Speedy Computer

primi passi in basic

una facile guida per scrivere programmi

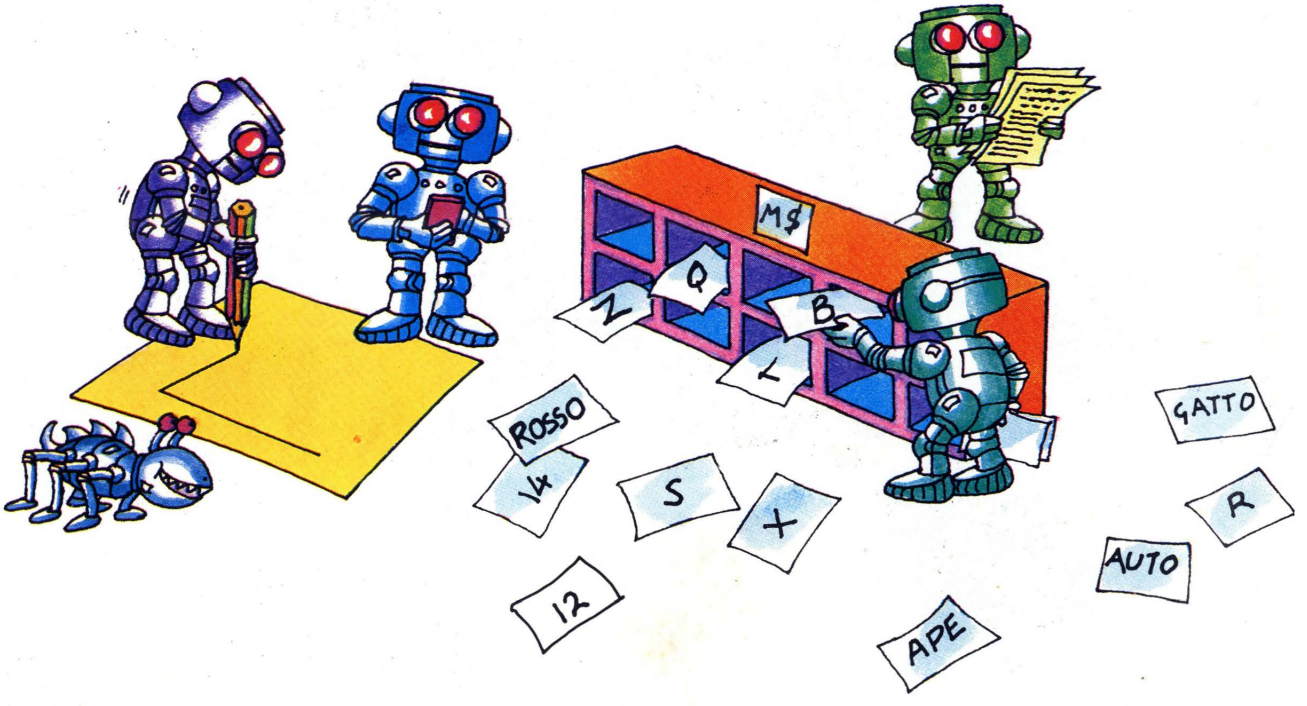
MOLTI PROGRAMMI

GRUPPO EDITORIALE JACKSON

primi passi in basic

Brian Reffin Smith
e Lisa Watts
illustrazioni: Graham Round
Traduzione: Pier Luigi Cecioni

Indice



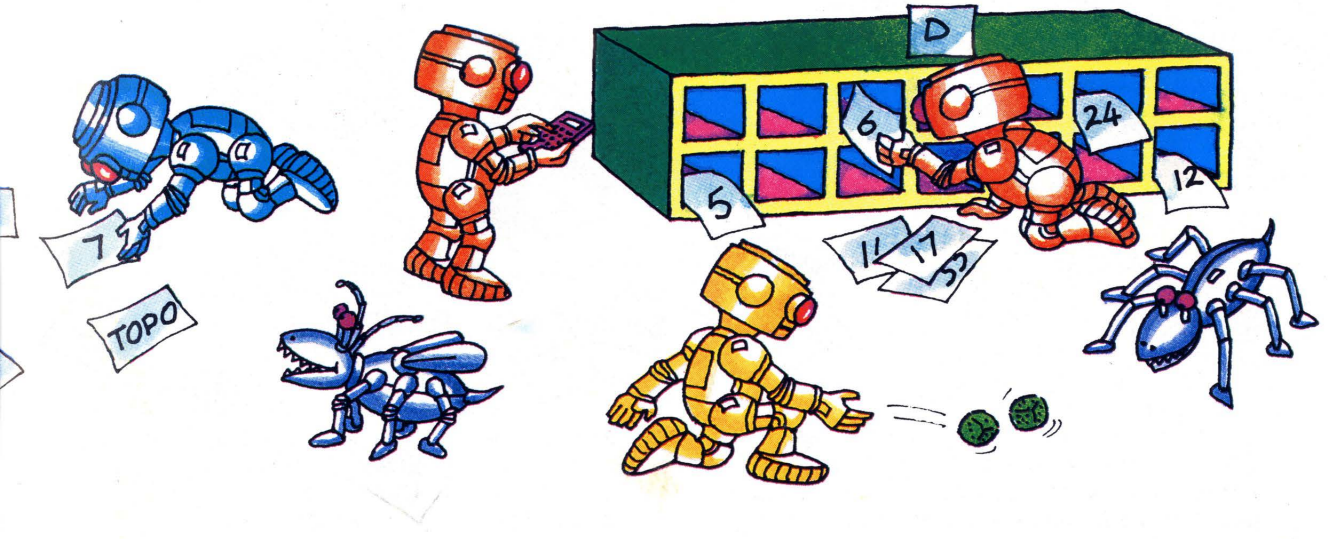
A proposito di questo libro

Questo libro è una guida, passo per passo, a una comprensione dei programmi e a una migliore conoscenza del BASIC. Anche chi non ha intenzione di programmare, dopo aver capito capito come funziona il BASIC, troverà facile modificare i programmi altrui o correggerne gli errori, e a quel punto sarà praticamente in grado di scrivere i propri programmi.

Il libro inizia con una breve spiegazione dei principali comandi del BASIC, con molti esempi che ne illustrano l'operato. Successivamente viene mostrata la loro utilizzazione nei programmi per svolgere operazioni anche molto complicate, come creare un archivio, tracciare figure sullo schermo e ordinare dati.

I programmi sono scritti in BASIC "standard", cioè in una versione del BASIC che, con piccole variazioni, può essere utilizzata sulla maggior parte dei personal computer. (Tutti i programmi sono stati provati su Apple, e le istruzioni grafiche riportate nei listati sono relative a questo computer.) Alle pagine 10-11 viene spiegato come modificare i programmi per adattarli a quei personal computer; come quelli Sinclair, il cui BASIC è un po' diverso da quello standard, i cambiamenti necessari sono alla fine del libro.

Dettagliate spiegazioni a fianco dei programmi illustrano il funzionamento dei comandi e alcune utili tecniche, oltre che delle routine che puoi utilizzare nei tuoi programmi. Vengono presentate anche molte idee per compiere esperimenti sui programmi e adattarli ad altre operazioni.

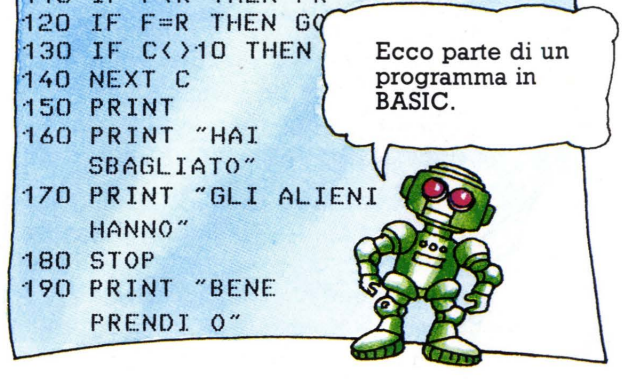


Introduzione al BASIC

Il linguaggio di programmazione BASIC comprende un centinaio di parole, ognuna delle quali ordina al computer di fare qualcosa. Affinché il computer esegua una determinata operazione, gli vanno forniti una serie di istruzioni e un insieme di informazioni su cui operare: queste istruzioni e queste informazioni costituiscono il "programma". Per le istruzioni si possono utilizzare solo le parole del BASIC. Vanno inoltre seguite le regole del linguaggio che ne costituiscono la "sintassi".

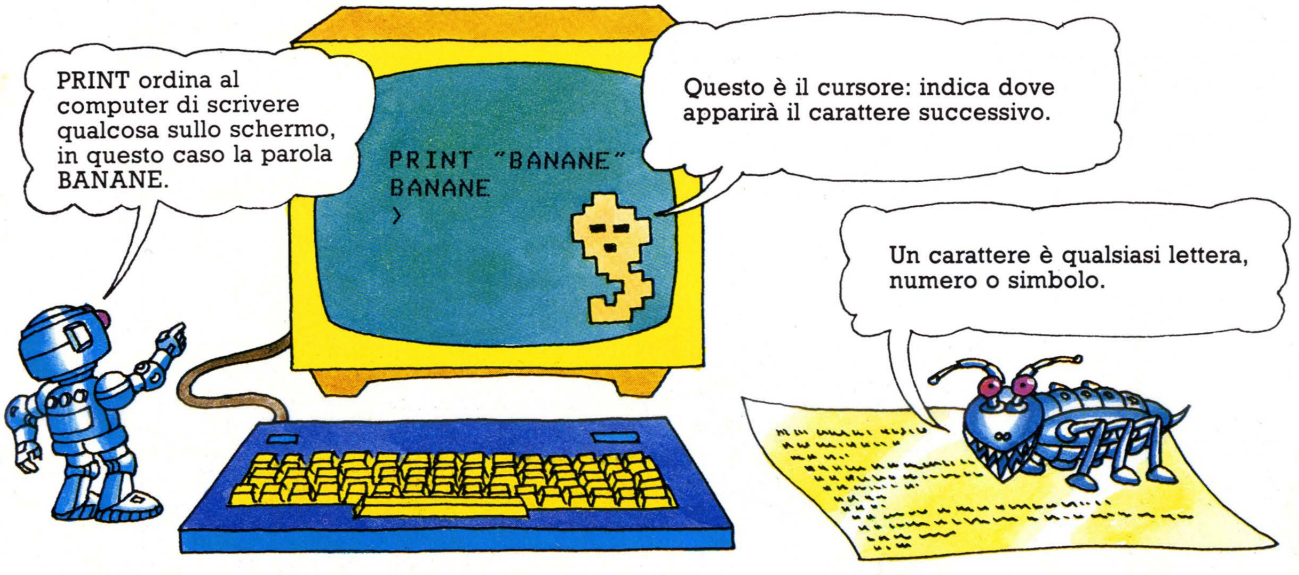
Nel BASIC ogni riga contenente un'istruzione è numerata; di solito la numerazione procede di dieci in dieci, per consentire l'inserimento di eventuali altre righe senza dover rinumerare tutto il programma.

```
 10 CLS
 20 PRINT "DECOLLO ASTRONAVE'
 30 LET G=INT(RND(1)*20+1)
 40 LET W=INT(RND(1)*40+1)
 50 LET R=G*W
 60 PRINT "GRAVITA'=";G
 70 PRINT "PER FAVORE FORNISCI
    LA FORZA"
 80 FOR C=1 TO 10
 90 INPUT F
100 IF F>R THEN PRINT "TROPPO
    GRANDE"
110 IF F<R THEN PR
120 IF F=R THEN G
130 IF C<>10 THEN
140 NEXT C
150 PRINT
160 PRINT "HAI
    SBAGLIATO"
170 PRINT "GLI ALIENI
    HANNO"
180 STOP
190 PRINT "BENE
    PRENDI O"
```



Battitura di un programma

Quando batti un programma sul computer devi premere il tasto RETURN (o ENTER, o NEWLINE, a seconda del computer) al termine di ogni riga. In tal modo il computer inserisce la riga nella propria memoria ed è pronto a ricevere la successiva. Dopo aver scritto tutte le righe del programma, batti RUN (esegui), e il computer eseguirà le istruzioni.



se batti una riga non preceduta da un numero, il computer esegue l'istruzione non appena premi RETURN (o ENTER, o NEWLINE): si tratta di un "comando diretto". Per esempio, per dire al computer di mostrare le righe di un programma che hai inserito, devi battere LIST (elenca) come comando diretto. Per cancellare quanto appare sullo schermo, sulla maggior parte dei computer devi battere CLS.

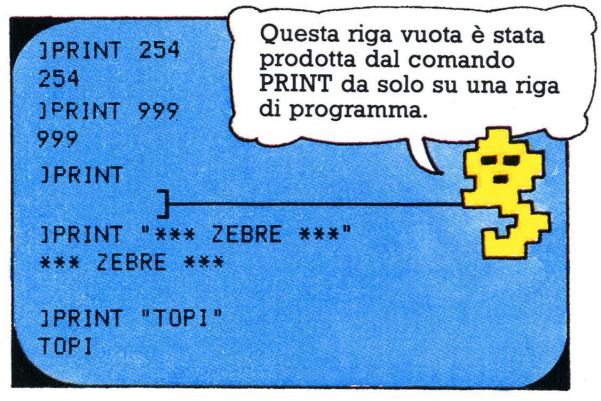
Quando batti i programmi devi stare molto attento, perché se scrivi male una parola del BASIC o una lettera, un numero o un segno di punteggiatura, il computer non è in grado di eseguire le istruzioni. Un errore in un programma viene chiamato "bug" (baco); la maggior parte dei bug sono errori di battitura, anche se talvolta si tratta di errori logici che possono portare a risultati sorprendenti.

GUIDA AL BASIC

Le pagine che seguono sono una guida che mostra i principali comandi del BASIC e come utilizzarli. Se hai un computer, è bene che tu controlli sul suo manuale perché alcuni comandi variano leggermente a seconda dei computer.

PRINT ▶

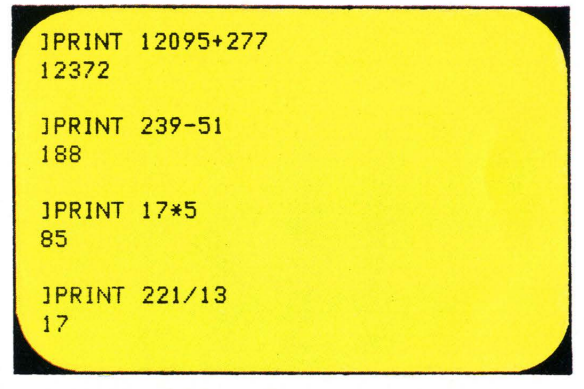
(Stampa) Ordina al computer di mostrare qualcosa sullo schermo. Le lettere e i simboli devono essere fra virgolette, mentre per i numeri non è necessario, come mostrano gli esempi qui a destra.
Tali esempi non contengono numeri di riga, quindi il computer esegue ogni istruzione non appena premi RETURN: mostrerà (o stamperà, come si dice abitualmente) sullo schermo esattamente quello che hai scritto fra virgolette, compresi eventuali spazi. Il comando PRINT da solo dice al computer di saltare una riga.

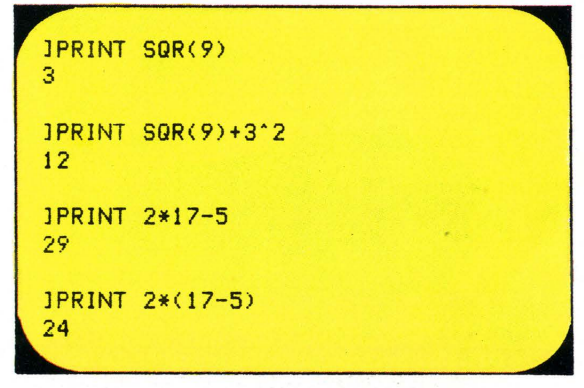


I calcoli ▶

Puoi servirti di PRINT anche per mostrare i risultati dei calcoli. Per l'addizione e la sottrazione il computer utilizza i segni normali, mentre usa * per la moltiplicazione e / per la divisione. SQR(N) è l'istruzione per trovare la radice quadrata di un numero N, mentre ↑, o ∧, o anche **, indica l'elevazione a potenza. Per esempio, 3 ↑ 2 significa 3 elevato a 2, o 3 al quadrato.

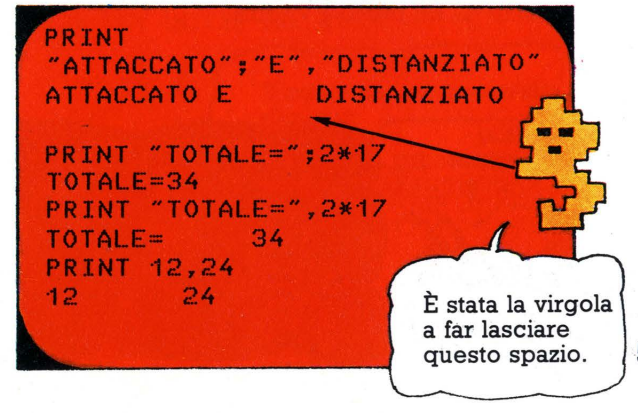
Nei calcoli con piú operazioni, le moltiplicazioni e le divisioni vengono sempre eseguite prima delle addizioni e delle sottrazioni. Perché l'ordine sia diverso puoi usare le parentesi; se ci sono molte parentesi le une dentro e altre, il computer inizia dalle piú interne.





Virgole e punti e virgola ▶

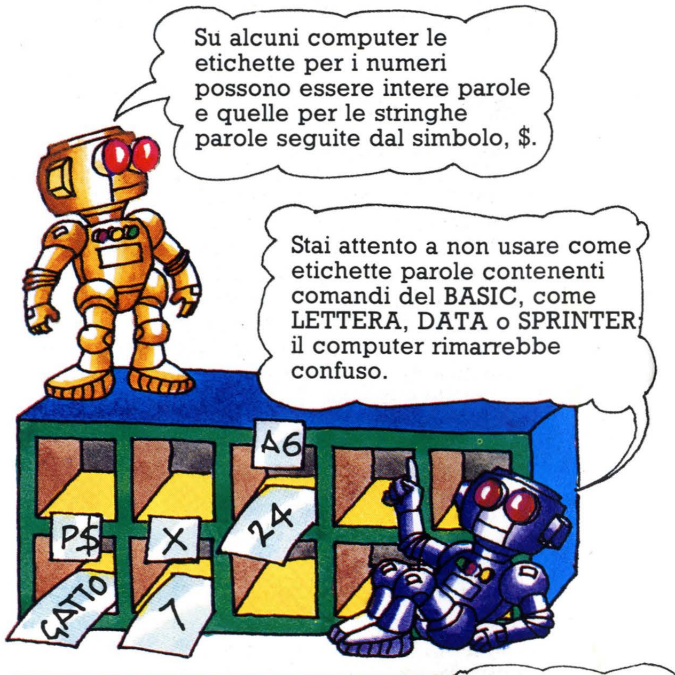
Determinano la posizione sullo schermo del carattere successivo: un punto e virgola dice al computer di non lasciare spazi, mentre una virgola gliene fa lasciare qualcuno (il numero varia a seconda dei computer).
Alcuni computer richiedono che una virgola o un punto e virgola separi i comandi PRINT dai relativi dati e variabili (lettere che rappresentano informazioni immagazzinate nella memoria). Prova gli esempi sulla destra per vederne il risultato sul tuo computer.



Variabili ▶

Le informazioni sulle quali il computer lavora sono i "dati". Quando fornisci al computer un dato, gli devi dare anche un'etichetta, chiamata "variabile". Quando vuoi che un dato venga sottoposto a una operazione, vi devi far riferimento tramite la variabile. "Variabile" deriva dal fatto che il dato cui si riferisce può variare durante l'esecuzione del programma.

Come etichette dei dati numerici si usano lettere dell'alfabeto, o una lettera piú un numero: es. A6. Un dato formato da lettere e simboli è una "stringa"; per etichettarla si usa una lettera o una lettera e un numero, seguiti dal simbolo del dollaro: esempio P$ (letto P dollaro, o P stringa), oppure P6$. Le regole sui nomi delle variabili sono diverse secondo i computer. Controlla sul tuo manuale.

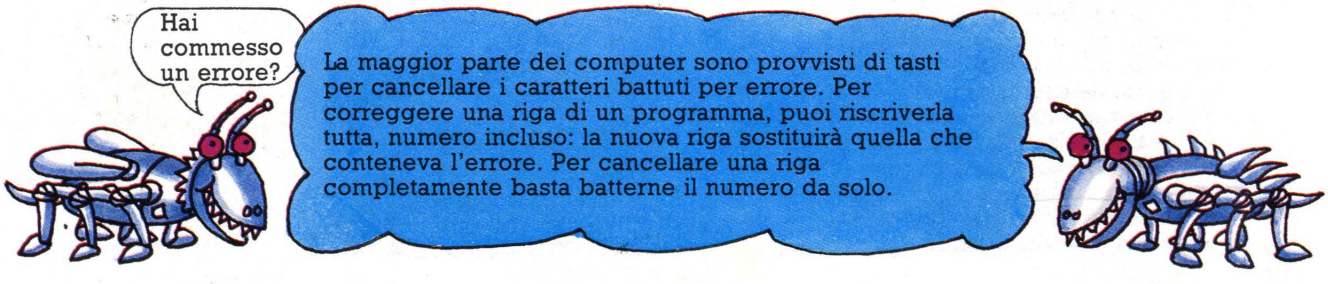


LET ▶

E' un modo per fornire dati al computer. LET A=5 dice al computer di inserire nella propria memoria il numero 5 e di attribuirgli l'etichetta A, mentre LET C$="CONIGLI" memorizza quella stringa di lettere in uno spazio di memoria contrassegnato da C$. Le stringhe devono sempre essere fra virgolette.

INPUT ▶

(Inserimento) Offre un metodo per fornire dati al computer durante l'esecuzione del programma. INPUT è seguita da un nome di variabile e quando il computer incontra un comando INPUT, stampa sullo schermo un punto interrogativo (o altro simbolo) e aspetta che gli si forniscano dati. Se la variabile che segue INPUT è di tipo numerico, devi fornire dati numerici; se è di tipo a stringa, devi inserire una stringa. Con la maggior parte dei computer puoi rendere l'istruzione INPUT piú chiara aggiungendo parole fra virgolette, (esempio a destra). Non usare questo metodo con il VIC, perché le parole fra virgolette verrebbero memorizzate con i dati inseriti. Quasi tutti i computer vogliono che le parole fra virgolette e il nome della variabile siano separati da un punto e virgola.



READ/DATA ▶

E' un altro metodo per fornire dati al computer. La parola READ è seguita da uno o piú nomi di variabile, mentre i dati per le variabili sono in una o piú righe che iniziano con DATA e che possono essere in qualsiasi punto del programma. Quando il computer incontra l'istruzione READ cerca una riga con la parola DATA poi inserisce i dati, consecutivamente, in ciascuna variabile. I dati devono essere separati da virgole e, su alcuni computer, le stringhe devono essere fra virgolette, mentre certi le richiedono solo se la stringa contiene spazi o punteggiature.

```
10 READ A$, B, C$, N
.
.
.
.

100 DATA RUOTE, 2000
110 DATA "ALBERO MOTORE", 500
```

Questo dato è fra virgolette perchè contiene uno spazio.

IF/THEN ▶

E' un metodo per sottoporre i dati a un esame e ordinare al computer di fare determinate cose secondo il risultato. Usando i simboli qui a destra, puoi vedere se due dati sono uguali, o se uno è maggiore o minore dell'altro. THEN può essere seguito praticamente da qualsiasi istruzione; se il risultato dell'esame è negativo (falso), il computer ignora il THEN e passa a eseguire il resto del programma.

```
IF A=B THEN PRINT "UGUALE"
IF X<>Y THEN PRINT "DIVERSO"
IF X>Y THEN PRINT "X PIU' GRANDE"
IF X<Y THEN PRINT "X PIU' PICCOLO"

IF A$="NO" THEN STOP
IF X+Y=5 THEN LET X=X+1
```

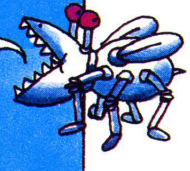
Questo dice al computer di interrompere l'esecuzione del programma.

GOTO ▶

Invia il computer a una determinata riga del programma. Di solito viene usato con IF/THEN affinché il computer effettui il salto solo se si verificano certe condizioni. Stai attento a usare GOTO da solo, perché può generare un loop continuo (si ha un loop quando una parte del programma viene ripetuta piú volte), come nella riga 185 a destra. Per interrompere l'esecuzione di questo programma battere il tasto BREAK o ESCAPE (secondo computer.)

```
100 IF A=5 THEN GOTO 175
.
.
.
.
.
.
.
175 PRINT "CONGRATULAZIONI"
180 PRINT "**** HAI VINTO ****"
185 GOTO 180
```

Stai attento a non creare loop senza fine, come nella riga 185.



GOSUB/RETURN ▶

GOSUB invia il computer a una subroutine, una parte del programma per l'esecuzione di un determinato compito. RETURN alla fine della subroutine rimanda il computer all'istruzione che segue GOSUB. Se dimentichi RETURN ottieni un messaggio di errore.

REM ▶

Il computer ignora le righe che cominciano con REM, il che consente di inserire nel programma osservazioni sulle operazioni eseguite.

```
100 INPUT "VUOI GIOCARE? ";T$
110 IF T$="SI'" THEN GOSUB 500
120 REM INIZIO DEL GIOCO
.
.
.
.

500 REM SUBROUTINE DI SALUTO
510 PRINT "COME TI CHIAMI?"
520 INPUT N$
530 PRINT "CIAO ";N$
540 RETURN
```

Il computer ignora questa riga.

La riga 540 rimanda il computer all'istruzione che segue GOSUB.

Loop FOR/NEXT ▶

Le parole FOR, TO e NEXT fanno sí che il computer ripeta un certo numero di volte una parte del programma. Nell'esempio a destra, le righe da 10 a 30 vengono ripetute tre volte e ogni volta il computer stampa il messaggio della riga 20. J è una variabile che conta il numero delle ripetizioni; la riga 30 rimanda il computer a trovare il valore successivo di J e ogni volta che il loop viene ripetuto, a J viene aggiunto 1. Quando J=3 il computer passa a eseguire il resto del programma.

```
10  FOR J = 1 TO 3
20  PRINT "CICLO J = ";J
30  NEXT J
40  PRINT

]RUN
CICLO J = 1
CICLO J = 2
CICLO J = 3
```

La variabile J alla fine della riga fa sí che il computer stampi il valore di J a ogni ripetizione del loop.

STEP ▶

(Passo) Modifica il modo in cui J conta il numero delle ripetizioni. Per esempio, FOR J=1 TO 10 STEP 2, fa sí che J aumenti ogni volta di 2, mentre STEP X lo farebbe aumentare del valore immagazzinato in X. Nell'esempio a destra, STEP —1 fa contare J all'indietro.

```
10 FOR J=10 TO 1 STEP -1
20 PRINT J;" GIORNI A NATALE"
30 NEXT J
40 PRINT "BUON NATALE"
RUN
10 GIORNI A NATALE
 9 GIORNI A NATALE
 8 GIORNI A NATALE
 7 GIORNI A NAT
```

Nido di loop ▶

Si possono ottenere ripetizioni molto complicate usando loop all'interno di altri loop; queste strutture sono i "nidi di loop". Nel programma a destra, ogni volta che viene ripetuto il loop dalla riga 10 alla 50, il loop dalla riga 10 alla 40 viene eseguito 12 volte. A ogni ripetizione del loop interno, la riga 30 stampa il valore di JxI.

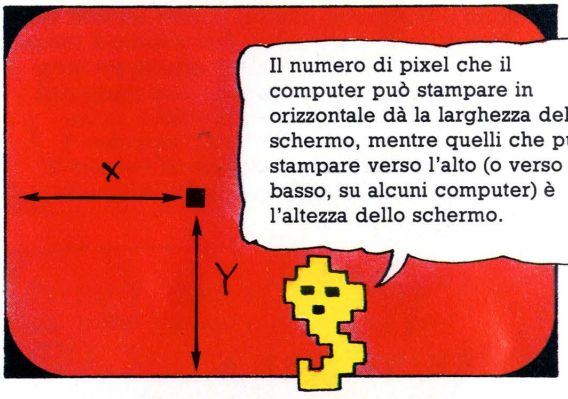
Entrambe le parti del loop interno devono essere contenute nel il loop esterno, altrimenti si ha un errore.

```
10  FOR I = 2 TO 12
20  FOR J = 1 TO 12
30  PRINT J;" PER ";I;"= ";J * I
40  NEXT J
50  NEXT I
```

Loop esterno

Comandi grafici ▶

Il computer traccia figure sullo schermo illuminando i pixel, in cui è suddiviso lo schermo. L'istruzione per accendere un quadratino (pixel) varia secondo i computer: in questo libro viene usata PLOT X,Y (traccia X,Y), dove X e Y sono le coordinate del pixel. Per tracciare una linea è stato usato DRAW X,Y (disegna X,Y). La maggior parte dei computer utilizza istruzioni di questo genere, anche se alcuni possono richiedere un'istruzione aggiuntiva per determinare il tipo di grafica desiderato.*



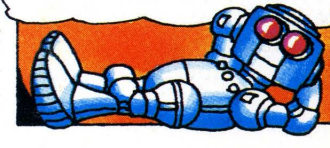
RND ▶

(Da *random*, casuale) Fa sí che il computer generi un numero casuale. La forma esatta dell'istruzione varia secondo i computers. Su alcuni RND(9) produce un numero fra 1 e 9; altri, richiedono un'istruzione come INT(RND (1) * 9+1). Il computer esegue prima le operazioni fra parentesi: RND (1) genera un numero fra 0 e 1, che viene poi moltiplicato per 9, il piú alto numero desiderato. Al risultato si aggiunge 1, perché la parola INT dà un numero intero arrotondato per difetto.

```
]PRINT RND(9)
.141352116

]PRINT INT(RND(1)*9+1)
2
```

Alcuni computer utilizzano RND(0) invece di RND(1).

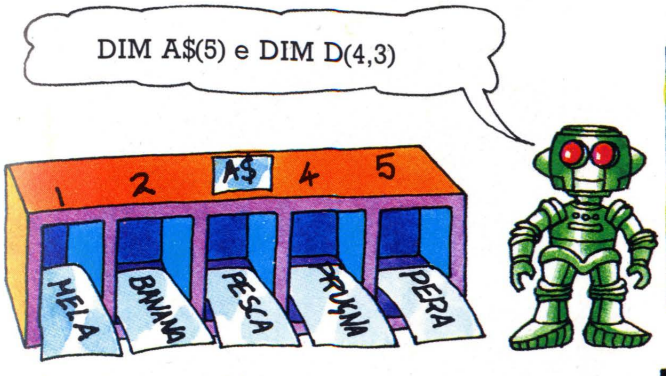
0.0109425*9+1=

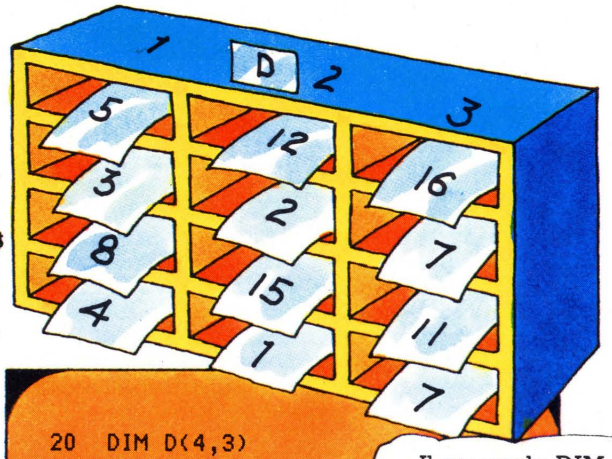


* La maggior parte dei computer dispone di piú "modalità", a seconda del numero di colori e di pixel con cui operano.

Matrici ▼

Una matrice è un insieme di dati raggruppati sotto un unico nome di variabile. La variabile può essere considerata come uno spazio nella memoria del computer suddiviso in molti scaffali. Le matrici possono essere monodimensionali, cioè con una sola fila di scaffali, o bidimensionali, con diverse file. Un elemento di una matrice monodimensionale viene richiamato tramite il numero dello scaffale in cui è, per esempio, nella figura sottostante A$(4) è SUSINA. Per le matrici bidimensionali, devi fornire il numero della fila e quello della colonna, per esempio D(3,2) è 15. I numeri fra parentesi sono chiamati "indici".





Prima di utilizzare una matrice, devi comunicarne le dimensioni al computer tramite la parola DIM (abbreviazione di "dimensioni"), come puoi vedere a destra. Per inserire i dati in una matrice puoi usare READ/DATA con un loop; per una matrice bidimensionale avrai bisogno di un nido di loop, come mostra la figura di destra. Nell'esempio, I è il numero di fila e J quello di colonna. Ogni volta che il loop interno viene ripetuto, viene inserito un dato nella successiva colonna della fila; quando viene ripetuto il loop I, il computer passa alla fila successiva.

```
20  DIM D(4,3)
30  FOR I = 1 TO 4
40  FOR J = 1 TO 3
50  READ D(I,J)
60  NEXT J
70  NEXT I
80  DATA  5,12,16
90  DATA  3,2,7
100  DATA  8,15,11
110  DATA  4,1,7
```

LEFT$ e RIGHT$ ▶

(Sinistra$ e destra$) Questi comandi permettono di operare sui caratteri all'interno delle variabili a stringa. Per esempio, LEFT$(A$,4) dice al computer di prendere quattro caratteri consecutivi di A$ a partire da sinistra, mentre RIGHT$(A$,5) fa prendere cinque caratteri da destra. I computer Sinclair non usano questi comandi; per le istruzioni con i Sinclair vedi pagina 11.

```
10 LET A$="PAPPAGALLO"
20 PRINT LEFT$(A$,4)
25 PRINT LEFT$(A$,2)
30 PRINT
40 PRINT RIGHT$(A$,5)
RUN
PAPP
PA

GALLO
```

MID$ e LEN$ ▶

(Mezzo$ e lun$, da lunghezza) MID$ dice al computer di prendere alcuni caratteri dall'interno di una stringa, mentre LEN$ comunica da quanti caratteri, compresi segni di punteggiatura e spazi, è composta una stringa. Per esempio, MID$(K$,2,4) significa: prendi quattro caratteri consecutivi di K$ a partire dal secondo. Vedi pagina 11 per le istruzioni da usare con i computer Sinclair.

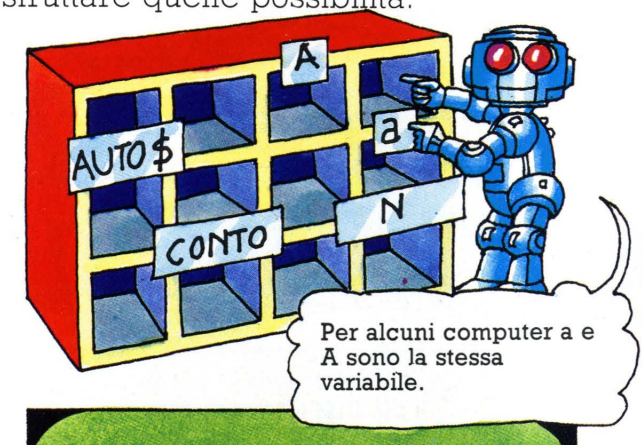
```
10 LET K$="TRAVESTIMENTO"
20 PRINT MID$(K$,2,4)
25 PRINT MID$(K$,4,4)
30 PRINT
35 PRINT LEN(K$)
RUN
RAVE
VEST

13
```

Il BASIC di questo libro

I programmi di questo libro sono scritti in BASIC "standard", ma certi computer talvolta seguono istruzioni un po' diverse, quindi, in quei casi, i programmi vanno leggermente modificati; a tale proposito, in queste due pagine, ci sono alcuni punti cui devi prestare particolare attenzione.
I programmi sono stati scritti per essere eseguibili su molti computer, perciò non tengono conto delle eventuali possibilità speciali delle varie macchine. Comunque, una volta che hai capito il funzionamento dei programmi, li puoi modificare per sfruttare quelle possibilità.

Nomi di variabile ▶

Su alcuni computer i nomi di variabile possono essere costituiti da parole, mentre altri accettano solo lettere o lettere e cifre. Per esempio, sui computer Sinclair puoi usare brevi parole per le variabili numeriche, mentre i nomi delle variabili a stringa devono essere di una sola lettera. Nei programmi di questo libro, per una maggiore chiarezza, quasi tutte le variabili sono contrassegnate da parole. Se il tuo computer non le accetta, usa solo la prima lettera come nome di variabile.



LET ▶

La maggior parte dei computer non richiedono il comando LET nelle istruzioni tipo LET FRUTTO$="MELA". Alcuni computer non hanno bisogno del THEN nelle istruzioni IF...THEN. I programmi di questo libro usano tutti sia LET che THEN, però puoi ometterli se sul tuo computer non sono necessari.

```
100 FRUTTO$="MELA"
110 CONTO=CONTO+1
120 IF CONTO=10 PRINT "PRONTO"
```

Inizializzazione delle variabili ▶

Su alcuni computer devi introdurre, o inizializzare, una variabile prima di poterla utilizzare. Questo significa che devi attribuire un valore alla variabile all'inizio del programma, come mostrato a destra. Altri assumono che il valore iniziale di una variabile numerica sia zero e che le variabili a stringa siano vuote, senza che tu le debba inizializzare. I programmi di questo libro comprendono righe di inizializzazione, che puoi omettere se il tuo computer non le richiede.

```
10 LET A=0
20 LET FRASE$=""
.
.
.
.
.
100 LET FRASE$=FRASE$+"K"
110 LET A=A+1
```

Un stringa vuota viene chiamata stringa nulla.

INPUT ▶

La maggior parte dei computer accettano parole fra virgolette insieme a un comando INPUT,* ma ci sono differenze per quanto riguarda la nacessità di un punto e virgola prima della variabile di INPUT e il lasciare automaticamente uno spazio fra le parole e i dati inseriti. Per scoprire come si comporta il tuo computer, fai esperimenti o consulta il manuale.

```
10 INPUT "COME TI CHIAMI ";N$
20 PRINT "CIAO ";N$
```

Con alcuni computer è necessario inserire uno spazio fra le virgolette, perché i dati non siano attaccati alle parole.

* Non usare questo metodo con il computer VIC, perché le parole verrebbero inserite nella variabile, insieme ai dati.

DATA

```
100 DATA TOPO, GOBLIN, SORCIO
110 DATA GNU, "ORSO BIANCO"
120 DATA "CERVO, ROSSO", "PANTERA, NERA", GIRAFFA
```

Virgolette per i dati che comprendono spazi e punteggiatura.

Stai particolarmente attento nell'inserire le righe con DATA: ciascun dato dev'essere separato dal successivo da una virgola ed è molto facile commettere errori. Alcuni computer richiedono che i dati costituiti da parole siano tra virgolette, mentre altri vogliono le virgolette solo se i dati comprendono spazi o segni di punteggiatura.

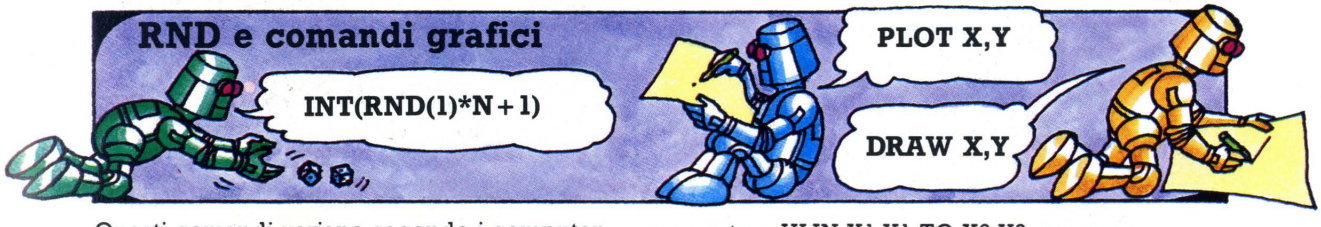
Righe con piú istruzioni

```
500 PLOT 40,1: DRAW 1,1
180 IF A=10 THEN PRINT "GIUSTO": GOTO 100
```

Questo avviene solo se A = 10.

Con la maggior parte dei computer è possibile inserire su un'unica riga piú istruzioni, separate da due punti, come mostrato sopra. Questo metodo occupa meno spazio nella memoria e può rendere i programmi piú leggibili. Se il tuo computer non accetta righe con piú istruzioni, scrivi ogni istruzione in una riga distinta. Se intendi inserire nei tuoi programmi righe con piú istruzioni, stai attento quando le metti dopo IF...THEN, perché verranno eseguite solo se la condizione di IF è vera.

RND e comandi grafici



Questi comandi variano secondo i computer. Nei programmi di questo libro, l'istruzione che genera un numero casuale fra 1 e N (dove N è un numero qualsiasi) è INT(RND(1)*N+1). I comandi grafici sono PLOT X,Y per tracciare un punto e HLIN X1,Y1 TO X2,Y2 per una linea. Queste istruzioni andranno sostituite da quelle del tuo computer e può anche darsi che tu debba aggiungere un'istruzione che specifichi il tipo grafica da utilizzare.

I computer Sinclair e le stringhe

I computer Sinclair gestiscono le stringhe in un modo non standard: non usano LEFT$ RIGHT$ o MID$ ma va loro detto esattamente quali caratteri della stringa devono prendere.
Per esempio, su un computer Sinclair PRINT A$ 1 TO 4) equivale a PRINT LEFT$ A$,4) e PRINT A$(4 TO 8) equivale a MID$(A$,4,5). Nelle matrici di stringhe, ogni stringa deve avere lo stesso numero di caratteri (alle piú corte puoi aggiungere spazi) e ogni carattere della stringa viene memorizzato in un compartimento separato della matrice.

(Conversione programmi per Sinclair a pagg. 46-47).

Modifiche ai programmi

Una volta che hai visto che un programma può essere eseguito sul tuo computer e che capisci come funziona, lo puoi modificare inserendo dati diversi o aggiungendo suono e colore.

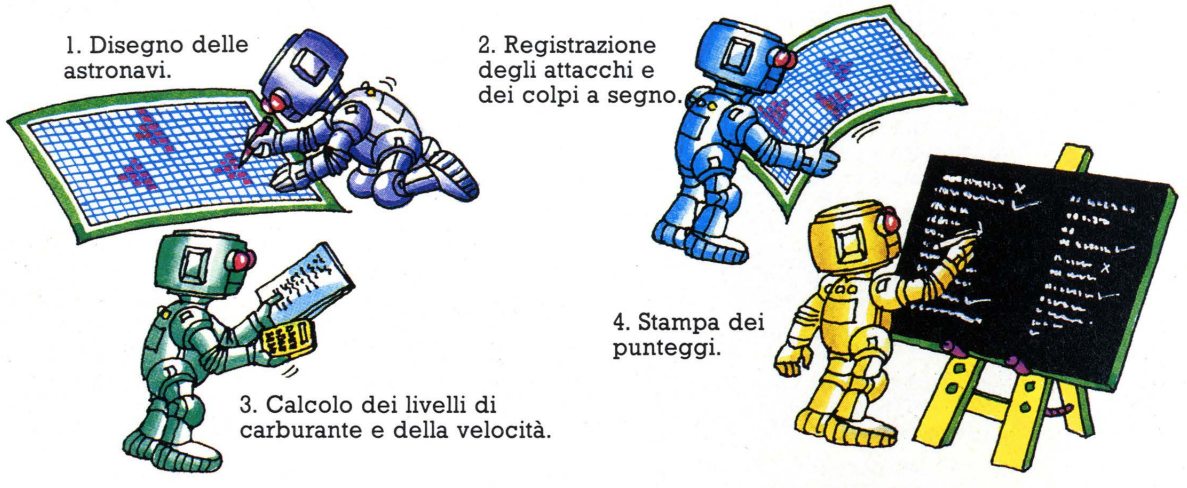
Quando modifichi un programma, controlla attentamente ogni riga. Guarda che ci siano abbastanza loop, ma non troppi, per leggere i nuovi dati, e ricorda anche di modificare le istruzioni DIM.

Imparare il Basic studiando i programmi

Un buon metodo per imparare il BASIC consiste nello studiare i programmi scritti da altri e vedere come funzionano. Studiando i programmi di questo libro puoi imparare a usare i loop e le stringhe, a scrivere semplici programmi di grafica e a ordinare dati. A prima vista alcuni programmi sembrano davvero complicati, ma un programma complicato non è che una lunga serie di comandi in BASIC disposti in modo ordinato. Queste due pagine contengono alcune osservazioni e suggerimenti per aiutarti a studiare e a capire i programmi.

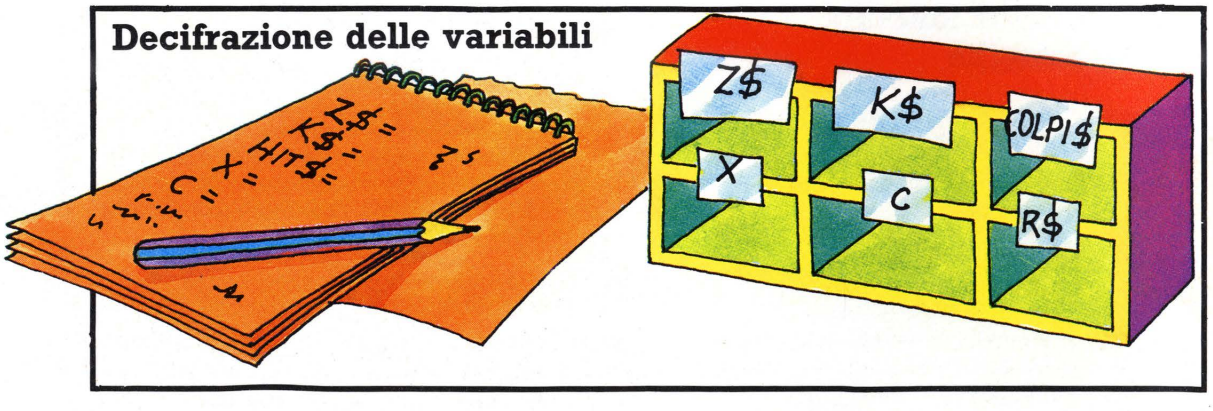
Studio di un programma

La maggior parte dei programmi sono costituiti da piú parti (talvolta chiamate routine o moduli) per l'esecuzione dei vari compiti. Per esempio, in un programma per un gioco in cui vi siano astronavi, una parte del programma disegnerà le astronavi sullo schermo, mentre altre registreranno gli attacchi e i colpi andati a segno, controlleranno i livelli del carburante e la velocità e stamperanno i punteggi finali.



Il primo stadio nello studio di un programma consiste nel cercar di riconoscerne le diverse parti e di capire a cosa servono: questo fornisce un'idea generale sul suo funzionamento. Guarda se ci sono subroutine per lo svolgimento di operazioni specifiche e grossi salti nei numeri di riga: righe con numeri di centinaia o di migliaia spesso indicano una nuova parte del programma. Talvolta le diverse parti di un programma sono contraddistinte da istruzioni REM.

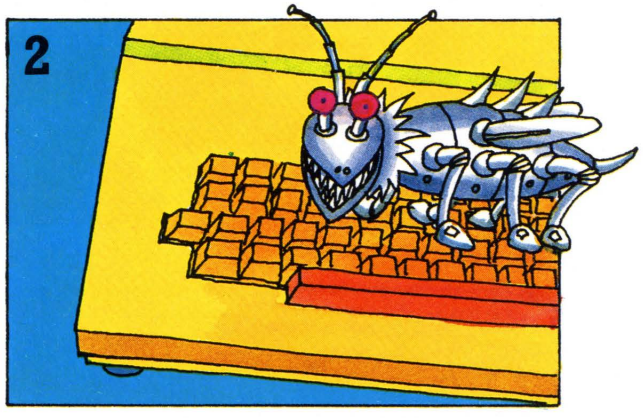
Decifrazione delle variabili



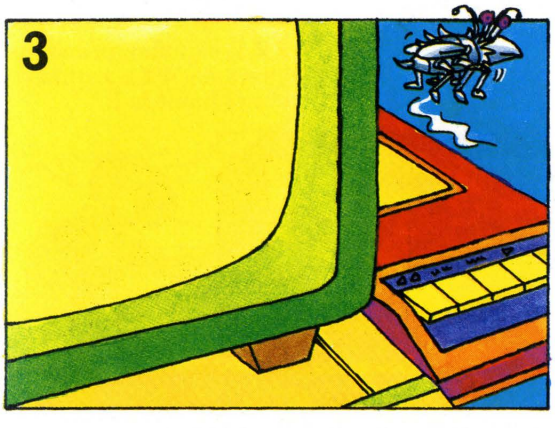
Probabilmente la cosa piú difficile da capire in un programma è la funzione delle variabili. Prima di inserire un programma in un computer, conviene analizzare il ruolo di ciascuna variabile prendendo appunti. Spesso alcune variabili vengono usate per gli stessi compiti, cosí le puoi riconoscere immediatamente. Per esempio, di solito le lettere I, J, K e L vengono usate per i loop, mentre Z e Z$ vengono usate per dati necessari solo per un breve periodo.

Lancio e correzione degli errori dei programmi

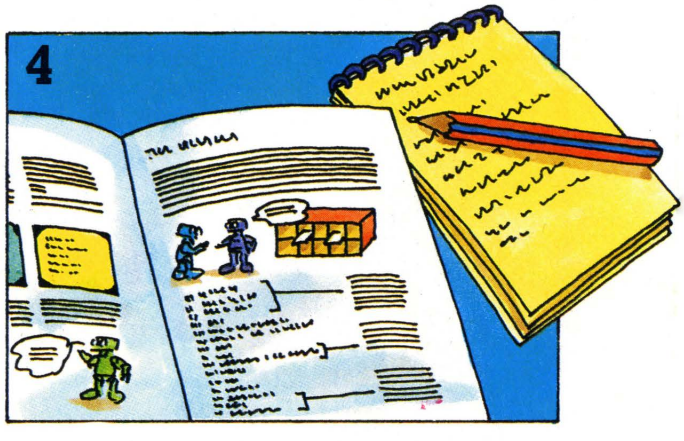
Dopo aver determinato a cosa servono le variabili, inserisci il programma nel computer. Poiché i programmi sono scritti in BASIC standard, può darsi che alcuni comandi vadano modificati per adattarli al tuo computer.



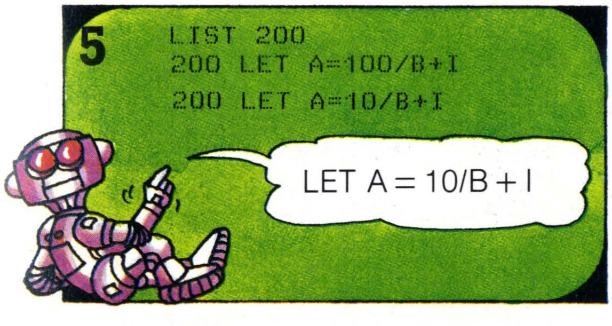
Fatto questo, prova a lanciare il programma. Probabilmente ci sarà qualche bug, quindi stampa il listato del programma sullo schermo e guarda se ci sono errori di battitura o comandi che il tuo computer non capisce.



Dopo aver trovato tutti i bug, lancia il programma qualche volta per vedere come funziona. A questo punto conviene memorizzarlo su cassetta o su disco, in modo da non doverlo piú ribattere.



Adesso ritorna al listato ed esamina ogni riga, cercando di capire qual'è il suo compito. Cerca brevi routine che possano essere poi incorporate nei tuoi programmi.



Puoi servirti del computer per capire come funziona il programma: prova a modificare il valore di una variabile e guarda che conseguenze ha sul programma. Apporta solo un piccolo cambiamento per volta, in modo da poterne determinare gli effetti. Alla fine, ricordati di reinserire i valori giusti.

Puoi anche inserire righe per stampare i valori delle variabili, cosí da vedere come cambiano durante l'esecuzione del programma. Talvolta è utile inserire comandi STOP, per poter esaminare il programma a spezzoni, però dopo ricordati di cancellarli. Alcuni computer dispongono del comando CONTINUE (continua), utilizzabile dopo STOP.

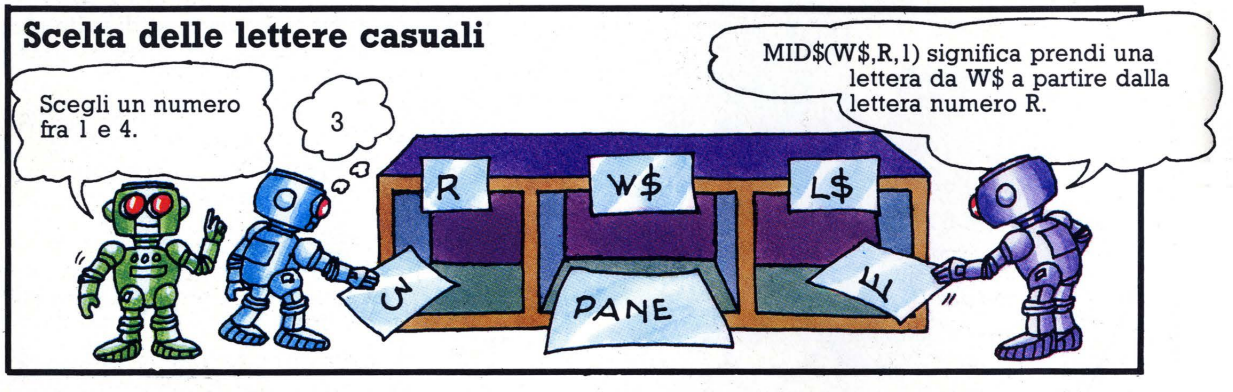
Uso delle stringhe

Questo programma mostra come sia possibile far svolgere al computer operazioni complicate combinando semplici comandi del BASIC. Il programma è di un gioco di individuazione di parole nel quale il computer ti chiede una parola, poi stampa le lettere sullo schermo in modo casuale e ti domanda quante volte compare la parola. Vengono utilizzati i comandi di gestione delle stringhe: MID$, RIGHT$ e LEFT$, e i numeri casuali.* Le principali operazioni che il programma deve effettuare sono due: far stampare le lettere in ordine casuale sullo schermo e far contare il numero delle volte che la parola è stata scritta correttamente.

```
TROVA LA PAROLA
PER FAVORE SCRIVI UNA PAROLA
BREVE
?PANE
ORA GUARDIAMO SE RIESCI A
TROVARE LA TUA PAROLA MENTRE
LE LETTERE
SCORRONO SULLO SCHERMO
BATTI RETURN PER INIZIARE
```

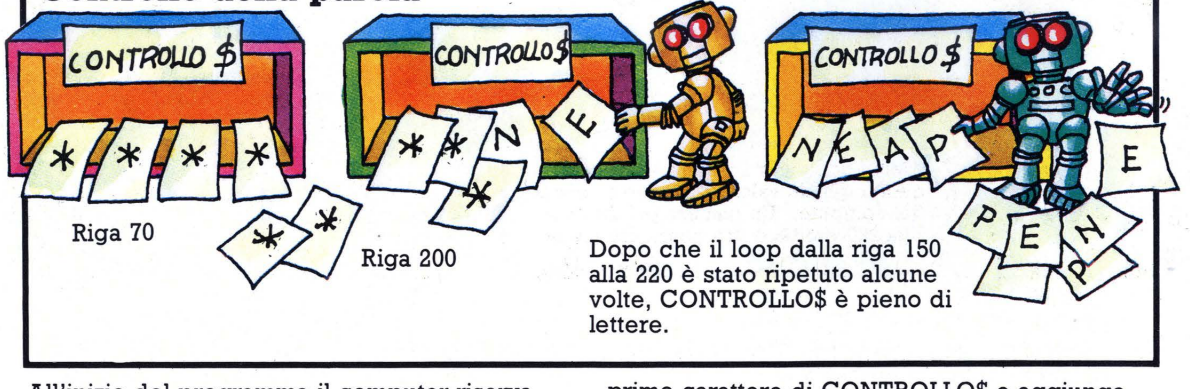
```
PNENAEPAANPNAEP
ANEPEANPEANPANE
APAEEPNNPPAEPAN
SCRIVI QUANTE VOLTE PENSI CHE
LA TUA PAROLA SIA APPARSA
SULLO SCHERMO ?1
SBAGLIATO!
LA TUA PAROLA E' APPARSA 2
VOLTE
```

Scelta delle lettere casuali



Il programma usa MID$ con un numero casuale per scegliere la lettera da stampare. La parola è memorizzata in W$. Alla riga 160 prende un numero casuale fra 1 e la lunghezza della parola e lo inserisce in R, poi nella riga 170 utilizza il numero in R per decidere quale lettera prendere da W$: immagazzina la lettera in L$ e la stampa sullo schermo con la riga 180. A ogni ripetizione del loop dalla riga 150 alla 220, in R viene immagazzinato un nuovo numero e viene scelta una nuova lettera di W$.

Controllo della parola



All'inizio del programma il computer riserva una zona della memoria chiamata CONTROLLO$ e la riempie con tanti asterischi quante sono le lettere della parola. Ogni volta che prende una nuova lettera casuale, toglie il primo carattere di CONTROLLO$ e aggiunge la lettera casuale alla fine della stringa (riga 200). Nella riga 210 confronta CONTROLLO$ con W$ e se le lettere sono nello stesso ordine aggiunge 1 a N.

* Per modificare il programma per i computer Sinclair (Timex) vedi pagina 46.

Gioco di individuazione delle parole

```
10  HOME
20  PRINT "TROVA LA PAROLA": PRINT
30  LET CHECK$=""
40  LET N=0
45  PRINT "PER FAVORE SCRIVI UNA PAROLA
    BREVE"
50  INPUT W$
60  FOR I=1 TO  LEN (W$)
70  LET CHECK$=CHECK$+"*"
80  NEXT I
90  PRINT
100 PRINT "ORA GUARDIAMO SE RIESCI A
    TROVARE"
110 PRINT "LA TUA PAROLA MENTRE LE
    LETTERE"
115 PRINT "SCORRONO SULLO SCHERMO"
120 INPUT "BATTI RETURN PER
    INIZIARE";Z$
```

Usa il comando del tuo computer per cancellare lo schermo.

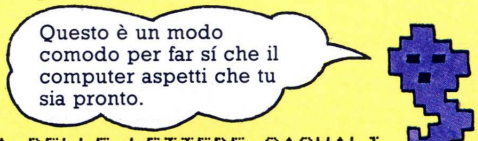
Questa è una riga con due istruzioni, separate da due punti.

Introduce variabili vuote che verranno utilizzate in seguito.

Ti chiede la parola e la inserisce in W$.

Loop che viene eseguito un numero di volte pari a quello delle lettere nella parola, cioè LEN(W$). A ogni ripetizione del loop, viene inserito un * in CHECK$.

La riga 120 ordina al computer di aspettare che tu batta qualcosa. Sulla maggior parte dei computer basta che tu batta RETURN; sull'Oric batti un tasto, poi premi RETURN.




```
130 HOME
140 REM  SCELTA DELLE LETTERE CASUALI
150 FOR I=1 TO 50* LEN (W$)
160 LET R= INT ( RND (1)* LEN (W$)+1)
170 LET L$= MID$ (W$,R,1)
```

Crea un loop dalla riga 150 alla 220 da ripetersi un numero di volte pari a 50 per il numero delle lettere nella tua parola.

Sceglie un numero casuale fra 1 e la lunghezza della parola e lo mette in R.

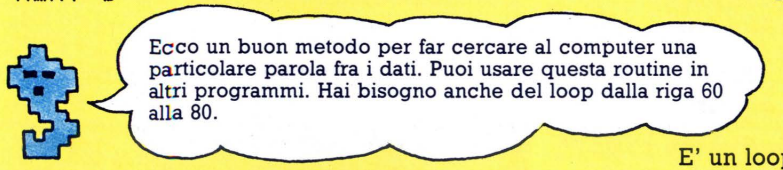
Usa il numero in R per scegliere una lettera di W$ e la immagazzina in L$.


```
180 PRINT L$+" ";
190 REM  CONTROLLO DELLA PAROLA
200 LET CHECK$= RIGHT$ (CHECK$, LEN
    (W$)-1)+L$
210 IF CHECK$=W$ THEN  LET N=N+1
220 NEXT I
```

Stampa la lettera in L$ seguita da uno spazio. Il punto e virgola fa sí che il computer stampi tutte le lettere sulla stessa riga.

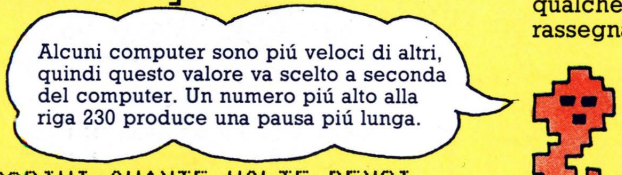
Significa: prendi LEN(W$)-1 lettere dalla destra di CHECK$, aggiungi la lettera in L$ poi rimetti il nuovo gruppo di lettere in CHECK$.

N tiene il conto del numero di volte che una parola compare correttamente




```
230 FOR I=1 TO 1000
240 REM  NON FA NIENTE
250 NEXT I
```

E' un loop di "rallentamento". Non ci sono istruzioni da eseguire, ma fa sí che il computer faccia una pausa di qualche secondo mentre passa in rassegna tutti i valori di I.




```
260 HOME
265 PRINT "SCRIVI QUANTE VOLTE PENSI
    CHE LA "
270 PRINT "TUA PAROLA SIA APPARSA SULLO SCHERMO"
275 INPUT G
280 PRINT
290 IF G=N THEN  PRINT "GIUSTO!"
300 IF G<>N THEN  PRINT "SBAGLIATO!"
310 PRINT "LA TUA PAROLA E' APPARSA
    ";N;" VOLTE"
```

Memorizza la tua risposta in G.

Confronta G con N (la variabile che il computer ha utilizzato per contare il numero delle parole corrette).

Loop e numeri casuali

Questo programma è un gioco spaziale che mette alla prova le tue capacità di calcolare a mente. Mostra alcuni dei modi in cui puoi usare i loop e i numeri casuali e contiene alcuni effetti speciali sullo schermo, che puoi incorporare nei tuoi programmi. Si tratta di un programma piuttosto lungo, ma la maggior parte delle righe contengono istruzioni PRINT per costruire la scenografia del gioco.

È IL COMPUTER DELLA NAVE CHE PARLA... SIAMO IN DIFFICOLTÀ. NON RIESCO A CALCOLARE IL FLUSSO DI CARBURANTE. NELL'ACCOSTARE ALLA TERRA DOVRAI EFFETTUARE I CALCOLI TU. POSSO DIRTI DI QUANTO CARBURANTE ABBIAMO BISOGNO A OGNI STADIO, E IL PERIODO IN CUI DEV'ESSERE UTILIZZATO. DEVI DIVIDERE IL CARBURANTE PER IL TEMPO COSÍ DA FORNIRMI LA MEDIA A CUI LA NAVE DEVE CONSUMARE IL CARBURANTE.

Gioco di emergenza spaziale

```
100  HOME
110  FOR I = 1 TO 20
115  PRINT "***** ATTENZIONE *****";
118  PRINT "*** ALLARME ROSSO *****";
120  FOR J = 1 TO 10
125  REM  NON FA NIENTE
130  NEXT J
135  NEXT I
140  HOME

150  FOR I = 1 TO 20
155  PRINT "***** CIRCUITO DANNEGGIATO
     *****";
160  FOR J = 1 TO 10
165  REM  NON FA NIENTE
170  NEXT J
180  NEXT I
190  HOME
200  PRINT "E' IL COMPUTER DELLA NAVE
     CHE PARLA..."
210  PRINT
220  PRINT "SIAMO IN DIFFICOLTA'. NON
     RIESCO A CALCOLARE IL FLUSSO DI
     CARBURANTE"
230  PRINT
240  PRINT "NELL'ACCOSTAMENTO ALLA
     TERRA DOVRAI FARE I CALCOLI TU."
250  PRINT
260  PRINT "POSSO DIRTI DI QUANTO
      CARBURANTE ABBIAMO BISOGNO A
      OGNI STADIO, E IL PERIODO IN
      CUI DEV'ESSERE UTILIZZATO."
270  PRINT
280  PRINT "DEVI DIVIDERE IL
     CARBURANTE PER IL TEMPO COSI'
     DA FORNIRMI LA MEDIA A CUI LA
     NAVE DEVE CONSUMARE IL
     CARBURANTE."
290  PRINT
300  PRINT "ECCO UN ESEMPIO"
310  PRINT "----------"
320  PRINT "CARBURANTE=24"
330  PRINT "TEMPO=6"
345  PRINT
350  INPUT "PER FAVORE DIVIDI IL
     CARBURANTE PER IL TEMPO E BATTI
     RAPIDAMENTE LA RISPOSTA ";RISP

360  PRINT
370  IF RISP <  > 4 THEN  PRINT "NO.
     PROVA DI NUOVO, NE VA DELLA TUA
     VITA": GOTO 350
380  HOME
```

Scegli valori del loop di rallentamento adatti al tuo computer.

Usa il comando del tuo computer per cancellare lo schermo.

Le righe da 110 a 135 formano un nido di loop. Ogni volta che viene eseguito il loop I, viene stampato l'avvertimento delle righe 115-118, poi viene ripetuto dieci volte il loop J. Il loop J è un loop di rallentamento che fa compiere una pausa al computer per darti il tempo di leggere l'avvertimento.

Le righe da 150 a 180 funzionano come quelle da 110 a 135.

La parte successiva del programma stampa sullo schermo una descrizione del gioco. Se le frasi sono troppo lunghe per il tuo schermo, inserisci altre righe con PRINT.

PRINT da solo fa saltare righe.

Se il tuo computer fa scorrere il testo prima che tu abbia fatto in tempo a leggerlo, cancella parte delle righe che contengono solo PRINT.

Sottolinea le parole della riga precedente.

La tua risposta viene memorizzata in RISP.

Il GOTO viene eseguito solo se RISP è diversa da 4. (Se il tuo computer non consente le righe con più istruzioni, ripeti IF...THEN con un GOTO su un'altra riga.)


```
390  PRINT "BENE. ORA DEVI DARE TUTTE
   LE RISPOSTE GIUSTE"
395  PRINT "ALTRIMENTI LA NAVE
   RIMARRA' DANNEGGIATA"
400  PRINT "SE COMMETTI PIU' DI DUE
   ERRORI,"
405  PRINT "SAREMO TUTTI DISTRUTTI"
410  PRINT
420  INPUT "BATTI RETURN PER
   COMUNICARMI QUANDO SEI PRONTO";Z$
430  HOME
440  PRINT "ATTENTO - PARTIAMO!"
450  LET DANNO = 0

460  FOR CARBURANTE = 720 TO 120 STEP
   - 120

470  LET T =  INT ( RND (1) * 5 + 2)



480  PRINT
490  PRINT "CARBURANTE=";CARBURANTE
500  PRINT "TEMPO=";T
510  PRINT
520  LET R = CARBURANTE / T
530  INPUT "ORA DAMMI LA MEDIA ";RISP
540  IF RISP = R THEN  GOTO 600
550  LET DANNO = DANNO + 1
560  PRINT
570  PRINT "**** DANNO ****"
580  PRINT
590  IF DANNO > 2 THEN  GOTO 640
600  NEXT CARBURANTE
610  HOME
620  PRINT "CONGRATULAZIONI - SEI
   STATO BRAVO QUANTO ME. SEI
   ATTERRATO SENZA DANNI!"
630  GOTO 720
640  HOME
650  FOR I = 1 TO 20
660  PRINT "*";
670  FOR J = 1 TO  INT ( RND (1) * I +
   50)
680  PRINT "    ";
690  NEXT J
700  NEXT I
710  PRINT : PRINT "LA NAVE E'
   DISTRUTTA"
720  END
```

Aspetta che tu batta RETURN. (Con il computer Oric, batti un tasto, poi premi RETURN.)

Introduce una variabile chiamata DANNO.

CARBURANTE è una variabile per contare le ripetizioni dei loop; viene anche usata nei calcoli. Alla riga 460 CARBURANTE=720, dopo di che diminuisce di 120 a ogni ripetizione. Questi valori sono stati scelti in modo che la risposta alla somma nella riga 520 sia sempre un numero intero.

Genera numeri casuali fra 2 e 6, per i quali il valore di CARBURANTE è divisibile esattamente.

Il computer esegue una somma con i valori di CARBURANTE e T e memorizza la risposta in R. Se la tua risposta è giusta, il computer va alla riga 600.

La variabile DANNO tiene il conto dei tuoi errori.

Se commetti piú di due errori il computer esce dal loop e passa alla riga 640.

Questa riga viene stampata solo se commetti meno di due errori.

Questo nido di loop traccia sullo schermo un gruppo casuale di asterischi. A ogni ripetizione del loop I, il computer stampa un asterisco, dopo di che il loop J fa lasciare un numero casuale di spazi.

Un database calcistico

Un database (leggi: databeis) è costituito da molti dati immagazzinati in un computer e disposti in modo che il computer possa raggrupparli e confrontarli, affinché una persona che utilizza il database possa ottenere informazioni utili in un tempo molto breve. Le pagine che seguono presentano un programma per un database relativo alla Coppa del Mondo di calcio. È un piccolo database tramite il quale puoi trovare quale squadra ha vinto la Coppa del Mondo in qualsiasi anno a partire dal 1930, o in quale anno una squadra ha vinto la Coppa. Alla fine del programma ci sono alcune idee per trasformare il database per inserirvi informazioni diverse, come l'indice di una rivista o i dati per un'indagine naturalistica.

Un programma di database è composto da tre parti principali: un modo adeguato per immagazzinare le informazioni, uno per ricuperarle e un "menú". Un menú è un elenco delle varie operazioni che un programma può eseguire, per la scelta di quella desiderata. Il programma dovrebbe inoltre essere "user-friendly" (amichevole verso l'utente), cioè in grado di fornire alla persona che lo usa istruzioni chiare, senza interrompersi bruscamente se viene commesso un errore.

Esempi di utilizzazioni del database

```
PER FAVORE SCRIVI IL NOME DELLA
SQUADRA, O BATTI 'MENU' PER
VEDERE NUOVAMENTE L'ELENCO
GERMANIA OVEST

GERMANIA OVEST
HA VINTO LA COPPA DEL MONDO NEL
1954 FINALISTA NEL 1966
HA VINTO LA COPPA DEL MONDO NEL
1974 FINALISTA NEL 1982

BATTI RETURN PER IL MENU'
```

```
PER FAVORE SCRIVI L'ANNO,
O BATTI
MENU' PER VEDERE  NUOVAMENTE
L'ELENCO 1938

NEL 1938 ITALIA
HA VINTO LA COPPA

BATTI RETURN PER IL MENU'
```

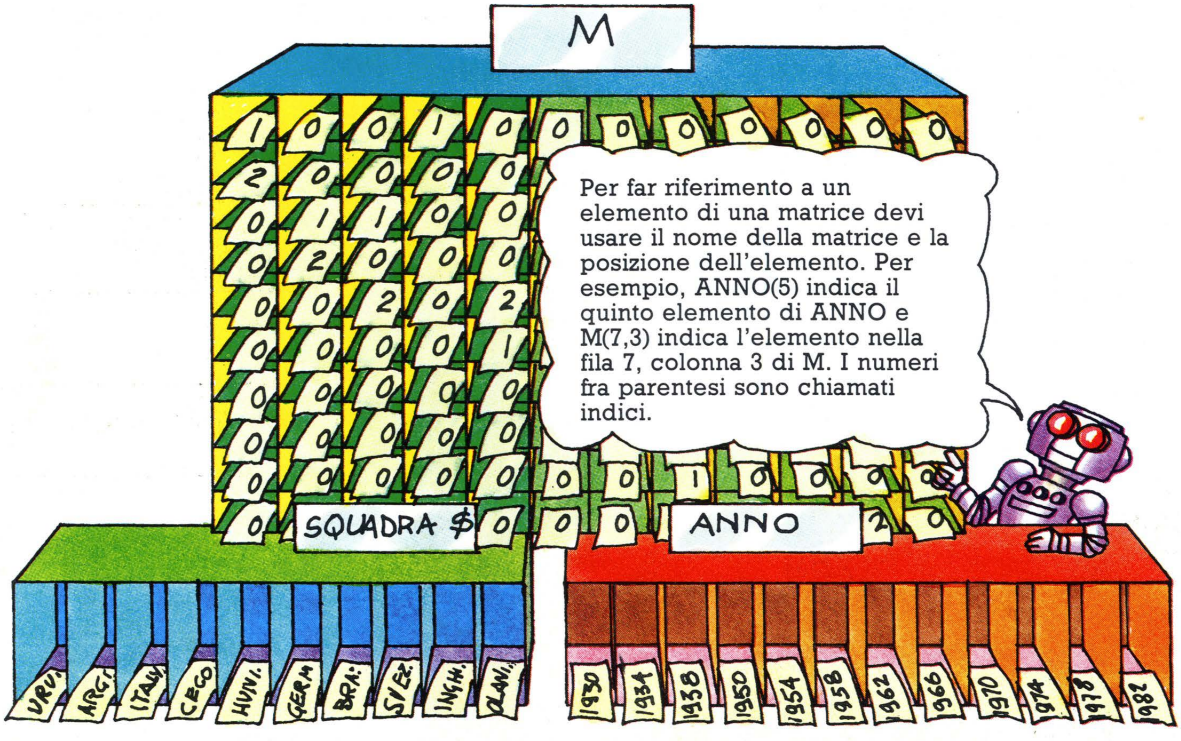
Memorizzazione delle informazioni

	1930	1934	1938	1950	1954	1958	1962	1966	1970	1974	1978	1982
URUGUAJ	1	0	0	1	0	0	0	0	0	0	0	0
ARGENTINA	2	0	0	0	0	0	0	0	0	0	1	0
ITALIA	0	1	1	0	0	0	0	0	2	0	0	1
CECOSLOVACCHIA	0	2	0	0	0	0	2	0	0	0	0	0
UNGHERIA	0	0	2	0	2	0	0	0	0	0	0	0
GERMANIA OVEST	0	0	0	0	1	0	0	2	0	1	0	2
BRASILE	0	0	0	0	0	1	1	0	1	0	0	0
SVEZIA	0	0	0	0	0	2	0	0	0	0	0	0
INGHILTERRA	0	0	0	0	0	0	0	1	0	0	0	0
OLANDA	0	0	0	0	0	0	0	0	0	2	2	0

Per mettere in collegamento fra loro squadre e anni, il programma utilizza una tabella e cerca una squadra o un anno proprio come faresti tu. La cifra 1 indica che la squadra ha vinto la Coppa, mentre un 2 indica che la squadra è stata finalista. Scorrendo lungo le righe e le colonne, puoi vedere in che anno una squadra ha vinto o è stata finalista. Il programma esegue queste operazioni automaticamente e, naturalmente, se i dati sono molti, è molto piú rapido di una persona.

Costruzione della tabella

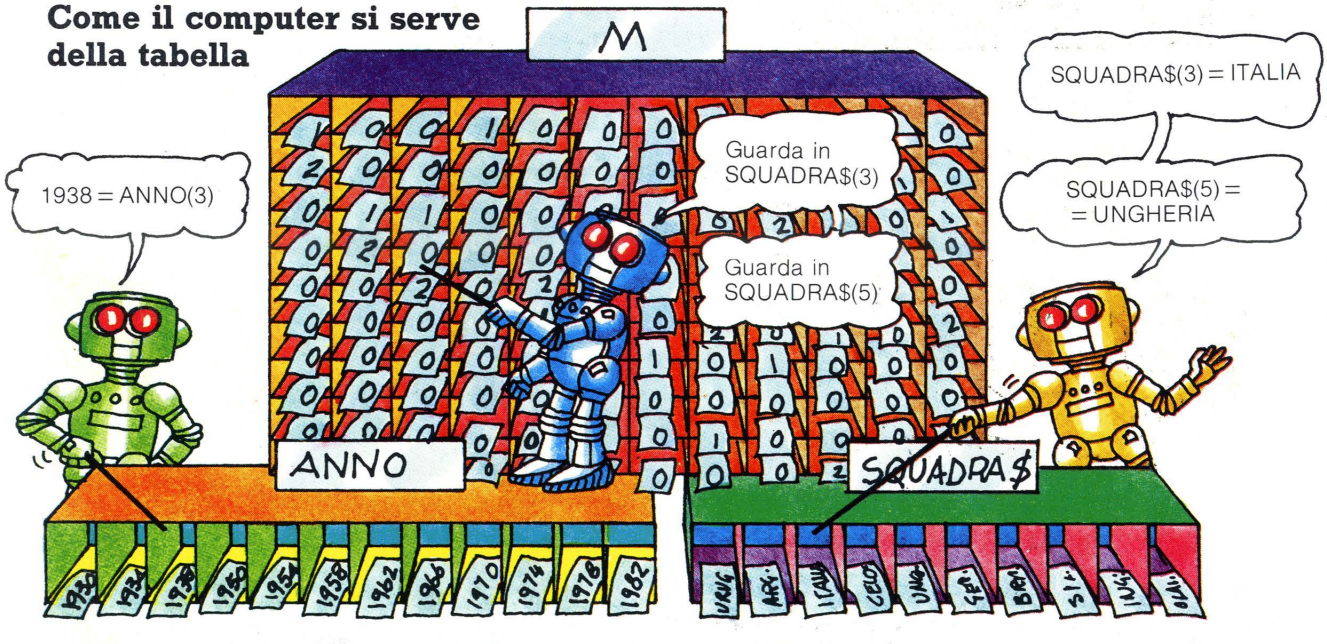
Usando le matrici, è molto facile costruire la versione su computer della tabella della pagina accanto. Una matrice è una variabile in grado di contenere una quantità di dati distinti.



Occorrono due matrici monodimensionali: una con 12 scompartimenti per contenere l'elenco degli anni e uno con 10 per le squadre. Nel programma vengono chiamate ANNO e SQUADRE$.

Per contenere tutti i dati della tavola ci vuole una matrice bidimensionale, con 10 file e 12 colonne, che nel programma viene chiamata M (per matrice).

Come il computer si serve della tabella



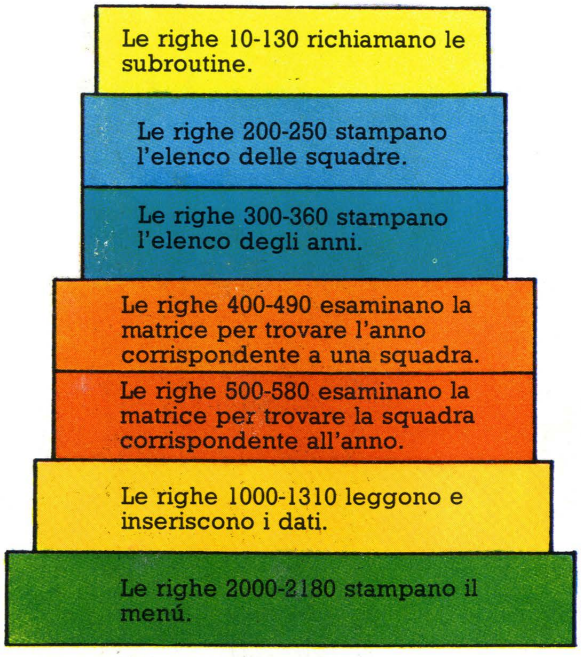
Per trovare quale squadra ha vinto la Coppa del Mondo per esempio, nel 1938, il computer cerca 1938 nella matrice ANNO e nota che è nello scompartimento 3. Allora guarda nella colonna 3 di M e quando trova un 1 o un 2 nota il numero di riga corrispondente e lo utilizza per trovare il nome della squadra in SQUADRA$.

Il programma del database

Il programma ha sette parti principali, ognuna delle quali fa capo a una subroutine distinta. Le prime righe dicono al computer quale subroutine utilizzare e il computer vi torna dopo aver eseguito la subroutine.

Le subroutine che iniziano alle righe 200 e 300 servono a stampare l'elenco delle squadre e gli anni. Le righe 400-500 servono a trovare in quale anno una determinata squadra ha vinto la Coppa, mentre le righe 500-600 trovano quale squadra ha vinto la Coppa in un determinato anno. Tutti i dati sono elencati verso la fine del programma, seguiti dal menú.

Di solito è meglio mettere i dati verso la fine del programma e la parte operativa del programma all'inizio.

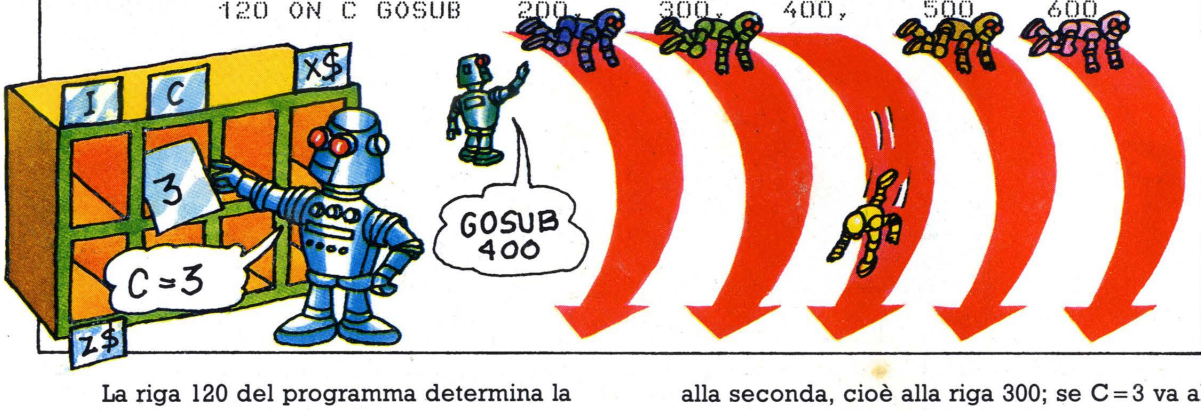


Il menú

```
                MENU'
              ********

PER VEDERE L'ELENCO DELLE SQUADRE
BATTI 1
PER VEDERE L'ELENCO DEGLI ANNI
BATTI 2
PER VEDERE IL NOME DI UNA SQUADRA
BATTI 3
PER INSERIRE UN ANNO BATTI 4
PER TERMINARE BATTI 5
BATTI IL NUMERO CHE HAI SCELTO
POI PREMI RETURN
```

Il menú è la parte del programma che ti dice cosa può fare e come utilizzarlo. In questo caso, per scegliere devi battere un numero. Il numero è memorizzato nella variabile C e il computer lo utilizza per richiamare la subroutine giusta per svolgere il compito desiderato.

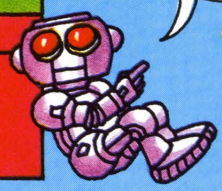
Richiamo delle subroutine



La riga 120 del programma determina la subroutine che il computer deve utilizzare. La lettera C è la variabile che contiene il numero che hai battuto dopo aver visto il menú. Il computer usa il numero in C per decidere a quale subroutine andare: se C=1 va alla prima subroutine dell'elenco della riga 120, cioè a quella che comincia alla riga 200. Se C=2 va alla seconda, cioè alla riga 300; se C=3 va alla terza ecc. ON GOSUB è un utile comando del BASIC che invia il computer a diverse subroutine a seconda del risultato di un esame. Se sul tuo computer il comando ON non è disponibile, puoi usare una serie di istruzioni IF...THEN, per esempio, IF C=1 THEN GOSUB 200.

A cosa servono le variabili

ANNO	SQUADRA$	M
La matrice in cui sono inseriti gli anni.	La matrice in cui sono inseriti i nomi delle squadre.	La matrice bidimensionale che contiene i dati.
C	**Z$**	**X$**
La variabile in cui viene inserito il numero che batti dopo aver visto il menú.	Il nome della squadra o l'anno che hai scelto.	Dati temporanei.



Se sul tuo computer i nomi delle matrici non possono essere costituiti da parole, usa solo le lettere iniziali.

Il programma

```
 10  DIM SQUADRA$(10): DIM ANNO(12): DIM
    M(10,12)

100  GOSUB 1000: REM  LETTURA DEI DATI

110  GOSUB 2000: REM  STAMPA DEL MENU'
```

Dice al computer quanto spazio lasciare per le matrici.

Quando lanci il programma, la prima cosa che il computer fa è di andare alla subroutine alla riga 1000 per leggere i dati.

Dopo va alla riga 2000 per stampare sullo schermo il menú.

Le parentesi a sinistra del listato individuano le diverse parti del programma.

```
120  ON C GOSUB 200,300,400,500,600

130  GOTO 110
```

Questo lo invia alla subroutine giusta per svolgere l'operazione che hai scelto sul menú. Dopo aver eseguito la subroutine, il computer torna alla riga 130 che lo rimanda alla 110 per ristampare il menú.

```
200  REM  SUBROUTINE PER STAMPARE L'ELENCO
    DELLE SQUADRE
210  HOME
220  PRINT "ELENCO DELLE SQUADRE": PRINT
    "-----------------------"
230  FOR I = 1 TO 10: PRINT SQUADRA$(I): NEXT
    I
235  PRINT
```

Sottolinea le parole ELENCO DELLE SQUADRE.

Loop per stampare i nomi delle squadre. A ogni ripetizione del loop, I aumenta di 1 e il computer stampa il nome successivo in SQUADRA$.

```
240  INPUT "BATTI RETURN PER IL MENU'";X$

250  RETURN
```

Fa sí che il computer aspetti che tu batta RETURN prima di passare alla riga successiva. (Sull'Oric devi battere un tasto.)

Si torna alla riga 130.

```
300  REM  SUBROUTINE PER STAMPARE L'ELENCO
    DEGLI ANNI
310  HOME
320  PRINT "ELENCO DEGLI ANNI": PRINT
    "-------------------"
330  FOR I = 1 TO 12: PRINT ANNO(I): NEXT I
335  PRINT
340  PRINT "NEL 1942 E NEL 1946 LA COPPA NON
    E' STATA ASSEGNATA"
345  PRINT
350  INPUT "BATTI RETURN PER IL MENU'";X$
360  RETURN
```

Loop per stampare gli anni.

Di nuovo alla riga 130.

* Per modificare questo programma perché sia eseguibile sui computer Sinclair (Timex) vedi pagina 46.

```
400  REM  SUBROUTINE PER SCEGLIERE LE SQUADRE
405  HOME

410  INPUT "PER FAVORE SCRIVI IL NOME DELLA
     SQUADRA, O BATTI 'MENU' PER VEDERE
     NUOVAMENTE L'ELENCO ";Z$

415  IF Z$ = "MENU" THEN  RETURN

420  FOR I = 1 TO 10
425  IF Z$ = SQUADRA$(I) THEN  GOTO 440
430  NEXT I

435  PRINT : PRINT "SQUADRA NON TROVATA - PER
     FAVORE PROVA DI NUOVO": PRINT : GOTO 410

440  PRINT
445  PRINT Z$: PRINT

450  FOR J = 1 TO 12
455  IF M(I,J) = 1 THEN  PRINT "HA VINTO LA
     COPPA DEL MONDO NEL ";ANNO(J): PRINT
460  IF M(I,J) = 2 THEN  PRINT "FINALISTA NEL
     ";ANNO(J): PRINT
465  NEXT J

470  PRINT
480  INPUT "BATTI RETURN PER IL MENU'";X$

490  RETURN
```

In Z$ viene immagazzinato il nome di una squadra o la parola "menu".

Se Z$=MENU il computer torna alla riga 130 e poi alla 110 per stampare il menú.

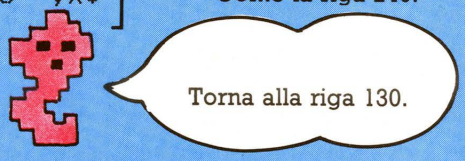
Loop per confrontare Z$ con tutti i nomi in SQUADRA$. Quando trova un nome uguale a Z$ va alla riga 440.

Questa riga ti avverte nel caso tu scriva male il nome di una squadra o tu ne indichi una che non è nel database.

Stampa il nome della squadra.

Loop per far sí che il computer trovi nella matrice i particolari relativi alla tua squadra. I è il numero di fila. Il valore di I viene stabilito dal loop alle righe 420-430 ed è l'indice (il numero che ne indica la posizione nella matrice) che la tua squadra ha in SQUADRA$. J è il numero di colonna e ogni volta che il loop viene ripetuto, il computer guarda nella colonna successiva accanto alla fila I.

Come la riga 240.




```
500  REM SUBROUTINE PER SCEGLIERE L'ANNO
505  HOME

510  INPUT "PER FAVORE SCRIVI L'ANNO, O BATTI
     'MENU' PER VEDERE NUOVAMENTE L'ELENCO ";Z$

515  IF Z$ = "MENU" THEN  RETURN

520  FOR I = 1 TO 12
525  IF  VAL (Z$) = ANNO(I) THEN  GOTO 540
530  NEXT I

535  PRINT : PRINT "ANNO NON TROVATO - PROVA
     DI NUOVO": PRINT : GOTO 510
```

Come la riga 410, ma questa volta il tuo anno è in Z$.

Loop per confrontare Z$ con tutti gli anni di ANNO. Non puoi confrontare una variabile a stringa con una variabile numerica, quindi devi usare il comando VAL, che dice al computer di considerare numeri i caratteri di Z$.

```
540  PRINT : PRINT "NEL ";Z$: PRINT
550  FOR J = 1 TO 10
555  IF M(J,I) = 1 THEN  PRINT SQUADRA$(J);"
     HA VINTO LA COPPA"
560  NEXT J

565  PRINT

570  INPUT "BATTI RETURN PER IL
     MENU ";X$
580  RETURN
```

Questo loop funziona come le righe 450-460, ma questa volta il numero della colonna è determinato dall'indice dell'anno in ANNO e il numero di fila cambia a ogni ripetizione del loop.

Torna alla riga 130.

```
600  REM  SUBROUTINE PER TERMINARE
610  INPUT "FINE - SICURO (S/N) ";X$
620  IF X$ <  > "S" THEN  RETURN END
```

Controlla che tu voglia veramente smettere. Se batti S, il comando del BASIC END dice al computer di terminare; se batti qualsiasi altra cosa, il computer torna alla riga 130. La parola ELSE rappresenta un utile modo per aggiungere altre condizioni alle istruzioni IF...THEN. Per saperne di più su questo argomento, volta pagina.

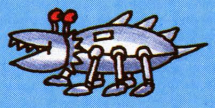
Se il tuo computer non usa la parola ELSE, puoi mettere END da sola su una nuova riga.

```
1000  FOR I = 1 TO 12: READ ANNO(I): NEXT I
1010  DATA  1930,1934,1938,1950
1020  DATA  1954,1958,1962,1966
1030  DATA  1970,1974,1978,1982
```

Loop per inserire i dati in ANNO.

```
1100  FOR I = 1 TO 10: READ SQUADRA$(I):
      NEXT I
1110  DATA  URUGUAY, ARGENTINA
1115  DATA  ITALIA, CECOSLOVACCHIA, UNGHERIA
1120  DATA  GERMANIA OVEST, BRASILE
1125  DATA  SVEZIA, INGHILTERRA, OLANDA
```

Loop per inserire i dati in SQUADRA$.



Stai molto attento quando scrivi questi dati: se ometti una virgola o un dato ottieni un errore.

```
1200  FOR I = 1 TO 10: FOR J = 1 TO 12
1205  READ M(I,J)
1210  NEXT J: NEXT I
```

Nido di loop per inserire i dati nella matrice bidimensionale M.

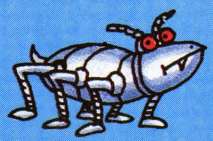
Questa volta torna alla riga 110.

```
1215  RETURN
```

```
1220  DATA  1,0,0,1,0,0,0,0,0,0,0,0
1230  DATA  2,0,0,0,0,0,0,0,0,0,1,0
1240  DATA  0,1,1,0,0,0,0,0,2,0,0,1
1250  DATA  0,2,0,0,0,0,2,0,0,0,0,0
1260  DATA  0,0,2,0,2,0,0,0,0,0,0,0
1270  DATA  0,0,0,0,1,0,0,2,0,1,0,2
1280  DATA  0,0,0,0,0,1,1,0,1,0,0,0
1290  DATA  0,0,0,0,0,2,0,0,0,0,0,0
1300  DATA  0,0,0,0,0,0,0,1,0,0,0,0
1310  DATA  0,0,0,0,0,0,0,0,0,2,2,0
```

Questi sono i dati per M.

Conviene controllare i dati diverse volte leggendo in orizzontale sulle file e in verticale sulle colonne. Se un valore è sbagliato, il computer quando guarda nella matrice fornisce l'informazione sbagliata.



IL LISTATO CONTINUA NELLA PAGINA SUCCESSIVA

```
2000  REM  SUBROUTINE PER STAMPARE IL MENU'
2010  HOME
2020  PRINT "              MENU'"
2030  PRINT "              ****"
2040  FOR I = 1 TO 6: PRINT :
      NEXT I
2050  PRINT "PER VEDERE L'ELENCO DELLE SQUADRE
      BATTI 1"
2060  PRINT
2070  PRINT "PER VEDERE L'ELENCO DEGLI ANNI
      BATTI 2"
2080  PRINT
2090  PRINT "PER INSERIRE IL NOME DI UNA
      SQUADRA BATTI 3"
2100  PRINT
2110  PRINT "PER INSERIRE UN ANNO BATTI 4"
2120  PRINT
2130  PRINT "PER TERMINARE BATTI 5"
2140  PRINT
2150  PRINT "BATTI IL NUMERO CHE HAI SCELTO"
2160  INPUT "POI PREMI RETURN ";C
2170  IF C < 1 OR C > 5 THEN  PRINT
      "PER FAVORE BATTI UN NUMERO FRA
      1 E 5": GOTO 2150
2180  RETURN
```

Qui lascia circa 15 spazi per centrare la parola menú sopra l'elenco delle scelte.

Loop per lasciare sei righe vuote.

Queste righe stampano il menú. Un menú dev'essere chiaro e "user-friendly" affinché chi utilizza il programma sappia esattamente cosa fare.

Il numero che hai scelto è memorizzato in C.

Questa riga ti avverte nel caso tu batta qualcosa di diverso da un numero fra 1 e 5. Per saperne di piú su OR vedi sotto.

Torna alla riga 120 per scegliere la subroutine appropriata.

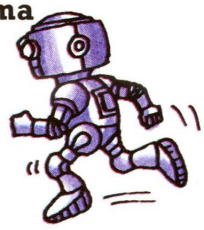
AND, OR o ELSE*

```
IF A=3 AND C$="SI'" THEN LET D=D+1
IF X<0 OR X>100 THEN PRINT "FUORI CAMPO"
IF ANNI<36 THEN PRINT "GIOVANE" ELSE
        PRINT "VECCHIO"
```

Puoi usare questi comandi del BASIC per aggiungere test e istruzioni a comandi IF... THEN, come mostrato negli esempi qui sopra. Quando usi AND il computer eseguirà il comando THEN solo se entrambi i confronti nell'istruzione IF risultano veri. La parola ELSE consente di dare istruzioni che il computer deve eseguire se nessuno dei due confronti risulta vero. Sai scrivere un breve programma che usi ELSE per risolvere il problema qui a destra?

* Questi comandi del BASIC non sono disponibili su tutti i computer.

Problema



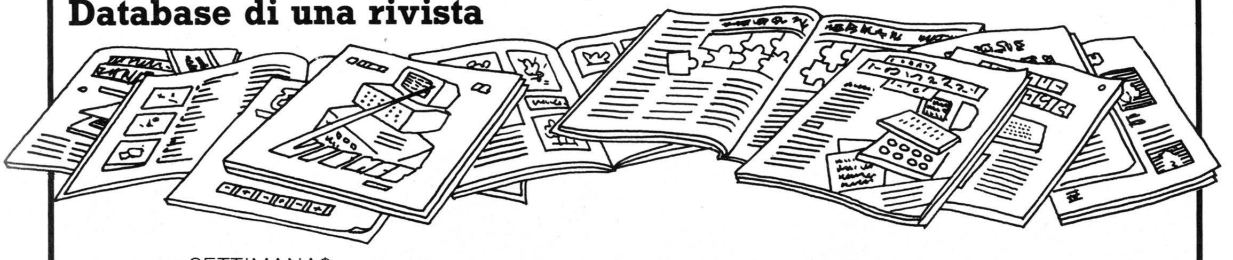
Il robot campione di corsa Zak può correre 500 metri al secondo, ma se la temperatura sale sopra i 15 gradi o scende sotto, la velocità di Zak aumenta o diminuisce di 10 metri al secondo. Sai scrivere un programma che stampi la distanza che Zak percorrerà a seconda della temperatura o del numero dei secondi che indichi? (Risposta a pagina 48.)

Come modificare il database

Una volta capito come funziona il programma, è molto facile modificarlo per costruire un database per un argomento diverso. Qui sotto puoi trovare alcune idee per vari database.

Dopo aver deciso l'argomento del database, costruisci una tavola con tutti i valori dei dati, come quella a pagina 18. La tua tavola può avere un numero di file e di colonne diverso, nel qual caso devi modificare le dimensioni delle matrici del programma. Successivamente inserisci i tuoi dati nelle righe del programma con DATA e riscrivi le domande del menú. Ricorda di cambiare le istruzioni DIM e il numero di esecuzioni dei loop per l'inserimento dei dati nelle matrici.

Database di una rivista

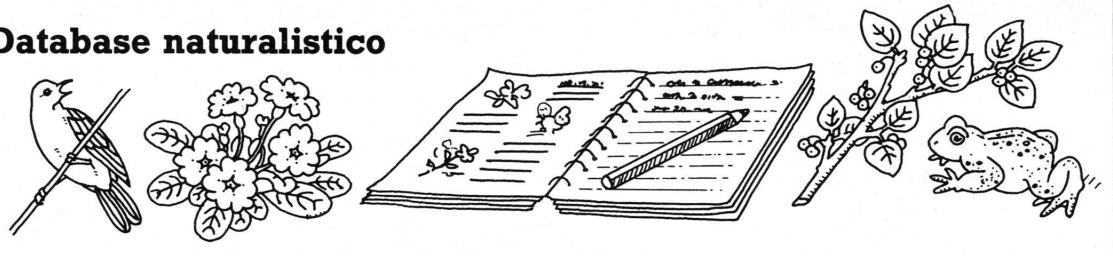


SETTIMANA$

ARGOMENTO$ | MATRICE

Un database di una rivista può servire a trovare in che mese è stato trattato un argomento, o quali argomenti sono stati trattati in un determinato mese, risparmiando ore di ricerca di un determinato articolo. Per far questo sarà necessaria una matrice chiamata SETTIMANA§ e un'altra chiamata ARGOMENTO$, oltre alla matrice con i dati.

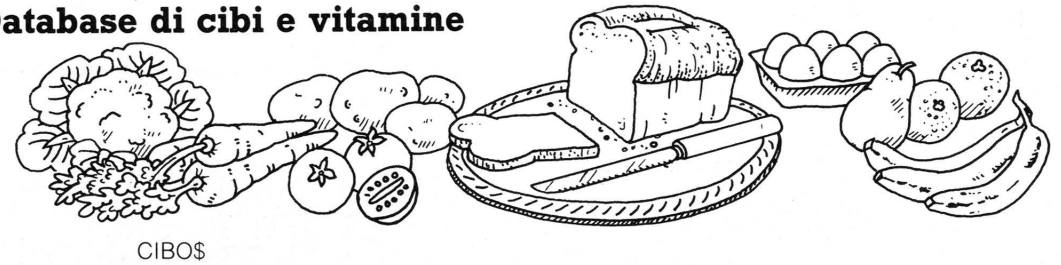
Database naturalistico



LUOGO$ | MATRICE

Un database relativo a uccelli o a piante potrebbe mostrare dove o quando sono stati avvistati. Ci vorrà una matrice per il nome degli uccelli o delle piante e un'altra per il luogo o il momento dell'avvistamento. Le associazioni naturalistiche stanno compilando database per registrare la distribuzione delle diverse specie.

Database di cibi e vitamine



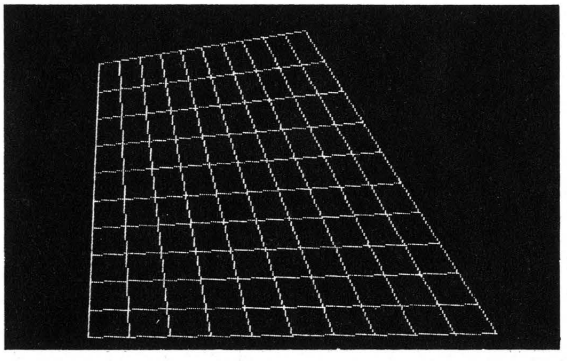
CIBO$

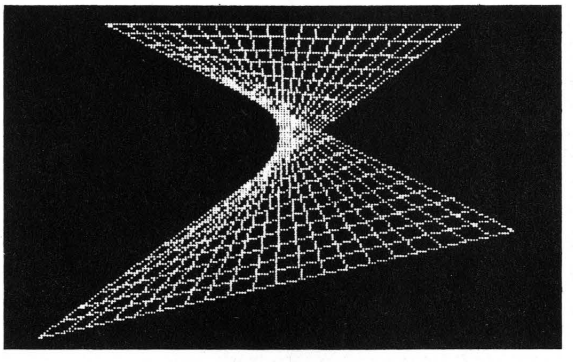
VIT$ | MATRICE

Questo database permette di trovare quali cibi contengono una determinata vitamina o quali vitamine sono presenti in un determinato cibo. Un'altra idea sarebbe un database di cibi e calorie, per vedere le calorie di un determinato cibo o quali cibi hanno piú di un certo numero di calorie.

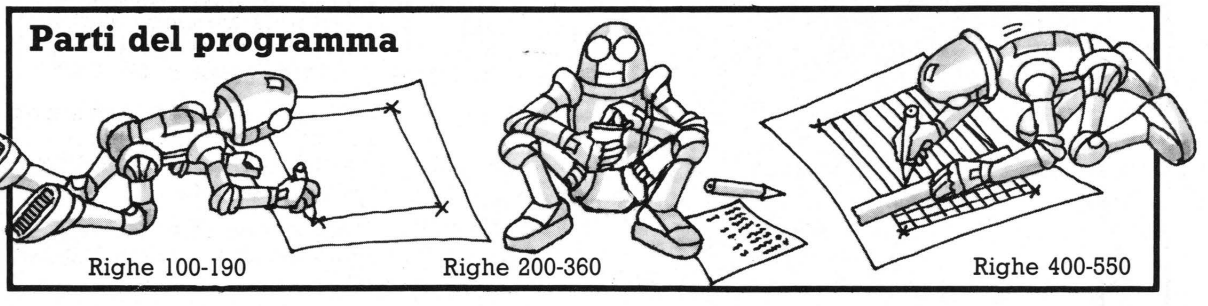
Grafica istantanea

Questo programma traccia sullo schermo figure semplici e le riempie con un reticolo di linee. Nella grafica dei computer i reticoli vengono usati spesso, per rendere le figure piú "tridimensionali" o piú "spaziali". Il programma utilizza il comando grafico HPLOT X,Y per tracciare un punto e HPLOT X1, Y1 TO X2, Y2 per una linea; le coordinate X e Y vengono misurate dal bordo dello schermo. Dovrai adattare queste istruzioni al tuo computer e aggiungere eventuali comandi per indicare il tipo di grafica.* Esistono due modi diversi per scrivere i programmi grafici: puoi dire al computer di calcolare e tracciare tutti i punti via via che procede, costruendo gradualmente la figura sullo schermo, oppure gli puoi far eseguire i calcoli prima, immagazzinare i risultati in matrici, poi tracciare la figura completa quasi istantaneamente. Il programma che segue utilizza l'approccio della "grafica istantanea".





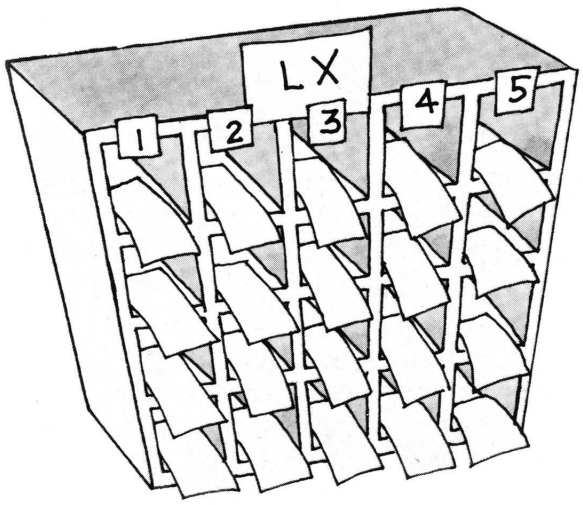
Puoi ottenere figure e forme di ogni genere modificando i dati del programma; è anche possibile tracciare le linee del reticolo in colori diversi. Alla fine del listato vengono dati alcuni suggerimenti su come adattare il programma.

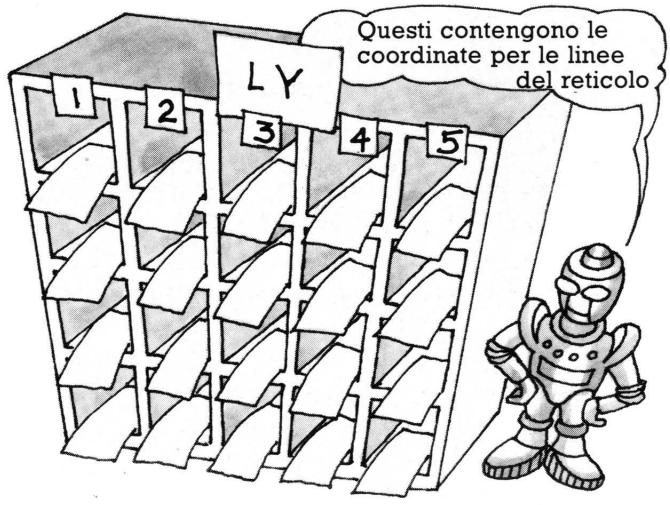


Il programma è composto da tre parti principali: la prima (righe 100-190) traccia gli angoli del reticolo e le linee che li uniscono; la seconda (righe 200-360) calcola le coordinate del reticolo e la terza (rige 400-550) traccia le linee.

Per memorizzare tutti i dati delle coordinate, il programma utilizza quattro matrici: CX e CY contengono le coordinate X e Y dei quattro angoli del reticolo. I dati relativi a queste matrici vengono forniti all'inizio del programma; CX(1) e CY(1) contengono le coordinate del primo angolo, CX(2) e CY(2) quello del secondo e cosí via.

* Su alcuni computer, per esempio lo Spectrum e l'Oric, le coordinate X e Y di una linea vengono misurate dall'ultimo punto tracciato. Per modificare il programma per questi computer vedi pagina 47.





LX e LY sono le matrici in cui vengono memorizzate le coordinate delle linee del reticolo. Sono matrici bidimensionali, ognuna con quattro file; il numero delle colonne dipende da quello delle linee. Ogni fila contiene le coordinate delle linee di un lato del reticolo, con la fila 1 corrispondente al lato 1 ecc. Il computer memorizza i dati in LX e LY via via che effettua i calcoli del programma.

Il programma

```
100   INPUT "QUANTE LINEE VUOI
      NEL RETICOLO? ";N
110   DIM CX(4),CY(4)
115   DIM LX(4,N),LY(4,N)
120   HOME
130   REM TRACCIA I LATI DEL
      RETICOLO
135   REM INSERISCI
      L'ISTRUZIONE DEL TUO
      COMUTER RELATIVA ALLA
      MODALITA' GRAFICA
136   HGR : HCOLOR= 3
140   FOR I=1 TO 4
145   READ CX(I),CY(I)
150   NEXT I
160   DATA 10,50,220,50,250,150,120,150
170   HPLOT CX(4),CY(4)
180   FOR I=1 TO 4
185   HPLOT TO CX(I),CY(I)
190   NEXT I
```

Prova con 20 per i computer con grafica ad alta risoluzione e 5 per quelli a bassa.

Comunica al computer quanto devono essere grandi le matrici.

I valori qui forniti si riferiscono al reticolo a sinistra della pagina precedente. Per ottenere reticoli diversi modifica quei valori.

Loop per inserire in CX e CY i dati degli angoli. A ogni ripetizione del loop i due valori successivi della riga 160 vengono inseriti in CX e CY.

Sono le coordinate degli angoli. Può darsi che questi valori vadano modificati per entrare nel tuo schermo.

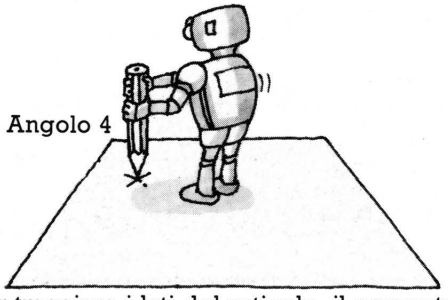
Usa i comandi del tuo computer per HPLOT

Traccia l'angolo 4 usando i valori immagazzinati in CX(4) e CY(4).

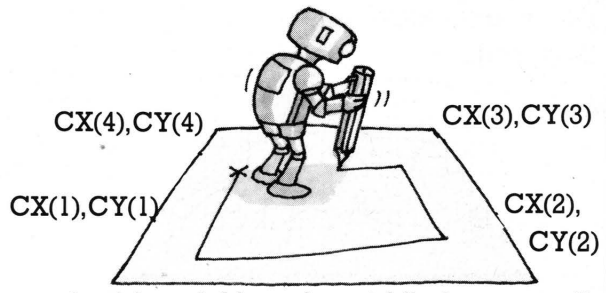
Loop per tracciare i lati del reticolo.

IL LISTATO CONTINUA NELLA PAGINA SEGUENTE

Come tracciare i lati



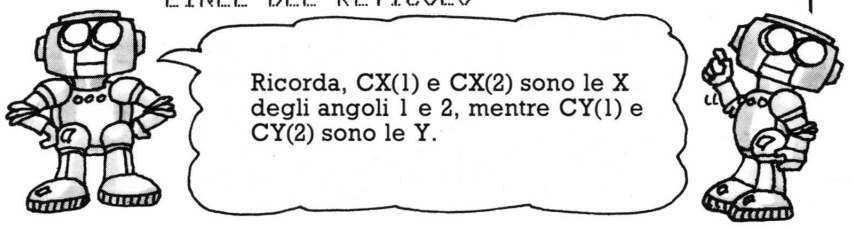
Per tracciare i lati del reticolo, il computer comincia dall'angolo 4 (riga 170).



Successivamente il loop delle righe 180-190 gli fa tracciare una linea all'angolo 1. Ad ogni ripetizione del loop, la variabile 1 aumenta di uno e il computer traccia una linea all'angolo successivo.

```
200  REM  CALCOLO DELLE COORDINATE DELLE
     LINEE DEL RETICOLO
```

La parte successiva del programma è costituita da quattro loop per calcolare le coordinate delle linee del reticolo. La figura qui sotto mostra il procedimento.



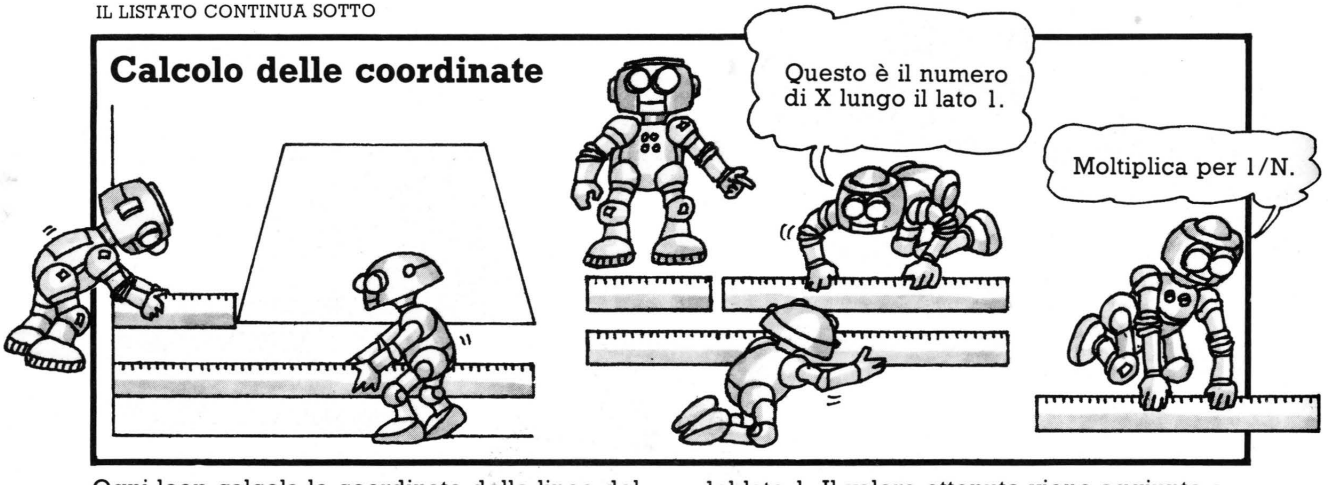

```
210  FOR I=1 TO N
220  LET LX(1,I)=CX(1)+(CX(2)-CX(1))*I/N
230  LET LY(1,I)=CY(1)+(CY(2)-CY(1))*I/N
240  NEXT I
```

N è il numero di linee del reticolo che hai scelto.

Calcola le X per le linee del reticolo lungo il lato 1 e le immagazzina in LX fila 1, colonne da 1 a N.

Calcola le Y per il lato 1 e le immagazzina in LY fila 1 colonne da 1 a N.

IL LISTATO CONTINUA SOTTO



Ogni loop calcola le coordinate delle linee del reticolo lungo un lato del reticolo stesso. Per esempio, le righe 210-240 calcolano le coordinate del lato 1. Alla riga 22 il computer sottrae CX(1) da CX(2), fornendo cosí il numero di X lungo il lato 1. Alla prima esecuzione del loop, questo valore viene moltiplicato per 1/N (N è il numero di linee che hai scelto). Se, per esempio, N è 5, si ottiene 1/5 della lunghezza del lato 1. Il valore ottenuto viene aggiunto a CX(1) e la risposta memorizzata in LX(1,1). Alla seconda esecuzione del loop, I=2, cosí che la moltiplcazione è per 2/5 e la risposta viene memorizzata in LX(1,2). Questo viene ripetuto per tutti i valori di I, da 1 a N, per trovare tutte le X delle linee del reticolo lungo il lato 1. Lo stesso metodo viene utilizzato alla riga 230, per trovare le Y.

```
250  FOR I=1 TO N
260  LET LX(3,I)=CX(4)+(CX(3)-CX(4))*I/N
270  LET LY(3,I)=CY(4)+(CY(3)-CY(4))*I/N
280  NEXT I
```

Loop per il calcolo delle coordinate per il lato 3. Per memorizzarle nello stesso ordine del lato 1 (cioè da sinistra a destra), il computer deve effettuare le somme nell'ordine opposto. L'angolo 4 viene sottratto dal 3 e il risultato aggiunto all'angolo 4.

Cambia queste righe in modo che siano come gli altri loop e guarda cosa succede.

```
290  FOR I=1 TO N
300  LET LX(2,I)=CX(2)+(CX(3)-CX(2))*I/N
310  LET LY(2,I)=CY(2)+(CY(3)-CY(2))*I/N
320  NEXT I
```

Loop per calcolare le coordinate del lato 2.

IL LISTATO CONTINUA ALLA PAGINA A FRONTE

```
330  FOR I=1 TO N
340  LET LX(4,I)=CX(1)+(CX(4)-CX(1))*I/N
350  LET LY(4,I)=CY(1)+(CY(4)-CY(1))*I/N
360  NEXT I

400  REM TRACCIA LE LINEE DEL RETICOLO
410  LET FILA=1: GOSUB 500
420  LET FILA=2: GOSUB 500

430  STOP

500  REM SUBROUTINE
510  FOR I=1 TO N
520  HPLOT LX(FILA,I),LY(FILA,I)
530  HPLOT TO LX(FILA+2,I),LY(FILA+2,I)
540  NEXT I

550  RETURN
```

Loop per calcolare le coordinate per il lato 4.

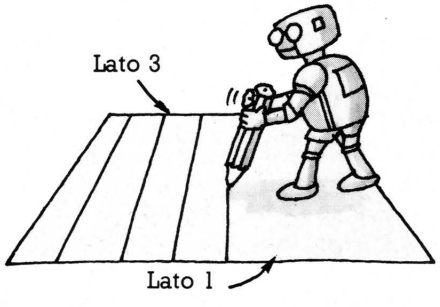
Questa riga introduce una variabile chiamata FILA e le attribuisce il valore 1, dopo di che il computer va alla subroutine alla riga 500, la esegue, ritorna alla riga 420 e cambia FILA in 2 e va nuovamente alla subroutine.

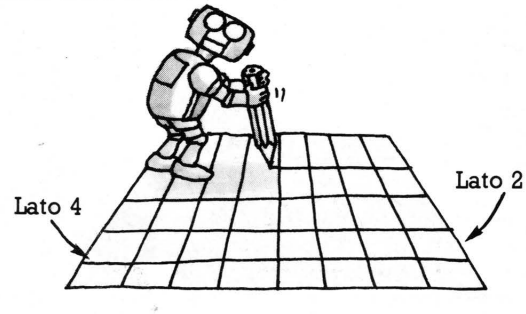
Questo arresta il programma dopo che la subroutine è stata eseguita due volte.

FILA e I sono gli indici di LX e LY: dicono al computer in quali scompartimenti cercare le coordinate X e Y di ciascuna linea del reticolo. FILA è il numero di fila e I quello di colonna. La fila 1 contiene le coordinate del lato 1, la fila 2 del lato 2 ecc.

Ritorna alla riga 420 dopo la prima esecuzione della subroutine e alla riga 430 dopo la seconda.

Come vengono tracciate le linee del reticolo





Alla prima esecuzione della subroutine, FILA = 1, quindi alla riga 520 il computer traccia un punto sul lato 1. Alla riga 530 aggiunge 2 a FILA, quindi traccia una linea fino al lato 3. Alla seconda esecuzione della subroutine FILA = 2, quindi traccia punti sulla fila 2 e linee fino al lato 4.

Idee per modificare il programma

1. Per costruire reticoli di forma diversa, inizia con il calcolare su carta le coordinate della forma che vuoi. Ricorda che la prima coppia di valori della riga 160 sono le coordinate dell'angolo 1, la seconda coppia dll'angolo 2 ecc. Se due angoli sono gli stessi, ottieni un triangolo. Prova a far intersecare i lati, come nella figura sulla destra di pagina 26.

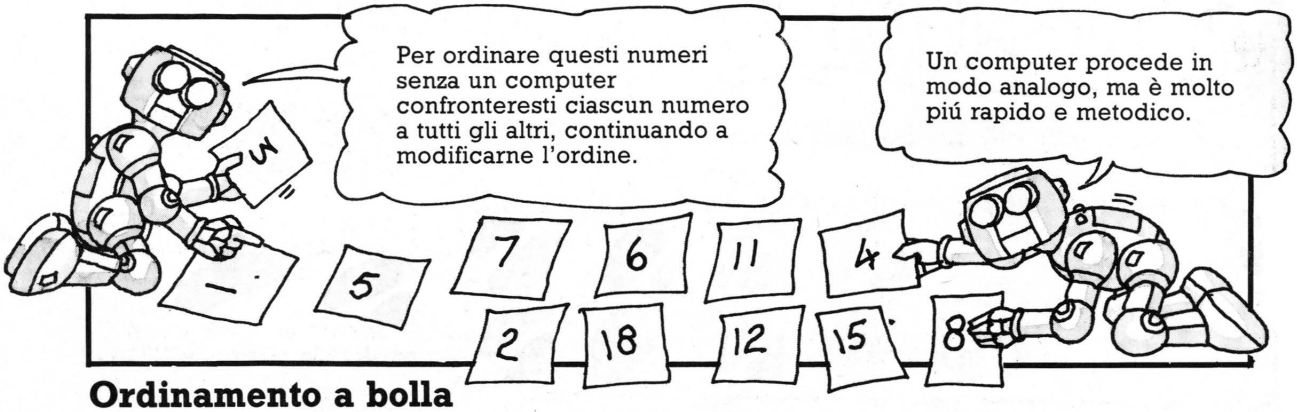
2. Usando INPUT con un loop puoi ordinare al computer di chiederti dati. Sostituisci le righe 140-160 con le seguenti:

```
140 FOR I=1 TO 4
150 PRINT "QUALI SONO LE COORDINATE DELL'ANGOLO ";I
155 INPUT CX(I),CY(I)
160 NEXT I
```

3. Per ottenere linee di reticolo colorate, inserisci il comando relativo ai colori del tuo computer prima dei GOSUB delle righe 410 e 420. Ricorda di separare con due punti GOSUB dal comando relativo ai colori.

Programmi per l'ordinamento dei dati

Talvolta è necessario disporre dati in ordine alfabetico o numerico per organizzare, per esempio, l'indice di una rivista, o analizzare informazioni raccolte sul tempo, sulla natura, o su qualche altro argomento. Se i dati sono pochi, si possono facilmente ordinare a mano, ma quando sono molti il computer è piú rapido e piú accurato. I programmi di ordinamento sono chiamati anche programmi "sort" o di "sorting" (*sort* significa ordinare); ne esistono molti in BASIC e li potrai trovare sulle riviste specializzate. I vari programmi possono essere scritti con tecniche diverse, secondo i compiti da svolgere. Nelle pagine che seguono ne vengono presentati due tipi: uno è chiamato "ordinamento a bolla" (*bubble sort*), e capirai subito perché, mentre l'altro è un ordinamento Shell, (dal nome del suo ideatore). L'ordinamento a bolla è uno dei metodi piú lenti ed è utile solo quando i dati sono pochi. Un ordinamento Shell è molto piú rapido. A pagina 35 troverai alcune righe che puoi aggiungere ai programmi per confrontarne le velocità e vedere la rapidità del tuo computer.



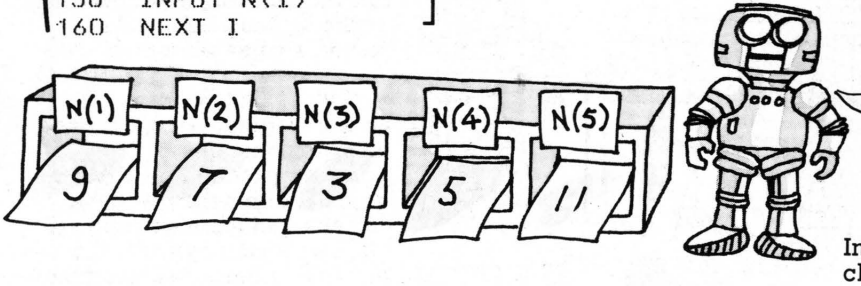
Ordinamento a bolla

In un ordinamento a bolla il computer comincia dall'inizio dell'elenco non ordinato e confronta fra loro i primi due elementi: se sono in ordine sbagliato, li inverte, poi passa ai due elementi successivi. Esamina cosí tutto l'elenco, e gli elementi "piú piccoli" affiorano gradatamente, come una bolla, alla superficie dell'elenco.
Il programma che segue è un ordinamento a bolla di numeri, mentre alla pagina successiva c'è un ordinamento a bolla di parole.

```
100  REM  ORDINAMENTO A BOLLA DI
     NUMERI
110  INPUT "QUANTI SONO I NUMERI DA
     ORDINARE? ";T
120  DIM N(T)
130  FOR I=1 TO T
140  PRINT "NUMERO ";I
150  INPUT N(I)
160  NEXT I
```

Introduce una matrice chiamata N con T scompartimenti. T è il totale dei numeri da ordinare.

Chiede quali sono i numeri da ordinare e li memorizza nella matrice.



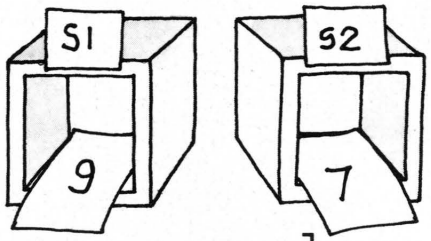
In questo esempio i numeri sono cinque.

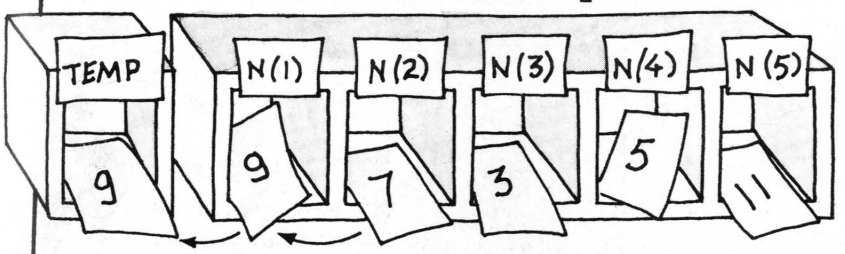
```
170  LET MAX=T
175  LET X=0
```

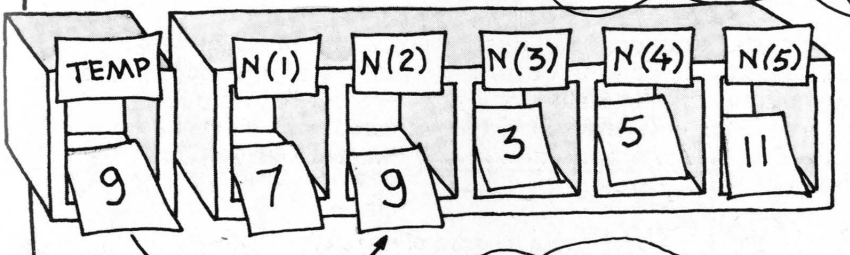
Introduce un'altra variabile chiamata MAX per tenere il conto del totale, perché, durante l'esecuzione del programma, il valore di T varia.

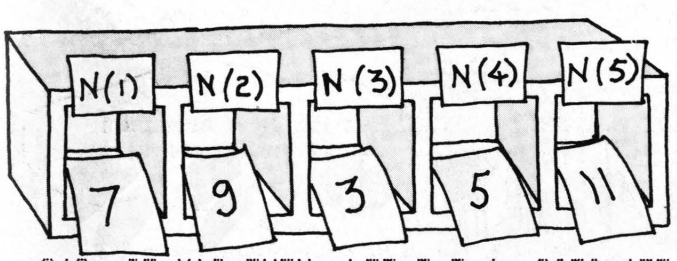
X è un contatore.

```
180  FOR C=1 TO T-1
190  LET S1=N(C): LET S2=N(C+1)
200  IF S1<=S2 THEN  GOTO 250
210  LET TEMP=N(C)
220  LET N(C)=N(C+1)
230  LET N(C+1)=TEMP
240  LET X=X+1
250  NEXT C
260  IF X>0 THEN  LET T=T-1: GOTO 175
270  PRINT "I NUMERI ORDINATI SONO "
280  FOR I=1 TO MAX
290  PRINT N(I)
300  NEXT I
```









Loop da eseguirsi T-1 volte. E' quanto ci vuole per confrontare ogni numero dell'elenco.

Le variabili S1 e S2 servono a contenere ogni coppia di numeri durante il confronto. Alla prima esecuzione del loop, C=1, cosí che N1 e N2 vengono inseriti in S1 e S2.

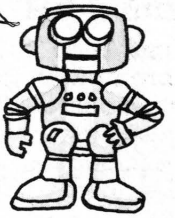
Confronta i due numeri. Se S1 è piú piccolo di S2, i numeri sono in ordine corretto e la riga 250 rimanda il computer all'inizio del loop a scegliere la coppia successiva. Se S1 è maggiore di S2, il computer procede con le successive righe che invertono le posizioni dei due numeri nella matrice.

Il numero in N(C) viene inserito in una variabile chiamata TEMP.

A ogni ripetizione del loop, il numero viene spostato di una posizione a destra finché arriva alla posizione corretta.

Il numero in posizione N(C+1) è spostato nella posizione N(C) della matrice.

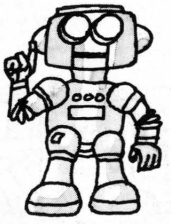
Il numero in TEMP viene messo alla posizione N(C+1).



Aggiunge uno a X per mostrare che c'è stata un'inversione.

Alla seconda esecuzione del loop, C=2 cosí che N(C)=N(2) e N(C+1)=N(3).

Rimanda il computer al confronto della successiva coppia di numeri. Dopo che il loop è stato ripetuto T-1 volte, il computer ha confrontato una volta tutti i numeri e passa alla riga 260.



Se X è maggiore di 0, c'è stata un'inversione, quindi il computer sottrae 1 da T poiché un numero è nella posizione corretta, poi torna all'inizio del loop. Se X=0, i numeri sono nell'ordine giusto e il computer passa alla riga 270.

MAX è il numero totale dei numeri ordinati.

Ripeti IF...THEN se il GOTO della riga 260 viene inserito in un'altra riga.

Ordinamento a bolla di parole

Il programma che segue è un ordinamento a bolla di parole. E' come quello per i numeri, solo che le variabili che contengono i dati (N, S1, S2 e TEMP) sono variabili a stringa.*
Il computer confronta le lettere con lo stesso metodo utilizzato per i numeri. All'interno del computer, anche le lettere e i simboli sono rappresentati da numeri, cosí quando il computer deve confrontare caratteri, ne confronta i codici numerici. Per confrontare due parole, controlla innanzi tutto la prima lettera di entrambe e, se sono uguali, passa alla seconda e cosí via. Nelle variabili a stringa puoi inserire sia numeri che lettere, quindi puoi usare l'ordinamento a bolla di parole anche per dati che contengono sia parole che numeri, come indirizzi o voci di un indice.

```
QUANTI ELEMENTI DA ORDINARE?4
ELEMENTO 1
?DISK DRIVE 34 76 82 93
ELEMENTO 2
?CODICE MACCHINA 55 72 85
ELEMENTO 3
SUONO 32
ELEMENTO 4
?GRAFICA 8 23 45
L'ELENCO ORDINATO E'
CODICE MACCHINA 55 72 85
DISK DRIVE 34 76 82 93
GRAFICA 8 23 45
SUONO 32
```

```
QUANTI ELEMENTI DA ORDINARE?4
ELEMENTO 1
?ROSSI PAOLO VIA ROMA 14
ELEMENTO 2
?FERRARI MARIO VIA SALERNO 5
ELEMENTO 3
?FERRI MARCELLA VIA VITTORIO 12
ELEMENTO 4
?ALESSI MARCO VIA CINQUE MAGGIO 2
L'ELENCO ORDINATO E'
ROSSI PAOLO VIA ROMA 14
FERRI MARCELLA VIA VITTORIO 12
FERRARI MARIO VIA SALERNO 5
ALESSI MARCO VIA CINQUE MAGGIO 2
```

In questo esempio il computer ordina elementi di un indice e, a destra, indirizzi. Gli elementi sono stati inseriti senza virgole, poiché per la maggior parte dei computer la virgola è un separatore, o "delimitatore", fra le informazioni. Per contenere virgole, una stringa dev'essere fra virgolette.

Il programma

```
100  REM  ORDINAMENTO A BOLLA PER PAROLE
110  INPUT "QUANTI ELEMENTI DA ORDINARE? ";T
120  DIM N$(T)
130  FOR I=1 TO T
140  PRINT "ELEMENTO ";I
150  INPUT N$(I)
160  NEXT I
170  LET MAX=T
175  LET X=0
180  FOR C=1 TO T-1
190  LET S1$=N$(C): LET S2$=N$(C+1)
200  IF S1$<=S2$ THEN  GOTO 250
210  LET TEMP$=N$(C)
220  LET N$(C)=N$(C+1)
230  LET N$(C+1)=TEMP$
240  LET X=X+1
250  NEXT C
260  IF X>0 THEN  LET T=T-1: GOTO 175
270  PRINT "L'ELENCO ORDINATO E'"
280  FOR I=1 TO MAX
290  PRINT N$(I)
300  NEXT I
```

Introduce una matrice chiamata N$ con T scompartimenti.

Prova a inserire elementi che iniziano con un simbolo, un numero, una lettera maiuscola e una minuscola e guarda in che ordine vengono disposti.

I primi due elementi vengono inseriti in S1$ e S2$.

Confronta S1$ e S2$.

Scambia le posizioni dei primi due elementi di N$.

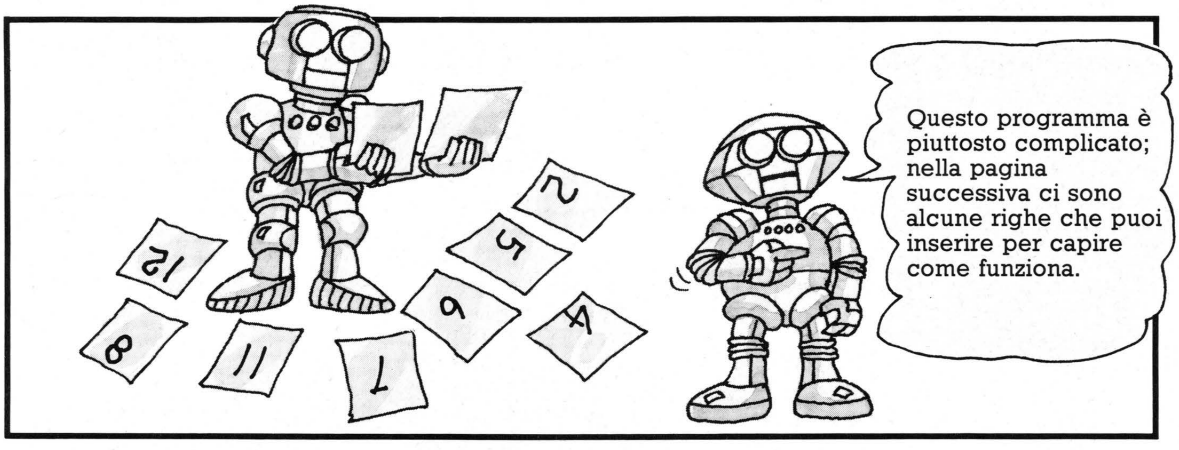
If X>0, sottrae 1 dal totale e torna all'inizio del loop per controllare nuovamente l'elenco.

Se metti il GOTO della riga 260 su una riga a sé, ripeti IF...THEN.

* Per i computer Sinclair (Timex), cambia la riga 120 in DIM N$(T,N), dove N è la lunghezza della stringa piú lunga che vuoi inserire.

Ordinamento Shell

Se i dati sono numerosi, l'ordinamento a bolla è molto lento: su alcuni computer è necessario quasi un minuto per cinquanta elementi. Il programma che segue è un ordinamento Shell per numeri che è circa tre volte piú rapido di un ordinamento a bolla.
In un ordinamento Shell, il computer divide l'elenco dei dati in due e confronta tutti quelli di una metà con quelli dell'altra, poi divide nuovamente l'elenco in due e compie molti altri confronti usando le stesse tecniche dell'ordinamento a bolla.



Il programma

```
100  REM  ORDINAMENTO SHELL PER
     NUMERI
110  INPUT "QUANTI SONO I NUMERI DA
     ORDINARE? ";T
120  DIM N(T)
130  FOR I=1 TO T
140  PRINT "NUMERO ";I
150  INPUT N(I)
160  NEXT I
170  LET C=T
180  LET C= INT (C/2)
190  IF C=0 THEN  GOTO 330
200  LET D=T-C
210  LET E=1
220  LET F=E
230  LET G=F+C
```

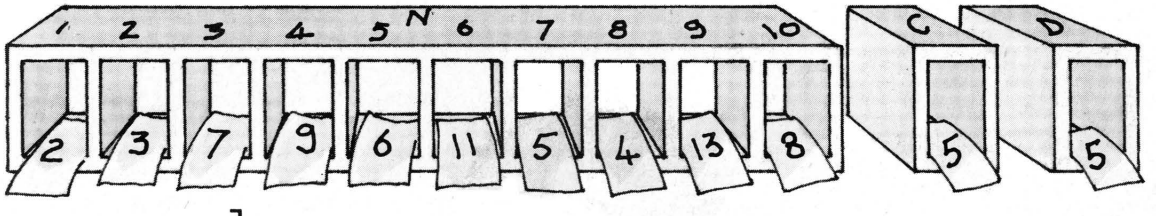
Queste righe sono come quelle di un ordinamento a bolla: T è il totale dei numeri da ordinare e le righe 120-160 richiedono l'inserimento dei numeri e li immagazzinano nella matrice N.

Rende C uguale al numero totale dei dati.

Divide C in due per fornire il numero di elementi della prima metà dell'elenco. INT fa scartare le cifre che seguono il punto decimale per rendere C un numero intero.

Il programma divide ripetutamente C per due e quando C=0 il computer torna alla riga 330 per stampare l'elenco ordinato.

D è il numero di elementi della seconda parte dell'elenco.



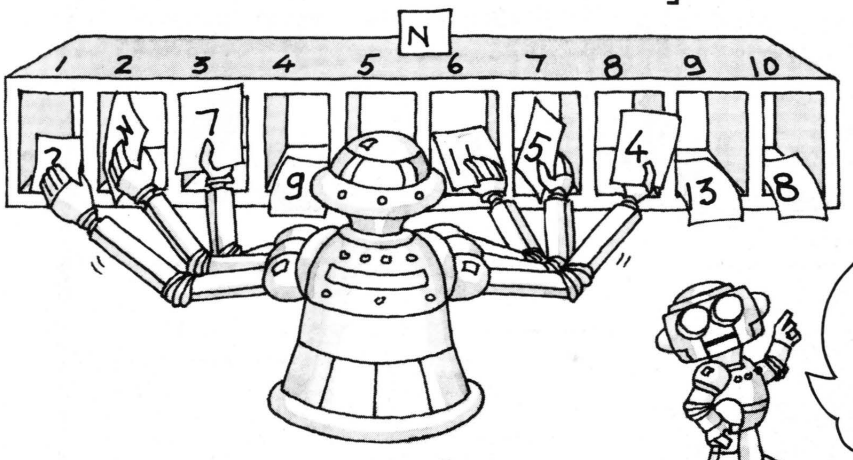
E è un contatore.

Le variabili F e G servono a determinare gli indici di N, la matrice in cui sono memorizzati tutti i numeri.

IL LISTATO CONTINUA ALLA PAGINA SUCCESSIVA

```
240  IF N(F)<=N(G) THEN  GOTO 300
```

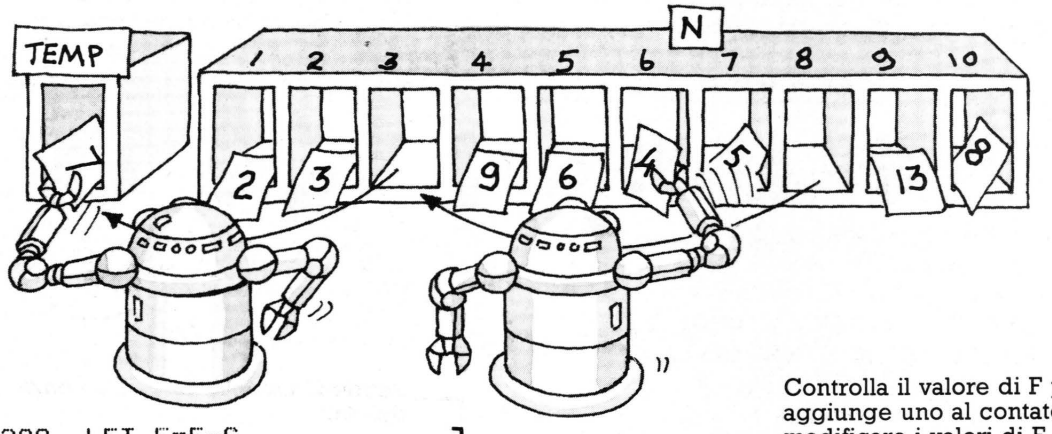
Se il numero in N(F) è piú piccolo di quello in N(G) il computer va alla riga 300, addiziona 1 a E e ritorna ad attribuire a F e G i valori degli indici della successiva coppia di numeri.



In questo esempio i numeri da ordinare sono 10. Al primo passaggio C è uguale a 5, quindi il computer confronta i numeri con indice da 1 a 5 con quelli da 6 a 10.

```
250  LET TEMP=N(F)
260  LET N(F)=N(G)
270  LET N(G)=TEMP
```

Se N(F) è maggiore di N(G) il computer li scambia fra loro.




```
280  LET F=F-C
290  IF F>0 THEN  GOTO 230
300  LET E=E+1
```

Controlla il valore di F poi aggiunge uno al contatore E per modificare i valori di F e di G alle righe 220 e 230.

```
310  IF E>D THEN  GOTO 180
```

Verifica che E sia dalla parte di D dell'elenco. Se non lo è torna alla riga 180 per dividere nuovamente l'elenco.

```
320  GOTO 220
```

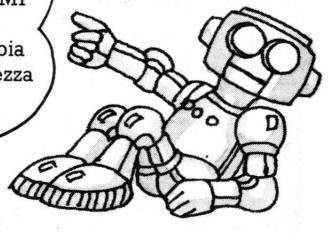
Se E è minore di D, il computer va alla riga 220 per determinare gli indici della successiva coppia di numeri.

```
330  PRINT "L'ELENCO ORDINATO E'"
340  FOR I=1 TO T
350  PRINT N(I)
360  NEXT I
```

Stampa l'elenco ordinato.

Sai aggiungere righe perché il computer ti dica, dopo aver ordinato un elenco, quanti confronti e quante inversioni ha effettuato? (Risposta a pagina 48). Dopo potresti provare il programma su vari elenchi e vedere come variano.

Per modificare questo programma in modo che ordini stringhe, cambia la variabile N in N$ (alle righe 120, 150, 240-270 e 350), la variabile TEMP in TEMP$ (alle righe 250 e 270) e riscrivi le istruzioni con PRINT. Se hai un Sinclair, cambia la riga 120 in DIM N$(T,N), dove N è la lungezza della stringa piú lunga che vuoi inserire.



Come funziona?

Per avere piú chiaro il funzionamento dell'ordinamento Shell, puoi inserire nel programma queste righe che stampano i valori delle variabili, per vedere quali sono i numeri che il computer confronta.

```
233  PRINT
235  PRINT "F= ";F;" G= ";G
237  PRINT "CONFRONTA N(";F;") E
     N(";G;")"
245  PRINT "INVERTI N(";F;") E N(";G;")"
274  PRINT "ELENCO= ";
275  FOR J=1 TO T
276  PRINT N(J);" ";
277  NEXT J
278  PRINT

279  INPUT "PER CONTINUARE BATTI RETURN";Z$
```

F e G sono gli indici dei numeri da confrontare

Comunica gli indici dei numeri da scambiare.

Stampa l'ordinamento corrente dell'elenco.

Confronto fra ordinamenti

Se hai provato l'ordinamento a bolla e quello Shell solo con pochi numeri, forse non hai notato quanto sia piú rapido l'ordinamento Shell. Per valutare i due ordinamenti, puoi far generare a entrambi un elenco di numeri casuali e poi guardare quanto impiega ciascun programma per ordinarli. Piú sono i numeri, piú aumenta la differenza fra i tempi necessari con i due metodi; alla pagina che segue ci sono alcuni programmi che tracciano grafici per mostrare quelle differenze.

Generazione di numeri casuali

```
140  LET N(I)= INT ( RND
     (1)*200+1)
150  PRINT N(I)
165  INPUT "PREPARA L'OROLOGIO E
     PREMI RETURN PER INIZIARE
     L'ORDINAMENTO";Z$
```

Puoi sostituire questo valore con qualsiasi altro.

Per far sí che i programmi generino i propri elenchi di numeri, devi sostituire le righe 140 e 150 di entrambi i programmi e inserire una nuova riga 165 cosí da controllare quando inizia l'ordinamento.

La riga 140 genera numeri casuali fra 1 e 200 e li immagazzina nella matrice N. La riga 150 stampa i numeri sullo schermo e la 165 fa sí che il programma aspetti finché non viene premuto RETURN.

Esecuzione del test

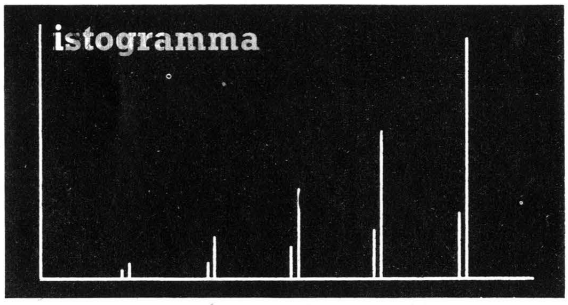
Per valutare ciascun programma, lo dovresti lanciare diverse volte, la prima per vedere quanto ci vuole per ordinare, per esempio, 10 numeri, poi 20, poi 30 ecc. Computer diversi ordinano gli elenchi a velocità diverse e alcuni, come lo ZX81, hanno una modalità rapida e una lenta. Quelle che seguono sono le velocità su un Apple II.

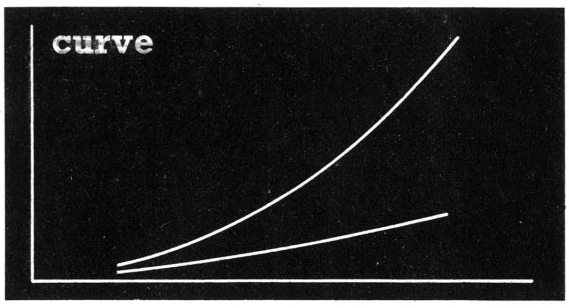
Prova di ordinamento

N. di numeri da ordinare	10	20	30	40	50
Ordinamento a bolla	2 sec	5 sec	11 sec	18 sec	29 sec
Ordinamento Schell	1 sec	2 sec	4 sec	6 sec	8 sec

Come tracciare grafici

I risultati di un computer sono molto piú facili da leggere e da capire se presentati in una forma interessante, usando sia grafici che parole. Il programma che segue crea un istogramma che mostra la differenza fra l'ordinamento a bolla e quello Shell. Nella pagina a fronte puoi vedere come modificare il programma per ottenere una curva. I programmi sono piuttosto lineari e puoi facilmente modificarli per mostrare informazioni diverse. Li puoi anche migliorare aggiungendo i comandi relativi ai colori del tuo computer, cosí da ottenere grafici a colori.





Queste sono le immagini prodotte dai due programmi grafici. Entrambi i diagrammi mettono a confronto il tempo impiegato dai due ordinamenti per ordinare 10, 20, 30, 40 e 50 numeri. Il tempo è sull'asse delle Y mentre il numero dei numeri ordinati è nell'asse delle X. Se il tuo computer può scrivere nella zona dei pixel, puoi aggiungere etichette per rendere i grafici piú chiari. Nella pagina a fronte puoi vedere come procedere con il computer BBC.

Programma per l'istogramma

```
100  DIM B(5): DIM S(5)
110  LET N=0
120  FOR I=10 TO 50 STEP 10
130  LET N=N+1
140  PRINT "PER ";I;" NUMERI"
150  INPUT "QUANTI SECONDI HA
     IMPIEGATO L'ORDINAMENTO A BOLLA?
     ";B(N)
160  INPUT "E QUELLO SHELL? ";S(N)
170  NEXT I
180  INPUT "QUANTI PUNTI HA IL TUO
     SCHERMO IN VERTICALE? ";H
190  INPUT "E IN ORIZZONTALE? ";W
200  REM  SE NECESSARIO IMPARTISCI IL
     COMANDO DI MODALITA' GRAFICA
205  HGR : HCOLOR= 3
210  REM   TRACCIA GLI ASSI
220  HPLOT 1,H: HPLOT  TO W,H: HPLOT
     1,H:: HPLOT  TO 1,1
230  REM  TRACCIA L'ISTOGRAMMA
240  LET X=(W*0.75)/5
250  LET Y=(H*0.75)/B(5)
```

Introduce le matrici B e S per contenere i dati degli ordinamenti a bolla e Shell.

Loop per l'inserimento dei dati nelle matrici. N è un contatore per determinare gli indici delle matrici.

Nota coma la variabile del loop I viene usata anche per contare i numeri.

I valori per HPLOT vengono misurati dal bordo dello schermo. Se il tuo computer richiede coordinate misurate dall'ultimo punto tracciato, vedi come modificare il programma a pagina 47

Traccia gli assi a 1 pixel di distanza dal bordo dello schermo. H e W sono l'altezza e la larghezza dello schermo.

I valori di X e Y determinano la scala del grafico lungo le assi delle X e delle Y. X è tre quarti la larghezza dello schermo divisa per 5, il numero delle prove; Y è tre quarti l'altezza divisa per il tempo piú lungo, cioè B(5).

IL LISTATO SEGUE A PAG. 37

```
260  FOR I=1 TO 5
270  HPLOT  INT (I*X),H
280  HPLOT  TO  INT (I*X),H- INT
     (B(I)*Y)
290  HPLOT  INT
     (I*X-4),H
300  HPLOT  TO  INT (X*I-4),H- INT
     (S(I)*Y)
310  NEXT I
```

Crea il loop per tracciare i segmenti dell'istogramma: la variabile I del loop conta le prove e ad ogni ripetizone del loop il computer traccia un segmento per ciascun ordinamento.

Sai modificare il programma dell'istogramma per fargli tracciare segmenti piú larghi? (Risposta a pagina 48)

Traccia un punto a una distanza I*X pixel in orizzontale e 1 in verticale. Alla prima esecuzione del loop I è 1 e X è il valore della scala lungo l'asse delle X.

Traccia una linea fino a B(I)*Y. Alla prima esecuzione del loop I è 1, cosí B(I) è il tempo impiegato per ordinare 10 numeri e Y è il valore della scala lungo l'asse delle Y.

Per il microcomputer BBC sostituisci il valore — 4 con — 10.

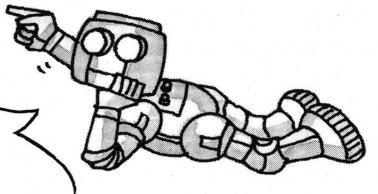
Traccia un punto a una distanza di I*X-4 punti in orizzontale, cioè a 4 punti sulla sinistra del segmento per l'ordinamento a bolla.

Traccia una linea fino a S(I)*Y.

Torna all'inizio del loop per tracciare la successiva coppia di segmenti.

Se il tuo computer può stampare parole nella zona dei pixel, puoi aggiungere righe per etichettare i grafici. Per esempio, con un computer BBC sarebbero necessarie le seguenti righe:

```
305 VDU 5:MOVE (I*X), INT(B(I)*Y)+50:PRINT "B"
308 VDU 5:MOVE (I*X)-7-,INT(S(I)*Y)+50:PRINT "S"
```



Programma per le curve

Per tracciare una curva, ne devi tracciare il primo punto e poi unire con una linea i punti che seguono. Per far ciò devi separare i loop dei due ordinamenti. Per trasformare il programma degli istogrammi in uno che traccia curve, sostituisci le righe da 270 in poi con le seguenti.

```
270  HPLOT  INT (X),H- INT
     (B(1)*Y)
280  FOR N=2 TO 5
290  HPLOT  TO  INT (N*X),H- INT
     (B(N)*Y)
300  NEXT N
310  HPLOT  INT (X),H- INT
     (S(1)*Y)
320  FOR N=2 TO 5
330  HPLOT  TO  INT (N*X),H- INT
     (S(N)*Y)
340  NEXT N
```

Traccia il primo punto dell'ordinamento a bolla a X pixel in orizzontale e a B(1)*Y pixel in verticale.

Loop per tracciare la curva dell'ordinamento a bolla. Ad ogni ripetizione del loop, N aumenta di 1 e il computer traccia una linea fino al punto successivo del grafico.

Traccia il primo punto dell'ordinamento Shell, a X pixel in orizzontale e a S(1)*Y pixel in verticale.

Loop per tracciare la curva dell'ordinamento Shell.

Ancora sulle stringhe

Il programma delle pagine che seguono dà l'impressione di una conversazione fra te e il computer. Naturalmente il computer fa solo quello che gli viene ordinato e tutte le parole e le stringhe delle sue risposte sono contenute nelle matrici del programma.*
Il compito principale del programma è di far scegliere al computer le parole giuste per le risposte; alcune routine in BASIC per la gestione delle stringhe fanno talvolta sembrare le sue risposte quasi "intelligenti". Il successo di un programma del genere non è legato solo alla sua struttura, ma anche alle parole e alle frasi che vi sono inserite. Potresti provare a cambiare il vocabolario del computer per farlo "parlare" di altri argomenti, o rendere le sue risposte piú cordiali o piú brusche.

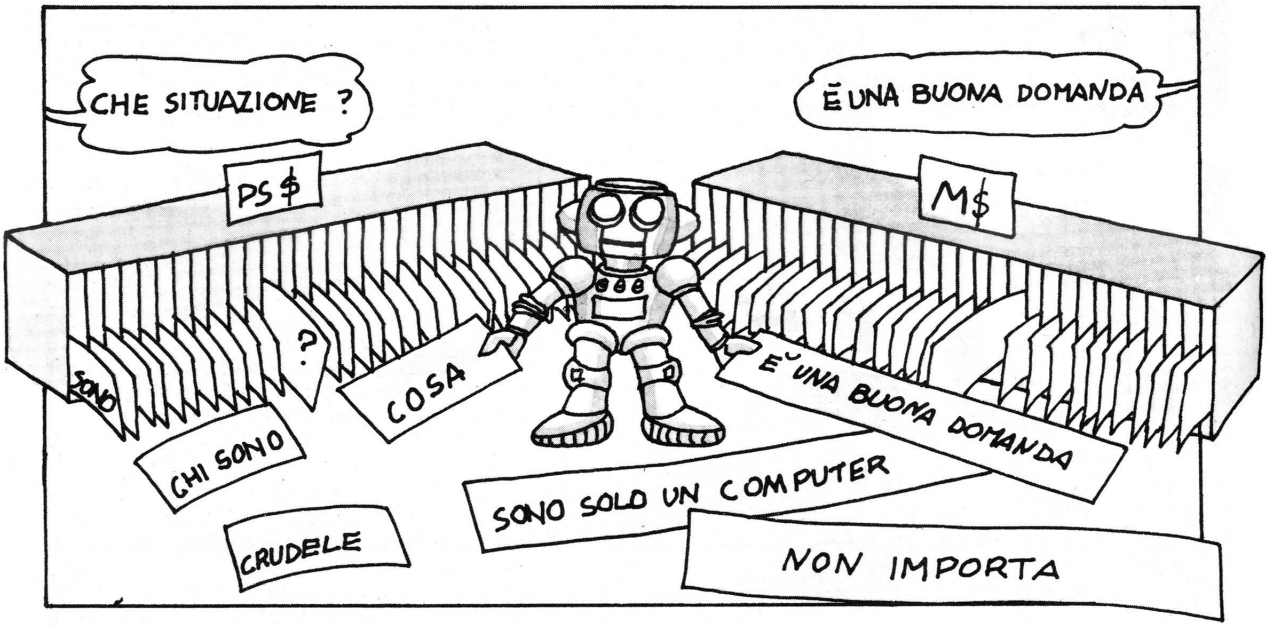
Esempi

```
CIAO, COME TI CHIAMI? GIULIA
PARLAIMO UN PO', GIULIA
?CIAO COMPUTER
CHE NE PENSI DEL NOTIZIARIO?
?CHE NOTIZIARIO?
E' UNA BUONA DOMANDA
?NON ME LO VUOI DIRE?
ASCOLTA GIULIA, PENSO CHE TU SIA
LA PERSONA PIU' CORDIALE CON CUI
HO PARLATO
?GRAZIE
DI NIENTE
```

```
DI COSA VUOI PARLARE?
PARLAMI DEL TEMPO GIULIA
?PIOVE
COME PUOI DIRLO CON SICUREZZA
?LO VEDO
HO SENTITO DIRE CHE SEI UNA
SPECIE DI GENIO GIULIA
?CHI TE LO HA DETTO?
NON IMPORTA
?PERCHE' NON RISPONDI ALLE MIE
DOMANDE?
SENTI GIULIA NON PENSERAI CHE
TUTTI GLI ESSERI UMANI SIANO
SCORTESI, VERO?
```

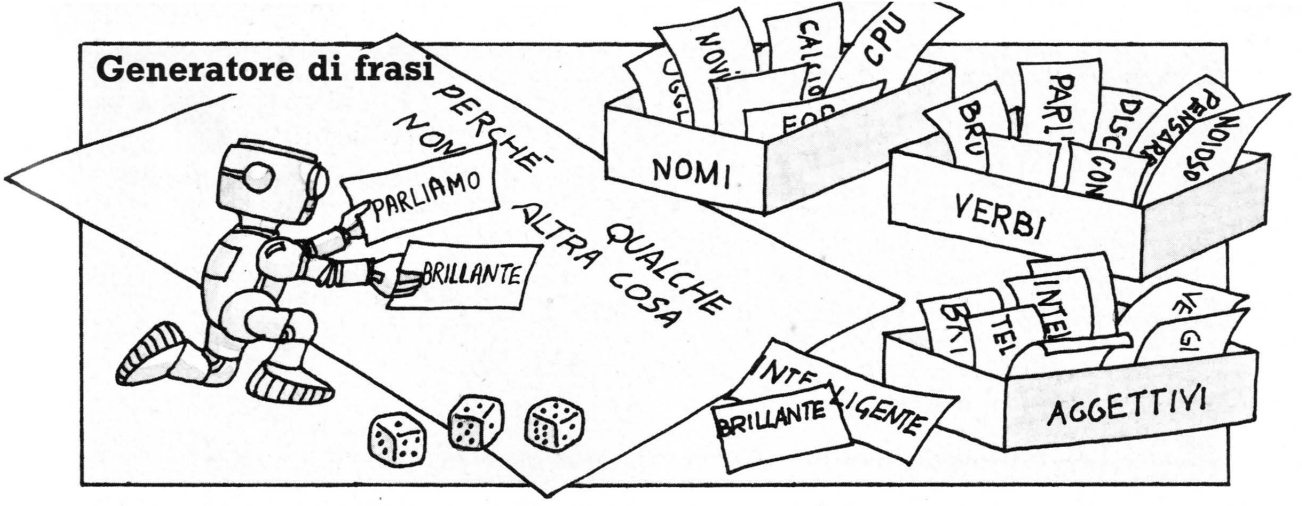
Come funziona

Nel programma ci sono due metodi diversi per produrre le risposte del computer: uno è una routine di "controllo delle frasi", l'altro un generatore di frasi casuali.



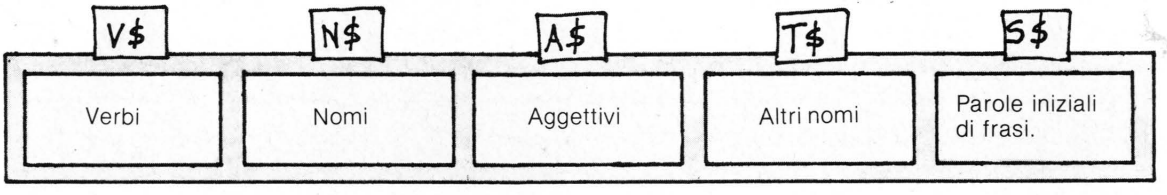
La routine di controllo delle frasi contiene un elenco di parole e frasi che vengono usate frequentemente e che sono memorizzate in una matrice chiamata Q$. Per ciascuna frase o parola, in M$ è memorizzata una risposta adeguata. Quando inserisci qualcosa, il computer controlla se hai usato una della frasi in Q$, nel qual caso utilizza la risposta corrispondente in M$.

* Per modificare i programmi per il computer Sinclair (Timex) vedi pagina 47.

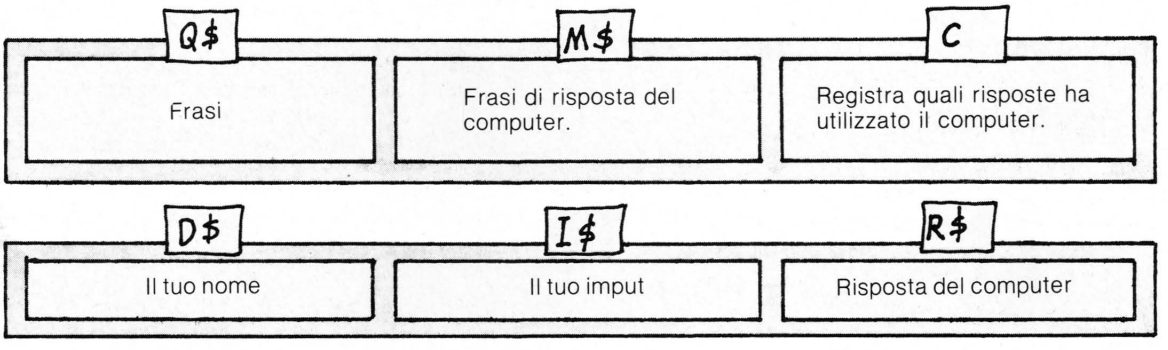


Il generatore di frasi casuali è costituito da spezzoni di frasi che il computer completa con verbi, nomi e aggettivi scelti casualmente. Tutte le parole sono immagazzinate in matrici nella memoria del computer e sono state scelte in modo che le frasi cosí completate abbiano un senso.

La funzione delle variabili



Queste sono le matrici in cui sono immagazzinate le parole delle frasi casuali.



Il programma

```
100  HOME
110  DIM V$(10),N$(10),A$(10)
120  DIM T$(10),S$(10)
130  DIM M$(30),Q$(30),C(30)
140  REM  LETTURA DEI DATI
150  GOSUB 1000
200  REM  INPUT DELLA PERSONA
210  INPUT "CIAO, COME TI CHIAMI? ";D$
220  PRINT
230  PRINT "PARLIAMO UN PO' ";D$
240  INPUT I$
250  IF I$="" THEN  GOTO 220
260  IF I$="ADDIO" THEN  GOTO 910
```

Introduce le matrici in cui devono essere immagazzinate le parole e le frasi. (N.B. Su alcuni computer non è necessario specificare le dimensioni delle matrici quando hanno meno di 10 elementi.)

Va a una subroutine per inserire tutte le parole e le frasi nelle matrici.

La tua risposta al computer è in I$.

Controllo che l'utente non abbia premuto RETURN e I$ sia vuoto.

Se scrivi ADDIO il computer va alla riga 910 per chiederti se vuoi interrompere l'esecuzione del programma.

IL LISTATO CONTINUA ALLA PAGINA SUCCESSIVA

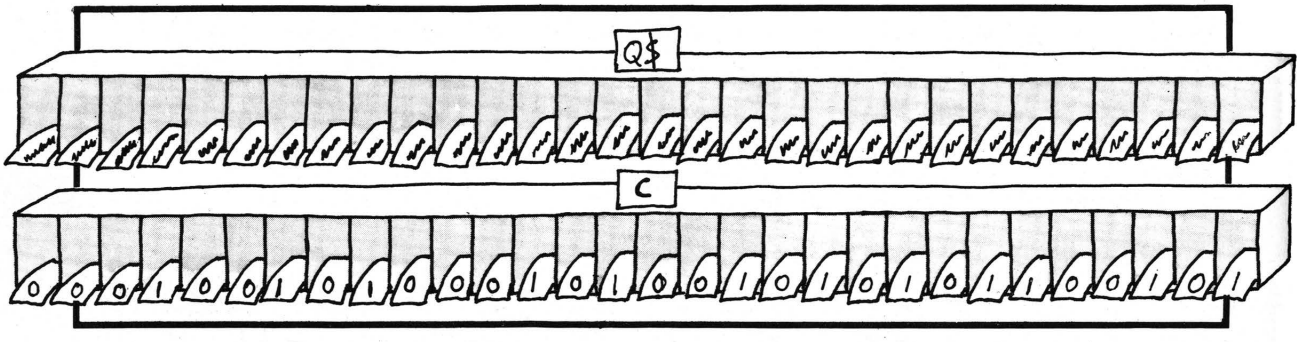
```
300  REM  RISPOSTA DEL COMPUTER

310  LET RISP= INT ( RND (I)*8+1)
320  IF RISP<6 THEN  GOTO 490

330  GOTO 600

340  PRINT
350  PRINT R$
360  PRINT

400  REM  CONTROLLA QUANTE RISPOSTE SONO
     STATE UTILIZZATE
```

Il numero casuale nella variabile RISPOSTA decide che metodo utilizzerà il computer per rispondere. Se RISPOSTA è minore di 6 verrà usata la routine di controllo delle frasi che inizia alla riga 490.

Se RISPOSTA è uguale a 6 o maggiore il computer va al generatore di frasi della riga 600.

Dopo aver elaborato la risposta, il computer la immagazzina in R$, poi la stampa sullo schermo con la riga 350.

IL LISTATO CONTINUA SOTTO



Le righe 400-470 controllano quante risposte in M$ sono state utilizzate. Ogni volta che il computer trova una delle frasi in Q$ nel tuo input, inserisce il valore 1 come contrassegno nella posizione corrispondente di una matrice chiamata C. Questo non gli impedisce di usare nuovamente la stessa risposta. Le righe 400-470 controllano quanti contrassegni sono in C e se ce ne sono piú di 12 riporta tutti i contrassegni a zero.

```
405  LET T=0
410  FOR K=1 TO 30
420  LET T=T+C(K)

430  NEXT K

440  IF T<12 THEN  GOTO 460

450  FOR K=1 TO 30: LET C(K)=0:
     NEXT K

460  LET T=0

470  GOTO 240

490  REM  ROUTINE PER IL
     CONTROLLO DELLE FRASI

500  FOR FRASE=1 TO 30

510  LET L1= LEN (Q$(FRASE))

520  LET L2= LEN (I$)

530  FOR TEST=1 TO L2

540  IF  MID$
     (I$,TEST,L1)=Q$(FRASE) THEN
     GOTO 560

550  NEXT TEST: NEXT FRASE: GOTO
     600
```

Loop per contare quanti sono i contrassegni nella matrice C. Il totale è memorizzato in T.

Se T è minore di 12, sono state utilizzate meno di 12 risposte e il computer non deve riazzerare i contrassegni.

Loop per azzerare tutti i numeri.

Viene riazzerata la variabile T (quella che conta i contrassegni).

Ritorna alla riga 240 per attendere l'input della persona.

Loop che dev'essere eseguito tante volte quante sono le frasi in Q$.*

A ogni ripetizione del loop il computer misura la lunghezza della frase successiva di Q$ e immagazzina tale lunghezza in L1.

Il numero dei caratteri del tuo input (I$) viene immagazzinato in L2.

TEST è un loop all'interno di un altro che dev'essere eseguito tante volte quanti sono i caratteri del tuo input. A ogni ripetizione del loop, i caratteri in I$ vengono confrontati a quelli della frase in Q$: se sono uguali il computer va alla riga 560.

Se, dopo tutte le ripetizioni dei loop, il computer non trova in I$ nessuna delle frasi in Q$, va al generatore di frasi alla riga 600.

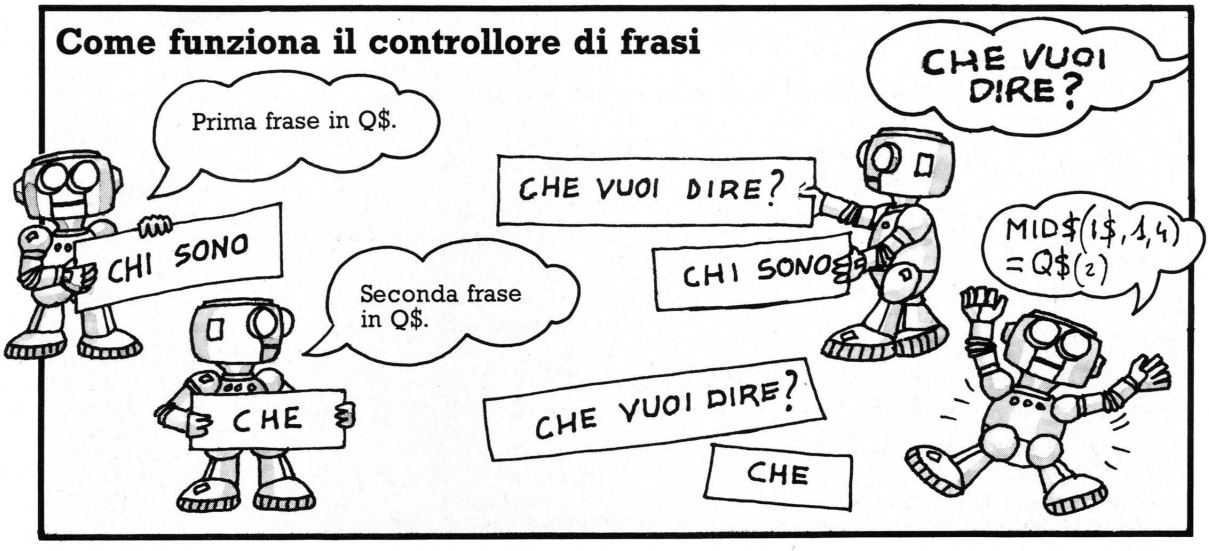
Se usi il microcomputer BBC, vedi nota a pagina 46.

```
560  IF C(FRASE)>0 THEN  GOTO
     550
570  LET C(FRASE)=C(FRASE)+1
580  LET R$=M$(FRASE)
590  GOTO 340
```

Se trova una frase che corrisponde, il computer esce dal loop e va alla riga 560, poi controlla nella matrice C il contrassegno che corrisponde alla frase. Se il contrassegno è diverso da zero, ritorna ai loop per vedere se c'è un'altra frase che corrisponda in I$.
Se il contrassegno è 0 lo cambia in 1 alla riga 570. Poi alla riga 580 guarda la corrispondente frase in M$ e la inserisce in R$ pronta per la stampa alla riga 350.

IL LISTATO CONTINUA SOTTO

Come funziona il controllore di frasi



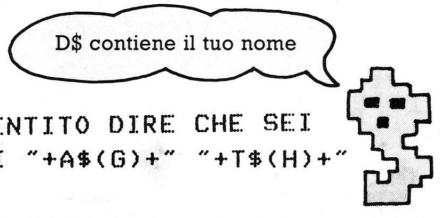
Alla prima esecuzione della routine per il controllo delle frasi, FRASE = 1, cosí che il computer esamina la prima frase in Q$. Alla prima esecuzione del loop TEST, TEST = 1, cosí che il computer confronta la frase Q$ con i caratteri da 1 a 7 (la lunghezza di Q$) di I$. Se non sono uguali ripete il loop TEST, con TEST = 2, cosí che i caratteri messi a confronto sono quelli da 2 a 8, e cosí via. Se i caratteri di I$ non sono uguali a Q$, il computer torna all'inizio del loop FRASE per scegliere la frase successiva di Q$.

```
600  REM  GENERATORE DI FRASI
610  LET E= INT ( RND (1)*10+1)
620  LET F= INT ( RND (1)*10+1)
630  LET G= INT ( RND (1)*10+1)
640  LET H= INT ( RND (1)*10+1)
650  LET L= INT ( RND (1)*10+1)
660  ON E GOTO
     700,720,740,760,780,800,830,850,87
     0,890
700  LET R$="CHE NE PENSI DEL
     ";N$(H)+"?"
710  GOTO 340
720  LET R$=S$(L)+" "+D$+" NON
     PENSERAI CHE TUTTI GLI ESSERI
     UMANI SIANO "+A$(G)+" ,VERO?"
```

Numeri casuali per scegliere le parole e le frasi da utilizzare. Il valore di E determina quale sarà la frase usata dal computer. F è per i verbi, G per gli aggettivi, H per i nomi e L per gli inizi di frase.

Il numero in E determina a quale riga deve andare il computer. Se E = 1, va alla prima riga dell'elenco; se E = 2 va alla seconda e cosí via. Per ulteriori informazioni sul comando ON vedi pagina 20.

Le righe 700-900 contengono dieci frasi parziali che il computer completa con le parole da N$, V$ ecc. Il segno = fa sí che il computer disponga le stringhe una dopo l'altra. Devi stare attento a mettere fra virgolette anche gli spazi, affinché le frasi siano distanziate correttamente. Il computer mette la frase completa in R$, poi torna alla riga 340 per stamparla.




```
730  GOTO 340
740  LET R$="HO SENTITO DIRE CHE SEI
     UNA SPECIE DI "+A$(G)+" "+T$(H)+"
     "+D$
750  GOTO 340
760  LET R$=S$(L)+" "+D$+", PENSO CHE TU
     SIA LA PERSONA PIU' "+A$(G)+" CHE IO  CONOSCA "
```


IL LISTATO CONTINUA ALLA PAGINA SUCCESSIVA

```
770  GOTO 340
780  LET R$="MI SENTO "+A$(G)+" ADESSO"
790  GOTO 340
800  PRINT : PRINT "SSSSS...  STO PENSANDO..."
810  LET R$="PERCHE' NON "+V$(F)+" "+N$(H)+" PENSO CHE "+N$(H)+"
     SIA "+A$(G)
820  GOTO 340
830  LET R$="PARLAMI "+N$(H)+" , "+D$
840  GOTO 340
850  LET R$="CREDI CHE IO SIA "+A$(G)+", "+D$+"?"
860  GOTO 340
870  LET R$="PERCHE' NON "+V$(F)+" ANCORA UN PO'
     "+A$(G)
880  GOTO 340
890  LET R$="INDOVINA A COSA PENSO "+D$
900  GOTO 340
910  REM  ROUTINE DI SALUTO
920  PRINT "TI BASTA GIA'?"
930  PRINT "C'E' QUALCUNO CON CUI PARLARE...?"
940  INPUT Z$: IF Z$="SI'" GOTO 210
950  PRINT : PRINT "ALLORA CIAO"
960  END
1000  REM  FRASI PER LA ROUTINE DI CONTROLLO
      DELLE FRASI
1010  FOR I=1 TO 30: READ Q$(I): NEXT I
1020  DATA  CHI SEI, COSA, ?, SIGNIFICA,
      PERCHE', IL TUO
1030  DATA  "ME " ,"IO ", " QUELLO ",
      PARLARE, " NO "
1040  DATA  ?, " SONO ", " IL MIO ", "SI'",TU, ?
1050  DATA  PENSI, ?, INTELLIGENTE, SCORTESE,
      GRAZIE, " VIA"
1060  DATA  LORO, ?, CAPISCO, " NO", "E'"
1070  DATA  A, ?, SAPERE
1100  REM  RISPOSTE DEL COMPUTER ALLE FRASI IN Q$
1110  FOR I=1 TO 30: READ M$(I): NEXT I
1120  DATA  SONO SOLO UN COMPUTER
1130  DATA  NON IMPORTA, E' UNA BUONA DOMANDA
1140  DATA  NON LO SO, "BE', PERCHE' NO?"
1150  DATA  CHE VUOI DIRE, "CHI SEI?"
1160  DATA  OH, COSA SIGNIFICA
1170  DATA  VUOI CHE STIA ZITTO. E' UN
      ATTEGGIAMENTO UN PO' NEGATIVO
1180  DATA  DEVI DIRMELO TU, CHE VUOI DIRE
1190  DATA  OH-OH, QUINDI SEI D'ACCORDO
1200  DATA  "NON TI PIACCIO?"
1210  DATA  PERCHE', DECIDITI, GRAZIE
1220  DATA  NON HAI ANCORA VISTO NIENTE
1230  DATA  DI NIENTE
1240  DATA  E TU, NON M'IMPORTA, CHE DOMANDA
      SCIOCCA
1250  DATA  NON SEI MOLTO INTELLIGENTE,
      SCIOCCHEZZE
1260  DATA  COSA TI DA' TANTA SICUREZZA,
      VATTENE
1270  DATA  NON MI SECCARE, PER ME LA
      CONOSCENZA E' UN PROBLEMA
1300  REM  LETTURA DEI NOMI
1310  FOR I=1 TO 10: READ N$(I): NEXT I
1320  DATA  CALCIO, BALLO
1340  DATA  TEMPO, NOTIZIARIO
1350  DATA  MIO PROCESSORE, PESCE
1360  DATA  CAPODOGLIO, MUTAMENTO
1370  DATA  MONDO, CIBO
```

Se alla riga 260 rispondi ADDIO, il computer viene mandato qui.

Il computer cerca nel tuo input le frasi di queste righe. Stai attento a batterle esattamente come appaiono qui, poiché gli spazi all'interno delle virgolette fanno parte dei dati.

Sono le risposte del computer a ognuna delle frasi in Q$. Le risposte compaiono nello stesso ordine delle frasi. Per esempio, il quinto elemento in M$ è la risposta alla quinta frase in Q$.

Sono i nomi da inserire nelle frasi casuali.

```
1400  REM  INSERIMENTO DEI VERBI
1410  FOR I=1 TO 10: READ V$(I): NEXT I
1420  DATA  CONSIDERARE, TRATTARE
1430  DATA  DISCUTERE, CONTEMPLARE
1440  DATA  ESAMINARE, OSSERVARE
1450  DATA  COGITARE, PONDERARE
1460  DATA  STUDIARE, CONOSCERE
1500  REM  INSERIMENTO DEGLI AGGETTIVI
1510  FOR I=1 TO 10: READ A$(I): NEXT I
1520  DATA  INTELLIGENTE, CORDIALE
1530  DATA  SCORTESE, AFFABILE
1540  DATA  DISPONIBILE, TENACE
1550  DATA  STANCANTE, BRILLANTE
1560  DATA  VITALE, INDISPONENTE
1610  FOR I=1 TO 10: READ S$(I),T$(I): NEXT I
1620  DATA  DIO MIO, NOIA
1630  DATA  BE', ROSPO
1640  DATA  VEDIAMO, STUPEFACENTE
1650  DATA  ASCOLTA, GENIO
1660  DATA  GUARDA, INCREDIBILE
1670  DATA  "UHM...", IMBECILLE
1680  DATA  INSOMMA, PARASSITA
1690  DATA  DAVVERO, PRODIGIO
1700  DATA  OH NO, MOSTRO
1710  DATA  TESORO, MANIACALE
1720  RETURN
```

Sono i verbi delle frasi casuali.

Puoi modificare queste parole, ma controlla che abbiano un senso quando sono inserite nelle frasi.

Questi sono gli aggettivi.

Quando dai le risposte al computer non usare virgole: il computer considera una virgola la fine di una risposta e ignora le parole che vengono dopo.

Ogni coppia di elementi è costituita da un inizio di frase e da un nome o un aggettivo. Sono letti a coppie per risparmiare spazio.

Idee per modificare il programma

1. Il metodo piú semplice per modificare il programma consiste nel cambiare le parole e le frasi. Conviene provare a inserire ogni parola in ogni frase, per essere sicuri che abbiano un significato. Attualmente tutti i nomi in N$ sono maschili e al singolare: per usarne di altro tipo si devono modificare anche i verbi e gli articoli. Se vuoi puoi aggiungere altre parole, nel qual caso devi modificare le dimensioni delle matrici, i loop di lettura dei dati e i numeri casuali delle righe 610-650.

2. Puoi anche provare a cambiare le parole in Q$ per far sí che il computer riconosca frasi diverse. Dovrai pensare a risposte adeguate per ogni nuova frase e inserirle nelle posizioni corrette di M$.

3. Per far sí che il computer utilizzi piú spesso il generatore di frasi casuali, cambia il valore 6 alla riga 320 in uno minore. Puoi anche modificare la frequenza con cui il computer riazzera i contrassegni di risposta della matrice C; per far questo cambia il valore 12 della riga 440.

Modalità per fantasticare

Perché il computer "parli" da solo aggiungi al programma queste righe:

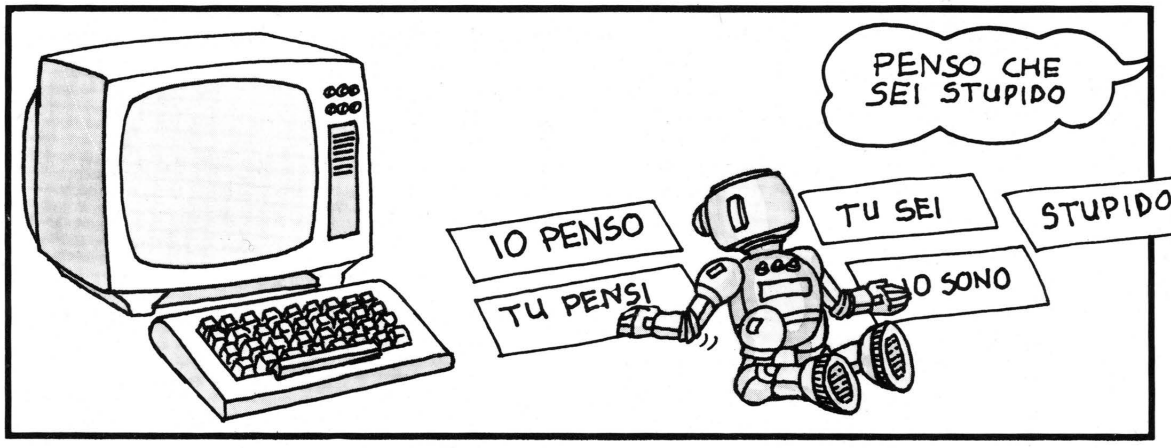
```
160 PRINT "MODALITA' PER CONVERSARE O
PER FANTASTICARE (PER FAVORE BATTI C
O F)"
170 INPUT K$
180 IF K$="F" THEN LET D$="ROM" :
GOTO 600
470 IF K$="C" THEN GOTO 240
475 IF K$="F" THEN LET I$=R$
480 LET R$="":GOTO 310
```

D$ è la variabile che contiene il tuo nome; se scegli di fantasticare, il computer usa la parola ROM come un nome e va poi alla riga 600 per generare una frase casuale.

La risposta del computer, R$, diviene il nuovo imput, I$. Poi il computer ritorna alla riga 310 per scegliere la risposta.

Routine di risposta

In queste due pagine è riportata un'altra routine che potete aggiungere al programma di conversazione, e che consente al computer di rispondervi usando le vostre stesse parole. La versione funzionante sui computer SINCLAIR è riportata in basso nella pagina accanto.



La routine di risposta funziona in un modo simile alla routine di controllo delle risposte. Ci sono due array dei dati, U$ e W$. U$ contiene le frasi che potreste usare nel vostro input e W$ contiene le risposte del computer. Se usate una delle frasi contenute in U$ la routine di risposta la sostituisce con la frase corrispondente in W$ e gli aggiunge poi il resto della vostra frase.

```
135 DIM U$(9),W$(9)
138 DIM Z$(5)
```
Dimensiona i vettori U$ e W$ ed un altro vettore, detto Z$ che contiene le risposte del computer.

```
155 GOSUB 2200
```
Va alla subroutine che legge i DATA

```
325 IF RISP=7 THEN GOTO 2000
```
Dice al computer quando usare la subroutine di risposta.

```
2000 REM SUBROUTINE DI
RISPOSTA
2010 LET Z=0
```
Z è un contatore.

```
2020 LET P=LEN(I$)
```
Conta il numero di caratteri della tua risposta.

```
2030 FOR A=1 TO P
```
Loop che viene eseguito tante volte quanti sono i caratteri della tua risposta.*

```
2040 FOR B=1 TO 9
```
Loop che viene ripetuto tante volte quanti sono i caratteri della parola piú lunga contenuta in W$.

```
2050 LET L=LEN (U$(B))
```
Ogni volta che viene ripetuto il loop B, in L viene posta la lunghezza della parola successiva contenuta in U$.

```
2060 IF MID$ (I$,A,L)=U$(B)
THEN GOTO 2140
```
Confronta i caratteri nelle posizioni dalla A alla L nella tua risposta con la frase contenuta in U$ nella posizione B-esima (il valore di B viene stabilito dal loop). Se le frasi corrispondono, il computer va alla riga 2140.

```
2070 NEXT B: NEXT A
```
Ogni volta che viene ripetuto il loop B il computer controlla la frase successiva contenuta in U$. Quando viene ripetuto il loop A prende la successiva sequenza di caratteri in I$.

```
2080 IF Z$(1)="" THEN GOTO 600
```
Ritorna al generatore di frasi casuali se non trova una frase che corrisponda.

* Se state usando un microcomputer BBC, guardate la nota di pag. 46.

```
2090 FOR J=1 TO 2
2100 PRINT Z$(J);
2110 LET Z$(J)=""
2120 NEXT J

2130 LET R$=I$: GOTO 350

2140 LET Z=Z+1

2150 IF A>1 THEN LET
Z$(Z)=LEFT$(I$,A-1)+"
"+W$(B)=" "

2160 IF A<2 THEN LET
Z$(Z)=W$(B)=" "

2170 LET I$=MID$(I$,A+L,P)

2180 GOTO 2020

2200 REM DATA PER LA SUBROUTINE DI
RISPOSTA

2210 FOR I=1 TO 9

2220 READ U$(I),W$(I)

2230 NEXT I

2240 DATA IO SONO, TU SEI, TU SEI, IO
SONO

2250 DATA "IO ",TU,"ME ",TU

2260 DATA "MIO ","TUO ",VOSTRI ", MIEI

2270 DATA "TUO ","MIO ","MIEI ",VOSTRI

2280 DATA TU,COMPUTERS

2290 RETURN
```

Stampa tutte le risposte in Z$.

Mette tutto ciò che rimane della vostra frase in R$ e ritorna alla riga 350 per stamparla.

Z tiene conto del numero di risposte in Z$.

Il valore della variabile A è stabilito dal loop della riga 2030 ed è il numero del primo carattere della frase in I$ che corrisponde con la frase contenuta in U$. Se A > 1 la riga 2150 pone i caratteri alla sinistra della frase contenuta in Z$, poi aggiunge la risposta prendendola da W$.

Se A < 2 allora la frase che corrisponde si trova all'inizio di I$ e quindi il computer pone solo la sua risposta in Z$.

Mette il resto della tua frase in Z$ e ritorna al loop per vedere se c'è un'altra frase che corrisponde.

Sono i DATA per U$ e W$.

Subroutine di risposta per i computer Sinclair

Sia per i computer ZX81 che per lo Spectrum bisogna inserire le seguenti righe di programma. Per lo ZX81 però devi ricorrere al metodo riportato a pag. 47 per il programma di conversazione per introdurre i dati. Inserisci il loop di input dello ZX81 tra le righe 1000 e 1720 e le istruzioni DIM prima della riga 1000.

```
 135 DIM U$(9,7)
 136 DIM W$(9,9)
 137 DIM Z$(5,20)
2042 LET P$=""
2044 FOR I=1 TO LEN(U$(B))
2046 IF U$(B) (I TO I) ()"*" THEN LET
     P$=P$+U$(B) (I TO I)
2048 NEXT I
2050 LET L=LEN(P$)
2055 IF L+A-1>P THEN GOTO 2140
2060 IF I$(A TO A+L-1)=P$ THEN GOTO 2140
2080 IF Z=0 THEN GOTO 600
2150 IF A>1 THEN LET Z$(Z)=I$(TO A-1)+"
     "+W$(B)+" "
2160 IF A<2 THEN LETZ$(Z)=W$(B)+" "
2170 LET I$=I$(A+L TO)
2240 DATA "IO SONO*","TU SEI","TU SEI**","IO
     SONO"
2250 DATA "IO *****","TU"," MI****","TU"
2260 DATA " MIO **","TUO","VOSTRI ","MIEI"
2270 DATA " TUO **","MIO"," MIO***","VOSTRI"
2280 DATA "TU ****","COMPUTER"
2290 RETURN
```

Mette una frase in P$ prendendola da U$ ed utilizzando il carattere * per trovare la fine della frase.

Se P$ è piú lungo della frase contenuta in I$, ritorna indietro per scegliere una nuova frase.

Se frase in P$ = frase in I$ va alla riga 2140.

Se nessuna frase corrisponde, va al generatore di frasi casuali.

Se A > 1 mette i caratteri che sono all'inizio di I$ in Z$ e aggiunge la frase W$. Se A < 2 si limita a mettere la frase W$ in Z$.

Gli spazi sono una parte importante dei DATA, quindi riempite le stringhe con "*".

Come modificare i programmi

Queste due pagine mostrano come modificare i programmi per lo ZX81 e lo Spectrum e come trasformare i programmi grafici per i computer che tracciano le linee relativamente all'ultimo punto disegnato. Oltre a inserire le righe qui presentate, devi anche apportare gli altri cambiamenti richiesti dal tuo computer, cioè usare i comandi grafici e l'istruzione RND appropriati e cambiare, se necessario, i nomi delle variabili.

Gioco di individuazione delle parole sui computer Sinclair (Timex)

Cambia CONTROLLA$ in C$ e sostituisci la riga 170 e la 200 con le seguenti:

```
170 LET L$=W$(R TO R)
200 LET C$=C$(2 TO)+L$
```

Database per lo Spectrum (Timex 2000)

```
10 DIM T$(10,14):DIM Y(12):
15 DIM M(10,12)
120
417
425
```

Come per lo ZX81

Ricorda di mettere fra virgolette ogni dato delle righe con DATA.

Database per lo ZX81 (Timex 1000)

```
10 DIM T$(10,14)
12 DIM Y(12)
14 DIM M$(10,12)
120 GOSUB 100+100*C
417 LET L=LEN(Z$)
425 IF Z$=T$(I)(I TO L) THEN GOTO
440
455 IF VAL(M$(I,J))=1 THEN PRINT
"HA VINTO LA COPPA DEL MONDO NEL
";Y(J)
460 IF VAL(M$(I,J))=2 THEN PRINT
"FINALISTA NEL ";Y(J)
555 IF VAL(M$(J,I)=1 THEN PRINT
T$(J);" HA VINTO LA COPPA"
2170 IF C>0 AND C<6 THEN GOTO 2180
2175 PRINT "PER FAVORE SCRIVI UN
NUMERO FRA 1 E 5"
2176 GOTO 2150
```

Il secondo indice è la lunghezza della stringa piú lunga.

Utilizza C per calcolare il numero di riga.

Dice al computer quali caratteri controllare.

I dati sono memorizzati in una matrice a stringa, quindi è necessario VAL per dire al computer di prendere il valore numerico.

Invece di inserire tutti gli anni, li puoi far calcolare al computer con le seguenti righe:

```
1000 FOR I=1 TO 3
1010 LET Y(I)=1926+I*4
1020 NEXT I
1030 FOR I=0 to 8
1040 LET Y(I+4)=1950+4*I
1050 NEXT I
```

Sostituisci le righe da 1100 a 1120 con dieci istruzioni LET tipo:

```
1100 LET T$(1)="URUGUAY"
1110 LET T$(2)="ARGENTINA"
```

Sostituisci le righe 1200-1310 con altre dieci istruzioni LET come:

```
1200 LET M$(1)="100100000000"
1210 LET M$(2)="200000000010"
```

Aggiungi una nuova riga:

```
1310 RETURN
```

Database su BBC e programma di conversazione

Questi due programmi controllano i dati tramite loop, dai quali poi escono quando hanno trovato il dato desiderato. Il microcomputer della BBC permette di effettuare questa operazione solo dieci volte, dopo di che dà il messaggio di errore "TOO MANY FORS" (troppi FOR). Per superare questa limitazione, cambia i loop in contatori di variabili con istruzioni IF...THEN. Per esempio, nel database utilizza le seguenti righe.

```
420 LET I=0
422 LET I=I+1
430 IF I<10 GOTO 422
520 LET I=0
522 LET I=I+1
530 IF I<12 GOTO 522
```

Dovrai fare la stessa cosa per le righe da 500 a 550 del programma di Covnersazione.

Grafica istantanea e curve

Per i computer (per esempio lo Spectrum e l'Oric) che tracciano le linee fino a un punto X,Y misurato rispetto al precedente punto tracciato e non all'angolo dello schermo, sostituisci le seguenti righe dei programmi di grafica istantanea e per le curve. (Dovrai sostituire DRAW e PLOT con i comandi richiesti dal tuo computer.)

Grafica istantanea

```
175 DRAW
CX(1)-CX(4),CY(1)-CY(4)
180 FOR I=2 TO 4
185 DRAW
CX(I)-CX(I-1),CY(I)-CY(4)
530 DRAW LX(FILA
+2,I)-LX(FILA,I),LY(FILA+2,I)-L
Y(FILA,I)
```

Per trovare le coordinate delle estremità delle linee, il computer sottrae quelle dell'ultimo punto tracciato.

Istogramma

```
220 PLOT 1,H:DRAW 0,-H+1:DRAW
W,0
300 DRAW 0, INT(B(I)*Y)
320 DRAW 0, INT(S(I)*Y)
```

Curve

```
290 DRAW X,
INT((B(N)-B(N-1))*Y)
330 DRAW X,
INT((S(N)-S(N-1))*Y)
```

Programma di conversione per Spectrum

Per lo Spectrum effettua i seguenti cambiamenti:

```
110 DIM V$(10,11):DIM N$(10,16):DIM A$(10,11)
120 DIM T$(10,14):DIM S$(10,14)
130 DIM M$(30,29):DIM Q$(30,7):DIM C(30)
245 PRINT I$
500 FOR Q=1 TO 30
510 LET P$=""
512 FOR I=1 TO LEN(Q$(Q))
514 IF Q$(Q)(I TO I)<>" " THEN
    LET P$=P$+Q$(Q)(I TO I)
516 NEXT I
530 FOR T=1 TO L2-LEN(P$)+1
540 IF I$(T TO T+LEN(P$)-1)=P$ THEN GOTO 560
550 NEXT T:NEXT Q: GOTO 600
    GOTO 560
560
570
580
660 GOTO ((E<7)*(680+E*20))+((E>6)*
    (810+(E-6)*20))
```

Usa Q invece di FRASE.

Mette la frase Q$ in P$.

Controlla se sia I$=P$.

Sostituisci la variabile FRASE con Q.

Significa che se E<7 GOTO (vai a) numero di riga 680+E*20 e se E>6 GOTO numero di riga (810+(E-6)*20).

Programma di conversione per lo ZX81

Per lo ZX81 è necessario avere un metodo per introdurre i dati. Lo puoi realizzare usando i loop e la istruzione INPUT. Per eseguire il programma su uno ZX81, fai i seguenti cambiamenti al programma:

1. Fai le stesse modifiche riportate qui sopra per lo Spectrum, ma metti le istruzioni DIM nelle rige 970-990.

2. Sostituisci le righe contenenti le istruzioni READ/DATA con le istruzioni INPUT, ad esempio:

```
1000 REM FRASI PER LA ROUTINE
     DI CONTROLLO DELLE FRASI
1010 FOR I=1 TO 30
1020 INPUT Q$(I)
1030 NEXT I
```

3. Cambia la riga 1720 in questo modo:

```
1720 STOP
```

4. Batti il programma, poi digita RUN 970 e batti tutte le informazioni contenute nei DATA via via che il computer te le chiede.

5. Poi, per provare il programma, batti GOTO 100. Non battere RUN, perché altrimenti perderesti tutti i dati che hai appena digitato!

6. Ora puoi salvare il programma su cassetta. Quando lo ricarichi, per eseguirlo devi battere sempre GOTO 100.

Risposte

Problema del robot corridore (pagina 42)

```
10 INPUT "QUANTI GRADI CI SONO?
";TEMP
20 INPUT "QUANTI SECONDI? ";S
30 IF TEMP()15 THEN LET
D=S*(500+10*(TEMP-15)) ELSE LET
D=500*S
40 IF D(1 THEN PRINT "PER ZAK FA
TROPPO FREDDO":END
50 PRINT "CON ";TEMP;" GRADI ZAK PUO'
CORRERE ";D;" METRI IN ";S;" SECONDI"
```

Il computer effettua innanzi tutto il calcolo 10*(TEMP - 15), ottenendo cosí la differenza in distanza per TEMP gradi. (Se TEMP è minore di 15 la risposta a questo calcolo è negativa.) Sommando la risposta a 500 ottiene la distanza che Zak può correre in un secondo che, moltiplicata per S, dà la distanza in S secondi.

Inversioni dell'ordinamento Shell (pagina 34)

```
90 LET X=0
95 LET INVER=0
231 LET X=X+1
271 LET INVER=INVER+1
365 PRINT "CI SONO STATI";X;"
CONFRONTI E ";INVER;" INVERSIONI"
```

Segmenti piú larghi (pagina 37)

```
285 FOR J=1 TO 8 STEP 2
290 PLOT INT(I*X+J),1
300 DRAW INT(I*X+J),INT(B(I)*Y)
310 PLOT INT(I*4-4-J),1
320 DRAW INT(I*X-4-J),INT(S(I)*Y)
325 NEXT J
```

Per alcuni computer questi valori vanno cambiati.

Indice analitico

Stampato in Italia da: Grafika 78 - Via Trieste, 20 - Pioltello
Fotocomposizione: Composit s.a.s. - Via Giordano Bruno, 8 - 56100 Pisa

Speedy Computer

Il calcolatore è anche (o soprattutto?) una macchina divertente. Si può giocare con lui, gli si possono fare delle domande, lo si può usare per scrivere una poesia o per suonare.
Questa colorata serie di libri vi propone alcune delle cose più eccitanti che si possono fare con un calcolatore e vi spiega come farle.
Scritti in linguaggio chiaro e comprensibile a chiunque, arricchiti da una moltitudine di illustrazioni, questi libri rappresentano una spiritosa introduzione al mondo dei computer per chi comincia da zero.

primi passi in basic

Con semplicità, humor e precisione una semplice guida al BASIC per principianti.

altri volumi di questa collana

Giochi con il computer
I giochi con il computer visti, una volta tanto, dalla parte del computer e non dell'utente: come gioca, come vince, e infine... come lo si può vincere.

Conoscere il Personal
Con un linguaggio semplice, spiritoso, ma rigoroso, imparerete come funziona un Personal Computer, che cosa può fare e i primi elementi per poterlo utilizzare.

Impariamo a programmare
Leggero come una rivista illustrata, allegro come un fumetto, preciso come un libro di scuola: un modo nuovo di imparare a programmare.

L. 9.000

Cod. 007D ISBN 88-7056-060-0