei vari comandi occuperà buona parte del resto di questo manuale. Per il momento descriveremo l'uso di due soli comandi: Lasciá e Zero.

Lascia viene usato al termine delle operazioni con Quill. Per tornare a SuperBASIC usando **Lascia** premi i tasti **F3** e L. A questo punto **Lascia** ti chiederà se vuoi memorizzare il tuo documento attuale su una cartuccia di Microdrive prima di abbandonare Quill. Per memorizzarlo premi **ENTER**, per abbandonarlo premi **A**.

Puoi anche premere **ESC** per annullare il comando.

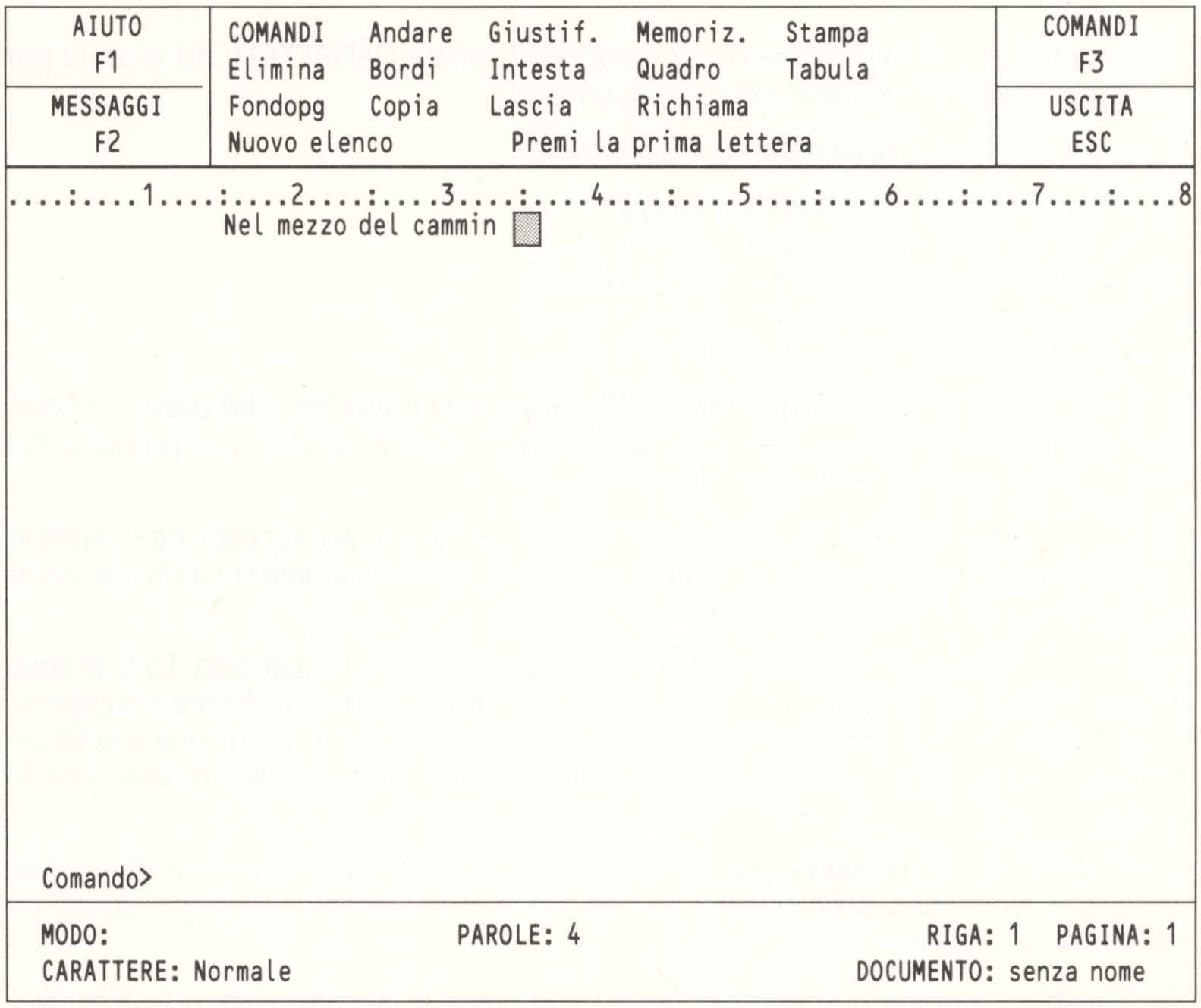


Figura 2.11 Il primo menù dei comandi.

Il comando **Zero** si trova nel menù COMANDI II, perciò, prima di selezionare **Zero** devi usare **Nuovo**. Devi premere **F3**, **N** oppure 2 volte **F3** poi **Z**. **Zero** cancella dalla memoria il testo del documento attuale, ma non ritorna a SuperBASIC.

```
AIUTO                COMANDI II  Ricerca        Unione          COMANDI
F1                   File        Sostituzione   Zero            F3
MESSAGGI             Pagina      Trattino                       USCITA
F2                   Nuovo elenco     Premi la prima lettera    ESC

....:....1....:....2....:....3....:....4....:....5....:....6....:....7....:....8
              Nel mezzo del cammin

Comando II>

MODO:                              PAROLE: 4          RIGA: 1   PAGINA: 1
CARATTERE: Normale                                    DOCUMENTO: senza nome
```

Figura 2.12 Il secondo menù dei comandi

Se cancelli il testo prima di averlo memorizzato su una cartuccia di Microdrive, non potrai ripristinarlo, ma dovrai riscriverlo. Perciò Quill ti chiede di confermare la scelta premendo **ENTER**. In alternativa, puoi premere **ESC** per cancellare il comando e ritornare al documento.

CAPITOLO 3

EDIZIONE AL CURSORE

In questo capitolo imparerai come usare le facili modalità di edizione di QUILL. Il testo viene cambiato sempre in corrispondenza della posizione del cursore. Quindi devi usare i tasti del cursore per portarlo nella posizione opportuna prima di fare delle variazioni.

Questo metodo di edizione conosciuto, per ovvi motivi, come *edizione al cursore*. Abituati ad usare queste tecniche su un testo scritto da te, oppure usa un testo fornito da Quill. Se scrivi il tuo testo, non preoccuparti degli errori che fai. In effetti può essere una buona idea aggiungere deliberatamente degli errori – ogni errore ti darà un' ulteriore occasione di conoscere i comandi di edizione.

INSERIMENTO DI TESTO

Quill è inizialmente nel *modo inserimento*, quindi il testo che scrivi viene automaticamente inserito nella posizione del cursore. Per inserire lettere o parole nel mezzo del testo devi fare quanto segue:

Porta il cursore, usando i quattro tasti del cursore, nel punto dove vuoi fare l' inserimento.

Scrivi le lettere o le parole che vuoi inserire. I caratteri vengono inseriti immediatamente sotto la posizione del cursore, e il testo esistente si sposta a destra per fare spazio.

Il testo viene riaggiustato automaticamente mentre fai l' inserimento.

Se vuoi inserire parecchie parole sarebbe fastidioso dover aspettare che tutto il testo venga aggiustato ogni volta che premi un tasto. Quill individua questa situazione e reagisce dividendo la riga nel punto in cui stai inserendo il testo. Questa operazione viene chiamata *divisione automatica del testo*. A questo punto puoi scrivere quanto vuoi.

Quill ripristinerà in quel punto il testo alla fine dell' inserimento (cio quando premi un tasto di cursore, o un tasto di funzione o **ESC**).

CANCELLAZIONE DEL TESTO

Anche la cancellazione del testo nella posizione di cursore è molto semplice. Viene usato il tasto **CTRL** assieme ai tasti del cursore.

Per vedere l' effetto del tasto di movimento verso sinistra del cursore, metterlo immediatamente dopo il carattere da cancellare. Ora tieni premuto il tasto **CTRL** e premi brevemente il tasto del cursore verso sinistra. La lettera immediatamente a sinistra della posizione del cursore verrà cancellata ed il cursore si sposterà di una posizione verso sinistra. Ogni volta che premi il tasto del cursore a sinistra, con il tasto **CTRL** abbassato, verrà cancellata una lettera. Se desideri cancellare parecchie lettere puoi tenere premuti i tasti **CTRL** e cursore verso sinistra, usando la funzione di autoripetizione. Premi sempre il tasto **CTRL** prima del tasto del cursore.

Se usi il tasto **CTRL** assieme al tasto di movimento verso destra del cursore, il testo verrà cancellato, carattere per carattere, da sotto la posizione del cursore, ed il testo a destra si sposterà per riempire la posizione lasciata libera.

Puoi cancellare intere parole alla volta, sia a sinistra che a destra del cursore, usando insieme **SHIFT** e **CTRL** e premendo il tasto di movimento verso destra o sinistra del cursore.

Puoi cancellare tutta la riga a destra o sinistra del cursore. Tieni premuto il tasto **CTRL** e premi il tasto di movimento verso l' alto del cursore. La riga a sinistra del cursore scomparirà. Analogamente, se premi il tasto di movimento verso il basso del cursore, cancellerai tutta la riga a destra del cursore.

In tutti i casi il testo verrà riaggiustato automaticamente.

RISCRITTURA

Nel modo riscrittura puoi riscrivere sul testo esistente e sostituirlo con testo nuovo.

Puoi passare al modo riscrittura tenendo premuto **SHIFT** e premendo **F4**. L' indicatore di modo sulla sinistra della zona di stato cambia da "INSERIMENTO" a "RISCRITTURA" per indicare che il testo battuto sulla tastiera sostituirà il testo esistente. Se premi nuovamente **SHIFT** e **F4** ritorni al modo inserimento.

Con Quill impostato nel modo riscrittura, posiziona il cursore all' inizio del testo che vuoi cancellare e scrivi il testo sostitutivo sopra quello vecchio. Quando hai terminato di fare sostituzioni, ritorna al modo inserimento premendo di nuovo **SHIFT** e **F4**.

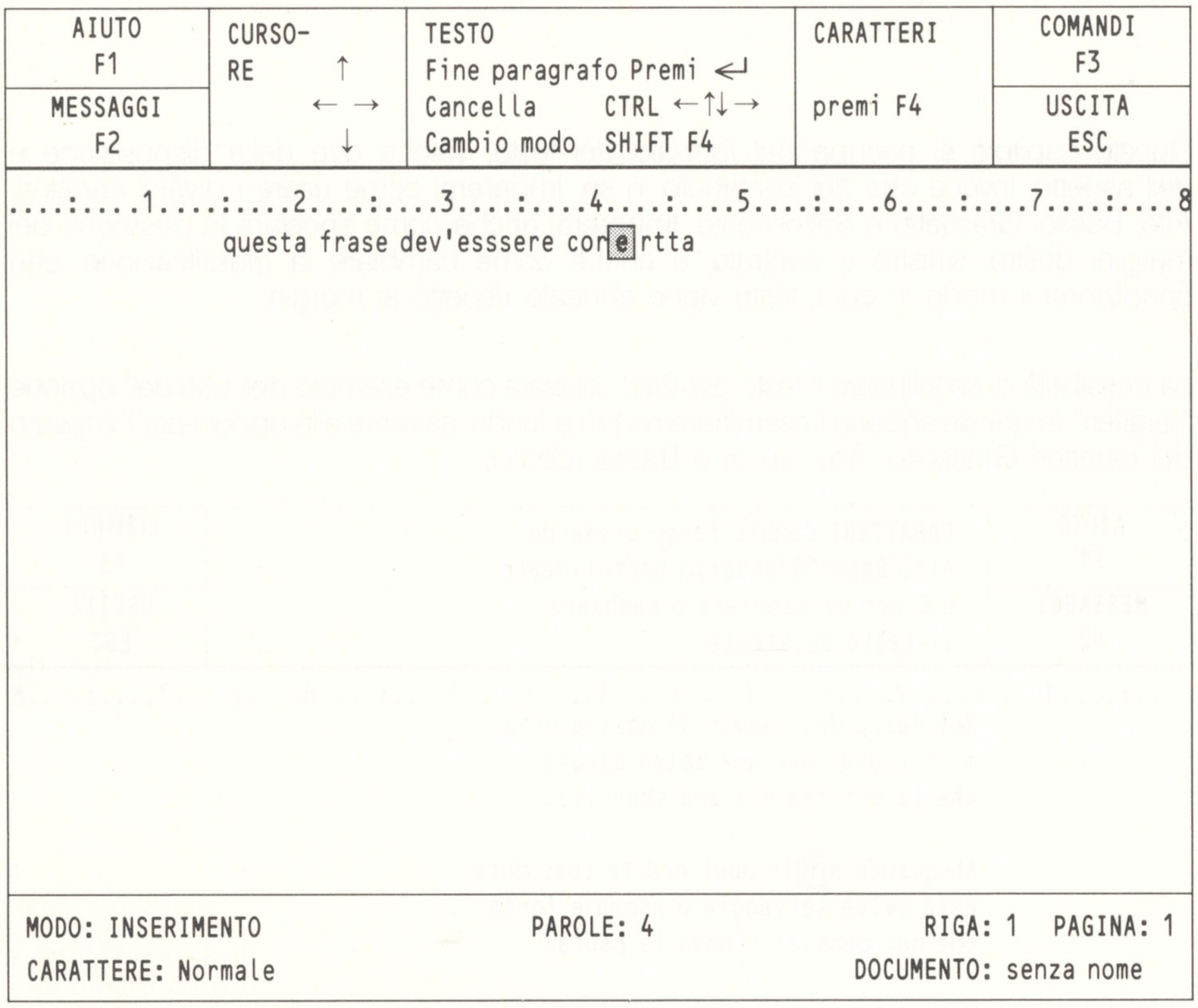


Figura 3.1 Riscrittura.

La figura 3.1 indica una situazione tipica in cui hai bisogno di usare il modo riscrittura. Con lo schermo illustrato, e con il cursore sulla "e" di "corerta", puoi riscrivere "r" ed "e" per correggere la parola. (Puoi anche impratichirti nella cancellazione di una lettera togliendo una delle "s" da "esssere"!).

CAPITOLO 4
FORMATO DEL TESTO

Questo capitolo si occupa del *formato* del testo; vale a dire della disposizione e dell'aspetto, invece che del contenuto in sé. Imparerai come usare i diversi caratteri, Alto, Basso, Grassetto e Sottolineato. Imparerai anche come spostare la posizione dei margini destro, sinistro e rientrato, e anche come cambiare la giustificazione, che condiziona il modo in cui il testo viene allineato rispetto ai margini.

CARATTERI

La possibilità di sottolineare il testo già stata utilizzata come esempio dell'uso dell'opzione "caratteri". In questa sezione l'esamineremo più a fondo, assieme alle opzioni per l'impiego dei caratteri Grassetto, Alto (apici) e Basso (pedici).

```
| AIUTO F1          | CARATTERI Cambia forma premendo                  | COMANDI F3  |
|                   | Alto Basso Grassetto Sottolineato                |             |
| MESSAGGI F2       | o E per evidenziare o cambiare                   | USCITA ESC  |
|                   | il testo esistente.                              |             |

....:....1....:....2....:....3....:....4....:....5....:....6....:....7....:....8
          Nel mezzo del cammin di nostra vita
          mi ritrovai per una selva oscura
          che la diritta via era smarrita.

          Ah quanto a dir qual era te cose dura
          esta selva selvaggia e aspra e forte
          che nel pensier rinova la paura!

          Tant'te amara che poco te pitu morte;
          ma per trattar del ben ch'io vi trovai,
          dirto dell'altre cose ch'i' v'ho scorte.

          Io non so ben ridir com'io v'entrai,
          tant'era pien di sonno a quel punto
          che la verace via abbandonai.

               Ah quanto a dir

Caratteri>

MODO:                          PAROLE: 85          RIGA: 17  PAGINA: 1
CARATTERE: Sotto                                   DOCUMENTO: senza nome
```

Figura 4.1 Scelta di un tipo di carattere.

Generalmente puoi scegliere qualsiasi opzione premendo **F4** e quindi la lettera appropriata – **G**rassetto, **S**ottolineato, **A**lto o **B**asso. Se in un dato momento attivata una di queste opzioni, puoi disattivarla di nuovo usando esattamente lo stesso metodo con cui l'hai attivata – premendo **F4** e quindi la lettera corrispondente.

Nota che il testo che scrivi sarà sempre nel carattere indicato nella zona di stato. Se sposti il cursore in una sezione scritta in grassetto, per esempio, la zona di stato indicherà carattere Grassetto, e le parole che scriverai in questa sezione saranno anch'esse in grassetto. Il carattere cambia automaticamente appena ti sposti in una sezione che contiene un carattere diverso.

Naturalmente, nel caso di **A**lto o **B**asso puoi avere solo una delle due opzioni in un dato momento. Se scegli una di esse, l'altra viene automaticamente disattivata.

L'opzione CARATTERI può essere utilizzata in tre modi:

Inserire nuovo testo in un particolare carattere
Variare il carattere del testo esistente
Cambiare o togliere un carattere esistente

Se vuoi scrivere del testo in un carattere particolare devi premere **F4** e selezionare il carattere desiderato. Da quel momento il testo che scrivi apparirà nel tipo di carattere scelto. Quando vuoi ritornare al testo normale devi disattivare il carattere premendo **F4** e quindi la lettera relativa al tipo di carattere che vuoi togliere.

La procedura per cambiare il carattere usato nel testo esistente è molto facile. Il metodo viene chiamato *evidenziamento* perchè pone in evidenza i nuovi caratteri, passandoci sopra con il cursore.

Per prima cosa porta il cursore all'inizio del testo da cambiare, poi premi **F4** e poi il tasto **E**, per scegliere l'opzione evidenziamento. Poi, seleziona la combinazione di tipi di caratteri che desideri. Usa i tasti di movimento del cursore verso destra e verso il basso per sportare il cursore sul testo che vuoi cambiare. Quando arrivi alla fine del testo che vuoi modificare, esci dall'opzione premendo **ENTER**. Non c' bisogno che disattivi la scelta dei caratteri; esso ritornerà al carattere corretto appena sposti il cursore dalla zona evidenziata con il nuovo carattere. La figura 4.2 mostra l'aspetto dello schermo mentre si evidenzia lo schermo con la sottolineatura.

```
AIUTO F1          CARATTERI Cambia forma premendo                COMANDI F3
                  Alto Basso Grassetto Sottolineato
MESSAGGI F2       poi usa ↓ ou → per evidenziare il testo        USCITA ESC
                  esistente. Premi ↵ per finire
....:....1....:....2....:....3....:....4....:....5....:....6....:....7....:....8
          Nel mezzo del cammin di nostra vita
          mi ritrovai per una selva oscura
          che la diritta via era smarrita.

          Ah quanto a dir qual era me cosa dura
          esta selva selvaggia e aspra e forte
          che nel pensier rinova la paura!

          Tant'me amara che poco me pimu morte;
          ma per trattar del ben ch'io vi trovai,
          dirmo dell'altre cose ch'i' v'ho scorte.

          Io non so ben ridir com'io v'entrai,
          tant'era pien di sonno a quel punto
          che la verace via abbandonai.

          Ah quanto a dir qual era me cosa dura
MODO:                        PAROLE: 89          RIGA: 17  PAGINA: 1
CARATTERE: Sotto                                 DOCUMENTO: senza nome
```

Figura 4.2 Evidenziamento della sottolineatura.

E' possibile cambiare, o togliere, un carattere esistente usando il sistema usato per aggiungere un nuovo carattere al testo esistente. Anche in questo caso devi portare il cursore all'inizio del testo prima di premere **F4**. Premi il tasto **E** e poi scegli (o disattiva) la combinazione di caratteri che desideri. Passa il cursore sopra il testo che desideri cambiare e poi premi **ESC**.

Quando cambi il carattere esistente di un testo, Quill non ricorda il carattere originale. Per esempio, immaginiamo che cambi del testo che era originariamente sottolineato e lo faccia diventare in grassetto. Se successivamente togli il carattere grassetto, il testo finale sarà in carattere normale, e non ritornerà sottolineato.

MARGINI

La larghezza dei margini viene cambiata con il comando **Bordi**. Ogni nuova posizione di margine ha effetto dal paragrafo corrente ed influisce su tutti i paragrafi seguenti, fino a che fai un'altra variazione alla posizione del margine.

Per usare questo comando premi il tasto **F3** e poi il tasto **B**. Oltre ad altri cambiamenti nella zona di comando, vedrai che compariranno tre scelte – **SINISTRA, RIENTRO** e **DESTRA**, e che evidenziata l'opzione SINISTRA. Queste opzioni rappresentano i tre margini, e puoi muovere quella che evidenziata. Puoi passare da una opzione all'altra usando lo spazio, o puoi scegliere una opzione particolare premendo il tasto corrispondente alla sua prima lettera. Quando il nome di un margine viene evidenziato nella zona di comando puoi cambiare la posizione di quel margine con i tasti di movimento verso sinistra o destra del cursore.

Immaginiamo che tu desideri spostare il margine sinistro verso destra di tre caratteri, iniziando dal secondo paragrafo del documento.

Per prima cosa porta il cursore in qualsiasi punto del secondo paragrafo e poi premi:

F3 B

Come indicato dall'evidenziamento, puoi muovere il margine sinistro, per cui devi solo premere tre volte il tasto del cursore verso destra. Il cambimento del margine si verifica immediatamente, per cui puoi vederne l'effetto prima di uscire dal comando.

Puoi uscire immediatamente dal comando premendo **ENTER**, o puoi continuare a fare altri cambiamenti di margine. Premi la barra dello spazio fino a che hai scelto il margine giusto e spostalo con i tasti di movimento del cursore verso destra e sinistra. Puoi usare i tasti del movimento verso l'alto o il basso del cursore per portarlo ad un altro paragrafo e fare ulteriori cambiamenti ai margini. Dopo che hai fatto tutti i cambiamenti desiderati, puoi uscire dal comando premendo **ENTER**.

Il margine di rientro segna la posizione usata per il primo carattere di un nuovo paragrafo. Per uno schermo di 80 caratteri esso viene inizialmente impostato nella posizione del quindicesimo carattere.

Non c' alcuna limitazione alle posizioni rispettive del rientro e del margine sinistro. Se non vuoi usare i paragrafi rientrati, puoi spostarli in modo che coincidano. Puoi persino mettere il margine di rientro a sinistra del margine sinistro. Questo è utile per produrre paragrafi numerati, come illustrato nel seguente esempio:

```
Margine rientrato
|
|  Margine sinistro
|  |
1) Questo è il primo di due
   paragrafi che mostrano come
   possibile usare i margini rientrati.
2) Il margine rientrato si trova a sinistra
   di tre caratteri rispetto al margine
   sinistro.
```

In questo caso, se inizi un nuovo paragrafo (premendo **ENTER** esso verrà scritto nella posizione "rientrata". Tutto il testo successivo verrà collocato tra le posizioni del margine destro e sinistro finché non premi nuovamente **ENTER**.

GIUSTIFICAZIONE

Il comando **Giustifica** permette di variare il tipo di allineamento usato nel documento. Come il comando **Bordi**, tutti i cambiamenti hanno effetto dal paragrafo corrente (quello che contiene il cursore) e rimangono validi fino alla fine del documento, o fino al prossimo cambiamento di giustificazione. Quando scegli questo comando, vedrai che ti viene offerta la scelta di giustificazione a sinistra, in centro o a destra.

Inizialmente Quill effettua la giustificazione *Destra*. Il testo allineato ai margini destro e sinistro, producendo un testo dall'aspetto simile a questo manuale. Se in una riga non ci sono caratteri sufficienti per far corrispondere i margini, verranno aggiunti spazi aggiuntivi tra le parole finché i bordi saranno allineati. L'aspetto finale è molto professionale. Comunque, se in un documento vengono usate un numero eccessivo di parole lunghissime o con trattino, possono risultare degli spazi dall'aspetto sgradevole.

Se scegli la giustificazione a *Sinistra*, premi **S** dopo aver chiamato il comando **Giustifica**. Questo produrrà un testo il cui aspetto sarà simile a questo paragrafo. Il margine sinistro allineato, ma la spaziatura del testo sulla riga non viene aggiustata, per cui il margine destro sarà irregolare.

La giustificazione nel *Centro*, scelta usando l'opzione **C** del comando **Giustifica**, provoca la centratura del testo di ciascuna riga tra i margini destro e sinistro. In questo caso il testo avrebbe l'aspetto di questo paragrafo. La giustificazione al centro è utile, per esempio, per centrare le intestazioni e i titoli, o per mettere diciture a diagrammi.

In modo analogo al comando **Bordo**, puoi premere i tasti di movimento del cursore verso l'alto o il basso e spostarti ad un altro paragrafo e fare ulteriori cambiamenti della giustificazione. Premi **ENTER** per abbandonare il comando.

CAPITOLO 5
COMANDI DI EDIZIONE

Questo capitolo amplierà le tue conoscenze relative alle possibilità di modifica del testo, e comprenderà la copiatura, il movimento e la cancellazione di blocchi. In aggiunta, verranno presentate le tecniche estremamente potenti di ricerca e sostituzione. Questi servizi sono disponibili mediante i comandi di edizione di Quill – **Copia, Elimina, Ricerca** e **Sostituisci.**

COPIA

Oltre a copiare blocchi di testo da un posto all'altro nel documento, il comando **Copia** ti permette di spostare blocchi di testo.

La sola differenza tra la copiatura e il movimento del testo consiste nel fatto che, nel caso di un **copia**, il testo originale viene lasciato in posizione, per cui puoi avere due "copie". Puoi utilizzarlo per esempio, se hai bisogno di creare una tabella, con un testo ripetuto molte volte, o se vuoi vedere il punto migliore per inserire un particolare paragrafo.

Se muovi del testo, la nuova copia viene inserita e la vecchia viene cancellata, per cui ti rimane usa sola versione.

Il comando **Copia** ti dà l'opzione di conservare o cancellare la vecchia versione, per cui ti dà entrambe le possibilità in un solo comando.

Quando scegli il comando **Copia**, (premendo **F3** e poi il tasto **C**), per prima cosa devi portare il cursore all'inizio del testo che vuoi copiare e poi premere **ENTER**. Porta poi il cursore alla fine del testo da copiare. Quando sposti il cursore, il testo che sarà interessato dal comando viene evidenziato, per cui facile vedere quanto testo sarà copiato. Se per sbaglio marchi troppo testo, puoi usare i tasti di movimento verso l'alto o sinistra del cursore, ma non puoi superare il punto di partenza. Dopo aver segnato il testo devi premere nuovamente **ENTER**.

Ti viene quindi chiesto se vuoi conservare la vecchia versione. Devi premere **C** per conservare la vecchia versione (e produrre l'effetto di copiatura) o premere **ENTER** per accettare il suggerimento di cancellarlo. In risposta al sollecito successivo devi portare il cursore nel punto in cui vuoi inserire il testo prescelto e premere il tasto **C**. La copia verrà fatta e inserita immediatamente.

A questo punto puoi terminare il programma premendo **ENTER**, il che ti riporta allo schermo principale.

Comunque, hai la possibilità di fare copie ulteriori dello stesso testo in altri punti del tuo documento. Devi semplicemente portare il cursore nel punto dove vuoi un'altra copia e premere il tasto **C**. Puoi ripetere questo procedimento quante volte vuoi. Mentre fai queste copie addizionali non ti viene chiesto se conservare o cancellare la vecchia versione. Quando avrai finito di fare le copie devi premere **ENTER** per uscire dal comando.

Come normale in Quill, se premi **ESC** annulli delle azioni parzialmente completate, ma non distruggi quello che stato completato. Se premi **ESC**, tutte le copie fatte rimarranno nel testo.

ELIMINA

Questo comando dev'essere usato per cancellare larghe porzioni di testo dal documento (premi **F3** e poi **E**). Ricordati che più semplice cancellare piccole parti di testo con le funzioni di edizione al cursore descritte nel capitolo 3.

Analogamente al comando **Copia**, ti viene chiesto di portare il cursore all'inizio del testo da eliminare e poi di premere **ENTER**. Devi poi portare il cursore alla fine del testo. Anche in questo caso il testo interessato viene evidenziato. Dopo aver segnato il quantitativo corretto di testo devi premere **ENTER** ed il testo marcato verrà eliminato immediatamente.

RICERCA

Il comando **Ricerca** ti permette di cercare una parola o frase particolare, in tutto o parte del documento. Puoi usarlo, per esempio, per controllare se hai ripetuto una parola o frase particolare troppo spesso. La prima ricerca inizierà all'inizio del testo, ma successivamente può essere continuata dalla posizione corrente del cursore.

Il comando **Ricerca** si trova nel secondo menù di comandi, per cui lo scegli premendo **F3**, **N** e poi **R**.

Quando usi il comando, ti viene chiesto di scrivere il testo che vuoi trovare, e di finire con **ENTER**. Quill inizia immediatamente ad esaminare il documento dall'inizio, finché trova la prima volta che compare il testo. Il cursore rimane nella posizione dell'inizio del testo trovato. Se quello che cerchi, puoi uscire dal comando premendo **ENTER**.

Comunque, una volta che hai dato al comando **Ricerca** del testo da cercare, puoi usarlo ancora per trovare quando il testo si presenta nuovamente. Invece di premere **ENTER**, premi il tasto **C**. Se lo fai, Quill continua la ricerca dalla posizione corrente del cursore finché non trova la prossima volta che il testo dato si presenta. Puoi ripeterlo tutte le volte che vuoi, trovando le occorrenze successive. Premi **ENTER** per uscire dal comando quando hai trovato il testo che desideri.

Se, in qualsiasi fase, Quill non trova nuovamente il testo nel documento, te lo dice e attende che tu prema la barra spaziatrice. Quando premi un tasto Quill ritorna allo schermo principale.

SOSTITUZIONE

Il comando **Sostituzione** simile al comando **Ricerca**, ma in aggiunta ti dà la possibilità di sostituire il testo trovato alcune volte o tutte le volte. Il comando si trova nel secondo menù dei comandi, per cui lo scegli premendo **F3**, **N** e poi **S**.

Ti viene chiesto di scrivere il testo da trovare. Quando premi **ENTER** alla fine del testo, Quill trova immediatamente la prima volta in cui si incontra il testo e ti chiede di scrivere il testo sostitutivo (non dimenticare di premere **ENTER** alla fine del testo).

Poi Quill ti chiede se vuoi sostituire il testo trovato. Premi il tasto **S** per sostituire il testo – se premi il tasto **N** il testo non viene sostituito. In entrambi i casi Quill continua a cercare la prossima occorenza e ti offre le stesse scelte di conservare o di sostituire il testo trovato. Questo continua finché non si trovano più nuove occorrenze del testo o finché non premi **ENTER**.

Se, in qualsiasi momento, Quill non trova nuovamente il testo nel documento te lo dice e aspetta finché non premi la barra spaziatrice. Quando premi un tasto Quill ritorna allo schermo principale.

Puoi usare il comando per fare sostituzioni, inserzioni o cancellazioni multiple come viene illustrato nei seguenti tre esempi:

Per *sostituire* la parola "fiume" con "torrente", scrivi "fiume" come testo da trovare e "torrente" come testo sostitutivo.

Per *inserire* "o torrente", scrivi "fiume" come testo da trovare e "fiume o torrente" come testo sostitutivo.

Per *cancellare* "fiume", scrivi "fiume" come testo da trovare e non dare alcun testo sostitutivo (premi semplicemente **R**).

CAPITOLO 6
NOTIZIE AGGIUNTIVE SUL FORMATO

In questo capitolo ci occuperemo delle altre opzioni per modificare l'aspetto del testo. Esso contiene delle informazioni sull'impostazione degli arresti del tabulatore e dei fine pagina. Inoltre c'è una sezione sul comando **Quadro**, che puoi usare per cambiare i valori delle varie opzioni (per esempio la dimensione della pagina) che controllano l'aspetto generale dei documenti.

TABULATORE

Impiego degli arresti del tabulatore.

Un modo molto comune di controllare l'aspetto di un documento è l'impiego degli arresti del tabulatore. Essi consistono in posizioni marcate in corrispondenza di particolari colonne del testo del documento. Quando premi il tasto **TABULATE**, il cursore si muove verso destra, dalla sua posizione attuale al prossimo arresto di tabulatore sulla riga. Se hai superato l'ultima tabulazione, quando premi il tasto **TABULATE** andrai all'inizio della riga successiva.

Per farti sapere dove si trovano gli arresti del tabulatore, nella riga immediatamente sotto la riga del margine vengono tracciate le posizioni del tabulatore e i loro tipi (che vengono descritti qui sotto).

Quill ti permette di usare parecchi tipi di arresti di tabulatore, e di metterli in qualsiasi colonna. Una riga può contenere fino a sedici arresti di tabulatore.

Tipi arresti di tabulatore

Ci sono quattro tipi diversi di arresti di tabulatore.

Il tipo più è comune conosciuto come arresto di tabulatore *Sinistro* ed esso funziona esattamente allo stesso modo delle posizioni del tabulatore su una normale macchina da scrivere. Quando premi il tasto **TABULATE**, il cursore si porterà sulla posizione successiva del tabulatore e il testo che scrivi inizierà alla colonna di tabulazione. Viene chiamato tabulazione Sinistra perchè le righe degli arresti di tabulatore sono allineate sul bordo sinistro.

Un secondo tipo costituito dalla tabulazione *Destra*. Quando ti porti su questo arresto di tabulatore ed inizi a scrivere, il cursore rimarrà nella posizione della tabulazione ed il testo comparirà a sinistra, per cui esso finirà alla posizione della tabulazione. Questo continuerà finché il testo a sinistra della posizione della tabulazione ha riempito lo spazio disponibile o finché premi nuovamente il tasto **TABULATE** per portarti alla posizione successiva del tabulatore. Le righe di testo con questo tipo di arresto di tabulatore sono allineate al bordo destro.

Esiste inoltre una tabulazione *Centrata*. Il testo scritto in tale posizione del tabulatore verrà aggiustato in modo che il suo carattere centrale sia posizionato sulla tabulazione. Anche in questo caso l'allineamento del testo continuerà fino a che lo spazio disponibile (fino al testo esistente o fino al margine sinistro) sia riempito, o finché premi nuovamente il tasto **TABULATE**.

Il quarto tipo di arresto di tabulatore è quello relativo al *Punto* decimale, e viene usato per scrivere valori numerici. Quando scrivi un numero in corrispondenza di tale arresto di tabulatore esso viene posizionato in modo che il punto decimale si trovi nella colonna del tabulatore. Se non scrivi un punto decimale nel testo, esso si comporterà come una tabulazione Destra.

Sinistra	Centro	Destra	Decimale
qualche	qualche	qualche	qualche
parola di	parola di	parola di	parola di
testo	testo	testo	testo
12.345	12.345	12.345	12.345
123.4	123.4	123.4	123.4
1234.56	1234.56	1234.56	1234.56

Figura 6.1 I quattro tipi di arresti di tabulatore

La figura 6.1 illustra l'aspetto di un testo scritto usando uno dei quattro differenti tipi di arresti di tabulatore.

Il comando del tabulatore

All'inizio gli arresti del tabulatore sono posti in corrispondenza di ogni dieci caratteri e sono tutti tabulatori sinistri. Puoi cambiare il numero, la posizione e il tipo di arresto con il comando **Tabula**.

Puoi mettere gli arresti di tabulatore in qualsiasi punto della riga e mescolare i differenti tipi in qualsiasi modo desideri. L'unica limitazione consiste nel fatto che non è possibile avere più di sedici arresti di tabulatore per riga. I nuovi arresti di tabulatore hanno effetto dal paragrafo corrente (quello in cui si trovava il cursore quando hai usato il comando **tabula**) fino alla fine del documento, o al successivo cambiamento di posizione del tabulatore.

Quando scegli il comando **Tabula** (**F3** e **T**), le posizioni vengono tracciate sullo schermo, immediatamente sotto la riga di riferimento.

Ogni arresto di tabulatore marcato da una lettera (**S, C, D** o **P**) per indicarne il tipo. Il cursore è posizionato all'inizio della riga e puoi spostarlo verso destra o sinistra usando i tasti del cursore appropriati.

Puoi cambiare il tipo di arresti di tabulatore che vuoi. Puoi anche premere i tasti di movimento verticale del cursore per portarti in un altro paragrafo e fare altri cambiamenti agli arresti di tabulatore. Quando hai fatto tutti i cambiamenti che vuoi, premi **ENTER** per uscire dal comando e tornare allo schermo principale.

Inserzione di un tabulatore

Per inserire un fermo di tabulatore, scegli il tipo che vuoi, usa i tasti di movimento verso destra e sinistra del cursore per portarlo alla posizione richiesta per il tabulatore e premi **T**.

Quando hai selezionato il comando **Tabula**, nella zona dei comandi vengono riportati i vari tipi di tabulazione.

La zona dei comandi contiene le parole S(inistra), C(entro), D(estra) e P(unto), e la parola S(inistra) evidenziata. Questo indica che il prossimo arresto di tabulatore che verrà inserito sarà un tabulatore sinistro.

Puoi cambiare il tipo di arresto di tabulatore da inserire premendo lo spazio (ogni volta che lo premi l'evidenziamento si sposta da un tipo al successivo) o premendo il tasto corrispondente alla sua prima lettera. Per esempio, se vuoi cambiare ad una tabulazione Destra, puoi continuare a premere lo spazio fino a che viene evidenziata la parola D(estra) o semplicemente premere il tasto **D**.

Cancellazione di una tabulazione

Togli un arresto di tabulatore spostando il cursore finché non sia sopra il marcatore di tabulazione che vuoi cancellare e premi il tasto **X**.

QUADRO

Usa il comando **Quadro** per cambiare le caratteristiche dello schermo principale, quali:

Il numero di caratteri per riga
La spaziatura tra le righe
Il numero di righe per pagina del documento

AIUTO F1	QUADRO determina il FORMATO di stampa. Premi la prima lettera dell'opzione	COMANDI F3
MESSAGGI F2	Alla fine premi ↵	USCITA ESC

Bordo inferiore 3

Larghezza schermo 80,64,40 (8,6,4) 8

Interlinea (0,1,2) 0

Righe stampate per pagina 66

Numero prima pagina 1

Colore testo - Verde o Bianco ..... VERD

Margine superiore 6

Figura 6.2 Il comando Quadro

Il comando viene illustrato nella figura 6.2 ed una descrizione completa di ciascuna opzione viene riportata nel capitolo 8.

Per scegliere il comando **Quadro** premi **F3** e poi **Q**. Immediatamente Quill riporta l'elenco delle opzioni. Se, per esempio, vuoi scegliere una visualizzazione a 40 caratteri, premi il tasto L per l'opzione "Larghezza schermo". Questa opzione verrà evidenziata e Quill attende che tu prema 4, 6 o 8 per selezionare una visualizzazione di 40, 64 od 80 caratteri. Non potrai scegliere alcuna altra opzione finché non avrai scelto tra una di queste tre.

Poi hai la possibilità di cambiare una o tutte le caratteristiche elencate sullo schermo. Dopo aver fatto tutti i cambiamenti che vuoi, devi uscire dal comando premendo **ENTER**.

DOCUMENTI LARGHI

Se sposti il margine destro in modo che il numero di caratterl della riga sia superiore alla larghezza dello schermo, Quill non può mostrare tutta la larghezza del documento sullo schermo. In questa situazione la zona di visualizzazione si comporta come una finestra, attraverso cui vedi solo una parte di tutta la larghezza del documento. Man mano che sposti il cursore lungo la riga, la finestra scorrerà lungo il documento, per cui mostrerà sempre la regione contenente il cursore.

Righe per pagina

Una delle opzioni del comando **Quadro** ti permette di impostare le dimensioni della pagina, cioè il massimo numero di righe di testo che possono stare su una pagina del documento. In aggiunta al testo, questo numero di righe comprende il margine inferiore e superiore, l'eventuale intestazione e piè di pagina e le interlinee tra queste ed il testo.

Supponiamo, per esempio, che tu abbia un margine superiore di 3 righe, una intestazione separata da 2 righe vuote dal testo, un piè di pagina separato dal testo da 4 righe vuote e un margine inferiore di 5 righe. Questo dà un totale di 3+1+2+4+6=16 righe. Con una dimensione di pagina di 66 righe ci saranno 66-16=50 righe di testo su ciascuna pagina. Se usi il comando **Quadro** per impostare l'interlinea 1 (spaziatura doppia), avresti solo 25 righe di testo per pagina.

Fine pagina

Un fine pagina segna il punto del tuo documento in cui inizierà una nuova pagina, a seconda della lunghezza di pagina impostato nel comando **Quadro**. Viene raffigurato come una linea orizzontale che attraversa lo schermo e contiene il numero di pagina. Nel calcolare la lunghezza della pagina Quill tiene conto del margine superiore ed inferiore, dell'intestcazione e del piè di pagina. Nell'esempio di cui sopra, con l'interlinea zero Quill inserisce un fine pagina dopo ciascun blocco di 50 righe di testo.

Se imposti una dimensione di pagina che non lascia spazio per cinque o più righe di testo per pagina, Quill disattiverà la funzione di fine pagina. In questo caso non vengono indicati i fine pagina e Quill tratterà tutto il documento come una pagina singola. Per disattivare l'impaginazione automatica devi scrivere 0 alla voce del numero di righe per pagina.

Fine pagina obbligati

Per forzare un fine pagina ad una riga particolare puoi usare il comando **Pagina**. Questo è molto utile per garantire che una sezione di testo, per esempio un elenco o una tabella, cominci all'inizio di una nuova pagina e non venga spezzato in due parti su pagine differenti.

Il comando **Pagina** ti permette di impostare un fine pagina obbligato a piacere (esso si trova nel secondo menù di comandi – premi **F3**, il tasto **N** e poi il tasto **P**). Poi devi posizionare il cursore sulla riga dove vuoi finire la pagina (in un punto qualsiasi) e premere il tasto **P**. Quill inserirà un fine pagina dopo la riga scelta.

Puoi impostare parecchi fine pagina obbligati, in differenti posizioni, ma non puoi impostare più di un fine pagina obbligato sulla stessa riga di un documento. Quando hai finito, premi **ENTER** per uscire dal comando.

Puoi togliere un fine pagina forzato dal tuo documento in qualsiasi momento – per farlo non hai bisogno di usare il comando **Pagina**. Il fine pagina viene tolto mettendo il cursore sulla linea di fine pagina (usando i tasti di movimento verticale del cursore). Poi premi il tasto **CTRL** e, tenendolo schiacciato, premi il tasto di movimento verso sinistra del cursore.

CAPITOLO 7
OPERAZIONI SUI FILE

Una volta prodotto un documento vorrai probabilmente memorizzarne una copia su una cartuccia di Microdrive. Forse più tardi vorrai fare dei cambiamenti e conservare una copia della nuova versione. Se disponi di una stampante vorrai senz'altro produrre delle copie su carta.

I documenti vengono memorizzati in una cartuccia di Microdrive sotto forma di un *file* – una serie di informazioni dotata di nome. Questo capitolo descrive i comandi forniti per memorizzare, richiamare e stampare i file.

MEMORIZZA

Impieghi questo comando per memorizzare in una cartuccia di Microdrive una copia del testo di un documento di Quill. Se non memorizzi un documento dopo averlo scritto ne perderai il contenuto quando esci da Quill.

Quando usi il comando **Memorizza** (**F3** e **M**), ti viene chiesto di scrivere il nome del documento. Perciò, il modo più semplice di usare il comando quello di battere una sequenza simile alla seguente.

F3 `M mialett` ENTER

Questo memorizza il documento con il nome "mialett_doc" sulla cartuccia nel Microdrive 2.

Se il nome uguale a quello di un documento già memorizzato sul Microdrive, Quill ti ricorderà che quel documento esiste già, e ti chiede se vuoi scrivere sopra quello nuovo. Premi **S** (sì) per sostituire il documento o **ESC** per memorizzare il documento con un nome differente.

Una volta che il documento e' stato memorizzato, Quill ti chiede se vuoi continuare a scrivere nel documento. Premi **ENTER** per continuare oppure premi la sbarra spaziatrice per cambiare documento.

Quando dai un nome o se richiami un documento memorizzato in precedenza, Quill mostra il nome del documento nella zona di stato. Se più tardi desideri rimemorizzare il documento, Quill ti suggerisce di usare il nome corrente del documento. Se scrivi un nome di tua scelta, esso sostituirà il nome suggerito. In alternativa puoi accettare il suggerimento di Quill semplicemente premendo **ENTER**. In tale caso puoi battere:

F3 `M` ENTER

A questo punto la nuova versione verrà memorizzata sulla cartuccia nel Microdrive, sostituendo quella vecchia.

RICHIAMA

Il comando **Richiama** dev'essere usato quando vuoi copiare nella memoria dell'elaboratore un documento che si trova sulla cartuccia nel Microdrive, in modo che, per esempio, possa essere modificato.

Per prima cosa ti viene chiesto di scrivere il nome del documento che vuoi richiamare. Se l'hai dimenticato, puoi battere il punto di domanda, seguito da **ENTER**. A questo punto Quill ti presenta un elenco di tutti i documenti del Microdrive 2 e ti chiede nuovamente di scrivere il nome.

Se il nome che scrivi non corrisponde al nome di un documento esistente, QUILL ti dirà che il documento non esiste e ti dà un'altra possibilità di scrivere il nome.

FILE E UNIONE

Il comando file comprende quattro opzioni:

BACKUP	– per copiare un documento del Microdrive o un altro file del Microdrive
CANCELLA	– per cancellare un documento del Microdrive o un altro file del Microdrive
FORMATTA	– per formattare una cartuccia nel Microdrive
IMPORTA	– per inserire dalla posizione del cursore del documento corrente un file di Microdrive, esportato da Abacus, Archive o Easel.

Il comando **Unione** ti permette di prendere un documento presente in una cartuccia nel Microdrive e di inserirlo nel documento corrente dalla posizione del cursore.

Con questi comandi spesso sarà necessario usare una seconda cartuccia di dati. Per esempio, di solito desidererai fare una copia di backup di un documento su una cartuccia differente, ed un file di importazione non si troverà solitamente sulla stessa cartuccia del documento Quill.

Puoi togliere la cartuccia Quill nel Microdrive 1 e sostituirla con un'altra cartuccia. Ricordati di rimettere la cartuccia Quill prima di stampare un documento o chiedere Aiuto. Se usi dei Microdrive aggiuntivi normalmente non sarà necessario togliere la cartuccia Quill dal Microdrive 1.

STAMPA

Questo comando viene usato per produrre una copia stampata di tutto o parte di un documento QUILL. Naturalmente, necessario avere a disposizione una stampante e che essa sia collegata correttamente all'elaboratore, altrimenti non succederà molto!

Quill suggerisce che tu stampi il documento su cui stai attualmente lavorando, ed attende che tu prema un tasto. Premi **ENTER** per accettare il suggerimento, oppure scrivi il nome del documento da stampare (che deve essere un documento presente nella cartuccia nel Microdrive 2).

QUILL ti chiederà se vuoi che venga stampato tutto il documento. Premi **ENTER** per accettare il suggerimento. Altrimenti scrivi il numero della prima pagina del documento da stampare e successivamente il numero dell'ultima pagina da stampare, premendo **ENTER** dopo aver scritto ogni numero. Puoi stampare soltanto pagine complete del documento.

Il comando stampa ha l'opzione di inviare il testo ad un file su Microdrive invece che alla stampante. Premi **ENTER** per usare la stampante, o scrivi un nuovo nome di file se vuoi inviare il testo ad un file. Il file prodotto in tale modo conterrà tutti i caratteri e codici di comando che altrimenti sarebbero stati inviati alla stampante.

L'impiego più semplice quello di stampare tutto il documento corrente. I tasti da premere in questo caso sono:

F3 s ENTER ENTER ENTER

Per stampare le pagine da 2 a 4 di un documento chiamato "mialett__doc" su un nuovo file chiamato "mialett__ele" (entrambi nel drive n. 2) devi scrivere:

F3 `S mialett` ENTER `2` ENTER `4` ENTER `mialett` ENTER

Prima di iniziare la stampa, Quill leggerà il programma di comando stampante dalla cartuccia Quill nel Microdrive 1. Esso dirà che funzioni sono disponibili su una particolare stampante e come possono essere usate. Quill funzionerà con la maggior parte delle stampanti ed i particolari di come fare le modifiche per uno specifico tipo di stampante potranno essere trovati nella sezione delle *Informazioni*, dove viene descritto il programma di comando della stampante.

Puoi anche desiderare cambiare delle caratteristiche, quali lo spazio tra le righe ed il numero di righe per pagina del documento stampato. Esse sono tutte riportate nel comando **Quadro**, descritto nel capitolo 6.

CAPITOLO 8
CONSULTAZIONE QUILL

I TASTI DELLE FUNZIONI

In aggiunta all'impiego normale dei tasti di funzione **F1**, **F2** e **F3**, viene usato il tasto di funzione **F4** per i sequenti scopi:

F4	cambia carattere
SHIFT e **F4**	commuta tra inserimento e riscrittura

Quill non usa il tasto di funzione **F5**.

Scegli un comando premendo **F3**. Questo fa comparire il menù dei comandi di Quill. Puoi ancora spostare il cursore, ma non puoi inserire o cancellare testo.

Il contenuto della zona dei comandi cambia e compare un elenco dei comandi disponibili. Scegli un comando battendo la sua iniziale. Hai a disposizione un secondo elenco di comandi (COMANDI II), che puoi vedere usando il comando Nuovo elenco.

Poiché nei due elenchi ci sono comandi che iniziano con la stessa lettera, devi sempre accertarti che il comando desiderato si trovi nella zona dei comandi prima di sceglierlo.

In generale, puoi abbandonare un comando parzialmente completato premendo **ESC**.

Alla fine della maggior parte dei comandi di Quill ritorna allo schermo principale. L'eccezione costituita dai comandi che hanno un loro menù interno (per es. **File**). In questi casi rimani nel menù interno e devi premere **ESC** per tornare allo schermo principale.

In tutti i comandi che richiedono una immissione di testo (per es. **memorizza, richiama, file, sostituzione**), puoi modificare il testo con l'editore di riga, descritto nella presentazione del programma QL.

I COMANDI

Sono disponibili i seguenti comandi:

COPIA

Usa questo comando per spostare o copiare del testo da una posizione del documento ad un'altra.

Essi sono elencati in ordine alfabetico. Se sono parte del secondo elenco dei comandi questo viene indicato da un simbolo II dopo il nome dei comandi.

Ti viene dapprima chiesto di portare il cursore all'inizio del testo da copiare e poi di premere **ENTER**. Successivamente porta il cursore alla fine del testo che vuoi copiare. Nota che puoi muovere all'indietro il cursore, con i tasti con freccia verso sinistra o verso l'alto, ma non puoi superare il punto di inizio. Il testo interessato viene evidenziato. Premi **ENTER** quando hai finito. Premi nuovamente **ENTER** per cancellare il testo marcato originariamente, o premi il tasto **C** per conservarlo. Porta il cursore nella posizione dove vuoi che compaia il testo marcato e premi il tasto **C** per inserire il testo nella nuova posizione.

Puoi fare copie addizionali del testo in un altro punto del documento. Metti il cursore dove vuoi che compaia un'altra copia e premi il tasto **C**. Puoi fare quante copie vuoi. Quando hai finito premi **ENTER** per uscire dal comando.

QUADRO

Questo comando ti permette di impostare o cambiare un certo numero di caratteristiche che controllano l'aspetto generale del documento. All'interno del comando ti viene chiesto di scegliere, premendo il tasto appropriato, tra le seguenti opzioni:

Bordo inferiore

scrivi il numero di righe da lasciare vuote alla base di ogni pagina stampata del documento. Premi **ENTER** dopo aver scritto il numero. L'impostazione iniziale per un margine inferiore di 3 righe.

Larghezza schermo

scrivi 4, 6 o 8 per scegliere una larghezza di schermo di 40, 64 o 80 caratteri per riga. Quill non accetta nessun altro carattere. L'impostazione iniziale di 80 o 64 caratteri, a seconda se usi un monitor o un televisore.

Interlinea
scrivi 0, 1 o 2 per scegliere quante linee vuote saranno lasciate tra ogni riga di documento. Quill non accetta nessun altro carattere. L'impostazione iniziale 0.

Righe stampate per pagina
scrivi il numero totale di righe da stampare in ogni pagina di documento e premi **ENTER**. Questo numero comprende le linee vuote del margine inferiore e superiore. Se scrivi uno zero il documento non verrà diviso in pagine. L'impostazione iniziale a 66. (Normalmente possibile stampare 66 righe su un normale foglio A4).

Numero prima pagina
scrivi un numero, seguito da **ENTER**. Questo numero viene usato per numerare la prima pagina del documento. Le pagine successive vengono numerate progressivamente partendo da questo valore. Il valore iniziale 1, ma puoi desiderare cambiarlo se il tuo documento la continuazione di un altro.

Colore testo
per cambiare i colori usati per il testo normale o grassetto. ogni volta che scegli questa opzione i colori del testo cambiano tra verde e bianco. L'impostazione iniziale prevede il testo normale in verde ed il testo in grassetto bianco.

Margine superiore
scrivi il numero di linee da lasciare vuote all'inizio di ogni pagina del documento. Dopo aver scritto il numero premi **ENTER**. L'impostazione iniziale di 6 linee.

Alla fine di ogni opzione puoi scegliere un'altra opzione, o premere **ENTER** per uscire dal comando.

ELIMINA

Questo comando ti permette di eliminare righe di testo dal documento. Per prima cosa ti viene chiesto di mettere il cursore (usando i tasti con freccia) sul primo carattere che vuoi eliminare, premere **ENTER**, e far scorrere il cursore lungo il testo che vuoi eliminare. Il testo viene evidenziato man mano che il cursore ci passa sopra. Una volta marcato il testo devi premere nuovamente **ENTER**. Il testo marcato viene cancellato immediatamente.

FILE (II)

Questo comando mette a disposizione quattro opzioni.

Cancella
per cancellare dalla cartuccia nel Microdrive il documento o file nominato. Ti viene chiesto di scrivere il nome del file che vuoi cancellare. Il file viene cancellato quando premi **ENTER**.

Formatta
per formattare una cartuccia nel Microdrive 2. Poiché questo cancella tutte le informazioni sulla cartuccia, devi confermare la scelta.

Attenzione: tutte le informazioni sulla cartuccia vengono cancellate quando la formatti.

Backup
per fare una seconda copia di riserva di un documento su cartuccia di Microdrive. Ti viene chiesto di scrivere il nome del documento ed il nome che vuoi dare alla nuova copia. Normalmente farai la copia su una cartuccia differente, perciò puoi usare lo stesso nome per la copia. Importa per inserire dalla posizione del cursore del tuo documento un file preso dalla cartuccia di Microdrive. Il file deve essere stato esportato da QL Abacus o QL Archive, o essere un file di testo prodotto, per esempio, da SuperBASIC. Vedi la sezione sulle *informazioni*.

FONDOPAGINA

Questo comando ti permette di specificare una riga di testo da usare nell'ultima riga di ciascuna pagina. Il fondopagina non compare nel documento sullo schermo, ma solo sulla pagina stampata.

Per prima cosa ti viene chiesto di scegliere la posizione del fondopagina tra quattro opzioni:

Nessuno – nessun testo di fondopagina
Sinistra – al margine sinistro
Centro – centrato nella pagina (l'impostazione iniziale)
Destra – al margine destro

Premi lo spazio fino a che l'opzione richiesta sia evidenziata e poi premi **ENTER**. Ti viene poi chiesto di scrivere il testo a piè di pagina, e per finire premi ancora **ENTER**.

Se avevi precedentemente specificato un testo a pi di pagina, esso viene visualizzato nella zona di stato. Hai l'opzione di variarlo con l'editore di riga, invece di scrivere tutto il testo revisionato.

Il numero di pagina può essere inserito nel testo in qualsiasi punto desiderato. La posizione e il tipo del numero di pagina marcato da un codice a tre caratteri:

Caratteri	Tipo di numerazione
`nnn` o **`NNN`**	Numeri arabi per es. 1, 2, 3, 4
`rrr` o **`RRR`**	Numeri romani per es. I, II, III, IV
`aaa` o **`AAA`**	Alfabeto per es. a, b, c, d

Ti viene poi chiesto di scrivere un numero, da 0 a 9, per indicare il numero di spazi da lasciare tra il fondo del testo ed il pi di pagina.

ANDARE

Puoi usare questo comando per portare il cursore all'inizio, alla fine o ad una pagina specificata del documento. Ti vengono offerte tre opzioni:

Inizio
: per portare il cursore all'inizio del documento.

Fine
: per portare il cursore alla fine del documento. un numero di pagina se scrivi un numero, seguito da **ENTER** mandi il cursore all'inizio di quella pagina del documento. Se non ci sono fine pagina questa opzione manda il cursore alla fine del documento.

INTESTAZIONE

Questo comando ti permette di specificare una riga di testo da usare come prima riga di ciascuna pagina. Nota che l'intestazione non compare nel documento sullo schermo. Quill non fornisce una intestazione automatica al documento.

Per prima cosa ti viene chiesto di scegliere la posizione dell'intestazione tra le quattro opzioni:

Nessuno – nessuna intestazione (l'impostazione iniziale)
Sinistra – al margine sinistro
Centro – al centro della pagina
Destra – al margine destro

Premi lo spazio finché l'opzione richiesta viene evidenziata e poi premi **ENTER**. A questo punto ti viene chiesto di battere il testo dell'intestazione, e per finire premi **ENTER**.

Se hai aggiunto una intestazione in una fase precedente, il testo esistente compare nella zona di stato. A questo punto hai l'opzione di alterarlo con l'editore di riga, invece di scrivere tutto il testo.

Puoi inserire il numero di pagina in qualsiasi punto del testo. La posizione ed il tipo del nomero sono contraddistinti da un codice a tre caratteri:

Caratteri	Tipo di numerazione
`nnn` o **`NNN`**	Numeri arabi per es. 1, 2, 3, 4
`rrr` o **`RRR`**	Numeri romani per es. I, II, III, IV
`aaa` o **`AAA`**	Alfabeto per es. a, b, c, d

TRATTINO (II)

Questo comando ti permette di specificare un punto di una parola in cui essa può essere spezzata, con un trattino inserito automaticamente, se esso si estende oltre la fine della riga. Le parole non marcate in questo modo, se necessario, saranno completamente trasferite alla riga successiva.

Il trattino particolarmente utile se stai usando la giustificazione destra, per evitare che tra le parole si creino degli spazi eccessivi.

Porta il cursore sul primo carattere dopo la posizione in cui vuoi che la parola venga divisa e premi il tasto **T**. Puoi ripetere questo procedimento quanto vuoi. Premi **ENTER** per uscire dal comando.

Se la parola non si trova alla fine di una riga sembrerà che il comando non abbia alcun effetto.

GIUSTIFICA

Usa questo comando per scegliere il tipo di giustificazione desiderato. Esso ha effetto dall'inizio del paragrafo contenente il cursore, e rimane in effetto fino alla fine del documento, o fino al prossimo cambiamento di giustificazione.

Ti vengono offerte le seguenti opzioni, selezionate premendo il tasto corrispondente all'iniziale dell'opzione:

Sinistra	il testo viene allineato al margine sinistro, ma il margine destro irregolare.
Centro	il testo di ogni riga centrato rispetto ai margini.
Destra	tra le parole di ciascuna riga vengono inseriti spazi aggiuntivi in tutte le righe, in modo che il testo sia allineato sia al margine sinistro che destro.

Puoi fare cambiamenti di giustificazione a più di un paragrafo. Premi i tasti di movimento verticale per muovere il cursore verso l'alto o verso il basso di un paragrafo e cambia la giustificazione come descritto più sopra. Premi **ENTER** per uscire dal comando.

RICHIAMA

Questo comando ti permette di caricare in memoria un documento da una cartuccia di Microdrive, pronto per la stampa o l'edizione.

Scrivi il nome del documento (il nome che gli hai dato quando l'hai memorizzato). Se premi solo "?" più **ENTER**, Quill ti mostrerà un elenco dei nomi di tutti i documenti memorizzati sul Microdrive 2. Se vuoi l'elenco dei file di un Microdrive diverso cambia il nome del Microdrive. Quando Quill ha mostrato l'elenco, ti chiede nuovamente di scrivere il nome del documento.

BORDI

Usa questo comando per impostare o cambiare le posizioni del margine sinistro, rientrato e destro del tuo documento. Tutti i cambiamenti dei margini hanno effetto sul testo man mano che li fai.

La zona dei comandi riporta le parole Sinistra, Rientro e Destra, e appena si entra nel comando illuminata la parola Sinistra. Questo significa che puoi spostare il margine sinistro usando i tasti di movimento verso destra e sinistra del cursore.

Puoi scegliere uno qualsiasi dei tre margini premendo lo spazio fino a che nella zona dei comandi sia evidenziato il nome del margine corretto. Il margine prescelto viene spostato premendo il tasto di movimento verso destra o sinistra del cursore.

Il cambiamento dei margini ha effetto dal paragrafo che contiene il cursore. Esso rimane valido fino alla fine del documento, o fino al prossimo cambiamento di posizione di quel margine.

Puoi fare cambiamenti alle posizioni dei margini in più di un paragrafo. Premi i tasti del movimento verticale del cursore per spostarlo verlo l'alto o il basso di un paragrafo e cambiare i margini come descritto più sopra. Premi **ENTER** per uscire dal comando.

UNIONE (II)

Il comando **Unione** preleva dalla cartuccia di Microdrive una copia del documento di Quill nominato e lo inserisce, dalla posizione corrente del cursore, nel documento attualmente in memoria.

Questo comando ti dà l'opzione di sostituire la cartuccia di con una cartuccia di dati. Devi sostituire la cartuccia di Quill nel Microdrive 1 alla fine del comando.

Posiziona il cursore nel punto in cui vuoi inserire il documento prima di selezionare il comando. Quill ti chiede di scrivere il nome del file che vuoi inserire. Se inserisci il documento nel mezzo di un paragrafo, Quill lo dividerà in due paragrafi dalla posizione del cursore e inserirà il documento tra di essi.

NUOVO ELENCO

Questo comando ti permette di visualizzare una seconda serie di comandi nella zona dei comandi. Nella zona dei comandi comparirà la prima o la seconda serie ogni volta che premi Nuovo.

Poiché parecchi comandi iniziano con la stessa lettera, accertati che il comando che vuoi sia uno di quelli visualizzati prima di sceglierlo.

PAGINA (II)

Puoi usare questo comando per marcare un punto del documento dove vuoi che cominci una nuova pagina.

Porta il cursore dove vuoi che cominci la nuova pagina e premi il tasto **P**.

Puoi aggiungere questi fine pagina in parecchi punti del documento. Porta il cursore nel punto in cui vuoi che cominci un'altra pagina e premi il tasto **P**. Premi **ENTER** per uscire dal comando.

Non usare il comando **Pagina** per cancellare un fine pagina obbligato. Puoi annullare un fine pagina portando il cursore in qualsiasi punto della riga di fine pagina e premendo contemporaneamente **CTRL** e la freccia di movimento verso sinistra del cursore.

STAMPA

Questo comando stampa tutto o in parte il documento attualmente nella memoria dell'elaboratore, o un qualsiasi altro documento nella cartuccia nel Microdrive 2.

Premi **ENTER** per stampare il documento corrente, o scrivi il nome di file del documento da stampare, seguito da **ENTER**.

Quill suggerisce poi di stampare l'intero documento. Se rispondi premendo **ENTER** verrà stampato tutto il documento. Se vuoi che vengano stampate alcune delle pagine, scrivi il numero della prima pagina che vuoi stampare, seguito da **ENTER**. Poi scrivi il numero dell'ultima pagina che deve essere stampata, e per terminare premi ancora **ENTER**.

Per finire, premi **ENTER** per inviare il testo alla stampante, o scrivi il nome di un nuovo file, seguito da **ENTER** per inviare l'uscita ad un file su Microdrive.

Prima di stampare, Quill leggerà un file "printerdot" che immette le informazioni del programma di comando della stampante.

LASCIA

Questo comando ti permette di uscire da Quill e ritornare a SuperBASIC. Hai tre opzioni:

ENTER	per memorizzare il documento corrente prima di ritornare a SuperBASIC. Ti viene offerta anche l'opzione di scrivere il nome del documento memorizzato. Se premi soltanto **ENTER**, il documento verrà memorizzato col suo vecchio nome, e sostituirà la versione originaria del documento nella cartuccia di Microdrive.
A	per abbandonare il documento corrente e tornare a SuperBASIC senza memorizzarlo.
ESC	per annullare il comando e tornare al tuo documento.

SOSTITUZIONE (II)

Puoi usare questo comando per sostituire un testo con un altro tutte o alcune volte in cui si presenta.

Per prima cosa scrivi il testo da sostituire, seguito da **ENTER**. Poi scrivi il testo sostitutivo, seguito nuovamente da **ENTER**.

Quill esamina il documento dall'inizio fino a che incontra la prima volta il vecchio testo. Poi ti offre l'opzione di sostituire il vecchio testo con il nuovo. Premi il tasto **S** per sostituire il testo, o **P** per passare al prossimo.

Quill si metterà a cercare la prossima volta in cui si incontra il testo e ti offrirà ancora la possibilità di fare la sostituzione. Questa procedura continua fino a che raggiungi la fine del documento, o finché finisci il comando premendo **ENTER**.

MEMORIZZA

Questo comando viene usato per memorizzare su cartuccia di Microdrive una copia del documento.

Scrivi il nome del documento, in modo da poterlo identificare. Il documento viene quindi memorizzato sotto quel nome. Se, invece di scrivere un nome, premi soltanto **ENTER**, il documento verrà memorizzato con il suo vecchio nome, sostituendo la versione originale.

Poi Quill ti chiede se vuoi continuare l'edizione del documento appena memorizzato. Se premi **ENTER**, il testo del documento rimane nella memoria dell'elaboratore e puoi continuare a lavorarci sopra.

Oppure premi la barra spaziatrice se vuoi continuare con un altro documento.

RICERCA (II)

Questo comando ricerca una particolare parola o frase nel documento.

Per prima cosa scrivi il testo che vuoi trovare. Quando premi **ENTER** Quill comincia all'inizio del documento e cerca la prima volta in cui scritto il testo.

SEARCH (II)

Questo comando ricerca nel testo una parola o una frase determinata.

A questo punto puoi premere il tasto **C** per continuare la ricerca, per trovare la prossima volta in cui si incontra il testo. Premi **ENTER** per terminare il comando quando hai trovato quello che vuoi.

TABULATORE

Il comando **Tabula** ti permette di specificare le posizioni e i tipi di arresti del tabulatore su una riga di testo. Il tasto TABULATE ti porterà direttamente al prossimo arresto di tabulazione lungo il righello da te impostato. I cambiamenti degli arresti del tabulatore avranno effetto dall'inizio del paragrafo corrente (quello che contiene il cursore). Essi rimarranno validi fino alla fine del documento, o fino alla prossima volta che cambi gli arresti del tabulatore.

Sono utilizzabili quattro tipi di arresti di tabulatore:

Sinistra — l'arresto del tabulatore si comporta come un margine sinistro; il testo viene posizionato alla destra dell'arresto del tabulatore.

Centro — il testo viene centrato attorno all'arresto del tabulatore.

Destra — l'arresto del tabulatore si comporta come un margine destro; il testo viene posizionato alla sinistra dell'arresto del tabulatore.

Punto — viene usato per allineare numeri decimali. Ogni numero verrà posizionato in modo che il suo punto decimale si trovi all'arresto del tabulatore. Esso si comporta come un tabulatore destro se non trova un punto decimale.

Le posizioni del tabulatore vengono tracciate sullo schermo, sotto il righello, usando i seguenti simboli.

S – sinistra
C – centro
D – destra
P – punto

Il cursore viene posizionato all'inizio di quella riga. Puoi spostare il cursore lungo la riga usando i tasti di movimento verso destra e sinistra del cursore.

Puoi togliere un indicatore di tabulazione spostando il cursore con i tasti di movimento verso destra e sinistra del cursore finché si trovi sopra l'indicatore di tabulazione nella riga sotto il righello e poi premendo il tasto **X**.

Per inserire un indicatore di tabulazione devi per prima decidere che tipo vuoi premendo lo spazio finché viene evidenziato il tipo corretto nella zona dei comandi; oppure premendo i tasti **S, C, D,** o **P**. Successivamente porta il cursore nel punto appropriato e premi il tasto **T**.

Puoi mescolare l'inserzione e la cancellazione di indicatori di tabulazione in qualsiasi combinazione. Puoi anche premere i tasti di movimento verticale del cursore per portarti su un altro paragrafo e fare altri cambiamenti agli arresti del tabulatore. Quando avrai fatto tutti i cambiamenti che vuoi, premi **ENTER** per uscire dal comando e ritornare allo schermo principale.

ZERO (II)

Questo comando distrugge tutto il documento corrente, senza memorizzarlo sulla cartuccia nel Microdrive. Ti permette di scartare il documento corrente e ricominciare.

CARATTERI

Puoi cambiare i caratteri del testo del documento premendo **F4** e poi la prima lettera di una delle quattro opzioni elencate più sotto. Il carattere selezionato ha effetto su tutto il testo che viene scritto successivamente.

In alternativa puoi premere **F4** e poi il tasto **E** per cambiare il testo esistente e dargli la nuova forma.

Ti vengono offerte le seguenti opzioni di caratteri:

Grassetto
il testo viene convertito in un carattere grassetto, o pesante.

Sottolineato
il testo viene sottolineato.

Alto (apice)
il testo viene scritto nella metà superiore della riga.

Basso (pedice)
il testo viene scritto nella metà inferiore della riga.

Puoi scegliere qualsiasi combinazione di queste opzionmi, ma naturalmente non puoi avere assieme i caratteri Alto e Basso. Se scegli uno di essi, l'altro verrà spento automaticamente.

Se vuoi scegliere una combinazione di caratteri devi sceglierli l'uno dopo l'altro, premendo **F4** e la lettera appropriata.

Se premi il tasto E per scegliere l'opzione evidenziamento, Quill ti permette di scegliere uno o più tipi di carattere. Poi sposta il cursore con i tasti di movimento verso il basso e verso destra per convertire il testo al nuovo carattere. Se usi i tasti di movimento verso l'alto e verso sinistra, il cursore si sposta all'indietro, ripristinando il testo al carattere normale.

Puoi disattivare un qualsiasi carattere usando lo stesso sistema impiegato per attivarlo – cio premendo **F4** e poi i tasti appropriati (**A, B, G,** o **S**).

MODI INSERIMENTO E RISCRITTURA

All'inizio Quill si trova nel modo inserimento e il testo che scrivi verrà aggiunto nel tuo documento dalla posizione del cursore. Il testo circostante verrà aperto per lasciare posto al testo inserito.

Se tieni schiacciato **SHIFT** e premi **F4**, Quill passerà al modo riscrittura. In questo modo, il testo che scrivi sostituirà, carattere per carattere, il vecchio testo dalla posizione del cursore in avanti.

Puoi ritornare al modo inserimento con lo stesso sistema, cio tenendo schiacciato **SHIFT** e premendo **F4**.

I PARAMETRI DI AVVIAMENTO

Al primo caricamento QUILL si trova nello stato descritto dalla tabella sottostante. Puoi cambiare ciascuna delle caratteristiche con il metodo indicato nella colonna di destra.

Caratteristica	Inizialmente	Cambia con
Modo:	inserimento	**SHIFT + F4**
Larghezza schermo	80 (mon) 64 (tv)	Quadro
Margine sinistro:	10 0	Bordi
Margine rientrato:	15 5	Bordi
Margine destro:	70 64	Bordi (max 160)
Margine superiore:	6	Quadro
Margine inferiore:	3	Quadro
Giustificazione:	Destra	Giustifica
Arresti tabulatore:	Sin, col. 10,20,..,80	Tabula
Altezza pagina:	66	Quadro
Interlinea:	0	Quadro
Intestazione:	nessuna	Intesta
Fondopagina	centrato "pagina nnn"	Fondopg
Numero prima pagina	1	Quadro
Colore testo:		
Normale	verde	Quadro
Grassetto	bianco	Quadro
Caratteri:		
Grassetto	spento	**F4**
Sottolineato	spento	**F4**
Alto (apice)	spento	**F4**
Basso (pedice)	spento	**F4**

CAPITOLO 1

CENNI SU QL ABACUS

QL Abacus è una tabella elettronica, che può essere usata per la progettazione, la tabulazione dei dati, i bilanci preventivi, la memorizzazione delle informazioni oppure per la presentazione delle informazioni. Le informazioni vengono rappresentate su di un *reticolo* formato da 255 righe per 64 colonne. La zona dei dati che vedi sullo schermo è una *finestra*, attraverso la quale puoi vedere parte del reticolo. La finestra può essere fatta scivolare lungo il reticolo. Le intersezioni delle righe con le colonne formano più di 16.000 *celle* o caselle nel reticolo. Il testo può essere immesso in una cella o in celle, oppure le celle possono essere utilizzate per memorizzare numeri e dati.

La vera potenza di Abacus, comunque, deriva dall'uso delle regole e delle *formule*, che permettono di collegare blocchi, righe o colonne di celle diverse, o anche celle individuali del reticolo. Il che significa che le informazioni, inserite in una determinata zona, possono essere elaborate immediatamente e rapppresentate in forma diversa in un'altra zona.

Per esempio, puoi utilizzare dodici colonne per rappresentare i mesi dell'anno e puoi immettere i dati delle vendite sulla riga delle "vendite". Le due righe seguenti possono contenere formule per calcolare il costo delle vendite (come percentuale delle vendite più un costo fisso) e il profitto. Il profitto mensile viene così elaborato automaticamente ogni volta che scrivi una cifra di vendita. I totali annui possono anche essere ottenuti utilizzando un'altra formula, in modo che le vendite di, diciamo, marzo daranno immediatamente un profilo di profitto diverso con un totale annuo variato. Tutte le cifre vengono elaborate automaticamente da Abacus.

Puoi anche utilizzare i dati di Abacus in forma grafica o in tabelle facendo uso dei comandi di *esportazione* del package QL della Psion.

Sotto molti punti di vista Abacus è un linguaggio visivo per la programmazione. Puoi manipolare il testo, i dati, o le formule, puoi usare immissione ed emissione degli estratti conti e variabili di testo.

Se, in qualsiasi momento, non sei sicuro di ciò che devi fare ricordati che puoi chiedere Aiuto premendo **F1**. Ricordati pure che puoi cancellare qualsiasi parte di un'operazione non completata (per esempio se batti un numero, o usi un comando) premendo **Esc**.

CAPITOLO 2
INIZIO

RICHIAMO QL ABACUS

Richiama QL Abacus come descritto nell'Introduzione dei Programmi QL, non dimenticare che Abacus richiede una cartuccia formattata nel Microdrive 2. Al richiamo avvenuto il seguente messaggio si leggerà sullo schermo.

```
QL ABACUS
tabella elettronica
Versione x.xx
Copyright ©1984 PSION Ltd.
Tutti i diritti riservati
```

dove x.xx rappresenta il numero della versione (es. 2.23).

Il programma attende per alcuni secondi prima di essere disponibile.

L'informazione di Aiuto non viene caricata nella memoria del computer insieme al programma. Viene semplicemente letta dalla cartuccia di Abacus quando se ne ha bisogno. Per questa ragione non devi rimuovere la cartuccia di Abacus dal Microdrive 1 se ti vuoi avvalere dell'assistenza delle note di **Aiuto**

Appena Abacus viene caricato nella memoria, lo schermo appare come illustrato alla Figura 2.1. Questo è il *quadro principale* di visualizzazione.

ASPETTO GENERALE

Il Quadro di Abacus permette la visualizzazione a 80, 64 o 40 caratteri. Se stai usando un apparecchio televisivo, la visualizzazione potrebbe non essere abbastanza chiara da permetterti di leggere 80 caratteri per riga e per questa ragione dovresti usare 64 o 40 caratteri. La visualizzazione a 64 caratteri è molto simile a quella a 80 caratteri, mentre la visualizzazione a 40 caratteri è impostata un po' più diversamente. Questo è illustrato alla Figura 2.2.

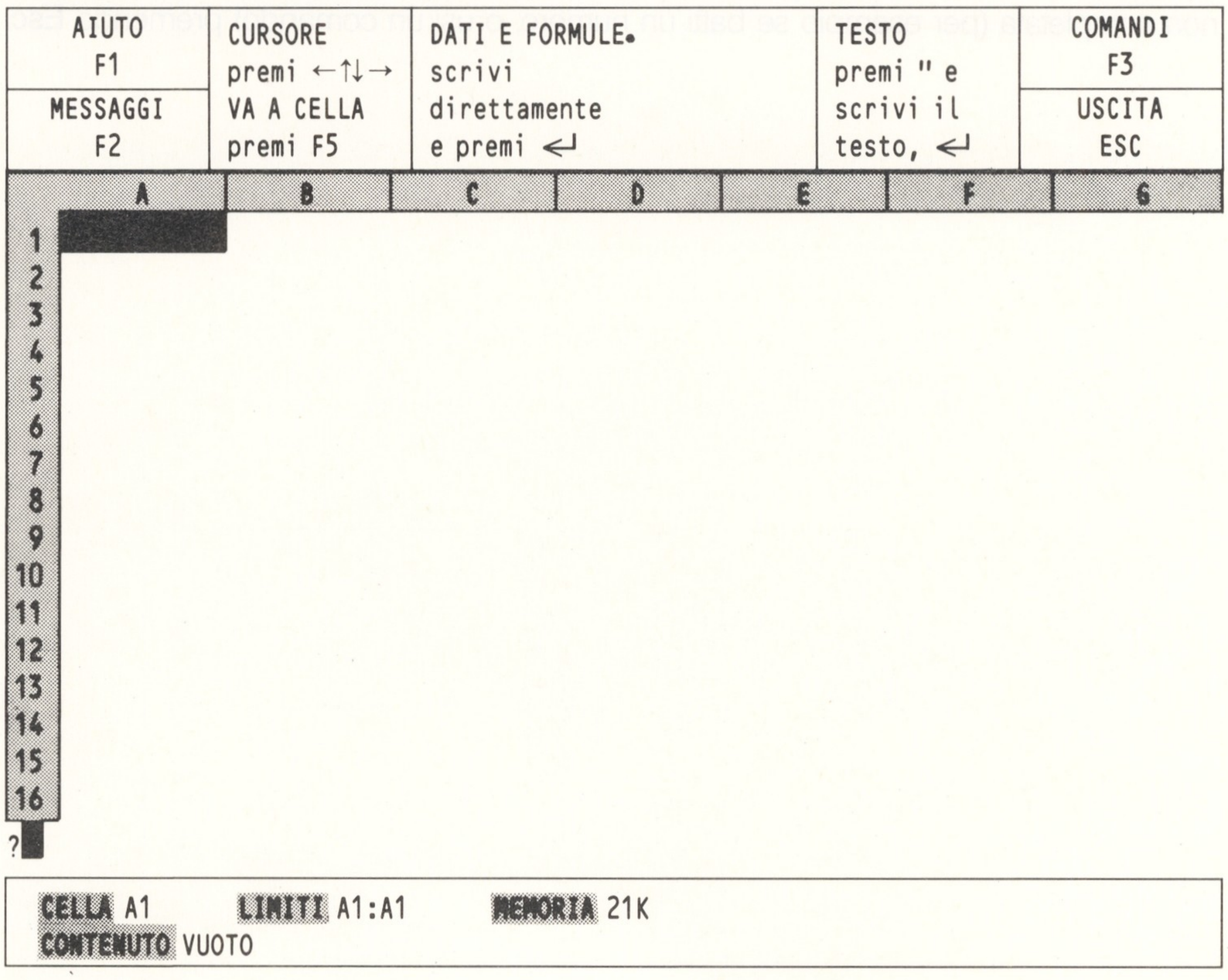


Figura 2.1 Quadro principale di visualizzazione con video (80 caratteri)

Abacus inizialmente seleziona la visualizzazione a 80 o 64 caratteri a seconda se hai premuto il tasto dato F1(monitor) o F2(tv).

A parte la differenza in aspetto, Abacus opera esattamente allo stesso modo con tutti e tre i formati dei quadri. La maggior parte dei diagrammi in questo manuale sono illustrati con visualizzazione a 80 caratteri.

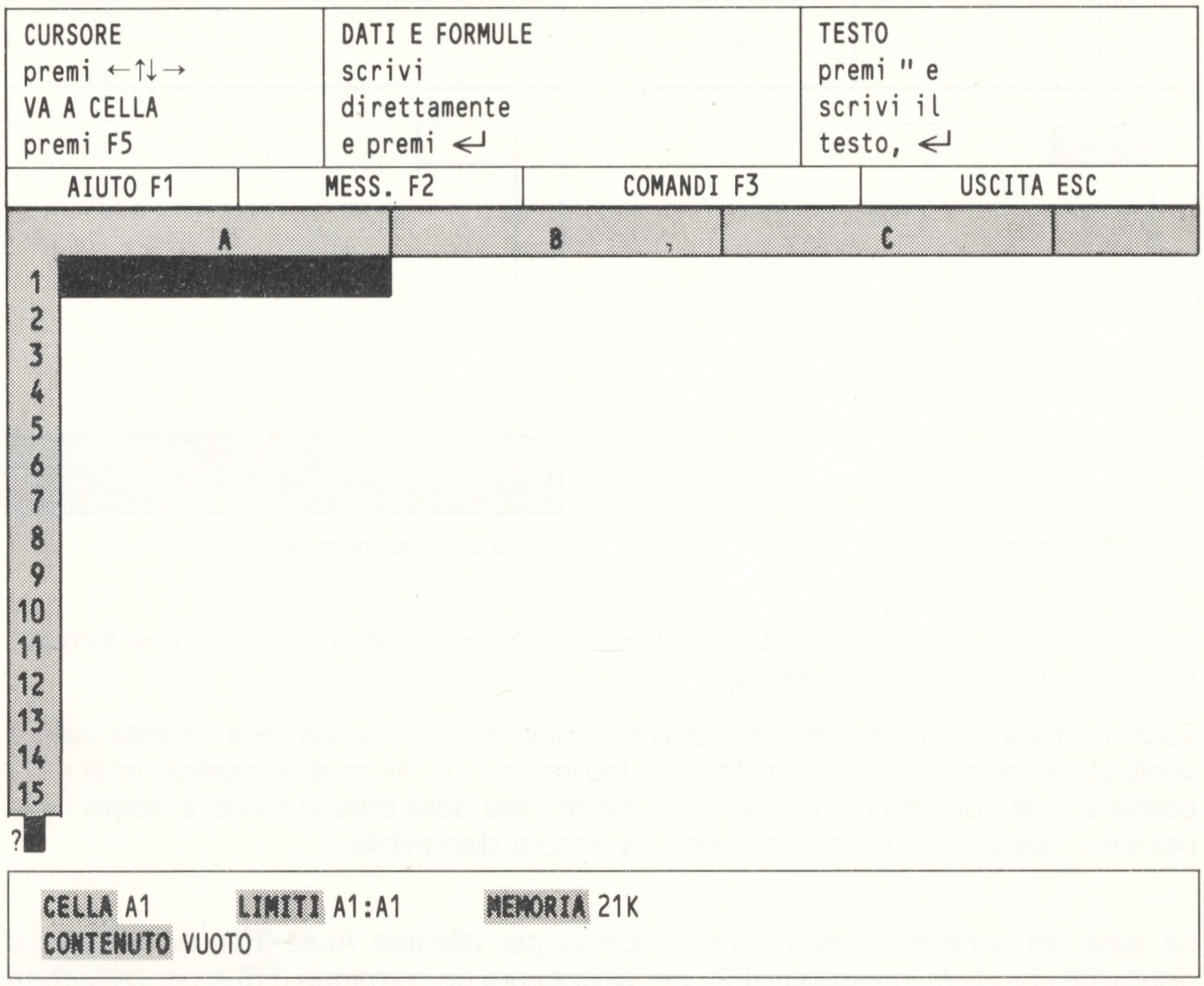


Figura 2.2 Quadro principale di visualizzazione per leggere 40 caratteri.

La Finestra

La zona centrale dello schermo contiene la *finestra* che mostra parte del reticolo.

Lungo la parte superiore della finestra vedrai una riga in cui appaiono un numero di lettere. Queste lettere denominano le colonne verticali che formano il reticolo. Come potrai vedere le colonne A,B,C, e così via, sono visibili. Lungo il lato della finestra, in basso c'è una serie di numeri, da 1 a 15. Questi numeri denominano le righe nel reticolo.

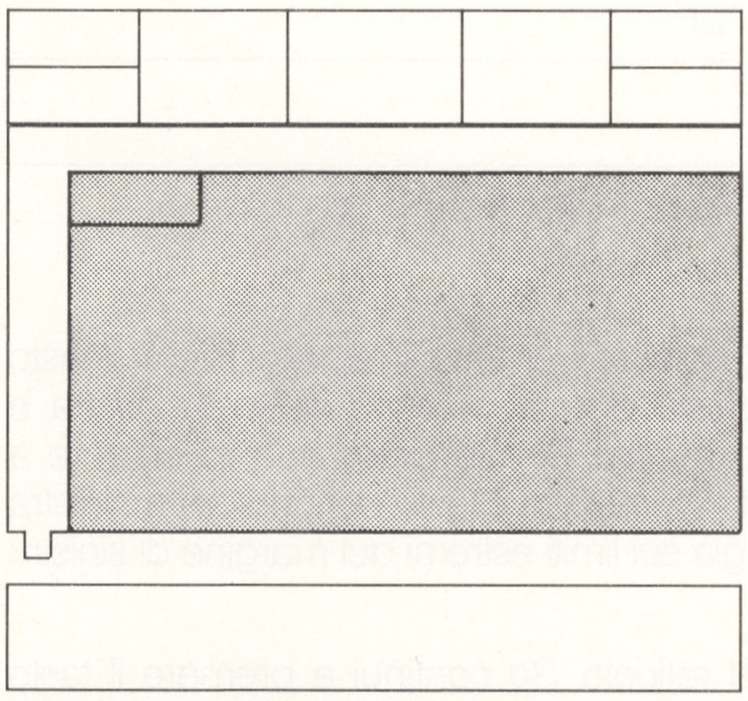


Figura 2.3 La finestra

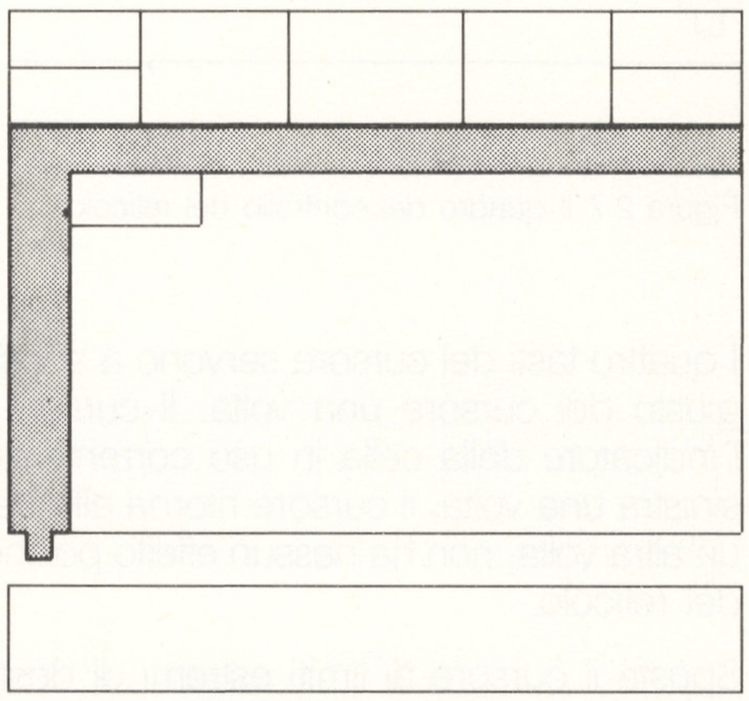


Figura 2.4 La nomeclatura del reticolo

Una combinazione di una lettera con un numero identifica una specifica cella, ed è conosciuta col nome di *rifcella*. Per esempio, A1. Questo si riferisce alla cella che si trova nella colonna A e riga 1, (la cella nella parte sinistra superiore della finestra).

Avrai notato che questa cella è diversa da tutte le altre, nel senso che contiene una striscia rettangolare. Questo rettangolo si chiama il cursore e identifica la *cella in uso corrente*, quella la cella che riceverà i dati che immetterai con la battitura dei tasti.

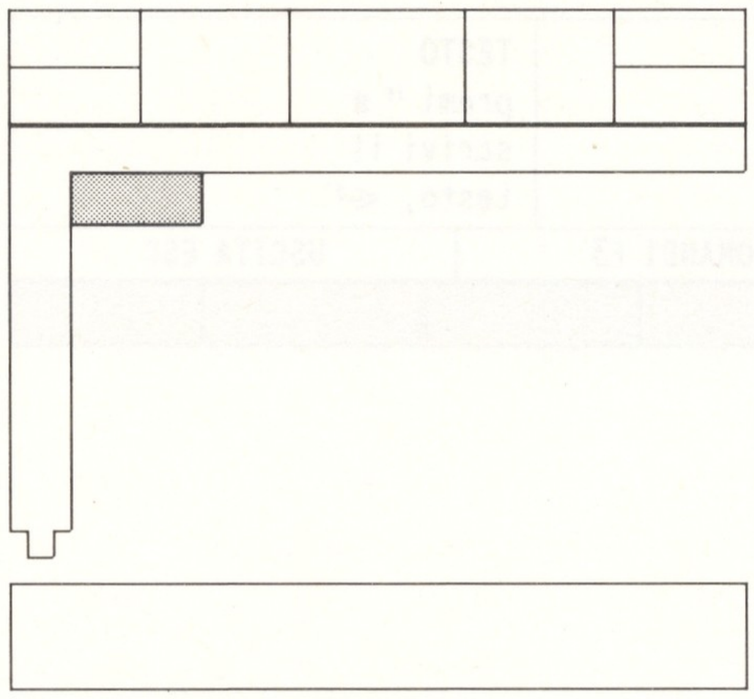


Figura 2.5 Il cursore

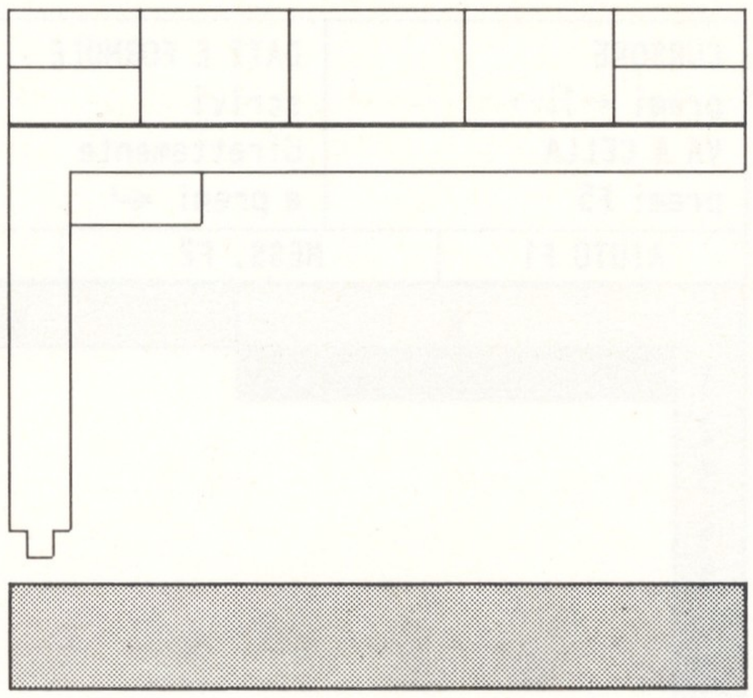


Figura 2.6 Riga informativa di controllo

La riga informativa di controllo

La sezione alla base del quadro, contiene la *riga informativa controllo*, che fornisce informazioni sullo stato del reticolo.

Essa mostra il riferimento cella e il relativo contenuto della stessa cella. Questa cella è vuota subito dopo il richiamo di Abacus. Inoltre, la zona informativa mostra i limiti della porzione usata del reticolo (come il riferimento cella della cella in fondo a destra della porzione usata) e la quantità di memoria ancora disponibile.

Il quadro del controllo del reticolo

La zona del controllo mostra normali opzioni per ottenere Aiuto (**F1**), per attivare e disattivare i prompts (messaggi) (**F2**), per selezionare un comando (**F3**) e per cancellare una selezione incompleta (**ESC**). Inoltre vi sono tre opzioni che sono specificamente di Abacus. Queste sono:

muovi il cursore
batti i dati o le formule,
batti il testo.

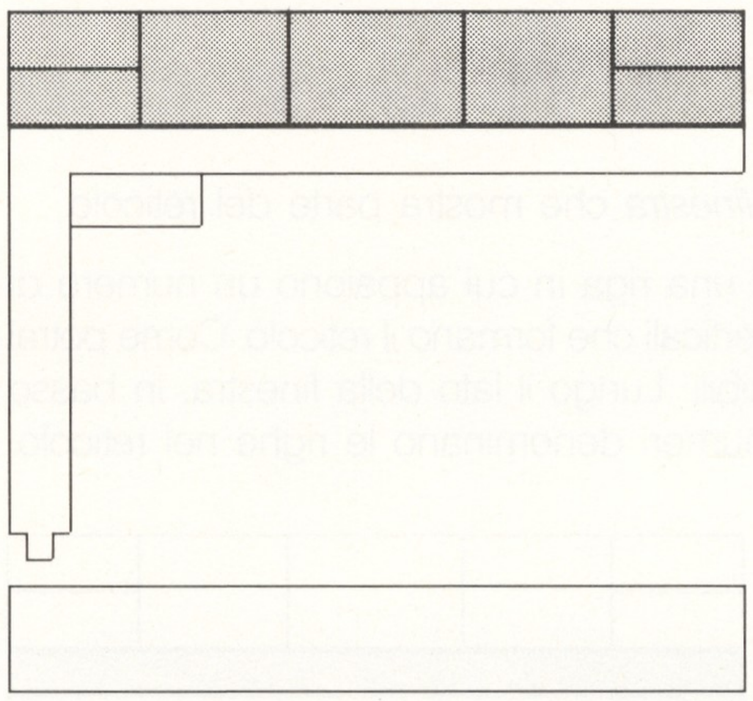


Figura 2.7 Il quadro del controllo del reticolo

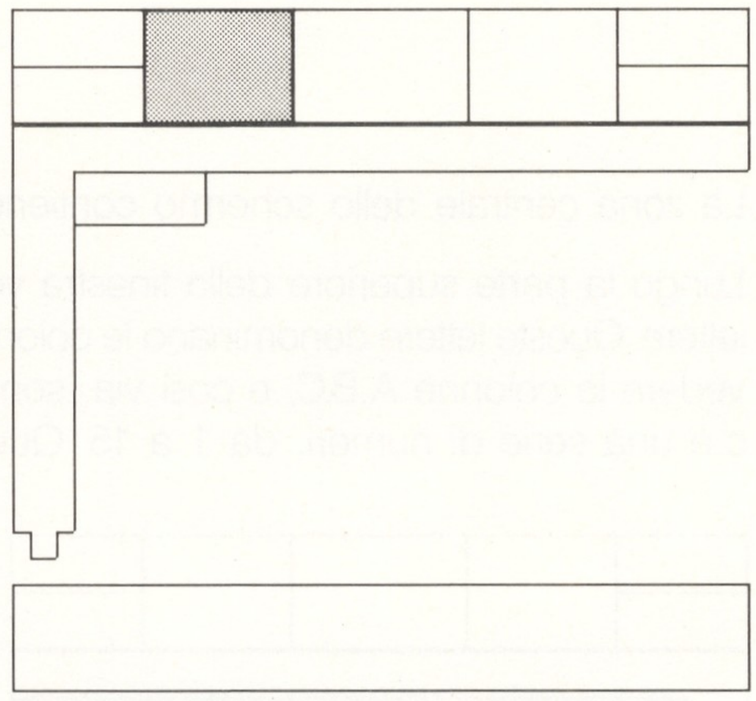


Figura 2.8 Spostamento del cursore

SPOSTAMENTO DEL CURSORE

I quattro tasti del cursore servono a spostare il cursore per tutto il reticolo. Premi il tasto giusto del cursore una volta. Il cursore si sposta di una colonna verso la destra e l'indicatore della cella in uso corrente adesso mostra B1. Se premi il tasto cursore a sinistra una volta, il cursore ritorna alla cella A1. Premendo il tasto del cursore a sinistra un'altra volta, non ha nessun effetto poichè sei già sui limiti estremi del margine di sinistra del reticolo.

Sposta il cursore ai limiti estremi di destra del reticolo. Se continui a premere il tasto per lo spostamento del cursore, questo non si sposta mentre le lettere della testata della finestra cambiano. Quando tenti di spostare il cursore dalla zona visibile del reticolo, la finestra si sposta lungo il reticolo, in modo che il cursore rimanga sempre in vista.

I tasti del cursore rappresentano un metodo utile per il movimento del cursore, purchè desideri spostarlo di una o due celle. Essi sono inefficienti per effettuare grandi spostamenti sul reticolo. Per questo tipo di spostamenti è più conveniente andare direttamente alla cella richiesta. E' possibile farlo premendo il tasto **F5** per selezionare l'opzione **VA A CELLA**, e poi battere il rifcella richiesto, seguito da **ENTER**.

Come esempio dell'uso dell'opzione **VA A CELLA**, chiedi ad Abacus di spostare il cursore alla cella D11. Per prima cosa batti il tasto **F5** per selezionare l'opzione **VA A**. Le parole **VA A>A1**" appariranno nella riga subito sotto la finestra. Abacus suggerisce che il cursore venga spostato nell'angolo superiore sinistro del reticolo. Se accetti questo suggerimento (premendo solamente il tasto **ENTER**) il cursore si sposterà in quella posizione. Per spostare il cursore ad un'altra cella, batti il rifcella – in questo caso batti:

d11

e premi **ENTER**. Nota che la lettera "d" potrà essere maiuscola o minuscola – Abacus non fa nessuna distinzione tra l'una e l'altra. Il Rifcella che batti sostituisce quello suggerito da Abacus e il cursore si sposta direttamente alla cella che hai specificato.

A questo punto, sposta il cursore dove si trovava all'angolo superiore sinistro del reticolo, usando questa opzione di nuovo. Questa volta puoi accettare il rifcella suggerito da Abacus (A1) e quindi devi solo battere:

F5 ENTER

Troverai che ritorni così alla posizione originale di visualizzazione, con il cursore all'angolo superiore sinistro della finestra, alla cella A1.

Adesso sposta il cursore alla cella Y1, battendo

F5 y1 ENTER

Osserva le lettere che identificano le colonne lungo la testata della finestra e vedrai che la colonna a destra della colonna Z è stata denominata AA, la colonna seguente denominata AB, e così via di seguito. Questo ti permette di riferirti a più di 26 colonne.

Vi sono 64 colonne in totale e dopo la colonna AZ, le colonne sono state tutte denominate BA, BB e così via di seguito. L'ultima colonna nel reticolo stata denominata BL.

Puoi anche effettuare lo spostamento in basso lungo il reticolo, ma questa è un'operazione piuttosto lunga; vi sono infatti più di 255 righe nel reticolo.

IMMISSIONE DI NUMERI

Riposiziona il cursore alla cella A1 e batti

100

ma non premere **ENTER** per il momento. La casella delle opzioni "DATI E FORMULE" viene evidenziata, a conferma dell'operazione. Il **valore>**, del prompt, seguito dal numero 100 appare anche nella riga subito sotto la finestra.

Tutta l'immissione battuta, e il testo che Abacus mostra mentre stai usando un comando, appare in questa riga. Questa è la *riga d'immissione*.

Il rettangolino alla riga d'immissione marca il punto dove il carattere di immissione apparirà, ed conosciuto col nome di cursore d'immissione, per distinguerlo dal cursore principale nella finestra. Se fai uno sbaglio, in qualsiasi momento durante la scrittura della riga d'immissione, puoi correggerlo usando l'editore di riga, come descritto nelle Istruzioni dei Programmi QL.

Quando premi **ENTER**, il valore 100 verrà trasferito alla cella in uso (A1) e la riga d'immissione viene liberata, pronta per altre immissioni. Vedrai che il valore 100 appare anche alla riga informativa in calce al quadro visualizzato.

IMMISSIONE DI TESTO

Scrivere del testo nella cella è lo stesso che immettere un numero, con la differenza che il testo è preceduto da due virgolette. Appena batti le virgolette, Abacus risponde evidenziando la casella di opzione TESTO nella zona del controllo e mostrando **testo>** alla riga dell'immissione. Quindi puoi battere tutto ciò che vuoi che appaia nella cella, seguito da **ENTER**. Non c'è bisogno di scrivere le virgolette di chiusura. Prova a scrivere del testo in alcune celle e, in particolare, nota la differenza tra l'immissione, diciamo:

1000 ENTER (un numero)

e

"1000 ENTER (testo)

Un numero viene mostrato a destra della cella, mentre testo viene scritto alla sinistra. La zona informativa di controllo mostra pure il tipo d'informazione; testo, numeri e così via, nella cella in uso corrente.

I COMANDI

Un comando viene selezionato col tasto **F3**.

Il quadro centrale per il controlo mostra una lista, o menù, dei comandi disponibili ed conosciuto col nome di *menù comandi*, illustrato alla Figura 2.9.

La maggior parte dei comandi sono descritti in altri capitoli. Intanto possiamo dare uno sguardo a due di essi. Questi sono **Zero (pulisci), che usi per cancellare il contenuto della tabella elettronica e Lascia** che permette di abbandonare Abacus e ritornare a SuperBASIC.

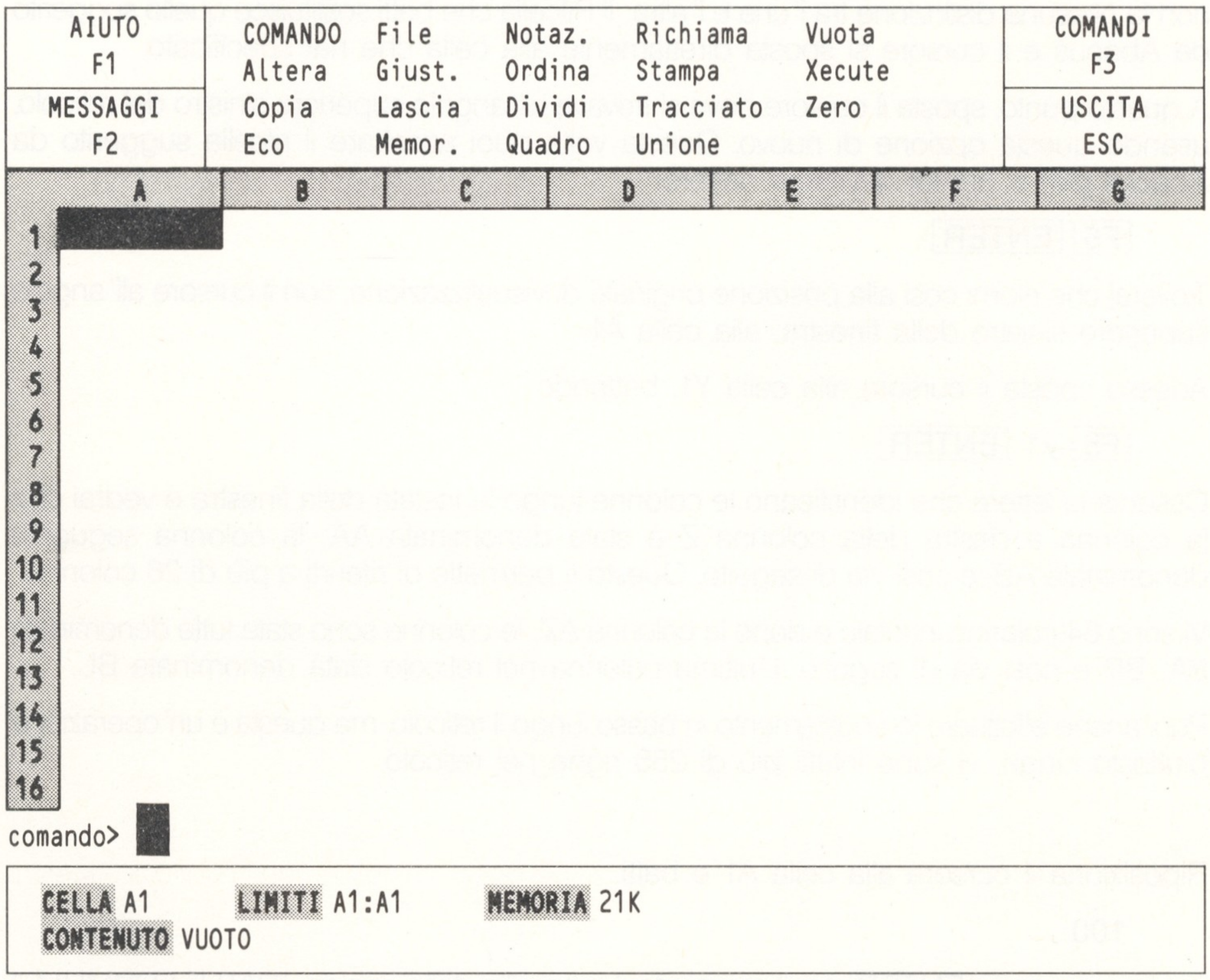


Figura 2.9 I menù comandi

Prova ad usare il comando **zero** per iniziare. Premi **F3** e trova il comando **Zero** nel menù che viene mostrato sullo schermo. Se premi la lettera **Z**, la parola zero appare nella riga d'immissione – è sufficiente battere solo la prima lettera del comando. Allo stesso tempo, la casella dei comandi nella zona del controllo, cambia per mostrare le funzioni del comando **Zero**. Prova a premere **ESC**, per annullare il comando.

Adesso visualizza il menù comando premendo **F3**, quindi premi **Z** per richiamare il comando **Zero** di nuovo, ma questa volta continua l'operazione premendo **ENTER** subito dopo, per pulire tutta la tabella. Sullo schermo vedrai il reticolo liberato e il cursore alla cella A1, pronto per il riavvio.

Ogni volta che vuoi abbandonare Abacus e ritornare a SuperBASIC, devi usare il comando **Lascia**. Questo funziona in modo analogo a **Zero**, (premi **F3** e poi la prima lettera del comando (**L**)). "Lascia" causa la perdita del contenuto del reticolo, quindi puoi annullare l' esecuzione.

CAPITOLO 3
CELLE, RIGHE, COLONNE E SERIE

L'abilità di Abacus è la sua capacità di trattare righe intere, colonne o serie di celle con un'unica operazione. Questo lo fai usando semplici espressioni che ti permettono, per esempio, di riempire tutta o parte di una riga di celle. I valori nelle celle possono essere tutti uguali o possono variare in un modo regolare.

Questo capitolo descrive alcune caratteristiche delle celle e i vari modi con cui puoi far riferimento ad esse.

CELLE

La cella è l'unità base per contenere informazioni in Abacus. Ogni cella può contenere una voce d'informazione che potrebbe essere testo, un numero o una formula.

Per ogni cella che contiene informazione, Abacus mantiene anche un record di come si deve visualizzare quell'informazione. Puoi, per esempio, visualizzare numeri o testo alla sinistra, al centro o alla destra della cella, e puoi visualizzare numeri in diversi formati.

Giustifica

Il comando **Giustifica** è usato per cambiare la posizione della visualizzazione entro una cella. Ti permette di scegliere la posizione dei numeri o del testo dentro una cella o un gruppo di celle.

Scrivi un valore di 100 nella cella A1, poi usa il comando **giustifica** premendo **F3** seguito dal tasto **G**. Abacus ti chiede prima di selezionare le opzioni Celle e Defaults; seleziona l'opzione Celle premendo **ENTER**. Abacus quindi ti chiede di scegliere tra testo o numeri. Scegli numeri, premendo il tasto **N**. Dopo di che devi scegliere giustifica a Sinistra, Centro o Destra. Dal momento che Sinistra viene suggerita da Abacus sceglila premendo **ENTER**.

Infine Abacus ti chiede di specificare la serie di celle che ne subiranno l'effetto. In questo caso premi **ENTER**. Vedrai che il valore di 100 nella cella A1 si sposta alla sinistra della cella.

Nota che puoi cambiare il formato dei numeri o la giustificazione dei numeri di una cella che adesso contiene testo. Non si vede accadere niente. Se, comunque, più tardi cambi il contenuto della cella a numerico, lo visualizzerai nel formato e giustificazione che avevi specificato. Questo vale anche se cambi giustificazione del testo in una cella che correntemente contiene informazione numerica.

Celle vuote

Celle che non contengono nessuna informazione non esistono per quanto riguarda Abacus, e non usano alcuna parte della memoria. Perciò non posseggono nessuna caratteristica. Se provi ad usare l'opzione Celle del comando **Giustifica** oppure **Notazione** su un cella vuota, questi comandi non avranno nessun effetto. Numeri che in seguito vengono inseriti in una di queste celle, verranno mostrati nel formato generale di default.

Se vuoi cambiare questi default devi usare l'opzione di Default dei comandi **Giustifica** oppure **Notazione** (o ambedue). Per esempio, usa l'opzione di Default del comando **Notazione** (premi **F3**, **N** e poi (**P**) per selezionare la percentuale con una cifra decimale. Le scelte sono analoghe a quelle nelle opzioni di Celle, solo che non ti viene chiesto di specificare una serie di celle.

L'opzione di Default del comando **Giustifica** opera allo stesso modo. Anche qui non ti viene chiesto di scrivere la serie di cella poichè Abacus usa il nuovo default ogni volta che scrivi una nuova informazione in una cella che precedentemente era vuota.

La nuova registrazione dei default rimane effettiva fino a quando non la cambi di nuovo, o fino a quando non finisci di usare Abacus e ritorni a SuperBASIC.

Per ripristinare i default allo stato originale – giustifica numeri a destra, giustifica testo a sinistra e numeri mostrati in formato Generale – usa la seguente procedura:

```
F3 G V N D          {numero giustificato a destra}
F3 G V ENTER ENTER  {testo giustificato a sinistra}
F3 U V G            {numero visualizzato in formato generale}
```

RIGHE

Molto spesso vorrai scrivere un determinato valore in diverse celle di una determinata riga. Abacus dispone di semplici metodi per effettuare quest'operazione. Uno dei metodi di riferirsi alle celle di una riga è un *identificatore*. Vi sono due identificatori: **riga** e **col**. Essi si riferiscono alle celle della riga o colonna in uso – la riga o la colonna che contiene il cursore.

Per esempio, scriviamo il valore di 100 nella prima riga, dalla colonna B alla colonna D. Useremo l'identificatore di **riga** nel modo seguente. Sposta il cursore alla cella A1 e poi batti:

```
riga = 100 [ENTER]
```

Appena premi **ENTER** un prompt appare alla riga d'immissione per suggerire di iniziare a scrivere nella riga alla colonna A (la colonna che contiene il cursore). Il sistema offre sempre un suggerimento come punto d'inizio, e questo suggerimento può essere accettato premendo il tasto **ENTER**. In questo caso, comunque, noi vogliamo iniziare alla colonna B quindi devi scrivere:

```
B [ENTER]
```

La riga dell'immissione cambia, per indicare che la scrittura nella riga deve iniziare alla colonna B e un altro prompt appare con il suggerimento BL (l'ultima colonna nel reticolo), per indicare la colonna dove si vuole terminare. Ancora a volta questo deve essere cambiato, dato che noi vogliamo terminare alla colonna D. Quindi devi scrivere:

```
D [ENTER]
```

L'istruzione viene così completata e sarà eseguita – il valore 100 appare in ogni cella da B1 a D1 inclusa e la riga della immissione viene cancellata, pronta per la successiva immissione.

COLONNE

La scrittura in una colonna funziona in modo analogo, con l'eccezione, naturalmente, che farai riferimento alle colonne con una o due lettere anzichè con il numero che identifica una riga. Supponiamo di voler scrivere il testo "ciao" in ogni cella della colonna D, dalla riga 5 alla riga 11. Noi possiamo effettuare quest'operazione usando il secondo identificatore **col**. Sposta il cursore alla cella D5 e scrivi:

```
col = "ciao" [ENTER]
```

Questa volta Abacus suggerisce il punto d'inizio giusto (riga 5). Poichè questa riga contiene il cursore, puoi accettare questo suggerimento premendo il tasto **ENTER**. La riga 255 viene poi suggerita come punto di termine della scrittura. Deve essere modificata scrivendo:

```
11 [ENTER]
```

Il testo apparirà dalla cella D5 alla cella D11 inclusa e la riga della immissione viene cancellata pronta per la succesiva immissione.

Ogni volta che usi **col** ti verrà chiesto di specificare la prima e l'ultima riga che vuoi utilizzare per la scrittura. Puoi, di solito, accettare o cambiare il valore suggerito da Abacus.

Oltre a questo uso gli identificatori di **riga** e **col**, sono usati per definire la serie di celle per qualsiasi funzione per cui necessita tale serie. Es.: Per esempio, scrivi dei numeri in tutte le celle della zona rettangolare il cui angolo superiore sinistro è la cella A1 e il cui angolo inferiore destro è C3 (nove gruppi di cifre in tutto). Adesso sposta il cursore alla cella D1 e batti:

```
col = somma(riga) [ENTER]
```

Questa operazione scriverà nelle celle della colonna D il totale dei valori contenuti nelle celle della corrispondente riga. Abacus ha bisogno di sapere la serie sia della **riga** che della **col**. Perciò ti chiederà di definire la serie di colonne per **riga** (Abacus suggerisce da colonna A a colonna C, che è giusto – accetta il suggerimento premendo il tasto **ENTER** e poi per la serie di righe da essere usata per **col**. Abacus suggerisce da riga 1, che è giusta, a riga 255 (oppure riga 11 se batti quest'esempio subito dopo quello precedente). Accetta il primo suggerimento premendo il tasto **ENTER** e batti il valore giusto, 3 (non dimenticare di premere **ENTER**) di nuovo. Abacus allora calcola il totale per ognuna delle tre righe e mostra il risultato nelle celle della colonna D.

NOMI (NOMECLATURA)

Gli esempi precedenti si riferiscono a righe e colonne con l'uso specifico del riferimento del numero e della lettera del rifcella. Un'importante alternativa per identificare righe e colonne è l'uso di **nomi**, che puoi scegliere tu stesso. Questi nomi sono usati per riferirsi a determinate righe, colonne o celle.

Il testo che scrivi in una cella può essere usato come un nome. Puoi usare i nomi in comandi o formule dove altrimenti useresti un riferimento con una lettera e una cifra. Il vantaggio è nel fatto che più facile ricordare nomi che lettere e numeri, quando vuoi riferirti ad una determinata cella.

Questo è un metodo estremamente flessibile ed efficiente, che puoi usare con grande vantaggio, per semplificare la disposizione e la gestione di un reticolo. Le due sezioni seguenti spiegano come puoi utilizzare questi nomi.

Nomi di colonne e di righe

Un nome si può riferire ad una riga od una colonna, secondo il contenuto delle altre celle nel reticolo. La regola fondamentale, quando usi un nome per identificare una riga o una colonna di cifre, è che Abacus esegue la ricerca da una posizione inferiore e verso la destra dalla cella che contiene il nome. La cella più vicina che contiene un numero, inferiore o alla destra della posizione del nome, determina se il nome si riferisce ad una riga o ad una colonna. Figura 3.1 e Figura 3.2 dovrebbero rendere più chiaro quanto detto. Nella Figura 3.1 la nomeclatura si riferisce ad una riga e nella Figura 3.2 si riferisce ad una colonna.

In casi più complessi, per esempio dove vi sono numeri sia alla destra che sotto il nome, il numero più vicino (misurato dal numero di celle che separano il numero dal nome) determina se si tratta di riferimento riga oppure di riferimento colonna. Se i due numeri sono alla stessa distanza dal nome, Abacus da questo messaggio:

```
Il nome non una riga o colonna.
```

e attende che prema la barra spaziatrice. Abacus quindi rimette il testo della formula sulla linea dell'immissione, in modo che possa correggerla con l'editore di riga. Dovresti sostituire il riferimento non risolto con **riga** oppure **col** e di nuovo premere **ENTER**. Dovresti anche prendere in considerazione la revisione della nomeclatura in modo che Abacus possa risolvere il riferimento in futuro.

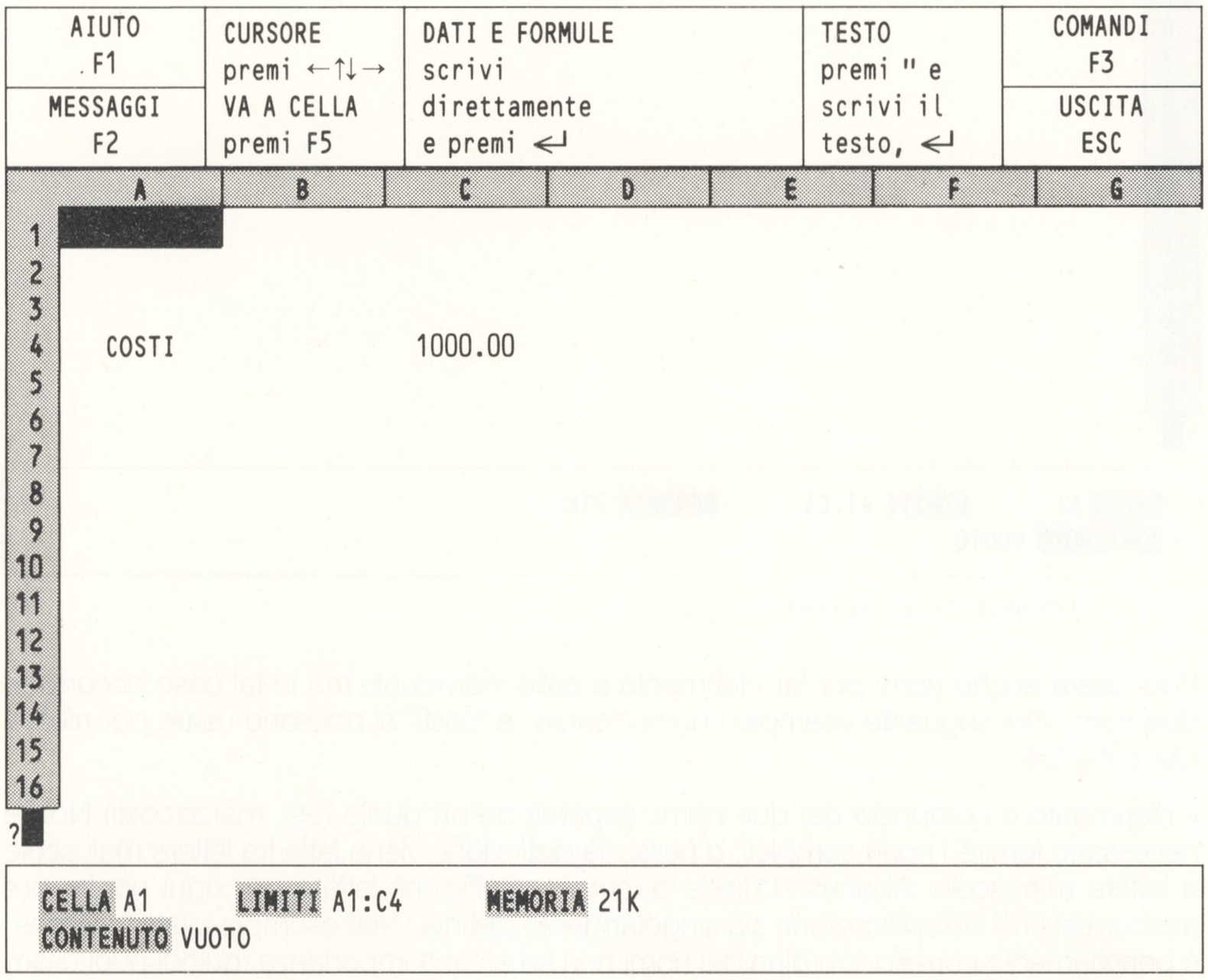


Figura 3.1 Nomeclatura di riga

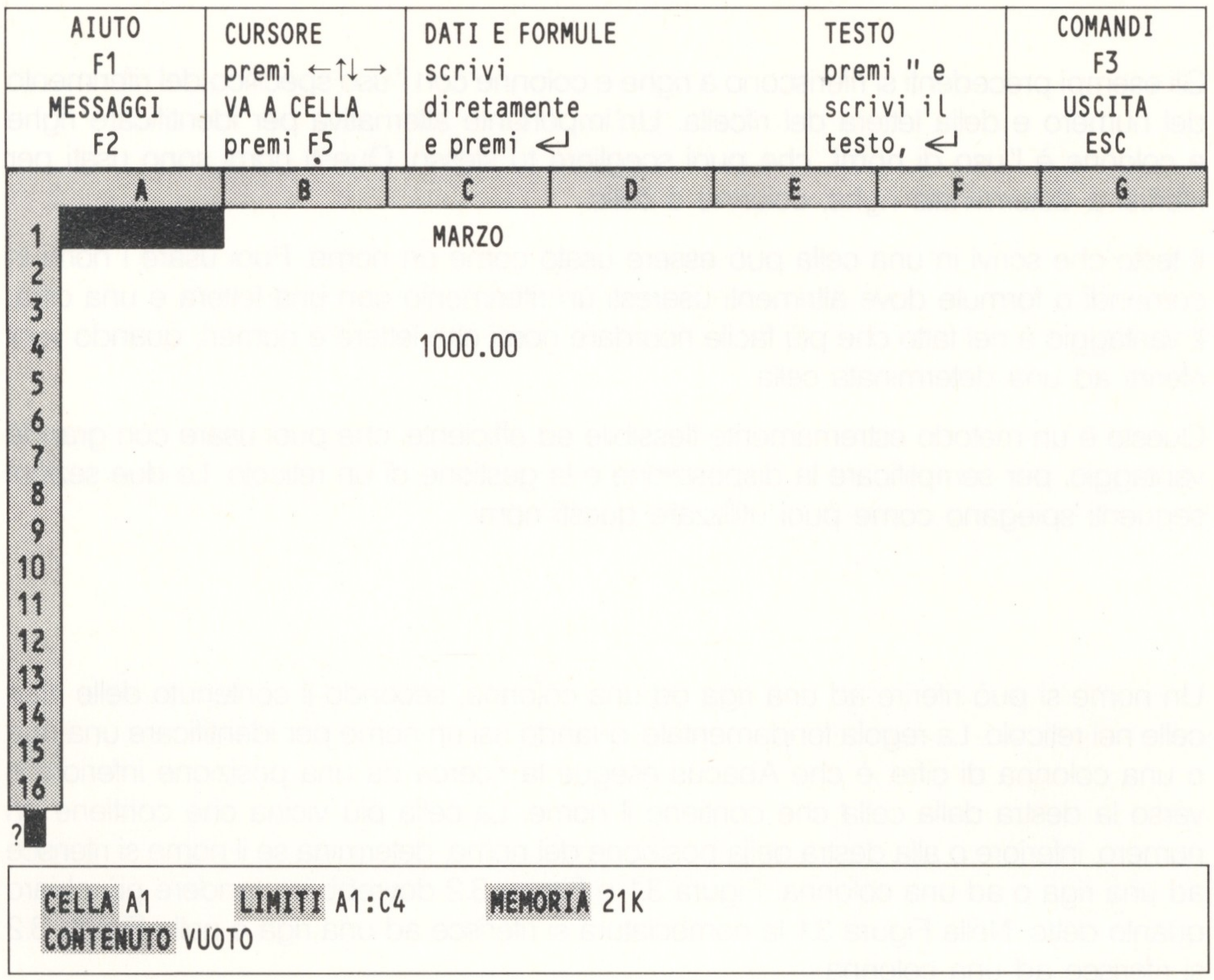


Figura 3.2 Nomeclatura di colonna

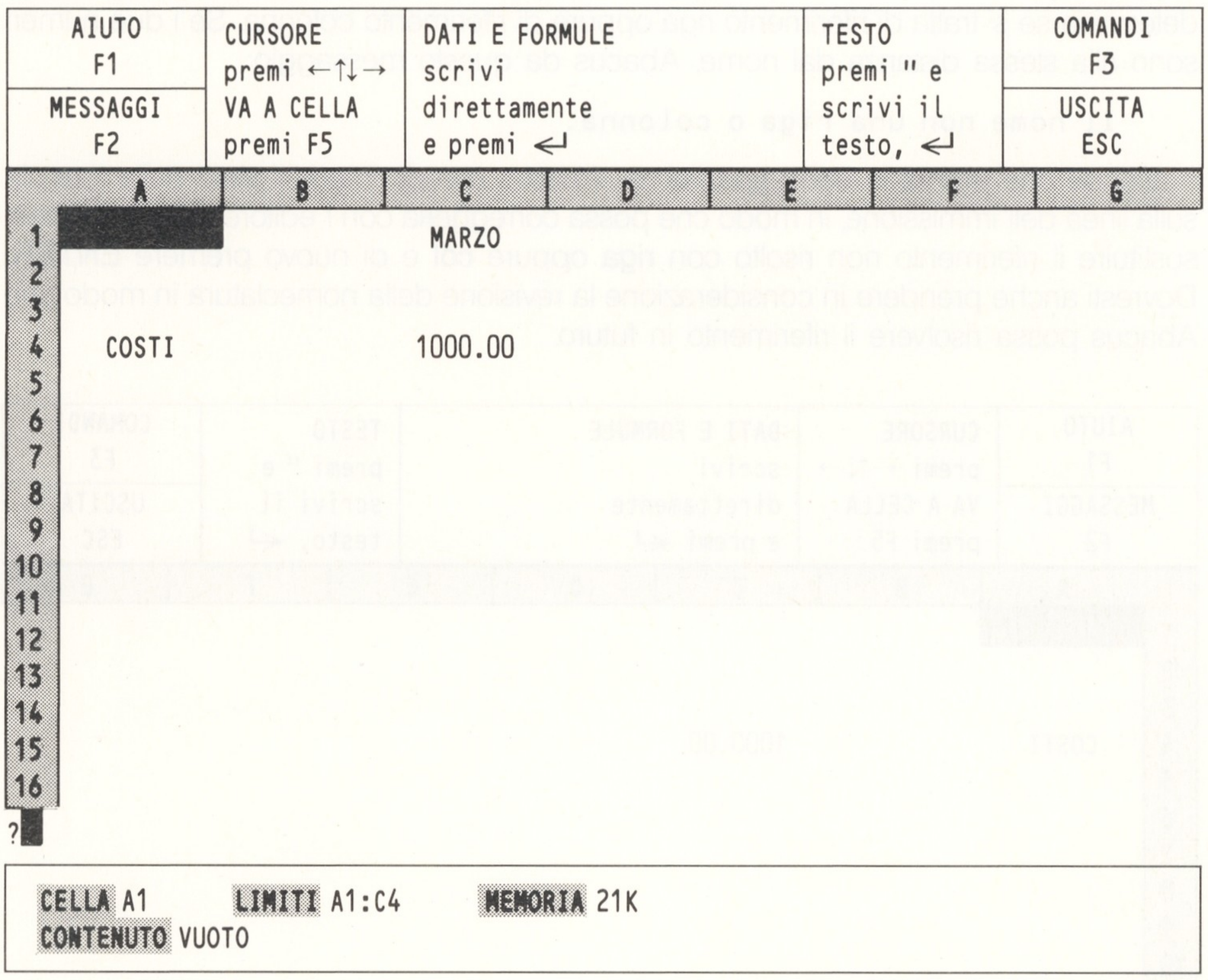


Figura 3.3 Nomeclatura di una cella

Nomeclatura delle celle

Puoi usare anche nomi per far riferimento a celle individuali, ma in tal caso occorrono due nomi. Nel seguente esempio i nomi "marzo" e "costi" si possono usare per riferirsi alla cella C4.

Il riferimento è composto dai due nomi, separati da un punto (es., marzo.costi) Non è necessario fornire i nomi completi, e nessuna distinzione viene fatta tra lettere maiuscole e lettere minuscole. Abacus richiede solamente sufficienti lettere per ogni nome, per assicurarsi che l'identificazione sia singolarmente distinta. Nell'esempio citato "mar.cos" è perfettamente corretto. L'ordine dei nomi non ha alcuna importanza, quindi puoi usare "cos.mar" per riferirti alla stessa cella.

SERIE

Oltre alla possibiltà di poter far riferimento a un' intera riga o un' intera colonna puoi anche far eseguire i comandi su un blocco rettangolare, o *serie*, di celle.

Un riferimento di serie è composto da due parti. La prima parte è il riferimento della riga e colonna della cella superiore sinistra della serie (rifcella). Questa è separata da due punti (:) dalla seconda parte, che rappresenta il riferimento della riga e colonna (rifcella) dell' angolo inferiore destro della serie. Un esempio del riferimento della serie :

```
A2:D27
```

Un esempio dell' uso del riferimento serie è nell' uso del comando **Copia**, per copiare il contenuto di una serie di celle ad una simile serie in un' altra posizione nel reticolo.

Molti comandi ti chiedono di battere il riferimento della serie, per identificare le celle su cui devono effettuare l' esecuzione. Dal momento che un riferimento di serie possiede una serie di possibilità più vasta di un riferimento di riga o di colonna, Abacus non può suggerire una serie alternativa. Devi battere l' intero riferimento della serie tu stesso. Puoi specificare la serie in quattro modi diversi. Questi sono:

1) Con specifici dettagli di lettere e numero di riga e di colonna,
   es. A1:C7

2) Con nomi,
   es. gennaio.vendite:marzo.costi

3) Con una combinazione dei due metodi citati,
   es. A1:marzo.costi

4) Con un identificatore di serie,
   es. riga (o col)

Questo si riferisce alle celle della riga (o colonna) che contiene il cursore. In questo caso, Abacus può suggerire punti iniziali e finali adatti.

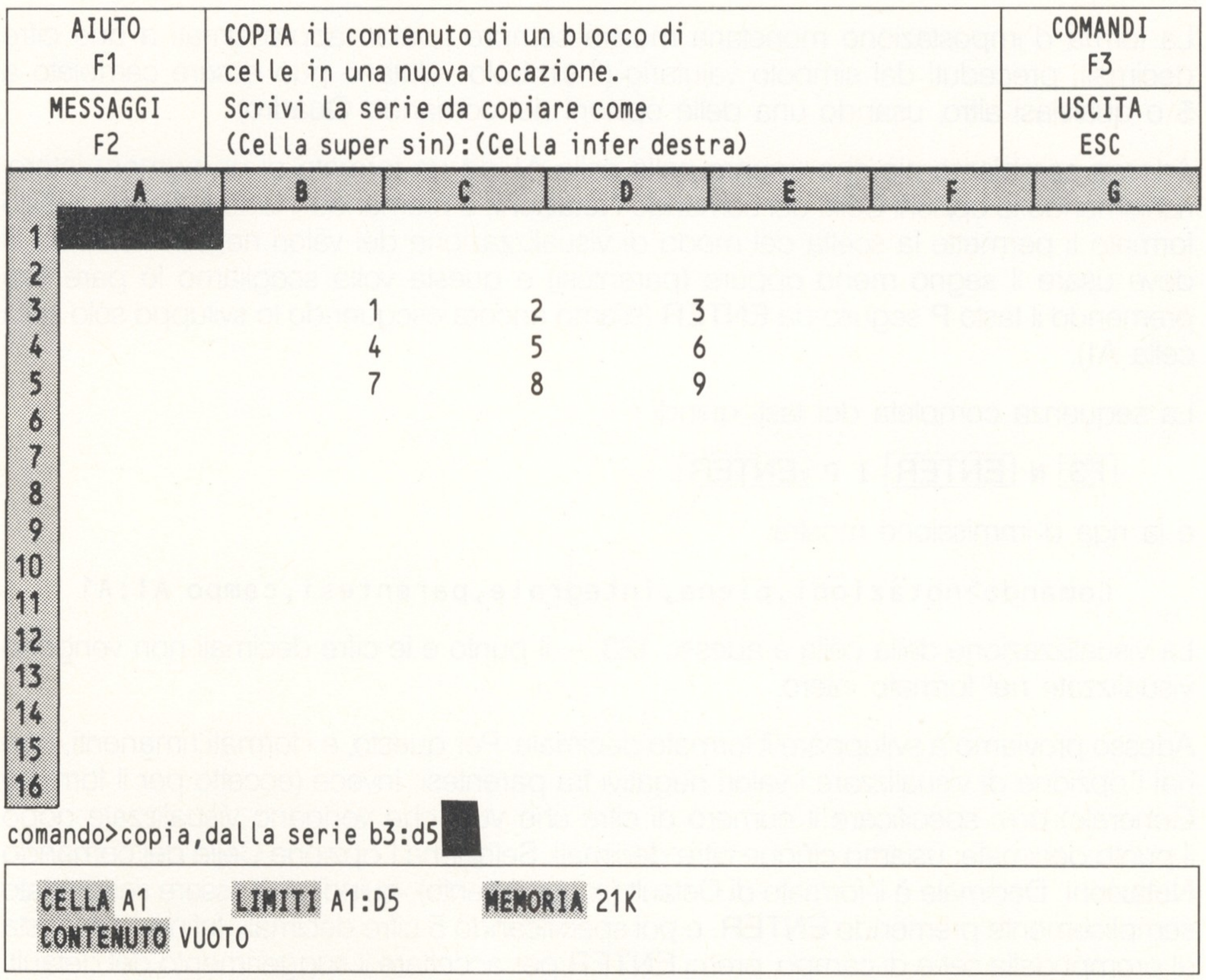


Figura 3.4 Un riferimento di gamma

ALTRI DETTAGLI SUI NUMERI E SUL TESTO

Adesso che abbiamo visto come definire una cella o una serie di celle, possiamo proseguire per illustrare come il contenuto di queste celle può essere modificato. Prima è necessario spiegare come avviene la memorizzazione. Sposta il cursore alla cella A1 e scrivi il numero 123.456. Abacus memorizza tutti i numeri con una precisione di 16 cifre significative e può mostrare fino a 14 cifre significative – le due cifre extra vengono usate per essere sicuri che il valore calcolato venga mostrato con precisione. Sebbene Abacus calcoli e memorizzi tutti i numeri con questa precisione, non c'è bisogno di mostrare tutte le cifre significanti.

Seleziona il comando **Notazioni** (premendo il taso **F3** seguito dal tasto (**N**). Vi sono due opzioni: Celle o Default. In questo caso premi il tasto **ENTER** per scegliere l'opzione "Celle" suggerita.
Abacus ti offre varie forme d'impostazione.

Premi il tasto **M** per scegliere la forma d'impostazione Monetaria. Abacus ti chiede come vuoi visualizzare i valori negativi. Abacus suggerisce che essi vengano mostrati con segno – (meno) questo suggerimento può esere accettato premendo il tasto **ENTER**. Alternativamente puoi far mostrare questi valori in parentesi e in tal caso devi premere invece il tasto **P**. In quest'esempio non importa quale opzione scegli, ma ammettiamo di scegliere l'opzione meno.

Abacus quindi ti chiede di definire la serie di celle che devono gestire lo sviluppo. Puoi rispondere scrivendo un riferimento di serie (es. A1:B3) oppure semplicemente il riferimento di una singola cella. Abacus cerca sempre di prevenirti fornendoti la serie richiesta. Comunque, in alcune circostanze Abacus non può far altro che suggerire la serie A1:A1. Questo riferimento di serie è identico al singolo riferimento di cella A1. (rifcella). Puoi accettare il suggerimento premendo il tasto **ENTER**, oppure scrivere un riferimento di tua scelta seguito da **ENTER**.

Supponiamo che Abacus suggerisca la serie default A1:A1 la completa sequenza da seguire nel premere i tasti :

F3 N ENTER M ENTER ENTER

Prima che tu prema **ENTER** per la terza volta, la riga d'immissione dovrebbe contenere:

```
Comando>notazioni,piena,monetaria,segno meno,campo A1:A1
```

Quando premi il tasto **ENTER**, il contenuto della cella A1 è $123.46, anche se il valore reale (123.456) viene ancora mantenuto e mostrato nella riga informativa dello stato della cella. Abacus automaticamente ti riporta allo stato di visualizzazione principale.

La forma d'impostazione monetaria mostra sempre i valori approssimati a due cifre decimali, preceduti dal simbolo valutario (Il simbolo valutario può essere cambiato a $ o qualsiasi altro, usando una delle opzioni nel comando **Quadro**).

Adesso cambiamo ciò che appare nella cella A1 ad un formato di un numero intero, richiamando le opzioni Celle del comando **Notazioni**, e premendo il tasto **I**. Anche questo formato ti permette la scelta del modo di visualizzazione dei valori negativi, cioè se si deve usare il segno meno oppure (parentesi) e questa volta scegliamo le parentesi premendo il tasto **P** seguito da **ENTER** (stiamo ancora eseguendo lo sviluppo solo sulla cella A1).

La sequenza completa dei tasti quindi :

F3 N ENTER I P ENTER

e la riga d'immissione mostra:

```
Comando>notazioni,piena,integrale,parentesi,campo A1:A1
```

La visualizzazione della cella è adesso 123 – il punto e le cifre decimali non vengono visualizzate nel formato intero.

Adesso proviamo a sviluppare il formato decimale. Per questo, e i formati rimanenti, non hai l'opzione di visualizzare i valori negativi fra parentesi. Invece (eccetto per il formato Generale) devi specificare il numero di cifre che vuoi che vengano visualizzate dopo il punto decimale; usiamo cinque cifre decimali. Seleziona l'opzione Celle del comando **Notazioni**. Decimale è il formato di Default (suggerimento), quindi può essere selezionato semplicemente premendo **ENTER**, e poi specificando 5 cifre decimali. Infine, in risposta al prompt della serie di campo, premi **ENTER** per accettare il suggerimento del default. La completa sequenza dei tasti e la visualizzazione della relativa riga d'immissione sono:

F3 N ENTER ENTER 5 ENTER ENTER

```
Comando>notazioni,piena,decimale,cifre decimali 5,campoA1:A1
```

Nella cella A1 adesso si visualizza 123.45600

Adesso usa nuovamente il comando. Questa volta premi il tasto **P** per specificare il formato Percentuale. Utilizza una cifra decimale e seleziona la cella A1. La completa sequenza dei tasti :

F3 N ENTER P 1 ENTER ENTER

La visualizzazione adesso 12345.6%. L'opzione della percentuale mostra un numero moltiplicato per 100 con il simbolo % aggiunto. Nota che il valore memorizzato, come appare alla riga d'informazione stato, rimane ancora 123.456, senza riguardo alla visualizzazione della cella.

Adesso possiamo provare il formato Esponenziale, con tre cifre decimali, premendo:

F3 N ENTER E 3 ENTER ENTER

Prima di battere **ENTER** per la terza volta, la riga d'immissione deve apparire con la seguente leggenda:

```
Comandi>numerazioni,celle,esponenziali,cifre decimali 3,
    campo A1:A1
```

e dopo averlo scritto, la visualizzazione della cella sarà 1235E+02.

Il formato esponenziale viene usato per visualizzare numeri troppo lunghi o troppo corti per essere scritti in formato decimale. Il numero viene scritto come valore tra 1 e 10, moltiplicato alla potenza di dieci. Il numero 2 300 000 000, per esempio, può essere scritto come 2.3 moltiplicato per 1 000 000 000 e 1 000 000 000 come 10 elevato alla nona potenza (moltiplicato nove decine di volte). Cosicchè il numero 2 300 000 000 può essere scritto in formato esponenziale come 2.3 E+09. Numeri molto piccoli vengono scritti usando potenze negative di dieci. Quindi, il numero 0.000123, che non altro che 1.23 diviso per 1000 (poi riportato alla quarta potenza), può essere scritto in formato esponenziale come 1.23 E−04.

La rimanente opzione è quella del formato Generale, che puoi visualizzare nella cella A1 premendo:

F3 N ENTER G ENTER

La riga d'immissione adesso contiene:

```
Comando>notazioni,piena,generale,campo A1:A2
```

Neppure questo comando ti chiede di specificare il numero di cifre decimali. Usando il formato Generale lasciamo che Abacus scelga una forma adatta per mostrare ogni numero. Abacus fa il meglio possibile nella visualizzazione con la massima precisione possibile nello spazio disponibile.

Prima di lasciare il comando **Notazioni**, prova a visualizzare il numero nella cella A1 in formato decimale, con nove cifre decimali. Batti:

F3 N ENTER ENTER 9 ENTER ENTER

Nella cella A1 adesso visualizzi # # # # # # # # # per indicare che il formato non può essere contenuto nello spazio disponibile. Ogni volta che visualizzi questi segni, devi cambiare il formato da visualizzazione, oppure aumentare la larghezza di quella cella.

Pulisci il tabellone usando il comando **Zero**. Con il cursore alla cella A1, scrivi:

```
"Questo un lungo bit di testo
```

Sebbene il testo sia troppo lungo per essere contenuto nella cella, viene tutto visualizzato. Sconfina nelle celle attigue. Adesso scrivi il numero 1 nella cella B1. Il testo viene tagliato dall'estremità della cella A1 poichè è non ammesso lo sconfinamento in un'altra cella piena. Sposta il cursore di nuovo alla cella A1 e verifica che il testo intero sia memorizzato dando uno sguardo alla riga informativa di stato della tabella.

Sposta il cursore di nuovo alla cella B1 e usa il comando **Vuota** per vuotare celle. Quando usi questo comando ti si chiede di specificare la serie di celle il cui contenuto deve essere cancellato. In questo caso si vuole vuotare solo la cella B1 e quindi possiamo effettuare quest'operazione premendo il tasto **ENTER**. La sequenza completa da seguire nel premere i tasti e:

F3 v ENTER

Adesso che la cella B1 è stata vuotata, il testo completo alla cella A1 appare di nuovo.

CAPITOLO 4
FUNZIONI E FORMULE

FUNZIONI

Abacus possiede diverse *funzioni* predefinite che vengono utilizzate per eseguire lo sviluppo di determinati calcoli sul contenuto di una o più celle. Una funzione prende un numero di valori d'immissione, conosciuti col nome di *argomenti*, e da essi calcola specifici risultati. Il risultato è il valore che la funzione *riporta*.

In Abacus l'argomento deve essere scritto in parentesi dopo il nome della funzione e, se c'è più di un argomento, questi devono essere separati ciascuno da una virgola. La maggior parte delle funzioni fornite, riportano un valore numerico, per esempio la funzione **somma** ( ). Questa accetta come argomento un riferimento di serie e riporta un valore numerico uguale alla somma dei valori numerici contenuti in tutte le celle della serie.

Alcune funzioni, come per esempio **mese**( ), riportano un valore di testo. (**mese(1)**, per esempio, riporta il testo "gennaio"). Alcune funzioni non richiedono alcun argomento, ma è sempre necessario includere le parentesi. Per esempio la funzione **pi** riporta il valore numerico della costante matematica "pigreco" (3.14 circa).

Due funzioni particolarmente utili sono **col()** e **riga()**. Queste riportano il numero della colonna (o riga) che s'incrocia con la cella che contiene la funzione. Sono molto usate negli esempi del prossimo capitolo.

Per esempio, **col**( ) riporta un valore di 1 dalla colonna A, 2 dalla colonna B, e via di seguito. La funzione **riga**( ) riporta semplicemente il numero della riga.

Come esempio, possiamo usare le due funzioni **mese**( ) e **col**( ) per denominare le colonne del reticolo. L'obiettivo sarà di scrivere i titoli gennaio, febbraio, e così via in cima alle colonne da B ad M. Noi usiamo la funzione **col( ) per fornire il numero che mese( )** richiede per argomento, in modo che riporti un valore diverso per ogni colonna. Batti e scrivi:

```
riga = mese (col())
```

e premi il tasto **ENTER**. Seleziona la serie da B ad M quando Abacus ti chiede di definire la colonna iniziale e la colonna finale. Vedrai che il risultato non quello che volevamo poichè, sebbene i nomi iniziano alla colonna B, il primo nome è febbraio e non gennaio. Questo accade perchè nella colonna B, **col**( ) riporta un valore di 2 e **mese(2)** nel testo corrisponde a "Febbraio". Non ci rimane che modificare l'istruzione in modo che 1 venga sottratto dal valore riportato da **col**( ), prima che sia calcolato il mese. Scrivi:

```
riga = mese(col()-1)
```

(Non dimenticare di premere il tasto **ENTER** alla fine dell'immissione). Seleziona il campo di serie dalla colonna B alla colonna M, come prima.

FORMULE

Una formula viene generalmente usata per riferirsi al contenuto di una o più celle nel reticolo. L'idea delle formule è molto importante nell'usare Abacus, perchè ti permette di descrivere anche i calcoli più complicati in modo semplice.

Scrivi una formula in una cella allo stesso modo come scrivi numeri, cioè spostando il cursore alla cella, con i tasti e premendo **ENTER**. Abacus presume che tutto ciò che non riconosce come numero (che inizia con una cifra) o come valore di testo (che inizia con le virgolette) sia una formula.

Sposta il cursore alla cella B3 e scrivi il numero 100, sposta il cursore alla cella C3 e scrivi 200. Adesso sposta il cursore alla cella D3 e scrivi la seguente formula

```
B3 + C3
```

Quando premi **ENTER** vedi che succedono due cose. Prima, il valore 300 appare nella cella D3; il risultato della formula stato è calcolato sommando il contenuto della cella B3 + C3 e il totale è stato scritto nella cella D3. Inoltre vedi che alla riga informativa di stato, in fondo allo schermo, si vede la formula usata per calcolare il valore in questa cella. Una cella che contiene una formula ti mostra sempre il risultato del calcolo. Se metti il cursore su quella cella, allora Abacus mostra la formula stessa alla riga informativa di stato in fondo allo schermo. Il resto degli esempi relativi all' uso delle formule utilizzano la possibilità della nomeclatura e gli identinficatori di serie di **riga** e **col**. Essi offrono metodi per scrivere informazioni nel reticolo molto più efficaci dell'uso diretto dei riferimenti di

lettera e numero della cella. Nota che tutte le formule numeriche, che non hanno riferimenti di cella, non vengono memorizzate come formule. In tal caso Abacus calcola il relativo valore e ne memorizza il risultato come semplice numero. Per esempio, 37 + 100/20 viene memorizzato come valore 42, e non come la formula originale.

UN SEMPLICE ESEMPIO DI CASH FLOW

```
   |     A     |    B     |    C     |    D     |    E
 1 |           Gennaio    Febbraio   Marzo      Aprile
 2 | Vendite      1000.00    1050.00    1102.50    1157.63
 3 | Costi         722.00     749.50     778.38     808.69
 4 | Profitto      278.00     300.50     324.13     348.93
 5 |
```

Figura 4.1 Semplice esempio di analisi di cash flow.

Se hai qualcosa nel reticolo devi cancellarlo usando il comando **Zero**. Poi inizia quest'esempio con i mesi dell'anno visualizzati nel reticolo, dalla colonna B1 alla colonna M1. Adesso sposta il cursore alla cella A2, scrivi il testo "vendite e poi scrivi il valore 1000 nella cella B2. Adesso sposta il cursore alla cella C2 e batti la formula:

```
riga=vendite.gennaio*1.05
```

Accetta la serie suggerita da Abacus (Colonna C a colonna M) premendo **ENTER** due volte. Nota che Abacus già sa che la fine della riga è alla colonna M perchè questa è la cella dove finisce la riga precedente. Quando premi **ENTER** per la seconda volta, vedrai apparire tutta una serie di valori nella riga 2, dalla colonna C in avanti, e la formula B2*1.05 appare nella riga informativa di stato in fondo allo schermo.

Se sposti il cursore lungo la riga due, vedrai che la formula per ogni cella è leggermente diversa. In ogni caso la formula prende il contenuto e lo moltiplica per 1.05 per ottenere il valore da mettere nella cella in uso. Per esempio, la formula nella cella E2 si riferisce alla cella D2, e la formula nella cella H2 si riferisce alla cella G2, e così via.

In Abacus tutte le formule operano in questo modo, a meno che non specifichi diversamente. Ogni formula si ricorda la relativa posizione di tutte le celle a cui si riferisce. Quando questa formula viene usata in più di una cella, i riferimenti vengono modificati per mantenere un *rifcella relativo*.

Potrebbe essere di aiuto far presente che il valore iniziale di 1000 scritto nella cella B2 è necessario per due scopi; per assicurarsi che il nome "vendite" venga ricosciuto come rifriga (riferimento di una riga) ed anche per specificare il primo valore da essere usato dalla formula.

Adesso sposta il cursore alla cella A3 e scrivi il testo "Costi. Senza muovere il cursore, batti la seguente formula:

```
costi = vendite * 0.55 + 172
```

Questa formula calcola il costo da due componenti. Essi possono essere considerati come costi d'esercizio (55% delle vendite) e costi fissi per un totale di $172.00.

Usa il punto iniziale e finale suggeriti, dalla colonna B alla colonna M. Dal momento che il contenuto della riga è definito in termini di rifriga "Vendite", il nome "Costi", viene anche preso come rifriga, con gli stessi limiti di serie di "Vendite".

Ancora una volta dovresti spostare il cursore lungo tutta la riga, esaminando le diverse formule mostrate in fondo allo schermo, in modo da comprendere come sono stati calcolati i risultati.

Infine, scrivi il testo "Profitto" nella cella A4 e scrivi la seguente formula:

```
profitto = vendite - costi
```

con la stessa serie precedente (cioè da colonna B a colonna M). Abacus esegue il resto del lavoro riportando un semplice ma, completo esempio. Se adesso cambi la visualizzazione a formato monetario con il comando:

```
F3 Notazioni,piena,monetaria,segno meno,campo B2:M4
```

troverai che le prime colonne sono visualizzate come alla Figura 4.1.

CALCOLO AUTOMATICO

Quando hai scritto l'applicazione del semplice cash flow descritta nella sezione precedente, prova a cambiare i numeri nella cella B2 (Vendite.gennaio).

Sposta il cursore a questa cella – il modo più facile è di premere tasto **F5** e poi scrivi il rifcella, o B2 o (vend.genn) seguito da **ENTER**. A questo punto batti un numero qualsiasi. Quando premi **ENTER** vedrai che tutti i numeri nel reticolo cambiano!

Tutte le formule nelle celle del reticolo vengono calcolate automaticamente, ogni volta che effettui un immissione in una cella. Dal momento che tutte le formule in quest'esempio si riferiscono, direttamente o indirettamente, al valore mantenuto nella cella B2, tutti i valori di esse cambiano quando alteri il contenuto di questa cella. (Ricordati che abbiamo presunto che le vendite aumenterebbero del 5% al mese, basato sulle cifre di gennaio).

Il calcolo automatico può essere disattivato col comando **Quadro**. Questo è utile, per esempio, quando hai molte formule complicate nel reticolo e non vuoi attendere per un ricalcolo ogni volta che cambi un singolo valore.

Seleziona il comando **Quadro** premendo **F3** seguito dal tasto **Q**. La visualizzazione cambia mostrando una lista di opzioni, come illustrato alla Figura 4.2. Puoi selezionare qualsiasi opzione di quelle mostrate battendo la prima lettera. Seleziona il calcolo automatico premendo **C** e la funzione del calcolo automatico cambia automaticamente. Abbandona il comando premendo **ENTER**.

AIUTO F1	QUADRO permette di impostare le opzioni elencate. Premi la prima lettera	COMANDI F3
MESSAGGI F2	per scegliere l'opzione Alla fine premi ↵	USCITA ESC

```
CALCOLO AUTOMATICO all'immissione ..... si
VUOTA se zero ..... no
ORDINE di calcolo ..... RIGA
MONITOR 80,64,40 colonne (8,6,4) ..... 80
AVANZAMENTO modulo tra le pagine ..... si
INTERLINEE sulla stampante ..... 0
RIGHE stampate per pagina ..... 64
SIMBOLO valutario (p.es. £,$) ..... $
LARGHEZZA - carta della stampante ..... 80
```

Figura 4.2 Il comando Quadro.

Se adesso cambi il contenuto della cella **B2** vedrai che nessun altro cambiamento avviene nel contenuto delle altre celle.

Puoi anche forzare un ricalcolo di tutte le formule nel reticolo in qualsiasi momento usando il comando **Xecute**. Mentre hai il calcolo automatico disattivato, prova ad usare questo comando. Assicurati che il menu dei comandi sia visualizzato nel quadro di comando (premi **F3**) e poi premi il tasto **X**. I valori nelle celle del reticolo vengono ricalcolati.

Prima di proseguire devi attivare il calcolo automatico usando il comando **Quadro** di nuovo. Seleziona l'opzione calcolo automatico premendo il tasto **C**, come prima, e abbandona il comando premendo **ENTER**.

CAPITOLO 5
GLI ESEMPI

Le seguenti sezioni illustrano l'uso di Abacus mediante lo sviluppo di un numero di esempi. Oltre a spiegare il modo in cui diverse caratteristiche eseguono la gestione dello sviluppo, gli esempi sono stati scelti per illustrare alcune delle applicazioni realizzabili con Abacus. Il miglior modo per imparare tutto su Abacus è di usarlo. Gli esempi sono stati scritti proprio con questo concetto in mente.

Ti si raccomanda di sviluppare tutti gli esempi da te stesso, scrivendoli progressivamente. Ogni esempio possiede informazioni in più, oltre ad offrire possibiltà di più pratica con gli argomenti trattati nei precedenti esempi. Potresti anche modificare o migliorare gli esempi; questo ti dovrebbe dare altre idee su come sviluppare nuove applicazioni per conto proprio.

In tutti gli esempi, testo, numeri e formule sono illustrate esattamente nel modo in cui le devi scrivere. Se si richiede una serie di cella, verrà data in parentesi alla fine della riga. In molti casi, la serie che richiedi sarà quella suggerita da Abacus, che tu sceglierai battendo il tasto **ENTER**. In altri casi devi assegnare la serie. Se devi spostare il cursore in una determinata cella, il rifcella viene illustrato in parentesi quadre all'inizio della riga – non battere questo riferimento. Per esempio, la riga:

```
[A4] riga=mese(col()-1)        {colonne da B a M}
```

deve essere interpretata come:

> sposta il cursore alla cella A4 e poi scrivi
> `riga=mese(col()-1)`
>
> Se necessario, modifica la serie suggerita da ABACUS, che va dalla colonna B alla colonna M. Dove devi scrivere un determinato riferimento di serie, es. b3:e15, verrà data in quella forma.

Quando i comandi vengono dati per intero, essi sono illustrati esattamente come appariranno sullo schermo. Ricordati che è necessario premere solo la prima lettera di ogni opzione – il resto viene fatto da Abacus. Se vuoi usare l'opzione di default non hai che da premere il tasto **ENTER**.

Per ogni esempio, si presume che stia iniziando con una nuova tabella. Se necessario cancella la tabella con l'uso del comando **Zero**, prima di iniziare a scrivere è esempio successivo.

SVILUPPO "CASH FLOW"

Questa è una versione più completa del cash flow semplificato del capitolo 4. Quando hai finito il reticolo dovrebbe apparire come all'illustrazione alla Figura 5.1.

Le prime due immissioni di cella riportano un titolo sottolineato nel reticolo.

```
[C1] "CASH FLOW"
[C2] ripro("=",lun(c1))
```

La seconda immissione utilizza la funzione **ripro()**, che richiede due argomenti. Il primo è un testo, oppure un riferimento ad una cella che contiene un valore di testo e il secondo è numerico. La funzione riporta tanti numeri di ripetizioni del primo carattere del testo. In questo caso sottolinea il titolo con il segno " = ", per tutta la lunghezza del titolo. Se decidi di cambiare il titolo non c'è bisogno di cambiare la formula alla cella C2, dal momento che utilizza la funzione **lun( )** per leggere la lunghezza del testo nella cella C1.

```
[A4] riga=mese(col()-1)          {colonne B ad M}
[A5] riga=ripro("=",larg()+1)    {colonne A ad M}
```

Queste immissioni di riga producono le intitolazioni dei mesi, e una linea per tutta la zona usata del reticolo. La funzione **larghezza( )** fornisce la larghezza, in spazi di carattere, per ogni colonna. Può quindi essere usata per scrivere linee per tutto il reticolo con colonne di diversa larghezza. C'è un altro spazio di carattere che separa ogni colonna del reticolo, questa è la ragione per cui si richiede l'extra +1.

```
[A6] "VENDITE
[B6] 4000
[C6] riga=vend.gen*1.02        {colonne C ad M}
```

Queste immissioni completano le cifre di vendita per tutto l'anno, con l' ipotesi che le vendite per gennaio erano 4000 e che aumentano del 2% al mese.

```
[A7] "COSTO DI VENDITE
cost=vend*0.5+750        {colonne B ad M}
```

	A	B	C	D	E
1			CASH FLOW		
2			=========		
3					
4		Gennaio	Febbraio	Marzo	Aprile
5					
6	Vendite	40000	40800	41616	42448.32
7	Costi di vendite	27500	27900	28308	28724.16
8					
9	Profitto Lordo	1250.00	1290.00	1330.80	1372.42
10					
11	Spese				
12	Paghe	700.00	700.00	700.00	700.00
13	Pubblicità	100.00	100.00	100.00	100.00
14	Fitto	200.00	200.00	200.00	200.00
15	Elettricità	50.00	50.00	50.00	50.00
16	Ammortamento	90.00	90.00	90.00	90.00
17					
18	TOTALE SPESE	1140.00	1140.00	1140.00	1140.00
19					
20	NETTO PROFITTO	110.00	150.00	190.80	232.42
21					

Figura 5.1. Il reticolo del cash flow completato (le prime 5 colonne)

(I costi sono ipotizzati la metà del prezzo di vendita più una cifra fissa di $750,00.)

```
[A8] riga=a5        {colonne A ad M}
[A9] "PROFITTO LORDO
pro=ven-cos         {colonne B ad M}
```

Questa formula immette un'altra linea nel reticolo e calcola le cifre per il profitto lordo.

```
[A11] "SPESE
[A12] "Paghe
riga=700        {colonne B ad M}
[A13] "Pubblicità
riga=100        {colonne B ad M}
[A14] "Affitto
riga=200        {colonne B ad M}
[A15] "Elettricità
riga=50         {colonne B ad M}
[A16] "Ammortamento
riga=90         {colonne B ad M}
```

Queste immissioni scrivono le cifre di spese, presumendo che esse rimangano costanti per tutto l'anno. Puoi, naturalmente, cambiare i titoli di spese e la relativa cifra come ti fa comodo. Puoi includere più o meno immissioni, purchè effettui le modifiche necessarie al rifcella nel resto dell'esempio. Vorresti ottenere diversi valori per ogni mese, ma è più rapido stendere la tabella con valori fissi e modificarli in seguito.

```
[A17] riga = a5                 {colonne A ad M}
[A18] "TOTALE SPESE
[B18] riga=somma(col)           {righe da 12 a 16 colonne da B ad M}
[A19] riga=a5                   {colonne da A ad M}
```

Adesso hai il totale delle spese mensili.

La funzione **Somma** totalizza il contenuto di tutte le celle numeriche nella serie specificata come argomento. Tutte le celle vuote, comprese quelle che contengono testo, vengono ignorate. La serie può essere data come riferimento di serie specifico – B12:B16 per esempio. In questo caso, comunque, ogni serie è solo una singola colonna, così dobbiamo usare la specifica "col". Tutto ciò che ti rimane da fare è di rispondere alla serie che ti chiede Abacus, e quindi basta che premi è **ENTER** se la serie suggerita è quella che vuoi.

Nota che questa formula usa l'identificatore di serie **riga** e **col** nei due modi diversi. Prima, **riga** viene usata per indicare che la formula deve essere scritta in diverse celle della riga in uso. Secondo, **col** viene usata per specificare la serie di celle su cui si deve effettuare la somma. In entrambi i casi è richiesta la conferma (o modifica) del punto iniziale e finale. In questo caso Abacus tratta prima la serie per la funzione **somma( )**.

```
[A20] "NETTO PROFITTO
net=pro-tot      {colonne B to M}
[A21] riga=ripro("=",larghezza()+1)      {colonna A ad M}
```

La tabella adesso è completa, con le cifre del profitto netto calcolato come la differenza tra il profitto lordo e il totale delle spese. Tutto ciò che ti rimane da fare è di aggiustare la visualizzazione della tabella, usando alcuni comandi. Ricordati di premere **F3** ogni volta che vuoi usare i comandi.

Innanzitutto cambiamo la larghezza di colonna A (Nota che il comando **Tracciato** dispone del proprio menù di opzioni.

```
Tracciato>Larghezza, 15 da A ad A
```

Poi cambiamo la giustificazione ed il formato della visualizzazione numerica per alcune celle:

```
Giustifica,Piena,Testo,Destra,Campo a4:m4
Giustifica,Piena,Testo,Destra,Campo a12:a16
Notazioni,Piena,Decimale, cifre decimali 2,Campo A1:M21
```

Abbiamo scelto di visualizzare i numeri in forma decimale, con due cifre decimali. Se preferisci visualizzare il simbolo monetario, devi sostituire l'ultimo comando con

```
Notazioni,Piena,Monetaria,segno meno,campo a1:m21
```

E' molto semplice modificare qualsiasi numero. Supponiamo di voler aumentare le cifre per la pubblicità per il mese di febbraio. Tutto ciò che devi fare è premere il tasto **F5** (va a cella) e scrivere il riferimento della cella (rifcella)

```
feb.pub
```

Il cursore si sposta a quella cella e così puoi scrivere il nuovo valore.

Ricordati che i numeri vengono calcolati con una formula che assume un aumento mensile del 2%. Se cambi una di queste celle ad un valore numerico, cancelli la formula in quella cella. Le formule nelle altre celle, comunque, rimangono invariate. Il valore nelle celle seguenti aumenta del 2% mensile, a partire dal nuovo valore.

LA TAVOLA PITAGORICA

Questo semplice esempio potrebbe essere utile ad un bambino che vuole imparare la tavola pitagorica. Ti permette di chiedere qualsiasi tavola e la fa visualizzare.

Quando hai scritto l'esempio, puoi usarlo forzando un ricalcolo del reticolo col comando **Xecute**, es. batti:

F3 x

Abacus quindi ti chiede di scrivere un numero e mostra la corrispondente tavola di moltiplicazione.

La tavola di moltiplicazione alla Figura 5.2 illustra un esempio della visualizzazione che produce.

Innanzi tutto scrivi il titolo dell'applicazione nel modo normale:

```
[B1] "TAVOLA DI MOLTIPLICAZIONE
[B2] ripro("="'lun(b1))
```

Le seguenti tre righe danno un titolo alla tavola.

```
[B3] "Per
[C3] chieden("Quale tavola di moltiplicazione vuoi")
[D3] "Moltiplicazione
```

Qui abbiamo usato la funzione **chieden** per chiedere un'immissione; ti permette di selezionare quale tavola vuoi, premendo il numero desiderato.

```
   |      A      |   B   |   C   |   D   |   E   |   F
 1 |                TAVOLA DI MOLTIPLICAZIONE
 2 |                ========================
 3 |                  Per 7 Moltiplicazione
 4 |                     1 *     7     =     7
 5 |                     2 *     7     =    14
 6 |                     3 *     7     =    21
 7 |                     4 *     7     =    28
 8 |                     5 *     7     =    35
 9 |                     6 *     7     =    42
10 |                     7 *     7     =    49
11 |                     8 *     7     =    56
12 |                     9 *     7     =    63
13 |                    10 *     7     =    70
14 |                    11 *     7     =    77
15 |                    12 *     7     =    84
```

Figura 5.2 Una tavola di moltiplicazione.

Questa funzione prende una stringa di testo come argomento e mostra il testo nella riga d'immissione, seguito da un punto interrogativo. Poi attende che scriva un numero, completando con **ENTER**. Il numero che scrivi, verrà visualizzato nella cella che contiene **chieden**.

Nota che **chieden** non attende per l'immissione durante un calcolo automatico normale del reticolo. Visualizza solo il messaggio e chiede l'immissione quando metti la formula inizialmente nella cella, oppure quando forzi un ricalcolo del reticolo, usando il comando **Xecute**. Una volta che hai immesso un valore nella cella verrà mantenuto, fino a quando non forzi un ricalcolo di nuovo con il comando **Xecute**.

Le rimanenti immissioni del reticolo usano la possibilità di riempire le colonne per produrre il corpo della tavola di moltiplicazione.

[B4] `col=str(riga()-3,2,0)+"*"` {righe da 4 a 15}

Questa la formula è più complicata dell'esempio. Viene usata per la visualizzazione del moltiplicatore in ogni riga della tavola. Il numero viene convertito ad una stringa di testo, in modo che si possa unificarlo con il simbolo della moltiplicazione e visualizzarli tutte e due in una singola cella.

La funzione **str** converte un numero nella stringa equivalente di cifre. Prende tre valori: il numero da convertire, un codice per il formato (0 = decimale, 1 = esponenziale, 2 = intero, 3 = generale) in cui il numero deve essere visualizzato, e il numero di cifre decimali da mostrare. In questo caso il numero viene convertito a formato intero.

In questo caso il valore viene ottenuto dall'espressione **riga()−3**, che è 1 nella riga quattro, 2 nella riga cinque, e così via di seguito, fino a 12 nella riga 15. Il valore seguente (2) seleziona la visualizzazione come numero intero. Il terzo numero generalmente specifica quante cifre decimali si devono usare. È sempre necessario fornire questo valore, ma viene ignorato nelle forme intere. Gli abbiamo dato il valore di zero (si poteva usare qualsiasi altro valore). Infine il risultato viene *concatenato* (il termine corretto per congiungere una stringa di testo ad un'altra) con la stringa `*`, in modo che il simbolo e il moltiplicatore della moltiplicazione vengano visualizzati in una singola cella.

[C4] `col=$c3` {righe 4 a 15}

La colonna C contiene copie del valore scritto in risposta alla domanda della funzione **chieden()**. Il rifcella è preceduto dal simbolo **$** in modo da renderlo uno *rifcella assoluto*. Quando hai scritto la formula, sposta il cursore in basso e in alto lungo la colonna della cella C e osservane il contenuto. Vedrai che tutte hanno il rifcella $C3. Il riferimento non è stato aggiustato in nessuna riga. Un rifcella assoluto si riferisce sempre ad una particolare cella, da qualsiasi posizione nel reticolo. Puoi rendere assoluto qualsiasi rifcella facendolo precedere dal simbolo **$**.

[D4] `col="="` {righe 4 a 15}
[E4] `col=$c3*(riga()-3)` {righe 4 a 15}

Queste due ultime righe non hanno bisogno di molte spiegazioni. Esse sono usate per riportare simboli uguali e la risposta per ogni riga della tavola. L'ultima formula moltiplica il valore dalla funzione **chieden( )** nella cella C3 (un'altro rifcella assoluto) per l'espressione **riga( )−3** che, come abbiamo già visto, dá un valore di 1 alla riga quattro, 2 alla riga cinque, fino a 12 alla riga quindici.

Adesso dobbiamo usare alcuni comandi per cambiare il formato della visualizzazione della tavola ad un formato più adatto. Usa i seguenti comandi

```
Giustifica,piena,testo,destra,campo b3:b15
Giustifica,piena,testo,destra,campo d4:d15
Giustifica,piena,numeri,centro,campo C3
Tracciato>larghezza, 5 da b ad b
Tracciato>larghezza, 3 da c ad c
Tracciato>larghezza, 2 da d ad d
Tracciato>larghezza, 4 da e ad e
```

Puoi forzare un ricalcolo nel reticolo con il comando **Xecute** . Il testo della funzione **chieden( )** appare alla riga d'immissione – devi scrivere un numero da 1 a 12.

QUADRATURA DEL LIBRETTO DEGLI ASSEGNI

Quest'esempio ti permette di controllare il tuo conto corrente. Scrivi i particolari degli assegni nello spazio disponibile. Alla fine del mese aggiungi tutti i particolari dello stipendio, ordini di pagamento permanenti ecc. Quindi ti verrà fornito un bilancio che puoi paragonare con l'estratto conto bancario.

Il risultato, con alcune cifre aggiunte, illustrato alla Figura 5.3.

```
  |        A          |        B          |        C          |        D
1 |  QUADRATURA DEL LIBRETTO DEGLI ASSEGNI BANCARI
2 |  ========================================
                                        ==
3 |
4 |                Mese                Gennaio
5 |
6 | Bilancio d'apertura                    2000
7 | Stipendio                           5273.50
8 | Redditi vari                              0
9 |
10|                          ACCREDITO                          7273.50
11|                                                             ======
12|
13| Ordini di pagamento                                          1300
14| Spese bancarie                                                  0
15|
16| Assegni          Data          Assegno Nr.                   Valore
17|                3/01/85               123456                    500
18|               10/01/85               123457                    500
19|               14/01/85               123458                  322.10
20|               17/01/85               123459                    500
21|               24/01/85               123460                    500
22|               31/01/85               123461                    500
23|                  ---                   ---                       ---
24|                  ---                   ---                       ---
25|                  ---                   ---                       ---
26|                  ---                   ---                       ---
27|
28|                          ADDEBITO                           4122.10
29|                                                             ======
30| Bilancio di chiusura                3151.40
31|                                     ======
```

Figura 5.3 Quadratura del libretto degli assegni bancari.

```
[B1] QUADRATURA DEL LIBRETTO DEGLI ASSEGNI BANCARI
[B2] ripro("=",lun(b1))

[C4] "Mese
[D4] chiedet("Immissione mese")
```

La funzione **chiedet( )** effettua la stessa esecuzione della funzione **chieden**, con la differenza che l'immissione chiesta è in questo caso, di testo anzichè di numero. Quando usi **Xecute**, Abacus mostra il messaggio alla riga d'immissione e attende che scrivi del testo. Devi scrivere il nome del mese a cui si riferisce il bilancio.

```
[A6] "Bilancio d'apertura
[A7] "Stipendio
[A8] "Redditi vari
[C6] chieden(a6+" per "+$d4)
```

La stringa del prompt per **chieden( )** è costruita dal testo immesso in altre celle, usando sia rifcelle relativi che quelli assoluti.

Adesso utilizziamo il comando **Eco** per copiare la formula dalla cella C6 nella cella C7 e C8. Anzichè battere il rifcella C7:C8, possiamo usare l'identificatore **col**.

```
Eco,cella c6,nella serie col da 6 ad 8

[B10] "ACCREDITO
[C10] somma(col)        {righe da 6 ad 8}
```

La cella C10 viene usata per contenere il totale di tutti gli accrediti per quel mese. Questa cella ha un nome; il riferimento "accredito.mese".

Il contenuto delle celle viene calcolato con l'uso della funzione **somma**, che abbiamo già visto nel primo esempio di questo capitolo. Questa funzione somma il contenuto numerico di tutte le celle del campo della serie specificata dall'argomento. Ricordati che ignora le celle vuote della serie oppure quelle che contengono testo.

In questo caso è stata di nuovo usata come somma(col), che specifica che le celle da sommare si trovano nella colonna in uso. Normalmente, Abacus ti chiede di specificare l'esatto campo della serie, suggerendo adatti valori basati su operazioni precedenti.

```
[C11] ripro("=",lun(str(accredito.mese,0,2)))
```

La cella C11 sottolinea il totale, con l'uso delle funzioni **ripro())** e **lun( )**) In questo caso, comunque noi non sappiamo in anticipo il numero di caratteri da sottolineare. Quindi dobbiamo convertire il numero ad una stringa di caratteri con l'uso della funzione **str( )**, presumendo che verrà visualizzata in formato decimale con due cifre decimali. La lunghezza di questa stringa dà il corretto numero di caratteri da sottolineare.

```
[A13] "Ordini di pagamento
[A14] "Spese bancarie
[D13] chieden(a13+" per "+$d4)
[D14] chieden(a14+" per "+$d4)
```

Queste ti permettono d'immettere gli addebiti mensili in risposta ai prompt, con l'uso di **chieden( )** allo stesso modo come descritto precedentemente.

```
[A16] "Assegni
[B16] "Data
[C16] "Assegno Nr.
[D16] "Valore
[B17] riga="---"        {colonne B ad D}
```

Queste celle impostano una zona del reticolo, che in seguito userai per scrivere i particolare degli assegni.

```
[B28] "ADDEBITO
[D28] somma(col)        {righe da 13 a 26}
```

Questo calcola il totale degli addebiti. Ricordati che **somma( )** effettua solo l'addizione dei valori numerici nelle celle della serie specificata. Le celle che contengono testo, e celle vuote, non sono incluse. Quindi, la somma ignora tutte le immissioni non utilizzate nella lista degli assegni, come pure i titoli della colonna D nel tabellone.

```
[A30] "Bilancio di chiusura
[C30] accredito.mese.-addebito.mese
```

Il calcolo del bilancio di chiusura completa le immissioni nel reticolo. Adesso devi usare i comandi per mettere in ordine la visualizzazione dell'applicazione.

Innanzitutto con l'uso del comando **Eco** riempiamo il resto della tabella degli assegni e completiamo la sottolineatura dei totali. Questo comando copia il contenuto di una singola cella in tutte le celle della serie. Il primo dei seguenti tre, per esempio, copia il contenuto della cella B18 e D26 rispettivamente.

```
Eco,Cella b17,nella serie b18:d26
Eco,Cella c11,nella serie d29:d29
Eco,Cella c11,nella serie c31:c31
```

Poi dobbiamo fissare il formato numerico a decimale con due cifre decimali, per tutta l'applicazione, con formato intero per il numero degli assegni.

```
Notazioni,piena,decimale,cifre decimali 2,campo a1:d3
Notazioni,piena,integrale,segno meno,campo c17:c2l1
```

Abbiamo già spiegato che celle vuote non esistono per quanto riguarda Abacus e quindi una variazione del formato ha solo effetto sulle celle non vuote. Possiamo riempire la tabella degli assegni con "---" prima di effettuare la variazione per essere sicuri che queste celle vengano cambiate a formato decimale. Un'altro metodo alternativo di cambiare il formato di visualizzazione.

Infine possiamo modificare la giustificazione del testo, compresa la sottolineatura per rendere l'aspetto finale esteticamente migliorato.

```
Giustifica,piena,testo,destra,campo b16:d26
Giustifica,piena,testo,destra,campo c11
Giustifica,piena,testo,destra,campo d29
Giustifica,piena,testo,destra,campo c31
```

La parte del reticolo usata è troppo larga per essere visualizzata sulla finestra per intera. Per visualizzare i risultati finali, con i valori immessi mediante le funzioni **chiedet( )** e **chieden( )**, puoi usare la possibilità della finestra divisa. Il comando **Finestra** effettua la divisione verticale, oppure orizzontale della finestra in due sezioni, con l'uso della posizione del cursore si determina il punto della divisione. Una divisione verticale è molto adatta a questo reticolo e la si può effettuare con lo spostamento del cursore al centro della finestra e poi usare il comando:

```
Dividi,Verticale,Separatamente
```

Puoi spostare il cursore da una finestra all'altra premendo il tasto **F4**. Per quest'esempio devi usare il cursore per regolare la finestra di sinistra per visualizzare la cella A1 all'angolo sinistro superiore, e la cella B15 all'angolo superiore destro della finestra.

SCOSTAMENTO QUADRATICO MEDIO

Quest'esempio calcola la media e lo scostamento quadratico di una serie di numeri. Utilizza il sistema di nomeclatura di Abacus per cui le formule usate non hanno bisogno di spiegazioni.

```
  |       A      |    B     |    C     |    D     |    E
1 |                SCOSTAMENTO QUADRATICO MEDIO
2 |                ============================
3 |
4 |                Valore     Deviazione Quadrato deviazione
5 |                   5.00      -4.50       20.25
6 |                   6.00      -3.50       12.25
7 |                   7.00      -2.50        6.25
8 |                   8.00      -1.50        2.25
9 |                   9.00      -0.50        0.25
10|                  10.00       0.50        0.25
11|                  11.00       1.50        2.25
12|                  12.00       2.50        6.25
13|                  13.00       3.50       12.25
14|                  14.00       4.50       20.25
15|
16|  Media           9.50    Varianza        8.25
17|                          Scost.          2.87
18|                          ------------------
```

Figura 5.4 Calcolo scostamento quadratico medio

Inoltre esso utilizza un tracciato che richiede calcolo in ordine di colonna, anzichè in ordine normale di riga.

Generalmente una formula dovrebbe solo riferirsi alle celle che sono nella zona al di sopra e alla sinistra della cella che contiene la formula, compreso la riga e la colonna che contengono la formula.

Se non segui questa regola, come in quest'esempio, è probabile che il risultato sia scorretto. Nella maggior parte dei casi puoi ottenere un risultato corretto forzando un ricalcolo del reticolo con il comando **Xecute**, oppure, come in questo caso, calcolando il reticolo in ordine di colonna.

```
[B1] "SCOSTAMENTO QUADRATICO MEDIO
[B2] ripro("=",lun(b1))
[B4] "Valore
[C4] "Deviazione
[D4] "Quadrato deviazione
[B5] col=riga()        {da riga 5 ad 14}
```

Questa ultima formula inserisce una serie di numeri fittizi nelle celle della colonna B, per provare l'applicazione. Quando le immissioni del reticolo sono complete, le puoi sostituire con altri valori. La tabella descritta in quest'esempio può contenere solo dieci valori – puoi cambiare per trattare con più valori, se lo desideri.

```
[A16] "Media
[B16] media(valore)        {riga da 5 a 14}

deviazione=valore-$media.valore      {riga 5 a 14}
quadrato=deviazione*deviazione       {riga 5 a 14}

[C16] "Varianza
[D16] media(quadrato)      {riga da 5 a 14}
```

Queste formule dimostrano che la varianza di una serie di numeri è definita come la media dei quadrati delle deviazioni dalla media.

```
[C17] "Scost."scost.
[D17] quad(varianza)       {da colonna D a D}
```

e che lo scostamento medio può essere calcolato come la radice quadrata della varianza.

```
[D18] ripro("=",lun(str(quad.var,3,0)))
```

I numeri, in quest'esempio, sono stati lasciati in formato generale, in modo che possa trattare qualsiasi serie di valori. La sottolineatura utilizza la lunghezza della stringa di testo corrispondente al numero della cella soprastante (con il rifcella "quad.var"), espresso in formato generale.

Puoi migliorare la visualizzazione della tabella cambiando la giustificazione del testo a sinistra della serie B4:D4, e usando giustificazione a sinistra dei numeri nella serie B4:d18. Se provi ad usare quest'esempio scrivendo valori diversi nelle celle alla colonna B, troverai che non ti darà risposte corrette. La ragione è che il calcolo del reticolo viene effettuato riga per riga, da quella superiore a quella inferiore. Qualsiasi modifica apporti quindi verrà sviluppata sulla base di un valore medio scorretto (dal momento che la nuova media non verrà calcolata se non dopo gli scarti dalla media). La soluzione è nel fare i calcoli del reticolo per colonna, da destra a sinistra. Questo lo puoi fare con il comando **Quadro**.

Usa l'opzione **"O"** per cambiare l'ordine di calcolo a colonna e lascia il comando premendo il tasto **ENTER**, come indicato al quadro di controllo. La prossima volta che cambi il valore nella colonna B, il calcolo sarà corretto, dal momento che la nuova media adesso viene calcolata prima degli scostamenti. Sebbene la possibilità di cambiare l'ordine di calcolo sia molto utile, non devi abituarti ad usarla troppo spesso – il calcolo per colonna più lento del calcolo per riga.

Se salvi un reticolo su un file del Microdrive, le impostazioni in uso delle opzioni del **Quadro** vengono salvate con esso e vengono usate ogni volta che richiami il file.

UN BILANCIO FAMILIARE

Quest'esempio ti aiuterà a programmare le spese familiari per tutto l'anno. Le spese preventivate possono essere scritte sotto uno svariato numero di titoli per ogni trimestre. Di consequenza, ti verranno forniti totali trimestrali, le spese per l'intero anno e le spese medie mensili. Non scrivere alcun numero nella tabella, fino a quando non riceverai istruzioni di effettuare quest'operazione. Questo ti permette di cambiare il formato numerico di visualizzazione, con il comando di **Notazioni** e la scelta dell'opzione di Default.

```
    A |     B      |C|   D    |E|   F    |G|   H    |I|   J    |K
  1 |                  BILANCIO DOMESTICO
  2 |                  ==================
  3 |
  4 |  ================================================
  5 |  !                !          SPESE PREVENTIVATE          !
  6 |  ! Voce           ! gen-mar  ! apr-giu  !lugl-set  ! ott-dic !
  7 |  ================================================
  8 |  !Affitto/mutuo   ! 4000.00  ! 4000.00  ! 4000.00  ! 4000.00 !
  9 |  !Tasse comunali  !          ! 4500.00  !          !          !
 10 |  !Gas             ! 1500.00  !  800.00  !  600.00  ! 1500.00 !
 11 |  !Elettricità     !  400.00  !  300.00  !  300.00  !  400.00 !
 12 |  !Acqua           !          !  350.00  !          !  350.00 !
 13 |  !Telefono        ! 1500.00  ! 1500.00  ! 1500.00  ! 1500.00 !
 14 |  !Assicurazione   !          !          !          !          !
 15 |  !Vestiari        !          !          !          !          !
 16 |  !Acquisti rateali!          !          !          !          !
 17 |  !Bollo macchina  !          !          !          !          !
 18 |  !Benzina         !          !          !          !          !
 19 |  !Abbonamento TV  !          !          !          !          !
 20 |  !Risparmi        !          !          !          !          !
 21 |  ================================================
 22 |   Tot. trimestre  !  7400    !  11450   !  6400    !  7750    !
 23 |                   ========================================
 24 |
 25 |                      Annue     Mensili
 26 |
 27 |            Uscite  33000.00   2750.00
 28 |                    ==========  =========
```

Figura 5.5 Esempio di bilancio familiare

```
[D1] "BILANCIO FAMILIARE
[D2] ripro("=",lun(d1))
```

Adesso possiamo impostare la struttura della tabella con le divisioni per linee.

```
[A4] ripro ("-",larghezza()+1)      {colonne A ad K}
[A5] col="!"      {righe da 5 a 20}
```

I seguenti comandi completano la struttura della tabella.

```
Tracciato>Larghezza, 16 da b ad b
Tracciato>Larghezza, 8 da d ad j
Tracciato>Larghezza, 1 da a ad a
Tracciato>Larghezza, 1 da c ad c
Tracciato>Larghezza, 1 da e ad e
Tracciato>Larghezza, 1 da g ad g
Tracciato>Larghezza, 1 da i ad i
Tracciato>Larghezza, 1 da k ad k

Eco,cella a5,nella serie c5:c22
Eco,cella a5,nella serie e6:e22
Eco,cella a5,nella serie g6:g22
Eco,cella a5,nella serie i6:i22
Eco,cella a5,nella serie k5:k22
Eco,cella a4,nella serie b7:j7
Eco,cella a4,nella serie b21:k21
Eco,cella a4,nella serie c23:k23

[A7] "!-
[F5] "SPESE PREVENTIVATE
[B6] "Voce
[D6] "gen-mar.
[F6] "apr.giu.
[H6] "lugl.set.
[J6] "ott.nov.
```

```
[B8] "Affitto/mutuo
[B9] "Tasse comunali
[B10] "Gas
[B11] "Elettricità
[B12] "Acqua
[B13] "Telefono
[B14] "Assicurazione
[B15] "Vestiari
[B16] "Acquisti rateali
[B17] "Bollo macchina
[B18] "Benzina
[B19] "Abbonamento TV
[B20] "Risparmi

[B22] "Tot. trimestre

[D22] somma(col)      {riga da 8 a 20}
[F22] somma(col)      {riga da 8 a 20}
[H22] somma(col)      {riga da 8 a 20}
[J22] somma(col)      {riga da 8 a 20}

[D25] "Annue
[F25] "Mensili
[B27] "Uscite

[D27] somma (d22:j22)
[F27] uscite.annue/12
[D28] ripro("=",lun(str(usc.ann,0,2))+1)
[F28] d28
```

Nota che la sottolineatura delle due cifre finali presume un formato monetario. La lunghezza della sottolineatura per un formato monetario, è con due cifre decimali, più una (per il simbolo monetario).

Dovresti anche usare alcuni comandi, per la giustificazione del testo a destra nella serie B22:B27 (totale uscite trimestrali) e per la giustificazione a sinistra dei numeri per la serie di celle che contengono le uscite mensili ed annuali. Devi anche modificare il formato di visualizzazione numerica. Dato che la maggior parte delle celle sono ancora vuote, la semplice modifica del formato non avrà alcun effetto. Devi cambiare il formato del Default delle celle per rendere l'esecuzione permanente.

Il seguente comando cambierà il default di visualizzazione di notazioni monetarie su tutta l'applicazione del bilancio.

```
Notazioni,Vuota,Monetaria,segno meno
```

L'illustrazione alla Figura 5.5 usa un formato decimale, con due cifre decimali, ad eccezione delle uscite annue e mensili, che sono in formato monetario. I comandi da usare sono:

```
Notazioni,Vuota,Decimale, cifre decimali 2
Notazioni,piena,Monetaria,segno meno,campo d27:f27
```

Con questo ultimo comando puoi usare l'opzione di celle piene dal momento che le celle in oggetto sono già in esistenza.

Puoi scrivere i valori in questa tabella spostando il cursore alla cella richiesta e scrivendo il numero. Il modo più facile per spostare il cursore è di premere il tasto **F5** (Va a Cella) e poi usare un rifcella, come per esempio

```
apr.gas
```

ISTOGRAMMA A GRADAZIONE AUTOMATICHE

Il grafico rappresenta dodici valori, denominati per mese. I valori vengono letti da dodici celle in cima al grafico. La scala verticale viene aggiustata automaticamente per assicurare che tutti i valori siano in forma adatta per la visualizzazione. Esso è adatto solo per mostrare valori positivi.

Prima di tutto devi registrare la larghezza delle colonne, la colonna A a cinque, la colonna B ad uno e dalla colonna C alla colonna N a tre, con l'uso dell'opzione Larghezza del comando **Tracciato**.

```
[C2] riga=0      {colonna da C ad N}
```

```
   | A    |B|  C  |  D  |  E  |  F  |  G  |  H  |  I  |  J  |  K  |  L  |  M  |  N
 1 |
 2 |            3     4     3     2     3     4     3     4     1     2     3     2
 3 |                                GRAFICO A BARRE GRADUATO
 4 |    5   !
 5 |  4.5   !
 6 |    4   !       ***                       ***         ***
 7 |  3.5   !       ***                       ***         ***
 8 |    3   !  ***  ***  ***        ***  ***  ***  ***                    ***
 9 |  2.5   !  ***  ***  ***        ***  ***  ***  ***                    ***
10 |    2   !  ***  ***  ***  ***  ***  ***  ***  ***        ***  ***  ***  ***
11 |  1.5   !  ***  ***  ***  ***  ***  ***  ***  ***        ***  ***  ***  ***
12 |    1   !  ***  ***  ***  ***  ***  ***  ***  ***  ***  ***  ***  ***  ***
13 |  0.5   !  ***  ***  ***  ***  ***  ***  ***  ***  ***  ***  ***  ***  ***
14 |    0   !  ----------------------------------------------------------------
15 |             Gen  Feb  Mar  Apr  Mag  Giu  Lug  Ago  Set  Ott  Nov  Dic
```

Figura 5.6 Un istogramma graduale

La riga due contiene adesso i valori che devono essere visualizzati – che inizialmente vengono mostrati con caratteri fittizi.

```
[F3] "ISTOGRAMMA GRADUATO

[P2] int(max(c2:n2)/5+1)*5
[Q2] int(min(c2:n2)/5)*5
```

Le celle P2 e Q2 contengono il valore massimo e minimo per la scala di graduazione verticale del grafico. Queste celle sono state scelte in modo che non siano viste nella visualizzazione finale del grafico. I valori iniziali sono cinque e zero rispettivamente.

La funzione **max()**riporta il valore numerico massimo nella serie delle celle specificata dall'argomento. Allo stesso modo, la funzione **min()** riporta il valore numerico minimo nella serie.

Esaminiamo la formula nella cella Q2. La funzione **min( )** trova il più piccolo, o il minimo valore nella serie delle celle specificata e poi lo divide per cinque. La funzione **int()** quindi elimina la parte frazionale del risultato della divisione. Se, per esempio, il valore minimo è 13, diviso per 5 abbiamo un valore di 2,6, e **int(2,6)** è 2. Quando questo viene moltiplicato per 5 abbiamo il risultato finale di 10, che è il più grande multiplo di 5 che è meno del valore minimo.

La formula nella cella P2 è analoga, con l'eccezione che trova il valore massimo nella serie a cui aggiunge 1, prima che il numero venga infine moltiplicato per 5. Se, per esempio, noi presumiamo che il valore massimo 21, tu puoi verificare che la formula riporti un valore di 25 – il multiplo più piccolo di 5 che è superiore al massimo valore.

I due valori in queste celle quindi metteranno sempre in parentesi i valori nelle celle da C2 a N2. La differenza è sempre un multiplo di cinque.

La formula seguente mostra la scala graduata del grafico nella colonna A.

```
[A4] col=$q2+(14-riga())*($p2-$q2)/10      {riga 4 ad 14}
```

L'intervallo è tra i numeri successivi nella scala graduata **(P2-Q2)/10**. Nota che abbiamo fatto in modo che la differenza tra il contenuto di P2 e Q2 sia un multiplo di 5, in modo che quest'intervallo abbia sempre un semplice valore.

Quest'intervallo viene moltiplicato per un numero **(14–riga()** che inizia a zero alla riga quattordici. Il risultato viene aggiunto al valore più piccolo, dalla cella Q2, per riportare il numero per ogni cella.

Il risultato è che il valore nella cella Q2 viene visualizzato in A14, il valore da P2 viene visualizzato in A4 e le celle tra A4 e A14 contengono una serie di valori equidistanziati tra i due limiti.

```
[B4] col="!"      {iga 4 ad 14}
[B14] riga=ripro("-",larghezza()+1)      {colonne B ad N}
[C15] riga=mese(col()-2) (ad 3)      {colonne C ad N}
```

Queste tracciano gli assi per il grafico e aggiungono i nomi degli assi orizzontali, usando i mesi dell'anno. Nota che abbiamo usato l'operatore di abbreviazione di stringa, analogo a quello di SuperBASIC, per visualizzare solo tre caratteri di ogni mese.

```
[C4] se(indice(1,riga()) > indice(col(),2) ,"","***")
```

Questa è la formula che effettua tutte le operazioni che producono le barre stesse. Essa deve essere copiata in ogni cella visualizzata:

```
Eco,Cella c4,nella serie c4:n13
```

La formula stessa ha bisogno di qualche spiegazione. Essa utilizza la funzione **se()** per decidere se mostrare parte della barra. La funzione **se()** prende tre argomenti. Il primo è un'espressione che deve riportare un risultato numerico. Se questo è non-zero la cella mostra il secondo argomento che può essere testo o numerico. Se, comunque, il risultato è zero il terzo argomento viene visualizzato nella cella. Anche questo può essere testo o numerico.

In ogni cella la formula paragona il numero in colonna uno di quella riga (il valore che etichetta l'asse verticale) con il numero in riga due di quella colonna (il valore che si visualizzerà nel grafico). Se l'etichettamento dell'asse è superiore al valore di visualizzazione, la condizione è vera (riportata a 1) e non si visualizza niente. Se l'etichettamento dell'asse è inferiore o uguale al valore di visualizzazione, la condizione risulta in un valore di zero, e tre asterischi vengono visualizzati nella cella. Il risultato è che una barra viene graficamente espressa alla giusta altezza in ogni colonna.

Dal momento che una singola formula viene usata per tutte le celle visualizzate, il riferimento di cella non può essere nè assoluto nè relativo. Il riferimento al valore di visualizzazione deve cambiare mentre ci spostiamo da colonna a colonna (es. deve essere relativo lungo una colonna), ma deve sempre riferirsi alla riga due mentre ci spostiamo in basso, da riga a riga. Abbiamo bisogno di una forma di rifcella che sia relativa in riguardo a colonne, ma assoluta in riguardo a righe.

Fortunatamente la funzione **indice()** può essere utilizzata per produrre quest'effetto. Essa richiede due parametri, un numero di una colonna e poi un numero di riga, riportando il contenuto della cella specificata. Con questo possiamo costruire qualsiasi combinazione di riferimenti assoluti e relativi. Per esempio:

Funzione	Rifcol	Rifriga
indice(5,5)	assoluto	assoluto
indice(col( ),5)	relativo	assoluto
indice(5,riga( ))	assoluto	relativo
indice(col(),riga())	relativo	relativo

La funzione **indice(col(),2)** riporta il contenuto delle celle nella riga due della colonna in uso, e la funzione **indice(1,riga())** riporta il contenuto della cella nella colonna uno (A) della riga in uso.

Adesso sostituisci i caratteri fittizi nelle celle da C2 a N2 per vedere l'effetto che ne risulta sulla visualizzazione.

MUTUO IPOTECARIO CALCOLO DELLE RATE

Quest'esempio ti permette di calcolare le rate mensili di un mutuo ipotecario. Devi scrivere il valore del prestito, il tasso d'interesse il periodo del prestito, il mese in cui devi fare il primo pagamento. I pagamenti richiesti vengono calcolati e visualizzati, insieme ad una tabella di pagamenti, per tutto il periodo del prestito ipotecario. Questa tabella ti mostra il bilancio ancora scoperto al principio di ogni mese fino al saldo completo del prestito.

Molti calcoli nel reticolo utilizzano i valori immessi mediante la funzione **chieden( )**.

Calcolo per ammortamento mutuo

In questa sezione produrremo la parte del reticolo che accetta la scrittura delle immissioni e calcola i pagamenti mensili del prestito. Quando hai scritto le formule e aggiunto alcune cifre in risposta alla funzione **chieden( )**, la visualizzazione dovrebbe apparire come alla Figura 5.7.

```
[C1] "IPOTECA - CALCOLO DELLE RATE
[C2] ripro("=",lun(c1))
[B4] "Prestito
[C4] chieden ("Somma prestito")
```

	A	B	C	D	E
1			IPOTECA - CALCOLO DEI RIPAGAMENTI		
2			==========		
3					
4		Prestito	25,000.00		
5		Interesse%	14.00%		Mese
6		Periodo anni	25	Inizio	4
7					(Aprile)
8				RIPAGAMENTI	
9				----------	
10			Annui	3637.46	
11			Mensili	303.12	
12				----------	

Figura 5.7 Calcolo dell'ammortamento mutuo

Le tre immissioni seguenti chiedono di immettere il tasso d'interesse. L'immissione originale è in una cella (H4) molto distanziata dalla parte che è visualizzata nel reticolo, quindi normalmente non la vedi. Scrivi i valori della percentuale, es. scrivi 12 per significare 12%. Il valore richiesto dal resto della formula un valore decimale (es. 12% deve essere convertito a 0,12) e questo viene calcolato dal valore d'immissione dalla formula nella cella C5.

```
[H4] chieden("Interesse percento")
[B5] "Interesse %
[C5] h4/100

[B6] "Periodo anni
[C6] chieden("Periodo anni prestito [massimo 35]")

[E5] "Mese
[D6] "Inizio

[E6] chieden ("Mese primo pagamento [gen=1,feb=2, ecc.]")
[E7] "(' + mese(e6) + ")"
```

In quest'ultima formula abbiamo incluso il testo letterale con singole virgolette. Se il primo carattere fosse stato doppia virgoletta, Abacus avrebbe interpretato i caratteri seguenti come immissione di testo, anzichè immisione di formula.

```
[D8] "RIPAGAMENTI
[D9] ripro("-",lun(d8))
[C10] "Annui
[D10] ipo.pre*ipo.int/(1-(1+ipo.int)^(-ipo.per))
```

Questa formula che calcola le rate annue, presume che gli interessi vengono calcolati su base annuale e aggiunti al prestito prima che i dodici pagamenti mensili vengano fatti.

```
[C11] "Mensili
[D11] ann.rip/12
[D12] d9
```

Il reticolo è adesso sufficientemente completo per calcolare le rate del mutuo. Prova ad usare il comando **Xecute** e scrivi le cifre richieste, in modo da osservarlo in funzione.

Per migliorare l'aspetto di quest'esempio, possiamo cambiare il formato di alcuni numeri con il comando **Notazioni**. In quest'esempio non c'è bisogno di alterare il formato numerico del default, dal momento che non devi effettuare nuove immissioni in alcuna delle celle del reticolo, una volta che l'applicazione è stata completata.

```
Notazioni,piena,decimale,cifre decimali 2 campo c5
Notazioni,piena,monetaria,segno meno,campo c4
Notazioni,piena,monetaria,segno meno,campo d10:d11
```

Per migliorarne oltre la visualizzazione, possiamo spostare i numeri nelle righe 4, 5 e 6 a sinistra delle celle:

```
Giustificazione,piena,numeri,sinistra,campo c4:e6
```

Tabella di rateazione del mutuo

Questa sezione descrive come puoi aggiungere la tabella della rateazione del mutuo ipotecario al calcolo dell'ipoteca. La prima parte della tabella delle rate per i valori visualizzati alla Figura 5.7 è illustrata alla Figura 5.8.

```
   |      A      |      B      |      C      |      D      |      E
15 |                           TABELLA DEI RIPAGAMENTI
16 |                           ======================
17 |
18 |                 Anno                 1             2             3
19 |                           ------------------------------------------
20 |                 Aprile          28500.00      28343.30      28164.65
21 |                 Maggio          28196.88      28040.17      27861.53
22 |                 Giugno          27893.76      27737.05      27558.41
23 |                 Luglio          27590.63      27433.93      27255.29
24 |                 Agosto          27287.51      27130.81      26952.17
25 |                 Settembre       26984.39      26827.69      26649.04
26 |                 Ottobre         26681.27      26524.57      26345.92
27 |                 Novembre        26378.15      26221.44      26042.80
28 |                 Dicembre        26075.03      25918.32      25739.68
29 |                 Gennaio         25771.90      25615.20      25436.56
30 |                 Febbraio        25468.78      25312.08      25133.44
31 |                 Marzo           25165.66      25008.96      24830.31
32 |
33 |                 Anno                 1             2             3
34 |
35 |  Bilancio fine anno             24862.54      24705.84      24527.19
```

Figura 5.8 La tabella dei ripagamenti

Se hai già un prestito ipotecario, scrivi le cifre. Non perdere tempo sui risultati dei primi anni – la loro lettura piuttosto deprimente!

```
[C15] "TABELLA RATEAZIONI
[C16] ripro("=",lun(c15))
[B18] "Anno
[C18] riga=col()-2          {colonne da C ad AK}
[B19] riga=ripro("-",larghezza()+1)          {colonne da B ad AK}
[B20] col=mese(riga()-20+$mese.iniz.)          {righe da 20 a 31}
```

Queste immissioni scrivono i titoli della tabella: adesso dobbiamo aggiungere le formule che devono calcolare i valori. Iniziamo con la prima voce, che è la somma iniziale del prestito. Essa viene calcolata con l'addizione degli interessi del primo anno al valore del prestito.

```
[C20] ipo.pre*(1+ipo.int)
```

Poi il resto della prima riga viene calcolato con la sottrazione del pagamento annuale e l'addizione degli interessi per l'anno in corso. Questi valori non devono essere calcolati oltre l'anno in cui il prestito deve essere saldato. Questo è fatto con l'uso della funzione **se()**. Se il numero dell'anno (dato da **col()-2**) è superiore al periodo del prestito ipotecario, zero viene scritto nella cella.

```
[D20] riga=se((col()-2)>$ipo.per,0,(c20-$ann.rip)+
   (1+$ipo.int))    {colonne da D ad AK}
```

Il resto della tabella può essere riempito con una singola formula. Una formula riempie la prima cella che non fa altro che sottrarre le rate mensili dal valore della cella soprastante. Anche qui usiamo la funzione **se()** per evitare che il calcolo vada oltre l'anno in cui si deve saldare il prestito.

```
[C21] se((col()-2)>$ipo.per,0,c20-$men.rip)
```

Puoi quindi usare il comando **Eco** per copiare la formula dalla cella C21 alla serie C21:AK31.

```
Eco,Cella c21,nella serie c21:ak31
```

Adesso possiamo completare la tabella aggiungendo una riga finale per dare il bilancio scoperto alla fine di ogni anno. Probabilmente una buona idea è aggiungere una copia dell'anno, dalla riga 18, per facilitare il riferimento.

```
[B33] riga=anno.per          {colonna da B ad AK}
[A35] "Bilancio fine anno
[C35] riga=se((col()-2)>$ipo.per,0,c31-$men.rip)
                                   {colonne da C ad AK}
```

L'intera tabella, e il bilancio scoperto a fine d'anno devono essere impostati a formato monetario oppure a formato decimale con due cifre decimali. La serie per queste impostazioni sono C20:AK30 e C35:AK35

ANALISI DI FOURIER

Lo scienziato francese Fourier ha dimostrato che una ripetizione di onde di ogni forma può essere composta da una serie di sinusoidi o di cosinusoidi della giusta ampiezza e frequenza. La scomposizione di onde complesse in sinusoidi o cosinusoidi è conosciuta come la sintesi di Fourier e viene usata, per esempio, nella maggior parte dei sintetizzatori musicali in uso oggigiorno.

Il processo opposto, la scomposizione della forma di un'onda complessa in un numero di pure sinusoidi o cosinusoidi, è conosciuto come l'analisi Fourier. Quest'esempio ti permette di sviluppare un analisi Fourier per qualsiasi forma di onda. Tutto ciò che devi fare, è di scrivere l'altezza dell'onda a sedici intervalli equidistanti e lasciare che le formule nel reticolo facciano il resto. Le formule presumono che l'onda ripeta la forma dopo il sedicesimo valore, cioè il diciasettesimo valore uguale al primo, il diciottesimo uguale al secondo e così via.

Calcolo della Trasformata di Fourier

Dal momento che il calcolo richiede un bel po' di tempo vale la pena di disattivare il calcolo automatico, usando il comando **Quadro**, prima di scrivere l'esempio.

```
[C1] "ANALISI DI FOURIER
[C2] ripro("=",lun(c1))

[B3] "Funzione:
[A7] "Immissione
[A8] "Valori
```

I valori d'immissione vengono scritti nelle sedici celle da B9 a B24 incluso.

```
col=riga()-9        {righe da 9 a 24}
```

I componenti del coseno

Adesso impostiamo i titoli per la tabella che deve calcolare i componenti del coseno dell'onda. Il risultato contiene la quantità di tutte le cosinusoidi nell'immissione.

```
[E3] "Trasformata:
[E4] "Coseno
[D6] "Cicli riga=col()-5        {colonne da E ad T}
[D8] "Campione
```

In modo sorprendente, l'intera trasformazione del coseno può essere sviluppata con una singola formula. In ogni riga il valore d'immissione viene moltiplicato dal coseno di un angolo (in radianti) che è calcolato nel modo seguente:

angolo = 2 * pi() * numeroriga * numerocol / 16

Il numero di riga e il numero di colonna sono valori dati nella riga col nome "Ciclo" e nella colonna col nome "Campione" rispettivamente. Tutte e due numerano da zero a quindici. Il divisore finale è semplicemente il numero di punti nell'immissione (o emissione).

```
[E9] indice(2,riga())*cos(pi()*(riga()-9)*(col()-5)/8)
```

Adesso usa il comando **Eco** per copiare il contenuto della cella E9 alle celle nella serie da E10 a T24.

Il risultato finale viene calcolato sommando il contenuto di ogni colonna per produrre i sedici valori d'emissione.

```
[A26] "Componenti
[E26] riga=somma(col)        {righe da 9 a 24, colonne da E ad T}
```

I componenti del seno

Il calcolo dei componenti del seno segue esattamente la stessa forma usata per il calcolo del coseno. I risultanti valori sono le quantità di tutte le onde sinusoidali nella immissione.

```
[X4] "Seno
[X6] riga=col()-24
[X9] indice(2,riga())*sen(pi()*(riga()-9)*(col()-24)/8)
                                {colonna X ad AM}
```

Adesso **Eco** il contenuto della cella X9 sulla serie da X10 ad AM24, per riempire il resto della tabella, e poi **Eco** il contenuto della cella C9 alla colonna V, da V9 a V24 (questo copia tutti i valori del "Campione").

[X26] `riga=somma(col)` {righe da 9 a 24, colonne X ad AM}

Lo Spettro della Potenza

Le onde d'immissione che non fossero nè cosinusoidi nè sinusoidi producono generalmente componenti in trasformata sia seno che coseno. Inoltre, quando calcoli la trasformata di molti tipi di onde, alcuni componenti risulteranno negativi. In modo da ottenere risultati che unificano tutte e due le trasformate, e non siano mai negativi, dobbiamo fare un altro calcolo in più. Questo addiziona il quadrato dei componenti del seno e del coseno. Nel caso di un'onda reale questo risultato dimostra quanta potenza (energia per secondo) sia presente nell'onda ad ogni frequenza, rispettivamente nei componenti seno o coseno. Questo è generalmente conosciuto come lo spettro della potenza (uno spettro registra quanta frequenza di ciascuno presente). In questo caso noi calcoliamo la radice quadrata dello spettro della potenza, per evitare di ottenere valori troppo grandi per la semplice illustrazione grafica che vogliamo rappresentare.

```
[C28] "Potenza
[E28] riga=qdr(comp.cos*comp.cos + comp.sen*comp.sen)
```
{colonne da E a T}

Illustrazione Grafica della Trasformata di Fourier

I risultati di questo calcolo possono apparire più chiari presentandoli in forma grafica. Se desideri ottenere grafici di alta qualità, il miglior modo è di usare il comando **Esporta** per creare file che possono essere letti con Easel, che contengono valori d'immissione e d'emissione del calcolo. Le seguenti addizioni al reticolo ti permettono di vedere risultati in forma grafica molto semplice.

```
    |      A      |     B      |     C      |      D
 30 |  Grafica
 31 |  max =                      10.00          80.00 = max
 32 |  min =                     -10.00           0.00 = min
 33 |  ..................*           *
 34 |  ..................*           ...................*
 35 |  ...............*              *
 36 |  ............*                 *
 37 |  .........*                    *
 38 |  ......*                       *
 39 |  ..*                           *
 40 |  *                             *
 41 |  *
 42 |  *
 43 |  ..*
 44 |  ......*
 45 |  .........*
 46 |  ............*
 47 |  ..............*
 48 |  ................*
                  IMMISSIONE              SPETTRO DELLA POTENZA
```

Figura 5.9 Semplice emissione grafica

I grafici d'emissione sono la metà, in dimensione, del grafico d'immissione, dal momento che la frequenza rilevabile numericamente è uguale alla metà dei numeri di punti d'immissione. L'informazione completa è presente nella prima metà dei risultati.

La prima parte produce un istogramma dei valori d'immissione.

```
[A30] "Grafico
[A31] "max=
[B31] max(col)        {righe da 9 a 24}
[A32] "min=
[B32] min(col)        {righe da 9 a 24}
[A33] col=ripro(".",(max.funz-$min.funz)
        *18/($max.funx-$min.funz+1))+"*"        {righe da 33 a 48}
```

La seconda serie di immissioni forma il grafico dello spettro.

```
[D31] "= max
[C31] max(e28:t28)
[D32] "= min
[C32] 0
[C33] col=ripro(".",(indice(riga()-28,28)-$min.pot)
        *18/($max.pot-$min.pot+1))+"*"        {righe da 33 a 40}
```

La seguente serie di immissioni forma il grafico dei componenti del coseno.

```
[F31] "= max
[E31] max(e26:t26)
[F32] "= min
[E32] min(e26:t26)
[E33] col=ripro(".",(indice(riga()-28,26)-$min.cos)
        *18/($max.cos-$min.cos+1))+"*"      {righe da 33 a 40}
```

La serie finale delle immissioni forma un grafico dei componenti del seno.

```
[Y31] "= max
[X31] max(x26:am26)
[Y32] "= min
[X32] min(x26:am26)
[X33] col=ripro(".",(indice(righe()-9,26)-$min.sen)
        *18/($max.sen-$min.sen+1))+"*"      {righe da 33 a 40}
```

Uso della Trasformata di Fourier

Come già stato accennato in precedenza, tu dovresti mettere i valori d'immissione nella celle B9 a B24 inclusa. Puoi provare qualsiasi serie di valore desideri, ma ecco qui di seguito alcuni suggerimenti

```
[B9] col=10*cos(pi()*(riga()-9)/8)      {righe da 9 a 24}
[B9] col=10*cos(pi()*(riga()-9)/4)      {righe da 9 a 24}
[B9] col=10*sin(pi()*(riga()-9)/8)      {righe da 9 a 24}
[B9] col=10*sgn(cos(pi()*(riga()-9)/8))     {righe da 9 a 24}
[B9] col=10     {righe da 9 a 24}
```

Ricordati che, dal momento che il calcolo automatico è stato disattivato, devi usare **Xecute** per calcolare ogni risultato.

Un altro vantaggio che offre l'inclusione di molte etichette di nomi, è che ti permette di spostare la finestra alla maggior parte dei punti interessanti nel reticolo con l'uso del tasto **va a(F5)**, seguito dal rifcella nella forma di etichettamento.

CAPITOLO 6
RIFERIMENTO QL ABACUS

I TASTI PER LA FUNZIONE

Oltre all'uso normale dei tasti per la funzione, **F1**, **F2** ed **F3**, i tasti **F4** e **F5** vengono utilizzati nel modo seguente:

F4	sposta il cursore tra le due metà divise della finestra
F5	Va a cella

RIFCELLA

Puoi far riferimento a singole celle, colonne o serie con l'uso determinato del riferimento di una lettera con un numero oppure con l'uso dei nomi di testo (etichettamento).

Singole celle

Un riferimento ad una singola cella (rifcella) consiste di due parti, un riferimento di colonna e uno di riga.

Vi sono 64 colonne nel reticolo ed esse sono etichettate da A a BL. Vi sono 255 righe, numerate da 1 a 255. Caratteristiche rifcelle sono

```
A1 AC13 BD200
```

Riferimanto serie

Un riferimento di serie è formato da due rifcelle, separati da due punti. Devi sempre scrivere il due punti per separare le due parti del riferimento. Il primo rifcella specifica l'angolo superiore sinistro del blocco e il secondo identifica l'angolo inferiore destro. Esempi di riferimenti di serie (serie) sono:

```
B5:D9
AZ23:BA155
```

Rifriga e rifcol (riferimento a riga e a colonna)

Una parte di una riga o di una colonna può essere considerata come una serie che è larga una riga (o profonda una colonna). Quindi puoi usare un riferimento di serie per specificare parte di una riga o di una colonna, come per esempio:

```
A3:L3      {celle A ad L di riga 3}
D7:D11     {celle da 7 a 11 della colonna D}
```

Identificatore di serie

Vi sono due identificatori di serie: **riga** e **col**. Esse si riferiscono rispettivamente alle celle in uso (quelle che si incrociano alla cella che corrisponde all'identificatore della serie).

Ogni volta che ne usi una in una formula ti viene chiesto di specificare la serie della serie esatta delle celle nella riga o nella colonna. Abacus ti suggerisce dei punti iniziali e finali per la serie e tu sei libero di accettare o cambiare questa scelta.

Vi sono due modi in cui puoi usare gli identificatori di serie. Puoi riempire la riga o la colonna in uso, utilizzando

riga = (*formula*) oppure **col** = (*formula*)

Puoi anche usarli come argomenti per qualsiasi funzione che richiede una serie, per esempio, conta(riga). Puoi, naturalmente, adottarli in questo modo quando vuoi semplicemente riferirti alle celle di una singola riga o colonna.

Puoi mischiare i due metodi liberamente, per esempio,

col = **media**(*riga*)

Ogni volta che usi uno di questi in una formula Abacus, ti chiede di specificare una determinata serie.

Rifcella relativo e assoluto

Abacus normalmente presume che i riferimenti di celle sono relativi. Il fattore importante è la differenza della posizione tra la cella che contiene il riferimento e la cella a cui ti riferisci. Quando fai copia di tale riferimento in un'altra cella, i riferimenti vengono modificati per mantenere questa differenza relativa. Per esempio, immagina che una formula nella cella B2 contenga un riferimento alla cella A1 (una colonna a sinistra e una riga in alto). Se la formula nella cella B2 viene copiata nella cella D4 farà, in questa nuova posizione, riferimento alla cella C3 (di nuovo una colonna a sinistra e una riga in alto).

Questo è illustrato alla Figura 6.1. Una formula nella cella X contiene un riferimento alla cella leggermente oscurata. Se questa formula viene copiata nella cella Y, allora farà riferimento alla cella oscurata con più densità. Le due celle in ognuna delle due coppie hanno la stessa posizione relativa.

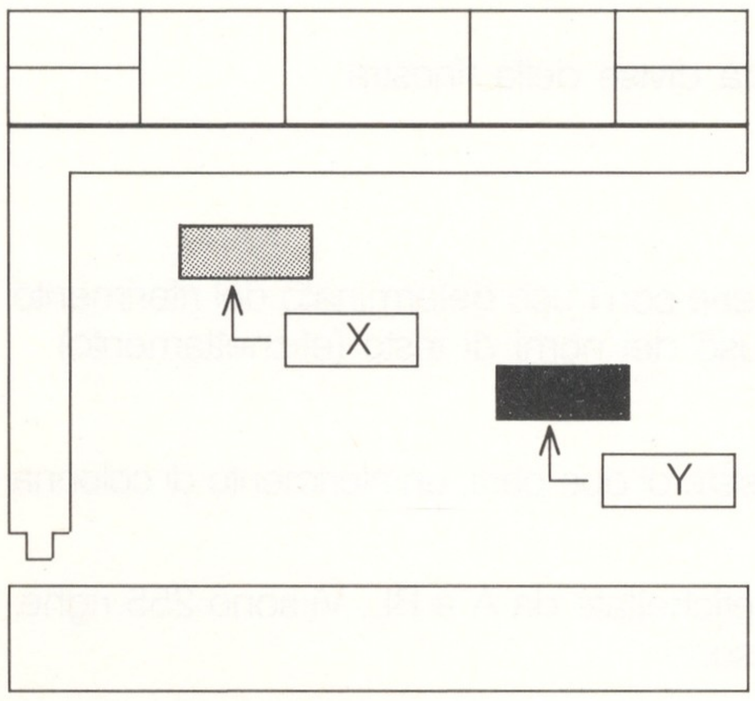


Figura 6.1 Rifcella relativo

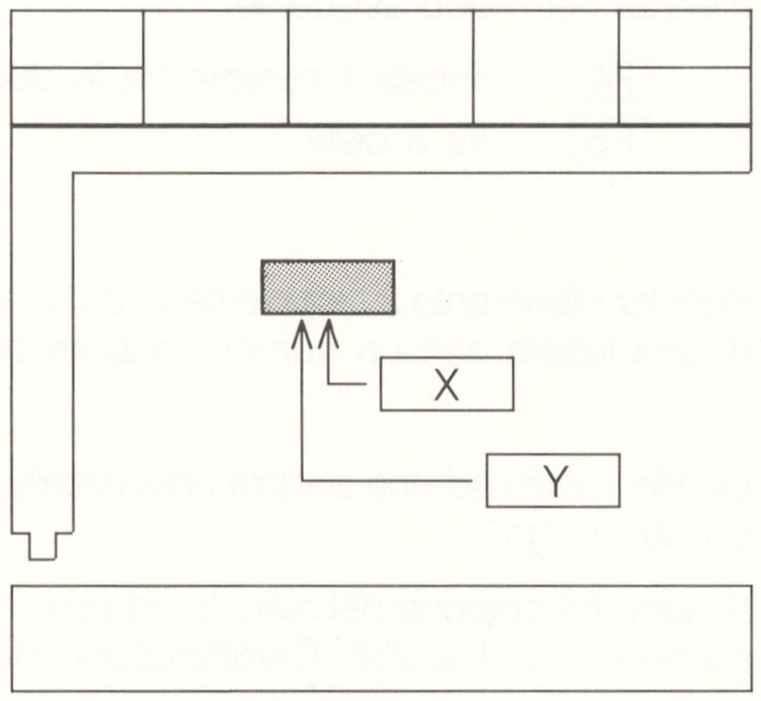


Figura 6.2 Rifcella assoluto

Supponiamo di copiare la formula A1 * 2 nella cella A2, e poi usiamo il comando **Eco** per copiare la formula nelle celle della serie B2:G2. Se esaminiamo il contenuto delle celle nella riga 2, troviamo che queste hanno il seguente contenuto:

Cella:	A2	B2	C2	D2	E2	F2	G2
Contenuto:	A1 * 2	B1 * 2	C1 * 2	D1 * 2	E1 * 2	F1 * 2	G1 * 2

Puoi rendere assoluto ogni rifcella se lo fai precedere dal segno monetario $. Questo riferimento non viene modificato quando la formula viene copiata in altre celle. Per esempio, se un riferimento nella cella B2 era $A1, tutte le copie della formula avranno anche il riferimento $A1. Puoi anche usare nomi per far riferimento ad un rifcella assoluto (es. $costi.marzo).

Figura 6.2 illustra l'effetto di un rifcella assoluto. Una formula nella cella X contiene un riferimento assoluto alla cella leggermente oscurata. Una copia della formula nella cella Y fa riferimento alla stesa cella.

Proviamo adesso l'esempio precedente, ma questa volta usiamo un riferimento assoluto. Scrivi la formula $A1 * 2 nella cella A2 e **Eco** la stessa formula nelle celle B2 a G2 incluse. Troverai che il contenuto delle celle sarà il seguente;

Cella:	A2	B2	C2	D2	E2	F2	G2
Contenuto:	$A1 * 2	$A1 * 2	$A1 * 2	$A1 * 2	$A1 * 2	$A1 * 2	$A1 * 2

Vedi anche la funzione **Indice**.

Gamme di celle, in qualsiasi forma (compreso identifiche, righe e col) sono sempre relative.

NOMI

Nomi di righe e colonne

Un nome è il testo che scrivi in una cella. Il testo deve includere solo lettere e cifre. Una di queste celle può essere usata per identificare una riga o colonna nel reticolo. Puoi anche usare nomi per far riferimento ad una singola cella, ma non puoi usarli per sostituire un riferimento di una serie oppure per far riferimento ad un intero blocco di celle. Ogni volta che fai riferimento ad un nome in un'espressione o in una formula, Abacus usa una serie di regole per determinare se si riferisce ad una riga, una colonna oppure una cella. Le regole per righe e colonne sono i seguenti:

1 La riga e la colonna che s'incrociano al nome vengono esplorate (alla destra e in basso) per trovare l'immissione numerica.

   a) Se solo un'immissione di riga viene trovata, il nome si riferisce alla riga, iniziando dall'immissione trovata.

   b) Se solo un'immissione di colonna viene trovata, il nome si riferisce alla colonna, iniziando dall'immissione di colonna.

   c) Se le immissioni vengono trovate sia nella riga che nella colonna, l'immissione più vicina alla cella denominata viene usata per effettuare la scelta.

2 Se non è possibile nessuna decisione sotto la regola 1), ma il nome viene usato alla sinistra dell'espressione, gli verrà dato il tipo di qualsiasi nome/i usato/i alla destra. Per esempio, se "Costi" è un nome di riga:

```
Vendite = Costi * 0,5
```

allora "Vendite" sarà anche un nome di riga.

Se tutte e due queste regole non funzionano, vieni informato che Abacus non può decidere il significato del nome.

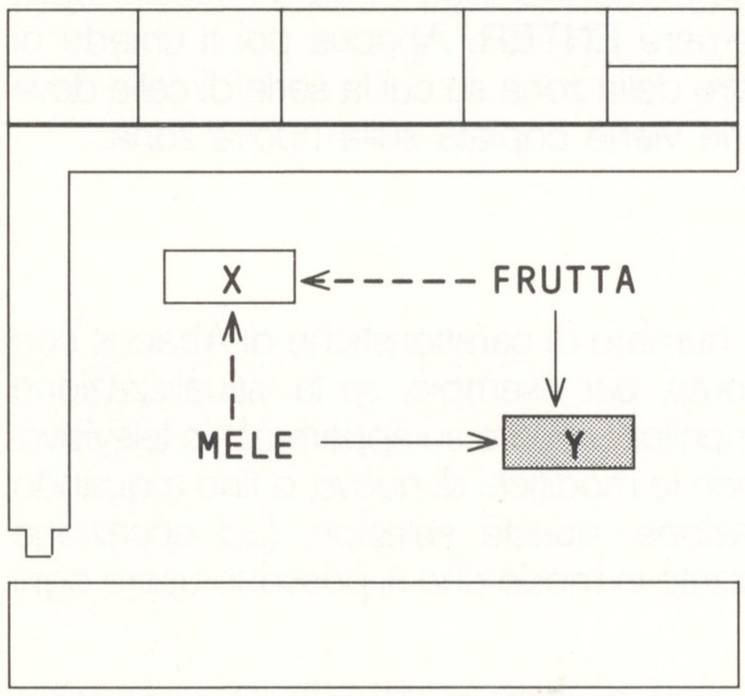


Figura 6.3 de Nominazione di una cella

Nomi delle celle

Hai bisogno di usare due nomi per identificare una cella singola; formi il Rifcella dando tutte e due i nomi, separati da un punto. Per esempio, se hai due nomi "frutta" e "mele", puoi far riferimento ad una cella come

```
frutta.mele
```

(oppure con un'abbreviazione unica, come per esempio fru.mel). L'ordine dei due nomi non è importante quindi, puoi usare mele rutta, melru e così via

Questo riferimento si riferisce alla cella all'incrocio delle righe con le colonne che contengono i nomi ma, come illustrato alla Figura 6.3, vi sono due celle di questo tipo (etichettate X e Y).

La cella che viene selezionata è quella nella colonna più a sinistra e nella riga più in basso delle due. Nell'esempio precedente, la cella etichettata Y viene scelta. Devi, quindi, sempre posizionare i nomi sopra o alla sinistra delle celle a cui si riferiscono.

FORMULE

Una formula è formata da una combinazione permessa di funzioni, rifcelle, nomi ed operatori aritmetici. Gli esempi son:

```
A1*B3
mese(col()-1)
se(istr(B6,""),1,0)
ripro("=",lun(G23))+":"
```

Formule maestre

Ogni nuova formula, oltre ad essere usata in una o più celle del reticolo, viene memorizzata separatamente in una lista di formule maestre. Ogni formula maestra può quindi apparire in una cella o in molte celle. Quando riempi le celle con l'uso delle operazioni per il riempimento delle celle, oppure con l'uso dei comandi **Copia** o **Eco**, tutte le celle riempite condividono una formula maestra. Se una formula maestra contiene rifcelle, questi vengono aggiustati in ogni cella in cui si usa la formula, in modo che sia valida per quella particolare posizione. Le formule potranno apparire superficialmente diverse, ma sono tutte basate sulla formula maestra.

Puoi modificare tutte le copie delle formule apportando la modifica ad una copia sola. Se usi il comando **Altera** per modificare una copia qualsiasi della formula maestra, la maestra viene anche modificata e tutte le copie vengono cambiate simultaneamente.

I COMANDI

Questa sezione contiene una descrizione completa di tutti i comandi disponibili in Abacus.

ALTERA (A)

Questo comando ti permette di cambiare il contenuto di una cella. Il contenuto della cella in cui si trova il cursore viene copiato alla riga informativa d'immissione, pronto per essere modificato con l'uso dell'editore di riga descritto nelle istruzioni dei Programmi QL. Quando premi **ENTER**, la versione modificata sostituisce il contenuto originale della cella.

COPIA (C)

Questo comando è usato per copiare una serie di celle da una zona del reticolo ad un'altra serie analoga in un'altra zona. Abacus prima ti chiede di dare il rifcella della serie da copiare, es. A1:B3 e quindi devi premere **ENTER**. Abacus poi ti chiede di specificare il rifcella per l'angolo superiore sinistro della zona su cui la serie di celle deve essere copiata. Quando premi **ENTER** la serie viene copiata sulla nuova zona.

QUADRO (Q)

Il comando **Quadro** è usato per modificare un numero di caratteristiche di Abacus che hanno effetto sull'aspetto di tutto il reticolo, come, per esempio, se la visualizzazione deve essere impostata per essere mostrata su "monitor" oppure su apparercchio televisivo. Le selezioni rimangono attivate fino a quando non le modifichi di nuovo, o fino a quando non lasci Abacus. Se memorizzi un'applicazione, queste selezioni (ad eccezione dell'opzione **Quadro**) vengono anche memorizzate in modo che si possono usare ogni volta che richiami l'applicazione.

Il cambiamento dei default, comunque, non ha nessun effetto su Abacus. Devi registrarli ai valori richiesti ogni volta che richiami Abacus da SuperBASIC.

Quando hai completato l'operazione, ritorni di nuovo alla visualizzazione della tabella elettronica premendo **ENTER**. Le opzioni sono le seguenti:

Calcolo automatico all'immissione
usato per specificare se si desidera l'autocalcolo oppure no. Ogni volta che premi il tasto **C** l'opzione dell'autocalcolo viene attivata e disattivata tra SI e NO.

Se selezioni SI, tutta la tabella viene calcolata dopo ogni immissione. Selezionando NO, la tabella effettua il calcolo solo quando usi il comando **Xecute**. Lo stato iniziale è SI.

Vuota se zero
cambia tra due modi nel trattare i valori di zero nel reticolo. L'opzione originale è di visualizzare il valore di zero nell'esatto formato di quella cella. Puoi selezionare l'alternativa, che è di visualizzare una cella vuota se il suo contenuto risulta zero.

Nota che, in questa opzione, una cella vuota viene visualizzata solo se il valore è realmente zero. Supponiamo di aver selezionato un formato a visualizzazione decimale, con due cifre decimali, e il valore in questa cella è 0,003. La cella mostra 0,00, anzichè essere vuota, dal momento che il valore reale è diverso da zero.

Ordine di calcolo
seleziona il calcolo della tabella elettronica per RIGA oppure per COLONNA. L'opzione cambia ogni volta che premi il tasto **O** (come accade per il calcolo automatico). L'ordine specificato viene usato sia per il comando **Xecute**, che per il calcolo automatico. Il valore iniziale è per ordine di calcolo per riga.

Monitor 80,60,40 colonne
seleziona la visualizzazione di tutto la tabella. Ti si chiede di scrivere 8,6 o 4 (seguito da **ENTER**) per scegliere la visualizzazione a 80, 60 o 40 caratteri. Il valore iniziale è registrato a 80 o a 40 a seconda se scegli l'opzione TV o Monitor quando richiami Abacus dalla cartuccia del Microdrive.

Avanzamento modulo tra le pagine
seleziona se un avanzamento modulo viene emesso alla fine di ogni pagina di emissione stampata. Allo stesso modo dell'autocalcolo il valore iniziale è SI.

Interlinee sulla stampante
assegna lo spazio tra le righe sullo stampato, specificando il numero di spazi tra le righe di testo. Ti si chiede di scrivere 0.1 o 2 (non c' bisogno di battere **ENTER**). Puoi registrare stampa ad interlinee doppie, specificando lo spazio tra ogni riga. Il valore iniziale è zero.

Righe
specifica il numero di righe stampate per pagina. Devi scrivere un numero, seguito da **ENTER**. Il valore iniziale è 66 e il massimo è 255.

Simbolo valutario
specifica il simbolo valutario da usare nella visualizzazione di valori monetari. Devi scrivere il singolo carattere che desideri (non c'è bisogno di battere **ENTER**). Il valore iniziale è il simbolo monetario del Dollaro.

Stampante
assegna il numero di caratteri per riga da stampare. Devi scrivere un numero, seguito da **ENTER**. Il valore iniziale è 80 e il massimo è 255.

ECO (E)

Il comando **Eco** copia un dato o una formula da una determinata cella in tutte le celle di una serie specificata.

Hai l'opzione di specificare il rifcella della cella da copiare, oppure premere **ENTER** per copiare la cella in uso. Dopo di che, devi scrivere la serie su cui il contenuto della cella deve essere copiato, seguito da **ENTER**.

FILE (F)

Questo comando ti permette di eseguire operazioni sui file di Abacus, che sono stati memorizzati precedentemente su una cartuccia Microdrive. L'opzione ti chiede di scrivere i nomi di file. Ogni volta che ti si chiede il nome di un file puoi premere **?** per visualizzare un'elenco di tutte i file nel Microdrive 2.

Ti verranno offerte le seguenti opzioni:

Backup
usato per fare una copia di un file di Abacus. Ti si chiede il nome del file da copiare. Ti si raccomanda di fare delle copie di tutti i file, per protezione contro perdita accidentale, o danneggio della cartuccia.

Cancella
cancella un file nominato dalla cartuccia di un Microdrive. **Nota che questo comando NON revocabile e quindi deve essere usato con GRANDE CAUTELA**.

Esporta
esporta un nomefile. Il file viene memorizzato in una forma adatta per essere importato da Archive, Easel e Quill.

Abacus prima ti chiede se vuoi esportare a Quill, Archive o Easel. Accetta il suggerimento di esportare a Quill premendo **ENTER**, oppure seleziona l'esportazione ad Archive o Easel premendo il tasto **A** oppure il tasto **E**.

In tutti i casi ti viene chiesto di scrivere il riferimento della serie per la sezione del reticolo che vuoi esportare, terminando l'immissione premendo **ENTER**.

Se hai scelto di esportare ad Archive o ad Easel puoi effettuare l'esportazione per righe o per colonne. Abacus ti chiede di premere **ENTER** per accettare il suggerimento di esportare per righe, oppure di premere il tasto **C** per esportare per colonne. Nessuna opzione ti viene data quando esporti a Quill. In questo caso i dati vengono sempre esportati per righe.

Infine Abacus ti chiede di scrivere un nome per il file esportato. Se non specifichi una estensione del nome del file, Abacus fornisce una estensione di **__exp.**

Formatta
formatta la cartuccia nel Microdrive 2. Abacus ti offre la specificazione del Microdrive, mdv2__ e tu devi scrivere il nome di un volume per la cartuccia.

Assicurati che la cartuccia nel Microdrive 2 non contenga nessun file che vuoi conservare – TUTTO il contenuto della cartuccia viene cancellato.

Importa
importa un nomefile. Permette ad Abacus di leggere file esportati da Archive o da Easel. C'è una descrizione completa su importa nella sezione *Informazioni* della Guida dell'Utente.

Puoi importare un file per riga o per colonna, ti viene chiesto di selezionare in quale ordine. Ti viene anche chiesto il rifcella dell'angolo superiore sinistro della zona in cui i dati devono essere importati.

Se non specifichi una estensione del nome del file Abacus presume l'estensione di **exp**.

TRACCIATO (T)

Il comando **Tracciato** viene usato per apportare delle modifiche che influiscono sull'intera tabella. Ti permette d'inserire o di cancellare un'intera riga o colonna, o di cambiare il numero di caratteri visualizzati in una o più colonne.

Le opzioni sono le seguenti:

Inserisci
ti permette d'inserire righe o colonne vuote nel reticolo. Prima ti viene chiesto se vuoi inserire righe (premi **ENTER**) oppure colonnè (premi **C**). Poi ti viene chiesto di dare un rifriga (o rifcol) e il numero di righe o colonne da inserire. Quando premi **ENTER** righe o colonne vuote vengono inserite davanti a quelle specificate. Le ultime righe (o colonne) vengono perdute dal reticolo. Se, per esempio, inserisci tre righe, le ultime tre righe del reticolo visualizzato precedentemente saranno perse.

Non puoi recuperare nessun dato che era contenuto in queste righe, a meno di non riscriverli.

Cancella
ti permette di cancellare una o più righe o colonne dal reticolo. Prima ti si chiede se vuoi cancellare righe (premi **ENTER**) oppure colonne (premi **C**). poi ti viene chiesto di fornire un riferimento della riga iniziale (o colonna) della zona che vuoi cancellare, seguito da **ENTER**. Poi ti viene ancora chiesto di fornire il riferimento della riga (o colonna) finale della zona.

Quando poi premi **ENTER**, la zona selezionata viene cancellata e le righe seguenti (o colonne) si ristringono per riempire lo spazio. Righe (o colonne) vuote vengono inserite in fondo (o all'estrema destra) al reticolo.

In tutte e due queste opzioni tutte le formule nelle righe o nelle colonne vengono aggiustate per correggerle per la nuova posizione.

Larghezza
ti permette di cambiare la larghezza (numero di caratteri) di una o più colonne. Prima ti viene chiesto di specificare il numero di caratteri in una colonna, e poi di specificare le colonne iniziali e finali su cui vuoi effettuare la modifica.

GIUSTIFICA (G)

Il comando **Giustifica** viene usato per modificare il posizionamento del testo o dei numeri in una serie di celle. Offre due opzioni; la modifica di celle Piene, oppure la modifica di default per cambiare il posizionamento di celle Vuote. Per scegliere celle Piene premi **ENTER**, per scegliere l'opzione di default di celle Vuote premi **V**.

Poi ti viene chiesto di specificare se vuoi modificare la giustificazione del testo (premi **ENTER**) o dei numeri (premi il tasto **N**). In tutte e due i càsi puoi allora selezionare giustificazione a sinistra (**ENTER**, centro (**C**) o destra (**D**).

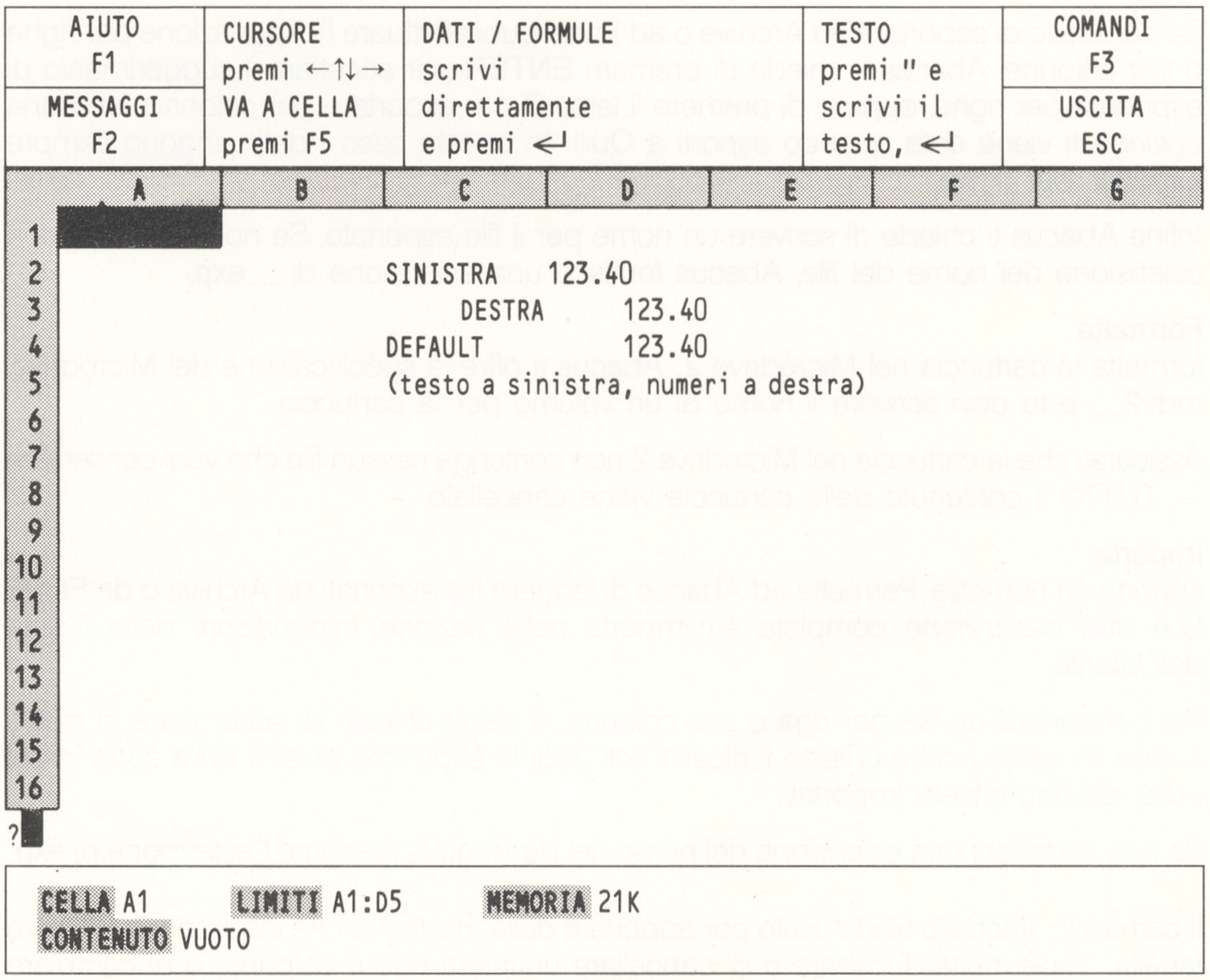


Figura 6.4 Giustifica

Nel caso dell'opzione di celle piene, ti viene infine chiesto di fornire il campo della serie su cui deve essere effettuata la giustifica.

Se selezioni l'opzione del Default, (**V**) non devi fornire il campo della serie. Il nuovo default viene applicato a tutte le celle che vengono formate, in qualsiasi punto del reticolo, fino a quando non apporti un'altra modifica al default di giustificazione.

Alcuni dei diversi tipi di giustificazione, insieme alle registrazioni originali (testo giustificato a sinistra e numeri giustificati a destra) sono illustrati alla Figura 6.4.

RICHIAMO (R)

Questo comando viene usato per richiamare un file dal Microdrive. Prima ti viene chiesto di specificare il riferimento del file; premendo il tasto **?** a questo punto puoi visualizzare un elenco di file nel Microdrive 2.

Se non includi una estensione nel nome del file che scrivi, Abacus presume un'estensione **__aba**.

UNIONE (U)

Questo comando viene usato per unificare o per consolidare, dati da file memorizzati precedentemente con i dati nel reticolo in uso. Prima ti viene chiesto di fornire il nome del file da unire dalla cartuccia del Microdrive e poi devi indicare se i dati in questo file devono essere aggiunti a (premi **ENTER**) o sottratti dai (premi **S**) dati nella tabella in uso.

Ogni qualvolta una cella (nel file), che contiene un numero o una formula, trova dati corrispondenti e uguali nella tabella, il valore viene aggiunto o sottratto dai dati della tabella. Il contenuto di altre celle non viene toccato. Il comando non ha nessun effetto sulle celle del reticolo che contengono testo, che sono così protette da modifiche.

La tabella risultante consiste solamente di valori numerici in ogni cella che ha subito l'effetto del comando unione. Le formule che hanno riportato questi valori nelle celle originali del reticolo vengono distrutte. Le formule non hanno alcun significato sulla tabella consolidata.

Questo comando offre un metodo rapido e facile di combinare i dati in due modelli analoghi. Naturalmente è importante impostare i due tabelloni allo stesso modo, usando la stessa posizione delle celle, perchè il risultato abbia senso.

ORDINA (O)

Devi usare questo comando per l'ordinamento del reticolo in ordine ascendente, basato sul contenuto di una determinata colonna.

Prima ti viene chiesto di specificare la colonna su cui deve essere basato l'ordinamento. La precisa sequenza usata per l'ordinamento è la seguente:

Celle vuote
Celle numeriche, in ordine numerico ascendente
Celle di testo in ordine alfabetico

Usa il comando **Ordina** solo su celle o su colonne che contengono dati. Le formule presenti nelle sezioni in uso possono con probabilità venire invalidate perchè non sono aggiustate per la nuova posizione.

STAMPA (S)

Questo comando viene usato per mandare una parte selezionata dal reticolo alla stampante oppure ad un file del Microdrive. Prima ti viene chiesto se vuoi che i valori o le formule di ogni cella vengano stampati. Premi **ENTER** se vuoi che vengano stampati i valori o premi il tasto **F** se vuoi che vengano stampate le formule. Poi Abacus ti chiede di specificare la serie di celle che vuoi che sia stampata. Poi ti viene chiesto se vuoi che il bordo del reticolo deve essere stampato (premi **ENTER**) oppure no (premi il tasto **N**). A seguito di questo devi specificare se l'emissione deve essere mandata alla stampante (premi **ENTER**) oppure ad un file del Microdrive (premi il tasto **F**). Se scegli di mandare l'emissione ad un file, ti viene anche chiesto di scrivere il nome del file seguito da **ENTER** per completare l'operazione.

La parte selezionata del reticolo viene inviata alla destinazione prescelta. Lo stampaggio può essere fermato in qualsiasi momento premmendo **ESC**.

Se hai chiesto per la visualizzazione delle formule, Abacus stampa prima un elenco numerato di tutte le formule usate nel reticolo. Poi stampa il reticolo stesso. Il numero della formula appare nella cella che contiene quella formula.

Se tu, nel caso dell'opzione di stampa ad un file, non specifichi una estensione quando scrivi il nomefile, Abacus presume una estensione di **__lis**.

LASCIA (L)

Usi questo comando per abbandonare Abacus quando hai finito di utilizzarlo.

Quando lasci Abacus il contenuto del reticolo in uso sarà perso. Ti viene chiesto di confermare la richiesta, in modo che tu abbia l'opportunità di cambiare idea. Puoi annullare il comando e ritornare alla tabella elettronica premendo il tasto **ESC**. Se invece premi **ENTER**, confermi il comando per abbandonare Abacus e ritorni a SuperBASIC.

VUOTA (V)

Questo comando viene usato per cancellare o vuotare il contenuto di una o più celle nel reticolo. Quando usi questo comando, ti viene chiesto di specificare il campo della serie di celle. Tutte le celle in quella serie saranno vuotate.

MEMORIZZA (M)

Questo viene usato per memorizzare un file su un Microdrive. Prima ti viene chiesto di specificare il nomefile; a questo punto se premi il tasto **?** potrai visualizzare un elenco di file sul Microdrive 2.

Se non includi una estensione quando scrivi il nomefile, Abacus assume una estensione di **__aba**.

NOTAZIONI (N)

Il comando **Notazioni** viene usato per cambiare l'impostazione numerica in una cella, o un gruppo di celle. Non influisce affatto sul valore numerico.

Prima ti viene chiesto di selezionare l'opzione per cambiare celle piene (premi **ENTER** o registrare il formato del default che Abacus userà per tutte le celle che verranno successivamente formate (premi il tasto **V**).

In tutte e due i casi, poi, ti viene chiesto di selezionare il formato dalla lista seguente:

Decimale

I numeri vengono visualizzati in una notazione decimale a punto fisso, cioè vale a dire, tutti i numeri vengono visualizzati allo stesso modo, con un numero fisso di cifre decimali. Numeri che effettivamente hanno più cifre decimali di quelle visualizzate vengono arrotondati al valore più adatto. Ti viene chiesto di scrivere il numero delle cifre decimali che richiedi. Il numero massimo di cifre decimali è 14.

Se vuoi che i valori stessi delle celle vengano arrotondati, anzichè essere solamente visualizzati in forma arrotondata, devi effettuare l'arrotondamento da te stesso. Per esempio, per arrotondare un valore a due cifre decimali:

1) moltiplica per 100 (1000 per arrotondamento a tre cifre decimali, e così via di seguito
2) aggiungi 0,5
3) scarta la frazione decimale con la funzione int()
4) dividi per 100 (o per 1000)

La seguente formula arrotonda il valore nella cella C3 a due cifre decimali:

```
int(c3100+0.5)/100
```

Intero

I numeri vengono visualizzati in forma intera, come dalla funzione **int()**. Hai l'opzione di racchiudere valori negativi tra parentesi, anzichè di farli precedere dal segno meno. Premi il tasto **P** per racchiudere i valori negativi fra parentesi.

Usa la funzione **int()** se vuoi che i valori nella cella vengano convertiti a interi, anzichè essere solo visualizzati in formato intergrale.

Esponenziale

I numeri vengono visualizzati in forma esponenziale, o notazione scientifica. La notazione ti chiede di scrivere il numero di cifre decimali che vuoi visualizzare. Esso non accetta più di 14 cifre. Anche qui il numero visualizzato viene arrotondato al livello più adatto al numero di cifre decimali che hai scelto.

Percentuale

Visualizza i numeri in forma percentuale in modo che, per esempio, il valore 0,55 viene rappresentato come 55%. L'opzione ti chiede di scrivere il numero di cifre decimali che vuoi visualizzare. Il numero massimo di cifre decimali accettabili è 14.

Generale

Questo è un formato numerico generale in cui qualsiasi dei formati precedenti viene selezionato, a seconda del valore del numero, per utilizzazione ottimale dello spazio disponibile nella cella.

Monetario
I numeri vengono rappresentati in formato decimale, con due cifre decimali e preceduti da un segno valutario. Hai l'opzione di racchiudere i valori negativi tra parentesi, anzichè di farli precedere dal segno meno. Premi il tasto **P** per racchiudere i valori negativi tra parentesi, oppure premi **ENTER** per il segno meno.

Nel caso dell'opzione di celle piene, Abacus ti chiede alla fine delle scelte suddette, di specificare la serie su cui effettuare l'esecuzione. Puoi scrivere qualsiasi forma di rifcella o riferimento di serie (compreso nomi o identificatori di gamme). Premi **ENTER** per segnare la fine del riferimento. Abacus non ti chiede di specificare la serie se scegli l'opzione del default. In questo caso il formato selezionato viene usato per tutte le celle, come vengono formate.

DIVIDI (D)

Usi questo comando per controllare se la visualizzazione è una finestra singola oppure se la finestra è divisa in due parti che possono essere usate per visualizzare due porzioni separate del reticolo.

Prima ti viene chiesto di selezionare la divisione verticale (**V**) o quella orizzontale (**O**), oppure di riunire (**R**) la visualizzazione divisa in una finestra singola. Se la finestra inizialmente è divisa e vuoi cambiare, diciamo, una divisione orizzontale ad una verticale devi riunire le due finestre prima di effettuare la nuova divisione.

Se scegli di dividere la finestra, la divisione avviene alla colonna o riga che contiene il cursore. Quindi devi posizionare il cursore dove desideri effettuare la divisione, prima di iniziare l'esecuzione di divisione. Colonne intere vengono sempre visualizzate. Ogni finestra in una divisione verticale non ha mai meno di dieci caratteri di larghezza.

Poi ti viene data un'altra scelta. Se vuoi che le due finestre si devono muovere assieme (premi **ENTER**) o separatamente (premi (il tasto **S**). Se specifichi l'opzione "assieme", significa che ogni cambiamento nella posizione della finestra – nella direzione parallela alla divisione – causa un corrispondente cambiamento nella posizione dell'altra. Movimenti ad angolo retto alla divisione non sono correlati in questo modo. L'opzione **S** permette che le due finestre si muovono per tutto il reticolo indipendentemente l'una dall'altra.

XECUTE (X)

Questo comando viene usato per forzare il calcolo delle formule che sono nel reticolo. Un calcolo viene generalmente effettuato automaticamente dopo ogni immissione nel reticolo. Hai bisogno di questo comando soltanto se hai disattivato l'opzione per il calcolo automatico se vuoi attivare qualsiasi delle funzioni **chieden()** o **chiedet** memorizzate nelle celle del reticolo.

ZERO (Z)

Questo comando cancella tutto il reticolo e ritorna al riavvio di Abacus. Dal momento che questo comando drastico (e irrevocabile) nell'esecuzione, ti viene chiesto di confermare la richiesta. Se premi **ESC** ritorni al menù comandi senza effettuare nessuna cancellazione. Se premi **ENTER** confermi la richiesta di cancellare il reticolo.

FUNZIONI

Considera una funzione come una specie di ricetta che converte un numero di valori, conosciuti come gli *argomenti* della funzione ad un valore diverso, che si dice il valore riportato dalla funzione. In Abacus questo il valore si visualizza in una cella che contiene la funzione.

Le funzioni offerte da Abacus possono prendere tre, due, uno o no argomenti che vengono messi fra parentesi dopo il nome della funzione. Fra il nome e la parentesi di apertura non ci deve essere spazio, mentre spazio viene permesso fra voci racchiuse da parentesi. Se una funzione richiede più di un argomento, allora questi sono separati da virgole. Tutte le funzioni devono essere seguite da parentesi, anche se non richiedono alcun argomento. La presenza di parentesi è un utile metodo per ricordarsi che si fa riferimento ad una funzione.

Nelle descrizioni delle funzioni:

n	un'espressione numerica oppure un riferimento ad una cella che mostra un valore numerico,
testo	un'espressione di testo oppure un riferimento ad una cella che mostra un valore di testo,
serie	un riferimento di serie nel reticolo

Un' espressione numerica, un numero, oppure un' espressione che dà un risultato numerico.

Un' espressione di testo, una stringa di testo, (fra virgolette) oppure un' espressione che dà un risultato di testo.

Le seguenti funzioni sono disponibili:

ASS(*n***)**
Riporta il valore assoluto (cioè, il valore ignorando segni meno) dell' argomento.

Per esempio, **ass(3)** riporta 3 e **ass(–7)** riporta 7.

CHIEDEN(*testo***)**
Questa funzione viene usata per l' immissione di dati numerici. Visualizza il dato *testo* (che può essere fino a 40 caratteri lungo) come un "prompt" alla riga d' immissione, seguito da "?", e attende che venga scritta la risposta. La risposta viene visualizzata nella cella che contiene la funzione. Immissione viene chiesta solo dopo che hai scritto la funzione in una cella per la prima volta, e quando effettui il calcolo con l' uso del comando **Xecute**. Non viene chiesta durante un calcolo automatico dopo ogni immissione nel reticolo.

CHIEDET(*testo***)**
Questa funzione viene usata per l' immissione delle stringhe di testo. Funziona allo stesso modo di **chieden()**, con l' eccezione che devi scrivere testo invece di numeri.

ATG(*n***)**
Riporta l' angolo, in radianti, la cui tangente è *n*.

MEDIA(*serie***)**
Riporta la media dei valori numerici contenuti in tutte le celle della *serie*. Celle vuote e celle che contengono testo vengono ignorate nel calcolo della media. Se non vi sono celle numeriche nella serie, la media viene riportata al valore di zero.

CAR(*n***)**
Questa funzione riporta il carattere del codice ASCII il cui codice è *n*. Un carattere con codice ASCII inferiore a 32 non ha alcun effetto sullo schermo, ma viene mandato alla stampante (quando stampi la porzione del reticolo che lo contiene) se preceduto dal carattere "null" del codice ASCII. Per esempio, **car(0)+car(13)** passa il carattere del codice ASCII per salto riga ad una stampante, quando la cella che lo contiene viene stampata.

Puoi visualizzare "A" sullo schermo con **car(65)**.

CODICE*testo***)**
Riporta il valore del primo carattere del codice ASCII trovato nel *testo* specificato.

COL( )
Riporta il numero della colonna in uso.

COS(*n***)**
Riporta il coseno di una dato angolo (in radianti).

CONTA(*serie***)**
Riporta il numero di celle non-vuote della serie. Solo celle numeriche sono incluse nel conteggio.

DATA(*n***)**
Riporta la data odierna come valore di stringa di testo.

n	data stringa
0	"AAAA/MM/GG"
1	"GG/MM/AAAA"
2	"MM/GG/AAAA"

Per prima cosa devi aver impostato l' orologio del sistema, come descritto nella guida parola chiave del sistema SuperBASIC.

GIORNI(*testo***)**
Riporta il numero di giorni, dal 1mo. gennaio 1583, ad una successiva data, da un' espressione di testo della forma "AAA/MM/GG". La conversione presume l' uso del calendario Gregoriano (moderno). La forma, quindi, è valida per date dopo il 1582.

RD(*n*)
Prende un angolo, misurato in radianti, e lo riporta misurato in gradi.

ESP(*n*)
Riporta il valore di e (circa 2,718) elevato alla potenza di *n*. Il valore riportato sarà scorretto se non rimane tra la serie da −87 a +88, dal momento che il risultato supera la serie numerica di Abacus.

SE(*espressione*,*vera*,*falsa*)
Il valore dell'espressione viene calcolato e usato per determinare quale dei seguenti argomenti deve essere riportato.

espressione:= *n*
vero:= *n* | *testo*
falso:= *n* | *testo*

Se l'espressione valuta 0, viene considerata falsa quindi l'argomento di valore "falso" viene riportato. Per l'espressione, qualsiasi argomento non-zero viene interpretato come vero e quindi causa il riporto del valore di argomento "vero" Gli argomenti del valore di "vero" e "falso" possono essere valori numerici o letterali. Perciò i seguenti esempi rappresentano usi positivi della funzione

```
se(A1=B1,"uguale","non uguale")
se(A1,1,0)
```

Puoi anche mischiare un argomento numerico e un argomento di testo come nel seguente esempio. Prova a sviluppare il seguente esempio se non sei sicuro di come funzioni **se()**.

[A1] 1
[B1] 0
[C1] `se(A1 or B1,"antivalenza",0)`

Dovresti vedere apparire la parola "antivalenza" nella cella C1, dal momento che il parametro di **se()** riporta un valore di non-zero (vero) se cella A1 o cella B1 contiene un valore di zero. Se il valore della cella A1 viene cambiato a zero, allora la visualizzazione della cella C1 sarà zero.

INDICE(*colonna*,*rig*)
colonna:= *n*
riga:= *n*

Riporta il contenuto della cella all'incrocio della colonna e la riga specificata.

STRGIINT(*testo1*,*testo2*)
testo1:= *testo*
testo2:= *testo*

Riporta la posizione della prima volta in cui si incontra il *testo2* nel *testo1*. Se non si trova il *testo1* la funzione riporta zero.

```
strgint("gennaio","gen")      {riporta 1}
strgint("gennaio","an")       {riporta 2}
strgint("gennaio","AN")       {riporta 0}
```

INT(*n*)
Riporta il valore intero del numero, tralasciando la parte decimale. Il troncamento della parte decimale rende sempre il numero più piccolo. Quindi:

```
int(3.7)      {riporta 3}
int(-4.8)     {riporta −4}
```

TIR(*serie*,*periodo*)
periodo:= *n*

Calcola il Tasso di interesse della somma degli utili per i dati numerici in una serie specificata, che può essere una riga o una colonna.

I dati nella serie rappresentano il cash flow per ogni serie di periodi, separati da *n* mesi. Valori negativi rappresentano uscite di liquidi e valori positivi rappresentano somma degli utili.

La funzione riporta il tasso d'interesse necessario perchè il costo dell'investimento quadri con gli utili proposti.

La funzione riporta il tasso d'interesse necessario perchè il costo dell'investimento quadri con gli utili proposti.

Per esempio, supponiamo che ti venga offerto un utile di ventimila dollari alla fine di ogni anno per i prossimi sette anni, in remunerazione per un investimento iniziale di centomila dollari. Credi che questo sia un buon affare?

```
[A1] "flusso
[A2] -100000
[A3] col=20000        {righe da 3 a 9}
```

Possiamo far riferimento alla serie dei dati con il nome "flusso" e l'intervallo tra i successivi periodi dodici mesi:

```
[C2] tir(flu,12)      {righe da 2 a 9}
```

Il reticolo completato deve apparire come alla Figura 6.5, mostrando che tasso d'interesse 9,1%. Se puoi investire i centomila dollari ad un tasso d'interesse più alto, dovresti farlo e dimenticare la proposta di quest'affare.

```
  |      A      |      B      |      C
1 | flusso
2 |     -100000.00                    9.10
3 |      20000.00
4 |      20000.00
5 |      20000.00
6 |      20000.00
7 |      20000.00
8 |      20000.00
9 |      20000.00
```

Figura 6.5 Tasso di remunerazione d'interesse

Nota che la prima voce nella serie ha un valore di zero, la seconda uno, e così via. La funzione presume che ogni mese deve essere pagato in pieno alla fine di tutto il periodo.

LUN(*testo*)
Riporta il numero di caratteri nel testo specificato.

LN(*n*)
Riporta il logaritmo naturale, base e, di *n*. Un errore ne risulta se *n* è negativo o zero, dal momento che i logaritmi non sono definiti in questa serie.

GUARDA(*serie*,*sfasata*,*valore*)
sfaso:= *n*
valore:= *n*

La funzione imposta una tabella per guardare nel reticolo. Due tabelle di valori, si presumono essere presenti. La prima tabella occupa la serie specificata (che può essere una riga o una colonna). La seconda tabella, parallela alla prima, nella riga o colonna seguente. Per esempio, se la prima tabella è nella G, da G10 a G25, la seconda si presume di essere nella serie h10 a H25. Ogni immissione nella prima tabella deve avere un'immissione corrispondente nella seconda. Nota che si presume per la corretta operazione di questa funzione, che tutte e due le tabelle contengono valori numerici, e che quelli nella prima tabella siano impostati in ordine ascendente.

Il primo valore nella prima tabella è finto. Deve essere inferiore al secondo valore, che è il limite inferiore per il procedimento dell'esecuzione di "guarda" della tabella. Viene, altrimenti, ignorato. Il primo valore nella seconda tabella è il valore riportato, se si invoca **guarda()** con qualsiasi numero inferiore al limite più basso.

MAX(*serie*)
Riporta il valore numerico massimo trovato nelle celle della serie specificata. Se nella serie non vi sono celle numeriche, la funzione riporta il più piccolo numero possibile (–1,7 E+38).

MIN(*serie*)
Riporta il valore numerico minimo trovato nelle celle della serie specificata. Se non vi sono celle numeriche nella serie, la funzione riporta il numero più grande possibile (–1,7 E+38).

MESE(*n*)
Riporta il mese come valore di testo, il nome del mese *n*

Per esempio **mese(3)** riporta il testo **"marzo"**.

Se un argomento superiore a 12 viene usato, viene sostitituito dal rimanente dopo la divisione per 12 di modo che, per esempio, **mese(13)** e **mese(1)** daranno tutte e due il risultato di **'gennaio'**.

VAN(*serie*,*periodo*,*tasso*)
tasso:= *n*
periodo:= *n*

Riporta il Valore Annuale Netto per i dati del cash flow nella serie specificata. Tasso l'interesse annuo (14 rappresenta un tasso del 14%). I dati si riferiscono ad una serie di periodi, separati da uguali intervalli di mesi.

Il valore attuale netto è la somma dei dati del cash flow richiesta adesso per riportare un futuro cash flow, presumendo un tasso d'interesse. Per esempio supponiamo ti venga data l'opportunità di comprare, con un singolo pagamento di settantamila dollari, un contratto di affitto per un negozio che al momento produce un reddito annuo netto di diecimila dollari. Prevedi che il reddito aumenti del 10% per anno. Se non acquisti il negozio i settantamila dollari ti renderanno il 14% d'interesse. Che devi fare?

Devi calcolare il valore attuale netto del reddito e paragonarlo alla somma che ti si chiede di pagare

```
[A1] "flusso
[A2] 0
[A3] 10000
[A4] col=a3+1.1        {righe da 4 a 12}
[A14] van(flusso,14,12)      {righe da 2 to 12}
```

Il risultato illustrato alla Figura 6.6

```
   |          A          |       B
 1 |  flusso
 2 |                0.00
 3 |            10000.00
 4 |            11000.00
 5 |            12100.00
 6 |            13310.00
 7 |            14641.00
 8 |            16105.10
 9 |            17715.61
10 |            19487.17
11 |            21435.89
12 |            23579.48
13 |
14 |            75088.51
```

Figura 6.6 Valore attuale netto

Il valore attuale netto (nella cella A14) del cash flow dal negozio è più del prezzo chiesto, quindi devi acquistare il negozio.

La prima voce nella lista per periodo zero, il secondo periodo uno, e così via. Questo consistente con la presunzione, fatta dalla funzione, che gli utili vengono ricevuti alla fine di ogni periodo. Quindi devi attendere per tutto un periodo prima che ricevi degli utili dall'investimento. In una situazione reale di questo tipo probabilmente baseresti la funzionalità su base mensile, anzichè di periodi di dodici mesi.

PI( )
Riporta il valore della costante di matematica PI-GRECO.

RD(*n*)
Riporta un angolo, misurato in gradi, e converte lo stesso angolo in radianti.

RIPRO(*testo*,*n*)
Questa funzione riempie la cella in uso con *n* copie del primo carattere di un determinato testo. Per esempio,

```
ripro("*",5)        {scriverà cinque asterischi nella cella in uso}
ripro("abc",3)      {effettua tre ripetizioni di "a"}
```

RIGA( )
Riporta il numero della riga in uso.

SGN(*n*)
Riporta +1, –1, o 0, a seconda se l'argomento è positivo, negativo o zero.

SEN(*n*)
Riporta il valore del seno di un angolo specificato (in radianti).

STR(*n*,*tipo*,*cd*)
num:= *n*
tipo:= *n*
cd:= *n*

Converte un numero, *num*, all'equivalente stringa di testo. Tipo indica la forma della stringa convertita nel modo seguente:

0	decimale (formato)
1	esponenziale o notazione scientifica
2	intero **3** generale

Il terzo parametro,*cd*, specifica il numero di cifre decimali nella stringa convertita. Deve essere sempre incluso, sebbene il suo valore venga ignorato nella forma intera, nella forma generale e in quella monetaria.

QUAD(*n*)
Riporta la radice quadrata del numero *n*, che non deve essere negativo.

SOMMA(*serie*)
Nota che il valore riportato dalla funzione è la somma del valore esatto delle celle in oggetto. Non prende in considerazione nessun arrotondamento che possa risultare dall'uso del comando **Notazioni**. Per esempio, se due celle hanno i valori 3,44 e 9,73, la funzione **somma()** li addiziona riportando un valore di 13,17. Se scegli la visualizzazione a formato decimale con una cifra decimale, i due numeri vengono arrotondati a 3,4 e 9,7. La somma, il cui valore rimane ancora 13,17, verrà arrotondata al valore che appare inaccurato di 13,2. Vedi il comando **Notazioni**.

TAN(*n*)
Riporta il valore dell'angolo specificato in radianti.

ORA( )
Riporta, in valore di testo, l'orario del giorno in formato "OO:MM:SS". Devi prima aver impostato l'orologio del sistema, come descritto nella guida parole chiave del SuperBASIC.

VAL(*testo*)
Riporta il valore di *testo* al corrispodente valore numerico. Converte solo testo composto di caratteri numerici validi e la conversione si ferma al carattere che non può essere interpretato come cifre. Per esempio, **val("2.2ABC")** riporterà il valore 2,2, e **val("ABC")** riporterà il valore 0,0.

LARG( )
Riporta la larghezza, in caratteri, della colonna in uso. Nota che c'è uno spazio che separa le colonne adiacenti.

ERRORI

Errori del reticolo

Qualsiasi errore in una formula – come per esempio la scrittura di numero errato di argomenti di funzione, parentesi sbagliate – è riferito quando scrivi la formula. Vieni informato della natura dell'errore e la formula viene lasciata alla riga d'immissione. Quindi puoi esaminarla, e apportarne la correzione con l'editore di riga.

Una lista messaggi di errori che si possono incontrare appare qui di seguito:

Messaggi	Esempio
Virgolette mancanti dopo la parola	`"abc" + "def`
Costante numerica formata male	`1.5e` (numero mancante dopo "e")
Numero fuori serie	`1.5e99`
Carattere illegale	`12_5` (sottolinea invece del meno)
I nomi devono riferirsi a colonne Tutti i nomi devono riferirsi a righe I riferimenti dei nomi possono essere solo relativi (vedi sezione Rifcella, in questo capitolo)	
Riferimento di serie formato male	`a1:`
Riferimento di nome formato male	`c3.`
Il nome non una riga o colonna	(see Chapter 3)
Riferimento del primo nome non definito	
Riferimento del secondo nome non definito (il testo no si trova nel reticolo, sopra e alla sinistra di questa cella)	
La funzione richiede un riferimento serie	`tir(1,2,3)` (vedi descrizione di **tir()**)
Serie scorretta Errore di sintassi Parentesi sbagliate	
Errore di tipo	`1 + "abc"`
Numero errato di argomenti di funzione	`quad(1,2)`
Stringa più lunga di 255 caratteri	`ripro("*",256)`
Divisione per zero	
Argomenti della funzione illegali	`quad(-1)`
Deponenziale di stringa fuori serie (o deponenziale meno di zero o più grande di 255, oppure il primo deponenziale più grande del testo)	
Riferimento fuori serie (ad una cella fuori del reticolo)	
Riferimento ad una cella di errore (la formula si riferisce ad una cella che contiene una formula che produce uno degli errori descritti qui di seguito)	
Memoria satura VUOTA per far spazio	

Altri errori, come, per esempio, una formula che somma il contenuto di due altre celle, una che contiene un valore numerico e l'altra che contiene un valore di testo, non viene rilevato fino a quando il risultato della formula non viene calcolato – dopo che la formula è stata posizionata nel reticolo.

Se una formula contiene un riferimento ad una cella vuota, Abacus presume che la cella abbia un valore numerico di zero. Questo potrebbe essere la possibile causa di un errore quando la formula viene calcolata.

Se Abacus rileva un errore quando la formula viene calcolata, dà un breve messaggio di errore nella relativa cella. Puoi allora spostare il cursore a quella cella per esaminare la formula e verificare che cosa è sbagliato.

Gli errori che possono succedere sono:

 # #TIPO – la formula contiene un riferimento ad una cella in cui c'è informazione del tipo sbagliato, cioè numerica invece di testo, o viceversa.

 # #LUNGA – la formula contiene un riferimento ad una stringa di testo che è più lunga di 255 caratteri.

 # #ZERO – La formula sta cercando di dividere qualcosa per zero.

#ARG – La formula contiene una funzione chiamata con un valore non-valido per uno o più dei suoi argomenti es. ln(–5).

#SUB – La formula usa un operatore di divisione di stringa con un errore in uno o più dei suoi deponenti.

#RIF – La formula contiene un riferimento ad una cella che si trova fuori del reticolo. La formula in questa cella mostrerà la parola "ERRORE" per ogni rifcella che non valido.

#ERR – La formula contiene un riferimento ad una cella in cui c'è un errore. Questi messaggi possono essere ignorati dal momento che scompariranno quando l'errore originale, nella cella a cui si riferisce la formula, viene corretto.

Errori di file

I seguenti messaggi di errore si visualizzano solo mentre stai usando un comando relativo al file es. Richiama o File.

Il file non esiste
il nomefile che hai dato non può essere trovato nella cartuccia del Microdrive2.

I/E del file non completo
il richiamo o la memorizzazione di un file è iniziato poi ha incontrato degli ostacoli – questo potrebbe significare che i dati del file hanno subito un'avaria, oppure che la cartuccia è stata danneggiata

Impossibile aprire un file
Il file non può essere aperto – per una delle ragioni date per l'errore precedente.

Tipo di file sbagliato
l' estensione del nomefile non è quella che si aspettava Abacus – es. tentativo di richiamare l'esportazione di un file anzichè di importarlo.

Nome di file non valido
es. "3test" i nomi di file devono iniziare con un carattere alfabetico e non può essere più di 8 caratteri.

Formato di file da importare non valido
questo potrebbe verificarsi solo se tenti d'importare un file che non stato formato con il comando **esporta**.

CAPITOLO 1
QL ARCHIVE

QL Archive una base dei dati che ti permette di creare sistemi di archiviazione per qualsiasi tipo di informazioni da te prescelto. Sei libero di decidere come queste informazioni saranno immagazzinate e ricercate.

Scoprirai presto in che modo Archive può essere usato per creare dei semplici sistemi di indice su scheda, quali elenchi d'indirizzi o registrazioni dei clienti. Dopo aver imparato a creare dei sistemi semplici come questi, probabilmente desidererai sviluppare dei sistemi più complessi, dove le informazioni vengono condivise, per esempio, tra registrazioni di acquisto e controllo giacenze.

Le informazioni possono essere presentate usando il tracciato dello schermo fornito da Archive, oppure puoi disegnarne uno tuo. Le informazioni del file possono essere utilizzate per produrre moduli e rapporti stampati in qualsiasi formato da te prescelto.

Una delle caratteristiche più potenti di Archive è costituita dalla struttura dei suoi comandi. Una volta creato un file e immagazzinate delle registrazioni, questi comandi possono essere usati per trovare delle registrazioni particolari, fare delle ricerche e scelte o visualizzare le informazioni nel file in un ordine particolare.

I comandi si combinano per formare un potente linguaggio di programmazione simile al SuperBASIC, che può essere usato per molteplici impieghi specialistici.

Sarai costantemente guidato da una serie di messaggi d'informazione, che non ti lasciano mai in dubbio sulle tue opzioni e su che cosa fare. Se hai bisogno di informazioni ulteriori puoi usare i file d'Aiuto. Puoi chiedere aiuto in qualsiasi momento, qualsiasi cosa stia facendo, e ti verrà data automaticamente l'informazione più importante per i tuoi bisogni del momento.

La potenza reale di Archive si rivela quando scrivi le tue procedure nel linguaggio di comando. Puoi creare una procedura dotata di nome, che faccia esattamente quello che vuoi, e poi usarla come un comando addizionale, allo stesso modo in cui useresti i comandi forniti da Archive.

La meccanica di scrittura e modifica di un programma viene aiutata da un *editore di procedura* completo, che, assieme all'*editore della riga d'immissione* (che è sempre disponibile), rende l'edizione un lavoro semplice e non faticoso.

I file dei dati stessi usano campi e registrazioni di lunghezza variabile. Questo porta non solo ad un uso più efficiente della memoria e dello spazio nella cartuccia, ma semplifica anche la creazione di file. Non c'è mai bisogno di decidere in anticipo quanto dev'essere lungo un record.

Questo manuale contiene un certo numero di esempi operativi. Provali per vedere alcune delle cose che è possibile fare. Essi contengono molte procedure di uso generale che puoi inserire nei tuoi programmi.

Se in qualsiasi momento non sei sicuro di cosa fare, ricorda che che puoi chedere aiuto premendo **F1**. Ricorda anche che puoi annullare qualsiasi operazione parzialmente completata (per es. la scrittura di un numero, o l'uso di un comando) premendo **ESC**.

Archive è stato disegnato per darti la massima flessibilità possibile. Di conseguenza, esso non può dare tutta l'assistenza nella selezione delle opzioni che viene data dagli altri programmi QL. Se non conosci bene gli elaboratori e la programmazione, puoi trovare utile leggere la parte introduttiva al SuperBASIC del manuale prima di cercare di scrivere programmi con Archive.

CAPITOLO 2
PARTENZA

CARICAMENTO DI QL ARCHIVE

Il caricamento del QL Archive viene eseguito come descritto nell'introduzione dei programmi QL. Durante il caricamento, Archive presenta il seguente messaggio:

```
QL ARCHIVE
Archivio elettronico
Versione x.xx
Copyright ©1984 PSION Ltd.
Tutti i diritti riservati
```

in cui x.xx rappresenta il numero della versione, per es. 2.23.

Il programma attende alcuni secondi prima di iniziare. Premi qualsiasi tasto per cominciare immediatamente.

Le informazioni d'Aiuto non vengono richiamate nella memoria dell'elaboratore assieme al programma. Esse vengono lette dalla cartuccia di Archive quando ce n'è bisogno. **Perciò non devi togliere la cartuccia di Archive dal Microdrive 1 se intendi utilizzare il servizio di Aiuto.**

Dopo il caricamento di Archive, lo schermo deve avere l'aspetto della figura 2.1. Questo è lo *schermo principale.*

ASPETTO GENERALE

Se stai usando un apparecchio televisivo, lo schermo è disposto in un modo leggermente differente. Questo deriva dal fatto che, in genere, un televisore non è in grado di mostrare chiaramente 80 caratteri per riga. Perciò Archive visualizza solo 64 caratteri.

```
AIUTO                COMANDI   apri     guarda    visualizza   cerca      COMANDI
F1                   cancella  crea     chiudi    inserimento  lascia     F3
MESSAGGI             modifica  prox     primo     precedente   ultimo     USCITA
F2                   Scrivil il comando e ↵       (F3 per altri)          ESC

>
```

Figura 2.1 Lo schermo principale usando un monitor. (80 caratteri)

Lo schermo è suddiviso in tre sezioni: la zona di visualizzazione, la zona di lavoro e la zona dei comandi.

Le zone di visualizzazione e di lavoro

Come suggerito dal nome, in queste zone vengono mostrate tutte le informazioni prodotte da Archive.

La zona di lavoro usa le quattro righe inferiori dello schermo. Tutti i comandi da te scritti, assieme ai messaggi di errore, vengono visualizzati qui.

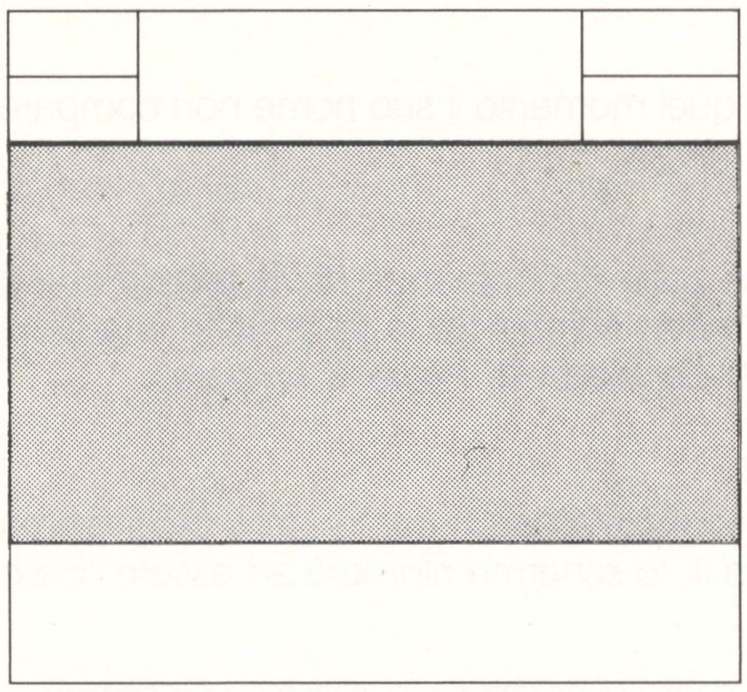


Figura 2.2 La zona di visualizzazione

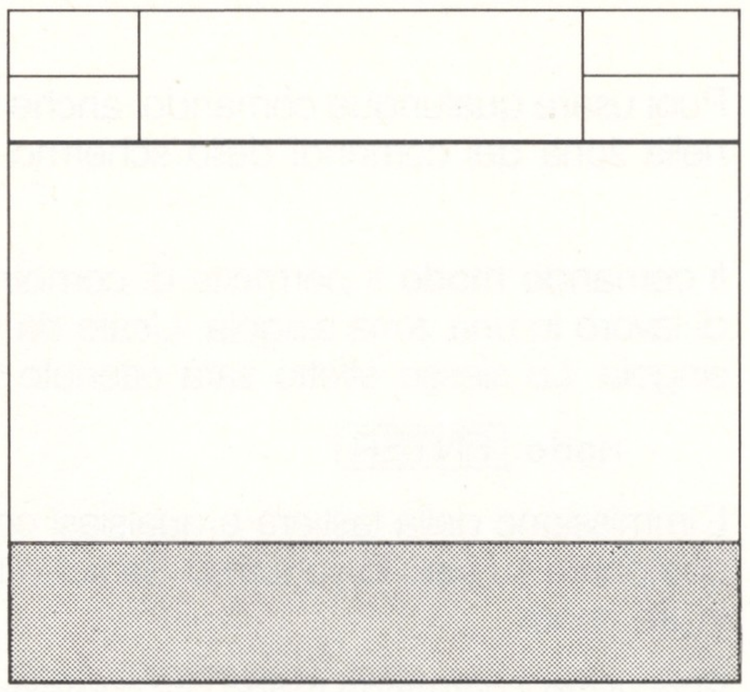


Figura 2.3 La zona di lavoro

Queste due zone lavorano quasi sempre assieme, poiché i comandi scritti nella zona di lavoro producono i loro risultati nella zona di visualizzazione.

Per esempio, scrivi il seguente breve programma, esattamente come viene riportato più sotto.

```
poni x=15:mentre x>0:scrivi x:poni x=x-1:finementre ENTER
```

Il testo di questo programma comparirà nella prima riga della zona di lavoro. Quando premi **ENTER**, verranno scritti, in righe successive della zona di visualizzazione, i numeri decrescenti da quindici a uno. La riga inferiore della zona di visualizzazione verrà lasciata vuota, eccetto che per un cursore rosso che indica la posizione successiva in cui il testo verrà visualizzato. Vengono visualizzati i numeri da quindici a uno che, assieme alla riga vuota inferiore, occupano tutte e sedici le righe della zona di visualizzazione (con lo schermo a 80 colonne).

Il comando:

```
pulisci ENTER
```

pulirà completamente la zona di visualizzazione.

La zona dei comandi

La zona dei comandi occupa le righe superiori dello schermo. Essa contiene le opzioni normali: AIUTO premendo **F1**, MESSAGGI, che vengono attivati e disattivati premendo **F2**, USCITA, per annullare operazioni incomplete premendo **ESC**, e impiego di un comando **F3**.

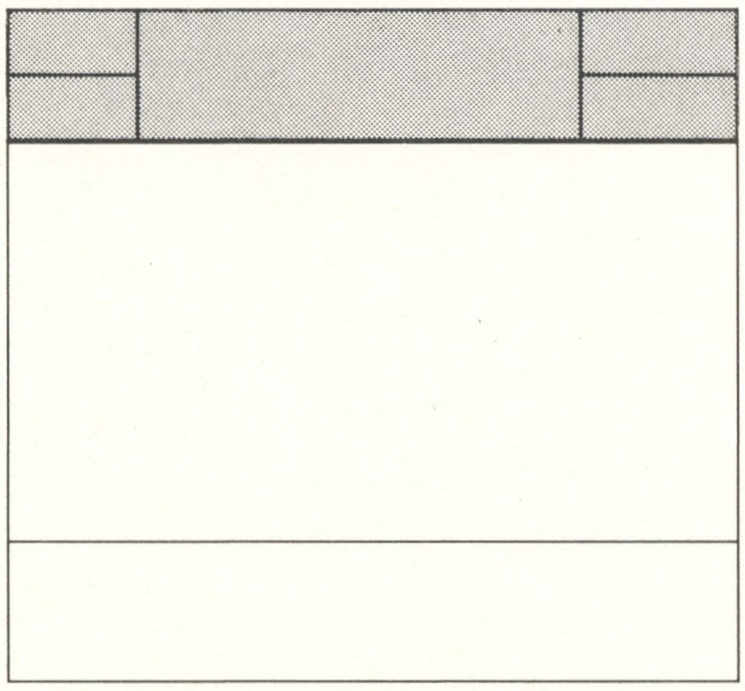


Figura 2.4 La zona dei comandi

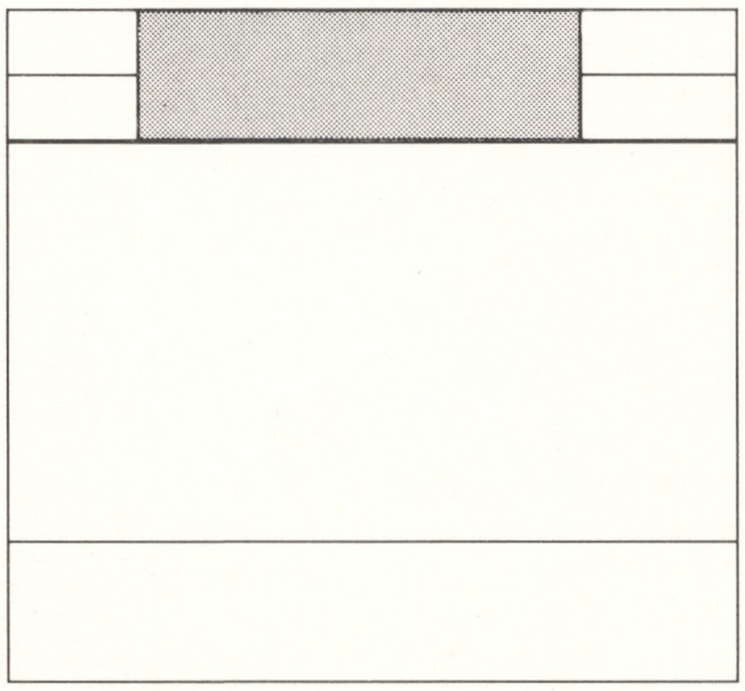


Figura 2.5 I comandi

IMPIEGO DEI COMANDI

I comandi di Archive formano un linguaggio di programmazione ed il loro nome deve essere completo. Dapprima questo può sembrare troppo lungo, ma più tardi vedrai come creare procedure che ti permettono di attivare i comandi battendo un solo tasto.

Premendo **F3** possono essere visualizzati quattro differenti elenchi di comandi. Se sullo schermo è già presente un elenco di comandi, premendo **F3** passerai all'elenco successivo. Questi comandi vengono usati scrivendo il loro nome e premendo **ENTER**. Comunque, alcuni comandi hanno bisogno di ulteriori informazioni e ti chiederanno di fornirle.

Puoi usare qualunque comando, anche se in quel momento il suo nome non compare nella zona dei comandi dello schermo.

IL COMANDO "MODO"

Il comando **modo** ti permette di combinare le zone dei comandi, di visualizzazione e di lavoro in una zona singola. Usato da solo, **modo** combinerà le tre zone in una zona singola. Lo stesso effetto sarà ottenuto scrivendo **modo 0**. Prova a scrivere

```
modo ENTER
```

L'immissione dalla tastiera e qualsiasi cosa visualizzata da un comando o programma comparirà su tutto lo schermo. Se scrivi la cifra 1, lo schermo ritornerà ad essere diviso in tre zone.

Puoi usare il comando **modo** per cambiare il numero di caratteri visualizzati sullo schermo. Per farlo, devi scrivere un secondo numero separato dal primo da una virgola. Il secondo numero deve essere un 4, 6 o 8 per selezionare uno schermo da 40, 64 o 80 colonne. Prova a scrivere

```
modo 0,4 ENTER
```

per cambiare la visualizzazione a 40 caratteri e per combinare tutte e tre le zone dello schermo. Nota che lo 0, che originariamente era opzionale, deve essere scritto per cambiare la dimensione della visualizzazione.

Cerca delle combinazioni differenti per vederne l'effetto sullo schermo. Finisci con un comando che lascia lo schermo diviso nelle tre zone, ma scegli il numero di caratteri che dia una chiara visione sul tuo televisore o monitor.

CAPITOLO 3

FILE DI QL ARCHIVE

RECORD E CAMPI DI FILE

Un file di Archive si comporta quasi come uno schedario, formato da una scatola contenente una serie schede, ognuna delle quali contiene varie informazioni. Affinché un tale schedario sia utile, ci devono essere regole per determinare dove si trovi ciascuna informazione scritta.

Immaginiamo per esempio di avere uno schedario organizzato per nomi ed indirizzi. Normalmente scriveresti il nome della persona in cima, seguito dall'indirizzo e dal numero di telefono (se esistente). Sarebbe molto difficile da usare se su alcune schede il nome fosse scritto in alto e in altre fosse scritto in basso. Normalmente ti aspetti di usare uno schedario dando una scorsa alle schede, leggendo solo la riga superiore, fino a che trovi il nome che cercavi.

Se tu avessi due serie di schede, per esempio una serie di record organizzati per nome ed indirizzo ed una serie di record organizzati per giacenze, non le terresti nello stesso schedario. Devi usare due schedari ed etichettarli, per esempio "Record clienti" e "Record giacenze".

Il sistema dello schedario contiene la maggior parte delle idee necessarie a capire come funziona un *file* di Archive. Un è file simile ad uno schedario e gli viene dato un nome per identificarlo. Il file è composto da una raccolta di *record*, ciascuno dei quali serve allo stesso scopo di una scheda. Per cui un file è semplicemente una raccolta di record collegati.

Le informazioni contenute in ogni record sono organizzate in modo regolare, esattamente come in uno schedario. Dati individuali, per esempio numeri di telefono, possono essere tenuti in una zona specifica della scheda. Un record in un file di Archive è organizzato allo stesso modo. Ogni dato viene immagazzinato in una regione distinta del record, conosciuta come il *campo*. Un record in un file del cliente, come quello descritto sopra, conterrebbe un campo nome, un campo indirizzo, un campo sconto ecc.

Se questo fosse tutto, non varrebbbe la pena di usare un file di dati di Archive, al posto di uno schedario fisico. Comunque, l'uso di record elettronici presenta molti vantaggi. Normalmente uno schedario clienti è organizzato in ordine alfabetico secondo i nomi dei clienti, e questo lo rende efficiente per trovare le informazioni su un cliente specifico. Comunque, immagina di voler inviare una lettera a tutti i clienti che non hanno fatto ordini nel corso degli ultimi sei mesi. Sarebbe molto noioso esaminare tutte le schede per compilare tale elenco. Con Archive è possibile fare una tale ricerca usando pochi semplici comandi. Inoltre, è facile dare i comandi per stampare contemporaneamente una serie di etichette.

Puoi risparmiare molto tempo e fatica se usi Archive per immagazzinare e manipolare i tuoi record.

CAPITOLO 4
ESAME DI UN FILE

Il modo migliore per imparare a usare Archive è quello di leggere il file di dimostrazione **"gazzetta"**, fornito nella cartuccia di Archive. Questo file contiene informazioni su vari paesi – il continente, la capitale, la valuta, la lingua, la popolazione, la superficie e il prodotto nazionale lordo pro capite.

La maggior parte degli esempi dei capitoli 4 e 5 si riferiscono al file "gazzetta". Prima di usarlo, fanne una copia usando la seguente procedura:

Una volta caricato Archive, metti una cartuccia formattata nel Microdrive 2 e scrivi:

```
backup                  ENTER
mdv1_gazzetta_dbf       ENTER
mdv2_gazzetta_dbf       ENTER
```

Aspetta fino a che i due Microdrive si siano fermati; ci vuole un po' di pazienza in quanto il file lungo e ci vuole un po' di tempo a copiarlo. Usa la copia, che ora si trova nella cartuccia nel Microdrive 2, per fare delle prove.

Da ora in avanti non troverai scritto **ENTER** alla fine di tutti i comandi, ma ricordati che deve essere sempre usato.

Il comando **guarda** apre un file in modo da lasciartene leggere il contenuto, ma non potrai fare modifiche o aggiunte al file. E' un comando più sicuro di **apri** se stai semplicemente consultando un file, in quanto esso protegge il file da variazioni accidentali. Puoi esaminare la copia del file "gazzetta" del Microdrive 2 scrivendo:

```
guarda "gazzetta"
```

VISUALIZZAZIONE DI UN RECORD

Per guardare il primo record, scrivi:

```
primo
visualizza
```

Non dimenticare di premere **ENTER** dopo ogni comando. A questo punto sullo schermo comparirà il primo record del file.

Nota che la prima riga indica il nome logico del file: Archive fornisce automaticamente il nome "principale" per un file unico. Di solito i nomi di file logici vengono usati quando usi contemporaneamente più di un file ed essi vengono descritti più avanti.

ESAME DI ALTRI RECORD

Dopo aver esaminato il primo record del file, vorrai probabilmente passare al prossimo record. Scrivi:

```
prox
```

e lo schermo mostra il record successivo del file. Quando scrivi comandi unici dopo un comando **visualizza**, la zona di visualizzazione viene aggiornata continuamente per mostrare il contenuto del record corrente. Puoi usare il comando **prox** per esaminare tutto il file, record dopo record, fino a che raggiungi la fine (esso non supererà l'ultimo record).

Esistono altri tre comandi che puoi usare per controllare quale record del file viene visualizzato.

```
precedente  – che visualizza il record precedente,
primo       – che visualizza il primo record,
ultimo      – che mostra l'ultimo record del file.
```

Cerca di usare questi comandi per spostarti nel file, visualizzando tutti i record che vuoi. Nota che i quattro comandi **primo, ultimo, prox** e **precedente** non visualizzano il record di per sé. Essi semplicemente si spostano di record in record, indipendentemente dal fatto che tu abbia usato il comando **visualizza**

RICERCA IN UN FILE

Cerca

Il comando di ricerca più semplice è **cerca**. Esso cercherà dall'inizio di un file la prima volta che si incontra il testo specificato in un qualsiasi campo del file. Per esempio:

```
cerca "africa"
```

Quando premi **ENTER** ci sarà una breve pausa e quindi verrà visualizzato il primo record che contiene la parola "africa" in uno qualsiasi dei suoi campi di testo. Nota che questa ricerca non tiene alcun conto del fatto che il testo sia in lettere maiuscole o minuscole, e perciò cercherà "Africa", "AFRICA" o "africa".

Continua

Se il primo record trovato che contiene il testo non quello che vuoi, puoi cercare la successiva occorrenza scrivendo:

```
continua
```

Il comando **continua** ripeterà la ricerca precedente, cercando la successiva occorrenza del testo in un qualsiasi campo dei record successivi.

E' possibile che tu debba ripetere una ricerca parecchie volte prima di trovare il record che desideri. Premi **F5** ed Archive metterà il comando precedente nella riga di comando. Premi **ENTER** ed il comando verrà eseguito.

Ricerca

Un altro sistema per localizzare un particolare record è quello di usare il comando **ricerca**. Esso ti permette di trovare un record, specificando il contenuto di uno o più campi specifici, per esempio:

```
ricerca continente$= "EUROPA" e lingue$="ITALIANO"
```

cercherà il primo record del file che soddisfa entrambe le condizioni. Devi scrivere il nome del campo per intero.

A differenza di **cerca**, il comando **ricerca** cercherà solo nei campi da te specificati e distinguerà tra maiuscole e minuscole. Usa le funzioni **maiusc( )** o **minusc( )** per rendere indipendente la ricerca dallla scrittura, per esempio:

```
ricerca minusc(continente$)="europa"
```

Anche qui il comando **continua** può essere usato per cercare la prossima volta che si incontra il testo.

Seleziona

In molti casi è possibile che tu voglia guardare un sottogruppo dei record nel file. Per esempio, supponiamo che tu voglia guardare solo i particolari che si riferiscono a paesi europei. Puoi usare il comando **seleziona** per scegliere dal file tutti i record che soddisfano una certa condizione. Il file si comporterà poi come se fossero presenti solo i record selezionati. Prova questo comando con il file "gazzetta". Per prima cosa scrivi:

```
scrivi conta()
```

che ti dirà quanti record sono presenti nel file. Poi scrivi:

```
seleziona continente$="EUROPA"
scrivi conta()
```

E vedrai quanti record sono stati selezionati. I record tolti dal file sono ancora conservati nella memoria e puoi ripristinarli nel file in qualsiasi momento usando il comando **ripris**. Scrivi:

```
ripris
```

e scrivi nuovamente il valore di **conta( )**, per controllare che il file sia stato ripristinato nel suo stato originario.

Quando usi il comando **scrivi** dalla tastiera, il record presente sullo schermo verrà cancellato. Questo perché, di solito, **visualizza** e **scrivi** usano le stesse zone dello schermo. Dopo aver usato **scrivi** devi digitare nuovamente **visualizza** per ripristinare la visualizzazione.

ORDINAMENTO DI UN FILE

E' possibile che i record del file non siano sempre nell'ordine di cui hai bisogno. Puoi ordinare il file per contenuto dei campi numerici o alfabetici. Il comando **ordina** prende in considerazione solo i primi otto caratteri di testo.

Per esempio, immaginiamo che tu voglia fare l'ordinamento dei record di "gazzetta" secondo l'ordine alfabetico della capitale. Puoi farlo usando nel seguente modo il comando **ordina**:

```
ordina capitale$;a
```

La "a" che segue il punto e virgola specifica che vuoi fare l'ordinamento del file in ordine ascendente. Se vuoi che il file venga riordinato in ordine discendente devi scrivere una "d". Il campo capitale$ diventa la **chiave di ordinamento** per quel file. Puoi specificare una chiave di ordinamento composta da un massimo di quattro campi, dando un elenco di campi dopo il comando **ordina**. Per ciascuna di queste chiavi, devi specificare se l'ordinamento deve essere in ordine ascendente o discendente. Per esempio, il comando seguente ordinerà il file in ordine discendente per popolazione e ordine ascendente per capitale.

```
ordina pop;d,capitale$;a
```

Nota che il nome del campo è separato dalla "a" o "d" che specifica l'ordine ascendente o discendente, da un punto e virgola, ma che ogni coppia (nome di campo e lettera) è separata dalla successiva da una virgola.

Quando si esegue l'ordinamento di più di un campo, i record vengono ordinati a cominciare dal contenuto del primo campo dell'elenco. Se ci sono due o più record che hanno lo stesso contenuto in questo campo, essi vengono ordinati secondo il prossimo campo dell'elenco. Se ci sono dei record uguali per quel che riguarda il contenuto di questi due campi, essi vengono ordinati secodo il contenuto del terzo campo, ecc.

TROVA

Quando un file è stato ordinato, puoi usare il comando **trova** affinché un record specifico diventi quello corrente del file. Esso trova il primo record il cui primo campo d'ordinamento sia maggiore o uguale all'espressione data. Questo record diventa quello corrente del file.

Per esempio, se il file "gazzetta" è stato ordinato come descritto nell'ultimo esempio, il comando:

```
trova "100"
```

trova il primo paese del file ordinato che abbia una popolazione di 100 milioni. Nel caso che tale paese non esista, Archive trova il primo paese la cui popolazione sia inferiore a 100 milioni (ricorda che il file stato ordinato in ordine discendente).

Trova è seguito da una *espressione* che può essere alfabetica o numerica, ma dev'essere dello stesso tipo del campo usato per ordinare il file. (Vedi il capitolo *Consultazione*).

Puoi trovare un record, che si riferisca al contenuto di più di un campo di ordinamento, usando **trova** con espressioni multiple, separate da virgole. Per esempio,

```
poni a="100"
poni b$="D"
trova a,b$
```

trova il primo paese la cui popolazione, uguale o inferiore a 100 milioni, e la cui capitale inizia con "D" o segue la "D" nell'alfabeto. In questo esempio, Archive troverà il Bangladesh, che ha una popolazione di 76,1 milioni di abitanti e la cui capitale Dacca.

La sola limitazione al numero di espressioni che puoi usare con **trova**, costituita dal numero di campi usati per ordinare il file.

Dopo **trova**, non puoi usare **continua**. Se ripeti un **trova** con la stessa condizione, trovi sempre lo stesso record.

Trova, il modo più rapido di cercare un record in un file ordinato. Poiché esiste sempre una certa insicurezza del record trovato, è possibile che tu debba fare un ulteriore controllo del record per accertarti che sia quello che vuoi.

CHIUSURA DI UN FILE

Quando hai finito di guardare un file, devi dirlo ad Archive. Per farlo scrivi

```
chiudi
```

Questo avrà effetto solo sui file di dati, e lascia intatti programmi e aspetto dello schermo. Puoi chiudere tutti i tuoi file, annullare i dati e ripulire la zona di visualizzazione scrivendo

```
nuovo
```

Questo riporterà Archive al suo stato iniziale, immediatamente successivo al caricamento.

Oppure, se hai finito di usare Archive, puoi ritornare a SuperBASIC usando **lascia**. Questo comando chiude automaticamente tutti i file aperti prima di uscire da Archive.

Ricorda che non devi mai togliere la cartuccia dal Microdrive mentre esso contiene dei file aperti.

CAPITOLO 5
MODIFICA DI UN FILE

Prima di scrivere gli esempi di questo capitolo. scrivi **nuovo** per essere sicuro che Archive sia pulito e pronto per un nuovo inizio.

Il comando **apri** prepara un file sia per la lettura che per la scrittura.

Se apri un file con il comando **apri**, invece di **guarda**, sarai in grado di scrivere sul file per cambiarne il contenuto, oltre che di leggerlo. Questo significa che qualsiasi aggiunta, cancellazione o modifica farà un cambiamento permanente alla copia del file quando verrà chiuso. Scrivi:

```
apri "gazzetta"
```

Se hai aperto un file per sola lettura con **guarda**, non devi usare alcun comando che tenti di modificare i dati. Se lo fai, Archive darà un messaggio di errore. I comandi descritti in questo capitolo modificano i file di dati, per cui devono essere usati soltanto con un file aperto con **apri**.

Visualizza il primo record del file con:

```
primo
visualizza
```

CHIUSURA DEL FILE

Quando hai finito di modificare il file, devi chiuderlo, (usando il comando **chiudi** o **nuovo**) per essere sicuro che tutti i cambiamenti siano registrati.

Se non chiudi correttamente un file (per esempio, se spegni l'elaboratore quando hai finito) è possibile che sul file non siano state registrate le modifiche più recenti. **Accertati sempre che non ci siano file aperti su una cartuccia prima di toglierla dal Microdrive. Non spegnere l'elaboratore senza aver prima chiuso tutti i file aperti e tolto le cartucce dal Microdrive.**

INSERIMENTO

Il comando **inserimento** viene usato per aggiungere uno o più record al file corrente. Quando usi **inserimento** ti viene chiesto di scrivere il contenuto di ciascun campo del nuovo record. Scrivi:

```
inserimento
```

La zona di visualizzazione ora mostra:

```
Nome logico      : principale
stato$           :
continente$      :
capitale$        :
moneta$          :
lingue$          :
pop              :
superfice        :
pnl              :
```

Ora puoi scrivere il contenuto di ogni campo. Puoi passare da un campo all'altro premendo **ENTER** o **TABULATE**, o puoi ritornare al campo precedente tenendo premuto **SHIFT** e premendo **TABULATE**. Puoi fare tutti i cambiamenti che desideri ai campi finché sei soddisfatto. Il nuovo record può essere inserito nel file premendo **F5**. Premi **F4** per uscire da **inserimento**. Prova a scrivere:

SCOZIA	TABULATE
EUROPA	TABULATE
EDIMBURGO	TABULATE
STERLINA	TABULATE
INGLESE	TABULATE
10	TABULATE
30	TABULATE
50	TABULATE

Ora la zona di visualizzazione deve contenere:

```
Nome logico          : principale
stato$               : SCOZIA
continente$          : EUROPA
capitale$            : EDIMBURGO
moneta$              : STERLINA
lingue$              : INGLESE
pop                  : 10
superfice            : 30
pnl                  : 50
```

Una volta che sei sicuro di aver scritto correttamente le nuove informazioni, premi **F5** per inserire il nuovo record nel file. I campi che hai appena scritti verranno svuotati, e saranno pronti per l'inserimento di un nuovo record. Premi **F4** quando hai finito di fare l'inserimento.

Puoi anche finire l'immissione di ciascun campo e passare al successivo premendo **ENTER**. Il nuovo record viene aggiunto automaticamente al file quando premi **ENTER** dopo l'ultimo valore.

Se il file è stato ordinato, il nuovo record viene inserito nella posizione corretta per mantenere l'ordine.

CANCELLA

Puoi usare il comando **cancella** per togliere un record dal file. **Cancella** toglie dal file il record corrente (quello mostrato con **visualizza**). Per togliere uno specifico record, devi semplicemente visualizzarlo e, dopo esserti accertato che sia quello giusto, scrivere:

```
cancella
```

VARIAZIONE DI UN RECORD

Anche la modifica del contenuto di uno o tutti i campi di un record esistente è una procedura facile. Ci sono due sistemi.

Modifica

Scegli il record che vuoi cambiare (usando **visualizza** e **cerca**) e poi scrivi **modifica. Modifica** opera allo stesso modo di **inserimento**, eccetto che ogni campo riporta il vecchio contenuto. Puoi oltrepassare i campi che non vuoi cambiare (usa **TABULATE** o **ENTER**). Scrivi il nuovo valore o usa i tasti del cursore per modificarne uno vecchio. Premi **F5** per sostituire il record.

Come nel caso di **inserimento**, il record viene sostituito automaticamente se premi **ENTER** dopo l'ultimo campo del record.

Aggiorna

Scegli il record che vuoi variare e poi cambia il contenuto delle variabili di campo fino a che il record visualizzato corrisponda alle necessità. Scrivi **aggiorna** per cambiare il record.

Per esempio, immaginiamo che tu decida che l'Islanda debba essere in Europa invece che nell'Artico. Cerca il record scrivendo

```
cerca "Islanda"
visualizza
```

Usa il comando poni per cambiare il contenuto del campo **continente$**:

```
poni continente$ = "EUROPA"
```

Per finire memorizza questa variazione nel record scrivendo **aggiorna**.

Con entrambi i sistemi il nuovo record verrà inserito nella posizione corretta se il file è stato ordinato. Altrimenti il record sostitutivo viene inserito in una qualsiasi posizione nel file.

Il comando **modifica** è più semplice da usare, ma influisce sempre sul record corrente. Il comando **aggiorna** può essere utile quando stai usando file multipli.

Ricorda che devi chiudere il file con comandi **chiudi**, **nuovo** o **lascia**, prima di spegnere l'elaboratore.

CAPITOLO 6
CREAZIONE DI UN FILE

Se hai seguito gli esempi fino a questo punto, avrai usato Archive solo per esaminare il file che era stato preparato per te. Questo capitolo ti mostrerà come creare un tuo file e sarai tu a scegliere i nomi di file.

Se necessario, scrivi **nuovo** per pulire la memoria dell'elaboratore e per chiudere i file aperti. Accertati che nel Microdrive 2 si trovi la cartuccia formattata su cui saranno memorizzati i file.

Immaginiamo che tu voglia usare ARCHIVE per fare un catalogo dei tuoi libri. Per farlo, dovrai creare un nuovo file chiamato "libri". Nel creare un file, la prima cosa da fare è decidere quello che conterrà, cioè, quali campi userai in ogni record. In questo caso, naturalmente avrai bisogno di registrare l'autore, il titolo e il soggetto; forse desideri anche includere altri particolari, quali il tipo (narrativa o non narrativa), ISBN (International Standard Book Number), posizione dello scaffale, una breve descrizione ecc. In questo esempio useremo solo tre campi di testo che conterranno l'autore, il titolo ed il soggetto, ed un campo numerico che verrà usato per contenere l'ISBN.

CREA

Il file viene creato usando il comando **crea**. Devi precisare il nome del file da creare ed i nomi dei campi da usare in ciascun record. Il segno **$** indica che il campo contiene del testo. Quando avrai finito di definire i campi del record, termina il comando **crea** con **finecrea**. Puoi creare il semplice file di catalogo libri descritto più sopra con la seguente sequenza.

```
crea "libri"
autore$
titolo$
soggetto$
isbn
finecrea
```

Nota che non devi scrivere il comando **finecrea** finale. Se vuoi puoi farlo, ma puoi terminare la creazione del file semplicemente premendo **ENTER** su una linea di immissione vuota. Se usi **crea** in un programma di Archive, devi usare **finecrea**.

AGGIUNTA DI RECORD

Una volta che hai creato un file, esso è aperto sia per la lettura che per la scrittura, ma non contiene alcun record. I record possono essere aggiunti usando **inserimento**. Scrivi:

```
inserimento
```

e nella zona di visualizzazione comparirà:

```
nome logico   : principale
autore$       :
titolo$       :
soggetto$     :
isbn          : 0
```

Dovrai semplicemente scrivere il contenuto di ciascun campo. Per esempio, scrivi:

```
Blocchi, G                TABULATE
Un manuale pesante        TABULATE
Costruzione cannoni       TABULATE
1234567
```

e nella zona di visualizzazione deve comparire

```
Nome logico   : principale
autore$       : Blocchi, G
titolo$       : Un manuale pesante
soggetto$     : Costruzione cannoni
isbn          : 1234567
```

Inserisci il record nel file "libra" premendo **F5**. I campi si svuoteranno e saranno pronti per inserire un altro record.

Ricorda che puoi terminare l'immissione di ciascun campo e passare al successivo premendo **ENTER** e che premendo **ENTER** dopo l'ultimo valore aggiungi il record al file.

Una volta che hai finito premi **F1** e ricordati di usare **chiudi** o **lascia**.

CAPITOLO 7
TRACCIATI DELLO SCHERMO

Quando usi il comando **visualizza** con un file creato da te, i record vengono mostrati con il tracciato normale di schermo di Archive.

DEFINIZIONE DEL TRACCIATO DELLO SCHERMO

Puoi disegnare un tracciato dello schermo, che sia più adatto alle informazioni del tuo file di dati. **Apri** un file esistente e scrivi:

```
visualizza
```

L'edizione di schermo viene attivata con il comando **edisch**. Scrivi:

```
edisch
```

La zona di visualizzazione mostra il tracciato dello schermo, che sarà quello creato automaticamente da Archive. Se nella memoria dell'elaboratore non c'è alcun tracciato, la zona di visualizzazione può essere vuota.

Noterai che i valori dei campi dei file non vengono indicati. Gli spazi dove normalmente vengono riportati questi valori sono marcati da righe di puntini. Devi pensare al tracciato dello schermo come ad uno sfondo in cui i valori delle variabili vengono mostrati in specifiche posizioni. Archive mostra il tracciato dello schermo in due fasi – prima traccia il testo dello sfondo e poi mostra i valori delle variabili nelle posizioni marcate sullo schermo.

Inizialmente sei al *livello principale* del comando e hai tre opzioni:

scrivere il testo di sfondo sullo schermo
premere **ESC** per uscire da **edisch**
premere **F3** per usare un comando di edizione di schermo

Per disegnare un tracciato di schermo, premi **F3** e poi **P** per pulire lo schermo. Premi **ENTER** per confermare la scelta; qualsiasi altro tasto ti rimanderà al livello principale di **edisch**.

Scegli i colori della carta e dell'inchiostro premendo **C** o **I** e premendo qualsiasi tasto per passare da un colore all'altro tra i quattro disponibili. Premi **ENTER** per ritornare al livello principale per scrivere il testo dello sfondo.

Il testo di sfondo potrebbe essere una spiegazione, per esempio:

L'informatore mondiale di Andrew Young

o potrebbe consistere di un nuovo nome per uno dei campi del tuo file:

Popolazione (milioni):

Puoi portare il cursore in qualsiasi punto della zona di visualizzazione usando i quattro tasti del cursore. Qualsiasi cosa da te scritta comparirà immediatamente nella zona di visualizzazione nella posizione del cursore e diventerà parte dello sfondo del tracciato. La sola eccezione si ha quando il cursore si trova nella zona dello schermo riservata alla visualizzazione di una variabile. Archive riporta il nome della variabile nella zona di lavoro alla base dello schermo. E' impossibile scrivere del testo in questa zona senza averla prima liberata, come descritto più avanti.

I quattro comandi di edizione dello schermo ti permettono di produrre la visualizzazione dei dati in formati attraenti e colorati. E' già stato spiegato come fare a pulire lo schermo. Per padroneggiare a fondo gli altri tre dovrai fare delle prove, per cui accertati di usare una copia "sacrificabile" dei tuoi file di dati.

COMANDI DELL'EDITORE DI SCHERMO

Poni Variabile (V)

Immagina di voler far apparire il valore dellla variabile stato$ in una particolare posizione sullo schermo. Porta il cursore in quel punto e premi **F3** e poi il tasto **V**. Archive ti chiede di scrivere il nome della variabile. Scrivi:

```
stato$
```

Nota che questo nome non compare sullo schermo. Stai soltanto marcando il punto in cui deve comparire il *valore*. Quando premi **ENTER**, Archive ti chiede di indicare quanto spazio deve essere riservato per l'indicazione del valore. Premi qualsiasi tasto eccetto **ENTER** per marcare lo spazio con una serie di puntini. Per cancellare lo spazio riservato vengono utilizzati **CTRL** ed il tasto di movimento verso sinistra del cursore. Dopo aver riservato abbastanza spazio premi **ENTER** ed Archive ti riporta al livello principale di **edisch**.

Se porti il cursore in una delle zone riservate, (segnate con puntini), Archive ti indica il nome della variabile per cui è riservato spazio nella zona di lavoro.

Se riservi spazio ad una variabile in una regione che si sovrappone a una zona che sia già riservata, ti viene data l'opzione di cancellare la vecchia zona. A questo punto puoi usare nuovamente l'opzione per riservare spazio ad una nuova variabile.

Inchiostro (I)

Immagina di voler cambiare il colore dell'inchiostro. Porta il cursore nel punto dove vuoi che cominci il nuovo colore e premi **F3** e poi il tasto **I**. Archive elenca i quattro colori disponibili nella zona di lavoro. Quello selezionato sarà evidenziato. Premi un tasto qualsiasi per cambiare il colore prescelto e poi premi **ENTER** per registrare la scelta. Eventuale testo che scriverai successivamente, comparirà nel nuovo colore fino a che userai nuovamente il comando **inchiostro**.

Carta (C)

Per cambiare il colore della carta devi procedere allo stesso modo premendo **F3** e poi il tasto **C**.

Se vuoi che un cambiamento di colore influisca solo su una parte della riga, devi portare il cursore all'inizio della regione e scegliere i colori di carta e inchiostro di cui hai bisogno. Poi porta il cursore alla fine della regione e fai una seconda scelta dei colori di carta e inchiostro, ripristinandoli ai loro valori originari.

ATTIVAZIONE DI UN TRACCIATO DI SCHERMO

Una volta che tu hai disegnato un tracciato di schermo e sia uscito da **edisch**, il tracciato dello schermo sarà *attivo*. Questo significa che i valori di tutte le variabili del tracciato di schermo verranno visualizzate automaticamente ogni volta che Archive completa un comando o un programma. Se, per esempio, scrivi il comando **prox**, Archive passa al record successivo del file corrente e mostra i campi compresi nel tracciato dello schermo. Lo schermo attivo viene disattivato quando usi il comando **chiudi**.

Se il tracciato dello schermo non è attivo, puoi attivarlo con il comando **schermo**. Questo mostra il testo di sfondo del tracciato dello schermo, ma non mostra i valori correnti delle variabili.

MEMORIZZAZIONE E RICHIAMO DI SCHERMI

Puoi memorizzare il disegno dello schermo su cartuccia di Microdrive usando il comando **memosch**:

```
memosch "nomefile"
```

in cui "nomefile" un nome di tua scelta. Il tracciato dello schermo viene memorizzato esattamente con il suo aspetto.

Puoi richiamare il tracciato dello schermo scrivendo il comando:

```
risch "nomefile"
```

Quando richiami un tracciato di schermo, esso viene visualizzato automaticamente sullo schermo e reso attivo.

Archive non aggiornerà automaticamente un tracciato di schermo attivo dall'interno di un programma. Immagina di voler mostrare tutti i record del file corrente, uno dopo l'altro, e di aver cercato di farlo scrivendo il programma di una riga:

```
primo: poni x=0: mentre x<conta():prox:poni x=x+1:finementre
```

(i comandi **mentre** e **finementre** provocano l'esecuzione ripetuta della sezione di programma compresa tra di essi, finchè la condizione che segue **mentre** è vera. Per una esecuzione corretta, ogni comando **mentre** deve avere un **finementre** corrispondente.)

Questo programma non farebbe quello che vuoi, in quanto Archive aggiorna il contenuto del tracciato dello schermo alla fine del programma.

IL COMANDO MOSTRASCH

Comunque, puoi forzare la visualizzazione dei valori delle variabili in uno schermo attivo dall'interno di un programma, usando il comando **mostrasch**. Il seguente programma di una riga mostrerà i comandi richiesti.

```
primo:poni x=0:mentre x<conta():mostrasch:prox:
    poni x=x+1:finementre
```

Se non c'è schermo attivo, **mostrasch** non ha alcun effetto.

IL COMANDO VISUALIZZA

Ricorda che il comando **visualizza** usa il tracciato normale. Esso sostituirà sempre un tracciato di schermo con il suo semplice elenco dei campi del record corrente del file corrente. Devi perciò **memorizzare** il tracciato dello schermo prima di usare nuovamente **visualizza**. In caso contrario, il tracciato dello schermo verrà sostituito e potrai riaverlo solo se lo disegni nuovamente con **edisch**.

CAPITOLO 8
PROCEDURE

Per usare gli esempi di questo capitolo, per prima cosa scrivi **nuovo** per pulire l'elaboratore, poi scrivi **guarda "gazzetta"** per aprire il file d'esempio sulla tua cartuccia dei dati, che si presume sia nel Microdrive 2.

I comandi e le funzioni di Archive, formano assieme un linguaggio di programmazione che puoi usare per scrivere programmi che manipoleranno i tuoi file. Troverai che i programmi di Archive sono facili da scrivere.

Un programma di Archive è composto da una o più sezioni separate. Ogni sezione è conosciuta come una *procedura*, che è semplicemente una sezione di programma dotata di nome. Puoi riferirti ad una procedura con il suo nome, come le procedure che scrivi e usi in SuperBASIC. In Archive puoi eseguire una procedura scrivendo il suo nome sulla tastiera. Quando scrivi una procedura stai, in effetti, aggiungendo un nuovo comando ad Archive.

Nessuna procedura può contenere più di 255 rige, e ogni riga non deve contenere più di 160 caratteri.

CREAZIONE DI UNA PROCEDURA

Usi l'*editore di programma* tutte le volte che vuoi scrivere o cambiare una procedura. Questo editore ti permette di fare cambiamenti, cancellazioni o aggiunte al testo delle procedure.

L'editore di programma viene descritto nei particolari nel Capitolo 9, ma in questo capitolo osserveremo in breve alcune delle sue caratteristiche, in modo da poter scrivere alcune brevi procedure. Presumeremo che nella memoria dell'elaboratore non siano presenti procedure.

Scrivi:

```
edita
```

per entrare nell'editore di programma. La zona dei comandi cambia, ed indicherà che devi scrivere il nome della procedura. Quando entri nell'editore, potrai sempre creare una nuova procedura se non ne è stata definita o richiamata nessuna.

Perciò, la prima cosa da fare è di decidere cosa deve fare la nuova procedura. Iniziamo con un compito molto semplice; facilitarci la vita ribattezzando il comando **visualizza**. Risparmieremo tempo di scrittura se gli diamo il nome "v".

Scrivi semplicemente

```
v
```

Ora nel lato sinistro della zona di visualizzazione c'è il nome, e nel lato destro c'è un listato della procedura. La procedura, per il momento, non contiene alcun comando; il **proc** e **fineproc**, che marcano l'inizio e la fine della procedura, sono stati aggiunti automaticamente da Archive.

Bisogna aggiungere il *corpo* della procedura; cioè la sequenza di azioni che deve eseguire.

La zona dei comandi indica che puoi aggiungere righe di testo alla nuova procedura. Nei termini dell'esempio attuale questo testo costituito dal comando **visualizza**. Scrivi:

```
visualizza
```

ed Archive inserirà il nuovo testo nella procedura, sotto la riga evidenziata. Se hai seguito questo esempio lo schermo conterrà:

```
v   proc v
    visualizza
    fineproc
```

potresti aggiungere altre righe di testo – ogni riga verrebbe inserita sotto quella evidenziata.

Comunque, in questo caso la procedura è completa, per cui puoi lasciare il comando **edita** premendo due volte **ESC**.

Per usare questta procedura devi soltanto scrivere il suo nome, seguito da **ENTER**. Questa nuova procedura eseguirà la stessa funzione che se scrivessi il comando **visualizza** per intero.

LISTA E STAMPA DI PROCEDURE

Quando chiami il comando **edita**, ti viene mostrato un elenco dei nomi di tutte le procedure definite, presenti nella memoria dell'elaboratore.

Puoi elencare qualsiasi procedura dall'interno di **edita** premendo il tasto **TABULATE** per spostarti lungo l'elenco verso il basso o i tasti **SHIFT** e **TABULATE** assieme per muoverti nell'elenco verso l'alto, finché viene evidenziato quel particolare nome di procedura. La procedura viene inserita automaticamente nel lato destro dello schermo. Se la procedura troppo lunga per essere contenuta nella zona di visualizzazione, verrà mostrata la prima parte e potrai quindi far scorrere la procedura verso l'alto o il basso usando i tasti di movimento verticale del cursore. Una volta finito, puoi uscire dal comando **edita** premendo **ESC**.

Se vuoi un listato stampato delle procedure puoi usare il comando **lista**. Scrivi:

```
lista
```

e tutte le procedure attualmente nella memoria dell'elaboratore verranno listate su una stampante.

ATTENZIONE: Non usare questo comando se non c' una stampante collegata, in quanto questo provocherà il "blocco" del programma.

PROCEDURE DI MEMORIZZAZIONE E RICHIAMO

Se vuoi conservare le procedure che hai definito, puoi usare il comando **memorizza**. Questo immagazzina tutte le procedure definite in un file unico dotato di nome nella cartuccia del Microdrive. Se vuoi memorizzare le nuove procedure di visualizzazione, che hai appena definito, in un file chiamato "mieproc", devi scrivere

```
memorizza "mieproc"
```

Successivamente puoi riportare queste procedure nella memoria dell'elaboratore scrivendo:

```
richiama "mieproc"
```

Il comando **richiama** cancella le procedure esistenti in memoria prima di caricare quelle nuove dalla cartuccia del Microdrive. Se vuoi aggiungere le nuove procedure a quelle già in memoria, puoi usare il comando **unione**. Per esempio:

```
unione "mieproc"
```

Questo funziona come **richiama**, eccetto che le procedure esistenti non sono cancellate. Se una nuova procedura ha lo stesso nome di una esistente, la nuova versione sostituirà quella vecchia.

ESAME DEI RECORD DEL FILE

Un modo per renderti la vita più facile è quello di dare nuovi nomi composti da una lettera sola ai comandi usati più frequentemente. Una alternativa sarebbe costituita dalla scrittura di una procedura più lunga per sostituire parecchi comandi con azionamenti di tasti singoli. Prova ad usare il comando **edita** per definire la seguente procedura. Essa ti permette di aprire ed esaminare qualsiasi file di dati, a condizione, naturalmente, che il file che desideri usare non sia già stato richiamato.

Se hai già definito una procedura, scrivendo:

```
edita
```

non avrai automaticamente l'opzione di creare una nuova procedura. Quando sei in **edita** devi premere **F3** e poi il tasto **N** per iniziare una nuova procedura.

Non preoccuparti se fai degli errori nello scrivere questo esempio – imparerai come correggerli nel prossimo capitolo.

```
proc vedifile
  pulisci
  immetti "quale file? ";file$
  guarda file$
  visualizza
  poni premi$="Z"
  mentre premi$<>"L"
    mostrasch
    poni premi$=maiusc(tasto())
    se premi$="1":primo:finese
    se premi$="U":ultimo:finese
```

```
        se premi$="P":prox:finese
        se premi$="A":precedente:finese
        finementre
    chiudi
    fineproc
```

Ricorda che per uscire da **edita** devi premere due volte **ESC**.

Puoi usare la procedura scrivendo:

```
vedifile
```

Questo, per prima cosa, pulirà la zona di visualizzazione e poi ti inviterà a scrivere un nome di file, per esempio "gazzetta". Comunque, se "gazzetta" è già stato richiamato, riceverai un messaggio di errore. Per ricominciare, scrivi **nuovo** e richiama ed esegui nuovamente la procedura. Quando avrai scritto il nome di uno dei tuoi file di dati, la procedura aprirà quel file nella modalità di sola lettura e visualizzerà il suo primo record. Esso aspetterà che tu prema un tasto e risponderà ai tasti **1**, che sostituisce il comando "primo"; **U**, che sostituisce il comando "ultimo"; **P**, che sostituisce il comando "prox"; **A**, che sostituisce il comando "precedente" (pensa alla parola "antecedente") **L**, che lascia la procedura. Nota che convertendo tutte le lettere a maiuscole, non importa se premi maiuscole o minuscole. Le prime quattro provocheranno le azioni di visualizzazione appropriate (primo, ultimo, prox o precedente) e se premi il tasto L (lascia) chiuderai il file e finirai la procedura.

Poiché questo è il primo programma di una certa lunghezza da noi scritto, possono essere utili alcune osservazioni. Per prima cosa notiamo che l'esempio è rientrato per chiarire la struttura del procedimento. Non c' bisogno che tu lo scriva in questo modo, i rientri vengono aggiunti automaticamente man mano che scrivi, elenchi o stampi la procedura.

La parte principale della procedura (in attesa che venga premuto un tasto e venga eseguita l'azione appropriata), è racchiusa tra i comandi **mentre** e **finementre**. Si uscirà da questa serie di ripetizioni solo quando la condizione che segue **mentre** falsa, in questo caso, quando premi il tasto L.

Anche il comando **se**, usato parecchie volte in questo loop, richiede che ciascun **se** abbia un **finese** corrispondente, per marcare la fine della sequenza di instruzioni da eseguire se la condizione è vera. **Se** e **finese** sono comandi indipendenti e possono essere usati in righe differenti. Per esempio, potremmo aver scritto il primo delle istruzioni **se** in questa procedura come:

```
se premi$="1"
   primo
   finese
```

Puoi comprendere parecchie righe di istruzioni tra **se** e **finese**; esse saranno tutte eseguite, quando la condizione che segue **se** è vera. Nella procedura **vedifile** queste istruzioni sono sufficientemente corte da poter essere su una riga singola, usando il due punti per separare le singole istruzioni.

Come puoi vedere, all'interno del loop principale viene usato un comando **mostrasch**, per accertarti che tutti i nuovi record siano mostrati sullo schermo. Ricorda che, sebbene i comandi di visualizzazione (**primo**, **ultimo** ecc.) portano sempre al record corretto, i dati della zona di visualizzazione non vengono cambiati automaticamente fino alla fine della procedura. Se non avessimo inserito il comando **mostrasch**, nella zona di visualizzazione non sarebbe stata mostrata alcuna informazione fino al momento dell'azionamento del tasto **I** per lasciare la procedura. In quel caso vedresti soltanto il risultato dell'ultima delle sequenze di azionamenti di tasti da te fatte.

CAPITOLO 9
EDIZIONE

Questo capitolo descrive l'editore di programma. Inseriremo alcuni semplici esempi, ma il miglior modo di imparare ad usarlo è utilizzarlo. Inizia scrivendo **nuovo** per pulire la memoria dell'elaboratore.

Dopo aver letto questo capitolo puoi cercare di scrivere alcuni semplici programmi per conto tuo, o puoi cercare di modificare le procedure da te scritte nell'esercitazione del capitolo precedente. Se vuoi usare degli esempi più lunghi, puoi usare l'editore per scrivere tutti o parte dei programmi dei capitoli seguenti.

L'EDITORE DI PROGRAMMA

Per entrare nel *livello principale* dell'editore di programma usa il comando **edita**.

Come esempio possiamo creare una procedura e aggiungerci una coppia di istruzioni. Dal livello principale di **edita**, premi **F3** e **N** per creare una nuova procedura. Quando ti viene chiesto il nome della procedura scrivi **prova**.

Premi due volte **ESC** per uscire dall'editore senza aggiungere alcuna istruzione. Poi usa nuovamente il comando **edita**. Se non hai richiamato nessuna altra procedura, lo schermo mostrerà

```
prova    proc prova
             fineproc
```

Se le procedure da te create nell'ultimo capitolo sono ancora caricate, sulla sinistra sarà evidenziato **prova** come procedura corrente tra queste altre procedure. Premi **F4** per inserire righe di testo. La riga contenente **proc** sarà evidenziata.

Ora scrivi:

```
scrivi "questa una prova" ENTER
scrivi "ci sono due istruzioni" ENTER ENTER
```

Se premi due volte consecutive **ENTER** esci da **inserto**. Alla fine lo schermo avrà questo aspetto:

```
prova    proc prova
         scrivi "questa una prova"
         scrivi "ci sono due istruzioni"
         fineproc
```

La riga che contiene la seconda istruzione di stampa è evidenziata.

Ricorda che fino a quando premi **ENTER**, puoi usare l'editore di riga per correggere il testo che scrivi. Comunque, dopo aver premuto **ENTER** la riga viene inserita nella procedura. Per estrarla nuovamente per **edita**rla devi premere **F5**. Se poi premi **ENTER**, metterai la nuova riga al posto di quella vecchia.

Non puoi editare l'istruzione **fineproc**. Non puoi editare la parola **proc**, ma puoi editare il resto del contenuto di questa riga. Perciò, puoi dare un nuovo nome ad una procedura usando l'editore di riga per cancellare quello vecchio. L'elenco di procedure alla sinistra dello schermo viene riordinato automaticamente per mantenere le procedure in ordine alfabetico.

Comandi di edizione

Nella sezione dei comandi, avrai notato quattro distinti comandi di edizione che vengono usati per creare una nuova procedura. Puoi sceglierne uno premendo **F3** e poi scrivendo la prima lettera del nome.

Nuova procedura (N)

Puoi scrivere il nome della procedura che vuoi creare. Se scrivi il nome di una procedura esistente, verrà creata una nuova procedura che prenderà il posto di quella originale.

Quando premi **ENTER** alla fine del nome, la nuova procedura diventa quella corrente, elencata alla fine dello schermo. Ti viene presentata una procedura vuota – cioè che contiene solo le istruzioni **proc** e **fineproc**.

Distruggi procedura (D)

Questo comando cancella la procedura corrente dal tuo programma. Per prima cosa devi scegliere la procedura desiderata usando i tasti **SHIFT** e **TABULATE**, come descritto precedentemente, per farla diventare la procedura corrente. Poi scegli il comando premendo **F3** e il tasto **D**.

Devi premere **ENTER** per confermare che vuoi veramente cancellare la procedura. Se cambi idea a questo punto, puoi premere qualsiasi altro tasto per ritornare a **edita** senza cancellare la procedura.

Stai attento quando usi questo comando, in quanto è impossibile ripristinare una procedura cancellata, a meno di riscriverla.

Taglia (T)

Questo comando toglie una o più righe di testo dalla procedura corrente. Il testo che viene tolto può essere inserito in un'altra posizione, o persino in un'altra procedura, mediante il comando **incolla**.

Prima di scegliere il comando devi usare i tasti di movimento verticale del cursore per far diventare quella corrente la prima o l'ultima riga della sezione che vuoi togliere. A questo punto puoi scegliere il comando premendo **F3** e poi il tasto **T**.

Se successivamente premi **ENTER**, la riga corrente verrà tolta dalla procedura. Oppure puoi usare il tasto di movimento verso l'alto o il basso del cursore per portarlo all'altra estremità della sezione di testo che vuoi togliere. La regione di testo che verrà tolta viene indicata dall'evidenziamento. Dopo aver marcato il testo che vuoi togliere devi premere **ENTER**. Archive toglierà immediatamente il testo marcato, e lo metterà in una zona riservata della memoria conosciuta col nome di buffer di incolla.

Incolla (I)

Questo comando inserisce il testo tolto l'ultima volta che hai usato il comando **taglia** nella procedura corrente, sotto la riga corrente. Il testo può essere inserito in un'altra posizione, o persino in un'altra procedura.

Prima di scegliere il comando devi, se necessario, usare i tasti **SHIFT** e **TABULATE** per selezionare la procedura in cui vuoi inserire il testo. Devi anche usare i tasti di movimento verticale del cursore per evidenziare la riga immediatamente sopra la posizione dove vuoi inserire il testo.

Archive inserisce il testo immediatamente al di sotto della riga corrente. Dopo aver usato **incolla** per inserire il testo, il buffer di incolla è vuoto. Perciò, puoi inserire lo stesso testo in più di una posizione.

CAPITOLO 10
PROGRAM-MAZIONE IN ARCHIVE

Questo capitolo descriverà lo sviluppo di un esempio di lavoro e ciascuna nuova tecnica verrà descritta al momento in necessaria.

Immaginiamo che tu sia interessato alla gestione di un club o società che richieda una inscrizione e produca un bollettino. Avrai bisogno di inviare una copia di ogni numero a ogni iscritto. Dovrai inoltre inviare un sollecito a ciascun socio al momento della scadenza del rinnovo dell'iscrizione.

Questo esempio ti permette di costruire un elenco postale e poi stampare una serie di etichette d'indirizzo su richiesta. L'etichetta d'indirizzo comprende un sollecito quando è dovuta l'iscrizione. L'esempio presume che tu invii sei numeri del bollettino all'anno e che l'iscrizione scada quando il socio ha ricevuto sei numeri. Potrebbe essere facilmente adattato a qualsiasi situazione in cui regolarmente invii qualche tipo di circolare ad un certo numero di persone su un elenco postale.

UN ELENCO POSTALE

In questo esempio useremo il più possibile i comandi già noti e ne introdurremo alcuni nuovi. Se hai bisogno di aiuto riguardo ad una caratteristica o comando che non hai ancora incontrato, cerca aiuto nella sezione di consultazione o usa la funzione di **aiuto** premendo **F1**. Useremo i comandi **inserimento** e **modifica** per le aggiunte e modifiche ai record dei file. Comunque, avremo bisogno di scrivere procedure speciali per stampare le etichette degli indirizzi.

Dovremo soddisfare la seguente serie di requisiti:

Aggiungere un nuovo record al file.
Cancellare un record.
Modificare un record.
Registrare il pagamento delle iscrizioni.
Produrre le etichette degli indirizzi.
Lasciare il programma.

Scriveremo una procedura per trattare ciascuno di questi compiti e li collegheremo assieme con un'altra procedura che ti permetterà di scegliere una qualsiasi di queste opzioni.

In questo è impiego molto chiaro quello che i campi di ciascun record devono contenere. Il nome e l'indirizzo sono essenziali, ed inoltre un campo per registrare il numero di bollettini che la persona ha ricevuto. Possiamo creare immediatamente il file necessario, che avrà l'aspetto dell'esempio sottostante:

```
crea "posta"
   titolo$
   cognome$
   nome$
   via$
   cap$
   città$
   numeri
   finecrea
```

Abbiamo usato tre campi alfabetici per il nome della persona; per contenere rispettivamente il titolo (Dott., Sig., Sig.ra ecc.), il cognome ed il nome. Probabilmente avremmo potuto cavarcela anche con un solo campo.

Per l'indirizzo abbiamo creato tre campi, riservati nominalmente alla via o piazza, al codice postale e alla città. Non c' sempre bisogno di usarli in questo modo, ma puoi trattarli come tre campi generali dove mettere l'indirizzo. Di solito tre campi dovrebbero essere sufficienti.

C'è un solo campo numerico, per inserirvi le informazioni riguardanti quanti numeri devono essere ancora inviati.

Ora che abbiamo il file, possiamo usarlo per mettere alla prova le varie procedure man mano che le scriviamo. E' una buona idea provare le procedure man mano che procedi. Così potrai individuare gli errori e correggerli immediatamente. Se lasci tutte le prove alla fine, sarà molto più complicato, in quanto possibile che parecchie cose

non funzionino allo stesso tempo. Mentre stai provando le procedure fai le cose nel modo più semplice possibile. Cerca di accertarti che ogni procedura funzioni correttamente prima di passare alla prossima. In questo modo troverai che il tuo programma finale funzionerà appena avrai scritto l'ultima procedura.

Inserimento

Non c' bisogno di scrivere una procedura per aggiungere un record. Possiamo usare **inserimento**. Ricorda che devi usare **mostrasch** per forzare la visualizzazione del contenuto del record mentre sei in una procedura. Puoi aggiungere immediatamente **inserimento** per aggiungere alcuni record al file in modo che tu possa provare le altre procedure in un file vero.

Cancellazioni

Ci sarà un momento in cui vorrai togliere i record delle persone che non hanno rinnovato l'iscrizione. Scriveremo una procedura, che chiameremo **elimina**, che ti permette di scorrere nel file, esaminando i record di tutte le persone che non hanno rinnovato l'iscrizione, e di decidere quali devono essere cancellati.

Useremo la variabile di campo **bollettini** per inserirvi il numero di numeri che una persona ha il diritto di ricevere. Tutti i record per cui il valore di **numeri** è zero, sono perciò candidati per la cancellazione.

```
proc elimina
   nota ***** cancella iscritti che non pagano *****
   pulisci
   visualizza
   seleziona numeri=0
   tutto
      mostrasch
      scrivi da 10,0; "CANCELLA (s/n)? ";
      poni vabene$ =maiusc(tasto())
      scrivi vabene$
      scrivi da 10,0;
      se vabene$ ="S"
         cancella
         scrivi "CANCELLATO"; tab 20
         finese
      scrivi tab 20
      finetutto
   ripris
fineproc
```

Poiché un record cancellato non può essere recuperato, viene visualizzato il contenuto completo del record e ti viene chiesto di confermare se vuoi veramente cancellarlo. Usiamo la funzione **tasto()** che attende l'azionamento di un tasto e poi presenta il codice ASCII di quel tasto. Nota che **minuscolo()** converte il codice in carattere minuscolo, per cui puoi scrivere le lettere sia maiuscole che minuscole.

Una volta che sei sicuro di aver scritto correttamente questa procedura, puoi provarla sui tuoi file, (naturalmente, a condizione di aver scritto dei record di prova). Per prima cosa esci da **edita**, premendo **ESC** (due volte se necessario) e memorizza la tua procedura in un file chiamato "Eleposta".

Scrivi:

```
memorizza "Eleposta"
```

La procedura chiamata **elimina** è ora immagazzinata e può essere utilizzata tutte le volte che viene richiamato "Eleposta".

Dopo aver immesso ciascuna delle seguenti procedure, ripeti questi passi, registrando ogni volta le nuove procedure in "Eleposta".

Pagamenti

Normalmente avrai bisogno di registrare un gruppo di pagamenti di iscrizioni da un elenco di nomi ed indirizzi. Perciò avrai bisogno di prendere il record di una persona specifica. Il modo più semplice quello di scrivere una procedura distinta, **prendrec**, per localizzare un particolare record e poi incorporarlo in una procedura **rinnovo**.

La procedura **prendrec** ti chiede una stringa di testo (n$) e poi indivudua il primo record del file che contiene quel testo. Se rispondi premendo semplicemente **ENTER**, n$ viene impostato sulla stringa vuota e non viene fatta alcuna ricerca. Comunque, questo indicherà che hai finito di registrare i pagamenti.

Dal livello **edita**, premi **F3** e **N** per iniziare ad immettere **prendrec**

```
proc prendrec
  nota ***** localizzare un record specifico *****
  pulisci
  poni vabene$ ="N"
  immetti "Nome dell'iscritto? "; n$
  se n$ <>""
    cerca n$
    mentre vabene$ <>"S" ed trovato()
      scrivi titolo$ ; " "; nome$(1); ". "; cognome$
      scrivi via$
      scrivi "VABENE (s/n)? ";
      poni vabene$ =maiusc(tasto())
      pulisci
      se vabene$ <>"S"
        continua
        finese
      finementre
    se non trovato()
      scrivi n$ ; " non trovato."
      finese
    finese
fineproc
```

La ricerca usa il comando **cerca**, per cui il testo viene cercato in tutti i campi di testo. Perciò puoi identificare un record per nome o per indirizzo. Naturalmente, il primo record che corrisponde può non essere quello che vuoi, per cui dobbiamo essere in grado di continuare la ricerca. La procedura ripetitiva **mentre finementre** ha questo scopo specifico. Essa scrive il nome e la prima riga dell'indirizzo, per identificare il record, e ti chiede se questo è il record giusto. Se non rispondi premendo il tasto S, esso continua la ricerca. La procedura ripetitiva finisce quando rispondi premendo il tasto S o quando il testo non viene trovato in nessuno dei record rimanenti. Nota che la funzione **trovato()** dà un valore vero (non zero) se la ricerca ha successo.

Poiché vabene$ potrebbe essere un "S" iniziale (da una ricerca precedente che ha avuto successo) devi darle qualche altro valore all'inizio della procedura, prima di immettere la procedura ripetitiva. Questo garantisce che la procedura verrà utilizzata almeno una volta.

Ora puoi scrivere la procedura **rinnovo**:

```
proc rinnovo
  nota ***** registra il pagamento dell'iscrizione *****
  pulisci
  poni n$ ="X"
  mentre n$ <>""
    prendrec
    se vabene$ ="S"
      poni numeri =numeri +6
      aggiorna
      finese
    finementre
fineproc
```

La ripetizione di questa procedura continua fino a che n$ sia una stringa vuota. Questo ti permette di registrare parecchi pagamenti senza dover selezionare l'opzione **rinnovo** per ciascuno di essi. Quando hai finito, premi **ENTER** in risposta al messaggio "chi?". Se il valore di vabene$ è "S" dopo la chiamata di **prendrec**, il pagamento viene registrato e marcato come valido per altri sei numeri.

Anche in questo caso dobbiamo impostare il valore iniziale di n$ a qualche valore appropriato (qualunque, purché non sia una stringa vuota) per garantire che la procedura non sia influenzata da una operazione precedente.

Cambiamenti

La procedura che ti permette di cambiare il contenuto di un record è ora molto facile. Anche in questo caso devi essere in grado di selezionare un record particolare da cambiare, per cui la struttura generale può essere identica a quella di **rinnovo**.

```
proc cambia
   nota ***** modifica il record *****
   pulisci
   poni n$ ="X"
   mentre n$ <>""
      prendrec
      se vabene$ ="S"
         modifica
         pulisci
         finese
      finementre
   fineproc
```

PARAMETRI

Ora faremo un piccolo intervallo dallo sviluppo del programma per descrivere l'uso dei *parametri* con le procedure. Puoi usare un *parametro* per inviare un valore ad una procedura, invece di usare il valore di una variabile. Ti mostreremo alcuni esempi di come possono essere usati. Non c'è bisogno di memorizzare queste procedure in "eleposta" e puoi cancellarle prima di passare alla sezione del programma che tratta le etichette.

Prova il seguente sempice esempio. Usando l'editore di riga, aggiungi il parametro alla riga che contiene il nome della procedura.

```
proc prova; a
   scrivi 5*a
fineproc
```

Questo definisce una procedura chiamata **prova**, che richiede un parametro, "a". Nota che il parametro è separato dal nome della procedura da un punto e virgola. Quando usi la procedura devi sempre fornire il valore del parametro. Per esempio, puoi scrivere:

```
prova; 3
```

che stamperà il valore 15 – il numero (3) stato inviato alla procedura come il valore della variabile a.

In una procedura puoi specificare qualsiasi numero di parametri, a condizione che li separi da virgole. Per esempio:

```
Xchange
proc tenta; a,b,c
   scrivi a * b * c
fineproc
```

che puoi chiamare con:

```
tenta; 3,4,5
```

I valori che fornisci non devono essere necessariamente alfabetici, ma potrebbero essere variabili, come illustrato più sotto:

```
poni x = 2
poni y = 5
poni z = 7
tenta; x,y,z
```

Nota che i nomi delle variabili non devono essere gli stessi dei nomi usati all'interno della procedura. Nella definizione della procedura possiamo distinguere tra i *parametri formali* (per es. a,b,c), ed i *parametri effettivi*, cio i valori effettivi che vengono inviati alla procedura.

Puoi anche inviare il risultato delle espressioni:

```
tenta; x*2,z/y,(z-y)*x
```

Non sei limitato all'impiego di variabili numeriche, ma puoi inviare come parametri anche stringhe (o espressioni stringa) a condizione che specifichi le variabili stringa nella definizione della procedura. Per esempio:

```
proc dai; a$
   scrivi a$
fineproc
```

```
poni t$ = "messaggio"
dai; t$
```

Il solo requisito è che il numero ed il tipo di parametri forniti deve corrispondere alla lista di parametri formali della definizione della procedura.

Etichette d'indirizzo

La ragione di questa breve introduzione dei parametri, è che essi forniscono un modo chiaro di scrivere la procedura per stampare una etichetta d'indirizzo. Per questa prova scriveremo per prima cosa la procedura per mostrare gli indirizzi sullo schermo, e poi la convertiremo per inviare l'uscita alla stampante. Presumeremo che le etichette siano lunghe otto righe di scrittura. Se questo non va bene per la tua combinazione di stampante ed etichetta, dovrai cambiare il numero di righe di spazio della procedura, in modo che corrisponda ai tuo requisiti. Ricorda di iniziare memorizzando nuovamente le tue procedure in "Eleposta".

Per prima cosa scriveremo una procedura che visualizzi una sola riga, il cui contenuto viene inviato attraverso un parametro.

```
proc fariga; x$
  scrivi x$
  fineproc
```

Ora potremo usare questa procedura per visualizzare otto righe di testo per l'etichetta d'indirizzo.

```
proc faetichetta
  nota ***** scrivi etichette *****
  se numeri
    se numeri =1
      fariga; "SOLLECITO - Iscrizione in scadenza"
      altrimenti
      fariga; ""
      finese
      fariga; ""
      fariga; titolo$ +" "+nome$ (1)+". "+cognome$
      fariga; via$
      fariga; cap$+" "+città$
      fariga; ""
      poni numeri =numeri - 1
      aggiorna
      finese
  fineproc
```

La procedura comprende un commento nell'etichetta d'indirizzo se la persona sta per ricevere l'ultimo numero. Ogni volta che viene stampata un'etichetta, il conteggio dei numeri della persona viene ridotto di uno. Se questo numero diventato zero l'etichetta non viene stampata.

Puoi cominciare a vedere che i parametri possono essere veramente utili – senza di essi questa procedura sarebbe molto più lunga. Vedi come facile combinare il titolo, l'iniziale e il cognome per la prima riga dell'indirizzo.

Forse ti domandi perché ci siamo presi il disturbo di definire **fariga**, quando avremmo potuto usare semplicemente istruzioni **scrivi** per tutto **faetichetta**. La ragione è che la routine nella sua forma presente mostra gli indirizzi sullo schermo. Possiamo convertirla per inviare la sua uscita alla stampante semplicemente cambiando una riga di **fariga**, invece di dover cambiare tutte le istruzioni di stampa di **faetichetta**. Dobbiamo cambiare **fariga** e farla diventare:

```
proc fariga; x$
  stampa x$
  fineproc
```

Finalmente possiamo scrivere la procedura per stampare tutte le etichette d'indirizzo:

```
proc spedisci
  pulisci
  tutto
    faetichetta
    finetutto
  fineproc
```

Uscita dal programma

L'opzione finale è quella di uscire dal programma quando hai finito. Questa procedura può essere molto semplice – devi soltanto accertarti che il file sia chiuso correttamente prima di restituire il comando alla tastiera. Abbiamo anche aggiunto un breve messaggio di commiato per rendere chiaro che il programma è finito.

```
proc ciao
   pulisci
   scrivi da 10;10 "CIAO "
   chiudi
   ferma
   fineproc
```

ERRORI

E' molto probabile che prima o poi nell'impiegare questo programma farai un errore. Per esempio, puoi premere per sbaglio il tasto **ESC**, oppure puoi inserire nel testo quando ci si aspetta un numero. Questo tipo di errore viene individuato da Archive e normalmente porta alla visualizzazione di un messaggio d'errore ed al ritorno alla tastiera dal programma.

Puoi usare il comando **errore** per controllare, in una procedura, l' esecuzione degli errori. Se nella procedura marcata, o in una procedura da essa chiamata, si verifica un errore, questo porterà ad un ritorno immediato.

Il sistema normale di trattare gli errori non è *attivo* per la procedura marcata e sarà compito tuo decidere come trattarlo. Puoi trovare il numero dell'ultimo errore verificatosi usando la funzione **numerr()**. Puoi usarla per leggere più volte il numero di errore, in quanto il valore viene portato a zero dall'uso successivo del comando **errore**. Se dall'inizio del programma, o dall'ultima volta che stato eseguito **errore** non si è verificato nessun errore, **numerr( )** darà il valore di zero.

Questo metodo, sebbene difficile da capire all'inizio, ti dà un modo molto potente e flessibile di trattare gli errori. Il seguente esempio mostra un modo tipico di usare **errore**. Ti dà un sistema a prova di errore per immettere un numero.

```
proc faprova
   immetti "Immetti numero: ";x
   fineproc

proc prova
   poni n =1
   mentre n
      errore faprova
      poni n =numerr()
      se n
         scrivi " Hai fatto l'errore num. " ;n ;", riprova"
         finese
      finementre
   fineproc
```

La prima procedura attende semplicemente che tu immetta la variabile x. La seconda procedura tratta gli errori durante l'esecuzione della procedura d'immissione. Se durante **faprova** si verificano degli errori, essa verrà interrotta prematuramente e verrà riportato il numero di errore. Questo numero viene poi letto da **numerr( )** e, se è differente da zero, viene scritto il messaggio di errore (naturalmente, questo messaggio di errore può essere qualsiasi cosa a tuo piacere). Poiché queste istruzioni sono racchiuse in un loop **mentre finementre**, qualsiasi errore provocherà la ripetizione della loro esecuzione. Il numero di errore viene tolto da **errore**, per cui sarà pronto per il prossimo tentativo. Non puoi uscire da **prova** fino a che non avrai scritto un numero corretto.

Questo esempio riporta il numero di errori individuati. Nella maggior parte delle occasioni non ti interesserà quale errore si è verificato. L'impiego principale di **numerrn( )** è di differenziare tra l'assenza di errori e la presenza di un errore di qualsiasi tipo. Nel capitolo di consultazione, si troverà un elenco di numeri di errore e possibili spiegazioni

Ora possiamo scrivere una procedura che ti permette di scegliere una qualsiasi tra le sei opzioni mediante l'azionamento di un solo tasto. E' semplice quanto basta per non aver bisogno di alcuna spiegazione.

```
proc scegli
   nota ***** scegli una opzione *****
   pulisci
   scrivi
   scrivi " Aggiungi Spedisci Rinnovo Cambia Elimina Lascia";
   scrivi " Quale opzione? ";
   poni scelta$ =maiusc(tasto())
   scrivi scelta$
   se scelta$ ="A": inserimento : finese
   se scelta$ ="S": spedisci : finese
   se scelta$ ="R": rinnovo : finese
   se scelta$ ="C": cambia : finese
   se scelta$ ="E": elimina : finese
   se scelta$ ="L": ciao : finese
   fineproc
```

Per completare il nostro programma rimane soltanto da scrivere una precedura di partenza che apra il file e chiami **scegli**. Dobbiamo includere **scegli** nella procedura ripetitiva, in modo che ti venga offerta nuovamente l'opzione, ogni volta che completi la selezione precedente.

Vedrai che il loop **mentre finementre** della procedura seguente non finirà mai. Questo loop finirà soltanto quando l'espressione che segue **mentre** ha il valore zero. Nella procedura di cui sopra, l'espressione ha sempre il valore 1, per cui il loop continuerà indefinitamente. Il solo modo di uscire da questo loop è di scegliere l'opzione Lascia. Il comando **ferma** di **ciao** restituisce immediatamente il comando alla tastiera.

```
proc start
   nota ***** procedura di avviamento *****
   pulisci
   apri "posta dbf"
   mentre 1
      errore scegli
      poni n =numerr()
      se n
         scrivi da 2,1; "Errore - Premi un tasto per proseguire"
         poni m$ =tasto()
      finese
   finementre
fineproc
```

All'interno di questo loop si trova una sequenza di istruzioni che tratta tutti gli errori, usando un sistema simile a quello descritto nella sezione precedente. Se fai un errore, il programma non continuerà fino a che premi un tasto. Questo ti permette di guardare quello che hai appena fatto, in modo da poter trovare la causa dell'errore.

IL COMANDO ESEGUI

La procedura principale del programma di elenco postale è chiamata "start" in modo che tu possa usare il comando **esegui** nell'usare il programma.

Memorizza questa procedura finale in "eleposta". Quando vuoi eseguire il programma avrai bisogno di richiamare le procedure nella memoria dell'elaboratore e poi eseguire la procedura principale, che chiamerà tutte le altre. Un modo è quello di usare il comando **richiama** e poi scrivere il nome della procedura principale, per esempio:

```
richiama "eleposta"
start
```

Il comando **esegui** richiamerà il programma di cui si dà il nome e poi eseguirà la procedura chiamata "start" (se esiste). Puoi eseguire il programma esattamente come nell'esempio precedente scrivendo semplicemente:

```
esegui "eleposta"
```

Le due sezioni rimanenti di questo capitolo comprendono delle procedure a scopo generale che puoi trovare utili.

VARIABILI LOCALI

La maggior parte delle variabili che compaiono nelle procedure sono *globali*. Questo significa che esse sono riconosciute in tutto il programma. Esse possono essere usate o cambiate in qualsiasi procedura, e non soltanto nella procedura in cui sia stata assegnato per la prima volta un valore.

Le variabili usate come parametri formali in una procedura sono *variabili locali* ed esse non vengono riconosciute al di fuori della procedura in cui compaiono.

L'esempio seguente può aiutare a chiarire la differenza. Prima di cominciare, scrivi **nuovo** per pulire la memoria dell'elaboratore. Per prima cosa creiamo una procedura che usa due variabili locali, a e b$, ed inoltre assegnamo i valori a due variabili normali (globali) u e v$.

```
proc dimostra; a,b$
scrivi a;b$
poni u=3
poni v$="testo"
scrivi u;v$
fineproc
```

Poi usiamo **dimostra**:

```
dimostra;5,"parole"
```

Tutti e quattro i valori vengono stampanti, a dimostrazione che all'interno di **dimostra** vengono riconosciute tutte e quattro le variabili. Scrivendo

```
scrivi u;v$
```

dimostri che queste due variabili sono riconosciute anche fuori della procedura. Comunque, se scrivi

```
scrivi a;b$
```

Hai come risultato un errore perché a e b$ non vengono riconosciute al di fuori di **dimostra**. Tutti i parametri formali sono variabili locali, ma puoi anche dichiarare altre variabili locali come nel seguente esempio:

```
proc sapoco
   scrivi "dentro sapoco"
   scrivi p; q; r
   fineproc
proc satutto
   locale q,r
   poni p = 2
   poni q = 3
   poni r = 4
   scrivi "dentro satutto"
   scrivi p; q; r
   sapoco
   fineproc
```

Se cerchi di usare **satutto** scrivendo:

```
satutto
```

troverai che i valori di p, q e r sono tutti riconosciuti (e perciò scritti) in **satutto**, ma **sapoco** non conosce i valori di q ed e, che sono locali di **satutto**.

I valori delle variabili locali non sono definiti in nessun altro posto, eccetto che nella procedura in cui sono dichiarati – nemmeno nelle procedure chiamate dalla procedura di dichiarazione. La variabile p è globale ed è riconosciuta ovunque.

Ti puoi domandare perché siano necessarie le variabili locali. Per illustrare la loro utilità, immagina di scrivere un programma contenente parecchie procedure che tu, o qualcun altro, avevi scritto originariamente per usarle con altri programmi. E' del tutto possibile che due o più di queste procedure possano usare variabili con lo stesso nome per scopi completamente differenti. Se queste variabili fossero globali, una procedura potrebbe variare un valore, che potrebbe risultare sbagliato per un'altra. In una tale situazione dovresti controllare tutte le procedure che usi e, se necessario, cambiare i nomi delle variabili. Comunque, se le variabili fossero locali, non avrebbe alcuna importanza il nome. A condizione che siano in procedure differenti, se ne cambi una, non influirai sull'altra.

Inoltre, non importa se una procedura chiama un'altra che usa lo stesso nome per una variabile – a condizione che almeno una di esse sia locale. Per esempio, la procedura **scegli**, nella sezione sugli errori, più sopra in questo capitolo, dichiarava che la variabile scelta$ era locale. Questo significa che non c'è bisogno di controllare se alcune delle procedure chiamate da **scegli** usano scelta$ – le procedure chiamate non possono cambiare il valore di scelta$ in **scegli**.

MESSAGGI

La visualizzazione di un messaggio e l'attesa che si prema un tasto è una delle azioni richieste più frequentemente, per cui vale la pena di scrivere una procedura di scopo generale. Questa procedura deve essere in grado di visualizzare un'ampia serie di messaggi. Un modo semplice, per permettere alla procedura di visualizzare qualsiasi messaggio, quello di passare il messaggio alla procedura sotto forma di un parametro.

```
proc messaggio; m$
   scrivi m$ + ": ";
   poni x$ =maiusc(tasto())
   scrivi x$
   fineproc
```

Il messaggio da visualizzare viene passato alla procedura sotto forma di un parametro della variabile locale m$. La funzione **tasto( )** attende che venga premuto un tasto e dà il valore ASCII del tasto. In questa procedura il codice ASCII viene convertito in maiuscolo dalla funzione **maiusc( )**, per cui il risultato è indipendente dal carattere maiuscolo o minuscolo. Per finire, il valore risultante viene assegnato alla variabile x$. Essa è una variabile globale, per cui il tasto che è stato effettivamente premuto è disponibile per qualsiasi altra procedura del programma.

PAUSA

Pausa costituisce una procedura utile. Essa usa **messaggio** per scrivere un messaggio e poi attende semplicemente finché si preme un tasto. Poiché, di solito, non ti interessa sapere quale tasto sia stato effettivamente premuto, essa usa una variabile locale, y$, per conservare il contenuto originario di x$. Nota che bisogna aver assegnato un valore a x$ prima di usare Pausa.

```
proc pausa
   nota ***** attendi un tasto *****
   locale y$
   poni y$ =x$
   scrivi messaggio; "Premi un tasto per proseguire"
   poni x$ =y$
   fineproc
```

IMMISSIONE DI DATI

Testo

L'accettazione di testo come immissione digitata è molto semplice. Ogni insieme di caratteri costituisce una stringa di testo valida (anche se non ha alcun significato) e non provocherà un errore del sistema. Normalmente non ci sarà bisogno che tu prenda delle precauzioni speciali nell'accettare l'immissione di testo. Di solito sarà sufficiente usare una riga simile alla seguente, che ti chiede di scrivere il tuo nome:

```
immetti "Scrivi il tuo nome: ";nome$
```

Nota che l'ultimo carattere del testo del messaggio è costituito da uno spazio; questo porterà ad una grande differenza nell'aspetto del tuo programma quando lo usi.

Puoi immettere parecchie voci con una istruzione d'immissione. Devi soltanto inserire tutti i messaggi e i nomi variabili, separati da punti e virgole.

```
immetti "Il tuo nome? ";nome$;" Il tuo cognome? ";cognome$;
```

Anche l'ultima istruzione d'immissione finisce con un punto e virgola – questo impedisce che il cursore vada alla linea seguente dopo che hai scritto l'immissione.

Numeri

Quando usi il comando **immetti** per scrivere del testo in una variabile stringa, l'elaboratore accetterà tutto quello che scrivi, senza protestare. Però, nel caso di una variabile numerica, se scrivi qualsiasi cosa che non sia un numero o una espressione numerica valida, questo non verrà accettato e riceverai un messaggio d'errore. Presumendo che non voglia uscire dal programma ogni volta che ti scivolano le dita mentre stai scrivendo un numero, devi fare in modo che il programma sia in grado di far fronte a tali errori.

Il modo più utile è usare il comando **errore**, che stato descritto precedentemente. La procedura seguente, per esempio, accetterà qualsiasi numero valido entro una serie specificata. Essa dà persino la visualizzazione di qualsiasi messaggio che vuoi far comparire.

```
proc prendnum; m$,min,max
   nota ***** prendi numero nella gamma specificata *****
   locale sbagliato
   poni sbagliato=1
   mentre sbagliato
      scrivi m$; "? ";
      errore legnum
      poni sbagliato=numerr()
      se non sbagliato
         se num<min od num>max
         poni sbagliato=1
         scrivi "La gamma permessa ";min;" ad ";max
         finese
      finese
      se sbagliato
         scrivi " Riprova"
         finese
      finementre
   fineproc
```

Poiché **errore** dev'essere seguito dal nome di una procedura, definiamo **legnum**, in modo da immettere un valore per la variabile **num**.

```
proc legnum
   immetti num
   fineproc
```

Immaginiamo che tu voglia una procedura che controlla che un numero sia entro i limiti da 1 a 10. Puoi farlo usando **prendnum** nel modo seguente:

```
proc verifica
   prendnum; "Valore numerico",1,10
fineproc
```

CAPITOLO 11
IMPIEGO DI FILE MULTIPLI

NOMI DI FILE LOGICI

Questo capitolo amplia la spiegazione di come usare il linguaggio di programmazione di Archive, descrivendo come lavorare con due o più file aperti. Quando hai più di un file aperto contemporaneamente, devi essere in grado di identificare quale vuoi usare per una specifica operazione. Devi dare a ciascun file un *nome di file logico* quando lo apri o crei e poi indicarlo con quel nome in tutti i comandi che si riferiscono al file.

Archive fornisce automaticamente il *nome di file logico*, "principale", quando apri un file singolo. Esso viene chiamato nome di file logico per distinguerlo dal *nome di file fisico* – il nome che dai al file quando lo memorizzi.

Poiché un programma si riferisce ad un file con il suo nome di file logico, puoi scrivere un programma che funzionerà con parecchi file differenti. I nomi di file logici sono essenziali per le operazioni con file multipli, in quanto l'unico modo di aprire un secondo file è di usare sia il nome fisico che logico di file. Nota che il nome di file logico non viene memorizzato con il file quando esso viene chiuso e deve essere specificato ogni volta che il file è aperto.

Due o più file di dati potrebbero contenere campi con lo stesso nome. Quando questo succede puoi identificare il file a cui appartiene il campo aggiungendo il nome del file logico al nome del campo. Per esempio, se il campo stato$ appare in due file i cui nomi di file logici sono "principale" e "b", puoi riferirti a ciascuno di essi rispettivamente come "principale.stato$" e "b.stato$".

VARIAZIONE DEI RECORD DI UN FILE

Il primo esempio dimostra come aggiungere, cancellare o rinominare i campi all'interno di un file esistente.

Immaginiamo che tu voglia fare dei cambiamenti al file "gazzetta", per creare un nuovo file contenente solo paesi europei. Il campo "continenti$" diventa inutile e non c'è bisogno di includerlo. Inoltre chiameremo il campo "pop" con il nuovo nome "popolazione".

Il modo più conveniente di variare il file è quello di creare un secondo file contenente i campi che vuoi e poi di copiare i record richiesti dal vecchio file a quello nuovo. Chiamiamo il nuovo file "europa". La seguente procedura farà il resto dell'operazione.

```
proc start
   nota ***** crea il file europa *****
   crea "europa" logico "e"
      stato$
      capitale$
      lingue$
      moneta$
      popolazione
      pnl
      superfice
      finecrea
   guarda "gazzetta" logico "g"
   seleziona continente$="EUROPA"
   tutto "g"
      scrivi at 0,0;g.stato$;tab 30
      poni e.stato$=g.stato$
      poni e.capitale$=g.capitale$
      poni e.lingue$=g.lingue$
      poni e.moneta$=g.moneta$
      poni e.popolazione=g.pop
      poni e.pnl=g.pnl
      poni e.superfice=g.superfice
      aggiungi "e"
      finetutto
   chiudi "e"
   chiudi "g"
   scrivi
   scrivi "FATTO"
   fineproc
```

IL FILE CORRENTE

Da questo esempio puoi vedere che puoi usare lo stesso nome per un campo in entrambi i file – essi possono essere distinti includendo il nome di file logico. Se non includi il nome di file logico, si presumerà che debba essere usato il *file corrente*. L'ultimo file da aprire diventa automaticamente il file corrente. In questo esempio il file corrente sarà "gazzetta" (con nome di file logico "g"), per cui possiamo utilizzarlo semplicemente evitando di scrivere la g prima del nome del campo nel programma precedente.

Se non inserisci il nome di nome di file logico in nessun caso in cui è opzionale, Archive presumerà che il comando si riferisca al file corrente. Di solito, per evitare qualsiasi possibilità di confusione, è più sicuro inserire esplicitamente il nome di file logico.

In qualsiasi momento puoi specificare il file corrente mediante il comando **usa**. Se nell'esempio suddetto tu inserissi il comando:

```
usa "e"
```

"europa" sarebbe il file corrente fino a quando tu lo cambiassi nuovamente, aprendo un altro file o mediante il comando **usa**.

GESTIONE DEL MAGAZZINO

Facciamo ora un esempio più complesso. In un sistema di gestione del magazzino hai bisogno di:

Cercare le informazioni su un particolare articolo di magazzino
Farsi dare un rapporto sui livelli attuali di magazzino di tutti gli articoli
Registrare le vendite e modificare in conformità le registrazioni di magazzino
Ordinare nuove forniture, per mantenere livelli adeguati di giacenze.
Registra le consegne di giacenze.

Avrai ovviamente bisogno di un file per conservare i particolari di tutti gli articoli mantenuti in magazzino ed è conveniente avere un secondo file per conservare i particolari di tutti i tuoi fornitori. Avrai bisogno di poter avere accesso a un file dall'altro – per esempio è possibile che tu voglia conoscere tutti i possibili fornitori di un particolare articolo, o trovare quali articoli sono forniti da una particolare società.

Per mantenere l'impiego il più semplice possibile, non useremo l'approccio guidato da menù degli esempi dei due capitoli precedenti. Lo scriveremo come una serie di comandi separati che possono essere usati – come i comandi normali – scrivendo i loro nomi.

Poiché le procedure saranno fortemente dipendenti dalla struttura dei file che usiamo, dobbiamo per prima cosa pensare al loro aspetto.

Il file di magazzino

Il file di magazzino deve contenere particolari completi sulla situazione di magazzino per ciascun articolo. Il seguente elenco spiega tutti i campi che useremo.

Nome campo	Uso	Esempio
`Nummaga$`	Il codice interno di mag.	`A101`
`descrizione$`	Descrizione della voce	`Dispositivo, grande`
`quantità`	Quantità in magazzino	`500`
`prezvend`	Prezzo di vendita	`1.25`
`livriordino`	Riordinare quando il livello in magazzino scende sotto questo valore	`200`
`quantacq`	Quantità da ordinare	`400`

Possiamo creare il file con:

```
crea "giacenza" logico "mag"
   nummaga$
   descrizione$
   quantità
   prezvend
   livriordino
   quantacq
   finecrea
```

Il file fornitori

Questo file contiene i nomi, indirizzi e numeri di telefono delle ditte che forniscono le merci da te vendute. Sarà utile inserire anche il nome della persona che contatti nella ditta. Allo scopo di poter aver accesso in modo efficiente a queste informazioni, dovremo includere un codice per ciascuna ditta. Useremo i campi seguenti:

Nome campo	Uso	Esempio
`nomeditta$`	Il nome della ditta	`Super Carri SpA`
`via$`	Prima riga d'indirizzo	`Via Belmonte 37`
`cap$`	Seconda riga d'indirizzo	`12345`
`città$`	Terza riga d'indirizzo	`Modena`
`stato$`	Ultima riga d'indirizzo	`Italia`
`contatto$`	Nome del contatto	`Andrea Comino`
`tel$`	Numero di telefono	`051-532 7133`
`codice$`	Il codice della ditta	`a`

Possiamo creare il file con:

```
crea "fornitor" logico "forni"
   nomedit$
   via$
   cap$
   città$
   stato$
   contatto$
   tel$
   codice$
   finecrea
```

Il file ordini

Questo file forma il collegamento tra i due file precedenti. Esso usa i seguenti campi:

Nome campo	Uso	Esempio
`nummaga$`	Il tuo codice di magazzino	`A101`
`codice$`	Il tuo codice del fornitore	`a`
`codiforn$`	Il codice del fornitore dell'articolo	`123-456`
`prezzo`	Il prezzo di vendita del fornitore	`0.87`
`consegna`	Il tempo di consegna del fornitore, in giorni	`28`

Ciascun record di questo file collega un record del file di magazzino con un record nel file del fornitore. L'esempio di cui sopra mostra che Super Carri (codice di fornitore "a") può fornirti dispositivi grandi (codice di magazzino "A101"). In aggiunta, inseriamo dei particolari del prezzo, tempo di consegna e codice di magazzino del fornitore. Queste voci sono utili quando ordini altro materiale.

L'impiego di questo file ti permette di tener conto dei casi in cui un fornitore fornisce più di un articolo di magazzino (valori uguali per codice$, ma valori differenti per nummaga$) e in cui un articolo di magazzino è ottenibile da parecchi fornitori (nummaga$ uguale, ma codice$ differente).

Crea il file con:

```
crea "ordini" logico "ord"
   nummaga$
   codice$
   codiforn$
   prezzo
   consegna
   finecrea
```

Richieste

Avendo creato questi file, ora abbiamo bisogno di alcune procedure per trattare le informazioni che conterranno. Troverai che il servizio di cui c'è bisogno più frequentemente è quello di trovare le informazioni su un particolare articolo di magazzino in risposta alle richieste del cliente. Avrai bisogno di trovare l'informazione il più rapidamente possibile, ma probabilmente avrai bisogno di trovare un record particolare dal numero di particolare o dalla descrizione. Perciò useremo il comando **cerca** in modo che tu possa dare un qualsiasi testo valido per iniziare la ricerca.

La procedura deve essere in grado di chiederti di confermare che il record è quello di cui hai bisogno. Delegheremo questo compito a una procedura distinta, in modo che, se necessario, possiamo usarla in situazioni differenti.

```
proc conferma
  scrivi : scrivi "Conferma (s/n) ";
  poni si=maiusc(tasto())="S"
  pulisci
  fineproc
```

Questo dà il valore 1 alla variabile **sì**, se premi il tasto S – altrimenti il valore è zero. Nota l'impiego del segno = nelle assegnazioni e anche in una condizione logica.

```
proc domanda
  nota ***** domanda articolo a magazzino *****
  scrivi
  immetti "Articolo di magazzino? ";nome$
  usa "mag"
  cerca nome$
  poni si=0
  mentre trovato() ed non si
    visualizza
    mostrasch
    conferma
    se non si
      continua
    finese
  finementre
  se non trovato()
    scrivi
    scrivi nome$; " inesistente"
  finese
  fineproc
```

Questa procedura localizza semplicemente il record corretto. Una procedura utilizzabile maggiormente per interrogare il file di magazzino è **chiedi**:

```
proc chiedi
  domanda
  spazza
  fineproc
```

Essa utilizza un'altra procedura, **spazza**, che attende fino a che premi un tasto, pulisce lo schermo, e poi presenta un elenco dei comandi che puoi usare. Tralasceremo questa procedura finché avremo scritto le procedure che deve elencare. Ricorda di uscire di volta in volta da **edita** per memorizzare queste procedure man mano che le scrivi.

Situazione magazzino

Puoi scrivere anche una semplice procedura per produrre un resoconto generale di magazzino.

```
proc resoconto
  nota ***** resoconto magazzino *****
  pulisci
  scrivi "ARTICOLO"; tab 25; "CODICE";
  scrivi tab 33; "QUANTITA"; tab 47; "PREZZO";
  scrivi tab 55; "VALORE DI MAGAZZINO"
  scrivi
  poni totale=0
  usa "mag"
  tutto
    scrivi descrizione$( ad 25);tab 27;mag.nummaga$;tab 35;
      quantità;
```

```
      scrivi tab 47;"£";prezvend; tab 59;"£";prezvend*quantità
      poni totale=totale+prezvend*quantità
      finetutto
   scrivi
   scrivi "Valore totale magazzino = £"; totale
   spazza
   fineproc
```

Registrazione vendite

Per registrare una vendita, dobbiamo soltanto sottrarre il numero di articoli venduti dal relativo record di magazzino. Si consiglia di inserire una qualche forma di conferma che trattiamo la voce di magazzino corretta e che la giacenza è sufficente a soddisfare l'ordine.

```
proc quantità
  nota ***** scrivi articoli a magazzino *****
  domanda
  se trovato()
    scrivi immetti "Quanti? "; num
    scrivi
    scrivi num;" * ";mag.nummaga$;" (";mag.descrizione$;")"
    finese
  fineproc

proc vendita
  nota ***** elabora vendita *****
  quantità
  se trovato()
    se num<=mag.quantità
      scrivi "Valore ordine:- £"; num*mag.prezvend
      conferma se si
        poni mag.quantità=mag.quantità-num
        aggiorna
        mostrasch: nota *** mostra il record modificato ***
        finese
      altrimenti
      scrivi "Giacenze non sufficienti"
      finese
    finese
  spazza
  fineproc
```

Registrazione di merci in arrivo

La seguente procedura permette di registrare la consegna di merci a magazzino. Anche in questo caso, ti chiede i particolari che immetti prima di accettarli e aggiornare il record di magazzino relativo.

```
proc consegna
  nota ***** nel caso merce in consegna *****
  quantità
  se trovato()
    conferma
    scrivi
    se si
      scrivi "Accettato"
      poni mag.quantità=mag.quantità+num
      aggiorna
      mostrasch
      altrimenti
      scrivi "Consegna non registrata"
      finese
    finese
  spazza
  fineproc
```

Riordino di merci

Fino a questo momento le nostre procedure si sono riferite solo al file di giacenze. Quando vogliamo ordinare nuova merce dovremo riferirci ai file del fornitore e degli ordini per trovare il nome e l'indirizzo del ditta, il prezzo, ecc.

Presumendo di aver identificato l'articolo nel file di magazzino, (con **domanda**) scegliamo, dal file degli ordini, i record che hanno il codice di magazzino corretto. Questi record contengono i codici di tutte le ditte che possono fornire questo articolo. Poiché i record contengono anche il prezzo e il tempo di consegna per ogni fornitore, possiamo decidere se vogliamo l'articolo più a buon mercato o il tempo di consegna più breve.

Usiamo **trova** per trovare rapidamente il record del fornitore richiesto. Questo significa che bisogna fare l'ordinamento del file del fornitore (rispetto al codice del fornitore, codice$) prima di usare **faordine**.

```
proc faordine
   nota ***** ordine rinnovo merce *****
   domanda
   se trovato()
      usa "ord"
      seleziona mag.nummaga$=ord.nummaga$
      scrivi
      scrivi "Consegna rapida o a buon prezzo (r/b)?";
      se maiusc(tasto())="R"
         rapido
         altrimenti : buonpr
         finese
      poni rif$=codiforn$
      ripris
      usa "forni"
      trova migliore$
      famodulo
      scrivi
      scrivi "Consegna prevista tra ";con;" gg"
      finese
spazza
fineproc
```

La procedura **buonpr** trova il fornitore con il prezzo più basso, e **rapido** funziona allo stesso modo per trovare il fornitore con il tempo di fornitura più breve.

```
proc buonpr
   nota ***** cerca il prezzo migliore *****
   usa "ord"
   poni pre=prezzo
   poni migliore$=codice$
   poni con=consegna
   tutto
      se prezzo<pre
         poni pre=prezzo
         poni migliore$=codice$
         poni con=consegna
         finese
      finetutto
   fineproc

proc rapido
   nota ***** consegna più rapida *****
   usa "ord"
   poni con=consegna
   poni migliore$=codice$
   poni pre=prezzo
   tutto
      se consegna<con
         poni con=consegna
         poni migliore$=codice$
         poni pre=prezzo
         finese
      finetutto
fineproc
```

La procedura **famodulo** produce l'effettivo modulo di ordinazione. Devi modificarla per adattarla ai tuoi requisiti. Useremo una versione semplice che mostra i particolari dell'ordine sullo schermo.

```
proc famodulo
  nota ***** produce modulo d'ordine *****
  pulisci
  scrivi
  scrivi forni.nomedit$
  scrivi forni.via$
  scrivi forni.cap$;" ";forni.città$
  scrivi
  scrivi "Vogliate fornire "; mag.quantacq;" ";
  scrivi mag.descrizione$;" (vostro rif. ";rif$;")";
  scrivi "a £"; pre; " per unità."
  scrivi
  scrivi "Valore totale: £"; mag.quantacq*pre
  fineproc
```

Abbiamo bisogno di un comando finale per chiudere tutti i file, una volta che abbiamo finito di usarli.

```
proc ciao
  conferma
  se si
    pulisci
    scrivi : scrivi "Operazione terminata. Arrivederci"
    chiudi "mag"
    chiudi "forni"
    chiudi "ord"
    pulisci
    finese
  fineproc
```

Possiamo scrivere una breve procedura per eseguire l'impiego. Essa deve aprire tutti e tre i file con i nomi corretti di file logici, pulire lo schermo e mostrarti i comandi aggiuntivi che hai. Nota che, nell'uso normale, il file di magazzino è il solo i cui record dovranno essere cambiati. Gli altri due file vengono aperti come file di sola lettura. Inoltre esso fa l'ordinamento del file fornitori in modo che possiamo **trovare** una ditta mediante il suo codice di riferimento.

```
proc start
  pulisci
  modo 0
  scrivi da 5,21; "PROGRAMMA DI GESTIONE MAGAZZINO"
  scrivi
  apri "giacenza" logico "mag"
  guarda "fornitor" logico "forni"
  guarda "ordini" logico "ord"
  usa "forni"
  ordina codice$; a
  spazza
  fineproc
```

Per finire possiamo scrivere **spazza**, che pulisce lo schermo e mostra un elenco dei comandi addizionali disponibili:

```
proc spazza
  nota ***** spazza schermo e mostra comandi *****
  locale x$
  scrivi
  scrivi "Premi un tasto per proseguire ";
  poni x$=tasto()
  pulisci
  scrivi
  scrivi "Comandi disponibili"
  scrivi "chiedi resoconto consegna faordine vendita ciao":
  scrivi
  scrivi "Scrivi la tua scelta"
  fineproc
```

CAPITOLO 12
CONSULTAZIONE DI QL ARCHIVE

VARIABILI

I nomi delle variabili possono essere lunghi fino a tredici caratteri, e non devono iniziare con una cifra (da 0 a 9). Essi possono contenere qualsiasi combinazione di lettere maiuscole e minuscole, o numeri. Non sono permessi altri caratteri, eccetto **$** e **.** che hanno significati speciali.

Se un nome di variabile finisce con una **$**, esso è una variabile stringa. Le stringhe possono essere lunghe fino a 255 caratteri. Se il nome non finisce con una **$**, la variabile è numerica. Un nome di variabile può riferirsi al contenuto di un record in un file ed è conosciuto come una variabile di campo. Si assume normalmente che le variabili di campo si riferiscano al file corrente, ma si può fare in modo che si riferiscano ad un altro file aperto, inserendo un nome di file logico, separato da un **.** dal nome della variabile. Una tale variabile di campo viene scritta come:

nome_file_logico . nome_campo

Per esempio **principale.continente$**. Se un nome di variabile comprende un punto, si deve riferire ad un campo in un file aperto. Se non c'è un punto si farà un tentativo di far corrispondere il nome ad una variabile esistente nella seguente sequenza:

1 un campo del file corrente
2 una variabile locale (un parametro nella procedura corrente, se esistente,)
3 una variabile globale

Viene dato un messaggio di errore se non viene trovata alcuna corrispondenza.

SINTASSI

Il termine *sintassi* si riferisce alla struttura esatta di un comando o funzione. La sintassi di un comando specifica i parametri necessari al comando, in quale ordine devono comparire, e i simboli (se esistenti) usati per separarli.

Questa sezione descrive la notazione usata per esprimere la sintassi del linguaggio di programmazione di Archive.

ESPRESSIONI

Una *espressione* è una combinazione di valori, variabili, funzioni ed operatori letterali che dà come risultato un valore singolo. Una *espressione numerica* dà come risultato un valore numerico ed una *espressione stringa* dà come risultato un valore di testo.

```
3 * y * sen(x) + lun(a$)        {numerica}
"abc" + a$ + ripro("-", 5)      {stringa}
```

Una espressione può, come nell'esempio di cui sopra, essere composta di parecchie sottoespressioni. In tale caso non puoi mescolare sottoespressioni di tipi differenti. Esse devono essere tutte espressioni stringa o tutte espressioni numeriche.

Convenzioni sintattiche

Le definizioni sintattiche sono simili a quelle usate per definire la sintassi di SuperBASIC, cioè

Simbolo	Significato
italico	indica una entità sintattica
[ ]	racchiude una voce opzionale
* *	racchiude le voci che possono essere ripetute
|	o
{ }	commento

Entità sintattiche

s.let	stringa letterale
esp.s	espressione stringa
esp.n	espressione numerica
esp	espressione, stringa o numerica
vsc	voce di scrittura
var	nome di variabile, stringa o numerica
nfl	nome di file logico
nff	nome di file fisico (fino a 8 catteri)
npr	nome di procedura

Una stringa letterale consiste di testo racchiuso tra virgolette, per esempio "testo" o "testo".

Una espressione stringa è una stringa letterale, o una combinazione di stringhe letterali, variabili di stringa e funzioni di stringa, che dà come risultato un valore di testo, per esempio:

```
"fred"+a$+car(72)
```

Una espressione numerica è costituita da un numero o una combinazione di numeri, variabili numeriche ed operatori, (+, −, *, /, ecc.) che dà come risultato un valore numerico. Per esempio:

```
(3+x)/sen(y)
```

Una voce di scrittura è costituita da quattro possibilità, **da, tab, inchiostro, carta.** Una descrizione completa di una voce di scrittura è nella nostra notazione sintattica :

voce__di__scrittura:= | **da** *n esp , n esp*
| **tab** *n esp*
| **inchiostro** *n esp*
| **carta** *n esp*

I nomi di file logici e i nomi di procedura hanno le stesse limitazioni dei nomi variabili. I nomi di file tipici non possono eccedere otto caratteri.

Come esempio di una *definizione di sintassi*, considera la sintassi del comando ordina. Nella nostra notazione esso appare come:

ordina spec:= var; a | d
ordina *ordina spec* *[, *ordinaspec*]*

Ordina, perciò, dev'essere seguito da almeno una *specifica d'ordina*, che consiste in una *variabile* separata da una virgola da una lettera che dev'essere una **a** o **d**. Inoltre, puoi anche inserire fino a tre ulteriori specifiche d'ordina, a condizione che ciascuna coppia sia separata da virgole. Chiaramente la notazione sintattica fornisce una descrizione molto più compatta.

Nota che la notazione sintattica non ti dice il significato o scopo dei simboli, in modo che dovrai leggere il resto della descrizione di ciascun comando. La sintassi ti dà soltanto una descrizione formale del numero e genere di voci che costituiscono un comando valido. In aggiunta, la notazione sintattica, non ti dice il numero massimo di ripetizioni permesse per le voci ripetute. **Ordina** accetterà fino a quattro coppie di una variabile e una lettera.

FILE DI DATI DI ARCHIVE

Un campo

Un *campo* è lo spazio riservato a contenere una stringa o un numero.

In Archive, ogni campo è identificato da un nome di variabile di campo.Se un campo specifico può contenere una stringa o un numero dipende dal nome dato al campo nel momento in cui è stato creato – il nome dei campi stringa finisce con una **$**. Un campo stringa di Archive può contenere fino a 255 caratteri. Il nome di un campo numerico non finisce con un segno di **$**. Tutti i numeri vengono immagazzinati nello stesso spazio, indipendentemente dal loro valore. La gamma possibile per un numero è la stessa della gamma numerica valida per le operazioni aritmetiche.

Un record

Un *record* è una raccolta di campi, i cui contenuti sono collegati in qualche modo. I campi di un record possono, per esempio, essere usati per conservare il nome, l'indirizzo ed il numero di telefono di una particolare persona. In Archive i record sono di *lunghezza variabile*, per cui ciascun record occupa solo lo spazio necessario per alloggiare le informazioni contenute nei suoi campi. In un record di Archive ci possono essere fino a 255 campi.

Un file

Un *file di dati* è formato da un certo numero di record collegati. Per continuare l'esempio di cui sopra, un file di dati potrebbe consistere in una raccolta di record contenenti nome, indirizzo e numero di telefono di molte persone differenti. Il numero di record in un file di dati di Archive è limitato a circa 15.000. In pratica, sei limitato dalla capacità di una cartuccia di Microdrive, che può contenere circa circa 1.000 record di 100 caratteri. Un file è l'unità di base che puoi memorizzare, su una cartuccia di Microdrive, o che puoi richiamare da essa. Per identificare un file si usa un nome. In Archive dai un nome fisico al file al momento in cui lo crei, ma puoi cambiarne il nome logico in qualsiasi momento.

Apertura e chiusura di file

Quando vuoi leggere o scrivere un file di dati, devi per prima cosa *aprirlo*. In generale, l'apertura di un file di dati trasferisce una copia del file dalla cartuccia del Microdrive alla memoria, anche se, nel caso di un file lungo, è possibile che solo parte del file sia presente in memoria in un momento specifico.

Puoi aprire un file di dati nel modo di *sola lettura* con **guarda**, che, come suggerito dal nome, significa che non puoi cambiarne il contenuto. Inoltre, hai l'opzione di aprire un file di dati nel modo *aggiorna* con **apri**, in modo di poterne leggere e cambiare il contenuto.

Ogni volta che apri un file di dati, Archive riserva dello spazio per le variabili di campo necessarie ad un record entro il file. Le variabili di campo contengono sempre i valori del record corrente.

Quando chiudi un file di dati con **chiudi** o **lascia**, tutte le variazioni da te fatte vengono copiate nel file registrato nella cartuccia di Microdrive. La copia conservata in memoria viene scartata. La chiusura di un file è il solo modo di garantire che la copia nella cartuccia di Microdrive contenga la tua versione più recente. Poiché una file aperto usa parte della memoria dell'elaboratore, non devi lasciare file aperti se non li stai usando.

Quando esci da Archive con il comando **lascia**, tutti i file aperti vengono chiusi automaticamente.

Non spegnere l'elaboratore, o togliere una cartuccia dal Microdrive, mentre la cartuccia contiene dei file aperti.

Nomi di file logici

Ogni file di dati aperto ha un *nome di file logico*, che gli viene dato quando il file viene aperto. Se quando apri un file non specifichi un nome logico, gli viene dato automaticamente il nome di file logico "principale".

Il nome di file logico viene usato per identificare un file specifico quando usi contemporaneamente parecchi file.

PROCEDURE

Una procedura consiste in una sezione di programma dotata di nome, che inizia con una dichiarazione di procedura avente la forma:

proc *npr*[; *var* *[, *var*]*]

e che finisce con:

fineproc

Può essere indicata col suo nome da qualsiasi altro programma o procedura, compresa se stessa. Essa si comporta come se il suo codice fosse stato inserito nel punto da cui viene chiamato.

In Archive, i comandi **proc** e **fineproc** non possono essere immessi direttamente dalla tastiera, ma vengono aggiunti automaticamente quando usi l'editore di programma per creare una procedura.

L'EDITORE DI PROGRAMMA

Si ottiene accesso all'editore di programma usando il comando **edita**.

Se nella memoria dell'elaboratore non sono presenti procedure, ti verrà immediatamente offerta l'opzione di creare una nuova procedura. Altrimenti, sul lato sinistro della zona

di visualizzazione, ti viene dato un elenco di tutte le procedure in memoria. La prima procedura è evidenziata ed è elencata per esteso sul lato destro dello schermo. La prima riga della procedura evidenziata per marcare la procedura corrente e la riga corrente.

Una volta in **edita**, hai cinque opzioni:

Seleziona una procedura
Premi **TABULATE** per spostarti verso il basso nell'elenco di procedure, premi **SHIFT** e **TABULATE** per spostarti verso l'alto nell'elenco. Il listato sullo schermo mostra sempre la procedura corrente.

Seleziona una riga
Usa i tasti di movimento verticale del cursore per selezionare una riga nella procedura corrente. La riga corrente viene evidenziata.

Premi F3 per il menù dei comandi di edizione.
Esistono quattro comandi, che vengono selezionati premendo il tasto corrispondente alla prima lettera.

Cancella
Premi **ENTER** per cancellare la procedura evidenziata sulla sinistra dello schermo. Premi qualsiasi altro tasto per uscire dal comando senza cancellare la procedura.

Nuovo
Scrivi il nome della nuova procedura e premi **ENTER**. Se esiste già una procedura con quel nome, ti verrà offerta l'occasione di editarla.

Taglia
Toglie del testo dalla procedura corrente e lo trasferisce al buffer di incolla. Prima di chiamare questo comando, usa i tasti di movimento verticale del cursore, in modo che la prima (o l'ultima) riga della zona da togliere diventi la riga corrente. Poi usa i tasti di movimento verticale del cursore per marcare la regione di testo da togliere. Premi **ENTER** per trasferire il testo nel buffer di incolla.

Incolla
Copia il contenuto del buffer di incolla nella procedura corrente sotto la riga corrente. **Incolla** svuota il buffer di incolla.

Inserimento di testo
Premi **F4** per inserire una o più righe di testo sotto la riga corrente della procedura. Scrivi il testo e premi **ENTER**. Se premi **ENTER** senza alcun testo precedente, uscirai dall'opzione di inserimento.

Edizione di testo
Premi **F5** per editare la riga corrente della procedura corrente. La riga di testo viene copiata nella riga di immissione e può essere modificata con l'editore di riga. Premi **ENTER** per sostituire la riga vecchia con la riga nuova.

L'EDITORE DI SCHERMO

Si accede all'editore di schermo con il comando **edisch**. Esso ti permette di disegnare un nuovo tracciato di schermo o modificarne uno esistente. Una volta che hai disegnato un tracciato, puoi memorizzarlo in una cartuccia di Microdrive mediante il comando **memosch** e richiamarlo mediante il comando **risch**.

Un tracciato di schermo è composto da due parti, il testo di sfondo fisso e i valori variabili che vengono visualizzati su di esso. Il comando **schermo** mostra il testo di sfondo e il comando **mostrasch** aggiunge i valori correnti delle variabili che contiene.

Edisch ha due opzioni:

Scrivi il testo nello sfondo dello schermo
premi **F3** per usare un comando di edizione di schermo.

Dopo aver premuto **F3** sono disponibili quattro comandi di edizione di schermo:

P – pulisci lo schermo
V – marca una regione per definire una variabile
I – imposta il colore dell'inchiostro
C – imposta il colore della carta.

Il tracciato dello schermo viene reso attivo da:

risch
schermo

Quando un particolare schermo è attivo, esso mostrerà i valori correnti delle sue variabili dopo **mostrasch**, o quando la tastiera riprende il controllo, dopo aver eseguito un programma (o un comando). Il tracciato dello schermo viene reso inattivo ripulendo lo schermo con **pulisci**. Se non c'è uno schermo attivo, **mostrasch** non ha alcun effetto. La memoria dell'elaboratore può contenere solo un tracciato di schermo alla volta.

Il comando **visualizza** crea e usa il suo proprio tracciato di schermo. Esso perciò sostituirà qualsiasi altro tracciato di schermo con il suo disegno.

I COMANDI

Sono disponibili i seguenti comandi.

TUTTO

Esamina tutti i record del file presenti logicamente nel tempo più rapido possibile.

Sintassi: **tutto** [*nfl*] : ... : **finetutto**

In generale, questo esame non sarà in una sequenza particolare. Il nome di file logico opzionale lo obbligherà a riferirsi ad un file aperto specificato. Se non viene dato il nome di file logico, esso esaminerà il file corrente.

Il loop **tutto** è disegnato principalemente per esaminare i record del file, piuttosto che per cambiarli. Non usare **aggiorna** all'interno di un loop **tutto**, a meno di non essere sicuro che la lunghezza del record rimarrà invariata. Per esempio puoi cambiare il valore di un numero, o convertire un campo di testo in maiuscole. Se sei in dubbio, usa un loop **mentre** – usando il valore di **fdf()** per trovare la fine del file. Per esempio

```
primo
mentre non fdf( )
   ...
   aggiorna
   ...
   prox
   filementre
```

MODIFICA

Modifica il tracciato dello schermo corrente in modo che visualizzi i valori correnti delle variabili.

Sintassi: **modifica**

Puoi cambiare il contenuto di uno o più campi qualsiasi del file corrente, i cui valori sono mostrati sul tracciato dello schermo. Nota che non è necessario che tutte le variabili di campo vengano mostrate. Non puoi cambiare un campo che non viene mostrato. Se nessuna delle variabili di schermo compare sullo schermo, Archive obbliga una **visualizzazione** del file.

Per prima cosa seleziona il campo da cambiare premendo **TABULATE** o **ENTER** fino a che il cursore sia nel campo corretto (le variabili che non siano campi del file vengono saltate). Successivamente puoi scrivere un nuovo valore o usare l'editore di riga per modificare il valore esistente. Premi **TABULATE** o **ENTER** per portarti al campo successivo. (Se premi contemporaneamente **SHIFT** e **TABULATE** ritorni al campo precedente.)

Una volta fatti tutti i cambiamenti che vuoi, premi **F5** per mettere il nuovo record al posto del vecchio. La sostituzione del record è automatica se premi **ENTER**. Se il file è ordinato, la nuova versione del record viene inserita in sequenza.

AGGIUNGI

Aggiunge un record al file specificato, o al file corrente se non viene dato il nome del file logico.

Sintassi: **aggiungi** [*nfl*]

I campi del record prendono i valori correnti delle variabili di campo. Se il file è ordinato, l'inserimento è in sequenza.

PRECEDENTE

Si muove indietro di un record nel file specificato, o nel file corrente se non viene dato il nome di file logico.

Sintassi: **precedente**[*nfl*]

BACKUP

Fà una copia del file specificato. Devi fare delle copie di tutti i tuoi file, come salvaguardia contro danni o cancellazioni accidentali.

Sintassi: **backup** *vecchionf* **come** *nuovonf*

CHIUDI

Chiude il file specificato, o il file corrente se non viene specificato nessun nome di file.

Sintassi: **chiudi** [*nfl*]

PULISCI

Pulisce la zona di visualizzazione e spegne qualsiasi schermo di visualizzazione. Vedi **schermo**, **risch**, **mostrasch**.

Sintassi: **Pulisci**

CONTINUA

Continua il comando **ricerca** o **cerca** precedente dal record successivo a quello corrente nel file in use.

Sintassi: **continua**

CREA

Crea un file aperto, dotato di nome, i cui record contengono i campi dati dall'elenco di variabili specificato nel comando. Hai l'opzione di specificare un nome di file logico – se non lo fai, il file viene creato con il nome di file logico "principale".

Sintassi: **crea** *nmf* [**logico:** *nfl*] : *var* *[: *var*]* : **finecrea**

CANCELLA

Cancella il record corrente dal file specificato, o dal file corrente se non viene dato nessun nome di file logico.

Sintassi: **cancella** [*nfl*]

Attenzione: Usa questo comando con attenzione poiché non puoi recuperare il record cancellato.

ELENCO

Mostra l'elenco dei file nella cartuccia di Microdrive

Sintassi: **elenco** [*drive*]

Puoi specificare che il Microdrive sia **mdv1__** o **mdv2__**. Se non scrivi il nome del Microdrive, Archive elencherà automaticamente i file della cartuccia nel Microdrive 2.

Prima di mostrare l'elenco di file, Archive mostra il nome del volume della cartuccia (il nome che gli hai dato quando l'hai formattata).

VISUALIZZA

Visualizza il nome di file logico del file corrente e una lista dei nomi di campo ed i valori delle variabili di campo del record corrente. Se il file è ordinato, esso mostra anche i campi di ordinamento e le loro priorità di ordinamento.

Sintassi: **visualizza**

Il comando mette questa lista al posto di qualsiasi tracciato dello schermo definito dall'utente, ed essa diventa il tracciato attivo dello schermo.

OUTPUT

Sintassi: **output** [; *var*] *[, *var*]*

Scrive i campi specificati dei record selezionati del file corrente come uscita in forma tabulare su **ser1**. Se non dai una lista di nomi di variabili di campo, vengono scritti *tutti* i campi.

Puoi inviare l'uscita ad un file di Microdrive con **sispool**.

EDITA

Chiama l'editore di procedura per creare una nuova procedura o per editare una procedura esistente.

Sintassi: **edita**

FINETUTTO

Vedi **tutto.**

FINECREA

Vedi **crea.**

ERRORE

Sintassi: **errore** *npr*

Marca una procedura allo scopo di trattare gli errori. Qualsiasi errore che si verifichi durante l'esecuzione di questa procedura, o qualsiasi procedura chiamata da essa, provoca un ritorno prematuro dalla procedura marcata. La precedura può determinare la natura dell'errore usando la funzione **numerr( )**, per leggere il numero dell'errore. Questo numero di errore viene cancellato tutte le volte che viene eseguito **errore**.

ESPORTA

Memorizza i campi nominati dei record selezionati del file di Archive corrente in una cartuccia di Microdrive, in una forma adatta per l'importazione in QL Abacus o QL Easel.

Sintassi: **esporta** *nmf* [; *var*] *[, *var*]* [**quill**]

Se non specifichi una lista di nomi di variabili di campo, vengono esportati *tutti* i campi. Se non inserisci il parametro opzionale **quill**, (separati dall'ultimo nome di variabile da almeno uno spazio), il file viene esportato in una forma adatta all'importazione da parte di QL Quill.

Il file di esporta viene chiamato *nmf* e, a meno che tu specifichi il tuo suffisso di nome di file, Archive usa il suffisso **__ESP**.

Vedi la *Sezione informativa* per una discussione completa di importa ed esporta.

CERCA

Va all'inizio del file e cerca il primo record che contiene una corrispondenza alla stringa specificata in qualsiasi campo stringa. La corrispondenza è indipendente dal fatto che il testo sia in caratteri maiuscoli o minuscoli.

Sintassi: **cerca** *esp.s*

Puoi continuare la ricerca con il comando **continua**, e determinare se la ricerca ha avuto successo esaminando il valore reso dalla funzione **trovato( )**.

PRIMO

Cerca il primo record del file specificato, o il file corrente se non specificato nessun nome di file logico.

Sintassi: **primo** [*nfl*]

FORMATTA

Formatta la cartuccia nel Microdrive 2 (il drive destro). Esso dà alla cartuccia il nome da te specificato. Questo nome viene riportato quando, successivamente, usi **elenco** per mostrare l'elenco dei file in quella cartuccia.

Sintassi: **formatta** "*specificato da te*"

SE

Permette ad una condizione specificata di controllare l'elaborazione successiva.

Sintassi: **se** *esp.n* : ... [: **altrimenti** : ...] : **finese**

Senza **altrimenti**.
Se l'espressione è diversa da zero, vengono eseguite le seguenti istruzioni. Se l'espressione è zero, l'esecuzione viene trasferita all'istruzione che segue **finese**.

Con **altrimenti**.
Se l'espressione numerica è diversa da zero, vengono eseguite le istruzioni tra **se** e **altrimenti**. Altrimenti vengono eseguite le istruzioni tra **altrimenti** e **finese**. In entrambi i casi l'esecuzione continua con le istruzioni che seguono **finese**.

IMPORTA

Legge un file, *nome1*, esportato da QL Abacus o QL Easel, e produce in file di dati di Archive *nome2*. In modo analogo a **apri** e **guarda**, hai l'opzione di specificare un nome di file logico per il file di dati.

Sintassi: **importa** *nome1* **come** *nome2* [**logico** *nfl*]

in cui: *nome1*:= *nmf*
nome2:= *nmf*

Vedi la *Sezione informativa* per una descrizione completa di importa ed esporta.

INCHIOSTRO

Imposta il colore di scrittura del testo successivo nel colore specificato dal valore dell'espressione.

Sintassi: **inchiostro** *esp.n*

I colori sono: 0 e 1 nero
2 e 3 rosso
4 e 5 verde
6 e 7 bianco

Se l'espressione dà il valore superiore a 7, il valore preso è il resto, dopo una divisione per 8. Per esempio **inchiostro** 9 è equivalente a **inchiostro**1, ed entrambi impostano il colore di stampa a nero. Se inchiostro viene usato entro un comando **scrivi**, esso cambierà solo il colore di scrittura per la durata di quel comando.

IMMETTI

Richiede l'immissione dalla tastiera alle variabili elencate nel comando. Ogni variabile di una lista d'immissione può essere preceduta da una stringa iniziale, che verrà visualizzata come messaggio d'immissione. Tutte le voci d'immissione devono essere separate l'una dall'altra da punti e virgole. Se la lista ha un punto e virgola finale, il cursore non si porterà su una nuova riga dopo l'immissione.

Sintassi: **immetti** [*var* | *s.let* | *svo* *[; *var* | *s.let* | *svo*]*] ;]

L'elenco di voci d'immissione può comprendere le voci di posizionamento del cursore

da *riga,colonna*
tab *colonna*

in cui: *riga*:=*esp.n*,
colonna:=*esp.n*

La prima di esse posiziona il cursore alla riga e colonna specificata, e **tab** porta il cursore alla colonna specificata all'interno della riga corrente. Se il cursore è già a destra della colonna specificata, tab non avrà alcun effetto.

Queste due voci non possono essere usate al di fuori dei comandi **immetti** o **scrivi**.

Puoi usare anche inchiostro e carta come voci d'immissione. Se usati all'interno di un comando d'immissione, essi influiranno sui colori dell'inchiostro e della carta fino alla fine dell'immissione, dopo di che i colori ritorneranno alle loro impostazioni originarie.

INSERIMENTO

Aggiunge un nuovo record ad un file.

Sintassi: **inserimento**

Usa il tracciato corrente dello schermo per visulizzare i valori correnti delle variabili. Puoi scrivere un nuovo valore per uno qualsiasi o più campi del file corrente, i cui valori siano riportati nel tracciato dello schermo. Nota che non è necessario che siano mostrate tutte le variabili di campo. Non puoi scrivere un valore per un campo che non è visualizzato. Se nessuna delle variabili del campo comparre nello schermo, Archive obbliga una **visualizzazione** del file.

Per prima cosa scegli un campo premendo **TABULATE** o **ENTER**, fino a che il cursore sia nel campo corretto (i valori che non sono campi del file vengono saltati). Puoi poi scrivere un nuovo valore. Premi **TABULATE** o **ENTER** per portarti al campo successivo. (se premi contemporaneamente **SHIFT** e **TABULATE** ritorni al campo precedente.)

Dopo aver scritto tutti i valori che vuoi, devi premere **F5** per aggiungere il nuovo record al file. Il record verrà aggiunto al file anche se non premi **ENTER**, quando il cursore si trova nell'ultimo campo. Tutti i campi a cui non avrai dato un valore saranno zero (se un campo numerico) o una stringa vuota (se un campo di testo). Se il file è ordinato, il nuovo record viene inserito in sequenza, altrimenti l'inserzione avviene in una posizione non specificata.

DISTRUGGI

Cancella il file specificato dalla cartuccia di Microdrive.

Sintassi: **distruggi** *nmf*

Attenzione: Usa questo comando con attenzione in quanto non puoi recuperare il file cancellato.

ULTIMO

Cerca l'ultimo record del file specificato, o del file corrente, se non specifichi un nome di file logico.

Sintassi: **ultimo** [*nfl*]

PONI

Usato per assegnare un valore ad una variabile.

Sintassi: **poni** *var* = *esp*

LISTA

Elenca su una stampante tutte le procedure attualmente in memoria.

Sintassi: **lista**

RICHIAMA

Richiama nella memoria dell'elaboratore il file di procedura specificato da una cartuccia del Microdrive.

Sintassi: **richiama** [**oggetto**] *nmf*

LOCALE

All'interno di una procedura, obbliga la lista di variabili seguente a diventare le variabili locali. Queste variabili esistono solo entro le procedure in cui esse sono dichiarate e sono indefinite in qualsiasi altra procedura. I loro valori vengono distrutti all'uscita dalla procedura.

Sintassi: **locale** *var* *[, *var*]*

TROVA

Trova, in un file che è stato ordinato, il primo record il cui contenuto del campo corrisponde alla/e espressione/i.

Sintassi: **trova** *esp* * [,*esp*]*

Il record viene trovato molto più rapidamente che se usassi **cerca**, ma **il file dev'essere stato ordinato in precedenza**. Ogni espressione deve riferirsi esplicitamente al contenuto di un particolare campo di ordinamento. Nel caso di un campo stringa la corrispondenza dipende dalla scrittura maiuscola o minuscola.

Se hai fatto l'ordinamento del file nei confronti di più di un campo, puoi specificare parecchie espressioni (una per ciascun campo di ordinamento). Le espressioni sono separate da virgole e si devono riferire ai campi usati per ordinare il file. Essi devono essere nella stessa sequenza del comando **ordina** antecedente. Per esempio:

```
ordina animale$ ; a, peso ; a
trova "Elefante", 2000
```

troverà il primo record in cui il campo animale$ contiene il testo "Elefante" e un peso pari (o superiore) a 2000.

Trova individua un record anche se non c'è una corrispondenza esatta. Questo record sarà il primo in cui il contenuto del campo "supera" – nel senso dell'ordinamento (cioè, "d" viene dopo "e" se il file ordinato in ordine discendente) – i valori specificati.

GUARDA

Apre il file specificato solo per lettura. Se il nome di file logico non viene specificato, gli viene dato automanticamente il valore "principale".

Sintassi: **guarda** *nmf* [**logico** *nfl*]

STAMPA

Stampa su una stampante collegata a SER1 i valori della lista di voci seguente, allo stesso modo di **lista**.

Sintassi: **stampa** [*esp* | *svo* *[; *esp* | *svo*]*] [;]

UNIONE

Aggiunge le procedure del file di programma specificato alle procedure che si trovano già nella memoria dell'elaboratore. Se il file contiene una procedura dello stesso nome di una già in memoria, la nuova procedura sostituisce quella vecchia.

Sintassi: **unione** [oggetto] *nmf*

Se inserisci l'**oggetto** opzionale, Archive esporterà il file in formato binario, piuttosto che ASCII. Vedi **Memorizza**.

MODO

Cambia la forma della visualizzazione.

Sintassi: **modo** *var*,*var*

La prima variabile può avere un valore di 0 o 1. Un valore di 0 unisce le zone dei comandi, visualizzazione e lavoro in una zona unica. Il valore di 1 le separa nuovamente in zone distinte.

La seconda variabile può avere il valore di 4, 6 o 8 e cambia lo schermo per una visualizzazione a 40, 64 o 80 caratteri per riga.

L'impostazione iniziale, quando richiami Archive per l'impiego con un monitor, è equivalente a:

```
modo 1,8
```

NUOVO

Cancella tutti i dati dalla memoria dell'elaboratore, pronto per una nuova partenza. Tutti i file aperti sono chiusi. (Il comando non cancella i file immagazzinati nella cartuccia di Microdrive.)

Sintassi: **nuovo**

PROX

Va al record successivo nel file specificato, o nel file corrente se non è specificato un nome di file logico.

Sintassi: **prox** [*nfl*]

APRI

Apre il file specificato sia per la lettura che per la scrittura. Il file riceve il nome di file logico "principale" se non ne specifichi uno.

Sintassi: **apri** *nmf* [**logico** *nfl*]

ORDINA

Ordina i record del file in conformità al contenuto dei campi specificati.

Sintassi: **ordina** *ordina_spec* *[, *ordina_spec*]*

in cui : *ordina_spec*:= *var*; **a** | **d**

Il primo campo specificato nella lista è il campo di ordinamento primario. I record, i cui campi di ordinamento primario hanno contenuto uguale, vengono ulteriormente ordinati secondo il contenuto del campo successivo della lista (se stato specificato) e così via. Per ciascun campo specificato bisogna dare la direzione ordinamento. Essa dev'essere **a** o **d**, per specificare rispettivamente l'ordine ascendente o discendente.

L'ordine prende in considerazione solo i primi 8 caratteri del campo di testo e non puoi specificare più di quattro campi per **ordinare** il file.

CARTA

Imposta il colore di sfondo di tutto il testo successivo nel colore specificato dal valore dell'espressione.

Sintassi: **carta** *esp.n*

I colori sono: 0 e 1 nero
2 e 3 rosso
4 e 5 verde
6 e 7 bianco

Se l'espressione dà come risultato più di 7, il valore preso è il resto dopo la divisione per 8, cioè **carta** 11 equivalente a **carta** 3, ed entrambi impostano il colore rosso.

Se **carta** viene usata entro un comando **scrivi**, essa cambia solo il colore di sfondo per la durata di quel comando.

POSIZIONA

Fa diventare record corrente quello il cui numero dato dall'espressione.

Sintassi: **posiziona** *n.exp*

SCRIVI

Visualizza i valori della lista di voci seguente – che deve essere separata da punti e virgole – sullo schermo. Se la lista ha un punto e virgola finale, il cursore non si porterà su una riga successiva dopo la visulizzazione. Vedi anche **stampa**.

Sintassi: **scrivi** [*esp* | *vsc*]*[; *esp* | *svo*] *[;]

LASCIA

Chiude tutti i file e rimanda a SuperBASIC.

Sintassi: **lascia**

NOTA

Quando usato in una procedura, indica che il resto della riga contiene un commento. Il testo seguente su quella riga viene ignorato al momento dell'esecuzione della procedura.

Sintassi: **nota**

RIPRIS

Questo comando ripristina tutti i record del file corrente, che erano stati tolti mediante un precedente impiego di **seleziona**. Esso distrugge l'eventuale ordinamento del file.

Sintassi: **ripris**

RITORNA

Usato con una procedura, per provocare la fine immediata della procedura, ritornando alla procedura di chiamata.

Sintassi: **ritorna**

MEMORIZZA

Memorizza in una cartuccia di Microdrive tutte le procedure attualmente nella memoria dell'elaboratore, creando un file dotato di nome unico.

Sintassi: **memorizza** [**oggetto**] *nmf*

Se comprendi l'**oggetto** opzionale, Archive memorizza il file in un formato binario invece che ASCII. Questo significa che Archive non deve convertire il programma in caratteri ASCII prima di memorizzarlo, ed perciò molto più rapido. Puoi **richiamare, eseguire** o **unire** un tale programma, aggiungendo l'**oggetto** opzionale al comando appropriato. Queste operazioni funzioneranno più rapidamente anche perché non è necessaria alcuna conversione. Questi file hanno il suffisso __pro, piuttosto che il normale prg.

Puoi anche memorizzare questo programma **oggetto** in una forma che sia protetta contro l'esame o la modifica. Includi, invece di **oggetto**, il **proteggi** opzionale. Un programma memorizzato in questo modo può essere richiamato, eseguito o unito solo con l'uso dell'**oggetto** opzionale con il comando appropriato.

Un programma protetto non può essere listato, editato o memorizzato. Se unisci un programma protetto con qualsiasi altro programma, la combinazione sarà similarmente protetta. Il solo modo di eliminare lo stato di protetto è di utilizzare il comando **nuovo**.

La memorizzazione della versione protetta non influisce sulla copia del programma nella memoria dell'elaboratore. Puoi ancora listare, editare o memorizzare il programma nel modo normale.

SCHERMO

Visualizza il tracciato dello schermo formattato richiamato in precedenza. Non fà nulla se non presente alcun tracciato di schermo. Non visualizza alcuna variabile nello schermo.

Sintassi: **schermo**

RICERCA

Esplora il file corrente dall'inizio fino a che viene trovato un record per il quale l'espressione specificata sia vera. Questo record diventa quello corrente.

Sintassi: **ricerca** *esp.n*

EDISCH

Chiama l'editore di schermo, per permetterti di definire un nuvo tracciato di schermo. Vedi il capitolo 7.

SELEZIONA

Esamina tutto il file selezionando solo i record per i quali è vera l'espressione specificata. Il file si comporta successivamente come se fossero presenti solo i record selezionati.

Sintassi: **seleziona** *esp.n*

Puoi ripristinare tutti i record scartati col comando **ripris**.

INSCH

Attende l'immissione delle variabili nella lista seguente, usando l'ordine specificato nella lista. Tutte le variabili della lista devono essere visualizzate in quel momento in un tracciato attivo di schermo.

Sintassi: **insch** *var* *[, *var*]*

RISCH

Richiama un tracciato di schermo definito e memorizzato in precedenza. Inoltre visualizza il tracciato dello schermo e attiva la visualizzazione delle variabili dello stesso.

Sintassi: **risch** *nmf*

I valori visualizzati vengono poi aggiornati automaticamente tutte le volte che il controllo ritorna da una procedura all'interprete di tastiera.

NOSPOOL

Dirige verso la stampante tutte le uscite che seguono **stampa** e **lista**. Questo annulla l'effetto di **sispool**.

Sintassi: **nospool**

SISPOOL

Dirige verso il file specificato – o verso lo schermo – tutte le uscite che seguono **stampa, lista** e **output** , invece che verso la stampante.

Sintassi: **sispool** *nmf* [**esporta** | *output*]
o:
sispool schermo

Se stai inviando l'uscita ad un file, viene diretta attraverso il programma di comando della stampante attualmente installato, per cui contiene tutti i codici speciali di cui ha bisogno la tua stampante.

Se inserisci l'**esporta** opzionale, Archive garantisce che il file contenga solo codici ASCII stampabili, gli a capo e gli avanzamenti di riga. Il file risultante è adatto per l'importazione in Quill.

L'**output** opzionale permette al testo di essere trasmesso al file senza essere elaborato dal programma di comando della stampante. In questo caso tutti i codici ASCII (compresi i codici di comando) vengono trasmessi immediatamente al file.

A meno che tu specifichi un suffisso di nome di file, Archive presume il suffisso **__ele** (__esp o __out se inserisci gli **esporta** o **output** opzionali).

La forma alternativa del comando – **sispool schermo** – invia l'uscita allo schermo del monitor invece che alla stampante.

MOSTRASCH

Usato all'interno di una procedura per obbligare la visualizzazione dei campi del record corrente.

Sintassi: **mostrasch**

Ci deve essere un tracciato attivo di schermo (il tracciato dello schermo viene reso attivo da un uso precedente di **schermo**, **risch** o **visualizza**). Se non c'è un tracciato di schermo attivo, il comando non avrà nessun effetto.

MEMOSCH

Memorizza come tracciato di schermo, definito la zona visualizzata in quel momento, in un file con nome nella cartuccia di Microdrive.

Sintassi: **memosch** *nmf*

Esso memorizza il testo dello schermo ed una lista delle variabili dello schermo, assieme alla loro posizione.

FERMA

Arresta l'esecuzione di tutte le procedure e restituisce il comando alla tastiera.

Sintassi: **ferma**

TRACCIA

Attiva e disattiva il modo traccia.

Sintassi: **traccia**

Scrivi:

```
traccia
```

per attivare la traccia. Nel modo traccia, ogni riga del programma viene visualizzata nella zona di lavoro dello schermo, man mano che viene eseguita. Premi la barra spaziatrice e tienila premuta per arrestarla. La traccia continuerà quando lasci andare la barra spaziatrice. Per disattivare nuovamente la traccia scrivi:

```
traccia
```

AGGIORNA

Sostituisce il record corrente nel file specificato (o il file corrente se non viene dato nessun nome di file logico) con un record che contiene i valori correnti delle variabili di campo.

Sintassi: **aggiorna** [*nfl*]

USA

Prende il file specificato e lo fa diventare il file corrente.

Sintassi: **usa** *nfl*

MENTRE

Esegue ripetutamente le istruzioni tra **mentre** e **finementre** per tutto il tempo in cui il valore dell'espressione è diverso da zero (vera).

Sintassi: **mentre** *esp.n* : ... : **finementre**

FUNZIONI

Pensa ad una funzione come una specie di ricetta che trasforma uno o più valori iniziali, conosciuti come gli *argomenti* della funzione, in un valore differente, che viene indicato come il valore *dato* dalla funzione.

Le funzioni fornite da Archive possono prendere nessuno, due o tre argomenti. Gli argomenti di una funzione vengono messi tra parentesi dopo il suo nome. Non devi lasciare spazi tra il nome e la parentesi aperta, ma sono permessi spazi tra le voci all'interno delle parentesi. Se una funzione prende più di un argomento, essi vengono separati da virgole. Tutte le funzioni devono essere seguite dalle parentesi, anche se non prendono alcun argomento. La presenza delle parentesi serve a rammentare che ti riferisci ad una funzione. Essi ti permettono di distinguere tra una variabile ed una funzione, anche se hanno lo stesso nome.

Vengono fornite le seguenti funzioni.

ASS(*esp.n*) Dà il valore assoluto dell'argomento, ignorando l'eventuale segno meno.

ATN(*esp.n*) Dà l'angolo, in radianti, la cui tangente è esp.n.

CAR(*esp.n*) Questa funzione dà il carattere ASCII il cui codice è esp.n. Un carattere con un codice ASCII inferiore a 32 viene inviato alla stampante solo se preceduto da uno zero ASCII. Per esempio:

stampa car(0)+car(13)

invia alla stampante il carattere ASCII per un ritorno di carrello. Questo utile se la tua stampante ha bisogno di sequenze di codice di comando per produrre effetti speciali – consulta il manuale della tua stampante per eventuali codici speciali di cui abbia bisogno.

Per esempio, puoi inviare allo schermo una "A" con:

stampa car(65).

CODICE(*esp.s*) Esso dà valore ASCII del primo carattere trovato nel testo specificato.

COS(*esp.n*) Dà il coseno delll'angolo dato (in radianti).

CONTA([*nfl*]) Dà il conteggio del numero di record del file corrente.

DATA(*esp.n*) Dà la data odierna sotto forma di stringa di testo, in una di queste tre forme:

esp.n	stringa di data
0	"AAAA/MM/GG"
1	"GG/MM/AAAA"
2	"MM/GG/AAAA"

Devi aver impostato precedentemente l'orologio del sistema, come descritto nella *Guida delle parole chiave di SuperBASIC*.

GIORNI(*esp.s*) Dà il numero di giorni, dal primo gennaio 1583 alla data che viene data come espressione di testo della forma "AAAA/MM/GG". La conversione presume che venga usato il calendario gregoriano (moderno). Perciò la formula valida soltanto per date successive al 1582.

DECI(*valore,pd,larghezza*)
valore:=(esp.n)
pd:= (esp.n)
larghezza:= (esp.n)

Converte il *valore* numerico dato nella stringa di testo equivalente, in formato decimale, con *pd* posizioni decimali. Il testo viene giustificato a destra in un campo di *larghezza* caratteri. Per esempio:

dec(1.23e1,3,10) dà il testo " 12.300" (con 4 spazi davanti).

GRAD(*esp.n*) Prende un angolo, misurato in radianti, e lc converte nello stesso angolo in gradi.

FDF([*nfl*]) Dà un valore che indica se hai cercato di leggere oltre la fine del file corrente, o del file specificato se viene dato un idenficatore di file. Il valore dato è 1, se hai cercato di leggere oltre la fine del file, altrimenti è zero.

NUMERR() Dà il numero dell'ultimo errore verificatosi (zero come numero di errori è significa nessun errore). Il numero di errori lo stesso di quello visualizzato assieme al messaggio di errore quando Archive comunica l'individuazione di un errore.

ESP(*esp.n*) Dà il valore di e (circa 2.2718) elevato alla potenza di (*esp.n*). Il valore dato, sarà errato se *esp.n* è superiore a +88, in quanto il risultato supererà il limite numerico di Archive.

NOMCAM(esp.n|, *nfl*|)
Dà il nome del campo specificato nel record corrente del file specificato (o del file corrente se non viene dato nessun nome di file logico). Nota che **nomcam(0)** dà il nome del primo campo.

TIPCAM(*esp.n* [, *nfl*])
Dà il tipo del campo specificato nel record corrente del file specificato (o del file corrente se non viene dato nessun nome di file logico). Nota che **tipcam(0)** dà il tipo del primo campo.
Dà il valore 0 se il campo numerico, altrimenti dà 1.

VALCAM(*esp.n*[, *nfl*])
Dà il valore del campo specificato del record corrente del file specificato (oppure del file corrente se non viene dato nessun nome difile logico). Nota che **valcam(0)** dà il valore del primo campo.

TROVATO() Dà uno se viene trovato un record usando **ricerca** o **cerca**, altrimenti dà zero.

GEN(*valore*,*larghezza*)
valore:=esp.n
larghezza:=esp.n

Converte il *valore* numerico dato nella stringa di testo in un formato generale. Il testo viene giustificato a destra, in un campo di *larghezza* caratteri. Per esempio:

```
gen(1.23e1,10)
```

dà il testo " 12.3" (preceduto da 6 spazi).

TASTO() Attende che venga premuto un tasto e dà un carattere singolo di testo che corrisponde al tasto premuto.

PREMUTO() Dà il carattere singolo di testo corrispondente al tasto premuto quando viene chiamata la funzione. Non attende l'azionamento di un tasto, ma dà una stringa nulla ("") se non viene premuto nessun tasto.

STRGINT(*principale*,*sotto)*
principale:= *esp.s*
sotto:= *esp.s*

Trova la prima volta che *sotto* viene incontrato in *principale* e dà la posizione del primo carattere di *sotto* in *principale*. Esso darà un valore di zero se non viene trovata alcuna corrispondenza. La corrispondenza dipende dal carattere maiuscolo o minuscolo.

```
strgint("Gennaio","Gen")     {dà 1}
strgint("Gennaio","en")      {dà 2}
strgint("Gennaio","EN")      {dà 0}
```

INT(*esp.n*) Dà il valore intero del numero, troncandolo al punto decimale. Il troncamento opera sempre verso zero. Per cui

```
int(3.7)      {dà 3}
int(-4.8)     {dà -4}
```

LUN(*esp.s*) Dà il numero di caratteri del testo specificato.

MINUSC(*esp.s*) Converte il testo specificato in lettere minuscole.

MEMORIA( ) Dà il numero di byte inutilizzati che rimangono in memoria.

MESE(*esp.n*) Dà, come testo, il nome di un mese.

Per esempio, **mese(3)** dà il testo "Marzo".

Se viene usato un argomento maggiore di 12, esso viene sostituito dal resto dopo la divisione per 12, per cui, per esempio, **mese(13)** e **mese(1)** daranno entrambi il risultato "Gennaio".

NUM(*valore, larghezza*)

valore:= esp.n
larghezza:= esp.n

Converte il *valore* numerico dato nella stringa di testo in formato intero. Il testo viene giustificato a destra, in un campo di *larghezza* caratteri. Per esempio:

num(1.23e1,10) dà il testo " 12" preceduto da 8 spazi).

NUMCAM([*lfn*])

Dà il numero di campi dei record del file specificato (o del file corrente se non dai un nome di file logico).

PI() Dà il valore della costante matematica pi greca.

RAD(*esp.n*) Prende un angolo, misurato in gradi, e lo converte in radianti.

NUMREC([*nfl*])

Dà il numero (contando il primo record come zero) del record corrente del file specificato (o del file corrente se non dai un nome di file logico).

RIPRO(*esp.s*,*esp.n*)

Questa funzione dà una striga composta di un certo numero di copie del primo carattere del testo dato. Il testo risultante può essere lungo fino a 255 caratteri. Per esempio,

```
scrivi ripro("*",5)      {scriverà cinque asterischi}
scrivi ripro("abc",3)      {scriverà "aaa"}
```

SGN(*esp.n*) Dà +1, –1 o 0, a seconda se l'argomento è positivo, negativo o zero.

SEN(*esp.n*) Dà il valore del seno dell'angolo specificato (in radianti).

QUAD(*esp.n*) Dà la radice quadrata dell'argomento, che non deve essere negativo.

STR(*n*,*tipo*,*pd*) *n*:=*esp.n*
tipo:=*esp.n*
pd:=*esp.n*

Converte il numero, n, nella stringa di testo equivalente.

Il secondo parametro, tipo, indica la forma della stringa convertita nel modo seguente:

0 decimale (punto mobile)
1 notazione esponenziale, o scientifica
2 intero
3 formato generale

Il terzo parametro, pd, indica il numero di cifre che seguono il punto decimale nella stringa convertita. Esso deve essere sempre specificato, sebbene il suo valore sia ignorato nei formati interi e generali.

Per esempio:

```
poni a$=str(12.3456,0,2)   {dà ad a$ il valore "12.35"}
poni a$ str(12.3456,1,4)   {dà ad a$ il valore "1.2346e1"}
```

TAN(*esp.n*) Dà la tangente dell'angolo (in radianti) specificato.

ORA( ) Dà, sotto forma di testo, l'ora del giorno nel formato "OO:MM:SS". Devi avere precedentemente impostato l'orologio del sistema, come descritto nella Guida delle parole chiave di SuperBASIC.

MAIUSC(*esp.s*) Converte la stringa specificata in lettere maiuscole.

VAL(*esp.s*) Converte il testo nel suo valore numerico. Essa convertirà solo testo composto di caratteri numerici validi e la conversione si fermerà al primo carattere che non può essere interpretato come un numero. Per esempio, **val("1.1ABC")** darà il valore numerico 1.1, e **val("ABC")** darà 0.0.

VALVAR(*esp.s*) Dà il valore della variabile il cui nome viene dato da *esp.s* – per esempio:

```
poni a="lun"
poni lunghezza=15
scrivi valore(a"gth")
```

scriverà il valore 15.

Nota che valore(numcam(y)) esattamente equivalente a valcam(y).

ERRORI

Quando ARCHIVE individua un errore in un comando digitato alla tastiera o in una procedura, visualizza un numero di errore ed un breve messaggio di errore. Esempi di errori che verrebbero individuati sono:

tentativo di divisione per zero, **se** senza il corrispondente **finese**, assegnazione del numero di parametri errati ad una procedura.

Se l'errore deriva dall'immissione da tastiera, il testo dell'istruzione rimane visibile nella zona di lavoro. Puoi premere **F5** per ripetere il testo, in modo da poter usare l'editore di riga per correggere l'errore. Successivamente puoi premere **ENTER** per eseguire l'istruzione corretta. Se l'errore deriva da una istruzione di programma, ARCHIVE mostra il nome della procedura e la riga in cui si verificato l'errore. A questo punto puoi usare l'editore di programma per correggere l'errore.

Quando usi il comando **errore** nei tuoi programmi, ARCHIVE non riporta gli errori da esso individuati in una procedura marcata con **errore**. Sei libero di trattare tale errore in qualsiasi modo tu voglia (puoi anche ignorarlo). Puoi trovare quale errore si è verificato esaminando il valore dato da **numerr( )**. Questo è numero uguale a quello dato da ARCHIVE quando visualizza un messaggio di errore.

L'elenco seguente mostra i numeri d'errore di ARCHIVE, assieme ai messaggi corrispondenti. Dove possibile, l'elenco comprende un breve esempio di una istruzione che darebbe l'errore. I messaggi di errore non sono destinati a focalizzare l'errore, ma sono intesi a darti un'idea di che tipo di errore cercare.

I messaggi di errore per i quali non c'è un breve esempio sono segnati da un asterisco. Essi vengono trattati nelle note che seguono l'elenco.

No.	Messaggio	Esempio
0	nessun errore	
1	comando non riconosciuto	aggiungi
2	istruzione incompleta	poni x=3 poni y=4
3	nome della variabile assente	poni 31=x
4	voce di stampa non riconosciuta	scrivi crea
5	tipo di dati sbagliato	* (1)
6	espressione numerica richiesta	poni x="fred"
7	espressione alfanumerica richiesta	poni x$=4
8	variabile non trovata	poni x=qq (qq non definita)
9	variabile non trovata	scrivi qq
10	separatore mancante	scrivi da 5
11	nome troppo lungo	poni nomelunghissimo=4
12	nome duplicato	crea:n$:n$:finecrea
13	attesa stringa letterale	* (2)
14	fineproc mancante	* (3)
15	instruzione proc scorretta	* (3)
16	fine d'istruzione prematura	crea"prova" ENTER ENTER
17	struttura di programma scorretta	* (4)
18	troppi numeri	* (5)
50	virgolette finali mancanti	poni x$="fred
51	esponente mancante dopo 'E'	poni x=1.2E
52	numero troppo grande	poni x=1.2E100
53	simbolo sconosciuto	poni x=%
70	errore di sintassi di valutatore	poni x=3+
71	parentesi mancante	poni x=(3+5)/7)
73	tipo non corrispondente	poni x$="fred"+3

74	numero di argomenti scorretto	`poni x$=str(1,2)`
75	stringa troppo lunga	`poni x$=rept("*",256)`
76	divisione per zero	`poni a=0: poni x=5/a`
77	argomenti di funzione scorretti	`poni x$=quad(-4)`
78	errore d'indice di stringa	`poni x$="fred" (ad 97)`
80	memoria satura	* (6)
90	manca spazio per aprire file	* (7)
91	trasferimento di file incompleto	* (8)
93	fuori dei limiti	`scrivi da 100,100;37`
94	file non aperto	`aggiungi` (senza aver aperto il file)
100	impossibile aprire file	`guarda"xxx"` (inesistente)
101	scrittura su file sola lettura	`guarda "nomi":inserimento`
103	tipo di file sbagliato	`risch"nomi"` (file di dati)
104	nome di file scorretto	`memorizza"3prova"`
105	errore di lettura di file	* (9)

Note

1) La causa più probabile dell'errore 5 – "tipo di dati sbagliato" è l'immissione del testo quando è atteso un numero, per esempio in risposta ad una istruzione **immetti** quale:

```
immetti x
```

2) Errore 13 – "attesa stringa letterale" – può verificarsi per esempio, durante l'importazione di un file costruito da te stesso (senza usare nessuno dei comandi **esporta** dei programmi QL). Esso significa che Archive ha trovato un numero, o una espressione numerica o letterale, dove si aspettava di trovare un valore di testo letterale. Nella maggior parte delle situazioni in cui Archive trova dei dati numerici quando si aspetta del testo, o viceversa, darà l'errore 7 o 8.

3) Errori 14 – "fineproc mancante" – e 15 – "istruzione proc scorretta" – non dovrebbero mai verificarsi nell'uso normale. Essi indicano che Archive ha individuato un **fineproc** mancante o un errore nella struttura di una istruzione **proc** in una procedura. Si possono verificare solo se costruisci un file di programma con un editore che non sia quello di Archive.

4) Errore 17 – "struttura di programma scorretta" – di solito indica che un **tutto**, **se** o **mentre** non ha un **finetutto**, **finese** o **finementre** in una procedura. Puoi anche generare questo errore inserendo un **fineproc** dentro la struttura di un altro programma, o usando **ritorna** direttamente dalla tastiera.

5) Errore 18 – "troppi numeri" – indica che stai cercando di immettere più numeri di quello che la memoria riservata all'immissione può contenere. L'errore può verificarsi in una riga d'immissione dalla tastiera, oppure durante il caricamento di un programma che comprende una procedura contenente molti numeri in una delle sue righe. Il limite esatto dipende dalle circostanze – un limite tipico è di 15 a 20 numeri, per cui non ci sono molte probabilità di fare questo errore.

6) Errore 80 – "memoria satura" – dovrebbe essere dato solo se usi un programma molto grande. La grandezza di un file di dati normale non è limitato dalla memoria dell'elaboratore, in quanto soltanto una parte di un file si trova in memoria in un particolare momento. Se Archive ti dà questo errore, dovrai ridurre la grandezza del tuo programma prima di continuare. Se necessario, puoi spezzare il tuo programma in parecchie sezioni, in file differenti, ed usare **unione** per richiamare ciascuna sezione, quando necessaria. Comunque, questa tecnica richiederà una considerevole abilità di programmazione.

7) Errore 90 – "manca spazio per aprire un file" – si verifica quando la zona di memoria riservata da Archive per immagazzinare informazioni interne sui file attualmente in memoria si riempie. Questo può succedere anche se c'è ancora memoria disponibile (cioè, se il valore dato da memoria() non si avvicina ancora a zero).

8) Errore 91 – "trasferimento di file incompleto" – significa che il richiamo o la memorizzazione di un file non è riuscito per una qualche ragione. Questo può significare che i dati sono stati corrotti, o che la cartuccia o il Microdrive stata danneggiata.

9) Errore 105 – "errore di lettura di file" – significa che alcuni dei dati in un file sono nel formato sbagliato, nell'ordine sbagliato, o sono stati corrotti. Ci sono possibilità che questo succeda solo se costruisci il tuo file di importazione – o il tuo file di programma, senza usare l'editore di programma di Archive (impieghi avanzati).

QL Easel

CAPITOLO 1
CENNI SU QL EASEL

QL Easel è *completamente interattivo*, il che significa che il risultato di tutto ciò che fai è visualizzato subito. Per cominciare è sufficiente scrivere una serie di valori perchè questi siano visualizzati graficamente. Non devi mai preoccuparti di dover impostare tabelle di valori; di questo se ne occupa Easel per te, conservandoli – fuori vista.

Puoi immettere *testo* sul grafico nello stesso modo di come immetti dati e, una volta scritto, lo puoi cambiare o spostare come vuoi (con facilità!) fino a quando non sei soddisfatto del risultato.

Easel è organizzato in una serie di livelli e presenta una struttura *piramidale*. Il livello superiore, che è disponibile appena inizi ad usare Easel, ti permette di effettuare le operazioni più comunemente necessarie, per esempio, immisssione di dati o di testo. La completa potenza di Easel diventa evidente mentre ti familiarizzi con i comandi.

Anche se potente, Easel rimane semplice per l'uso a tutti i livelli. Non hai bisogno di ricordare molti comandi, dal momento che vieni guidato attraverso ogni operazione da una sequenza di *suggerimenti* (prompts) che ti spiegano cosa puoi fare in ogni momento. In particolare Easel offre la possibilità di *disegna con gli esempi* che ti permette di selezionare o di effettuare qualsiasi disegno da una singola riga o barra ad un completo grafico, mediante la semplice selezione da una serie di illustrazioni. Con questa possibilità sarai sempre sicuro di come apparirà la visualizzazione finale del grafico.

Se, in qualsiasi momento non sei sicuro di cosa fare, ricordati che puoi chiedere Aiuto premendo il tasto **F1**. Ricordati anche che puoi cancellare qualsiasi operazione non completata premendo il tasto **ESC**.

CAPITOLO 2
INIZIO

RICHIAMO DI QL EASEL

Richiama QL Easel come descritto nella Introduzione ai Programmi QL. Al richiamo, Easel mostra il seguente messaggio:

```
QL EASEL
grafica gestionale
versione x.xx
Copyright ©1984 PSION Ltd
Tutti i diritti riservati
```

dove x.xx il numero della versione, es. 2.23.

Di tanto in tanto, Easel leggerà più informazioni dalla cartuccia. **Non devi rimuovere la cartuccia dal Microdrive 1 fino a quando non hai finito di usare Easel e sei ritornato a SuperBASIC.**

PRESENTAZIONE

Appena hai richiamato Easel la visualizzazione dovrebbe essere come quella illustrata alla Figura 2.1. La visualizzazione è divisa in tre finestre principali, conosciute come la finestra dei comandi, la finestra di stato e la finestra di visualizzazione.

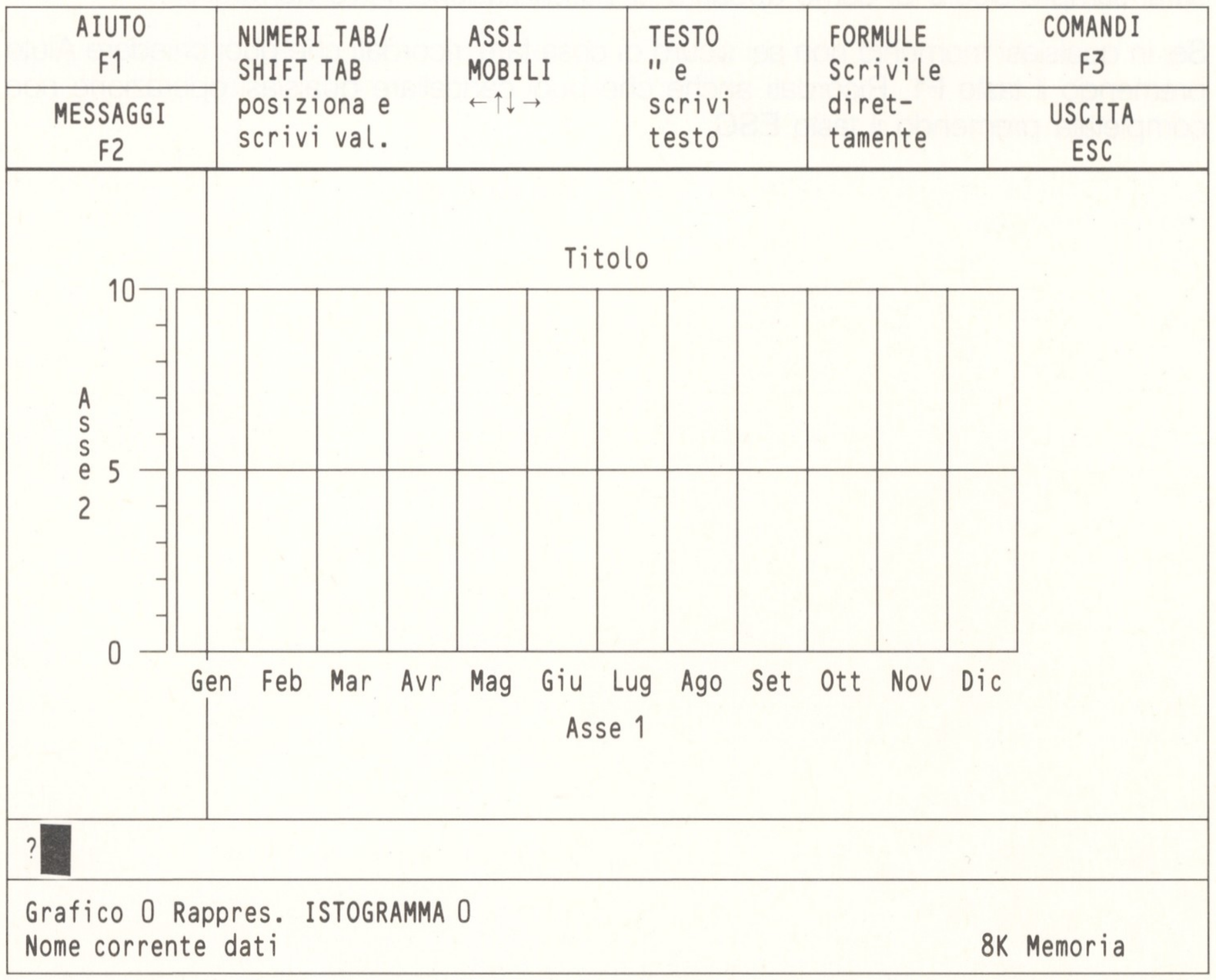


Figura 2.1 La visualizzazione principale

La finestra di stato

Il *formato* t'informa come verranno visualizzati i valori che scrivi. Vi sono otto diversi formati per la rappresentazione (numerati da 0 a 7) fra cui puoi effettuare la scelta, predefiniti in modo da offrire visualizzazione grafica a barre, a righe o a torta. Inizialmente, il formato di rappresentazione è quello che ti permette di visualizzare grafico a barre (grafico 0).

Visualizzazione grafica (grafico 0)

Vengono anche date informazioni sulla serie dei dati (o valori) del grafico. Se hai più di un grafico ci sarà una predeterminata serie di valori per ogni grafico. La serie corrente di valori è la serie che viene cambiata quando scrivi dei numeri.

Oltre a questo vieni anche informato dello *stile* che verrà usato. Easel può mostrare una serie di valori in una delle tre rappresentazioni come istogramma cartesiano oppure grafico a torta. Inizialmente Easel seleziona l'istogramma e usa il numero relativo al grafico a barra 0 (vi sono 16 disegni disponibili per la tua scelta). Lo spazio di memoria disponibile in qualsiasi momento viene visualizzato nella finestra di stato insieme ai *messaggi di errore* quando necessario.

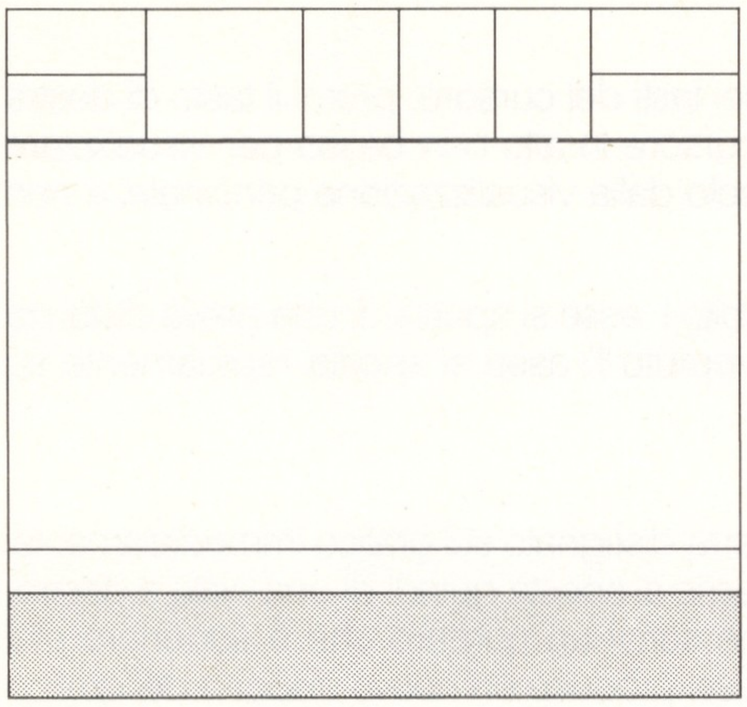


Figura 2.2 La finestra stato

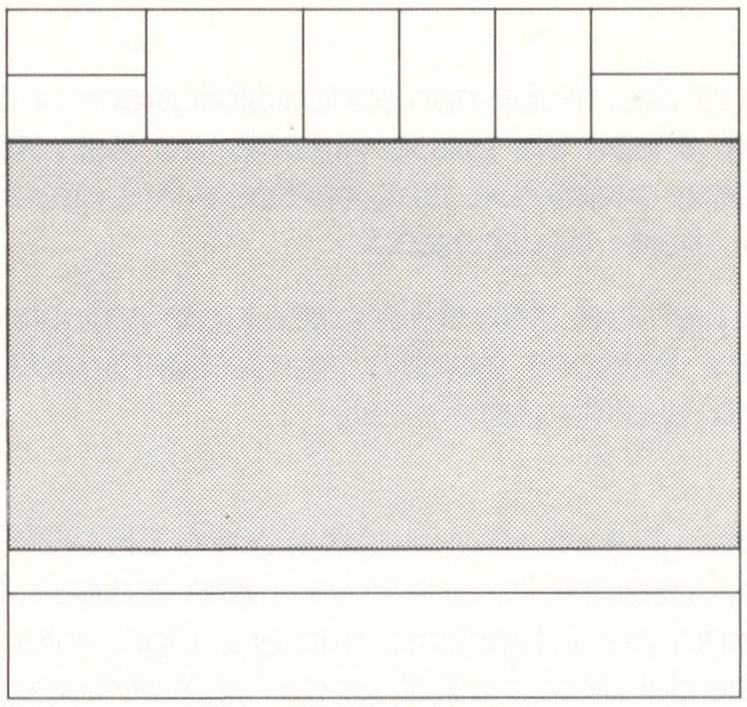


Figura 2.3 La finestra visualizzazione

La finestra di visualizzazione

Tutti i grafici riportati da Easel vengono visualizzati nella finestra di visualizzazione. Inizialmente, nella finestra di visualizzazione c'è un istogramma marcato con righe orizzontali e verticali. Le righe orizzontali corrispondono ai valori che appaiono sull'asse verticale ("Asse 2") mentre le righe verticali dividono il grafico in *celle*. Ogni cella contrassegna la posizione in cui verrà tracciato ogni valore di una serie di valori.

Ogni cella ha un *nome* (etichetta), lungo l'asse orizzontale ("Asse 1"). Easel automaticamente ne fornisce il testo "Gen", "Feb" e così via, fino a "Dic". Se vuoi, puoi cambiare il testo con qualsiasi altro nome che desideri.

Immagina ogni serie di dati come una riga di celle, ognuna contenente i valori che devono essere disegnati.

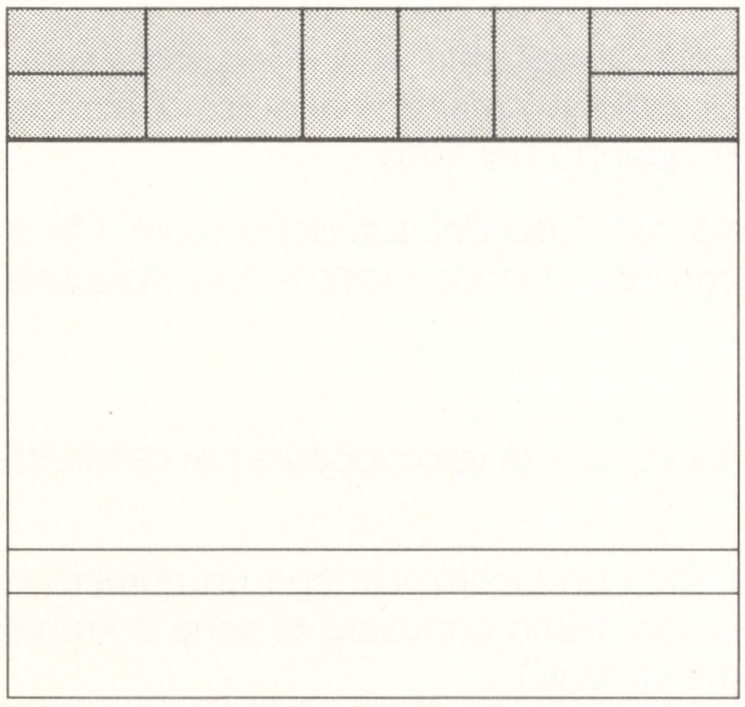


Figura 2.4 La finestra dei comandi

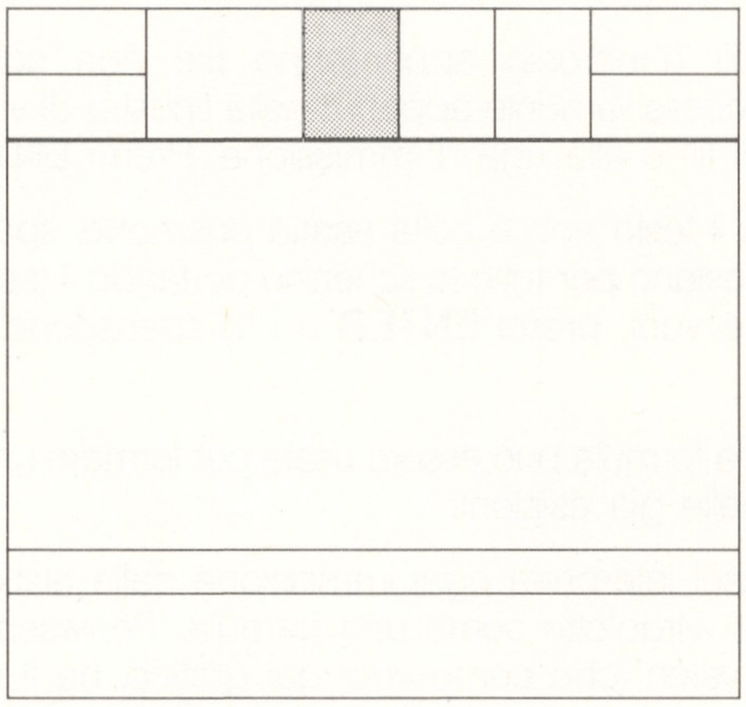


Figura 2.5 Gli assi mobili d'incrocio

La finestra dei comandi

La finestra comandi mostra le normali opzioni: Aiuto (**F1**), per attivare o disattivare i suggerimenti (prompts) (**F2**), per selezionare un comando (**F3**) e per cancellare una selezione non completata (**ESC**). Oltre a questo, vi sono quattro opzioni che sono proprie di Easel. Queste sono:

sposta gli assi mobili
scrivi un numero,
scrivi testo,
scrivi una formula.

GLI ASSI MOBILI

Premi il tasto del cursore opportuno e tienilo premuto per alcuni istanti. Vedrai che l'asse mobile verticale si sposta lungo lo schermo, da sinistra a destra. I tasti cursore spostano gli assi mobili a sinistra, a destra, in alto e in basso.

Puoi indicare qualsiasi punto nella zona di visualizzazione spostando gli assi mobili in quel punto.

Oltre a questo, l'asse mobile verticale marca la posizione nel grafico dove un numero che scrivi verrà disegnato in formato grafico.

Se gli assi mobili non sono visibili premi uno dei tasti del cursore; premi il tasto di destra o di sinistra per l'asse verticale, e il tasto del cursore in alto o in basso per visualizzare e'asse orizzontale. Nota che puoi fare questo solo dalla visualizzazione principale, e non dal menu dei comandi.

Se premi un tasto del cursore e poi lo rilasci subito l' asse si sposta di una breve distanza nella direzione indicata, ma se tieni il tasto premuto l' asse si sposta rapidamente su tutta la zona determinata.

NUMERI

Scrivi un numero qualsiasi (e premi **ENTER**). Verrà disegnato sul grafico immediatamente, alla posizione in cui si trova l' asse verticale. L' asse si sposta quindi di una cella a destra, pronto per il prossimo numero. Ogni volta che scrivi un numero che è superiore alla serie dei valori mostrati lungo l'asse verticale, Easel produce un nuovo disegno del grafico con una scala graduata che permette la visualizzazione del nuovo valore.

Ogni volta che premi il tasto **TABULATE**, l' asse mobile si sposta a destra di una cella. Tieni premuto il tasto **SHIFT** e premi il tasto **TABULATE**, e l' asse verticale si sposta a sinistra di una cella. La posizione del filo d'incrocio marca la cella (*corrente*) – la cella in cui sarà inserito il numero che scriverai. Puoi correggere un valore sbagliato spostando l' asse verticale alla cella dove appare l'errore e scrivendo il valore giusto. Se ti accorgi di un errore prima di premere il tasto **ENTER** lo puoi correggere con l'editore di riga. Oppure puoi cancellare l'errore premendo **ESC** e poi scrivere il valore giusto.

Se sposti il filo d'incrocio con il tasto **TABULATE** oppure con i tasti del cursore, il valore che scrivi successivamente sarà sempre visualizzato nella cella che contiene l' asse verticale.

TESTO

Puoi aggiungere testo al grafico scrivendo virgolette doppie o singole ("oppure") come primo carattere dell'immissione.

I fili d'incrocio appariranno (se non sono già visualizzati) e il testo che scrivi successivamente apparirà nella finestra di visualizzazione iniziando dal punto d'incrocio dei fili e alla riga d'immissione. Premi **ENTER** quando hai finito.

Se il testo non è nella esatta posizione, spostalo con l'uso dei tasti del cursore. I fili si spostano per tutto lo schermo portando il testo con loro. Quando il testo è nella posizione che vuoi, premi **ENTER** e i fili spariscono.

FORMULE

Una formula può essere usata per formare una nuova serie di valori, oppure per cambiare quelle già esistenti.

Easel interpreta ogni immissione dalla tastiera che non incomincia con un numero o con virgolette come una formula. Per esempio, possiamo cambiare la serie corrente di valori (che come vedi dal grafico, ha il nome di "dati").

```
dati = dati + 2 ENTER
```

Il nuovo grafico simile a quello precedente, tranne che per i valori che sono stati aumentati di 2. Se vuoi ritornare al grafico originale, scrivi un'altra formula:

```
dati = dati - 2 ENTER
```

Una formula inizia sempre col nome di una serie di valori. Questo può essere Il nome di una serie già esistente oppure un nuovo nome. In un caso o nell'altro il contenuto di quella serie di dati viene definito dall'espressione a destra del segno uguale (=) nella formula. E' importante rendersi conto che la formula modifica tutti i valori della serie, anzich è un solo valore.

I COMANDI

I comandi ti permettono di usare alcuni dei più sofisticati aspetti di Easel. Premi il tasto **F3** per selezionare il comando. La finestra comandi cambia e ti mostra un elenco di comandi disponibili – il menù dei comandi.

Quando il menù dei comandi viene visualizzato puoi scegliere un comando battendo la prima lettera del comando che vuoi usare.

Per esempio, Il comando **Lascia** abbandona Easel e ritorna al SuperBASIC. Per scegliere questo comando premi **F3** seguito da **L**. Easel ti offre l'opzione di premere **ESC** per rimanere in Easel (questo in caso tu abbia scelto il comando per errore). Se decidi che effettivamente vuoi lasciare Easel, allora premi **ENTER**.

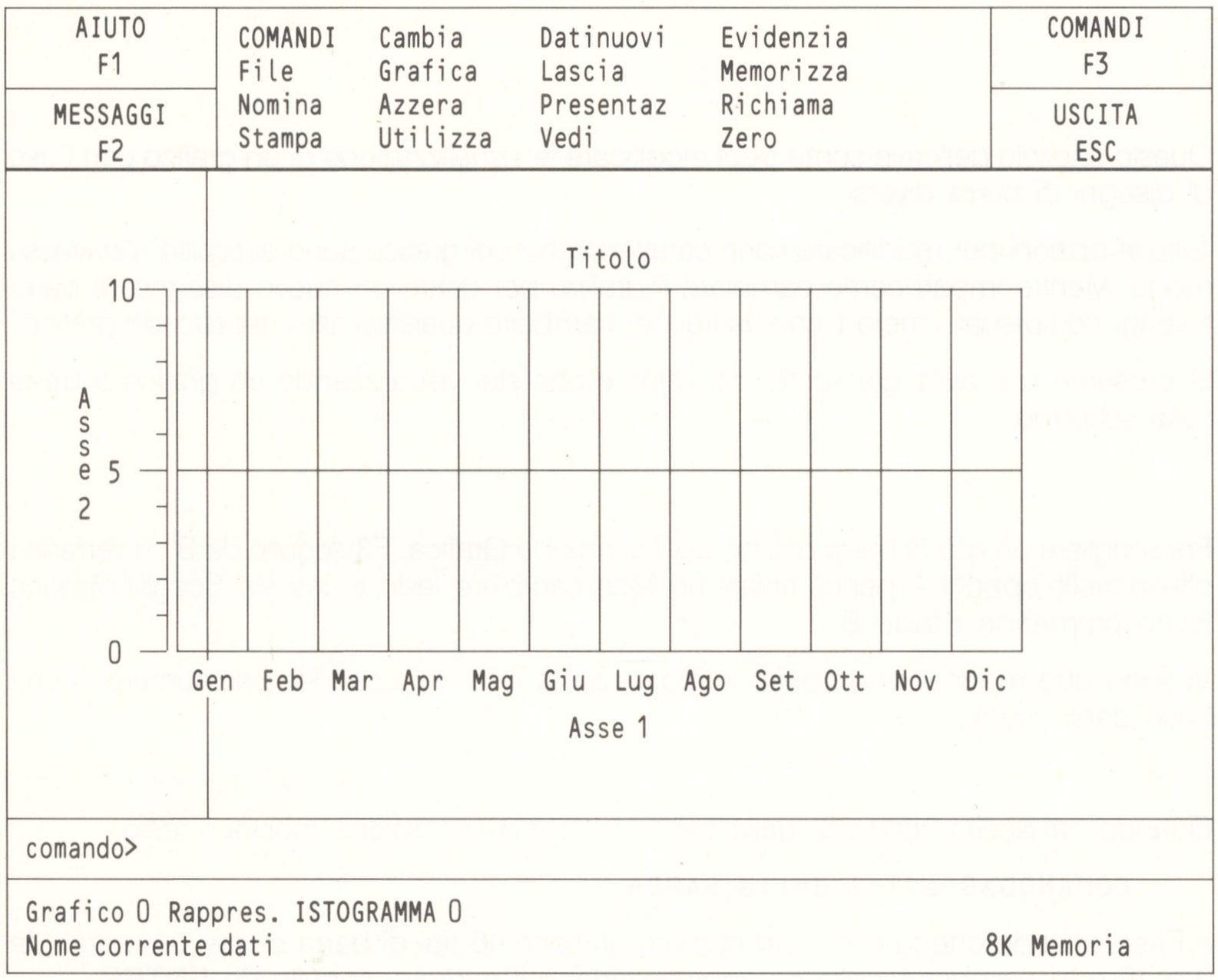


Figura 2.6 Il menù dei comandi

Quando il menù dei comandi visibile, non puoi scrivere alcun numero in una cella ne' scrivere una formula. Non puoi spostare i fili d'incrocio, eccetto quando ti viene data questa opzione come parte del comando.

Al completamento dell'esecuzione del comando, Easel rimane nel menù dei comandi. Per ritornare alla visualizzazione principale devi premere il tasto **ESC**.

CANCELLA UN VALORE

Puoi cancellare un valore dal grafico. Usa il tasto **TABULATE** (o i tasti **SHIFT** e **TABULATE**) per posizionare il filo verticale sul numero che vuoi cancellare e poi premi **F4**. Se il grafico mostra più di una serie di valori, tutti i valori in quella cella verranno cancellati quando premi **F4**. Esso non ha nessun effetto sui valori non visualizzati. Se cancelli i valori da una cella che non ha nessun nome (etichetta), allora quella cella non sarà inclusa nel grafico quando lo disegni di nuovo.

Easel non cancella una cella che non ha un nome e un valore. Se vuoi cancellare una cella devi cancellarne il contenuto ed anche l'etichetta. Quella cella non sarà inclusa quando tracci di nuovo il grafico con il comando **Vedi**.

IMMISSIONE DI UN VALORE

Puoi inserire un nuovo valore alla destra di quello segnato dall'asse verticale. Premi il tasto **F5** e uno spazio si apre, pronto a ricevere il numero che tu scriverai. La nuova cella non avrà etichetta, ma tu gliene puoi dare una.

L' immissione e la cancellazione di valori da un grafico a torta è un po' diversa ed è spiegata al capitolo 9.

CAPITOLO 3

DISEGNO DI UNA BARRA

Questo capitolo descrive come puoi modificare la visualizzazione di un grafico con l'uso di disegni di barra diversi.

Tutte le opzioni per modificare varie caratteristiche del grafico sono eseguite allo stesso modo. Mentre impari come cambiare il grafico per usare un nuovo disegno di barra ti vengono spiegati i metodi che userai per cambiare qualsiasi altro aspetto del grafico.

Si presume che avrai già scritto dei valori e che stai visualizzando un grafico a barre sullo schermo.

SCEGLI UN TIPO DI BARRA

Per scegliere un tipo di barra diverso usi il comando **Grafica**, **F3** seguito da **B**. Ti verranno offerte molte opzioni – per cambiare un Asse, cambiare Testo, e così via. Scegli l'opzione Barra premendo il tasto **B**.

Vi sono due modi per cambiare il tipo di barra che stai usando: per numero o con l'uso degli esempi.

Selezione per numero

Quando hai scelto l'opzione della barra, la riga d'immissione mostra il testo:

```
COMANDO>Grafica della BARRA ?
```

e Easel attende che tu scriva un numero. Vi sono 16 tipi di barra diversi, numerati da 0 a 15 e tu ne puoi scegliere uno scrivendone il numero, seguito da **ENTER**.

Questo è un metodo molto veloce per cambiare il tipo di barra che stai usando, purchè conosca il numero relativo alla barra che vuoi.

Scelta con l'uso degli Esempi

Se non sai il numero relativo al tipo di barra che vuoi usare, oppure vuoi creare un disegno per conto tuo, premi **ENTER**, invece di scrivere un numero. Prova questo metodo battendo

F3 G B ENTER

(Non c' bisogno che premi **F3** se sei ancora nel menù dei comandi.) La visualizzazione cambia per mostrare esempi dei tipi di barra disponibili, insieme ai relativi numeri di riferimento.

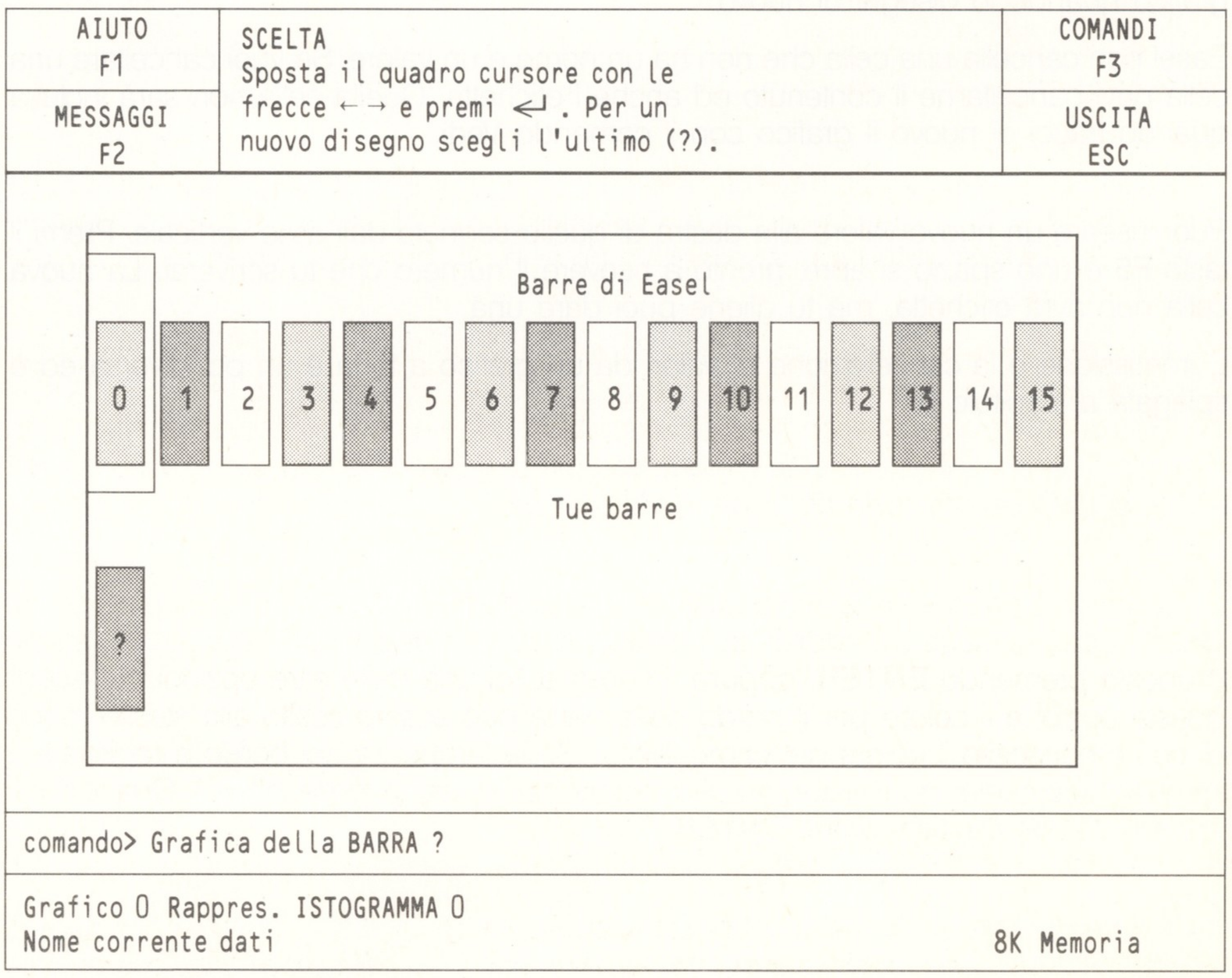


Figura 3.1 Selezione di una barra

La barra è scelta racchiusa in un rettangolo. Per spostare questo rettangolo da una barra all'altra usa i tasti destra e sinistra del cursore fino a quando il rettangolo è posizionato sulla barra che vuoi usare. Quando premi **ENTER**, la barra che hai scelto viene visualizzata.

DISEGNO DELLA BARRA

Quando usi gli esempi per operare delle scelte noterai che vi una barra presente, alla seconda riga, che al posto del numero di riferimento ha un punto interrogativo. Se vuoi creare un nuovo disegno scegli questa barra.

Posiziona il rettangolo della selezione su questa barra e poi premi **ENTER**. Il disegno con l'uso degli esempi continua presentandoti con un disegno di barra vuoto e una lista di opzioni.

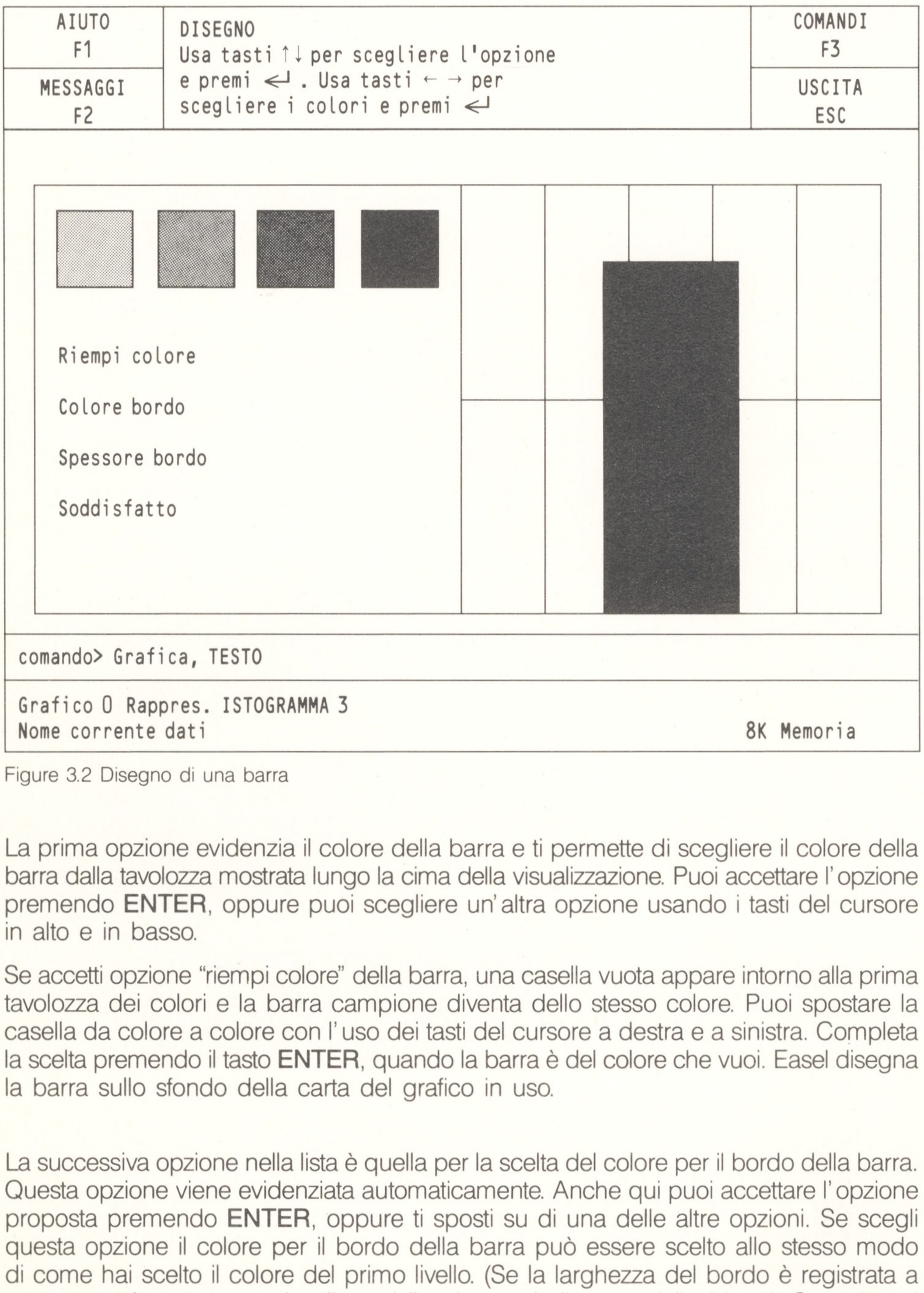


Figure 3.2 Disegno di una barra

Colore della barra

La prima opzione evidenzia il colore della barra e ti permette di scegliere il colore della barra dalla tavolozza mostrata lungo la cima della visualizzazione. Puoi accettare l'opzione premendo **ENTER**, oppure puoi scegliere un'altra opzione usando i tasti del cursore in alto e in basso.

Se accetti opzione "riempi colore" della barra, una casella vuota appare intorno alla prima tavolozza dei colori e la barra campione diventa dello stesso colore. Puoi spostare la casella da colore a colore con l'uso dei tasti del cursore a destra e a sinistra. Completa la scelta premendo il tasto **ENTER**, quando la barra è del colore che vuoi. Easel disegna la barra sullo sfondo della carta del grafico in uso.

Colore del bordo

La successiva opzione nella lista è quella per la scelta del colore per il bordo della barra. Questa opzione viene evidenziata automaticamente. Anche qui puoi accettare l'opzione proposta premendo **ENTER**, oppure ti sposti su di una delle altre opzioni. Se scegli questa opzione il colore per il bordo della barra può essere scelto allo stesso modo di come hai scelto il colore del primo livello. (Se la larghezza del bordo è registrata a zero, naturalmente, non visualizzerai il colore nel disegno della barra.) Quando sei soddisfatto del risultato, premi **ENTER**.

Spessore del bordo

La terza opzione è quella per selezionare lo spessore del bordo. In questo caso ti viene chiesto di scrivere un numero per rappresentare lo spessore del bordo come percentuale della metà dello spessore della barra.

Infine ti viene data l'opzione di decidere se sei soddisfatto del disegno. Se ne sei soddisfatto devi premere **ENTER**, e il nuovo disegno viene aggiunto alla lista dei tipi dei vari disegni delle barre e viene automaticamente usato per visualizzare la serie di valori in uso. Se non sei soddisfatto del disegno puoi ritornare ad una delle altre opzioni, con l'uso dei tasti in alto e in basso del cursore, e provare una nuova combinazione di colori. In qualsiasi momento, prima di terminare il disegno, puoi mettere termine al comando premendo il tasto **ESC**. Così lasci il comando senza aver formato alcun altro disegno della barra.

CAPITOLO 4
USO DEL TESTO

Ogni volta che modifichi del testo, o aggiungi altro testo, questo viene visualizzato nel colore e direzione orizzontale o verticale dell'ultima registrazione, con l'uso dell'opzione Testo, dal comando **Cambia**.

Easel riconosce tre tipi di testo:

Testo ordinario (compreso il Titolo)
Nomi degli Assi
Nomi delle celle (Etichette)

TESTO ORDINARIO

Testo ordinario – cioè tutto il testo ad eccezione dei nomi degli assi – si comporta come se fosse stato incollato sullo schermo. Viene sempre stampato sul grafico o sul quadro statistiche e rimane sullo schermo fino a quando non lo cancelli, indipendentemente da altre modifiche che apporti.

Il comando **Cambia** ha tre opzioni per modificare i tre tipi di testo succitati, e una quarta opzione che si riferisce alla Legenda. L'opzione della Legenda ha importanza solo quando hai più di una serie di valori nel grafico. Essa viene descritta nel prossimo capitolo.

Premi il tasto **T** per selezionare l'opzione di Testo. Poi con l'uso dei tasti del cursore sposta il punto di intersezione degli assi mobili vicino al testo che vuoi modificare. E' importante che l'intersezione degli assi venga posizonata con precisione; premi **ENTER** e i fili si agganciano al testo più vicino. Una copia del testo appare anche alla riga della immissione. Puoi cancellare il testo premendo il tasto **F4**, o lo puoi modificare con l'uso dell'editore. Se scegli di cancellare il testo questo termina anche il comando.

Quando sei completamente soddisfatto con i vocaboli che hai usato per testo allora devi premere **ENTER**. Easel allora ti dà l'opportunità di riposizionare il testo. Premi **ENTER** quando sei soddisfatto della posizione.

Easel tratta il titolo di un grafico come testo. La sola differenza è nel fatto che Easel fornisce il testo del "Titolo", centrato sopra il grafico, quando lo richiami dalla cartuccia del Microdrive.

NOMI DEGLI ASSI

I nomi degli assi appaiono solo su grafici a barre o a righe e non vengono visualizzati nella rappresentazione grafica a torta.

Scegli l'opzione del comando **Cambia** per modificare un nome o l'altro degli assi. Premi **V** per modificare gli assi verticali oppure **O** per modificare gli assi orizzontali. Allora puoi modificare, cancellare o spostare il testo come descritto all'opzione Testo. Easel disegna di nuovo il testo nel colore corrente d'inchiostro o di carta.

ETICHETTE DELLE CELLE (NOMI)

Le celle del grafico hanno etichette per nome che inizialmente sono state impostate per definire i mesi dell'anno da Gennaio a Dicembre. Queste etichette sono visualizzate lungo l'asse orizzontale di un grafico a barre o uno a linee. Nel caso del grafico a torta, le etichette sono usate per nominare i segmenti del grafico.

Per cambiare l'etichetta delle celle, usa l'opzione "Etichetta" del comando **Cambia**. Quando effettui quest'operazione, i fili d'incrocio si attaccano al nome più vicino che viene quindi visualizzato per intero. Etichette di celle possono avere fino a dieci caratteri ma generalmente solo i primi caratteri vengono visualizzati. Il testo viene anche copiato nella riga di immissione. Allora puoi cancellare il nome dell'etichetta premendo il tasto **F4**, oppure modificarlo con l'editore di riga. Premi **ENTER** per il completamento della modifica. Sebbene le etichette abbiano inizialmente il proprio colore di testo, quando modifichi le etichette, vengono visualizzate nel colore del testo corrente. Una etichetta di una cella non può essere spostata.

COLORE E DIREZIONE DEL TESTO

Per cambiare il colore del testo e lo sfondo usa l'opzione **Testo** del comando **Grafica**. Puoi anche scegliere di visualizzare il testo verticalmente oppure orizzontalmente.

Easel usa il nuovo colore e la nuova direzione di testo per tutto il testo che aggiungi successivamente al grafico – ed anche per il testo già in esistenza a cui apporti delle modifiche. Un modo conveniente per cambiare il colore del testo è di cambiarne il colore e poi usare il comando **Grafica** – descritto nella seguente sezione – sul testo esistente senza effettivamente cambiarne i vocaboli o la posizione.

Seleziona l'opzione **Testo** dal comando **Grafica**. Easel ti offre una lista di opzioni per il disegno del testo. Puoi passare da un'opzione all'altra con i tasti del cursore in basso e in alto, e scegliere l'opzione evidenziata premendo **ENTER**.

Colore dell'inchiostro

La prima opzione serve per scegliere il colore dell'inchiostro. Con l'uso del tasto a freccia a destra e a sinistra puoi scegliere il colore seguito da **ENTER**, quando il testo è del colore che vuoi. La successiva opzione viene evidenziata automaticamente, pronta per la selezione con l'uso del tasto **ENTER**.

Colore della carta

Questa seconda opzione serve per scegliere il colore di sfondo della carta. Per scegliere il colore della carta devi usare i tasti a frecce del cursore. Premi **ENTER** per confermarne la scelta e spostarti sulla succesiva opzione.

La terza opzione serve per scegliere uno sfondo trasparente per il testo. Se scegli questa opzione, Easel ignora la scelta del colore della carta e permette che lo sfondo del grafico sia visto per tutto il testo. Ogni volta che scegli quest'opzione, lo sfondo cambia tra lo sfondo trasparente e il colore scelto.

Direzione del testo

La quarta opzione serve per determinare la direzione in cui viene stampato il testo. Ad ogni scelta di quest'opzione Easel cambia il testo tra visualizzazione verticale e quella orizzontale.

Infine ti viene data l'opzione di decidere se sei soddisfatto della presentazione del testo. A questo punto puoi premere **ENTER** per lasciare la selezione così come l'hai fatta e ritornare al menù dei comandi. Alternativamente con l'uso dei tasti a frecce in alto e in basso, puoi effettuare altre modifiche.

CAPITOLO 5
SERIE DI VALORI DIVERSI

Fino ad ora abbiamo descritto solo come formare e rappresentare una singola serie di valori. Molte altre volte vorrai rappresentare due o più serie di valori sullo stesso grafico, per esempio per paragonare i valori di vendita di due anni successivi. Questo capitolo descrive la tecnica da adottare per la formazione, la modifica e la visualizzazione di grafici che hanno serie di di valori diversi.

I VALORI CORRENTI

Indipendentemente dalle serie di valori in un grafico, puoi modificare solo una serie alla volta. La serie che vuoi modificare o a cui vuoi aggiungere è conosciuta col nome di *dati correnti*, e il nome appare alla riga del grafico. Inizialmente hai una serie di valori chiamata "dati".

IL COMANDO NOMINA

Supponiamo che tu abbia scritto una serie di "dati" e vuoi cambiarne il nome a "vendite". Lo puoi fare con l'uso del comando **Nomina**. Premi **F3** seguito da **N**. Easel ti suggerisce di cambiare la serie corrente. Puoi accettare questo suggerimento premendo **ENTER**, oppure scrivere un altro nome seguito da **ENTER**. Per cambiare il nome corrente dei "dati" al nuovo nome "vendite", devi premere:

F3 N ENTER vendite ENTER

La serie dei valori che hai nominato adesso diventata la serie corrente.

DATI NUOVI

Vi sono due metodi che puoi usare per riportare nuove serie successive di valori. Questi sono l'uso del comando **Datinuovi**, oppure l'uso di una formula. I due metodi sono descritti in questa sezione e in quella successiva.

Supponiamo che tu voglia formare una serie di valori nominata "vendita" come sopra descritto, contenente i valori delle vendite mensili, e che tu adesso vuoi includere la rappresentazione dei costi mensili. Lo puoi fare premendo il tasto **F3** e poi il tasto **D**, per selezionare il comando **Datinuovi**. Quindi scrivi un nome per la nuova serie di valori, per completare premi il tasto **ENTER**.

Per formare una nuova serie di valori chiamata "costi" devi quindi premere:

F3 D costi ENTER

Easel ti offre immediatamente un nuovo grafico libero (assumendo che stai usando grafico a barre o a righe) con il filo verticale impostato nella prima colonna. La riga informativa del grafico ti mostra che i valori correnti sono adesso "costi" (nuova serie). Non ti resta che scrivere i nuovi valori, che verranno rappresentati immediatamente sul grafico in modo normale.

Se vuoi formare un'altra serie di valori, puoi usare il comando **Datinuovi** un'altra volta, esattamente nel modo descritto, dando un nuovo nome alla nuova serie di valori. Puoi formare quante serie vuoi. Il solo limite è la capacità di memoria disponibile.

USO DI UNA FORMULA

Qualche volta vorresti riportare una nuova serie di valori che fanno riferimento in un modo o nell'altro ad uno o più serie già esistenti.

Supponiamo che già immesso i valori per "vendite" e "costi" e voglia produrre un grafico che rappresenti il risultato del profitto. Non hai che da scrivere la formula che descrive la nuova serie di valori, per esempio:

```
profitto = vendite - costi
```

Questa formula crea una nuova serie di valori, nominata "profitto", il cui valore è quindi la differenza tra i corrispondenti valori delle "vendite" e dei "costi". "Profitto" diventa il valore corrente e il grafico viene visualizzato immediatamente.

Puoi anche usare una formula che non faccia riferimento alla serie di valori in esistenza. Potresti, per esempio, scrivere una formula come

```
sinusoide = 10 * seno(cella/2)
```

Questa formula definisce e mostra una nuova serie di valori che chiama "sin", il cui valore viene calcolato con l'uso della funzione sen(). In questa formula abbiamo usato anche "cella". Questo dà il numero della cella, iniziando da 1 a sinistra del grafico. Per vedere come funziona, scrivi la formula:

```
a = cella
```

e osserva il grafico che viene tracciato sullo schermo. Quando usi "cella" in una formula, il numero dei valori in una serie di dati viene riportato uguale al numero di colonne correntemente visualizzate nel grafico.

C' un altro termine riservato a Easel – "cellmax". Esso ha un valore uguale al numero di celle correntemente visualizzate sullo schermo. Puoi usare "cellmax" per aggiustare la scala dell'asse orizzontale in una formula. Per esempio, la formula:

```
sin = sen(2*pi()*(cella - 1)/(cellmax - 1))
```

traccia un ciclo completo di un'onda sinusoidale, indipendentemente dal numero di celle sullo schermo.

IL COMANDO UTILIZZA

Quando usi il comando **Datinuovi** la serie dei dati che formi diventa la serie corrente. Ricordati che è la serie a cui puoi aggiungere o modificare dei numeri. Puoi modificare una serie di valori già in memoria, che non è la serie corrente, con il comando **Utilizza**. Quando selezioni questo comando, ti viene chiesto di scrivere il nome di una serie di valori attualmente nella memoria, e quella serie diventa la serie corrente.

Supponiamo che tu abbia le tre serie di valori chiamati "Vendite", "Costi" e "Profitto", e che profitto sia il nome della serie corrente. Se vuoi modificare o apportare delle aggiunte devi selezionare il comando **Utilizza**. I valori dei costi appaiono sul grafico e quindi ne puoi modificare i dati.

Nota che le modifiche che apporti ai valori dei "costi" non influiscono automaticamente sul grafico del "profitto". (Questo è una gestione di Abacus).

VISUALIZZAZIONE DEI DATI

Puoi visualizzare tutti i valori in un singolo grafico con l'uso del comando **Vedi**.

Prova a selezionare questo comando. Come vedi, Easel suggerisce che tutti i valori vengano visualizzati. Puoi accettare questo suggerimento premendo il tasto **ENTER**. Easel poi ti suggerisce il grafico con cui visualizzare i valori. Anche qui accetti il suggerimento premendo **ENTER**. Il grafico viene tracciato immediatamente, mostrando tutti i dati che hai definito – insieme alla *casella della legenda*, in cui si vedono i nomi dei valori e il modo in cui sono rappresentati per la visualizzazione (la legenda non viene visualizzata se hai solo una serie di valori sul grafico).

Se hai definito molti valori, il grafico sarebbe troppo pieno e non avrebbe molto senso. Generalmente è una buona idea visualizzare solo alcuni valori sul grafico per ottenere visualizzazione ad impatto ottimale. Questo non significa che devi definire solo poche serie di valori, dal momento che il comando Vedi ti permette di scegliere quale serie di valori vuoi vedere.

Puoi scegliere i valori che vuoi vedere, non accettando "tutti i valori" suggerito da Easel, con il comando **Vedi**. Invece di premere **ENTER**, a questo punto, scrivi un elenco delle serie di valori che vuoi che vengano visualizzate, separate da virgole. Quando hai scritto tutti i nomi delle serie di valori che vuoi che siano visualizzate, premi il tasto **ENTER**.

Puoi anche scegliere un grafico diverso per la visualizzazione invece di accettare quello suggerito da Easel. Anzich, semplicemente battere **ENTER** a questo punto, puoi scrivere un numero da 0 a 16. Easel dispone di 8 possibilità che offrono tipi di grafici diversi, a barre, (Istogrammi) a righe (cartesiana) o a torta. Per vedere il menù di tutti i grafici disponibili, basta scrivere un punto interrogativo. Prova ad usare il comando **Vedi** per visualizzare tre o quattro serie di valori in vari grafici tra quelli disponibili predefiniti.

LA LEGENDA

Una delle opzioni del comando **Cambia**, quella per lo spostamento della legenda. La legenda viene sostituita con la casella, e poi ti viene offerta l'opzione di cancellare la legenda – premendo il tasto **F4** – oppure di spostare la legenda con l'uso dei tasti del cursore con frecce. Se scegli l'opzione spostare, i tasti del cursore spostano una casella di dimensione uguale alla legenda per tutta la zona della visualizzazione. Quando premi **ENTER** il grafico viene tracciato di nuovo con la legenda nella nuova posizione.

del cursore con frecce. Se scegli l'opzione spostare, i tasti del cursore spostano una casella di dimensione uguale alla legenda per tutta la zona della visualizzazione. Quando premi **ENTER** il grafico viene tracciato di nuovo con la legenda nella nuova posizione.

Vorrai ad un certo punto riposizionare la visualizzazione della legenda che hai cancellato precedentemente. Lo puoi fare con l'uso del comando **Cambia** e la scelta dell'opzione Legenda. La casella della legenda apparirà. Puoi spostare la legenda ad un'altra posizione. Il grafico viene tracciato di nuovo insieme alla legenda se premi **ENTER**.

La sola modifica che puoi fare alla legenda è di cambiarne il colore del testo. Questo testo è impostato l'ultima volta che si usato il comando **Cambia**. I simboli che appaiono nella casella della legenda saranno sempre corrispondenti, naturalmente, a quelli usati nella visualizzazione del grafico.

CAPITOLO 6
FORMATO DEL GRAFICO

PER CAMBIARE FORMATO

Easel è fornito di 8 *tipi di grafici* (numerati da 0 a 7). Con l'uso di uno di questi numeri, puoi specificare quale tipo di grafico vuoi che venga usato ogni volta che premi il comando **Vedi**. Oltre ad usare tipi diversi dello sfondo e del colore della barra, questi grafici ti offrono una gamma di tipi per la rappresentazione.

Puoi anche usare l'opzione "Grafico" del comando **Grafica** per scegliere uno degli 8 tipi. Easel imposta il testo:

```
COMMANDO> Grafica, grafico ?
```

alla riga della immissione e tu puoi scegliere un determinato tipo, se ne scrivi il relativo numero da 0 a 7 (seguito da **ENTER**). Se premi **ENTER**, allora Easel ti mostra gli esempi degli 8 tipi di grafici e ti chiede di nuovo di scrivere il numero relativo al grafico.

PER DISEGNARE UN NUOVO GRAFICO

Puoi disegnare la completa apparenza di qualsiasi degli otto grafici diversi che ti offre Easel.

Normalmente avrai un'idea di come vuoi che il grafico appaia. In questo caso scegli il grafico del tipo più simile a quello che vuoi e poi lo modifichi fino a quando appare secondo le tue esigenze.

Usa le opzioni del comando **Cambia** per cambiare il testo delle etichette delle celle e i nomi degli assi. Per modificare il disegno del testo e della rappresentazione della barra usa il comando **Grafica**.

Se vuoi un grafico a righe (Cartesian) puoi usare il grafico numero 3, oppure l'opzione **Riga** del comando **Grafica**. I grafici a torte sono descritti al capitolo 8.

Carta__del__grafico

L'opzione "Carta__del__grafico" del comando **Grafica** ti permette la scelta del colore della carta del grafico e il colore del reticolo. Puoi scegliere la carta del grafico da una serie di sette tipi già esistenti, oppure ne puoi disegnare uno di tua scelta. Il metodo per disegnare è esattamente come quello descritto per l'opzione Barra del comando **Grafica**.

Asse

L'opzione Asse del comando **Grafica** ti permette la scelta del tipo di asse per il grafico. Puoi scegliere un asse da una serie già esistente di dieci tipi oppure ne puoi disegnare uno di tua scelta. Il metodo per il disegno di un tipo proprio esattamente come descritto per le opzioni di Barra e Carta del grafico del comando **Grafica**.

L'opzione per il disegno dell'asse ti permette la scelta del colore della riga dell'asse, indipendentemente dal fatto che la riga sia disegnata o meno, del colore dei numeri che etichettano l'asse verticale e dei limiti dell'asse.

Easel normalmente sceglie i limiti per la serie di valori mostrati sull'asse verticale. Sceglie una serie che ti permette di visualizzare tutti i valori del grafico. Se scegli l'opzione di cambiare i limiti dell'asse, ti viene offerta una delle tre possibilità.

Premi il tasto **A** per scegliere i limiti Automatici. Easel seleziona una serie adatta, secondo i valori del grafico. La serie potrebbe non comprendere il punto zero se, per esempio, tutti i valori sono grandi e positivi.

Premi il tasto **Z** per selezionare limiti automatici che comprendono lo Zero. Questo il tipo dei limiti dell'asse che usa Easel se non fai la tua scelta.

Premi il tasto **M** se vuoi fare la tua propria scelta dei limiti (Manuali). Easel ti chiede di scrivere il limite inferiore e poi i limiti superiori (marca la fine di ogni valore premendo **ENTER**). Nota che Easel esclude la tua scelta se i limiti non coprono la serie completa dei valori nel grafico.

In tutti i casi, i due limiti vengono arrotondati in modo che gli intervalli nello scalamento dell'asse siano sensati. Nota che l'esempio dell'asse, visualizzato alla destra dello schermo, non mostra necessariamente la giusta serie che verrà usata nel grafico. Come esempio è solo rappresentativo ed è usato solo per illustrare il tipo generale di asse che hai scelto.

CAPITOLO 7
GRAFICI A LINEE (RIGHE) (CARTESIANA)

Come avrai visto quando sviluppavi i diversi tipi di visualizzazione, le serie di valori possono essere rappresentate anche in grafico di formato a linea, o a torta. Questo ti permette di rappresentare una data serie di valori in molti modi diversi, in modo che puoi scegliere il metodo più adatto alle tue esigenze.

Il grafico numero 3, usa linee per rappresentare la serie di valori. Ogni valore può essere marcato con un simbolo e i valori sono congiunti da righe di vari colori e spessori. Puoi anche usare linee "piene" dove lo spazio tra la linea e il livello zero viene completamente riempito di colore.

Potrai trovare le linee piene utili per rappresentare "valori critici", come per esempio, un livello del punto di pareggio.

Dal momento che barre e linee vengono visualizzate sullo stesso tipo di reticolo, puoi mettere insieme linee e barre in qualsiasi combinazione. Titoli, nomi di assi e casella della legenda si comportano tutte allo stesso modo, sia per la rappresentazione a barre che per quella a linee.

PER SELEZIONARE UN TIPO DI LINEA

Se selezioni l'opzione Riga del comando **Grafica**, puoi cambiare la rappresentazione di una serie dei valori che devono essere usati nel grafico. Prima cambia il grafico che vuoi modificare a valori correnti (es. con Utilizza). Poi seleziona l'opzione Riga del comando **Grafica**.

Vi sono 16 tipi di linee predefinite e Easel per prima cosa ti chiede di dare il numero relativo alla linea che vuoi, premi **ENTER**, oppure premi solo **ENTER** per vedere la selezione disponibile. Scegli una linea premendo i tasti del cursore a destra o a sinistra. Quando la casella racchiude la linea che vuoi, premi **ENTER**. Easel immediatamente traccia la serie di valori col tipo di linea che hai scelto (questo è un grafico CARTESIANO).

Scegli la linea visualizzata con il punto interrogativo invece del numero se vuoi disegnare la linea per conto tuo.

Easel ti offre una lista di opzioni per la linea e tu le usi esattamente allo stesso modo descritto per l'opzione della Barra, al capitolo 3. Premi **ENTER** per selezionare l'opzione evidenziata o usa i tasti del cursore per passare all'opzione che vuoi.

Colore della linea

Scegli il colore della linea. Con l'uso dei tasti del cursore e il tasto **ENTER** per passare alla successiva opzione.

Simboli

Ti permette di scegliere se marcare ogni punto della linea con simboli. Ogni volta che scegli questa opzione i simboli vengono attivati e disattivati.

Colore del simbolo

Scegli il colore del simbolo allo stesso modo di come scegli il colore della linea. Puoi trascurare questa opzione se hai deciso di non usare alcun simbolo.

Linee piene

Cambia da linee normali a linee che sono piene con il colore usato per la linea all'asse orizzontale. Ogni volta che scegli questa opzione Easel passa da una all'altra.

Spessore della linea

Ti permette la scelta dello spessore della linea. Scrivi un numero tra 0 (il più sottile) e 100 (il più spesso) e premi **ENTER**. Puoi trascurare questa opzione se hai scelto una linea piena dall'asse. Easel ti offre un'altra opzione per controllare che sei soddisfatto del risultato. Premi **ENTER** per visualizzare il grafico con il tuo tipo di linea, oppure usa i tasti del cursore con frecce in basso e in alto per modificare le scelte ancora una volta.

Figura 7.1 stata formata col grafico numero 2 (barre ammassate) con una serie di valori cambiati per rappresentazione a linea a quella a barra.

IMMISSIONE NUMERICA PER LINEE

Se scegli un formato che usa grafico a linee per la rappresentazione dei valori correnti, puoi immettere i dati esattamente come descritto per i grafici a barre – è sufficiente scriverli.

La sola vera differenza tra rappresentazioni a linee e rappresentazioni a barre appare quando stai scrivendo dei numeri in una serie di valori rappresentati con una linea. Per da permetterti di scrivere nuovi numeri – oppure di modificare quelli già esistenti – senza dover disegnare tutto il grafico per ogni numero, Easel non usa il vero colore della linea.

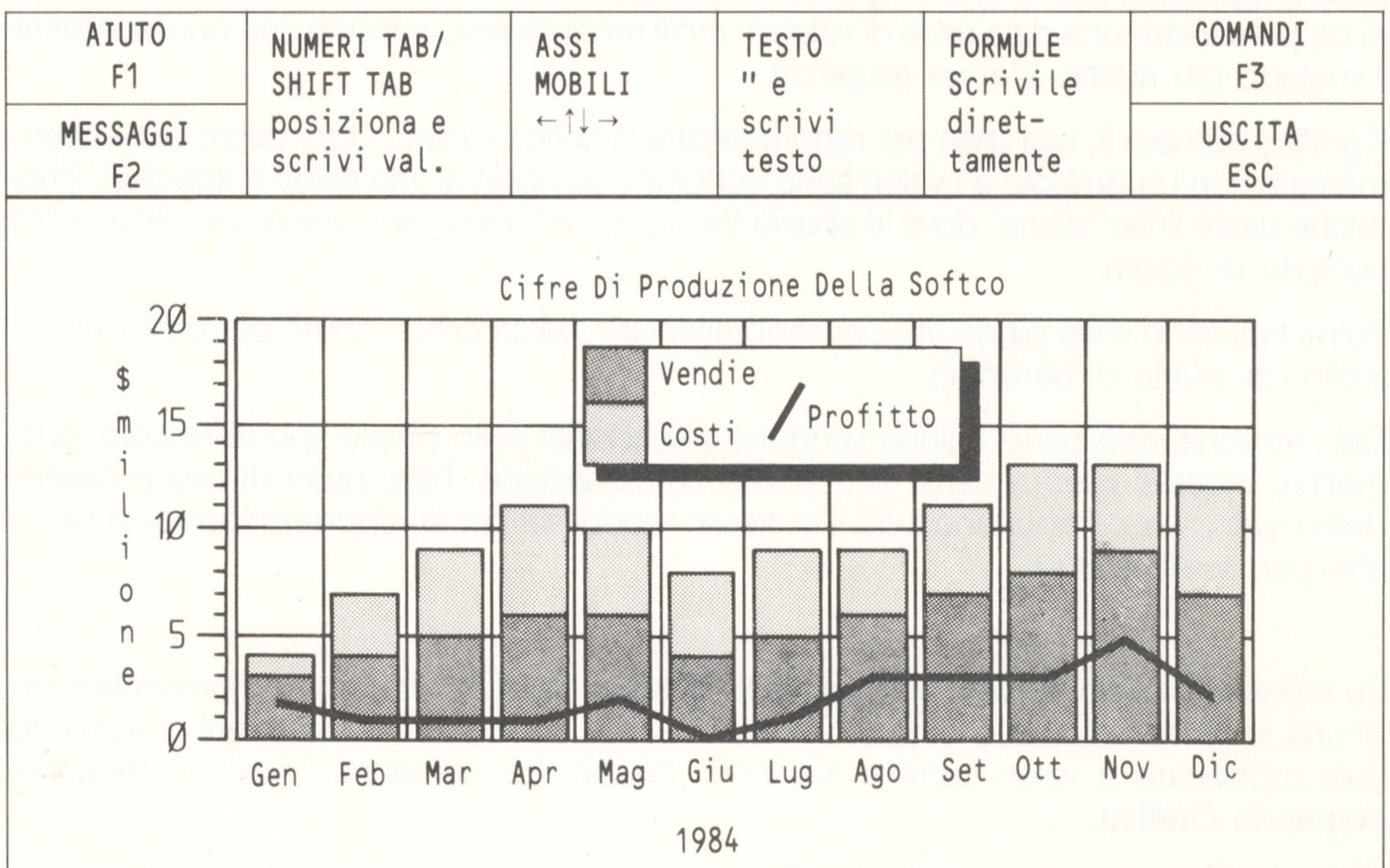


Figura 7.1 Barre ammassate e linee

Mentre stai scrivendo i numeri, il grafico viene tracciatao con l'uso di una sottile linea bianca – secondo il tipo di linea che hai scelto. Il colore della linea cambia quando passa su di una barra, su di una linea o su del testo. Easel ti avverte, alla riga di stato che il colore della linea visualizzata non è quello vero. Quando hai finito di scrivere i valori, usa il comando **Vedi** per visualizzare il grafico con il giusto colore e spessore della linea.

CAPITOLO 8
GRAFICI A TORTA

Sebbene un grafico a torta sia molto diverso in apparenza da un Istogramma o un grafico Cartesiano, Easel ti permette di formarne uno esattamente allo stesso modo. Il formato numero 7 rappresenterà una serie di valori con grafico a torta. Nota che puoi visualizzare solo una serie di valori alla volta in un formato di grafico a torta, e che i valori negativi vengono ignorati. Easel t'informa se dei dati sono stati omessi.

Dal momento che puoi rappresentare solo una serie di valori in una torta, il comando **Vedi** ti offre l'opzione di vedere i valori correnti – invece della normale "vedi tutti i valori" –. Puoi scrivere la sostituzione di un nome. Se scrivi una lista di nomi (separati da virgole), Easel rappresenta la prima serie di valori, ignorando il resto.

IMMISSIONE DI NUMERI

Per illustrare l'immissione in un grafico a torta, usa il comando **Grafica**, e con l'opzione GRAFICO cambia la rappresentazione al formato 7, che è il formato del grafico a torta. Poi usa il comando **Datinuovi** per formare una nuova serie di valori vuota chiamata, per esempio "costi". Easel disegna un cerchio pieno, etichettato con il primo nome della cella che, a meno di non averlo cambiato, è "Gen".

Scrivi un numero e premi **ENTER**. Easel traccia di nuovo il cerchio, ma questa volta il numero che hai scritto viene visualizzato sotto l'etichetta. Il diagramma è un grafico a torta con un solo valore. Scrivi gli altri numeri, esattamente allo stesso modo come scrivevi i numeri per l' istogramma.

Durante l'immissione dei valori in un grafico a torta, la cella successiva che riceverà i dati viene indicata evidenziando il suo nome. Se questo non è possibile, verrà indicato con una casella speciale, evidenziata all'angolo sinistro inferiore della finestra di visualizzazione. Alla Figura 8.1 questa casella ha il nome "PORTOGALLO".

AIUTO F1	NUMERI TAB/ SHIFT TAB posiziona e scrivi val.	ASSI MOBILI ←↑↓→	TESTO " e scrivi testo	FORMULE Scrivile diret- tamente	COMANDI F3
MESSAGGI F2					USCITA ESC

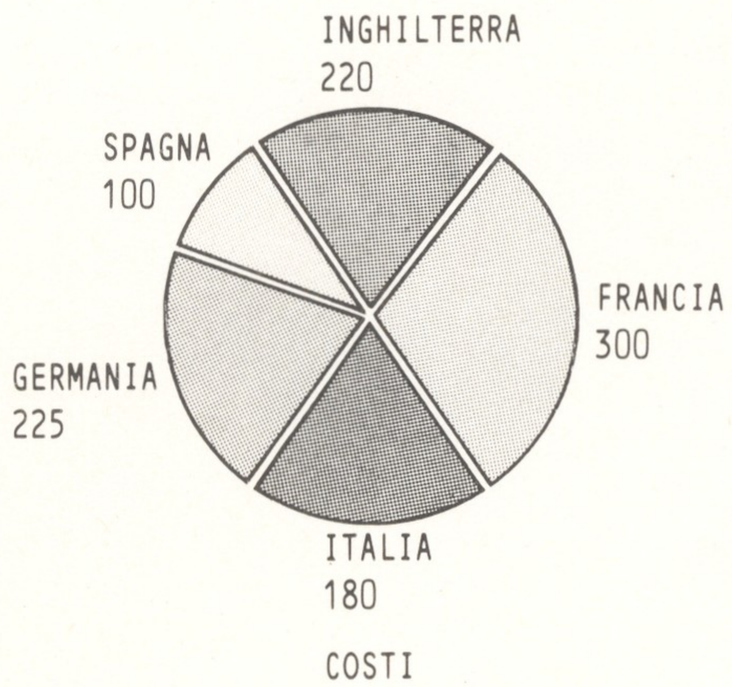


Figura 8.1 Un grafico a torta

Dal momento che la torta viene tracciata ogni volta che immetti o modifichi dei dati, potresti trovare più conveniente immettere i valori in uno degli altri formati grafici e poi cambiare la rappresentazione grafica a torta.

NOMI DELLE CELLE

Per spostarti da una cella all'altra usi il tasto **TABULATE** oppure con i tasti **SHIFT** + **TABULATE**, proprio come in un grafico a barre. Ricordati che, come sempre, questa opzione non è disponibile dal menù dei comandi.

Premi **F5** per aggiungere un'altra cella dopo quella il cui nome è evidenziato. Easel dà ad ogni nuova cella l'etichetta "senza nome". Scrivi un numero in quella cella nel modo normale. Puoi modificare il nome della cella con l'opzione Etichetta del comando **Cambia**. La cella che puoi modificare, viene evidenziata. Puoi passare da cella a cella con i tasti **TABULATE** e **SHIFT**.

Come per i grafici a barre e quelli a linee, devi cancellare il nome della cella (usa l'opzione Etichetta del comando **Cambia**, e premi **F4**) e il numero nella cella (sposta alla cella con il tasto **TABULATE** e premi **F4**) prima che Easel possa cancellare tutta la cella. Easel cancella una cella vuota la prossima volta che usi il comando **Vedi**.

TESTO

Puoi aggiungere, modificare o spostare testo e titoli esattamente come descritto per i grafici a barre e a righe al Capitolo 2 e 4. Usa l'incrocio dei fili (orizzontale e verticale) nel modo normale per aggiungere, modificare o spostare sil testo ordinario.

IL COMANDO GRAFICA

Le opzioni Testo e Grafico del comando **Grafica** operano esattamente allo stesso modo con un formato a torta come per qualsiasi altro formato.

Carta__del__grafico, Barra, Righe e assi dei grafici non hanno alcun significato per un grafico a torta e Easel non ti permette di usare queste opzioni.

L'opzione Settore può essere usata solo in un formato grafico a torta. Ti permette di cambiare il colore di un segmento del grafico. Prima scegli l'opzione Settore del comando **Grafica**. Poi seleziona il segmento di cui vuoi cambiarne il colore (premi **TABULATE** fino a quando il relativo nome viene evidenziato.

Easel mostra nella finestra di visualizzazione la tavolozza dei colori che sono disponibili. Premi il tasto del cursore a destra o a sinistra per scegliere il colore che vuoi e poi premi **ENTER**. Easel traccia di nuovo la torta con il segmento del colore che hai scelto.

CAPITOLO 9

COPIE PERMANENTI DEI GRAFICI

STAMPA

Se hai una stampante ad aghi che è compatibile con la Epson JX80 (per esempio la Brother HR-15 o la Canon PW1080A) puoi fare delle copie stampate dei grafici immediatamente.

Il comando **Stampa** produce una copia stampata del grafico che è sullo schermo. Premi il tasto **S** per selezionare il comando **Stampa**. Prima di stampare, Easel legge il file di comando della stampante dal file "gprint__prt" dalla cartuccia del Microdrive 1. Vedi Appendice.

Premi il tasto **M** per memorizzare il grafico in un file del Microdrive; devi scrivere il nome del file da usare seguito da **ENTER**. Questo file può in seguito essere usato in per esempio, da un programma SuperBASIC e inviato ad una stampante non supportata da Easel. Nota che questo file è molto grande e normalmente non più di tre file possono essere memorizzate su una cartuccia del Microdrive.

Premi Il tasto **I** per installare un nuovo file di comando della stampante. Diversi altri file comando stampante vengono forniti sulla cartuccia di Easel. Essi sono file con l'estensione **__prt**. Alcuni di queste sono stampanti a colore, per esempio la Intgx132 e la Okimate 80. Scrivi il nome del file di comando stampante che vuoi e premi **ENTER**.

Il nuovo file comando stampante non viene installato permanentemente e Easel ritorna all' Epson JX80 la prossima volta che viene richiamato. Il comando stampante di default si trova nel file "gprint__prt".Così puoi rendere l'installazione di una nuova stampante permanente semplicemente cambiando il nome dei file. Per prima cosa copia l'originale "gprint__prt" su un altro file, per esempio "JX80__prt" e cancella il file originale. Poi copia il file che contiene il file di comando della stampante che richiedi sul file "gprint__prt". Nota che la cartuccia originale di Easel è protetta dalla scrittura. Quindi devi usare la copia che hai fatto.

Puoi usare un baud diverso da 9600 assunto dal QL. Per esempio, se vuoi impostare 4800 baud inizia Easel scrivendo:

```
BAUD 4800: LRUN mdv1_boot
```

Invece di avere cartuccia Easel nel Microdrive 1 quando premi **F1**o**F2**.

Alternativamente potresti rendere la modifica della velocità permanente con l'aggiunta di un'altra riga al programma "boot". Per prima cosa richiama e dai un nuovo numero al programma scrivendo:

```
LOAD mdv1_boot: RENUM 10,10
```

Poi aggiungi, per esempio, la riga:

```
5 BAUD 4800
```

Cancella la vecchia copia del programma e memorizza la nuova versione sulla cartuccia di Easel, scrivi:

```
SAVE mdv1_boot
```

Anche questa modifica deve essere fatta sulla copia della cartuccia Easel.

FOTOGRAFIA

Il modo più semplice e più rapido per ottenere copie permanenti di uno dei grafici è quello di fotografare lo schermo. Devi, comunque, avere un po' di cura se vuoi ottenere buoni risultati.

Una delle cause più comune di una cattiva foto dello schermo della televisione è l'uso di breve tempo d'esposizione. La visione composta da 625 linee separate, mostrate una dopo l'altra. Ci vuole un 25mo. di secondo per mostrare tutte le linee nella visione e se usi un tempo d'esposizione di circa questa lunghezza, o più corto, la foto sarà di luce disuguale. Sarebbe meglio usare un tempo di esposizione di circa un quarto di secondo – questo significa che devi sistemare la macchina fotografica su un cavalletto.

Una pellicola a colori normale (per foto o per diapositive) con un velocità di, diciamo, ASA 100 necessita un apertura di circa F5,6. Usa una lente a lunga distanza focale (circa 100 mm) se ne hai una, perchè questa riduce la distorsione causata dalla superficie curva dello schermo della TV.

Cerca di fare la foto in una stanza buia, per evitare riflessioni del dintorno della superficie dello schermo. Ti meraviglieresti vedere quanto forti siano queste riflessioni nella foto, anche se non le noti quando guardi nella macchina.

Premi **F2** per rimuovere la finestra di comando in modo da darti un grafico più grande. Puoi anche premere **SHIFT** e, mentre lo tieni premuto, premi **F2** per cancellare il testo nella riga di stato.

Prima di fare la foto assicurati che tutto il testo, nomi di celle, nomi di assi e la legenda appaiono esattamente come le vuoi.

CAPITOLO 10
RIFERIMENTO QL EASEL

TASTI DI FUNZIONE

Oltre all'uso normale di **F1**, **F2** e **F3**, tasti di funzione 4 e 5 vengono usati come segue:

F4 cancella
- testo
- etichette (nomi)
- numeri
- la legenda
- oggetti definiti dall'utente

Nota: oggetti definiti dall'utente son barre, linee, carta del grafico e assi.

F5 inserisci una cella

I COMANDI

I comandi ti portano più a fondo dei livelli di Easel e ti permettono l'uso di operazioni molto più difficili. I seguenti comandi sono disponibili:

GRAFICA

Il comando **Grafica** ti permette di modificare la visualizzazione di qualsiasi caratteristica del grafico.

Ti vengono offerte le seguenti opzioni:

Asse — per selezionare il marcamento degli assi. Puoi alterarne il colore e i numeri che etichettano l'asse-y. Puoi anche scegliere se le linee degli assi devono essere tracciate o no. Easel non ti permette di selezionare questa opzione nel formato numero 7 (grafico a Torta).

L'opzione per cambiare i limiti degli assi ti permette di scegliere tra limiti automatici oppure manuali. Premi il tasto **A** per limiti automatici o il tasto **Z** per limiti automatici che includono sempre lo zero.

Alternativamente, premi il tasto **M** per selezionare limiti manuali. In questo caso devi scrivere i valori per tutte e due i limiti. Easel può modificare la tua scelta dei limiti per assicurarsi che tutto il grafico venga visualizzato, con semplici valori numerici sullo scalamento.

Barra — per selezionare o per definire il tipo di barra da usare per rappresentare la serie di valori corrente. Puoi scegliere una delle 16 barre predefinite col relativo numero, o con l'uso degli esempi. Il disegno con l'opzione degli esempi ti permette di selezionare una barra oppure di disegnarne una per conto tuo. Easel non ti permette di selezionare quest'opzione nel grafico numero 7 (grafico a Torta).

Formato — per definire di nuovo la rappresentazione di tutto il grafico. Puoi scegliere uno degli 8 formati già definiti con il relativo numero, oppure con l'uso degli esempi.

Carta__del__grafico — per selezionare uno dei sette tipi diversi della carta del grafico, oppure per sostuirne uno con un disegno per conto tuo. Puoi selezionare il colore di sfondo e le linee del reticolo. Easel non ti permette di selezionare quest'opzione nel formato numero 7 (grafico a Torta).

Riga — per selezionare uno dei 16 tipi di linee definite, oppure per disegnarne una per conto tuo. Puoi scegliere il colore e lo spessore della linea, e il colore del simbolo usato ad ogni punto della linea, oppure scegliere una linea piena, dove lo spazio tra la linea e il livello zero sul grafico tutto riempito di colore. Easel non ti permette di selezionare quest'opzione nel formato numero 7 (grafico a Torta).

Settore — per selezionare il colore di un determinato segmento della torta Easel ti permette di selezionare quest' opzione nel formato numero 7 (grafico a Torta).

Testo — per selezionare il colore usato per il testo e per lo sfondo. Puoi scegliere uno sfondo trasparente in modo che il grafico sottostante venga visualizzato. Puoi anche scegliere se il testo deve essere visualizzato orizzontalmente o verticalmente.

Testo già esistente ritiene il colore e la direzione originale, ma nuovo testo verrà visualizzato nello stile selezionato, fino a quando non lo cambi di nuovo. (Il testo nella Casella della Legenda viene sempre rappresentato nel colore del testo corrente.)

PRESENTAZIONE

Il comando **Presentazione** ti permette di selezionare la visualizzazione a 40, 64 o 80 caratteri. Premi 4, 6 o 8.

CAMBIA

Il comando **Cambia** ti permette di modificare o di spostare il testo, le etichette e la legenda.

Ti viene chiesto di scegliere fra le seguenti opzioni:

Testo — gli assi mobili si agganciano al testo più vicino e puoi usare l' editore di riga per cambiarne i vocaboli. Premi **ENTER** e ti viene offerta l' opzione di spostare il testo ad un' altra posizione. Premi **ENTER** quando sei soddisfatto della nuova posizione.

Etichette — gli assi mobili si agganciano all' etichetta della cella più vicina e tu ne puoi modificare il testo dell' etichetta come all' opzione Testo. Premi **ENTER** quando hai finito. Etichette di celle non possono essere spostate.

Legenda — ti viene subito offerta l' opzione di spostare la casella della legenda con i tasti del cursore. Premi **ENTER** quando il rettangolo della casella è nella posizione che vuoi tu. La legenda con la casella viene tracciata di nuovo nella nuova posizione.

Asse — ti viene chiesto di premere il tasto **V** o il tasto **O** per selezionare il nome dell' asse verticale o orizzontale. I fili d' incrocio si agganciano al nome scelto e tu ne puoi modificare il testo e poi riposizionarlo.

Dopo che hai effettuato la modifica, tutto il testo visualizzato nel colore e nella direzione impostata con l' ultimo uso dell' opzione Testo del comando **Grafica**. Le sole eccezioni sono i nomi degli assi, che sono a direzione fissa.

FILE

Questo comando ti permette di modificare file di Easel memorizzati precedentemente su una cartuccia del Microdrive, o di trasferire dati ad un altro dei programmi QL Psion.

Ti vengono offerti le seguenti opzioni:

Backup — usato per fare una copia di riserva di un file di Easel. Ti viene chiesto il nome del file da copiare e il nome che vuoi dare alla copia del file. **Ti viene raccomandato di fare copie dei file, per protezione contro danneggi, perdita, o avaria della cartuccia, e contro possibili errori che causano risultato sbagliato o cancellazione dell' applicazione**.

Cancella — cancella un file dalla cartuccia del Microdrive.

Attenzione – questo comando non è reversibile e deve essere usato con estrema cautela.

Esporta — esporta un file. Il file contiene tutte le serie di valori attualmente nella memoria del computer. E' stato memorizzato in una forma adatta da poter essere letta con QL Abacus o QL Archive. Importa ed Esporta vengono spiegati nell' Appendice.

Se non specifichi una estensione del nomefile per un file esportato, Easel fornisce una estensione di **__exp**.

Importa `importa un file e permette ad Easel di leggere file di dati esportati da QL Abacus o QL Archive e li rappresenta in forma grafica.

Se non specifichi una estensione di un file importato, Easel assume una estensione di **__exp**.

Formatta formatta la cartuccia in Microdrive 2 oppure un altro Microdrive nominato. Accetta il suggerimento di Easel di formattare mdv2 oppure scrivi un altra specifica, es. mdv3. Easel ti chiede di confermare la selezione di questa opzione.

Attenzione – tutte le informazioni sulla cartuccia vengono cancellate quando effettui la formattazione.

EVIDENZIA

Questo comando ti permette di usare uno stile speciale per rappresentare un particolare numero in una serie di valori, o tutti i valori negativi in un grafico a barre.

Easel per prima cosa ti chiede di premere il tasto **V** per evidenziare un particolare valore, o di premere **N** per evidenziare tutti i valori Negativi. Non puoi selezionare questa seconda opzione per un grafico a Torta.

Se scegli di evidenziare un valore, Easel ti chiede di selezionare il valore, premi **TABULATE** (**SHIFT** e **TABULATE**) per selezionare la cella che vuoi evidenziare e poi premi **ENTER**. Nel caso di un istogramma ti viene mostrata la selezione di barre definite, e puoi sceglierne una – oppure ne puoi disegnare una per conto tuo. In un diagramma a torta il segmento selezionato viene staccato dal resto della torta.

Se selezioni l'opzione di evidenziare valori negativi, Easel ti chiede immediatamente di selezionare o disegnare una barra.

AIUTO F1	NUMERI TAB/ SHIFT TAB posiziona e scrivi val.	ASSI MOBILI ←↑↓→	TESTO " e scrivi testo	FORMULE Scrivile diret- tamente	COMANDI F3
MESSAGGI F2					USCITA ESC

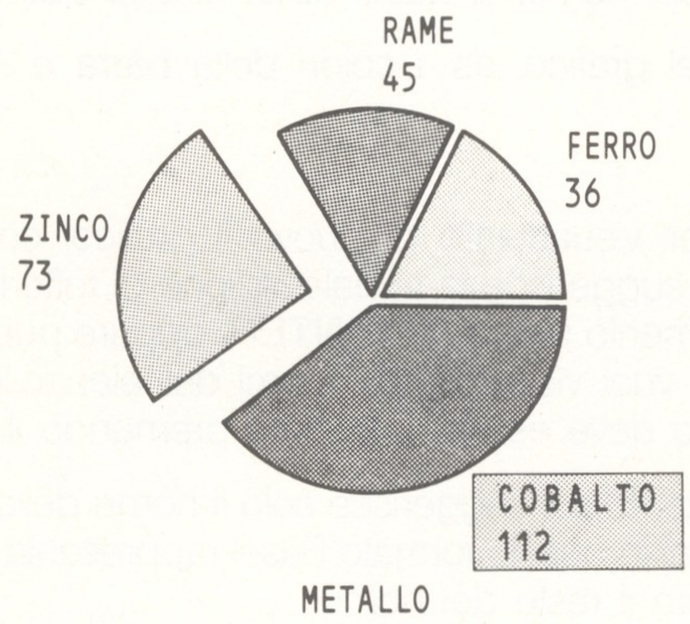


Grafico 7 Rappres. TORTA
Nome corrente Metallo 8K Memoria

Figura 10.1 Un settore della torta evidenziato.

AZZERA

Cancella una o più serie di valori dal grafico e ne distrugge i dati. Quando selezioni questo comando ti viene chiesto di scrivere i nomi dei valori che vuoi cancellare, separati da virgole e completando con **ENTER**. Se premi solo **ENTER**, Easel cancella i valori correnti. Puoi, se vuoi, scrivere il testo **tutti i valori**.

RICHIAMA

Richiama un grafico memorizzato precedentemente nella cartuccia del Microdrive. Easel ti chiede di scrivere il nome del file che deve essere richiamato. Tutte le opzioni di disegno vengono richiamate con i dati quindi il grafico dei dati richiamati ha esattamente lo stesso aspetto che aveva quando è stato memorizzato.

Se non hai specificato una estensione del nomefile, Easel assume una estensione di **__grf**.

DATINUOVI Ti permette di formare un nuova serie di dati, che diventa la "serie corrente". Ti viene chiesto di scrivere il nome della nuova serie di dati (non c'è bisogno di virgolette). Quando premi **ENTER**, ritorni allo stato di immissione dati, pronto per la scrittura di nuovi valori.

UTILIZZA Il comando **Utilizza** ti permette di far diventare una vecchia serie i "dati correnti". Ti viene chiesto di scrivere il nome della vecchia serie (non c' bisogno di virgolette). Quando premi **ENTER** ritorni allo stato di immissione, pronto per la scrittura di modifiche o aggiunte ai valori.

STAMPA Stampa il grafico che attualmente visualizzato sullo schermo.

Il comando offre tre opzioni. Premi il tasto **S** per stampare il grafico, con l'uso del file del comando della stampante dei grafici corrente.

Premi il tasto **M** per memorizzare lo schermo su un file nel Microdrive. Nella opzione di memorizzazione su file, Easel ti chiede di scrivere il nome del file. Premi il tasto **I** se vuoi installare un nuovo file del comando della stampante grafica. Easel attende che tu scrivi il nome di uno dei file di comando della stampante (con un'assunta estensione di **__prt**) fornito sulla cartuccia di Easel. Per ulteriori informazioni vedi il Capitolo 9.

LASCIA Usi questo comando per lasciare Easel e ritornare a SuperBASIC. Ti viene offerta la scelta tra le due opzioni di premere **ENTER** per confermare la tua scelta e ritornare a SuperBASIC, o di premere **ESC** per cancellare il comando e ritornare al menu dei comandi.

NOMINA Questo comando ti permette di cambiare il nome della serie dei valori correnti. Easel ti chiede di scrivere il vecchio nome, suggerendo la serie di valori correnti, e poi il nuovo nome. Premi **ENTER** alla fine di ogni nome.

MEMORIZZA Memorizza tutte le serie di valori rappresentati sullo schermo nella cartuccia del Microdrive. Ti viene chiesto di scrivere un nome sotto cui i valori verranno memorizzati. Se non specifichi la estensione del nomefile, Easel assume una estensione di **__grf**.

Tutte le caratteristiche del grafico, es. i colori della barra e il tipo di assi, vengono memorizzati con i valori.

VEDI Usi questo comando per visualizzare di nuovo il grafico, con tutte, o alcune serie selezionate di dati. Easel suggerisce la visualizzazione di tutte le serie di dati e tu puoi accettare questo suggerimento premendo **ENTER**, oppure puoi scrivere un elenco dei nomi di quelle serie che vuoi visualizzare. I nomi dell'elenco devono essere separati da una virgola e l'elenco deve essere terminato premendo il tasto **ENTER**.

Nel formato grafico a Torta, Easel suggerisce solo il nome della serie corrente dei dati. Se scrivi un elenco di nomi in questo formato Easel rappresenta solo la torta per il primo nome dell'elenco e ignora il resto dei nomi.

Poi ti viene offerto il numero di un formato suggerito per la rappresentazione. Puoi accettare il formato suggerito (l'ultimo che stavi usando) premendo **ENTER**, oppure puoi scrivere il numero del formato di tua scelta, seguito da **ENTER**.

ZERO Questo comando cancella tutto il testo, tutte le serie di dati e tutti gli oggetti definiti dall'utente (barre, linee e così via). Ripristina pure il nome del mese delle celle.

FUNZIONI

Considera una funzione come un tipo di ricetta che converte un numero di valori iniziali, conosciuti come gli argomenti della funzione, in valori diversi, che si dice di essere il valore riportato dalla funzione.

Le funzioni offerte da Easel prendono uno o nessun argomento. L'argomento per una funzione viene messo tra parentesi subito dopo il nome. Non devi lasciare spazio tra il nome e la parentesi di apertura, spazio comunque, permesso fra le parentesi. Tutti i nomi delle funzioni devono essere seguiti da parentesi, anche se non hanno alcun argomento. La presenza delle parentesi è un utile modo per ricordarti che stai facendo riferimento ad una funzione. Ti permette di fare la distinzione tra il nome di una serie di dati e una funzione.

Le seguenti funzioni sono disponibili:

ASS(n) Calcola il valore assoluto , cio il valore numerico senza segno, dell'argomento. Per esempio, ass(5) e ass(−5) riportano il valore di 5.

ATG(n) Calcola l'angolo, in radianti, la cui tangente è n.

COS(n) Calcola il coseno di un dato (in radianti) angolo.

GRAD(n) Prende un angolo, misurato in radianti, e lo riporta misurato in gradi.

ESP(n) Calcola il valore di e (circa 2,718) elevato alla potenza di n. il valore riportato sarà scorretto se non è compreso fra E-87 e E+88, dal momento che il risultato supera i valori numerici di Easel.

INT(n) Calcola il valore intero del numero, troncandolo al punto decimale. Il troncamento avviene sempre verso numeri più piccoli Quindi:

```
int(3.7)  riporta 3
int(-4.8) riporta −5
```

LN(n) Calcola il logaritmo naturale di n. Un errore ne risulta se n negativo o zero.

PI( ) Calcola il valore della costante matematica pi greeco.

RD(n) Prende un angolo misurato in gradi e lo riporta misurato in radianti.n

SGN(n) Riporta +1, −1, o 0, a seconda se l'argomento positivo, negativo o zero.

SEN(n) Calcola il valore del seno di un angolo specificato (in radianti).

QUAD(n) Calcola la radice quadrata del numero n, che non deve essere negativo.

TAN(n) Calcola la tangente dell'angolo specificato (in radianti)

Cella e Cellmax

Le due voci seguenti non sono funzioni, anche se riportano un valore. Possono solo essere usate in una formula che determina una nuova serie di valori.

CELLA In una formula, **cella** riporta il numero di ogni cella. Ha un valore di 1 nella prima cella, 2 nella seconda, e così di seguito. Un semplice esempio, per formare una serie di valori chiamata "esempio" :

```
esempio = cella
```

CELLMAX In una formula, **cellmax** riporta – in ogni cella – il numero totale di celle in uso. Puoi, per esempio, trovare il numero di celle nel grafico, formando una nuova serie di valori chiamata "dimensione" con la formula:

```
dimensione = cellmax
```

Information

PROGRAMMI QL

IMPORTA ED ESPORTA

Puoi trasferire le informazioni tra i quattro programmi del QL mediante i comandi **importa** ed **esporta**.

Le informazioni immagazzinate in QL Abacus, QL Archive e QL Easel sono simili e possono essere rappresentate in forma tabulare. Il trasferimento di informazioni tra di essi è estremamente semplice. In Abacus e Easel, i comandi **importa** ed **esporta** sono opzioni del comando file. In Archive essi sono due comandi separati.

Per prima cosa esaminiamo le operazioni di importa ed esporta tra Abacus, Archive ed Easel. La struttura dei file di esportazione prodotti mediante i tre programmi è identica e può essere importata da uno qualsiasi di essi.

Per esempio, immaginiamo di avere una tabella di Abacus che contiene le seguenti informazioni:

```
        A          B          C          D
1 |  cassa     Gennaio    Febbraio     Marzo
2 |  vendite      1000       1050      1100
3 |  costi         500        530       560
4 |  profitti      500        520       540
```

Tabella Abacus per esportazione

Se questi dati fossero importati in Easel, sarebbero interpretati come tre serie di cifre, chiamate *costi, vendite e profitti.* Easel usa i nomi dei mesi come nomi delle celle dei grafici. Le informazioni potrebbero essere:

```
etichette celle      Gennaio    Febbraio     Marzo
grafico vendite         1000       1050      1100
grafico costi            500        530       560
grafico profitti         500        520       540
```

Importato in Easel

Easel non usa la prima parola del testo *cassa*. Quando esporti una serie di cifre da Easel, esso inserisce automaticamente in questa posizione il testo, *etichette*, per mantenere la compatibilità.

Se dovessimo importare la stessa serie di cifre in Archive, il risultato sarebbe un file di dati contenente tre record, ciascuno dei quali avrebbe quattro campi, con i nomi di campo cassa$ (un campo di testo), costi, vendite e profitti (campi numerici). Il file sarebbe:

```
Campi      Record 1   Record 2   Record 3
cassa      Gennaio    Febbraio     Marzo
vendite       1000       1050      1100
costi          500        530       560
profitti       500        520       540
```

Importato in Archive

REGOLE

Affinché i dati possano essere scambiati tra i tre programmi, necessario ricordare alcune regole:

1. Quando esporti il contenuto di una tabella da Abacus, la sezione della tabella che viene esportata deve avere del testo nella prima cella di ciascuna riga (o ciascuna colonna se l'esportazione avviene in ordine di colonna).
2. Se la prima cella di ogni riga (o colonna) è vuota, quella riga (o colonna) non viene esportata.
3. Nella cella immediatamente successiva quella del testo di ogni riga (o colonna) ci devono essere dei dati. Il tipo di questi dati determina il tipo di dati nel resto della riga (o colonna). Ogni riga (o colonna) deve contenere dati tutti numerici o tutti alfabetici.
4. Puoi esportare dei file da Abacus o Archive che contengono diverse serie di dati, di testo. Easel può esportare solo un file che contenga una sola serie di dati di testo – le etichette delle celle.
5. Se importi in Easel un file che contiene più di una serie di dati di testo, Easel usa la prima come etichette di cella e ignora il resto.

STRUTTURA DEL FILE

La struttura del file di esportazione consiste in una serie di record, ognuno dei quali finisce con un <**CR**> (codice ASCII 13) e <**LF**> (codice ASCII 10). Comunque, i comandi d'importazione accettano uno di questi caratteri o i due assieme, in qualsiasi ordine. La fine del file è marcata dal carattere **CTRL Z** (codice ASCII 26).

Ogni record consiste di una serie di valori separati da virgole. I valori sono lettere (che devono essere racchiuse tra virgolette) o numeri.

Il primo valore di ogni record dev'essere alfabetico, e se il suo nome finisce con il segno di dollaro, tutti i valori seguenti devono essere alfabetici.

Il file di esportazione è prodotto esportando la serie originaria di dati di esempio da Abacus:

```
"cassa$","vendite","costi","profitti"<LF>
"Gennaio",1000,500,500<LF>
"Febbraio",1050,530,520<LF>
"Marzo",1100,560,540<LF>
```

Un file di esportazione

E' possibile generare un file di esportazione da SuperBASIC. Il programma seguente produrrà un file di esportazione, chiamato *esempio_esp*, per i dati normali.

```
100 OPEN_NEW#4,mdv2_esempio_esp
120 PRINT #4,'"cassa$","vendite","costi","profitti"'
130 PRINT #4,'"Gennaio",1000,500,500'
140 PRINT #4,'"Febbraio",1050,530,520'
150 PRINT #4,'"Marzo""1100,560,540'
160 PRINT #4, CHR$(26)
170 CLOSE #4
```

SuperBASIC aggiungerà automaticamente un carattere di alimentazione di fine riga (codice ASCII 10) alla fine di ciascun record.

ESPORTAZIONE A QUILL

QL Quill opera con testo formattato per cui i file esportati a Quill devono contenere del testo formattato piuttosto che la normale struttura di file di esportazione. Quill accetterà qualsiasi testo contenente caratteri di avanzamento di modulo (codice ASCII 12) e di avanzamento di riga (codice ASCII 10) e qualsiasi carattere ASCII stampabile. Gli avanzamenti di riga sono interpretati come indicatori di fine di paragrafo e gli avanzamenti di modulo come fine di pagina. Qualsiasi altro carattere del file viene ignorato.

Abacus ed Archive possono produrre dei file speciali per l'importazione da parte di Quill. Archive può esportare a Quill producendo un "Rapporto formattato" prodotto da **stampa**. Per esportare il rapporto invii l'uscita stampata ad un file di Microdrive usando l'opzione **esporta** del comando **sispool** (vedi capitolo 12 della Guida di Archive).

STAMPANTI CON I PROGRAMMI QL

La cartuccia originale del programma QL è protetta dalla scrittura, per cui non può essere sottoposta al procedimento di installazione della stampante. Bisogna per prima cosa fare una copia di backup della cartuccia e successivamente installare la stampante con la copia.

Ognuno dei quattro programmi Psion QL può stampare testo su quasi tutte le marche di stampanti che abbiano una interfaccia RS-232-C.

E' possibile impostare la stampante in modo che possa stampare carta continua o fogli singoli. Se la stampante usa fogli di carta singoli, essa si fermerà alla fine del foglio, e sullo schermo comparirà un messaggio che chiede di introdurre un altro foglio. Premi **ENTER** per continuare o **ESC** per abbandonare il documento.

PROGRAMMI DI COMANDO DELLA STAMPANTE

La stampante viene comandata da un programma speciale chiamato **printer driver**. Esso può essere modificato per usare qualsiasi stampante.

Un carattere non stampabile, che non sia un fine riga o un a capo, deve essere preceduto da un codice ASCII 0 (ZERO), per indicare al programma di comando della stampante che deve essere emesso. Per esempio, il comando della Epson FX-80 per stampare caratteri in grassetto è il codice ASCII 27 **(ESC)** e **E**. ESC è un carattere non stampabile e perciò dev' essere preceduto da uno ZERO. Puoi inviare i codici necessari da Archive con l' istruzione:

```
stampa car(0) + car(27) + "E"
```

In ABACUS lo stesso compito può essere eseguito scrivendo:

```
car(0) + car(27) + "E"
```

in una cella nel punto in cui deve cominciare la stampa in grassetto.

La modifica di QL Quill, QL Abacus e QL Archive per adattarli alle altre stampanti viene chiamata l' installazione del software e viene fatta usando il programma di installazione da SuperBASIC. Il programma d' intallazione (**install__bas**), i dati di installazione delle varie stampanti (**install__dat**) e i dati d' installazione della stampante corrente (**printer__dat**) sono sulle cartucce di programma QL Quill e QL Abacus. Puoi usare il programma per installare una stampante per QL Archive anche se la cartuccia di Archive non contiene il programma d' installazione o i dati d' installazione.

I programmi di Abacus, Archive e Quill usano solo le informazioni di **printer__dat**.

INSTALLAZIONE DI UNA STAMPANTE SERIALE

Per esempio, per installare Quill in modo che funzioni con una Epson FX-80, dotata di una interfaccia RS-232-C, inserisci la cartuccia Quill nel Microdrive 1, ma non farla girare. Mentre sei in SuperBASIC scrivi:

```
lrun mdv1_install_bas
```

e verrà eseguito il programma d' installazione. Il programma richiede che **"install__dat"** sia nella cartuccia nel Microdrive 1, in modo da non cancellarlo.

Per prima cosa scegli il Microdrive in cui verrà installata la stampante. In questo caso premi 1, seguito da **ENTER**, per installarla nel Microdrive 1. Poi premi **ENTER** per selezionare una stampante seriale (collegata all' elaboratore attraverso la porta seriale **ser1** o **ser2**).

Successivamente il programma legge i dati d' installazione e mostra l' elenco dei nomi delle stampanti per cui viene fornito un programma di comando della stampante.

Scegli una stampante dall' elenco mediante i tasti di movimento verticale del cursore, finché è evidenziata la stampante richiesta e poi premi **F5** per installarla. Devi confermare l' installazione premendo **ENTER**; qualsiasi altro tasto annullerà l' installazione e rimanderà all' elenco dei nomi delle stampanti.

Quando l' installazione è completa, verrai rimandato a SuperBASIC. La prossima volta che Quill viene caricato, sarà impostato per usare la stampante da te scelta, compresi i caratteri in grassetto, la sottolineatura, i pedici e gli apici.

Puoi togliere una stampante dall'elenco premendo **F3**, e memorizzare tutti i programmi di comando delle stampanti dell'elenco premendo **F4**. Poiché queste due opzioni operano delle modifiche irreversibili alle informazioni del drive della stampante devono essere confermate premendo **ENTER**.

ALTRE STAMPANTI SERIALI

Se la tua stampante non è compresa nell'elenco mostrato dal programma d'installazione, hai due possibilità:

Non fare nulla

Esci dal programma d'installazione premendo **ESC**. Tutti e quattro i programmi dispongono di un semplice programma di stampa che dovrebbe essere in grado di stampare testo ordinario con quasi tutte le stampanti.

Installarlo

Aggiungere un nuovo nome all'elenco delle stampanti. Per farlo ci sono tre modi:

1. Usa il tasto di movimento verso il basso del cursore per scegliere la voce chiamata "ALTRA". Premi **F1** o **F2** per creare una nuova voce, che potrai impostare per la tua stampante.
2. Scegli un nome di stampante esistente e premi **F1** per creare una nuova stampante con gli stessi valori della vecchia. Usa questa opzione se la tua stampante è simile a una stampante che si trova già nell'elenco.
3. Scegli una stampante esistente e premi **F2**. Questo non produce una nuova copia, ma permette di cambiare i valori di una stampante esistente. Non usare questa opzione a meno di non essere sicuro dei cambiamenti che intendi fare.

In ciascun caso ti viene mostrato l'elenco dei parametri della stampante da cambiare. Premi i tasti di movimento verticale per scegliere una voce e i tasti di movimento orizzontale del cursore per cambiarla.

Nell'elenco ci sono due tipi di voci:

- quelle con una varietà di valori possibili, quali il NOME DEL PROGRAMMA DI COMANDO DELLA STAMPANTE, e IL CODICE DI FINE RIGA.
- e quelle con una serie di valori limitata, quali la PARITA.

	Default	Altre opzioni
NOME DRIVER	: DEFAULT	....
PORTA	: ser1	ser2
BAUD RATE	: 9600	75, 300, 600, 1200, 2400, 4800
PARITA	: NESSUNO	SPAZIO, SEGNO, DISPARI, PARI
RIGHE/PAGINA	: 66	da 0 a 255
CARATTERI/RIGA	: 80	da 0 a 255
MODULI CONTINUI	: SI	NO
CODICE FINE RIGA	: CR, LF	....
PREAMBOLO	: NESSUNO	....
CODICE POSTAMBOLO	: NESSUNO	....
GRASSETTO ACCESO	: NESSUNO	....
GRASSETTO SPENTO	: NESSUNO	....
SOTTOLINEA ACCESO	: NESSUNO	....
SOTTOLINEA SPENTO	: NESSUNO	....
PEDICE ACCESO	: NESSUNO	....
PEDICE SPENTO	: NESSUNO	....
APICE ACCESO	: NESSUNO	....
APICE SPENTO	: NESSUNO	....
TRADUZIONE1	: NESSUNO	....
TRADUZIONE2	: NESSUNO	....
TRADUZIONE3	: NESSUNO	....
TRADUZIONE4	: NESSUNO	....
TRADUZIONE5	: NESSUNO	....
TRADUZIONE6	: NESSUNO	....
TRADUZIONE7	: NESSUNO	....
TRADUZIONE8	: NESSUNO	....
TRADUZIONE9	: NESSUNO	....
TRADUZIONE10	: NESSUNO	....

I valori di ciascun tipo vengono cambiati in modi differenti. Lo schema più sotto riporta i valori dati alla stampante di DEFAULT. Alla destra dello schema ci sono altri valori possibili (per quelli con serie limitata).

Per ciascuna delle voci che ha un numero limitato di opzioni, il valore cambia ogni volta che si preme il tasto di movimento verso sinistra o destra.

Per le altre voci, se si preme uno di questi tasti del cursore si cancella il valore esistente; a questo punto scrivi il tuo valore e premi **ENTER**. Tutte queste voci, eccetto che il NOME DEL PROGRAMMA DI COMANDO STAMPANTE, accetteranno elenchi lunghi fino a dieci codici separati da virgole. Ogni codice può essere scritto in differenti modi:

1. Un numero tra 0 e 255
2. Un numero esadecimale, preceduto da un segno di dollaro, tra $0 e $FF
3. Qualsiasi carattere singolo, preceduto da un simbolo di virgolette (" o ')
4. Un normale codice mnemonico di controllo ASCII, in maiuscole o minuscole

NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL
BS	HT	IF	VT	FF	CR	SO	SI
DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB
CAN	EM	SUB	ESC	FS	GS	RS	US

5. Il testo DEF (o def) fa usare alla stampante una azione di default, che fa arretrare la stampante per produrre l'effetto desiderato. Dev' è essere usato solo per enfatizzare e sottolineare. Queste voci devono essere usate in coppia, per esempio, se in DEF impostato SOTTOLINEA ACCESO dev'essere impostato anche SOTTOLINEA SPENTO. La stampante dev'essere in grado di rispondere al codice di arretramento ASCII.

Oppure puoi semplicemente premere **ENTER** per selezionare NESSUNO. Sei libero di mescolare i vari metodi in qualsiasi modo desideri.

Il NOME DRIVER contiene il nome del modello della stampante. E' il nome mediante il quale puoi identificare il programma di comando della stampante. Il nome non dev'essere lungo più di 16 caratteri. Per cambiare questa voce premi un tasto di movimento orizzontale, scrivi il nome che vuoi e premi **ENTER**.

La PORTA ser1 e ser2 e sceglie una delle porte seriali dei due standard.

Il BAUD RATE determina la velocità con cui i caratteri vengono trasmessi attraverso una interfaccia seriale, in termini di numero di bit al secondo. 110 baud equivalente a circa 10 caratteri al secondo, 300 baud circa 30 caratteri al secondo, ecc. Il baud rate del programma di comando della stampante deve corrispondere a quello dell'interfaccia seriale della tua stampante.

La voce PARITA dipende dal modo in cui la tua stampante tratta il bit più significativo (cifra binaria) nei dati inviati dall'elaboratore. Tutti i codici ASCII si trovano tra 0 e 127 e possono essere rappresentati da un numero binario a 7 cifre. Molte stampanti seriali si aspettano che venga inviato un carattere con un valore di sette bit. Altre stampanti possono aspettarsi valori a otto bit, che accettano codici tra 0 e 255. I codici addizionali, tra 128 e 255, possono essere stampati come caratteri grafici o accentati. La tua stampante può interpretare l'ottavo bit di un codice a 8 bit come un *codice di parità*, usato per controllare se c'è stato un errore durante la trasmissione di un carattere. La tua stampante può usare una parità PARI (il bit di parità viene impostato a 0 od 1, per cui il numero totale è di 1 in ogni codice di carattere pari) o parità DISPARI (il numero totale di 1 dispari). Se la tua stampante non controlla la parità, puoi scegliere SPAZIO (l'ottavo bit sempre 0) o SEGNO (l'ottavo bit sempre 1). Se imposti NESSUNO, permetti a tutti e otto i bit di essere inviati alla stampante.

RIGHE/PAGINA e CARATTERI/RIGA specifica il numero massimo di righe di testo (comprese le righe vuote se vuoi stampare del testo con spaziatura doppia o tripla) su ogni pagina, e il numero massimo di caratteri su ogni riga. I valori usati nel programma di comando della stampante fornito sono adatti per l'uso su fogli di formato A4.

MODULI CONTINUI specifica se la tua stampante usa dei moduli continui (SI) o fogli singoli (NO). Se stai stampando su fogli singoli, la stampante si fermerà alla fine di ogni pagina. Sullo schermo comparirà un messaggio, che ti chiederà di inserire un nuovo foglio. Premi **ENTER** per ricominciare la stampa, o premi **ESC** per terminare la stampa.

Il CODICE DI FINE RIGA è la sequenza di codice da inviare alla stampante per indicare la fine della riga. La maggior parte delle stampanti accetterà un a capo seguito da un avanzamento di riga. Scegli un avanzamento di riga come indicatore di fine di riga se vuoi stampare in un file un programma di SuperBASIC.

I CODICI DI PREAMBOLO E POSTAMBOLO possono essere necessari se la tua stampante richiede una sequenza di inizializzazione prima di usarla per la prima volta. Per esempio, puoi desiderare di impostare le posizioni del margine della stampante, o scegliere un set di caratteri particolare. E' anche possibile che tu voglia ripristinare i valori originari, una volta finito di usare uno dei programmi. Le voci di preambolo e postambolo ti permettono di specificare una sequenza lunga fino a 10 caratteri da inviare alla stampante per questi due scopi.

Le voci GRASSETTO ACCESO e SPENTO contengono i codici per attivare e disattivare la stampa in grassetto. Se la tua stampante non può stampare caratteri in grassetto puoi usare il valore DEF, descritto precedentemente, a condizione che la stampante risponda ad un carattere di arretramento.

SOTTOLINEA ACCESO e SPENTO attiva e disattiva la sottolineatura, a condizione che la tua stampante abbia la possibilità di fare la sottolineatura automatica. Se la tua stampante non può stampare caratteri sottolineati, puoi usare il valore DEF, descritto in precedenza, a condizione che la stampante risponda ad un carattere di arretramento.

Usa le voci di PEDICE ACCESO/SPENTO ed APICE ACCESO/SPENTO per la sequenza di codici necessari alla tua stampante per attivare e disattivare la stampa di caratteri sopra il rigo e sotto il rigo.

Da TRADUZIONE1 a TRADUZIONE10 accettano fino a dieci caratteri. Il primo carattere specificato viene tradotto nella sequenza successiva di caratteri prima di essere inviato alla stampante. Il primo carattere non dev'essere un carattere di controllo (il suo codice ASCII dev'essere nella serie da 32 a 255). La traduzione può contenere qualsiasi carattere. Il risultato deve essere in un carattere singolo una volta stampanto.

Per esempio, creiamo un secondo programma di comando stampante per la Epson FX-80. Cominciamo richiamando ed eseguendo il programma d'installazione da SuperBASIC. Scegliamo il programma chiamato ALTRO e premiamo **F1** o **F2**. I valori iniziali vengono elencati più sotto, e la colonna a destra riporta i valori necessari alla FX-80:

	Default	Altre opzioni
NOME DRIVER	: ALTRA	....
PORTA	: ser1	: ser2
BAUD RATE	: 9600	: 9600
PARITA	: NESSUNO	: NESSUNO
RIGHE/PAGINA	: 66	: 66
CARATTERI/RIGA	: 80	: 80
MODULI CONTINUI	: NO	: SI
CODICE FINE RIGA	: CR, LF	: CR, LF
CODICE PREAMBOLO	: NESSUNO	: esc,@,ESC,R,NUL
CODICE POSTAMBOLO	: NESSUNO	: NESSUNO
ENFASI ACCESO	: NESSUNO	: ESC,E
ENFASI SPENTO	: NESSUNO	: ESC,F
SOTTOLINEA ACCESO	: NESSUNO	: ESC,1
SOTTOLINEA SPENTO	: NESSUNO	: ESC,0
PEDICE ACCESO	: NESSUNO	: ESC,S,1
PEDICE SPENTO	: NESSUNO	: ESC,T
APICE ACCESO	: NESSUNO	: ESC,S,0
APICE SPENTO	: NESSUNO	: ESC,T
TRADUZIONE1	: NESSUNO	: £,ESC,R,ETX,@,ESC,R,NULL
TRADUZIONE2	: NESSUNO	: NESSUNO
TRADUZIONE3	: NESSUNO	: NESSUNO
TRADUZIONE4	: NESSUNO	: NESSUNO
TRADUZIONE5	: NESSUNO	: NESSUNO
TRADUZIONE6	: NESSUNO	: NESSUNO
TRADUZIONE7	: NESSUNO	: NESSUNO
TRADUZIONE8	: NESSUNO	: NESSUNO
TRADUZIONE9	: NESSUNO	: NESSUNO
TRADUZIONE10	: NESSUNO	: NESSUNO

Per prima cosa cambia il nome del driver, premi il tasto di movimento verso destra del cursore per cancellare il testo esistente e scrivi

```
FX-80 [ENTER]
```

Se fai un errore puoi ripetere il procedimento.

Premi il tasto di movimento verso il basso del cursore, finché viene evidenziata la voce MODULI CONTINUI. Ci sono solo due opzioni; scegli SI' premendo il tasto di movimento verso destra o sinistra del cursore.

Una sequenza di PREAMBOLO adatta all'Epson FX-80 è **ESC**, che inizializza la stampante e svuota il suo buffer di stampa. Bo'sogna, inoltre, impostare la stampante per usare il set di caratteri americano (per stampare il simbolo di hash e il segno di sterlina – vedi più avanti). Il codice FX-80 per farlo è **ESC R NUL**

Usa i tasti del cursore per scegliere il PREAMBOLO e premi il tasto di movimento del cursore verso destra (o sinistra) per cancellare il valore corrente. Le seguenti tre opzioni producono tutte lo stesso risultato ed inizializzano la stampante:

```
ESC,"@,ESC,"R,NUL
27,64,27,82,0
$1B,$40,$1B,$52,$0
```

Potresti usare questa voce per impostare altre caratteristiche della stampante, quale la spaziatura della riga o i caratteri italici. Se la tua stampante non richiede alcuna inizializzazione, puoi lasciare l'impostazione iniziale a NESSUNO.

La FX-80 non richiede un POSTAMBOLO, per cui l'impostazione può essere mantenuta a NESSUNO.

I codici di ENFASI ACCESO ed ENFASI SPENTO per la Epson FX-80 sono rispettivamente ESC E ed ESC F. Puoi impostarle scrivendo

```
esc,"E
esc,"F
```

I codici rimanenti possono essere impostati scrivendo:

Voce	Scrivi
SOTTOLINEA ACCESO	`esc,"-,"1`
SPENTO	`esc,"-,"0`
PEDICE ACCESO	`esc,"S,"1`
SPENTO	`esc,"T`
APICE ACCESO	`esc,"S,"0`
SPENTO	`esc,"T`
TRADUZIONE1	`£,esc,R,ETX,#,ESC,R,NUL`

Nell'esempio di cui sopra, **TRADUZIONE1** permette alla Epson FX-80 di stampare un segno di sterlina, che è disponibile solo nel set di caratteri inglese. Il segno di sterlina del QL viene tradotto per:

Attivare il set di caratteri inglesi, stampare un simbolo di hash, (che compare come un segno di sterlina) ritornare al set di caratteri americano

Quando hai finito di editare i codici della stampante, puoi installarla premendo **F5**. Oppure puoi ritornare all'elenco di stampanti, pronto a fare cambiamenti di modo.

INSTALLAZIONE PER QL ARCHIVE

Metti una cartuccia QL Quill o QL Abacus nel Microdrive1 e una cartuccia di QL Archive nel Microdrive2. Richiama ed esegui **install_bas** dal Microdrive1, ma poi premi 2, seguito da **ENTER**, per indicare che vuoi installare una stampante nel Microdrive2.

Segui normalmente la procedura di installazione. I dati della installazione saranno letti dal Microdrive1, ma la stampante sarà installata sulla cartuccia nel Microdrive2.

STAMPANTI PARALLELE

il programma d'installazione permette l'installazione di una stampante collegata al QL attraverso delle porte che non siano **ser1** o **ser2**. Usa questa opzione se, per esempio, hai aggiunto una interfaccia parallela opzionale. Richiama ed esegui **install_bas** come descritto in precedenza. Dopo che hai scelto l'installazione sul Microdrive 1 o 2, premi lo spazio per scegliere l'opzione di porta parallela.

L'elenco di stampanti compare come in precedenza, ma quando premi **F1** o **F2** l'elenco di parametri appare come riportato nella tabella seguente.

	Default	Altre opzioni
NOME DRIVER	: DEFAULT	....
PORTA	: NESSUNO	....
RIGHE/PAGINA	: 66	da 0 a 255
CARATTERI/RIGA	: 80	da 0 a 255
MODULI CONTINUI	: SI'	NO
CODICE FINE RIGA	: CR, LF	....
CODICE PREAMBOLO	: NESSUNO	....
CODICE POSTAMBOLO	: NESSUNO	....
ENFASI ACCESO	: NESSUNO	....
ENFASI SPENTO	: NESSUNO	....
SOTTOLINEA ACCESO	: NESSUNO	....
SOTTOLINEA SPENTO	: NESSUNO	....
PEDICE ACCESO	: NESSUNO	....
PEDICE SPENTO	: NESSUNO	....
APICE ACCESO	: NESSUNO	....
APICE SPENTO	: NESSUNO	....
TRADUZIONE1	: NESSUNO	....
TRADUZIONE2	: NESSUNO	....
TRADUZIONE3	: NESSUNO	....
TRADUZIONE4	: NESSUNO	....
TRADUZIONE5	: NESSUNO	....
TRADUZIONE6	: NESSUNO	....
TRADUZIONE7	: NESSUNO	....
TRADUZIONE8	: NESSUNO	....
TRADUZIONE9	: NESSUNO	....
TRADUZIONE10	: NESSUNO	....

Non ti viene data l'opzione di scegliere il baud rate o la parità poiché essi si riferiscono solo ad una interfaccia seriale attraverso ser1 o ser2. Anche la selezione della PORTA differente. Cambia questa voce premendo il tasto di movimento verso sinistra o destra del cursore e poi scrivendo qualsiasi *nome di dispositivo* valido lungo fino a sedici caratteri. Consulta la sezione *dispositivi* dei *Concetti del QL*, o il manuale che accompagna una interfaccia aggiuntiva.

A parte queste differenze, il resto dell'installazione è esattamente lo stesso descritto per una interfaccia seriale.

L'UTILITA CONVERT

La versione 2.0 del programma **install_bas** è stata modificata per offrire una serie più ampia di opzioni della stampante. Questo significa che non è compatibile con i file **install_dat** creati con la versione 1. Per convertire i file **install_dat** della versione 1, viene fornito un programma di conversione, chiamato convert_bas, in modo che siano leggibili dal programma d'installazione della versione 2.0.

Per prima cosa copia **convert_bas** su un'altra cartuccia. Metti la cartuccia contenente la copia di **convert_bas** nel Microdrive1 e una cartuccia contenente il tuo file **install_dat** versione 1 nel Microdrive2. Esegui il programma scrivendo:

```
lrun mdv1_convert_bas
```

Il programma legge il file install_dat nel Mricrodrive2 e scrive la nuova versione sul Microdrive1. Nota che la nuova versione sostituirà qualsiasi file install_dat su questa cartuccia. Successivamente puoi, se necessario, copiare il nuovo file install_dat su un'altra cartuccia.

PROGRAMMA QL
CONFIG

Programma **config_bas** ti permette di specificare i dispositivi di default in alternativa e per il programma QL e di modificare l'ordine di ordinamento nei comandi Ordina di Abacus e Archive.

I programmi presumono di usare il Microdrive2 per l'immagazzinamento dei dati, e mentre che le informazioni di Aiuto e il programma di comando della stampante sono sul Microdrive1. E' possibile che tu voglia modificarli per usare Microdrive aggiuntivi, unità a disco ecc.

Forse desidererai anche modificare l'ordine di ordinamento dei record di Archive, o di un tracciato di Abacus. Questo può essere utile, per esempio, se vuoi ordinare un testo che comprende dei caratteri accentati di una lingua straniera.

IMPIEGO DI CONFIG

Puoi eseguire **config_bas** da qualsiasi Microdrive, e modificare un programma di QL su una cartuccia nel Microdrive 1 o Microdrive 2. Immagina di voler eseguire **config_bas** dal Microdrive 2 per modificare una copia di un programma QL nel Microdrive 1. Esegui il programma scrivendo

```
lrun mdv2_config_bas
```

Quando ti verrà chiesto, scrivi il nome del programma che vuoi modificare (Quill, Abacus, Easel od Archive) e premi **ENTER**. Poi, quando ti viene chiesto quale drive contiene il programma, batti il numero 1.

Il programma attende che tu prema la barra spaziatrice dopo che ti sei accertato che la cartuccia del programma sia nel Microdrive corretto. Una volta che l'hai fatto, il programma ti mostra il menù delle opzioni principali che sono: **Selezionare nuovi dispositivi di default Modifica l'ordine di ordinamento Lasciare il programma.**

Ordinamento

Per scegliere l'opzione di modificare l'ordine di ordinamento premi **ENTER**. Quando ti viene chiesto di farlo, premi la barra spaziatrice.

La zona più grande dello schermo mostra un blocco di 256 caratteri che definisce l'ordine di ordinamento. La *posizione* nel blocco, leggendo da sinistra a destra e dall'alto in basso, determina quali caratteri vengono ordinati; il *contenuto* in quella posizione mostra in che modo il carattere verrà provato dal comando Ordina. La parte destra dello schermo mostra informazioni addizionali sul carattere indicato dal cursore. Sposta il cursore di carattere in carattere con i tasti del cursore.

Il blocco di caratteri alla base dello schermo viene usato per modificare l'ordine. Inoltre esso contiene un cursore, che puoi spostare con **SHIFT** e i tasti del cursore. Questo blocco mostra solo metà della serie completa di caratteri – premi **F1** per passare da una metà all'altra.

Il miglior modo per descrivere come modificare l'ordine di ordinamento è di usare degli esempi. Inizialmente i caratteri minuscoli verranno ordinati in modo da essere successivi a tutti quelli maiuscoli, cioè "a" verrà dopo "Z". Immagina di voler rendere l'ordine indipendente dalle maiuscole o minuscole, cosicché, per esempio, "A" e "a" non vengano distinte.

Affinché "a" venga ordinata come se fosse una "A", porta il cursore nel blocco principale di caratteri sulla lettera "a" e premi il tasto "A" (assicurati di premere il carattere maiuscolo). La "a" del blocco superiore diventa una "A", e l'informazione sulla destra dello schermo mostra che il carattere "a" verrà ora considerato come equivalente a "A" per l'ordinamento.

Ripeti questo procedimento per ogni lettera minuscola, rendendo la "b" equivalente alla "B", la "c" alla "C" eccetera.

Un altro modo per cambiare un carattere è di portare il cursore nel blocco inferiore di caratteri, usando **SHIFT** e i tasti del cursore, finché si trova sul carattere che vuoi, e poi premi **F2**. Questo metodo è particolarmente utile per i caratteri, come quelli stranieri accentati, che non sono indicati sui tasti. Questo metodo viene usato nell'esempio seguente.

Immagina di voler invertire l'ordine di ordinamento normale, lasciando immutato il resto dell'ordinamento. Per farlo devi cambiare la parte del blocco principale su cui è scritto "A B C ... X Y Z" in modo che ci sia scritto "Z Y X . . . C B A". Porta il cursore principale su "A" e il cursore inferiore su "Z" e premi **F2** per immettere il nuovo carattere. Il carattere "A" verrà quindi ordinato come se fosse "Z". Ripeti la procedura per ogni lettera maiuscola, cambiando "B" in "Y", "C" in "X" eccetera.

Una volta che hai completato la definizione dell'ordine di ordinamento, premi **F5** per memorizzare il nuovo ordine nel programma QL, sostituendo quello vecchio. Premi **ESC** per ritornare al menù principale.

Selezione del dispostivo

Inizialmente i programmi QL usano il Microdrive 1 per l'informazione del sistema (il programma di comando della stampante installato, per esempio) e per l'Aiuto. Per i loro dati usano tutti il Microdrive 2.

Dal menù principale premi la barra spaziatrice per scegliere i nuovi dispositivi di default. Premi nuovamente la barra quando ti viene chiesto.

Dopo aver letto le impostazioni correnti dalla cartuccia del programma, il programma mostra questi valori e attende che tu scriva le nuove scelte. Premi **ENTER** per mantenere il vecchio valore, o scrivi la nuova selezione e premi **ENTER**

Dopo aver fatto la tua scelta, puoi memorizzare i nuovi dispositivi, riselezionare i dispositivi o annullare questa opzione e ritornare al menù principale.

Se memorizzi la selezione del dispositivo, il programma QL userà questi dispositivi fino a che usi **confing_bas** per cambiarli nuovamente.

ARITMETICA DEL QL

Eccetto che per Easel, la serie valida dei numeri nei programmi QL è:

da ± 2.9*10^39 a ± 1.7*10^38

Tutti calcoli sono accurati fino alla sedicesima cifra significativa, ma possono essere visualizzati solo fino a quattordici cifre.

In Easel la serie di numeri validi

da ± 1.0^ −35 a ± 1.0 x 10^ +36

Abacus, Archive ed Easel forniscono i seguenti operatori aritmentici:

Operatore	Funzione
+	Addizione sui numeri o concatenazione sulle stringhe
−	Sottrazione
*	Moltiplicazione
/	Divisione
^	Elevamento a potenza
=	Uguale
>	Maggiore di
<	Minore di
< =	Minore di o uguale a
> =	Maggiore di o uguale a
< >	Diverso da

Tra i tipi di dati non c'è coerzione automatica. Perciò, gli operandi devono essere dello stesso tipo. Il risultato è sempre un numero, 1 se il confronto è vero e 0 se falso.

Funzioni ed operatori hanno la seguente precedenza:

Operazione	Precedenza
Pedici e slicing	12
Tutte le funzioni	11
^	10
Meno unitario	9
*,/	8
+.−	6
=,>,<,< =,> =,< >	5
non	4
ed	3
od	2

PROCEDURA DI FORMAT-TAZIONE

La formattazione di una cartuccia provoca la cancellazione di eventuali dati precedentemente registrati sulla cartuccia. Questi dati non possono essere recuperati, per cui accertati di formattare soltanto cartucce vuote o cartucce che non contengono informazioni utili.

Per prima cosa decidi il nome per la cartuccia, che non sia più lungo di dieci caratteri. Con il QL acceso e con il cursore lampeggiante sullo schermo, metti la cartuccia da formattare nel Microdrive 1. Immaginiamo che il nome della cartuccia sia "dati". Poi scrivi

```
FORMAT mdv1_dati
```

Non confondere il simbolo di sottolineatura (_) con il segno meno (–), poiché sono sullo stesso tasto. Il simbolo di sottolineatura è quello superiore, per cui devi tenere premuto **SHIFT** mentre premi il tasto.

Premi **ENTER** e la luce del Microdrive sinistro si accenderà per circa trenta secondi. Sullo schermo comparirà un messaggio che indicherà quanto spazio è disponibile su quella cartuccia. Il comando **FORMATTA** descritto a fondo nella sezione *Parole chiave*.

E' buona pratica formattare parecchie volte una cartuccia nuova. Questo aiuterà a far scorrere bene il nastro e può aumentarne la capacità.

Sarebbe stato ugualmente possibile formattare la cartuccia nel Microdrive2 scrivendo **mdv2_** invece di **mdv1_**.

PROCEDURA DI BACKUP

Si fa il backup copiando tutti i file contenuti in una cartuccia su una cartuccia vuota. E' meglio che la cartuccia vuota sia stata appena formattata e che il suo nome indichi che si tratta di un backup.

Scegli una cartuccia vuota o una cartuccia che non contenga alcuna informazione utile e mettila nel Microdrive1. Decidi il nome della cartuccia, per esempio, se il nome della cartuccia da copiare "QL_dati", un buon nome per la cartuccia di backup sarebbe "QL_dati_bak". Poi scrivi:

```
FORMAT mdv1_dati_bak
```

seguito da **ENTER**. La luce del Microdrive sinistro si accenderà per circa trenta secondi.

Metti la cartuccia da copiare nel Microdrive2 e scrivi

```
DIR mdv2_
```

questo elencherà tutti i file contenuti su questa cartuccia.

Per ogni file elencato scrivi:

```
COPY mdv2_nomefile TO mdv1_nomefile
```

mettendo in nome di file relativo dov'è indicato. Questo comando copierà tutti i file specificati dal Microdrive 2 al Microdrive1. La velocità di questa operazione dipende dalle dimensioni dei file che vengono copiati: è possibile che per l'operazione ci voglia parecchio tempo.

Ripeti il comando **COPIA** per ciascuno dei file elencati. Una volta completa, bisogna segnare sulla cartuccia di backup (quella nel Microdrive2) i dati e il nome della cartuccia di cui una copia, e poi metterla in un posto sicuro.

Normalmente, per ciascuna cartuccia con cui lavori e che contiene i dati, devi avere uno, due o più cartucce di backup, a seconda di quanto sono importanti i dati. Se usi questo sistema, fai sempre una copia sulla più vecchia cartuccia di backup della serie.