

dossier

TEXAS 99/4A

MENSILE - SPED. AB POST. GR. III/70 - N° 3 - FEBBRAIO 1985 - EDIZIONI CO STA - PISA



**PROGRAMMI PER
TI 99/4A**

L. 2000

dossier

TEXAS

Mensile

di giochi e programmi
per TEXAS 99/4A

Direttore Responsabile:

Marco Bindi

Direttore Tecnico:

Gianroberto Negri

Edizioni **CO.STA** srl

via Mazzini 128

56100 PISA

Registrazione al Tribunale

di Pisa n. 20 del 20/10/84

Spediz. in abbon. post.

gruppo III / 70

TEXAS 99/4A

il mensile per realizzare la tua
Biblioteca di Software in tre
sezioni:

— **Giochi**

— **Programmi personali**

— **Programmi di utilità**

Lire 2000

I numeri arretrati, a lire 4000,
possono essere richiesti sen-
za oneri di spedizione con pa-
gamento anticipato in contan-
ti, o tramite assegno o vaglia
postale intestato alla Società
editrice.

Impianti:

Cencograf

Stampa:

Grafika 78 - Pioltello (MI)

Distribuzione:

Messaggerie Periodici (MI)

dossier

Ogni mese in edicola
il **DOSSIER**
per il tuo computer

- **COMMODORE DOSSIER**
- **SINCLAIR DOSSIER**
- **TEXAS 99/4A DOSSIER**

**giochi, programmi di utilità
e programmi personali per
realizzare una fantastica
biblioteca software**



UTILITY



**PROGRAMMI
PERSONALI**



GIOCHI

CODIFICA SIMBOLI

TEXAS 99/4A



ANIMAZIONE



Avete mai provato il desiderio di realizzare dei cartoni animati?... Nel caso la Vostra risposta fosse affermativa, questo programma Ve ne darebbe la possibilita', fornendoVi come attore niente meno che Charlot!!!... Intendiamoci quanto in questa sede e' proposto non e' una novita', perche' la realizzazione di cartoni animati facendo uso del Computer e' attualmente una tecnica a cui molti creatori ricorrono. E' comunque una novita' il farlo con un computer che molti giudicano poco piu' di un giocattolo e che in realta' a ben poco da invidiare anche a modelli piu' grossi e ben piu' costosi. Quindi questa e' una proposta per continuare a percorrere questa interessante ed istruttiva via della creazione di film d'animazione facendo uso del computer ed in questo caso del TI99/4A.

Ma veniamo alla descrizione del programma e del suo utilizzo. Esso e' strutturato in maniera da accettare una intera riga di comandi atti a produrre la particolare situazione da Voi scelta. La riga di comandi di cui e' riportato un esempio a

TEXAS 99/4A

video viene letta, scandendo un carattere e quindi un comando alla volta e vengono così richiamate le routine interessate al particolare movimento che il nostro Charlott effettuerà. Sono disponibili molti movimenti ed essi sono ampiamente elencati sul video del Vostro teleschermo, una volta lanciato il programma. Le routine interessate al particolare movimento sono evidenziate mediante delle parole chiave REM ed il nome con cui sono elencate a video.

Il programma necessita del BASIC ESTESO per poter funzionare.

```

100 !*****
110 !* ANIMAZIONE *
120 !*****
130 !-----
140 !*****
150 RANDOMIZE
160 CALL CLEAR
170 CALL CHAR(100,"01030B04010001060E1E3F3F7
7776767C0E0E8104000B0607078FCECECECE
CEC")
180 CALL CHAR(104,"6F6F7F9F2F2F2F2E2E2E4E4
E4E9EFEFCEFCFCFBF0F0F070707070707070
B7E")
190 CALL MAGNIFY(3)
200 CALL SPRITE(#1,100,2,140,200,0,0,#2,104,
2,156,200,0,0)

```

TEXAS 99/4A

```

210 PRINT TAB(8);" ANIMAZIONE " :: PRINT "-"
-----"
220 PRINT "FORMATO COMANDI:NUMERI DA 1 A 9"
230 PRINT "PRIMA CIFRA:SITUAZIONE 1 FABBRIC
A 2 VIA"
240 PRINT "SECONDA : COLORI DEL TI99/4A DA 1
A 9"
250 PRINT "SEGUENTI : AZIONI: 1 SALTO 2 MAR
CIA 3 BASTONE 4 INDIETRO 5 SCIVOL
ONE 6 SALUTO 7 PIEGARSI 8 SALIRE 9 SCEND
ERE"
260 FOR I=1 TO 9 :: PRINT :: NEXT I
270 !-----analisi
280 DISPLAY AT(23,1):"AZIONE?&EX:27882263236
526&"
290 ACCEPT BEEP VALIDATE(DIGIT):C$
300 CALL CLEAR
310 GOSUB 1540
320 CALL SPRITE(#20,120,9,5,100,0,15,#19,92,
16,10,70,0,8,#18,92,16,15,120,0,5,#1
7,120,3,1,200,0,3)
330 CO=VAL(SEG$(C$,2,1))
340 CALL COLOR(5,CO,1)
350 CALL COLOR(7,CO,1)
360 ON VAL(SEG$(C$,1,1))GOSUB 510,700
370 GOSUB 570
380 FOR X=3 TO LEN(C$)
390 C=VAL(SEG$(C$,X,1))
400 ON C GOSUB 1010,1060,1180,1220,1260,1360
,1400,1450,1520
410 GOSUB 470
420 NEXT X
430 CALL SOUND(10,1000,0)

```

TEXAS 99/4A

```
440 CALL CLEAR
450 END
460 !-----normalita
470 CALL CHAR(100,"01030B04010001060E1E3F3F7
7776767C0E0EB10400080607078FCECECECE
CEC")
480 CALL CHAR(104,"6F6F7F9F2F2F2F2E2E2E2E4E4
E4E9EFEFCEFCFC8F0F0F0707070707070707
87E")
490 RETURN
500 !-----fabbrica
510 FOR I=11 TO 17
520 FOR J=2 TO 16
530 CALL HCHAR(I,J,86)
540 NEXT J :: NEXT I
550 CALL HCHAR(10,2,87,15)
560 RETURN
570 !-----strada
580 CALL HCHAR(20,1,63,24)
590 CALL HCHAR(23,1,63,22)
600 CALL HCHAR(20,28,63,3)
610 CALL HCHAR(23,26,63,4)
620 CALL HCHAR(19,29,62)
630 CALL HCHAR(24,25,62)
640 CALL HCHAR(18,30,62)
650 CALL HCHAR(24,22,62)
660 CALL HCHAR(20,25,62)
670 CALL HCHAR(19,26,62)
680 CALL HCHAR(18,27,62)
690 RETURN
700 !-----case
710 FOR R=1 TO 24 STEP 9
720 H=INT(RND*8)+5
730 CALL VCHAR(H,R,65,19-H)
```

TEXAS 99/4A

```
740 CALL VCHAR(H-2,R+8,64,19-H)
750 CALL VCHAR(H,R+6,64,19-H)
760 CALL HCHAR(H,R+1,66)
770 CALL HCHAR(H,R+6,64)
780 CALL HCHAR(H,R+2,66,4)
790 CALL HCHAR(19,R+1,66,5)
800 CALL HCHAR(H-3,R+3,67,5)
810 CALL HCHAR(H-1,R+1,68)
820 CALL HCHAR(H-2,R+2,68)
830 CALL HCHAR(H-1,R+6,68)
840 CALL HCHAR(H-2,R+7,68)
850 CALL HCHAR(18,R+6,71)
860 CALL HCHAR(17,R+7,68)
870 CALL VCHAR(15,R+2,64,4)
880 CALL VCHAR(15,R+3,65,4)
890 CALL HCHAR(14,R+2,67,2)
900 FOR F=1 TO 4 STEP 3
910 FOR CF=1 TO 9 STEP 3
920 IF H+CF+1>=14 THEN 980
930 CALL HCHAR(H+CF,R+F,66,2)
940 CALL HCHAR(H+CF+1,R+F,67,2)
950 CALL VCHAR(H+CF,R+F-1,65,2)
960 CALL VCHAR(H+CF,R+F+1,64,2)
970 NEXT CF
980 NEXT F
990 NEXT R
1000 RETURN
1010 !-----SALUTO
1020 CALL CHAR(100,"81838B84818081868E9EBFBF
77776767C0E0E810400080607078FCECECEC
ECEC")
1030 CALL CHAR(104,"6F7F9F0F0F0F0E00000000
00000000FCFCFCFBF7F7FF7D000000000000
```

TEXAS 99/4A

```

0000")
1040 CALL SOUND(10,1000,0)
1050 RETURN
1060 !-----MARCIA
1070 S=1
1080 CALL MOTION(#1,0,-3*S,#2,0,-3*S)
1090 FOR I=1 TO 3
1100 CALL CHAR(107,"7038381C1E0E0C38")
1110 CALL CHAR(105,"0E1C1C1C1CDCFC3C")
1120 CALL CHAR(107,"38383838383878F8")
1130 CALL CHAR(105,"2E2E2E4E4E4E9EFE")
1140 CALL SOUND(30,-6,15)
1150 NEXT I
1160 CALL MOTION(#1,0,0,#2,0,0)
1170 RETURN
1180 !-----BASTONE
1190 CALL CHAR(100,"81838B84818081868E9EBFBF
77776767")
1200 CALL CHAR(104,"6F7F9F0F0F0F0F0E0E0E0E0E
0E0E1EFE")
1210 RETURN
1220 !-----INDIETRO
1230 S=-1
1240 GOSUB 1080
1250 RETURN
1260 !-----SCIVOLONE
1270 GOSUB 1400
1280 CALL CHAR(100,"000000000000000000000001
020408100000000000000000304888000000
0000")
1290 CALL CHAR(104,"2F0E7C6D6C6C6C6C6C666677
731B1F0FE0003C7DFE0020C01C3C3CFEF7F3
FFFF")

```


TEXAS 99/4A

```

1300 CALL MOTION(#1,0,-3,#2,0,-3)
1310 FOR S=20 TO 0 STEP -1
1320 CALL SOUND(-10,-6,S)
1330 NEXT S
1340 CALL MOTION(#1,0,0,#2,0,0)
1350 RETURN
1360 !-----SALUTO
1370 CALL CHAR(102,"COEEEA164303837F7E7CF8E0
EOEOEOEO")
1380 CALL CHAR(106,"FOFOFOFOFOFOFOFO70")
1390 RETURN
1400 !-----PIEGARSI
1410 CALL CHAR(100,"0000000000000000001030B04
010001060000000000000000C0E0EB104000
B060")
1420 CALL CHAR(104,"0E1E3F3F777767676F6F7F9F
2F2F2F2E7078CECECECECECFCFCFBF7F7F
F7D")
1430 CALL SOUND(10,110,0)
1440 RETURN
1450 !-----SALIRE
1460 S=1
1470 CALL MOTION(#2,-2*S,0,#1,-2*S,0)
1480 GOSUB 1090
1490 CALL MOTION(#1,0,0,#2,0,0)
1500 RETURN
1510 !-----SCENDERE
1520 S=-1 :: GOSUB 1470
1530 RETURN
1540 !-----DEFINIZIONE
1550 CALL CHAR(62,"0912244890204080000000007
F01017F")
1560 CALL CHAR(64,"B080B080B080B080010101010

```

TEXAS 99/4A

```

1010101FF000000000000000000000000000000
OFF")
1570 CALL CHAR(68,"0102040810204080FF8080808
0808080FF010101010101018182848890A0C
080")
1580 CALL CHAR(86,"FFFFFFFFFFFFFFFF"):: CALL
CHAR(87,"0103070FFFFFFFFF")
1590 CALL CHAR(92,"0F3F3F7FFF7F3F1F00000000
0000000F8FCFEFEFEFEFEFC00000000000000
000")
1600 CALL CHAR(86,"FFFFFFFFFFFFFFFF"):: CALL
CHAR(87,"0103070FFFFFFFFF")
1610 CALL CHAR(120,"C0C1E3FF7F03070F00000000
00000000E09189FFDF9C0000000000000000
0000")
1620 RETURN

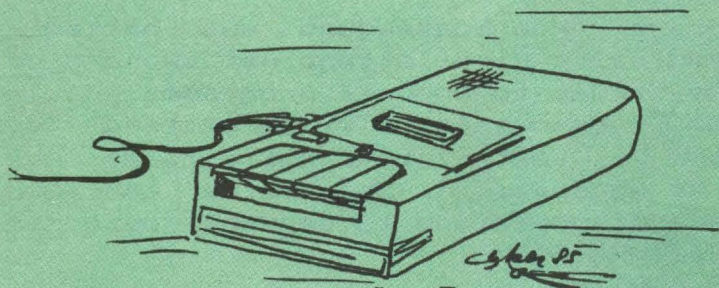
```

**PASS. VARIABILI**

Uno dei problemi di chi realizza applicazioni sofisticate, utilizzando il registratore a nastro per registrare i dati e' il passaggio dei valori, contenuti nelle variabili tra un programma ed un altro. Questo programma d'utilita' provvede appunto a fare cio'. Il suo utilizzo e'

TEXAS 99/4A

molto semplice basta lanciarlo e caricare il programma a cui passare i valori delle variabili.



```

100 OPTION BASE 1
110 FOR I=1 TO NV :: A$(I)=STR$(X(I)):: NEXT
    I
120 FOR I=1 TO NV :: FOR II=1 TO 16-(LEN(A$(
    I))
130 A$(I)="0"&A$(I):: NEXT II :: NEXT I
140 FOR I=1 TO NV
150 IF SEG$(A$(I),1,1)="-" THEN A$(I)="A"&(S
    EG$(A$(I),2,LEN(A$)))::: NEXT I
160 FOR I=1 TO NV :: FOR II=1 TO LEN(A$(I))
170 IF SEG$(A$(I),II,1)="." THEN A$(I)=SEG$(
    A$(I),1,II-1)&"B"&SEG$(A$(I),II+1,LE
    N(A$(I)):: NEXT I :: NEXT II
180 REM PASSAGGIO VARIABILI
190 I=143 :: FOR II=1 TO NV :: CALL CHAR(I,A
    $(II)):: I=I-1 :: NEXT II
    
```

TEXAS 99/4A



GENERATORE



Per quanti non posseggono una batteria elettronica e si dilettono nel suonare un qualche strumento ecco il programma adatto a loro. Esso infatti Vi permetterà di scrivere i Vostri ritmi nella forma che piu' Vi aggrada. Una volta lanciato avrete a disposizione le seguenti opzioni:

- a) SCRITTURA RITMO
- b) LISTA RITMO
- c) ASCOLTO RITMO
- d) CORREZIONI
- e) CARICAMENTO RITMO DA NASTRO
- f) SALVATAGGIO RITMO SU NASTRO

Immettendo la lettera posta davanti ad ogni opzione potrete gestirVi i Vostri ritmi. La prima cosa ovviamente da effettuare sara' l'opzione a) SCRITTURA RITMO, a questo punto Vi verra' richiesta la lunghezza in STEP (NUMERO DELLE NOTE DEL VOSTRO RITMO). Il massimo numero permesso dal programma e' di 255 STEP. Non rimane che comunicare i vari strumenti con cui desiderate creare il Vostro ritmo, strumenti che potrete scegliere tra i seguenti:

SIDE: tamburo
 SIDES: rullante

TEXAS 99/4A

TOM: tam-tam
 TOMS: tam-tam con suono secco
 BASS: gran-cassa
 HAT: piatto secco
 CYM: da 1 a 14 , vari tipi di piatti
 DEL: serve per dare un DELEY particolare e compreso tra 1 e 255.

Una volta scritta la sequenza potrete sentirla mediante l'opzione c) ASCOLTO RITMO.

Per interrompere il ciclo continuo della batteria dovrete premere il tasto FCTN ed il tasto contrassegnato dal numero 4.

```

1 REM
2 REM
3 DIM D(100)
4 DIM S(27)
5 DIM C$(256)
6 GOTO 24
7 REM
8 OPEN #1:"CS1",INTERNAL,INPUT,FIXED
9 INPUT #1:B
10 FOR A=1 TO B STEP 2
11 INPUT #1:C$(A),S(A),D(A),C$(A+1),S(A+1),D(A+1)
12 NEXT A
13 CLOSE #1
14 GOTO 25
15 REM
16 OPEN #1:"CS1",INTERNAL,OUTPUT,FIXED
    
```

TEXAS 99/4A

```

17 PRINT #1:B
18 FOR A=1 TO B STEP 2
19 PRINT #1:C$(A),S(A),D(A),C$(A+1),S(A+1
),D(A+1)
20 NEXT A
21 CLOSE #1
22 GOTO 25
23 REM
24 CALL CLEAR
25 PRINT "IMMETTERE SCELTA          "
26 PRINT
27 PRINT "a) SCRITTURA RITMO          b) LIS
TA RITMO          c) ASCOLTO RITMO
          d) CORREZZIONI          "
28 PRINT "e) CARICAMENTO RITMO  f) SALVAT
AGGIO RITMO "
29 CALL KEY(5,Z,X)
30 IF X=0 THEN 29
31 IF Z=65 THEN 39
32 IF Z=66 THEN 83
33 IF Z=67 THEN 100
34 IF Z=68 THEN 66
35 IF Z=69 THEN 8
36 IF Z=70 THEN 16
37 GOTO 25
38 REM
39 CALL CLEAR
40 INPUT "NUMERO DI STEP DEL RITMO?":B
41 IF B>256 THEN 40
42 FOR A=1 TO B
43 PRINT A;
44 INPUT C$(A)
45 IF C$(A)="DEL" THEN 51

```

TEXAS 99/4A

```
46 IF C$(A)="CYM" THEN 54
47 GOSUB 113
48 IF FLAG=1 THEN 58
49 NEXT A
50 GOTO 60
51 INPUT "LUNGHEZZA DELAY ?":D(A)
52 IF D(A)>1024 THEN 51
53 GOTO 49
54 GOSUB 123
55 IF FLAG=1 THEN 49
56 PRINT " * NON VALIDO   *"
57 GOTO 54
58 PRINT " * NON VALIDO *"
59 GOTO 43
60 PRINT "CORREZZIONI      ?(S/N)"
61 CALL KEY(5,Z,X)
62 IF X=0 THEN 61
63 IF Z=78 THEN 24
64 IF Z<>83 THEN 60
65 REM
66 PRINT "CORREZZIONE ALLA LINEA  ?"
67 INPUT A
68 IF A>B THEN 66
69 INPUT "NUOVO RITMO ?":C$(A)
70 IF C$(A)="DEL" THEN 75
71 IF C$(A)="CYM" THEN 78
72 GOSUB 113
73 IF FLAG=1 THEN 58
74 GOTO 60
75 INPUT "LUNGHEZZA DELAY ?":D(A)
76 IF D(A)>1024 THEN 75
77 GOTO 60
78 GOSUB 123
```

TEXAS 99/4A

```

79 IF FLAG=1 THEN 60
80 PRINT " * NON VALIDO  **"
81 GOTO 78
82 REM
83 CALL CLEAR
84 FOR A=1 TO B
85 PRINT A;" ";
86 IF C$(A)="DEL" THEN 95
87 IF C$(A)="CYM" THEN 97
88 PRINT C$(A)
89 NEXT A
90 PRINT " * PRONTO *"
91 PRINT " PREMI UN TASTO      "
92 CALL KEY(5,Z,X)
93 IF X=0 THEN 92
94 GOTO 25
95 PRINT "DEL ";D(A)
96 GOTO 89
97 PRINT "CYM ";S(A)
98 GOTO 89
99 REM
100 FOR A=1 TO B
101 CALL KEY(5,Y,Z)
102 IF Y=83 THEN 24
103 IF C$(A)="DEL" THEN 146
104 IF C$(A)="CYM" THEN 131
105 IF C$(A)="HAT" THEN 159
106 IF C$(A)="SIDE" THEN 162
107 IF C$(A)="SIDES" THEN 150
108 IF C$(A)="TOM" THEN 167
109 IF C$(A)="TOMS" THEN 153
110 IF C$(A)="BASS" THEN 156
111 NEXT A

```


TEXAS 99/4A

```

112 GOTO 100
113 FLAG=1
114 IF C$(A)="TOM" THEN 121
115 IF C$(A)="TOMS" THEN 121
116 IF C$(A)="BASS" THEN 121
117 IF C$(A)="SIDE" THEN 121
118 IF C$(A)="SIDES" THEN 121
119 IF C$(A)="HAT" THEN 121
120 GOTO 122
121 FLAG=0
122 RETURN
123 INPUT "TIPO CYMBAL ?":S(A)
124 SA=S(A)
125 IF SA*(SA-1)*(SA-2)*(SA-3)*(SA-4)*(SA
-5)*(SA-6)*(SA-8)*(SA-10)*(SA-14)*(SA-2
6)=0 THEN 128
126 FLAG=0
127 GOTO 129
128 FLAG=1
129 RETURN
130 REM
131 REM
132 O=S(A)
133 FOR X=0 TO 10 STEP 2
134 IF O>=3 THEN 136
135 CALL SOUND(-40,-5,X+5)
136 NEXT X
137 IF O<>1 THEN 140
138 CALL SOUND(3,-6,0)
139 GOTO 111
140 IF O<>2 THEN 142
141 GOTO 156
142 FOR X=0 TO 12 STEP 0-2

```

TEXAS 99/4A

```

143 CALL SOUND(-40,-5,X)
144 NEXT X
145 GOTO 111
146 REM
147 FOR DEL=1 TO D(A)
148 NEXT DEL
149 GOTO 111
150 REM
151 CALL SOUND(3,-6,1)
152 GOTO 111
153 REM
154 CALL SOUND(3,140,3,-7,8)
155 GOTO 111
156 REM
157 CALL SOUND(-40,-7,5,110,5)
158 GOTO 111
159 REM
160 CALL SOUND(2,-5,12)
161 GOTO 111
162 REM
163 FOR X=1 TO 18 STEP 6
164 CALL SOUND(-40,-6,X)
165 NEXT X
166 GOTO 111
167 REM
168 FOR X=0 TO 18 STEP 9
169 CALL SOUND(-40,140,X+2,-7,X+07)
170 NEXT X
171 GOTO 111

```

TEXAS 99/4A

•DIDATTICA.

Nel proseguimento di questa rubrica dedicata alla didattica ed atta ad aiutare quanti si avvicinino per la prima volta a questa MERAVIGLIOSA passione chiamata INFORMATICA ci occuperemo questa volta di ESPLORATORI e CANNIBALI!!... I cannibali sono razze decisamente in estinzione a parte rari casi nelle intricate foreste del BORNED e della NUOVA GUINEA.

E per fortuna si trovano in paesi molto lontani da noi e quindi il pericolo e' scongiurato, in caso contrario poveri noi!!... Ma veniamo al dunque e vediamo se possiamo far attraversare tre ESPLORATORI e tre CANNIBALI facendo ovviamente in modo che siano sempre in numero uguale?... Arduo dilemma... Non e' vero?... Meditate gente... Meditate... Fa' bene alle meningi. Ma veniamo alle descrizioni: I tre ESPLORATORI ed i tre CANNIBALI hanno a disposizione una barca per poter attraversare, ma essa non e' in grado di trasportare piu' di due persone contemporaneamente. Si rendono quindi necessarie piu' traversate per poter trasbordare tutti sulla spiaggia di destra. Ad ogni passaggio il programma chiede il numero di cannibali e di

TEXAS 99/4A

esploratori. Il totale d'essi non deve essere superiore a due (controllo effettuato all'istruzione 160, mediante la parola chiave IF). Nel caso che gli esploratori diventino meno dei cannibali il gioco termina. Per conoscere il numero degli esploratori e dei cannibali nel medesimo momento, e' stata definita una MATRICE mediante la parola chiave DIM N(4) nell'istruzione 30, e nella quale vengono caricati quanti d'essi rispettivamente trasbordano. La variabile di tipo numerico B permette di conoscere su quale spiaggia Vi trovate (B=0 per la spiaggia di sinistra e B=1 per la spiaggia di destra). Dopo aver visualizzato il numero di spiaggia sulla quale Vi trovate, il programma domanda il numero d'esploratori e di cannibali che devono attraversare e controlla che non abbiate commesso degli errori. Le istruzioni che vanno dalla 180 alla 250, servono a modificare il contenuto della MATRICE N() in funzione del numero degli esploratori e dei cannibali scelto (variabili di tipo numerico C ed M) e modificano il contenuto della variabile B in riferimento alla nuova spiaggia. Ogni volta prima di proseguire il programma controlla se avete vinto oppure perso ed in tal caso visualizza i messaggi opportuni. La ROUTINE che inizia all'istruzione 360 e termina all'istruzione 560, visualizza le

TEXAS 99/4A

due spiagge e le lettere (CARATTERI) C per i cannibali ed E per gli esploratori. Questa ROUTINE viene utilizzata, richiamandola mediante la parola chiave GOSUB rispettivamente alle istruzioni 70 e 260. E questo viene fatto piu' volte nell'ambito del programma. La tecnica di ricorrere a piccole parti di programma da riutilizzare ogni volta che se ne presenti il bisogno mediante la parola chiave GOSUB e' comunemente utilizzata anche nei programmi di tipo PROFESSIONALE e GESTIONALE e si rivela molto pratica per OTTIMIZZARE i programmi e renderli piu' facilmente modificabili ed ampliabili.



```

10 REM  ESPLORATORI E CANNIBALI
20 REM  -----
30 DIM N(4)
40 LET N(1)=3
50 LET N(2)=3
60 LET B=0
70 GOSUB 360
80 PRINT
90 PRINT "VI TROVATE SULLA SPIAGGIA ";B+1
100 PRINT "NUMERO DI CANNIBALI "
110 INPUT C
    
```

TEXAS 99/4A

```
120 IF C>N(1+(B*2))THEN 70
130 PRINT "NUMERO D'ESPLORATORI "
140 INPUT M
150 IF M>N(2+(B*2))THEN 70
160 IF C+M>2 THEN 70
170 IF C+M<1 THEN 70
180 N(1+(B*2))=N(1+(B*2))-C
190 N(2+(B*2))=N(2+(B*2))-M
200 IF B=1 THEN 230
210 LET B=1
220 GOTO 240
230 LET B=0
240 N(1+(B*2))=N(1+(B*2))+C
250 N(2+(B*2))=N(2+(B*2))+M
260 GOSUB 360
270 IF N(3)+N(4)=6 THEN 340
280 IF N(1)<=N(2)THEN 300
290 IF N(2)<>0 THEN 320
300 IF N(3)<=N(4)THEN 80
310 IF N(4)=0 THEN 80
320 PRINT "      AVETE PERSO "
330 END

340 PRINT "      AVETE VINTO "
350 END
360 REM -----
370 REM   IL GIOCO
380 REM -----
390 PRINT "SPIAGGIA 1      SPIAGGIA 2"
400 FOR I=1 TO 4
410 IF I/2=INT(I/2)THEN 440
420 LET C$="C"
430 GOTO 450
440 LET C$="E"
```

TEXAS 99/4A

```

450 FOR J=1 TO 3
460 IF N(I)<J THEN 490
470 PRINT C$;
480 GOTO 500
490 PRINT " ";
500 NEXT J
510 IF I<>2 THEN 530
520 PRINT "          ";
530 PRINT " ";
540 NEXT I
550 PRINT
560 RETURN
    
```



TEXAS MIX



L'USO DEI COLORI
di Renato Gualtieri

Più che parlare diffusamente dell'uso dei colori in alta risoluzione vorrei soffermarmi su di un aspetto molto importante: la pianificazione del sets di colore. Per definire i colori per i vostri grafici, si usa l'istruzione CALL COLOR. La sintassi è la seguente: CALL COLOR (a, b, c), dove a è il numero del set di colori, b è il colore del primo piano (o dei caratteri) e c è il colore dello sfondo. Sia a che b che c possono variare da 1 a 16. Ogni carattere

TEXAS 99/4A

grafico che definite può avere due specificazioni, quella del carattere (detta foreground) e quella dello sfondo (o background), scelta nella lista di 16.

I colori

Nel Texas TI-99 ci sono 16 insiemi di colori. Ogni insieme contiene il codice ASCII di otto caratteri. Se prendete la 'BASIC Reference Card' che vi è stata data con il computer, potrete fare dei segni sulla tabella dei codici dei caratteri, in modo tale che possiate identificare 'al volo' in che 'set' di colori si trova un dato carattere, e quali altri caratteri sono nello stesso set. Il procedimento è semplice: basta che dapprima facciate un segno ogni otto codici: in tal modo avrete identificato i 16 sets. Ora basta semplicemente numerarli ed il gioco è fatto. Per comodità potete trovare nella tabella la lista dei sets di caratteri con i codici compresi in ogni set.

La CALL COLOR

Per vedere subito come funziona la CALL COLOR provate a immettere il programma del Listato 1. Date il RUN. Le linee 10-20 stampano delle parole sullo schermo. Comunque non avendo fatto la CALL CLEAR vi troverete anche con il listato del programmino davanti. Appena il programma parte, lo schermo diventerà verde. La linea 30 cambia tutti i caratteri nel 'set' numero cinque in rosso (colore 7), con lo sfondo trasparente (colore 1). La linea 40 mantiene i

TEXAS 99/4A

colori sullo schermo fino a che non premete FCTN 4 per fare un CLEAR o fermare il programma. Noterete che quando avete fatto partire il programma lo schermo è diventato verde e che tutte le lettere nel set 5 (A, B, C, D, E, F, G) sono rosse. Lo sfondo trasparente fa in modo che lo sfondo di ogni carattere sia dello stesso colore dello sfondo dello schermo (verde). Dopo aver interrotto il programma, provate a cambiare la linea 30 in: 30 CALL COLOR (5, 6, 1). Le lettere ora diventeranno blu. Andate avanti provando diversi colori, per trovare gli effetti che vi piacciono.

Proviamo ora a cambiare il colore dello sfondo: aggiungete al vostro programma le linee del listato 2 e fate girare il tutto.

La linea 30 cambierà il colore delle lettere in quello che voi avete specificato (blu se non avete modificato il programma). La linea 60 cambia il colore delle lettere del set N.6 (H, I, J, K, M, N, O) in rosso, con uno sfondo bianco (colore 16). Ogni carattere vi comparirà come una lettera rossa in un quadrato bianco. Dopo un breve ritardo (linea 70), la linea 90 cambierà le lettere nel set 6 in segni bianchi su sfondo rosso (l'opposto di com'erano prima). La linea 100 torna alla 40, per cui vedrete i caratteri dapprima rossi su sfondo bianco poi bianchi su sfondo rosso, poi ancora rossi su sfondo bianco ecc.

Avrete sicuramente notato che non appena usate la CALL COLOR, tutti i caratteri del set interessato cambiano il loro colore: sia quelli sullo schermo sia quelli che potreste stampare in seguito. Per questa ragione bisogna porre molta attenzione nella pianificazione, per sapere esattamente quali caratteri state definendo essere di un certo colore piuttosto

TEXAS 99/4A

che non un altro. Se volete cambiare il colore dello schermo, potete farlo con CALL SCREEN (c), dove c è il numero di un colore (al solito tra 1 e 16). Se per esempio al nostro listato aggiungiamo la linea: 5 CALL SCREEN (12) possiamo pilotare il colore dello schermo. Provare ora un altro effetto, aggiungete la linea: 35 CALL COLOR (1,2,8). Questa cambierà tutti i caratteri nel set 1 in modo da avere un nero su azzurro (invece che su verde).

Notato l'effetto? Tutti gli spazi tra una parola e l'altra sono ora azzurrini (ciano). Il motivo è molto semplice: lo 'spazio' ha anch'esso un codice, il 32, che è, appunto, nel set N.1. Una cosa molto importante che si può notare da questi semplici esperimenti è che si ha un effetto molto diverso se prima si 'attiva' il colore e poi si stampano i caratteri, oppure se si stampano i caratteri e poi gli si cambia il colore. Ancora: pianificate bene i vostri colori perchè si comportino esattamente nel modo da voi voluto, a seconda degli effetti che voi volete ottenere.

Provate ora a caricare il programma del listato 3 (dopo aver dato un NEW).

Guardate attentamente dopo aver dato il RUN: lo schermo si pulisce, i caratteri sono stampati in nero, dopodichè alcuni caratteri diventano rossi. Se preferite avere gli asterischi in rosso dall'inizio, allora la CALL COLOR deve essere fatta prima della CALL VCHAR. Provate a cancellare la linea 80 e ad aggiungere la linea 15 CALL COLOR (2, 7, 1). Questo esempio serve a mostrarvi la differenza di una colorazione a stampa avvenuta e di una stampa a cui segue la colorazione: a voi scegliere l'approccio che più vi è congeniale.

TEXAS 99/4A

I caratteri invisibili

C'è un terzo modo di lavorare con i colori ed i caratteri: stampare i caratteri in modo trasparente (saranno così invisibili ma presenti sullo schermo) e poi farli 'comparire' ad un certo punto del programma. Questa soluzione può essere molto utile se disegnate delle figure complesse: non si vedrà la 'creazione' ma solo il disegno finito. Per avere un esempio di quello che vado cianciando, fate le seguenti modifiche al listato N.3:

15 CALL COLOR (2,1,1)

16 CALL COLOR (1,1,1)

80 CALL COLOR (2,7,1)

85 CALL COLOR (1,2,1)

Inizialmente i caratteri del set 2 e del set 1 sono resi invisibili, mettendo sia il loro colore che lo sfondo in trasparente. Poi vengono disegnati i caratteri con le CALL HCHAR e CALL VCHAR. Questo processo non lo si può vedere (è, appunto, invisibile). Infine, la linea 80 colora gli asterischi in rosso e la linea 85 i punti esclamativi in nero, in modo tale che i 'saluti' appaiano all'improvviso.

I caratteri grafici

Quando voi definite i vostri caratteri grafici, potete usare qualunque codice di carattere che volete. Se volete mantenere l'alfabeto intatto, usate i numeri sopra 95. Raggruppate i vostri caratteri in blocchi di 8 in modo tale che tutti i caratteri dello stesso colore siano nello stesso set. Se per esempio fate una barca che occupa 9 caratteri, cercate di comprimerla

TEXAS 99/4A

in 8, per farla stare, appunto, in un set di colori.

READY.

```
10 PRINT "CIAO A TUTTI!"
20 PRINT "QUESTO E' UN ESEMPIO"
30 CALL COLOR (5,7,1)
40 GOTO 40
```

READY.

Listato 1

READY.

```
40 FOR ATTESA=1 TO 100
50 NEXT ATTESA
60 CALL COLOR (6,7,16)
70 FOR ATTESA=1 TO 100
80 NEXT ATTESA
90 CALL COLOR (6,16,7)
100 GOTO 40
```

READY.

Listato 2

READY.

```
10 CALL CLEAR
20 CALL VCHAR (10,5,42,9)
```

TEXAS 99/4A

30 CALL VCHAR (10,10,42,9)
 40 CALL HCHAR (14,6,42,4)
 50 CALL VCHAR (10,17,42,9)
 60 CALL VCHAR (10,24,33,6)
 70 CALL VCHAR (18,24,33)
 80 CALL COLOR (2,7,1)
 90 GOTO 90

READY.

Listato 3

TI 99/4A- I Sets dei colori/caratteri

<u>Set</u>	<u>Cod.Caratteri</u>	<u>Set</u>	<u>Cod.Caratteri</u>
1	32-39	9	96-103
2	40-47	10	104-111
3	48-55	11	112-119
4	56-63	12	120-127
5	64-71	13	128-135
6	72-79	14	136-143
7	80-87	15	144-151
8	88-95	16	152-159

TEXAS 99/4A

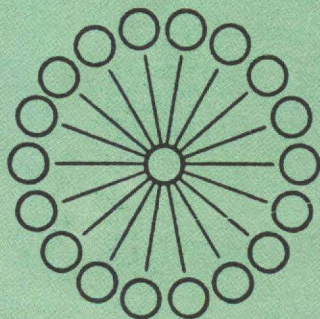
GIOCHI

di Leonardo Felician

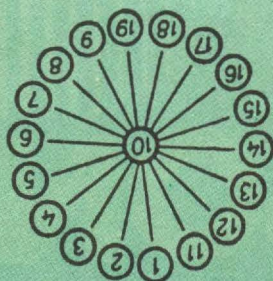
1-Amedeo possiede un Personal con dischetti. Un suo programma sta caricando record in un archivio sequenziale. Improvvisamente cade la corrente e la macchina si spegne. Dopo averla riaccesa, Amedeo scopre con sorpresa che non ha perso solo l'ultimo record su cui stava lavorando, ma non trova nell'archivio la bellezza di 15 records. Come mai? Cosa si può fare per evitare il ripetersi dell'inconveniente?

2-Avete un Personal con unità a dischetti? Se sì, può essere importante conoscere l'ampiezza del buffer di memoria. Se questa informazione non sta scritta sul manuale, sapreste inventare un metodo per ricavarla empiricamente?

3-Scrivete nei cerchi della figura sottostante i numeri da 1 a 19 in modo che la somma dei numeri posti su 3 cerchi in linea retta sia sempre 30. Naturalmente la soluzione può essere trovata ad occhio oppure con un opportuno programma.



TEXAS 99/4A



3.

2-Scrivete un programma fatto da un loop di WRITE: ogni volta che lampeggia la spia rossa dell'unità dischetto avviene un trasferimento fisico di dati dal buffer di memoria al dischetto. Provando con varie lunghezze record (32, 64, 128, etc.) e mostrando a video il contatore del loop non è difficile scoprire "ogni quanti" record il buffer pieno viene trasferito.

1- Un calcolatore non esegue le istruzioni di WRITE quando sono scritte, ma solo nel momento in cui è piena l'area di passaggio tra la memoria centrale e il dischetto, detta buffer. Perciò se la lunghezza record dell'archivio è piccola rispetto al buffer, molti record "restano in memoria" prima di essere scritti fisicamente su dischetto, e non son protetti dalle cadute di corrente. Soluzione: anziché aggiungere record in coda all'archivio, creare un archivio "diretto" vuoto, e andare in riscrizione. Così facendo il buffer viene scaricato ad ogni operazione.

Soluzioni



PERSONAL COMPUTER **CLUB**

GIOCHI-CONCORSO

PROGRAMMI E TRUCCHI

PROVE HARDWARE E SOFTWARE

PROGETTI SOFTWARE

BIBLIOTECA PERSONALE

e le rubriche

TUTTOCOMMODORE

TUTTOSINCLAIR

TUTTOMSX

**OGNI MESE
IN EDICOLA
LIRE 3.000**