IBM

Personal Computer

MANUALE BASIC

Informazioni generali sulla programmazione

6834178

IBM United Kingdom International Products Limited
PO Box 41, North Harbour, Portsmouth PO6 3AU, England

Printed in Great Britain by Ben Johnson & Co. Ltd., York.

In Italia valgono solo le seguenti disposizioni in lingua italiana. Eventuali altre condizioni o contratti di licenza di programmi IBM, stampati all'interno del prodotto non sono applicabili e devono pertanto essere ignorate.

Il cliente è responsabile della scelta di questo prodotto software al fine del raggiungimento dei risultati voluti, nonché dell'installazione, dell'uso dello stesso e dei relativi risultati.

Condizioni di uso

Questo prodotto software contiene materiale oggetto di diritti esclusivi della IBM Italia-Distribuzione Prodotti s.r.l. (qui di seguito denominata IBM) o dei suoi danti causa. In ogni caso devono essere tassativamente osservate le seguenti condizioni:

1. L'uso del programma è consentito su una sola macchina per volta.

2. Il programma può essere copiato in forma leggibile dal sistema elaborazione o stampato esclusivamente come supporto a tale uso. Inoltre, i programmi contrassegnati "Copy Protected" oppure "Copia Protetta" possono contenere dispositivi per limitarne o impedirne la copiatura.

3. Il programma può essere modificato o collegato ad altri programmi esclusivamente per uso su detta macchina; ogni copia integrale, parziale o modificata sarà soggetta alle presenti condizioni.

4. L'indicazione di copyright deve essere riprodotta ed inclusa in ogni copia integrale, parziale o modificata del programma.

5. Il cliente che ceda il prodotto software ad altri deve immediatamente cessarne l'uso e non può trattenerne alcuna copia sia essa integrale, parziale o modificata. Il nuovo acquirente è tenuto ad osservare le presenti condizioni.

E' altrimenti vietata la riproduzione, l'uso, ed il trasferimento del prodotto software.

In caso di esportazione, il cliente dovrà ottenere le eventuali licenze di esportazione nazionali e degli Stati Uniti d'America che siano necessarie per l'esportazione o la riesportazione di questo prodotto software.

Caratteristiche del programma e avvertimenti per l'uso

Il programma non è garantito dalla IBM né dai suoi concessionari. La IBM non garantisce che le funzioni contenute nel programma soddisfino le esigenze dell'utente o funzionino in tutte le combinazioni che possono essere scelte per l'uso da parte dell'utente. L'utente inoltre dovrà controllare il programma ed ovviare a proprie spese ad eventuali errori o malfunzionamenti.

IBM *Personal Computer*

MANUALE BASIC

Informazioni generali sulla programmazione

Prima Edizione, (Maggio 1984)

La IBM Italia fornisce questo manuale senza offrire le garanzie usuali come per i prodotti IBM. La IBM si riserva il diritto di apportare miglioramenti o modifiche al prodotto od al programma descritto nel manuale in qualsiasi momento e senza preavviso.

Questa pubblicazione potrebbe contenere informazioni tecniche inconsistenti od errori tipografici.
Le correzioni relative saranno incluse nelle nuove edizioni della pubblicazione.

E' possibile che questo materiale contenga riferimenti od informazioni su prodotti (macchine o programmi), non ancora annunciati. Tali riferimenti od informazioni non possono significare in alcun modo che la IBM Italia intenda annunciare tali prodotti, programmi o servizi.

Richieste di ulteriori copie di questo prodotto od informazioni tecniche sullo stesso, vanno indirizzate al punto di vendita autorizzato.

PREFAZIONE

L'interprete BASIC del Personal Computer IBM comprende tre versioni progressive compatibili: su Cassetta, su Disco e Avanzato. Il manuale di riferimento del BASIC ed il manuale del BASIC costituiscono un riferimento per queste 3 versioni e per i diversi livelli di rilascio di programma BASIC a partire da 1.Ø.

È opportuno tener presente che i suddetti manuali costituiscono *unicamente dei riferimenti* per il linguaggio BASIC e non dovranno essere considerati testi di apprendimento per la programmazione in BASIC.

Volendo imparare a programmare in BASIC è consigliabile consultare specifici volumi.

Il manuale contiene informazioni generali sull'utilizzo del BASIC; i capitoli iniziali potranno essere usati per l'avvio all'utilizzo del BASIC, mentre quelli contenenti informazioni più complesse potranno essere rivolti a programmatori esperti.

Il manuale di riferimento del BASIC contiene, in ordine alfabetico, la sintassi e la semantica di ciascun comando, specifica e funzione in linguaggio BASIC.

I due manuali costituiscono un utile riferimento, di facile consultazione. Hanno riferimenti incrociati e sono provvisti di indici, appendici ed utili informazioni.

È disponibile, inoltre, un manualetto che riassume per categoria, i comandi, le specifiche e le singole funzioni.

Nota: Se si possiede un PCjr usare il volume BASIC specifico per questo tipo di PC.

Il compilatore BASIC del Personal Computer IBM è un programma facoltativo disponibile presso i venditori PC. Se si utilizza il Compilatore BASIC consultare unitamente a questo volume ed al manuale di riferimento, il manuale relativo al Compilatore BASIC del Personal Computer IBM.

Riferimenti

Utili riferimenti possono essere trovati nelle seguenti pubblicazioni:

- *Guida Operativa* del Personal Computer IBM.
- *Dos* del Personal Computer IBM.
- *Technical Reference* al DOS del PC IBM.
- *Technical Reference* del PC IBM.

Riepilogo delle modifiche

Il riepilogo elenca nell'ordine le modifiche apportate al BASIC livello 2.Ø e 3.Ø.

Variazioni fra BASIC livello 2.Ø e 2.1

Al BASIC livello 2.Ø sono state apportate le seguenti variazioni:

- Alla linea di comando BASIC sono state apportate tre modifiche:
  - E' stato aggiunto all'opzione /M: il parametro facoltativo *mass ampbloc* che consente di riservare dello spazio oltre il segmento di dati del BASIC per i sotto programmi in linguaggio macchina.
  - Specificando /D nella linea di comando BASIC è ora possibile il calcolo a precisione doppia con le funzioni ATN, COS, EXP, LOG, SIN, SQR e TAN.
  - Usando i parametri < *stdin* o > *stdout* nella linea di comando BASIC è possibile variare il dispositivo standard di immissione e emissione. stdin (reindirizza l'immissione standard), stdout (reindirizza l'emissione standard).
- La pressione dei tasti Ctrl-Stampa causa la stampa contemporanea di ciò che viene emesso dallo schermo.
- Nei comandi ove è presente il parametro *specfile* è stata ampliata la sintassi che ora consente di specificare un collegamento ad un dispositivo o ad un file. Tutti i comandi e specifiche che accettavano il parametro *specfile* accettano ora anche specifica di collegamento. I comandi che consentono il collegamento sono

BLOAD, BSAVE, KILL, LOAD, MERGE, NAME, RUN e SAVE. Le specifiche che consentono il collegamento sono CHAIN e OPEN.

- La sintassi del comando DELETE è stata ampliata per permettere la cancellazione di righe di un programma, da una determinata riga sino alla sua fine.

- Alla specifica OPEN "COM... è stato aggiunto il parametro PE per permettere il controllo di parità.

- La specifica PLAY ha due nuovi parametri. (Disponibili solo col BASIC avanzato).

  – > *n* aumenta l'ottava e suona la nota *n*.

  – < *n* diminuisce l'ottava e suona la nota *n*.

- La specifica DRAW ha due nuovi parametri. (Disponibili solo col BASIC avanzato).

  – TA*(n)* ruota l'angolo *n* da -360 a 360 gradi.

  – P *figura, bordo* imposta il colore della figura e del bordo.

- La funzione POINT ammette ora il formato v = POINT *(n)* che ripristina il valore corrente delle coordinate grafiche *x* o *y*. (Disponibile solo con il BASIC avanzato).

- La specifica RANDOMIZE permette ora espressioni a precisione doppia.

- La specifica LINE ha il nuovo parametro, *style*, che usa valori esadecimali per tracciare una sequenza di punti sullo schermo. (Disponibile solo con il BASIC avanzato).

- La specifica PAINT ha il nuovo parametro, *sfondo*, che permette di colorare un'area mediante un modello piuttosto che tramite un colore. (Disponibile solo con il BASIC avanzato).

- Nelle specifiche ON KEY (n), KEY (n), e KEY è possibile ora definire sei tasti in aggiunta ai precedenti, n = 15-20. (Disponibile solo con il BASIC avanzato).

- Le specifiche GET e PUT consentono, ora, l'utilizzo di numeri di record compresi tra 1 e 16.777.215 per l'uso di grandi file con ridotta lunghezza record.

- La funzione EOF (Ø) ripristina la condizione di fine file per i dispositivi standard di immissione usata con reindirizzamento di I/E.

- La funzione LOF ripristina l'effettivo numero di byte riservati a un file.

- Le specifiche grafiche CIRCLE, DRAW, LINE, PAINT, POINT, PSET, PRESET, VIEW e WINDOW ora usano il taglio delle linee anziché la sovrapposizione.

Sono state aggiunte tre nuove funzioni:

- La funzione PLAY (n) ripristina il numero di note attualmente presenti nella memoria di transito della musica di sottofondo (MB). (Disponibile solo con il BASIC avanzato).

- La funzione PMAP consente di tradurre coordinate spaziali in coordinate fisiche e viceversa. (Disponibile solo con il BASIC avanzato).

- La funzione TIMER riporta il numero di secondi trascorsi dalla mezzanotte o dall'ultimo ripristino del sistema (ripristino del sistema con i tasti Ctrl, Altrn e Canc).

Sono state aggiunte quattro nuove specifiche:

- La specifica ON PLAY consente di mantenere continuo il suono di una musica durante l'esecuzione di un programma. (Disponibile solo con il BASIC avanzato).

- La specifica ON TIMER, trascorso un periodo di tempo definito, trasferisce il controllo ad un determinato numero di riga del programma BASIC. (Disponibile solo con il BASIC avanzato).

- La specifica VIEW permette di definire una "finestra" o porzione di schermo entro i limiti fisici del video. (Disponibile solo con il BASIC avanzato).

- La specifica WINDOW consente di ridefinire le coordinate dello schermo o una porzione di esso. (Disponibile solo con il BASIC avanzato).

Sono stati aggiunti tre nuovi comandi:

- Il comando CHDIR consente la variazione dell'indirizzario corrente.

- Il comando MKDIR crea un indirizzario sul disco specificato.

- Il comando RMDIR elimina un indirizzario dal disco specificato.

Variazioni nel BASIC 3.Ø

Nel BASIC 3.Ø sono state eseguite le seguenti modifiche:

- Il supporto unità consente al BASIC di comunicare con i programmi guida dell'unità, installati dall'utente, tramite la specifica IOCTL e la funzione IOCTL$.

- IOCTL e IOCTL$ vengono utilizzate per ricevere/inviare informazioni da/a canali di unità.

- La specifica ENVIRON e la funzione ENVIRON$ consentono di modificare i parametri nella tabella BASIC.

- ERDEV e ERDEV$ sono variabili di errore dell'unità che consentono la lettura dei codici d'errore INT 24.

- SHELL consente l'esecuzione di comandi DOS e di elaborazioni "derivate" (secondarie) dal BASIC.

  Nota: Nel manuale vengono citati i termini "disco" "minidisco" e "disco fisso". "Minidisco" fa riferimento unicamente ad unità a minidisco e minidischi, "disco fisso", fa riferimento unicamente a unità disco fisso, "disco" fa riferimento sia ai dischi fissi che ai minidischi.

Indice

Capitolo 1. Le versioni del BASIC

Indice

Le versioni del BASIC

Per il Personal Computer IBM sono disponibili tre diverse versioni del BASIC:

- Su cassette
- Su disco
- Avanzato

Le tre versioni di BASIC sono compatibili verso l'alto; cioè, il BASIC su disco esegue tutte le funzioni del BASIC su cassette più altre, mentre il BASIC avanzato esegue tutte le funzioni del BASIC su disco più altre.

I dispositivi speciali disponibili con le tre versioni di BASIC sono:

- Serie di caratteri estesa. È possibile visualizzare un insieme di 256 caratteri diversi. Oltre alle lettere, numeri e simboli speciali, si può disporre di caratteri internazionali. Sono disponibili anche i simboli più comuni usati nella terminologia matematica e scientifica, come ad esempio le lettere greche ed altri simboli.

- Possibilità grafiche. Se si possiede l'Adattatore video colore/grafici, è possibile disegnare punti, linee e figure. È possibile indirizzare *tutti i punti* sullo schermo sia a media sia ad alta risoluzione. Per ulteriori informazioni vedere il Capitolo 3 e il Manuale di riferimento del BASIC.

- Dispositivi speciali di immissione/emissione. Il Personal Computer IBM possiede un altoparlante che emette dei suoni. Il BASIC prevede anche una penna luminosa e dei comandi per giochi che rendono più interessanti e divertenti i programmi.

Nel Manuale di riferimento del BASIC vengono descritti, in dettaglio, i comandi, le specifiche e le funzioni delle tre versioni del BASIC.

Per ciascuna descrizione è prevista una riga "versioni" che illustra le versioni BASIC che utilizzano il comando, la specifica o la funzione.
Se nel manuale di riferimento si ricerca, ad esempio, la specifica CHAIN, si può leggere:

Versioni	Cassetta	Disco	Avanzato	Compilatore
		***	***	(**)

Gli asterischi indicano quale versione del BASIC supporta la specifica. Questo esempio indica che si può utilizzare la specifica CHAIN nei programmi scritti in BASIC Avanzato o su Disco.

Si noterà anche che nell'esempio gli asterischi sotto la parola "Compilatore" sono tra parentesi. Ciò significa che con l'interprete BASIC la specifica opera in modo diverso rispetto al Compilatore BASIC del Personal Computer IBM. Il Compilatore è un pacchetto di programmi facoltativo disponibile presso l'IBM. (Il manuale relativo al Compilatore BASIC illustra queste differenze).

BASIC su cassetta

Il nucleo del BASIC è la versione su Cassetta che viene generata nella memoria di sola lettura di 32 Kbyte del Personal Computer IBM. È possibile utilizzare il BASIC su Cassetta con qualsiasi quantità di memoria ad accesso casuale disponibile nel Personal Computer IBM. La ampiezza di memoria utilizzata per i programmi e i dati dipende dalla memoria del PC. Il numero di byte disponibili viene visualizzato dopo l'accensione dell'elaboratore.

L'unico dispositivo di memoria utilizzabile per salvare le informazioni con il BASIC su Cassetta è un registratore a cassette. Non si possono utilizzare minidischi con il BASIC su Cassetta.

BASIC su disco

Questa versione del BASIC è un programma contenuto nel minidisco DOS del Personal Computer IBM. Il DOS è un altro prodotto software disponibile presso la IBM. E' indispensabile caricare in memoria, dal minidisco DOS, il BASIC su disco prima di utilizzarlo. La quantità di memoria disponibile per i dati ed i programmi viene visualizzata dopo l'avvio del BASIC.

Con il BASIC su disco sono disponibili i seguenti dispositivi:

- Immissione/emissione su disco oltre che su cassetta. Vedere l'Appendice A per ulteriori informazioni.
- Un orologio interno che registra la data e l'ora.
- Un supporto per le comunicazioni asincrone (RS232). (Per ulteriori dettagli vedere l'Appendice C del manuale di riferimento BASIC).
- Un supporto per due ulteriori stampatrici (tre in totale).

BASIC avanzato

Il BASIC avanzato è la forma più estesa di BASIC disponibile sul Personal Computer IBM ed esegue tutte le funzioni del BASIC su cassetta e su disco più altre. Come per il BASIC su disco è necessario caricarlo in memoria per poterlo utilizzare. Come per le altre versioni, dopo l'avvio del BASIC viene visualizzato il numero di byte disponibili per i dati ed i programmi.

Soltanto con il BASIC avanzato si hanno le seguenti funzioni:

- Salto condizionato. Un programma può rispondere ad una determinata condizione saltando automaticamente ad una specifica riga di programma. Le condizioni comprendono: le comunicazioni, sia la pressione di un tasto funzionale sulla tastiera che di un tasto sul comando per i giochi, l'attivazione della penna luminosa.

- Grafici avanzati. Esistono ulteriori specifiche (CIRCLE, DRAW, GET, PAINT, PMAP, POINT, PSET, PUT, VIEW e WINDOW) per la realizzazione di grafici più complessi.

- Supporto avanzato per la musica. La specifica PLAY consente l'utilizzo dell'altoparlante incorporato per creare della musica.

Capitolo 2. Come usare il BASIC

Indice

Avvio del BASIC

È facile avviare il BASIC sul Personal Computer IBM.

Avvio del BASIC su cassetta

Importante: Usare il BASIC su cassetta solo se nel sistema non esiste alcun tipo di disco. Se si dispone di un sistema a dischi è consigliabile l'uso del BASIC su disco o avanzato.

Se il computer è spento:

Accendere il computer.

Appariranno le parole "Version C" e il numero della versione del programma unitamente alla quantità dei byte disponibili.

Se il computer è acceso:

1. Togliere l'eventuale minidisco dall'unità A.
2. Tenendo premuti i tasti Ctrl e Altrn premere il tasto Canc.

Appariranno le parole "Version C" e il numero della versione del programma unitamente alla quantità dei byte disponibili.

Avvio del BASIC su disco

1. Inserire il minidisco IBM DOS nell'unità A.
2. Accendere il computer.

3. Immettere il comando BASIC, quando il DOS richiede l'immissione di un comando (A >).

   Saranno visualizzate le parole "Version D" e il numero della versione del programma unitamente alla quantità di byte disponibili.

Avvio del BASIC Avanzato

1. Avviare il DOS come descritto precedentemente.

2. Immettere il comando BASICA in risposta alla richiesta DOS (A >).

   Appariranno le parole "Version A" e il numero della versione del programma unitamente alla quantità di byte disponibili.

Ritorno al DOS

Talvolta è necessario ritornare al DOS dopo l'esecuzione di un programma BASIC: se si vuole, ad esempio, passare da BASIC su disco a BASIC avanzato.

Per ritornare al DOS dal BASIC:

1. Alla richiesta di comando del BASIC battere:

   ```
   SYSTEM
   ```

   quindi premere il tasto di immissione.

2. Quando appare la richiesta da parte del DOS, il DOS è pronto a ricevere un comando.

Per ulteriori informazioni consultare il Manuale di riferimento del BASIC (Comando SYSTEM).

Parametri del comando BASIC

Quando si avvia il BASIC su disco o avanzato si possono immettere dei parametri nel comando BASIC o BASICA. Questi parametri specificano l'ampiezza di memoria utilizzata dal BASIC per i programmi, i dati e le aree di memoria di transito; è possibile anche richiedere al BASIC di caricare immediatamente ed eseguire un programma.

Questi parametri non sono indispensabili; il BASIC opera anche senza di essi. Se non si è esperti di BASIC è consigliabile passare prima alla successiva sezione "Modi Operativi" per poi tornare a consultare questa parte.

Questo è il formato completo del comando BASIC:

BASIC[A] [*specfile*] [< *stdin*] [>] [> *stdout*]
[/F:*file*] [/S:*ampm*] [/C:*memcom*] [/M:[*mass spaziolav*]
[,*mass ampbloc*]] [/D]

Per la sintassi sono previste le seguenti convenzioni:

- Le parole in lettere maiuscole sono parole-chiave e devono essere immesse come illustrato. È possibile la combinazione di maiuscola e minuscola; il BASIC converte sempre le parole in maiuscole (a meno che non facciano parte di una stringa fra apici, di una annotazione o della specifica DATA).
- Le voci in caratteri corsivi devono essere sempre immesse.
- Le voci tra parentesi quadre sono facoltative.
- Una ellissi (...) indica una voce che può essere ripetuta più volte.
- Tutti i segni di punteggiatura ad eccezione delle parentesi quadre (virgole, parentesi, punto e virgola, trattini o segni di uguale) dovranno essere inclusi come illustrato.

specfile Questo parametro indica il nome del programma che deve essere caricato ed eseguito immediatamente. Il BASIC procederà come se fosse stato specificato RUN < *nome file* > dopo il completamento dell'inizializzazione. Nel BASIC 2.0 e nelle versioni successive il parametro *specfile* è stato esteso per permettere di specificare un percorso. Deve attenersi alle regole per la specifica dei file descritte nel Capitolo 3. In mancanza di indicazioni circa l'estensione, verrà assunta per difetto **.BAS**.
Usando questo formato di comando in un file AUTOEXEC. BAT si possono eseguire programmi BASIC con elaborazioni batch (a blocchi). I programmi così eseguiti dovrebbero terminare con il comando SYSTEM per consentire l'esecuzione del successivo comando per il file AUTOEXEC. BAT.
Notare che quando si indica *specfile* non viene visualizzata l'intestazione BASIC con le notizie relative al copyright.

< stdin Un programma BASIC normalmente riceve l'immissione dalla tastiera (dispositivo standard di immissione).
Usando il parametro < *stdin* il BASIC potrà ricevere l'immissione dal file specificato;
Quando si usa < *stdin* deve essere posizionato prima di ogni switch nella linea di comando.
Ci sono dei parametri che iniziano con una / (barra) e vengono chiamati "switch".
Per ulteriori informazioni consultare la parte "Reindirizzamento dell'Immissione ed Emissione standard" (disponibile solo col BASIC 2.0 e successivi livelli).

> stdout Un programma BASIC normalmente scrive i dati di emissione sullo schermo video (dispositivo standard di emissione). Usando il parametro > *stdout* il BASIC potrà scrivere

i dati su di un file o sul dispositivo specificato. Se usato, questo parametro dovrà essere indicato prima di ciascun switch sulla linea di comando. Per ulteriori informazioni consultare la parte "Reindirizzamento delle immissioni ed emissioni standard. (Disponibile solo col BASIC 2.0 e successivi livelli).

I seguenti parametri nella linea di comando BASIC sono switches:

/F:file Indica il numero massimo di file che possono essere aperti contemporaneamente durante l'esecuzione di un programma BASIC. Ciascun file richiede 62 byte di memoria per il blocco di controllo del file (FCB), con l'aggiunta di 128 byte per la memoria di transito. /S: modifica l'ampiezza della memoria di transito. Se si omette l'opzione **/F:,** si assume il valore di tre file. Il numero effettivo di file che possono essere aperti contemporaneamente dipende dal valore del parametro FILES = nel file di configurazione DOS "CONFIG. SYS". Il valore massimo di file è 15. Se non è specificato in CONFIG. SYS, il valore assunto è FILES = 8. I primi 3 agganci al file vengono fissati da *stdin, stdout, stderr, stdaux* e *stdprn*. Si rende necessario un ulteriore aggancio da parte del BASIC per LOAD, SAVE, CHAIN, NAME e MERGE. Nel caso in cui FILES sia uguale a 10 (FILES = 10) verranno riservati 6 file come file di I/E per il BASIC. /F:6 sarà, quindi, il valore massimo da assegnare nel caso in cui FILES sia uguale a 10 e si vogliano aprire tutti i file contemporaneamente.

Se si tenta l'apertura di un file ad esaurimento di tutti gli agganci, si verificherà errore **Too many files.**

/S:ampm Indica l'ampiezza della memoria di transito utilizzata con i file ad accesso casuale. Il parametro relativo alla lunghezza del record

nella specifica OPEN non può superare questo valore. L'ampiezza di memoria di transito assunta è di 128 byte. È possibile avere un valore massimo di 32767 byte.

/C:memcom

Imposta l'ampiezza della memoria di transito che deve ricevere i dati quando si usa l'Adattatore per le comunicazioni asincrone. Per utilizzare questa opzione è dunque indispensabile che il sistema disponga di questo Adattatore. La memoria di transito per la trasmissione dei dati con le comunicazioni è sempre allocata a 128 byte. Per **/C:** è possibile immettere un valore massimo di 32767 byte . Se si omette l'opzione /C:, vengono in ogni caso allocati 256 byte per la memoria di transito in ricezione. Se si possiede una linea ad alta velocità, è consigliabile usare /C: 1024. Se il sistema dispone di due Adattatori per le comunicazioni asincrone, le due memorie di transito in ricezione vengono impostate con l'ampiezza specificata da questa opzione. È possibile disabilitare il supporto RS232 usando un valore zero (/C:0), nel quale caso non verrà riservato dello spazio alla mer oria di transito per le comunicazioni ed il suppo. er le comunicazioni non sarà incluso nel caricamento del BASIC. Qualsiasi tentativo di successiva comunicazione I/E determinerà un errore **Device unavailable.**

/M:mass spaziolav

Indica il numero massimo di byte che possono essere utilizzati come spazio di lavoro del BASIC. Il BASIC tenterà di allocare 64K di ampiezza memoria per i segmenti dati e di stack. Se con il BASIC si utilizzano i sotto programmi in linguaggio macchina, si potrà specificare **/M:** nel comando BASIC.

Il parametro /**M: 32768,** ad esempio assegna 32768 byte ai segmenti dati e stack del BASIC consentendo all'utente l'utilizzo degli altri 32768 Byte per le eventuali routine in linguaggio macchina. Per ulteriori informazioni su come il BASIC utilizza la memoria consultare nell'Appendice B il paragrafo "Mappa della Memoria" nel manuale di riferimento del BASIC.

:mass ampbloc

Se si desidera caricare dei sottoprogrammi oltre il massimo indirizzo a cui può accedere il BASIC (65536), per riservare loro lo spazio necessario si deve usare l'opzione **mass ampbloc.**
Questa operazione risulta indispensabile se s'intende utilizzare la specifica SHELL (per ulteriori ragguagli vedere la specifica SHELL nel Manuale di riferimento del BASIC). In fase di esecuzione di una specifica SHELL si possono determinare degli inconvenienti quando si carica COMMAND. COM sopra i programmi di lavoro.

Questa opzione specifica il numero massimo di paragrafi assegnati al BASIC, in aggiunta all'ampiezza oltre il segmento stack, utilizzato per i sottoprogrammi in linguaggio macchina. I paragrafi consistono in blocchi di 16 byte ciascuno. Se: **mass ampbloc** viene omesso, viene assunto **& H 1000** (4096) e vengono assegnati 65536 byte (4096* 16) per i dati del BASIC e l'area di lavoro.

Qualora si vogliano assegnare, ad esempio, 65536 byte per il BASIC e 512 byte per i sottoprogrammi in linguaggio di macchina, occorrerà specificare /**M:, & H 1020** (4096 paragrafi per il BASIC e 32 paragrafi per i sottoprogrammi).

L'opzione **/M:, 2048** segnala al BASIC l'assegnazione e l'utilizzo di 2048 paragrafi (32768 byte) per i segmenti dati e di stack. Specificando **/M: 32000, 2048** verrà assegnato un valore massimo di 32768 byte, mentre il BASIC utilizzerà unicamente i primi 32000 byte, lasciando 768 byte per i sottoprogrammi. (Disponibile solo con il BASIC livello 2.0 e successivi livelli).

/D Questo parametro indica al BASIC che si vuole usare la doppia precisione nelle funzioni trascendenti (disponibile solo col BASIC 2.0 e successivi livelli). Se si specifica /D circa 3000 byte aggiuntivi vengono usati dalle funzioni trascendenti. Le funzioni che possono essere trasformate in precisione doppia sono: ATN, COS, EXP, LOG, SIN, SQR e TAN. Se **/D** è omesso, la funzione precisione doppia viene ignorata e lo spazio viene reso disponibile per il programma.

Nota: i valori *file, mass spaziolav, mass ampbloc, ampm e memcom* potranno essere espressi in numeri decimali, ottali (preceduti da & O) o esadecimali (preceduti da & H).

Esempi di utilizzo del comando BASIC:

```
BASIC PAGHE
```

Utilizza 64K di memoria e 3 file; il programma PAGHE .BAS viene caricato ed eseguito.

```
BASICA INVEN/F:6
```

Utilizza 64K di memoria e 6 file; il programma INVEN.BAS viene caricato ed eseguito (se si utilizzano 6 file occorrerà modificare il valore del file CONFIG. SYS in FILES = 10).

```
BASIC /C:0/M:32768
```

Disattiva il supporto RS232 ed utilizza unicamente 32K di memoria per l'area di lavoro BASIC.

```
BASIC /F:4/S:512
```

Utilizza 4 file e consente una massima lunghezza record di 512 byte

```
BASIC TTY/C:512
```

Utilizza 64K di memoria e 3 file; assegna 512 byte alle memorie di transito per la ricezione con supporto RS232 e 128 byte per le memorie di transito per la trasmissione. Il programma TTY.BAS viene caricato ed eseguito.

Reindirizzamento dell'immissione ed emissione standard.

Col BASIC livello 2.Ø e successivi livelli si possono reindirizzare l'immissione e l'emissione per il BASIC. L'immissione che normalmente avviene da tastiera può essere reindirizzata ad un qualsiasi file specificato nella riga di comando BASIC. L'emissione che normalmente avviene sullo schermo può essere indirizzata su qualsiasi file o dispositivo specificato nella riga di comando BASIC.

BASIC *specfile* [< *stdin*] [>] [> *stdout*]

Note:

- In fase di reindirizzamento tutte le specifiche INPUT, INPUT$, INKEY$, e LINE INPUT leggeranno dai file di immissione specificati anziché dalla tastiera.

- Se reindirizzate, per l'immissione ed emissione tutte le specifiche PRINT ed i messaggi d'errore scriveranno sul file o dispositivo di emissione specificato anziché sullo schermo.

- Se un file viene reindirizzato unicamente per l'emissione, tutti i dati che appaiono sullo schermo passeranno sul file di emissione specificato.

- Nel caso in cui venga reindirizzata solo l'emissione, i messaggi di errore continueranno ad apparire sull'unità di emissione reindirizzata e sullo schermo. Tutti i file vengono chiusi, il programma termina ed il controllo ritorna al DOS.

- Il file di immissione da "KYBD:" viene letto sempre dalla tastiera.

- L'emissione file su "SCRN" appare sempre sullo schermo.

- Il BASIC imposta sempre il codice dei tasti se è specificata la specifica ON KEY *(n)*.

- L'utilizzo contemporaneo dei tasti Ctrl-Stampa non determina la stampa del contenuto dello schermo quando la emissione standard viene reindirizzata.

- L'utilizzo contemporaneo dei tasti Ctrl-Interr riporta l'emissione all'unità di emissione standard, determina la chiusura di tutti i file e il controllo ritorna al DOS.

Esempi:

```
BASIC MYPROG >DATA.OUT
```

Nell'esempio i dati letti con INPUT, INPUT$, INKEY$ e LINE INPUT continueranno a essere immessi da tastiera. Tutte le emissioni sullo schermo andranno sul file DATA.OUT.
Ciò comprende, quindi, i dati stampati sullo schermo con una specifica PRINT, tutti i messaggi di errore che il BASIC stampa sullo schermo e i dati immessi in modo diretto.

```
BASIC MYPROG <DATA.IN
```

In questo esempio i dati letti con INPUT, INPUT$, INKEY$ e LINE INPUT verranno presi dal file DATA.IN e i dati emessi con PRINT continueranno a essere visualizzati sullo schermo.

```
BASIC MYPROG <MYINPUT.DAT >MYOUTPUT.DAT
```

Nell'esempio i dati letti con INPUT, INPUT$, INKEY$ e LINE INPUT verranno presi dal file MYINPUT.DAT e i dati scritti con PRINT andranno sul file MYOUTPUT.DAT.

```
BASIC MYPROG< \SALES\JOHN\TRANS.>>\
SALES\SALES.DAT
```

Nell'esempio i dati letti con INPUT, INPUT$, INKEY$ e LINE INPUT verranno dal file \ SALES \ JOHN \ TRANS. I dati scritti con PRINT saranno aggiunti al file\ SALES \ SALES.DAT.

Modi operativi

Dopo aver avviato il BASIC, viene visualizzato lo schermo **Ok. Ok** significa che il BASIC è pronto ad operare. Talvolta questo stato viene definito come *livello di comando*. A questo punto è possibile colloquiare con il BASIC in due modi: *diretto* o *indiretto*.

Modo diretto

Significa che il BASIC deve eseguire immediatamente la richiesta dopo che questa viene immessa. *Non* si deve far precedere alla specifica o al comando un numero di riga. Si possono visualizzare i risultati di operazioni aritmetiche o logiche immediatamente, oppure memorizzarli per usi successivi, ma le istruzioni non vengono salvate dopo la loro esecuzione. Questo modo è utile nella ricerca e correzione di errori, oppure per calcoli veloci che non richiedano un programma completo. Ad esempio:

```
PRINT 20+2
 22
```

Modo indiretto

Nel modo indiretto, per informare il BASIC che la riga che si sta immettendo fa parte di un programma, si inizia la riga con un *numero*. Essa viene così memorizzata come parte del programma. E' possibile eseguire il programma in memoria immettendo il comando RUN, ad esempio:

```
1 PRINT 20+2
RUN
 22
```

Tastiera

La tastiera è suddivisa in tre aree:

- Dieci tasti funzionali, F1 - F10, posti sul lato sinistro della tastiera.
- I tasti "tipo macchina per scrivere" si trovano nella parte centrale; comprendono le lettere ed i numeri.
- La tastiera numerica, simile a quella di una calcolatrice, è sul lato destro.

Tutti i tasti nelle tre aree della tastiera sono ripetitivi se si mantengono premuti.

Tasti funzionali

È possibile utilizzare i tasti funzionali:

- Come "semplici" tasti; per fare in modo che ciascun tasto batta automaticamente una sequenza di caratteri (fino a 15). Infatti, a questi tasti sono stati associati alcuni dei comandi più usati. È possibile modificarli, qualora lo si desideri. Per ulteriori dettagli vedere la parte "Specifica KEY" nel Manuale di riferimento BASIC.
- Per interrompere il programma nel BASIC avanzato usando la specifica ON KEY (vedere la specifica ON KEY (n) nel Manuale di riferimento).

Combinazione speciale di tasti

La parte che segue illustra alcune combinazioni di tasti utilizzabili nel BASIC.

Tasto Altrn

Il tasto Altrn consente di battere un'intera parola chiave BASIC con la sola pressione di un tasto.
Ciò avviene quando si preme e si tiene premuto il tasto Altrn unitamente ad uno dei tasti alfabetici (A-Z). Qui di seguito vengono elencate le parole chiave associate ad ogni lettera; le lettere a cui non sono state assegnate parole riservate vengono indicate con "(nessuna parola)".

A	AUTO	N	NEXT
B	BSAVE	O	OPEN
C	COLOR	P	PRINT
D	DELETE	Q	(nessuna parola)
E	ELSE	R	RUN
F	FOR	S	SCREEN
G	GOTO	T	THEN
H	HEX$	U	USING
I	INPUT	V	VAL
J	(nessuna parola)	W	WIDTH
K	KEY	X	XOR
L	LOCATE	Y	(nessuna parola)
M	MOTOR	Z	(nessuna parola)

Si usa il tasto Altrn anche con i tasti della tastiera numerica per immettere caratteri che non si trovano sui tasti. Per far ciò si tiene premuto il tasto Altrn e si batte il codice di tre cifre ASCII relativo a quel carattere (per una lista dei codici ASCII vedere il Manuale di riferimento, Appendice D).

Tasto Ctrl

Si usa il tasto Crtl anche per immettere alcuni codici e caratteri che altrimenti non sono disponibili sulla tastiera.

Ctrl-G, ad esempio è il carattere relativo alla *suoneria;* quando si stampa questo carattere l'altoparlante emette dei suoni. Per attivare il segnale acustico, premere Ctrl unitamente al tasto G.

Ctrl-Interr

La combinazione di questi 2 tasti interrompe l'esecuzione del programma alla successiva istruzione BASIC e lo riporta a livello di comando.
I due tasti vengono utilizzati, anche per porre termine alla numerazione automatica ottenuta mediante AUTO. Per far ciò, tenendo premuto Ctrl, premere Interr.

Ctrl-Stampa

Nel BASIC livello 2.Ø e successivi si possono utilizzare i tasti Ctrl e Stampa come interruttori di attivazione e disattivazione per inviare una parte di testo allo schermo ed alla stampatrice. Tenendo premuto il tasto Ctrl, premere Stampa e rilasciare, poi, entrambi i tasti per stampare l'emissione sulla stampante. Premere e rilasciare entrambi i tasti nuovamente per arrestare la stampa. Anche se l'utilizzo di questi due tasti consente alla stampatrice di registrare l'attività del sistema, alcune operazioni risultano rallentate, dal momento che l'elaborazione non prosegue finché la stampatrice non si arresta. Questa funzione viene disattivata quando si verifica una condizione di errore.

Si può anche utilizzare il tasto Ctrl unitamente ad altri tasti per "editare" programmi con l'editore di programmi BASIC.
Queste operazioni vengono illustrate nel capitolo seguente.

Editore di programmi BASIC

Qualsiasi riga di testo immessa quando il BASIC è a livello di comando viene elaborata dall'editore di programmi BASIC. L'editore di programmi è un "editore di riga di schermo"; ciò permette di modificare una riga ovunque sullo schermo, ma solo una riga per volta. La modifica diventa effettiva quando si preme il tasto Immiss su quella riga.

Per familiarizzarsi con tutti i suoi aspetti è consigliabile immettere un programma esemplificativo ed esercitarsi nell'utilizzo di tutte le possibilità di edizione, tentando, poi, l'edizione di alcune righe mentre si prendono in esame le informazioni che seguono.

Tasti speciali per l'editore di programmi

Si usano i tasti sulla tastiera numerica, il tasto di ritorno unitario e il tasto Ctrl per spostare il cursore ad una determinata posizione sullo schermo, per inserire o cancellare dei caratteri.

Tasto	Funzione
Rit Curs	Sposta il cursore nell'angolo superiore sinistro dello schermo video.
Ctrl-Rit Curs	Elimina totalmente il contenuto dello schermo, posizionando il cursore nell'angolo superiore sinistro dello schermo video.

↑ (Cursore verso l'alto) — Sposta il cursore di una riga in senso ascendente

↓ (Cursore verso il basso) — Sposta il cursore di una riga in senso discendente.

← Cursore verso sinistra) — Sposta il cursore di una posizione verso sinistra. Se il cursore avanza oltre il limite sinistro del video, si sposta sul lato destro dello schermo alla riga precedente.

→ (Cursore verso destra) — Sposta il cursore di una posizione verso destra. Se il cursore avanza oltre il limite destro del video, si sposta sul lato sinistro dello schermo alla riga successiva.

Ctrl → (Parola successiva) — Sposta il cursore direttamente alla *parola successiva*. Una parola viene definita come un carattere o un gruppo di caratteri che iniziano con una lettera o un numero. Le parole sono separate da spazi in bianco o caratteri speciali.

Supponiamo ad esempio di avere la riga seguente:

```
LINE (L1,LOW2)-(MAX,48) ,3 , BF
```

Come si può notare il cursore si trova sulla O della parola LOW2. Se si utilizzano Ctrl + → (Parola successiva) il cursore si sposterà all'inizio della parola successiva (MAX):

```
LINE (L1,LOW2)-(MAX,48) ,3 , BF
```

Se si premono ancora i due tasti il cursore si sposterà alla parola successiva, che è il numero 48;

```
LINE (L1,LOW2)-(MAX,48) , 3 , BF
```

Ctrl ← (Parola precedente)	Sposta il cursore a sinistra, all'inizio della parola precedente. La parola precedente è la lettera o il numero alla sinistra del cursore preceduto da uno spazio in bianco o da un carattere speciale.
	Ad esempio, si supponga di avere:
	`LINE (L1,LOW2)-(MAX,48) ,3 , BF`
	Se si premono Ctrl + ← (Parola precedente) il cursore si sposta all'inizio della parola BF:
	`LINE (L1,LOW2)-(MAX,(48+) ,3 ,BF`
	Se si premono ancora Ctrl + ← il cursore si sposta alla parola precedente, che è il numero 3:
	`LINE (L1,LOW2)-(MAX,48) ,3 , BF`

Fine	Sposta il cursore alla fine della riga logica.
Ctrl – Fine	Cancella tutti i dati fino alla fine di una riga logica.

Ins

Attiva o disattiva il modo «Inserimento»; in questo modo operativo il cursore copre la metà inferiore della posizione di un carattere.

Quando si opera un inserimento il carattere sopra il cursore e quello ad esso successivo vengono spostati a destra e i caratteri vengono inseriti. Quando i caratteri superano il lato destro dello schermo video, vengono riportati sul margine sinistro della riga successiva.

Se l'inserimento risulta disattivato, qualsiasi carattere immesso sostituisce il carattere già esistente sulla riga.

Nota: l'inserimento viene inoltre disattivato premendo uno dei tasti di spostamento del cursore, oppure il tasto di immissione.

Canc

Cancella il carattere nella posizione del cursore. Tutti i caratteri alla destra di quello cancellato si spostano a sinistra di una posizione per riempire lo spazio vuoto. Si verifica l'andata a capo della riga; cioè, se la riga logica supera quella fisica, i caratteri sulla riga successiva si spostano a sinistra di una posizione per riempire lo spazio precedente ed il carattere nella prima colonna di ogni riga successiva si sposta alla fine della riga precedente.

← (Ritorno unitario)	Cancella l'ultimo carattere immesso, cioè il carattere alla sinistra del cursore. Tutti i caratteri alla destra di quello cancellato si spostano a sinistra di una posizione per riempire lo spazio. I caratteri e le righe successive, nell'ambito della riga logica, si spostano in senso ascendente (come nell'utilizzo di Canc).
Esc	Se si utilizza questo tasto in un punto qualsiasi della riga, l'intera riga logica viene eliminata dallo schermo; la riga non viene trasferita al BASIC per l'elaborazione, ma se si tratta di una riga di programma (che inizia con un numero di riga) non viene eliminata dal programma in memoria.
Ctrl-Interr	Ritorna a livello di comando senza salvare le eventuali modifiche eseguite sulla riga da editare. Non elimina, come il tasto Esc, la riga dallo schermo.

→|
(Tabulazione) Sposta il cursore al successivo fermo di tabulazione, che si trova ogni otto caratteri (a posizione 1,9,17 ecc.)

Se l'inserimento è disattivato, premendo il tasto di tabulazione si sposta il cursore fino al successivo fermo di tabulazione.

Ad esempio, si supponga di avere la seguente riga:

```
10 REM  questa è una nota
```

Se si preme il tasto di tabulazione, il cursore si sposta alla nona posizione, come illustrato:

```
10 REM questa è una nota
```

Quando si è in modo inserimento, premendo il tasto di tabulazione vengono inseriti degli spazi in bianco a partire dalla posizione del cursore fino al successivo fermo di tabulazione. Si verifica l'andata a capo della riga come descritto per il tasto **Ins.**
Per esempio supponiamo di avere questa riga:

```
10 REM questa è una nota
```

Premendo Ins e poi →|, vengono inseriti degli spazi in bianco fino alla posizione 17:

```
10 REM qu      esta è una nota
```

Come effettuare le correzioni sulla riga

Poiché ogni riga di testo immessa mentre il BASIC è a livello di comando viene elaborata dall'editore di programmi, per modificare la riga corrente è possibile utilizzare uno dei tasti descritti nella precedente sezione. Il BASIC è sempre a livello di comando dopo la richiesta OK e finché non viene specificato il comando RUN.

È possibile estendere la riga logica oltre quella fisica dello schermo, superando il limite dello schermo con la battitura. Il cursore *passerà* alla successiva riga.

Si può utilizzare anche l'avanzamento di riga (Ctrl-tasto di immissione) per stampare il testo seguente sulla successiva riga dello schermo. La riga logica non viene elaborata; ciò si verifica solo premendo il tasto di immissione.

Notare che, in realtà, l'avanzamento di riga produce il riempimento con caratteri in bianco del resto della riga fisica dello schermo; questi spazi in bianco sono compresi nei 255 caratteri consentiti per una riga BASIC (il carattere di avanzamento riga non viene aggiunto al testo e non viene considerato nella riga).

Quando, alla fine, si preme il tasto di immissione, l'intera riga logica viene passata al BASIC per l'elaborazione.

Modifica di caratteri: Se si sta immettendo una riga e si scopre che si è battuto qualcosa in modo errato, è possibile effettuare delle modifiche. Usare il tasto di spostamento del cursore verso sinistra o altri tasti di spostamento per portare il cursore sulla posizione nella quale si è verificato l'errore e ribattere le lettere corrette (sopra quelle errate). Si può riportare poi il cursore alla fine della riga con il tasto di spostamento verso destra o con un qualsiasi tasto di spostamento e continuare l'immissione.

Cancellazione di caratteri: Se si nota di aver battuto un carattere in più sulla riga che si sta immettendo, è possibile

cancellarlo usando il tasto Canc. Usare il tasto di spostamento del cursore verso sinistra o altri tasti di spostamento per portare il cursore sul carattere da cancellare, premere poi il tasto Canc. Utilizzare, quindi, i tasti di spostamento del cursore verso destra o il tasto Fine per portare il cursore alla fine della riga e continuare l'immissione.

Aggiunta di caratteri: Se sono stati omessi dei caratteri nella riga che si sta immettendo, spostare il cursore alla posizione in cui si vuole inserire il nuovo carattere. Premere il tasto Ins per attivare il modo inserimento; battere i caratteri che si vogliono aggiungere: questi verranno inseriti nella posizione del cursore ed i caratteri successivi risulteranno spostati verso destra. Quando si è pronti a continuare l'immissione alla fine della riga, usare i tasti di spostamento del cursore verso destra o il tasto Fine per portare il cursore in posizione e continuare l'immissione. Usando uno di questi tasti il modo inserimento verrà disattivato automaticamente.

Cancellazione di una parte della riga: Per troncare una riga nella posizione del cursore, premere i tasti Ctrl-Fine

Ad esempio, si supponga di avere:

```
10 REM *** fiori fiori fiori
```

Il cursore è posizionato sotto la **f** della prima parola **fiori;** di conseguenza per cancellare dalla posizione del cursore sino al termine della riga premere Ctrl-Fine

```
10 REM ***
```

Annullamento di una riga: Per annullare una riga che si sta immettendo, premere il tasto Esc in qualsiasi punto della riga. Non è necessario premere il tasto di immissione; l'intera riga immessa precedentemente verrà cancellata.

Ad esempio, si supponga di avere:

```
QUESTA RIGA NON HA SIGNIFICATO_
```

Anche se il cursore si trova alla fine della riga, il tasto Esc la cancella interamente:

—

Immissione o modifica di un programma BASIC

Ogni riga di testo immessa, che inizia con un numero, viene considerata come *riga di programma.*

Una riga di programma BASIC inizia sempre con un numero, termina con un ritorno a margine e può contenere al massimo 255 caratteri, compreso il ritorno a margine. Se una riga contiene più di 255 caratteri, quelli in eccesso vengono troncati (eliminati) quando si preme il tasto Immiss. Anche se i caratteri in eccesso appaiono sullo schermo, essi non vengono elaborati dal BASIC.

Le parole chiave BASIC ed i nomi di variabili possono essere immessi in una combinazione qualsiasi di maiuscole/minuscole. L'editore di programmi convertirà tutto in caratteri maiuscoli eccettuate le note, le specifiche DATA e le stringhe racchiuse tra apici.

Il BASIC talvolta modifica l'immissione; si supponga, ad esempio, di usare il punto interrogativo (?) al posto della parola PRINT in una riga di programma. Quando successivamente si lista (LIST) la riga, il punto interrogativo viene modificato in PRINT seguito da uno spazio, dato che il punto interrogativo è l'abbreviazione per PRINT. Questa espansione può troncare la fine di una riga, se la lunghezza supera i 255 caratteri.

Edizione di righe in qualsiasi punto dello schermo

È possibile editare qualsiasi riga di programma sullo schermo usando semplicemente i tasti di spostamento del cursore per spostare il cursore fino alla posizione che richiede la modifica. A questo punto è possibile utilizzare una delle tecniche precedentemente descritte per modificare, cancellare o aggiungere caratteri alla riga. Le modifiche non diventeranno parte del programma finché non si usa Immiss.

Se si vogliono modificare delle righe di un programma non visualizzate al momento, è possibile visualizzarle utilizzando il comando LIST o EDIT. Vedere il comando EDIT e LIST nel Manuale di riferimento BASIC.

La riga di un programma può essere duplicata spostando il cursore sulla riga da duplicare e riscrivendo sul vecchio il nuovo numero di riga. Premendo il tasto Immiss, nel programma si troveranno entrambe le righe.

È opportuno considerare che quando si apportano delle modifiche ad una riga già immessa, non è necessario spostare il cursore alla fine della riga logica prima di premere Immiss. L'Editore di programma conosce la fine di ogni riga logica ed elabora l'intera riga anche se si preme Immiss all'inizio della riga stessa.

> **Nota:** Se si sta immettendo un programma, è molto utile usare il comando AUTO. Si deve uscire dal modo AUTO premendo i tasti Ctrl-Interr prima di modificare qualsiasi riga che non sia quella corrente.

Ricordarsi che le modifiche effettuate usando queste tecniche modificano il programma solamente in memoria. Per salvare permanentemente il programma modificato si deve usare il comando SAVE prima di immettere un comando NEW o uscire dal BASIC.

Vedere la parte riguardante il comando SAVE nel Manuale di riferimento BASIC.

Aggiunta di una nuova riga di programma: Immettere un numero di riga valido (Ø-65529) che non sia già stato usato nel programma, seguito da almeno un carattere non in bianco e da un ritorno a margine. La riga sarà salvata come parte del programma BASIC in memoria.

Se esiste già una riga con lo stesso numero, la vecchia riga viene cancellata e sostituita da quella nuova.

Se si tenta di aggiungere una riga ad un programma e non c'è più spazio in memoria, si verifica un errore e la riga non viene aggiunta.

Sostituzione di una riga di programma: si può sostituire una riga già esistente specificando il numero di riga già presente nel programma, il nuovo testo e premendo, poi, il tasto Immiss.

Cancellazione di righe di programma: Per cancellare una riga di programma esistente, battere solo il numero di riga ed usare il tasto Immiss. Ad esempio, se si immette:

```
1Ø
```

questa operazione cancella la riga 10 dal programma.

Si può usare il comando DELETE per cancellare un gruppo di righe di programma. Vedere la descrizione del comando DELETE nel Manuale di riferimento BASIC.

Non usare il tasto Esc per cancellare righe di programma, poiché questo tasto cancella la riga solamente dal video. Se la riga esiste nel programma BASIC, vi rimane.

Notare che se si tenta di cancellare una riga inesistente si verifica l'errore **Undefined line number.**

Cancellazione di un intero programma: Per cancellare un intero programma attualmente residente in memoria, immettere il comando NEW.

Errori di sintassi

Se durante l'esecuzione di un programma si riscontra un errore di sintassi, il BASIC evidenzia il messaggio **Sintax error** unitamente al numero di riga che contiene l'errore. Il numero di riga viene visualizzato, poi, per la correzione. Esempio:

```
10 A = 2$12
RUN
Errore sintassi in 10
10 A = 2$12
```

Il BASIC ha posizionato il cursore sotto alla cifra 1. Per correggere l'errore posizionare il cursore sotto il segno di dollaro ($) e modificarlo nel segno più (+) e premere, poi, il tasto Immiss. La riga corretta viene memorizzata nel programma.

Quando si esegue l'edizione di una riga e la si memorizza di nuovo nel programma, mentre questo viene interrotto (come nell'esempio), può accadere quanto segue:

- Si perdono tutte le variabili e le schiere; esse vengono ripristinate a zero o annullate.
- I file aperti vengono chiusi.
- Non si può utilizzare CONT per continuare il programma.

Esecuzione di un programma BASIC

L'esecuzione di un programma BASIC memorizzato su di un minidisco si svolge in due fasi.

La prima fase consiste nel trasferire una copia del programma dal minidisco alla memoria dell'elaboratore. Questa fase è chiamata caricamento del programma e si effettua tramite il comando LOAD.

La seconda fase consiste nell'effettiva esecuzione delle istruzioni del programma e si effettua tramite il comando RUN.

RUN *specfile* carica ed esegue il programma specificato.

Maggiori informazioni sulla sequenza dei comandi si avranno consultando i programmi SAMPLES nel minidisco Programmi Complementari DOS.

Esecuzione di programmi SAMPLES

1. Assicurarsi che il DOS sia pronto e la A > sia visualizzata

2. Inserire il minidisco DOS nel modulo A.

3. Battere:

   ```
   basic
   ```

   e premere il tasto Immiss.

   Il BASIC su disco viene caricato nella memoria dell'elaboratore. Apparirà sullo schermo la risposta del BASIC (OK), il copyright IBM BASIC ed i 10 tasti funzionali del BASIC.

4. Estrarre il minidisco DOS. Inserire nell'unità A il minidisco Programmi Complementari DOS.

5. Battere:

   ```
   run "samples
   ```

   quindi premere il tasto Immiss.

6. Quando sullo schermo appare il titolo SAMPLES, premere la barra di spaziatura: apparirà una lista chiamata "menù".

7. Selezionare il tasto corrispondente alla scelta che si vuole effettuare. Le note successive al menù indicano cosa sia necessario per eseguire il programma scelto: se occorre utilizzare il Basic avanzato (BASICA), oppure se sia necessario un video grafico a colori e l'ampiezza di memoria necessaria per l'esecuzione, da parte del sistema, dei programmi esemplificativi.

   Il Basic avanzato è richiesto per alcuni programmi esemplificativi, ma è possibile utilizzarlo per eseguirli tutti.
   Per tentare l'esecuzione del programma H, COLORBAR, si specifica:

   h

Non occorre premere il tasto di Immissione; nel caso di un video monocromatico appariranno due file di barre in un unico colore. Nel caso di video a colori, appariranno barre di vari colori. Sistemare il video, regolandone i comandi, in modo da ottenere una buona visualizzazione.

Dopo aver immesso la scelta programma seguire le istruzioni sullo schermo video, premere il tasto Esc per arrestare un programma e ritornare al menù.

Dopo aver provato i programmi da esaminare ed essere ritornati al menù, premere di nuovo Esc. Apparirà sullo schermo OK.

Esecuzione di programmi COMM

Sul minidisco Programmi complementari DOS è presente anche un esempio di programma per telecomunicazioni. Questo consente di stabilire un collegamento per comunicazioni asincrone con un altro Personal Computer IBM, con un elaboratore Serie/1 IBM, o con altre 2 reti per comunicazioni.

I programmi COMM funzioneranno solo se è presente il dispositivo adeguato (Adattatore per comunicazioni asincrone). Consultare il punto di vendita per qualsiasi aiuto nell'utilizzo dei dispositivi esterni.

Nota: i programmi COMM possono anche essere utilizzati come traccia per scrivere i propri programmi di telecomunicazione.

Osserviamo il menù dei programmi COMM (si può fare anche se non si prevede di comunicare con un altro elaboratore).

Esecuzione di un programma BASIC da un altro minidisco.

Si può eseguire un programma BASIC da un altro minidisco caricando prima il BASIC e specificando, poi, l'unità e/o il percorso sui quali è posto il programma.

Con una unità minidischi

In un sistema che preveda una sola unità minidischi non occorre modificare o aggiungere una specifica di unità a *specfile*; occorre inserire unicamente il minidisco nell'unità dopo aver richiamato il BASIC.

1. Caricare il BASIC.
2. Estrarre dal modulo A il minidisco DOS ed inserirvi il minidisco contenente il programma.
3. Caricare ed eseguire il programma.

Con più di una unità minidischi

Con un sistema con più di una unità minidischi occorrerà indicare l'ubicazione del programma.

1. Caricare il BASIC.
2. Caricare ed eseguire il programma dall'unità o dal percorso indicato.

Capitolo 3. Informazioni in BASIC

Indice

Note:

Formato della riga

Le righe di programma nel BASIC hanno il seguente formato:

```
nnnnn specifica BASIC[:specifica BASIC...] ['commento]
```

e terminano con l'utilizzo del tasto di immissione.

Numeri di riga: "nnnnn" indica il numero di riga (da 1 a 5 cifre). Ogni riga di programma BASIC inizia con un numero. Si usano i numeri per indicare l'ordine in cui le righe di programma vengono memorizzate e anche come punti di riferimento per salti ed edizioni. I numeri di riga devono essere compresi tra 0 e 65529. Si può usare un punto (.) nei comandi LIST, AUTO, DELETE e EDIT per fare riferimento alla riga.

Specifiche BASIC: una specifica BASIC può essere *eseguibile* oppure *non eseguibile*. Le specifiche eseguibili sono istruzioni che segnalano al BASIC le operazioni da eseguire durante l'esecuzione di un programma. PRINT X ad esempio, è una specifica eseguibile. Le specifiche non eseguibili come DATA o REM, ad esempio, contengono unicamente informazioni e non determinano alcuna azione da parte del programma quando il BASIC le esamina. Il Manuale di riferimento BASIC contiene la spiegazione in dettaglio delle specifiche BASIC.

Su di una riga è possibile indicare più di una specifica BASIC, ma ciascuna di esse deve essere separata dall'altra da due punti (:); il numero totale di caratteri non dovrà superare 255, tasto Immiss compreso.
Ad esempio:

```
10 FOR I=1 TO 5: PRINT I: NEXT
RUN
 1
 2
 3
 4
 5
```

Commenti: è possibile aggiungere dei commenti alla fine di una riga usando (') per separare il commento dal resto della riga.

Insieme di caratteri

L'insieme di caratteri riconosciuti dal BASIC comprende caratteri alfabetici, numerici e speciali.

Nel BASIC i caratteri alfabetici sono le lettere maiuscole e minuscole dell'alfabeto. I caratteri numerici sono le cifre da 0 a 9.

I seguenti caratteri speciali hanno un significato specifico nel BASIC.

Carattere	Nome
	spazio in bianco
=	uguale o simbolo di assegnazione
+	segno più o simbolo di concatenazione
−	segno meno
*	asterisco o simbolo di moltiplicazione
/	barra o simbolo di divisione
\	barra al contrario o simbolo di divisione di interi o delimitatore percorso
^	segno di omissione o simbolo esponenziale
(	aperta parentesi
)	chiusa parentesi
%	segno di percentuale o di carattere di dichiarazione intero
#	segno di numero o carattere di dichiarazione precisione doppia
$	segno di dollaro o carattere di dichiarazione della stringa
!	punto esclamativo o carattere di dichiarazione precisione singola
&	'e' commerciale
,	virgola
.	punto o punto decimale
'	apice singolo o delimitatore
;	punto e virgola
:	due punti o separatore di specifica
?	punto interrogativo (abbreviazione per PRINT)
<	minore di
>	maggiore di
"	apici doppi o delimitatore di stringa
_	sottolineatura

È possibile visualizzare o stampare molti caratteri anche se essi non hanno un particolare significato per il BASIC. Vedere nell'Appendice D "Codici carattere ASCII" la lista completa di questi caratteri, nel Manuale di riferimento BASIC.

Parole riservate

Le seguenti parole hanno un significato speciale per il BASIC e sono denominate *parole riservate*. Esse comprendono: comandi BASIC, specifiche, nomi di funzioni e nomi di operatori. Non possono essere utilizzate come nomi di variabili.

Si debbono sempre delimitare le parole riservate separandole dai dati o dalle altre parti di una specifica BASIC a mezzo di spazi o caratteri speciali consentiti dalla sintassi, in modo che il BASIC le riconosca.

ABS
AND
ASC
ATN
AUTO
BEEP
BLOAD
BSAVE
CALL
CDBL
CHAIN
CHDIR
CHR$
CINT
CIRCLE
CLEAR
CLOSE
CLS
COLOR
COM
COMMON
CONT
COS
CSNG
CSRLIN
CVD
CVI
CVS
DATA
DATE$
DEF
DEFDBL
DEFINT
DEFSNG
DEFSTR
DELETE
DIM
DRAW
EDIT
ELSE
END
ENVIRON
ENVIRON$
EOF
EQV
ERASE
ERDEV
ERDEV$
ERL
ERR
ERROR
EXP
FIELD
FILES

FIX
FN*xxxxxxxx*
FOR
FRE
GET
GOSUB
GOTO
HEX$
IF
IMP
INKEY$
INP
INPUT
INPUT #
INPUT$
INSTR
INT
INTER$
IOCTL
IOCTL$
KEY
KILL
LEFT$
LEN
LET
LINE
LIST
LLIST
LOAD
LOC
LOCATE
LOF
LOG
LPOS
LPRINT
LSET
MERGE
MID$
MKDIR
MKD$
MKI$
MKS$
MOD
MOTOR
NAME
NEW
NEXT
NOT
OCT$
OFF
ON
OPEN
OPTION
OR
OUT
PAINT
PEEK
PEN
PLAY
PMAP
POINT
POKE
POS
PRESET
PRINT
PRINT #
PSET
PUT
RANDOMIZE
READ
REM
RENUM
RESET
RESTORE
RESUME
RETURN
RIGHT$
RMDIR
RND
RSET
RUN
SAVE
SCREEN
SGN
SHELL
SIN

SOUND	TO
SPACE$	TROFF
SPC(	TRON
SQR	USING
STEP	USR
STICK	VAL
STOP	VARPTR
STR$	VARPTR$
STRIG	VIEW
STRING$	WAIT
SWAP	WEND
SYSTEM	WHILE
TAB(	WIDTH
TAN	WINDOW
THEN	WRITE
TIME$	WRITE #
TIMER	XOR

Costanti

Le costanti sono valori forniti quando si scrive un programma BASIC e che non cambiano valore durante l'esecuzione. Vi sono due tipi di costanti: caratteri (stringhe) e numeriche.

Una stringa è una sequenza di 255 caratteri, al massimo, racchiusi tra virgolette. I caratteri possono essere lettere, numeri o simboli. Esempi di costanti a stringa:

```
"CIAO"
"25.000"
"Numero di impiegati"
```

Vi sono alcuni casi in cui il BASIC sa che un particolare dato dovrà essere una costante stringa e non sono richiesti apici; questi casi vengono elencati in questo manuale. Se si inizia una stringa con un apice e non viene indicato l'apice di chiusura, il BASIC assume un apice a fine riga. Ciò risulta corretto, quindi, solo nel caso in cui la stringa sia l'ultimo dato della riga.

Le costanti numeriche sono numeri positivi o negativi. Il segno più (+) è facoltativo in un numero positivo. Le costanti numeriche nel BASIC non possono contenere virgole. Ci sono cinque modi per indicare le costanti numeriche:

Intero

Numeri interi tra −32768 e +32767, compreso. Le costanti intere non hanno segni decimali.

Segno decimale fisso

Numeri reali positivi o negativi, cioè numeri che contengono segni decimali.

Segno decimale mobile

Numeri positivi o negativi rappresentati in forma esponenziale (come nei termini scientifici). Una costante con segno decimale mobile consiste in un intero, facoltativamente con segno, o in un numero con segno decimale fisso (la mantissa) seguito da una lettera E e un intero, facoltativamente con segno (l'esponente). Le costanti con segni decimali mobili a doppia precisione usano la lettera D invece di E.
La E (o D) significa "dieci volte alla potenza di".

È possibile rappresentare qualsiasi numero da 2,9E-39 a 1,7E+38 (positivo o negativo) costante con segno decimale mobile.

Per maggiori informazioni vedere "Precisione numerica".

Ad esempio,

```
23E-2
```

In cui 23 è la mantissa e -2 rappresenta l'esponente. Questo numero può essere letto come "ventitre per dieci alla meno due". È possibile scriverlo nella forma con segno decimale fisso 0.23.

Altri esempi:

```
235.988E-7
```

equivale a: 0000235988
(precisione singola)

```
2359D6
```

equivale a: 2359000000
(precisione doppia)

Esadecimale Numeri interi esadecimali con quattro cifre al massimo ed un prefisso &H. Le cifre esadecimali sono i numeri Ø-9,A,B,C,D,E e F. Esempi:

```
&H76
&H32F
```

Ottale Numeri ottali con sei cifre al massimo ed un prefisso &O oppure solo &. Le cifre ottali sono comprese tra Ø e 7. Esempi:

```
&0347
&1234
```

Precisione numerica

È possibile memorizzare internamente i numeri come interi, precisione singola, precisione doppia. Le costanti immesse nel formato intero, esadecimale o ottale vengono memorizzate in due byte di memoria e vengono interpretate come numeri interi. Con la precisione doppia, i numeri vengono memorizzati con 17 cifre di precisione e stampati con 16 cifre al massimo. Con la precisione singola, vengono memorizzate e stampate al massimo sette cifre, benché siano significative solo sei cifre.
In precisione singola vengono stampate 7 cifre, dal momento che qualsiasi elaborazione intermedia utilizza tutti e sette i caratteri. Per assicurarsi la precisione dei risultati finali, occorre specificare tutte le 7 cifre prima dei risultati finali, in fase di elaborazioni interattive.

Una costante a precisione singola è una qualsiasi costante numerica scritta in uno qualsiasi dei seguenti modi:

- sette cifre al massimo
- forma esponenziale con E
- punto esclamativo (!) finale

Una costante a precisione doppia è una qualsiasi costante numerica scritta in uno qualsiasi dei seguenti modi:

- otto o più cifre
- forma esponenziale con D
- segno numerico (#) finale

La tabella che segue riassume la precisione e l'intervallo di numeri interi, numeri a precisione singola e numeri a precisione doppia:

TIPO	LIMITI	PRECISIONE
Intero	da −32768 a 32767	Esatta
Precisione singola segno decimale mobile	da 10E-38 a 10+38	6 cifre decimali
Precisione doppia segno decimale mobile	da 10D-38 a 10D+38	16 cifre decimali

Questi sono esempi di costanti a precisione singola o doppia:

Precisione singola	Precisione doppia
46.8	345692811
−1.09 E-06	−1.09432D-06
3489.0	3489.0#
22.5!	7654321.1234

Conversione nel BASIC dei numeri da una precisione all'altra.

Se necessario, il BASIC converte un numero da una precisione all'altra. Tenere presenti le seguenti regole ed esempi.

1. Se si assegna un valore numerico di una precisione ad una variabile numerica di precisione diversa, il numero sarà memorizzato con la precisione dichiarata nel nome della variabile di arrivo.

   Esempio:

   ```
   10 A% = 23.42
   20 PRINT A%
   RUN
    23
   ```

2. Si verifica l'arrotondamento, invece del troncamento, quando si assegna un valore di precisione più alto ad una variabile con precisione minore (ad esempio, passando da precisione doppia a singola).

   Esempio:

   ```
   10 C = 55.8834567#
   20 PRINT C
   RUN
    55.88346
   ```

   Ciò riguarda non solo le specifiche di assegnazione (ad es. I% = 2.5 diventa I% = 3), ma anche la valutazione delle funzioni e specifiche (ad es. TAB(4.5) va alla quinta posizione, A(1.5) corrisponde ad A(2) ed X =11.5 MOD 4 darà un valore 0 per X).

3. Se si esegue una conversione da un numero con precisione inferiore ad uno con precisione superiore, quest'ultimo non può essere più preciso di quello con precisione inferiore. Ad esempio, se si assegna un valore a precisione singola (A) ad una variabile con precisione doppia (B#), solo le prime sei cifre di B# saranno precise, poiché A ne fornisce solo sei. L'errore può essere circoscritto con questa formula:

```
ABS(B#-A) < 6.3E-8 * A
```

Cioè il valore assoluto della differenza tra il numero a precisione doppia stampato e il valore a precisione singola originale è inferiore di 6.3E-8 volte rispetto al valore a precisione singola originale.

Esempio:

```
10 A = 2.04
20 B# = A
30 PRINT A;B#
RUN
 2.04  2.039999961853027
```

4. Quando un'espressione viene eseguita, tutti gli operandi di un'operazione aritmetica o relazionale vengono convertiti allo stesso grado di precisione, cioè quello dell'operando più preciso. Il risultato di un'operazione aritmetica inoltre, viene riportato con questo grado di precisione.

Esempi:

```
10 D# = 6#/7
20 PRINT D#
RUN
 .8571428571428571
```

L'operazione aritmetica è stata eseguita con precisione doppia ed il risultato viene riportato in D# come valore a precisione doppia.

```
10 D = 6#/7
20 PRINT D
RUN
 .8571429
```

L'operazione aritmetica è stata eseguita con precisione doppia ed il risultato è stato riportato a D (variabile a precisione singola) arrotondato e stampato come valore a precisione singola.

5. Gli operatori logici convertono gli operandi in interi e riportano un risultato intero. Gli operandi devono essere compresi tra –32768 e 32767, altrimenti si verifica un errore per condizione di **overflow** (eccedenza).

Variabili

Le variabili sono dei nomi utilizzati per rappresentare i valori usati in un programma BASIC.
Come per le costanti, vi sono due tipi di variabili: numerica e stringa. Una variabile numerica ha sempre un valore numerico, mentre una variabile a stringa può avere solamente una stringa di caratteri.

La lunghezza di una variabile a stringa può essere al massimo di 255 caratteri.

Si può impostare il valore di una variabile a una costante, oppure impostare il suo valore come risultato di calcoli o varie specifiche di immissione dei dati nel programma. In entrambi i casi, il tipo di variabile (stringa o numerica) deve corrispondere al tipo di dati che le vengono assegnati.

Se si usa una variabile numerica prima di assegnarle un valore, si assume che questo sia zero. Si assume che il valore delle variabili a stringa sia inizialmente nullo; cioè che non contengano caratteri e la lunghezza sia zero.

Come denominare una variabile

I nomi di variabili BASIC possono avere una lunghezza massima di 40 caratteri.

I caratteri consentiti in un nome di variabile sono le lettere, i numeri ed il punto decimale. Il primo carattere deve essere una lettera. Come ultimo carattere del nome sono consentiti anche i caratteri speciali che identificano il tipo di variabile. Per maggiori informazioni sui tipi, vedere la sezione seguente, "Come dichiarare i tipi di variabile".

Un nome di variabile non può essere una parola riservata, ma può esserne parzialmente costituito (fare riferimento a "Parole riservate" per una lista completa di queste parole).

```
10 EXP = 5
```

non è valido, poiché EXP è una parola riservata. Tuttavia,

```
10 EXPONENT = 5
```

è valido, poiché EXP è solo una parte del nome dalla variabile.

> **Nota:** si assume che una variabile che inizia con FN richiami una funzione definita dall'utente (vedere la Specifica DEF FN nel Manuale di riferimento BASIC).

Come dichiarare i tipi di variabile

Il nome della variabile determina il suo tipo (stringa o numerica ed, in questo, caso la sua precisione).

I nomi di variabili a stringa vengono scritti con un segno di dollaro alla fine. Ad esempio:

```
A$ = "RIEPILOGO VENDITE"
```

Il segno di dollaro è un carattere di dichiarazione del tipo di variabile. "Dichiara" che la variabile rappresenta una stringa.

I nomi di variabili numeriche possono dichiarare valori interi, a precisione singola o doppia. Sebbene la precisione dei calcoli con le variabili a precisione singola sia inferiore, vi sono dei vantaggi per utilizzare questo tipo di calcoli dal momento che:

- Richiedono meno spazio in memoria.
- Il tempo di elaborazione del calcolo è minore.

I caratteri di dichiarazione del tipo per le variabili numeriche ed il numero di byte necessario per memorizzare ogni tipo di valore sono:

% variabile intera (2 byte)

! variabile a precisione singola (4 byte)

variabile a precisione doppia (8 byte)

Nota: se il tipo di variabile non viene dichiarato esplicitamente, si assume la precisione singola.

Esempi:

PI#	dichiara un valore a precisione doppia
MINIMUM!	dichiara un valore a precisione singola
LIMIT%	dichiara un valore intero
N$	dichiara un valore di stringa
ABC	rappresenta un valore a precisione singola

È possibile dichiarare i tipi di variabile anche in altri modi. È possibile includere in un programma le specifiche BASIC DEFINT, DEFSNG, DEFDBL e DEFSTR per dichiarare i tipi di determinati nomi di variabili. Vedere la parte Specifiche DEFtipo nel Manuale di riferimento BASIC. Tutti gli esempi che seguono in questo manuale assumono che non siano state eseguite queste dichiarazioni, a meno che nell'esempio non siano illustrate esplicitamente le specifiche.

Schiere

Una *schiera* è una lista o una tabella di valori a cui si fa riferimento con un nome. Ogni singolo valore della schiera viene chiamato *elemento*. Gli elementi di schiera sono variabili e possono essere usati nelle espressioni e in qualsiasi specifica o funzione BASIC che usa le variabili.

Il *subscritto* (il numero fra parentesi) indica la posizione dell'elemento nella schiera. Zero rappresenta la posizione minima, a meno che venga esplicitamente modificata (vedere la Specifica OPTION BASE nel Manuale di riferimento BASIC).

La dichiarazione del nome e del tipo di una schiera e l'impostazione dei numeri di elementi e la loro disposizione nella schiera viene denominata *denominazione (definizione)* oppure *dimensionamento* della schiera.

Il massimo numero di dimensione per una schiera è 255.

Se si utilizza un elemento di schiera prima di definire (dimensionare) la schiera stessa, si assume che essa venga dimensionata con un subscritto massimo di 10; solo i primi undici (quelli con indice da 0 a 10) elementi, quindi, verranno riconosciuti. Per definire una schiera si utilizza la specifica DIM.

Ad esempio:

```
DIM B$(5)
```

Questa specifica crea una schiera monodimensionale di variabili stringa B$ con un numero massimo di 6 elementi. La schiera B$ potrebbe essere considerata come una lista di stringhe di caratteri.

B$(0)
B$(1)
B$(2)
B$(3)
B$(4)
B$(5)

Segue un altro esempio:

```
DIM A(2,3)
```

Questo crea una schiera bidimensionale denominata A. Poiché il nome non ha un carattere di dichiarazione del tipo di variabile numerica si assume che la schiera sia composta di valori a precisione singola.

La schiera A potrebbe essere considerata come una tabella di righe e colonne come questa:

A(0,0)	A(0,1)	A(0,2)	A(0,3)
A(1,0)	A(1,1)	A(1,2)	A(1,3)
A(2,0)	A(2,1)	A(2,2)	A(2,3)

L'elemento nella seconda riga, prima colonna, è denominato A(1,Ø).

Un esempio di programma è:

```
10 DIM YEARS(3,4)
20 YEARS(2,3)=84
30 FOR ROW=0 TO 3
40 FOR COLUMN=0 TO 4
50 PRINT YEARS(ROW,COLUMN);
60 NEXT COLUMN
70 PRINT
80 NEXT ROW
RUN
 0  0  0  0  0
 0  0  0  0  0
 0  0  0 84  0
 0  0  0  0  0
```

In questo programma la riga 10 dimensiona una schiera con 20 elementi (4 righe e 5 colonne); la riga 20 assegna all'elemento di schiera nella posizione 2,3 il valore di 84. Le iterazioni concatenate nelle righe 30 - 80 stampano la schiera come una matrice 4 per 5.

> **Nota:** Una variabile scalare potrà avere la stessa denominazione di una variabile schiera, dal momento che A$ differisce da qualsiasi elemento della schiera A$ (n, ...).

Tecniche per la formattazione dell'emissione

Nei programmi BASIC è possibile usare particolari specifiche e funzioni che permettono di visualizzare i numeri nel formato e con l'accuratezza desiderati.

- Usare la DEFDBL per formattare l'emissione e definire costanti e variabili come numeri a doppia precisione. Esempio:

```
10 WIDTH 80
20 DEFDBL A
30 A=70#
40 PRINT A/100#,
50 A=A+1
60 IF A<100# GOTO 40

RUN

.7          .71         .72         .73         .74
.75         .76         .77         .78         .79
.8          .81         .82         .83         .84
.85         .86         .87         .88         .89
.9          .91         .92         .93         .94
.95         .96         .97         .98         .99
```

- Usare le specifiche PRINT USING e LPRINT USING per ottenere che i dati emessi dal programma vengano espressi in numeri decimali. Queste specifiche consentono di scegliere il formato voluto.

```
10 FOR I=4 to 5 STEP .1
20 PRINT USING "#.##  ";I;
30 NEXT

RUN

4.0  4.1  4.2  4.3  4.4  4.5  4.6  4.7  4.8  4.9  5.0
```

Note:

1 Evitare di usare nella stessa formula la singola e la doppia precisione insieme poiché questo può ridurre l'esattezza del calcolo.

2 Per una maggiore esattezza usare la doppia precisione.

Espressioni numeriche e operatori

Un'espressione numerica può essere semplicemente una costante numerica o una variabile. Può essere usata anche per combinare costanti e variabili usando operatori per produrre un valore numerico singolo.

Gli operatori numerici eseguono operazioni matematiche o logiche principalmente su valori numerici e talvolta su stringhe. Si fa riferimento ad essi come operatori "numerici" poiché producono un valore che è un numero. Gli operatori numerici BASIC possono essere divisi in queste categorie:

- Aritmetici
- Di relazione
- Logici
- Funzioni

Operatori aritmetici

Gli operatori aritmetici eseguono le normali operazioni di aritmetica, come la somma e la divisione. Sono, in ordine di precedenza:

Operatore	Operazione	Espressione di esempio
^	Esponente	X ^ Y
–	Negativo	–X
*	Moltiplicazione	X*Y
/	Divisione in virgola mobile	X/Y
\	Divisione per intero	X \ Y
MOD	Aritmetica modulare	X MOD Y
+	Somma,	X+Y
–	Sottrazione	X–Y

(Se si possiede una conoscenza aritmetica, si noterà che questo è l'ordine standard di precedenza). Benché la maggior parte di queste operazioni possa sembrare familiare, due di esse non lo sono, cioè la divisione per intero e l'aritmetica modulare.

Divisione per intero: è indicata da una barra al contrario (\ ). Gli operandi vengono arrotondati agli interi (da –32768 a 32767) prima che venga eseguita la divisione; il quoziente è troncato all'intero.

Ad esempio:

```
10 A = 10\4
20 B = 25.68\6.99
30 PRINT A;B
RUN
 2  3
```

Aritmetica modulare: è indicata dall'operatore MOD. Fornisce il valore intero che è il resto di una divisione per intero.

Ad esempio:

```
10 A = 7 MOD  4
20 PRINT A
RUN
 3
```

Si ha questo risultato perché 7/4 dà 1 con il resto di 3.

```
PRINT 25.68 MOD 6.99
 5
```

Il risultato è 5 poiché 26/7 dà 3 con il resto di 5 (ricordare che il BASIC arrotonda quando converte in interi).

Operatori di relazione

Gli operatori di relazione mettono a confronto due valori. I valori possono essere entrambi numerici o stringhe. Il risultato del confronto può essere "vero" (-1) o "falso" (Ø). Si usa generalmente questo risultato per prendere una decisione riguardo il flusso di programma (vedere la Specifica IF nel Manuale di riferimento BASIC).

Operatore	Relazione provata	Espressioni di esempio
=	Uguaglianza	X=Y
<>o><	Disuguaglianza	X<>Y X><Y
<	Minore di	X<Y
>	Maggiore di	X>Y
<=o=<	Minore di o uguale a	X<=Y X=<Y
>=o=>	Maggiore di o uguale a	X>=Y X=>Y

(si usa il segno uguale anche per assegnare un valore ad una variabile. Vedere la Specifica LET nel Manuale di riferimento BASIC).

Confronti numerici: quando si combinano gli operatori aritmetici con quelli di relazione in un'espressione, i primi ad essere eseguiti sono quelli aritmetici. Ad esempio, l'espressione:

è vera (-1) se il valore di X più Y è inferiore al valore di T-1 diviso Z.

Confronti fra stringhe: possono essere considerati "alfabetici". Cioè, una stringa è "minore" di un'altra se la sua prima lettera si trova in ordine alfabetico prima della corrispondente lettera dell'altra stringa. Le lettere minuscole sono "maggiori" delle maiuscole. I numeri sono "minori" delle lettere.

Il modo per confrontare le due stringhe è quello di considerare un carattere di ogni stringa per volta e confrontarlo con i codici ASCII (vedere l'Appendice D del Manuale di riferimento BASIC per una lista completa dei codici ASCII). Se tutti i codici ASCII sono uguali, le stringhe sono uguali. In caso contrario, la stringa con il numero di codice inferiore è minore di quella con il numero di codice superiore. Se, durante il confronto tra due stringhe, si raggiunge la fine di una stringa, quella più corta viene considerata la più piccola.

Gli spazi iniziali e finali sono significativi. Ad esempio, tutte le espressioni di relazione seguenti sono vere (cioè, il risultato dell'operazione di relazione è –1).

```
"AA" < "AB"
"FILENAME" = "FILENAME"
"X$" > "X#"
"kg" > "KG"
"RICH" < "RICHB"
B$ < "718"              (dove B$ ="12543")
```

Tutte le costanti della stringa usate nell'espressione di confronto devono essere racchiuse tra apici.

Operatori logici

Gli operatori logici eseguono operazioni logiche o booleane sui valori numerici. Come per gli operatori di relazione che vengono usati solitamente per prendere decisioni sul flusso di programma, si usano gli operatori logici per collegare due o più relazioni e fornire un valore vero o falso da utilizzare per una decisione (vedere la Specifica IF nel Manuale di riferimento BASIC).

Un operatore logico esamina una combinazione di valori veri-falsi e fornisce un risultato vero o falso. Un operando di un operatore logico viene considerato "vero" se non è uguale a zero (come –1 fornito da un operatore di relazione), oppure "falso" se è uguale a zero. Il numero viene calcolato eseguendo l'operazione bit per bit.

Gli operatori logici sono NOT (complemento logico), AND (congiunzione), OR (disgiunzione), XOR (o esclusivo), IMP (implicazione) e EQV (equivalenza). Ciascun operatore fornisce i risultati come indicato nella seguente tabella (T indica un valore vero, o non-zero; F un valore falso, o zero). Gli operatori vengono elencati in ordine di precedenza.

NOT

X	*NOT X*
T	F
F	T

AND

X	*Y*	*X AND Y*
T	T	T
T	F	F
F	T	F
F	F	F

OR

X	*Y*	*X OR Y*
T	T	T
T	F	T
F	T	T
F	F	F

XOR

X	*Y*	*X XOR Y*
T	T	F
T	F	T
F	T	T
F	F	F

EQV

X	*Y*	*X EQV Y*
T	T	T
T	F	F
F	T	F
F	F	T

IMP

X	*Y*	*X IMP Y*
T	T	T
T	F	F
F	T	T
F	F	T

Alcuni esempi su come utilizzare gli operatori logici nelle decisioni:

```
IF HE>60 AND SHE<20 THEN 1000
```

Qui il risultato è vero se il valore della variabile HE è maggiore di 60 ed il valore di SHE è minore di 20.

```
50 IF NOT (P=-1) THEN 100
```

Il programa salta alla riga 100 se P non è uguale a –1. Notare che NOT (P=–1) non produce lo stesso risultato di NOT P. (Vedere la parte seguente, Come lavorano gli operatori logici).

```
100 FLAG% = NOT FLAG%
```

Questo esempio modifica il valore da vero a falso e viceversa.

Come lavorano gli operatori logici: gli operandi vengono convertiti in interi nella gamma tra –32768 e 32767 (se gli operandi non si trovano in questa gamma si verifica un errore di **OVERFLOW**). Se l'operando è negativo, si usa la forma del complemento a due. Ciascun operando viene modificato in una sequenza di 16 bit. L'operazione viene eseguita su queste sequenze; cioè, ogni bit del risultato è determinato dai corrispondenti bit dei due operandi, in base alle tabelle per l'operatore listate in precedenza. Il bit 1 viene considerato "vero" e il bit 0 "falso".

Si possono usare gli operatori logici per verificare una particolare struttura di bit. È possibile ad esempio usare l'operatore AND per creare una maschera per tutti i bit ad eccezione di quello di stato in una porta I/E.

I seguenti esempi dimostrano come lavorano gli operatori logici.

```
A = 63 AND 16
```

Qui A è impostato a 16. Poiché 63 è, in formato binario, 111111 e 16 è, in formato binario, 10000, 63 AND 16 è uguale, in formato binario, a 010000 che è uguale a 16.

```
B = -1 AND 8
```

B è impostato a 8. Poiché –1 è in formato binario 11111111 11111111 e 8 è in formato binario 1000, –1 AND 8 è uguale in formato binario a 0000000000001000, oppure 8.

```
C = 4 OR 2
```

In questo caso, C è uguale a 6. Poiché 4 è in binario 100 e 2 è in binario 010, 4 OR 2 è in binario 110, che è uguale a 6.

```
X = 2
TWOSCOMP = (NOT X) + 1
```

Questo esempio mostra come formare il complemento a due di un numero. X è 2, che è 10 in formato binario. NOTX è in binario 11111111 11111101, che è –3, in decimale; –3 più 1 è –2, complemento a 2; il complemento a due di un intero è, quindi, il complemento del bit più uno.

Notare che se entrambi gli operandi sono uguali a 0 oppure a – 1, un operatore logico fornirà 0, oppure –1.

Funzioni numeriche

Si usa una funzione come una variabile in un'espressione per richiamare un'operazione predeterminata che deve essere eseguita su uno o più operandi. Il BASIC ha delle funzioni numeriche incorporate che risiedono nel sistema, come la radice quadrata (SQR) od il seno (SIN).

È possibile definire le proprie funzioni usando la specifica DEF FN. (Vedere la parte riguardante DEF FN nel Manuale di riferimento BASIC)

Ordine di esecuzione

Nella sezione precedente le categorie delle operazioni numeriche sono state trattate in ordine di precedenza e la precedenza di ogni operazione all'interno di una categoria è stata indicata nella trattazione della categoria stessa. In breve:

1. Vengono valutati prima i richiami di funzioni.

2. Vengono poi eseguite le operazioni aritmetiche in questo ordine:

   a. ^
   b. un operando
   c. *,/
   d. \
   e. MOD
   f. +,-

3. Vengono poi, eseguite, le operazioni di relazione.

4. Le operazioni logiche vengono eseguite per ultime, in questo ordine:

   a. NOT
   b. AND
   c. OR
   d. XOR
   e. EQV
   f. IMP

Le operazioni allo stesso livello nella lista vengono eseguite in ordine da sinistra a destra. Per cambiare l'ordine di esecuzione delle operazioni, usare le parentesi. Le operazioni tra parentesi vengono eseguite per prime. Tra parentesi viene mantenuto l'ordine abituale delle operazioni.

Seguono alcuni esempi di espressioni algebriche e le relative controparti BASIC.

Espressione algebrica	Espressione BASIC
$X+2Y$	X+Y*2
$X-\frac{Y}{Z}$	X-Y/Z
$\frac{XY}{Z}$	X*Y/Z
$\frac{X+Y}{Z}$	(X+Y)/Z
$(X^2)^Y$	(X^2)^Y
X^{Y^Z}	X^(Y^Z)
$X(-Y)$	X*(-Y)

Nota: due operatori consecutivi devono essere separati da parentesi, come mostrato nell'esempio X*(-Y).

Espressioni e operatori di stringa

Un'espressione di stringa può essere semplicemente una costante o variabile di stringa, oppure può combinare costanti e variabili utilizzando operatori per produrre un valore di stringa singolo.

Si usano gli operatori di stringa per disporre le stringhe di caratteri in modi diversi. Le due categorie di operatori di stringa sono:

- Concatenazione
- Funzione

Notare che sebbene si possano usare gli operatori di relazione =, <>, <, >, <=, e >=, per confrontare due stringhe, questi non sono considerati "operatori di stringa" perché producono un risultato numerico, non una stringa. Vedere la parte "Operatori di relazione" in questo capitolo, per una spiegazione su come confrontare le stringhe usando gli operatori di relazione.

Concatenazione

L'unione di due stringhe è denominata *concatenazione*. Le stringhe vengono concatenate usando il simbolo più (+). Ad esempio:

```
10 COMPANY$ = "IBM"
20 TYPE$ = " Personal"
30 FULLNAME$ = TYPE$ + " Computer"
40 PRINT COMPANY$+FULLNAME$
RUN
IBM Personal Computer
```

Funzioni di stringa

Una funzione di stringa è simile ad una funzione numerica, eccetto nel fatto che fornisce un risultato a stringa. È possibile utilizzare una funzione di stringa in un'espressione per richiamare un'operazione predeterminata che deve essere eseguita su uno o più operandi. Il BASIC possiede delle funzioni integrate di stringa che risiedono nel sistema, come ad esempio MID$, che fornisce una stringa dalla metà di un'altra stringa, oppure CHR$, che fornisce il carattere con il codice ASCII specificato. È possibile definire le proprie funzioni di stringa usando la specifica DEF FN. Vedere la Specifica DEF FN nel Manuale di riferimento BASIC.

Immissione ed emissione

Il seguito di questo capitolo contiene informazioni sulla Immissione ed Emissione (I/E) in BASIC. Verranno trattati i seguenti argomenti: denominazione e utilizzo di file e collegamenti, utilizzo di dispositivi di I/E, indirizzari strutturati ad albero, altri dispositivi e visualizzazione dell'emissione su schermo video, sotto forma di testo o grafici.

File

Un file è una raccolta di informazioni tenute ovunque, ma non nella memoria ad accesso casuale del Personal Computer IBM (ad esempio, memorizzate in un file su minidisco o cassetta). Per accedere alle informazioni, è necessario *aprire* il file (con la specifica OPEN) per indicare al BASIC dove si trovano le informazioni. In seguito è possibile utilizzare il file per l'immissione e/o l'emissione.

Il BASIC supporta il concetto di generico file di I/E. Ciò significa che qualsiasi tipo di immissione/emissione può essere trattato come I/E su un file, sia che si utilizzi un file su cassetta o su minidisco, oppure che si stia comunicando con un altro elaboratore.

Numero del file

Il BASIC esegue operazioni di I/E utilizzando un numero di file. Ad un file o ad un dispositivo viene assegnato un numero quando se ne effettua l'apertura con una specifica OPEN (vedere la Specifica OPEN nel Manuale di riferimento BASIC). Il numero del file può essere rappresentato da un numero qualsiasi, da una variabile o da un'espressione compresa fra 1 ed il numero massimo di file apribili contemporaneamente. Vedere il parametro **/F** nella sezione "Parametri del comando BASIC" al capitolo 2.

Nome del file

Il nome del file dovrà seguire determinate regole:

Nel caso di un file su cassetta:

- Il nome non dovrà superare 8 caratteri.
- Il nome non potrà contenere due punti (:), ØØ esadecimali, oppure FF esadecimali (decimale 255).

Nel caso di file su disco il nome dovrà seguire le regole del DOS:

- Potrà essere costituito da 2 parti separate da un punto (.)

```
nome.estensione
```

  Il *nome* utilizzerà da 1 ad 8 caratteri; *l'estensione* da 1 a 3 caratteri.

  Se per *l'estensione* vengono immessi più di 3 caratteri, i caratteri in più vengono troncati. Se il nome supera gli 8 caratteri e non è inclusa l'estensione, il BASIC inserisce un punto dopo gli 8 caratteri ed utilizza i caratteri in eccesso (fino a 3) per l'estensione. Se il *nome* supera 8 caratteri e si include l'estensione, si verifica un errore.

- Nel *nome* e nell'*estensione* è consentito specificare i seguenti caratteri:

  da A a Z
  da Ø a 9
  ( ) { }
  @ # $ % ^ & !
  - _ ` ' / ~

  Nota: nelle precedenti versioni BASIC era consentito l'utilizzo dei caratteri < > , e \ per il nome del file, ma dal momento che questi caratteri hanno un significato speciale per il BASIC 2.Ø e le versioni successive, non potranno essere utilizzati, quindi, per il nome del file.

Esempi di nomi di file:

27 HAL.DAD
VLD
PROGRAM 1.BAS
$$@(!).123

Il BASIC, poi, abbrevia nomi ed estensioni troppo lunghi nel modo seguente:

A23456789JKLMN diventa A2345678.9JK
@HOME.TRUM 10 diventa @ HOME.TRU
SHERRYLYNN.BAS determina un errore.

Nome dell'unità:

Il nome di unità consiste al massimo di quattro caratteri alfanumerici, seguiti da due punti (:). Segue una lista completa di nomi di unità, con l'indicazione del tipo di unità a cui si riferiscono.

KYBD: Tastiera. Solo immissione, tutte le versioni di BASIC.

SCRN: Video. Solo emissione, tutte le versioni di BASIC.

LPT1: Prima stampatrice. Emissione, per tutte le versioni, accesso casuale per BASIC su disco e avanzato.

LPT2: Seconda stampatrice. Emissione oppure accesso casuale, BASIC su disco e avanzato.

LPT3: Terza stampatrice. Emissione oppure accesso casuale, BASIC su disco e avanzato.

UNITA' DI COMUNICAZIONE

COM1: Primo adattatore per le comunicazioni asincrone. Immissione ed emissione, BASIC su disco e avanzato.

COM2: Secondo adattatore per le comunicazioni asincrone. Immissione ed emissione, BASIC su disco e avanzato.

UNITA' DI MEMORIA

CAS1: Registratore a cassette. Immissione ed emissione, tutte le versioni.

A: Prima unità a minidischi. Immissione ed emissione, BASIC su disco e avanzato.

B: Seconda unità a minidischi. Immissione ed emissione, BASIC su disco e avanzato.

C: Prima unità disco fisso. Immissione e emissione, BASIC su disco e avanzato.

D: Seconda unità disco fisso. Immissione e emissione, BASIC su disco e avanzato.

Nota: Se si dispone di tre unità minidisco, il terzo modulo è il C:, il primo disco è il D: e il secondo disco fisso è E:. Analogamente, se si dispone di quattro unità minidisco la quarta unità minidisco è il D:, il primo disco fisso è E:, il secondo disco fisso è F:.

Denominazione dei file

Il file fisico è descritto dalla *specifica del file.*

La specifica del file è una espressione a stringa nella forma:

```
[unità] [percorso][nomefile]
```

Il nome dell'unità indica al BASIC l'unità di I/E da utilizzare. Il percorso segnala al BASIC quale indirizzario contiene il file, mentre il nome del file indica al BASIC quale file cercare su una particolare unità. Talvolta non è necessario specificare entrambi i nomi, dell'unità e del percorso, quindi la specifica dei due nomi è facoltativa. Tutti i nomi delle unità terminano con due punti (:). I due punti sono parte del nome dell'unità e debbono essere inclusi ogniqualvolta è specificata quell'unità.

> **Nota:** la specifica del file per le unità di comunicazione è leggermente diversa. Il nome del file viene sostituito da una lista di opzioni che specificano alcune cose, come la velocità della linea. Vedere la specifica "OPEN COM..." nel Manuale di riferimento BASIC.

Ricordare che se si usa una costante a stringa per la *specfile*, è necessario racchiuderla tra apici. Ad esempio:

```
LOAD "B:PROG.BAS"
```

Nel BASIC livello 2.0 (BASIC su disco e avanzato) e per i livelli successivi le modalità di definizione dei file sono state ampliate consentendo la definizione di percorsi.

Un percorso è una lista di nomi di indirizzario separati da \ .

Si può specificare il percorso per tutti i comandi seguenti:

BLOAD	CHDIR	LOAD	NAME	RUN
BSAVE	FILES	MERGE	OPEN	SAVE
CHAIN	KILL	MKDIR	RMDIR	

Note:

1. Un percorso non deve contenere più di 63 caratteri.

2. Prima del percorso occorrerà indicare l'unità, diversamente apparirà il messaggio di errore **Bad file name.**

3. Se per il percorso si usa una costante di tipo stringa, essa dovrà essere posta tra apici.

```
"B: \ SALES \ JUNE \ REPORT \ "
```

Se il file non si trova nell'indirizzario in uso, per localizzare il file occorre fornire al BASIC un percorso di nomi di indirizzario.

Tipi di indirizzari

Come nei precedenti livelli di BASIC un unico indirizzario viene creato su ogni disco quando si usa il comando DOS FORMAT; questo viene chiamato indirizzario radice. Il numero massimo di file in un indirizzario radice su minidisco dipende dal tipo di minidisco utilizzato. Il numero massimo di file che può contenere un indirizzario radice su disco fisso, dipende dalla grandezza della partizione DOS nel disco.

Oltre a contenere nomi di file, l'indirizzario radice può contenere nomi di altri indirizzari chiamati sottoindirizzari. Diversamente dall'indirizzario radice questi sottoindirizzari sono dei veri e propri file e possono contenere a loro volta un numero qualunque di file e sottoindirizzari, limitati solo dalla quantità di spazio disponibile sul disco.

I nomi di sottoindirizzari sono nello stesso formato dei nomi dei file. Gli stessi caratteri validi per gli uni sono validi anche per gli altri. Ogni indirizzario può quindi contenere nomi di file o di indirizzari che possono apparire anche in altri indirizzari.

Indirizzario corrente

Come il BASIC ricorda un dispositivo assunto, se questo viene omesso, così ricorda anche un indirizzario assunto per ogni unità presente sul sistema; questo viene chiamato indirizzario corrente ed è qui che il BASIC effettuerà la ricerca, qualora non gli venga comunicato il nome dell'indirizzario contenente il file. Si può cambiare l'indirizzario corrente usando il comando CHDIR (vedere Comando CHDIR nel Manuale di riferimento BASIC).

Se in un percorso viene incluso un nome di file, esso dovrà essere separato dal precedente nome di indirizzario, tramite una barra rovesciata (\ ). Se il parametro "percorso" inizia con questo simbolo, il BASIC inizia la ricerca dall'indirizzario radice; diversamente la ricerca inizia dall'indirizzario corrente (in uso).

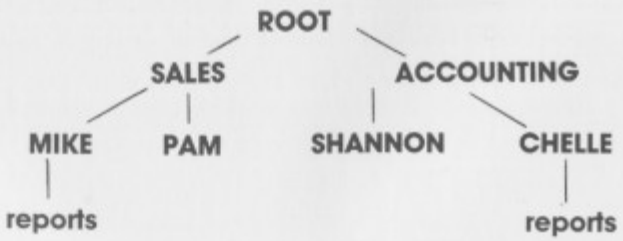
Indirizzari strutturati ad albero

Nei precedenti livelli di BASIC venivano usati indirizzari a struttura semplice sufficienti per gestire file su minidisco. Dal BASIC livello 2.Ø (e successivi) con le nuove unità a disco fisso un solo disco può contenere anche migliaia di file e quindi è necessario un metodo più sofisticato per gestirli.

Il BASIC livello 2.Ø (e successivi) dà la possibilità di gestire e organizzare meglio i dischi raggruppando e cancellando file nei propri indirizzari, siano essi su disco o minidisco.

Questi indirizzari vengono definiti "strutturati ad albero"; essi iniziano con l'indirizzario radice e contengono sottoindirizzari.

Se la società XYZ, ad esempio, ha due settori, Vendite e Contabilità che utilizzano lo stesso Personal Computer IBM, tutti i file della società sono contenuti sul disco fisso. L'organizzazione dei file per categoria potrebbe essere vista come la seguente.



Supporto unità

Il sistema di I/E del file consente l'utilizzo di unità installate dall'utente. (Vedere il Manuale di riferimento tecnico relativo al DOS 3.Ø per informazioni su tali unità).

Si può, ad esempio, scrivere il proprio programma di controllo unità per sostituire LPT1. Il programma aprirà il programma di controllo come segue:

```
OPEN "LPT1" FOR mode AS#filenum
```

Nella compilazione di un programma di controllo unità occorre seguire alcune norme:

- Il nome dell'unità installata non può terminare con il segno di due punti (:); i nomi che terminano con tale simbolo sono riservati a determinate unità (KEYBD:, SCRN:, etc).

- LPT1:, LPT2: ed LPT3: sono gli unici programmi controllo unità sostituibili da un programma controllo dispositivo installato con lo stesso nome.

- La lunghezza record è impostata ad 1 a meno che non sia stata modificata dalla specifica OPEN.

  ```
  OPEN filespec [FOR MODE] AS filenum LEN=recl
  ```

  Se si usa questa opzione il BASIC memorizzerà nella memoria di transito un numero di caratteri corrispondenti a "recl" prima di inviarli al programma guida.

- il BASIC invia solo un ritorno a capo (&HØD) a fine riga, se si rende necessaria un'interlinea (&HØA), il programma guida deve fornirla.

- Il programma guida unità dovrà produrre una condizione **End of file** (Fine file) per il BASIC qualora si voglia chiudere un file di immissione sequenziale aperto per un programma guida unità.
  Se il BASIC tenta la lettura dopo il termine del flusso di immissione per l'unità, il programma guida produrrà una condizione ^Z (Control-Z) utilizzata dal BASIC per determinare un errore **Input past end.**

Il BASIC può visualizzare testo, caratteri speciali, punti, linee o forme più complesse a colori o in bianco e nero. Le possibilità di visualizzazione dipendono dal tipo di adattatore per il video installato sul Personal Computer IBM.

Adattatori per il video

Il Personal Computer IBM ha due adattatori per il video: l'Adattatore per il video monocromatico IBM e stampatrice parallela e l'Adattatore di controllo per il video grafico a colori.

Con l'Adattatore per il video monocromatico IBM e stampatrice parallela, è possibile visualizzare il testo in bianco e nero. Il testo si riferisce a lettere, numeri e tutti i caratteri speciali del gruppo normale di caratteri. È possibile disegnare figure con i caratteri speciali di riga e blocco. Si possono creare anche caratteri con lampeggiamento, in negativo, invisibili, con contrasto immagine evidenziati e sottolineati ad alta intensità impostando i parametri nella Specifica COLOR.

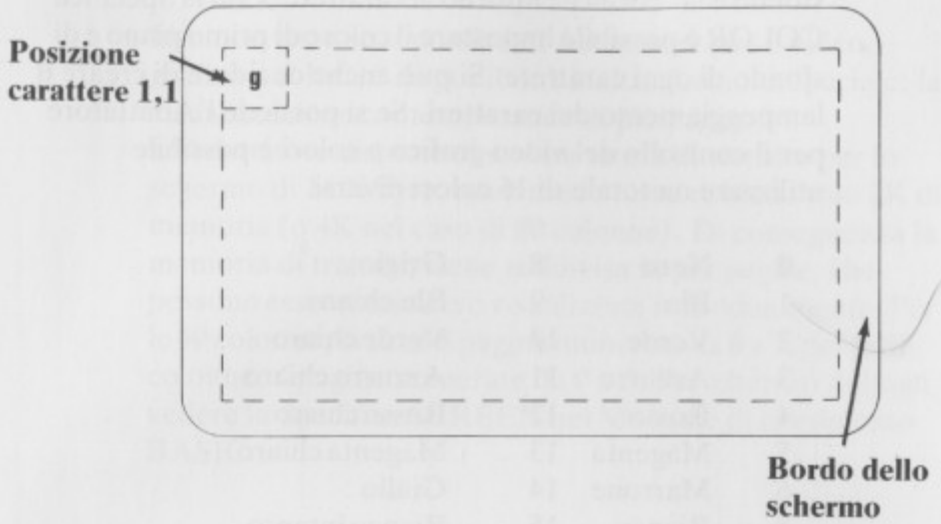
Anche l'Adattatore di controllo del video grafico a colori opera in modo testo, ma consente di visualizzare il testo in 16 colori diversi (è possibile visualizzarlo anche in bianco e nero impostando i parametri nella Specifica SCREEN o COLOR). È anche possibile eseguire disegni complessi tramite la possibilità grafica, che rende *indirizzabili tutti i punti* dello schermo a risoluzione media o alta. Questo modo è più versatile di quello che consente di eseguire disegni con i caratteri speciali di riga e blocco che si hanno in modo testo.

In questo manuale e nel Manuale di riferimento il termine "GRAFICI" è applicabile solo a questa particolare possibilità dell'Adattatore per il controllo del video grafico a colori. L'utilizzo della serie estesa di caratteri (con i

caratteri speciali di riga e blocco) non è da considerarsi relativo ai GRAFICI.

Modo testo

Lo schermo può essere raffigurato in questo modo:



I caratteri vengono visualizzati su 25 righe orizzontali, queste righe sono numerate da 1 a 25, dall'alto in basso. Ogni riga contiene 40 posizioni carattere (oppure 80, se si è impostata un'ampiezza diversa). Le posizioni sono numerate da 1 a 40 (o 80) da sinistra a destra. I numeri di posizione vengono utilizzati dalla specifica LOCATE e sono i valori riportati dalle funzioni POS (0) e CSRLIN. Il carattere nell'angolo superiore sinistro dello schermo ad esempio è sulla riga 1, posizione 1.

Normalmente i caratteri vengono posti sullo schermo con la specifica PRINT. I caratteri vengono visualizzati nella posizione del cursore da sinistra a destra di ogni riga, da riga 1 a riga 24. Quando il cursore passa alla riga 25 dello schermo, le righe 1-24 *scorrono* di una riga in senso ascendente in modo che la prima scompaia. La riga 24 è in bianco ed il cursore vi rimane per continuare la stampa.

La riga 25 viene solitamente usata per i "tasti di programma" (vedere la Specifica KEY nel Manuale di riferimento BASIC), ma è possibile scrivere in questa area

dello schermo se si disattiva la visualizzazione dei "tasti di programma". Il BASIC non esegue mai lo scorrimento della venticinquesima riga.

Ogni carattere sullo schermo è composto di due parti: primo piano e sfondo. Il primo piano è il carattere stesso, lo sfondo è la "cornice" attorno al carattere. Con la Specifica COLOR è possibile impostare il colore di primo piano e di sfondo di ogni carattere. Si può anche decidere di creare il lampeggiamento dei caratteri. Se si possiede l'Adattatore per il controllo del video grafico a colori è possibile utilizzare un totale di 16 colori diversi.

0	Nero	8	Grigio
1	Blu	9	Blu chiaro
2	Verde	10	Verde chiaro
3	Azzurro	11	Azzurro chiaro
4	Rosso	12	Rosso chiaro
5	Magenta	13	Magenta chiaro
6	Marrone	14	Giallo
7	Bianco	15	Bianco intenso

I colori possono variare in base alla particolare unità video.

La regolazione del colore sull'unità può consentire di ottenere dei colori il più possibile corrispondenti a quelli della tabella.
La maggior parte degli apparecchi televisivi e dei video dispone di un'area di "extra scansione" al di fuori dell'area usata per i caratteri. Questa area di extra scansione è denominata *bordo dello schermo*. Con la specifica COLOR è possibile impostare il colore per il bordo dello schermo.
Per visualizzare le informazioni in modo testo si usano le seguenti specifiche:

CLS	SCREEN
COLOR	WIDTH
LOCATE	WRITE
PRINT	

In modo testo possono essere usate le seguenti funzioni e variabili di sistema:

CSRLIN SPC
POS TAB
SCREEN

In modo testo se si possiede l'Adattatore per il video grafico a colori, è disponibile un'altra funzione speciale: la visualizzazione contemporanea di più pagine.
L'Adattatore dispone di una memoria di transito per lo schermo di 16 K byte, ma il modo testo richiede solo 2K di memoria (o 4K nel caso di 80 colonne). Di conseguenza la memoria di transito viene suddivisa in più *pagine,* che possono essere scritte e/o visualizzate individualmente. Per le 40 colonne, vi sono 8 pagine numerate da 0 a 7; per le 80 colonne, 4 pagine numerate da 0 a 3. Per ulteriori dettagli vedere la Specifica SCREEN nel Manuale di riferimento BASIC.

Modo grafici

I modi grafici sono disponibili solo se si possiede l'Adattatore per il controllo del video grafico a colori.

Si possono utilizzare le specifiche BASIC per eseguire i disegni in due soluzioni grafiche:

- Media: 320 per 200 punti e in 4 colori.
- Alta: 640 per 200 punti e in 2 colori.

Si può scegliere la soluzione desiderata con la Specifica SCREEN.

Le specifiche usate nel BASIC per i grafici sono:

CIRCLE	PRESENT
COLOR	PSET
DRAW	PUT
GET	SCREEN
LINE	VIEW
PAINT	WINDOW

Le funzioni grafiche sono:

PMAP
POINT

Attributi per il colore: esiste un valore numerico che descrive il colore di ciascun punto sullo schermo; questo valore viene chiamato **attributo.** Se per la specifica COLOR, ad esempio, si fa riferimento (nel manuale BASIC), alla lista dei colori disponibili (nel modo testo), si noterà che il colore blu è contraddistinto dall'attributo 1.

L'attributo varia a seconda del modo operativo sullo schermo, le variazioni sono basate sull'estensione dell'attributo per ciascun modo. L'attributo massimo per ciascun modo determina i bit necessari per la definizione dell'attributo di un punto — il numero di bit per pixel.

Consideriamo, ad esempio, l'estensione attributo (Ø-3) per lo schermo 1. In binario sono necessari 2 bit per rappresentare un numero decimale equivalente a 3. Nel caso di visualizzazione con attributo compresa fra Ø ed 1, occorrerà solo 1 bit per rappresentare questi valori. Da un punto di vista matematico il numero di bit per pixel corrisponde ad un logaritmo in base 2 dell'attributo massimo disponibile per lo schermo. Questo concetto è particolarmente importante nell'utilizzo del colore di sfondo. Vedere la Specifica PAINT nel Manuale di riferimento BASIC.

Media risoluzione: vi sono 32Ø punti orizzontali e 2ØØ verticali. Questi punti sono numerati da sinistra a destra e dall'alto verso il basso, a partire da zero. Ciò significa che

l'angolo superiore sinistro dello schermo è il punto (Ø,Ø) e quello inferiore destro è (319,199) (se si conosce il metodo matematico usuale per la numerazione delle coordinate, questo può sembrare contrario).
La media risoluzione è determinata da una specifica SCREEN 1.
La media risoluzione è insolita per la difficoltà di gestione del colore. Se si porta a video qualcosa a media risoluzione, si può specificare un numero di colore Ø, 1, 2 o 3. Questi colori non sono fissi, come i 16 colori del modo testo. Si sceglie il colore reale per l'attributo del colore Ø e poi si sceglie una delle due "tavolozze" per gli altri tre colori usando la specifica COLOR. Una tavolozza è un gruppo di tre colori da associare ai numeri 1,2 e 3. Se si modifica la tavolozza con una specifica COLOR, cambiano tutti i colori sullo schermo in modo da corrispondere alla nuova tavolozza.

Anche in modo "grafici" è possibile visualizzare i caratteri di testo. Le dimensioni del carattere saranno uguali a quelle del modo testo, cioè, 25 righe di 40 caratteri a media risoluzione, il primo piano avrà il colore numero 3 e lo sfondo il colore numero Ø.

Alta risoluzione: vi sono 64Ø punti orizzontali e 2ØØ verticali. Come per la media risoluzione, questi punti sono numerati a partire da zero, in modo che l'angolo inferiore destro sia (639,199).

L'alta risoluzione è determinata dalla specifica SCREEN2.

L'alta risoluzione è un po' più facile da descrivere rispetto a quella media, poiché vi sono solo due colori: bianco e nero. Il nero è sempre zero (Ø) ed il bianco è sempre (1).

Se si visualizzano caratteri di testo in alta risoluzione, si ottengono 8Ø caratteri per riga. Il colore in primo piano è 1 e quello di sfondo è Ø, ottenendo così caratteri bianchi su fondo nero.

Come specificare le coordinate: le specifiche grafiche richiedono informazioni su dove si desidera eseguire il disegno sullo schermo. Si danno queste informazioni sotto forma di coordinate: queste hanno generalmente il seguente formato *(x,y)* in cui *x* rappresenta la posizione orizzontale e *y* quella verticale. Questa forma è conosciuta come *forma assoluta* e fa riferimento alle reali coordinate del punto sullo schermo, senza considerare l'ultimo punto a cui si è fatto riferimento.

Vi è un altro modo per indicare le coordinate, conosciuto come *forma relativa*. Con questa forma si indica al BASIC l'ubicazione del punto relativo all'ultimo punto a cui si è fatto riferimento. Questa è la forma:

STEP (*xpos. relativa, ypos. relativa*)

Tra parentesi si indica la *posizione relativa* nelle direzioni orizzontale e verticale dall'ultimo punto a cui si è fatto riferimento. Inizialmente (dopo la specifica SCREEN 1, SCREEN 2, WIDTH o CLS) l'ultimo punto a cui viene fatto riferimento è quello al centro dello schermo: (160, 100) nel caso di risoluzione media oppure (320, 100) per l'alta risoluzione. In seguito le specifiche cambiano l'ultimo punto di riferimento. L'impostazione dell'ultimo punto di riferimento dei "grafici" viene specificata nel Manuale di riferimento BASIC.

> **Nota:** attenzione a non disegnare oltre i limiti dello schermo con una specifica dei grafici, poiché potrebbe confondersi l'ultimo punto a cui si è fatto riferimento.

Questo esempio illustra l'utilizzo di entrambe le forme di coordinate:

```
100 SCREEN 1
110 PSET (200,100) 'forma assoluta
120 PSET STEP (10,-20) 'forma relativa
```

Questo imposta due punti sullo schermo. Le loro coordinate sono (200, 100) e (210, 80).

Appendici

Indice

Appendice A. Immissione ed emissione su disco nel BASIC

Questa appendice descrive procedure e considerazioni speciali per l'uso dell'immissione ed emissione su disco. Contiene liste dei comandi e delle specifiche utilizzati con i file su disco e spiegazioni su come usarli. L'appendice comprende anche molti programmi di esempio che aiutano a chiarire l'uso dei file di dati su disco. Se non si è esperti di BASIC o se si verificano degli errori relativi al disco, leggere queste procedure e gli esempi di programma per assicurarsi di usare tutte le specifiche, relative al disco, correttamente.

Per altre informazioni su dischi e relativi file, consultare il Manuale di riferimento.

> **Nota:** la maggior parte delle informazioni di questa appendice sui file di programma e sequenziali è valida anche per l'I/E su cassetta. Non è possibile tuttavia aprire la cassetta in modo casuale.

Come specificare i nomi dei file

I nomi dei file su disco devono attenersi alle convenzioni per la denominazione del DOS in modo che il BASIC possa leggerli. Per maggiori informazioni in proposito vedere il Capitolo 3.

Comandi per file di programma

Questa è una lista, con breve descrizione, dei comandi che si possono utilizzare con i file di programma BASIC. Per maggiori informazioni sui comandi consultare il Manuale di riferimento BASIC.

CHAIN	NAME
KILL	RUN
LOAD	SAVE
MERGE	

File di dati su minidisco — I/E sequenziale e casuale

È possibile creare due tipi di file di dati su disco ai quali può accedere un programma BASIC: file sequenziali e ad accesso casuale.

File sequenziali

I file sequenziali sono più facili da creare di quelli ad accesso casuale, ma presentano una flessibilità e velocità limitate nell'accesso ai dati. I dati scritti su di un file sequenziale vengono memorizzati sequenzialmente, un elemento dopo l'altro, nell'ordine d'invio. Ogni elemento viene letto nello stesso modo, dal primo nel file fino all'ultimo.

Le specifiche e le funzioni usate con i file sequenziali sono:

CLOSE	LOF
EOF	OPEN
INPUT #	PRINT #
INPUT$	PRINT # USING
LINE INPUT #	WRITE #
LOC	

Come creare ed accedere ad un file sequenziale

Per creare un file sequenziale ed accedere ai suoi dati, si specificano nel programma i passi che seguono:

1. Aprire il file per l'emissione o l'aggiunta con la specifica OPEN

2. Scrivere i dati nel file usando le specifiche PRINT # WRITE #, PRINT # USING.

3. Per accedere ai dati nel file, chiudere il file (con CLOSE) e riaprirlo per l'immissione (OPEN).

4. Usare le specifiche INPUT # oppure LINE INPUT # per leggere i dati da un file sequenziale nel programma.

Le seguenti sono righe di un programma 1 esemplificativo che dimostrano questi passi:

```
10 REM PROGRAM1 - SEQUENTIAL FILES
20 OPEN "DATA" FOR OUTPUT AS #1 'PASSO 1
30 FOR I=1 TO 1040 PRINT#1,I 'PASSO 2
50 NEXT
60 CLOSE 'PASSO 3
70 OPEN "I", 1,"DATA"
80 IF EOF(1) THEN CLOSE:END
90 INPUT#1,A 'PASSO 4
100 PRINT A
110 GOTO 80
```

Notare entrambe le indicazioni della specifica OPEN (riga 20 e riga 70). Consultare il Manuale di riferimento BASIC per ulteriori dettagli sulla specifica OPEN.

Nella riga 80 la funzione EOF verifica la fine file, evitando un errore per "Immissione dopo la fine". La fine del file viene indicata da un carattere speciale nel file; questo carattere ha il codice ASCII 26 (esadecimale 1A). In un file sequenziale, quindi, non dovrà essere indicato CHR$ (26).

Un programma che crea un file sequenziale può anche scrivere dati organizzati sul disco con la specifica PRINT # USING. Ad esempio, la specifica:

```
PRINT# 1,USING "###.##.";A,B,C,D
```

può essere usata per scrivere dati numerici sul disco senza delimitatori espliciti. Lo spazio alla fine della stringa di formato viene usato per separare gli elementi nel file su disco.

La funzione LOC, se utilizzata con un file sequenziale, ritorna il numero di record che sono stati scritti o letti dal file dopo l'apertura (un record è un blocco di dati di 128 byte).

La funzione LOF ritorna il numero di byte allocati al file. Nel caso di file creati con il BASIC versione 1.10, LOF riporterà un numero di byte multiplo di 128 (arrotondato per eccesso, se necessario). Per i file creati con EDLIN, ad esempio, o con il BASIC 2.0 e rilasci successivi, LOF riporta il numero effettivo di byte allocati.

Come aggiungere dati ad un file sequenziale

Nel caso di un file sequenziale su disco al quale si vogliano aggiungere altri dati alla fine, non è possibile aprire semplicemente il file per l'emissione ed iniziare la scrittura dei dati. Non appena si apre un file sequenziale per l'emissione se ne distrugge il contenuto. Occorre, invece, aprire il file per l'aggiunta (APPEND). Vedere la Specifica OPEN nel Manuale di riferimento BASIC.

File ad accesso casuale

La creazione e l'accesso a file ad accesso casuale richiedono più passi di programma rispetto ai file sequenziali, ma esistono dei vantaggi nell'uso dei file ad accesso casuale. I numeri nei file ad accesso casuale, ad esempio, vengono generalmente memorizzati su disco in formato binario, mentre nei file sequenziali vengono memorizzati come caratteri ASCII. Perciò, in molti casi i file ad accesso casuale richiedono meno spazio su disco rispetto ai file sequenziali.

Il più grosso vantaggio con i file ad accesso casuale è quello di poter accedere ai dati in modo casuale cioè, ovunque su disco. Non è necessario leggere tutte le informazioni per intero come per i file sequenziali. Ciò è possibile dal momento che le informazioni vengono memorizzate ed esaminate in unità distinte, chiamate record ed ogni record risulta numerato.

I record possono essere lunghi fino a 32767 byte. L'ampiezza di un record non è collegata a quella di un settore su disco (512 byte). Il BASIC usa automaticamente tutti i 512 byte di un settore per memorizzare le informazioni. Effettua l'operazione bloccando i record ed espandendo i limiti del settore (parte di un record, cioè può trovarsi alla fine di un settore e l'altra parte all'inizio del settore successivo).

Le specifiche e le funzioni usate con i file ad accesso casuale sono:

CLOSE	OPEN
FIELD	PUT
GET	
LSET/RSET	
CVD	MKD$
CVI	MKI$
CVS	MKS$
LOC	
LOF	

Come creare un file ad accesso casuale

Per creare un file ad accesso casuale sono necessari i seguenti passi di programma.

1. Aprire il file per l'accesso casuale. L'esempio che illustra questi passi specifica una lunghezza record di 32 byte. Se si omette la lunghezza record, si assumono 128 byte.

2. Usare la specifica FIELD per allocare spazio nella memoria di transito casuale per le variabili che saranno scritte nel file ad accesso casuale.

3. Usare LSET o RSET per spostare i dati nella memoria di transito casuale. I valori numerici devono essere tramutati in stringhe quando vengono posti nella memoria di transito. Per questa operazione si usano le funzioni MKI$ per tramutare un valore intero in una stringa, MKS$ per un valore a precisione singola e MKD$ per un valore a precisione doppia.

4. Scrivere i dati dalla memoria di transito sul disco con la specifica PUT.

Queste fasi sono elencate in PROGRAM2.

Nota: Non utilizzare una variabile di stringa definita in una specifica FIELD di una specifica di immissione o sul lato sinistro di una specifica di assegnazione (LET).
Il puntatore della variabile punta, quindi, allo spazio della stringa anziché alla memoria di transito del file ad accesso casuale.

Esaminiamo il programma che segue. La specifica PUT viene eseguita, viene scritto un record sul file; il codice di 2 cifre che costituisce l'immissione, sulla riga 30, diventa il numero di record.

```
10 REM PROGRAM2 - CREARE UN FILE RANDOM
20 OPEN "DATAY" AS#1 LEN=32  'PASSO1
30 FIELD#1,20 AS N$, 4 AS A$, 8 AS P$  'PASSO 2
40 INPUT "2-DIGIT CODE";CODE%
50 IF CODE% =99 THEN CLOSE: END
60 INPUT "NAME";X$
70 INPUT "AMOUNT";AMT
80 INPUT "PHONE";TEL$: PRINT
90 LSET N$= X$ 'PASSO 3
100 LSET A$=MKS$(AMT)   'PASSO 3
110 LSET P$=TEL$  'PASSO 3
120 PUT#1,CODE% 'PASSO 4
130 GOTO 40
```

Come accedere ad un file ad accesso casuale

Per accedere ad un file ad accesso casuale sono necessari i seguenti passi di programma:

1. Aprire il file per l'accesso casuale.

2. Usare la specifica FIELD per allocare spazio nella memoria di transito casuale per le variabili che saranno lette dal file.

   Nota: in un programma che esegue sia l'immmissione che l'emissione sullo stesso file ad accesso casuale, è possibile solitamente usare solo una specifica OPEN ed una FIELD.

3. Usare la specifica GET per spostare il record desiderato nella memoria di transito casuale.

4. Il programma ora può accedere ai dati della memoria di transito. I valori numerici devono essere riconvertiti in numeri con le funzioni: CVI per gli interi, CVS per i valori a precisione singola, CVD per i valori a precisione doppia.

   Questi passi vengono indicati in PROGRAM 3.

Program 3 accede in maniera casuale al file creato da PROGRAM 2. Quando dalla tastiera viene immesso il codice di 2 cifre, le informazioni associate a quel codice vengono lette dal file e visualizzate.

```
10 REM PROGRAM3 - ACCESSO AD UN FILE RANDOM
20 OPEN "DATAY" AS#1 LEN=32   'PASSO 1
30 FIELD#1,20 AS N$, 4 AS A$, 8 AS P$ 'PASSO 2
40 INPUT "2-DIGIT CODE";CODE%
50 IF CODE%=99 THEN CLOSE: END
60 GET#1, CODE%   'PASSO 3
70 PRINT N$ 'PASSO 4
80 PRINT USING "$$###.##";CVS(A$)   'PASSO 4
90 PRINT P$
100 GOTO 40
```

La funzione LOC, con i file ad accesso casuale, ritorna l'attuale numero record, che è l'ultimo utilizzato in una specifica GET o PUT. Ad esempio la specifica:

```
IF LOC(1)>50 THEN END
```

termina l'esecuzione del programma se l'attuale numero record del file # 1 è superiore a 50.

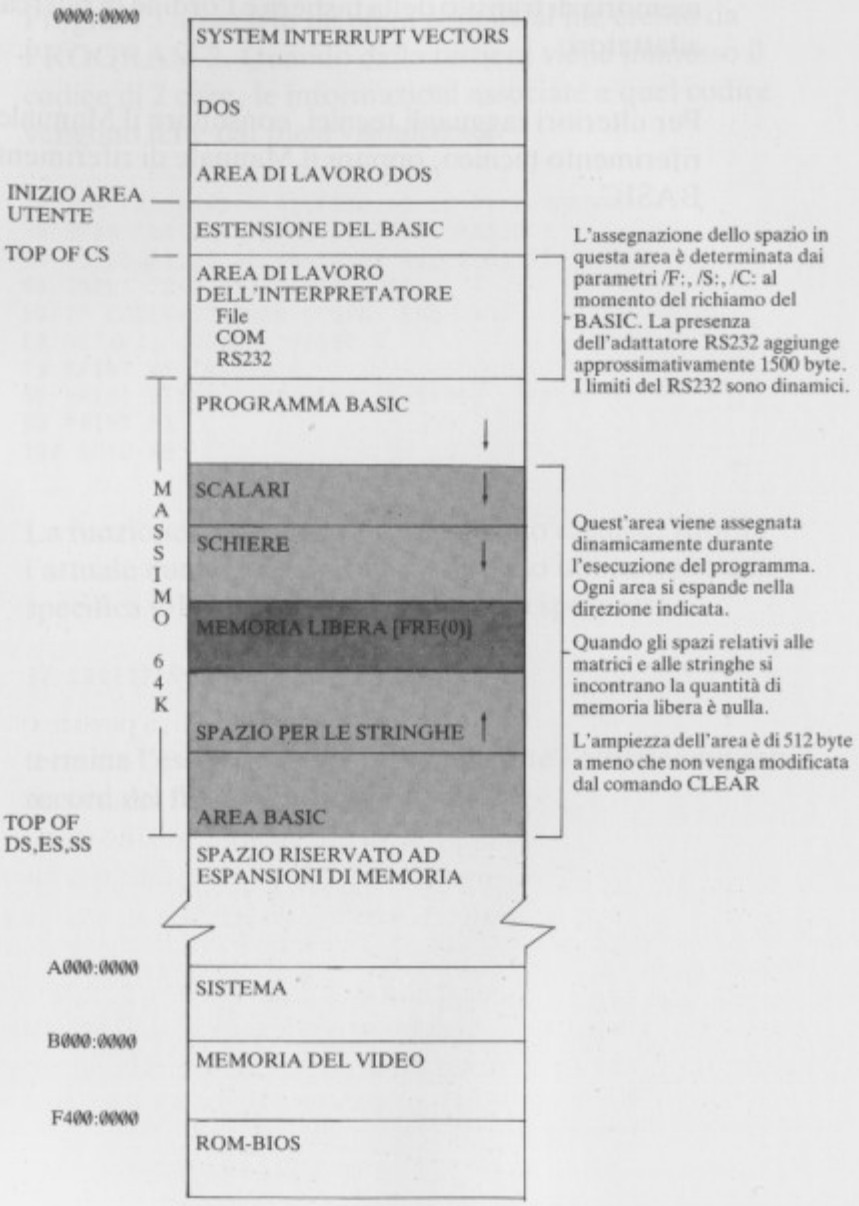
Appendice B. Informazioni sulla memoria

Questa appendice contiene informazioni relative alla memoria per il BASIC, compresi la mappa di memoria ed alcuni dettagli sulla memorizzazione delle variabili, la memoria di transito della tastiera e l'ordine di ricerca per gli adattatori.

Per ulteriori ragguagli tecnici, consultare il Manuale di riferimento tecnico, oppure il Manuale di riferimento BASIC.

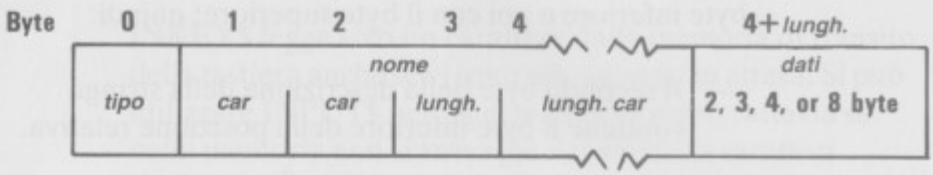
Mappa della memoria

Questa è una mappa di memoria per il BASIC su disco e Avanzato. Le estensioni del DOS e del BASIC non sono presenti nel BASIC su cassetta. Gli indirizzi sono espressi in formato esadecimale SEGMENT:OFFSET.



Come vengono memorizzate le variabili

Le variabili scalari vengono memorizzate nell'area dati del BASIC nel modo seguente:



tipo Identifica il tipo di variabile:

2	intero
3	stringa
4	precisione singola
8	precisione doppia

nome è il nome della variabile. I primi due caratteri del nome vengono memorizzati nei byte 1 e 2. Il byte 3 indica quanti caratteri si trovano ancora nel nome della variabile. Questi caratteri aggiuntivi sono memorizzati a partire dal byte 4.

Ciò significa che un nome di variabile occupa almeno tre byte. Un nome di uno o due caratteri occupa esattamente tre byte; un nome di *x* caratteri occupa *x*+1 byte.

dati segue il nome della variabile e può essere di 2,3,4 o 8 byte (come per *tipo)*. Il valore ritornato dalla funzione VARPTR punta a questi dati.

Nelle variabili a stringa, *dati* è la *descrizione della stringa:*

- Il primo byte della descrizione della stringa contiene la lunghezza della stringa (Ø-255).
- Gli ultimi due byte della descrizione della stringa contengono gli indirizzi della stringa nello spazio dati del BASIC (la posizione relativa nel segmento assunta per difetto). Gli indirizzi sono memorizzati prima con il byte inferiore e poi con il byte superiore; quindi:
  - — Il secondo byte della descrizione della stringa contiene il byte inferiore della posizione relativa.
  - — Il terzo byte della descrizione della stringa contiene il byte superiore della posizione relativa.

Nelle variabili numeriche, *dati* contiene l'effettivo valore della variabile:

- I valori interi sono memorizzati in due byte, prima quello inferiore, poi quello superiore.
- I valori a precisione singola sono memorizzati in quattro byte nel formato binario interno del BASIC in punto decimale mobile.
- I valori a precisione doppia sono memorizzati in otto byte nel formato binario interno del BASIC in punto decimale mobile.

Memoria di transito della tastiera

I caratteri immessi da tastiera vengono salvati nella memoria di transito della tastiera finché non vengono elaborati. La memoria può contenere 15 caratteri al massimo; se si tenta di immettere più di 15 caratteri, l'elaboratore emette un suono.

INKEY$ legge solo un carattere dalla memoria di transito della tastiera anche se vi sono più caratteri in attesa. Si può utilizzare INPUT$ per leggere più caratteri; tuttavia se nella memoria non si trova già il numero di caratteri richiesto, il BASIC attende finché non vengono immessi caratteri sufficienti.

La memoria di transito della tastiera dell'elaboratore può essere svuotata dalle seguenti righe di codifica:

```
DEF SEG=0: POKE 1050, PEEK(1052)
```

Questa tecnica può essere utile, ad esempio, per svuotare la memoria di transito prima di richiedere all'utente di premere un tasto qualsiasi.

Ordine di ricerca per gli adattatori

Le stampatrici associate ad LPT1:, LPT2: e LPT3: vengono assegnate quando si accende l'elaboratore. Il sistema ricerca gli adattatori per la stampatrice in una particolare sequenza: il primo adattatore trovato diventa LPT1:, il secondo (se esiste) LPT2: ed il terzo (se esiste) LPT3:. L'ordine di ricerca è il seguente:

1. Un adattatore per il video monocromatico IBM e stampatrice parallela.

2. Un Adattatore per la stampatrice parallela.

3. Un Adattatore per la stampatrice parallela che è stato modificato per variare il suo indirizzo di base.

Se una stampatrice viene reindirizzata con il comando MODE dal DOS, la modifica è valida anche nel BASIC.

I dispositivi per la comunicazione COM1: e COM2: vengono assegnati in maniera simile alle stampatrici. L'ordine di ricerca è il seguente:

1. Un Adattatore per le comunicazioni asincrone.

2. Un Adattatore per le comunicazioni asincrone modificato.

Scambio delle unità video

Se sul Personal Computer IBM sono installati gli Adattatori per il video grafico a colori e per il video IBM monocromatico e stampatrice parallela, il BASIC normalmente userà il video monocolore IBM. Tuttavia è possibile passare da un video all'altro, nel BASIC, usando la codifica che segue:

```
10 ' passaggio all'adattatore per video monocromatico
20 DEF SEG = 0
30 POKE &H410, (PEEK(&H410) OR &H30)
40 SCREEN 0
50 LOCATE ,,1,12,13
10 ' passaggio all'adattatore per video a colori
20 DEF SEG = 0
30 POKE &H410, (PEEK(&H410) AND &HCF) OR &H10
40 SCREEN 1,0,0,0
50 SCREEN 0
60 WIDTH 40
70 LOCATE ,,1,6,7
```

Nota: se si usa questa tecnica, lo schermo al quale si passa appare privo di contenuto, inoltre può essere necessario tenere traccia della posizione del cursore indipendentemente da ciascun video.

Note: