Commodore plus/4

MANUALE PER L'UTENTE

COMMODORE PLUS/4

MANUALE PER L'UTENTE

Commodore Italiana SpA

Via F.lli Gracchi, 48 - 20092 Cinisello Balsamo (Milano)
Tel. 02/618321

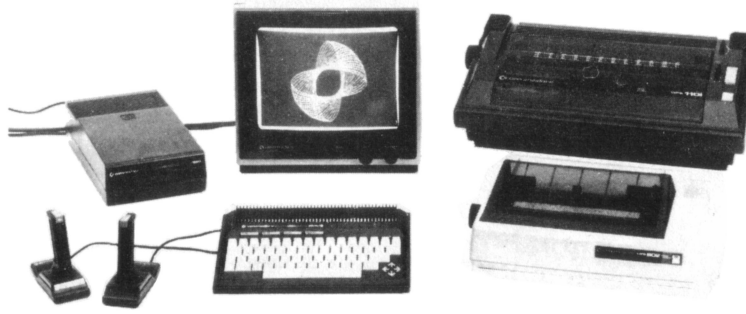
IMPORTANTE "Questa apparecchiatura genera ed utilizza energia a frequenza radio e se, non installata correttamente, vale a dire in stretta osservanza delle istruzioni del costruttore, può causare interferenze nella ricezione radio-televisiva. È stata controllata e ritenuta conforme ai limiti previsti per le unità di calcolo Classe B, in rispetto al sottocapitolo J del capitolo 15 delle direttive FCC, stabilite per fornire sufficiente protezione da tali interferenze in installazioni residenziali. Si può accertare la presenza di eventuali interferenze accendendo e spegnendo l'apparecchiatura. Se questa apparecchiatura dovesse causare interferenze alla ricezione radio-televisiva, è possibile cercare di ovviare all'inconveniente seguendo una o più delle seguenti regole:

- Riorientare l'antenna di ricezione
- Allontanare il calcolatore dal ricevitore
- Riposizionare il calcolatore rispetto al ricevitore
- Inserire la spina del calcolatore in una presa differente in modo che il calcolatore ed il ricevitore siano su circuiti di rete diversi.

Se necessario, l'utente dovrà consultare il fornitore della Commodore o un tecnico specialista di radio/TV per ulteriori suggerimenti. Può essere utile consultare il manuale edito dal FCC "Come identificare e risolvere problemi di interferenze Radio/TV". Questo opuscolo è disponibile presso l'Ufficio Stampa Governativo.
Si consiglia di usare solo cavi, accessori e periferiche consigliati dalla Commodore per il Plus/4. Tutti i cavi, inclusi i cavi per l'allacciamento televisivo, la porta seriale, la porta video, il Datasette e i joystick, sono appositamente schermati, in conformità con i regolamenti del FCC. La mancata utilizzazione degli accessori e cavi appropriati rende nulla l'approvazione FCC e può causare dannose interferenze radio.

INDICE

INTRODUZIONE



Il Plus/4 è il primo home Computer appositamente studiato per applicazioni commerciali, allo stesso tempo mantenendo le usuali caratteristiche di un "home Computer".
In questo manuale verranno esaminate le particolari funzioni del Plus/4. Si vedrà come:

- installare il Plus/4
- usare le differenti funzioni di tutti i tasti
- accedere ai vari tipi di Programmi Commodore
- usare le capacità del Plus/4 relative ai calcoli matematici, ai grafici, alla produzione di suoni e alla programmazione.

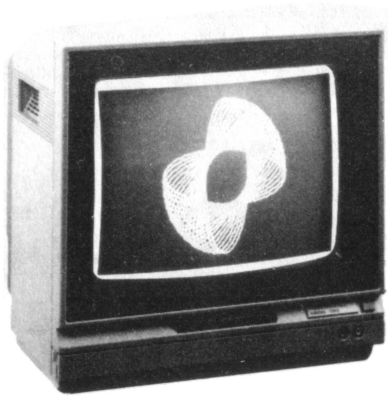
L'altro manuale allegato al computer (Manuale del Software integrato incorporato nel Plus/4), guida all'uso dei pacchetti di elaborazione testi, foglio elettronico (spreadsheet), data base e grafica.
Anche se l'interesse del lettore è principalmente orientato verso queste applicazioni commerciali, ugualmente si raccomanda di leggere almeno il primo Capitolo di questo manuale ("Disimballaggio e Installazione"), prima di passare al Manuale del Software Incorporato.

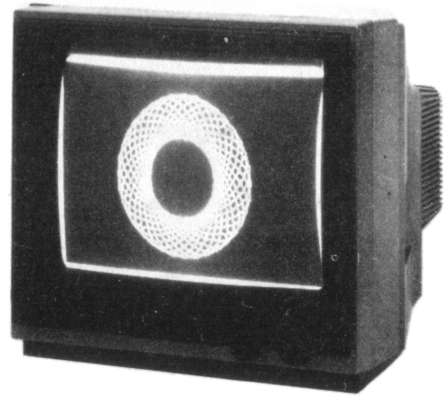
CARATTERISTICHE PARTICOLARI DEL PLUS/4

- RAM da 64K (60K disponibili per la programmazione in BASIC)
- Tastiera tipo macchina per scrivere
- Software opzionale incorporato
- Schermo con capacità di finestramento
- Tasto HELP
- 8 tasti funzione pre-programmati e riprogrammabili
- 4 tasti separati per controllo cursore
- Possibilità di usare gran parte delle periferiche COMMODORE 64 e VIC-20
- 121 colori (16 colori primari, 8 livelli di luminanza)
- Più di 75 comandi BASIC
- Tracciamento grafico ad alta risoluzione
- Possibilità di dividere lo schermo per testo e grafici ad alta risoluzione
- Set di caratteri grafici sulla tastiera
- Controlli di colore sulla tastiera
- Risoluzione di schermo a 320 x 200 punti indirizzabili (pixel)
- Caratteri invertiti e lampeggianti
- Generatori di suono a 2 tonalità
- Monitor linguaggio macchina incorporato (17 comandi).

CREAZIONE DI UN SISTEMA COMPLETO DI ELABORAZIONE

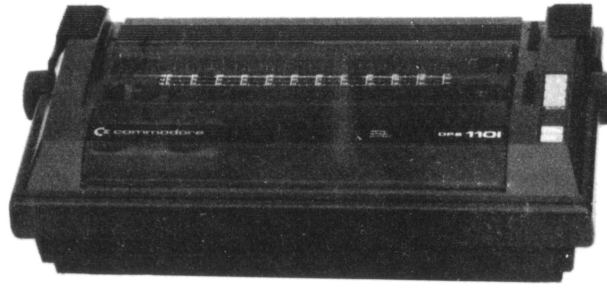
Elaboratore: Commodore PLUS/4





Video: Monitor a Colori Commodore 1702 oppure 1802/1803

Memorizzazione: Datassette Commodore 1531 (mangianastri) o Unità a Disco Commodore



Stampante: stampante Commodore

Comandi: Joystick del Commodore PLUS/4

CONCLUSIONE Si consiglia, di abbonarsi alla rivista Commodore per essere aggiornati sulle ultime novità relative al nostro computer e di associarsi ai club locali per l'utente Commodore. Leggere questo manuale, fare gli esercizi e provare dei programmi. Tenersi in contatto con i rivenditori Commodore locali per aggiornamenti sul software, sui libri e sulle periferiche. Imparando a programmare, a giocare e ad esercitarsi, apprezzerete il nuovo PLUS/4 COMMODORE.

CAPITOLO 1

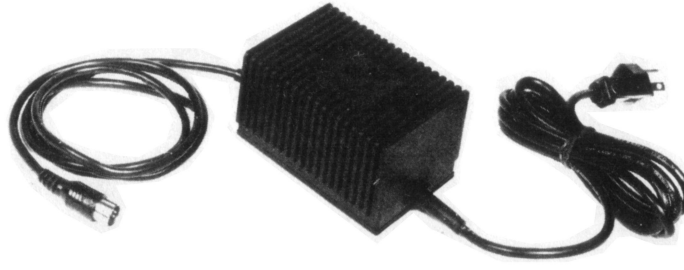
DISIMBALLAGGIO E INSTALLAZIONE

- Disimballaggio del Plus/4 Commodore
- Interruttori e connettori
- Installazione del Plus/4 Commodore
- Diagramma dell'individuazione e dell'eliminazione degli inconvenienti
- Periferiche

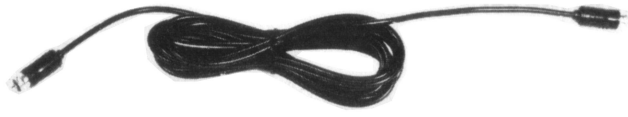
DISIMBALLAGGIO DEL PLUS/4 COMMODORE

Una volta aperta la scatola del Plus/4, la prima cosa da fare è assicurarsi della presenza di tutti gli articoli contenuti nella seguente lista:

1. Plus/4 Commodore



2. Alimentatore
   Uno dei 2 connettori va collegato a una presa a muro e l'altro sul lato sinistro del computer.



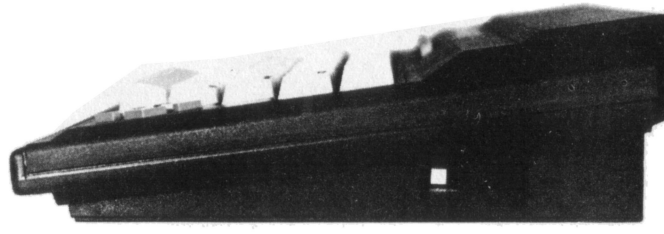
3. Cavo RF
   Collega l'interfaccia TV al connettore di output RF che si trova sulla sinistra del Plus/4. Questo cavo non è necessario per collegare il Plus/4 a un monitor.

4. Manuale per l'utente

5. Manuale per il software integrato incorporato nel Plus/4.

Se uno o più articoli dovessero risultare mancanti, contattare subito il rivenditore per il completamento della fornitura. Prima di procedere a qualsiasi collegamento, osservare attentamente le figure rappresentanti il computer che evidenziano i connettori per una facile e veloce installazione.

INTERRUTTORI E CONNETTORI Lato destro del PLUS/4



2 1

1 Interruttore di Accensione/Spegnimento

Il Plus/4 si deve SPEGNERE quando si inseriscono o si tolgono cartucce o si collegano e scollegano periferiche come la stampante o l'unità a disco. Al disotto del lato sinistro della tastiera è allocata una spia rossa di alimentazione che indica lo stato di accensione.

2 Pulsante di Reinizializzazione (RESET)

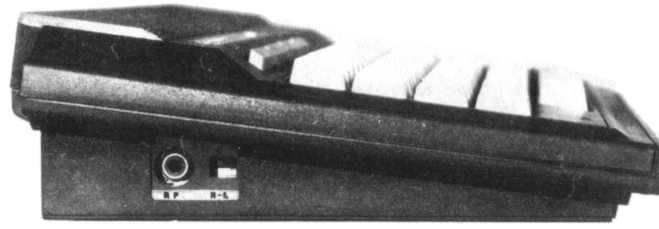
Ci sono due modi di utilizzare il pulsante RESET:

1. Per reinizializzare il computer simulando l'accensione basta premere il pulsante RESET una sola volta. Avvertenza: quando si preme il pulsante RESET si perde qualsiasi programma BASIC correntemente in memoria.*

2. Per reinizializzare il Plus/4 e mantenere il programma BASIC, premere contemporaneamente il tasto RUN/STOP e il pulsante RESET. Dopo di che il Plus/4 cede il controllo al monitor del linguaggio macchina incorporato. Battere X e premere RETURN per tornare al programma BASIC che non ha subito modifiche nella memoria del Plus/4. Battere LIST per visualizzare il programma sullo schermo.

*Quando si preme RESET, il Plus/4 emula automaticamente il comando NEW ripulendo lo schermo e viceversa. Consultare la Guida di Riferimento per il programmatore del Plus/4 per ulteriori informazioni sulla non cancellazione del programma in caso di pressione accidentale del pulsante RESET.

Lato Sinistro del PLUS/4



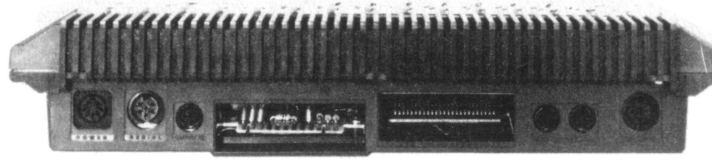
3

Sia il connettore che l'interruttore sul lato sinistro del Plus/4 sono utilizzati nei collegamenti con TV. Non sono necessari per il collegamento del Plus/4 al monitor.

3 Connettore RF

Collegare qui il cavo RF (il cavo sottile nero): un capo a questo connettore e l'altro al televisore.

Retro del Computer



5 6 7 8 9 10 11

I connettori sul retro del computer collegano vari accessori al Plus/4. Ogni connettore è diverso. Assicurarsi di effettuare correttamente i collegamenti.

5 Connettore d'alimentazione
Collegare qui un capo del cavo dell'alimentatore e l'altro capo ad una presa standard a muro.

6 Uscita Seriale
Si possono collegare a questo connettore sia l'unità disco che la stampante. Volendo collegare entrambe, inserire per prima l'unità disco e poi il cavo della stampante sul retro dell'unità disco.

7 Porta per la cassetta
Collegare qui il mangianastri Datassette 1531 della Commodore.

8 Porta RS-232
Collegare qui il modem, l'adattatore RS-232 o dispositivi simili. Un adattatore RS-232 rende possibile l'allacciamento di accessori non conformi alle porte standard dell'apparecchiatura Commodore.

9 Porta di Espansione della Memoria
Collegare qui le cartucce del software Plus/4 e l'unità disco SFD-481 del Plus/4. Prima di inserire o rimuovere le cartucce, assicurarsi che il Plus/4 sia spento.

10 Joy 1 e Joy 2: Porte di Gioco
Collegare qui i joystick. Il Plus/4 utilizza particolari modelli di joystick disponibili presso il vostro fornitore Commodore.

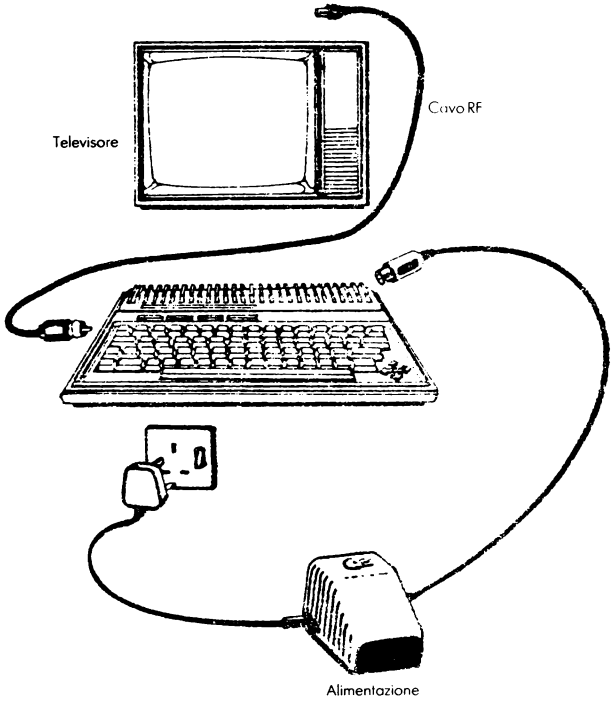
11 Connettore Video
Collegare qui il cavo che unisce un monitor al Plus/4. Sebbene il connettore sia a 8 pin, si può anche usare un cavo da 5 pin. I monitor a colori della Commodore vengono corredati da un cavo a 8 pin da usare con il Plus/4.

INSTALLAZIONE DEL PLUS/4

- Per installare il Plus/4 sono necessarie almeno 2 prese a muro: una per il Plus/4 e una per il TV o per il monitor.
- Se si vuole installare un'unità disco e una stampante, sono necessarie ulteriori prese a muro.
- Il Plus/4 deve essere installato ad una sufficiente distanza dal televisore.
- Prima di iniziare l'installazione, assicurarsi che il computer sia SPENTO controllando che la SPIA di ALIMENTAZIONE sulla parte anteriore sinistra non sia accesa.

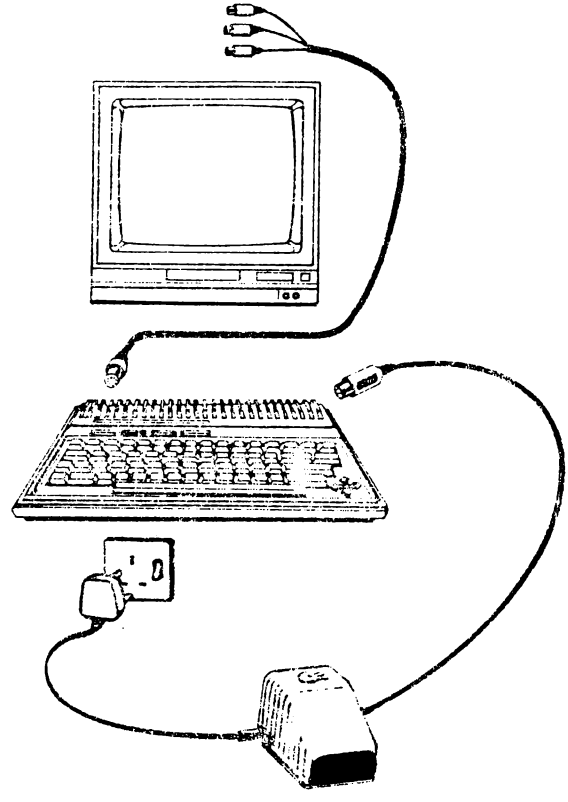
Collegamento del PLUS/4 al Televisore

Inserite una estremità del cavo TV nella presa del vostro televisore. Inserite l'altra estremità del cavo nella presa RF del computer.



Collegamento del PLUS/4 Commodore ad un Monitor

Per collegare il computer ad un monitor anziché al TV, seguire le istruzioni contenute nel manuale allegato al monitor. L'allacciamento ad un monitor, come ad esempio Monitor a Colori 1702 della Commodore, è facile. Richiede solamente un cavo che dal monitor colleghi direttamente il connettore VIDEO sul retro del computer.



Fasi Finali

1. Collegare il cavo dell'alimentatore al Plus/4. Collegare il connettore quadrato del cavo al connettore POWER situato sul retro del computer; collegare l'alimentatore alla presa a muro.
2. Accendere il computer (l'interruttore è sul lato destro, guardando il Plus/4).
3. Usando il TV, sintonizzarsi usando la banda di frequenza UHF. Con un monitor a colori della Commodore, utilizzare gli spinotti posteriori e controllare che l'interruttore back/front sia posizionato su back.

Se ogni operazione ha avuto esito positivo, questo messaggio appare sullo schermo:

```
COMMODORE BASIC 3.5 60671 BYTES FREE
READY.
```

Il cursore lampeggiante sotto al messaggio READY indica che il Plus/4 attende l'introduzione. Il colore di fondo è bianco, mentre le lettere sono stampate in nero e la cornice dello schermo è viola chiaro.

4. All'insorgere di qualsiasi problema, consultare il diagramma dell'individuazione e dell'eliminazione degli inconvenienti. Potrebbe essere necessario regolare l'apparecchio TV per avere un'immagine più nitida.

DIAGRAMMA DI INDIVIDUAZIONE ED ELIMINAZIONE DEGLI INCONVENIENTI

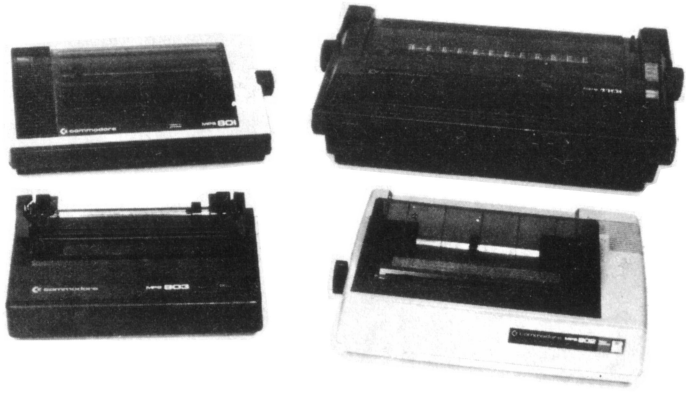
Sintomo	Causa	Rimedio
Spia di accensione spenta (non su ON)	Computer non ACCESO	Assicurarsi che l'interruttore di alimentazione sia in posizione "ON".
	Cavo dell'alimentazione non inserito	Controllare che nella presa di corrente non ci siano fili allentati o scollegati.
	Alimentatore non inserito	Controllare il collegamento con la presa a muro.
	Il fusibile del computer è saltato	Portare il sistema da un fornitore autorizzato per sostituire il fusibile.
Nessuna immagine	TV su canale errato	Verificare il canale per l'immagine.
	Allacciamento errato	Il computer si allaccia ai terminali UHF.
	Cavo del video non inserito	Controllare il collegamento del cavo d'uscita del TV.
Immagine confusa sullo schermo del TV (cartuccia inserita)	Cartuccia inserita non appropriatamente	Reinserire la cartuccia dopo aver spento il computer.
Immagine senza colore	TV sintonizzato male	Risintonizzare il TV.
	TV non collegato	Controllare i collegamenti.
Immagine buona, senza sonoro	Volume del TV troppo basso	Regolare il volume.
	TV mal sintonizzato	Risintonizzare il TV.
		Controllare il collegamento.

IMPORTANTE: Alcuni apparecchi televisivi non sono in grado di visualizzare l'intero schermo del Plus/4 e perciò rimangono fuori la colonna all'estrema sinistra e quella all'estrema destra. In tal caso si raccomanda l'uso di un diverso apparecchio o di un monitor come quello a colori 1702 o 1703 Commodore.

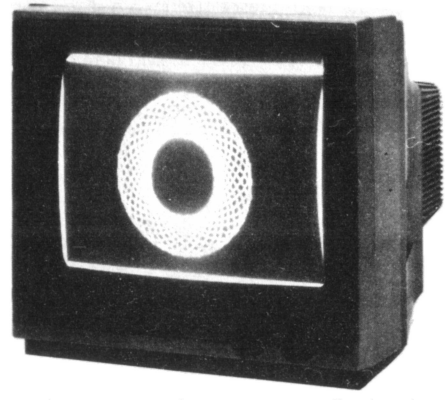
Se ciò è impossibile, si può risolvere il problema premendo il tasto ESC, seguito dal tasto "R". Questo riduce le dimensioni della visualizzazione dell'elaboratore, consentendo all'intera immagine di adattarsi ogni volta che si accende o si reinizializza il Plus/4.

PERIFERICHE Le periferiche sono accessori che si possono aggiungere per completare il sistema Plus/4. Sono disponibili presso i fornitori Commodore e consentono un utilizzo più completo del Plus/4. Le periferiche permettono al sistema Plus/4 di conservare e memorizzare dati, stampare su carta (in bianco e nero o a colori), utilizzare software su dischi o cassette e accedere alle informazioni e ai servizi disponibili attraverso le telecomunicazioni.

Per memorizzare o richiamare programmi, è necessario un dispositivo che memorizzi i dati. I dati possono essere memorizzati e recuperati sia da cassetta sia da dischetto. Per l'utilizzo dei dischetti è necessaria una unità a disco. Le unità a disco sono particolarmente veloci ed efficienti. Le unità a disco compatibili con il Plus/4 sono i modelli Commodore 1541 e 1551. Per la memorizzazione e il ricupero da cassette, è sufficiente il registratore Commodore 1531 DATASSETTE.



Mentre si usa un programma di elaborazione testi o un pacchetto grafico con il Plus/4, una stampante permette di riprodurre su carta ciò che compare sullo schermo. Ci sono diversi modelli disponibili di stampanti Commodore compatibili con il Plus/4, tra cui la Stampante MCS-801, MPS-802, MPS-803 (con trascinamento a trattore) e la DPS-1101 (qualità lettera). Molte stampanti si specializzano in diversi tipi di elaborati. Chiedere al rivenditore quella che meglio si adatta alle vostre esigenze.



Nel caso in cui il TV non dia immagini chiare come quelle desiderate per il computer, si tenga presente che i monitor a colori della Commodore sono stati appositamente progettati per dare un'immagine di elevata luminosità e definizione dell'output del Plus/4 per lo schermo. Sono disponibili diversi modelli di monitor Commodore fra cui il 1702 e 1802/1803.

CAPITOLO 2

USO DELLA TASTIERA E DELLO SCHERMO

- Panoramica della tastiera
- Tasti speciali
- Tasti grafici
- Tasti funzione programmabili
- Tasto HELP

PANORAMICA DELLA TASTIERA

Gran parte dei tasti del Plus/4 sono identici a quelli di una normale macchina per scrivere, ma rispetto a questi ultimi hanno più applicazioni. In questa sezione verrà mostrato come usare tasti speciali come il tasto C= e i tasti di controllo cursore e verranno illustrate le caratteristiche di ogni tasto, compreso l'uso dei simboli grafici raffigurati sulla parte frontale di molti di questi.
Mentre procede l'esame della tastiera del Plus/4, si consiglia di esercitarsi nell'uso dei tasti.

TASTI SPECIALI

RETURN

Il tasto RETURN va premuto alla fine di ogni riga di istruzioni digitata sul Plus/4 Commodore. In realtà questo tasto si può definire un tasto di INTRODUZIONE, in quanto RETURN introduce nell'elaboratore informazioni e istruzioni.

SHIFT

Questo tasto funziona come il tasto delle maiuscole di una normale macchina per scrivere. Il Plus/4 ha due tasti SHIFT e un tasto SHIFT LOCK che funziona come il tasto blocca maiuscole di una macchina per scrivere.
Premendo il tasto SHIFT si possono ottenere i simboli grafici rappresentati sul lato destro di ogni tasto grafico, se in modalità maiuscola/grafica.

Quando viene acceso, il Plus/4 si trova automaticamente nella modalità maiuscola/grafica. Nella modalità maiuscola/grafica, tutte le lettere appaiono maiuscole senza l'uso del tasto SHIFT Premendo il tasto SHIFT insieme al tasto di una lettera, si ottiene quella lettera maiuscola quando si è nel modalità testo maiuscola/minuscola (come avviene con il tasto delle maiuscole di una macchina per scrivere). Quando si è in questa modalità, le lettere che vengono battute sono minuscole, a meno che non si usi il tasto SHIFT.

NOTA: Si può passare dalla modalità maiuscola/grafica a quella di testo maiuscola/minuscola e viceversa, premendo contemporaneamente i tasti SHIFT e C=.

Premere questo tasto per interrompere l'esecuzione di un programma (funzione STOP). Premendo questo tasto si restituisce il controllo all'utente e alla tastiera. Tenendo premuti contemporaneamente i tasti SHIFT e RUN/STOP, il Plus/4 carica ed esegue il primo programma presente su un disco contenuto nell'unità a disco (funzione RUN).

Tasti Controllo Cursore

Spostare velocemente il cursore in ogni direzione dello schermo è un'operazione semplice. Premere il tasto controllo cursore che indica la direzione in cui si desidera spostarlo. Come tutti i tasti del Plus/4, ogni tasto controllo cursore ripete la sua funzione finché è tenuto premuto, questa funzione di ripetizione automatica continua a spostare il cursore finché non si rilascia il tasto.

NOTA: Il cursore si può spostare sopra lettere e numeri sullo schermo senza modificarli.

INST/DEL

Premendo questo tasto, si possono inserire (INSERT) e cancellare (DELETE) lettere e numeri dalla riga che si sta battendo. Quando si preme questo tasto da solo, il carattere immediatamente alla sinistra del cursore scompare e il cursore si sposta al posto del carattere cancellato. Si possono usare i tasti del cursore per tornare al centro di una riga e quindi usare **DEL** per cancellare una lettera. Quando si esegue questa operazione, la lettera sulla sinistra del cursore viene cancellata e il resto delle lettere si sposta di uno spazio a sinistra per riempire il vuoto.
Si può creare uno spazio per inserire lettere e numeri usando i tasti **SHIFT** e **INST**. Lo spazio si forma a destra del cursore, mentre il cursore stesso non si sposta. Quando si inserisce uno spazio al centro di una riga, il resto della riga si sposta a destra.
Il tasto **INST/DEL** consente un notevole risparmio di tempo quando si desideri correggere o modificare ciò che si è battuto.

CLR/HOME

Questo tasto adempie a tre funzioni: HOME, CLEAR e CLEAR WINDOWS. Quando viene premuto, il cursore si sposta immediatamente nell'angolo in alto a sinistra dello schermo. Questa viene definita posizione HOME. Il resto dello schermo rimane immutato (funzione HOME). Se si preme **CLR/HOME** insieme al tasto **SHIFT**, non solo il cursore si sposta a HOME, ma lo schermo di cancella (funzione CLEAR). Tutto quello che rimane sullo schermo è il cursore lampeggiante in alto a sinistra. Se si preme questo tasto insieme al tasto **C=**, qualsiasi finestra impostata viene cancellata (funzione CLEAR WINDOW). Le finestre sono aree di lavoro delimitate su una parte dello schermo; di esse si parlerà più esaurientemente in seguito.

CONTROL

Questo tasto funziona sempre in coppia con un altro tasto. Il tasto **CONTROL** funziona come il tasto **SHIFT**: va tenuto premuto mentre si preme l'altro tasto.

1. Come è illustrato anche nella sezione TASTI COLORE, premendo **CONTROL** e un tasto colore, il testo sullo schermo apparirà del colore prescelto.
2. Si può interrompere momentaneamente un programma che sta scorrendo sullo schermo in seguito a un istruzione PRINT o LIST premendo i tasti **CONTROL** e S (premere un tasto qualsiasi per riprendere l'operazione interrotta).
3. **CONTROL** è anche usato con i tasti **REVERSE ON/OFF** e **FLASH ON/OFF**.

Inoltre, alcuni dei programmi in vendita si servono del tasto CONTROL per particolari funzioni.

C=

Come il tasto **CONTROL**, il tasto Commodore funziona insieme ad altri tasti ed ha quattro funzioni:

1. Quando viene usato con il tasto **SHIFT**, il tasto **C=** cambia la modalità maiuscola/grafica in modalità testo maiuscola/minuscola.
2. Quando ci si trova in una di queste due modalità, il tasto **C=** funziona come un tasto SHIFT, consentendo di battere il simbolo grafico rappresentato sul lato frontale sinistro di ogni tasto. Tenere premutó **C=** e il tasto grafico che si desidera.

3. Quando si vuole cambiare il colore corrente in uno degli 8 colori riportati nella parte inferiore del lato frontale dei tasti colore, premere **C=** e il tasto colore desiderato.
4. Quando si desidera rallentare lo scorrimento del programma sullo schermo tenere premuto il tasto **C=** . La velocità di scorrimento diminuisce notevolmente. Quando si rilascia il tasto, lo scorrimento riacquista la sua velocità normale.

Tasti Colore

È possibile cambiare il colore delle lettere, dei numeri e dei simboli grafici dello schermo in uno dei 16 colori disponibili nel Plus/4. L'operazione è semplice:

- Se si desidera uno degli 8 colori elencati nella parte superiore del lato frontale dei tasti colore (ad esempio BLK per nero), premere contemporaneamente il tasto CONTROL e il tasto con il colore desiderato.
- Se si desidera uno degli 8 colori elencati nella parte inferiore del lato frontale dei tasti colore (ad esempio arancione), tenere premuto il tasto C= e quindi premere il tasto del colore desiderato.

È consigliabile esercitarsi nel cambiare i colori, per essere certi di aver capito le istruzioni impartite. Notare che dopo aver cambiato colore, ogni lettera e numero battuto in seguito apparirà del colore scelto per ultimo.

REVERSE ON/OFF

Il Plus/4 consente di stampare l'immagine invertita di lettere e numeri. In altre parole, se si stanno usando lettere nere su fondo giallo, usando i tasti di immagine invertita si stamperanno lettere gialle su sfondo nero. Per ottenere immagini invertite: premere il tasto CONTROL e il tasto RVS ON. Tutto quello che si batterà adesso apparirà invertito, finché non si premeranno i tasti CONTROL e RVS OFF, il tasto RETURN o i tasti ESC e 0. Sarà così ripristinata la battitura di caratteri normali (ossia non invertiti).

FLASH ON/OFF

È possibile far lampeggiare i caratteri sullo schermo. Premere i tasti CONTROL e FLASH ON per far lampeggiare ciò che si batte. Premendo CONTROL e FLASH OFF, RETURN o ESC si ritornerà a dei caratteri normali (non lampeggianti).

TASTI GRAFICI Come già detto in precedenza, quando si accende il Plus/4 si è automaticamente nella modalità testo maiuscola/grafica. Quando si è in questa modalità, è possibile battere l'intero set di più di 60 caratteri grafici visibili sul lato frontale di molti tasti, come pure tutte le lettere maiuscole, senza usare il tasto SHIFT.
Il tasto SHIFT, usato in questa modalità consente di battere caratteri grafici, invece di lettere maiuscole.
Su ogni tasto grafico si trovano due simboli grafici:

- Per stampare il simbolo grafico sulla destra, tenere premuto il tasto SHIFT mentre si preme il tasto appropriato.
- Per stampare il simbolo grafico sulla sinistra, tenere premuto il tasto α mentre si preme il tasto prescelto.

È possibile creare disegni, diagrammi e progetti stampando i caratteri grafici l'uno di fianco all'altro o uno sopra l'altro. È consigliabile esercitarsi nella stampa dei caratteri grafici, per meglio comprenderne il funzionamento. Il capitolo 7 fornisce ulteriori informazioni riguardo ai grafici.
Si può passare dalla modalità maiuscola/grafica alla modalità maiuscola/minuscola premendo contemporaneamente i tasti SHIFT e α . In entrambe le modalità, battere i comandi BASIC senza tenere premuto il tasto SHIFT.
In questa modalità, si possono battere lettere maiuscole e minuscole, proprio come in una normale macchina per scrivere. Per ottenere lettere maiuscole è necessario usare il tasto SHIFT. È anche possibile usare i caratteri grafici (sul lato frontale sinistro dei tasti), che stampano come nella modalità maiuscola/grafica; premere contemporaneamente il tasto α e quello grafico. I caratteri grafici sul lato sinistro sono adatti alla creazione di grafici, diagrammi e moduli commerciali.

ESCAPE

Il tasto ESC consente l'esecuzione di numerose funzioni speciali di screen editor (correzione sullo schermo), incluse quelle che utilizzano la prestazione di finestramento del Plus/4.
Le finestre sono aree dello schermo, definite dall'utente, che possono essere usate come aree di lavoro senza influenzare il resto dello schermo. Il tasto ESC può eseguire diverse funzioni di correzione di finestre, come pure molte altre funzioni usuali, come ad esempio inserimenti, le cancellazioni e lo scorrimento dell'immagine.
Il tasto ESC è particolarmente usato con tasti alfabetici standard. Per attivare una funzione, premere il tasto ESC, seguito da uno dei tasti sotto elencati.

A Modalità di inserimento automatico
B Impostazione dell'angolo inferiore destro della finestra (alla posizione corrente del cursore)
C Annullamento della modalità di inserimento automatico
D Cancellazione della riga corrente
I Inserimento di una riga
J Spostamento all'inizio della riga corrente
K Spostamento alla fine della riga corrente
L Attivazione dello scorrimento dell'immagine
M Disattivazione dello scorrimento dell'immagine
N Ritorno alle normali dimensioni di visualizzazione
O Annullamento delle modalità di inserimento, "quote", inversione e lampeggiamento
P Cancellazione totale fino all'inizio della riga corrente
Q Cancellazione totale fino alla fine della riga corrente
R Riduzione della visualizzazione
T Impostazione dell'angolo superiore sinistro della finestra
V Scorrimento verso l'alto
W Scorrimento verso il basso
X Annullamento della funzione escape

Simboli Speciali

La tastiera del Plus/4 contiene anche particolari simboli che non compaiono su molte macchine per scrivere e persino su gran parte degli elaboratori. Questi simboli speciali comprendono il simbolo della sterlina inglese (L.), il pi greco (π), i simboli maggiore di e minore di (< >), parentesi quadre ([ ]), e frecce (↑). Questi tasti con simboli speciali vengono utilizzati nella programmazione del Plus/4.

TASTI FUNZIONE PROGRAMMABILI

I quattro tasti nella parte superiore della tastiera sono tasti funzione speciali che consentono un risparmio di tempo eseguendo compiti ripetitivi utilizzando un solo tasto.
È possibile visualizzare ciò che viene eseguito da ogni tasto battendo KEY (tasto) e premendo RETURN.

Lo schermo visualizza:

```
KEY
KEY 1, "GRAPHIC"
KEY 2, "DLOAD"+CHR$(34)
KEY 3, "DIRECTORY"+CHR$(13)
KEY 4, "SCNCLR"+CHR$(13)
KEY 5, "DSAVE"+CHR$(34)
KEY 6, "RUN"+CHR$(13)
KEY 7, "LIST"+CHR$(13)
KEY 8, "HELP"+CHR$(13)
```

Funzione dei tasti:

TASTO 1 Introduce una delle modalità grafiche quando si fornisce il numero dell'area grafica (ad esempio, GRAPHICS 2, che corrisponde alla modalità di schermo diviso ad alta risoluzione) e **RETURN**. Negli elaboratori con software incorporato il tasto 1 è ridefinito, così che, premendolo, si attiva il pacchetto di software.

TASTO 2 Stampa DLOAD sullo schermo. Per caricare un programma dal disco è sufficiente battere il nome del programma seguito da **RETURN** subito dopo questo tasto.

TASTO 3 Fornisce un elenco dei file del disco contenuto nel drive.

TASTO 4 Cancella lo schermo (anche se si è in una delle modalità grafiche).

TASTO 5 Stampa sullo schermo DSAVE. Digitare il nome del programma per salvare il programma corrente su disco e premere **RETURN**.

TASTO 6 Esegue il programma corrente (funzione RUN)

TASTO 7 Visualizza un listato del programma corrente (funzione LIST)

TASTO 8 (Tasto **HELP**) evidenzia, facendoli lampeggiare, gli errori nelle righe del programma.

Per utilizzare una di queste funzioni, premere il tasto funzione appropriato. Per servirsi delle funzioni 4, 5, 6, 7 è necessario usare il tasto **SHIFT**.
È possibile ridefinire uno qualsiasi di questi tasti per eseguire una funzione che si adatti alle proprie necessità. La ridefinizione è una operazione semplice: usando il comando KEY, è possibile ridefinire i tasti con programmi BASIC, o modificarli in qualsiasi momento in modalità diretta (le nuove definizioni vengono annullate quando si spegne il Plus/4). Si possono ridefinire quanti tasti si vogliono, per un numero qualsiasi di volte.

IL TASTO HELP

Quando in un programma si commette un errore, il Plus/4 visualizza un messaggio di errore per evidenziare cosa si è sbagliato. Questi messaggi di errore sono illustrati più ampiamente nella sezione 4 del prontuario del Plus/4 nella seconda parte di questo manuale.
Per la correzione degli errori, è possibile ottenere un ulteriore ausilio usando il tasto **HELP**. Dopo un messaggio di errore, premere **HELP** per individuare esattamente l'errore. Quando si preme **HELP** viene visualizzata sullo schermo la riga contenente l'errore, mentre l'errore stesso lampeggia. Per esempio:

```
?SYNTAX ERROR IN          Messaggio del Plus/4
LINE 10
(?ERRORE DI SINTASSI
NELLA RIGA 10)

HELP                      È stato premuto HELP

10 PRONT "COMMODORE       Messaggio del Plus/4
COMPUTERS"                con errore lampeggiante
```

CAPITOLO 3

USO DEL SOFTWARE

- Introduzione
- Software incorporato
- Cartucce
- Cassette
- Dischetti

INTRODUZIONE

Il numero dei programmi disponibili per il Plus/4 è in rapida crescita. Presso i fornitori è possibile essere aggiornati sui nuovi prodotti ed avere informazioni sulle caratteristiche del software disponibile.
Il Plus/4 Commodore può utilizzare software su CARTUCCE, CASSETTE e DISCHETTI, tutti disponibili presso i fornitori Commodore. È sufficiente caricarli nel Plus/4. È anche possibile creare e memorizzare programmi propri su cassette o floppy disk.

SOFTWARE INCORPORATO

Il Plus/4 può essere attrezzato con un'ampia varietà di pacchetti di software incorporati per i tasti funzione. Questi programmi sono incorporati nel Plus/4 e vengono attivati premendo l'appropriato tasto funzione. Il Software incorporato nel Plus/4 trasforma il computer in elaboratore testi, database, foglio elettronico (spreadsheet) e macchina grafici. Un pacchetto incorporato è pronto per l'utilizzo all'accensione dell'elaboratore. Quando si accende il Plus/4 il messaggio di schermo comunica quali pacchetti siano disponibili e quale tasto funzione usare per attivarli. È possibile usare il comando KEY anche per visualizzare le definizioni dei tasti funzione. Se un software per tasti funzione è incorporato nel Plus/4, la definizione per KEY 1 sarà: SYS XXXX: nome del pacchetto. Premere il TASTO FUNZIONE 1 e premere RETURN per attivare il programma.

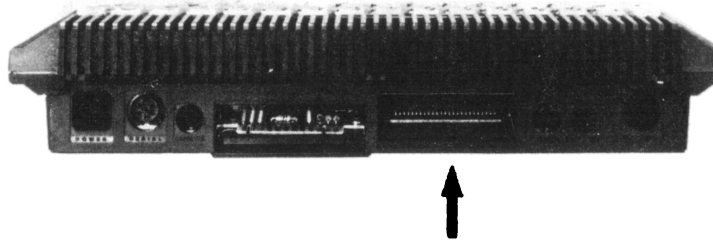
CARTUCCE Come Caricare le Cartucce

La Commodore produce un vasto assortimento di software su cartuccia per il Plus/4. È disponibile un'ampia scelta di programmi personali, didattici e commerciali e di giochi divertenti. Per usare le cartucce, attenersi alle seguenti istruzioni:

1. SPEGNERE il Plus/4

   IMPORTANTE: È ASSOLUTAMENTE NECESSARIO SPEGNERE L'ELABORATORE PRIMA DI INSERIRE O TOGLIERE LE CARTUCCE. IN CASO CONTRARIO, SI RISCHIA DI DANNEGGIARE SIA L'ELABORATORE SIA LA CARTUCCIA.

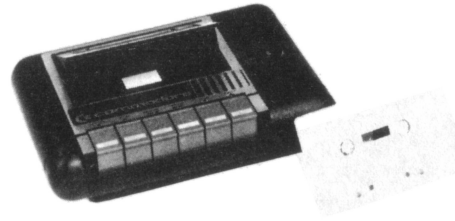
2. Inserire la cartuccia con l'etichetta verso l'ALTO, nella apposita sede sul retro dell'elaboratore.



3. ACCENDERE il Plus/4

4. Avviare il gioco o il programma seguendo le istruzioni allegate al software. Un programma su cartuccia si avvia immediatamente; mentre un software funzionale prende il via dopo aver premuto il tasto funzione.

CASSETTE Come Caricare le Cassette



Un'ampia gamma di prodotti software per il Plus/4 è disponibile su cassetta. Queste cassette sono simili alle musicassette che si ascoltano con registratori normali o stereo. I nastri per l'elaboratore funzionano con i registratori Datassette, disponibili presso i fornitori Commodore. Le cassette e il Datassette si possono anche utilizzare per memorizzare programmi scritti dall'utente. La sezione seguente spiegherà come salvare programmi su nastro.
Le operazioni necessarie per caricare un nastro sono le stesse, sia che si usi un software pre-registrato, sia che si usino programmi memorizzati dall'utente.

1. Inserire la cassetta nel Datassette e chiudere lo sportellino.

2. Riavvolgere il nastro all'inizio premendo sul Datassette il bottone REWIND.

3. Quando il nastro è stato completamente riavvolto, battere LOAD e premere il tasto **RETURN**, l'elaboratore risponde con il seguente messaggio:
   PRESS PLAY ON TAPE

4. Premere il bottone PLAY sul Datassette. Lo schermo si cancella quando si avvia il Datassette. Quando viene rintracciato un programma, lo schermo visualizza il messaggio:
   FOUND "nome del programma"

5. Premere il tasto Commodore per caricare il programma rintracciato. Se sul nastro si trova più di un programma e il programma rintracciato dal Plus/4 non è quello desiderato, premere la barra spaziatrice per continuare la ricerca.

NOTA: Per caricare sul nastro un programma specifico, usare il formato LOAD "nome del programma" del comando LOAD.

Quando il programma è caricato, appare la parola READY. Se si desidera arrestare il caricamento prima che sia completo, premere il tasto **RUN/STOP**. Dopo che il software è caricato, battere RUN per avviare il programma. È anche possibile listare il programma o modificarlo, se è un programma BASIC.

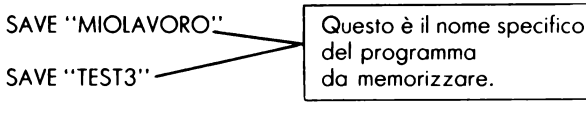
Come Memorizzare i Programmi su Cassetta

Se si scrive un programma e lo si vuole memorizzare su cassetta, seguire le seguenti istruzioni:

1. Battere:
   SAVE "nome del programma"
   Il nome del programma può essere un nome qualsiasi, tuttavia non può essere più lungo di 16 caratteri alfanumerici.

2. Premere il tasto **RETURN**, l'elaboratore visualizza il seguente messaggio:
   PRESS RECORD AND PLAY ON TAPE

3. Premere i tasti RECORD e PLAY sul Datassette. Lo schermo si cancella. Quando viene memorizzato il programma, compare sullo schermo la parola READY.
   Esempi di comandi SAVE per cassette:



NOTA: Quando si memorizza su cassetta, accertarsi sempre della posizione del nastro. In particolare, fare attenzione a non memorizzare un programma all'inizio esatto della cassetta, poiché molti nastri hanno una coda che non registra informazioni per cui una parte del programma andrebbe perduta.

Quando si carica o si memorizza un programma (comandi LOAD e SAVE) se si desidera interrompere l'operazione in corso prima del suo termine, premere il tasto RUN/STOP sulla tastiera e quindi il pulsante STOP del Datassette.

DISCHETTI Come Caricare dei Programmi da un Dischetto

Usare i dischi è semplice e veloce. I dischi e l'unità a disco vanno maneggiati con attenzione. I dischi possono essere anche chiamati, indifferentemente, dischetti, floppy disk o floppy.
Le seguenti istruzioni valgono per tutti i dischi.

1. Accertarsi che l'unità a disco sia accesa

2. Inserire il disco nel drive. Il lato del disco con l'etichetta deve essere rivolto verso l'alto. Inserire completamente il disco nell'apertura, ma in modo che l'estremità etichettata rimanga verso l'esterno. Su un lato del disco si trova una piccola tacca (può darsi che sia coperta da un adesivo). Mentre si inserisce il disco la tacca dovrebbe trovarsi alla sinistra di chi guarda l'unità a disco. Accertarsi che il disco sia completamente inserito.

3. Dopo aver inserito il disco chiudere lo sportellino di protezione del drive.

4. Battere:

DLOAD "nome del programma"

nome specifico del programma da caricare.

Per risparmiare tempo si può premere il TASTO FUNZIONE 2 e battere il nome del programma e le virgolette di chiusura.

5. Premere il tasto RETURN. Il disco inizia a girare e sullo schermo compare:

   SEARCHING FOR PROGRAM NAME (ricerca del nome del programma)

   LOADING (caricamento)

   READY

6. Il software è ora pronto per l'uso. Battere RUN e premere il tasto RETURN per avviare il programma. Se dopo che si è concluso il caricamento la spia rossa del drive comincia a lampeggiare, è segno che ci sono dei problemi.
   Battere:

   ?DS$ (e premere RETURN)
   per scoprire che problemi ci sono.

Esempi di comandi DLOAD:

DLOAD "*"	Carica il primo programma del disco
DLOAD "FILE"	Carica un programma su disco denominato FILE
DLOAD "SET*"	Carica il primo programma del disco che cominci con le lettere SET.
DLOAD "$"	Carica l'indice di tutti i programmai presenti sul disco nel drive.

Come Preparare un Dischetto con il Comando Header

Questa operazione prepara per l'uso un disco VERGINE. Qualsiasi disco vergine deve essere formattato prima dell'uso, servendosi del comando HEADER.

IMPORTANTE: NON FORMATTARE UN DISCO CONTENENTE DATI, A MENO CHE NON SI DESIDERI CANCELLARE L'INTERO DISCO. LA FORMATTAZIONE CANCELLA TUTTI I DATI PRESENTI SU UN DISCO.

Il formato per il comando HEADER è:

HEADER "nome disco", Udispositivo #, Ii.d. #, Ddrive #

- "Nome disco" sarà il nome dell'intero disco. È possibile assegnare qualsiasi nome che abbia fino a 16 caratteri.

- "dispositivo #" specifica il dispositivo usato per l'elaboratore (unità a disco o Datassette) ed è solitamente il numero 8.

- "i.d." è la lettera I e due caratteri alfanumerici qualsiasi, come I21, IR5 ecc. È possibile assegnare al disco qualsiasi i.d. si voglia, ma, per non creare confusione è consigliabile attribuire ad ogni disco un i.d. diverso.

- Se si è in possesso di un'unità a disco doppia aggiungere D0 o D1 per identificare il numero del drive.

CONFERMA

Non appena si preme **RETURN** dopo aver battuto il comando HEADER, il Plus/4 chiede ARE YOU SURE? Questo consente di cambiare decisione all'ultimo momento.
Per formattare il disco, battere YES o Y e premere **RETURN**. Se si decide di non formattare il disco, battere NO o N e premere **RETURN**

Alcuni esempi di comandi HEADER:

HEADER "LETTERE",U8,I07,D0

HEADER "FINANZE",U8,IS3,D0

Una volta appreso come formattare un disco, si è pronti ad usare i dischi per scrivere e memorizzare programmi sul Plus/4. La prima parte del prontuario Plus/4 contiene ulteriori informazioni riguardo al comando HEADER.

Come Memorizzare Programmi su Dischetto

Quando si desidera riutilizzare un programma che si è scritto, assicurarsi di memorizzarlo prima di caricare un altro programma o spegnere il Plus/4. Se non lo si fa, il programma andrà perduto. Quando si modifica un programma memorizzato, occorre memorizzarlo di nuovo se si vuole conservare la nuova versione. Quando si rimemorizza un programma, si sostituisce la vecchia versione con la nuova. Se si desidera conservare entrambe le versioni (sia la vecchia sia quella modificata), occorre assegnare a quella nuova un nome diverso quando si memorizza.
Per memorizzare un programma su disco, seguire le seguenti istruzioni:

1. Battere DSAVE "nome del programma"

2. Premere **RETURN**. Quando il programma viene memorizzato lo elaboratore visualizza il seguente messaggio:

   SAVING "nome del programma"
   OK
   READY.
   ■

Esempio:
DSAVE "PROG5"

Il nome del programma può essere composto da un massimo di 16 caratteri.

Se, dopo che la memorizzazione si è conclusa, la spia rossa del drive comincia a lampeggiare, è segno che c'è qualche problema. Battere:

?DS$ (e premere **RETURN**)

per sapere qual è il problema.

IL COMANDO DIRECTORY

Quando si memorizzano programmi su disco, l'elaboratore conserva una lista di tutti i "file" memorizzati su quel disco.
Usando il comando directory è possibile visualizzare la lista, sotto forma di indice per sapere cosa contiene un disco.

Battere: DIRECTORY quindi premere RETURN
(oppure premere il TASTO FUNZIONE 3)
Non appena si preme RETURN, il Plus/4 visualizza l'intero contenuto del disco.
È anche possibile visualizzare solo una parte dell'indice:

DIRECTORY "MY*" RETURN Elenca tutti i file del disco che iniziano con le lettere MY.

CAPITOLO 4

AVVIAMENTO

- Introduzione
- Colori della tastiera
- Stampa a colori e invertita
- Alcuni semplici programmi
- Correzione degli errori di battitura
- Introduzione allo schermo di testo del Plus/4
- Ulteriori informazioni sulla stampa su schermo
- Finestre dello schermo

INTRODUZIONE Lo scopo di questo capitolo è di consentire all'utente di cominciare ad acquistare confidenza con alcune delle caratteristiche e capacità del Plus/4, e con la programmazione dell'elaboratore di cui si dispone.

COLORI DELLA TASTIERA

Per migliorarne la leggibilità o per trovare una combinazione di colori di proprio gradimento, è possibile modificare il colore dei caratteri sullo schermo. Per constatare la resa sullo schermo dei diversi caratteri colorati:

1. Tenere premuto il tasto CONTROL
2. Premere il tasto 6 mentre si tiene premuto il tasto CONTROL. Il cursore diventa verde.
3. Rilasciare CONTROL e battere alcune lettere. Ora tutto ciò che si batte appare verde.

TASTO COLORE PREMUTO CON CONTROL	COLORE RISULTANTE
1	NERO
2	BIANCO
3	ROSSO
4	CIANO
5	VIOLA
6	VERDE
7	AZZURRO
8	GIALLO

L'uso del tasto CONTROL con i tasti numerati dall'1 all'8 consente di scegliere i colori che appaiono sulla parte superiore di ogni tasto colore.
Ora tenere premuto il tasto α. Battendo i tasti tra l'1 e l'8, il cursore appare di uno degli 8 colori stampati sulla parte inferiore di ogni tasto colore. Tutti e 16 i colori possono comparire sullo schermo contemporaneamente.

TASTO COLORE PREMUTO CON C=	COLORE RISULTANTE
1	ARANCIONE
2	MARRONE
3	VERDEGIALLO
4	ROSA
5	VERDAZZURRO
6	CELESTE
7	BLU NOTTE
8	VERDE CHIARO

STAMPA A COLORI E INVERTITA

Il Plus/4 può visualizzare numeri, lettere e simboli grafici nei 16 diversi colori. È possibile anche visualizzare questi caratteri con i colori di primo piano (cursore) e di sfondo invertiti.

1. Cancellare lo schermo premendo SHIFT e CLR/HOME
2. Premere contemporaneamente i tasti CONTROL e RVS ON:

3. Rilasciare i tasti e tenere premuta la barra spaziatrice (la lunga barra nella parte inferiore della tastiera).

4. Tenere la barra premuta quanto tempo si vuole. Mentre la si tiene premuta, una linea dello stesso colore delle lettere dello schermo si allunga a mano a mano. Quando la linea arriva al termine della riga, passa alla riga successiva.

5. Rilasciare la barra spaziatrice (ma non premere il tasto RETURN).

6. Tenere premuto il tasto CONTROL e premere uno dei tasti colore (di un colore che non risulti già presente sullo schermo). Non appena si esegue questa operazione, il cursore diventerà del colore del tasto premuto.

7. Tenere di nuovo premuta la barra spaziatrice, ora il Plus/4 traccerà una linea del nuovo colore. Continuare a cambiare i colori con i tasti CONTROL o C= e con i tasti colore. Quindi tenere premuta la barra spaziatrice per formare linee di colori differenti.

8. Disattivare la stampa invertita tenendo premuti contemporaneamente i tasti CONTROL e RVS OFF. Anche premendo il tasto RETURN si disattiva la stampa invertita.

Provare a battere alcune lettere invertite. Tenere premuti CONTROL e RVS ON per attivare l'inversione, quindi battere ciò che si vuole. Le lettere invertite sono particolarmente efficaci, se usate per titoli. Possono anche essere usate per evidenziare particolari parole e numeri. Operare come segue:

Ora operare allo stesso modo, ma sostituendo reverse on e off con FLASH ON e OFF:

Entrambe le funzioni possono essere usate sul Plus/4 anche come parte di una istruzione di programma.

ALCUNI SEMPLICI PROGRAMMI PLUS/4

Battere questo programma esattamente come compare qui. Accertarsi di non tralasciare i numeri all'inizio della riga, dal momento che questi comunicano l'ordine in cui le righe del programma devono essere implementate dall'elaboratore. Accertarsi di premere **RETURN** al termine di ogni riga che si batte.

```
10 PRINT "PLUS/4"
```
Questa riga ordina al computer di stampare sullo schermo Plus/4

```
20 GOTO 10
```
Questa riga ordina al computer di ritornare alla riga 10 e quindi di ristampare Plus/4

```
RUN
```
Questo ordina al computer di eseguire i comandi delle due righe precedenti.

Premere il tasto **RUN/STOP** per arrestare il programma.
Il motivo per cui il Plus/4 stampa il suo nome così tante volte, è perché GOTO ordina all'elaboratore di tornare indietro alla riga 10 e di stampare ripetutamente Plus/4. Questa ripetizione è chiamata loop.
Ora battere:

```
NEW RETURN
```
Questo ordina al computer di cancellare l'ultimo programma e di prepararsi per uno nuovo.
Il computer risponde:

```
READY.
```
Questo non è da battere: lo visualizza il Plus/4 per comunicare che è pronto per un nuovo programma.

```
10 PRINT "PLUS/4"
```
La stessa riga 10 come in precedenza. PRINT ordina al Plus/4 di visualizzare ciò che è inserito tra le virgolette.

```
20 COLOR 0,12
```
Questo comando ordina all'elaboratore di cambiare il colore dello schermo.

RUN

Questa volta non c'è un loop GOTO nel programma, così gli ordini impartiti vengono eseguiti una sola volta.

CORREZIONE DEGLI ERRORI DI BATTITURA

Esistono numerosi modi di correggere gli errori di battitura.

1. SI PUÒ RIBATTERE UNA RIGA in qualsiasi momento, anche dopo aver eseguito un programma. Il Plus/4 sostituisce automaticamente la vecchia riga con la nuova dopo aver premuto **RETURN** per inserire la nuova riga. La vecchia riga rimane ancora sullo schermo, ma il Plus/4 la ignora. Avendo due righe di programma con lo stesso numero di riga, il Plus/4 utilizza solo l'ultima introdotta. Ad esempio, se in un breve programma si commette un errore utilizzando il comando COLOR per cambiare il colore dello sfondo dello schermo:

```
10 COKOR 0,3 ———— errore
20 PRINT "PLUS/4"
```

Bisogna premere il tasto **RETURN** per ottenere una nuova riga e poi ribattere correttamente la riga 10

```
10 COLOR 0,3          RETURN
```

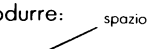
Ora la prima riga 10 è stata sostituita dalla seconda riga 10. Ciò è verificabile premendo LIST, che visualizza un listato del programma riga per riga, così come è stato introdotto nella memoria del computer. Dopo un'istruzione LIST, tutte le righe del programma appaiono in ordine esatto, senza le righe sostituite:

```
LIST                  RETURN
```

Sullo schermo si vedrà:

```
10 COLOR 0,3
20 PRINT "PLUS/4"
```

La sostituzione di righe in un programma è anche un buon modo per provare il proprio computer.
Per sostituire una riga, la nuova non deve essere necessariamente simile alla vecchia. Ad esempio, invece di correggere l'errore di battitura in COLOR, si può introdurre:



10 PRINT "TEN IS SIX " **RETURN**

Ora LANCIARE il programma e vedere cosa succede.

2. SI PUÒ CANCELLARE UNA RIGA NON VOLUTA semplicemente battendo il numero della riga e poi **RETURN**. Il computer ignora la riga anche se ancora appare sullo schermo. Battere LIST per il listato del programma per avere la certezza che la riga non è più nel programma.

10 PRINT "TEN IS SIX" **RETURN**

20 PRINT "PLUS/4" **RETURN**

10 **RETURN**

LIST **RETURN**

20 PRINT "PLUS/4"

3. CORREZIONE DI UNA RIGA. Utilizzare i tasti del cursore per spostarsi sul punto della riga che si vuol cambiare. Ora basta ribattere ciò che si vuole cambiare. Quindi premere **RETURN**

NOTA: Lavorando con un programma a righe numerate, non è necessario arrivare alla fine della riga per premere **RETURN**. Il Plus/4 ricorda l'intera riga anche se si preme **RETURN** a metà riga.

10 PRINT "SONO LE DUE"

Se si vuole cambiare l'ora alle tre, spostare il cursore sulla D di DUE

10 PRINT "SONO LE/D/UE"

Ora ribattere TRE su DUE e premere **RETURN**

10 PRINT "SONO LE TRE" **RETURN**

NOTA: Quando si battono le virgolette dopo l'istruzione PRINT, si inserisce la MODALITÀ QUOTE. In questa modalità alcuni tasti hanno una funzione diversa. Ad esempio, se si preme il tasto del cursore non si muoverà, si vedrà invece sullo schermo una Q invertita. Quando si esegue l'istruzione PRINT, la Q invertita non viene stampata, al contrario, il cursore si muove verso il basso. Nella modalità quote, il computer assume che tutto quello che viene battuto, sarà poi da visualizzare o fare quando si eseguirà l'istruzione PRINT.

4. INSERIMENTO DI SPAZI IN UNA PAROLA O IN UNA RIGA con il tasto **INST** (si ottiene premendo contemporaneamente **SHIFT** e **INST/DEL**). Tenere premuti i tasti finché non si sono inseriti tutti gli spazi desiderati. (Si noti che il cursore rimane nello stesso punto mentre gli spazi vengono inseriti dalla posizione del cursore verso destra). Quindi battere ciò che si vuole inserire.

10 PRINT "CORE" **RETURN**

Per cambiare questa parola abbreviata in COMMODORE Plus/4, portare il cursore sulla O, quindi premere i tasti **SHIFT** e **INST** fino ad avere spazio a sufficienza. Non è necessario contare gli spazi, se non sono sufficienti, se ne inseriscono altri.

10 PRINT "C ORE"
cursore

Ora aggiungere le altre lettere:

10 PRINT "COMMODORE" **RETURN**

5. CANCELLAZIONE DI CARATTERI ED ELIMINAZIONE DI SPAZI col tasto **DEL** (si ottiene premendo **INST/DEL**). Questo tasto cancella i caratteri o gli spazi che si trovano immediatamente alla SINISTRA del cursore.

10 PRINT "PROGRAMMA POMERIDIANO" **RETURN**

Si può cambiare in PROGRAMMA SETTIMANALE spostando il cursore sulla E di POMERIDIANO, premendo tre volte il tasto **INST/DEL** e quindi battendo SETTIMANALE".

10 PRINT "PROGRAMMA POM/E/RIDIANO" **RETURN**

e premere 3 volte INST/DEL

10 PRINT "PROGRAMMA /E/RIDIANO"

Battere SETTIMANALE" per sostituire ERIDIANO e premere RETURN

Esempio di Programma

Dopo essersi esercitati col Plus/4, si può ora provare un programma che richiederà un po' più di tempo per la battitura (qualsiasi programmatore esperto vi potrà dire che se non altro la programmazione migliorerà la vostra dattilografia). Per primo, ripulire lo schermo premendo contemporaneamente i tasti SHIFT e CLR/HOME. Poi, cancellare tutti i vecchi programmi dalla memoria battendo NEW e premendo RETURN.
Inserire questo programma esattamente come appare. Ricordare di inserire i numeri delle righe e tutti i segni di punteggiatura.
Ricordare i consigli per correggere eventuali errori di battitura. Non dimenticarsi di premere RETURN alla fine di ogni riga.

NOTA: il programma si può interrompere premendo il testo RUN/STOP.

```
NEW

10 COLOR 1,8

20 PRINT "È SUCCESSA UNA COSA BUFFA ";

30 COLOR 1,3,1

40 PRINT "MENTRE ANDAVO ALLA TASTIERA" ;

50 COLOR 1,7,1

60 PRINT " ♥♥♥♥♥

70 GOTO 60
```

Assicurarsi di aver lasciato uno uno spazio qui

I cuori si ottengono premendo contemporaneamente i tasti SHIFT e S 6 volte.

RUN

Dopo aver interrotto il programma (premendo il tasto **RUN/STOP**), provare a battere LIST. Dopo che il programma è stato visualizzato sullo schermo, ricordarsi i consigli per la correzione degli errori e provare a cambiare questo programma facendogli scrivere qualcosa di più significativo.

CONSIGLIO: per rallentare questo programma senza interromperlo, basta tenere premuto il tasto **C=** .

SCHERMO DI TESTO DEL PLUS/4

Provare a battere questo programma (ricordare di premere RETURN dopo l'inserimento di ogni riga).

```
NEW
10 PRINT " ♥";
20 GOTO 10
```

premere **SHIFT** e S

RUN

Ora lo schermo si riempie di cuori. Dopo che l'intero schermo è ricoperto di cuori, premere il tasto **RUN/STOP** per terminare il programma. Questo programma mostra l'ampiezza dello schermo del Plus/4.

Battere questo programma:

```
NEW
10 PRINT "■"
20 FOR X = 1 TO 40
30 PRINT " ♥";
40 NEXT X
```

premere **SHIFT** e **CLR/HOME**

premere **SHIFT** e S

RUN

Quando si avvia questo programma, la prima riga dello schermo si riempie completamente di cuori (40). A riga piena, si può notare che ci sono 40 posizioni attraverso lo schermo. Queste posizioni sono dette COLONNE.

Battere ora questo programma:

```
NEW
10 PRINT "♥"
20 FOR X = 1 TO 25
30 PRINT "♦";
40 NEXT X
RUN
```

premere SHIFT e CLR/HOME

premere SHIFT e Z

Quando si avvia questo programma, la prima riga dello schermo si riempie di rombi. I rombi stampati sono 25, ma i primi tre scompaiono dalla parte superiore dello schermo dato che alla fine del programma appare sempre la parola READY, inquadrata tra due linee vuote. Ci sono, quindi, 25 righe. È logico dedurre che il Plus/4 ha 40 colonne e 25 righe. Il Plus/4 ha 1000 diverse posizioni sullo schermo relative alle lettere, ai numeri, ai simboli grafici, ecc.

NOTA: a volte si deve battere sul Plus/4 una riga particolarmente lunga come:

10 PRINT "MI PIACE COME USI LA MIA TASTIERA. VIENI QUI SPESSO?"
(È una riga che supera i 50 caratteri!)

Si noterà che battendo questa riga si esce dallo spazio a disposizione. Ma si può continuare a battere perché il Plus/4 va automaticamente a capo e continua a stampare fino alla fine della riga. È possibile battere fino a 80 caratteri su una riga di programma (fino a due righe piene). Provare ora a LANCIARE questo programma di una riga. Il messaggio viene stampato su due righe. Se la frase è più lunga di una riga, il Plus/4 la lascia proseguire sulla riga successiva. Il Plus/4 considera terminata la riga quando si preme il tasto RETURN, non quando si arriva alla fine della riga.
Ci si abituerà a questo esercitandosi col Plus/4.
Battere questo programma:

```
NEW
10 PRINT " ♥ ";
```

lasciare uno spazio ai lati del cuore

Premere SHIFT e S

```
20 GOTO 10

RUN
```

Lanciando questo programma si può notare che è possibile dire esattamente al Plus/4 dove stampare qualcosa sullo schermo.

ULTERIORI INFORMAZIONI RELATIVE ALLA STAMPA SU SCHERMO

Provare a battere questo programma:

```
NEW
10 PRINT "A", "B"
20 PRINT "A"; "B"
RUN
```

Ecco come appare lo schermo:

```
A          B ---- Stampato dalla riga 10
AB ------------- Stampato dalla riga 20
```

La differenza di stampa sullo schermo, nonostante la riga 10 e la riga 20 appaiano praticamente identiche, dipende dalla punteggiatura tra gli elementi stampati da questo programma. Utilizzando una virgola per separare gli elementi dopo una istruzione PRINT, gli elementi vengono stampati a diversi spazi di distanza l'uno dall'altro.
Utilizzando un punto e virgola, gli elementi vengono stampati uno vicino all'altro.
Come si ricorderà, lo schermo del Plus/4 ha 40 colonne orizzontali. Tali colonne sono divise in 4 aree di 10 spazi, chiamate aree di stampa. Quando si usa la virgola per separare gli elementi stampati, il Plus/4 stampa il primo elemento nella prima area di stampa, il secondo elemento nella seconda area di stampa, ecc. Le virgole funzionano come i tabulatori delle macchine per scrivere.

```
PRINT 1          PRINT ZONE 2          PRINT ZONE 3          PRINT ZONE 4
1 2 3 4 5 6 7 8 9 10  11 12 13 14 15 16 17 18 19 2021 22 23 24 25 26 27 28 29 3031 32 33 34 35 36 37 38 39 40
A                B
```

Per stampare più di quattro elementi separati da virgole, il Plus/4 stampa automaticamente sulla riga successiva. Ad esempio:

```
PRINT"A","B","C","D","E","F"
```

Spazia le lettere sullo schermo in questo modo:

```
              1   11   21   31        COLONNA
RIGA 1        A    B    C    D
     2        E    F
```

Quando si usano i punti e virgola per separare gli elementi dopo un'istruzione PRINT, il Plus/4 ignora le aree di stampa e stampa tutti gli elementi in susseguenza.

```
PRINT"A";"B";"C";"D";"E";"F"
```

la STAMPA sarà:

```
ABCDEF
```

Stampando un primo elemento lungo 12 lettere e il secondo elemento separato da una virgola, si avrà:

```
PRINT"ABCDEFGHIJKL","M"
```

e la STAMPA sarà:

```
ABCDEFGHIJKL          M

area di               area di          area di
stampa 1              stampa 2         stampa 3
```

Ripulire lo schermo e battere questo programma:

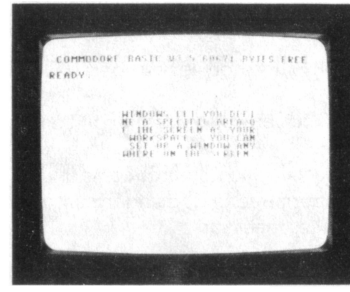
```
NEW
10 PRINT 1,2
20 PRINT 1;2
RUN
1    2
1 2
```

Questo programma mostra due novità:

1. I numeri non necessitano delle virgolette in un'istruzione PRINT.

2. I numeri sono visualizzati con una spaziatura ad entrambi i lati, perciò, usando un punto e virgola, i numeri non vengono stampati uno vicino all'altro, come invece accade con le lettere. Lo spazio permette di aggiungere un segno meno davanti a numeri negativi (se necessari).

FINESTRE DELLO SCHERMO



Le finestre permettono di definire sullo schermo un'area specifica di lavoro. Tutto ciò che viene battuto sulla tastiera (righe, listato dei programmi, ecc.) dopo aver inserito la finestra appare all'interno dei bordi della stessa, senza peraltro influire sullo schermo al di fuori dell'area della finestra. Si può impostare una finestra in qualsiasi zona dello schermo. Per impostare una finestra:

1. Spostare il cursore alla posizione desiderata sullo schermo per indicare l'angolo superiore sinistro della finestra.
2. Premere il tasto ESC e poi la lettera T
3. Spostare il cursore alla posizione in cui si desidera l'angolo inferiore destro della finestra.
4. Premere ESC e poi B. La finestra è impostata.

Tutto l'output dello schermo è racchiuso nel riquadro appena impostato. Premendo due volte il tasto HOME, si cancella la finestra e il cursore si posiziona nell'angolo superiore sinistro dello schermo.

CAPITOLO 5

NUMERI E OPERAZIONI ARITMETICHE

- Numeri e operatori fondamentali
- Esecuzione di operazioni
- Uso delle variabili
- Modalità immediata
- Funzioni numeriche
- Numeri casuali ed altre funzioni

NUMERI E OPERATORI FONDAMENTALI

Il Plus/4 si può utilizzare come una semplice calcolatrice. Oltre ai segni standard + e –, il Plus/4 utilizza anche il segno * per le moltiplicazioni e il segno / per divisioni e frazioni (i computer utilizzano il segno * invece delle x per le moltiplicazioni perché un computer non distingue la differenza tra la lettera x e il simbolo matematico x). Operatori e numeri si possono usare in modo diretto (senza righe numerate) o in un programma. Nell'esecuzione di operazioni matematiche col Plus/4, sia i numeri sia gli operatori non vanno messi tra virgolette.

OPERATORI MATEMATICI FONDAMENTALI		OPERATORI RELAZIONALI FONDAMENTALI	
Addizione	+	Maggiore di	>
Sottrazione	–	Minore di	<
Divisione e frazioni	/	Uguale	=
Moltiplicazione	*	Maggiore di uguale	= >
Esponenziali	↑	Minore di uguale	< =
		Diverso da	<> op. ><

NOTA: Il Plus/4 non accetta le virgole come parti di un numero, ad esempio, si deve battere 109.401 invece di 109,401, perché se si inserisce la virgola nel numero per separare i decimali dal numero intero, il Plus/4 interpreta la virgola come separatore di due numeri interi, leggendo quindi 109 e 401 invece di 109401.

Inoltre, non è possibile utilizzare il punto per separare le migliaia, dato che questo verrebbe interpretato come una virgola decimale, per cui 1.892.749 si scriverà 1892749.

FRAZIONI E DECIMALI

Una frazione si può scrivere : .5
oppure : 1/2 — Il Plus/4 sta effettivamente eseguendo la divisione

Se si inserisce una frazione in una istruzione PRINT, la risposta è sempre data in decimale o come numero intero.
Ad esempio:

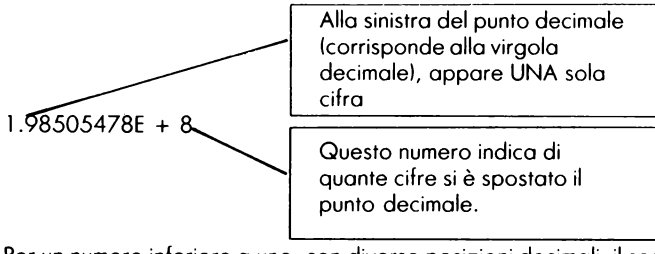
```
PRINT 139/493 + 5                    RETURN
5.28194726
```

pi greco (3.14159256...), rappresenta il rapporto tra la circonferenza di un cerchio e il suo diametro. Questo valore si ottiene premendo il tasto π ad esempio:

```
PRINTπ/374                           RETURN
8.39998036E-03
```

NOTAZIONE SCIENTIFICA

Ci si può chiedere cosa significhi E-03 nella risposta del Plus/4. Il Plus/4 visualizza sotto forma di numeri standard, i numeri decimali compresi fra -999.999.999 e 999.999.999. I numeri con più di 9 cifre vengono automaticamente visualizzati sotto forma di notazione scientifica. È possibile introdurre questi numeri già in questa forma, dal momento che il Plus/4 può leggerli senza difficoltà (in ogni caso con minori difficoltà di quelle che incontrerebbe l'utente convertendoli). La notazione scientifica è utile in molti casi, poiché consente al Plus/4 di visualizzare grandi numeri con poche cifre.
Se il numero 198.505.478 fosse scritto in notazione scientifica, apparirebbe così:



Per un numero inferiore a uno, con diverse posizioni decimali, il secondo numero sarà preceduto da un – anziché da un +, indicando che il punto decimale si è spostato a destra.

Per esempio:

.0003359 = 3.359E - 4

Altri esempi:

20 = 2E + 1	Il punto decimale si è spostato a sinistra di 1 cifra
105000 = 1.05E + 5	Il punto decimale si è spostato a sinistra di 5 cifre
.0666 = 6.66E - 2	Il punto decimale si è spostato a destra di 2 cifre.

ESECUZIONE DI OPERAZIONI

Per eseguire un'operazione, battere PRINT e quindi l'operazione matematica. Ricordarsi di non porre fra virgolette l'operazione. Battere questo programma:

```
NEW
10 PRINT 1+2, 2-1
20 PRINT 2*2, 4/2
RUN
3             1
4             2
```

Usare la linea di frazione sul tasto "?"

Per la prima volta PRINT non ha stampato esattamente ciò che si è battuto nell'istruzione. Il Plus/4, invece, ha risolto i calcoli e stampato i risultati. Tutto ciò che è necessario per usare PRINT per eseguire calcoli, è di omettere le virgolette. Ora:

```
NEW
10 PRINT "2001/2010"
20 PRINT 2*3
RUN
2001/2010
 6
```

In questo caso viene lasciato uno spazio per il segno del risultato.

Poiché il calcolo nella riga 10 è fra virgolette, il Plus/4 stampa l'operazione come se fosse un testo normale: esattamente come appare fra virgolette l'operazione non viene eseguita e non viene lasciato uno spazio per il segno del numero.
Ora, spostare il cursore alla riga 10 e modificarla:

```
10 PRINT "2*3+1=";2*3+1
```

Non dimenticare il punto e virgola

```
RUN
2*3-1 = 7
 6
```

Questo spazio è lasciato per il segno del risultato.

Il risultato della riga 20 rimane invariato.

Se si desidera sia stampare l'operazione, sia eseguirla, occorre battere due volte: una volta fra virgolette ed una volta senza virgolette, come segue:

```
10 PRINT "2+2="; 2+2
```

MODALITÀ IMMEDIATA

È possibile inserire in un programma qualsiasi calcolo od avere una risposta immediata battendo PRINT e l'operazione senza il numero della riga, premendo quindi RETURN, come segue:

```
PRINT 3-6
-3
PRINT 24/(6+2)
 3
```

Sia con i numeri, sia con i comandi ed il testo, quando non c'è un numero di riga prima di un'istruzione BASIC, non è necessario battere RUN per ordinare al computer di seguire l'istruzione; questa viene chiamata MODALITÀ DIRETTA, o IMMEDIATA.
La presenza di un numero di riga indica che l'istruzione è parte di un programma BASIC; questa viene definita MODALITÀ PROGRAMMA.
Entrambe le modalità vengono accettate.
Nella modalità immediata è anche possibile includere sia un istruzione di testo fra virgolette, sia un'operazione matematica da eseguire in una singola istruzione PRINT.

```
PRINT "2 ALLA TERZA
UGUALE";2↑3
```

Questo simbolo significa esponenziale; si ottiene battendo SHIFT e 0.

```
2 ALLA TERZA UGUALE 8
```

Viene stampato il messaggio e quindi il risultato dell'operazione.

ORDINE DI CALCOLO Il secondo esempio nell'ultima sezione dimostra che in una sola riga è possibile eseguire più di un calcolo. Battere:

PRINT 200+50/5

Il risultato è giusto? Battere:

PRINT (200+50)/5

Il Plus/4 esegue sempre i calcoli in un certo ordine. Le operazioni vengono eseguite da sinistra verso destra; all'interno di questa regola generale, alcuni tipi di calcolo vengono risolti per primi. L'ordine secondo il quale il Plus/4 valuta le espressioni viene definito l'ordine di precedenza dei simboli operatori.

PRIMO: Il Plus/4 considera i numeri negativi (non le sottrazioni, ma i numeri negativi).

SECONDO: Il Plus/4 esegue gli elevamenti a potenza.

TERZO: Il Plus/4 esegue tutte le moltiplicazioni e le divisioni, da sinistra verso destra.

QUARTO: Il Plus/4 esegue le addizioni e le sottrazioni, da sinistra verso destra.

NOTA: Il Plus/4 esegue per prime tutte le operazioni fra parentesi. È possibile anche porre delle parentesi all'interno di altre parentesi: 36*(12+(A/3)). Le operazioni nelle parentesi interne vengono eseguite per prime.

Talvolta può essere consigliabile per una maggior chiarezza porre i numeri negativi fra parentesi. Per esempio, se si vuole moltiplicare 45 per -5, battere in questo modo: 45*(-5). Il Plus/4 lo recepirà in ogni caso, anche senza parentesi.

USO DELLE VARIABILI

L'esempio 36*(12+(A/3)) mostra una delle caratteristiche più efficaci di un elaboratore. Quando in una operazione matematica si usa una lettera invece di un numero, si è usata una VARIABILE. Una variabile rappresenta un valore:

```
10 A = 3
20 PRINT "TOTAL:"; A*4
```

Se si esegue questo programma, sullo schermo risulterà:

```
TOTAL: 12
```

È possibile usare tre tipi diversi di variabile:

TIPO	SIMBOLO	DESCRIZIONE	ESEMPI	VALORI CAMPIONE
Virgola mobile		numeri reali (decimali) o interi	X,AB,T4	23.5, 12, 1.3E+2
numero intero	%	numeri interi	X%,AI%	15, 102, 3
stringa di testo	$	lettere, numeri, e tutti gli altri caratteri fra virgolette	X$,MS$	"TOTAL:", "DAY 1", "CBM"

Ogni volta che si desidera che una variabile sia intera, il simbolo di quella variabile dovrebbe includere il segno %. Una variabile che contiene testo DEVE terminare con un $ come parte della variabile stessa. Se non reca quel simbolo, il Plus/4 lo considera un numero a virgola mobile. Una variabile priva di entrambi i simboli (% o $) viene considerata come un numero a virgola mobile (un numero "regolare"). Le variabili intere sono un sottoinsieme delle variabili a virgola mobile; sono numeri senza cifre decimali.

Servirsi sempre del tipo corretto di variabile. Se si cerca, per esempio, di assegnare una parola a una variabile intera, il programma non funzionerà. Questo programma mostra quali possano o non possano essere usate in una certa situazione e si potrà scoprire personalmente cosa accade usando diversi tipi di dati:

```
10 REM QUESTO PROGRAMMA RICHIEDE DATI NUMERICI
20 PRINT "INTRODURRE UN NUMERO"
30 INPUT X%
40 PRINT "SEMPRE COSÌ, CAMPIONE!"
50 PRINT "HO LETTO IL TUO NUMERO COME"; X%
```

Questa è la variabile da introdurre.

Cercare di introdurre questi valori e attendere il risultato:

```
ONE FIFTH
.043
10
```

FUNZIONI NUMERICHE

Nel linguaggio BASIC 3.5 del Plus/4 sono incluse funzioni numeriche, che equivalgono ai calcoli di aritmetica superiori possibili con la maggior parte dei calcolatori scientifici (seno, coseno, tangente, ecc.).
Gran parte delle funzioni può essere utilizzata battendo il nome della funzione e, fra parentesi, il numero su cui eseguire l'operazione, come segue:

FUNZIONE (X)

Per esempio, per trovare il seno di una variabile, si batte:

*PRINT SIN (X)

con X che può essere un numero qualsiasi.

È anche possibile includere una delle funzioni in una riga di programma, come mostrato nell'esempio seguente:

```
10 FOR X = 1 TO 5
20 PRINT "LA RADICE QUADRATA DI"; X;"È" ;SQR(X)
30 NEXT X
```

Una lista completa delle funzioni numeriche si trova nel prontuario BASIC 3.5. Alcune delle funzioni più complesse sono illustrate nei paragrafi seguenti.

NUMERI CASUALI ED ALTRE FUNZIONI

Selezionare un numero casuale equivale a numerare 10 foglietti di carta dall'1 al 10, metterli in un cappello ed estrarne uno. Il numero scelto è un numero casuale. Il numero viene rimesso nel cappello e ne viene estratto un altro. Ogni volta che si estrae un numero, viene rimesso subito dopo nel cappello, mantenendo così ogni volta invariato l'insieme dei numeri possibili. Quando viene selezionato un numero, non è possibile sapere quale numero uscirà subito dopo, ma si è al corrente che sarà compreso fra l'1 e il 10. I NUMERI CASUALI funzionano su questa base.

I numeri casuali sono estremamente utili nella programmazione, in quanto forniscono l'elemento di possibilità o (ovviamente) di casualità. I numeri casuali solitamente hanno una data estensione, ossia c'è un limite massimo e minimo di numeri che si possono scegliere.
Nell'esempio del cappello, la gamma è compresa fra l'1 e il 10. Il limite minimo è "1". Il limite massimo è "10", il che significa che qualsiasi numero dall'uno al 10 può uscire a caso ogni volta che si seleziona un numero.
Si esaminerà ora come il Plus/4 gestisce i numeri casuali e alcune delle possibilità che essi consentono. Questo programma genera 5 numeri completamente casuali:

```
10 FOR X = 1 TO 5: PRINT RND(X): NEXT X
```

Questi numeri casuali sono tutti abbastanza complessi, con diverse cifre a destra del punto decimale... Ma la maggior parte degli usi dei numeri casuali necessitano di numeri interi. Usando la funzione che elimina tutti i decimali, si possono ottenere risultati interi (senza cifre decimali). Qui di seguito, viene fornita una formula per generare numeri casuali in qualsiasi estensione si voglia. Questa formula può essere usata, con pochissime eccezioni, ogni qualvolta si utilizzi nel programma una variabile o un numero.

```
INT (gamma*RND(1) limite minimo
```

Il comando INT ordina all'elaboratore di ignorare tutte le cifre decimali e fornisce solo numeri interi come, ad esempio, 1,45 o 320, invece di numeri come 1.223,45.6677, o 320.59.
Quando si utilizzano i numeri casuali, è più comodo servirsi di numeri interi.
Per **limite minimo,** nella formula, si intende il numero più basso a partire dal quale si vuole che l'elaboratore operi la sua scelta.

Per **gamma,** si intende quanti numeri sono compresi nell'insieme.

Per esempio, se si desidera scegliere un numero casuale compreso fra 1 e 5, il limite minimo è 1 e la gamma è 5. Se si desidera scegliere un numero casuale compreso fra 15 e 20, il limite minimo è 15 e la gamma è 6, perché si sceglie da un insieme di 6 numeri. Se si scelgono numeri da 2 a 100, il limite minimo è 2 e la gamma è 99. Si provi ora il seguente programma:

```
10 PRINT INT(5*RND(1)) + 1
```

Battere RUN e premere **RETURN**, lanciare alcune volte il programma. Ogni volta che si esegue, si ottiene un numero compreso fra 1 e 5. Si stampino ora 15 numeri casuali, con un limite minimo di 1 e una gamma di 5..: si noti che tutti i 15 numeri scelti sono selezionati a caso da 1 a 5:

```
10 FOR X = 1 TO 15              Imposta per 15 volte un loop
20 PRINT INT (5*RND(1)) + 1
30 NEXT X                       Seleziona un numero casuale
```

Battere RUN e premere RETURN

Un modo efficace di usare questa formula è di inserirla in una **funzione definita dall'utente.** Le funzioni definite dall'utente sono estremamente utili nei calcoli matematici ed estremamente semplici da implementare usando il Plus/4. Le funzioni definite dall'utente consentono di programmare una formula e quindi permettono al Plus/4 di introdurvi valori da calcolare. Questa capacità può essere utile per molteplici scopi, la sezione 10 del prontuario contiene una lista di derivate di funzioni matematiche che possono essere usate per definire funzioni. Questa è una istruzione che utilizza la funzione definita dall'utente per generare numeri casuali:

```
10 DEF FNR (X) = INT (X*RND(1)) + 1
```

Essa fornisce numeri casuali compresi fra 1 e X.
FNR è il nome della funzione definita da questa istruzione.

ESEMPIO di utilizzo di una funzione definita:

```
10 DEF FNR (X) = INT (X*RND(1)) + 1
20 DO
30 COLOR 1, FNR (16),5: REM SCEGLI UN COLORE DA 1 A 16
40 PRINT "STO CERCANDO"
50 LOOP
```

L'uso della funzione definita consente un risparmio dello spazio di memoria quando la funzione viene usata più di 1 volta, e rende i programmi più semplici da leggere e da comprendere.

CAPITOLO 6

NOZIONI FONDAMENTALI DI PROGRAMMAZIONE BASIC

- Introduzione
- Modalità di programmazione
- Istruzioni Input/Output
- Istruzioni di controllo e loop
- Istruzioni condizionali
- Subroutines
- REM (Annotazioni)

INTRODUZIONE La comprensione dei programmi finora provati che hanno introdotto alle capacità del Plus/4 potrà essere stata più o meno chiara, comunque sia, questo capitolo spiegherà i comandi BASIC già utilizzati. Inoltre si tratteranno alcune delle istruzioni BASIC più frequentemente usate e si analizzeranno alcune tecniche di programmazione. Sebbene questo capitolo introduca brevemente alla programmazione, resta comunque un'introduzione. Per imparare a programmare veramente, si consiglia l'acquisto di un buon manuale BASIC in una libreria specializzata (per titoli consigliati vedere il capitolo 14 del prontuario). Esistono numerose versioni BASIC, una diversa dall'altra. Il Plus/4 è fornito di una versione avanzata del linguaggio BASIC denominato BASIC 3.5 Commodore.

MODALITÀ DI PROGRAM-MAZIONE

Il Plus/4 permette l'uso di istruzioni e comandi BASIC in modalità diretta e in modalità indiretta. La modalità diretta è anche conosciuta come modalità immediata e la modalità indiretta come modalità di programma.

La MODALITÀ DIRETTA o IMMEDIATA, come suggerisce il nome, esegue immediatamente istruzioni e comandi (subito dopo aver premuto **RETURN** dopo aver digitato un comando). Quando si usano comandi o istruzioni in modalità diretta i numeri di riga non vanno battuti. Basta battere il comando o l'istruzione e premere il tasto **RETURN**. Questa modalità viene utilizzata se si vuole che il computer esegua calcoli con risultato immediato. I comandi LIST, SAVE, LOAD, VERIFY e RUN sono normalmente usati in modalità diretta. In modalità diretta operano la maggior parte di istruzioni BASIC (ma non tutte).

La MODALITÀ INDIRETTA o di PROGRAMMA, permette di organizzare una serie di istruzioni BASIC in un set di istruzioni che verranno eseguite nell'ordine prestabilito. Ogni riga del programma ha un numero di riga che istruisce il computer sull'ordine di esecuzione delle istruzioni. Nel capitolo 4 si sono già incontrati numerosi esempi di modalità di programma. Ricordare che per utilizzare la modalità di programma, bisogna premere **RETURN** per introdurre ogni riga del programma nella memoria del Plus/4. Se non si preme **RETURN** e si va semplicemente a capo, la riga battuta non verrà introdotta. Una volta che il programma è in memoria, non accade nulla finché non si impartisce il comando RUN. Il comando RUN lancia il programma iniziando dalla riga contrassegnata dal numero inferiore.

Le righe vengono numerate per comodità con multipli di 10, data la frequente necessità di inserire nuove righe in diverse posizioni durante la stesura di un programma. Ad esempio, si potrebbero inserire, in caso di necessità, 9 nuove righe tra la riga 10 e la riga 20 di un programma. Comunque sia, il Plus/4 prevede un comando BASIC, RENUMBER, che permette di aggiungere nuove righe e di cambiare la numerazione di quelle esistenti, evitando confusione nella sostituzione e nella reimpostazione delle righe.

ISTRUZIONI DI INPUT/OUTPUT

Le istruzioni di Input/output (I/O) sono utilizzate nei programmi per comunicare con la persona che utilizza il programma. Prima di lanciare il programma, se tutti i dati di calcolo sono disponibili, non è necessario aggiungere l'istruzione input. È comunque più utile se il computer ottiene i dati dalla persona che utilizza il programma (che chiameremo utente). I programmi risultano più versatili se non contengono dati fissi. I messaggi di output possono essere utilizzati dal computer per comunicare le risposte elaborate all'utente. Ovviamente l'output è molto importante: lanciare un programma che non produce un output non avrebbe senso.
Anche i programmatori esperti usano i messaggi I/O per comunicare con dispositivi piuttosto che con l'utente. Questa operazione è già stata fatta non in un programma, ma quando si è utilizzato LOAD o SAVE con il Datassette o l'unità a disco. LOAD è fondamentalmente un'istruzione di input, dato che il Plus/4 prende i dati (il programma) dal Datassette o dall'unità a disco, mentre SAVE è un'istruzione di output, dal momento che il Plus/4 invia i dati a queste periferiche. Nel caso di questa introduzione alle istruzioni I/O si tratteranno solo alcune delle più importanti, cioè quelle di cui si avrà bisogno immediatamente e cioè: PRINT, INPUT, GETKEY, e READ/DATA. PRINT è un'istruzione di output, mentre le altre sono istruzioni di input. (Ricordarsi che le istruzioni I/O BASIC saranno riassunte nel prontuario BASIC alla fine di questo manuale).

Nome dell'istruzione: PRINT

Formato: PRINT "testo fra virgolette" o variabili o numeri o operazioni, ecc.

Utilizzando l'istruzione PRINT nei programmi provati e osservando il formato del suindicato esempio, si può notare che PRINT è una istruzione molto versatile. Si utilizza per stampare messaggi, immagini formate da caratteri grafici, per eseguire operazioni, visualizzare il valore di una variabile e altro. Dato che l'istruzione PRINT viene utilizzata così spesso, vale la pena approfondirla.

Alternativa # 1: Visualizzazione del Testo

Supponendo che in un programma si voglia informare l'utente che la verifica del saldo è negativa o che non si possono portare le amichette nella sala di controllo, il modo più semplice è di utilizzare un'istruzione PRINT seguita da una stringa di testo. Le stringhe di testo vengono stampate esattamente come sono state battute e devono essere scritte tra virgolette (" "). Ad esempio:

Esempi:

BATTERE:

```
10 R = 10 * 2: N=R-5
20 PRINT "R È"; R; "E N È"; N
30 PRINT "MA R PER 2 È"; R*2
40 PRINT "E N MENO 2 È"; N-2
```

Di solito, dopo aver eseguito un'istruzione PRINT, il cursore va a capo automaticamente. Ciò si può evitare aggiungendo un punto e virgola (;) dopo l'istruzione PRINT, in questo modo:

```
200 PRINT "QUESTE DUE PARTI DI FRASE VERRANNO";
210 PRINT "STAMPATE SULLA STESSA RIGA"
```

Nome dell'istruzione: INPUT

Formato: INPUT "messaggio opzionale"; variabile da introdurre.

L'istruzione INPUT permette di ottenere dati dall'utente attraverso la tastiera e di utilizzarli nel programma. Il messaggio opzionale permette di informare esattamente l'utente sulla richiesta: il messaggio viene stampato dopo l'esecuzione dell'istruzione INPUT e seguito dal punto interrogativo.
A questo punto il Plus/4 attende l'introduzione della risposta da parte dell'utente seguita da **RETURN**. L'input dell'utente viene posto in una variabile. Si può anche richiedere una stringa all'utente utilizzando una variabile di stringa (ad esempio A$) o un numero utilizzando una variabile numerica. L'istruzione INPUT può essere utilizzata solo in modalità programma.

Esempi:

BATTERE:

```
10 PRINT "COME TI CHIAMI";
20 INPUT A$
30 PRINT "LIETO DI CONOSCERTI"; A$;"."
40 INPUT "QUANTI ANNI HAI"; AG
50 PRINT AG; "È UN PO' PIÙ DELLA MIA ETÀ"
RUN
```

100 PRINT "SEI AL VERDE!"

per informare l'utente che sono finiti i soldi, mentre

150 PRINT "NON PUOI PORTARE LA TUA AMICA NELLA STANZA DI CONTROLLO"

si può usare nel secondo esempio.
Qualsiasi cosa appaia tra le virgolette è definita "letterale" poiché viene stampata esattamente come appare. Non importa se si tratta di parole, lettere, numeri, punteggiatura, ecc. Alcuni tasti come il cursore e i tasti del colore, agiscono differentemente quando usati in una stringa di testo.
Invece di cambiare il colore o di spostare il cursore quando si batte il tasto, nella stringa viene stampato un carattere invertito. Quando il programma verrà lanciato il carattere produrrà l'azione voluta inizialmente. Questo all'interno di un programma permette di ripulire lo schermo, di cambiare il colore della stampa e di muovere il cursore. Ad esempio:

10 PRINT "TEST PER SHIFT CLR/HOME e CONTROL 3, TEST PER CRSR-DOWN e CONTROL 7"

Ricordarsi che i tasti preceduti da SHIFT e CONTROL devono essere battuti contemporaneamente agli stessi. I simboli invertiti agiscono come messaggi che il computer interpreta per cancellare lo schermo, cambiare il colore o spostare il cursore.

Alternativa # 2: Stampa di numeri e operazioni

PRINT può visualizzare il risultato di un'operazione eseguita all'interno di un'istruzione PRINT (VEDERE NUMERI e OPERAZIONI). Il Plus/4 esegue le operazioni richieste per dare il risultato che poi visualizza sullo schermo. Ad esempio:

100 PRINT 58*15,23,45+1000-45*(4-3)
stampa:

870 23 1000

Tali operazioni diventano ancora più interessanti con l'uso di variabili. L'input dell'utente può essere visualizzato e precedenti operazioni memorizzate in variabili possono essere visualizzate o perfino utilizzate in altre operazioni.

Nome dell'istruzione: GETKEY

Formato: GETKEY variabile da introdurre

GETKEY è un altro modo per introdurre dati mentre il programma è in corso. L'istruzione GETKEY accetta un solo tasto alla volta. Qualsiasi tasto venga premuto, sarà assegnato alla variabile di stringa specificata nell'istruzione GET (ad esempio A$). GETKEY è utile poiché permette di introdurre dati un carattere alla volta senza dover premere il tasto **RETURN** dopo ogni carattere. L'istruzione GETKEY può essere utilizzata solo in un programma.

Esempio di GETKEY in un programma:

```
1000 PRINT "PER FAVORE SCEGLI A,B,C,D,E oppure F"
1010 GETKEY A$
```

Nome dell'istruzione: READ/DATA

Formato: READ variabili da introdurre
DATA elementi di dati da leggere

Le istruzioni READ/DATA sono utili nell'assegnazione di valori alle variabili. L'istruzione READ si può paragonare a una istruzione INPUT che chiede dati al Plus/4, e non all'utente. I dati sono ovviamente contenuti nell'istruzione DATA. Quando il Plus/4 esegue un'istruzione READ, tiene presente i dati contenuti nell'istruzione DATA successiva e li assegna alla variabile contenuta in READ.
L'istruzione READ è sempre utilizzata con un'istruzione DATA. L'istruzione DATA è una riga di dati (parole o numeri) contenuta in un programma. L'istruzione READ è utilizzata per assegnare quei valori alle variabili. (Per ogni variabile contenuta nell'istruzione READ, il Plus/4 "legge" un valore dalla riga DATA per quella variabile). Un'istruzione DATA non è eseguibile e può essere introdotta in qualsiasi punto del programma. Ricordarsi che il tipo di variabile deve corrispondere al tipo di dati dell'istruzione DATA (variabili numeriche per i numeri, variabili di testo per i testi), altrimenti verrà visualizzato il messaggio TYPE MISMATCH ERROR (Errore di battitura).

Esempio:

```
10 READ A$, B$, C$, D$, E$
20 PRINT A$: PRINT B$: PRINT C$
30 PRINT D$: PRINT E$
40 DATA GROUCHO, HARPO, CHICO
50 DATA ZEPPO, GUMMO
```

Il computer risponde:

```
GROUCHO
HARPO
CHICO
ZEPPO
GUMMO
```

ISTRUZIONI DI CONTROLLO E LOOP

Sarebbe una gran perdita di tempo se il computer potesse eseguire linee di programma solamente in successione numerica. Il computer potrebbe solo partire dall'inizio e procedere passo per passo fino alla fine del programma. Ciò produrrebbe programmi molto lunghi; se si volesse ripetere due volte la stessa cosa (ad esempio PRINT "HELLO") si dovrebbero raddoppiare le linee di programma. In questo caso la differenza non sarebbe molta, ma potrebbe diventare difficoltoso con programmi più lunghi. Le funzioni di controllo permettono di ovviare a questo inconveniente indicando al computer di ignorare il normale ordine delle righe di programma e di saltare a un'altra riga senza tenere conto della sequenza. Nel Plus/4 sono previste numerose istruzioni di controllo: non condizionali (come GOTO) che trasferiscono *sempre* il controllo; istruzioni di conteggio (come FOR/NEXT) che trasferiscono il controllo un preciso numero di volte e, per i fanatici della programmazione strutturata, DO/LOOP.

Nome dell'istruzione: GOTO

Formato: GOTO riga #

GOTO informa il computer di spostarsi immediatamente dalla riga corrente al numero di riga specificata dall'istruzione GOTO. Ad esempio se nella riga n. 20 appare l'istruzione GOTO 40, il Plus/4 salta alla riga n. 40 evitando tutte le istruzioni tra 20 e 40.

Esempio di utilizzo dell'istruzione GOTO in un programma:

BATTERE:

```
10 PRINT: "UNA LIRA RISPARMIATA È MEGLIO DI NIENTE"
20 GOTO 10
```

Il computer stampa il messaggio della riga 10 e lo ripete finché non si preme il tasto STOP:

```
UNA LIRA RISPARMIATA È MEGLIO DI NIENTE
UNA LIRA RISPARMIATA È MEGLIO DI NIENTE
UNA LIRA RISPARMIATA È MEGLIO DI NIENTE

BREAK IN 10

READY
```

Se si preme il tasto di STOP

Questa istruzione di stampa continuerà ininterrottamente.
Ogni volta che il Plus/4 incontra GOTO alla riga 20, ritorna alla riga 10. Questo in "computerese" è denominato LOOP INFINITO. Nel caso in cui si voglia effettuare questa operazione, di solito si vuole fare una ripetizione solo un certo numero di volte, oppure finché non succede una cosa prestabilita. Questa è la ragione per cui nel BASIC sono disponibili le istruzioni FOR/NEXT e DO/LOOP.
GOTO può essere utilizzato anche in modalità diretta. GOTO riga # farà iniziare il programma alla riga specificata e contemporaneamente manterrà le variabili inalterate (invece di azzerarle come avviene col tasto RUN).

Nome dell'istruzione: FOR/NEXT

Formato: FOR variabile = valore iniziale TO valore finale
alcune istruzioni BASIC
NEXT variabile

Le istruzioni FOR/NEXT permettono di creare un loop che si ripeterà un certo numero di volte. Le istruzioni di programma tra l'istruzione FOR e la corrispondente istruzione NEXT vengono ripetute nel loop.
Nell'istruzione FOR, la variabile agisce come contatore. Si pone (inizialmente) al valore iniziale fornito, quindi vengono eseguite le righe di programma dopo FOR, fino all'istruzione NEXT corrispondente.
NEXT ordina al Plus/4 di incrementare di uno il contatore. Se il contatore è inferiore o uguale al valore finale, il computer ritorna alla riga di programma successiva alla istruzione FOR. Altrimenti, il Plus/4 continua con la prima istruzione successiva a NEXT.

Esempio di utilizzo di un loop FOR/NEXT

```
10 PRINT "CONTARE FINO A..."
20 FOR J = 1 TO 10
30 PRINT "NUMERO"; J
40 NEXT J
50 PRINT "ABBIAMO CONTATO FINO A"; J
```

Si può inoltre specificare un valore STEP in una istruzione FOR: invece di incrementare di 1 la variabile del contatore, il Plus/4 incrementa di un valore STEP. Usando ad esempio uno STEP di 5 con una istruzione FOR M = 10 TO 30, il contatore darebbe 10,15,20,25,30 dopo ogni loop. Il comando STEP permette anche di eseguire un conteggio alla rovescia (usando un valore STEP negativo).

Esempio con STEP negativo:

```
10 PRINT "CONTO ALLA ROVESCIA..."
20 FOR J = 10 TO 0 STEP - 1
30 PRINT "MENO"; J
40 NEXT J
50 PRINT "DECOLLO"; J
```

Nome dell'istruzione: DO UNTIL/WHILE... LOOP UNTIL/WHILE

Formato: DO UNTIL [condizione] WHILE [condizione]
alcune istruzioni BASIC
[EXIT]

LOOP UNTIL [condizione] WHILE [condizione]

La combinazione delle istruzioni DO/LOOP è un altro modo molto potente e versatile di creare un loop. Il metodo DO/LOOP è una tecnica comune ai linguaggi di programmazione strutturata. In questo capitolo verranno trattati solo alcuni usi possibili. Volendo creare un loop infinito, basta iniziare una sezione di righe di programma con DO, e concluderlo con una istruzione LOOP, in questo modo:

```
100 DO: PRINT "SALENDO"
110 LOOP
```

Premere il tasto **STOP** per interrompere il programma.
Una forma più utile è la combinazione DO/LOOP con l'istruzione UNTIL. Il loop verrà ripetuto senza interruzione finché la condizione UNTIL non verrà soddisfatta.

```
100 DO: INPUT "TI PIACE IL COMPUTER"; A$
110 LOOP UNTIL A$ = "SI"
120 PRINT "GRAZIE"
```

Per gli altri modi di usare DO/LOOP, vedere il prontuario BASIC alla fine del manuale.

ISTRUZIONI DECISIONALI O CONDIZIONALI

Le istruzioni condizionali vengono usate per prendere decisioni. Una delle capacità più importanti del computer è quella di poter prendere decisioni sulla base di quello che sta succedendo. L'istruzione condizionale disponibile nel Plus/4 è denominata istruzione IF/THEN.

Nome dell'istruzione: IF/THEN

Formato: IF condizione THEN fai questo (solo se la condizione è vera). Basilarmente l'istruzione IF/THEN agisce in questo modo:

IF (questa affermazione è vera) THEN (eseguire questa istruzione). In effetti, già si conosce il modo in cui agiscono le istruzioni condizionali. Si sarà già sentita questa famosa frase:

SE (IF) mangi tutta la verdura, POI (THEN) potrai mangiare il dolce. Può sembrare un po' banale, ma è questo il perno dell'istruzione IF/THEN.

Se la condizione della istruzione IF è vera, tutto ciò che segue THEN viene eseguito.

ESEMPIO:

```
10 INPUT "QUAL È LA DECIMA LETTERA DELL'ALFABETO"; A$
20 IF A$ = "J" THEN PRINT "GIUSTO":GOTO 100
30 INPUT "QUESTA È UNA A"; X$
40 IF X$ = "A" THEN 60
50 PRINT "SBAGLIATO, PROVA ANCORA": GOTO 30
60 PRINT "BATTERE A B"
70 GETKEY A$:IF A$ = "B" THEN PRINT "GIUSTO"
100 PRINT "PER OGGI BASTA COSÌ"
```

Alla riga 40 si è solo detto THEN 60, in effetti il significato è THEN GOTO 60, ma dal momento che la combinazione THEN GOTO viene utilizzata molto spesso, il BASIC permette di ignorare GOTO. Un passo opzionale dell'istruzione IF/THEN è la clausola ELSE, che dirige il computer ad una specifica azione se la condizione originale IF non si è soddisfatta. Si può esemplificare la clausola ELSE: IF B> 5 THEN 40 ELSE GOTO 10. Il prontuario BASIC spiega adeguatamente l'istruzione IF/THEN/ELSE.

SUBROUTINES Se nel programma si presenta qualcosa che deve essere ripetuta in più punti ci sono due alternative: si possono raddoppiare le routines oppure creare una subroutine.
Una subroutine è una sezione del programma che può venire usata da qualsiasi punto del programma. Quando la subroutine è terminata, il programma continua automaticamente dalla istruzione immediatamente successiva al punto in cui si era richiamata la subroutine.

Nome dell'istruzione: GOSUB/RETURN

Formato: GOSUB riga #

L'istruzione GOSUB viene utilizzata per chiamare una subroutine. Come per l'istruzione GOTO, il controllo viene trasferito al numero della riga specificata nell'istruzione. Tuttavia a differenza di GOTO, il Plus/4 ricorda il punto in cui GOSUB è allocato. Non appena si incontra un RETURN, il controllo torna al punto immediatamente successivo all'istruzione GOSUB.

Esempio:

```
5 T = 0:FOR J=1 TO 99
10 PRINT "DAMMI UN NUMERO DA 1 A 10"
20 INPUT N
30 IF N<1 THEN GOSUB 100:GOTO 20
40 IF N>10 THEN GOSUB 100:GOTO 20
50 T = T+N
60 NEXT J
70 PRINT "IL TOTALE È" J
80 END
100 PRINT "NUMERO FUORI GAMMA"
105 PRINT "SI PREGA DI BATTERE UN NUMERO TRA 1 E 10"
110 RETURN
```

Se si incontra un RETURN senza istruzioni GOSUB attive, verrà visualizzato un messaggio RETURN WITHOUT GOSUB ERROR (errore - RETURN senza GOSUB). Verificare che il computer non vada ad una subroutine se non tramite l'istruzione GOTO. Un modo per far ciò è quello di raggruppare le istruzioni GOSUB e GOTO, protette dall'esecuzione di un normale programma grazie ad una istruzione END.

Nome dell'istruzione: REM

Formato: REM messaggio

L'istruzione REM serve a introdurre commenti nei programmi. L'istruzione REM non viene eseguita come parte del programma: è un messaggio che si vede solo guardando il listato di programma. Spesso, quando non si commenta, può succedere che sei mesi dopo aver scritto il programma si dimentichi lo scopo di alcune parti. Si può usare l'istruzione REM per inserire promemoria che aiutino una più facile ricostruzione dell'originale significato, oppure per dare messaggi con ulteriori informazioni.

Esempio:

```
1560 E = R/I*9 : REM CIÒ RAPPRESENTA UNA "PITCHER'S ERA"
100 INPUT A,B:REM A È L'ALTEZZA IN POLLICI E B È IL PESO.
```

SOMMARIO Come sottolineato nell'introduzione, questo non vuol essere un corso introduttivo al BASIC. Se ne sono dati solo alcuni fondamenti. Tutti i comandi BASIC del Plus/4 sono contenuti nel prontuario BASIC, con formato, descrizione ed esempi.
Non si abbia timore di fare delle prove, chi vuole imparare seriamente il BASIC può procurarsi i libri sulla programmazione BASIC elencati alla sezione 14 del prontuario. Programmare è come mangiare le ciliege: una tira l'altra!

CAPITOLO 7

USO DELLA GRAFICA E DEL COLORE

- Caratteri grafici
- Animazione dei caratteri
- Controllo dei colori
- Grafici ad alta risoluzione
- Punti, linee ed etichette
- Quadrati, cerchi, poligoni e disegni
- Grafici multicolor

CARATTERI GRAFICI Ogni tasto alfabetico (compresi i tasti @ ,-,* e £) contiene anche 2 caratteri grafici. Per stampare i caratteri grafici, occorre tenere premuto il tasto SHIFT o il tasto C= insieme al tasto del simbolo grafico che si desidera.
Quando il Plus/4 è nella modalità maiuscola/grafica, tenere premuto SHIFT e premere un tasto alfabetico per visualizzare il carattere grafico sulla destra dello stesso. Questi caratteri includono i semi delle carte da gioco, una palla piena e una vuota, e un insieme di linee e di caratteri di collegamento che consentono di disegnare svariate figure sullo schermo. Seguono alcuni esempi per esercitarsi con questi caratteri:

Esercizio 1: Cerchio Ampio

1: Premere il tasto SHIFT LOCK
2: Premere la lettera U, quindi la lettera I
3: Premere il tasto RETURN
4: Premere la lettera J, quindi la lettera K
5: Premere il tasto RETURN

Esercizio 2: Serpentina

1: Premere il tasto SHIFT LOCK
2: Premere in susseguenza U,I,U,I,U, e I
3: Premere il tasto RETURN
4: Premere in susseguenza K,J,K,J,K, e J
5: Premere il tasto RETURN

Esercizio 3: Linea Curva

1: Premere il tasto SHIFT LOCK
2: Premere in susseguenza E,D,C,*,F, e R
3: Premere il tasto RETURN

Esercizio 4: Doppia Croce

1: Premere il tasto SHIFT LOCK
2: Premere in susseguenza M,SPAZIO,N,SPAZIO e -.
3: Premere il tasto RETURN
4: Premere in susseguenza SPAZIO, V,SPAZIO, *,+,e *.
5: Premere il tasto RETURN
6: Premere in susseguenza N,SPAZIO, M, SPAZIO e -.
7: Premere il tasto RETURN

Quando si sono concluse queste operazioni, premere di nuovo SHIFT LOCK portandolo alla posizione di riposo.
È probabile che ci si chieda perché l'elaboratore non abbia visualizzato il messaggio SYNTAX ERROR (errore di sintassi) quando si è premuto RETURN. Infatti, sulla riga si sono battuti dei caratteri che non erano comandi comprensibili all'elaboratore.
La ragione è che il Plus/4 non considera la riga che si è battuta, se si tiene premuto SHIFT insieme a RETURN. Se si preme RETURN senza il tasto SHIFT, l'elaboratore cerca di comprendere il messaggio inviatogli, mentre in effetti si stanno eseguendo solo dei disegni.
Finora non si sono presi in considerazione i caratteri grafici sulla sinistra dei tasti. Questi funzionano esattamente come i caratteri sulla destra, eccetto che occorre premere il tasto C= invece di SHIFT. Non essendo previsto un tasto blocca C=, è necessario tenerlo premuto.
Questo insieme di caratteri grafici può essere stampato sia nella modalità maiuscola/grafica sia in quella maiuscola/minuscola, entrambe sempre disponibili.
I caratteri grafici sulla sinistra includono linee ed angoli usati per disegnare diagrammi e tabelle. Per esempio, per sottolineare una parola:
per primo, spostare il cursore sulla riga posta sotto la parola che si vuole sottolineare. Quindi, premere contemporaneamente il tasto C= e il tasto T, che stampa un carattere di sottolineatura. Tenere premuti entrambi i tasti, finché la parola non sia sottolineata.

Esercizio 5: Semi Barra

1: Tenere premuto il tasto C= durante l'intero esercizio.
2: Premere in susseguenza D,I,I,F.
3: Premere il tasto RETURN

Esercizio 6: Cuneo

1: Tenere premuto il tasto C= e premere in susseguenza T,Y,U.
2: Tenere premuto il tasto CONTROL e premere 9.
3: Tenere premuto il tasto C= e premere in susseguenza I,O,P, @ , e SPAZIO.
4: Premere il tasto RETURN

Esercizio 7: Finestra

1: Tenere premuto il tasto C= fino al punto 4.
2: Premere in susseguenza A,R,S, e RETURN
3: Premere Q
4: Rilasciare C=
5: Tenere premuto SHIFT e premere +.
6: Rilasciare SHIFT e premere di nuovo C= (che non verrà più rilasciato fino alla fine dell'esercizio)
7: Premere W, quindi RETURN
8: Premere in susseguenza Z, E, X, e RETURN.

Lo scopo di questi esercizi è di mostrare come i simboli grafici del Plus/4 possano essere usati per creare forme e figure diverse. Queste sono solo alcune delle figure che è possibile sviluppare. Ora che si ha un'idea di che cosa comporta l'uso dei simboli grafici per ottenere forme diverse, è consigliabile esercitarsi e osservare i risultati.

ANIMAZIONE DEI CARATTERI

I film sono in realtà una sequenza di immagini statiche. Ogni immagine è leggermente diversa da quella che la precede. Il proiettore mostra ogni immagine per un tempo brevissimo, prima di passare alla successiva. La scena così si anima.
L'animazione con l'elaboratore si basa sullo stesso principio, per prima cosa l'elaboratore disegna un'immagine, quindi la modifica leggermente. Il Plus/4 è abbastanza veloce nel far muovere senza scatti oggetti su tutta la superficie dello schermo. Questo risulta utile nell'applicazione ai programmi di gioco e pratici.
Non è possibile battere abbastanza velocemente per creare animazione. Un film è animato ad una velocità di 30 immagini al secondo. La velocità deve essere abbastanza elevata per ingannare l'occhio dello spettatore.
Quindi sarà necessario un programma che visualizza un'immagine, attende pochi attimi e quindi passa a una nuova immagine. Per ordinare al programma di creare immagini, si utilizza l'istruzione PRINT con i caratteri grafici. Il tipo più semplice d animazione implica l'uso abbinato di 2 caratteri per ottenere l'effetto di animazione.

Questo programma simula il movimento di una palla che rimbalza.

Battere NEW e premere **RETURN** prima di introdurre un nuovo programma. Ricordarsi di premere **RETURN** al termine di ogni riga di programma.

```
10 PRINT " HOME SHIFT Q"
20 FOR L = 1 TO 100
30 NEXT L
40 PRINT " HOME SHIFT W"

50 FOR L = 1 TO 100
60 NEXT L
70 GOTO 10
```

Battere contemporaneamente questi tasti.

Battere RUN e premere **RETURN**

Per ottenere un effetto più interessante si potrà creare una piccola immagine utilizzando diversi caratteri grafici, e quindi cambiarne alcuni lasciando gli altri nella stessa posizione. Questo darà un effetto di movimento di una parte dello oggetto, come nel seguente programma.

NOTA IMPORTANTE: Ogni volta che viene detto di premere il tasto SHIFT o C= questo deve essere battuto CONTEMPORANEAMENTE al tasto successivo, durante l'introduzione del programma.

```
10 PRINT "HOME SHIFT M SHIFT W SHIFT N"
20 PRINT "SPAZIO C= + SPAZIO"
30 PRINT "SHIFT N SPAZIO SHIFT M"
40 FOR L = 1 TO 100:NEXT L
50 PRINT "HOME SPAZIO SHIFT W SPAZIO"
60 PRINT "C= T C= + C= T"
70 PRINT "SPAZIO C= G C= G"
80 FOR L = 1 TO 100:NEXT L
90 GOTO 10
```

Battere RUN e premere RETURN.

Negli esempi di animazione finora incontrati, si è utilizzata solo un'area dello schermo. Ora si faranno muovere le figure animate sullo schermo. La funzione TAB è d'ausilio quando si desidera spostare la figura dal margine sinistro. Il seguente programma rappresenterà un serpente che striscia sullo schermo.
Ricordarsi che SHIFT va premuto insieme ai tasti successivi.

```
 5 FOR A=0 TO 30
10 PRINT "SHIFT CLR"
20 PRINT TAB(A)"SHIFT U SHIFT I SHIFT U SHIFT I"
30 PRINT TAB(A)"SHIFT K SHIFT J SHIFT K SHIFT J"
40 FOR L = 1 TO 100:NEXT L
50 PRINT "SHIFT CLR"
60 PRINT TAB(A+1)"SHIFT I SHIFT U SHIFT I SHIFT U"
70 PRINT TAB(A+1)"SHIFT J SHIFT K SHIFT J SHIFT K"
80 FOR L = 1 TO 100:NEXT L
90 NEXT A
```

Utilizzando caratteri come la palla (SHIFT Q), si possono creare video giochi sullo schermo. Per spostare la palla, cancellare la palla e riprodurla nella nuova posizione, come nel seguente programma.

```
10 PRINT "SHIFT CLR"
20 PRINT "SPAZIO SHIFT Q← ";
30 FOR L = 1 TO 50:NEXT L
40 GOTO 20
```

tasto freccia a sinistra/cursore

Battere RUN e premere RETURN. Premere il tasto STOP quando si vorrà arrestare il movimento della palla.

CONTROLLO DEI COLORI

Diversi colori possono essere inseriti in ogni parte dello schermo. La cornice può essere di un colore, lo sfondo di un colore diverso e ogni carattere di un altro ancora. Si è già esaminato come si imposta il colore di un carattere utilizzando la tastiera. Il comando COLOR imposta il colore delle altre aree dello schermo.
Ora impostare il colore della cornice sul rosso, battendo il comando COLOR 4,3 e premendo **RETURN**. Il numero 4 del comando indica la cornice e il numero di colore 3 indica il rosso (il tasto 3 è il tasto contrassegnato anche con RED).
Ora battere COLOR 0,7 e premere **RETURN**. Lo sfondo dello schermo diventerà azzurro. Il numero 0 rappresenta lo sfondo, mentre 7 è l'azzurro (anche questo colore corrisponde al numero 7 sulla tastiera).
Il primo numero dopo la parola COLOR indica l'area dello schermo che si intende cambiare. L'area 0 è lo sfondo, 1 è il colore dei caratteri e 4 è la cornice. Si esamineranno le aree 2 e 3 quando si passerà alla parte grafica multi-color in questo stesso capitolo.

Numeri di Area di Schermo

AREA #	NOME AREA
0	Sfondo
1	Carattere
2	Multi-color 1
3	Multi-color 2
4	Cornice

Ogni colore ha un livello di luminosità regolabile chiamato "luminanza". È possibile aggiungere un numero da 0 (più scuro) a 7 (più chiaro) dopo il numero del colore per variare la luminanza. Battere COLOR 4,3,0 e poi **RETURN**. La cornice diventa rosso scuro. Battere COLOR 4,3,7 e la cornice diventa rosso brillante.

Numeri del Colore

COLORE #	COLORE	COLORE #	COLORE
1	NERO	9	ARANCIONE
2	BIANCO	10	MARRONE
3	ROSSO	11	VERDEGIALLO
4	CIANO	12	ROSA
5	VIOLA	13	VERDAZZURRO
6	VERDE	14	CELESTE
7	AZZURRO	15	BLU NOTTE
8	GIALLO	16	VERDE CHIARO

In breve, il comando COLOR si presenta così:
COLOR area, colore, luminanza
Ecco un breve programma che mostra tutti i colori del Plus/4:
Per prima cosa battere NEW seguito da **RETURN**. Ricordarsi di premere **RETURN** al termine di ogni riga di programma.

```
10 COLOR 0,7,7
20 FOR M = 0 TO 7
30 FOR N = 1 TO 2
40 FOR L = 1 TO 16
50 PRINT " CONTROL RVS ON ";
60 READ A
70 COLOR 1,A,M
80 PRINT "      ";
90 NEXT L
100 PRINT
110 RESTORE
120 NEXT N,M
130 COLOR 1,2,4
200 DATA 7,14,4,13,6,16,11,8,10,9,3,12,5,15,2,1.
```

Ora battere RUN e poi **RETURN** per vedere lo schermo azzurro brillante con ciascuno degli altri 15 colori visualizzati ad ogni livello di luminanza.

NOTA: Come la maggior parte di termini grafici BASIC riesaminati in questo capitolo, COLOR si può riferire intercambiabilmente ad un'istruzione o a un comando. Nella spiegazione di COLOR o di qualsiasi altro comando grafico, la diversità non è importante, dal momento che essa è basata sul fatto che il termine venga usato più frequentemente in modalità diretta o in modalità di programma.

GRAFICI AD ALTA RISOLUZIONE

Lo schermo del Plus/4 contiene 25 righe di 40 caratteri ciascuna, ossia in totale 1000 posizioni di carattere sullo schermo. Ogni carattere è formato da 8 file di 8 singoli punti. Lo schermo è formato da 200 file di punti composte di 320 punti ciascuna per un totale di 64.000 punti. Il Plus/4 è in grado di controllare ogni singolo punto.
Utilizzando grafici normali, si ha un controllo limitato sui punti individuali. Bisogna usare i 256 caratteri in ogni set di carattere incorporato nel Plus/4 per poter creare numerose immagini.
La capacità del Plus/4 di utilizzare grafici ad alta risoluzione permette di controllare ogni singolo punto e quindi di creare un numero grandissimo di immagini. Si possono usare comandi che permettono di tracciare e cancellare punti, linee, cerchi ed altre forme.
Esiste un limite ai grafici ad alta risoluzione. Il Plus/4 può usare solo due colori in ogni posizione di carattere 8x8. Ciò significa che ognuna di queste posizioni può utilizzare solo due colori contemporaneamente (colore di primo piano e di sfondo per quel riquadro). Si possono usare diversi colori per ogni differente posizione di carattere, ma solo due colori all'interno di quella posizione. Un'altra modalità grafica che verrà trattata più avanti in questo capitolo, la modalità multi-color, permette di usare fino a quattro colori per posizione di carattere a discapito della risoluzione disponibile nella modalità ad alta risoluzione.
Questo programma utilizza alcune delle capacità del Plus/4 di grafici ad alta risoluzione, in particolare il comando GRAPHIC. Iniziare battendo NEW e poi RETURN. Battere il tasto RETURN dopo ogni riga. Dopo aver battuto un intero programma, battere RUN e poi RETURN come sempre.

```
10 COLOR 0,1
20 GRAPHIC 1,1
30 FOR L = 2 TO 16
40 COLOR 1,L,2
50 DRAW 1,0 L*12 TO 319,L*12
60 DRAW 1,L*18,0 TO L*18,199
70 NEXT L
80 FOR L = 1 TO 5000:NEXT
90 COLOR 1,2,3
100 GRAPHIC 0
```

Notare che i colori cambiano vicino alle intersezioni.
Per passare da grafici normali (anche detti modalità testo) a quelli ad alta risoluzione, basta battere il comando GRAPHIC 2,1 e poi RETURN. A questo punto lo schermo si svuota e il cursore riappare vicino alla parte inferiore dello schermo. Il Plus/4 divide lo schermo in due sezioni separate: la parte superiore per i grafici e quella inferiore per cinque righe di testo. Nel caso in cui non si vogliano le cinque righe di testo, si può usare il comando GRAPHIC 1,1 anche se i comandi battuti non saranno visualizzati.
Si può passare indifferentemente dai grafici al testo usando il comando GRAPHIC. Il comando GRAPHIC 0 riporta lo schermo al testo, mentre GRAPHIC 2 riporta all'alta risoluzione senza cancellare lo schermo. Per cancellare lo schermo, bisogna aggiungere ,1 dopo il comando.
In genere il comando GRAPHIC si presenta così:

GRAPHIC modalità, cancellazione ——— questa parte è opzionale

Numero di Modalità	Effetto
0	Testo
1	Alta risoluzione
2	Alta risoluzione + testo
3	Multi-color
4	Multi-color + testo

Numero di Cancellazione	Effetto
0	Non azzera lo schermo
1	Azzera lo schermo

Un altro modo di cancellare lo schermo ad alta risoluzione è con l'utilizzo del comando SCNCLR che cancella lo schermo senza cambiare la modalità grafica.
Quando si impiegano grafici ad alta risoluzione, il computer riserva 10K di memoria per lo schermo ad alta risoluzione. Questa parte di memoria viene presa dall'area di programma BASIC. Quando non è più necessaria la modalità grafica, si può richiamare questa memoria con il comando GRAPHIC CLR.

PUNTI, LINEE ED ETICHETTE

Battere i comandi GRAPHIC 2,1: DRAW 1,0,0 e poi **RETURN**. Osservare attentamente l'angolo superiore sinistro dello schermo: il Plus/4 vi traccia un punto nero.
Nel comando DRAW, il primo numero può essere sia 1 (colore di primo piano) sia 0 (sfondo). I due numeri successivi si riferiscono alla posizione della fila e della colonna per il punto. Perciò se si volesse disegnare un punto alla fila 17, colonna 20, basterebbe battere DRAW 1,17,20. Per cancellare questo punto battere DRAW 0,17,20.
Il comando DRAW può anche tracciare una linea tra due punti qualsiasi. Basta aggiungere la parola TO e le coordinate dell'altro capo, in questo modo:
DRAW 1,1,1 TO 100,100, che traccia una riga da 1,1 a 100,100.
Chi è abituato a tracciare grafici matematici, all'inizio può sentirsi confuso con l'impiego del computer, difatti nel Plus/4 il sistema di coordinate è differente. In matematica, il punto 0,0 si troverebbe sia al centro sia nell'angolo inferiore sinistro dello schermo, mentre sul computer si trova nell'angolo superiore sinistro. Con la pratica ci si abituerà al sistema del computer.
Dopo aver impostato un punto o una linea sullo schermo, è possibile tracciare una linea da questa a qualsiasi altro punto, in questo modo: DRAW 1 TO 150,50, ciò traccia una linea dall'ultimo punto visualizzato alla fila 150, colonna 50. Se il programma usa molti comandi DRAW TO, si può allocare il primo punto sullo schermo usando il comando LOCATE, come in LOCATE 100,100.

Il comando DRAW può avere parecchi formati, ad esempio:

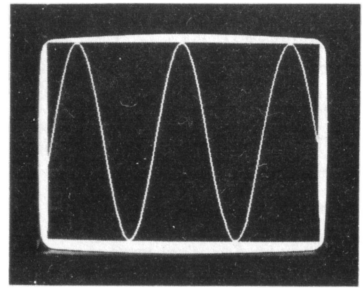
COMANDO	RISULTATO
DRAW sorgente del colore, colonna, fila	PUNTO
DRAW sorgente del colore, colonna, fila TO, colonna, fila	LINEA
DRAW sorgente del colore TO colonna, fila	LINEA TRACCIATA DALL'ULTIMO PUNTO

La sorgente del colore è 0 per lo sfondo e 1 per il primo piano.

Per cancellare punti o linee sullo schermo, bisogna usare il comando DRAW seguito dal numero 0. Se si crea un punto con DRAW 1,1,1, si può poi cancellare con DRAW 0,1,1. Una linea creata con DRAW 1,1,1 TO 100,100 si cancella con DRAW 0,1,1 TO 100,100.

Questo programma traccia una curva basata sulla funzione seno. Battere NEW e poi **RETURN**. Ricordarsi di premere il tasto **RETURN** dopo ogni riga e poi battere RUN.

```
10 COLOR 0,1
20 COLOR 1,2
30 GRAPHIC 1,1
40 LOCATE 0,100
50 FOR X = 1 TO 319
60 Y = INT (100+99*SIN(X/20))
70 DRAW 1 TO X,Y
80 NEXT X
90 FOR L = 1 TO 5000
100 NEXT L
110 GRAPHIC 0
```



Non battere NEW dopo aver lanciato l'ultimo programma. Per rappresentare graficamente il programma in modo diverso, cambiare la riga 70 in:

```
70 DRAW 1, X, Y
```

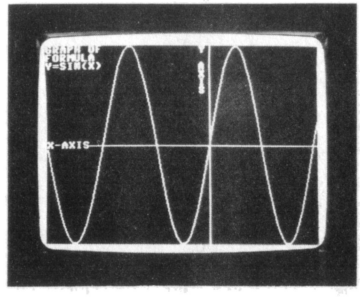
Questo programma traccia la stessa curva utilizzando punti al posto delle linee.

COMANDO CHAR I diagrammi risultano più comprensibili ed utili se li si etichetta. Si può usare il comando CHAR per introdurre un testo in un disegno ad alta risoluzione. Ad esempio, l'istruzione CHAR 1,0,5, "HELLO" inserisce la parola HELLO nella quinta fila al bordo sinistro dello schermo. Il primo numero dopo la parola CHAR può essere sia 1 (per tracciare) sia 0 (per cancellare). I due numeri successivi sono la colonna e la fila in cui appare il testo:

Lasciare nel computer gli ultimi due programmi: non battere NEW.

Aggiungere queste righe:

```
81 CHAR 1,0,0, "GRAPH OF": CHAR 1,0,1, "FORMULA"
82 CHAR 1,0,2, "Y = SIN(X)"
83 DRAW 1,0,100 TO 319,100,189,0 TO 189,199
84 CHAR 1,0,12, "X-AXIS":CHAR 1,22,0,"Y"
85 CHAR 1,22,2, "A":CHAR 1,22,3,"X"
86 CHAR 1,22,4, "I":CHAR 1,22,5,"S"
```



QUADRATI, CERCHI, POLIGONI E DISEGNI

Il comando DRAW permette di far disegni traccianto numerosi punti o linee. Per tracciare un quadrato si può usare il comando DRAW 1,0,0 TO 100,0 TO 100,100 TO 0,100 TO 0,0 (tracciando ogni punto finale del quadrato) oppure si può usare solamente il comando BOX.

Come Disegnare un Rettangolo

Comando BOX

Il Plus/4 include un comando che facilita il tracciamento di quadrati e di altre forme rettangolari. Il comando BOX permette di impostare i punti di due angoli opposti del quadrato. Per raddoppiare lo stesso riquadro del succitato esempio, usare BOX 1,0,0,100,100. Di nuovo, il numero 1 significa che si vuole tracciare e non cancellare. I quattro numeri successivi sono le coordinate degli angoli opposti del riquadro, (0,0) dell'angolo superiore sinistro e (100,100) vicino al centro dello schermo.
Il comando BOX può formare qualsiasi rettangolo semplicemente cambiando gli angoli. Si può persino ruotare il riquadro specificando un angolo (in gradi) dopo l'ultima coordinata, in questo modo: BOX 1,50,50,100,100,45. Il riquadro viene ruotato di 45 gradi in senso orario.
Se si vuole tracciare un riquadro pieno invece del solo perimetro, basta aggiungere virgola 1 dopo l'angolo. Un riquadro pieno al centro dello schermo si ottiene con BOX 1,100,50,220,150,,1. Notare che è necessaria una virgola per allocare l'angolo anche se non si vuole ruotare il riquadro.

Alcuni tipici formati del comando BOX:

COMANDO	EFFETTO
BOX 1, colonna 1, fila 1, colonna 2, fila 2	PERIMETRO
BOX 1, colonna 1, fila 1, colonna 2, fila 2, angolo	ROTAZIONE
BOX 1, colonna 1, fila 1, colonna 2, fila 2,, 1	RIQUADRO PIENO
BOX 0, colonna 1, fila 1, colonna 2, fila 2,angolo,, 1	CANCELLA L'AREA DELLO SCHERMO

Seguono due programmi che utilizzano il comando BOX:

Non dimenticarsi di battere NEW e poi RETURN prima di introdurre ogni programma, e di premere RETURN dopo aver battuto ogni riga.

```
10 COLOR 0,1
20 COLOR 1,2
30 GRAPHIC 2,1
40 A = RND(1)* 20+10
50 FOR L = 0 TO 359 STEP A
60 BOX 1,100,30,220,130,L
70 NEXT L
80 FOR L = 1 TO 2000:NEXT L
90 GRAPHIC 0,1
```

```
 5 TRAP 60
10 GRAPHIC 2,1
20 DEF FNA(X)=INT(RND(1)*X)
30 COLOR 1,FNA(15)+1
40 BOX 1,FNA(320),FNA(160),FNA(320),FNA(160),,1
50 GOTO 30
60 COLOR 1,2,3: GRAPHIC 0
```

Premere RETURN e battere RUN dopo che ogni programma è stato introdotto completamente. Tenere premuto il tasto STOP per interrompere il programma.
Il secondo programma traccia su tutto lo schermo dei riquadri di colori diversi. Si noterà che alcune parti dello schermo cambiano contemporaneamente ad altre parti vicino ad esse. Precedentemente, in questo capitolo, se ne è trattata la ragione.

Come Disegnare Cerchi

Il Plus/4 possiede anche dei comandi per disegnare cerchi. Come nel comando BOX, è possibile trasformare una forma iniziale (in questo caso un cerchio) in un'altra (in questo caso un ovale o ellisse), e ruotarla. È pure possibile tracciare solo una sezione della forma (detta arco). Questo comando traccia un cerchio nel centro dello schermo CIRCLE 1,160,100,50. Ciò indica al Plus/4 di tracciare un cerchio col centro alla fila 160 e colonna 100, con un raggio di 50. In effetti ciò produce un ovale, dato che i punti sullo schermo sono più alti che larghi. Per modificarlo in un vero cerchio, bisogna aggiungere un numero separato, per indicare che l'altezza è diversa dalla larghezza, in questo modo: CIRCLE 1,160,100,50,42.

Il Plus/4 può anche disegnare un quadrato, un triangolo o altri poligoni utilizzando il comando CIRCLE. È sufficiente indicare i gradi dell'angolo tra i punti sulla circonferenza, in questo modo: CIRCLE 1,160,100,50,42,,,,120.
Questo comando traccia un triangolo, dato che l'arco corrispondente ad ogni lato è di 120 gradi. (Omettendo valori numerici, ma mantenendo le virgole in un comando grafico, il computer legge valori standard di default, al posto dei numeri omessi).
Per esempio, per impostare l'angolo di un poligono con N lati, è sufficiente introdurre 360/N.

Segue un breve programma per tracciare poligoni:

```
10 GRAPHIC 2,1
20 INPUT "QUANTI LATI"; A
30 IF A<2 OR A>100 THEN PRINT "NON ESSERE RIDICOLO":
GOTO 20
40 CIRCLE 1,160,80,40,33,,,,360/A
50 GOTO 20
```

Si può scegliere di tracciare solo un arco invece di un cerchio intero. Il comando CIRCLE accetta in gradi gli angoli di partenza e di arrivo, subito dopo l'ottavo numero. Il comando CIRCLE 1,160,100,50,42,90,180 visualizza solamente la sezione inferiore destra del cerchio.
Per ruotare un ovale, aggiungere l'angolo di rotazione in senso orario dopo il comando, ad esempio: CIRCLE 1,160,100,100,20,,,30.
Solitamente i formati del comando CIRCLE sono:

COMANDO	EFFETTO
CIRCLE 1, colonna centrale, fila centrale, raggio	ovale
CIRCLE 1, colonna-c, fila-c, larghezza, altezza	cerchio/ovale
CIRCLE 1, col-c, fila-c, largh, alt, partenza, arrivo	arco
CIRCLE 1, col-c, fila-c, largh, alt,,, angolo	rotazionedell'ovale
CIRCLE 1, col-c, fila-c, largh, alt,,,, angolo fra i punti	poligono

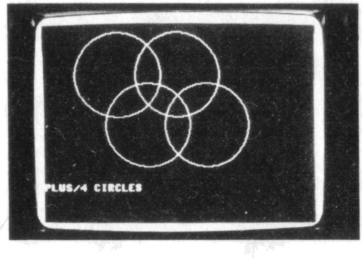
NOTA: Quando in un comando come CIRCLE oppure BOX si presentano virgole senza numeri, il Plus/4 interpreta la virgola come un input del valore di default per quel parametro del comando. Ad esempio, CIRCLE,160,40,100,100 viene letto dal computer come CIRCLE 1,160..., interpretando il valore di default 1 come la sorgente del colore.

Il prossimo programma utilizza il comando CIRCLE per un interessante effetto. Battere NEW e poi premere **RETURN** per cancellare l'ultimo programma della memoria prima di introdurvi questo.

```
10 COLOR 0,1
20 COLOR 1,2
30 GRAPHIC 1,1
40 A = RND(1)*20+10
50 FOR L = 0 TO 359 STEP A
60 CIRCLE 1,160,100,80,40,,,L
70 NEXT L
80 FOR L = 1 TO 2000:NEXT L
90 GRAPHIC 0,1
```

Si può ora provare un programma che utilizza il comando CIRCLE per creare un simbolo:

```
NEW
10 COLOR 0,1
20 COLOR 1,2
30 GRAPHIC 2,1
40 FOR L = 1 TO 4
50 Y = 50
60 IF L = 2 OR L = 4 THEN Y = 100
70 X = L*35+50
80 CIRCLE 1,X,Y,50,42
90 NEXT L
100 PRINT "CERCHI del PLUS/4"
```



COMANDO PAINT Il comando PAINT riempie qualsiasi area inclusa tra i margini formati dalle linee tracciate sullo schermo. Se non vi sono linee tracciate, lo schermo si riempie completamente. Il comando BOX può essere utilizzato per riempire di colore l'interno di quadrati e rettangoli. Il comando PAINT è in grado di colorare sullo schermo l'interno di forme irregolari ed altre aree non uniformi che non possono essere colorate con altri comandi.
Nell'ultimo esercizio eseguito, è stato creato un simbolo a quattro anelli. Aggiungendo alcuni comandi PAINT al programma, si possono riempire di colore solo le aree comprese tra gli anelli.

Aggiungere queste righe all'ultimo programma:

```
110 FOR L = 0 TO 1
120 PAINT 1,120+35*L,75
130 NEXT L
```

GRAFICI MULTICOLOR

I grafici ad alta risoluzione del Plus/4 permettono di controllare ogni singolo punto o "pixel" sullo schermo, ma si è constatato che vi è una limitata capacità di accostare vari colori. La maggior parte di programmi ad alta risoluzione può utilizzare solo uno o due colori. Per includere ulteriori colori diversi, il Plus/4 possiede una particolare modalità grafica di compromesso definita grafici multi-color. Nei grafici multi-color, si ha il controllo della metà dei punti presenti su ogni fila come nell'alta risoluzione, perché ogni punto è largo il doppio. Si hanno 160 punti su ogni fila, ma ancora 200 file. Il compromesso per l'uso di colori multipli, è una perdita di risoluzione.
Per iniziare ad usare grafici multi-color, rivedere il comando GRAPHIC precedentemente trattato in questo capitolo. Si vedrà che lo schermo multi-color senza testo corrisponde a GRAPHIC 3 e che lo schermo multi-color con 5 righe di testo corrisponde a GRAPHIC 4.
Osservare ora la tavola con il listato del comando COLOR. Vi sono due zone che non sono ancora state utilizzate le zone 2 e 3. In queste due zone si trovano due ulteriori colori. Si può usare qualsiasi dei tre colori (1, colore di testo; 2 colore extra; e 3 un altro colore extra). Tali colori non interferiscono l'uno con l'altro sullo schermo, come invece accade con i colori ad alta risoluzione analizzati in precedenti programmi di questo capitolo.
I seguenti due programmi fanno uso di grafici multi-color, il primo con anelli e il secondo mostra un effetto di insegna al neon.

```
10 COLOR 0,1
20 GRAPHIC 4,1
30 FOR L = 1 TO 4
40 Q = L:IF Q>3 THEN Q = Q-3
50 COLOR Q,L+1
60 Y = 50
70 IF L = 2 OR L = 4 THEN Y = 100
80 X = L*18+25
90 CIRCLE Q,X,Y,25,42
100 NEXT L
```

Battere NEW e poi **RETURN** . Non dimenticarsi di premere il tasto **RETURN** dopo ogni riga. Battere RUN e premere **RETURN** alla fine.

```
10 COLOR 0,1
20 GRAPHIC 3,1
30 COLOR 3,1
40 TRAP 200
50 DRAW 3,10,10 TO 10,100:DRAW 3,10,55 TO 30,55
60 DRAW 3,30,10 TO 30,100:DRAW 3,50,10 TO 80,10
70 DRAW 3,65,10 TO 65,100:DRAW 3,50,100 TO 80,100
80 FOR L = 0 TO 7
90 COLOR 3,2,L
100 FOR M = 1 TO 100:NEXT M
110 NEXT L
120 COLOR 3,1
130 FOR M = 1 TO 100:NEXT M
140 GOTO 80
200 GRAPHIC 0:COLOR 1,2,7
```

La zona 3 del colore, la seconda delle zone multi-color, ha una speciale capacità che nessuna delle altre possiede. Dopo aver tracciato sullo schermo utilizzando la zona 3, si può modificare quel colore usando il comando COLOR in qualsiasi punto esso appaia sullo schermo. Stabilito il colore con COLOR 3,5 e tracciando con quel colore, il grafico risulterà viola. Se quindi si cambia il colore con COLOR 3,6 tutte le zone viola si trasformeranno in verde. In nessuna altra zona è possibile effettuare questa operazione.
La guida di consultazione per il Programmatore del Plus/4 contiene ulteriori dettagli sui grafici.

CAPITOLO 8

COME PRODURRE SUONI E MUSICA CON IL PLUS/4

- Introduzione
- Comando VOL
- Comando SOUND
- Creazione di effetti sonori
- Produzione di musica
- La macchina musicale del Plus/4

INTRODUZIONE Per eseguire un breve programma di musica col Plus/4, bisogna battere il programma esattamente come appare e ricordare di premere **RETURN** alla fine di ogni riga. Una volta battuto il programma, battere RUN e premere **RETURN**. Quando il programma chiede l'inserimento di un numero, battere qualsiasi numero da 0 a 1023 e poi premere **RETURN**. Per interrompere il programma introdurre lo 0.

```
10 VOL 8
20 DO
30 INPUT X
35 IF X > 1023 OR X< 0 THEN PRINT "DA 0 a
   1023,PREGO":GOTO 30
40 SOUND 1,X,10
50 LOOP UNTIL X = 0
```

Premere il tasto **RUN/STOP** per interrompere il programma. Per produrre una nota col Plus/4, bisogna battere NEW e premere **RETURN** per azzerare la memoria del Plus/4.

Primo: Battere NEW e premere **RETURN**
Battere VOL 8 e premere **RETURN**

Secondo: Battere SOUND 1,266,60 e premere **RETURN**

Si dovrebbe udire il suono di una nota per circa un secondo. Se non si sente nulla, alzare il volume del televisore o del monitor e riprovare. Questi due comandi sono gli unici che è necessario conoscere per suonare col Plus/4.

COMANDO VOL VOL controlla il volume delle note emesse dal Plus/4. (Basta pensare alle prime tre lettere della parola "volume" per ricordare il comando VOL). Il numero che segue VOL è l'impostazione del volume (basta pensare al comando VOL come se fosse la manopola del volume sul Plus/4). Quando la manopola è sullo 0, il volume del Plus/4 è al minimo e non si percepirà alcun suono. Quando la manopola è sull'8, il volume del Plus/4 è al massimo. Provando il precedente esempio con vari numeri di volume si vedrà che più basso è il numero più attenuata è la nota.

COMANDO SOUND Il comando SOUND indica al Plus/4 tutto quanto è necessario sapere sul suono che si vuole produrre. SOUND è seguito da 3 numeri che descrivono la nota:

SOUND voce, valore della nota, durata.

Il primo numero del comando SOUND si riferisce alla voce e può essere 1,2 oppure 3. Il suono del Plus/4 viene prodotto da due diverse "voci", 1 per la prima voce e 2 per la seconda. La scelta della terza voce si applica alla possibilità della 2ª voce di produrre sia una nota che un rumore.

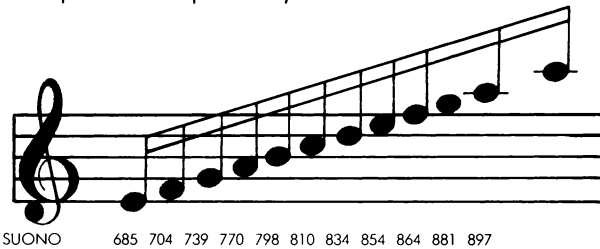
Voce 1 - Questa voce produce solo note. Selezionarla con 1 dopo il comando SOUND.

Voce 2 - Questa voce è simile alla 1, ma può essere utilizzata per produrre note oppure un rumore per effetti sonori. Per usare la voce per produrre note, battere il 2 nel comando; oppure il 3 per usare la voce per provocare un rumore, per creare un effetto sonoro come ad esempio il tuono o la pioggia.

Il secondo numero dopo la parola SOUND è il valore della nota (frequenza): può essere qualsiasi numero tra 0 e 1023 e serve ad indicare al Plus/4 l'altezza della nota. Più alto è il numero, più acuta è la nota. I valori più alti (attorno al 1023) non sono percepibili dall'orecchio umano.

NOTA: il rumore è "bianco" solo nella gamma tra 600 e 940 con voce 3. Utilizzando valori di registro al di fuori di questa gamma si possono ottenere interessanti effetti sonori.

Il seguente pentagramma mostra tutte le note su una scala con il valore di nota da usare. La sezione 11 del prontuario del Plus/4 include un diagramma completo di note per il Plus/4.



SUONO	685	704	739	770	798	810	834	854	864	881	897
NOTA	E	F	G	A	B	C	D	E	F	G	A
FREQUENZA	330		392.5		494.9		588.7		699		880.7
		349.6		440.4		522.7		658		782.2	

Provare sul Plus/4 il seguente programma:

```
NEW

10 VOL 7
20 X = 0
30 DO
40 SOUND 1,X,5
50 X = X + 5
60 LOOP UNTIL X = 1020
70 VOL 0
80 END
```

regola il volume

produce una nota

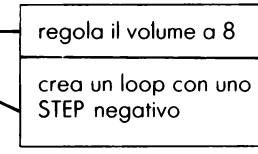
abbassa il volume completamente

Battere RUN e premere **RETURN**. Questo programma permette al Plus/4 di mostrare alcune delle sue gamme musicali.
Il terzo numero dopo la parola SOUND controlla la durata (lunghezza) della nota, informando quindi il Plus/4 sul tempo di emissione della nota. Può essere qualsiasi numero tra 0 e 65535. Questo numero imposta un timer che calcola il tempo in sessantesimi di secondo. Una durata di 60 mantiene la nota per 1 secondo. Per la regola del pollice, più grande è il numero, più a lungo dura la nota. Se ad esempio si utilizzasse 65535, la nota durerebbe più di 16'. Per eliminare il suono, usare la durata 0 che non permette l'emissione del suono.

UN EFFETTO MUSICALE

Usando sia note musicali sia rumore si possono creare effetti sonori. L'abbinamento di semplici programmi BASIC e comandi sound può produrre insoliti e piacevoli effetti. Ad esempio, il loop FOR...NEXT...STEP si può usare per creare effetti sonori. Il seguente programma utilizza un loop FOR... NEXT con uno STEP negativo per calcolare da un numero più alto ad uno più basso.

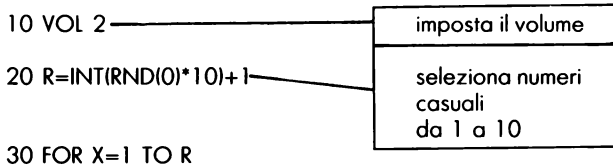
```
10 VOL 8
20 FOR S=1000 TO 700 STEP-25
30 SOUND 1, S, 1
40 NEXT S
```



Per udire l'effetto sonoro battere RUN e premere **RETURN**. La chiave è la riga 20 che seleziona una gamma di numeri da 1000 a 700, decrescendo sulla scala di 25 numeri per volta. Alla fine, la riga 30 dà istruzione al Plus/4 di emettere ogni nota per un istante utilizzando la DURATA 1 che equivale ad 1/60 di secondo. Eseguire prove con differenti numeri e valori di durata può dare alcuni effetti interessanti.

CREAZIONE DI UN RUMORE

Un rumore si specifica usando il valore 3 quando si seleziona una voce nel comando SOUND. Tale operazione è infatti usata per creare un effetto rumore piuttosto che una nota. Il seguente programma utilizza la voce 3 per creare i suoni di una tempesta.




```
30 FOR X=1 TO R
40 SOUND 3,600+30*X,10
50 NEXT X
60 FOR X=R TO 1 STEP-1
70 SOUND 3,600+30*X,10
80 NEXT X
90 T=INT(RND(0)*100)+30
100 SOUND 3,600,T
110 GOTO 20
```

Le righe 30 e 60 predispongono dei loop per il valore della nota (frequenza) del suono, uno crescente e uno decrescente, basato sul numero casuale della riga 20. È importante avere una variazione nel passo, dato che le tempeste hanno differenti intensità di raffiche di vento. Le righe 40 e 70 sono i comandi SOUND che creano il rumore. Le righe 90 e 100 predispongono un casuale ritardo che ricrea la natura irregolare di una tempesta, con intervalli di tempo tra il fischiare del vento. Il programma seleziona un numero casuale (RND) che viene utilizzato per la durata di un altro comando SOUND. Questo comando SOUND rimane allo stesso passo e produce un conforme rumore di fondo che serve da contrappunto alle raffiche di vento.
È estremamente eccitante creare degli effetti sonori rumorosi, cercando di impiegare precisamente gli elementi del suono desiderato.
Per ottenere buoni effetti sonori è comunque necessario effettuare numerose prove.

PRODUZIONE DI MUSICA

Dopo aver preso in considerazione alcuni modi per creare effetti sonori sul Plus/4 si può ora produrre della musica.
Provare i seguenti 2 programmi.
Il primo programma trasforma i tasti da 1 a 8 in un'ottava di pianoforte.
Inserire il programma e poi battere RUN.

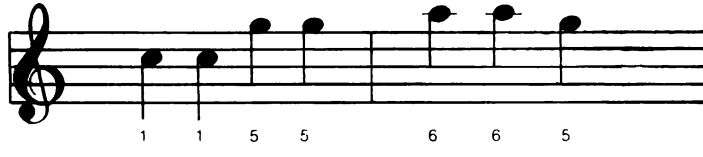
```
5 SCNCLR
10 FOR X =1 TO 8:READ N(X):NEXT X
20 VOL 7
30 DO
40 GET A$:IF A$ = THEN 40
50 A = ASC(A$):IF A < 49 OR A > 56 THEN 90
60 N = A-48
70 SOUND 1,N (N),5
80 COLOR 0,N,3
90 LOOP UNTIL A = 32
100 VOL 0:COLOR 4,2,7
110 DATA 169,262,345,383,453,516,571,596
```

azzera lo schermo

Per emettere delle note premere i numeri da 1 a 8. Suonando le note, la cornice cambia colore. Per smettere di suonare premere la barra spaziatrice per interrompere il programma. Ecco i numeri da battere per suonare un motivetto popolare:

Twinkle, Twinkle Little Star

1 1 5 5 6 6 5
4 4 3 3 2 2 1
5 5 4 4 3 3 2
5 5 4 4 3 3 2
1 1 5 5 6 6 5
4 4 3 3 2 2 1



Questo programma produce una canzone leggendo una lista a coppie di istruzioni DATA. Il primo numero è il valore della nota per il comando SOUND e il secondo è la durata del comando SOUND.

Row Boat

```
10 VOL 8
20 DO
30 READ X,Y
40 SOUND 1,X,Y
45 FOR D=1 TO 550:NEXT
50 LOOP UNTIL X=0
60 END
100 DATA 169, 45, 169, 45, 169, 30
110 DATA 262, 15, 345, 45, 345, 30
120 DATA 262, 15, 345, 30, 383, 15
130 DATA 453, 60, 596, 45, 453, 45
140 DATA 345, 45, 169, 45, 453, 30
150 DATA 383, 15, 345, 30, 262, 15
160 DATA 169, 60
200 DATA 0, 0
```

Questo loop crea un breve ritardo tra le note

Questo programma produce note ascendenti e discendenti con tempi differenziati e visualizza anche alcune barrette colorate insieme alle note.

```
10 VOL 8
20 DO
30 D=INT(RND(0)*5)+2:REM DURATA
40 S=INT(RND(0)*300)+700:REM INIZIO
50 R=INT(RND(0)*(1020-S)):REM GAMMA
60 P=INT(RND(0)*30)+5:REM STEP
70 T=SGN(RND(1)-.5):IF T=0 THEN 70
80 FOR Z=S TO S+T*R STEP P*T
90 SOUND 1, Z, D
100 Y=(Z AND 15) + 1:FOR X=1 TO D
110 PRINT CHR$ (18); :COLOR 1, Y:PRINT" ";
120 NEXT X,Z
130 LOOP
```

Queste REM indicano la funzione di ogni riga

lasciare qui uno spazio

LA GRANDE "MACCHINA MUSICALE" DEL PLUS/4

Quest'ultimo programma è leggermente più lungo: è la "GRANDE MACCHINA MUSICALE del PLUS/4". Battendo un tasto da 1 a 9, si emette la nota ed una nota appare sul pentagramma sulla riga esatta.

```
5 GOSUB 1000
6 FOR X=1 TO 9:READ N(X):NEXT X
8 CHAR 1,8,1"*LA GRANDE MACCHINA MUSICALE*"
10 VOL 7
20 DO
30 GET A$: IF A$="""THEN 30
35 A=ASC(A$): IF A<49 OR A>57 THEN 50
36 N= A - 48
40 SOUND 1, N(N), 4
45 GSHAPE N$,150, 8 * (6+(9-N)), 4
46 FOR Z=1 TO 50: NEXT Z
47 GSHAPE N$,150,8 * (6+(9-N)), 4
50 LOOP UNTIL A=32
55 VOL 0:  GRAPHIC 0: SCNCLR
60 END
100 DATA 345, 383, 453, 516, 571, 596, 643, 685, 704
1000 GRAPHIC 1,1
1010 FOR Y=60 TO 124 STEP 16
1020 DRAW 1, 100, Y TO 200, Y
1030 NEXT Y
1040 A$="FEDCBAGFE"
1050 FOR X=1 TO 9: C=13
1060 IF INT(X/2)= X/2 THEN C=14
1070 CHAR 1, C, X+6, MID$ (A$, X, 1), 0
1075 CHAR 1, C+10, X+6, RIGHT$ (STR$ (10-X), 1)
1080 NEXT X
1090 FOR X=1 TO 8: FOR Y=11 TO 16: DRAW 1, X, Y: NEXT Y, X
1100 Y=1: X=8: DRAW 1, 8, 16 TO X, Y
1110 SSHAPE N$, 1, 1, 8, 16
1120 GSHAPE N$, 1, 1, 4
1130 RETURN
```

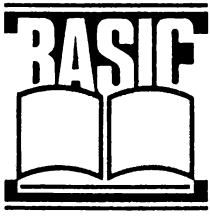
senza spaziatura

Come si può notare, non è difficile produrre i propri programmi musicali. Quelli descritti nel presente capitolo danno solo un'idea delle capacità musicali del Plus/4. Non si abbia timore di provare nuovi suoni e rumori e di creare i propri capolavori.

PRONTUARIO DEL PLUS/4 COMMODORE

Il prontuario del Plus/4 contiene informazioni utili sia al principiante che all'esperto di computer. Alcune parti sono obbligatorie per i principianti, come il Prontuario del BASIC 3.5, che elenca e spiega tutti i comandi, le istruzioni e la terminologia BASIC.
Altre parti saranno utili a quegli utenti la cui conoscenza và oltre il BASIC. TEDMON, il monitor del linguaggio macchina del Plus/4, fornisce alcune indicazioni per la programmazione in linguaggio macchina sul Plus/4. Le parti restanti possono essere utili a tutti gli utenti del Plus/4... listati di messaggi di errore, esempi di programma, la tavola delle note musicali e altro.
Il prontuario del Plus/4 fornisce indicazioni a tutti i programmatori di Plus/4, siano essi principianti oppure già esperti.

PRONTUARIO DEL BASIC 3.5



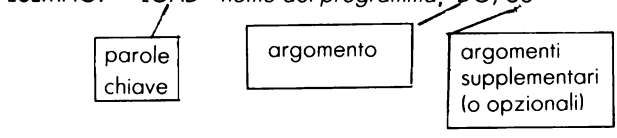
Fino a questo punto si è utilizzato il linguaggio BASIC per introdurre alla programmazione del computer e ad una parte dei termini necessari. Il prontuario fornisce una lista completa di regole (SINTASSI) del linguaggio BASIC 3.5, insieme ad una breve descrizione di ciascuna regola. Fare prove con questi comandi – ricordandosi che non è possibile danneggiare il Plus/4 inserendo i programmi – è il miglior modo per imparare ad elaborare. Il prontuario fornisce anche formati e brevi spiegazioni ed esempi dei comandi e delle istruzioni del BASIC 3.5. Tuttavia il suo scopo non è di insegnare il BASIC, volendolo imparare, la sezione 14 elenca una serie di manuali introduttivi attinenti. Comandi e istruzioni sono elencati in parti separate in ordine alfabetico. I comandi vengono principalmente usati in modalità diretta, mentre le istruzioni sono più frequentemente impiegate nei programmi. Nella maggior parte dei casi, i comandi possono essere utilizzati nei programmi come istruzioni, premettendo ad essi un numero di riga. È possibile usare un certo numero di istruzioni come comandi emettendoli in modalità diretta (cioè senza numeri di riga).
I vari tipi di operazioni in BASIC sono elencati nelle varie parti, sulla base dei seguenti criteri:

- COMANDI: comandi usati per operare con programmi, correggerli, memorizzarli e cancellarli.
- ISTRUZIONI: istruzioni di programma BASIC usate in righe numerate di programmi.
- FUNZIONI: funzioni di stringa, numeriche e di stampa.
- VARIABILI E OPERATORI: diversi tipi di variabili, nomi di variabile validi e operatori logici e aritmetici.

Una spiegazione più esauriente dei comandi BASIC 3.5 viene fornita dalla Guida di Consultazione per i Programmatori disponibile presso i rivenditori Commodore oppure in librerie specializzate.

FORMATO DEL COMANDO E DELL'ISTRUZIONE

Le istruzioni e i comandi presentati in questa parte del prontuario sono retti da logiche convenzioni di formato, studiate per renderle il più chiaro possibile. Nella maggior parte dei casi, vi sono numerosi esempi che illustrano l'effettivo aspetto del comando. Il seguente esempio mostra alcune delle convenzioni di formato utilizzate nei comandi e nelle istruzioni BASIC:



Le parti del comando o dell'istruzione che devono essere battute esattamente come appaiono, vengono evidenziate in neretto nel listato del formato, mentre il nome del comando è in lettere maiuscole. Le parole che non devono essere battute esattamente come riportate, come ad esempio il nome del programma, vengono stampate in corsivo.
Quando appaiono le virgolette (" ") (che di solito racchiudono il nome di un programma o di un file), devono essere introdotte nella posizione indicata dall'esempio del formato.

- PAROLE CHIAVE, anche dette PAROLE RISERVATE, appaiono maiuscole e in neretto. BISOGNA INTRODURRE QUESTE PAROLE CHIAVE ESATTAMENTE COME APPAIONO. Molte parole chiave hanno comunque abbreviazioni che possono venire utilizzate al loro posto (vedere Sezione 2).

Le parole chiave sono parole facenti parte del linguaggio BASIC conosciuto dal computer.
Le parole chiave sono la parte centrale di un comando o di una istruzione e informano il computer sul tipo di azione che si vuol fare eseguire. Tali parole non possono essere usate come nomi di variabile.

- ARGOMENTI (anche detti parametri), appaiono in corsivo minuscolo. Gli argomenti sono le parti di un comando o di una istruzione che viene selezionata; sono di complemento alle parole chiave poiché forniscono specifiche informazioni sul comando o sull'istruzione. Ad esempio, una parola chiave dice al computer di caricare un programma, mentre un argomento lo informa sullo specifico programma da caricare, ed un secondo argomento specifica in

quale drive è contenuto il disco del programma.
Gli argomenti includono i nomi di file, le variabili, i numeri di riga, ecc.

- PARENTESI QUADRE [] mostrano argomenti OPZIONALI. Si può selezionare uno qualsiasi degli argomenti listati oppure nessuno, a seconda della propria necessità.

- PARENTESI ANGOLARI <> indicano che si DEVE scegliere uno degli argomenti listati.

- BARRA VERTICALE separa gli elementi in una lista di argomenti quando le scelte sono limitate agli argomenti listati e non è possibile usare nessun altro argomento. Quando la barra verticale appare in un listato racchiusa tra PARENTESI QUADRE, le scelte sono limitate agli elementi listati, ma ancora sussiste l'opzione di non usare alcun argomento.

- PUNTINI DI SOSPENSIONE ..., una sequenza di tre punti significa che un'opzione o un argomento possono essere ripetuti più di una volta.

- VIRGOLETTE " " Racchiudono stringhe di caratteri, nomi di file e altre espressioni. Quando gli argomenti sono racchiusi da virgolette in un formato, bisogna includere le virgolette nel comando o nell'istruzione. Le virgolette non sono convenzioni utilizzate per descrivere formati; esse sono parti richieste di un comando o di un'istruzione.

- PARENTESI () Quando in un formato gli argomenti sono racchiusi da parentesi, queste vanno introdotte nel comando o nell'istruzione. Le parentesi non sono convenzioni utilizzate per descrivere formati; esse sono parti richieste di un comando o di un'istruzione.

- VARIABILE, si riferisce a qualsiasi nome valido di variabile BASIC, come X, A$, oppure T%.

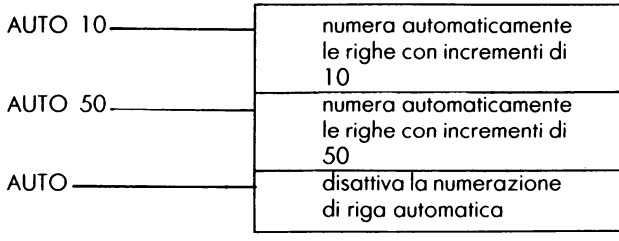
- ESPRESSIONE, significa qualsiasi valida espressione BASIC, come A+B+2 oppure .5*(X+3)

AUTO

AUTO [*riga #*]

Attiva la funzione di numerazione di riga automatica, che facilita l'introduzione dei programmi, battendo i numeri di riga al posto dell'utente. Dopo aver inserito ogni riga di programma ed aver premuto RETURN, il numero della riga successiva viene stampato sullo schermo, col cursore nella posizione da cui si può iniziare a battere quella riga. L'argomento [riga] si riferisce all'incremento tra i numeri di riga. AUTO non seguito dall'argomento, disattiva la numerazione di riga automatica. (Battendo RUN si ottiene lo stesso effetto). Tale istruzione si può eseguire solo in modalità diretta.

ESEMPI:

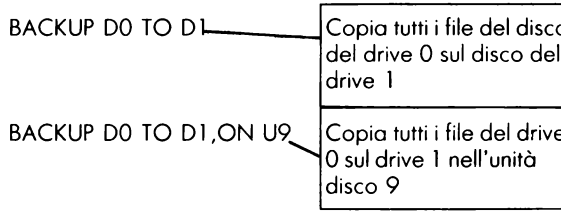


BACKUP

BACKUP D*drive #* TO D*drive #* [,ON U*unità #*]

Tale comando copia tutti i file di un dischetto su un altro dischetto su un sistema con unità disco doppia. È possibile effettuare la copia su un dischetto nuovo senza dover utilizzare prima un comando HEADER, perché il comando BACKUP copia tutte le informazioni contenute nel dischetto, compreso il formato. Si consiglia di effettuare sempre una copia di dischetti importanti, utilizzando il comando BACKUP, per prevenire la perdita o il danneggiamento dell'originale.
Poiché il comando BACKUP formatta anche i dischetti, esso distrugge qualsiasi informazione sul dischetto su cui si stanno copiando le informazioni. Per effettuare questa operazione su un dischetto usato precedentemente, assicurarsi che non contenga programmi che si vogliono conservare. Vedere anche il comando COPY.
NOTA BENE: Questo comando si può utilizzare solo con una unità disco doppia.

ESEMPI:



COLLECT COLLECT [D*drive #*] [,ON U*unità #*]

Usare questo comando per liberare spazio assegnato a file chiusi non correttamente e contemporaneamente per cancellare dall'elenco i riferimenti a questi file.

ESEMPIO:

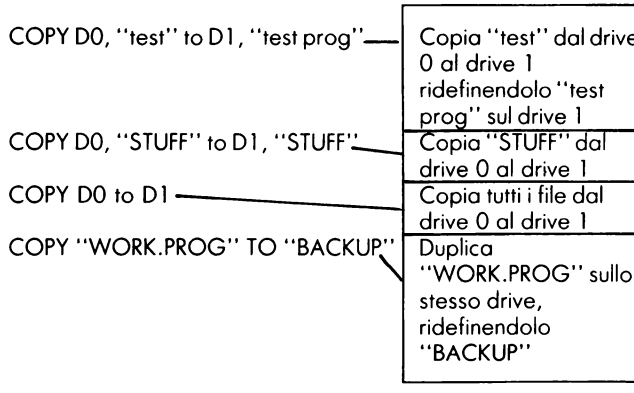
COLLECT D0

CONT (Continua) CONT

Tale comando è utilizzato per riprendere l'esecuzione di un programma che è stato interrotto sia usando il tasto STOP, sia un'istruzione STOP, oppure con un'istruzione END all'interno del programma. Il programma riprenderà l'esecuzione dal punto interrotto. CONT non funzionerà se si sono cambiate o aggiunte righe del programma (o anche se solo si è mosso il cursore ad una riga di programma e si è battuto **RETURN** senza alcuna modifica), se il programma si è fermato a causa di un errore, oppure se si è causato un errore prima di cercare di riprendere l'esecuzione del programma. In tal caso il messaggio di errore è CAN'T CONTINUE ERROR(ERRORE - NON SI PUÒ CONTINUARE).

COPY COPY [D*drive #*,] "*file sorgente*" TO [D*drive #*,] "*altro file*" [,ON U*unità #*]
Copia un file di un disco in un drive (file sorgente) sul disco nell'altro, solamente su un'unità disco doppia, oppure crea una copia di un file sullo stesso drive (con un diverso nome di file).

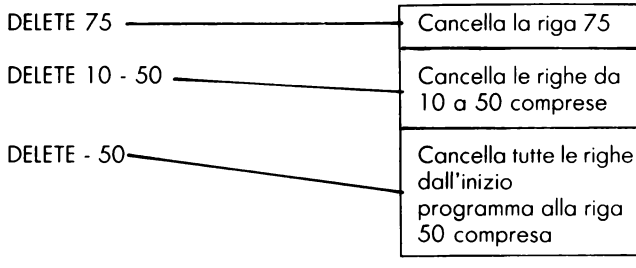
ESEMPI:



DELETE DELETE [*prima riga #*] [– *ultima riga #*]

Cancella le righe del testo BASIC. Questo comando può essere eseguito solo in modalità diretta.

ESEMPI:



DELETE 75 - → Cancella tutte le righe dalla 75 alla fine del programma

DIRECTORY DIRECTORY [D*drive #*], [,U*unità #*] [,"*nome del file*"]

Visualizza l'indice del disco sullo schermo del Plus/4. Usare CTRL-S per interrompere momentaneamente lo scorrimento della visualizzazione (premere qualsiasi altro tasto per riprendere lo scorrimento della visualizzazione). Usare il tasto C= (tasto Commodore) per rallentarlo. Il comando DIRECTORY non si può usare per stampare una copia su carta: per farlo, bisogna caricare l'indice del disco (distruggendo il programma in memoria in quel momento).

ESEMPI:

DIRECTORY → Lista tutti i file del disco

DIRECTORY D1, U9, "WORK" → Lista il file sull'unità disco 9 (8 è il valore di default), drive 1 denominato "work"

DIRECTORY "AB*" → Lista tutti i file che iniziano con le lettere "AB" come ABOVE, ABOARD, ecc.

DIRECTORY D0, "FILE ?.BAK" → Il punto di domanda è una variabile che corrisponde a qualsiasi singolo carattere in quella posizione: i FILE 1.BAK, 2.BAK, 3.BAK, corrispondono tutti a quella stringa.

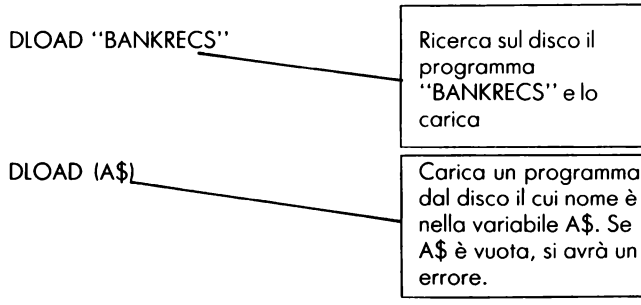
NOTA BENE: per stampare l'indice del drive 0, unità 8, battere:

```
LOAD "$0",8
OPEN4,4:CMD4:LIST
PRINT# 4:CLOSE4
```

DLOAD DLOAD "*nomefile*"[,D*drive #*][,U*unità #*]

Questo comando carica un programma del disco nella memoria corrente. (Usare LOAD per caricare programmi su cassetta). Bisogna fornire un nome di programma.

ESEMPI:

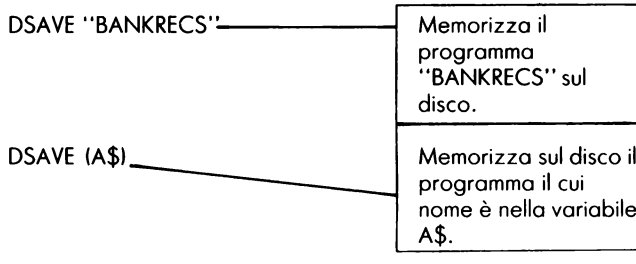


Il comando DLOAD può essere usato all'interno di un programma BASIC per trovare e lanciare un altro programma sul disco. Tale operazione viene definita concatenamento.

DSAVE DSAVE "*nomefile*"[,D*drive #*][,U*unità #*]

Questo comando memorizza un programma sul disco. (Usare SAVE per memorizzare programmi su cassetta). Bisogna fornire un nome di programma.

ESEMPI:



DSAVE "PROG 3",D0,U9

Memorizza il programma "PROG 3" sul drive con il numero di unità 9 (indirizzo di bus seriale).

HEADER HEADER "*nomedisco*"[,I*i.d.* #] [,D*drive* #], [,ON U*unità* #]

Prima di usare un nuovo dischetto per la prima volta, bisogna formattarlo col comando HEADER. Se si vuole cancellare un intero dischetto per riutilizzarlo, si può usare il comando HEADER. Questo comando divide il disco in sezioni denominate blocchi e crea sul disco un indice, denominato elenco o catalogo. Il *nome del disco* può essere un nome qualsiasi composto da un massimo di 16 caratteri. Il numero *i.d.* può essere qualsiasi combinazione di 2 caratteri. Dare ad ogni disco un numero i.d. esclusivo.
Quando si formatta un disco col comando HEADER, fare attenzione perché questo comando cancella tutti i dati memorizzati. Se non si fornisce il numero i.d. si può effettuare una formattazione più veloce; verrà utilizzato il vecchio numero i.d. Il metodo veloce di formattazione si può usare solo se il disco era stato precedentemente formattato, dato che la formattazione veloce cancella solo l'indice invece di formattare il disco.

ESEMPI:

HEADER "MYDISK",I23,D1
HEADER "RECS", I45, D1,U8

HELP HELP

Il comando HELP si utilizza quando nel programma si verifica un errore. Quando si batte HELP la riga in cui è contenuto l'errore viene listata e la parte errata viene visualizzata a caratteri lampeggianti.

KEY KEY [*tasto #, stringa*]

Sul computèr Plus/4 esistono otto (8) tasti funzione disponibili all'utente: 4 senza e 4 con uso del tasto SHIFT. Il Plus/4 permette di definire le funzioni espletate da ogni tasto battuto.
KEY senza alcun parametro specifico fornisce un listato che visualizza tutte le funzioni KEY correnti.
I dati assegnati ad un tasto vengono visualizzati quando quel tasto funzione viene premuto. La lunghezza massima di tutte le definizioni è di 128 caratteri.
Ad un tasto si possono assegnare comandi completi oppure una serie di comandi. Ad esempio:

```
KEY 7, "GRAPHIC0" + CHR$(13) + "LIST" + CHR$(13)
```

ordina al computer di selezionare la modalità testo e dà un listato del programma ogni volta che si preme il tasto "F7" (in modalità diretta). CHR$(13) è il carattere ASCII che indica **RETURN**. Usare CHR$(34) per incorporare una doppia virgoletta nella stringa KEY.
I tasti possono essere ridefiniti in un programma.
Ad esempio:

```
10 KEY 2, "TESTING" + CHR$(34):KEY3, "NO"
10 FORI=1 TO 8:KEY I, CHR$(I+132):NEXT
```

definisce i tasti funzione così come sono definiti sul Commodore 64 e VIC 20

Per riportare tutti i tasti funzione ai loro valori di default, reinizializzare il Plus/4 spegnendolo e riaccendendolo, oppure premere il pulsante RESET.

LIST LIST [*prima riga#*] [-[*ultima riga #*]]

Il comando LIST permette di visualizzare le righe di un programma BASIC che è stato battuto o caricato nella memoria del Plus/4. Quando LIST è utilizzato da solo (senza essere seguito da numeri), il Plus/4 visualizza sullo schermo un listato completo del programma, che può venire rallentato tenendo premuto il tasto C=, interrotto momentaneamente con CTRL -S (la visualizzazione può riprendere premendo un tasto qualsiasi), oppure interrotto col tasto RUN/STOP.
Se si aggiunge un numero di riga dopo la parola LIST, il Plus/4 visualizza solo quel numero di riga.
Se si batte LIST seguito da due numeri separati da una lineetta, il Plus/4 visualizza tutte le righe dal primo al secondo numero di riga.
Se si batte LIST seguito da un numero e dalla sola lineetta, verranno visualizzate tutte le righe da quel numero fino alla fine del programma.
Se si batte LIST, una lineetta e poi un numero, si otterranno tutte le righe dall'inizio del programma fino a quel numero di riga. Con l'uso di queste varianti è possibile esaminare qualsiasi porzione di programma, oppure portare facilmente le righe sullo schermo per modificarle.

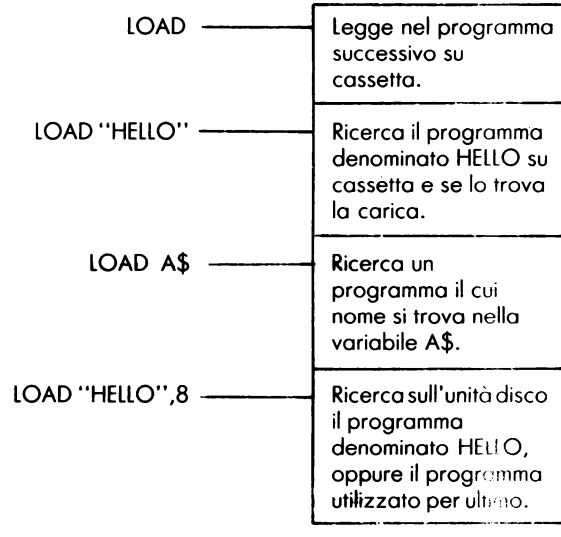
ESEMPI:

LIST	Visualizza l'intero programma.
LIST 100-	Visualizza dalla riga 100 alla fine del programma.
LIST 10	Visualizza solo la riga 10.
LIST-100	Visualizza le righe dall'inizio alla riga 100.
LIST 10-200	Visualizza le righe da 10 a 200 comprese

LOAD LOAD "*nomefile*"[,*dispositivo*]*][,flag di rilocazione*]

Questo è il comando da usare quando si vuole utilizzare un programma archiviato su cassette o su disco.
Battendo solo LOAD e premendo il tasto RETURN, lo schermo del Plus/4 si cancella. Premendo play, il Plus/4 inizia la ricerca di un programma su cassetta.
Quando ne trova uno, il Plus/4 stampa FOUND "*nomefile*". Per caricarlo battere il tasto C=.
Una volta che il programma è stato caricato, si può battere RUN, LIST oppure lo si può modificare.
Si può anche battere la parola LOAD seguita da un nome di programma, che il più delle volte è un nome fra virgolette ("*nome di programma*"). Il nome può essere seguito da una virgola (al di fuori delle virgolette) e da un numero (o variabile numerica), che agisce da numero dell'unità per determinare dove è stato memorizzato il programma (disco o cassetta). Se non è indicato alcun numero, il Plus/4 considera il numero dell'unità 1, che è il mangianastri.
Un altro dispositivo comunemente utilizzato con il comando LOAD è solitamente l'unità disco, che corrisponde al numero di dispositivo 8.

ESEMPI:



Il comando LOAD può essere utilizzato all'interno di un programma per trovare e lanciare il programma successivo su una cassetta.
Quest'operazione è denominata concatenamento.
Il punto della memoria in cui si è caricato un programma viene determinato dal FLAG DI RILOCAZIONE.
Un flag di rilocazione 0 ordina al computer di caricare il programma all'inizio dell'area del programma BASIC e un flag 1 gli ordina di caricare dal punto in cui esso è stato salvato. Il valore di default del flag di rilocazione è 0. Questo viene generalmente utilizzato solo per il caricamento di programmi in linguaggio macchina.

NEW NEW

Questo comando cancella l'intero programma in memoria e azzera tutte le variabili eventualmente utilizzate.
Se il programma non viene preventivamente memorizzato, viene perduto finché non lo si riintroduce, quindi attenzione all'uso di questo comando.
Il comando NEW può anche essere utilizzato come un'istruzione in un programma BASIC. Quando il Plus/4 raggiunge questa riga, il programma viene cancellato ed ogni funzione si interrompe. Ciò non è particolarmente utile in circostanze normali.

RENAME RENAME [,D*drive* ⇔] "*nome vecchio*" TO "*nome nuovo*" [,*Unità* ⇔]

Utilizzato per ridefinire un file su un dischetto.

ESEMPIO:

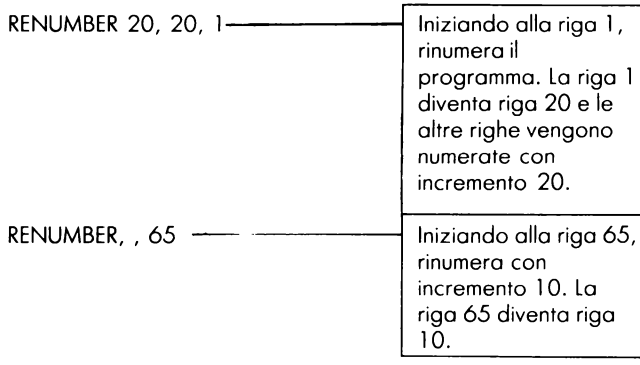
RENAME "TEST" TO "FINALTEST"

Cambia il nome del file da "TEST" a "FINALTEST"

RENUMBER RENUMBER [*riga nuova di inizio* #[,*incremento*[, *vecchia riga di inizio* #]]

La nuova riga di inizio è il numero della prima riga di programma dopo la rinumerazione. Il suo default è 10.
L'incremento è la spaziatura tra i numeri di riga, cioè 10,20,30 ecc. Il default è 10.
Il numero della vecchia riga di inizio è il numero di riga di programma prima della rinumerazione. Ciò permette di rinumerare una parte del programma. Il default è la prima riga del programma.
Questo comando può essere eseguito solo in modalità diretta.

ESEMPI:



RUN RUN [*riga* #]

Dopo che il programma è stato battuto in memoria oppure caricato, il comando RUN lo lancia. RUN cancella tutte le variabili del programma presenti prima di iniziare l'esecuzione. Se il comando RUN non è seguito da alcun numero, il computer inizia dalla riga più bassa del programma.
Se il comando RUN è seguito da un numero, l'esecuzione parte dalla riga corrispondente a quel numero. RUN può essere usato all'interno di un programma.

ESEMPI:

RUN	Lancia il programma dal numero di riga più basso.
RUN 100	Lancia il programma alla riga 100.

SAVE SAVE ["*nomefile*"[,*dispositivo*#[,*flag*EOT]]]

Questo comando memorizza un programma correntemente in memoria su nastro o su disco. Se si batte solo la parola SAVE e si preme RETURN, il Plus/4 inizia a memorizzare il programma su nastro.
Dato che SAVE non ha la possibilità di verificare se in quel punto del nastro è già presente un programma, bisogna fare molta attenzione nell'uso dei nastri.
Se si batte il comando SAVE seguito da un nome tra virgolette oppure da un nome di variabile di stringa, il Plus/4 dà quel nome al programma, in modo che sia più facilmente individuabile e rintracciabile in futuro.
Se si vuole specificare un numero dell'unità per il comando SAVE, fare seguire il nome da una virgola (dopo le virgolette) e da un numero o da una variabile numerica. L'unità 1 è l'unità nastro e il numero 8 corrisponde all'unità disco. Dopo il numero su un comando del nastro, ci può essere una virgola ed un secondo numero che può essere 0, 1 oppure 2. Se il secondo numero è 1 il programma verrà caricato nello stesso punto in cui è stato salvato, e se è 2 il Plus/4 introduce un'indicazione di fine nastro (flag EOT) dopo il programma. Se si sta provando a caricare un programma e il Plus/4 incontra uno di questi indicatori al posto del programma che si cerca di caricare verrà visualizzato il messaggio FILE NOT FOUND ERROR (ERRORE - FILE NON TROVATO).

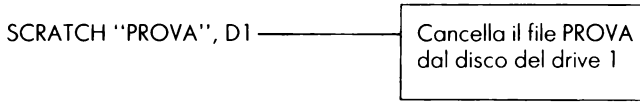
ESEMPI:

SAVE	Memorizza su nastro un programma senza nome
SAVE "HELLO"	Memorizza su nastro con il nome HELLO

SCRATCH SCRATCH "*nome file*"[,D *drive#*][,U *unità #*]

Cancella un file dall'indice del disco. Come precauzione, prima di completare l'operazione, il Plus/4 pone la domanda "are you sure" (confermare). Battere Y per eseguire il comando SCRATCH oppure N per annullare l'operazione. Usare questo comando per cancellare file non desiderati, in modo da creare maggior spazio sul disco.

ESEMPI:

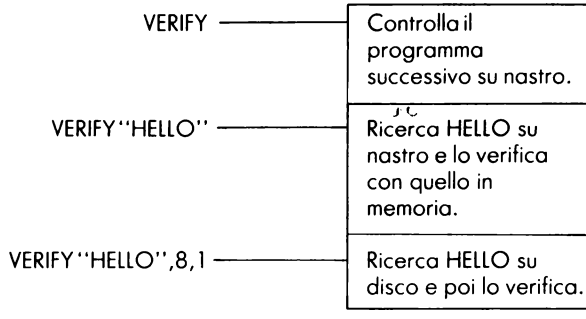


VERIFY VERIFY "*nomefile*"[,*dispositivo #*][,*flag di rilocazione*]

Questo comando permette al Plus/4 di confrontare il programma su nastro o su disco con quello in memoria. Ciò comprova che il programma appena salvato col comando SAVE è stato veramente memorizzato, nel caso in cui il nastro sia difettoso oppure che qualcosa non abbia funzionato.
Questo comando è anche molto utile per posizionare un nastro in modo tale che il Plus/4 scriva subito dopo l'ultimo programma del

nastro. È sufficiente ordinare al Plus/4 di verificare il nome dell'ultimo programma del nastro; dopo aver eseguito quest'operazione il computer informerà l'utente che i programmi non corrispondono (condizione già nota). Ora che il nastro si trova nel punto desiderato, si può memorizzare il programma successivo senza timore di cancellarne uno vecchio. Il solo comando VERIFY ordina al Plus/4 di controllare il programma successivo sul nastro, senza tener conto del nome, e di confrontarlo con quello ora in memoria.
VERIFY, seguito da un nome di programma (tra virgolette) o da una variabile di stringa, ricerca sul nastro quel programma e poi lo verifica.
VERIFY, seguito da un nome, da una virgola e da un numero, verifica il programma che si trova sul dispositivo con quel numero (1 per nastro, 8 per disco). Il flag di rilocazione è lo stesso del comando LOAD.

ESEMPIO:



ISTRUZIONI BASIC

BOX

BOX [*sorgente del colore #*,] *a1, b1, a2, b2,* [[,*angolo*][,*colorazione*]]

Sorgente del colore.....	Sorgente del colore (0-3); il default è 1 (colore di primo piano)
a1, b1.....	Coordinate dell'angolo (scalato)
a2, b2.....	Angolo opposto ad a1, b1 (scalato); il default è il CP
angolo.....	Rotazione in gradi in senso orario; il default è 0 gradi
colorazione.....	Riempie di colore la forma (0: disattivata, 1: attivata); il default è 0

Questo comando permette di tracciare un rettangolo di qualsiasi misura in qualsiasi punto dello schermo. Per ottenere il valore di default includere una virgola senza introdurre un valore. La rotazione è basata sul centro del rettangolo. Il Cursore Pixel (CP) è lasciato su a2, b2 dopo aver eseguito l'istruzione BOX.

ESEMPI:

BOX 1, 10, 10, 60, 60	Traccia il perimetro di un rettangolo
BOX, 10, 10, 60, 60, 45, 1	Traccia un riquadro pieno, ruotato (un rombo)
BOX, 30, 90,,45, 1	Traccia un poligono pieno, ruotato

CHAR

CHAR[*sorgente del colore #*],*x*,*y*[, *stringa*][,*flag di inversione*]
oppure
CHAR[*sorgente del colore #*],*x*,*y*[, *"stringa"*][,*flag di inversione*]

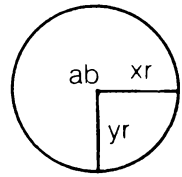
Sorgente del colore.....	Sorgente del colore (0 - 3)
x.....	Colonna del carattere (0 - 39)
y.....	Riga del carattere (0 - 24)
stringa	Stringa da stampare
inversione.....	Flag di inversione campo (0 = disattivata, 1 = attivata)

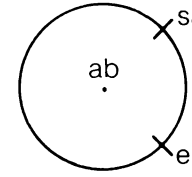
Il testo (stringhe alfanumeriche) può essere visualizzato su qualsiasi schermo, nel punto dato dal comando CHAR.
Il Plus/4 legge i dati del carattere nell'area ROM per i caratteri. Fornire le coordinate x e y della posizione di partenza e la stringa di testo che si vuole visualizzare; la sorgente del colore e la rappresentazione invertita sono opzionali.
Se la stringa supera il margine destro dello schermo, proseguirà sulla riga successiva.
Quando utilizzata in modalità TESTO, la stringa stampata con il comando CHAR funziona esattamente come una stringa PRINT, inclusi l'inversione campo, i tasti del cursore, flash on/off, ecc. Queste funzioni di controllo all'interno della stringa non operano quando il comando CHAR viene utilizzato per visualizzare il testo in modalità GRAFICA.

NOTA: in modalità multicolor, per visualizzare un carattere nel multicolor 2, portare la sorgente del colore su 0 e il flag di inversione su 1. Per visualizzare un carattere in multicolor 1, impostare la sorgente del colore su 0 e il flag di inversione su 0.

CIRCLE CIRCLE [*cs*],[*a,b*], *xr* [,[*yr*][, [*sa*] [, [*ea*] [, [*angolo* [, *inc*]]]]]

cs	Sorgente del colore (0 - 3)
a,b	Coordinate del centro (scalate) (il default è alla posizione del cursore pixel (CP))
xr	Raggio X (scalato)
yr	Raggio Y (il default è xr)
sa	Angolo di partenza arco (il default è 0)
ea	Angolo di arrivo arco (il default è 360)
angolo	Rotazione in senso orario in gradi (il default è 0 gradi)
inc	Gradi tra segmenti (il default è 2 gradi)





Il comando CIRCLE permette di tracciare un cerchio, un ellisse, un arco, un triangolo o un ottagono. La coordinata finale si trova sulla circonferenza del cerchio, all'angolo di arrivo dell'arco.
Tutte le rotazioni partono dal centro.
Dando valori uguali al raggio Y e al raggio X, la figura risultante non sarà un cerchio, perché le coordinate X e Y sono su scale diverse.
Gli archi vengono tracciati dall'angolo di partenza a quello di arrivo in senso orario. L'incremento del segmento controlla l'irregolarità della forma, valori inferiori di inc creano forme più arrotondate.

ESEMPI:

CIRCLE , 160,100,65,10	Traccia un'ellisse
CIRCLE , 160,100,65,50	Traccia un cerchio
CIRCLE , 60,40,20,18,,,,45	Traccia un ottagono
CIRCLE , 260,40,20,,,,,90	Traccia un rombo
CIRCLE , 60,140,20,18,,,,120	Traccia un triangolo

CLOSE CLOSE *file #*

Questo comando completa e chiude qualsiasi file utilizzato dalle istruzioni OPEN. Il numero che segue la parola CLOSE è il numero del file da chiudere.

ESEMPIO:

CLOSE 2 — Il file logico 2 viene chiuso

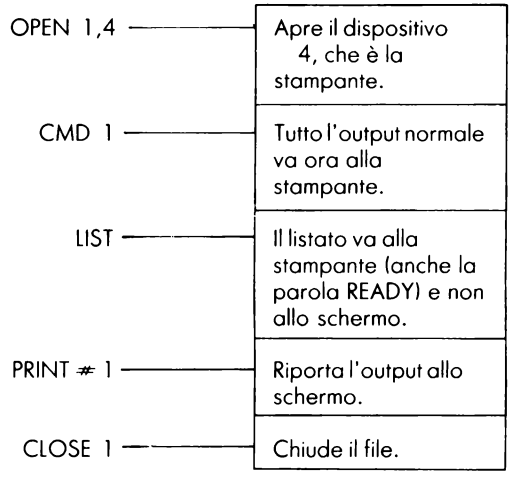
CLR CLR

Questo comando cancella qualsiasi variabile presente in memoria lasciando intatto il programma stesso.
Questo comando si esegue automaticamente quando viene dato un comando RUN o NEW, oppure quando si esegue una correzione.

CMD CMD *file* # [,*stringa di testo*]

CMD invia l'output che normalmente andrebbe allo schermo (ad esempio, un'istruzione PRINT, produrrà sempre un listato ma non sullo schermo) ad un altro dispositivo che potrebbe essere una stampante o un file di dati su nastro o su disco. Questo dispositivo o file deve essere inizialmente aperto con un'istruzione OPEN.
Il comando CMD deve essere seguito da un numero o da una variabile numerica che faccia riferimento al file.

ESEMPI:



COLOR COLOR *sorgente* #, *colore* # [,*luminanza* #]

Assegna un colore ad una delle 5 sorgenti di colore:

Numero	Sorgente
0	sfondo
1	primo piano
2	multicolor 1
3	multicolor 2
4	cornice

I colori che si possono utilizzare sono quelli compresi nella gamma tra 1 e 16 (BIANCO, NERO ...). Come opzione si può includere il livello di luminanza 0 - 7, dove 0 è il livello minimo e 7 è il livello massimo. Il valore default della luminanza è 7. La luminanza permette di selezionare tra otto livelli di luminosità per tutti i colori tranne il nero.

DATA DATA *costanti separate da virgole*

Questo comando è seguito da un elenco di elementi che devono essere utilizzati dalle istruzioni READ.
Gli elementi possono essere numeri o parole e sono separati da virgole. Non è necessario che le parole siano tra virgolette a meno che non contengano alcuni dei seguenti caratteri: SPAZIO, due punti, o virgola.
Se tra due virgole non è riportato nulla, il valore attribuito a un numero sarà zero, oppure una stringa vuota.
Considerare anche l'istruzione RESTORE, che permette al Plus/4 di rileggere i dati.

ESEMPIO:

DATA 100, 200, FRED, "CIAO,MAMMA", , 3, 14, ABC123

DEF FN DEF FN *nome (variabile) = espressione*

(DEFine FunctioN)

Questo comando permette di definire come funzione dei calcoli complessi.
Nel caso di una lunga formula, che deve essere frequentemente usata all'interno di un programma, ciò permette di risparmiare molto spazio. Il nome che si dà alla funzione inizia con le lettere FN, seguita da qualsiasi nome di variabile numerica valida. Per prima cosa bisogna definire la funzione utilizzando l'istruzione DEF seguita dal nome che si è dato alla funzione. Il nome deve essere seguito da una variabile numerica (in questo caso X) racchiusa tra parentesi. Segue poi un segno di uguale e quindi la formula che si vuole definire. Si può "chiamare" la formula sostituendo qualsiasi numero con X, usando il formato descritto alla riga 20 del seguente esempio:

ESEMPIO:

```
10 DEF FNA(X)=12*(34.75-X/.3)+X

20 PRINT FNA (7)
```

Il numero 7 è inserito in ogni punto in cui si trova una X nella formula data nell'istruzioneDEF.

DIM DIM *variabile (indici)* [, *variabile(indici)*] ...

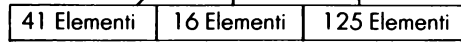
Prima di poter usare una matrice di variabili, il programma deve eseguire una istruzione DIM per stabilire una dimensione di quella matrice (a meno che non ci siano 11 o meno elementi nella matrice). L'istruzione DIM è seguita dal nome della matrice, che può essere qualsiasi nome di variabile valida. Poi, tra parentesi, si mette il numero (o la variabile numerica) degli elementi in ogni dimensione. Una matrice con più di una dimensione è detta matrix.
Si può usare qualsiasi numero di dimensioni, bisogna però ricordare che l'intera lista di variabili creata, occupa uno spazio nella memoria, da cui è facile uscire se si introducono troppi dati.

Per calcolare il numero di variabili creato con ciascun DIM bisogna moltiplicare il numero totale di elementi in ogni dimensione della matrice. (Ogni matrice inizia con l'elemento 0).

NOTA: Una matrice a numero intero (cifra-unica) occupa i 2/5 dello spazio di una matrice a virgola mobile.

ESEMPIO:

```
10 DIM A$ (40),B7(15),CC%(4,4,4)
```



È possibile dimensionare più di una matrice in una istruzione DIM separando le matrici con una virgola. Se il programma esegue un'istruzione DIM per qualsiasi matrice più di una volta, si otterrà un messaggio di errore che indica che si vuole ridimensionare la matrice. È una buona norma di programmazione posizionare istruzioni DIM vicino all'inizio del programma.

DO/LOOP/WHILE UNTIL/EXIT

DO [UNTIL *argomento booleano* WHILE *argomento booleano*] *istruzioni*[EXIT]

LOOP [UNTIL *argomento booleano* WHILE *argomento booleano*]

Esegue le istruzioni, comprese fra l'istruzione DO e l'istruzione LOOP. Se né UNTIL, né WHILE modificano l'istruzione DO o LOOP, l'esecuzione delle istruzioni contenute nel loop continua illimitatamente. Se si incontra un'istruzione EXIT durante l'esecuzione di un DO LOOP, tale esecuzione viene trasferita alla prima istruzione che segue l'istruzione LOOP.
I loop DO si possono nidificare seguendo le regole relative ai loop FOR-NEXT.
Se si utilizza un parametro UNTIL, il programma continua ad eseguire loop fino a che l'argomento booleano è soddisfatto (diventa VERO). Basilarmente, il parametro WHILE è l'opposto del parametro UNTIL: il programma continua ad eseguire dei loop finché l'argomento booleano è vero.
Un esempio di argomento booleano è A=1, oppure G>=65.

ESEMPIO:

```
DO UNTIL X=0 OR X=1
:
LOOP
DO WHILE A$=" ":GET A$:LOOP
```

DRAW **DRAW** *sorgente del colore#* [, *a1, b1,* [**TO***a2, b2 ...*]]

Questo comando permette di tracciare singoli punti, linee e forme. Si devono fornire la fonte del colore (0-3), i punti di partenza (a1, b1) e quelli di arrivo (a2, b2).

ESEMPI:

un punto:	DRAW 1, 100, 50 – non specificando il punto di arrivo, viene preso il valore di default a1, b1 per a2, b2 per creare un punto
linee:	DRAW, 10,10 TO DRAW, TO 25, 30 100,60
una forma:	DRAW , 10,10 TO 10,60 TO 100,60 TO 10,10

END END

Quando il programma esegue un'istruzione END, il programma si interrompe immediatamente. Usando il comando CONT, si re-inizia il programma dall'istruzione successiva all'istruzione END.

FOR...TO...STEP FOR *variabile* = *valore iniziale* TO *valore finale* [STEP *incremento*]

Questa istruzione funziona con NEXT per impostare una sezione del programma che si ripeta per uno stabilito numero di volte. Si potrebbe far si che il Plus/4 conti fino ad un numero elevato, in modo che il programma si interrompa per alcuni secondi nel caso in cui siano necessari dei calcoli oppure che si debba ripetere qualche funzione per un certo numero di volte (ad esempio stampare).
La variabile di loop è la variabile che si aggiunge o si sottrae durante un loop FOR/NEXT. I valori iniziali e finali sono i conteggi iniziali e finali della variabile di loop.
La logica dell'istruzione FOR è la seguente: per prima cosa la variabile di loop viene stabilita come valore iniziale. Quando il programma raggiunge una riga con il comando NEXT, questo aggiunge l'incremento STEP (default = 1) al valore della variabile di loop e controlla che sia maggiore del valore loop finale. Se non è maggiore, la riga eseguita successivamente è l'istruzione che viene subito dopo l'istruzione FOR. Se la variabile di loop è maggiore del numero di loop finale, allora l'istruzione eseguita successivamente è quella che segue l'istruzione NEXT. Vedere anche l'istruzione NEXT.

ESEMPIO:

```
10 FOR L = 1 TO 10
20 PRINT L
30 NEXT L
40 PRINT "SONO FINITO! L = "L
```

Questo programma stampa sullo schermo i numeri da 1 a 10, seguiti dal messaggio SONO FINITO! L = 11.
Il valore di loop finale può essere seguito dalla parola STEP e altri numeri o variabili. In questo caso, il valore che segue STEP viene aggiunto ogni volta anziché una volta sola. Ciò permette di contare alla rovescia, tramite frazioni o in qualsiasi modo necessario.
È possibile predisporre un loop all'interno di un altro. Ciò viene definito nidificazione di loop. Fare attenzione a nidificare i loop in modo che l'ultimo loop eseguito sia il primo a terminare.

ESEMPIO DI LOOP NIDIFICATO:

```
10 FOR L = 1 TO 100
20 FOR A = 5 TO 11 STEP 2


30 NEXT A
40 NEXT L
```

Questo loop FOR...NEXT è "nidificato" all'interno di quello più grande.

GET GET *variabili*

L'istruzione GET è un modo per ottenere dati dalla tastiera un carattere alla volta. Quando si esegue GET, viene ricevuto il carattere battuto. Se non si è battuto nulla, allora appare un carattere nullo (vuoto) e il programma continua senza aspettare un nuovo tasto. Non è necessario battere il tasto **RETURN**, dato che infatti lo si può ricevere con GET. La parola GET è seguita da un nome di variabile, solitamente una variabile di stringa.
Se si è usato un numero e si è battuto qualsiasi tasto che non sia un numero, il programma si interromperà con un messaggio d'errore.
L'istruzione GET si può anche inserire in un loop che controlla un risultato vuoto, in attesa della battitura di un tasto per continuare. In questo caso si potrebbe anche usare l'istruzione GETKEY. Tale comando può essere eseguito solo all'interno di un programma.

ESEMPIO:

```
10 GET A$:IF A$<> "A" THEN 10
```

Per proseguire, questa riga attende la battitura del tasto A

GETKEY GETKEY *variabili*

Il comando GETKEY è molto simile al comando GET. A differenza di GET, GETKEY attende che l'utente batta un carattere sulla tastiera. Ciò permette di usare quest'istruzione con lo scopo di far introdurre un carattere alla volta. Questo comando può essere eseguito solo all'interno di un programma.

ESEMPIO:

```
10 GETKEY A$
```

Questa riga attende che si batta un tasto. Qualsiasi tasto battuto farà proseguire il programma.

GET # GET # *numero di file, variabili*

Utilizzato con un dispositivo o file precedentemente aperto, serve per introdurre un carattere alla volta. Altrimenti, funziona come il comando GET.
Questo comando può essere eseguito solo in un programma.

ESEMPIO:

```
GET # 1,A$
```

GOSUB GOSUB *riga #*

Questo comando è simile al comando GOTO, solo che il Plus/4 tiene presente la riga di provenienza. Quando incontra una riga con un comando RETURN, il programma torna immediatamente al comando che segue GOSUB. L'oggetto di un comando GOSUB è denominato subroutine. Una subroutine è utile se nel programma è presente una routine che può essere utilizzata da varie differenti parti del programma. Invece di continuare a duplicare la sezione del

programma, la si può impostare come subroutine e saltare a questa dalle diverse parti del programma. Vedere anche il comando RETURN.

ESEMPIO:

```
20 GOSUB 800
:
:
800 PRINT "SALVE":RETURN
```

significa che bisogna andare alla subroutine cominciando dalla riga 800 ed eseguirla.

GOTO o GO TO GOTO *riga #*

Dopo che si è eseguito il comando GOTO, la riga che verrà successivamente eseguita sarà quella con il numero di riga fornito dopo la parola GOTO.
Quando è utilizzato in modalità diretta, GOTO riga permette di iniziare l'esecuzione del programma al numero di riga dato senza azzerare le variabili.

ESEMPIO:

```
10 PRINT "COMMODORE"
20 GOTO 10
```

GOTO alla riga 20 fa ripetere ininterrottamente la riga 10 finché si preme **RUN/STOP** .

GRAPHIC GRAPHIC < *modalità* [,*opzione di azzeramento*]/CLR>

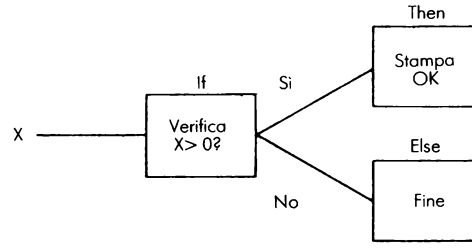
Questa istruzione imposta il Plus/4 in una delle 5 modalità grafiche.

modalità	descrizione
0	testo normale
1	grafici ad alta risoluzione
2	grafici ad alta risoluzione, schermo diviso
3	grafici multi-color
4	grafici multi-color, schermo diviso

Quando viene eseguito, GRAPHIC 1-4 posiziona un'area di 10K a bit-map e l'inizio dell'area di testo BASIC si sposta al di sopra dell'area di alta risoluzione.
Quest'area rimane allocata anche se l'utente ritorna alla modalità TESTO (GRAPHIC 0). Se nel comando GRAPHIC si dà 1 come secondo argomento, lo schermo viene anche cancellato. L'esecuzione di un comando GRAPHIC CLR rilascia l'area di 10K a bit-map e la rende nuovamente disponibile per il testo BASIC e le variabili.

IF...THEN [...:ELSE] IF *espressione* THEN *clausola then* [:ELSE *clausola else*]

IF...THEN permette al computer di analizzare un'espressione BASIC preceduta da IF e di scegliere uno dei due possibili sviluppi dell'azione. Se l'espressione è vera, viene eseguito il comando che segue THEN. Questa espressione può essere qualsiasi istruzione BASIC. Se l'espressione è falsa, il programma prosegue direttamente alla riga successiva, a meno che non sia presente una clausola ELSE.
L'espressione valutata può essere una variabile o una formula, per cui è considerata vera se non è zero e falsa se è zero. Nella maggior parte dei casi si tratta di un'espressione con operatori relazionali (=, <, >, <=, >=, <>, AND, OR, NOT).
Se la calusola ELSE è presente, deve trovarsi sulla stessa riga di IF-THEN. Quando una clausola ELSE è presente, questa viene eseguita quando la clausola THEN non viene eseguita. In altre parole, la clausola ELSE si esegue quando l'espressione IF è FALSA.



ESEMPIO:

```
50 IF X>0 THEN PRINT "OK":ELSE END
```

Verifica il valore di X. Se X è maggiore di 0 la clausola THEN viene eseguita, mentre la clausola ELSE no. Se X è minore di 0, la clausola ELSE viene eseguita, mentre la clausola THEN no.

INPUT INPUT [*stringa di prompt;*] *variabili*

L'istruzione INPUT permette al computer di richiedere dati all'utente e di introdurli in una variabile oppure in variabili. Il programma si interrompe, stampa un punto interrogativo (?) sullo schermo e attende che l'utente batta la risposta e prema il tasto **RETURN**.
La parola INPUT è seguita da un nome di variabile o da una lista di nomi di variabili separate da virgole. Ci può essere un messaggio fra virgolette prima della lista di variabili da introdurre. Se questo messaggio (denominato prompt) è presente, ci deve essere un punto e virgola (;) dopo le virgolette del prompt. Quando si deve introdurre più di una variabile, queste devono essere introdotte separate da virgole, in caso contrario, il computer richiede i restanti valori stampando due punti interrogativi (??). Se si preme **RETURN** senza introdurre un valore, la variabile INPUT mantiene il valore precedentemente introdotto per quella variabile.

Questa istruzione può essere eseguita solo all'interno di un programma.

ESEMPIO:

```
10 INPUT "BATTI UN NUMERO PREGO";A
20 INPUT "E IL TUO NOME";A$
30 INPUT B$
40 PRINT "SCOMMETTO CHE NON IMMAGINAVI COSA VOLEVO!"
```

INPUT # INPUT *# numero file, variabili*

Quest'istruzione funziona come INPUT ma prende i dati da un dispositivo o file precedentemente aperto.
Non è permessa alcuna stringa di prompt. Questo comando può essere utilizzato solo in modalità di programma.

ESEMPIO:

```
INPUT# 2, A$, C, D$
```

LET [LET] *variabile espressione*

La parola LET si usa molto raramente in un programma, dato che non è necessaria, tuttavia l'istruzione in se stessa è il nucleo di tutti i programmi BASIC. Ogni volta che una variabile è definita o le viene dato un valore, LET è sempre sottintesa. Il nome di variabile da assegnare al risultato di un calcolo si trova alla sinistra del segno uguale ed il numero o la formula si trova sulla destra.

ESEMPIO:

```
10 LET A = 5
20 B = 6
30 C = A * B + 3
40 D$ = "HELLO"
```

LOCATE LOCATE *coordinata-x, coordinata-y*

Il comando LOCATE permette di posizionare il cursore pixel (CP) in qualsiasi punto dello schermo. Il CP è la posizione corrente del punto iniziale del disegno successivo. A differenza del cursore normale, il CP non è visibile, tuttavia lo si può spostare con il comando LOCATE: ad esempio:

LOCATE 160, 100

posiziona il CP nel centro dello schermo ad alta risoluzione. Non si vedrà nulla finché non si inizierà effettivamente il disegno.
È possibile rintracciare in qualsiasi momento la posizione del CP utilizzando la funzione RDOT (0) per ottenere la coordinata-X e RDOT (1) per la coordinata-Y. La sorgente del colore del punto alla posizione CP può essere individuata stampando RDOT (2). (In tutti i comandi per disegnare in cui sia disponibile l'opzione del colore, si può selezionare un valore da 0 a 3, corrispondente allo sfondo, al primopiano, multicolor 1 o multi-color 2 come sorgente di colore).

MONITOR MONITOR

Questo comando porta dal BASIC al programma monitor in linguaggio macchina incorporato. Il monitor è usato per sviluppare, effettuare la messa a punto ad eseguire programmi in linguaggio macchina più facilmente che con il BASIC. Per ulteriori informazioni vedere la sezione sui comandi del monitor. (Quando si è nel monitor, battere una X e premere **RETURN** per ritornare al BASIC).

NEXT NEXT[*variabile,...variabile*]

L'istruzione NEXT è utilizzata con l'istruzione FOR. Quando il computer incontra un'istruzione NEXT, esso ritorna alla istruzione FOR corrispondente e verifica la variabile loop. (Per maggiori dettagli vedere l'istruzione FOR).
Quando il loop è terminato, l'esecuzione procede con l'istruzione successiva a NEXT.

La parola NEXT può essere seguita da un nome di variabile, da una lista di nomi di variabile separati da virgole, oppure da nessun nome di variabile. Se non ci sono listati di nomi, l'ultimo loop iniziato verrà completato. Se vengono fornite più variabili, verranno esaminate nell'ordine da sinistra verso destra.

ESEMPIO:

```
10 FOR L = 1 TO 10:NEXT
20 FOR L = 1 TO 10:NEXTL
30 FOR L = 1 TO 10:FOR M = 1 TO 10: NEXT M, L
```

ON ON *espressione*<GOTO/GOSUB> *riga # 1* [, *riga# 2*,...]

Questo comando può trasformare le istruzioni GOTO e GOSUB in versioni speciali dell'istruzione IF.
La parola ON è seguita da una formula, poi da GOTO oppure da GOSUB e da un listato di numeri di riga separati da virgole. Se il risultato del calcolo della formula (espressione) è 1, viene eseguita la prima riga del listato.
Se il risultato è 2, viene eseguito il secondo numero di riga del listato, e così via.
Se il risultato è 0 oppure maggiore del numero di numeri di riga del listato, la riga eseguita successivamente è l'istruzione che segue ON. Se il numero è negativo, risulta un ILLEGAL QUANTITY ERROR (ERRORE-QUANTITÀ ILLEGALE).

ESEMPIO:

```
10 INPUT X:IF X < 0 THEN 10
```

Se X = 1, ON invia il controllo al primo numero contenuto nella riga 20 e cioè 50

```
20 ON X GOTO 50, 30, 30, 70

25 PRINT "CADDE ATTRAVERSO":GOTO 10
```

Se X = 2, ON invia il controllo al secondo numero (30), ecc.

```
30 PRINT "TROPPO ALTO": GOTO 10
```

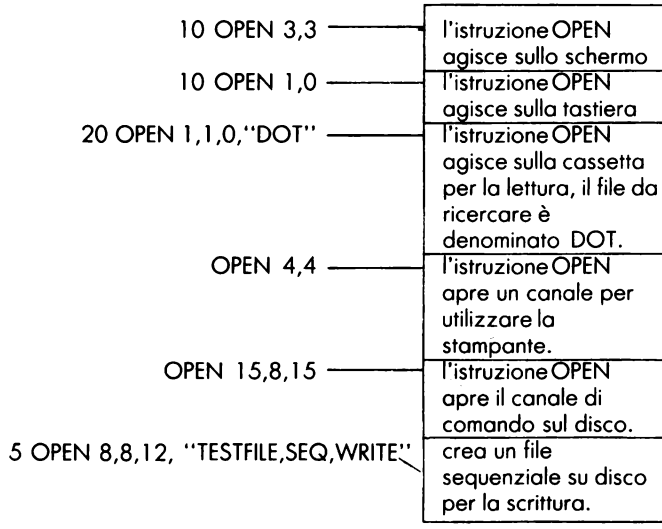
```
50 PRINT "TROPPO BASSO": GOTO 10

70 END
```

OPEN OPEN *file #, dispositivo #* [,*indirizzo secondario*][,"*nomefile, tipo, modalità*"]

L'istruzione OPEN permette al Plus/4 di accedere a dispositivi quali il registratore e il disco per i dati, alla stampante oppure anche allo schermo del Plus/4.
La parola OPEN è seguita da un numero logico di file, che è il numero a cui faranno riferimento tutte le altre istruzioni BASIC. Questo numero và da 1 a 255. C'è sempre un secondo numero dopo il primo, detto numero del dispositivo.
Il numero di dispositivo 0 è la tastiera del Plus/4, 3 è lo schermo del Plus/4, 1 è il registratore a cassette, 4 è la stampante, di solito 8 è il disco. Uno zero (0) può essere incluso prima del numero del dispositivo (ad esempio, 08 e 8 per il Plus/4 sono intercambiabili). Si consiglia di usare il numero del dispositivo come numero di file, in modo che sia facilitata l'individuazione del file oppure del dispositivo. Dopo il secondo numero ce ne può essere un terzo denominato indirizzo secondario. Riferito alla cassetta, questo può essere 0 per leggere, 1 per scrivere e 2 per scrivere con l'indicatore di fine nastro alla fine.
Riferito al disco, il numero indicato è quello del canale. Nella stampante, gli indirizzi secondari vengono utilizzati per stabilire la modalità della stampante. Per ulteriori informazioni vedere la Guida di Consultazione Per il Programmatore oppure il manuale di ogni specifico dispositivo per maggiori dettagli sugli indirizzi secondari.
Dopo il terzo numero si può avere anche una stringa, che potrebbe essere un comando all'unità disco, oppure il numero del file su nastro o su disco.
Il tipo e la modalità si riferiscono solamente ai file disco (i tipi di file sono prg, seq, rel e usr; le modalità sono leggere e scrivere).

ESEMPI:



Vedere anche le istruzioni CLOSE, CMD, GET #, INPUT#, e PRINT#, le variabili di sistema ST, DS, e DS$.

PAINT PAINT [*sorgente del colore*][,[*a,b*][,*modalità*]]

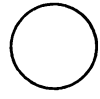
Sorgente del colore.... (0-3);(il default è 1, colore di primopiano)
a,b..... coordinata iniziale, scalata (il default è al CP)
modalità..... 0 = colora un'area definita dalla sorgente del colore selezionata
1 = colora un'area definita da qualsiasi sorgente non di sfondo.

Il comando PAINT permette di riempire di colore un'area. Esso riempie l'area circostante il punto specificato, fino a che non incontra un limite dello stesso colore (oppure qualsiasi colore di non-sfondo, a seconda della modalità prescelta).
La posizione finale del CP sarà nel punto iniziale (a,b).

NOTA: Se il punto iniziale è dello stesso colore della sorgente del colore prescelta (oppure qualsiasi non-sfondo, quando viene utilizzata la modalità 1), non avvengono cambiamenti.

ESEMPIO:

```
10 CIRCLE, 160,100,65,50 ——————→ traccia una circonferenza
```



```
20 PAINT, 160,100 ——————→ riempie di colore la circonferenza
```

POKE POKE *indirizzo, valore*

Il comando POKE permette di cambiare qualsiasi valore della memoria RAM del Plus/4 e permette di modificare numerosi registri Input/Output del Plus/4.
POKE è sempre seguito da due numeri (o equazioni). Il primo numero è una locazione all'interno della memoria del Plus/4, con qualsiasi valore da 0 a 65535. Il secondo numero è un valore da 0 a 255 che viene dato a quella locazione, sostituendo qualsiasi valore precedentemente attribuito.

ESEMPIO:

```
10 POKE 28000,8 ——————→ Imposta la locazione 28000 al valore 8
20 POKE 28*1000,27 ——————→ Imposta la locazione 28000 al valore 27
```

NOTA: PEEK, un comando relativo a POKE, è elencato nelle FUNZIONI.

PRINT PRINT *listatodistampa*

L'istruzione PRINT è l'istruzione più importante del BASIC.
Anche se PRINT è l'istruzione BASIC che si impara ad usare per prima, vi sono ugualmente numerose sottigliezze da approfondire. La parola PRINT può essere seguita da qualsiasi delle seguenti espressioni:

Caratteri all'interno di virgolette	("righe di testo")
Nomi di variabile	(A, B, A$, X$)
Funzioni	(SIN(23), ABS(33))
Punteggiatura	(; ,)

I caratteri all'interno di virgolette sono spesso definiti letterali, perché vengono stampati esattamente come appaiono. Il valore contenuto dai nomi di variabile (un numero o una stringa) viene stampato. Vengono stampati anche i valori del numero delle funzioni. La punteggiatura è utilizzata per facilitare una chiara formattazione di dati sullo schermo. La virgola divide lo schermo in 4 colonne per i dati, mentre il punto e virgola non aggiunge spazi. Entrambi i segni possono essere utilizzati quale ultimo simbolo dell'istruzione, dando come risultato la stampa dell'istruzione PRINT successiva, sulla stessa riga dell'istruzione PRINT precedente.

ESEMPIO:

```
                                        RISULTATO
10 PRINT "HELLO"                        HELLO
20 A$="THERE":PRINT"HELLO," A$          HELLO, THERE
30  A=4:B=2:PRINT A+B                   6
50 J=41:PRINT J;:PRINT J-1              41  40
60 C=A+B:D=A-B:PRINT A;B;C,D            4   2   6          2
```

Vedere anche: FUNZIONI POS(), SPC(), e TAB().

PRINT # PRINT *# file #, listato di stampa*

Vi sono alcune differenze tra questa istruzione e PRINT.
Innanzitutto la parola PRINT # è seguita da un numero che si riferisce al

dispositivo oppure al file di dati precedentemente aperti.
Il numero è seguito da una virgola e da un listato da stampare. La virgola e il punto e virgola agiscono per la spaziatura come nell'istruzione PRINT. Alcuni dispositivi possono non funzionare con TAB e SPC.

ESEMPIO:

```
100 PRINT #1, "HELLO THERE!",A$,B$,
```

PRINT USING PRINT [# *numerofile*]USING *listato del formato; listato di stampa;*

Queste istruzioni permettono di definire il formato della stringa e degli elementi numerici che si vuole stampare sullo schermo, sulla stampante, oppure su un altro dispositivo.
Il formato desiderato va racchiuso tra virgolette. Questo è il listato del formato. Quindi aggiungere un punto e virgola ed una lista di ciò che si vuole stampare nel formato del listato di stampa. La lista può contenere variabili oppure gli effettivi valori che si vuol stampare.
Ad esempio:

```
 5 X=32: Y=100.23: A$="CAT"
10 PRINT USING "$##.## ";13.25,X,Y
20 PRINT USING "## ##>## ";"CBM",A$
```

Quando viene lanciato questo programma, la riga 10 stampa:

```
$13.25    $32.00    $*****
```

stampa***** invece del valore Y, perché Y è di 5 cifre, il chè non è conforme al listato del formato (come spiegato qui di seguito).

La riga 20 stampa:

```
          CMB    CAT
```

lascia tre spazi prima di stampare "CBM", come definito nel listato del formato

CARATTERE	NUMERICA	STRINGA
Diesis (#)	X	X
Segno più (+)	X	
Segno meno (-)	X	
Virgola decimale (.)	X	
Virgola (,)	X	
Simbolo del dollaro ($)	X	
Quattro frecce (↑↑↑↑)	X	
Segno uguale (=)		X
Segno maggiore di (>)		X

Il simbolo di diesis (#) riserva spazio per un singolo carattere nel campo di output. Se l'elemento di dati contiene più caratteri di quanti siano presenti nel campo del formato, accade che:

in caso di un elemento numerico, l'intero campo si riempie di asterischi (*). Non viene stampato alcun numero.

Ad esempio:

```
10PRINTUSING"# # # # ",X
```

A causa di questi valori di X, questo formato visualizza:

```
A = 12.34          12
A = 567.89        568
A = 123456        ****
```

A causa di un elemento di STRINGA, i dati della stringa vengono troncati ai bordi del campo. Vengono stampati solo tanti caratteri quanti simboli diesis (#) si trovano nell'elemento del formato. Il troncamento avviene sulla destra.
I segni più (+) e meno (-) possono essere usati nella prima o nell'ultima posizione del campo del formato, ma non in entrambe. Se il numero è positivo viene stampato il segno più; se negativo, il segno meno.
Se si utilizza il segno meno per un numero positivo, verrà emesso uno spazio vuoto nella posizione del carattere indicata dal segno meno.
Se per un elemento di dati numerici non vengono utilizzati nè il segno più né il segno meno nel campo del formato, verrà stampato un segno meno davanti alla prima cifra o al simbolo del dollaro se il numero è negativo; se il numero è positivo non verrà stampato alcun segno.
Ciò significa che se il numero è positivo si può stampare un ulteriore carattere.

Se vi sono troppe cifre da collocare nel campo specificato dal simbolo # e dai segni +/-, incorre allora un overflow e il campo si riempie di asterischi (*).
Il simbolo della virgola decimale (.) designa la posizione della virgola decimale in un numero. Si può avere una sola virgola decimale in qualsiasi campo di formato. Se non si specifica una virgola decimale nel campo di formato, il valore viene arrotondato all'intero più vicino e poi stampato senza posizioni decimali.
Quando si specifica una virgola decimale, il numero di cifre che precede la virgola decimale (incluso il segno meno, se il valore è negativo) non deve superare il numero di # presenti prima della virgola decimale. Se le cifre sono troppe, si verificherà un overflow e il campo si riempirà di asterischi (*).
Una virgola (,) permette l'introduzione di virgole nei campi numerici. La posizione della virgola nel listato del formato indica il punto in cui la virgola appare nel numero stampato.
Vengono stampate solo le virgole all'interno di un numero.
Virgole non utilizzabili appaiono alla sinistra della prima cifra in qualità di carattere di riempimento. Almeno un # deve precedere la prima virgola in un campo.
Se si specificano le virgole in un campo e il numero è negativo, allora viene stampato un segno meno come primo carattere, anche se la posizione del carattere è specificata come virgola.

ESEMPI:

CAMPO	ESPRESSIONE	RISULTATO	COMMENTO
##.##+	-.01	0.01-	Aggiunta di uno 0 iniziale
##.##-	1	1.0	Aggiunta di uno 0 finale
####	-100.5	-101	Arrotondamento senza decimali
####	-1000	****	Overflow di campo (4 cifre più il segno meno immessi)
###.	10	10.	Aggiunta di virgola decimale
#$##	1	$1	Aggiunta di segno $ iniziale

Inserendo un simbolo di dollaro ($) questo verrà stampato nel numero. Se si vuole che il simbolo del dollaro sia mobile (cioè che venga sempre situato prima del numero) è necessario specificare almeno un # prima del simbolo dollaro.
Se si specifica un segno dollaro senza un # iniziale, il segno dollaro verrà stampato nella posizione data nel campo del formato. Se si specificano virgole e/o un segno più o meno in un campo di formato

contenente un simbolo dollaro, il programma stampa una virgola o un segno prima del simbolo dollaro.
Il simbolo delle quattro frecce verso l'alto serve a specificare che il numero deve essere stampato in formato E +. Bisogna aggiungere un # alle ↑↑↑↑ per specificare la larghezza del campo. Le ↑↑↑↑ possono apparire sia prima che dopo il # nel campo del formato.
Quando si vuole stampare un numero in formato E - (notazione scientifica), bisogna specificare quattro frecce (↑↑↑↑). Se se ne specificano più di una, ma meno di quattro, si ottiene un messaggio di errore "syntax errore" (errore di sintassi). Se si specificano più di quattro frecce, vengono utilizzate solo le prime quattro. La quinta viene interpretata come simbolo di non-testo.
Il segno uguale (=) è utilizzato per centrare una stringa in un campo. La larghezza del campo si specifica con il numero di caratteri (# e =) nel campo del formato. Se la stringa contiene un numero di caratteri inferiore alla larghezza del campo, la stringa viene centrata nel campo. Se la stringa presenta più caratteri, allora può essere contenuta nel campo. I caratteri all'estrema destra vengono troncati e la stringa riempie l'intero campo.
Il segno maggiore di (>) viene utilizzato per giustificare a destra una stringa in un campo.
La larghezza del campo si specifica con il numero di caratteri (# e =) del campo del formato.
Se la stringa contiene un numero di caratteri inferiore alla larghezza del campo, la stringa viene giustificata a destra nel campo. Se la stringa presenta più caratteri, di quanti ne possa contenere il campo, i caratteri all'estrema destra vengono troncati, e la stringa riempie l'intero campo.

PUDEF PUDEF "*da 1 a 4 caratteri*"

PUDEF permette di ridefinire fino a 4 simboli nell'istruzione PRINT USING. È possibile cambiare spazi vuoti, virgole, virgole decimali e simboli dollaro in altri caratteri, sostituendo il nuovo carattere nella posizione corretta della stringa di controllo PUDEF.
La posizione 1 è il carattere di riempimento.
Il default è uno spazio vuoto. Inserire qui un nuovo carattere quando si vuole che un altro carattere appaia al posto degli spazi vuoti.

La posizione 2 è il carattere virgola.
Il default è una virgola.

La posizione 3 è la virgola decimale.

La posizione 4 è il simbolo dollaro.

ESEMPI:

10 PUDEF "*"	Stampa * al posto degli spazi vuoti.
20 PUDEF " @"	Stampa @ al posto delle virgole.
30 PUDEF " .,"	Stampa punti decimali al posto delle virgole e le virgole al posto dei punti decimali.
40 PUDEF " ,.£"	Stampa il simbolo della sterlina inglese al posto del $, i punti decimali al posto delle virgole e viceversa. Gli altri segni sono i valori di default.

READ READ *variabili*

Quest'istruzione viene usata per portare informazioni dalle istruzioni DATA alle variabili, dove i dati possono essere utilizzati.
Le variabili dell'istruzione READ possono contenere sia stringhe, sia numeri. Fare attenzione a non leggere stringhe laddove l'istruzione READ si aspetta un numero, poiché ciò produrrebbe un messaggio di ERRORE.

ESEMPIO:

READ A$, G$, 45

REM REM *messaggio*

REM è semplicemente un'annotazione per chiunque stia leggendo un listato del programma.
Può dare spiegazioni su una parte del programma, informazioni sull'autore, ecc.
L'istruzione REM non influisce minimamente nelle operazioni del programma, se non nell'allungarlo (e quindi rallentarlo). La parola REM può essere seguita da qualsiasi testo, nonostante l'uso di caratteri grafici possa dare strani risultati.

ESEMPIO:

```
10 NEXT X: REM QUESTA RIGA NON È NECESSARIA.
```

RESTORE RESTORE [*riga #*]

Quando RESTORE viene eseguito in un programma, il puntatore sull'elemento in un'istruzione DATA che si dovrà leggere successivamente, viene reimpostato sul primo elemento del listato.
Ciò permette di rileggere l'informazione.
Se [riga #] segue l'istruzione RESTORE, il puntatore si imposta su quella riga.
Altrimenti il puntatore viene reimpostato sulla prima istruzione DATA del programma.

ESEMPIO:

```
RESTORE 200
```

RESUME RESUME [*riga #* /NEXT]

Viene usato per ritornare all'esecuzione dopo aver individuato un errore. Senza argomenti, RESUME prova a rieseguire la riga in cui è contenuto l'errore.
RESUME NEXT riprende l'esecuzione dall'istruzione successiva a quella in cui era contenuto l'errore; RESUME riga # tornerà alla riga specificata e riprenderà l'esecuzione da quel punto.

RETURN RETURN

Questa istruzione viene sempre utilizzata con l'istruzione GOSUB. Quando il programma incontra un'istruzione RETURN, esso va all'istruzione immediatamente successiva all'ultimo comando GOSUB eseguito.
Se GOSUB non era già stato emesso, allora appare un messaggio RETURN WITHOUT GOSUB ERROR (ERRORE - RETURN SENZA GOSUB) e l'esecuzione del programma si interrompe.

SCALE SCALE< *1/0*>

Nelle modalità multicolor e ad alta risoluzione, la variazione della scala degli assi della bit map si può cambiare con il comando SCALE. Introducendo:

SCALE 1

si attiva la modalità di variazione.
Le coordinate possono quindi essere scalate da 0 a 1023, sia X che Y, invece dei normali valori di scala che sono:

modalità multicolor.....	X = da 0 a 159	Y = da 0 a 199
modalità ad alta risoluzione.....	da 0 a 319	da 0 a 199

L'introduzione di "SCALE 0" disattiva la modalità di variazione.

SCNCLR SCNCLR

Cancella lo schermo corrente, i grafici, i testi oppure entrambi (schermo diviso).

SOUND SOUND *voce #, controllo della frequenza, durata*

Quest'istruzione produce un SUONO utilizzando una delle tre voci con un controllo della frequenza incluso nella gamma tra 0 - 1023 per una durata da 0 a 65535 sessantesimi di secondo.

V	Voce
1	Voce 1 (nota)
2	Voce 2 (nota)
3	Voce 2 (rumore bianco)

Se è richiesta un'istruzione SOUND per produrre la voce N, e la precedente istruzione SOUND per la stessa N sta ancora suonando, il BASIC attende che la precedente istruzione SOUND si completi. SOUND con durata 0 è un caso speciale. Ciò fa in modo che il BASIC disattivi immediatamente l'istruzione SOUND corrente per quella voce, senza tenere conto del tempo restante dell'istruzione SOUND precedente. Vedere la TABELLA delle NOTE MUSICALI (SEZIONE 11) per i valori di controllo della frequenza che corrispondono alle note reali.

ESEMPIO:

SOUND 2, 800, 360 ——— Produce una nota usando la voce 2 con la frequenza impostata a 800 per un minuto

SSHAPE/GSHAPE Le istruzioni SSHAPE e GSHAPE sono utilizzate per salvare e richiamare aree rettangolari di schermi multicolor o ad alta risoluzione, usando variabili di stringa BASIC.

Il comando per salvare un'area è il seguente:

SSHAPE *variabile di stringa, a1,b1* [*,a2,b2*]

variabile di stringa	Nome di stringa in cui salvare dati
a 1, b 1	coordinata dell'angolo (scalata)
a 2, b 2	Cooordinata dell'angolo opposto (a 1, b 1) (il default è il CP)

Dato che il BASIC limita la stringa ad una lunghezza fino a 255 caratteri, la misura dell'area che si vuole salvare è limitata.
La misura della stringa richiesta può essere calcolata utilizzando una delle seguenti formule (non scalate):

L(mcm) = INT ((ABS(a1-a2) + 1) / 4 + .99)* (ABS(b1-b2) + 1) + 4
L(h-r) = INT ((ABS(a1-a2) + 1) / 8 + .99)* (ABS(b1-b2) + 1) + 4
(mcm) si riferisce alla modalità multicolor; (h-r) è l'alta risoluzione.

La forma viene salvata riga per riga.
Gli ultimi quattro byte della stringa contengono le lunghezze della colonna e della riga meno uno (cioè: ABS(a1-a2) in un formato di byte basso/alto (se scalato divide le lunghezze di (X) per 3.2 e di (Y) per 5.12).

Il seguente comando visualizza una forma salvata su qualsiasi area dello schermo:

GSHAPE *stringa* [,[*a,b*][,*modalità*]]

stringa.....	Contiene la forma che si vuole tracciare
a,b.....	Coordinata superiore sinistra che indica dove tracciare la forma (scalata- il default è il CP)
modalità.....	Modalità di sostituzione 0: posiziona la forma così com'è (default) 1: posiziona la forma invertita del campo 2: esegue un OR logico tra la forma e l'area 3: esegue un AND logico tra la forma e l'area 4: esegue un XOR logico tra la forma e l'area

ESEMPI:

SSHAPE "SHIP",0,0	Salva la forma sull'area dello schermo dall'angolo superiore sinistro al cursore al di sotto del nome "SHIP"
GSHAPE "SHIP",,,1	Visualizza una forma "SHIP" invertita con l'angolo superiore sinistro nella posizione del cursore

STOP STOP

Questa istruzione interrompe il programma.
Verrà visualizzato il messaggio BREAK IN LINE #, (INTERRUZIONE ALLA RIGA #) dove # è il numero di riga che contiene STOP. Il programma si può riprendere dall'istruzione successiva a STOP usando il comando CONT.
Di solito l'istruzione STOP si usa durante la ricerca degli errori del programma.

SYS SYS *indirizzo*

La parola SYS è seguita da un numero decimale o da una variabile numerica nella gamma da 0 a 65535.
Il programma inizia eseguendo il programma in linguaggio macchina partendo da quella locazione di memoria. Ciò è simile alla funzione USR, ma non passa un parametro.
Vedere la Guida di Consultazione per il Programmatore del Plus/4 per ulteriori informazioni sui programmi in linguaggio macchina.

TRAP TRAP [*riga #*]

Quando è in funzione, TRAP intercetta tutte le condizioni di errore (incluso il TASTO STOP), eccetto "UNDEF'D STATEMENT ERROR" (ERRORE - ISTRUZIONE INDEFINITA). Nel caso di qualsiasi errore di

esecuzione, appare il flag di errore e l'esecuzione viene trasferita al numero di riga definito nell'istruzione TRAP.
Si può individuare il numero di riga in cui è contenuto l'errore utilizzando la variabile di sistema EL. La specifica condizione di errore è contenuta nella variabile di sistema ER. La funzione di stringa ERR$(ER) dà il messaggio di errore che corrisponde a qualsiasi condizione di errore ER.

NOTA: Un errore contenuto in una routine TRAP non può essere individuato.
L'istruzione RESUME può essere utilizzata per riprendere l'esecuzione.
L'istruzione TRAP senza l'argomento numero di riga impedisce l'individuazione degli errori.

TRON TRON

L'istruzione TRON viene utilizzata nella ricerca degli errori del programma. Questa istruzione attiva la modalità di traccia. Quando si è in modalità di traccia, mano a mano che le istruzioni vengono eseguite, viene stampato il numero di riga di quella istruzione.

TROFF TROFF

Questa istruzione disattiva la modalità di traccia.

VOL VOL *livello del volume*

Imposta il livello del volume corrente per i comandi SOUND. Il volume si può impostare da 0 a 8, dove 8 è il volume massimo e 0 è il volume minimo. VOL agisce su entrambe le voci.

WAIT WAIT *indirizzo, valore 1* [,*valore 2*]

L'istruzione WAIT viene utilizzata per interrompere il programma fino a

che il contenuto di una locazione in memoria non cambia in modo specifico.
L'indirizzo deve essere incluso nella gamma da 0 a 65535. Il valore 1 e il valore 2 devono essere inclusi nella gamma da 0 a 255.
Per prima cosa viene eseguito un OR esclusivo tra il contenuto della locazione di memoria e il valore 2 (se presente) e quindi viene eseguito un AND con il valore 1.
Se il risultato è zero, il programma controlla di nuovo la locazione della memoria.
Quando il risultato non è zero, il programma prosegue con l'istruzione successiva.

ULTERIORI INFORMAZIONI SULLE ISTRUZIONI GRAFICHE

Vi sono ancora alcuni concetti che si riferiscono a tutte le istruzioni grafiche della bit-map.
Innanzitutto il concetto del cursore Pixel (CP). Il CP è molto simile al cursore della modalità testo; è la posizione da cui verrà tracciato il punto successivo. A differenza del cursore di testo, il CP è invisibile.
Tutti i comando del disegno utilizzano il CP. Inoltre, il comando locate permette di riposizionare il CP senza disegnare nulla.
Ovunque si vogliano utilizzare le coordinate X,Y in un comando del disegno, esse possono essere sostituite dalle coordinate RELATIVE.
Le coordinate relative sono basate sul valore corrente del CP. Per utilizzare le coordinate relative è sufficiente aggiungere un + o un - davanti alle coordinate.
Un segno più davanti al valore di X sposta il CP sulla destra. Un segno meno, lo sposta sulla sinistra.
Analogamente, un segno meno davanti alla coordinata Y sposta il CP verso l'alto, mentre un segno più lo sposta verso il basso.

Ad esempio:

```
LOCATE +100,-25 ——— Sposta il CP di 100 pixel verso destra e di 25 verso l'alto

DRAW1,+10,+10TO100,100 ——— traccia una riga 10 pixel a destra e 10 pixel sotto il valore corrente del CP fino al punto assoluto 100,100.
```

Si può anche specificare una distanza e l'angolo relativo al corrente CP separando i due parametri con un punto e virgola.

Ad esempio:

LOCATE 50;45 ——— sposta il CP dalla propria locazione corrente per una distanza di 50 punti ad un angolo di 45 gradi.

FUNZIONI Funzioni Numeriche

Le funzioni numeriche vengono classificate come tali in quanto ritornano numeri. La loro gamma va dal calcolo di funzioni matematiche alla specificazione di locazioni sullo schermo. Le funzioni numeriche hanno i seguenti formati:

FUNZIONE (argomento)
dove l'argomento può essere un valore numerico, una variabile o una stringa.

ABS(X) (valore assoluto)
La funzione di valore assoluto ritorna il valore positivo dell'argomento X.

ASC(X$)
Questa funzione ritorna il codice ASCII (numero) del primo carattere di X$.

ATN(X) (arcotangente)
Ritorna l'angolo la cui tangente è X, misurata in radianti.

COS(X) (coseno)
Ritorna il valore del coseno di X, dove X è l'angolo misurato in radianti.

DEC (stringa esadecimale)
Ritorna il valore decimale della stringa esadecimale (0 < stringa esadecimale < FFFF)

ESEMPIO:

N=DEC("F4")

EXP(X)
Ritorna il valore della costante matematica e (2.71828183) elevata alla potenza di X.

FNxx(x)
Ritorna il valore della funzione xx definita dall'utente, creata in un'istruzione DEF FNxx.

INSTR (*stringa 1, stringa 2 [posizione di partenza]*)
Ritorna la posizione della stringa 2 nella stringa 1 alla [posizione di partenza] o dopo questa posizione. Il default per la posizione di partenza è all'inizio della stringa 2. Se non viene trovata alcuna corrispondenza, viene ritornato il valore 0.

ESEMPIO:

PRINT INSTR ("IL GATTO NEL CAPPELLO", "GATTO")
il risultato è 4, perché la parola GATTO inizia al quarto carattere della stringa 1.

INT (X) (intero)
Ritorna la parte intera di X, togliendo tutte le posizioni decimali a destra della virgola. Il risultato è sempre minore di o uguale a X. In questo modo tutti i numeri negativi con posizioni decimali si trasformano in numeri interi con valore inferiore al valore effettivo (es. INT(-4.5)=-5). Se la funzione INT viene utilizzata per arrotondare per eccesso o per difetto, il formato è INT(X +/- .5).

ESEMPIO:

X=INT(X*100 + .5)/100

arrotonda per eccesso.

JOY (n)

Quando n = 1..... la posizione del joystick è # 1
n = 2..... la posizione del joystick è # 2

I valori maggiori o uguali a 128 indicano che il pulsante fuoco è stato premuto. La direzione viene indicata come segue:

```
                    SU
fuoco = 128 +        1
              8              2
SINISTRA  7          0              3 DESTRA
              6              4
                     5
                    GIÙ
```

ESEMPIO:

```
JOY(2)=135
```
il joystick # 2 spara a sinistra

LOG (X) (logaritmo)
Questa funzione ritorna il logaritmo naturale di X, che è il logaritmo in base e (vedi EXP(X)). Per passare al logaritmo in base 10, dividere per LOG (10).

PEEK (X)
Questa funzione dà il contenuto della posizione di memoria X, dove X si trova in una gamma tra 0 e 65535 ritornando un risultato da 0 a 255. Questa funzione viene spesso usata insieme all'istruzione POKE.

RCLR (N)
Ritorna il colore corrente assegnato alla sorgente N (0 < N < 4) (0= sfondo; 1= primo piano; 2= multicolor 1; 3= multicolor 2; 4= cornice).

RDOT (N)
Ritorna le informazioni sulla posizione corrente del cursore pixel (CP) a XPOS/YPOS.
N - 0 per XPOS
1 per YPOS
2 sorgente di colore

RGR (X)
Ritorna la modalità grafica corrente (X è un argomento fittizio).

RLUM (N)
Ritorna il livello di luminanza corrente assegnato alla sorgente N.

RND (X) (numero casuale)
Questa funzione ritorna un numero casuale, 0 o 1. Questa funzione è utile nei giochi, per simulare il gioco dei dadi e altri giochi d'azzardo, e viene inoltre utilizzata in applicazioni statistiche. Il primo numero casuale deve essere generato dalla formula RND(-TI), per iniziare ogni volta in un modo diverso. Dopo di ciò, il numero assegnato a X deve essere 1 o un qualsiasi altro numero positivo (X rappresenta il seme o l'argomento del numero casuale). Se X è zero, a RND viene assegnato nuovamente un seme dal clock hardware ogni volta che viene usato RND. Un valore negativo per X assegna un seme al generatore di numero casuale usando X e dà una sequenza di numero casuale. L'uso dello stesso numero negativo per X come seme risulta nella stessa sequenza di numeri casuali. Un valore positivo dà numeri casuali basati sul seme assegnato precedentemente.
Per simulare un dado che rotola, utilizzare la formula INT (RND(1)*6+1). Per prima cosa il numero casuale 0-1 viene moltiplicato per 6, dando così una gamma di valori da 0 a 6 (in effetti, maggiore di 0 e minore di 6). Viene quindi aggiunto 1, ottenendo una gamma da 1 a 7 escluso. La funzione INT toglie tutte le posizioni decimali, trasformando il risultato in una cifra da 1 a 6.
Per simulare 2 dadi, sommare 2 dei numeri ottenuti con la formula precedente.

ESEMPIO:

```
100 X=INT(RND(1)*6)+INT(RND(1)*6)+2     Simula 2 dadi
100 X=INT(RND(1)*1000)+1                Numero da 1 a 1000
100 X=INT(RND(1)*150)+100               Numero da 100 a 249
```

SGN(X) (segno)
Questa funzione ritorna il segno, positivo, negativo o zero, di X.
Il risultato è +1 se positivo, 0 se zero, e -1 se negativo.

SIN(X) (seno)
Questa è la funzione trigonometrica del seno. Il risultato è il seno di X, dove X è un angolo in radianti.

SQR(X) (radice quadrata)
Questa funzione ritorna la radice quadrata di X, dove X è un numero positivo o 0. Se X è negativo, verrà visualizzato il messaggio di errore ILLEGAL QUANTITY ERROR (ERRORE - QUANTITÀ NON VALIDA)

TAN (X) (tangente)
Questa funzione dà la tangente di X, dove X è un angolo in radianti.

USR (X)
Quando viene utilizzata questa funzione, il programma salta a un programma in linguaggio macchina il cui punto di partenza è contenuto nelle posizioni di memoria 1281 e 1282.
Il parametro X viene passato al programma in linguaggio macchina nell'accumulatore di virgola mobile. Un altro numero viene ritornato al programma BASIC per mezzo della variabile di chiamata. In altre parole, questa funzione permette di scambiare una variabile tra codice macchina e BASIC.
Per maggiori dettagli su quanto sopra e sulla programmazione in linguaggio macchina, riferirsi al MANUALE DI CONSULTAZIONE PER IL PROGRAMMATORE del Plus/4.

VAL(X$).
Questa funzione converte la stringa X$ in un numero, ed è essenzialmente l'operazione inversa di STR$. Tutti i caratteri in formato di numero riconoscibile nella stringa vengono esaminati dal carattere all'estrema sinistra verso destra. Se il Plus/4 trova caratteri illegali, viene convertita solo la parte di stringa fino a quel punto.

ESEMPIO:

10 X=VAL("123.456")	X=123.456
10 X=VAL("3E03")	X=3000
10 X=VAL("12A13B")	X=12
10 X=VAL("RIUO17*")	X=0
10 X=VAL("-1.23.23.23")	X=-1.23

Funzioni di Stringa Le funzioni di stringa si differenziano dalle funzioni numeriche in quanto possono ritornare caratteri, grafici o numeri da una stringa (definiti da virgolette) invece che ritornare un numero.

CHR$(X)
Questa funzione ritorna un carattere di stringa il cui codice ASCII è X.

ERR$(N)
Ritorna la stringa che descrive una condizione di errore N (vedi TRAP).

HEX$(N)
Ritorna una stringa di 4 caratteri che contiene la rappresentazione esadecimale del valore N (0 <N< 65535).

LEFT$(X$,X)
Questa funzione ritorna una stringa contenente i caratteri all'estrema sinistra di X$.

LEN(X$)
Ritorna il numero di caratteri (compresi spazi ed altri simboli) nella stringa X$.

MID$ (X$,S,X)
Questa istruzione ritorna una stringa contenente X caratteri, iniziando dal carattere S in X$. MID$ può anche essere utilizzato alla sinistra dell'istruzione di assegnazione, sia come pseudo-variabile, sia come funzione. MID$ (variabile di stringa, posizione iniziale, lunghezza) = stringa sorgente.
Questa funzione riassegna valori di posizioni (da posizione iniziale) a (posizione iniziale + lunghezza) della stringa sorgente ai caratteri della variabile di stringa nelle locazioni corrispondenti. Il default della lunghezza è la lunghezza delle variabili di stringa e se (posizione di partenza + lunghezza) è maggiore della lunghezza della stringa sorgente, ne risulterà un errore:

ESEMPIO:

```
10 A$="IL CANE NEL CAPPELLO":
20 PRINT A$
30 MID$ (A$,4,4)="PANE"
40 PRINT A$
```

RIGHT$ (X$,X)
Ciò ritorna gli X caratteri all'estrema destra di X$.

STR$ (X)
Ciò ritorna una stringa che è identica alla versione stampata di X$.

ESEMPIO:

```
A$=STR$ (X)
```

Altre Funzioni

FRE(X)
Questa funzione ritorna il numero dei byte non utilizzati disponibili in memoria.
X è un argomento fittizio.

POS(X)
Questa funzione ritorna il numero della colonna 0-79) dove inizia la successiva istruzione PRINT sullo schermo. X è un argomento fittizio.

SPC(X)
Questa funzione viene utilizzata nell'istruzione PRINT per saltare oltre X spazi. X può avere un valore da 0 a 255.

TAB(X)
Questa funzione viene utilizzata nell'istruzione PRINT. Il successivo elemento da stampare è nella colonna numero X. X può avere un valore da 0 a 255.

π (PI greco)
Il simbolo π, quando viene usato in un'equazione, ha il valore di 3.14.

VARIABILI E OPERATORI

Il Plus/4 utilizza 3 tipi di variabili nel BASIC: le numeriche normali, le numeriche intere, e le variabili (alfanumeriche) di stringa.

Variabili

Le VARIABILI NUMERICHE normali, anche dette variabili in virgola mobile, possono avere qualsiasi valore da 10^{-10} a 10^{+10}, con un massimo di nove cifre di visualizzazione. Quando un numero diventa maggiore di nove cifre non in 10^{-10} o 10^{+10}, il computer lo visualizza sottoforma di notazione scientifica, con il numero normalizzato ad 1 cifra e otto decimali, seguiti dalla lettera E e la potenza di dieci per cui è stato moltiplicato il numero.
Ad esempio, il numero 12345678901 verrà visualizzato 1.23456789E+10

Le VARIABILI INTERE possono essere usate quando il numero và da +32767 a -32768 e senza parte frazionale. Una variabile intera è un numero come 5,10, oppure -100.
Gli interi occupano meno spazio delle variabili a virgola mobile, specialmente quando usati in una matrice.

Le VARIABILI di STRINGA sono quelle utilizzate per i dati del carattere e possono contenere numeri, lettere, e qualsiasi altro carattere il Plus/4 possa riprodurre.
"Plus/4" è un esempio di variabile di stringa.

Nomi di Variabile

I nomi di variabile possono consistere in una singola lettera, una lettera seguita da un numero, due lettere. I nomi di variabile possono essere più lunghi di 2 caratteri, ma sono significativi solo i primi due.
Una variabile intera viene specificata con l'uso del segno percento (%) situato dopo il nome di variabile.
Le variabili di stringa hanno i nomi seguiti dal simbolo del dollaro ($).

ESEMPI:

Nomi di variabile numerica: A, A5, BZ
Nomi di variabile intera: A%,A5%,BZ%
Nomi di variabile di stringa: A$,A5$,BZ$

Le MATRICI sono liste di variabili con lo stesso nome, che usano un numero (o numeri) extra, per specificare un elemento della matrice. Le matrici vengono definite con l'uso dell'istruzione DIM e possono essere a virgola mobile, intere, oppure matrici e variabili di stringa. Il nome di variabile della matrice è seguito da parentesi () che includono nella lista il numero della variabile.

ESEMPI:

A(7),BZ%(11),A$(87)

Le matrici possono avere più di una dimensione. Una matrice a due dimensioni può essere visualizzata con righe e colonne, laddove il primo numero identifica la riga e il secondo, tra parentesi, identifica la colonna (come per specificare un riquadro su un reticolo).

ESEMPI:

A(7,2), BZ%(2,3,4), Z$(3,2)

Nomi di Variabile Riservati

Vi sono sette nomi di variabile riservati per l'utilizzo del Plus/4 e non possono essere utilizzati per altri scopi. Essi sono le variabili DS, DS$, ER, EL, ST, TI, e TI$. Nemmeno è possibile usare PAROLE CHIAVE come TO e IF oppure nomi che contengono PAROLE CHIAVE come SRUN, RNEW, oppure XLOAD, come nomi di variabile.

ST è una variabile di stato per input e output (tranne operazioni normali di schermo e di tastiera). Il valore di ST dipende dal risultato dell'ultima operazione input/output.
Una spiegazione maggiormente dettagliata di ST si trova nel Manuale di Consultazione per il Programmatore, ma, in generale, se il valore di ST è 0, l'operazione è riuscita.

TI e TI$ sono variabili che si riferiscono all'orologio in tempo reale incorporato nel Plus/4.
L'orologio del sistema è aggiornato ogni sessantesimo di secondo. Inizia da 0 quando il Plus/4 viene acceso e viene reimpostato solo cambiando il valore di TI$.
La variabile TI fornisce il valore corrente dell'orologio in 1/60 di secondo.

TI$ è una stringa che legge il valore dell'orologio in tempo reale come un normale orologio a 24 ore. I primi due caratteri di TI$ contengono l'ora, il terzo e il quarto carattere sono i minuti e il quinto e il sesto carattere sono i secondi. Questa variabile può essere impostata su qualsiasi valore (numerico), e sarà automaticamente aggiornata come un orologio a 24 ore.

ESEMPIO:

TI$ = "101530" imposta l'orologio su 10:15 e 30 secondi

Il valore dell'orologio si perde quando il Plus/4 viene spento. Inizia da zero quando il Plus/4 viene acceso, e si riazzera quando il valore dell'orologio supera 235959 (23 ore, 59 minuti e 59 secondi).
La variabile DS legge il canale del comando del drive e ritorna lo stato corrente del drive. Per avere in caratteri alfabetici questa informazione battere PRINT DS$.
Queste variabili di stato sono utilizzate dopo una operazione su disco, come con DLOAD o DSAVE, per scoprire perché la spia rossa d'errore sul drive sta lampeggiando.

ER, EL e ERR$ sono variabili utilizzate nelle routine di ricerca dell'errore. Solitamente esse sono utili solo all'interno di un programma. ER ritorna l'ultimo errore incontrato dal momento in cui è stato lanciato il programma.
EL è la riga in cui è contenuto l'errore.
ERR$ è una funzione che permette al programma di stampare uno dei messaggi di errore BASIC.
PRINT ERR$ (ER) permette di visualizzare l'esatto messaggio di errore.

OPERATORI BASIC Gli operatori ARITMETICI includono i simboli seguenti:

+	addizione
−	sottrazione
*	moltiplicazione
/	divisione
↑	elevamento a potenza

In una riga che contiene più di un operatore, le operazioni verranno sempre eseguite in un ordine prestabilito.
Se vengono utilizzati più operatori insieme, il computer assegna le seguenti priorità: per primo l'elevamento a potenza, quindi la moltiplicazione e la divisione e infine l'addizione e la sottrazione. Se due operazioni hanno la stessa priorità, verranno eseguite in ordine, da sinistra a destra. Se si desidera però un ordine differente, il BASIC del Plus/4 permette di conferire una priorità più alta ad una determinata operazione includendola tra parentesi. Le operazioni tra parentesi

saranno calcolate prima di qualsiasi altra operazione. Assicurarsi che le equazioni posseggano lo stesso numero di parentesi aperte e chiuse. In caso contrario si avrà un messaggio di errore SYNTAX ERROR (ERRORE DI SINTASSI) al momento dell'esecuzione del programma.
Esistono anche operatori per eguaglianze e diseguaglianze, chiamati operatori RELAZIONALI. Gli operatori aritmetici hanno sempre la priorità sugli operatori relazionali.

=	è: uguale a
<	è: minore di
>	è: maggiore di
<= oppure =<	è: minore di o uguale a
>= oppure =>	è: maggiore di o uguale a
<> oppure ><	è: diverso da

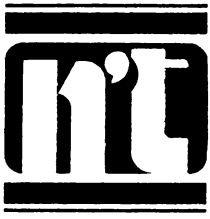
Infine, vi sono tre operatori LOGICI, con priorità inferiore agli operatori aritmetici e relazionali.

AND
OR
NOT

Questi operatori vengono utilizzati principalmente per collegare formule multiple all'interno di istruzioni IF...THEN. Se utilizzati con operatori aritmetici, vengono presi in considerazione per ultimi (cioè dopo + e –).

ESEMPI:

IF A=B AB AND C=D THEN 100	entrambe le condizioni A=B e C=D devono essere vere.
IF A=B OR C=D THEN 100	la condizione A=B o la condizione C=D deve essere vera.
A=5:B=4:PRINT A=B	visualizza un valore 0
A=5:B=4:PRINT A>B	visualizza un valore -1
PRINT 123 AND 15:PRINT 5 OR 7	visualizza 11 e 7



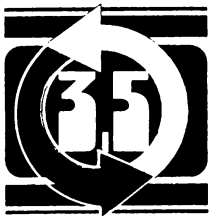
Abbreviazioni Basic 3.5

PAROLA CHIAVE		ABBREVIAZIONE		TIPO
ABS	a	SHIFT	B	funzione-numerica
ASC	a	SHIFT	S	funzione-numerica
ATN	a	SHIFT	T	funzione-numerica
AUTO	a	SHIFT	U	comando
BACKUP	b	SHIFT	A	comando
BOX	b	SHIFT	O	istruzione
CHAR	ch	SHIFT	A	istruzione
CHR$	c	SHIFT	H	funzione-stringa
CIRCLE	c	SHIFT	I	istruzione
CLOSE	cl	SHIFT	O	istruzione
CLR	c	SHIFT	L	istruzione
CMD	c	SHIFT	M	istruzione
COLLECT	col	SHIFT	L	comando
COLOR	co	SHIFT	L	istruzione
CONT	c	SHIFT	O	comando
COPY	co	SHIFT	P	comando
COS		nessuna		funzione-numerica
DATA	d	SHIFT	A	istruzione
DEC		nessuna		funzione-numerica
DEF FN	d	SHIFT	E	istruzione
DELETE	de	SHIFT	L	comando
DIM	d	SHIFT	I	istruzione
DIRECTORY	di	SHIFT	R	comando
DLOAD	d	SHIFT	L	comando
DO		nessuna		istruzione
DRAW	d	SHIFT	R	istruzione
DSAVE	d	SHIFT	S	comando
END	e	SHIFT	N	istruzione
ERR$	e	SHIFT	R	funzione-stringa
EXP	e	SHIFT	X	funzione-numerica
FOR	f	SHIFT	O	istruzione
FRE	f	SHIFT	R	funzione-numerica
GET	g	SHIFT	E	istruzione
GETKEY	getk	SHIFT	E	istruzione
GET #		nessuna		istruzione
GOSUB	go	SHIFT	S	istruzione
GOTO	g	SHIFT	O	istruzione
GRAPHIC	g	SHIFT	R	istruzione
GSHAPE	g	SHIFT	S	istruzione
HEADER	he	SHIFT	A	comando
HEX$	h	SHIFT	E	funzione-stringa
IF...GOTO		nessuna		istruzione
IF...THEN...ELSE		nessuna		istruzione
INPUT		nessuna		istruzione
INPUT #	i	SHIFT	N	istruzione

PAROLA CHIAVE	ABBREVIAZIONE		TIPO
INSTR	in	SHIFT S	funzione-numerica
INT	nessuna		funzione-numerica
JOY	j	SHIFT O	funzione-numerica
KEY	k	SHIFT E	comando
LEFT$	le	SHIFT F	funzione-stringa
LEN	nessuna		funzione-numerica
LET	l	SHIFT E	istruzione
LIST	l	SHIFT I	comando
LOAD	l	SHIFT O	comando
LOCATE	lo	SHIFT C	istruzione
LOG	nessuna		funzione-nnumerica
LOOP	lo	SHIFT O	istruzione
MID$	m	SHIFT I	funzione-stringa
MONITOR	m	SHIFT O	istruzione
NEW	nessuna		comando
NEXT	n	SHIFT E	istruzione
ON...GOSUB	on...go	SHIFT S	istruzione
ON...GOTO	on...g	SHIFT O	istruzione
OPEN	o	SHIFT P	istruzione
PAINT	p	SHIFT A	istruzione
PEEK	p	SHIFT E	funzione-numerica
POKE	p	SHIFT O	istruzione
POS	nessuna		funzione-numerica
PRINT	?		istruzione
PRINT #	p	SHIFT R	istruzione
PRINT USING	?us	SHIFT I	istruzione
PUDEF	p	SHIFT U	istruzione
RCLR	r	SHIFT C	funzione-numerica
RDOT	r	SHIFT D	funzione-numerica
READ	r	SHIFT E	istruzione
REM	nessuna		istruzione
RENAME	re	SHIFT N	comando
RENUMBER	ren	SHIFT U	comando
RESTORE	re	SHIFT S	istruzione
RESUME	res	SHIFT U	istruzione
RETURN	re	SHIFT T	istruzione
RGR	r	SHIFT G	funzione-numerica
RIGHT$	r	SHIFT I	funzione-stringa
RLUM	r	SHIFT L	funzione-numerica
RND	r	SHIFT N	funzione-numerica
RUN	r	SHIFT U	comando
SAVE	s	SHIFT A	comando
SCALE	sc	SHIFT A	istruzione
SCNCLR	s	SHIFT C	istruzione
SCRATCH	sc	SHIFT R	comando

PAROLA CHIAVE	ABBREVIAZIONE			TIPO
SGN	s	SHIFT	G	funzione-numerica
SIN	s	SHIFT	I	funzione-numerica
SOUND	s	SHIFT	O	istruzione
SPC	s	SHIFT	D	funzione-speciale
SQR	s	SHIFT	Q	funzione-numerica
SSHAPE	s	SHIFT	S	istruzione
STatus		nessuna		riservato-variabile-numerica
STOP	s	SHIFT	T	istruzione
STR$	st	SHIFT	R	funzione-stringa
SYS	s	SHIFT	Y	istruzione
TAB	t	SHIFT	A	funzione-speciale
TAN		nessuna		funzione-numerica
TI		nessuna		riservato-variabile-numerica
TI$		nessuna		riservato-variabile di stringa
TRAP	t	SHIFT	R	istruzione
TROFF	tro	SHIFT	F	istruzione
TRON	tr	SHIFT	O	istruzione
UNTIL	u	SHIFT	N	istruzione
USR	u	SHIFT	S	funzione-speciale
VAL		nessuna		funzione-numerica
VERIFY	v	SHIFT	E	comando
VOL	v	SHIFT	O	istruzione
WAIT	w	SHIFT	A	istruzione
WHILE	w	SHIFT	H	istruzione

SEZIONE 3



Programmi di Conversione

Conversione di Programmi Basic Standard in Programmi Basic 3.5 Commodore

Se si devono trattare programmi scritti in un BASIC diverso da quello Commodore, potranno essere necessarie alcune rettifiche prima di utilizzarli sul Plus/4. Ecco alcuni accorgimenti per rendere più semplice la conversione.

Dimensioni di Stringa

Cancellare tutte le istruzioni utilizzate per definire la lunghezza delle stringhe. Una istruzione tipo DIM. A$(I,J), che dimensiona una matrice di stringa per J elementi di lunghezza I, dovrebbe essere convertita nella istruzione BASIC Commodore DIM A$(J).
Alcuni BASIC utilizzano una virgola o una «e» commerciale per il concatenamento delle stringhe (collegamento). Queste virgole e queste «e» commerciali dovranno essere trasformate in un segno + che rappresenta l'operatore BASIC 3.5 Commodore per il concatenamento delle stringhe.
Nel BASIC Commodore, le funzioni MID$, RIGHT$ e LEFT$ vengono utilizzate per prelevare sottostringhe dalle stringhe. Per accedere alla posizione I di A$ in un formato A$(I) o per prelevare una sottostringa di A$ dalla posizione I alla posizione J nel formato A$(I,J) è necessario trasformare i formati in questo modo:

Altri BASIC	BASIC 3.5 Commodore
A$(I) = X$	MID$(A$,I,J) = X$
A$(I,J) = X$	MID$(A$,I,J) = X$

Assegnazioni Multiple

Per impostare B e C = 0, alcuni BASIC permettono questo tipo di formato: 10 LET B = C = 0
Il BASIC Commodore interpreterebbe il secondo segno di uguale come un operatore logico e imposterebbe B = -1 se C = 0.
Bisognerà dunque convertire questa istruzione in : 10 C = 0: B = 0

Istruzioni Multiple
Alcuni BASIC utilizzano una / per separare istruzioni multiple sulla stessa riga. Con il BASIC 3.5, si utilizzeranno i 2 punti (:) per separare tutte le istruzioni.

Funzioni Mat
I programmi che utilizzano le funzioni MAT disponibili in alcuni BASIC dovranno essere riscritti utilizzando loop FOR...NEXT per una esecuzione corretta.

Tasti Funzione Riprogrammabili
Si potranno riprogrammare i tasti funzione per renderli uguali ai tasti funzione del Commodore 64 e del VIC 20. (Questo facilita anche la portabilità di programmi da queste macchine al Plus/4).
Per riprogrammare i tasti, inserire nel programma la seguente riga:

```
10 FOR I=1 TO 8:KEY I, CHR$(I+132): NEXT
```

Da questo momento qualsiasi tasto funzione venga battuto provocherà un carattere non visualizzato da 133 a 140 come nel Commodore 64. Per controllare questo in un programma, si potrà usare il seguente metodo:

```
20 GETKEY A$: IF ASC(A$)=133 THEN PRINT "PRESSIONE DEL
TASTO FUNZIONE 1": GOTO 20
30 IF ASC(A$) > 133 AND ASC(A$) < 141 THEN PRINT
"PRESSIONE DI ALTRI TASTI FUNZIONE"
40 GOTO 20
```

Una volta completato il programma, si dovranno di nuovo ridefinire le funzioni dei tasti se si vorrà che corrispondano nuovamente a directory, dload, ecc. Si potrà farlo manualmente, in un programma, o reinizializzando il Plus/4.

SEZIONE 4

Messaggi d'Errore

Questi messaggi d'errore vengono stampati dal BASIC. Si potrà anche visualizzare i messaggi tramite l'uso della funzione ERR$ (). Per quanto riguarda questa funzione il numero d'errore si riferisce solo al numero assegnato all'errore.

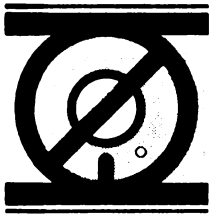
ERRORE #	NOME DELL'ERRORE	
1	**TOO MANY FILES** (TROPPI FILE)	Il limite dei file aperti contemporaneamente è 10.
2	**FILE OPEN** (FILE APERTO)	È stato fatto un tentativo di aprire un file utilizzando il numero di un file già aperto.
3	**FILE NOT OPEN** (FILE NON APERTO)	Il numero di file specificato in una istruzione I/O deve essere aperto prima di utilizzarlo.
4	**FILE NOT FOUND** (FILE NON TROVATO)	O non esiste nessun file con quel nome (disco) o è stato letto un indicatore di fine nastro (nastro).
5	**DEVICE NOT PRESENT** (DISPOSITIVO NON PRESENTE)	Il dispositivo I/O richiesto non è disponibile.
6	**NOT INPUT FILE** (FILE NON DI INPUT)	È stato fatto un tentativo per ottenere o introdurre dati di un file che è stato specificato di solo output.
7	**NOT OUTPUT FILE** (FILE NON DI OUTPUT)	È stato fatto un tentativo di trasmettere dati ad un file che è stato specificato di solo input.
8	**MISSING FILE NAME** (MANCA IL NOME DEL FILE)	Le istruzioni OPEN, LOAD, o SAVE trasmesse all'unità disco richiedono generalmente un nome di file.
9	**ILLEGAL DEVICE NUMBER** (NUMERO DI DISPOSITIVO ILLEGALE)	È stato fatto un tentativo di utilizzare non correttamente un dispositivo (istruzione SAVE applicata allo schermo, ecc.).

10	**NEXT WITHOUT FOR** (NEXT SENZA FOR)	O i loop non sono nidificati correttamente, o c'è un nome di variabile in una istruzione NEXT che non corrisponde a quello contenuto in un'istruzione FOR.
11	**SYNTAX** (SINTASSI)	Un'istruzione non è riconoscibile dal BASIC. Questo può accadere a causa della mancanza di parentesi, di una parola chiave mal formulata, ecc.
12	**RETURN WITHOUT GOSUB** (RETURN SENZA GOSUB)	Un'istruzione RETURN è stata incontrata quando nessuna istruzione GOSUB era attiva.
13	**OUT OF DATA** (DATI ESAURITI)	È stata incontrata un'istruzione READ (leggere) senza che nessun dato sia rimasto non-letto.
14	**ILLEGAL QUANTITY** (QUANTITÀ ILLEGALE)	Un numero usato come argomento di una funzione o di una istruzione è al di fuori della gamma permessa.
15	**OVERFLOW**	Il risultato di un calcolo è maggiore del numero massimo permesso (1.701411833E+38).
16	**OUT OF MEMORY** (MEMORIA ESAURITA)	O non c'è più spazio per il programma e le variabili di programma, o ci sono troppe istruzioni DO, FOR o GOSUB in funzione.
17	**UNDEF'D STATEMENT** (ISTRUZIONE NON DEFINITA)	Il numero di riga voluto non esiste nel programma.

18	**BAD SUBSCRIPT** (INDICE NON VALIDO)	Il programma ha cercato di indirizzare un elemento di una matrice fuori della gamma specificata dalla istruzione DIM.
19	**REDIM'D ARRAY** (MATRICE RIDIMENSIONATA)	Una matrice può essere dimensionata solo una volta. Sè una matrice viene indirizzata prima di essere dimensionata, viene effettuato un dimensionamento automatico (fino a 10).
20	**DIVISION BY ZERO** (DIVISIONE PER ZERO)	La divisione per zero non è ammessa.
21	**ILLEGAL DIRECT** (INDIRIZZO ILLEGALE)	INTRODURRE o OTTENERE istruzioni viene permesso solo all'interno di un programma.
22	**TYPE MISMATCH** (ERRORE DI BATTITURA)	Questo avviene quando un numero viene usato al posto di una stringa o viceversa.
23	**STRING TOO LONG** (STRINGA TROPPO LUNGA)	Una stringa può contenere fino a 255 caratteri.
24	**FILE DATA** (DATI DEL FILE)	Dati non corretti letti da un file su nastro.
25	**FORMULA TOO COMPLEX** (FORMULA TROPPO COMPLESSA)	Semplificarel'espressione (dividere in due parti oppure utilizzare meno parentesi)
26	**CAN'T CONTINUE** (NON È POSSIBILE CONTINUARE)	Il comando CONT non funziona se il programma non è stato lanciato, se c'è un errore o se è stata modificata una riga.
27	**UNDEF'D FUNCTION** (FUNZIONE NON DEFINITA)	Si è fatto riferimento ad una funzione definita dall'utente che non era mai stata definita.

28	**VERIFY** (VERIFICA)	Il programma su nastro o su disco non corrisponde al programma in memoria.
29	**LOAD** (CARICAMENTO)	C'è stato un problema nel caricamento. Riprovare.
30	**BREAK** (INTERRUZIONE)	Si è battuto il tasto STOP per interrompere l'esecuzione del programma.
31	**CAN'T RESUME** (RECUPERO IMPOSSIBILE)	Si è incontrata un'istruzione RESUME senza un'istruzione TRAP attiva.
32	**LOOP NOT FOUND** (LOOP NON TROVATO)	Il programma ha incontrato una istruzione DO e non può trovare il LOOP corrispondente.
33	**LOOP WITHOUT DO** (LOOP SENZA DO)	Si è incontrato un LOOP senza un'istruzione DO attiva.
34	**DIRECT MODE ONLY** (SOLO MODALITÀ DIRETTA)	Questo comando è permesso solo in modalità diretta, non da un programma.
35	**NO GRAPHICS AREA** (AREA NON GRAFICA)	Si è incontrato un comando (DRAW, BOX, ecc.) per creare grafici prima che fosse eseguito un comando GRAPHIC.
36	**BAD DISK** (DISCO DIFETTOSO)	Si è tentato di riformattare un dischetto non formattato o difettoso con il metodo di formattazione veloce (senza ID).

DESCRIZIONE DEI MESSAGGI D'ERRORE DOS



Questi messaggi d'errore vengono ritornati attraverso le variabili riservate DS e DS$.

NOTA: I numeri di messaggio d'errore inferiori a 20 dovrebbero venire ignorati eccetto 01, che informa sul numero di file cancellati con il comando SCRATCH.

20	**READ ERROR** (ERRORE DI LETTURA) (intestazione del blocco non trovata)	Il controllore del disco non può allocare l'intestazione del blocco di dati richiesto. Ciò può essere provocato da un numero di settore illegale, o dal fatto che l'intestazione è stata distrutta.
21	**READ ERROR** (ERRORE DI LETTURA) (carattere non sincronizzato)	Il controllore del disco non può rilevare un indicatore di sincronizzazione sulla traccia desiderata. Questo può essere provocato da un disallineamento della testina di lettura/scrittura, dalla mancanza del dischetto, da un dischetto non formattato o inserito non correttamente. Può anche indicare un guasto hardware.
22	**READ ERROR** (ERRORE DI LETTURA) (blocco dati non presente)	Al controllore del disco è stato richiesto di leggere o verificare un blocco di dati che non è stato scritto correttamente. Questo messaggio d'errore si presenta in congiunzione con i comandi BLOCK ed indica una richiesta di traccia e/o settore illegale.

23	**READ ERROR** (ERRORE DI LETTURA) (errore di checksum nel blocco di dati)	Questo messaggio d'errore indica che c'è un errore in uno o più d'uno dei byte dei dati. I dati sono stati letti nella memoria DOS, ma il checksum sui dati è in errore. Questo messaggio può anche indicare problemi di messa a terra.
24	**READ ERROR** (ERRORE DI LETTURA) (errore di decodificazione del byte)	I dati o l'intestazione sono stati letti nella memoria DOS, ma si è creato un errore hardware a causa di una configurazione di bit non valida in un byte di dati. Questo messaggio può anche indicare problemi di messa a terra.
25	**WRITE ERROR** (ERRORE DI SCRITTURA) (errore di verifica di scrittura)	Questo messaggio viene generato se il controllore rileva un'incongruenza tra i dati scritti e i dati della memoria DOS
26	**WRITE PROTECT ON** (PROTEZIONE IN SCRITTURA SULL'ON)	Questo messaggio viene generato quando è stato richiesto al controllore di scrivere un blocco di dati mentre l'interruttore di protezione di scrittura è sull'on. Di solito questo viene causato utilizzando un dischetto protetto in scrittura.
27	**READ ERROR** (ERRORE DI LETTURA) (errore di checksum nella intestazione)	Il controllore ha rilevato un errore nell'intestazione del blocco di dati richiesto. Il blocco non è stato letto nella memoria DOS. Questo messaggio può anche indicare problemi di messa a terra.

28	**WRITE ERROR** (ERRORE DI SCRITTURA) (blocco di dati lungo)	Il controllore tenta di rilevare l'indicatore di sincronizzazione dell'intestazione seguente dopo aver scritto un blocco di dati. Se l'indicatore di sincronizzazione non appare entro un tempo predeterminato, viene generato il messaggio d'errore. L'errore viene causato da una formattazione non corretta del dischetto (i dati si estendono nel blocco seguente) o da un guasto hardware.
29	**DISK ID MISMATCH** (ERRORE DI IDENTIFICAZIONE DEL DISCO)	Questo messaggio viene generato quando è stato richiesto al controllore di accedere a un dischetto che non è stato inizializzato. Il messaggio può anche presentarsi se un dischetto ha un'intestazione non corretta.
30	**SYNTAX ERROR** (ERRORE DI SINTASSI) (sintassi generale)	Il DOS non può interpretare il comando inviato al canale di comando. Solitamente questo errore viene provocato da un numero illegale di nomi di file, o dal fatto che le configurazioni vengono usate illegalmente. Per esempio, due nomi di file possono apparire sul lato sinistro del comando COPY.
31	**SYNTAX ERROR** (ERRORE DI SINTASSI) (comando non valido)	Il DOS non riconosce il comando. Il comando deve cominciare dalla prima posizione.

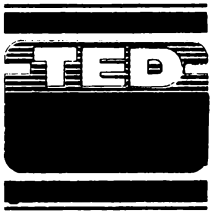
32	**SYNTAX ERROR** (ERRORE DI SINTASSI) (comando non valido)	Il comando trasmesso è più lungo dei 58 caratteri consentiti.
33	**SYNTAX ERROR** (ERRORE DI SINTASSI) (nome del file non valido)	La corrispondenza di configurazione è usata in modo non ammesso nei comandi OPEN o SAVE.
34	**SYNTAX ERROR** (ERRORE DI SINTASSI) (nessun file dato)	Il nome del file è stato escluso da un comando o il DOS non lo riconosce come tale. Di solito sono i 2 punti (:) ad essere esclusi dal comando.
39	**SYNTAX ERROR** (ERRORE DI SINTASSI) (comando non valido)	Questo errore può risultare se il comando trasmesso al canale di comando (indirizzo secondario 15) non viene riconosciuto dal DOS.
50	**RECORD NOT PRESENT** (RECORD NON PRESENTE)	Risultato della lettura del disco, al di là dell'ultimo record, attraverso i comandi INPUT# o GET# . Questo messaggio si presenterà anche dopo il posizionamento su un record oltre la fine del file in un dato file. Se lo scopo è di espandere il file aggiungendo il nuovo record (con un comando PRINT#) il messaggio d'errore può essere ignorato. I comandi INPUT o GET non si dovrebbero più tentare dopo il rilevamento di questo errore senza prima effettuare un riposizionamento.
51	**OVERFLOW IN RECORD** (OVERFLOW DEL RECORD)	L'istruzione PRINT# supera i limiti del record. L'informazione viene

		troncata. Dato che il ritorno del carrello, che viene trasmesso come indicatore di fine record, viene calcolato nella dimensione del record, questo messaggio si presenterà se l'insieme dei caratteri del record (incluso il ritorno carrello finale) supera le dimensioni stabilite.
52	**FILE TOO LARGE** (FILE TROPPO GRANDE)	La posizione del record all'interno di un dato file, indica che ci sarà un overflow del disco.
60	**WRITE FILE OPEN** (FILE DI SCRITTURA APERTO)	Questo messaggio viene generato quando un file di scrittura che non è stato chiuso viene aperto per la lettura.
61	**FILE NOT OPEN** (FILE NON APERTO)	Questo messaggio viene generato quando si accede a un file che non è stato aperto nel DOS. A volte, in casi simili, non viene neanche generato un messaggio; la richiesta viene semplicemente ignorata.
62	**FILE NOT FOUND** (FILE NON TROVATO)	Il file richiesto non esiste nel drive indicato.
63	**FILE EXISTS** (FILE ESISTENTE)	Il nome del file che si sta creando già esiste nel dischetto.
64	**FILE TYPE MISMATCH** (CORRISPONDENZA DI TIPO DI FILE ERRATA)	Il tipo di file non corrisponde a quello della posizione sull'elenco per il file richiesto.
65	**NO BLOCK** (NESSUN BLOCCO)	Questo messaggio si presenta in congiunzione col comando B-A ed indica che il blocco da allocare è già stato allocato. I parametri indicano la traccia e il

		settore disponibili con il numero successivo più alto. Se i parametri corrispondono a zero (0), vengono utilizzati tutti i blocchi di numero più alto.
66	**ILLEGAL TRACK AND SECTOR** (TRACCIA E SETTORE ILLEGALI)	Il DOS ha tentato di accedere a una traccia o a un blocco che non esiste nel formato utilizzato. Ciò può indicare un problema di individuazione del puntatore del blocco successivo.
67	**ILLEGAL SYSTEM T OR S** (TRACCIA O SETTORE ILLEGALE DI SISTEMA)	Questo speciale messaggio d'errore indica una traccia o un settore illegale del sistema.
70	**NO CHANNEL** (NESSUN CANALE) (disponibile)	Il canale richiesto non è disponibile, oppure tutti i canali sono già occupati. Un massimo di cinque file sequenziali può venire aperto contemporaneamente al DOS. I canali ad accesso diretto possono avere sei file aperti.
71	**DIRECTORY ERROR** (ERRORE D'ELENCO)	Il BAM non corrisponde al conteggio interno. C'è un problema nell'allocazione BAM oppure sono stati scritti dei dati nella zona riservata alla BAM nella memoria DOS. Per correggere questo problema, reinizializzare il dischetto per ripristinare la BAM nella memoria. Alcuni file attivi possono essere interrotti dall'azionecorrettiva. NOTA: BAM = Mappa di disponibilità del blocco.

72	**DISK FULL** (DISCO PIENO)	O sono stati usati tutti i blocchi del dischetto oppure è terminato lo spazio disponibile nell'elenco. Il messaggio DISK FULL viene trasmesso quando sul 1541 restano disponibili solo due blocchi per la chiusura del file corrente.
73	**DOS MISMATCH** (73, CBM DOS V2.6 1541)	I DOS 1 e 2 sono compatibili in lettura ma non in scrittura. I dischi possono essere letti indifferentemente con ambedue i DOS, ma un disco formattato in una certa versione non può venire scritto nell'altra versione perché il formato è diverso. Questo messaggio viene visualizzato ogni qual volta venga effettuato un tentativo di scrivere su un disco che è stato formattato in un formato non compatibile (è disponibile una routine di utilità come aiuto alla conversione da un formato all'altro). Questo messaggio può anche apparire dopo l'accensione.
74	**DRIVE NOT READY** (DRIVE NON DISPONIBILE)	È stato fatto un tentativo per accedere all'Unità Floppy Disk senza utilizzare un dischetto.

SEZIONE 5: Introduzione



Tedmon

Il TEDMON è un programma in linguaggio macchina incorporato che permette di scrivere facilmente programmi in linguaggio macchina. Il TEDMON comprende un monitor di linguaggio macchina, un miniassembler e un disassembler.
I programmi in linguaggio macchina scritti utilizzando il TEDMON possono essere utilizzati autonomamente oppure venire usati come subroutine molto veloci per programmi BASIC, dato che il TEDMON può tranquillamente coesistere con il BASIC.

Comandi Tedmon

A	**ASSEMBLA**	Assembla una riga del codice 6502.
C	**CONFRONTA**	Confronta due sezioni della memoria e segnala le differenze.
D	**DISASSEMBLA**	Disassembla una riga del codice 6502.
F	**RIEMPI**	Riempie la memoria con il byte specificato.
G	**ESEGUI**	Avvia l'esecuzione all'indirizzo specificato.
H	**CERCA**	Ricerca nella memoria tutte le posizioni di determinati byte.
L	**CARICA**	Carica un file dal nastro o dal disco.
M	**VISUALIZZA MEMORIA**	Visualizza i valori esadecimali delle locazioni di Memoria.
R	**VISUALIZZA REGISTRI**	Visualizza i registri 6502.
S	**SALVA**	Salva su nastro o su disco.
T	**TRASFERISCI**	Trasferisce il codice da una sezione della memoria a un'altra.
V	**VERIFICA**	Confronta la memoria con il nastro o con il disco.
X	**ESCI**	Uscita da TEDMON.
.	**(punto)**	Assembla una riga del codice 6502.
>	**(maggiore di)**	Modifica la memoria.
;	**(punto e virgola)**	Modifica la visualizzazione del Registro 6502.

La locazione $7F8 controlla se TEDMON vede la ROM e la RAM sopra $8000. Se questa locazione è impostata sullo 0, TEDMON visualizza il BASIC e il KERNAL dopo un comando di disassemblaggio o di una stampa di tutta la memoria sopra $8000. Se questa locazione viene impostata su $80, TEDMON visualizza la RAM sotto il BASIC e il KERNAL. Ciò è spesso conveniente per lo sviluppo di programmi in linguaggio macchina. Notare che la locazione $7F8 non ha alcuna influenza sul comando G. Il comando G avvia l'esecuzione nella mappa di memoria corrente (ROM on o RAM on) senza tener conto dell'impostazione della locazione $7F8.

Come Utilizzare Tedmon

Si accede a TEDMON battendo:

MONITOR

TEDMON risponde visualizzando i registri 6502 e facendo lampeggiare il cursore. Il cursore rappresenta il prompt che ricorda che TEDMON è in attesa dei comandi.

Descrizione dei Comandi

COMANDO: **A**

SCOPO: Introduce una riga del codice d'assemblaggio.

SINTASSI: **A** *<indirizzo> <codice operativo mnemonico> <operando> <indirizzo>*. È un numero esadecimale che indica la locazione della memoria per il collocamento del codice operativo.

<codice operativo mnemonico>. Mnemonico in linguaggio assembler per tecnologia MOS standard, per es. LDA, STX, ROR, ecc.

<operando>. L'operando, quando richiesto, può essere di una qualsiasi delle modalità d'indirizzamento legale. (Per le modalità di pagina-zero un numero esadecimale di due cifre è quello i cui valori sono inferiori a $100. Per indirizzi di pagina-non zero vengono richiesti numeri esadecimali di 4 cifre).

Un RETURN viene utilizzato per indicare la fine della riga di assemblaggio. Se nella riga risultano degli errori, verrà visualizzato un punto di domanda ad indicare un errore e il cursore si sposterà alla riga seguente. Per correggere eventuali errori della riga, potrà essere

utilizzato lo screen editor. Una volta assemblata con successo una riga del codice, l'assemblatore stampa un prompt contenente la successiva locazione legale di memoria per una eventuale istruzione, così che A e il numero di riga non dovranno essere battuti più di una volta in caso di introduzione nel C-264 di programmi in linguaggio assembler.

ESEMPIO:

.A 1200 LDX # $00
.A 1202

NOTA: Un punto (.) equivale al comando ASSEMBLA.

ESEMPIO:

. 2000 LDA # $23

COMANDO: **C**

SCOPO: Confronta due aree di memoria

SINTASSI: **C** *<indirizzo 1> <indirizzo 2> <indirizzo 3>*

<Indirizzo 1> è un numero esadecimale indicante l'indirizzo iniziale dell'area di memoria da confrontare.

<Indirizzo 2> è un numero esadecimale indicante l'indirizzo finale dell'area di memoria da confrontare.

<Indirizzo 3> è un numero esadecimale indicante l'indirizzo iniziale dell'altra area di memoria da confrontare.

Se le due aree di memoria sono uguali, TEDMON stampa un RETURN, che indica che la seconda area di memoria è uguale alla prima. Gli indirizzi dei byte delle due aree che presentano differenze vengono stampati sullo schermo.

COMANDO: **D**

SCOPO: Disassembla il codice macchina in mnemonici e operandi di linguaggio assembler.

SINTASSI: **D** [<*indirizzo*>] [<*indirizzo 2*>]

<*Indirizzo*> È un numero esadecimale che imposta l'indirizzo per avviare il disassemblaggio.

<*Indirizzo 2*> È un indirizzo finale esadecimale opzionale del codice da disassemblare.

Il formato del disassemblaggio è solo leggermente diverso dal formato di input di un assemblaggio. La differenza sta nel fatto che il primo carattere di un disassemblaggio è una virgola invece di una A (per la leggibilità), e che l'esadecimale del codice viene anch'esso listato.
Un listato di disassemblaggio può essere modificato utilizzando lo screen editor. Operare dei cambiamenti sul mnemonico o sull'operando sullo schermo e quindi premere return. Ciò introduce la riga e richiama l'assembler per ulteriori modifiche.
Un disassemblaggio può essere visualizzato a pagine. L'introduzione di una D fa si che la pagina successiva del disassemblaggio venga fatta scorrere sullo schermo.

```
ESEMPIO:  D     3000  3004
          .     3000  A900      LDA# $00
          .     3002  FF        ???
          .     3003  DO  2B    BNE $3030
```

COMANDO: **F**

SCOPO: Riempie una fascia di locazioni con un byte specificato

SINTASSI: **F** <*indirizzo 1*> <*indirizzo 2*> <*byte*>

<*Indirizzo 1*> È la prima locazione da riempire con il <*byte*>

<*indirizzo 2*> È l'ultima locazione da riempire con il <*byte*>

<valore del byte> È il numero esadecimale di 1 o 2 cifre che deve essere scritto.

Questo comando è utile per inizializzare le strutture dei dati o qualsiasi altra area RAM.

ESEMPIO: F 0400 0518 EA

Riempie con $EA (una istruzione NOP) le locazioni di memoria da $0400 a $0518.

COMANDO: **G**

SCOPO: Iniziare l'esecuzione di un programma all'indirizzo specificato.

SINTASSI: **G** [*<indirizzo>*]

<indirizzo> È un argomento opzionale che specifica il nuovo valore del contatore del programma e fornisce l'indirizzo da cui deve cominciare l'esecuzione. Nel caso in cui *<indirizzo>* venga tralasciato, l'esecuzione inizia dal CP corrente. (Il CP corrente può essere visualizzato utilizzando il comando R.)
Il comando GO ripristina tutti i registri (visualizzabili con il comando R) ed inizia l'esecuzione dall'indirizzo iniziale specificato. Si raccomanda cautela nell'uso del comando GO. Per tornare a TEDMON dopo l'esecuzione di un programma in linguaggio macchina, utilizzare l'istruzione BRK.

ESEMPIO: G 140C

L'esecuzione inizia dalla locazione $140C.

COMANDO: **H**

SCOPO: Ricerca tutte le posizioni di determinati byte all'interno di una fascia specificata nella memoria.

SINTASSI: **H** *<indirizzo 1>* *<indirizzo 2>* *<dati>*

<indirizzo 1> indirizzo iniziale della procedura di ricerca

<indirizzo 2> indirizzo finale della procedura di ricerca

<*dati*> l'insieme di dati per la ricerca dei dati può essere una stringa esadecimale o ASCII. Una stringa ASCII viene specificata facendo precedere il primo carattere da una sola virgoletta, per es.: 'STRING. I dati possono essere argomenti ad elementi singoli o multipli. Se multipli o in esadecimale, ogni numero deve essere separato da uno spazio.

ESEMPIO: H C000 FFFF 'READ	Ricerca la stringa ASCII leggendola da C000 a FFFF
H A000 A101 A9 FF 4C	Ricerca i dati $A9, $ff, $4C, da A100 a A101

COMANDO: **L**

SCOPO: Carica un file da una cassetta o da un disco.

SINTASSI: **L** <*nomefile*>, <*dispositivo*>

<*nomefile*> è qualsiasi nome legale di file del Plus/4 tra virgolette.

<*dispositivo*> è un numero esadecimale indicante il dispositivo dal quale caricare.

1 cassetta
8 disco (o 9, A, ecc.)

Il comando Load fa si che un file venga caricato nella memoria. L'indirizzo iniziale è contenuto nei primi due byte del file (file di programma). In altre parole, il comando Load carica sempre un file nello stesso posto da cui è stato salvato. Questo è molto importante per le operazioni in linguaggio macchina, dato che pochi programmi sono completamente rilocabili. Il file viene caricato nella memoria finché non si incontra un indicatore di fine file (EOF).

ESEMPIO:

L "SCREEN", 1	Legge un file dalla cassetta
L "TANK", 8	Legge un file dal disco

COMANDO: **M**

SCOPO: Visualizza la memoria come una stampa esadecimale e ASCII all'interno della specifica gamma indirizzo.

SINTASSI: **M** [<*indirizzo 1*>] [<*indirizzo*>]

[<*indirizzo 1*>] È il primo indirizzo di stampa della memoria. Opzionale. In caso venga omesso, verrà visualizzata una pagina. Il primo byte è l'ultimo indirizzo specificato.

[<*indirizzo 2*>] È l'ultimo indirizzo di stampa della memoria. Opzionale. In caso venga omesso, verrà visualizzata una pagina. Il primo byte è rappresentato dai dati di [<*indirizzo 1*>].

La memoria viene visualizzata nel seguente formato:

>A048 41 E7 00 AA AA 00 98 56 45 :A!.*..VE

Il contenuto della memoria può essere corretto utilizzando lo screen editor. Spostare il cursore sui dati da modificare, battere la correzione desiderata e premere RETURN. In caso di locazione RAM scorretta o di un tentativo di modifica della ROM, verrà visualizzato un flag (?) d'errore.
Una stampa della memoria ASCII dei dati viene visualizzata invertita (per contrastare la stampa degli altri dati visualizzati sullo schermo) alla destra dei dati esadecimali. Quando un carattere non è stampabile, verrà visualizzato come un punto invertito (.).
Come per il comando di disassemblaggio, si potrà visualizzare la pagina successiva battendo M e RETURN.

ESEMPIO:

```
M 1C00
>1C00 41 E7 00 AA AA 00 98 56 45 :A!.*..VE
>1C08 41 E7 00 AA AA 00 98 56 45 :A!.*..VE
>1C10 41 E7 00 AA AA 00 98 56 45 :A!.*..VE
>1C18 41 E7 00 AA AA 00 98 56 45 :A!.*..VE
>1C20 41 E7 00 AA AA 00 98 56 45 :A!.*..VE
>1C28 41 E7 00 AA AA 00 98 56 45 :A!.*..VE
>1C30 41 E7 00 AA AA 00 98 56 45 :A!.*..VE
>1C38 41 E7 00 AA AA 00 98 56 45 :A!.*..VE
>1C40 41 E7 00 AA AA 00 98 56 45 :A!.*..VE
>1C48 41 E7 00 AA AA 00 98 56 45 :A!.*..VE
>1C50 41 E7 00 AA AA 00 98 56 45 :A!.*..VE
>1C58 41 E7 00 AA AA 00 98 56 45 :A!.*..VE
```

COMANDO: **>**

SCOPO: Può essere utilizzato per impostare contemporaneamente da 1 a 8 locazioni di memoria.

SINTASSI: **>** *indirizzo byte di dati 1 <byte di dati da 2 a 8>*

indirizzo: primo indirizzo di memoria da impostare
byte di dati 1: dati da collocare all'indirizzo
<byte di dati da 2 a 8> : dati da collocare nelle locazioni di memoria successive al primo indirizzo. Opzionale.

ESEMPIO

>2000 08	colloca uno 08 alla locazione 2000
>3000 23 45 65	colloca un 23 alla locazione 3000, un 45 alla 3001 e un 65 alla 3002

COMANDO: **R**

SCOPO: Evidenzia importanti registri 6502. Vengono visualizzati il registro dello stato di programma, il contatore di programma, l'accumulatore, i registri di indice X e Y e il puntatore di stack.

SINTASSI: **R**

ESEMPIO:
```
R
     PC   SR   AC   XR   YR   SP
;  1002 01   02   03   04   F6
```

NOTA: il ; (punto e virgola) può essere usato per modificare le visualizzazioni di registro, allo stesso modo in cui usando > è possibile modificare i registri di memoria.

COMANDO: **S**

SCOPO: Salva il contenuto della memoria su nastro o su disco.

SINTASSI: **S** <*"nomefile"*>,<*unità*>,<*indirizzo 1*>,<*indirizzo 2*> <*"nomefile"*>. Tutti i nomi file del Plus/4 ammessi. Per salvare i dati, il nome del file deve essere racchiuso tra virgolette. Non è possibile utilizzare apici.

<*unità* > Due possibili dispositivi possono essere cassette e disco. Per memorizzare su cassetta, utilizzare l'unità 1: il numero dell'unità a disco del Plus/4 è solitamente 8. Comunque sia, questo può essere modificato (per esempio, quando vengono usate più unità). Vedere il MANUALE DELL'UNITÀ A DISCO del Plus/4.

<*indirizzo 1* > Indirizzo iniziale di memoria da salvare.

<*indirizzo 2* > Indirizzo finale di memoria da salvare + 1.
Vengono salvati tutti i dati fino al byte di dati di questo indirizzo escluso. Il file creato da questo comando è un file di programma. I primi due byte contengono l'indirizzo base <*indirizzo 1* > dei dati. Il file potrà essere richiamato utilizzando il comando L.

ESEMPIO: S "GAME", 8, 0400, 0BFF

Salva il contenuto della memoria sul disco a partire da $0400 fino a $0BFF

COMANDO **T**:

SCOPO: Trasferisce segmenti di memoria da un'area di memoria a un'altra.

SINTASSI: **T**<*indirizzo 1*> <*indirizzo 2*> <*indirizzo 3*>

<*indirizzo 1*> Indirizzo di partenza dei dati da trasferire.

<*indirizzo 2*> Indirizzo finale dei dati da trasferire.

<*indirizzo 3*> Indirizzo di partenza della nuova locazione (dove i dati devono essere trasferiti).

I dati possono essere trasferiti da indirizzi bassi o indirizzi alti o viceversa. Dei segmenti di memoria aggiuntivi di qualsiasi lunghezza possono essere spostati in avanti o all'indietro di un numero qualsiasi di byte (cioè trasferiti).

ESEMPIO: T 1400 1600 1401

Fa scorrere di un byte i dati in memoria da $1400 a $1600 incluso.

COMANDO **V**:

SCOPO: Verifica un file su cassetta o su disco confrontandolo con il contenuto della memoria.

SINTASSI: **V**<*"nomefile"*>, <*dispositivo*>

<*nomefile*> è un qualsiasi nome di file ammesso.

<*dispositivo*> è un numero esadecimale che indica in quale unità si trova il file, la cassetta è 1 o 01, il disco è 8 o 08, 09, ecc.

Il comando Verify confronta un file con il contenuto della memoria. Il Plus/4 risponde con VERIFYING (Verifica in corso). Se viene riscontrato un errore, viene aggiunta la parola ERROR (errore); se la verifica ha buon esito, riappare il cursore lampeggiante.

ESEMPIO: V "WORKLOAD", 8

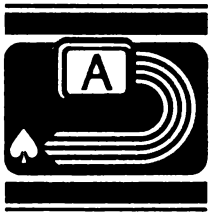
COMANDO **X**:

SCOPO: Torna al BASIC

SINTASSI: **X**

Quando viene dato il comando X, il puntatore stack della macchina viene portato al suo valore corrente (vedere il comando R). Se questo fosse modificato dopo essere tornati al BASIC, utilizzare il comando BASIC CLR per re-impostare i puntatori.

SEZIONE 6



Codificatore della Visualizzazione sullo Schermo

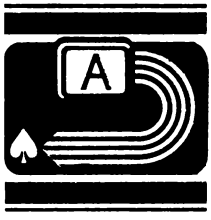
Il seguente diagramma elenca tutti i caratteri incorporati nel set di caratteri della Commodore. Mostra i numeri scritti con la funzione POKE nella memoria di schermo (posizioni da 3072 a 4095) per ottenere il carattere desiderato. Viene mostrata inoltre la corrispondenza di un carattere al numero letto dallo schermo con la funzione PEEK.
Sono disponibili due set di caratteri, da utilizzare uno per volta. Questo significa che non è possibile avere sullo schermo i caratteri di un set contemporaneamente ai caratteri dell'altro set. I set vengono selezionati premendo contemporaneamente SHIFT e C= .
Dal BASIC, PRINT CHR$(142) selezionerà la modalità maiuscola/grafica, e PRINT CHR$(14) selezionerà la modalità maiuscola/minuscola.
Tutti i numeri del diagramma possono essere visualizzati INVERTITI. Questo viene ottenuto aggiungendo 128 al valore mostrato.

SET 1	SET 2	POKE	SET 1	SET 2	POKE	SET 1	SET 2	POKE
@		0	T	t	20	(		40
A	a	1	U	u	21	)		41
B	b	2	V	v	22	*		42
C	c	3	W	w	23	+		43
D	d	4	X	x	24	,		44
E	e	5	Y	y	25	-		45
F	f	6	Z	z	26	.		46
G	g	7	[		27	/		47
H	h	8	£		28	0		48
I	i	9	]		29	1		49
J	j	10	↑		30	2		50
K	k	11	←		31	3		51
L	l	12	SPAZIO		32	4		52
M	m	13	!		33	5		53
N	n	14	"		34	6		54
O	o	15			35	7		55
P	p	16	$		36	8		56
Q	q	17	%		37	9		57
R	r	18	&		38	:		58
S	s	19	'		39	;		59

SET 1	SET 2	POKE	SET 1	SET 2	POKE	SET 1	SET 2	POKE
<		60		T	84			108
=		61		U	85			109
>		62		V	86			110
?		63		W	87			111
		64		X	88			112
	A	65		Y	89			113
	B	66		Z	90			114
	C	67			91			115
	D	68			92			116
	E	69			93			117
	F	70			94			118
	G	71			95			119
	H	72	SPAZIO		96			120
	I	73			97			121
	J	74			98		☑	122
	K	75			99			123
	L	76			100			124
	M	77			101			125
	N	78			102			126
	O	79			103			127
	P	80			104			
	Q	81			105			
	R	82			106			
	S	83			107			

I codici da 128 a 255 sono le immagini invertite dei codici da 0 a 127.

SEZIONE 7



Codici ASCII e CHR$

Questa appendice mostra i caratteri che appariranno battendo PRINT CHR$(X) per tutti i possibili valori di X. Vengono inoltre mostrati i valori ottenuti battendo PRINT ASC("X"), dove X è un qualsiasi carattere battuto. Questo è utile per valutare il carattere ricevuto in un'istruzione GET, per convertire le maiuscole in minuscole, e per stampare i comandi basati sui caratteri (come trasformazione da maiuscolo a minuscolo) che non possono essere tra virgolette.

STAMPA	CHR$	STAMPA	CHR$	STAMPA	CHR$	STAMPA	CHR$
	0	↓	17	"	34	3	51
	1	RVS ON	18		35	4	52
	2	CLR HOME	19	$	36	5	53
	3	INST DEL	20	%	37	6	54
	4		21	&	38	7	55
WHT	5		22	'	39	8	56
	6		23	(	40	9	57
	7		24	)	41	:	58
DISABLES SHIFT C=	8		25	*	42	;	59
ENABLES SHIFT C=	9		26	+	43	<	60
	10	ESCAPE	27	,	44	=	61
	11	RED	28	-	45	>	62
	12	→	29	.	46	?	63
RETURN	13	GRN	30	/	47	@	64
SWITCH TO LOWER CASE	14	BLU	31	0	48	A	65
	15	SPAZIO	32	1	49	B	66
	16	!	33	2	50	C	67

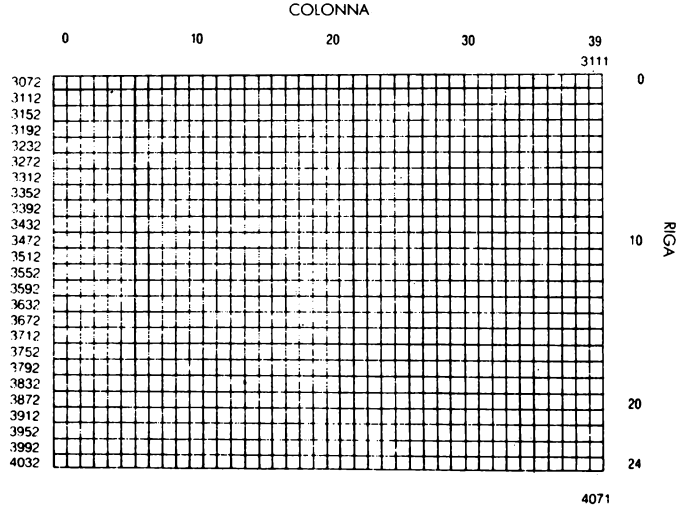
STAMPA	CHR$	STAMPA	CHR$	STAMPA	CHR$	STAMPA	CHR$
D	68		97		126	LT GREEN	155
E	69		98		127	PUR	156
F	70		99		128		157
G	71		100	ORANGE	129	YEL	158
H	72		101	FLASH ON	130	CYN	159
I	73		102		131	SPACE	160
J	74		103	FLASH OFF	132		161
K	75		104		133		162
L	76		105		134		163
M	77		106		135		164
N	78		107		136		165
O	79		108		137		166
P	80		109		138		167
Q	81		110		139		168
R	82		111		140		169
S	83		112	SHIFT RETURN	141		170
T	84		113	SWITCH TO UPPER CASE	142		171
U	85		114		143		172
V	86		115	BLK	144		173
W	87		116		145		174
X	88		117	RVS OFF	146		175
Y	89		118	CLEAR HOME	147		176
Z	90		119	INST DEL	148		177
[	91		120	BROWN	149		178
£	92		121	YEL/GREEN	150		179
]	93		122	PINK	151		180
↑	94		123	BL/GREEN	152		181
←	95		124	LT BLUE	153		182
	96		125	BLUE	154		183

STAMPA	CHR$	STAMPA	CHR$	STAMPA	CHR$	STAMPA	CHR$
	184		186		188		190
	185		187		189		191

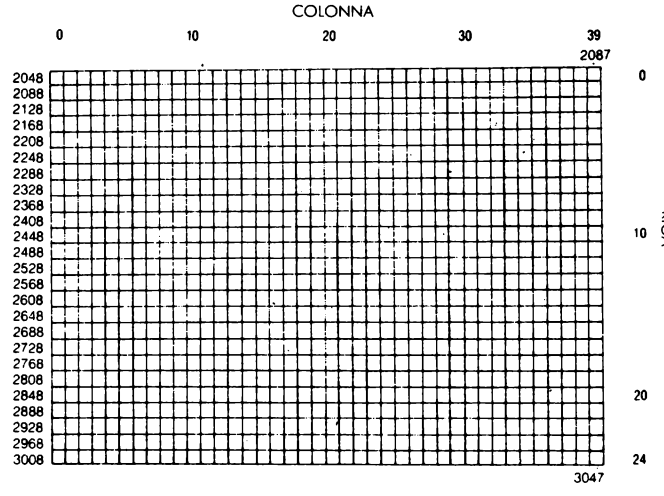
CODICI	192-223	COME	96-127
CODICI	224-254	COME	160-190
CODICE	255	COME	126

Il seguente diagramma elenca quali locazioni di memoria controllano la disposizione dei caratteri sullo schermo, e le locazioni utilizzate per variare i colori individuali dei caratteri, inoltre mostra i codici di colore dei caratteri.

MAPPA DI MEMORIA DELLO SCHERMO



MAPPA DI MEMORIA DI COLORE



I valori effettivi per cambiare il colore dei caratteri sono:

1 NERO	9 ARANCIONE
2 BIANCO	10 MARRONE
3 ROSSO	11 VERDE-GIALLO
4 CIANO	12 ROSA
5 VIOLA	13 VERDAZZURRO
6 VERDE	14 CELESTE
7 AZZURRO	15 BLU NOTTE
8 GIALLO	16 VERDE CHIARO

La luminanza del colore viene selezionata moltiplicando il valore di luminanza (0-7) per 16, e sommando il risultato al numero del colore. Per rendere il carattere lampeggiante, aumentare il valore di 128.

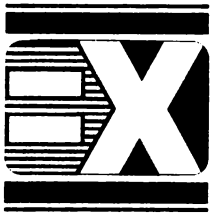
Mappa Registro Memoria PLUS/4

REG			DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	$FF00		TIMER # 1 RELOAD VALUE, BITS 0-7 (LOW)							
1	$FF01		TIMER # 1 RELOAD VALUE, BITS 8-15 (HIGH)							
2	$FF02		TIMER # 2 RELOAD VALUE, BITS 0-7 (LOW)							
3	$FF03		TIMER # 2 RELOAD VALUE, BITS 8-15 (HIGH)							
4	$FF04		TIMER # 3 RELOAD VALUE, BITS 0-7 (LOW)							
5	$FF05		TIMER # 3 RELOAD VALUE, BITS 8-15 (HIGH)							
6	$FF06		TEST	ECM	BMM	BLANK	# ROWS	Y2	Y1	Y0
7	$FF07		RVS OFF	PAL/	FREEZE	MCM	# COLS	X2	X1	X0
8	$FF08		KEYBOARD LATCH							
9	$FF09		IRQ	I-T3	NC	I-T2	I-T1	I-LP	I-RAS	NC
10	$FF0A		NC	EI-T3	NC	EI-T2	EI-T1	EI-LP	EI-RAS	RC8
11	$FF0B		RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0
12	$FF0C		NC	NC	NC	NC	NC	NC	C9	CUR8
13	$FF0D		CUR7	CUR6	CUR5	CUR4	CUR3	CUR2	CUR1	CUR0
14	$FF0E		SND1-7	SND1-6	SND1-5	SND1-4	SND1-3	SND1-2	SND1-1	SND1-0
15	$FF0F		SND2-7	SND2-6	SND2-5	SND2-4	SND2-3	SND2-2	SND2-1	SND2-0
16	$FF10		NC	NC	NC	NC	NC	NC	SND2-9	SND2-8
17	$FF11		SND-REL NOISE		V2-SEL	V1-SEL	VOL3	VOL2	VOL1	VOL0
18	$FF12		NC	NC	BMB2	BMB1	BMB0	R BANK	S1-9	S1-8
19	$FF13		CB5	CB4	CB3	CB2	CB1	CB0	SCLOCK	STATUS
20	$FF14		VM4	VM3	VM2	VM1	VM0	NC	NC	NC
21	$FF15	BKGD0	NC	LUM2	LUM1	LUM0	COLOR3	COLOR2	COLOR1	COLOR0
22	$FF16	BKGD1	NC	LUM2	LUM1	LUM0	COLOR3	COLOR2	COLOR1	COLOR0
23	$FF17	BKGD2	NC	LUM2	LUM1	LUM0	COLOR3	COLOR2	COLOR1	COLOR0
24	$FF18	BKGD3	NC	LUM2	LUM1	LUM0	COLOR3	COLOR2	COLOR1	COLOR0
25	$FF19	BKDG4	NC	LUM2	LUM1	LUM0	COLOR3	COLOR2	COLOR1	COLOR0
26	$FF1A		NC	NC	NC	NC	NC	NC	BRE9	BRE8
27	$FF1B		BRE7	BRE6	BRE5	BRE4	BRE3	BRE2	BRE1	BRE0
28	$FF1C		NC	NC	NC	NC	NC	NC	NC	VL8
29	$FF1D		VL7	VL6	VL5	VL4	VL3	VL2	VL1	VL0
30	$FF1E		H8	H7	H6	H5	H4	H3	H2	H1
31	$FF1F		NC	BL3	BL2	BL1	BL0	VSUB2	VSUB1	VSUB0
62	$FF3E		ROM SELECT							
63	$FF3F		RAM SELECT							

INDIRIZZO	CONTENUTO		NOTE
	<.....	>	
$FFFE-FFFF	<VETTORE IRQ	> *	
$FFFC	<VETTORE RES	> *	
$FFFA	<VETTORE NMI (non utilizzato)	< *	BANCO ROM ALTO (cont.)
	<	> *	
$FF81-FFF5	<TABELLA DI JUMP DEL KERNAL	> *	
$FF40-FF80	<	>	
$FF00-FF3F	<CHIP TED	> *	
	<	> *	
$FE00-FEFF	<SISTEMA DISCO DMA	> *	CHIP TED E SPAZIO
$FDE0-FDEF	<	> *	I/O SONO PRESENTI
$FDD0-FDDF	<PORTA DEL BANCO DI CARTUCCIA	> *	IN TUTTE LE MAPPE
$FD10-FD1F	<PORTA PARALLELA 6529	< *	DI MEMORIA
$FD00-FD0F	<ACIA	> *	
	<	>	
$FCD0-FCFF	<	>	ROUTINE DI BANCO ROM
	<	<	(presenti in tutte le mappe ROM)
$D800-FCFF	<	> *	
	<	> *	
$D000-D7FF	<ROM DEI CARATTERI	> *	BANCO DI ROM ALTO
	<	> *	
$C000-D7FF	<BASIC ADDIZIONALE	> *	
	<	>	
$8000-BFFF	<BASIC	>	BANCO DI ROM BASSO
	<	>	
$4000-FFFF	<RAM E INIZIO TESTO BASIC	>	
	<AREA DOVE SONO UTILIZZATI I GRAFICI	>	
	<AD ALTA RISOLUZIONE	>	
$2000-3FFF	<DATI BIT MAP DI SCHERMO	>	
	<	>	
$1C00-1FFF	<MATRICE DI VIDEO AD ALTA RISOLUZIONE	>	
	<	>	
$1800-1BFF	<BYTE DI ATTRIBUTI DI SCHERMO AD ALTA RISOLUZIONE	>	
	<	>	
$1000-	<AREA DI TESTO BASIC (BIT MAP DISATTIVATO)	>	
	<	>	
$0C00-0FFF	<MATRICE DI SCHERMO DI TESTO (memoria di schermo	>	
	<	>	
$0800-0BFF	<BYTE DI ATTRIBUTI DI TESTO (memoria colore)	>	
	<	>	
$0000-07FF	<MEMORIA DI SISTEMA	>	
	<.....	>	

* NOTA: Nel sistema RAM 64-K, la RAM va da $0000 a $FCFF e da $FF40 a $FFFF.

SEZIONE 10



Calcolo di Funzioni Matematiche

Le funzioni che non sono specifiche del BASIC 3.5 possono essere calcolate come segue:

FUNZIONE	EQUIVALENTE BASIC
SECANTE	SEC(X)=1/COS(X)
COSECANTE	CSC(X)=1/SIN(X)
COTANGENTE	COT(X)=1/TAN(X)
SENO INVERTITO	ARCSIN(X)=ATN(X/SQR(-X*X+1))
COSENO INVERTITO	ARCCOS(X)=-ATN(X/SQR (-X*X+1))+ π/2
SECANTE INVERTITA	ARCSEC(X)=ATN(X/SQR(X*X-1))
COSECANTE INVERTITA	ARCCSC(X)=ATN(X/SQR(X*X-1)) +(SGN(X)-1*π/2)
COTANGENTE INVERTITA	ARCOT(X)=ATN(X)+π/2
SENO IPERBOLICO	SINH(X)=(EXP(X)-EXP(-X))/2
COSENO IPERBOLICO	COSH(X)=(EXP(X)=+EXP(-X))/2
TANGENTE IPERBOLICA	TANH(X)=EXP(-X)/(EXP(x)+EXP (-X))*2+1
SECANTE IPERBOLICA	SECH(X)=2/(EXP(X)+EXP(-X))
COSECANTE IPERBOLICA	CSCH(X)=2/(EXP(X)-EXP(-X))
COTANGENTE IPERBOLICA	COTH(X)=EXP(-X)/(EXP(X)-EXP (-X))*2+1
SENO IPERBOLICO INVERTITO	ARCSINH(X)=LOG(X+SQR(X*X+1))
COSENO IPERBOLICO INVERT.	ARCCOSH(X)=LOG(X+SQR(X*X-1))
TANGENTE IPERBOLICA INV.	ARCTANH(X)=LOG((1+X)/(1-X))/2
SECANTE IPERBOLICA INVERT.	ARCSECH(X)=LOG((SQR (-X*X+1)+1)/X)
COSECANTE IPERBOLICA INV.	ARCCSCH(X)=LOG((SGN(X)*SQR (x*x+1)/x))
COTANGENTE IPERBOLICA INV.	ARCCOTH(X)=LOG((X+1)/(x-1))/2

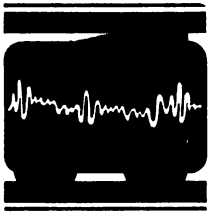


Tabella delle Note Musicali

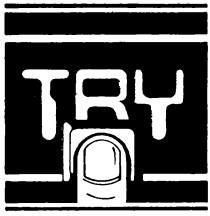
NOTA	VALORI SONORI DEI REGISTRI	FREQUENZA EFFETTIVA (HZ)
A la	7	110
B si	118	123.5
C do	169	130.8
D re	262	146.8
E mi	345	164.7
F fa	383	174.5
G sol	453	195.9
A la	516	220.2
B si	571	246.9
C do	596	261.4
D re	643	293.6
E mi	685	330
F fa	704	349.6
G sol	739	392.5
A la	770	440.4
B si	798	494.9
C do	810	522.7
D re	834	588.7
E mi	854	658
F fa	864	699
G sol	881	782.2
A la	897	880.7
B si	911	989.9
C do	917	1045
D re	929	1177
E mi	939	1316
F fa	944	1398
G sol	953	1575

La tabella precedente mostra i valori sonori dei registri di quattro ottave di note. I valori sonori dei registri sono usati come secondo parametro del comando SOUND. Per utilizzare la prima nota nella tabella (A - valore sonoro dei registri 7) utilizzare il 7 come secondo numero dopo il comando SOUND - SOUND 1,7,30.
Per trovare i valori sonori dei registri per frequenze non comprese nella tabella, utilizzare la formula seguente:

VALORE SONORO DEI REGISTRI = 1024- (111860.781/FREQUENZA)

Sia la tabella dei valori sonori dei registri sia la formula sono applicati a televisori NTSC, il sistema standard utilizzato negli Stati Uniti e in Canada. In paesi che utilizzano il sistema PAL, calcolare i valori sonori dei registri con l'aiuto della formula seguente:
VALORE SONORO DEI REGISTRI = 1024 - (111840.45/FREQUENZA)

SEZIONE 12



Esempi di Programma

```
5 GRAPHIC 3, 1: GRAPHIC 0, 1
10 INPUT "SHOULD I CLEAN UP MY MESS"; A$
20 INPUT "SHOULD I ROTATE"; B$
30 INPUT "SHOULD I VARY MOTION"; C$
40 INPUT "SHOULD I PICK THE START"; D$
50 IF A$="Y" THEN DIM A (3,200)
60 DEF FNA (X) = INT(RND(1) * X)
70 IF D$="Y" THEN X1=FNA(80)+80: X2=FNA(80)+80:
    Y1=FNA(100)+100
75 IF D$="Y" THEN Y2=FNA(100)+100
80 IF D$<>"Y" THEN X1=80: X2=80: Y1=100: Y2=100
90 GRAPHIC 3: FOR L=1 TO 3: COLOR L, FNA(15)+2, FNA(8):
    NEXT
100 IF C1<1 THEN COLOR FNA(3)+1, FNA(15)+2, FNA(8):
    C1=FNA(40)+20
110 IF C2<>0 THEN 140: ELSE XA=FNA(11)-5: XB=FNA(11)-5:
    YA=FNA(13)-6
115 YB=FNA(13)-6
120 IF C$="Y" THEN C2=FNA(10)+5
130 IF B$="Y" THEN XB=-XA: YB=-YA
140 IF C3<1 THEN C=FNA(3)+1: C3=FNA(10)
145 IF A$="Y" THEN DRAW 0, A(0,P), A(1,P) TO A(2,P), A(3,P)
150 X1= X1+ XA: X2= X2+ XB: Y1= Y1+ YA: Y2= Y2+ YB
160 IF X1<0 OR X1>159 THEN XA= -XA: X1= X1+XA
170 IF X2<0 OR X2>159 THEN XB= -XB: X2= X2+XB
180 IF Y1<0 OR Y1>199 THEN YA= -YA: Y1= Y1+YA
190 IF Y2<0 OR Y2>199 THEN YB= -YB: Y2= Y2+YB
200 DRAW C, X1, Y1 TO X2, Y2
210 IF A$="Y" THEN A(0,P)= X1: A(1,P)=Y1: A(2,P)=X2:
    A(3,P)=Y2: P= P+1
215 IF A$="Y" THEN IFP>200THENP=0
220 C1= C1-1: C2= C2-1: C3= C3-1: GOTO 100
```

EFFETTI SONORI

Ululato di Lupo

```
10 VOL7
20 FORL=400TO800STEP20
30 SOUND1,L,3:NEXT
40 FORL=300TO600STEP40
50 SOUND1,L,3:NEXT
60 FORL=600TO300STEP-40
70 SOUND1,L,3:NEXT
```

Computer Maniac

```
10 VOL7
20 FORL=1TO100
30 SOUND1,INT(RND(0)*500)+400,4
40 NEXT
```

Telephone

```
10 VOL7
20 FORL=1TO5
30 FORM=1TO60
40 SOUND1,466,1
50 SOUND1,1020,1
60 NEXT
70 FORZ=1TO2000:NEXT
80 NEXT
```

Busy Signal

```
10 VOL7
20 FORL=1TO15
40 SOUND1,466,20
50 SOUND1,1020,15
80 NEXT
```

Bubbles

```
10 VOL7
20 GRAPHIC1,1
30 FORM=1TO50
40 GOSUB80
50 SOUND1,900-R*20, (YR/2)+50
60 CIRCLE1,X,Y,R,YR
70 NEXT:GRAPHIC0:END
80 X=INT(RND(0)*280)+20
90 Y=INT(RND(0)*160)+20
100 R=INT(RND(0)*40)+5
110 YR=R/1.3
120 RETURN
```

Zap Beam

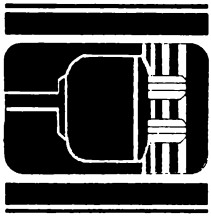
```
10 VOL7
20 FORM=1TO 20
30 FORL=900TO850STEP-10
40 SOUND1,L,1
```

```
50 NEXT
60 FORL=850TO900STEP10
70 SOUND1,L,1
80 NEXT
100 NEXT
```

Music Lines

```
10 VOL7
15 X1=0:Y1=0
20 GRAPHIC1,1
30 GETA$:IFA$=" "THENGRAPHIC0:END
40 GOSUB80
45 FORL=1TODSTEP2
50 SOUND1,X*3,5
55 SOUND2,Y*3,5
60 DRAW1,X,Y
65 X=X+2*DX:Y=Y+2*DY
70 NEXT:GOTO30
80 X=X1:X1=INT(RND(0)*280)+20
90 Y=Y1:Y1=INT(RND(0)*160)+20
100 A=X1-X:B=Y1-Y:D=SQR(A*A+B*B)
110 DX=A/(D+1):DY=B/(D+1)
120 RETURN
```

SEZIONE 13 Introduzione



Interfaccia RS-232

Il Commodore Plus/4 ha un'interfaccia RS-232 incorporata per il collegamento a qualsiasi modem, stampante o altri dispositivi RS232. Per collegare un dispositivo al Plus/4 occorrono un cavo e del software addizionale. Il modem Commodore viene collegato direttamente. L'RS-232 sul Plus/4 è il formato standard RS-232, ma i voltaggi sono a livelli TTL (da 0 a 5V), al contrario del normale RS-232 che possiede un voltaggio da −12 a 12 volt. Il cavo tra il Commodore Plus/4 e il dispositivo RS-232 deve essere provvisto dei necessari trasformatori di voltaggio (la cartuccia di interfaccia del Commodore RS-232 adempie a questa funzione).
È possibile avere accesso al software di interfaccia dell'RS-232 per la programmazione in linguaggio macchina dal BASIC o dal KERNAL. In questa sezione viene spiegato l'uso del RS-232 del BASIC. Per ulteriori informazioni e o per l'uso del linguaggio macchina, consultare il Manuale di Consultazione per il Programmatore del Commodore Plus/4. L'RS-232 a livello BASIC, utilizza i comandi BASIC standard: OPEN, CLOSE, CMD, INPUT#, GET#, PRINT# e la variabile riservata ST. INPUT# e GET# richiamano i dati dal buffer ricevente, mentre PRINT# e CMD inviano i dati al trasmettitore.

Come Aprire il Canale dell'RS-232

È consigliabile aprire un solo canale per volta dell'RS-232; una seconda istruzione OPEN provoca il ripristino del puntatore del buffer di ricezione, causando la perdita di tutti i caratteri presenti nel buffer.
Nel campo nomefile possono essere inviati fino a 4 caratteri. I primi due sono i caratteri del registro di controllo e di comando, gli altri due sono riservati alle opzioni future di sistema. Tramite queste funzioni è possibile selezionare velocità di trasmissione, parità e altre opzioni.

SINTASSI BASIC:

OPEN *lf*,**2,0,** <" > *registro di controllo registro di comando* <" >

ESEMPIO: OPEN 2,2,0, CHR$(5) + CHR$(15)

lf - Normale i.d. di file logico (1-255). Se lf è > 127, il ritorno carrello è seguito da un avanzamento di riga.

registro di controllo - Carattere di un byte (vedi figura 1)
(richiesto per specificare la velocità di trasmissione)

registro di comando - Carattere di un byte (vedi figura 2)

Come Ricevere i dati dal Canale dell'RS-232

Quando i dati vengono ricevuti, la capacità del buffer di ricezione del Commodore Plus/4 è di 127 caratteri prima di un overflow di buffer. Questo viene indicato dalla parola di stato dell'RS-232 (ST in BASIC). Tutti i caratteri ricevuti quando il buffer è pieno vengono persi. Naturalmente è consigliabile tenere il buffer più libero possibile. Se si vogliono ricevere ad alta velocità i dati dell'RS-232 si consiglia l'uso del linguaggio macchina, in quanto la velocità del BASIC è molto ridotta.

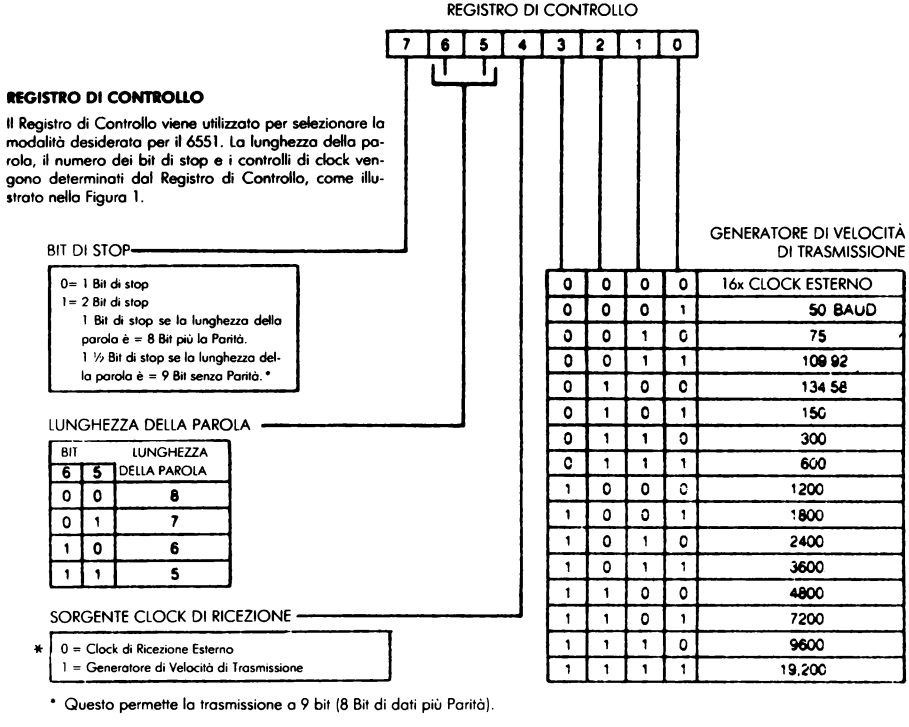


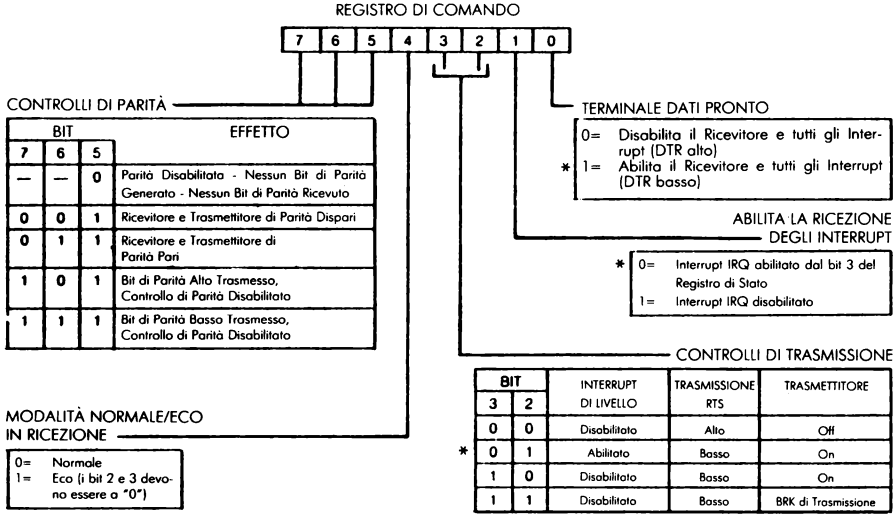
Figura 1. Formato del Registro di Controllo

SINTASSI BASIC

Consigliato: GET #If, <variabile di stringa >
NON Consigliato: INPUT #If, <variabili >

REGISTRO DI CONTROLLO

Il Registro di Comando viene usato per controllare le funzioni specifiche di Trasmissione/Ricezione, come mostrato nella Figura 2.



	7	6	5	4	3	2	1	0
RIPRISTINO HARDWARE	0	0	0	0	0	0	0	0
RIPRISTINO SOFTWARE	—	—	—	0	0	0	0	0

* Questi bit devono essere impostati ai valori dati.

Figura 2. Registro di Comando

NOTE

Se la lunghezza della parola è inferiore a 8 bit, viene assegnato un valore di zero a tutti i bit non utilizzati.

Se il comando GET # non trova alcun dato nel buffer, viene ritornato il carattere "" (carattere nullo).

Se viene utilizzato INPUT #, il sistema si arresta finché non viene ricevuto un carattere significativo seguito da un ritorno carrello. Perciò, se la linea CTS o DSR scompare durante l'introduzione (INPUT#) del carattere, il sistema si blocca sullo stato di "solo RIPRISTINO". Per questo motivo non sono consigliabili le routine INPUT # e CHRIN.

Come Inviare Dati ad un Canale RS-232

SINTASSI BASIC:

CMD *lf* - agisce come nelle specifiche BASIC

PRINT # *lf*, *<variabili>*

Come Chiudere un Canale di Dati RS-232

La chiusura di un file RS-232 provoca la cancellazione di tutti i dati nel buffer al momento dell'esecuzione, arresta tutte le trasmissioni e ricezioni dell'RS-232 e imposta le linee RTS e Sout nella condizione "ALTA".

SINTASSI BASIC:

CLOSE *lf*

Tabella 1. LINEE DI PORTA DELL'RS-232

PIN ID	DESCRIZIONE	EIA	ABV	OUT
C	DATI RICEVUTI	(BB)	Sin	IN
D	RICHIESTA DI INVIO	(CA)	RTS	OUT
E	TERMINALE DATI PRONTO	(CD)	DTR	OUT
F	INDICATORE DI CHIAMATA	(CE)	RI	IN
H	SEGNALE DI LINEA RICEVUTO	(CF)	DCD	IN
J	NON ASSEGNATO	( )	XXX	IN
K	LIBERO PER TRASMETTERE	(CB)	CTS	IN
L	MODEM PRONTO	(CC)	DSR	IN
B	DATI RICEVUTI	(BB)	Sin	IN
M	DATI TRASMESSI	(BA)	Sout	OUT
A	TERRA DI PROTEZIONE	(AA)	GND	
N	TERRA DI SEGNALE	(AB)	GND	

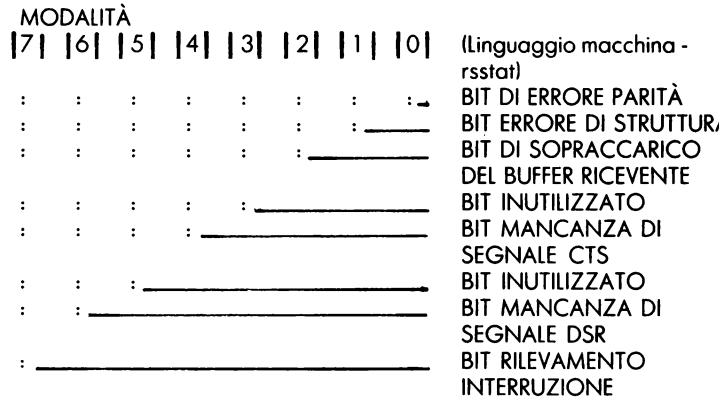


Figura A-3. Registro di Stato dell'RS-232

NOTE

Se Bit =0, non è stato trovato alcun errore.

Il registro di stato dell'RS-232 può essere letto dal BASIC con l'uso della variabile ST.

Se la ST viene letta dal BASIC o utilizzando la routine READST del KERNAL, la parola di stato dell'RS-232 viene azzerata all'uscita. Se sono necessarie diverse applicazioni della parola di STATO, la ST dovrà essere assegnata ad una altra variabile, cioè:

```
SR=ST: REM ASSEGNA ST A SR
```

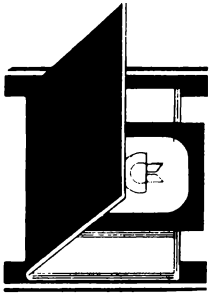
Lo stato dell'RS-232 viene letto (e azzerato) solo quando il canale dell'RS-232 è l'ultimo I/O esterno utilizzato.

ESEMPIO DI PROGRAMMA IN BASIC

Questo programma apre il canale di telecomunicazioni per dare la possibilità di utilizzare un modem.

```
100 OPEN 5, 2, 2, CHR$(22)+CHR$(5): REM ALLOCATE BUFFER
      AND OPEN CHANNEL
110 DIM F%(255), T%(255)
120 FOR J= 32 TO 64: T%(J)=J:NEXT
130 T%(13)=13: T%(20)=8: RV=18: CT=0
220 FOR J=65 TO 90: K=J+32: T%(J)=K: NEXT
230 FOR J=91 TO 95: T%(J)=J: NEXT
240 FOR J=193 TO 218: K=J-128: T%(J)=K: NEXT
250 T%(146)=16: T%(133)=16
255 T%(137)=3: T%(134)=17: T%(138)=19
260 FOR J=0 TO 255
270 K=T%(J)
280 IF K<>0 THEN F%(K)=J: F%(K+128)=J
290 NEXT
300 PRINT" "CHR$(147)
310 GET#5, A$
320 IF A$="" THEN 360
330 PRINT" "CHR$(157);CHR$(F%(ASC(A$)));
340 IF F%(ASC(A$))=34 THEN PRINT CHR$(27)"O";
350 GOTO 310
360 PRINT CHR$(RV)" "CHR$(157);CHR$(146);: GET A$
370 IF A$<>"" THEN PRINT#5,CHR$(T%(ASC(A$)));
380 CT=CT+1
390 IF CT=8 THEN CT=0: RV=164-RV
410 GOTO 310
```

SEZIONE 14



Altri Titoli Disponibili

Le liste seguenti riportano un elenco dei libri disponibili per computer e relativi alla programmazione. Viene riportato per primo il titolo del libro, seguito dall'autore e dall'editore.

Libri Commodore

VIC 20 Programmer's Reference Guide
Commodore 64 Programmer's Reference Guide
Commodore Plus/4 Programmer's Reference Guide
Mastering Your VIC 20
Four VIC 20 Computer Books:
- VIC Revealed, Nick Hampshire
- VIC Games, Nick Hampshire
- VIC Graphics, Nick Hampshire
- Stimulating Simulations for the VIC, C.W. Engel

Introduction to BASIC, Part 1 and 2, Andrew Colin
Commodore Software Encyclopedia, Third Edition

Programmazione in BASIC

Armchair BASIC: An Absolute Beginner's Guide to Programming in BASIC, Fox & Fox, Osborne/McGraw-Hill
BASIC Handbook, Second Edition, Lien, Compusoft
Basic Commodore 64 BASIC, Coan, Hayden
Elementary BASIC, Ledgard & Singer, SRA
How to Build a Program, Emmerichs, Dilithium Press
Istant Freeze-Dried Computer Programming in BASIC, Brown
My Computer Likes Me When I Speak in BASIC, Albrecht, Dilithium Press
Nailing Jelly to a Tree, Willis & Danley, Dilithium Press
The Programmer's Book of Rules, Ledin & Ledin, Lifetime Learning Publishers
Technical BASIC, Kassab, Prentice-Hall

Programmazione in Linguaggio Macchina

Machine Language for Beginners, Mansfield, COMPUTE! Books
Programming the 6502, Zaks, Sybex
6502 Assembly Language Programming, Leventhal, Osborne/Mc Graw-Hill
6502 Micro Chart, Micro Logic Corp
6502 Software Design, Scanlon, Sams
The 6502 Software Gourmet Guide & Cookbook, Findlay, Hayden

commodore COMPUTER

Via F.lli Gracchi, 48 - 20092 Cinisello Balsamo (Milano)
Tel. 02/618321