INTERPRETE «BASIC»

Vi presentiamo un interprete BASIC da 5,5 K tramite il quale potremo dialogare con il nostro microcomputer non più in esadecimale, bensì con delle parole di uso corrente che scriveremo sulla tastiera alfanumerica. Tale interprete è una novità in senso assoluto in quanto è «bilingue», cioè è in grado di accettare e di eseguire ordini sia in italiano che in inglese.

Sappiamo già che un microcomputer è in grado di comprendere solo il linguaggio macchina, cioè un linguaggio fatto esclusivamente di numeri binari, quindi ben diverso da quello usuale che noi adottiamo normalmente per comunicare con altre persone.

Sappiamo inoltre che affinché un microcomputer possa eseguire un qualcosa che noi gli ordiniamo è assolutamente necessario indicargli passo per passo come deve procedere per esplicare il compito affidatogli.

Per esempio se il microcomputer fosse il cameriere di un ristorante e noi fossimo il direttore, non potremmo limitarci a dirgli «servi il cliente», bensì dovremmo precisargli ogni volta tutto ciò che deve fare, cioè:

1) va al tavolo del cliente
2) chiedi se vuole l'antipasto
3) trascrivi su un foglio di carta quanto ti ordina
4) se vuole il vino chiedi se lo preferisce bianco o rosso
5) chiedi che tipo di minestra vuole
6) porta al tavolo forchetta, coltello, ecc.
7) va in cucina e metti nel piatto ciò che il cliente ha ordinato per antipasto
8) porta questo piatto al cliente poi leggi nella lista cosa ti ha ordinato come minestra e inizia a prepararla
9) quando il cliente ha finito l'antipasto togli il piatto e sostituisci le posate
10) versa la minestra nel piatto e servila
11) portala al tavolo del cliente, ecc.

In altre parole noi dovremmo specificare a questo cameriere ogni più piccolo particolare perché se ci dimenticassimo per esempio di dirgli «metti la minestra nel piatto», questi porterebbe al cliente il piatto vuoto oppure la minestra ancora nella pentola in quanto è talmente «stupido» che di sua iniziativa non prende alcuna decisione.

La grossa complicazione comunque non è neppure il dovergli dire tutte queste cose, quanto dovergliele dire in linguaggio macchina, cioè in binario, in quanto questo ci obbliga, per ogni istruzione, a scrivere una lunga sequenza di 1 e di 0.

Per alleviare questo fastidio noi abbiamo introdotto fin dal primo momento nel nostro microcomputer la tastiera esadecimale la quale, se vi ricordate l'esempio del ristorante cinese riportato sul n. 69 di Nuova Elettronica, si comporta in pratica come un «interprete francese» in grado di parlare e di farsi comprendere dai camerieri del ristorante i quali ovviamente sanno parlare solo il «cinese».

In pratica però con la tastiera esadecimale il problema è risolto solo a metà in primo luogo perché con questa tastiera noi dobbiamo ancora scrivere le istruzioni in «esadecimale», cioè in un linguaggio diverso dal nostro, anche se meno complicato del linguaggio macchina, cioè del binario, ed in secondo luogo perché la tastiera esadecimale rimane sempre un interprete non «specializzato», cioè si limita a tradurre parola per parola ciò che noi le diciamo ma non è in grado di aggiungere nulla di proprio a queste istruzioni.

Per esempio se noi vogliamo che un cameriere serva un cliente al tavolo dobbiamo sempre ripetere tutte le istruzioni che abbiamo visto in precedenza e l'unica agevolazione che ci viene fornita dalla tastiera esadecimale è che invece di dover scrivere, per ognuna di queste istruzioni, dei codici binari a 8 cifre, possiamo scrivere dei codici a 2 cifre.

Ovviamente ripetere sempre le stesse cose soprattutto quando queste sono ovvie e assodate è una enorme «scocciatura» nonché una enorme perdita di tempo infatti vi sarete già accorti che programmare il microcomputer in esadecimale, anche se dà maggior soddisfazione in quanto permette effettivamente di rendersi conto di ciò che avviene all'interno della macchina, in realtà è sempre molto laborioso e difficile.

Se invece potessimo disporre di un **interprete** «esperto del mestiere» al quale fosse sufficiente dire: «Servi il cliente» perché questi automaticamente si preoccupasse di fornire al cameriere in

Fornendovi insieme alla scheda video anche l'interprete BASIC possiamo dire di aver già completato al 90% il nostro microcomputer in quanto possiamo già lavorare con la tastiera alfanumerica, il monitor e la stampante realizzando così i primi programmi gestionali. Possiamo inoltre anticiparvi che gli attuali 5,5 K di BASIC sono destinati ad espandersi in quanto sui prossimi numeri, quando avrete una maggior dimestichezza con il computer, vi insegneremo ad aggiungere voi stessi, in coda a queste, altre «subroutine» che vi permetteranno di eseguire un numero maggiore di funzioni.

BILINGUE da 5,5 K

cinese tutte le istruzioni al riguardo, sarebbe senz'altro una grossa agevolazione nonché un notevole risparmio di tempo.

Se poi questo interprete, oltre ad essere un esperto del mestiere, quindi in grado di fare tutto ciò che abbiamo appena accennato, fosse anche **italiano**, quindi ci potesse capire immediatamente senza dover utilizzare come tramite per i nostri colloqui la lingua francese o inglese, l'agevolazione sarebbe doppia in quanto ci eviterebbe anche questo lavoro supplementare.

Ebbene noi oggi siamo in grado di fornirvi tutto questo, cioè un interprete «italiano» ed «esperto del mestiere» il quale vi permetterà di parlare con il vostro microcomputer proprio come se parlaste con un amico, senza cioè dover spendere ogni volta una montagna di parole per farvi comprendere.

Tanto per rendervi un'idea dei vantaggi che possono derivare dall'impiego di questo interprete, pensate solo un attimo a quante istruzioni sono necessarie programmando in esadecimale, per far stampare il listing di un programma (vedi in coda all'articolo «interfaccia per stampante» su questo stesso numero).

Con l'interprete BASIC invece basta dire al microcomputer **SLISTA** (cioè Stampa la Lista) ed automaticamente la stampante inizierà a scrivere sulla carta il listing del programma presente in memoria; se poi il listing del programma vogliamo che appaia sul video dovremo scrivere semplicemente **LISTA** ed automaticamente sul video ci appariranno le prime 8 linee di questo programma dopodiché pigiando il tasto **ESC** potremo far comparire la 9° linea, poi la 10°, l'11° e così di seguito.

Come vedete il risparmio di tempo è notevole in quanto si tratta di scrivere una sola istruzione e per lo più in linguaggio corrente, invece di decine e decine di istruzioni in un linguaggio astruso come lo è appunto il linguaggio macchina.

Le agevolazioni comunque non finiscono qui infatti quando sarete più esperti di programmazione, invece di scrivere ogni volta la parola LISTA per intero, potrete scrivere più semplicemente L. (cioè una L seguita da un punto) in quanto il nostro interprete è così intelligente che vi capirà al volo senza che sia necessario concludere la parola.

Tutto ciò potrebbe sembrare un'inezia però se pensate di dover scrivere un programma che di queste istruzioni ne comprende per esempio un centinaio, capirete che risparmiare mediamente 3 battute sulla tastiera per ciascuna istruzione significa alla fine risparmiare un qualcosa come 3x100=300 battute occupando inoltre una minor quantità di memoria con il programma, memoria che ovviamente resterà libera per i calcoli che si debbono eseguire.

Un'altra avevolazione di questo linguaggio BASIC è quella di poter accettare istruzioni indifferentemente in italiano (e questa è una novità in senso assoluto) oppure se lo preferite, in inglese, pertanto chi è abituato a programmare già da tempo in BASIC su computer commerciali potrà conservare i propri programmi in inglese senza doverli modificare, mentre chi è nuovo del mestiere e non conosce l'inglese, potrà programmare direttamente in italiano, evitando così noiose traduzioni mentali.

Con il BASIC inoltre, se ci si sbaglia a programmare o si dimentica un'istruzione, non si corre il

rischio, come avveniva con la tastiera esadecimale, di distruggere il programma in memoria oppure di doverlo riscrivere completamente per inserire i passi mancanti: un programma scritto in BASIC infatti non può autodistruggersi quindi può essere corretto in qualsiasi istante aggiungendo o togliendo delle istruzioni e se per caso c'è un'istruzione sbagliata sarà il computer stesso a **segnalarcelo** facendo comparire sul video la scritta: COME? proprio nel punto esatto di programma in cui c'è l'errore, pertanto risulterà facilissimo porvi rimedio.

Tanto per fare un esempio, se invece di dire all'interprete «Servi il cliente», noi dicessimo «metti il cliente in pentola», l'interprete essendo un esperto del mestiere e sapendo che in pentola ci si può mettere solo il manzo, il pollo o le patate, ci risponderà subito: «IMPOSSIBILE» facendo comparire questa scritta sul video.

Come già anticipato l'interprete BASIC che noi forniamo, una volta caricato in memoria, occupa 5,5 K di RAM, quindi è un interprete molto più specializzato rispetto ai normali BASIC da 2 o 3 K che generalmente vengono forniti in dotazione ai microcomputer commerciali ed essendo più specializzato è ovvio che ci permetterà di compiere un numero maggiore di operazioni.

A questo punto qualcuno obietterà che esistono in commercio anche dei BASIC da 8 K, ma qui vorremmo aggiungere che finché non è presente il floppy-disk, un BASIC da 5,5 K è più che sufficiente soprattutto se si tien conto che questo è «espandibile», cioè sui prossimi numeri vi presenteremo dei «programmi» che potrete aggiungere in coda a questo BASIC da 5,5 K trasformandolo così ben presto in un BASIC da 7-8 K.

Possiamo inoltre anticiparvi fin d'ora che quando presenteremo il floppy disk vi forniremo insieme ad esso anche un BASIC da 16 K, cioè un **maxi** BASIC.

Prima di passare alle caratteristiche vi precisiamo che il nostro interprete BASIC viene fornito su cassetta magnetica insieme all'interfaccia video (registrato alla solita velocità di 300 baud) quindi per poterlo utilizzare occorre ogni volta caricarlo sulle memorie del microcomputer leggendolo appunto dal registratore tramite l'interfaccia cassette LX385.

Una volta caricato in memoria il BASIC occupa, come abbiamo già detto, 5,5 K di memoria RAM, quindi per poterlo utilizzare è sufficiente avere montata sul BUS una sola scheda di espansione da 8 K (è ovvio però che montando 2 di queste schede avremo un maggior margine per i nostri programmi).

Nota: si consiglia per il momento di non montare sul BUS più di 2 schede di espansione della memoria.

CARATTERISTICHE del nostro BASIC

Come già detto il BASIC di Nuova Elettronica ha una caratteristica unica: la possibilità cioè di accettare istruzioni sia in italiano che in inglese.

In pratica il BASIC, appena caricato in memoria, si predispone per rispondere ai comandi in **italiano** tuttavia è sufficiente che il programmatore scriva sulla tastiera alfanumerica **ENG** e pigi subito dopo il tasto **RETURN** per passare automaticamente in lingua inglese.

Una volta che si è scelta la lingua, cioè inglese o italiano, tutte le istruzioni del programma che si vuole utilizzare debbono essere scritte in quella medesima lingua, cioè se abbiamo scelto l'inglese il programma deve essere scritto in inglese, viceversa se abbiamo scelto l'italiano il programma deve essere scritto in italiano.

Per eseguire il passaggio inverso, cioè per passare dall'inglese all'italiano, è invece sufficiente scrivere sulla tastiera alfanumerica la parola **ITA** e pigiare ancora il tasto **RETURN**.

Nella versione attuale il nostro BASIC permette di effettuare le seguenti operazioni:

1) di disegnare sullo schermo dei grafici avendo a disposizione 64x48 quadratini che possono essere accesi o spenti indirizzandoli uno per uno

2) di leggere e scrivere i programmi su nastro mediante l'interfaccia cassette già presentata sul n. 70 della rivista

3) di utilizzare nei programmi un max di 26 variabili numeriche

4) di compiere operazioni con dati interi e in virgola mobile con una precisione di 6 cifre

5) di utilizzare vettori e variabili alfanumeriche

6) di pilotare la stampante collegata all'apposita interfaccia presentata su questo stesso numero

7) di abbreviare le istruzioni in modo da occupare meno memoria con i programmi complessi

8) di registrare dati su nastro e rileggerli in qualsiasi momento

9) di chiamare delle subroutine

10) di eseguire listing di programmi ecc. ecc.

COME SI CARICA IL BASIC IN MEMORIA

Per caricare il BASIC in memoria e per poterlo quindi utilizzare occorre che il microcomputer sia completo delle seguenti parti:

1) Alimentatore e BUS

2) Scheda CPU

3) Tastiera esadecimale e relativa interfaccia

4) Interfaccia cassette con relativo registratore collegato

5) Interfaccia video

6) Tastiera alfanumerica

7) Monitor video o TV

8) Almeno 8 K di memoria RAM.

IMPORTANTE: Sulla scheda di memoria debbono essere stati effettuati tutti e tre i ponticelli P1-P2-P3.

Se si utilizzano due schede di espansione, sulla seconda dovranno essere effettuati solo i ponticelli P2-P3 e lasciato libero P1.

Ovviamente con tutte queste schede montate sul BUS è più che normale che tenendo acceso il microcomputer per qualche ora, l'alimentatore tenda a scaldare, quindi sarebbe consigliabile applicargli frontalmente una ventola per raffreddarlo.

A questo punto potremo iniziare il caricamento del BASIC procedendo come segue:

1) Inserite la cassetta del BASIC nel registratore e riavvolgete completamente il nastro

2) Pigiate sulla tastiera **esadecimale** i tasti CONTROL-6 poi pigiate sul registratore il tasto PLAY (cioè ascolto) avendo cura di controllare che il volume sia regolato in modo opportuno

3) Pigiate sulla tastiera esadecimale il tasto A ~~o B a seconda dell'uscita a cui vi siete collegati sull'interfaccia cassette~~ automaticamente il registratore si metterà in funzione mentre sui display vedrete scorrere i numeri corrispondenti agli indirizzi che di volta in volta vengono letti.

Durante questa fase sul video vi apparirà un rettangolo di caratteri del tutto casuali, cioè dei caratteri alfanumerici o grafici qualsiasi, mentre sulla destra in basso dello schermo noterete una riga di caratteri che cambiano in continuazione.

4) Se non avvengono errori la lettura si fermerà all'indirizzo 03FF e dopo qualche secondo anche il registratore si arresterà.

A questo punto, senza toccare i comandi del registratore e soprattutto senza spegnerlo, pigiate sulla tastiera esadecimale per due volte consecutive i tasti CONTROL-4.

5) Immediatamente sui display della tastiera esasecimale vi apparirà la scritta BASIC e il registratore riprenderà a girare per fermarsi definitivamente dopo circa 4 minuti.

Durante questa fase sullo schermo del monitor vi apparirà al centro un rettangolo con tutti i caratteri alfanumerici in positivo e negativo più la scritta BASIC NUOVA ELETTRONICA ecc. ecc.

6) Quando il registratore si fermerà, sullo schermo del monitor in alto a sinistra vi apparirà la scritta: BASIC V.1.0 in negativo (cioè con caratteri neri su fondo bianco) per indicarvi che potete iniziare a programmare.

ATTENZIONE: una volta caricato il BASIC in memoria non pigiate per nessun motivo il tasto RESET sulla tastiera esadecimale poiché così facendo il controllo del microcomputer ritorna al programma «monitor» contenuto nella EPROM, cioè all'interprete esadecimale e sui display appare la solita «n» seguita da 7 trattini orizzontali.

In queste condizioni, per rimettere il BASIC in condizione di funzionare occorre scrivere nel registro SP (Stack Pointer), servendosi sempre della tastiera esadecimale, il numero 1600, e nel registro PC (Program Counter) il numero 1000, dopodiché è sufficiente premere i due tasti CONTROL-4.

Così facendo, se il BASIC non si è «sporcato», sul video comparirà nuovamente la scritta:

BASIC V.1.0.

Se invece non compare questa scritta significa

Tabella riassuntiva delle istruzioni

Italiano	Abbrev.	Inglese	Abbrev.
ACCENDI	A.	SET	S.
AL	AL	AT	AT
ALLORA	A.	THEN	T.
ASS	A.	ABS	A.
CANCELLA	CA.	NEW	N.
CARICA	CAR.	CLOAD	CL.
CAS	C.	RND	R.
CHIAMA	C.	GOSUB	GOS.
CONTINUA	C.	CONT	C.
DATI	D.	DATA	D.
DUP	D.	DUP	D.
ENG	E.	ITA	I.
ESEGUI	ES.	RUN	R.
FINE	F.	END	E.
FINO	F.	TO	TO
INCREMENTA	I.	NEXT	N.
INT	I.	INT	I.
LEGGI	L.	READ	REA.
LISTA	L.	LIST	L.
MEM	M.	MEM	M.
NOTA	N.	REM	REM
PASSO	P.	STEP	S.
PER	P.	FOR	F.
PUNTO	P.	POINT	P.
RC	RC	RC	RC
REGISTRA	R.	CSAVE	CS.
RICHIEDI	R.	INPUT	I.
RILEGGI	RIL.	RESTORE	REST.
RITORNA	RIT.	RETURN	RET.
SIA	SIA	LET	L.
SCRIVI	S.	PRINT	P.
SE	SE	IF	IF
SECONDO	SEC.	ON	ON
SLISTA	S.	LLIST	LL.
SPEGNI	SP.	RESET	R.
SPSC	SPSC	CLS	C.
SSCRIVI	SS.	LPRINT	LP.
STOP	ST.	STOP	ST.
TAB	T.	TAB	T.
UT	UT	USR	U.
VA A	V.	GOTO	G.

In questa tabella sono riportate tutte le istruzioni del nostro BASIC sia in italiano che in inglese, comprese le abbreviazioni che potremo utilizzare per occupare meno memoria RAM.
Come potrete constatare le abbreviazioni debbono essere tutte completate con un «punto», diversamente l'interprete non sarà in grado di riconoscerle (vedi ad esempio CANCELLA = CA. oppure CARICA = CAR.).
Per quanto riguarda le abbreviazioni che usano la stessa lettera, come per esempio P. (PASSO-PER-PUNTO) sarà il microcomputer, controllando ciò che segue, a decidere ogni volta ciò che deve fare.
Sui prossimi numeri vi spiegheremo come si impostano i programmi in BASIC.

che il BASIC stesso in seguito al RESET si è cancellato quindi occorre ricaricarlo in memoria.

Ovviamente se durante la lettura del BASIC da cassetta il registratore improvvisamente si ferma senza che sul video compaia la scitta BASIC V.1.0 significa che si è verificato un errore di lettura pertanto in questo caso occorrerà ricominciare il tutto daccapo cioè ripetere dall'inizio tutte le operazioni precedentemente descritte per il caricamento del programma BASIC.

LE ISTRUZIONI

Dopo avervi parlato in generale del nostro BASIC ed avervi insegnato a caricarlo in memoria leggendolo dal nastro, passiamo ora ad occuparci di tutti i comandi (vedi tabella a pag. 101) che il BASIC stesso, nella versione attuale, è in grado di riconoscere e di eseguire.

Qui di seguito vi spiegheremo in dettaglio a cosa serve e come deve essere utilizzata ciascuna di queste istruzioni.

Istruzioni e comandi del BASIC

Italiano	Abbrev.	Inglese	Abbrev.
ITA	I.	ENG	E.

Sia ITA che ENG sono due comandi che possono essere eseguiti solo da **tastiera** e ci permettono, come già accennato in precedenza, di scegliere rispettivamente l'interprete **«italiano»** oppure l'interprete **«inglese»**.

Quando noi carichiamo il BASIC da cassetta questo automaticamente si predispone per ricevere comandi in italiano, pertanto se volessimo programmare in inglese dovremo scrivere sulla tastiera alfanumerica la parola **ENG** (o più semplicemente E.) e pigiare quindi il tasto RETURN, dopodiché potremo iniziare a scrivere tranquillamente il nostro programma, assegnando a ciascuna istruzione il proprio numero d'ordine (vedi oltre).

Per eseguire il passaggio inverso, cioè per poter programmare in italiano quando si è scelto in precedenza l'inglese, occorre invece scrivere sulla tastiera alfanumerica la parola **ITA** (o più semplicemente I.) e pigiare ancora il tasto RETURN.

LISTA	L.	LIST	L.

Anche questi sono comandi che possono essere eseguiti solo da tastiera, cioè non è possibile inserirli nel corso di un programma.

Scrivendo sulla tastiera alfanumerica LISTA e pigiando il tasto RETURN automaticamente vedremo comparire sul video le prime 8 linee di programma inserite in precedenza in memoria.

A questo punto se ci interessa vedere anche le righe successive, non dovremo fare altro che pigiare il tasto **ESC** posto sulla sinistra ed automaticamente comparirà in basso la 9ª riga di programma.

Continuando a pigiare **ESC** noi faremo ovviamente comparire in basso la 10ª, la 11ª, la 12ª riga fermo restando che non appena il quadro sarà pieno scomparirà in alto la 1ª riga, poi la 2ª riga, la 3ª, la 4ª ecc.

Ovviamente se avessimo in precedenza scelto l'interprete «inglese» e per errore scrivessimo sulla tastiera LISTA invece di LIST, il computer anziché mostrarci le righe di programma sul video, farà comparire sul video stesso la scritta COME? segnalandoci l'errore di scrittura da parte nostra.

LISTA nn	L. nn	LIST nn	L. nn

Queso comando è simile al precedente come effetto solo che a differenza del precedente ci permette di far comparire sul video il listing di un programma a partire da una determinata riga (che noi abbiamo indicato genericamente con «nn») anziché dalla prima istruzione.

Per esempio se noi avessimo scritto in memoria un programma che inizia dalla riga 10 e termina alla riga 400 e volessimo controllare solo le ultime istruzioni di questo programma, cioè quelle dalla riga 350 in poi, è inutile che facciamo il listing del programma dall'inizio e pigiamo poi tante volte il tasto ESC quanto è necessario per arrivare alla riga 350; meglio scrivere subito **LISTA 350** e pigiare il tasto RETURN in quanto così facendo ci verranno visualizzare solo le righe dalla 350 in poi.

Nota: Sia LISTA che LISTA nn, funzionano solo se in precedenza abbiamo scritto in memoria un qualsiasi programma in BASIC.

ESEGUI	ES.	RUN	R.

Anche questo comando può essere eseguito solo immediatamente da tastiera, cioè non può far parte di un programma scritto in memoria.

Scrivendo sulla tastiera alfanumerica ESEGUI (oppure anche solo ES.) e pigiando il tasto RETURN, automaticamente il programma inserito in precedenza in memoria verrà fatto eseguire dal computer partendo dal numero di linea più basso. È ovvio che se noi avessimo scelto la lingua inglese, anziché scrivere ESEGUI, dovremmo scrivere RUN (cioè la versione inglese di ESEGUI).

ESEGUI nn	ES. nn	RUN nn	R. nn

Questo comando è simile al precedente con la sola differenza che anziché far eseguire il programma presente in memoria a partire dal numero di linea più basso, fa eseguire questo programma partendo dal numero di linea specificato con «nn». Per esempio scrivendo sulla tastiera **ESEGUI 150** e pigiando il solito tasto RETURN noi faremo ese-

guire il programma in memoria a partire dalla riga 150, proprio come se le righe precedenti non esistessero.

CONTINUA	C.	CONT	C.

Anche questo comando può essere eseguito solo immediatamente da tastiera e ci permette in pratica di far ripartire l'esecuzione di un programma dal punto in cui si era arrestata in seguito ad un'istruzione di STOP (vedi oltre).

In altre parole se nel corso di un programma si incontra un'istruzione di STOP, l'esecuzione si interrompe e sul video compare la scritta **FERMO A** seguita da un numero di riga, per esempio 150 per indicarci che il programma è fermo appunto alla riga 150.

Una volta che il programma si è fermato in seguito ad uno STOP è comunque possibile farlo ripartire dallo stesso punto scrivendo sulla tastiera CONTINUA e pigiando il tasto RETURN.

CANCELLA	CA.	NEW	N.

Anche questo comando può essere eseguito solo da tastiera e ci permette in pratica di cancellare tutta la memoria RAM a disposizione dell'utente in modo tale da poter scrivere nuovi programmi.

Per esempio se dopo aver fatto un certo tipo di lavoro con un programma, noi volessimo passare ad eseguire un altro lavoro con un programma diverso, prima di scrivere in memoria questo nuovo programma dovremo ovviamente cancellare il precedente scrivendo sulla tastiera alfanumerica **CANCELLA** (o più semplicemente CA.) e pigiando quindi il tasto RETURN.

CARICA	CAR.	CLOAD	CL.

Questo comando può essere eseguito solo da tastiera e ci permette di caricare nella memoria RAM a disposizione dell'utente un determinato programma leggendolo dal nastro.

Per esempio se noi avessimo registrato sul nastro un programma per effettuare la fatturazione e volessimo caricare questo programma in memoria, per raggiungere il nostro scopo dovremmo inserire la cassettina sul registratore, riavvolgerla completamente, poi scrivere sulla tastiera alfanumerica **CARICA** (o più semplicemente CAR.) e pigiare quindi il tasto RETURN.

Così facendo il registratore si metterà in moto ed il programma verrà caricato in memoria nel giro di qualche minuto (vedi a fine articolo).

REGISTRA	R.	CSAVE	CS.

Questo comando, che come il precedente può essere eseguito solo da tastiera, ci permette di registrare sul nastro un programma da noi scritto in BASIC nella memoria RAM.

Per effettuare la registrazione dovremo inserire un nastro vergine sul registratore, riavvolgerlo completamente, poi scrivere sulla tastiera alfanumerica **REGISTRA** (o più semplicemente R.) e pigiare quindi il tasto RETURN.

Così facendo il registratore automaticamente si metterà in moto e dopo qualche minuto, a seconda della lunghezza del programma, si fermerà per indicarci che il programma è già stato completamente registrato.

SIA	SIA	LET	L.

Questa istruzione, per poter essere eseguita, deve necessariamente trovarsi all'interno di un programma ed è la tipica istruzione utilizzata nel BASIC per assegnare un valore ad una variabile.

Per esempio se noi vogliamo utilizzare in un programma la variabile X e vogliamo che il valore iniziale di questa variabile sia 15, dovremo scrivere la seguente istruzione:

10 SIA X = 15

dove il 10 è il numero di linea preso come esempio il quale ovviamente cambia da programma a programma.

Precisiamo comunque che nel nostro BASIC, per assegnare un valore ad una variabile, non è indispensabile scrivere SIA, infatti rifacendoci all'esempio precedente noi potremmo scrivere più semplicemente:

10 X = 15

ed automaticamente avremmo assegnato alla variabile X il valore 15.

STOP	ST.	STOP	ST.

Questa istruzione, se inserita nel corso di un programma, per esempio:

115 STOP

provoca l'interruzione del programma stesso in corrispondenza della riga 115 e contemporaneamente fa apparire sul video la scritta:

FERMO A 115.

In tal caso, per far ripartire il programma, abbiamo già visto che occorre scrivere sulla tastiera alfanumerica CONTINUA poi battere il tasto RETURN.

L'istruzione di **STOP** viene generalmente utilizzata per scoprire eventuali errori in un programma infatti inserendola in due o tre punti diversi si può facilmente individuare qual'è la parte di programma che viene eseguita correttamente e quale invece no.

RICHIEDI	R.	INPUT	I.

Quando incontra questa istruzione nel corso di

un programma il computer fa apparire sul video un punto interrogativo quindi si mette in attesa che noi gli forniamo tramite la tastiera il valore da assegnare alla variabile indicata in tale istruzione subito dopo RICHIEDI.

Per esempio se in un programma è inserita l'istruzione:

25 RICHIEDI C

arrivato alla riga 25 il computer si ferma e fa apparire sul video un punto interrogativo.

A questo punto se noi vogliamo assegnare alla variabile C il valore 3395, dovremo scrivere sulla tastiera alfanumerica 3395 poi pigiare il tasto RETURN.

Così facendo il progamma riprenderà a girare con la variabile C = 3395.

RICHIEDI"xx"	R."xx"	INPUT"xx"	I. "xx"

Questa istruzione è assolutamente identica alla precedente con la sola differenza che quando la incontra nel corso di un programma il computer, anziché far comparire sul video un solo punto interrogativo, prima di questo punto interrogativo fa comparire la frase che noi abbiamo scritto fra virgolette.

Per esempio se alla riga 120 di un programma il computer trova l'istruzione:

120 RICHIEDI "GIACENZA"; B

automaticamente fa comparire sul video la scritta: GIACENZA? quindi si mette in attesa che noi gli forniamo il valore da assegnare alla variabile B.

A questo punto se noi scriviamo sulla tastiera il numero 1500, quindi pigiamo il tasto RETURN automaticamente il computer assegnerà alla variabile B (che per noi rappresenta la giacenza) il valore 1500 e passerà quindi ad eseguire l'istruzione successiva.

CHIAMA n	C.	GOSUB n	GOS.

Quando il computer, nel corso di un programma, incontra questa istruzione, automaticamente passa ad eseguire la «subroutine» che inizia alla riga «n» poi quando trova un'istruzione RITORNA, ritorna al programma principale ed esegue l'istruzione che viene subito dopo CHIAMA.

Esempio:

```
100 CHIAMA 260
110 STOP
.....
.....
260 A = 22
270 RITORNA
```

Se noi avessimo inserito in memoria un programma con queste istruzioni, quando il computer arriva alla riga 100, automaticamente passa ad eseguire la «subroutine» posta alla riga 260, cioè assegna alla variabile A il valore 22 poi ritorna al programma iniziale ed esegue l'istruzione posta alla riga 110 che essendo uno STOP lo costringe a fermarsi facendo comparire sul video la scritta FERMO A 110.

RITORNA	RIT.	RETURN	RET.

Questa istruzione, come abbiamo visto nell'esempio precedente, deve sempre essere inserita alla fine di ogni subroutine in quanto è lei che permette al computer di ritornare ad eseguire il programma principale partendo dalla linea immediatamente successiva a quella in cui è posta l'istruzione CHIAMA.

ACCENDI	A.	SET	S.

Questa istruzione, posta nel corso di un programma, ci permette di accendere uno qualsiasi dei 64 x 48 quadratini in cui viene suddiviso lo schermo del monitor video nel funzionamento in semigrafico.

Le coordinate di questo quadratino debbono essere indicate fra parentesi dopo ACCENDI, per esempio:

```
50 X = 24
60 Y = 35
70 ACCENDI (X, Y)
80 STOP
```

Questo programma ci permetterà di accendere il quadratino 24 della riga 35 infatti avendo posto X = 24 e Y = 35 è come se noi dicessimo al computer: ACCENDI (24, 35).

Precisiamo che la numerazione delle righe, così come quella dei quadratini, inizia sempre da 0, pertanto noi avremo le righe da 0 a 47 e i quadratini da 0 a 63.

SPEGNI	SP.	RESET	R.

Questa istruzione è esattamente l'inverso della precedente infatti ci permette di spegnere uno qualsiasi dei quadratini presenti sullo schermo.

Per esempio se il computer alla riga 50 trova l'istruzione:

```
50 SPEGNI (28, 16)
```

arrivato a questo punto del programma provvede a spegnere il quadratino 28 della riga 16.

NOTA	N.	REM	REM

Questa non è un'istruzione vera e propria bensì serve al programmatore per inserire nel mezzo di un programma dei pezzi di «commento» al programma stesso in modo da potersi ricordare per esempio a cosa serve un determinato gruppo di istruzioni oppure il significato attribuito in tale programma ad una determinata variabile.

Esempio:

```
100 A = 22500
```

110 NOTA A = PREZZO VENDITA
120 B = 18000
130 NOTA B = PREZZO D'ACQUISTO
140 FINE

Durante l'esecuzione di questo programma il computer ignora le righe 110 e 130 le quali però compaiono regolarmente nel listing del programma stesso per ricordare al programmatore il significato attribuito alle variabili A e B.

SCRIVI	S.	PRINT	P.

È una delle istruzioni più usate nel corso di un programma e serve per visualizzare sul monitor il valore di una variabile o un numero qualsiasi.

Esempio:
150 A = 57
160 B = 223
170 SCRIVI A; B; 49
180 FINE

Facendo eseguire questo programma, quando il computer arriverà alla riga 170, sul video compariranno i seguenti numeri:
57 223 49
infatti con l'istruzione SCRIVI A; B; 49 noi diciamo in pratica al computer di visualizzare il valore di A (cioè 57), il valore di B (cioè 223) e il numero 49.

Da notare che sul video gli interspazi fra questi tre numeri vengono determinati automaticamente dal computer.

Se dopo SCRIVI nell'istruzione non compare nessuna variabile o numero, il computer salta una riga più avanti sul video, cioè lascia una riga bianca.

SCRIVI "xx"	S. "xx"	PRINT "xx"	P. "xx"

Questa istruzione è simile alla precedente con la sola differenza che invece di un numero ci permette di far apparire sul video l'insieme dei caratteri alfanumerici da noi inseriti fra virgolette nell'istruzione stessa.

Esempio:
100 SCRIVI ''ENTRATE USCITE GIACENZA''
110 FINE

Facendo eseguire queste due istruzioni sul video comparirà la scritta riportata tra virgolette, cioè
ENTRATE USCITE GIACENZA

Da notare che tra virgolette si può inserire un massimo di 64 caratteri.

SCRIVI TAB(n)"xx"	S.T.(n)"xx"	PRINT TAB(n)"xx"	P.T.(n)"xx"

È un'istruzione simile alla precedente e si differenzia da questa per il solo fatto che la scritta tra virgolette non viene visualizzata sullo schermo a partire dall'estremo lembo a sinistra, bensì ad una distanza da questo determinata dal numero «n».

Esempio:
100 SCRIVI TAB 15 "FATTURA N. 46 DE RICA"
105 SCRIVI TAB 15 "FATTURA N. 47 SOLEMAR"
110 FINE

Facendo eseguire queste istruzioni, sul video ci appariranno le due scritte da noi poste tra virgolette una sotto l'altra distanziate di 15 spazi dall'estremità sinistra del quadro.

SCRIVI AL n,"xx"	S.AL n,"xx"	PRINT AT n,"xx"	P.AT n,"xx"

Questa istruzione è ancora simile alle precedenti ma con la possibilità di indirizzare il cursore in una qualsiasi zona del video indicata da «n», dove n rappresenta il numero del rettangolino da cui deve iniziare la scritta medesima tenendo presente che nella prima riga del video abbiamo tutti i rettangolini da 0 a 31, nella seconda tutti quelli da 32 a 63, nella terza tutti quelli da 64 a 95 e così di seguito.

Esempio:
120 SCRIVI AL 64, "CODICE"

Con questa istruzione il computer scriverà la parola CODICE all'inizio della terza riga del video.

SPSC	SPSC	CLS	C.

Quando il computer incontra questa istruzione nel corso di un programma, per esempio:
10 SPSC
ripulisce tutto lo schermo del monitor dai caratteri e segni grafici visualizzati in precedenza e riporta il cursore in alto a sinistra.

CAS (n)	C. (n)	RND (n)	R. (n)

Ogni volta che il computer incontra questa istruzione nel corso di un programma genera automaticamente un numero casuale di valore compreso fra 0 e il numero o la variabile indicata tra parentesi.

Esempio:
100 A = CAS (10).

Quando il computer arriva ad eseguire la riga 100 genera internamente un numero casuale di valore compreso tra 0 e 10 ed assegna questo valore alla variabile A, pertanto ammesso che il numero generato sia per esempio 3,58, ad esecuzione avvenuta il valore della variabile A sarà 3,58.

ASS (n)	A. (n)	ABS (n)	A. (n)

Ogni volta che il computer trova questa istruzione nel corso di un programma considera solo il valore assoluto della variabile "n", vale a dire che se questa per caso fosse un numero negativo, lo considera come positivo.

Esempio:
100 C = — 35,78

110 B = ASS (C).

Quando il computer arriva ad eseguire l'istruzione 110 di questo ipotetico programma considera solo il valore assoluto della variabile C. cioè 35.78 senza il segno «meno» davanti ed assegna questo valore alla variabile B.

INT (n)	I. (n)	INT (n)	I. (n)

Ogni volta che il computer incontra questa istruzione considera solo la parte intera della variabile n, cioè se il valore di questa variabile fosse per esempio 72,331 il computer considera solo il 72, mentre se il valore della variabile è un numero negativo, per esempio — 72,331. il computer arrotonda in eccesso considerando questo come se fosse un —73.

Esempio:

```
100 C = — 35,78
110 D = 42,001
120 A = INT (C)
130 B = INT (D)
140 FINE
```

Facendo eseguire queste istruzioni al microcomputer, arrivati alla riga 140 avremo A = — 36 B = 42.

PUNTO (X, Y)	P. (X, Y)	POINT (X, Y)	P. (X, Y)

E un'istruzione che serve quando si fanno i grafici per verificare se il punto individuato dalle coordinate X, Y è acceso o spento sullo schermo.

Se il punto è acceso il risultato del test sarà 1; se il punto è spento sarà invece 0.

Esempio:

```
100 X = 32
110 Y = 25
120 A = PUNTO (X, Y)
130 FINE
```

Facendo eseguire questo programma se il quadretto 32 della riga 25 è acceso, il valore della variabile A sarà 1; se invece tale quadretto è spento, il valore della variabile A sarà 0.

SLISTA	S.	LLIST	LL.

È questo un comando che può essere eseguito solo immediatamente da tastiera. Scrivendo sulla tastiera alfanumerica SLISTA e pigiando il tasto RETURN, automaticamente il computer esegue il **listing** del programma che ha in memoria non sul video bensì sulla **stampante**, partendo dalla riga di programma più bassa.

SSCRIVI	SS.	LPRINT	LP.

Quando il computer incontra questa istruzione nel corso di un programma automaticamente fa stampare su carta il valore della variabile o del numero che segue l'istruzione stessa.

Esempio:

```
100 A = 125,33
110 SSCRIVI A; 58
120 FINE
```

Facendo eseguire questo programma quando il computer arriverà alla riga 110 farà scrivere alla stampante il valore di A, cioè 125,33 e di fianco ad esso il numero 58.

SSCRIVI TAB	SS. T.	LPRINT TAB	LP. T.

È un'istruzione simile alla SCRIVI TAB (scrivi sul video) che abbiamo visto in precedenza a proposito del video con la sola differenza che questa volta aggiungendo una S in più all'inizio, cioè SSCRIVI TAB, l'operazione viene eseguita sulla stampante.

UT	UT	USR	U.

Questa istruzione è simile all'istruzione CHIAMA vista in precedenza con la sola differenza che la subroutine questa volta è una subroutine scritta in linguaggio macchina, cioè in esadecimale.

Esempio:

```
100 UT 2564
110 STOP.
```

Arrivato alla riga 100 il computer passa ad eseguire la subroutine scritta in esadecimale a partire dalla riga di memoria 2564; al termine della subroutine il computer ritorna al programma principale eseguendo la successiva istruzione (cioè la 110) ed essendo questa uno STOP si ferma.

PER FINO PASSO	P. F. P.	FOR TO STEP	F. TO S.

E questa la classica istruzione impiegata per realizzare un loop, cioè la ripetizione automatica di un certo gruppo di istruzioni, e per poter essere eseguita deve sempre essere abbinata ad un'istruzione INCREMENTA, che vedremo in seguito.

Per spiegare come si utilizza questa istruzione facciamo un esempio pratico.

Supponiamo di voler calcolare la somma dei numeri dispari compresi fra 1 e 5.

Il programma che possiamo utilizzare è il seguente:

```
10 A = 0
20 NOTA LA VARIABILE A RAPPRESENTA
   LA SOMMA PARZIALE
30 PER X = 1 FINO 5 PASSO 2
40 A = A + X
50 INCREMENTA X
60 SCRIVI A
70 STOP.
```

Alla riga 10 di questo programma si azzera la variabile A che come spiega la nota riportata alla riga 20 ci servirà per conservare in memoria il totale

parziale del nostro calcolo.

Alla riga 30 si assegna alla variabile X il valore 1 poi alla successiva riga 40 si somma questo numero al valore precedente di A ottenendo così un nuovo «parziale» uguale a 1.

Alla riga 50 noi troviamo un INCREMENTA X che significa testualmente: «Aggiungi ad X il valore del passo, cioè 2; controlla se il nuovo valore di X è maggiore del numero riportato alla riga 30 dopo la parola FINO, cioè se è maggiore di 5; in caso negativo ritorna ad eseguire la riga di programma posta subito dopo l'istruzione PER, cioè la riga 40, quindi somma al totale parziale il nuovo valore di X; in caso contrario esegui invece l'istruzione posta alla riga 60, cioè visualizza sul monitor il valore di A poi fermati».

In pratica il computer addiziona al valore di X un 2 ottenendo così 1 + 2 = 3; poiché 3 è minore di 5, cioè del numero riportato dopo FINO nell'istruzione di loop, il computer addiziona questo 3 al valore precedente di A (cioè 1) ottenendo così come totale parziale 1 + 3 = 4; a questo punto ritorna ad aggiungere 2 al valore di X ottenendo così 3 + 2 = 5 e poiché 5 non è maggiore di 5 addiziona ancora il valore di X al valore di A ottenendo così una nuova somma parziale pari a: 4 + 5 = 9.

A questo punto il computer addiziona ancora 2 al valore di X, ottenendo 5 + 2 = 7 e poiché 7 è maggiore di 5, passa ad eseguire l'istruzione 60, cioè scrive il valore di A sul video, quindi essendo il valore attuale di A uguale a 9 è ovvio che sul video compare un 9 infatti questa è appunto la somma dei numeri dispari compresi fra 1 e 5.

Precisiamo che nel caso in cui il «passo» sia uguale a 1 può anche essere omesso nell'istruzione PER infatti se noi, invece della somma dei numeri dispari, avessimo voluto ottenere la somma di tutti i numeri compresi fra 1 e 5, alla riga 30 avremmo potuto scrivere tranquillamente:
30 PER X = 1 FINO 5.

INCREMENTA	I.	NEXT	N.

Questa istruzione, come abbiamo visto, fa parte integrante dell'istruzione PER ... FINO ... e deve essere sempre seguita dalla stessa variabile impiegata appunto in tale istruzione.

Per esempio se noi alla riga 10 abbiamo un'istruzione:
10 PER A = 5 FINO 33
al termine del gruppo di istruzioni che fanno parte di questo loop, per esempio alla riga 150 dovremo porre un:
150 INCREMENTA A
Nell'esempio precedente invece, avendo utilizzato come variabile nel loop la X, avevamo posto alla riga 50 un INCREMENTA X.

SE	SE	IF	IF

Un'istruzione di questo tipo obbliga il computer ad effettuare un confronto fra due variabili: se l'esito di questo confronto è positivo, il computer passa ad eseguire quanto indicato a fianco dell'istruzione SE.

Se l'esito è negativo il programma prosegue invece regolarmente con la successiva istruzione.

Esempio:
50 SE A = 15 VA A 150
60 STOP

Facendo eseguire queste istruzioni il computer confronta il valore della variabile A con il numero 15; se il valore di A è uguale a 15 passa ad eseguire la riga di programma 150; se invece il valore di A è diverso da 15 esegue la riga successiva, cioè la riga 60 ed essendo questa uno STOP si ferma.

2°Esempio:
80 SE A>B ALLORA C = 0
90 SCRIVI C
100 STOP

Facendo eseguire al microcomputer un programma in cui siano inserite queste istruzioni, quando arriva alla riga 80 il microcomputer stesso controlla se il valore della variabile A è maggiore di quello della variabile B; se è maggiore pone il valore di C uguale a 0 e scrive uno 0 sul video; se invece non è maggiore scrive sul video il valore di C che può essere un numero qualsiasi a seconda di ciò che gli è stato assegnato in precedenza.

ALLORA	A.	THEN	T.

Questa istruzione va sempre abbinata all'istruzione SE vista in precedenza e ne rappresenta in pratica la logica conclusione.

Il suo significato è più o meno questo:
«Se il confronto fra le due variabili ha dato esito positivo, allora esegui quanto testè specificato».

Dopo ALLORA possiamo trovare l'assegnazione di un valore ad una variabile, per esempio C = 0 oppure un semplice numero di riga in quanto occorre tener presente che ALLORA può essere usato in sostituzione del VA A per rinviare l'esecuzione ad una riga di programma diversa da quella immediatamente successiva.

Esempio:
50 SE A<150 ALLORA 100
60 STOP

In questo caso, quando arriva alla riga 50, il microcomputer guarda se il valore di A è minore di 150 ed in caso lo sia, salta alla riga di programma 100.

Se invece il valore di A è maggiore o uguale a 150 il computer esegue la riga 60 ed essendo questa uno STOP si ferma.

VA A n	V. n	GOTO n	G. n

È un'istruzione di salto incondizionato che può essere utilizzata indifferentemente da sola oppure

abbinata all'istruzione SE vista in precedenza.

Quando il computer nel corso di un programma trova un VA A, salta immediatamente e senza porsi alcun problema alla riga specificata dopo il VA A stesso.

Esempio:

```
50 VA A 70
60 STOP
```

Quando arriva alla riga 50 il computer salta direttamente alla riga 70 ignorando lo STOP posto nel mezzo, cioè alla riga 60.

Importante: le due parola VA A debbono sempre essere separate da uno spazio vuoto.

SECONDO	SEC.	ON	ON

Quando incontra questa istruzione il programma può saltare a differenti righe di memoria a seconda del valore che assume una determinata variabile.

Esempio:

```
50 SECONDO B VA A 500, 750, 900
```

Facendo eseguire questa istruzione il computer controlla il valore della variabile B; se il valore di B è uguale a 1 passa ad eseguire l'istruzione 500; se il valore di B è uguale a 2 passa ad eseguire l'istruzione 750; se il valore B è uguale a 3 passa ad eseguire l'istruzione 900.

Qualsiasi altro valore di B non potrebbe in questo caso essere accettato e darebbe luogo ad un errore segnalato sul video con la parola COME?

DATI	D.	DATA	D.

È un'istruzione che serve per assegnare dei valori numerici o alfabetici a delle variabili e deve sempre essere abbinata a un'istruzione LEGGI o RILEGGI.

Vedremo appunto sotto queste istruzioni come si impiega generalmente l'istruzione DATI.

LEGGI	L.	READ	REA.

Per spiegare come funziona questa istruzione, che deve sempre essere abbinata ad un'istruzione DATI, faremo subito un esempio pratico.

```
10 DATI 5, 80, 99, 57, 68
20 LEGGI X, Y
30 LEGGI A
40 LEGGI B, C
50 STOP
```

Facendo eseguire questo programma, quando arriva alla riga 20, il computer va a leggere i primi due dati da noi indicati, cioè 5 e 80 e li assegna rispettivamente come valore alla variabile X e alla variabile Y per cui avremo X = 5 e Y = 80.

Passando alla riga 30 il computer va a leggere il terzo dei nostri dati e lo assegna come valore alla variabile A, per cui avremo A = 99.

Passando infine alla riga 40 il computer va a leggersi gli ultimi due dati, cioè 57 e 68 e li assegna rispettivamente come valore alla variabile B e alla variabile C, quindi avremo B = 57 E C = 68.

RILEGGI	RIL.	RESTORE	REST.

Anche per spiegare questa istruzione, che va essa pure abbinata ad un'istruzione DATI, ci serviremo di un esempio pratico.

```
10 DATI 47, 89, 65, 102
20 LEGGI X, Y
30 RILEGGI A, B
40 STOP
```

Facendo eseguire questo programma, quando arriva alla riga 20 il computer legge i primi due dati, cioè 47 e 89 ed assegna questi valori rispettivamente alla variabile X e alla variabile Y, quindi avremo X = 47 e Y = 89.

Passando alla riga 30 il computer trova un RILEGGI quindi anziché leggere il terzo e il quarto dato come avveniva in precedenza, legge ancora il primo e il secondo ed assegna questi valori rispettivamente alla variabile A e B, quindi avremo A = 47 e B = 89.

DUP	D.	DUP	D.

Questo comando, che può essere eseguito solo da tastiera, serve come spiegheremo sul prossimo numero per **duplicare** la cassetta del BASIC.

FINE	F.	END	E.

Quando incontra questa istruzione nel corso di un programma il computer capisce che il programma stesso è finito quindi si ferma totalmente cioè non è più possibile farlo ripartire con un CONTINUA come avveniva per lo STOP.

MEM	M.	MEM	M.

Questo comando che può essere eseguito solo da tastiera, purché sia preceduto da uno SCRIVI, ci permette di far comparire sul video il numero corrispondente ai **byte di memoria ancora disponibili per l'utente** in modo tale da potersi regolare circa la lunghezza del programma.

In pratica noi dovremo scrivere sulla tastiera alfanumerica SCRIVI MEM poi pigiare il tasto RETURN ed automaticamente sul video ci apparirà un numero; se appare per esempio il numero 1325 significa che abbiamo ancora a disposizione per il nostro programma 1325 byte, cioè 1325 righe di memoria.

RC	RC	RC	RC

Scrivendo sulla tastiera alfanumerica RC e pigiando il tasto RETURN noi escludiamo automaticamente il controllo del computer sul comando REMOTE del registratore e questo ci servirà ad esempio per poter riavvolgere il tasto dopo aver caricato in memoria il BASIC.

Una volta effettuata questa operazione, per poter iniziare a programmare, sarà sufficiente battere sulla tastiera la **barra** di interlinea.

LE VARIABILI

Il nostro BASIC, nella versione attuale, permette l'utilizzazione di:

26 variabili numeriche
5 variabili alfanumeriche
1 vettore ad una dimensione

Le variabili **numeriche** possono essere siglate con le lettere dell'alfabeto dalla **A alla Z** e servono nel corso di un programma per svolgere una funzione analoga a quella che abbiamo visto svolgere dai registri della CPU quando programmavamo in esadecimale.

In pratica una variabile «numerica» è una specie di registro in cui noi depositiamo un certo numero o il risultato di una determinata operazione per poterlo eventualmente prelevare e modificare in seguito.

Un esempio di utilizzazione pratica di una variabile lo abbiamo già visto in precedenza quando abbiamo esaminato il programma per calcolare la somma dei numeri dispari compresi fra 1 e 5 tuttavia non riteniamo certo chiuso l'argomento e ci ripromettiamo senz'altro di riprenderlo in modo più approfondito sui prossimi numeri.

Per ora ci limitiamo soltanto a dire che nella scelta della sigla da assegnare ad una variabile è sempre conveniente seguire un metodo che sia il più mnemonico possibile in modo da potersi facilmente ricordare il significato della variabile stessa.

Per esempio se dovessimo realizzare una gestione di magazzino potremmo utilizzare la variabile G per la Giacenza, la variabile E per le Entrate, la variabile U per le Uscite e così di seguito.

Le variabili **alfanumeriche** o «string» come i programmatori più esperti sono abituati a chiamarle, possono essere contraddistinte con le lettere dell'alfabeto dalla A alla E seguite dal simbolo di «dollaro», cioè:

A$ - B$ - C$ - D$ - E$

Ognuna di queste «string» può contenere un massimo di 16 caratteri alfabetici o numerici, compresi gli spazi vuoti, e può essere utilizzata, riferendoci sempre ad un programma per una gestione di magazzino, per conservare in memoria ad esempio la descrizione di un determinato componente oppure il nome del fornitore da cui acquistiamo il componente stesso.

Per quanto riguarda i «vettori», come già anticipato, ne abbiamo uno solo a disposizione, cioè il **vettore A**, il quale ha una lunghezza praticamente illimitata, dipendente solo dall'area di memoria a disposizione per i programmi.

In pratica un vettore è un insieme di un certo numero di variabili con un indice fra parentesi, cioè A(1), A(2), A(3), A(4) ecc. ad ognuna delle quali può essere assegnato un determinato valore numerico.

Questo vettore potrebbe essere utilizzato ad esempio in un programma di contabilità per tenere memorizzati gli incassi giornalieri di un determinato mese; in tal caso avremo:

A(1) = incasso giorno 1
A(2) = incasso giorno 2
A(3) = incasso giorno 3
A(4) = incasso giorno 4
A(31) = incasso giorno 31.

OPERATORI MATEMATICI e RELAZIONALI

Gli operatori matematici e relazionali disponibili nel nostro BASIC per effettuare appunto operazioni matematiche o confronti fra due variabili sono rispettivamente:

= uguale a
+ più (addizione)
— meno (sottrazione)
* per (moltiplicazione)
<> diverso da
> maggiore di
< minore di
>= maggiore o uguale a
<= minore o uguale.

Per spiegare come si usano questi operatori vi faremo un esempio pratico che li comprende tutti.

```
10 A=5
20 B=A+3
30 C=B—6
40 D=C*9
50 E=D / 3
60 SE A <> B VA A 90
70 SE D > C VA A 100
80 SE E < B VA A 70
90 SE E >= C VA A 80
100 SE B <= C VA A 10
110 SCRIVI A; B; C; D; E
120 FINE.
```

Alla riga 10 il computer assegna alla variabile A il valore 5.

Alla riga 20 addiziona al valore di A, cioè 5, il numero 3 ottenendo così 8 ed assegna quindi questo 8 alla variabile B.

Alla riga 30 sottrae al valore di B, cioè 8, il numero 6 ottenendo così 2 ed assegna quindi questo 2 alla variabile C.

Alla riga 40 moltiplica il valore di C, cioè 2, per il numero 9 ottenendo così 18 ed assegna quindi il 18 alla variabile D.

Alla riga 50 divide il valore di D, cioè 18, per il

numero 3 ottenendo così 6 ed assegna quindi questo valore alla variabile E.

Alla riga 60 iniziano i confronti fra le variabili e precisamente il computer si chiede se il valore di A è diverso dal valore di B: essendo la risposta affermativa (infatti A è diverso da B) salta alla riga 90.

Alla riga 90 il computer si chiede se il valore di E, cioè 6, è maggiore o uguale a quello di C, cioè 2 ed essendo anche in questo caso la risposta affermativa, salta alla riga 80.

Alla riga 80 il computer si chiede se il valore di E, cioè 6, è minore di quello di B, cioè 8, ed essendo ancora la risposta affermativa salta alla riga 70.

Alla riga 70 il computer si chiede se il valore di D, cioè 18 è maggiore del valore di C, cioè 2, ed essendo la risposta ancora un «si» salta alla riga 100.

Alla riga 100 il computer si chiede se il valore di B, cioè 8, è minore o uguale a quello di C, cioè 2, ed essendo in questo caso la risposta negativa, anziché saltare alla riga indicata di fianco, cioè alla riga 10, prosegue con la riga che viene subito dopo, cioè con la riga 110.

Alla riga 110 il computer scrive sul video i valori delle variabili A, B, C, D, E cioè rispettivamente i numeri 5, 8, 2, 18, 6 poi passa alla riga 120 e si ferma.

PUNTEGGIATURA

Per concludere la nostra descrizione vediamo ora quale significato hanno nell'ambito del BASIC alcuni simboli grafici come per esempio il «punto e virgola», i «due punti», le «virgolette» e la «virgola», simboli questi che possono essere tutti utilizzati nelle istruzioni.

Punto e virgola (;)

Il punto e virgola si utilizza per poter visualizzare sul monitor o stampare due o più numeri sulla stessa riga; dopo un ordine di «stampa» infatti il cursore della stampante o del video tenderebbe automaticamente a ritornare a capo; trovando un punto e virgola invece, il cursore non ritorna a capo e quando arriva il successivo ordine di stampa il relativo numero viene scritto di fianco al primo.

Esempio:

```
10 A=58          10 A=58
20 B=32          20 B=32
30 SCRIVI A;     40 SCRIVI A
40 SCRIVI B      40 SCRIVI B
50 FINE          50 FINE
```

Nel primo caso (programma di sinistra) i numeri 53 e 32 appariranno sul video tutti sulla stessa riga, infatti essendoci un «punto e virgola» dopo A il cursore del video non ritorna a capo; nel secondo caso invece i numeri 58 e 32 ci verranno visualizzati uno sotto l'altro in quanto dopo aver scritto il valore di A il cursore del video, non trovando il punto e virgola, ritorna a capo.

Due punti (:)

I due punti ci permettono di inserire più di una istruzione sulla stessa riga di programma.

Esempio:

```
10 A=27:B=39:C=41
20 SCRIVI A; B; C
30 FINE
```

Come noterete sulla riga 10 noi abbiamo 3 istruzioni e precisamente la prima assegna il valore 27 alla variabile A; la seconda assegna il valore 39 alla variabile B e la terza assegna il valore 41 alla variabile C.

Alla riga 20 si visualizzano questi tre valori uno di fianco all'altro sul video in quanto abbiamo messo un punto e virgola dopo A e B ed alla riga 30 il programma si ferma.

Virgolette (")

Le virgolette servono sempre per racchiudere un testo alfanumerico che deve essere visualizzato sul video, stampato o assegnato ad una delle 5 variabili alfanumeriche.

Esempio:

```
10 SCRIVI AL 5, "GIACENZA"
20 FINE
```

Facendo eseguire questo programma al computer sulla prima riga del video, a partire dal 5° rettangolino, verrà visualizzata la parola GIACENZA, cioè il testo compreso fra virgolette.

2° Esempio:

```
10 A$="PRELIEVO"
20 SCRIVI A$
30 FINE
```

In pratica con questo programma si assegna alla variabile alfanumerica A$ il testo PRELIEVO quindi si dice al computer di visualizzare sul monitor tale variabile e di conseguenza sul monitor stesso compare la parola PRELIEVO.

Virgola (,)

La virgola viene utilizzata nelle istruzioni BASIC per separare fra di loro un blocco di dati, per esempio:

```
10 DATI 3, 67
20 LEGGI X, Y
30 SCRIVI X
40 SCRIVI Y
50 FINE.
```

Quando arriva alla riga 20 il computer legge i due dati riportati di fianco all'istruzione DATI ed assegna rispettivamente il valore 3 alla variabile X e il valore 67 alla variabile Y, poi visualizza questi due numeri uno sotto l'altro sul monitor.

Punto (.)

Fa le veci della nostra virgola nei numeri che non siano interi.

Per esempio il numero 30,19 sul computer deve essere scritto: 30.19

COME SI SCRIVE UN PROGRAMMA IN BASIC

Poniamoci ora un problema molto semplice e vediamo come lo si può risolvere in BASIC.

Supponiamo di voler calcolare la somma dei primi N numeri interi, dove N è un numero generico che noi forniamo di volta in volta al computer, e di voler quindi visualizzare il risultato sul monitor.

Il programma che utilizzeremo per raggiungere lo scopo è il seguente:

```
10 SPSC
20 X=0
30 RICHIEDI "NUMERO"; N
40 PER Y=1 FINO N
50 X=X+Y
60 INCREMENTA Y
70 SCRIVI "TOTALE="; X
80 VA A 20
90 FINE.
```

In pratica in questo programma noi utilizziamo la variabile X come totale parziale nei calcoli, la variabile N numero massimo da addizionare e la variabile Y come variabile di servizio.

Analizzando il programma riga per riga troviamo alla riga 10 un SPSC che serve per **cancellare** totalmente lo schermo del monitor; alla riga 20 noi azzeriamo la variabile X che rappresenta il totale parziale del nostro calcolo; alla riga 30 il computer ci chiede di specificargli il numero N facendo comparire sul video la parola NUMERO seguita da un punto interrogativo; a questo punto noi dovremo scrivere il numero che vogliamo sulla tastiera alfanumerica, per esempio 30, poi pigiare RETURN.

Ovviamente il numero 30 verrà assegnato come valore alla variabile N dopodiché inizierà il «loop» di calcolo vero e proprio compreso fra le righe 40 e 60.

Il computer pone inizialmente il valore di Y uguale a 1, addiziona questo valore a X poi aumenta di 1 il valore di Y e lo torna a addizionare ad X procedendo in questo modo finché il valore di Y, che ogni volta aumenta di 1, non supera 30, cioè il valore da noi assegnato ad N.

Arrivati a questo punto il computer passa ad eseguire l'istruzione 70, cioè fa apparire sul video la scritta TOTALE = e di fianco ad essa visualizza appunto il totale, poi passa ad eseguire la successiva istruzione (posta alla riga 80) che lo rimanda alla riga 20, cioè lo predispone per un nuovo ciclo di calcolo.

Come vedete il programma è abbastanza facile da comprendere tuttavia quello che a noi preme in questo momento non è tanto insegnarvi a programmare, quanto piuttosto insegnarvi ad inserire in memoria le varie istruzioni che costituiscono un programma ed a far quindi eseguire il programma stesso dal computer.

Per effettuare tale operazione occorre procedere come segue:

1) Innanzitutto per poter scrivere un programma in BASIC occorre che sia già stato caricato nel computer l'interprete BASIC, come spiegato nell'apposito paragrafo riportato all'inizio di questo articolo.

2) Una volta terminato di caricare il BASIC, cioè una volta che il registratore si è fermato e sul video è comparsa la scritta BASIC in alto a sinistra, scrivete sulla tastiera alfanumerica **RC** quindi pigiate il tasto **RETURN**.

Automaticamente vedrete il cursore del video (cioè il quadratino luminoso) andare a capo e contemporaneamente il computer perderà il controllo del registratore lasciandovi così liberi di riavvolgere il nastro anche con la spina del REMOTE inserita.

3) Dopo aver riavvolto completamente il nastro del BASIC battete uno spazio sulla tastiera (cioè pigiate la barra di interlinea) oppure pigiate il tasto RETURN per riprendere il controllo del registratore.

4) A questo punto potete iniziare a scrivere sulla tastiera le vostre istruzioni, ricordando che all'inizio di ognuna di esse deve sempre essere riportato il relativo numero di riga, cioè 10-20-30-40 ecc. e che alla fine di ogni riga, per poterla caricare in memoria, occorre sempre pigiare il tasto RETURN.

5) Scrivete sulla tastiera **10 SPSC** poi pigiate il tasto **RETURN**.

6) Automaticamente il cursore si porterà a capo della riga successiva. Scrivete in questa riga
20 X=0
e pigiate ancora **RETURN**.

7) Se per caso vi sbagliaste a scrivere, cioè scriveste per esempio 20 X=1 invece che 20 X=0, prima di pigiare il tasto RETURN potrete correggere questa istruzione pigiando il tasto **DEL** il quale ci permette in pratica di cancellare l'ultimo carattere da noi impostato sulla tastiera.

Ovviamente una volta cancellato il numero 1 non dovremo fare altro che riscrivere al suo posto uno 0 e pigiare al solito il tasto RETURN per passare alla riga successiva.

8) In questa riga dovremo scrivere:
30 RICHIEDI "NUMERO"; N
e tutto sarebbe più che normale se non fosse per le «virgolette» le quali però si possono ottenere molto facilmente tenendo pigiato con l'indice della mano destra il tasto SHIFT e contemporaneamente pigiando il tasto 2 in alto con l'indice della mano sinistra.

In pratica il tasto SHIFT svolge la stessa funzione del tasto per le lettere maiuscole in una normalissima macchina da scrivere.

E ovvio che se per ottenere le virgolette pigiassimo, invece del tasto SHIFT, il tasto SHIFT LOCK, dovremmo in seguito ricordarci di «sbloccare» questo tasto pigiandolo una seconda volta diversamente pigiando ad esempio il tasto 4 vedremmo comparire sul video il simbolo del «dollaro».

9) Alla fine della riga 30 pigiate come al solito il tasto RETURN per passare alla riga successiva e

così di seguito fino ad arrivare alla riga 90 in cui dovrete scrivere 90 FINE e pigiare ancora il tasto RETURN.

Giunti a questo punto, considerata la semplicità del programma, potreste anche farlo eseguire direttamente tuttavia essendo questa una prova esplicativa, supponiamo di voler fare il listing del medesimo sul video.

Per ottenere questo la procedura è molto semplice infatti, come già precisato in precedenza, basta scrivere sulla tastiera alfanumerica: **LISTA** e pigiare quindi il tasto **RETURN** per vedere automaticamente comparire sul video le prime 8 righe di programma, cioè dalla riga 10 alla riga 80.

A questo punto, se volete far comparire anche la riga 90, pigiate il tasto **ESC** ed automaticamente la in basso vedrete apparire la riga 90.

PER CORREGGERE UN'ISTRUZIONE

Se controllando il listing di un programma vi accorgeste di aver commesso un errore, per esempio se alla riga 60, invece di INCREMENTA Y, aveste scritto INCREMENTA X, per correggere questo errore occorrerà necessariamente riscrivere l'intera riga procedendo come segue:

1) Pigiate contemporaneamente sulla tastiera alfanumerica i due tasti BREAK-BREAK (pigiandone uno solo per volta non si ottiene la funzione voiuta In quanto sono dipendenti l'uno dall'altro); automaticamente vedrete ricomparire in alto a sinistra sul video la scritta BASIC ecc.

2) Riscrivete per intero la vostra istruzione, facendola precedere dal numero di linea, cioè:
60 INCREMENTA Y

3) Pigiate il tasto RETURN.

Così facendo, se voi ora eseguite di nuovo il listing del programma scrivendo LISTA sulla tastiera e pigiando il tasto RETURN vedrete che la vostra istruzione si è modificata come richiesto.

In pratica scrivendo una nuova istruzione con un numero di riga già esistente la vecchia istruzione automaticamente si cancella e viene sostituita da quest'ultima.

PER INSERIRE UNA NUOVA LINEA DI PROGRAMMA

Sempre riferendoci al programma precedente supponiamo di voler aggiungere, fra la riga 60 e la riga 70, un'istruzione che ci permetta di far comparire sul video il numero N, per esempio:
65 SCRIVI "N="; N.

Tale operazione è possibile per il semplice motivo che noi, scrivendo il programma iniziale, abbiamo volutamente saltato di 10 in 10 nel numerare le righe proprio in previsione che ci fosse stata qualche istruzione da aggiungere in seguito cioè abbiamo scritto sulle righe 10-20-30-40 ecc. saltando la 11-12-13 ecc.

Se il programma originario fosse stato numerato con 1-2-3-4 ecc. noi ora non potremmo inserire questa nuova istruzione nel mezzo delle altre, bensì dovremmo riscrivere totalmente parte del programma stesso.

Questo dovrebbe farvi comprendere che è sempre conveniente, scrivendo un qualsiasi programma, numerare di dieci in dieci le istruzioni poiché così facendo noi potremo sempre inserire altre righe nel mezzo, per esempio la riga 61-62-63-64 ecc.

Inserendo una sola nuova istruzione è consigliabile assegnarle un numero intermedio, per esempio dovendo inserirla come nel nostro caso fra il 60 e il 70 è consigliabile assegnarle il numero 65 in modo da lasciare spazio per altre istruzioni ancora.

Per inserire la nuova istruzione alla riga 65 dovrete procedere come segue:

1) Pigiate sulla tastiera i due tasti BREAK-BREAK (insieme).

2) Scrivete sulla tastiera stessa:
65 SCRIVI "N="; N.

3) Pigiate il tasto RETURN per memorizzare tale istruzione.

A questo punto eseguendo un nuovo listing come in precedenza indicato, vi accorgerete che l'istruzione 65 è stata inserita come richiesto fra la 60 e la 70.

PER FAR ESEGUIRE UN PROGRAMMA

Una volta verificato che tutte le istruzioni sono state scritte in modo corretto potremo finalmente far eseguire il nostro programma al computer procedendo come segue:

1) Pigiate i due tasti BREAK-BREAK sulla tastiera alfanumerica avendo cura di pigiarli contemporaneamente.

2) In alto a sinistra sul video vi apparirà la solita scritta BASIC ecc.

3) Scrivete sulla tastiera ESEGUI quindi pigiate il tasto RETURN.

Automaticamente il computer inizierà la sue esecuzione.

Se il programma è quello visto in precedenza per calcolare la somma dei primi N numeri interi il monitor si spegnerà e sulla sinistra in alto comparirà la domanda: NUMERO?

Scrivete il numero che vi interessa sulla tastiera alfanumerica, per esempio il numero 5, quindi pigiate il tasto RETURN.

Il computer vi fornirà subito il risultato facendo comparire sul video la scritta:
N=5
e sotto a questa la scritta:
TOTALE=15
infatti la somma dei primi 5 numeri interi da per totale 15.

Sotto al totale comparirà poi ancora la domanda

NUMERO? ed a questo punto, se volete far eseguire un nuovo totale, scrivete il numero sulla tastiera e pigiate il tasto RETURN.

Se invece volete fermare il vostro programma pigiate il tasto situato nel vertice in alto sulla destra della tastiera e contraddistinto da un = posto verticalmente.

Il programma in questo modo automaticamente si fermerà, tuttavia per poter riprendere il controllo del computer dalla tastiera occorrerà ancora pigiare i due tasti BREAK-BREAK.

Nota: i due tasti BREAK-BREAK svolgono in pratica su questa tastiera una funzione simile a quella del RESET sulla tastiera esadecimale infatti ogni volta che il computer si blocca per un qualsiasi motivo, per poter ritornare sotto il controllo del BASIC occorre pigiare BREAK-BREAK.

PER REGISTRARE UN PROGRAMMA SU NASTRO

Giunti a questo punto, se noi volessimo registrare questo programma su nastro per poterlo utilizzare di nuovo dopo qualche giorno (infatti occorre sempre ricordare che spegnendo il computer i programmi memorizzati su RAM automaticamente si cancellano) dovremmo procedere come segue:

1) Inserite il nastro completamente riavvolto sul registratore.

2) Pigiate sulla tastiera alfanumerica i due tasti BREAK-BREAK per ritornare al controllo del BASIC poi pigiate sul registratore i tasti PLAY-REC.

3) Quando vedrete comparire in alto a sinistra dello schermo la scritta BASIC ecc. scrivete sulla tastiera alfanumerica REGISTRA quindi pigiate il tasto RETURN e il registratore inizierà a girare.

A registrazione avvenuta il registratore si deve fermare automaticamente e sul video deve comparire di nuovo la scritta BASIC ecc. sulla sinistra in alto.

PER LEGGERE UN PROGRAMMA DAL NASTRO

Supponiamo ora che a distanza di qualche giorno si voglia rileggere questo programma per farlo eseguire di nuovo.

Per far questo è assolutamente necessario che nel computer sia già stato caricato il BASIC dopodiché potremo procedere come segue:

1) Posizionate il nastro all'inizio corsa e pigiate sul registratore il tasto PLAY.

2) Pigiate sulla tastiera alfanumerica i due tasti BREAK-BREAK.

3) Scrivete sempre sulla tastiera alfanumerica la parola **CARICA** poi pigiate il tasto RETURN.

4) Il registratore si metterà in funzione e dopo alcuni secondi sulla destra del video, in negativo, appariranno due **LL** di cui quella più a destra inizierà dopo qualche istante a lampeggiare.

5) Se durante la fase di lettura sul video compare la scritta: COSA? preceduta da un numero compreso tra 1 e 7 significa che si è verificato un errore di lettura, pertanto occorre riprendere la procedura daccapo.

6) Se invece tutto fila liscio, alla fine il registratore si ferma da solo e sul video torna a comparire la scritta BASIC ecc. per indicarci che il computer stesso è pronto a ricevere ordini in BASIC.