Locate per C64

di Marco Olivotto - Rovereto (TN)

La breve routine in LM che vi propongo serve ad emulare l'istruzione Locate presente in varie versioni del Basic più evolute di quella standard del C64. In generale, se si vuole posizionare una stringa di caratteri o un numero sullo schermo è necessario utilizzare caratteri di controllo CRSR (POKE a parte, naturalmente). L'uso di tali caratteri presenta almeno due inconvenienti: primo, è abbastanza macchinoso da risultare complicato quando si vogliano eseguire posizionamenti complessi; secondo, è sostanzialmente un metodo di posizionamento relativo, nel senso che il cursore viene spostato sempre in una posizione riferita all'ultima che aveva assunto. Sarebbe utile, per questi motivi, avere un'istruzione che permetta di spostare il cursore direttamente in una posizione specificata dalle coordinate di schermo. Il programma che presento realizza esattamente questo. Si tratta, come già detto, di una routine in LM, molto breve (45 byte in tutto), che inizia alla locazione 49152. Essa richiede che le vengano passati due parametri numerici R e C che rappresentano rispettivamente la Riga e la Colonna su cui si vuole posizionare il cursore. La sintassi corretta è la seguente:

```
SYS 49152,R,C
```

Di solito, per comodità mnemonica, io inserisco all'inizio del programma Basic che utilizza Locate la linea

```
LOCATE=49152
```

in modo da poter scrivere

```
SYS LOCATE,R,C
```

D'ora in poi mi riferirò sempre a tale forma dell'istruzione.

Ovviamente, affinché tutto funzioni a dovere, R e C devono cadere entro i limiti dello schermo. Dal momento che questo, di solito, ha 25 righe e 40 colonne, R dovrà essere compreso tra 0 e 24 e C tra 0 e 39. Se una o entrambe le condizioni non vengono rispettate, si avrà il messaggio

ILLEGAL QUANTITY ERROR

e l'esecuzione verrà sospesa.

L'effetto di SYS LOCATE,R,C è, ripeto, quello di posizionare il cursore nella riga R e nella colonna C. A questo punto per ottenere la visualizzazione nella posizione desiderata basterà usare PRINT, senza alcun bisogno di caratteri di controllo. A titolo di esempio, ecco un programma in grado ci centrare una stringa sullo schermo alla riga 0 (la prima dall'alto).

```
10 PRINT "(CLR)"
20 LOCATE=49152
30 A$="STRINGA CENTRATA"
40 L%=(40-LEN(A$))/2
50 SYS LOCATE,0,L%
60 PRINT A$
70 END
```

Dal momento che non tutti i lettori hanno pratica di LM, presento in figura 1 il caricatore Basic e il disassemblato della routine, in base al quale poi discuteremo la logica su cui si fonda l'istruzione.

Passiamo ora alla descrizione della routine riportata in figura 2.

Essa fa uso di 4 routine dell'interprete Basic:

```
CHKCOM  ($AEFD)
FRMNUM  ($AD8A)
GETADR  ($B7F7)
ERROR   ($A437)
```

e di una routine del Kernal:

```
PLOT    ($FFF0)
```

Il nucleo principale del programma è costituito dalla subroutine che inizia alla locazione C01B. Senza entrare nei dettagli, diremo che essa cerca un parametro dopo l'istruzione SYS e, se lo trova, lo converte in numero intero e lo scarica nell'accumulatore A (vedere appendice). Questa routine, se non trova il parametro o se esso è incompatibile con la sintassi del comando, produce un messaggio di SYNTAX ERROR ed arresta l'esecuzione del programma. Essa viene chiamata all'inizio della routine, in C000. Viene letto il parametro R che viene depositato provvisoriamente nella locazione $FB in pagina 0. Quindi alla locazione C005 si verifica se il valore di R cade nel range prestabilito. Se così non è, il controllo viene trasferito alla subroutine $C027 che stampa il messaggio d'errore ILLEGAL QUANTITY ERROR. Se invece R ha un valore accettabile, si passa alla lettura di C per cui vale una procedura esattamente analoga a quella descritta, salvo che il suo valore viene posto in $FC.

Per farla breve, se entrambi i parametri cadono nel range corretto, viene acceso il flag di carry (necessario per la chiamata della routine di posizionamento del cursore) e vengono caricati nei registri X ed Y i valori di R e C temporaneamente memorizzati in $FB e $FC. Quindi viene chiamata la subroutine PLOT del Kernal ($FFF0) che posiziona il cursore. Infine il controllo ritorna al Basic.

Riassumiamo quanto detto:

$C000 - $C002	Chiamata della subroutine di lettura parametro
$C003 - $C004	Caricamento di R in $FB
$C005 - $C008	Se R è fuori range, subroutine $C027
$C009 - $C00B	Chiamata della subroutine di lettura parametro
$C00C - $C00D	Caricamento di C in $FB
$C00E - $C011	Se C è fuori range, subroutine $C027
$C012 - $C016	Preparazione per chiamata PLOT
$C017 - $C019	Chiamata PLOT ($FFF0)
$C01A	Ritorna al Basic
$C01B - $C026	Subroutine di lettura parametro; restituisce R o C nell'accumulatore

Figura 1
Salvate il programma prima di dare il RUN. Infatti, dal momento che lo scopo di queste poche righe di Basic è solo quello di caricare in memoria dei codici macchina, una volta che questo è stato fatto, il programma non serve più. Così ho messo un NEW alla linea 160. Perciò, attenzione: il programma si autocancella dalla memoria

```
10 REM LOCATE EMULATOR
20 REM BY M.OLIVOTTO - 1988
30 REM
40 PRINT"(CLR)"
50 L=49152
60 FOR I=0 TO 44
70 READ ML%
80 POKE L+I,ML%
90 NEXT I
100 DATA 32,27,192,133,251,201,25,16,30
110 DATA 32,27,192,133,252,201,40,16,21
120 DATA 24,166,251,164,252,32,240,255,96
130 DATA 32,253,174,32,138,173,32,247,183
140 DATA 165,20,96,162,14,32,55,164,96
150 PRINT"LOCATE CARICATO."
160 NEW
170 END
```

C-64

```
10 REM LOCATE DEMO
20 REM BY M.OLIVOTTO - 1988
30 REM
40 LOCATE=49152
50 POKE53280,0
60 POKE53281,2
70 PRINT"(WHT)(CLR)"
80 FORI=0TO12
90 SYSLOCATE,I,I
100 PRINT"AUTO-INDENT"
110 NEXTI
120 FORI=13TO23
130 SYSLOCATE,I,24-I
140 PRINT"BACKWARDS..."
150 NEXTI
160 GOSUB10000
170 FORI=1TO25
180 SYSLOCATE,23,I-1
190 PRINT"                "
200 GOSUB20000
210 SYSLOCATE,23,I
220 PRINT"ONWARD..."
230 GOSUB20000
240 NEXTI
250 GOSUB10000
260 PRINT"(CLR)"
270 A$="AND AUTOCENTER"
280 B$="WITHOUT ANY CRSR CHARACTER!"
290 YA=(40-LEN(A$))/2
300 YB=(40-LEN(B$))/2
310 SYSLOCATE,5,YA
320 PRINTA$
330 SYSLOCATE,10,YB
340 PRINTB$
350 GOSUB10000
360 GOSUB10000
370 PRINT"(CLR)"
380 A$="AND NOW... RANDOMIZE!"
390 YA=(40-LEN(A$))/2
400 SYSLOCATE,10,YA
410 PRINTA$
420 GOSUB10000
425 PRINT"(BLK)"
430 FORI=1TO1000
440 X=RND(I)*23
450 Y=RND(I)*40
460 SYSLOCATE,X,Y
470 PRINT"Q"
480 NEXTI
490 GOSUB10000
495 PRINT"(WHT)"
496 PRINT"(CLR)"
500 A$="BY M.OLIVOTTO - 1988"
510 B$="CIAO, MC!!!"
520 YA=(40-LEN(A$))/2
530 YB=(40-LEN(B$))/2
540 SYSLOCATE,5,YA
550 PRINTA$
560 SYSLOCATE,12,YB
570 PRINTB$
580 GOSUB10000
590 GOSUB10000
600 PRINT"(CLR)"
9999 END
10000 FORK=1TO1000
10010 NEXTK
10020 RETURN
20000 FORK=1TO50
20010 NEXTK
20020 RETURN
```

Locate per C64.

$C027 - $C02C Stampa messaggio ILLEGAL QUANTITY ERROR

Come potete vedere, è una routine piuttosto standard e facilmente comprensibile. Spero di avere fatto qualcosa di utile a molti. Dal canto mio vi posso assicurare che i due minuti necessari a digitare e, in seguito, i pochi secondi necessari a caricare questo programma pagano in termini di flessibilità della gestione video. Provate.

Appendice

Vorrei fare a parte un paio di commenti. Il primo riguarda i puristi del LM. Come gli esperti ben sanno, la routine del Basic GETADR ($B7F7) coinvolge non una, ma due locazioni di pagina 0: la $14 e la $15. In esse vengono caricati rispettivamente i byte basso ed il byte alto del parametro letto da CHKCOM ($AEFD) e FRMNUM ($AD8A). Nel nostro caso, dato che entrambi i parametri sono minori di 256 (cioè, occupano un solo byte) ci siamo preoccupati solo del byte basso. Pertanto la locazione $15 non compare mai. Un'altra curiosità. Un modo «standard», senza uso di Locate per centrare una stringa, potrebbe essere il seguente:

```
DISASSEMBLATO DI LOCATE -

C000 JSR $C01B
C003 STA $FB
C005 CMP #$19
C007 BPL $C027
C009 JSR $C01B
C00C STA $FC
C00E CMP #$28
C010 BPL $C027
C012 CLC
C013 LDX $FB
C015 LDY $FC
C017 JSR $FFF0
C01A RTS
C01B JSR $AEFD
C01E JSR $AD8A
C021 JSR $B7F7
C024 LDA $14
C026 RTS
C027 LDX #$0E
C029 JSR $A437
C02C RTS
```

```
10 PRINT "(CLR)"
20 A$="STRINGA CENTRATA"
30 L%=(40-LEN(A$))/2
40 FOR I=1 TO L%
50 PRINT "(CRSR RIGHT)";
60 NEXT I
70 PRINT A$
80 END
```

Questo programma gira in circa 92 ms. Il programma analogo che usa Locate, invece, gira in circa 68 ms. Questo significa che il metodo standard è circa il 35% più lento, in questo caso.

Alfabeto Morse

di Luciano Rosa - Potenza

Il programma Alfabeto Morse è nato per la mia necessità di dovere imparare questa forma di comunicazione. A tal fine ho realizzato questo programma che si divide in più parti, tutte accessibili tramite un menu. Si comincia con l'assimilazione del «tratto» e del «punto». Con l'opzione ESERCITAZIONE, che compare nel menu principale, si possono eseguire i classici suoni, uno più prolungato per indicare il tratto, uno breve per il punto. Per semplicità, al punto corrisponde il tasto MENO ed al tratto il tasto PIÙ. Assimilato questo sistema si può passare alla TABELLA MORSE (opz. 4). Per facilitarne l'apprendimento e per verificare che l'utente abbia imparato correttamente useremo l'opzione 2, DECIFRAZIONE, dove immetteremo un carattere nel codice Morse. Per semplicità dell'utente, prima dell'immissione del carattere bisogna dare uno «start», cosa possibile tramite la pressione di un tasto qualsiasi, anche quelli usati per la trasmissione. Si consigliano questi ultimi in quanto tra la partenza e il carattere stesso c'è un limite di tempo di circa un secondo. Questo è necessario affinché il computer decifri rapidamente il codice. L'immissione del codice è semplificata poiché, oltre al suono, comparirà sul video la sequenza dei segni trasmessi istante per istante. Composta la serie di simboli formanti una lettera, il computer visualizzerà la lettera corrispondente. Inoltre, se la sequenza dei simboli non forma alcuna lettera, il computer visualizzerà la

```
asdfg   jytvn   ikpkp
fgdgv   refwm   jbnbf
htgdc   nhvhg   wqgqk
ghdfv   htgfh   hjkiw
hyfgn   uljsc   iiopw
yhfvb   ioisa   vxvbc
tghgh   wdnmo   erwxv
yijkh   fbgff   gnhjy
hgfgf   bgbfv   ikjhg
hthyh   bvbxf   ijoll
kiilk   rgevb   lkjxz
mjmhm   wqebv   vbvbv
bcdtu   qweuy   dgdsf
hjrbv   vfdvd   vnvfs
hnhgg   gregb   qdfvf
mnbnm   dfdfw   vbfcg
ffdfd   ijomp
```

Tabella 1 - Contenuto del file DATI

```
djdur   ofjds   13364
hdued   sjhfd   10384
djrus   adjkj   56373
oeurp   lucjd   37464
vcndj   ianof   57198
qrwye   fjrid   54764
ejdkf   fjrug   57384
eizvx   rkfir   47583
zbxnd   37342   68481
qyeix   27928   47563
eudjs   42030   57583
eudjc   00971   67584
qiwid   36274   49584
eiqid   32864   57584
cnqow   16838   68497
djwio   37482   58495
rerwd   48473
```

Tabella 2 - Contenuto del file DATI2

frase «non esiste». Per uscire dalle diverse opzioni basta la pressione di un tasto comunque sempre menzionato nel programma. Un'altra parte del programma, la più interessante, è quella selezionabile tramite l'opzione 3: TRADUZIONE MORSE. Selezionata questa funzione, sullo schermo apparirà un menu secondario. La TRASFORMAZIONE DIRETTA permette all'utente di ascoltare il segnale Morse delle diverse lettere e simboli disponibili, selezionabili tramite la pressione del tasto corrispondente alla lettera. Ritornati al menu secondario (selezionabile con l'opzione 3 da quello principale) è visibile il LIVELLO DIFFICOLTÀ (opz. 3). Selezionandolo, il C64 chiederà la velocità da 1 a 4. Per velocità si intende il tempo che corre tra un simbolo e l'altro di una lettera. Maggiore sarà il numero scelto, minore sarà la velocità. L'ultima opzione selezionabile tramite il menu secondario è l'ESERCITAZIONE DECIFRAZIONE, che metterà alla prova l'abilità dell'utente. Fatta questa scelta verrà chiesto il tipo di parole da leggere; DATI comprende 50 gruppi di 5 lettere, mentre DATI2 comprende 25 gruppi di lettere e 25 di numeri. Dopo aver operato la scelta desiderata il C64 chiederà di inserire il disco dati (allegati al disco del programma stesso), e di battere RETURN. Sullo schermo appariranno in alto le seguenti opzioni: V sta per verifica gruppo e visualizza sequenza di lettere o numero che il computer ha decifrato; S sta per altro gruppo di lettere; il 64 ne sceglierà un altro ed emetterà i suoni che contraddistinguono i caratteri formanti il gruppo.

```
1000 PRINT"(SC)"
1020 OPEN6,8,6,"@0:NOME,S,W"
1025 INPUT A$
1030 IF A$="*" THEN CLOSE6:END
1035 PRINT#6,A$
1040 GOTO1025
```

N.d.r. - Per usare la funzione ESERCITAZIONE DECIFRAZIONE del programma «Morse» bisogna creare su disco i due file sequenziali DATI e DATI2 il cui contenuto è riportato rispettivamente nelle tabelle 1 e 2. Per la creazione di tali file potete utilizzare questo piccolo programma.

Alfabeto Morse

```
50 LDF=50:DIMD$(50):GOTO100
60 A=53280:POKEA,14:POKEA+1,6
61 PRINT"(CLR)(LBLU)":END
70 FORX=1TO3000:NEXTX:RETURN
100 REM ------------------------
110 REM --SIMULAZIONE LINGUAGGIO--
120 REM ---------MORSE-----------
130 REM
140 REM -----BY  LUCIANO ROSA-----
150 REM
160 PRINT"(CLR)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(GRN)"
170 A=53280:POKEA,0:POKEA+1,0
180 PRINTTAB(5)"SIMULAZIONE LINGUAGGIO MORSE":A$="(HOME)(DOWN)(DOWN)(DOWN)(
DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)"
185 A$=A$+"(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGH
T)DI  LUCIANO ROSA"
190 FORX=1TOLEN(A$):PRINTLEFT$(A$,X);:FORY=1TO85:NEXTY:NEXTX:GOSUB70
200 PRINT"(CLR)(DOWN)(DOWN)(RGHT)(RGHT)OPZIONI:"
210 PRINT"(DOWN)(DOWN)(DOWN)(RGHT)1.ESERCITAZIONE"
220 PRINT"(DOWN)(RGHT)2.DECIFRAZIONE"
230 PRINT"(DOWN)(RGHT)3.TRADUZIONE MORSE"
232 PRINT"(DOWN)(RGHT)4.TABELLA MORSE"
235 PRINT"(DOWN)(RGHT)5.FINE"
240 GETB$:B=VAL(B$):IFB<1ORB>5THEN240
250 ONBGOTO260,360,690,1080,60
260 PRINT"(CLR)(RGHT)(RVS) + (OFF) = -     (RVS) - (OFF) = .";
265 PRINT"     (RVS) X (OFF) = FINE"
270 GETB$:IFB$=""+"THENTE=150:GOSUB300
280 IFB$="-"THENTE=70:GOSUB300
285 IFB$="X"THEN200
290 GOTO270
300 SI=54272:FL=SI:FH=SI+1:TL=SI+2
310 TH=SI+3:A=SI+4:H=SI+6:L=SI+24
320 POKEL,15:POKEA,16:POKEH,4*16+4
330 W=SI+4:POKEFH,29:POKEFL,69:POKEW,17
340 FORI=1TOTE:NEXTI:POKEW,0:POKEA,0
345 POKE H,0:RETURN
360 PRINT"(CLR)":REM DECIFRAZIONE
361 PRINT"(HOME)(RGHT)(RVS) + (OFF) = -     (RVS) - (OFF) = .";
365 PRINT"     (RVS) X (OFF) = FINE":PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(RG
HT)(RGHT)";
366 GETB$:IFB$=""THEN366
367 FORM=1TO80:GETB$
375 IFB$="+"THENB$="-":D$=D$+B$:GOTO400
380 IFB$="-"THENB$=".":D$=D$+B$:GOTO420
385 IFB$="X"THENU$="":GOTO200
390 NEXTM:IFB$=""THEN500
400 PRINTB$;:TE=150:GOSUB300:GOTO367
420 PRINTB$;:TE=70:GOSUB300:GOTO367
500 RESTORE:FORIL=1TO42
510 READF$,G$
520 IFD$=F$THEND$="":GOTO540
530 NEXTIL:GOTO550
540 U$="(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(
RGHT)(RGHT)(RGHT)(RGHT)(RGHT)":U$=U$+U$+U$
541 F=F+1:V$=LEFT$(U$,F)
542 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN
)(DOWN)(DOWN)"+V$;
543 PRINTG$;
544 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)                    "
545 GOTO361
550 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN
)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)NON ESISTE'"
555 GOSUB70:D$=""
560 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN
)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)                    "
570 GOTO361
600 DATA"A","-.","B","-...","C
605 DATA"-.-.","D","-..","E",".","F
610 DATA"..-.","G","....","H","..","I
615 DATA".---","J","-.-","K",".-..","L
620 DATA"--","M","-.","N","---","O
625 DATA".--.","P","--.-","Q",".-.","R
630 DATA"...","S","-","T","..-","U
635 DATA"...-","V",".--","W","-..-","X
640 DATA"-.--","Y","--..","Z
650 DATA".-.-.-","."
655 DATA"--..--",","
660 DATA".----","1","..---","2"
665 DATA"...--","3","....-","4"
670 DATA".....","5","-....","6"
675 DATA"--...","7","---..","8"
676 DATA"----.","9","-----","0"
680 DATA"........","(DEL)(DEL)","-..-.","-.--."
690 A$="----------":A$=A$+A$
700 PRINT"(CLR)(DOWN)(DOWN)(RGHT)(RGHT)OPZIONI:"
710 PRINT"(DOWN)(DOWN)(DOWN)(RGHT)1.TRASFORM.DIRETTA"
720 PRINT"(DOWN)(RGHT)2.ESERC.DECIFRAZIONE"
725 PRINT"(DOWN)(RGHT)3.LIVELLO DIFFICOLTA'"
730 PRINT"(DOWN)(RGHT)4.RITORNO MENU'"
740 GETB$:B=VAL(B$):IFB$="_"THEN200
750 IFB<1ORB>4THEN740
760 ONBGOTO770,910,1300,200
770 PRINT"(CLR)(RGHT)(RVS) _ (OFF) = USCITA(DOWN)":PRINTA$
780 GETB$:IFB$=""THEN780:
785 IFB$="_"THEN200
790 RESTORE:FORI=1TO42:READF$,G$
800 IFB$=G$THENF=0:GOTO815
810 NEXTI:GOTO780
815 GOSUB820:GOTO780
820 F=F+1:A$="(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(
RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)"
825 A$=A$+A$:WK$=LEFT$(A$,F)
830 PRINT"(HOME)(DOWN)(DOWN)(DOWN)"+WK$;G$;
835 LU=LU+1
840 IFLU>LEN(F$)THENLU=0:GOTO690
850 H$=MID$(F$,LU,1)
855 IFH$="-"THENTE=150:GOSUB300:GOTO870
860 IFH$="."THENTE=55:GOSUB300
870 FORX=1TOLDF:NEXTX
875 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN
)(DOWN)(DOWN)(DOWN)";F$
880 FORX=1TO200:NEXTX:GOTO835
890 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)    ":PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(
DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)          "
900 RETURN
910 PRINT"(CLR)(DOWN)(DOWN)(RGHT)(RGHT)(RGHT)OPZIONI:"
911 PRINT"(DOWN)(DOWN)(DOWN)(RGHT)(RGHT)1.DATI"
912 PRINT"(DOWN)(RGHT)(RGHT)2.DATI2"
913 PRINT"(DOWN)(DOWN)(DOWN)(DOWN)(RGHT)SCEGLI IL FILE LEGGERE"
914 GETFL$:FIL=VAL(FL$):IFFIL<1ORFIL>2THEN914
915 IFFIL=1THENFL$="DATI"
916 IFFIL=2THENFL$="DATI2"
920 PRINT"(DOWN)(DOWN)(DOWN)(DOWN)(RGHT)INSERISCI IL DISCHETTO CONTENENTE"
925 PRINT"(RGHT)I DATI (LETTURA 'DATI') E BATTI "
930 PRINT"(RGHT)(RVS) RETURN (OFF)"
935 GETA$:IFA$=""THEN935
940 IFA$<>CHR$(13)THEN935
945 PRINT"(CLR)":A$="----------------------"
946 A$=A$+A$:PRINT"(HOME)(DOWN)"A$
947 PRINT"(HOME)(RGHT)V=VERIF.GRUPPO  S=ALTRO GRUPPO  N=STOP"
950 OPEN1,8,2,FL$+",S,R"
955 FORI=1TO50:INPUT#1,D$(I)
960 NEXTI:CLOSE1
970 AB=INT(50*RND(1))+1
980 C$=D$(AB):MI=0
985 MI=MI+1
990 B$=MID$(C$,MI,1)
995 RESTORE:FORI=1TO42:READF$,G$
1000 IFB$=G$THENF=0:GOTO1020
1010 NEXTI:GOTO1030
1020 GOSUB820:GOTO1030
1030 IFMI<LEN(D$(AB))THENGOTO1070
1040 GETNM$:IFNM$=""THEN1040
1045 IFNM$="S"THEN970
1046 IFNM$="V"THEN1060
1050 GOTO200
1060 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOW
N)(DOWN)(DOWN)(DOWN)";C$:GOTO1040
1070 FORX=1TO250:NEXTX:GOTO985
1080 PRINT"(CLR)";:REM TABELLA
1090 PRINT"┌──────────┐"
1100 PRINT"│"
1101 PRINT"(HOME)(DOWN)"TAB(3)"TABELLA MORSE"
1110 PRINT"├──────┬──────┤"
1115 FORX=1TO19
1120 PRINT"│      │      │"
1135 NEXTX
1140 PRINT"│      │      │"
1150 PRINT"└──────┴──────┘"
1155 PRINT"(HOME)(DOWN)(DOWN)"
1160 RESTORE:FORI=1TO20:READF$,G$
1170 PRINTTAB(3)G$;TAB(9)F$:NEXTI
1180 PRINT"(HOME)";
1190 PRINTTAB(19)"──────────┐"
1200 PRINTSPC(38)"│"
1210 PRINTTAB(19)"┼──────┬──────┤"
1220 FORX=1TO20
1230 PRINTSPC(26)"│"SPC(11)"│":NEXTX
1240 PRINTTAB(19)"└──────┴──────┘"
1255 PRINT"(HOME)(DOWN)(DOWN)"
1260 FORI=21TO40:READF$,G$
1270 PRINTTAB(22)G$;TAB(28)F$:NEXTI
1280 GETA$:IFA$<>CHR$(13)THEN1280
1290 GOTO200
1300 INPUT"(DOWN)(DOWN)(RGHT)LIVELLO (1-4)    1(LEFT)(LEFT)(LEFT)";LD$
1400 LDF=VAL(LD$):IFLDF<1ORLDF>4THEN1300
1410 IFLDF=1THENLDF=50
1420 IFLDF=2THENLDF=100
1430 IFLDF=3THENLDF=210
1440 IFLDF=4THENLDF=320
1450 GOTO700
```

Il programma oltre al suono in Morse del singolo carattere, visualizzerà più in alto la lettera corrispondente e più in basso il codice in punti e tratti. L'opzione N dà lo stop, ritorna cioè al menu principale. Superate le cinque prove l'utente è sicuro di aver imparato bene e per sempre l'affascinante codice Morse e potrà operarlo tranquillamente. Inoltre il programma, data la versatilità, può essere usato da chi già conosce il codice e vuole provare la propria abilità.

I valori che possono essere modificati dall'utente per personalizzare il programma, sono i seguenti:

▼

Per il menu secondario:

	RIGHE	VALORI INIZIALI
DIFFICOLTA' LIVELLO	1410-1440	LDF=50-320
DURATA DEL TRATTO E PUNTO	855- 860	TE=150-55
TEMPO FRA CARATTERE E CARATT.	880	X=1TO200

Per il menu primario:

	RIGHE	VALORI INIZIALI
DURATA TRATTO E PUNTO DI ESERC.	270-280	TE=150-70
DURATA TRATTO E PUNTO DI DECIF.	400-420	TE=150-70
TEMPO DI ACCETTAZ. DECIFRAZ.	367	M=1TO80

N.B. Variando il tempo di accettazione varia l'intervallo di tempo fra la partenza ed il codice.

ERRATA CORRIGE

Segnaliamo alcune imprecisioni contenute nel listato dell'utility POINTER pubblicata sul n. 75 di MC alle pagine 241 e 242.

Si tratta di 4 numeri sbagliati nelle linee DATA 110, 220 e 310 che ripubblichiamo correttamente:

```
110 DATA 00,07,07,FA,CE,02,0C,D0,5B,AD
220 DATA B9,D7,11,B0,0A,CD,08,0C,F0,10
310 DATA B9,D6,11,38,E9,01,99,D6,11,C9
```

Inoltre, nel programma dimostrativo, le linee 100 e 180 sono corrette nel modo seguente:

```
100 FOR J=3 TO 25:READ J%:POKE 3648+J,J%:NEXT
180 X1%=RSPPOS(1,0)-24:Y1%=RSPPOS(1,1)-50
```

Programma menu per C64 + drive

di Fabrizio Lanni - Cassino (FR)

Questo programma metterà un po' di ordine nei vostri dischetti. Il tutto lo ottiene archiviando su un file sequenziale tutti i nomi di programma che incontra sul disco tranne file sequenziali, delete, relativi e il primo programma presente sul disco che di norma è il programma menu stesso. Una volta caricato e dato il run il programma controlla che il file sequenziale contenente i nomi dei programmi sia stato creato; se esso non è presente allora provvede a crearlo. Dopo questo passaggio appariranno sullo schermo 15 programmi per volta e, per vederne altri 15, basterà premere il tasto «F1». Sottolineo che verranno visualizzati altri programmi solo se que-

```
100 DIMN$(100),L$(100):A=1:DIMJ$(16):POKE53280,0:POKE53281,0:POKE646,1
200 OPEN8,8,8,"CCCCCCCCCC,S,R":OPEN15,8,15:INPUT#15,EN$:IFEN$<>"62"THEN2400
400 J=1:FORX=1TO90:N$(X)="":L$(X)="":NEXTX
500 CLOSE8:CLOSE15:OPEN1,8,0,"$0":FORI=1TO30:GET#1,A$:B$=B$+A$:NEXT:Z$=MID$
(B$,7,16)
600 GET#1,A$,B$
700 FORX=1TO2:GET#1,A$,B$:NEXTX:I=0
800 IFA$<>""THENI=ASC(A$)
900 IFB$<>""THENI=I+ASC(B$)*256
1000 B$=STR$(I):BL$=RIGHT$(B$,LEN(B$)-1):A$=""
1100 GET#1,B$:IFST<>0THEN1800
1200 IFB$<>CHR$(34)THEN1100
1300 GET#1,B$:IFB$<>CHR$(34)THENA$=A$+B$:GOTO1300
1400 GET#1,B$:IFB$=CHR$(32)THEN1400
1500 GET#1,B$:IFB$="E"THEN2200
1600 IFB$<>""THEN1500
1700 N$(J)=A$:L$(J)=BL$:J=J+1:GOTO700
1800 CLOSE1
1900 OPEN15,8,15,"S0:CCCCCCCCCC":CLOSE15:OPEN9,8,9,"CCCCCCCCCC,S,W":PRINT#
9,(J-1)
2000 PRINT#9,Z$:FORO=1TOJ:PRINT#9,N$(O)
2100 PRINT#9,L$(O):NEXTO:PRINT#9,I:CLOSE9:GOTO2600
2200 IFB$<>""THENGET#1,B$:GOTO2200
2300 A$="":GOTO700
2400 INPUT#8,F:INPUT#8,Z$:FORO=1TOF:INPUT#8,N$(O)
2500 INPUT#8,L$(O):NEXTO:INPUT#8,I:CLOSE8:CLOSE15
2600 B$="":PRINT"(CLR)┌CCCC┬CCCCCCCCCC─CCCCCCC──┬CCCCC┬CCC┐"
2700 PRINT"B(RVS)NUM.(OFF)B(RVS)    NOME PROGRAMMA     (OFF)B(RVS)BLOCKS(OFF)
B(RVS)N.D(OFF)B"
2800 PRINT"├CCCC┼CCCCCCCCCC─CCCCCCCC┼CCCCC┼CCC┤"
2900 FORX=1TO15:PRINT"B     B"SPC(21)"B      B    B":NEXTX
3000 PRINT"└CCCC┴CCCCCCCCCC─CCCCCCCC┴─CCCCC┴    B"
3050 PRINT"B     B"SPC(21)"B      B    B"
3060 PRINT"┌CCCC┬CCCCCCCCCC─CCCCCCCC┬─CCCCC┬CCC┐"
3100 PRINT"B     B"SPC(21)"B      B    B"
3150 PRINT"└CCCC┴CCCCCCCCCC─CCCCCCCC┴─CCCCC┴CCC┘"
3160 PRINT"B     B"SPC(21)"B      B    B"
3200 PRINT"┌CCCC┬CCCCCCCCCC─CCCCCCCC┬─CCCCC┬CCC┐(HOME)(DOWN)(DOWN)
3300 FORC=ATOA+14:PRINT"(WHT)"SPC(4-(LEN(STR$(C)))):ABS(C);
3350 PRINTTAB(9)N$(C)TAB(32-LEN(L$(C)))L$(C)
3400 NEXT
3500 PRINT"(HOME)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOW
N)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(RGHT)(RVS) F1 (OFF
)(RGHT)(RGHT)CONTINUA  LA LISTA":PRINT
3550 PRINT"(RGHT)(RVS) F3 (OFF)(RGHT)(RGHT)PER TERMINARE"
3600 PRINT"(DOWN)(RGHT)(RVS) F7 (OFF)(RGHT)(RGHT)CREA LA ";
3700 PRINT"(RGHT)(RGHT)DIRECTORY(HOME)(DOWN)(DOWN)":PRINTSPC(35)"(RVS)    (O
FF)"
3800 PRINT"(HOME)(DOWN)(DOWN)(DOWN)":FORS=1TO16:J$(S)=MID$(Z$,S,1):PRINTSPC
(35)"(RVS) "J$(S)" (OFF)":NEXTS
3900 PRINTSPC(28)"(UP)(RVS)BLOCKS(RGHT)(OFF)(RGHT)(RGHT)(RGHT)"SPC(30)"(DOW
N)(RVS)FREE. (OFF)":I;SPC(5-LEN(STR$(I)));"(LEFT)(LEFT)(LEFT)(LEFT)(LEFT)(L
EFT)B"
3950 PRINTSPC(28)"(DOWN)(RVS)PRG N.(RGHT)(OFF)"
4000 PRINT"(LEFT)(LEFT)(LEFT)(LEFT)(LEFT)(LEFT)B(RGHT)(RGHT)(RGHT)(RGHT)(RG
HT)";
4100 GETA$:IFA$=""THEN4100
4200 IFASC(A$)=13THEN5000
4300 IFA$="(F1)"THEN5200:REM F1
4400 IFA$="(F7)"THEN400:REM F7
4500 IFA$="(F3)"THENPRINT"(CLR)(DOWN)(DOWN)(DOWN)RUN(HOME)":END:REM F3
4600 IFA$<"0"ORA$>"9"THEN4100
4650 B$=B$+A$:B=VAL(B$)
4670 IFB>90THEN2600
4700 IFN$(B)=""THEN2600
4800 PRINT"(UP)"SPC(35)"(RVS) "B$"(OFF)"
4900 GOTO4100
5000 PRINT"(F3)(CLR)(DOWN)(DOWN)(DOWN)LOAD"CHR$(34)N$(B)CHR$(34)",8,1"
5100 PRINT"(HOME)":POKE198,4:POKE631,13:POKE632,82:POKE633,213:POKE634,13:E
ND
5200 IFA=76THENA=1:GOTO2600
5300 IFN$(A+15)<>""THENA=A+15:GOTO2600
5400 A=1:GOTO2600
```

Output video del programma menu per Commodore 64.

NUM	NOME PROGRAMMA	BLOCKS	N D
1	PROGRAMMA 1	XX	N
2	PROGRAMMA 2	XX	O
3			M
4			E
5			
6			D
7			I
8			S
9			C
10			O
11			
12			
13			
14			
15	PROGRAMMA 15	XX	
F1	CONTINUA LA LISTA	BLOCKS	
F3	PER TERMINARE	FREE	XXX
F7	CREA LA DIRECTORY	PRG N.	

sti saranno presenti sul dischetto. In tutto ne potranno essere visualizzati un massimo di 90. Sulla stessa schermata in cui apparirà la lista verrà anche visualizzato il nome del disco (a destra), i blocchi liberi sul disco (a destra in basso) la lunghezza di ogni programma (a destra di ciascuno) e, più in basso, il numero di programma digitato. Una volta scelto il programma da caricare, dovrete digitare il numero corrispondente al programma desiderato. Per uscire dall'elaborazione premere «F3». Per ricreare l'archivio, ad esempio quando sono stati aggiunti altri file sul disco, basterà premere il tasto «F7».

Elenco variabili

TN$(Nome dei programmi.
L$(Lunghezza programma in blocchi.
J$(Nome del disco carattere per carattere.
Z$ Nome dei programmi, deriva da B$ alla riga.
I Blocchi liberi su disco.
C Numero del programma.
BL$ Come L$(.
A$ Carattere letto nella directory, verrà caricato nella variabile B$ per dare il nome del programma e la sua lunghezza.
A Numero assegnato al primo programma che appare su video.

Notizie utili:

Il programma Menu è adatto per il Commodore 64 e Commodore 128 in modo 64 + drive. MK