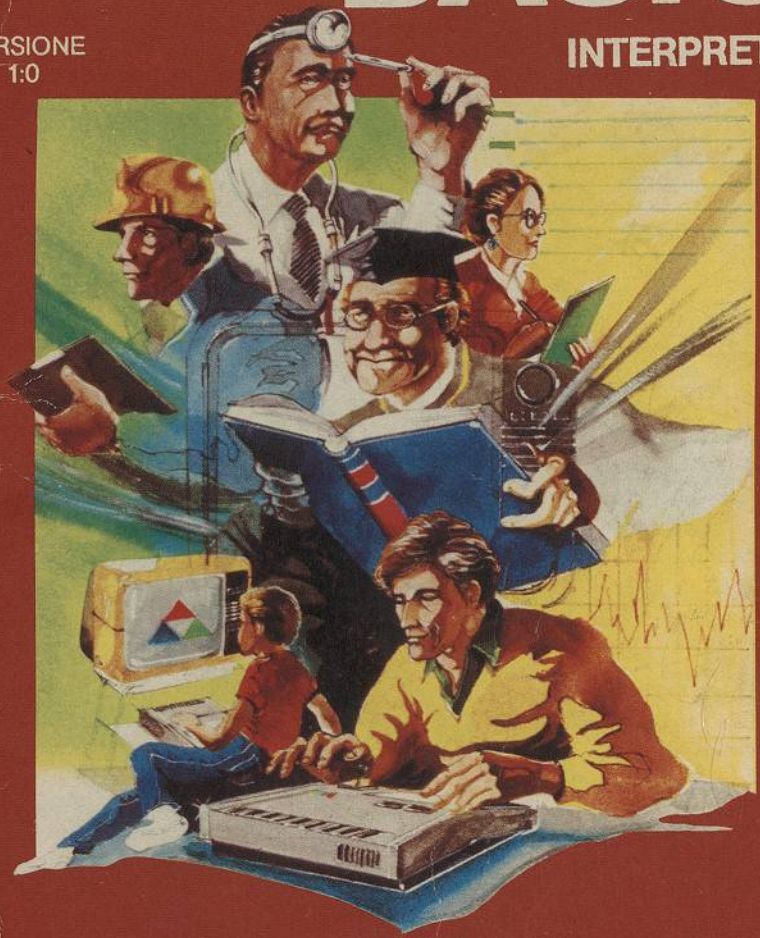
©CREATIVISION

BASIC

INTERPRETE

VERSIONE
1:0



MANUALE

I dati del presente documento possono essere modificati senza preavviso e non impegnano la Zanussi Elettronica. È vietato copiare il BASIC Creativision su cassetta, nastro, disco, ROM o su qualunque altro mezzo per qualsiasi scopo senza autorizzazione scritta della Zanussi Elettronica.

Prima edizione 1982

INDICE

PREFAZIONE

Questo manuale ha lo scopo di servire al principante come introduzione alla programmazione in BASIC nel Sistema Computer CreatiVision. Partendo dal livello più elementare, esso conduce il lettore agli aspetti fondamentali del BASIC e alle procedure per creare programmi sul Sistema Computer CreatiVision. Esso comprende tutti i concetti e le istruzioni occorrenti per la programmazione essenziale in BASIC e non richiede una precedente conoscenza del BASIC o della programmazione in generale.

Il lettore dovrebbe provare le nuove istruzioni e i concetti di programmazione man mano che vengono presentati, digitandoli sulla tastiera ed eseguendo i programmi di esempio. Si consiglia di fare prove sui programmi esempio introducendovi modifiche, immaginando gli effetti delle modifiche e successivamente confermando o correggendo le proprie conoscenze in base agli effetti che di fatto si osservano. Quanto prima possibile si dovrebbero cominciare a scrivere programmi per risolvere problemi semplificati nel campo particolare d'interesse del lettore. Soltanto in questo modo i concetti e le possibilità qui presentati possono divenire conoscenza utile.

Nulla di ciò che si può fare sulla tastiera può recare danno a un Sistema Computer CreatiVision. Nel testo sono introdotti avvertimenti quando vengono presentate istruzioni che potrebbero distruggere archivi di dati. Pertanto il lettore deve sentirsi libero di fare esperimenti con il Sistema Computer CreatiVision a tutti gli stadi di apprendimento.

In linea di massima, il lettore dovrebbe seguire l'ordine di presentazione del testo; vi sono, nondimeno, altre alternative.
Il capitolo 12, che tratta l'immagazzinamento su nastro, può esser letto in qualunque momento, allorché il lettore desideri conservare un programma sulla cassetta di nastro. Questo manuale, pur essendo scritto in maniera particolare per chi desidera imparare a programmare in BASIC sul Sistema Computer CreatiVision, può servire anche come introduzione generale alla programmazione in BASIC su qualsiasi sistema. Il lettore deve tuttavia ricordare che il linguaggio BASIC presenta molte forme. Ci sono leggere differenze tra una realizzazione di BASIC e un'altra.

Infine, ci auguriamo che questo manuale vi possa aiutare a imparare le nozioni fondamentali di BASIC e i concetti di base nella programmazione dei computer.

DIVERTITEVI COL VOSTRO SISTEMA COMPUTER CREATIVISION!

CAPITOLO 1

INTRODUZIONE
– Che cos'è un computer

Che cos'è un computer?

I computers sono apparecchi che eseguono varie operazioni in base a istruzioni assegnate dalle persone che li usano.
Un sistema computer comprende, in generale, le seguenti unità:

1) Unità di elaborazione centrale (CPU) – che esegue le operazioni specificate nell'istruzione, ad esempio operazioni aritmetiche e logiche; può essere considerata il cervello del sistema computer.
ii) Unità di memoria – che immagazzina le istruzioni e le informazioni generate dal computer o assegnate dall'utente. Questa unità é residente all'interno del computer, è la CPU ottiene direttamente da essa l'informazione.
iii) Unità di immagazzinamento di massa – che immagazzina istruzioni e informazioni generate dal computer o date dall'utente. Questa unità risiede all'esterno del computer. Per esempio, l'unità di memorizzazione su nastro e l'unità floppy disk (a dischi flessibili) sono unità di immagazzinamento di memoria. Le informazioni depositate in queste unità devono essere trasferite nell'unità di memoria prima che la CPU possa elaborarle.
iv) Dispositivo di input (ingresso) – che permette all'utente di introdurre istruzioni o informazioni al computer, p.es. la tastiera.
v) Dispositivo di output (uscita) – che riceve l'informazione o i risultati inviati dal computer, p.es. la stampante.

Gli apparecchi di input e output agiscono insieme come un canale di comunicazione nei due sensi tra l'utente e il computer.
Per quanto le dimensioni dei computers si differenzino l'una dall'altra, tutti i sistemi computers reali richiedono in linea di massima le unità sopra menzionate.

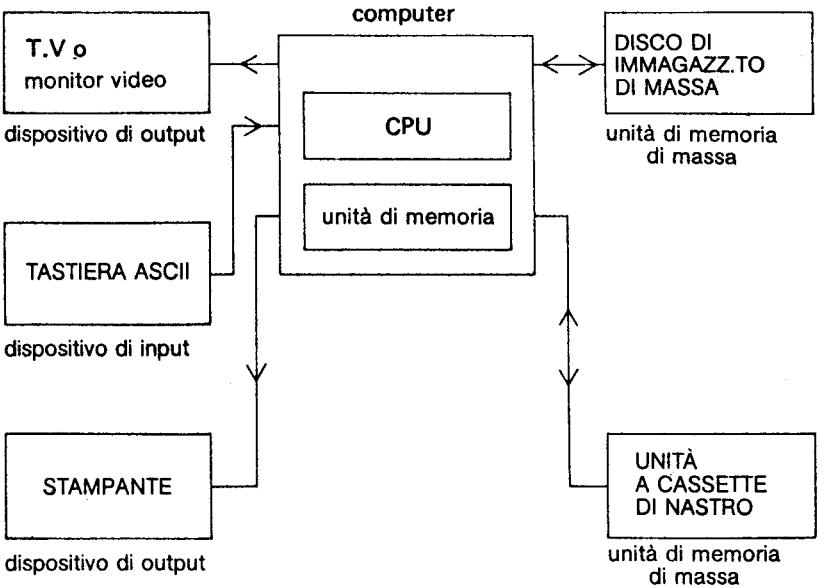


Fig. 1 Configurazione di un sistema computer in generale.

CAPITOLO 2

DEFINIZIONE
– Concetti di programmazione
– L'interprete BASIC

CONCETTI DI PROGRAMMAZIONE

Il processo di specificare un insieme di istruzioni per un computer si chiama programmazione, e l'insieme di istruzioni si chiama programma. La persona che prepara un programma si chiama programmatore.
Nel preparare un programma per un computer sono richiesti due passi. Anzitutto, il programmatore deve conoscere quali istruzioni indicare, e l'ordine con cui indicarle. Secondariamente, egli deve essere in grado di comunicare al computer le sue indicazioni. La comunicazione è realizzata per mezzo di un «linguaggio» di programmazione che il programmatore scrive e il computer «legge» per decidere che cosa fare.
Oggi sono utilizzati molti linguaggi di programmazione. Alcuni sono destinati ad applicazioni molto particolari e altri sono destinati a un uso più generale. Il BASIC è un linguaggio di quest'ultima categoria.

L'INTERPRETE BASIC

BASIC, acronimo di Beginner's All Purpose Symbolic Instruction Code (Codice di Istruzioni Simboliche per Applicazioni Universali per Principianti), possiede un semplice vocabolario inglese e poche regole grammaticali, e d'altro canto ricorda la normale notazione matematica. Sebbene il linguaggio sia fondamentalmente semplice, é tuttavia un linguaggio potente, che consente possibilità di calcolo aritmetico, di confronto logico, di utilizzo di variabili con indici, di funzioni trigonometriche, di liste, di variabili multiple e la manipolazione di stringhe di caratteri alfanumerici.
I programmi scritti di BASIC sono tradotti, da un programma di traduzione di linguaggio, in un linguaggio che il processore centrale è in grado di capire. Questo programma di traduzione del linguaggio è chiamato interprete BASIC e risiede nella cartuccia BASIC fornita.
Il prossimo capitolo vi indicherà come mettere in funzione il vostro Sistema Computer CreatiVision, e i seguenti capitoli illustreranno le istruzioni BASIC con esempi messi a disposizione per aiutarvi a comprenderle.

CAPITOLO 3

Preparazione del Sistema Computer CreatiVision

PREPARAZIONE DEL SISTEMA COMPUTER CREATIVISION

Per mettere in funzione il Sistema Computer CreatiVision, occorrono queste parti.

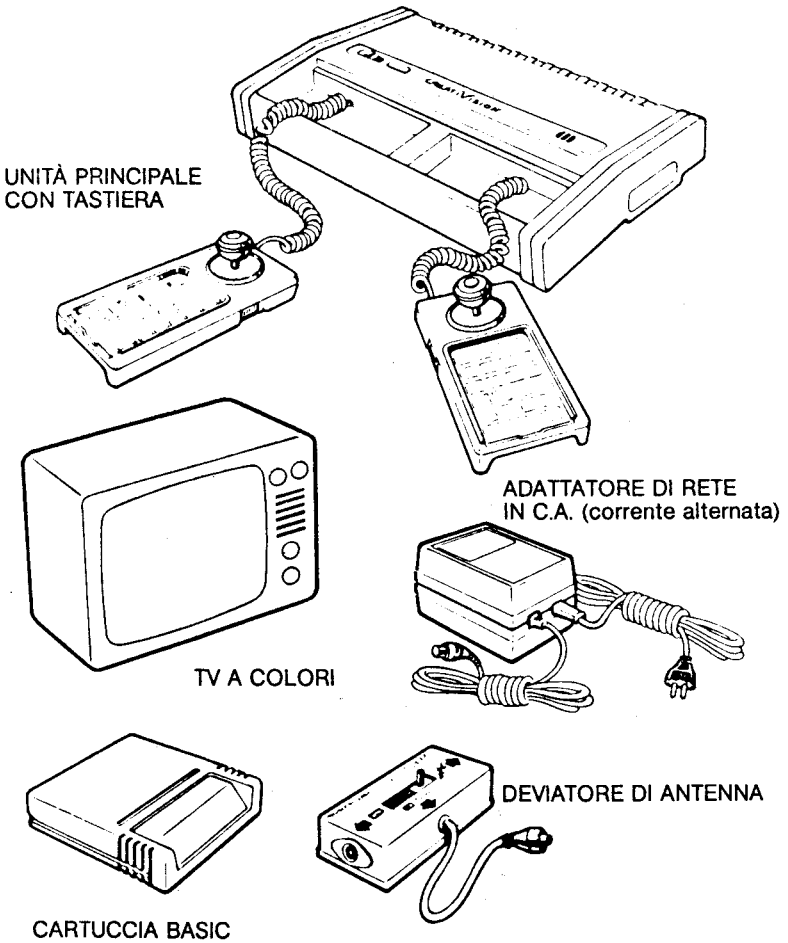


Fig. 3.1

DEVIATORE D'ANTENNA

Il deviatore d'antenna vi consente di avere un mezzo adatto per usare il vostro televisore sia per i normali programmi TV sia per i giochi.

- Staccare il cavo coassiale d'antenna dal televisore e collegarlo al deviatore.
- Collegare il cavo coassiale del deviatore alla presa d'antenna del televisore.
- Collegare il cavo coassiale dall'Unità Principale al deviatore.

Una volta realizzata l'installazione, potete azionare la leva per scegliere i programmi TV o i giochi.

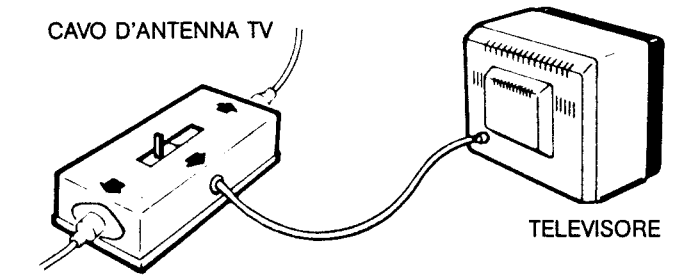


Fig. 3.2

OPERAZIONE PER METTERE IN FUNZIONE IL VOSTRO SISTEMA COMPUTER

1. Accertarsi che l'interruttore di alimentazione dell'Unità Principale sia in posizione off.
2. Collegare la spina di alimentazione dell'adattatore in c.a. alla presa d'alimentazione dell'Unità Principale.
3. Collegare la spina da parete dell'adattatore c.a. a una normale presa c.a. da muro.

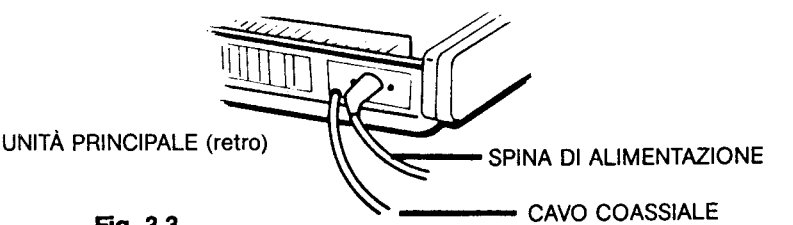


Fig. 3.3

4. Spostare la levetta del deviatore in posizione Game (gioco).
5. Inserire la cartuccia dell'interprete BASIC nella scanalatura dell'Unità Principale.

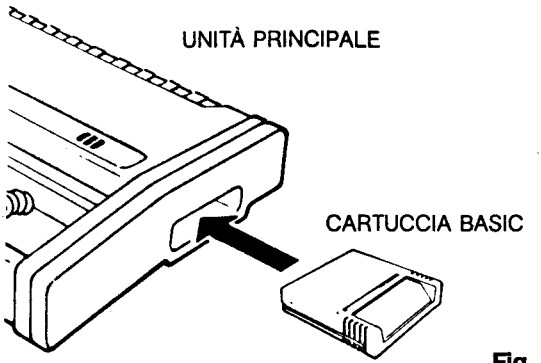


Fig. 3.4

6. Accendere il televisore e selezionare sul televisore il pulsante del programma prescelto. Il programma prescelto per il sistema è di solito quello di riserva che non si adopera per i normali programmi TV.
7. Spostare l'interruttore di alimentazione dell'Unità Principale in posizione ON. La prima volta che si effettua il collegamento, sintonizzate il canale per ricevere la visualizzazione BASIC CreatiVision.
   Nota: Se il vostro televisore ha un pulsante AFC (Sintonia Fine Automatica) accertarsi che questo interruttore sia in posizione off (staccato) quando si regola la sintonia.

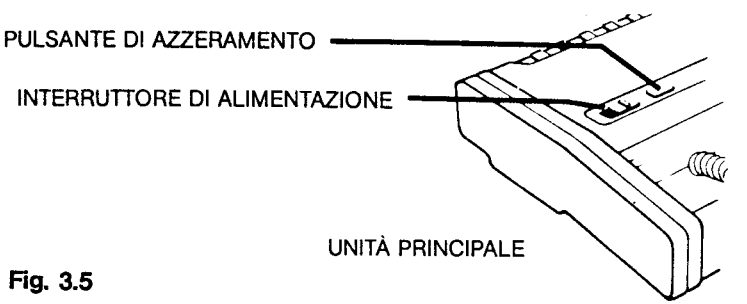


Fig. 3.5

8. Ora il computer è stato messo in funzione ed é pronto per la scrittura delle istruzioni.

AVVERTENZE

1. Tenere lontano dai liquidi la tastiera dell'Unità Centrale e la cartuccia BASIC.
2. Evitare di esporre al calore eccessivo la cartuccia BASIC, l'unità principale o la tastiera. Si raccomanda di mantenerle in condizioni di buona ventilazione.
3. Staccare l'alimentazione quando non si usa l'apparecchio.
4. Non far cadere l'Unità Principale, la tastiera o la cartuccia BASIC. Maneggiarle con cura.
5. Inserire la cartuccia BASIC nella scanalatura lentamente accertandosi che l'interruttore sia spento quando si inserisce o si estrae la cartuccia BASIC dall'Unità Principale.
6. Non mettere le dita nell'estremità aperta della cartuccia. L'elettricità statica delle dita può eventualmente danneggiare i sensibili componenti elettronici della cartuccia.
7. Togliere la cartuccia BASIC dall'Unità Principale quando non la si usa.

RI-COMMUTAZIONE DEL TELEVISORE ALL'USO NORMALE

1. Staccare l'interruttore di rete.
2. Spostare la levetta sul deviatore d'antenna alla posizione TV antenna.
3. Il televisore ora é pronto per l'uso normale.

RIASSUNTO DELLE FASI DI MESSA IN FUNZIONE

1. Inserire correttamente la cartuccia BASIC entro l'Unità Principale.
2. Allacciare correttamente alla presa di parete e, dall'altra parte, all'Unità principale l'adattatore c.a.
3. Il deviatore d'antenna è posizionato su Game. (gioco).
4. Collegare correttamente tutti i cavi coassiali.
5. Accendere gli interruttori di alimentazione dell'unità principale e del televisore.
6. Sintonizzarsi sul canale corretto.

CAPITOLO 4

COMINCIARE A PROGRAMMARE

- Modo di esecuzione immediata
- Modo di programmazione
- Formati di programma BASIC
- Composizione di una riga
- LIST
- Cancellazione di una riga dal programma
- LLIST
- NEW

COMINCIARE A PROGRAMMARE

Il computer esegue le istruzioni (assegnate dall'utente) in due modi, e precisamente il modo di esecuzione immediata e il modo di esecuzione posticipata (a programma).

MODO DI ESECUZIONE IMMEDIATA

Qualsiasi istruzione che non abbia un «numero di riga» viene eseguita immediatamente dopo che si é premuto il tasto RET'N (per RET'N si intende RETURN).

Esempi: *PRINT 2+4* RET'N
6
PRINT 2+4, 3/6 RET'N
6 Ø.5
PRINT «RESULT», (2+4)/(3/6) RET'N
RESULT 12

Il modo di esecuzione immediata fa lavorare il computer come una calcolatrice, ma nondimeno in questo tipo di funzionamento il computer é più potente di una calcolatrice. Esso é in grado di calcolare più di una espressione alla volta e può gestire anche stringhe (successioni ordinate) di caratteri.
L'istruzione PRINT è necessaria quando si usa il modo di esecuzione immediata, perché, altrimenti, il risultato non verrebbe visualizzato sullo schermo. Ora provate l'esempio qui sopra e osservate il risultato.

MODO DI ESECUZIONE POSTICIPATA (A PROGRAMMA).

Ogni istruzione provvista di un «numero di riga» all'inizio dell'istruzione non viene eseguita dopo che l'istruzione è stata terminata dal tasto RET'N, ma solo dopo aver digitato RUN e RET'N.

Esempi: *1Ø PRINT 2+4* RET'N
RUN RET'N
6
1Ø PRINT 2+4, 3/6 RET'N
RUN RET'N
6 Ø.5

1Ø PRINT 7+9 RET'N
2Ø PRINT 2+4, 3/6 RET'N
*30 PRINT «RESULT», (2*6)* RET'N
RUN RET'N
16 Ø,5
6
RESULT 12

Nell'ultimo esempio, si può osservare che il numero di riga «1Ø» è eseguito per primo, successivamente é eseguito il numero di riga «2Ø» e per ultimo, é eseguito il numero di riga «3Ø». Questo é l'ordine con cui il computer esegue le istruzioni in un programma. Le istruzioni (righe) con numero di riga più basso vengono eseguite prima delle istruzioni (righe) con numero di riga più alto.
Si può tuttavia indicare al computer quale istruzione si deve eseguire dopo una data istruzione utilizzando istruzioni come GOTO ecc. che saranno discusse al capitolo 9.
Si noti che utilizzando il comando RUN, il computer inizia a eseguire il programma dall'inizio. Se si vuole iniziare il programma da una riga diversa dalla prima, si utilizza: RUN seguito dal numero di riga.

Esempio: *1Ø PRINT 7+9* RET'N
2Ø PRINT 2+4, 3/6 RET'N
*3Ø PRINT «RESULT», (4/2*6)* RET'N
RUN 2Ø RET'N
6 Ø,5
RESULT 12

FORMATI DI PROGRAMMA BASIC

1. Ogni istruzione di programma inizia con un «numero di riga», p. es. 1Ø, 2Ø, 3Ø ecc.
2. Un numero di riga é un numero intero compreo tra Ø e 9999.
3. Dopo il numero di riga si trova l'istruzione BASIC corretta, altrimenti il computer risponde con un messaggio d'errore «SYNTAX ERROR» (errore di sintassi).

Esempio: *1Ø PRIMT 2+3* RET'N
RUN (RET'N)
SYNTAX ERROR

4. Ogni riga di programma viene terminata da un RET'N.

composizione di una riga

Se introducendo un'istruzione di programma si commette un errore, c'è il tasto che aiuta a tornare indietro al punto errato e a fare la correzione.

Esempio: *PRIMT*
si preme ora ← e lo schermo mostra
PRIM
si preme ora ← e lo schermo mostra
PRI
ora si può introdurre la lettera corretta.

LIST

Il comando LIST si adopera per visualizzare tutto il programma depositato in memoria appena scritto; le istruzioni sono listate in base al «numero di riga» in ordine crescente.

Esempio: *47 PRINT «GOOD BYE»* RET'N
25 PRINT 2+3 RET'N
33 PRINT 4+3 RET'N
12 PRINT «HELLO» RET'N
LIST RET'N
12 PRINT «HELLO»
25 PRINT 2+3
33 PRINT 4+3
47 PRINT «GOOD BYE»

Se dopo il comando LIST si introduce un numero di riga, viene listata solo la riga specificata.

Esempio: *LIST 47* RET'N
viene visualizzato
47 PRINT «GOOD BYE»

Se vengono introdotti, dopo il comando LIST, due numeri di riga separati da un virgola, vengono listate le righe di istruzione comprese tra i due numeri.

Esempio: *LIST 25, 33* RET'N
25 PRINT 2+3
33 PRINT 4+3

cancellare una riga di programma

Per cancellare una riga dal programma, basta digitare il «numero di riga» di programma che si vuole cancellare (eliminare) e digitare RET'N. Quella riga viene allora cancellata dal programma (usando il programma del precedente esempio):

Esempio: *12* RET'N
LIST RET'N
25 PRINT 2+3
33 PRINT 4+3
47 PRINT «GOOD BYE»

La riga di programma numero 12 é stata cancellata. Ora provate voi a cancellare la riga numero 33.

LLIST

LLIST è simile a LIST, tranne che in questo modo si lista tutto il programma sulla stampante anziché sullo schermo TV.

Sintassi: LLIST.

Se si adopera LLIST, devono essere preparate la stampante e la interfaccia I/O. Per maggiori particolari vedere il manuale d'uso dell'interfaccia I/O.

NEW

Il comando NEW si utilizza per cancellare (eliminare) tutto il programma attualmente depositato nella memoria.

Ora, digitare:
NEW RET'N
L'esempio soprastante viene cancellato.

Introdurre il comando LIST. Non viene listato niente.

NOTA

D'ORA IN POI SI CONSIDERA CHE IL LETTORE SI SIA IMPRATICHITO NELL'USO DEL TASTO RET'N. A PARTIRE DAL PROSSIMO CAPITOLO, NON SI STAMPERÀ NEGLI ESEMPI IL COMANDO RET'N.

CAPITOLO 5

NUMERI E VARIABILI

- Costanti numeriche
- Variabili numeriche
- Istruzione LET
- Istruzione REM
- Funzione ABS
- Funzione SGN
- Funzione RND

costanti numeriche

Costanti numeriche

Le costanti numeriche contengono un valore costante per tutto un programma, e possono essere positive o negative. Le costanti numeriche possono essere scritte utilizzando la notazione decimale come segue:

+2, 34, Ø.432, 3E18, 4E(−35) ecc.

La gamma dei numeri utilizzabili è compresa tra
$10^{-38} \leqq n \leqq 10^{38}$

variabili numeriche

Variabili numeriche

Una variabile é un dato il cui valore può essere modificato durante l'esecuzione di un programma. Una variabile é indicata da un nome della variabile fisso. Ognuna delle 26 lettere dell'alfabeto é un nome valido per una variabile. I valori delle variabili vengono assegnati con le istruzioni LET, INPUT. Il valore assegnato a una variabile resta immutato finché non si incontra un'altra volta un'istruzione LET o INPUT che assegna a quella variabile un nuovo valore o finché la variabile non viene incrementata da un'istruzione FOR. Tutte le variabili vengono fissate uguali a Ø prima dell'esecuzione del programma. Occorre assegnare un valore a una variabile solo quando si desidera un valore diverso da Ø. Tuttavia, é una buona regola di programmazione fissare uguali a Ø le variabili ogni qualvolta il programma lo richiede.
In tal modo si realizzerebbe una registrazione dei valori assegnati alle variabili.

Istruzione LET

L'istruzione LET assegna alla variabile alla sinistra del «segno di uguale» il valore dell'espressione alla destra.

Ogni istruzione LET ha la forma:
LET variabile = espressione.

L'istruzione non indica uguaglianza algebrica, ma effettua il calcolo dell'espressione (se c'é un'espressione) e assegna il valore risultante a una variabile indicata.

Esempio:
```
1Ø LET A = 5
2Ø LET B = 2Ø
3Ø LET C = 6Ø
4Ø LET A = A+5
5Ø PRINT A+B+C
6Ø END
RUN
9Ø
```

Alla riga 4Ø, il precedente valore di A é aumentato di 5 e diventa il nuovo valore di A.
L'interprete BASIC permette all'utente di tralasciare la parola LET dall'istruzione LET. All'utente può risultare più semplice scrivere: 1ØA = 5 invece che 1ØLET A = 5.

REMARK

È spesso desiderabile inserire note e messaggi entro un programma utente. Sono utili nel programma, per un rapido riferimento al lettore, informazioni quali il nome e lo scopo del programma, come usarlo, come funzionano certe parti del programma e i risultati attesi in vari punti. Viene ignorato dall'interprete BASIC e non viene eseguito.

Sintassi: REM (qualsiasi cosa si vuole digitare)

Esempio:
```
10 REM ISTRUZIONE REMARK
20 REM PROGRAMMA BASIC
30 A = 3
40 PRINT A
50 END
RUN
3
```

Funzione valore assoluto, ABS (X)

La funzione ABS calcola il valore assoluto dell'espressione.
Per esempio: ABS (−34.67) = 34.67.

Sintassi: ABS (espressione)

Esempio: *1Ø PRINT ABS (3+4−6*5)*
RUN
23

Funzione segno, SGN (X)

La funzione segno produce il valore + 1 se X é di valore positivo, Ø se X é uguale a Ø, e −1 se X é negativo. Per esempio:
SGN (3,42) = 1; SGN (−42) = −1 e SGN (23−23) = Ø

Sintassi: SGN (espressione)

Esempio: *1Ø A = −12*
3Ø PRINT SGN (A); SGN (A−A)
RUN
−1 Ø

RND (Ø)

La funzione Numero Casuale, RND (Ø), produce un numero casuale tra Ø e 1.

Sintassi: RND (Ø)

Esempio:
```
1Ø FOR I = 1 TO 5
2Ø PRINT RND (Ø)
3Ø NEXT I
4Ø END
RUN
Ø.53675
Ø.1463
Ø.8Ø221
Ø.34245
Ø.36985
```

RND (N)

La funzione Numero Casuale, RND (N), in cui N è un numero positivo, produce un numero casuale intero tra Ø e N.

Sintassi: RND (N)

Esempio:
```
1Ø FOR I = 1 TO 5
2Ø PRINT RND (1Ø)
3Ø NEXT
4Ø END
RUN
6
4
7
2
8
```

CAPITOLO 6

MANIPOLAZIONE DEI DATI
- aritmetici
- relazionali
- logici
- funzionali

Operatori

Vi sono quattro tipi di operatori e precisamente aritmetici, relazionali, logici e funzionali.

Operatori matematici

L'interprete BASIC esegue automaticamente le operazioni matematiche di addizione, sottrazione, moltiplicazione, divisione ed elevamento a potenza. Le formule da calcolare sono rappresentate in formato simile alla comune notazione matematica. Vi sono cinque operatori aritmetici, adoperati per scrivere tali formule nella maniera seguente:

Operatore	Esempio	Significato
+	A + B	aggiunge B ad A
−	A − B	sottrae B da A
*	A * B	moltiplica A volte B
/	A / B	divide A per B
**	A ** B	calcola A alla B-esima potenza, ossia A^B

Usando le parentesi nelle espressioni, si può cambiare l'ordine degli operatori. Le operazioni dentro le parentesi vengono calcolate per prime. Entro le parentesi viene comunque seguito il normale ordine di precedenza degli operatori. Le operazioni aritmetiche sono eseguite nell'ordine seguente, assegnando la precedenza all'operazione descritta nel paragrafo 1 qui sotto.

1. Si calcolano tutte le formule racchiuse tra parentesi prima di adoperare il risultato per i calcoli successivo. Se vi sono parentesi «nidificate», vale a dire una coppia di parentesi racchiusa entro un'altra coppia di parentesi, ad es. così: A+(B*(D**2)), si calcola per primo il valore della parentesi più interna.
2. In assenza di parentesi in una formula, l'interprete BASIC esegue le operazioni nel seguente ordine:
   1° Elemento a potenza
   2° Calcolo dei segni meno (—)
   3° Moltiplicazione e divisione
   4° Addizione e sottrazione.

Quindi, per es., −A**B con segno meno é un'espressione corretta ed é identica a −(A**B). Ciò implica che −2**2 viene calcolato pari a −4. L'unica eccezione a questa regola é che il termine A**−B è permesso e viene calcolato come A**(−B).

3. In assenza di parentesi in un'espressione che comprende più di un'operazione dello stesso livello (vedi paragrafo 2 qui sopra), le operazioni vengono eseguite da sinistra a destra, nell'ordine in cui é scritta la formula. Per esempio:
   A/B/C é calcolato come (A/B)/C
   AxB/C é calcolato come (A*B)/C
   L'espressione A+B*C**D é calcolata nell'ordine seguente:
   1. C è elemento alla D-esima potenza
   2. Il risultato dell'operazione é moltiplicato per B
   3. Il risultato della precedente operazione é sommato ad A

Si consiglia l'utente di adoperare le parentesi anche quando non sono strettamente necessarie allo scopo di rendere le espressioni più facili da leggere e per ridurre le possibilità di scrivere espressioni diverse da quelle desiderate.

Esempi:

Espressione algebrica	Espressione BASIC
3X−3Y	3*X−2*Y
$(X^2)Y$	X**2*Y
B^2-4AC	B**2−4*A*C

Operatori relazionali

Gli operatori relazionali confrontano due valori e prendono decisioni riguardo all'andamento di un programma BASIC. Il risultato del confronto può essere «1» oppure «Ø».«1» significa vero, mentre «ø» significa falso.

Operatore	Relazione esaminata	Espressione
=	Uguaglianza	X = Y
<>	Disuguaglianza	X <> Y
<	Minore di	X < Y
>	Maggiore di	X > Y
<= o =<	Minore o uguale a	X <= Y, X =< Y
>= o =>	Maggiore o uguale a	X >= Y, X => Y

Esempio:
```
1Ø INPUT A, B
2Ø IF A<B THEN PRINT «A<B»
3Ø IF A=B THEN PRINT «A=B»
4Ø IF A>B THEN PRINT «A>B»
5Ø END
RUN
?1Ø
?5
A>B
```

Operatori logici

Gli operatori logici sono usati in istruzioni tipo IF-THEN, nelle quali si assegna una condizione per determinare le operazioni successive all'interno del programma utente. Siano, a questo proposito, A e B due espressioni relazionali aventi unicamente i valori VERO (1) e FALSO (Ø). Gli operatori logici sono calcolati dopo gli operatori aritmetici e relazionali. Gli operatori logici sono i seguenti:

Operatore	Esempio	Significato
NOT (non)	NOT A	Il negativo logico di A. Se A é vero, NOT A é falso.
AND (e)	A AND B	Il prodotto logico di A e B. A AND B ha valore vero solo se A e B sono tutti e due veri e ha valore falso se A o B o entrambi sono falsi.
OR (o)	A OR B	La somma logica di A e B. A OR B ha valore vero se A oppure B oppure entrambi sono veri e ha valore falso solo se A e B sono entrambi falsi.

Le tabelle seguenti si chiamano tabelle di verità. Esse illustrano il risultato delle operazioni logiche sopra riportare con tutte le possibili combinazioni dei valori di A e B.
T (true) = vero
F (false) = falso

TABELLA DI VERITÀ PER FUNZIONE «NOT».

A	NOT A
T	F
F	T

TABELLA DI VERITÀ PER FUNZIONE «AND».

A	B	A AND B
T	T	T
T	F	F
F	T	F
F	F	F

TABELLA DI VERITÀ PER FUNZIONE «OR».

A	B	A OR B	
T	T	T	T
T	F	T	T
F	T	T	
F	F	F	

Esempio:
```
1Ø INPUT A, B, C
2Ø IF A > B AND B >C THEN PRINT «A>B>C»
3Ø IF NOT (A > B) OR NOT (B > C)
   THEN PRINT «A>B>C IS FALSE»
4Ø END
RUN
?1Ø
?5
?7
A > B > C IS FALSE
```

Nella pratica della programmazione, un utente incontra molti casi in cui vengono eseguite operazioni matematiche relativamente comuni. Il risultato di queste operazioni si può spesso trovare in volumi di tabelle matematiche, per esempio seni, coseni, radici quadrate, logaritmi ecc. Dato che queste sono operazioni che il computer esegue con rapidità e precisione, esse sono comprese nell'interprete BASIC. All'utente non occorre consultare tabelle per ricavare il valore del seno di 25 gradi o il logaritmo naturale di 150. Se in un'espressione si devono usare questi valori, si adoperano le funzioni intrinseche, come:

```
SIN (25*3,14/18Ø)
LOG (15Ø)
```

Le diverse funzioni matematiche disponibili nel nostro interprete BASIC sono descritte nella tabella seguente.

Codice della funzione	Significato
ABS (X)	Fornisce il valore assoluto di X.
SGN (X)	Fornisce la funzione segno di X, un valore di 1 preceduto dal segno di X, SGN Ø = Ø.
INT (X)	Fornisce il più grande numero intero minore o uguale a X, (INT (−Ø.5)=−1).
COS (X)	Fornisce il coseno di X (in radianti).
SIN (X)	Fornisce il seno di X (in radianti).
TAN (X)	Fornisce la tangente di X (in radianti).
SQR (X)	Fornisce la radice quadrata di X.
EXP (X)	Fornisce il valore e^x, in cui e = 2.71828.
LOG (X)	Fornisce il logaritmo naturale di X, $\log_e X$.
RND (Ø)	Fornisce un numero casuale tra Ø e 1.
RND (N)	Fornisce un numero casuale intero da Ø a N. −1

In sin (X), cos (X) e tan (X), il valore di X deve essere compreso tra i valori: $-1000 < X < 1000$.

Esempio:
```
1Ø REM TO PRINT A SINE AND COSINE TABLE
2Ø PRINT «SIN (X)», «COS (X)»
3Ø FOR X = Ø TO 2 STEP Ø.5
4Ø PRINT SIN (X), COS (X)
5Ø NEXT
RUN
SIN (X)          COS (X)
Ø                1
Ø.47942          Ø.87758
Ø.84147          Ø.54Ø3
Ø.9975           Ø.Ø7Ø73
Ø.9Ø93           -Ø.41614
```

CAPITOLO 7

FUNZIONI DI STRINGA

- Stringhe di caratteri
  - Costanti stringa
  - Variabili stringa
- Manipolazione delle stringhe
- Funzioni di stringa:
  - LEFT$
  - RIGHT$
  - MID$
  - CHR$
  - STR$
  - LEN
  - VAL
  - ASC
- Confronto di stringhe

STRINGHE DI CARATTERI

I capitoli precedenti descrivono la manipolazione dell'informazione numerica. L'interprete BASIC tuttavia elabora anche informazione sotto forma di stringhe di caratteri. In questo senso una stringa é definita come successione ordinata di caratteri considerata un tutt'uno. Una stringa può essere composta da qualsiasi combinazione di caratteri ASCII.

Esempio: *«ABC»*
«BASIC»

costanti stringa

Costanti stringa

Analogamente ai numeri, che possono essere usati come costanti o indicati con nomi di variabili, l'interprete BASIC elabora anche stringhe di caratteri costanti. Le stringhe di caratteri costanti sono delimitate dalle virgolette, per esempio:

1Ø LET A$ = «XYZ 123»

Ciò significa che il valore di A$ é la stringa di caratteri XYZ123.

variabili stringa

Variabili stringa

Alle stringhe si possono assegnare nomi di variabili. Ogni lettera dell'alfabeto seguita da un carattere dollaro ($) é un nome valido per una variabile stringa, come ad es. A$. La massima lunghezza di una variabile stringa é 32 caratteri. Durante l'elaborazione, nessuna espressione stringa può superare la lunghezza di 32 caratteri.

Manipolazione di stringhe

Il BASIC CreatiVision permette ai programmatori di elaborare le stringhe. Il solo operatore per le espressioni e le variabili stringa é «+».«+» significa concatenazione (collegamento).

Esempio 1:
```
10 LET A$ = «XYZ123»
20 P$ = A$ + «PQR»
30 PRINT P$
40 END
RUN
XYZ123 PQR
```

Esempio 2:
```
10 A$ = «  »
20 FOR I = 1 TO 5
30 A$ = A$ + «  »
40 PRINT A$
50 NEXT I
RUN
#
# #
# # #
# # # #
# # # # #
```

FUNZIONI DI STRINGA

Oltre alle funzioni matematiche intrinseche (p. es. SIN, LOG), il nostro interprete BASIC consente varie funzioni per il trattamento delle stringhe di caratteri. Queste funzioni permettono al programma di eseguire operazioni aritmetiche con stringhe numeriche, concatenare due stringhe, accedere a parte di una stringa, determinare il numero di caratteri in una stringa, generare una stringa di caratteri corrispondente a un numero dato o viceversa, cercare una sotto-stringa all'interno di una stringa più grande ed eseguire altre operazioni utili.

LEFT$

Sintassi: LEFT$ (espressione stringa, espressione numerica) p. es. LEFT$ (A$, N). Produce una sotto-stringa dal primo carattere (i caratteri a sinistra della stringa A$) fino all'ennesimo carattere.

Esempio:
```
1Ø A$ = «ABC»
2Ø C$ = LEFT$ (A$+«XYZ», 3+1)
3Ø PRINT C$
RUN
ABCX
```

RIGHT$

Sintassi: RIGHT$ (espressione stringa, espressione numerica) p. es. RIGHT$ (A$, N)
Produce una sotto-stringa della stringa A$ composta degli ultimi N caratteri.

Esempio:
```
1Ø A$ = RIGHT$ («BASIC», 3)
2Ø PRINT A$
RUN
SIC
```

MID$

Sintassi: MID$ (espressione stringa, espressione numerica, espressione numerica) per es. MID$ (A$, M, N).
Produce una sottostringa della stringa A$ a partire dal carattere ennesimo e di lunghezza pari a N.

Esempio:
```
10 A$ = «ABCDEFGH»
20 C$ = MID$ (A$, 3, 4)
30 PRINT C$
RUN
CDEF
```

CHR$

Sintassi: CHR$ (espressione numerica)
per es. CHR$ (N).
Genera una stringa di un carattere avente valore ASCII pari a N (vedi appendice D). Per esempio CHR$ (65) é equivalente ad «A». Può essere generato un solo carattere.

Esempio:
```
10 FOR I = 65 TO 70
20 PRINT CHR$ (I)
30 NEXT I
RUN
A
B
C
D
E
F
```

STR$

Sintassi: STR$ (espressione numerica)
per es. STR$ produce un'espressione stringa di un argomento numerico.

Esempio 1:
```
10 A$ = STR$ (3 *2+1)
20 C$ = «00» +A$
30 PRINT A$, C$
RUN
 7 007
```

Esempio 2:
```
10 A$ = STR$ (0.125 + 0.5)
20 C$ = A$ + «K»
30 PRINT C$
RUN
0.625K
```

LEN

Sintassi:
LEN (espressione *stringa)*
per es. LEN (A$)
Produce il numero di caratteri, compresi gli spazi, della stringa A$.

Esempio 1:
```
10 A$ = «ABCDEF»
20 X = LEN (A$)
30 PRINT X
RUN
 6
```

Esempio 2:
```
10 A$ = «XY»
20 C$ = «123»
30 X = LEN (A$+C$)+6
40 PRINT X
RUN
 11
```

val

Sintassi: VAL (espressione stringa)
per es. VAL (A$)
Produce il valore numerico dell'espressione stringa.

Esempio:
```
10 A$ = «123 + 1»
20 X = VAL (A$+«–100»)
30 PRINT X
RUN
24
```

asc

Sintassi: ASC (espressione stringa)
per es. ASC (A$)
Produce il valore decimale ASCII del primo carattere di A$.
Per esempio, il valore decimale ASCII di «X» é 88.
Se B$ = «XAB», allora ASC(B$) = 88.

Esempio:
```
10 X = ASC («AXY»)
20 PRINT X
RUN
65
```

Confronto di stringhe

Gli operatori relazionali si possono adoperare anche nelle espressioni stringa, analogamente che nelle espressioni numeriche,
per es. A$ = «1234», X.
Il confronto é effettuato prendendo i valori numerici dei corrispondenti caratteri delle stringhe e poi confrontandoli. Dall'appendice D sappiamo che il valore di «A» é 65, di «B» é 66, di «C» é 67, di «D» é 68, ecc. Possiamo allora effettuare i seguenti esempi.

Esempio:
```
10 A$ = «AA»
20 B$ = «BA»
30 IF A$ < B$ THEN PRINT 20
RUN
20
```

* Dato che il valore numerico del carattere A é minore del valore numerico del carattere B (65 é minore di 66, pertanto A$ é minore di B$).

Esempio:
```
10 A$ = «ABC»
20 B$ = «ABD»
30 IF B$>A$ THEN PRINT 20
RUN
20
```

* I primi due caratteri di ciascuna stringa sono uguali. Il computer allora cerca il terzo carattere ed effettua il confronto, ossia confronta «D» e «C». Sappiamo che il valore numerico di «D» é 68 e di «C» é 67. Possiamo dunque concludere che B$ é maggiore di A$.

Esempi:
```
10 A$ = CHR$ (30 *2+6)
20 IF A$ > = «B» THEN PRINT 1
30 IF A$>= «9» THEN PRINT 2
RUN
1
2
```

Esempi:
```
10 W$ = «BASIC»
20 X$ = «BA»
30 IF LEFT$ (W$, 3) > X$ THEN 60
40 STOP
60 PRINT X$
RUN
BA
```

Esempi:
```
10 W$ = «APPLE»
20 X$ = «ORANGE»
30 IF W$ <> X$ THEN 60
40 STOP
60 PRINT X$
RUN
ORANGE
```

Esempi:
```
10 REM...PRINT ASCII CHARACTERS
20 FOR I = 1 TO 128
30 PRINT CHR$ (I)
40 NEXT I
RUN
.
.
.
.
.
A
B
C
.
.
.
.
.
```

* Vengono stampati 128 caratteri ASCII. Sullo schermo sono mostrati solo i caratteri stampabili.

CAPITOLO 8

CONTROLLO DEL SISTEMA
- STOP
- END
- CONT
- CNTL/C
- RESET

Allo scopo di agevolare il controllo dell'esecuzione del vostro programma e la correzione degli errori, sono realizzati numerosi comandi/istruzioni. Si tratta delle istruzioni STOP ed END e dei comandi CONT e CNTL/C.
Si può fermare un programma «in loop» (ossia entrato in un ciclo di istruzioni da cui non é più possibile uscire) premendo il pulsante «RESET».

STOP

L'istruzione STOP termina l'esecuzione del programma e restituisce il controllo al modo comandi. Può comparire più volte in uno stesso programma con salti condizionati per stabilire la fine effettiva del programma.

Sintassi: numero di riga STOP

Dopo l'esecuzione, produce la stampa di STOP AT numero di riga.
Un comando CONTINUE introdotto a questo punto fa riprendere l'esecuzione dall'istruzione che segue lo STOP (vedi paragrafo seguente). L'istruzione STOP può facilitarvi la correzione dei programmi, dato che é possibile aggiungere punti di interruzione in vari punti del vostro programma. É quindi possibile seguire l'esecuzione del programma ed esaminare i risultati intermedi delle variabili nel modo di esecuzione immediata.

Esempio:
```
1Ø PRINT «TEST STOP»
2Ø PRINT 123
3Ø STOP
4Ø END
RUN
TEST STOP
123
STOP AT 3Ø
```

END

L'istruzione END si usa per terminare l'esecuzione del programma. Deve essere l'ultima istruzione in un programma BASIC.

Sintassi: numero di riga END.

Il numero di riga di un'istruzione END deve essere di norma il numero di riga più grande del programma. Un programma può essere eseguito anche senza un'istruzione END. È, comunque, buona norma aggiungere un'istruzione END, giacché é di norma accettata da tutti gli interpreti BASIC.
A differenza del comando STOP, un programma che é terminato dopo un END non può essere continuato da un CONT (paragrafo seguente).

CONT

L'utente può collocare istruzioni STOP a piacere in tutto il programma. Ogni istruzione STOP fa arrestare il programma e fa stampare il numero di riga in cui si é varificato lo STOP. L'utente può quindi esaminare vari valori dei dati, modificarli nel modo di esecuzione immediata e dare il comando CONT per riprendere l'esecuzione del programma.

CNTL/C

È già noto che lo STOP può arrestare il programma in punti prestabiliti. L'utente puó d'altronde anche arrestare l'esecuzione del programma ad ogni istante, premendo insieme CNTL e C. Si può quindi utilizzare il modo immediato per esaminare e/o modificare i valori dei dati. Digitando il comando CONT si riprende l'esecuzione del programma. Quando si interrompe un programma digitando la combinazione CNTL/C, si provoca la stampa del seguente messaggio:

STOP AT
numero di riga, istruzione.

Combinazione CNTL/C:
Premere contemporaneamente i due tasti.

RESET

Sull'unità principale, c'é un pulsante RESET vicino all'interruttore di alimentazione ON-OFF. Questo pulsante si adopera per rimettere a zero il sistema computer. Quando si preme questo pulsante, il computer termina l'esecuzione del vostro programma. Tutti i caratteri sono riportati alla forma normale.
Così, se il vostro programma si trova in un ciclo da cui non esce (loop) a causa di qualche errore di programmazione e non si può uscire dal «loop» con CNTL/C, è possibile premere questo pulsante. Il programma BASIC che avete scritto non é modificato.

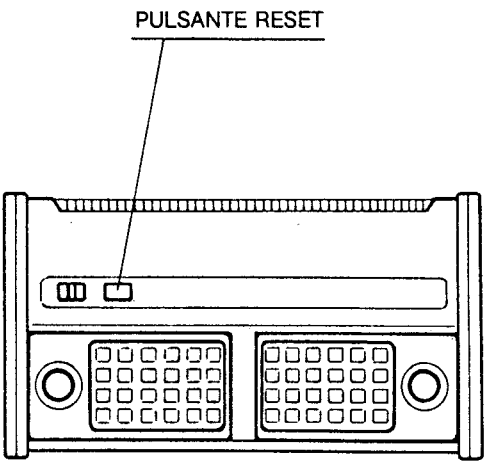


Fig. 8.1

CAPITOLO 9

ESECUZIONE CICLICA E CONDIZIONATA
- IF...THEN
- FOR...NEXT
- GOSUB/RETURN
- GO TO

In questo capitolo, esamineremo le istruzioni che consentono al programmatore di indirizzare il computer a eseguire un'istruzione di programma in base ai risultati o alle condizioni di operazioni aritmetiche e logiche; il programmatore può anche indirizzare il calcolatore a eseguire incondizionatamente una qualsiasi riga di programma.

IF THEN

Il corso del programma é regolato in base a condizioni indicate in un'espressione logica.
Scrivendo programmi, occorre un'istruzione che possa fornire un salto condizionato ad un'altra istruzione. L'istruzione IF-THEN fa proprio questo.

Sintassi: IF espressione logica THEN istruzione BASIC o numero di riga.

Esempio:
```
10 S = 0
20 N = 0
30 N = N + 1
40 S = S + N
50 IF N <> 10 THEN 30
60 PRINTS
70 END
RUN
55
```

FOR...NEXT

Un vantaggio dei computers é la loro capacitá di eseguire compiti ripetitivi. Il ciclo FOR...NEXT esegue una successione di istruzioni BASIC per un determinato numero di volte.
Supponiamo di volere una tabella di radici quadrate dei numeri interi da 1 a 5. Non é necessario scrivere la stessa espressione 5 volte per la stessa funzione. Usiamo invece, per eseguirla cinque volte, l'esecuzione ciclica.

Esempio:
```
10 FOR N = 1 TO 5
20 PRINT N, SQR (N)
30 NEXT N
RUN
1     1
2     1.41421
3     1.73205
4     2
5     2.23606
```

L'esempio soprastante ripete cinque volte la linea 20, incrementando ogni volta N di un'unità. È comunque possibile anche indicare l'entità dell'incremento con valori diversi da 1. Se vogliamo trovare solo le radici quadrate dei numeri dispari da 1 a 10, dobbiamo modificare l'esempi di sopra.

Esempio:
```
10 FOR N = 1 TO 10 STEP 2
20 PRINT «N=», SQR (N)
30 NEXT N
RUN
```

Sintassi: FOR VARIABILE = Espressione 1 TO Esp. 2 STEP Esp. 3.
STEP é il valore dell'incremento; se non viene indicato, il valore dell'incremento é 1. Esp. 1, Esp. 2., Esp. 3 sono semplici espressioni aritmetiche. Esp. 1 é il valore di partenza ed Esp. 2 il limite superiore.

Tutti i cicli FOR devono essere chiusi da un'istruzione NEXT.

Sintassi: NEXT (variabile)
Il nome della variabile dell'istruzione NEXT, deve essere lo stesso nome della variabile usata nell'istruzione FOR.
Si vedano gli esempi di sopra.

GOSUB/RETURN

Spesso operazioni identiche o quasi identiche devono essere eseguite ripetutamente in un programma. Non é conveniente ripetere continuamente lo stesso gruppo di istruzioni nel programma, giacché questo fa sprecare grandi quantità di memoria del computer e di tempo del programmatore. Se la stessa operazione deve essere eseguita in posizioni diverse in un programma, é buona cosa riportarle in un subroutine (sottoprogramma). Ogni volta che si desidera eseguire questa subroutine, si usa l'istruzione GOSUB. Tutte le subroutines devono essere terminate con un'istruzione RETURN che riporta il computer all'istruzione immediatamente successiva al GOSUB, allo scopo di riprendere l'esecuzione del resto del programma.

Sintassi: GOSUB (1° riga della subroutine)

```
:
:
(1ª riga della subroutine)
.
.                                  Subroutine
.
RETURN
```

GO TO

L'istruzione GO TO trasferisce l'esecuzione del programma immediatamente e senza condizioni a una riga di programma assegnata. Di solito la riga assegnata non è la riga immediatamente successiva del programma.

Sintassi: GO TO numero di riga.

Il numero di riga verso cui salta il programma può essere sia più grande sia più piccolo del numero di riga di partenza. È dunque possibile saltare in avanti o all'indietro entro un programma.

Esempio:
```
10 N = 1
20 S = 0
30 S = S+N
40 PRINT N, S
50 N = N + 1
60 GO TO 30
RUN
1     1
2     2
3     6
4     10
.     .
.     .
.     .
```

* N.B. Il programma continua l'esecuzione senza fine. Si adoperi CTL-C per farlo fermare.

Esempio:
```
10 FOR I = 1 TO 5
20 GOSUB 60
30 PRINT I, S
40 NEXT I
50 END
60 S = 0
70 FOR J = 1 TO I
80 S = S+J
90 NEXT J
100 RETURN
RUN
1     1
2     3
3     6
4    10
5    15
```

Esempio:
```
10 A = 30
15 PRINT A
20 GO TO A
25 PRINT A*A
30 END
RUN
30
```

Si può vedere che viene saltata la linea 25.

CAPITOLO 10

VARIABILI MULTIPLE – DIM

VARIABILI MULTIPLE

Una variabile multipla é un insieme di variabili o elementi, tutti con lo stesso nome, che si distinguono solo in base a un numero scritto tra parentesi dopo il nome (ossia l'indice della variabile). In questo modo si possono comporre entro il computer tabelle di variabili. A(I) è l'I-esimo elemento della variabile multipla A, in cui I è un numero intero.

Sintassi:

Esempio:
```
1Ø REM THE GAS BILLS OF HALF YEAR
2Ø DIM A (6)
3Ø REM ASK FOR 6 BILLS
4Ø FOR I = 1 TO 6
5Ø PRINT «THE BILL AMOUNT=»;
6Ø INPUT A (I)
7Ø S = S+A (I)
8Ø NEXT I
9Ø PRINT «THE TOTAL AMOUNT=»; S
```

L'istruzione DIM (dimensione) riserva caselle di memoria per le variabili multiple e le matrici. Le dimensioni possono essere 1 o 2.
A (6) é una variabile multipla a una dimensione con 6 elementi, mentre A (6, 6) é una variabile multipla a due dimensioni, o matrice, con 6x6 = 36 elementi.

La variabile multipla a due dimensioni richiede due indici per individuare ogni singolo elemento, come il numero di colonna e di riga in una comune matrice.
Ad esempio, per comporre una variabile multipla A con dimensioni 3 e 5, si usa un'istruzione DIM.

DIM A (3,5)

Ciò vi produce 3x5=15 variabili con indici

A(1, 1), A(1, 2)...A(1, 5)
A(2, 1), A(2, 2)...A(2, 5)
A(3, 1), A(3, 2)...A(3, 5)

Esempi:

```
1Ø DIM A (3, 3)
2Ø FOR I = 1 TO 3
3Ø FOR J = 1 TO 3
4Ø A (I, J) = I+J
5Ø PRINT A (I, J);
6Ø NEXT J
7Ø PRINT
8Ø NEXT I
RUN
2  3  4
3  4  5
4  5  6
```

CAPITOLO 11

COMANDI DI INPUT/OUTPUT
- INPUT
- PRINT
- TAB
- READ/DATA
- RESTORE

INPUT

L'istruzione INPUT permette di immettere dati dalla tastiera durante l'esecuzione di un programma. Come suggerimento viene visualizzato il «?».

Sintassi: INPUT variabile 1, variabile 2,...

Esempi:

```
1Ø INPUT A, B
2Ø PRINT A, B, A+B
3Ø END
RUN
? 3
? 4     qualsiasi numero venga digitato
3  4  7  (output del computer)
```

PRINT

PRINT visualizza sullo schermo il valore di una variabile o il valore di un'espressione.

Sintassi: PRINT elemento, elemento, elemento,...

Esempio:
```
1Ø PRINT «BASIC PROGRAM»
2Ø LET A = 3
3Ø PRINT A, A+A, A*A
4Ø END
RUN
BASIC PROGRAM (computer output)
3   6   9
```

Nell'istruzione PRINT, gli elementi possono essere separati da «;» anziché da «,». In questo caso non viene stampato spazio tra gli elementi.

Sintassi: PRINT elemento; elemento; elemento; elemento

Esempio:
```
1Ø PRINT 1; 2; 1+2
2Ø PRINT «A=»; 3
RUN
1  2  3  (output del computer)

RUN
1  2  3  (output del computer)
A = 3
```

L'istruzione PRINT con un punto e virgola alla fine non produce ritorno a capo.

Sintassi: PRINT elemento;

Esempio:
```
1Ø FOR A = 1 TO 1Ø
2Ø PRINT A;
3Ø NEXT
4Ø END
RUN
1 2 3 4 5 6 7 8 9 1Ø  (output del computer)
```

LPRINT

LPRINT è identico a PRINT con la differenza che la stampa viene prodotta sulla stampante anziché sullo schermo televisivo.

Sintassi: LPRINT elemento, elemento

Esempio:
```
1Ø LPRINT «THIS PROGRAM»
2Ø LPRINT «PRINT OUR THE CHARACTER SET»
3Ø FOR N = 32 TO 96
4Ø LPRINT CHR$ (N);
5Ø NEXT
RUN
```
(Viene stampato sulla stampante un elenco di caratteri).

* Nota: se si usa LPRINT, devono essere in funzione la stampante e l'interfaccia I/O. Per maggiori dettagli, fare riferimento al manuale di funzionamento dell'interfaccia I/O seriale e parallela.

TAB

TAB viene usato in un'istruzione PRINT per spostare il cursore a destra prima della stampa. Il numero di posizioni di cui ci si sposta è indicato fra parentesi ( ).

Sintassi: TAB (variabile)

TAB (n) sposta il cursore alla enn-esima colonna dello schermo. n può essere un numero intero qualsiasi tra Ø e 64.

Esempio: *PRINT TAB(5); «A»; TAB(1Ø); «A»*

```
    A         A
```

READ

Le istruzioni READ e DATA sono coordinate per fornire al programma utente una lista fissa di valori come dati. Un'istruzione READ inizializza le variabili prendendone i valori dalle istruzioni DATA.

Sintassi: READ (variabile 1, variabile 2...)

Esempio:

```
1Ø DATA, 1, 3, 5, 7
2Ø READ X, Y
3Ø READ A
4Ø READ B
5Ø PRINT X+Y, A+B
6Ø END
RUN
 4   12
```

DATA

L'istruzione DATA fornisce la tabella dei valori e dei dati a un'istruzione READ. L'istruzione READ legge elementi della tabella in base all'ordine del numero di riga dell'istruzione DATA. Tutte le istruzioni DATA devono trovarsi all'inizio di un programma BASIC o prima delle istruzioni READ.

Sintassi: DATA costante 1, costante 2, costante 3...

Esempio
```
10 DATA 8, 1, 6
20 DATA 3, 5, 7, 4, 9, 2
30 FOR I = 1 TO 3
40 READ A, B, C
50 PRINT A, B, C
60 NEXT I
70 END
RUN
8   1   6
3   5   7
4   9   2
```

Esempio:
```
10 REM FIND MAXIMUM
20 REM AND AVERAGE
30 DATA 0.125, 3, 0.6, 7
40 DATA 23, 9.3, 25.2, 8
50 M = -99
60 S = 0
70 FOR I = 1 TO 8
80 READ N
90 S = S+N
100 IF N>M THEN M=N
110 NEXT I
120 A = S/8
130 PRINT M, A
RUN 25.2 9.5281
```

RESTORE

Se dovesse essere necessario usare la stessa tabella di dati più di una volta in un programma, l'istruzione RESTORE permette di riportare l'indice dei valori DATA da leggere, al primo valore DATA.

Sintassi: RESTORE

Esempio:
```
1Ø DATA 1, 3, 5, 7, 9
2Ø READ A, B, C
3Ø RESTORE
4Ø READ X
5Ø PRINT A, C
6Ø PRINT X
7Ø END
RUN
1    5
1
```

CAPITOLO 12

IMMAGAZZINAMENTO SU NASTRO
– CSAVE
– CLOAD
– CRUN

Modulo di Immagazzinamento su Cassetta (registratore)

Potete ampliare ulteriormente il vostro Sistema Computer CreatiVision usando il Modulo di Immagazzinamento su Cassetta.
Il BASIC CreatiVision vi permette di caricare e conservare programmi da immettere nel computer. Registrando un programma su nastro, lo si può conservare in modo permanente. Successivamente é possibile caricare il programma dalla cassetta di nastro alla memoria del computer se si vuole riutilizzare il programma.
Il registratore é collegato al Sistema Computer con un cavo di interfaccia cassetta, come mostrato in figura 12. 1.

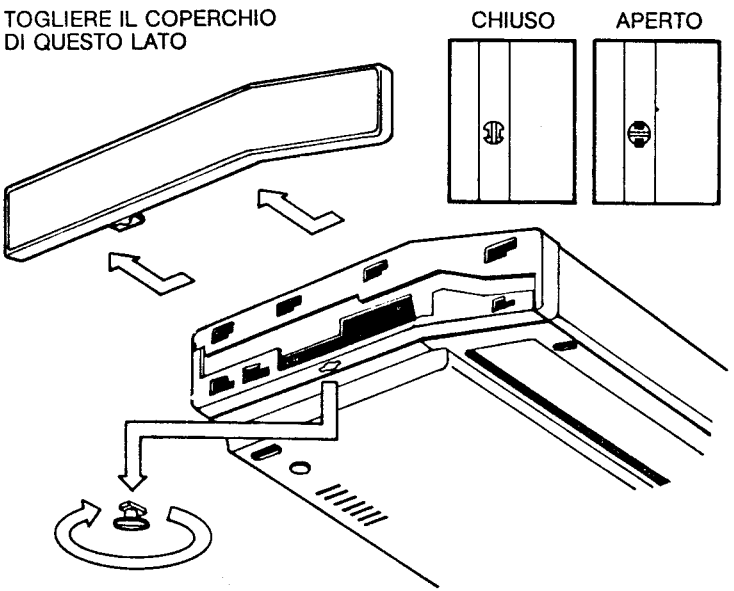


Fig. 12.1

CSAVE

Sintassi: CSAVE

Procedimento:

1. Collegare il registratore al sistema computer mediante il cavo.
2. Inserire una cassetta di nastro di buona qualità.
3. Premere i pulsanti «PLAY» e «RECORD» sul registratore per selezionare il modo «RECORD» (registrazione). Si noti che la cassetta non inizia a girare. Il computer regola l'alimentazione del motore del registratore. Il nastro quindi non si muove finché il computer non gli dice di farlo.
4. Prendere nota del numero del contatore di giri del registratore. Ciò serve per individuare facilmente la posizione corretta sul nastro quando volete caricare nuovamente il vostro programma nel computer.
5. Digitale il comando «CSAVE» (seguito da RET'N). In questo momento il programma comincia a essere registrato sul nastro, mentre lo schermo lista il programma.
6. Quando il programma sarà stato registrato, sullo schermo apparirà >□ Premere il pulsante «stop» sul registratore.

CLOAD

Sintassi: CLOAD

Procedimento:

1. Collegare il registratore tramite cavo al sistema computer.
2. Inserire la cassetta nell'apposito vano e riavvolgerla fino alla posizione corretta.
3. Premere solo il pulsante «play» per scegliere l'opzione «LOAD» del registratore.
4. Digitare il comando «CLOAD» (seguito da RET'N). In questo momento, il programma registrato inizia a caricarsi entro il sistema computer, e lo schermo ne produce la lista.
5. Dopo che il programma sarà stato caricato, sullo schermo apparirà >□ Premere il pulsante «stop» sul registratore.

N.B. Se il programma sul nastro proviene dalla Biblioteca su Nastro CreatiVision, vi saranno effetti di parlato e sonori provenienti dal televisore durante il caricamento del programma.

CRUN

Sintassi: CRUN

CRUN é equivalente a CLOAD + RUN. Il computer carica il programma e lo fa partire (RUN) automaticamente.
Il procedimento é analogo a CLOAD, eccetto il fatto di scrivere «CRUN» anziché «CLOAD».

CAPITOLO 13

FUNZIONI GRAFICHE E SONORE

- CLS
- COLOR
- CHAR
- PLOT
- SOUND
- JOY

Comandi grafici

I comandi grafici producono una visualizzazione sullo schermo di 32 colonne per 24 righe. Le 28 posizioni di stampa normalmente adoperate nel BASIC CreatiVision corrispondono alle colonne da 3 a 30. Dato che alcuni schermi possono non mostrare i due caratteri più a sinistra e i due più a destra, la grafica risulta più soddisfacente se adoperate le colonne da 3 a 30 ignorando le colonne 1 e 2 a sinistra e 31 e 32 a destra.

Vi sono quattro comandi grafici, essi sono:
1. CLS - cancella lo schermo.
2. COLOR - definisce un colore.
3. CHAR - definisce caratteri.
4. PLOT - posiziona e visualizza caratteri.

CLS

Il comando di cancellazione si adopera per cancellare l'intero schermo. Eseguendo il comando CLS, in tutte le posizioni dello schermo viene posizionato il carattere «spazio».

Esempio: *1Ø CLS*
2Ø PRINT «CLEAR SCREEN»

Eseguendo questo programma, lo schermo viene cancellato e quindi appare sullo schermo la scritta «CLEAR SCREEN» (cancella schermo).

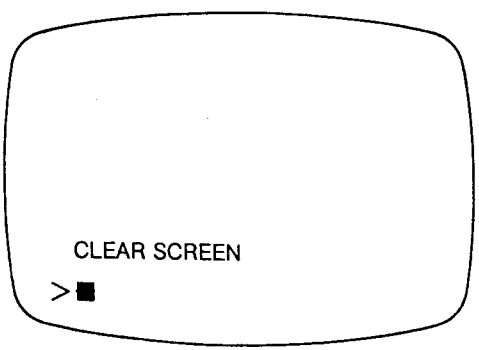


Fig. 13.1

COLOR

Il comando «COLOR» vi dà la possibilità di specificare il colore dei caratteri sullo schermo.

Sintassi: COLOR numero del gruppo di caratteri, codice del colore in primo piano, codice del colore di fondo.

Esempio: *COLOR 2, 1Ø, 14*

Ogni visualizzazione di carattere sullo schermo del vostro computer ha due colori. Il colore che compone il carattere in quanto tale si chiama colore del primo piano. Il colore che riempie il resto dello spazio in quel punto dello schermo si chiama colore dello sfondo. Col BASIC CreatiVision sono disponibili sedici colori. I codici di colore sono dati qui sotto.

Codice di colore	Colore
1	Trasparente
2	Nero
3	Verde medio
4	Verde chiaro
5	Azzurro scuro
6	Azzurro chiaro
7	Rosso scuro
8	Ciano
9	Rosso medio
10	Rosso chiaro
11	Giallo scuro
12	Giallo chiaro
13	Verde scuro
14	Magenta
15	Grigio
16	Bianco

Per usare i comandi COLOR dovete anche indicare a quale gruppo di caratteri appartiene quello da stampare.

Nell'appendice D é riportata la lista dei codici ASCII per i caratteri standard. ! numeri dei gruppi di carattere sono indicati qui sotto.

Numero del gruppo di caratteri	Codice ASCII	Numero del gruppo di caratteri	Codice ASCII
1	0-7	17	128-135
2	8-15	18	136-143
3	16-23	19	144-151
4	24-31	20	152-159
5	32-39	21	160-167
6	40-47	22	168-175
7	48-55	23	176-183
8	56-63	24	184-191
9	64-71	25	192-199
10	72-79	26	200-207
11	80-87	27	208-215
12	88-95	28	216-223
13	96-103	29	224-231
14	104-111	30	232-239
15	112-119	31	240-247
16	120-127	32	248-255

Si noti che lo schermo é riempito da caratteri spazio finché non si dispongono altri caratteri in alcune di queste posizioni.
Usando il gruppo di caratteri 5 nell'istruzione COLOR, tutti i caratteri spazio sullo schermo sono modificati con il colore dello sfondo specificato, dato che il carattere spazio é contenuto nel gruppo 5.
Ciò è dimostrato dal programma:

Esempio:
```
> 1ØØ REM COLOR DEMONSTRATION
> 2ØØ CLS
> 3ØØ COLOR 5, 6, 6,
> 4ØØ GO TO 4ØØ
> RUN
```
(il colore dello schermo passa da verde chiaro all'azzurro chiaro)

* Premere CTL-C per arrestare il programma.

CHAR

Il comando «CHAR» vi permette di definire i vostri particolari caratteri grafici. Si può ridefinire un nuovo gruppo di disegni di caratteri.

Sintassi: CHAR, codice-carattere, identificatore del disegno.

Il codice-carattere specifica il codice del carattere che si desidera definire; deve essere un'espressione numerica di valore compreso tra 0 e 255.
L'identificatore di disegno é un'espressione di 16 caratteri che specifica il disegno che si vuole usare nel programma. Questa espressione é una rappresentazione in codice dei 64 punti che compongono il disegno di un carattere sullo schermo. Questi 64 punti racchiudono un reticolo 8x8.

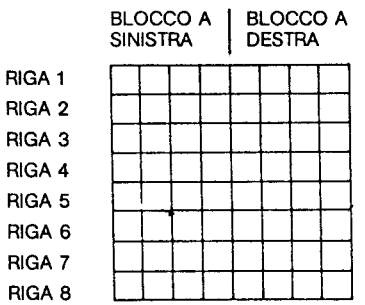
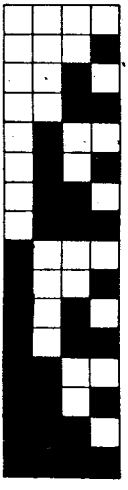


Fig. 13.2

Ogni riga é suddivisa in due blocchi di quattro punti ciascuno.

Fig. 13.3

Ogni carattere dell'espressione stringa descrive la configurazione dei punti in un blocco di una riga. Le righe vengono descritte da sinistra a destra e dall'alto in basso. Vale a dire che i primi due caratteri della stringa descrivono il disegno per la prima riga del reticolo di punti, i successivi due caratteri descrivono la seconda riga, e così via.
I caratteri sono creati «accendendo» alcuni punti e lasciando «spenti» gli altri. Per creare un nuovo carattere, occorre specificare al computer quali caratteri «accendere» o lasciare spenti in ciascuno dei 16 blocchi che compongono il carattere.
Se l'espressione stringa ha meno di 16 caratteri, il computer considera che i caratteri rimanenti siano zero. Se la stringa é più lunga di 16 caratteri, il computer ignora i caratteri in eccesso.



Blocchi	0=off (spento) 1=on (acceso)	Codice
	0000	0
	0001	1
	0010	2
	0011	3
	0100	4
	0101	5
	0110	6
	0111	7
	1000	8
	1001	9
	1010	A
	1011	B
	1100	C
	1101	D
	1110	E
	1111	F

Fig. 13.4

Esempio: per descrivere a punti la figura disegnata qui sotto.
CHAR 32, 18 18 FF 3C 7E FF 14 36

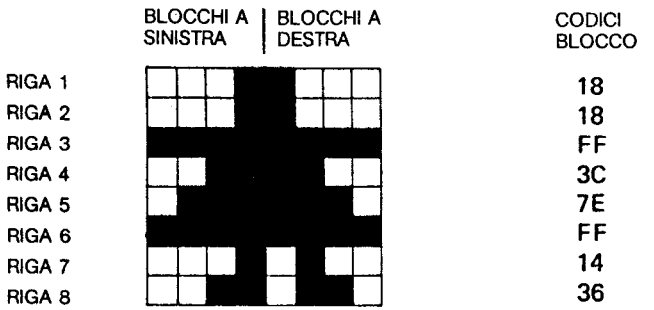


Fig. 13.5

DISEGNO PER IL CARATTERE CON CODICE 32

Si ricordi che il comando CHAR definisce soltanto il carattere.
Per visualizzare quel carattere sullo schermo, occorrerà usare il comando PLOT. Eseguendo il comando CHAR, tutti i caratteri già presenti sullo schermo con quello stesso codice di carattere vengono trasformati nel carattere nuovo.

Esempio: > *1Ø CLS*
> *15 COLOR 5, 4, 6*
> *2Ø CHAR 32, 18 18 FF 3C 7E FF 14 36*
> *3Ø GO TO 3Ø*
> *RUN*
(Lo schermo viene riempito col disegno sopra riportato, perché dopo il comando CLS lo schermo viene riempito di caratteri spazio, e lo spazio ha un codice di carattere 32 che é stato ridefinito col disegno di sopra).

*Si no*ti che i caratteri da 32 a 95 sono già definiti nel computer, Voi potete però ridefinirli come volete.
Nell'esempio soprastante, abbiamo ridefinito lo spazio con un carattere speciale. Premendo il pulsante reset, tutti i caratteri ridefiniti e il colore dello schermo vengono riportati ai valori iniziali.

PLOT

L'istruzione «PLOT» vi permette di posizionare un carattere in un punto qualunque dello schermo.

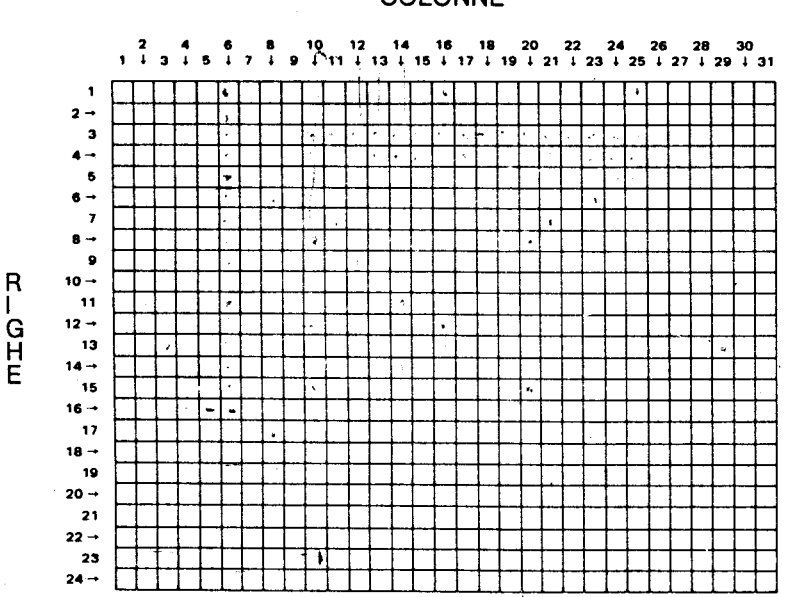


Fig. 13.6

Un valore di 1 per il numero della colonna indica la prima colonna a sinistra dello schermo.
Un valore di 1 per il numero della riga indica la prima riga in alto dello schermo. Lo schermo può essere anch'esso considerato un «reticolo» come é mostrato nella figura.
Dato che le colonne 1, 2, 31, 32 possono non comparire su alcuni schermi TV, si consiglia di usare solo le colonne di numero da 3 a 30.

Sintassi: PLOT numero delle colonne, numero della riga, codice del carattere.

Esempio: PLOT 15, 1Ø, 65

Questa istruzione fa comparire sullo schermo un carattere «A» nel punto di incontro della colonna numero 15 e della riga numero 10.

Esempio:
```
1Ø CLS
2Ø COLOR 5, 4, 6
3Ø CHAR 33, 18 18 FF 3C 7E FF 14 36
4Ø FOR X = 1 TO 32
5Ø PLOT X, 1Ø, 33
6Ø NEXT
7Ø GOTO 7Ø
RUN
```

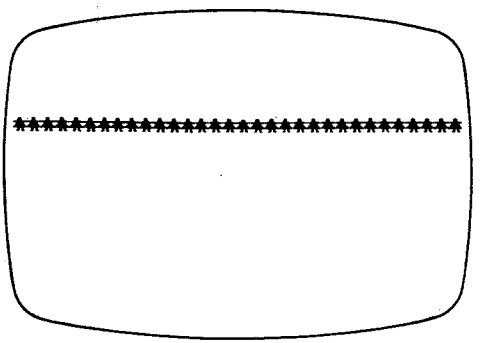


Fig. 13.7

I disegni formano una linea orizzontale che attraversa lo schermo.

Esempio:
```
Ø5 CLS
1Ø FOR A = 1 TO 16
2Ø COLOR A+16, A, A
3Ø NEXT
4Ø FOR X = 1 TO 32
5Ø FOR Y = 1 TO 2Ø
6Ø PLOT X, Y, X*4+128
7Ø NEXT
8Ø NEXT
9Ø GOTO 9Ø
RUN
```

Il programma mostra un gruppo di barre colorate sullo schermo TV.

SOUND

L'istruzione SOUND (suono) indica al computer di produrre suoni di diverse frequenze. Vi sono tre canali musicali, che si possono programmare indipendentemente.

Sintassi: SOUND codice di frequenza; codice di durata, codice di frequenza; codice di durata,...

Esempio: *1Ø SOUND 4;5, 6;7, 7;7*
(produce un suono di campane)

Il codice di frequenza e il codice di durata sono espressioni numeriche. Se il calcolo di qualche espressione numerica produce un risultato di valore non intero, tale risultato viene arrotondato per produrre un intero.

Codice di frequenza	
1 pausa	17 re diesis, mi bemolle
2 do	18 mi
3 do diesis, re bemolle	19 fa
4 re	20 fa diesis, sol bemolle
5 re diesis, mi bemolle	21 sol
6 mi	22 sol diesis, la bemolle
7 fa	23 la (successivo al do centrale)
8 fa diesis, sol bemolle	24 la diesis, si bemolle
9 sol	25 si
10 sol diesis, la bemolle	26 do (ottava del do centrale)
11 la (precedente al do centrale)	27 do diesis, re bemolle
12 la diesis, si bemolle	28 re
13 si	29 re diesis, mi bemolle1
14 do (do centrale)	30 mi
15 do diesis, re bemolle	31 fa
16 re	32 pausa

Codice di durata

Ø	¼	
1	½	
2	¾	
3	1	
4	1½	
5	2	
6	3	
7	4	

In un'istruzione SOUND, la prima coppia di dati si riferisce al primo canale, la seconda al secondo canale e la terza al terzo canale.

Esempio:
```
1Ø REM SONG
2Ø SOUND 26;3, 21;3, 14;3
3Ø SOUND 30;3, 25;3, 18;3
4Ø SOUND 28;3, 23;3, 16;3
5Ø SOUND 21;5, 16;5, 9;5
6Ø SOUND 26;3, 21;3, 14;3
7Ø SOUND 28;3, 23;3, 16;3
8Ø SOUND 30;3, 25;3, 18;3
9Ø SOUND 26;7, 21;7, 14;7
1ØØ GOTO 2Ø
RUN
```

Il computer suona la melodia mostrata in fig. 13.8

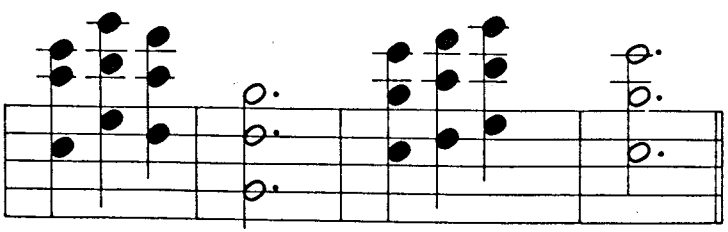


Fig. 13.8

JOY

La funzione JOY permette all'utente di introdurre nel computer informazioni basate sulla posizione della cloche. Questa funzione é utile all'utente per scrivere programmi di gioco.
La tastiera del Computer CreatiVision é divisa in due parti, una sinistra e una destra. Ogni parte ha una cloche e due pulsanti. La funzione cloche é adoperata per dare in input queste condizioni.

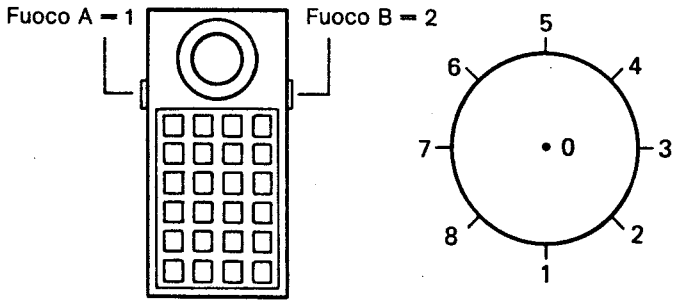


Fig. 13.8 Valori corrispondenti delle posizioni della cloche e dei pulsanti.

Sintassi: JOY (N) N = 1 per la posizione della cloche sinistra
N = 2 per la posizione della cloche destra
N = 3 per il pulsante del JOY-STICK di sinistra
N = 4 per il pulsante del JOY-STICK di destra

La funzione da il valore della posizione della cloche.

Esempio:
```
1Ø PRINT JOY (1)
2Ø GOTO 1Ø
RUN
3
.
.
.
```

Questo esempio produce la stampa della posizione della cloche sinistra. In questo caso, la cloche sinistra é spostata verso destra.

Esempio:
```
Ø5  CLS
1Ø  LET A=JOY (1)
2Ø  IF A=3 THEN GOTO 1ØØ
3Ø  IF A=7 THEN GOTO 2ØØ
40  GOTO 1Ø
1ØØ PLOT 4, 1Ø, 32
11Ø PLOT 28. 1Ø, 65
12Ø GOTO 1Ø
2ØØ PLOT 28, 1Ø, 32
21Ø PLOT 4, 1Ø, 65
22Ø GOTO 1Ø
RUN
```

Questo programma visualizza il carattere «A» a sinistra o a destra nello schermo TV in base alla posizione della cloche.

CAPITOLO 14

ACCESSO ALLA MEMORIA DI SISTEMA
– PEEK
– POKE

PEEK

PEEK é una funzione che dá in risposta il valore (numero intero di codice decimale compreso tra Ø e 255) letto nella posizione specificata della memoria.

Sintassi: PEEK (indirizzo).

È possibile per esempio vedere quale valore si trova in una determinata posizione di memoria.

Esempio:
```
1Ø PRINT «ADDRESS», «CONTENT»
2Ø FOR K = Ø TO 2Ø
3Ø PRINT K, PEEK (K)
4Ø NEXT K
```

POKE

Come spiegato più sopra, con la funzione PEEK si può leggere il contenuto di una posizione qualsiasi della memoria. La funzione di POKE é l'esatto complemento di PEEK. Con essa é possibile scrivere un dato di una posizione di memoria.

Sintassi: POKE, indirizzo, dato

Esempio: *POKE 1ØØØ, 48*

Il contenuto della posizione di memoria con indirizzo 1ØØØ viene sostituito dal valore 48. Se si digita.

PRINT PEEK (1ØØØ)

si ottiene il valore 48. (Provate a scrivere altri valori con l'istruzione POKE).

Nota: inserendo valori, con l'istruzione POKE, in determinate posizioni di memoria, si può «far impazzire» il computer. Per rimettere tutto a posto, premere RESET.

CAPITOLO 15

ESPANSIONE DEL SISTEMA CREATIVISION

Vi sono alcune possibilità opzionali, indicate più sotto, per espandere il vostro Sistema Creativision. Per maggiori particolari, si vedano i relativi manuali.

1. Modulo di Immagazzinamento su Cassetta (registratore) con il BASIC
   Creativision potete conservare i vostri programmi BASIC su cassette audio e ricaricali nel computer.
   Il funzionamento della cassetta può essere controllato da programma.
   La velocità di trasferimento dei dati é 600 baud (bit/secondo).
2. Interfaccia I/O Seriale e Parallela.
   Il modulo interfaccia I/O seriale e parallela consente un'ampia varietà di possibilità di input e di output. Il modulo ha una porta parallela per la vostra stampante. Questa porta é a standard «Centronics Bus», e si può quindi collegare a qualsiasi stampante Centronics Bus, per es. EPSOM MX80, SEIKOSHA GP-80, CENTRONICS 779, ecc.
   Per quanto riguarda le altre due porte, una é per il pilotaggio di Floppy Disk e una per Interfaccia Telefonica.
3. Modulo di Espansione Memoria.
   La memoria del Sistema Computer CreatiVision può essere ampliata aggiungendo un Modulo di Espansione Memoria al sistema.
   Il Modulo di espansione memoria è di 64 K byte.
4. Tastiera standard ASCII
   Una tastiera standard in gomma con sensibilità superiore e maggiore facilità nel digitare i dati.

APPENDICE A

TABELLA DELLE ISTRUZIONI BASIC

Il Manuale di Consultazione Rapida BASIC

I numeri sono immagazzinati con una precisione di 6 cifre. Il numero massimo ottenibile é 10^{38}. Tutte le istruzioni possono essere adoperate sia come comandi sia in un programma.

Funzioni:

1) Operatori artitmetici
   +, −, * , /, **
2) Operatori relazionali
   >, <, =, >=, <=, <>
3) Funzioni aritmetiche:
   SQR radice quadrata
   INT parte intera
   RND numero casuale
   ABS valore assoluto
   SGN segno
   COS coseno
   SIN seno
   EXP e^x
   TAN tangente
   LOG logaritmo naturale
4) Funzioni di stringa:
   LEN lunghezza
   STR$ stringa di un argomento numerico
   VAL valore numerico di una stringa
   ASC valore ASCII
   CHR$ carattere
   LEFT$ caratteri a sinistra
   MID$ caratteri in centro
   RIGHT$ caratteri a destra

5) Operatori logici
   AND
   OR Le espressioni relazionali o logiche hanno valore 1 se vere,
   Ø se false.
   NOT
6) Funzioni grafiche e sonore:
   CLS cancella lo schermo
   PLOT disegna un carattere sullo schermo
   COLOR predispone il colore
   SOUND produce suoni di differente frequenza e durata
   CHAR definisce un carattere
   JOY funzione Joy Stick
7) Istruzioni di programma
   DIM - dimensioni
   STOP
   END
   GOTO
   GOSUB
   RETURN
   FOR...TO...STEP
   NEXT
   REM
   IF...THEN
   INPUT
   PRINT
   PRINT TAB
   LET
   DATA
   READ
   RESTORE
8) Comandi:
   LIST
   RUN
   NEW
   CONT

CLOAD carica un programma presente su nastro
CSAVE immagazzina un programma su nastro
CRUN carica ed esegue un programma presente su nastro
CTL/C arresta il programma

9) Altre istruzioni
PEEK fornisce il valore depositato nella posizione di memoria indica indicata
POKE inserisce un valore in una posizione di memoria indicata
LPRINT stampa su stampante
LLIST lista del programma su stampante

APPENDICE B

MESSAGGI DI ERRORE

Messaggi di errore

Se il BASIC CreatiVison scopre un errore che provoca l'arresto dell'esecuzione del programma, viene stampata una segnalazione di errore. Qui sotto é fornita una spiegazione di ogni errore.

Messaggio di errore	
1. Next without for error	1. Istruzione next senza for
2. Without gosub error	2. Istruzione senza gosub
3. Missing line number error	3. Manca numero di riga
4. Missing operand error	4. Operando mancante
5. Syntax error	5. Errore di sintassi
6. Overflow error	6. Errore di overflow (n. troppo grande)
7. Illegal nested for error	7. Istruzione for nidificate (1ª entro l'altra)
8. Illegal nested gosub error	8. Istr. gosub nidificate non permesse
9. System error	9. Errore di sistema
10. Stack overflow error	10. Overflow della catasta di sistema
11. Illegal operand error	11. Operando non permesso
12. If error	12. Istruzione IF
13. ((.....)) error	13. Parentesi
14. ((.....)) level error	14. Livello parentesi
15. String not found error	15. Stringa non trovata
16. String evaluation error	16. Valutazione di stringa
17. Divided by zero error	17. Divisione per zero
18. Out of data error	18. Oltrepassamento istr. data
19. Data area overflow	19. Overflow (superamento dei limiti) zona dati
20. Dim error	20. Istruzione Dim
21. String length error	21. Lunghezza stringa

APPENDICE C

ESEMPI TIPICI DI PROGRAMMAZIONE

(i) Trovare il M.C.D. (massimo comune divisore)
(ii) Risoluzione equazioni di secondo grado
(iii) Trovare la radice quadrata con metodo iterativo
(iv) Trovare l'area di un triangolo
(v) Gioco Mark Six

Esempio (i):

```
1Ø REM FIND M.C.D.
2Ø INPUT X, Y
3Ø IF X>Y THEN X = X–Y
4Ø IF X<Y THEN Y = Y–X
5Ø IF X <> Y THEN 3Ø
6Ø PRINT X
7Ø END
RUN
? 3
? 9
  3
```

Esempio (iii):

```
1Ø REM: TAKE SQUARE ROOT
2Ø REM: BY ITERATIVE METHOD
3Ø REM: I = INITIAL VALUE
4Ø REM: N NUMBER
5Ø N = 2Ø
6Ø PRINT «VALUE I=»;
7Ø INPUT I
8Ø FOR J = 1 TO 1Ø
9Ø I = (I+N/I)*Ø.5
1ØØ PRINT J; «APPROXIMATION»; I
12Ø NEXT J
15Ø PRINT «SQUARE ROOT I =»; I
2ØØ END
RUN
VALUE I =
2Ø
1   APPROXIMATION 1Ø.5
2   APPROXIMATION  6.2Ø23
3   APPROXIMATION  4.7134
4   APPROXIMATION  4.4721
6   APPROXIMATION  4.4721
7   APPROXIMA TION  4.4721
8   APPROXIMATION  4.4721
9.  APPROXIMATION  4.4721
1Ø APPROXIMATION  4.4721
SQUARE ROOT I = 4.4721
```

Esempio (iv):

```
10 REM FIND AREA OF TRIANGLE
20 REM A, B, AND C ARE 3 SIDES
30 INPUT A, B, C
40 S = A+B+C
50 S = 0.5*S
60 P = S*(S-A)*(S-B)*(S-C)
70 A = SQR (P)
80 PRINT «AREA=»; A
90 END
RUN
A = 3
B = 4
C = 5
AREA = 6.0
```

* Questo programma non controlla se i valori A, B, C sono tali da poter formare un triangolo. Come esercizio, si possono aggiungere alcune istruzioni per questo controllo. (Perché i tre lati possano formare un triangolo deve essere: A minore di B+C; A maggiore di B−C; C minore di B+A).

Esempio (v):

```
1Ø REM MARK SIX GAME
2Ø DIM A(6)
3Ø FOR N = 1 TO 6
4Ø R = RND (Ø)*36
5Ø P = INT(R)
6Ø A(N) = P+1
7Ø IF N = 1 THEN 15Ø
8Ø M = N-1
9Ø J = Ø
1ØØ J = J+1
11Ø IF A(N) = A(J) THEN 4Ø
12Ø IF J<M THEN 1ØØ
15Ø PRINT A(N)
2ØØ NEXT N
3ØØ END
RUN
3
4
32
17  o qualsiasi numero casuale
24
15
```

APPENDICE D

CODICI ASCII

CODICE ASCII	CARATTERE	CODICE ASCII	CARATTERE
32	(Spazio)	64	@ (segno at, chiocciola)
33	! (punto esclamativo)	65	A
34	" (virgolette)	66	B
35	# (segno di numero o sterlina)	67	C
36	$ (dollaro)	68	D
37	% (per cento)	69	E
38	& (e commerciale)	70	F
39	' (apostrofo)	71	G
40	((parentesi aperta)	72	H
41	) (parentesi chiusa)	73	I
42	* (asterisco)	74	J
43	+ (più)	75	K
44	, (virgola)	76	L
45	— (meno)	77	M
46	. (punto)	78	N
47	/ (barra/diviso)	79	O
48	Ø	80	P
49	1	81	Q
50	2	82	R
51	3	83	S
52	4	84	T
53	5	85	U
54	6	86	V
55	7	87	W
56	8	88	X
57	9	89	Y
58	: (due punti)	90	Z
59	; (punto e virgola)		
60	< (minore di)		
61	= (uguale)		
62	> (maggiore di)		
63	? (punto interrogativo)		

CREATIVISION

ZANUSSI ELETTRONICA

Zanussi Elettronica spa
Viale Treviso 15
33170 PORDENONE PN

Modello

GCU 100

Codice
003590000 (S) ☐
013590000 (R) ☐

Matricola

Certificato di garanzia

Data di acquisto

Conservare questo certificato ed unirlo alla apparecchiatura difettosa insieme all'eventuale ricevuta fiscale o bolla di accompagnamento.

N. di ricevuta Fiscale o Bolla di Accompagnamento

Cognome e Nome

Timbro e firma del Rivenditore

Via N.

Località

C.A.P. Provincia

Limiti di garanzia

La Zanussi Elettronica declina qualsiasi responsabilità nei confronti dell'acquirente o di qualsiasi altra persona o ente di riguardo per perdite o danni causati direttamente dal presente prodotto, compresi, senza peraltro limitarsi ad essi, qualsiasi interruzione di servizio, e danni conseguenti derivanti dall'uso o dall'effetto del presente prodotto.
Il presente prodotto sarà sostituito entro 3 mesi dalla data dell'acquisto se presenta difetti di fabbricazione, ma, eccettuata tale sostituzione, la vendita o la successiva sistemazione del presente programma non è coperta da garanzia.

ZANUSSI ELETTRONICA
Zanussi Elettronica spa

La Casa costruttrice declina ogni responsabilità per le possibili inesattezze contenute nel presente opuscolo, imputabili ad errori di stampa o di trascrizione. Si riserva il diritto di apportare ai propri prodotti quelle modifiche che ritenesse necessarie o utili, senza pregiudicarne le caratteristiche essenziali.

Cod. 46.5407.000

91-92-07