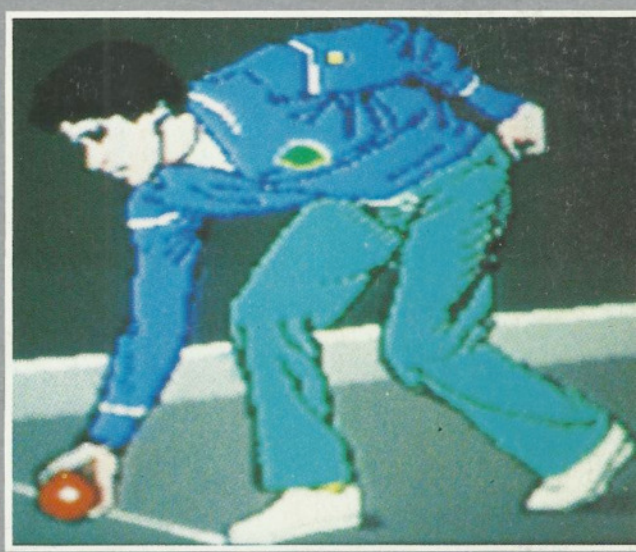
L. 8.000

C=64 C=128 C=16

VIDEOTECA COMPUTER

n 11

GRAFICA 64: L'ALTA RISOLUZIONE



C. 16 I COMANDI DEL BASIC

COMMODORE 16 • CALENDARIO • TALPE • SLOT MACHINE • ASTRONOMIA • READNIGHT

COMMODORE 64 - COMMODORE 128

• BOWLS (BOCCE) • NIKI • OCCHIO ALLA PAROLA • INCURSOR • CALENDARIO • ESERCIZI DI GRAFICA

EV EDITORIALE VIDEO

Dopo i primi dieci numeri VIDEOTECA COMPUTER, come già annunciato, cambia un po' la sua pelle. Le idee suggerite dai lettori e l'esperienza fatta in mesi e mesi di duro ma entusiasmante lavoro vanno prendendo corpo sempre di più. Innanzi tutto i programmi. Le edicole sono sommerse da pubblicazioni con cassetta che ripropongono programmi già noti, tutti stranieri, con titoli sempre più a sorpresa. Si tratta di veri e propri trabocchetti per tutti coloro che si illudono di acquistare programmi originali, magari creati su gusti, desideri ed esigenze di utilizzatori italiani del 1986, e non americani di uno o due anni fa.

VIDEOTECA COMPUTER ha prodotto i suoi sforzi per assicurare PROGRAMMI ORIGINALI di qualità sempre più elevata, per soddisfare quindi i gusti meno facili e più esigenti. Da questo numero la famiglia del nostro mensile si è allargata, abbiamo aperto le porte anche ai possessori di COMMODORE 16 e PLUS/4. La rivista presenterà sempre, quindi, articoli utili e pratici per i C.68, C.128, C.16 e PLUS/4.

La cassetta, invece, presenterà due facce diverse, una per C.64 e C.128, l'altra per C.16 e PLUS/4.

Produrre programmi originali per tutte queste macchine impone un impegno incredibile, probabilmente il massimo tra i periodici con cassetta di software. Di questo grande impegno noi siamo orgogliosi e ringraziamo ancora i lettori che ci hanno dato la forza e l'entusiasmo per sostenerlo.

VIDEOTECA COMPUTER - mensile di cultura informatica - anno IIIn. 11 - marzo 1986 - Direzione, redazione e amministrazione: Editoriale VIDEO - Via Castelvetro 9 - Milano - Direttore: Antonio Lucarella - Coordinamento tecnico: Roberto Treppiedi - Hanno collaborato: Franco Longoni, Fabrizio Iotti, Maria Russino, Aldo Campanozzi, Cinzia Agiman, Alessandro Moro, Roberta Di Pietro, Alessandro Vallone, Arnaldo Restelli, Dino Ticli, Giovanna Zampella - In copertina: disegno di Aldo Campanozzi - Stampa: Color Graf Milano - Fotocomposizione: ERREGI Milano - reg. del Trib. di Milano n. 517 del 10/11/84 - Stampato in Italia

INTRODUZIONE AI COMANDI DEL BASIC

La descrizione dei comandi basic del Commodore 16 consiste in una panoramica su tutte le possibilità offerte dal BASIC 3.5 con cui lavora questo microcomputer.
In questo lavoro, abbiamo puntato a descrivere in modo più accurato quei comandi che, non appartenendo al bagaglio di comandi del Commodore 64 o del Vic 20 non trovano facilmente una illustrazione chiara e completa.
Distingueremo quindi tali comandi attraverso un asterisco prima degli stessi non tralasciando peraltro tutti gli altri anche perché non tutti i lettori hanno avuto una qualche esperienza sugli altri due computer.
Al termine della descrizione, troverete, dopo il simbolo [L] il numero dei listati ove appare l'istruzione di cui abbiamo parlato.
Abbiamo di proposito tralasciato la descrizione dei comandi del MONITOR residente nel 16, perché di questi si parla in un apposito capitolo inserito in questo libro.
Va infine fatto notare che, inseriti accanto alle descrizioni, vi saranno dei brevissimi listati che serviranno a solo titolo di esempio, per illustrare in maniera più chiara tutto ciò che viene spiegato, proprio perché, soprattutto in questo campo, si dimostra sempre più utile la pratica sulla tastiera che non la sola teoria.
È proprio per questo che vi invitiamo, durante la prima consultazione di questa parte del testo, a tenere acceso il computer e a provare immediatamente i listati proposti, facendo seguire a tutto ciò una serie di prove a vostro piacimento, per "testare" ciò che avete imparato.
Una spiegazione dei simboli che troverete nei listati, si trova in appendice. Non scoraggiatevi quindi di fronte a un simbolo che in un primo momento può apparirvi un po' strano; sicuramente o le REM del listato, o in appendice, troverete spiegazione di tutto.

AUTO
Permette di avere impostato, in maniera automatica, il numero di linea seguente, secondo un intervallo da noi precedentemente scelto.
La sintassi è:
AUTO <N. LINEE DI INTERVALLO>
Se infatti noi, prima di procedere alla digitazione di un programma, impostiamo ad esempio AUTO 10 <RETURN>, nulla apparirà di particolare sullo schermo, ma dopo avere digitato la linea 10 e avere battuto <RETURN>, apparirà già impostato il numero di linea successivo (20) e il cursore si sposterà nella posizione giusta per proseguire.
Se mi interessa interrompere questa funzione, posso agire in 2

modi: 1 - digitare AUTO senza farlo seguire da alcun numero; 2 - battere <RETURN> 2 volte e fare listare il programma.
ESEMPIO DI APPLICAZIONE 1
AUTO 10 <RET>
10 PRINT"CIAO" <RET>
20 PRINT"COMMODORE 16":REM: IL NUMERO DI LINEA DA QUESTA RIGA IN POI È IMPOSTATO IN MANIERA AUTOMATICA <RET>
30 ecc...
ESEMPIO DI APPLICAZIONE 2
AUTO 125 <RET>
100 PRINT"CIAO" <RET>
225 PRINT"MAMMA" <RET>
350 PRINT"PAPÀ":END: <RET>
Come potete vedere le linee si alternano di 125 in 125

BACKUP
Questo comando permette la copia fisica del dischetto dal drive n. 0 al drive n. 1.
Ciò avviene anche se non è stato formattato in precedenza il dischetto "destinazione".
Sintassi:
BACKUP D0 TO D1, ON U9
La dicitura, ON U9 (sulla unità n. 9) può essere omessa la maggior parte delle volte.
Il comando ha senso solo se si possiedono 2 disk drive.
Massima attenzione va posta al fatto che sul dischetto destinazione viene cancellato tutto ciò che era presente in precedenza; tutti i dati vanno quindi persi per far posto ai nuovi.

COLLECT
Si tratta di un'istruzione di uso molto frequente e utile nel caso si sia in possesso di uno o due disk drive.
Essa, infatti, va a verificare sul dischetto la presenza di file mal chiusi o "spazi" utilizzabili e li rende disponibili all'utente, cancellando ciò che vi è di superfluo e inutilizzabile.
Ci ricorda da vicino la funzione VALIDATE, che però non sostituisce.
Sintassi:
COLLECT D0
Ricordarsi di lasciare uno spazio dopo la parola COLLECT. Il numero dopo D può essere 0 o 1 a seconda del drive scelto.

CONT
Questo comando permette di riprendere l'esecuzione del programma in memoria dal punto in cui è stato interrotto tramite il tasto STOP o i comandi END o STOP da BASIC.
Negli altri casi non produrrà altro effetto se non quello di causare l'errore ?CAN'T CONTINUE ERROR.
Per un studio più particolareggiato della gestione degli errori, vedere le istruzioni TRAP e RESUME.

COPY
Sintassi:
COPY Dn,"nome file"TO Dnn"nome file 2", ONmm
dove n è il numero del drive sorgente
nn è il numero del drive destinazione
mm è il numero dell'unità
nome file è il nome del file da copiare
nome file 2 è il nome da dare al file copiato.
Come si può intuire questo comando permette di copiare un singolo file da un dischetto a un altro o sullo stesso dischetto cambiandone o meno il nome.
Se non vengono indicati i nomi dei file, il comando eseguirà una copia di TUTTI i file presenti sul disco sorgente.**
**es. COPY D0 TO D1
Es COPY D0,"PIPPO"TOD1,"PAPPA",ONU8 copia un singolo file di nome PIPPO sullo stesso disco o su un altro con il nome PAPPA.

DELETE
Si tratta di un comando accettato solo in maniera diretta (cioè non da programma).
È uno dei più utili ausili alla programmazione che colma una grossa lacuna del C64 e VIC 20.
Esso permette di cancellare (to delete) le righe BASIC indicate in maniera istantanea, senza particolari operazioni lunghe e complicate.
Sintassi:
DELETE numero prima linea-numero ultima linea.
Esempi:
DELETE 10-70 Cancella tutte le linee dalla numero 10 alla 70
DELETE -70 Cancella TUTTE le linee fino alla 70 INCLUSA.
DELETE 10- CANCELLA TUTTE le linee dalla 10 IN POI.

DIRECTORY
Anche questa funzione rappresenta sicuramente un notevolissimo passo in avanti rispetto ai predecessori del C16.
Essa infatti permette di richiamare da disco l'indice dello stesso SENZA che esso si vada a sovrapporre al programma residente in memoria.
Questa decisiva particolarità permette una gestione ancor più professionale del dischetto in quanto anche durante l'elaborazione dei programmi possiamo cercare sulla DIRECTORY del dischetto il nome di qualche altro file che al momento non ci ricordiamo.
Sintassi:
DIRECTORY Dn,Unn,(nome file)
Se si tralasciano tutti i parametri verrà mostrata sullo schermo l'intera directory.
Nel caso si indichi un nome del programma, esso visualizzerà tutti i file con quel nome.
Molto utili sono anche i seguenti comandi:

DIRECTORY "lo*" e DIRECTORY "?fgh"
Il primo permetterà di visualizzare tutti i file che iniziano per le lettere "lo" e il secondo permette di vedere quelli le cui seconda, terza e quarta lettera sono "fgh", tralasciando la prima.
Lo scorrimento può essere rallentato attraverso la pressione del tasto commodore e la sua interruzione temporanea con la pressione simultanea di CTRL e S.
Poiché la DIRECTORY non va a sovrapporsi ai programmi essa non può essere listata su carta altro che con il vecchio sistema:
LOAD "$" ,8 <RETURN>
OPEN4,4 <RETURN>
CMD 4 : LIST : PRINT#3 <RETURN>
CLOSE 4

DLOAD
Si tratta di un comando caratteristico dei computer approntati per l'utilizzo del disk drive.
Esso permette di caricare da disco un qualsiasi programma, il cui nome venga specificato subito di seguito.
Sintassi:
DLOAD "nome programma", Dn , Unn
dove n e nn sono il numero del drive e dell'unità da cui si vuole caricare il programma.
Tale comando può essere inserito all'interno di un programma e, in particolare, si può utilizzare una variabile stringa al posto del nome programma.
È però necessario tenere presente che, nel caso tale stringa risulti vuota, viene generato un messaggio di errore.
L'utilizzo di tale comando è perfettamente simile a quello precedente, con la sola eccezione che, in questo caso, viene salvato sul disco in questione, il programma attualmente in memoria.
Sintassi:
DSAVE "nome programma" , Dn , Unn.

SAVE
Sintassi:
SAVE "nome del programma",n,nn
dove:
n è il device della periferica su cui si vuole salvare il programma
nn è un "flag" di fine cassetta che permette al C16 di riconoscere se il programma appena registrato è o meno l'ultimo della cassetta in questione.
Con il comando SAVE si trasferisce tutto il programma che risiede in memoria sul supporto magnetico desiderato.
Se esso non è seguito da alcun nome l'operazione può essere eseguita solo su cassetta.
Al posto del nome, può essere sostituita una variabile stringa, ad es.:
SAVE A$

SCRATCH
Sintassi:

SCRATCH"nf",Dn,unn
dove:
nf è il nome del file che si desidera cancellare
n è il numero del drive (Ø o 1)
nn è il numero dell'unità.
Questo comando serve a cancellare da un disco un file che si desidera non sia più presente su di esso.
Il C16 si assicurerà che voi non abbiate commesso alcun errore chiedendovi se siete sicuri dell'operazione compiuta, attraverso la domanda:
ARE YOU SURE?
a cui Voi risponderete Y se non avete alcun dubbio sulla correttezza dei dati impostati.

VERIFY
Il comando in esame consente un confronto tra il programma in memoria e quello che si vuole analizzare sul supporto desiderato. Questa fa sì che tale comando venga in particolare maniera utilizzato per "verificare" che le operazioni di registrazione di un programma siano state effettuate in modo corretto.
Solitamente viene utilizzato dopo un SAVE e la sua sintassi è del tutto identica.
Alcuni messaggi su video vi informeranno della corretta conclusione dell'operazione di confronto.

BOX
Sintassi:
BOXn,a1,b1,a2,b2,an,ri
dove:
a1 e b1 sono le coordinate di un vertice
a2 e b2 sono quelle del vertice opposto
an è l'angolazione, in senso orario di cui si vuole far ruotare il rettangolo
ri se è = a 1 riempie la figura del valore scelto, se = a Ø no.
Come è facilmente intuibile, questa istruzione permette di disegnare sullo schermo un rettangolo (box) di qualsiasi dimensione.
Al termine dell'operazione di pixel cursor si troverà nel punto di coordinate a2 e b2.

CHAR
Sintassi:
CHAR n,x,y,"stringa",f
dove:
n = numero del colore
x e y colonna(Ø-39) e riga(Ø-24) ove posizionare il carattere
stringa = messaggio da stampare nella posizione predetta f se = a Ø stampa in positivo, se = 1 stampa in reverse.
Rimandiamo alla sezione dedicata alla grafica ulteriori informazioni.

CIRCLE
Sintassi:

CIRCLE c,a,b,xr,yr,ap,aa,an,ng
dove:
c è il numero del colore
a e b sono le coordinate del centro
xr e yr sono le misure del raggio
ap e aa sono gli angoli di partenza e di arrivo
an è l'angolo di rotazione in senso orario espresso in gradi
ng è il numero dei gradi tra i segmenti.
Rimandiamo alla sezione dedicata alla grafica per una trattazione approfondita del comando.

CLOSE
Sintassi:
CLOSE nf
dove:
nf è il numero del file da chiudere.
È il comando che completa il comando OPEN.
Esso chiude il file indicato da nf.
La mancata chiusura di un file può provocare spiacevoli inconvenienti.

CLR
Il comando cancella tutte le variabili in memoria, lasciando intatto tutto il resto.
Esso viene eseguito automaticamente dopo un RUN o un NEW o nel caso si apporti una correzione di qualsiasi tipo in un programma.

CMD
Sintassi:
CMD numero del file
Il comando permette di inviare dei dati che normalmente verrebbero inviati allo schermo, sulla periferica specificata nel comando OPEN corrispondente.
Esempio:
OPEN3,4
CMD3:LIST:PRINT#3
CLOSE3
Il listato segnalato permette di inviare alla stampante l'output di un listato di programma in memoria.

COLOR
Sintassi:
COLOR de,nc,lu
dove:
de è la destinazione del comando
nc è il numero del colore (1-16)
lu è il livello di luminosità del colore scelto (Ø-7) ove Ø è la luminosità minima e 7 è quella normale.
Rimandiamo alla sezione dedicata alla grafica per una trattazione più particolareggiata del comando.

DATA
DATA lista dati separati da virgole
Si tratta del comando abbinato al comando READ.
Accanto a esso sono elencate le costanti stringa, numeriche che devono essere lette e utilizzate nel programma.
Attraverso il comando RESTORE si potrà poi indirizzare tale lettura (vedi).
L'assenza di dati tra due virgole viene interpretato come uno 0 o una stringa nulla.

DEFFN
DEFFN nome (variabile) = espressione
Questa funzione consente (come dice la stessa sintassi DEFinizione di FuNzione) di definire in una sola istruzione una funzione, i cui valori possono essere utilizzati in altra parte del programma.
Ne risulta un notevole risparmio di memoria (il C16 in fondo non ne ha moltissima).
Esempio:
10 DEF FN(X) = X↑2*2
20 PRINT FN(3)
RUN
18
Sostituisce cioè alla linea 20 il valore di 3 all'interno della funzione e ne calcola il risultato stampandolo.

DIM
DIM variabile(indice)
L'istruzione DIM permette di DIMensionare una matrice di una o più dimensioni.
Esempio:
DIM A$ (10) 11 elementi
DIM S$ (4,4,4) 125 elementi
Moltiplicando fra loro gli elementi si ottiene lo spazio di memoria occupato dalle variabili in termini di numero di esse.
Facendo tale operazione fate bene attenzione a considerare lo 0, cioè una matrice dimensionata così:
DIM (4)
è costituita da 5 elementi, non da 4!!!

DO/LOOP/WHILE/UNTIL/EXIT
Sintassi:
DO UNTIL argomento booleano-istruzioni-LOOP UNTIL argomento booleano
oppure
DO WHILE argomento booleano-istruzioni-LOOP WHILE argomento booleano
All'interno del ciclo DO vi può essere un'istruzione EXIT la quale ha il compito di portare l'esecuzione del programma alla prima istruzione dopo il ciclo (cioè dopo il LOOP).
L'argomento booleano può essere ad esempio S=6 oppure M<=78, ecc.
Il ciclo si attiene alle regole dei cicli FOR - NEXT.

Con il parametro UNTIL il ciclo si ripete fino a quando l'argomento booleano diventa vero; viceversa se viene utilizzato il WHILE, il ciclo si ripete fino a quando l'argomento booleano diventa falso.

DRAW
Sintassi:
DRAW nc,a1,b1 TO a2,b2
dove:
nc è il numero del colore da utilizzare.
a1 e b1 sono le coordinate del punto di partenza.
a2 e b2 sono le coordinate del punto di arrivo.
Il comando, che appartiene al tool grafico, permette di tracciare una linea tra i punti specificati.
Può essere utilizzato anche con parametri relativi, ad esempio:
DRAW 1, 10,10 TO 20,20 e
DRAW 1, 10,10 TO +10,+10
sono istruzioni che hanno il medesimo effetto finale di tracciare una linea tra i punti le cui coordinate sono quelle espresse nel primo dei due esempi.
Una trattazione più approfondita si trova in questo volume nella parte relativa alla grafica.

END
Se il programma incontra questa istruzione si ferma e può essere fatto ripartire dall'istruzione successiva mediante un comando CONT.

FOR..TO..STEP
Sintassi:
FORvar = vinTOvfiSTEPinc
dove:
var è la variabile su cui si opera
vin è il valore iniziale della variabile
vfi è il valore finale di tale variabile
inc è l'incremento del contatore relativo a questo ciclo.
Il comando può essere solamente utilizzato in combinazione con NEXT in quanto è solo così che si può chiudere il ciclo senza incappare in un errore.
È la maniera più semplice affinché un computer compia ciclicamente delle operazioni e all'interno di esso può essere rinchiuso un secondo, un terzo, ecc. ciclo seguendo le classiche regole del BASIC.
Esempio:
10 FOR X = 1 TO 100
20 PRINT X
30 NEXT X
Stamperà tutti i numeri da 1 a 100
10 FOR X = 100 TO 1 STEP -1
20 PRINT X
30 NEXT X
Stamperà tutti i numeri compresi tra 100 e 1. Ecco quindi l'utilità del comando STEP all'interno di questo ciclo.

GET
Sintassi:
GET 1v
dove:
1v è la lista di variabili numeriche o stringa.
Essa permette di acquisire dalla tastiera un carattere alla volta. Nel caso di mancata pressione il carattere sarà considerato nullo e il programma proseguirà l'esecuzione.
Attenzione però perché se il programma attende una variabile numerica e riceve una stringa o viceversa, si arresta con un messaggio di errore.
Tipico esempio è:
10 GET A$:IF A$ <> "S" THEN 10
La linea consente l'attesa della pressione del tasto S prima di proseguire.

GETKEY
Sintassi:
GETKEY lista variabili
A differenza dell'istruzione di GET essa attende la pressione di un tasto e ne associa quel valore alla variabile.
10 GETKEY A$
attende la pressione di un tasto per proseguire.

GET#
Sintassi:
GET# numero file, lista variabili
Essa ha la stessa funzione del get, dove, in questo caso, il computer accetterà i caratteri uno alla volta.
Il comando non può essere dato in maniera diretta, ma solo da programma.

GOSUB
Sintassi:
GOSUB numero di linea
Consente di saltare alla linea di programma specificata, eseguire le istruzioni contenute fino all'istruzione RETURN, e riprendere l'esecuzione del programma dalla linea successiva a quella ove è collocata.
Lavora quindi in coppia con l'istruzione RETURN e non può essere utilizzata singolarmente.
Esempio:
10 GOSUB 1500
.....
1500 PRINT "PIPPO" :RETURN

GOTO
Sintassi:
GOTO numero di linea
Salta l'esecuzione del programma alla linea specificata a lato.
Può essere utilizzata in maniera diretta per fare iniziare l'esecuzione di un programma in una determinata linea.

GRAPHIC
Sintassi:
GRAPHIC modo, oc
dove:
modo è il modo grafico scelto (vedi elenco e parte relativa alla grafica)
oc se è = a 1 cancella tutto ciò che era presente prima sullo schermo.

MODO	DESCRIZIONE
0	testo
1	alta risoluzione
2	alta risoluzione + testo
3	multicolore
4	multicolore + testo

Per rendere disponibili al BASIC i 10 K di memoria occupati dalla gestione grafica bisogna utilizzare il comando:
GRAPHIC CLR

IF...THEN...ELSE
Sintassi:
IF espressione THEN istruzione ELSE istruzione
La traduzione di questo comando chiarisce molti dubbi:
SE (IF) una qualsiasi espressione è vera ALLORA (THEN) esegui la seguente istruzione ALTRIMENTI (ELSE) esegui questa altra istruzione.
Se si desidera, si può tralasciare la istruzione ELSE senza peraltro compromettere il funzionamento del programma.
Esempio:
10 IF A = 0 THEN PRINT "MAMMA" ELSE PRINT "PAPÀ"
va a controllare il valore di A: se esso è = a 0 stamperà la parola MAMMA altrimenti stamperà PAPÀ.

1 - continua

L'ALTA RISOLUZIONE

Nella puntata precedente di questa serie di articoli abbiamo visto com'è organizzata la grafica del C64 nel modo testo, ossia quando vengono visualizzati solo dei caratteri, sia alfanumerici che grafici. Abbiamo anche visto come sia possibile ridefinire dei caratteri per creare disegni apparentemente in alta risoluzione. Questo secondo articolo invece sarà interamente dedicato all'alta risoluzione vera e propria, comunemente abbreviata in HI-RES (high resolution). C'è un bit tra i registri del VIC II che controlla se siamo in modo testo o in alta risoluzione: è il bit 5 della locazione 53265: se è settato siamo in alta risoluzione, altrimenti siamo in modo testo. Come si fa a settarlo? Ma nel solito modo! Basta una: POKE53265,PEEK(53265)OR32 e per resettarlo POKE53265,PEEK(53265)AND223.

Il C64 dispone di due modi di alta risoluzione: il primo, generalmente meno usato, ma utilissimo per disegni precisi è il modo normale, a due colori. Esso ci garantisce una risoluzione notevole, esattamente di 320 × 200 pixel. Sapete che cos'è un pixel? È il più piccolo puntino che possa essere rappresentato sullo schermo: se volete avere una chiara idea della sua dimensione scrivete sul computer una -i- minuscola: il puntino è esattamente un pixel. Non è tale invece il punto fermo, che è composto di ben quattro pixel disposti in quadrato.

Disporre di una risoluzione di 320 × 200 pixel significa che lo schermo è composto di 200 righe ciascuna delle quali è composta di 320 pixel affiancati e noi possiamo attribuire a ciascun pixel due colori per ogni gruppo di otto per otto. Infatti ad ogni quadrato di lato otto pixel corrisponde esattamente un carattere nel modo testo: potete verificare questo fatto semplicemente pensando che nel modo testo ogni riga è composta di 40 caratteri. Se moltiplicate i 40 caratteri per gli otto pixel di cui sono composti in senso orizzontale ottenete esattamente 320 pixel. Viceversa ogni colonna è composta da 25 caratteri nel modo testo. Se moltiplicate i 25 caratteri per gli otto pixel che li compongono in senso verticale ottenete esattamente 200 pixel. È importantissima questa corrispondenza tra caratteri e gruppi di pixel, innanzitutto per la gestione dei colori. Dicevo infatti che ad ogni gruppo di otto per otto pixel è possibile attribuire due colori. Voi sapete, dalle lezioni precedenti, che per definire un colore sono necessari 4 bit, infatti i colori disponibili sono sedici, numerati da 0 a 15, e 4 bit rappresentano esattamente un numero tra 0 e 15. Ciò significa che ogni byte può memorizzare due colori il primo nei quattro bit più significativi (4-7) e il secondo nei quattro bit meno significativi (0-3).

È esattamente quanto avviene con la grafica in alta risoluzione, dove i bit più bassi definiscono il colore di fondo e i bit più alti il

colore del disegno. Ma dove sono memorizzati questi bytes contenenti i colori? Ma nella memoria di schermo naturalmente! A ciascun byte della memoria di schermo corrisponde un quadrato di otto pixel come nel modo testo, solo che in alta risoluzione non contiene il codice di schermo del carattere rappresentato, bensì appunto i colori utilizzati in quel quadrato. Ovviamente la memoria di schermo copre esattamente l'intero schermo anche se utilizzata come memoria colore dell'alta risoluzione: quello che desidero sia ben chiaro è che ciascun byte della memoria di schermo contiene informazioni relative solo e solamente a una certa posizione dello schermo, indipendentemente dalla grafica utilizzata. Viene spontaneo pensare: ma allora se vogliamo una schermata grafica ci tocca rovinare la schermata in modo testo! No! Non è necessario: come avete imparato dalla prima lezione di grafica del C64 è possibile spostare la memoria di schermo. Conviene sicuramente lasciare la memoria per il modo testo nel suo posto normale, dalla locazione 1024 alla locazione 2023, mentre potete scegliere un'altra zona di memoria per la memoria di schermo per il modo HI-RES.
Ora che abbiamo visto come si controllano i colori in alta risoluzione dobbiamo vedere come sia possibile gestire i singoli pixel. È facile intuire che per determinare lo stato di un pixel è sufficiente un solo bit: esso sarà 1 se il bit ha il colore carattere e sarà 0 se il pixel ha il colore di fondo. Ovviamente un byte conterrà le informazioni relative a 8 pixel, e siccome i pixel sono $320 \times 200 = 64000$ ci vorranno $64000:8 = 8000$ bytes per coprire l'intera area della memoria di schermo. Voi sapete che il VIC II, il formidabile circuito integrato che gestisce la grafica del C64, può accedere a soli 16K ovvero a 16384 bytes di memoria contemporaneamente, e tali blocchi sono detti banchi di memoria. Dal momento che l'intera memoria di schermo può essere contenuta due volte in 16K, con un piccolo avanzo, sono possibili due diversi posizionamenti: dal primo byte del banco fino al byte 7999 oppure dal byte 8192 al byte 16191. È possibile effettuare tale scelta modificando il bit 3 della locazione 53272: se è 0 otteniamo la prima configurazione, se è 1 otteniamo la seconda. Attenzione che qualora sia selezionato il banco 0, come all'accensione del computer, non è possibile utilizzare la prima configurazione perchè la memoria di schermo si sovrapporrebbe alle locazioni di memoria dove il microprocessore registra dei dati importanti.
Se non si vuolo modificare il banco cui punta il VIC II è perciò necessario settare il bit 3 della locazione 53272. Se avete seguito le precedenti lezioni lo saprete fare anche a occhi chiusi, comunque è opportuno ripetere l'istruzione necessaria:

```
POKE53272,PEEK(53272)OR8
```

Viceversa per resettare tale bit si farà:

```
POKE53272,PEEK(53272)AND247
```

Generalmente, nel caso di programmi piuttosto semplici, si usa proprio questa configurazione, dove la memoria di schermo di alta risoluzione va dalla locazione 8192 alla locazione 16191: é questo il caso anche del programma GRAFICA 2 che vedremo tra po-

co. Un'avvertenza: se utilizzate questo tipo di configurazione ovviamente il programma basic non puó essere troppo lungo pena la sovrapposizione del medesimo alla memoria di schermo. È perciò necessario limitarne la lunghezza con due POKE: POKE52, 32 e POKE56, 32 seguite da un CLR.
Se ricordate abbiamo usato la stessa tecnica per riservare la memoria per i caratteri programmabili. Ovviamente se il programma basic richiede piú spazio sará necessario trovare un'altro posto per la memoria di schermo, ma di questo ne parleremo dopo.
Manca ancora la cosa più importante: che corrispondenza c'é tra gli 8000 bytes della memoria di schermo e i 64000 pixel? Ad esempio quale byte controlla il pixel di coordinate 36-48? Sicuramente la Commodore non si é sprecata in praticitá: in effetti il calcolo di tale valore è un po' laborioso, anche se basterá un'apposita routine che faccia per noi il lavoro per risolverci il problema.
La memoria di schermo é organizzata nel seguente modo: a ogni byte corrispondono otto pixel disposti in senso orizzontale: il byte piú significativo (bit 7) rappresenta il pixel piú a sinistra della fila, mentre il bit meno significativo il pixel piú a destra. Come ho giá detto, se il bit é settato, il pixel corrispondente ha il colore carattere, altrimenti il colore di schermo.
Il primo byte della memoria di schermo ha associato la prima fila di otto pixel in alto a sinistra, il secondo byte ha associati gli otto pixel sottostanti ai primi e cosı via fino all'ottavo byte, corrispondente alla locazione 8199 (durante questa spiegazione per semplicitá riterró scontato l'uso della configurazione più semplice, con la memoria di schermo da 8192 a 16191).
Il nono byte non rappresenta la fila di pixel sotto quella dell'ottavo byte, bensì la fila di otto pixel a destra del primo byte, il decimo byte la fila sotto al nono e cosı via fino al sedicesimo byte. Il ciclo riprende dal diciassettesomo byte, fino a esaurire la prima riga. Penso che la seguente tabella possa chiarire le idee:

BYTE1	BYTE 9	BYTE17	BYTE25	BYTE33 . . .
BYTE2	BYTE10	BYTE18	BYTE26	BYTE34 . . .
BYTE3	BYTE11	BYTE19	BYTE27	BYTE35 . . .
BYTE4	BYTE12	BYTE20	BYTE28	BYTE36 . . .
BYTE5	BYTE13	BYTE21	BYTE29	BYTE37 . . .
BYTE6	BYTE14	BYTE22	BYTE30	BYTE38 . . .
BYTE7	BYTE15	BYTE23	BYTE31	BYTE39 . . .
BYTE8	BYTE16	BYTE24	BYTE32	BYTE40 . . .

Terminata la prima riga, dove l'ultimo byte sará il 319, si riprende allo stesso modo da sotto il byte 8: il byte 320 infatti rappresenta la fila di pixel sotto il byte 8, il byte 321 la fila sotto il 320 e così via. Le righe ovviamente sono 25 (200 pixel diviso otto file di pixel per riga sono 25 righe).
Avrete perció capito quale é l'altro motivo per cui é importante la corrispondenza tra caratteri e gruppi di otto per otto pixel: la memoria di schermo in alta risoluzione, infatti, é organizzata come se avessimo 40x25 caratteri definitivi.

Questa similitudine sará utilissima qualora desiderassimo scrivere dei caratteri sullo schermo in alta risoluzione, poiché sara sufficiente cercare gli otto byte che definiscono un carattere e portarli così come sono in memoria di schermo, naturalmente avendo l'accortezza di calcolare a quale byte corrisponde la posizione dove volete che il carattere sia scritto.
Vediamo quindi come si può eseguire questo calcolo. Sappiamo che a ogni carattere corrispondono 8 bytes, perciò se volessimo scrivere il carattere sulla prima riga la locazione cercata sará BASE + 8*COLONNA dove BASE é la locazione in cui inizia la memoria di schermo (8192 nel nostro caso) e COLONNA é la colonna desiderata, considerando che le colonne siano numerate da 0 a 39.
Ma se volessimo scrivere su un'altra riga? Sappiamo che a ogni riga sono associati 320 bytes, perció dovremo aggiungere al valore calcolato prima 320*RIGA, dove RIGA é la riga desiderata, contando le righe da 0 a 24. Trovato il valore della locazione, ci basterá cercare i bytes che definiscono il carattere desiderato e trasferirli dalla locazione trovata in poi. È quanto fa il programma GRAFICA 1:

CARICATELO CON L'ISTRUZIONE LOAD "GRAFICA 1".

Questo programma dispone lo schermo in alta risoluzione e trasferisce il set dei caratteri alle locazioni da 16384 in poi, per semplicitá d'uso. Un'apposita routine provvede inoltre a leggere la tastiera: se un tasto alfabetico risulta premuto il corrispondente carattere viene scritto sullo schermo in alta risoluzione. Provate il programma e poi fermatelo premendo lo spazio e listatelo per studiare la routine appena descritta. Non é troppo difficile vero?.
Passiamo ora a vedere come ottenere l'alta risoluzione vera e propria. Supponiamo di volere *accendere* il pixel di coordinate 125,87 dove il primo numero é l'ascissa, ovvero il numero del pixel in senso orizzontale, contato da sinistra verso destra ricordando che il primo ha ascissa 0, mentre il secondo numero é l'ordinata, cioé la posizione del pixel in senso verticale contando dall'alto verso il basso. La prima operazione da compiere é stabilire a quale carattere appartiene il pixel, ovvero a quale gruppo di otto per otto pixel.
Ció puó essere calcolato facilmente: la colonna é la parte intera dell'ascissa diviso 8, la riga é la parte intera dell'ordinata diviso 8. Suponiamo che X e Y siano le due coordinate: serviranno le seguenti istruzioni Basic:
RI = INT(Y/8):CO = INT(X/8)
In base a quanto detto prima, ora sapete a quale gruppo di otto bytes si deve accedere per modificare il pixel desiderato. Nell'esempio indicato si otterrà RI = 10 e CO = 15, perciò il gruppo di otto bytes è quello che inizia dal byte: BASE + 8*15 + 320*10 = 11512. Per sapere quale degli otto bytes tra 11512 e 11519 è quello che ci serve basterà la semplice operazione LI = YAND7. La formula generale per conoscere il byte che controlla il pixel desiderato è perciò:

BY = BASE + 8*(INT(Y/8) + 320*(INT(X/8)) + (YAND7).
Non mi stancherò di ripetere che BASE è l'inizio della memoria di schermo dell'alta risoluzioe, mentre X e Y sono le coordinate del pixel desiderato. Nell'esempio considerato prima otterremo BY = 11519. I problemi però non finiscono qui: voi infatti sapete che un byte individua ben otto pixel, uno per bit: qual'è quello giusto? Semplice: basta fare BI = 7-(XAND7). Otterremo in BI un valore tra 0 e 7 che corrisponde esattamente al numero del bit che dobbiamo modificare. Nell'esempio precedente sarà BI = 2. Ora che disponiamo delle variabili BY e BI possiamo procedere: per accendere il pixel si farà: POKEBY,PEEK(BY)OR(2↑BI) e per spegnere il pixel: POKEBY,PEEK(BY)AND(255-2↑BI).
Sembra difficile, ma in realtà una volta che avete scritto in un programma due brevi subroutine (una per accendere, l'altra per spegnere) che contengono le tre istuzioni che ho appena mostrato, per accendere o spegnere un pixel è sufficiente assegnare il valore desiderato alle variabili X e Y e chiamare la sobroutine opportuna. Un esempio di utilizzo della grafica in alta risoluzione lo potete avere dal programma GRAFICA 2.

CARICATELO CON L'ISTRUZIONE LOAD"GRAFICA 2".

Date il RUN: comparirà uno schermo bianco con un puntino nero in mezzo (un pixel naturalmente!). Inserite il joystick in porta 2 e provate a muoverlo: si muoverà anche il puntino, lasciando una traccia. In questo modo potete provare a disegnare sullo schermo in alta risoluzione. Per cancellare lo schermo premete lo spazio e ricordate che l'operazione di cancellazione, essendo in basic, richiede un pò di tempo.

CARICATE ORA IL PRORAMMA GRAFICA 3 CON L'ISTRUZIONE LOAD"GRAFICA 3".

Questo programma è simile al precedente, nel senso che è un esempio di grafica ad alta risoluzione. Questo però è un programma di geometria: se date il RUN compare un menù che vi offre diverse scelte tra figure geometriche: premete il tasto numerico corrispondente e il computer vi disegnerà in alta risoluzione tali figure. Al solito, quando avete terminato di vedere un programma, listatelo e cercate di capirne il funzionamento, che altro non è che un'applicazione delle regole e delle formulette viste finora. Nel prossimo articolo parleremo dell'altra grafica di cui dispone il C64, il modo multicolor in alta risoluzione, e vi prometto delle immagini strabilianti, che vi faranno pensare che il vostro monitor o il vostro televisore siano collegati a un computer da 10 megalire. Oltre alle meraviglie dell'alta risoluzione multicolor vedremo lo scrolling di schermo, una tecnica che permette di creare animazioni di notevole effetto.

L'ALTERAZIONE DELLO STATO DEI FLAGS

Esistono istruzioni che permettono al programmatore di alterare lo stato di alcuni flags durante l'esecuzione di un programma; ciò può tornare utile se vogliamo per esempio disabilitare o abilitare l'interrupt o se durante una rotazione, vogliamo inserire un "1" nel registro coinvolto e così via. Tali istruzioni funzionano settando (portare a "1") il flag o cancellandolo (portarlo a "0"). È naturale che il tipo di indirizzamento in queste istruzioni è implicito in quanto l'istruzione stessa implica il flag che deve essere coinvolto.

6.1.1 SEC

```
  N V B D I Z C
  - - - - - - 1
```

Questa istruzione permette di settare il flag di carry, cioè portarlo a "1".

6.1.2 CLC

```
  N V B D I Z C
  - - - - - - 0
```

Esattamente il contrario di SEC, azzera il flag di carry.

6.1.3 SEI

```
  N V B D I Z C
  - - - - 1 - -
```

Permette di portare il bit di disabilitazione di interrupt a 1, ciò significa che qualsiasi richiesta di interrupt proveniente da un dispositivo esterno sarà da ora in poi ignorata.

6.1.4 CLI

```
  N V B D I Z C
  - - - - 0 - -
```

Il Contrario di SEI. Dopo l'esecuzione di questa istruzione la CPU risponderà a ogni richiesta di interruzione.
Si ricordi che le istruzioni CLI e SEI agiscono sull'interrupt mascherabile e che quindi un interrupt non mascherabile provocherà sempre un'interruzione di esecuzione da parte della CPU.

6.1.5 CLV

```
  N V B D I Z C
  - 0 - - - - -
```

Questa istruzione permette di cancellare il flag di overflow. Non esiste un'istruzione che abbia funzione contraria.

6.1.6 SED

N V B D I Z C
- - - 1 - - -

Abilita il modo di calcolo decimale. Tutti i calcoli che saranno d'ora in poi eseguiti nell'accumulatore saranno fatti usando codice BCD (binary coded decimal).

6.1.7 CLD

N V B D I Z C
- - - 0 - - -

Azzera il modo decimale. I calcoli saranno eseguiti in modo binario.

6.2 BIT

N V B D I Z C
M_7 M_6 - - - ↕ -

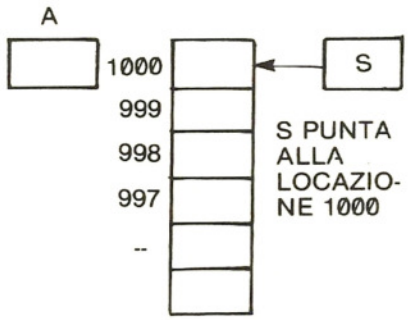
Questa istruzione esegue un and tra memoria e accumulatore. Inoltre i bit 7 e 6 della memoria sono riportati nei flag N e V. L'and comunque viene calcolato ma il risultato non viene memorizzato, per cui dopo l'operazione, accumulatore e memoria rimangono inalterati.
Indirizzamenti possibili:

ABSOLUTE
ZERO PAGE

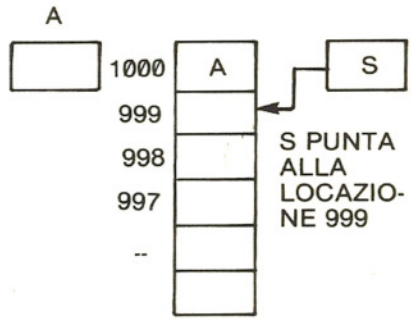
6.3 SALVATAGGIO DEI REGISTRI.

Esistono particolari istruzioni che permettono di salvare il contenuto dei registri in una determinata zona di memoria e in seguito richiamarla. Tale zona di memoria è puntata dallo stack pointer (S).
Facciamo un esempio salvando il contenuto dell'accumulatore per vedere l'esatto funzionamento della procedura di salvataggio.
Incontrata l'istruzione PHA la CPU controlla a che locazione punta S, in seguito preleva il dato dall'accumulatore e lo trasferisce nella memoria puntata da S. In seguito S viene decrementato per evitare che altri salvataggi avvengano nella stessa locazione.

PRIMA
DELL'ESECUZIONE



DOPO
DELL'ESECUZIONE



La locazione 1000 contiene un dato identico a quello in A.

L'operazione di recupero di un dato dallo stack (la zona di memoria in cui si salva) è esattamente il contrario di quella di salvataggio, anche come ordine di esecuzione. Per cui viene incrementato S ed il dato da esso puntato trasferito in A.
Si ricordi che lo stack non è in grado di riconoscere se un dato, precedentemente salvato, appartiene a un registro o a un altro, per cui molto importante è che le operazioni di salvataggio seguano un ordine preciso in modo che l'ultimo dato salvato sia il primo che viene recuperato.
Se per esempio dovessimo salvare lo stato dei registri A e P e dovessimo in seguito recuperarli avremmo quest'ordine:

```
1 SALVATAGGIO A
2 SALVATAGGIO P
3
4
5 ESECUZIONE
  DI
  PROGRAMMA
n
n+1   recupero P
n+Z   recupero A
```

Ecco le istruzioni che permettono il salvataggio e il recupero dei registri:

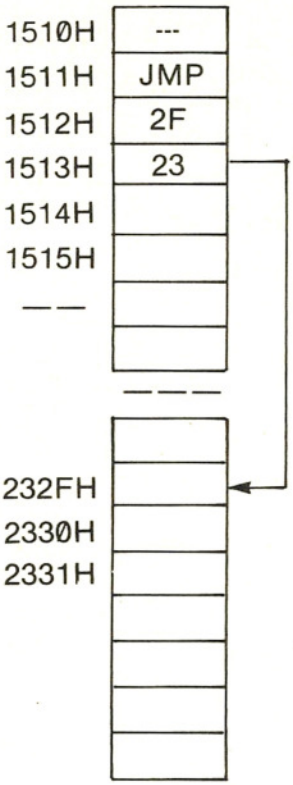
```
PHA ; salvataggio accumulatore sullo stack
PLA ; recupero
PHP ; salvataggio "status register"
PLP ; recupero
```

Ricordiamo che lo stack pointer punta ad una pagina di 256 bytes tra gli indirizzi 511 e 256, per cui esiste una limitazione del numero di salvataggi possibili.
In questa ultima puntata del l/m affronteremo tutte le istruzioni relative al cambio di indirizzo di elaborazione dei dati. Il che significa scegliere se far eseguire una parte di programma piuttosto che un'altra alla CPU. Praticamente è l'equivalente del "GOTO" o "GOSUB" del Basic.
L'unica differenza è il modo di indirizzamento che in Basic vuole dire fornire al computer l'indirizzo di linea Basic da cui continuare l'elaborazione, in l/m corrisponde a dare un indirizzo di memoria da cui continuare.
In una CPU 6510 esistono tre tipi di "salti" i BRANCHES, i JUMPS e le chiamate a SUBROUTINES.

6.4 JUMP

Questa è l'unica istruzione di salto incondizionato esistente nella 6510. Quando incontrate questo tipo di istruzione, durante un'elaborazione la CPU carica l'indirizzo assoluto seguente l'istruzione nel PC e continua da tale indirizzo l'elaborazione. Ec-

co un esempio:

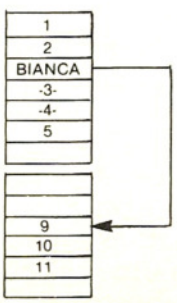


Come potete notare la CPU ricava dai due bytes seguenti l'istruzione jump l'indirizzo 232FH cui "saltare".

Notate inoltre che l'indirizzamento è assoluto.

6.5 BRANCHES

I Branches sono del tutto simili ai Jumps, ma usano un indirizzamento relativo e inoltre la loro esecuzione è soggetta a determinate condizioni dei flags.
Alla CPU si può praticamente dire di eseguire il salto se un determinato flag è in un determinato stato, altrimenti, continuare oltre l'istruzione.



in questo esempio la CPU esegue i passi 1 e 2, poi l'istruzione branch gli chiede di provare se un certo flag è settato, se ciò è vero allora la CPU passa ai passi 9, 10 e così via, altrimenti continua i passi 3,4 ecc.
Il branch è praticamente l'equivalente Basic di IF...THEN...

Esistono vari branches, a seconda del flag da testare, comunque nessuno di loro altera lo stato del registro flags.

6.5.1 BCC

Salto nel caso il flag di Carry sia a 0 (branch on Carry clear).

6.5.2 BLS

Salto nel caso di flag di Carry settato (branch on carry set)

6.5.3. BEQ

Salto su carry Zero settato (branch on zero)

6.5.4. BNE

Salto su flag Zero resettato (branch on not zero)

6.5.5 BMI

Salto su flag Negative settato (branch on minus)

6.5.6 BPL

Salto su flag Negative resettato (branch on plus)

6.5.7 BVC

Salto su Overflow non settato (branch on overflow clear)

6.5.8 BVS

Salto su Overflow non settato (branch on overflow set)

Tutte queste istruzioni permettono di far eseguire un programma in un modo o nell'altro a secondo di ben precise condizioni. Per esempio, se volessimo confrontare con un numero e in seguito far eseguire una cosa o un'altra a seconda che il numero sia uguale o no potremmo scrivere:

1) CPX #27 : confronta il contenuto di X con 27
2) BEQ 65F : se esso è uguale salta 5FH loc. avanti
3) LDA #11 : altrimenti continua riempiendo
4) STA..... : a con 11 e così via...
5) .....

6.5.9 JSR-RET

Questa istruzione fa eseguire un salto a subroutine il cui indirizzo è specificato dai due bytes seguenti l'istruzione. L'indirizza-

mento è assoluto e l'indirizzo di ritorno è memorizzato nello stack. La subroutine deve terminare con RET e l'indirizzamento è assoluto.

6.5.10 BRK-RTI

Questa istruzione simula un interrupt esterno ed esegue un salto alla subroutine il cui indirizzo è specificato agli indirizzi FFFH ed FFFFH.
Il flag I e settato a 1 e il ritorno da subroutine è eseguito da RTI.

6.5.11 NOP

Questa istruzione semplicemente significa no-operation e dice alla CPU di non eseguire nessuna operazione. Si usa di solito nei loops dove necessita un certo ritardo.

Abbiamo, con quest'ultima puntata, finito il corso sul linguaggio macchina della CPU 6510; speriamo di essere stati abbastanza chiari ed esaurienti e di avervi fornito le basi necessarie alla programmazione.

6 — FINE

A.V.S.

ISTRUZIONI PER IL CARICAMENTO DELLA CASSETTA

Se il vostro computer è un **Commodore 64** o un **Commodore 128**, prendete la cassetta e inseritela nel registratore con la facciata prescelta - A oppure B - in alto. Accendete il computer e premete contemporaneamente i tasti <SHIFT> e <RUN STOP>. A questo punto il monitor visualizzerà la scritta PRESS PLAY ON TAPE (schiacciare il tasto <PLAY> sul registratore).

Eseguite il comando e attendete che appaia la scritta FOUND (nome programma). Premete allora la barra spaziatrice e attendete. Se dovesse apparire qualche scritta di errore, riavvolgete la cassetta e ripetete le operazioni indicate. Se invece tutto è ok, schiacciate il tasto <STOP> sul registratore: dopo qualche istante apparirà la prima pagina della nostra videorivista con l'elenco dei programmi contenuti nella cassetta.

Quando apparirà la scritta PREMERE UN TASTO premete un tasto a caso sulla tastiera.

Non appena lo schermo ripresenterà i colori e le scritte che normalmente vedete quando accendete il computer, premete nuovamente <SHIFT> e <RUN STOP>; schiacciate quindi il tasto <PLAY> sul registratore, attendete che il computer trovi il programma successivo, premete la barra spaziatrice e pazientate un attimo per permettere al computer di caricare il programma. Se non appaiono scritte di errore, premete <STOP> sul registratore.

Se il vostro computer è invece un **Commodore 16** o un **Commodore plus/4**, dopo aver preso la cassetta e averla inserita nel registratore con la facciata prescelta - A oppure B - in alto, accendete il computer, poi digitate sulla tastiera LOAD e premete il tasto <RETURN>. A questo punto il monitor visualizzerà la scritta PRESS PLAY ON TAPE (schiacciare il tasto <PLAY> sul registratore).

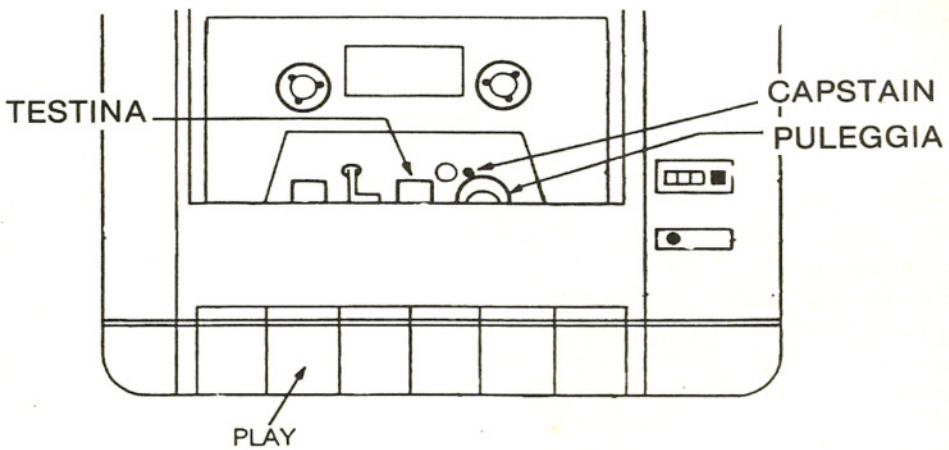
Eseguite il comando e attendete che appaia la scritta FOUND (nome programma): dopo alcuni secondi il computer inizierà il caricamento e dopo circa 1 minuto comparirà il cursore lampeggiante. Digitate allora RUN e premete il tasto <RETURN> e vedrete la lista dei programmi contenuti nella cassetta.

Per caricare il primo programma digitate LOAD e il tasto <RETURN> e quando compare il cursore lampeggiante ancora LOAD e il tasto <RETURN>. Dopo alcuni secondi (circa 40) comparirà sul video il programma scelto.

Seguite queste semplici istruzioni ogni volta che volete caricare un programma di Videoteca Computer. Per cambiare programma spegnete e riaccendete il computer e ricominciate seguendo le solite procedure per il caricamento.

Qualora trovaste difficile caricare i programmi o apparissero sullo schermo le scritte di errore (es. LOAD ERROR), vi consigliamo di procedere nel modo seguente:

1. Spegnete e riaccendete il computer
2. Alzate lo sportello del registratore
3. Premete il tasto <PLAY> del registratore
4. Dopo aver imbevuto di alcool un Cotton Fioc, passatelo sulla testina del registratore e sulla puleggia di gomma nera alla destra della testina
5. Attendete qualche minuto fino alla completa evaporazione dell'alcool
6. Inserite la cassetta e procedete secondo le istruzioni.



Se dopo aver eseguito le suddette operazioni i programmi non vengono ancora letti correttamente, vi consigliamo di far controllare il vostro registratore da un laboratorio tecnico specializzato.

N.B. I nostri programmi sono registrati su supporti magnetici di alta qualità. Ciò ne garantisce la perfetta registrazione e riproduzione. A ogni modo, nel caso in cui, dopo aver tentato le soluzioni sopra indicate, non riusciate ancora a caricare il programma, telefonate al nostro ufficio tecnico: i nostri esperti sono a vostra disposizione.

PER COMMODORE 64

BOCCE (BOWL)

Nonostante l'apparente semplicità, il gioco delle bocce richiede non poca abilità: prontezza di riflessi, precisione, senso della strategia, e, naturalmente, buona mira. Dopo essere stato per

qualche anno abbastanza trascurato, questo gioco sta ritornando di moda e anche i giovani sono tornati a sfidarsi la sera nei bocciodromi annessi ai bar o alle trattorie delle città e dei paesi dove abitano. Ma forse pochi sanno che quest'avvincente attività ha origini antichissime e che se ne trovano testimonianze già in alcuni geroglifici dell'antico Egitto risalenti al 5000 a.C. nei quali, alcune figure, rappresentano uomini impegnati a lanciare oggetti sferici. Anche i romani ne erano appassionati, nel Medioevo si giocava a bocce nei monasteri e durante il Rinascimento questo era un passatempo molto gradito persino dai re.

Ciò non deve sorprendere perché, provate per credere, giocare a bocce è veramente divertente, un pretesto per trascorrere una sera in compagnia impegnati in sfide all'ultimo punto con gli amici, ma, come abbiamo già detto, per poter ottenere buoni risultati questo sport richiede certe capacità e soprattutto acutezza visiva.

Per questa ragione, affinchè possiate diventare dei campioni e sbaragliare sul campo tutti gli avversari, abbiamo ideato per il vostro **Commodore 64** il programma BOCCE grazie al quale il computer sarà per voi un inflessibile e instancabile allenatore.

Gli sfidanti sono due per volta.

Appena caricato il gioco appare la presentazione iniziale; premete il tasto -SPACE- e inserite le risposte che vi vengono richieste:

— il nome del primo e del secondo giocatore;
— quale dei due giocatori (1/2) deve tirare il pallino;
— quale punteggio si vuole raggiungere (vedi tasti corrispondenti);
— quale giocatore vuole le bocce rosse.

A questo punto il giocatore prescelto posizionerà il pallino premendo i seguenti tasti:

-Z- per muoverlo a sinistra
-X- per muoverlo a destra
-A- per muoverlo verso il basso
-Q- per muoverlo verso il fondo del campo

Premete il tasto -←- dopo l'inserimento di ogni istruzione.

È ora il turno del primo giocatore che deve scegliere se 'bocciare' o 'andare a punto' (1/2). Prima di tirare deve stabilire:

— la 'posizione' del punto di partenza (tasti -Z- e -X-+ -←-);
— la 'potenza del tiro', cioè la lunghezza del lancio, tenendo premuto -←- per alcuni secondi; premere -←- per continuare;
— la 'direzione' del lancio, che viene indicata da una lineetta che si muove sul lato di fondo del campo. Premere -←- per bloccarla nel punto in cui si vuole che termini il lancio.

Dopo che il primo giocatore ha eseguito il lancio, tocca al secondo fare lo stesso e si continua così alternativamente.

Attenzione, bisogna cercare di avvicinarsi il più possibile al pallino; se si toccano il pallino o le sponde laterali o se il tiro non ha superato la metà del campo, la boccia viene eliminata.

Vince chi raggiunge per primo il punteggio stabilito.

NIKI

I circuiti integrati VIC del vostro **Commodore 64** offrono tante possibilità di invenzione grafica, spesso però i videogiochi si limitano a riproduzioni molto stilizzate di situazioni realistiche. Accogliamo perciò sempre con piacere i giochi in cui l'autore è riuscito a creare assurdi personaggi, ai quali subito ci si affeziona, e ambientazioni fantastiche, che stimolano la nostra fantasia. È questo il caso di NIKI, che qui vi presentiamo, il piccolo robot che caracolla trafelato tra saltelli, rapide corsette, cadute e capriole su un improbabile paesaggio irto di difficoltà da superare: montagnette, salite, improvvise discese e fossi da brivido. Come se non bastasse, il suo avanzare è ostacolato da grossi massi rotanti che precipitano dalle pendenze rischiando di travolgerlo.

Il suo tragitto diventa inoltre sempre più difficoltoso; a un certo punto arriverà addirittura ad attraversare un fiume saltellando sui pezzi di ghiaccio che vi galleggiano e dovrà fare di tutto per mantenersi in equilibrio per evitare di fare un bagno che gli sarebbe fatale.

Caricato il programma, potrete quindi selezionare le seguenti opzioni:

-F1- per iniziare il gioco
-F3- per scegliere il livello di difficoltà
-F5- per scegliere se giocare in 1 o 2 persone.

Se invece aspetterete qualche secondo otterrete automaticamente la DEMO.

Per aiutare NIKI a raggiungere la sua meta potrete servirvi del joystick o della tastiera.

Inserite il joystick in porta 2 e muovetelo:

A SINISTRA per andare a sinistra
A DESTRA per andare a destra

premete invece il FIRE per saltare.

Se utilizzate i tasti:

-CRSR- (↑ ↓) per andare a sinistra

-CRSR- (← →) per andare a destra

-SPACE- per saltare.

Nel corso del gioco appaiono:

— il punteggio raggiunto
— le vite ancora a disposizione (NIKI ne ha 3)
— il tempo restante
— il livello raggiunto.

Al termine appare la classifica: un asterisco indica il punteggio che vi sarete meritati come consiglieri, scrivete lì a fianco il vostro nome digitandolo sulla tastiera e premete -RETURN- se volete ritentare, NIKI sarà contento di rimettersi in viaggio, ma siate prudenti la sua sorte è nelle vostre mani.

OCCHIO ALLA PAROLA

Grazie a una routine che permette di visualizzare le lettere che compongono di volta in volta una parola scelta a caso nell'ambito di un repertorio prestabilito, OCCHIO ALLA PAROLA costituisce un divertentissimo video indovinello che metterà alla prova e permetterà di acuire la vostra velocità di percezione visiva e capacità di lettura in modo piacevole e divertente.

Accendete quindi il vostro **Commodore 64** e sedetevi vicino al vostro amico/a che considerate più abile e difficile da sfidare. Caricato il programma, inserite il nome del primo giocatore digitando sulla tastiera le lettere che lo compongono e poi premete -RETURN-; fate lo stesso per il secondo giocatore. A questo punto vi apparirà un quadro con tutte le lettere dell'alfabeto e l'indicazione 1 o 2 con il nome di chi deve giocare per primo.

Toccherà a lui aguzzare la vista, seguire attentamente l'illuminazione delle lettere che compongono la misteriosa parola da indovinare e quindi scrivere il vocabolo che pensa sia giusto (digitandolo sulla tastiera e premendo -RETURN-). Se avrà indovinato apparirà sullo schermo la gratificante scritta 'ESATTO!!!' altrimenti, come Edipo con la Sfinge, il malcapitato sarà condannato a morte (scherziamo, naturalmente).

Premendo un tasto si passa la mano all'altro giocatore e così via di seguito in un succedersi di lampi che danno flash di parole sempre più lunghe e complicate.

Al termine della partita appare un quadro con l'indicazione del nobile vincitore.

Se ve la sentite di cimentarvi nuovamente nell'impresa, rispondete positivamente alla domanda del computer-Sfinge che vi sfida a giocare ancora.

INCURSOR

I videogiochi di guerra sono tra i più appassionanti perché richiedono concentrazione, velocità di riflessi, abilità strategica e anche perché la lotta contro un nemico stimola a impegnarsi per un naturale istinto di difesa e anche, diciamolo, di aggressività. E a noi, grintosi amanti dell'elettronica, ma anche della vita e della fantasia, non dispiace affatto scaricare liberamente questa aggressività, purché sia per gioco, naturalmente, e contro le sprites di un computer!

Accendiamo quindi il nostro **Commodore 64** e scateniamoci con INCURSOR, che ci carica a bordo di una portaerei che si sta avvicinando alla costa per un'incursione sul territorio nemico. Naturalmente il nostro è il ruolo di massima responsabilità: quello del pilota dell'aereo che a cui è affidata la missione di bombardamento.

Appena caricato il programma, appare l'elenco dei tasti da usare:

-F5- per far decollare l'aereo
-F7 per abbassare di quota l'aereo

-Z- per virare a sinistra
-X- per virare a destra
-C- per ridurre la velocità e atterrare sulla portaerei
-SPACE- per sparare i missili
-F3- per sganciare le bombe
-C- per lanciarsi col paracadute.

L'aereo può essere manovrato anche con il joystick, che va inserito in porta 2.

Premere un tasto qualsiasi per iniziare il gioco e selezionare il livello di difficoltà, che è progressivo da 1 a 5.

Dopo aver decollato dalla portaerei, l'aereo deve velocemente dirigersi sopra la zona nemica per iniziare la missione di bombardamento durante la quale dovrà fare attenzione a evitare i colpi della contraerei.

Nel quadro di gioco lo stato della situazione è indicato con:
— il record da superare
— il punteggio raggiunto
— la quantità di carburante rimasta
— la disponibilità di bombe
— la velocità di volo
— la quantità di missili a disposizione.

Quando la scorta di carburante sta per terminare, l'aereo deve tornare a bordo della nave per fare rifornimento.

Se l'aereo viene colpito, l'intrepido pilota può ancora tentare di salvarsi lanciandosi con il paracadute.

Al termine del gioco il comandante potrà inserire il suo nome, che apparirà a fianco del punteggio conquistato nella postazione meritata nella classifica dei grandi.

PER COMMODORE 64 e COMMODORE 16

CALENDARIO

Oltre a una serie di fantastici giochi di abilità e di rapidità di riflessi oppure capaci di farci evadere e vivere situazioni esotiche, di sogno o futuribili, il nostro computer **Commodore 16** o **Commodore 64** ci permette di servirci anche di programmi 'seri', che ci mettono rapidamente a disposizione un elevato numero di dati e di risultati di calcoli complessi che si possono rivelare assai utili per i nostri studi, interessi o per altre esigenze personali. Tra questi programmi, particolare e aperto a molte possibili utilizzazioni da parte vostra è quello che qui vi presentiamo: CALENDARIO.

I calendari sono i sistemi di regole che fin dall'antichità gli uomini hanno inventato per ripartire il tempo in periodi più o meno lunghi (giorni, mesi, anni) e per stabilire le date rispetto a un determinato punto di partenza (anno zero).

Nel tentativo di avvicinarsi sempre di più alla realtà dell'anno solare, che regola i cicli dell'agricoltura, nel corso della storia sono stati adottati diversi sistemi di calendario; già gli antichi

egizi e i greci ne avevano uno, e naturalmente anche i razionalissimi romani.

Il calendario attualmente adottato in quasi tutti i paesi del mondo è il calendario gregoriano, che venne istituito nel 1582 da papa Gregorio XIII, ma il programma che qui vi proponiamo vi fornisce la numerazione dei giorni secondo il sistema del GIORNO GIULIANO, che fu inventato da Giuseppe Giusto Scaligero. Questo sistema numera progressivamente i giorni a partire dal 1 gennaio 4713 a.C. ed è particolarmente utile a chi si interessa di astronomia e di astrologia perché permette di determinare con rapidità ed estrema precisione le fasi lunari.

La prima parte del programma permette quindi di trasformare qualsiasi data del calendario attuale secondo il sistema di numerazione del giorno giuliano e ne dà la fase lunare corrispondente. Potrete così, per esempio, scoprire immediatamente qual era il giorno della settimana e la fase lunare del 5 maggio del 320 a.C. senza dover fare complicati calcoli e senza i molto probabili errori.

Grazie alla seconda parte è possibile invece avere il numero dei giorni che intercorrono tra due date qualsiasi; inoltre, su video o su carta, il calendario di un mese tra il 1600 e il 1999 e, su carta, il calendario di un intero anno. Ciò si presta a varie e personali utilizzazioni ed è prezioso per chi, per esempio, deve fare previsioni a lunga scadenza.

Caricato il programma potrete quindi scegliere tra 5 opzioni, premendo i seguenti tasti:

-1- per convertire nella numerazione del giorno giuliano una data del calendario che utilizziamo attualmente. Vengono forniti il giorno della settimana e la fase lunare;
-2- per ottenere il numero dei giorni che intercorre tra due date;
-3- per ottenere la visualizzazione o la stampa (V/S) di tutti i giorni di un determinato mese dell'anno che ci interessa;
-4- per ottenere la stampa su carta di un calendario annuale. In questo caso si deve prima rispondere positivamente (S) alla domanda se è accesa la stampante;
-5- per ottenere le istruzioni su video.

Provate allora a scegliere l'opzione 2, a digitare la data odierna e quella della vostra nascita. Il computer calcolerà immediatamente il numero dei giorni trascorsi. Non sentitevi troppo vecchi se scoprirete di avere più di 7000 giorni: non sono poi così tanti!.

PER COMMODORE 16

TALPE

Un gioco vivacemente colorato per il vostro **Commodore 16**, apparentemente semplice ma che in realtà richiede maggior attenzione e rapidità di riflessi di quanto non sembri a prima vista, è questa caccia a una dispettosa stirpe di talpe che all'improvviso sbucano dalle montagnette di terra intorno alle uscite delle lo-

ro tane, che si diramano in un intricato labirinto di gallerie sotterranee.

Potrete stabilire voi stessi la durata di questa caccia alle devastatrici di giardini premendo, subito dopo aver caricato il programma, il tasto corrispondente al numero di minuti di gioco.

Ogni montagnetta è contrassegnata da una lettera dell'alfabeto (Q,W,E,A,S,D,Z,X,C) e, per colpire la talpa, dovrete premere il tasto di quella lettera nel momento in cui l'animaletto è in fase di uscita. Il vostro successo sarà segnalato sullo schermo da un fantastico scrolling colorato lampeggiante.

Nel quadro di gioco sono sempre segnalati il record assoluto e il punteggio da voi raggiunto fino a quel momento.

Impegnatevi al massimo per indovinare da quale tana la prossima volta le imprevedibili talpe decideranno di sbucare e ricordatevi che per ogni colpo andato a vuoto vi verranno assegnati dei punti negativi.

Allo scadere del tempo apparirà un quadro con la classifica dei record e il risultato da voi raggiunto.

Se decidete di riprovare o che è il momento di passare la mano a un altro cacciatore rispondete positivamente (s) alla domanda 'vuoi rigiocare', in caso contrario (n) il gioco terminerà.

Buon divertimento e...in bocca alla talpa!

SLOT MACHINE

Per lei milioni di persone hanno perso la testa. Pur essendo sconvolgentemente semplice e non eccessivamente appariscente ha un fascino straordinario, dovuto alla sua imprevedibilità e al 'modo di fare' misterioso. È la SLOT MACHINE, la famosa e irresistibile macchina mangiasoldi che a Las Vegas, in America, in Inghilterra e in tutto il mondo tiene avvinti infiniti ammiratori di tutte le età che la corteggiano scongiurandola di concedere le sue grazie. Lei, regina indifferente, molte volte è purtroppo crudele e si rifiuta ostinatamente di esaudire le loro richieste, ma loro non smetteranno di interrogarla facendosi ridurre sul lastrico prima di rinunciare a tentare. Questo perché sanno che ogni tanto, per le stesse imprevedibili ragioni, si concede, rendendo felice e ricco quel fortunato.

Ora, grazie al vostro **Commodore 16** e al programma SLOT MACHINE, che troverete nella cassetta acclusa a questo fascicolo, potrete avere quest'affascinante oracolo a casa vostra.

Caricato il programma, appariranno quindi sul vostro schermo le combinazioni vincenti: da quelle che capitano ogni tanto a quelle più rare, destinate ai toccati dalla Fortuna.

Poi, eccola. Per interrogarla premete un tasto e lei comincerà a rispondervi.

In un riquadro apparirà il vostro punteggio che salirà o diminuirà a seconda delle combinazioni uscite.

Naturalmente, prima di presentarvela l'abbiamo domata, togliendole la pericolosa facoltà di ridurvi in rovina.

Non giocherete quindi i vostri soldi, ma potrete stabilire un limite di tempo e sfidare i vostri amici per vedere chi avrà avuto maggiore capacità di sedurla, lei, la dea della Fortuna.

ASTRONOMIA

Sono sempre di più gli appassionati di astronomia, che leggono libri e riviste specializzate o quelli che trascorrono ore, nelle notti d'estate, a scrutare il cielo con i loro telescopi. Spesso però, anche per i più esperti, è assai difficile individuare le varie costellazioni e, consultando le complesse tavole astronomiche, calcolare le distanze dei pianeti.

Grazie al computer, ora, con il programma scientifico ASTRONOMIA ideato per il **Commodore 16** gli interessati potranno immediatamente avere la posizione di Venere e di Marte per qualunque data a partire dal 1 gennaio 1950.

Il computer calcolerà la longitudine dei pianeti rispetto alla terra (in gradi); la latitudine rispetto al piano dell'eclittica, la distanza dalla terra in unità astronomiche (un'U.A. corrisponde a 145 milioni di chilometri) e la costellazione in cui si trova il pianeta.

Le costellazioni indicate dal computer sono ovviamente quelle reali che, a causa della precessione dell'asse della terra (dovuta alla sua inclinazione), non coincidono esattamente con i segni astrologici.

Caricato il programma si può quindi scegliere se vedere prima le istruzioni (I) o iniziare subito a utilizzarlo (P); premere poi un tasto per cominciare.

Dopo aver inserito una data, avremo immediatamente le corrispondenti posizioni e le distanze dei pianeti in U.A.; se poi ci interessa un grafico della situazione astronomica basterà rispondere positivamente (S) alla domanda relativa.

Per continuare digitate infine -S- alla domanda 'ancora?', altrimenti digitate -N-.

Chissà che, grazie a questo programma non aumentino i vostri interessi astronomici, magari un giorno, o meglio una notte, ormai che il Voyager ci ha reso familiare persino Urano, potreste arrivare a scoprire un decimo pianeta!

REDKNIGHT

Entra nel mondo magico e avventuroso del Medioevo con questo "Adventure Game" in italiano creato per il tuo **Commodore 16**.

Nell'Adventure, come certamente saprai, non bisogna "smanettare" col Joystick o con la tastiera, per risolvere le varie situazioni di gioco, ma dare delle istruzioni scritte al computer. Se per

esempio il computer ti dice: SEI IN UNA STANZA. Tu gli puoi rispondere: OSSERVO.

Se c'è qualcosa da vedere, il computer te lo dice: SEI IN UNA STANZA E CI SONO DELLE FRECCE.

La tua risposta può essere: PRENDO FRECCE.

Per spostarti da un luogo all'altro, invece, le indicazioni sono semplici: N = NORD, S = SUD, E = EST, O = OVEST.

In questo Adventure sei REDKNIGHT (Cavaliere Rosso) e vuoi sposare la Principessa di Viclandia.

Ma la figlia di un Re non si sposa così facilmente. Devi prima superare tre dure prove per dimostrare al tuo futuro Suocero di essere un suo degno successore al trono. Nella prima prova devi scoprire il nome del Re, che solo una persona conosce. Scopri, perciò, chi sia costui e arriva al suo cospetto attraverso varie peripezie, che però non sono casuali: ogni cosa che incontri, ogni azione che fai crea i presupposti per questo incontro.

Perciò, trovandoti davanti a qualcuno, se alla domanda CHIEDO NOME non hai alcuna risposta, vuol dire che non è la persona giusta; se, invece, hai la risposta: NON POSSO DIRTELO, la persona è giusta, ma non hai fatto, durante il gioco, qualcosa che gli permetta di dirtelo.

Quando hai scoperto il nome torna dal Re e...

Nella seconda avventura devi porre fine alle scorribande del Drago Alato e tornare dal Re.

Affronta, ora, nella terza impresa, il Folle Mago, uccidilo e torna dal Re.

Un consiglio: per meglio giocare un Adventure prendi carta e penna e scriviti lo schema del percorso che stai facendo.

Così, dopo una morte, ricominciando, non correrai lo stesso rischio.

Ti forniamo ora un piccolo glossario delle parole che il computer accetta. Alcune di queste sono valide per tutte e tre le avventure (es. Osservo-Os), altre solo in una o due.

Mancano, naturalmente, delle parole che dovrai scoprire durante il gioco e che servono per la soluzione.

Apro, brucio, incendio, alzo, aiuto, offro, osservo, os, mago, me, N, S, E, O, prendo, chiedo, masso, drago, grotta, fiori, dico, scrivo, do, dono, sollevo, esco, cibo, infilzo, mangio, sughero, chiave, torcia, sussurro, mormoro, cuoco, caverna, talismano, re, nome, attraverso, passo, urlo, prendo, raccolgo, cascata, foglie, scendo, trafiggo, insetti, spada, rupe, porta, savana.

Più di così non posso dirti.

Parti, lancia in resta, e sposati.

VIDEOTECA COMPUTER

11

COMMODORE 64

Lato A

- BOWLS (BOCCE)
- NIKI
- OCCHIO ALLA PAROLA
- INCURSOR
- CALENDARIO
- ESERCIZI DI GRAFICA

COMMODORE 16

Lato B

- CALENDARIO
- TALPE
- SLOT MACHINE
- ASTRONOMIA
- READNIGHT

EV EDITORIALE VIDEO