Computer Games Series

GIOCHIAMO CON TI99/4A

GREMESE EDITORE

Computer Games Series

GIOCHIAMO CON **TI 99/4A**

Computer Games Series

GIOCHIAMO CON TI 99/4A

Andrew Nelson

GREMESE EDITORE

Computer Games Series
Periodico mensile
N. 10 - Dicembre 1984
Registrazione Tribunale di Roma N. 138/84
del 24 marzo 1984
Direttore Responsabile: Grazia Valci

Volumi pubblicati nella collana:

Giochiamo con COMMODORE 64
Giochiamo con VIC 20
Giochiamo con ZX 81
Giochiamo con ZX SPECTRUM
Giochiamo con ATARI
Giochiamo con TRS-80
Altri giochi per il vostro COMMODORE 64
Altri giochi per il vostro VIC 20
Altri giochi per il vostro ZX SPECTRUM
Giochiamo con TI 99/4A

Volumi di prossima pubblicazione:

Altri giochi per il vostro ATARI
Altri giochi per il vostro ZX 81

Titolo originale
More Games for your TI 99/4A

Traduzione dall'inglese
Gloria Sportoletti

Edizione italiana a cura di
Giancarlo Zagarese

Design
Ray Hyden

Illustrazioni
Sue Walliker

Fotocomposizione
Linotipia De Angelis - Roma

Stampa
Arti Grafiche LABOR - Todi

ISBN 88-7605-149-X

ANDREW NELSON — AUTORE

L'attenzione di Andrew Nelson si rivolge ad una ampia varietà di giochi, sia a quelli contro il computer sia alle partite tra amici. Dalla fusione di questi suoi interessi è scaturito il suo primo libro, *Creating Adventure Programs on your Computer* (Interface Pubblications, 1983). Andrew, è anche l'autore di un altro volume di questa collana: *Altri giochi per il vostro VIC 20*.

TIM HARTNELL
CURATORE DELL'EDIZIONE INGLESE

Tim Hartnell è un eminente giornalista la cui esperienza computeristica ha contribuito a determinare il successo della Technical Consumer Press. È anche autore di diversi libri, tra i quali: *Getting Acquainted With Your ZX 81, Let Your BBC Micro Teach You to Program* e *Programming Your ZX Spectrum.*

GIANCARLO ZAGARESE
CURATORE DELL'EDIZIONE ITALIANA

Giancarlo Zagarese, insegnante di discipline scientifiche, è autore di oltre 250 articoli e di vari volumi nei settori dell'elettronica e dell'attività subacquea. Per GREMESE EDITORE ha già collaborato in *Il Sub per tutti* e *Il Sub in apnea* della serie "gli Abbiccì".

SUE WALLIKER — ILLUSTRATRICE

Sue Walliker è un'illustratrice free-lance.

RINGRAZIAMENTI

L'autore desidera ringraziare Robert Pollack, direttore del negozio di microcomputer *Gametronics*, per l'assistenza e per la preziosa consulenza che ha fornito. Grazie a Robert, infatti, l'autore ha potuto incontrare i suoi maggiori collaboratori, Danny Olesh e Damon Pillinger. Alla redazione di questo libro hanno inoltre preso parte James Turner, Wayne Southwick, Tim Hartnell e Kevin Burfitt.

INDICE

Introduzione

Battere il programma di un computer è come aprire una porta sconosciuta. Finché non l'avete dischiusa — o, nel nostro caso, finché non avete fatto girare il programma — non potete sapere quali esperienze vi attendono. Naturalmente la targa sulla porta può fornirvi qualche indicazione, ma nulla che possa essere paragonato ad un'esperienza diretta.

Non sapete con esattezza quali emozioni vi riservano i grandi programmi contenuti in questo libro. Evidentemente se l'introduzione vi presenta un gioco di tipo spaziale è probabile che il programma, una volta fatto girare, non vi proporrà un gioco sul tipo di 'indovina che numero è'.

Nelle spiegazioni raramente si fa cenno alla strategia di gioco del computer, alla visualizzazione grafica sullo schermo o al divertimento che vi attende.

Questo libro contiene molte porte sconosciute — porte che conducono nello spazio eterno, nel diabolico mondo dei cervelli elettronici, dei maghi e dell'Avventura.

Noi vi abbiamo fornito le porte ...e le chiavi. Tutto ciò che dovete fare per aprire la serratura è battere il programma e farlo girare. Qualsiasi cosa troviate dietro ciascuna porta, vi garantisco che non vi deluderà.

Introduzione dell'autore

Il TI 99/4A è un computer che, a mio avviso, è stato sottovalutato troppo a lungo. In realtà l'audio superbo, i caratteri grafici definibili dall'utente e la buona tastiera fanno di questa macchina un ottimo strumento.
Sebbene alcuni comandi siano leggermente atipici, controllarli ed usarli nel modo più efficace non è affatto difficile, come spero di aver sufficientemente dimostrato in questo libro.
Affinché il volumetto vi sia della massima utilità, per tutti i programmi, ad eccezione di uno, è stato adottato il BASIC on board di tipo standard.
Se pensate che questo linguaggio standardizzato sia troppo limitato per una programmazione veramente efficace, allora vi attende una gradita sorpresa.
I programmi di questo libro vi piaceranno sicuramente, e spero che vi divertirete riadattandoli e migliorandoli.

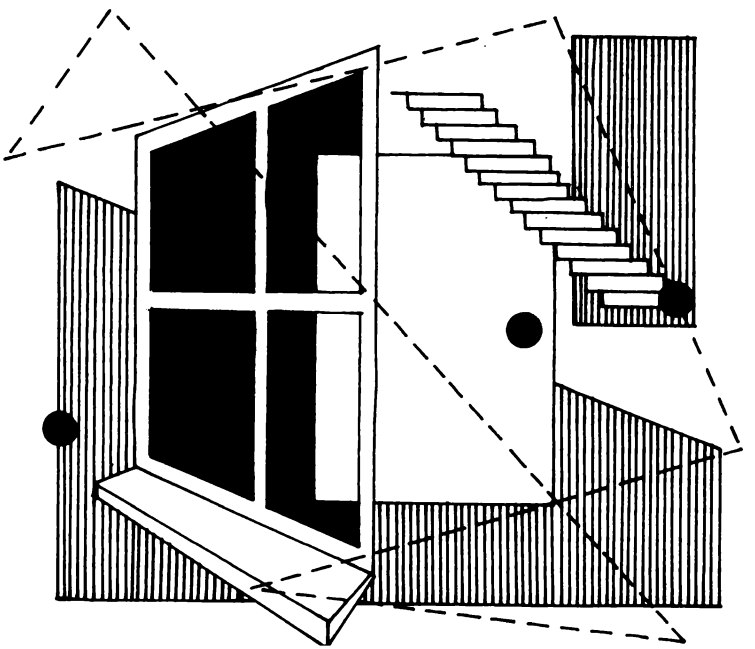
Prefazione all'edizione italiana

È con piacere che ho accolto l'invito dell'editore Gremese per curare una serie, finalmente in italiano, di volumetti sui videogiochi. Molti acquirenti di piccoli e medi personal computer, sia che lo utilizzino personalmente sia che ne abbiamo fatto oggetto di regalo ai propri figli, si sono fatti trascinare dalla pubblicità che precisava « al prezzo di un semplice video-gioco acquistate un intero computer ». È vero ed è stato un buon acquisto. Per imparare ad utilizzare un computer in modo semplice e divertente non c'è però niente di meglio della via ludica. Non giochi comprati e fruiti passivamente, però, ma "creati" e vissuti, istruzione dopo istruzione, prima compiendo e poi personalizzando o "inventando", in modo da accedere gradualmente nel nuovo mondo dell'informatica.

g.z.

3-D MAZE

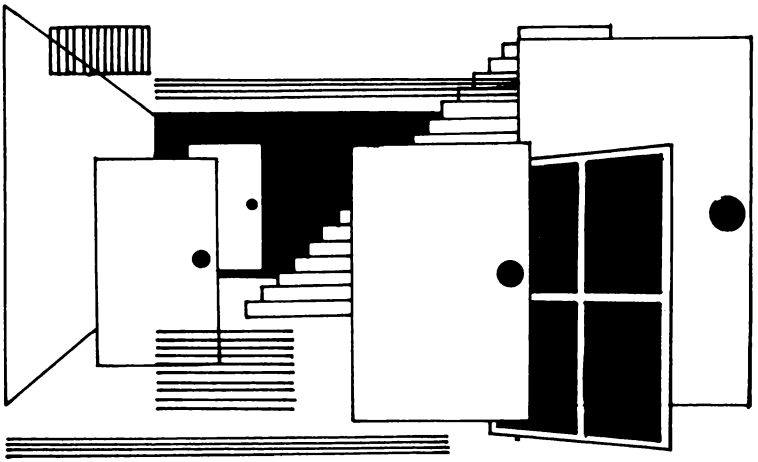
In questo gioco tenterete di scovare una stanza segreta, celata all'interno di un'imponente villa progettata dall'architetto pazzo T.I. McNinetyniner. Sebbene il tracciato rimanga sempre lo stesso, la posizione della camera che state cercando cambia ogni volta che fate girare il programma.
La rappresentazione grafica degli interni, le cui pareti si spostano proprio sotto i vostri occhi, produce una strana sensazione di "realtà".
Per muovere servitevi dei tasti contrassegnati dalle frecce.
Al momento di far girare il programma vi sarà chiesto di inserire i colori che intendete utilizzare. Iniziate con il 14,2, che lavora in modo ottimale, riservandovi la scelta di accoppiamenti di colore più personali per le sequenze successive.



```
10 REM  3-D MAZE
20 REM
30 INPUT "FOREGROUND,BACKGROUND COLOR ?":FG,BG
40 DIM ROOM(10,10),HUH(11,4),C(11,4)
50 GOSUB 680
60 A$(1)="SOUTH"
70 A$(2)="WEST"
80 A$(3)="NORTH"
90 A$(4)="EAST"
100 FOR X=1 TO 10
110 FOR Y=1 TO 10
120 READ ROOM(X,Y)
130 NEXT Y
140 NEXT X
150 FOR I=1 TO 11
160 READ HUH(I,3),HUH(I,4),HUH(I,1),HUH(I,2)
170 NEXT I
180 XX=1
190 YY=1
200 FACE=1
210 FOR I=1 TO 11
220 READ C(I,1),C(I,2),C(I,3),C(I,4)
230 NEXT I
240 REM
250 REM   ^SETUP,RUNv
260 GOTO 390
270 CALL SOUND(-500,110,0)
280 CALL KEY(1,K,S)
290 IF S=0 THEN 280
300 K=-(K=5)-2*(K=3)-3*(K+1=1)-4*(K=2)
310 K=K-(K=0)
320 FACE=FACE+(K=4)-2*(K=3)-(K=2)
330 FACE=FACE+4*(FACE>4)-4*(FACE<1)
340 IF K=1 THEN 360
350 GOTO 390
360 IF HUH(ROOM(XX,YY),FACE)=0 THEN 270
370 XX=XX+(FACE=3)-(FACE=1)
380 YY=YY+(FACE=2)-(FACE=4)
390 R=C(ROOM(XX,YY),FACE)
400 CALL CLEAR
410 REM  PRINT UP INFO
420 A=1
430 IF (R=1)+(R=2)+(R=4)+(R=7)THEN 450
440 A=2
450 GOSUB 890
460 A=7
470 IF (R=2)+(R=3)+(R=6)+(R=9)THEN 490
480 A=8
490 GOSUB 890
```

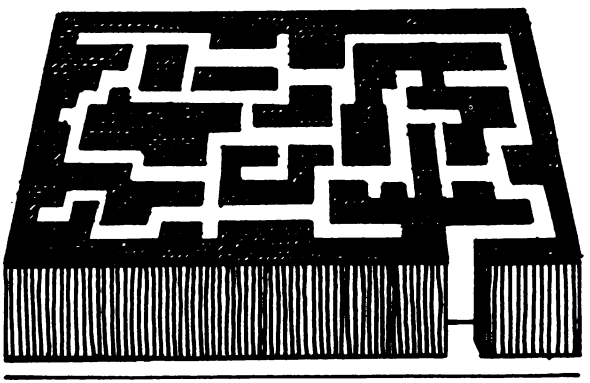
```
500 A=11
510 IF (R=4)+(R=5)+(R=6)+(R=10)THEN 520
                                  ELSE 540
520 GOSUB 890
530 GOTO 280
540 R=C(ROOM(XX+(FACE=3)-(FACE=1),
      YY+(FACE=2)-(FACE=4)),FACE)
550 A=3
560 IF (R=1)+(R=2)+(R=4)+(R=7)THEN 580
570 A=4
580 GOSUB 890
590 A=5
600 IF (R=2)+(R=3)+(R=6)+(R=9)THEN 620
610 A=6
620 GOSUB 890
630 A=10
640 IF (R=4)+(R=5)+(R=6)+(R=10)THEN 660
650 A=9
660 GOSUB 890
670 GOTO 280
680 REM  SEND THE 'SETUP' TO HERE
690 A$(1)="80C0E0F0F8FCFEFF"
700 A$(2)="FFFEFCF8F0E0C080"
710 A$(3)="FF7F3F1F0F070301"
720 A$(4)="0103070F1F3F7FFF"
730 A$(5)="FFFFFFFFFFFFFFFF"
740 A$(6)="0"
750 CALL CHAR(139,A$(1))
760 CALL CHAR(140,A$(2))
770 CALL CHAR(141,A$(3))
780 CALL CHAR(142,A$(4))
790 CALL CHAR(143,A$(5))
800 CALL CHAR(144,A$(6))
810 CALL COLOR(14,FG,1)
820 CALL SCREEN(BG)
830 IF BG>2 THEN 870
840 FOR I=1 TO 13
850 CALL COLOR(I,16,BG)
860 NEXT I
870 REM   139=\B,140=\T,141=/T,142=/B,
                  143=FULL,144=BLANK
880 RETURN
890 RE I  SEND THE MAP TO HERE
900 IF (A<1)+(A>12)THEN 920
910 ON A GOSUB 930,1000,1070,1140,1210,
    1280,1350,1420,1490,1610,1660,1710
920 RETURN
930 REM  LHS FRONT SOLID, COLOR=N
```

```
940 FOR I=1 TO 6
950 CALL HCHAR(I,I,139)
960 CALL HCHAR(25-I,I,140)
970 CALL VCHAR(I+1,I,143,24-2*I)
980 NEXT I
990 RETURN
1000 REM  LHS FRONT DOOR, COLOR=N
1010 FOR I=1 TO 6
1020 CALL VCHAR(6,I,143,14)
1030 NEXT I
1040 CALL HCHAR(6,6,139)
1050 CALL HCHAR(19,6,140)
1060 RETURN
1070 REM  LHS SIDE WALL,COLOR=N
1080 FOR I=7 TO 10
1090 CALL HCHAR(I,I,139)
1100 CALL HCHAR(25-I,I,140)
1110 CALL VCHAR(I+1,I,143,24-I*2)
1120 NEXT I
1130 RETURN
1140 REM  LHS SIDE DOOR, COLOR=N
1150 FOR I=7 TO 10
1160 CALL VCHAR(10,I,143,6)
1170 NEXT I
1180 CALL HCHAR(10,10,139)
1190 CALL HCHAR(15,10,140)
1200 RETURN
1210 REM  RHS SIDE WALL,COLOR=N
1220 FOR I=15 TO 18
1230 CALL HCHAR(I,I,141)
1240 CALL HCHAR(25-I,I,142)
1250 CALL VCHAR(26-I,I,143,I*2-26)
1260 NEXT I
1270 RETURN
1280 REM  RHS SIDE DOOR,COLOR=N
1290 FOR I=15 TO 18
1300 CALL VCHAR(10,I,143,6)
1310 NEXT I
1320 CALL HCHAR(10,15,142)
1330 CALL HCHAR(15,15,141)
1340 RETURN
1350 REM  RHS FRONT WALL,COLOR=N
1360 FOR I=19 TO 24
1370 CALL HCHAR(I,I,141)
1380 CALL HCHAR(25-I,I,142)
1390 CALL VCHAR(26-I,I,143,I*2-26)
1400 NEXT I
1410 RETURN
1420 REM  RHS FRONT DOOR,COLOR=N
```



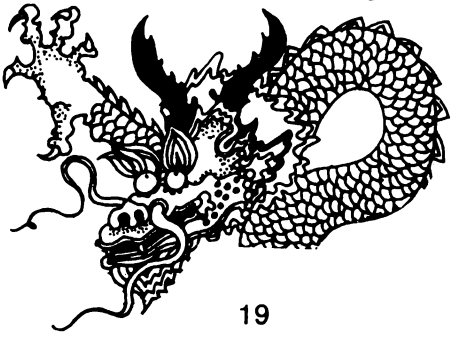
```
1430 FOR I=19 TO 24
1440 CALL VCHAR(6,I,143,14)
1450 NEXT I
1460 CALL HCHAR(6,19,142)
1470 CALL HCHAR(19,19,141)
1480 RETURN
1490 REM  CENTER FOREVER
1500 FOR I=11 TO 12
1510 CALL HCHAR(I,I,139)
1520 CALL HCHAR(25-I,I,140)
1530 NEXT I
1540 FOR I=13 TO 14
1550 CALL HCHAR(I,I,141)
1560 CALL HCHAR(25-I,I,142)
1570 NEXT I
1580 CALL VCHAR(12,11,143,2)
1590 CALL VCHAR(12,14,143,2)
1600 RETURN
1610 REM  CENTER WALL
1620 FOR I=10 TO 15
1630 CALL VCHAR(10,I,143,6)
1640 NEXT I
1650 RETURN
1660 REM  MIDDLE 'BIG' WALL
1670 FOR I=6 TO 19
1680 CALL VCHAR(6,I,143,14)
```

```
1690 NEXT I
1700 RETURN
1710 CALL CLEAR
1720 RETURN
1730 DATA 4,5,5,5,5,5,5,5,5,6
1740 DATA 7,8,9,7,8,9,7,8,9,2
1750 DATA 7,1,7,9,1,7,9,4,9,2
1760 DATA 1,8,8,8,8,11,8,8,8,8
1770 DATA 1,3,1,3,1,3,1,3,1,3
1780 DATA 4,6,4,6,4,6,4,6,4,6
1790 DATA 1,3,1,3,1,3,1,3,1,3
1800 DATA 1,1,1,1,1,1,1,1,1,1
1810 DATA 1,1,1,1,1,1,1,1,1,1
1820 DATA 1,1,1,1,1,1,1,1,1,1
1830 DATA 1,1,0,0,1,0,1,0,1,0,0,1,0,1,1,0,
          0,1,0,1,0,0,1,1,1,1,1,0,1,1,0,1,
          1,0,1,1
1840 DATA 0,1,1,1,1,1,1,1
1850 DATA 6,4,1,3,2,5,2,5,4,1,3,6,3,6,4,1,
          5,2,5,2,1,3,6,4,9,10,7,8,10
1860 DATA 7,8,9,7,8,9,10,8,9,10,7,
          11,11,11,11
```



SEARCH FOR THE HOLY GRAIL

Eccovi offerta la possibilità di sfidare il gruppo dei Monty Python nella loro ricerca: il Sacro Gral, sulle cui tracce attraverserete le distese solitarie... dello schermo televisivo.
In realtà non dovrete cercare a lungo, dato che il Gral compare distintamente sul video sin dall'inizio del gioco. Tuttavia sapere dove si trova e raggiungerlo di fatto, sono cose ben diverse. Per giungere alla vostra meta dovrete districarvi in un labirinto di muri costituiti di pietre dalle fogge più strane.
Per spostarvi all'interno di questo dedalo servitevi dei tasti contrassegnati dalle frecce. Attenzione! Il cammino che dovete percorrere è intralciato da un terribile drago, più che risoluto a distruggervi. Una programmazione intelligente ha reso questo mostro relativamente astuto: è costantemente alle vostre calcagna e possiede la facoltà di passare attraverso i muri. In queste condizioni non è che avete molte speranze!
Se riuscite a raggiungere il Sacro Gral durante il primo gioco, il computer genererà per voi un secondo labirinto: questa volta invisibile!
Sullo schermo potrete vedere soltanto il Gral, voi stessi ed il drago. Come ultimo imprevisto sul video comparirà una palla rimbalzante, da cui è bene guardarsi.

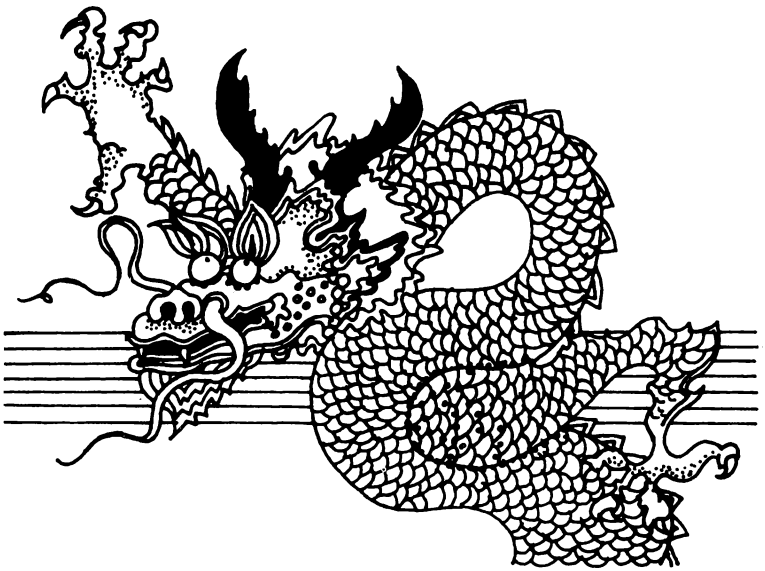


```
10 REM  SEARCH FOR THE HOLY GRAIL
20 GAMES=1
30 TOTS=387
40 HS=387
50 RANDOMIZE
60 CALL SCREEN(2)
70 SPEED=4
80 Y=23
90 A=8
100 B=8
110 R=10
120 C=16
130 X=2
140 CALL CHAR(131,"FFC3A59999A5C3FF")
150 OP=0
160 X$(0)="0000001"
170 X$(1)="0000001818"
180 X$(2)="0000183C3C18"
190 X$(4)="3C7EFFFFFFFF7E3C"
200 X$(3)="00183C7E7E3C18"
210 X$(5)=X$(4)
220 X$(6)=X$(4)
230 X$(7)=X$(3)
240 CALL CHAR(145,X$(0))
250 X$(8)=X$(2)
260 X$(9)=X$(1)
270 CALL CLEAR
280 CALL HCHAR(24,1,131,64)
290 CALL VCHAR(1,32,131,48)
300 CALL HCHAR(11,16,138)
310 CALL HCHAR(3,3,131,13)
320 CALL HCHAR(3,17,131,10)
330 CALL VCHAR(3,15,131,9)
340 CALL CHAR(133,"00101038101")
350 CALL VCHAR(3,17,131,9)
360 CALL HCHAR(12,15,131,3)
370 CALL HCHAR(2,2,133,1)
380 CALL HCHAR(9,5,131,10)
390 CALL HCHAR(11,5,131,10)
400 CALL HCHAR(10,15,32,4)
410 CALL VCHAR(5,3,131,10)
420 CALL VCHAR(3,30,131,10)
430 CALL HCHAR(22,2,131,8)
440 CALL CHAR(138,"00FF7E3C18187E7E")
450 CALL HCHAR(22,11,131,20)
460 CALL HCHAR(20,5,131,6)
470 CALL HCHAR(20,15,131,10)
480 CALL HCHAR(18,6,131,25)
490 CALL HCHAR(16,3,131,4)
```

```
500 CALL HCHAR(16,10,131,10)
510 CALL VCHAR(19,12,131,3)
520 CALL HCHAR(6,4,131,8)
530 CALL HCHAR(11,14,32)
540 CALL CHAR(144,"60E2151838181838")
550 CALL HCHAR(7,3,32)
560 CALL HCHAR(6,2,131,6)
570 CALL VCHAR(5,19,131,12)
580 CALL VCHAR(18,13,32)
590 CALL VCHAR(12,8,131,6)
600 CALL VCHAR(17,3,131,4)
610 CALL VCHAR(20,5,131,3)
620 CALL HCHAR(11,22,133)
630 CALL HCHAR(14,20,131,5)
640 CALL CHAR(150,"003C3C183C3C2424")
650 CALL HCHAR(12,22,131,6)
660 CALL HCHAR(15,8,32)
670 CALL HCHAR(14,10,131,3)
680 CALL VCHAR(14,16,131,2)
690 CALL HCHAR(20,12,32)
700 CALL HCHAR(15,19,32)
710 CALL VCHAR(12,26,131,6)
720 CALL HCHAR(16,4,129)
730 CALL HCHAR(8,15,129)
740 CALL HCHAR(16,15,129)
750 CALL HCHAR(3,14,129)
760 CALL HCHAR(11,7,129,3)
770 CALL HCHAR(19,26,133)
780 FOR I=1 TO 12
790 CALL COLOR(I,16,1)
800 NEXT I
810 CALL COLOR(13,6,1)
820 CALL COLOR(14,11,1)
830 CALL COLOR(15,10,1)
840 CALL COLOR(16,4,1)
850 CALL CHAR(129,"FFC3A59999A5C3FF")
860 CALL HCHAR(Y,X,150)
870 FOR I=1 TO 19-SPEED+(SCORE/15)
880 S=INT(RND*18)+3
890 T=INT(RND*26)+3
900 IF (S=11)*(T=16)THEN 880
910 CALL HCHAR(S,T,129-2*(RND>.8))
920 NEXT I
930 CALL HCHAR(INT(RND*22)+2,
                INT(RND*30)+2,133)
940 RR=R+(R>Y)-(R<Y)
950 CC=C+(C>X)-(C<X)
960 CALL GCHAR(R,CC,K)
```

```
970 IF (K=131)+(K=145)+
        (K=138)+(K=144)THEN 1020
980 IF K=150 THEN 1590
990 CALL HCHAR(R,C,32)
1000 CALL HCHAR(R,CC,144)
1010 C=CC
1020 CALL GCHAR(RR,C,K)
1030 IF (K=131)+(K=145)+(K=138)+
                    (K=144)THEN 1080
1040 IF K=150 THEN 1590
1050 CALL HCHAR(R,C,32)
1060 CALL HCHAR(RR,C,144)
1070 R=RR
1080 FOR I=1 TO SPEED
1090 CALL KEY(1,K,S)
1100 IF (S=0)+(K>5)+(K=1)THEN 1300
1110 XX=X+(K=2)-(K=3)
1120 YY=Y+(K=5)-(K+1=1)
1130 CALL GCHAR(YY,XX,K)
1140 IF K<>133 THEN 1190
1150 CALL SOUND(20,440,0)
1160 CALL SOUND(40,480,0)
1170 CALL SOUND(20,440,0)
1180 SCORE=SCORE+5
1190 SCORE=SCORE-(K=131)*(SCORE>0)
1200 IF (K<>129)*(K<>131)THEN 1220
1210 CALL SOUND(-10,440,0,-7,7)
1220 IF (K=129)+(K=131)+(K=150)THEN 1300
1230 IF (K=145)+(K=144)THEN 1590
1240 I=I+9*(K=133)
1250 IF K=138 THEN 1420
1260 CALL HCHAR(Y,X,32)
1270 CALL HCHAR(YY,XX,150)
1280 Y=YY
1290 X=XX
1300 NEXT I
1310 IF OP THEN 1390
1320 CALL HCHAR(A,B,32)
1330 A=INT(RND*22)+2
1340 B=INT(RND*30)+2
1350 IF (A=11)*(B=16)THEN 1330
1360 OP=10
1370 CALL HCHAR(A,B,145)
1380 GOTO 940
1390 OP=OP-1
1400 CALL CHAR(145,X$(OP))
1410 GOTO 940
1420 SPEED=SPEED+(SPEED>1)
```

```
1430 FOR I=1 TO 15
1440 CALL SOUND(-100,110+20*I,0)
1450 NEXT I
1460 SCORE=SCORE+20
1470 IF SPE=1 THEN 1540
1480 IF SPEED<>3 THEN 1580
1490 PRINT ::::::::::::::::::::::::::::
 ::::::::::::::::::"       BONUS LEVEL":::
:::::::::::::::::::::::::::::::::
1500 CALL SCREEN(6)
1510 IF SPE=1 THEN 1540
1520 SPE=1
1530 GOTO 80
```

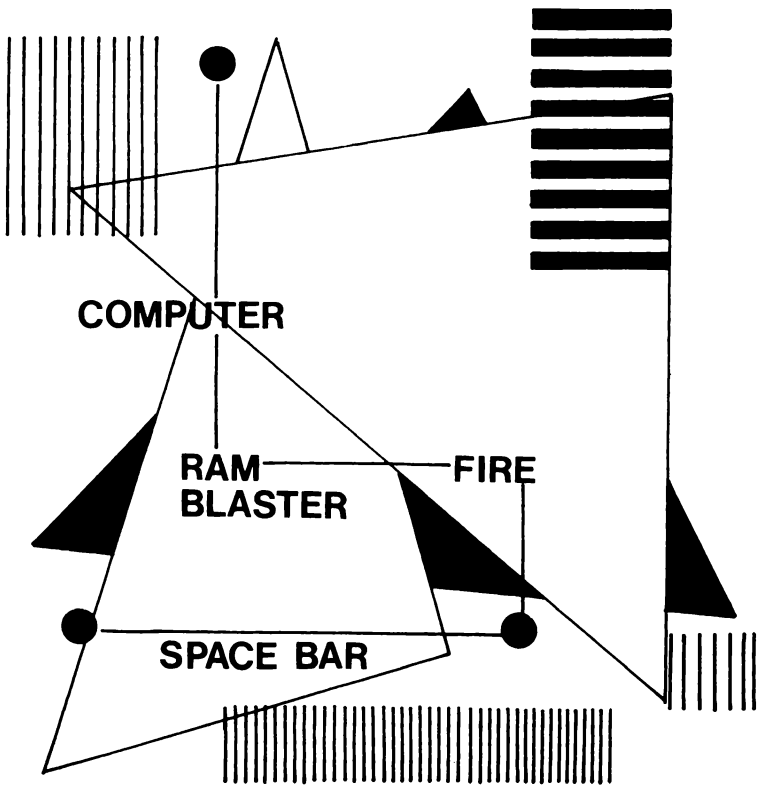


```
1540 CALL SCREEN(2)
1550 SPEED=2
1560 SCORE=SCORE+15
1570 SPE=0
1580 GOTO 80
1590 CALL SOUND(100,440,0)
1600 CALL SOUND(20,440,30)
1610 CALL SOUND(100,340,0)
1620 CALL SOUND(20,340,30)
```

```
1630 CALL SOUND(100,340,0)
1640 CALL SOUND(20,380,30)
1650 CALL SOUND(100,380,0)
1660 CALL SOUND(20,330,30)
1670 CALL SOUND(100,330,0)
1680 CALL SOUND(80,380,30)
1690 CALL SOUND(100,380,0)
1700 CALL SOUND(20,430,30)
1710 CALL SOUND(100,410,0)
1720 GAMES=GAMES+1
1730 PRINT ::"SCORE --->";SCORE
1740 TOTS=TOTS+SCORE
1750 PRINT "HIGH SCORE ===>";HS
1760 PRINT ::"AVERAGE SCORE ==>":
   "FROM ";GAMES;" GAMES":"=";TOTS/GAMES:
1770 IF SCORE>HS THEN 1840
1780 INPUT "PLAY AGAIN ?":A$
1790 IF SEG$(A$,1,1)<>"Y" THEN 1830
1800 SCORE=0
1810 SPE=0
1820 GOTO 50
1830 STOP
1840 HS=SCORE
1850 GOTO 1780
```

RAM BLASTER

In questo gioco siete un temuto RAM-icida che vuole distruggere tutti i pezzi di computer che appaiono sullo schermo.
Durante la prima partita potrete vedere quanti punti vi vengono assegnati per ogni tipo di componente che riuscite a colpire. Per spostarvi usate i tasti contrassegnati dalle frecce e aprite il fuoco premendo la barra spaziatrice. Non dimenticate che potete sparare solo mentre siete in movimento. Il piccolo proiettile nero procede sempre seguendo la direzione del vostro moto.



```
10 REM   RAM BLASTER
20 REM  BY D&D
30 RANDOMIZE
40 LEVEL=1
50 CALL CLEAR
60 PRINT "       ALL SYSTEMS GO":
   :"            STANDBY":::::::
70 FOR I=127 TO 159
80 CALL CHAR(I,STR$(INT(RND*999999999)+1))
90 NEXT I
100 KILL=0
110 A=9
120 X=10
130 DIM SC(5)
140 SC(1)=40
150 SC(2)=70
160 SC(3)=100
170 SC(4)=200
180 XM=0
190 CALL CLEAR
200 DIM HS(20),HS$(20)
210 GOSUB 2840
220 Y=10
230 CALL CHAR(105,"FF81C399C381FF")
240 DIM XX(10),YY(10)
250 CALL CHAR(104,"0000003C3C000000")
260 SC(5)=220
270 CALL CHAR(127,"")
280 YM=1
290 LEVEL=1
300 SCORE=0
310 CALL CHAR(103,"00183C7E7E3C18")
320 DIM T(10)
330 FOR I=1 TO A
340 T(I)=INT(RND*3)+1
350 XX(I)=INT(RND*24)+1
360 YY(I)=INT(RND*32)+1
370 NEXT I
380 CALL CHAR(100,"7E3C7E3C7E3C7E")
390 CALL CHAR(101,"0000FFE7E73C3C")
400 CALL CHAR(102,"0000FF421824FF")
410 FREE=1
420 CALL KEY(3,K,S)
430 LIFE=LIFE+1
440 IF LIFE>3 THEN 1000
450 QQ=12
460 FOR I=1 TO 16
470 CALL COLOR(I,02,QQ)
480 NEXT I
```

```
490 CALL COLOR(9,05,QQ)
500 CALL COLOR(10,2,QQ)
510 AX=22
520 XX$="     "&STR$(4-LIFE)&" LIVES      "
530 GOSUB 3220
540 KILL=0
550 AX=23
560 XX$="LEVEL -->"&STR$(LEVEL)
570 GOSUB 3220
580 AX=24
590 XX$="PRESS ANY KEY TO BEGIN LEVEL"
600 GOSUB 3220
610 CALL KEY(0,K,S)
620 IF S=0 THEN 610
630 AX=10
640 XX$="GET READY"
650 GOSUB 3220
660 AX=11
670 XX$=" GET SET "
680 GOSUB 3220
690 AX=13
700 XX$="                    GO                    "
710 GOSUB 3220
720 CALL SCREEN(QQ)
730 CALL CLEAR
740 CALL KEY(0,KEY,STATUS)
750 IF KEY=32 THEN 1970
760 XM=(XM-(KEY=ASC("X"))+(KEY=ASC("E")))*
      (KEY<>ASC("S"))*(KEY<>ASC("D"))
770 YM=(YM-(KEY=ASC("D"))+(KEY=ASC("S")))
      *(KEY<>ASC("E"))*(KEY<>ASC("X"))
780 CALL HCHAR(X,Y,32)
790 X=X+XM
800 Y=Y+YM
810 X=X-24*(X<=0)
820 X=X+24*(X>=25)
830 Y=Y-32*(Y<=0)
840 Y=Y+32*(Y>=33)
850 CALL GCHAR(X,Y,HIT)
860 IF HIT=32 THEN 890
870 GOSUB 1810
880 GOTO 990
890 CALL HCHAR(X,Y,100)
900 ABC=N
910 GOTO 930
920 IF N=ABC THEN 2330
930 N=N+1
940 N=N+A*(N>A)
950 IF T(N)=0 THEN 920
```

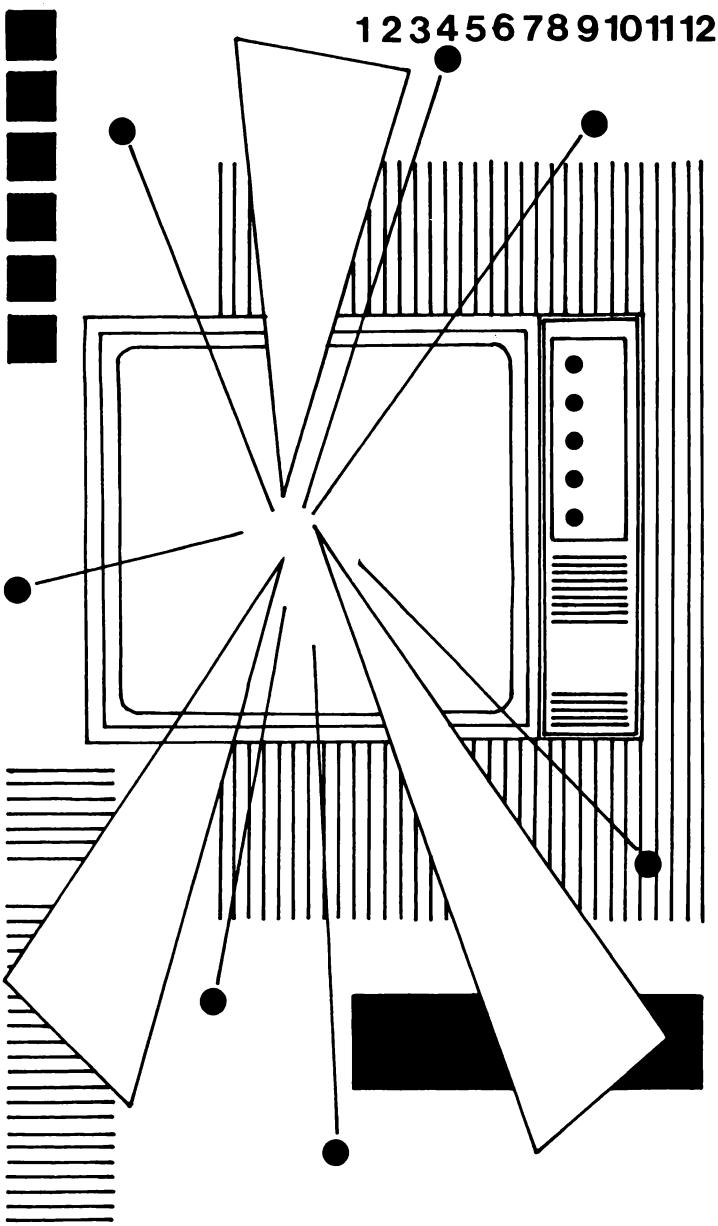
```
960 XXX=1
970 ON T(N)GOSUB 1030,1180,1290,1310,1420
980 IF XXX=1 THEN 1010
990 GOTO 430
1000 GOTO 1950
1010 CALL SOUND(1,1440,0)
1020 GOTO 740
1030 S=(XX(N)>X)-(XX(N)<X)
1040 REM  S=S+(S=0)
1050 D=(S=0)*((YY(N)<Y)-(YY(N)>Y))
1060 D=D+10000
1070 CALL HCHAR(XX(N),YY(N),32)
1080 XX(N)=XX(N)+S
1090 YY(N)=YY(N)+D
1100 XX(N)=XX(N)+(XX(N)>24)-(XX(N)<=0)
1110 YY(N)=YY(N)+(YY(N)>32)-(YY(N)<=0)
1120 CALL GCHAR(XX(N),YY(N),HIT)
1130 IF HIT=32 THEN 1160
1140 GOSUB 1820
1150 RETURN
1160 CALL HCHAR(XX(N),YY(N),101)
1170 RETURN
1180 CALL HCHAR(XX(N),YY(N),32)
1190 XX(N)=XX(N)+(XX(N)>X)-(XX(N)<X)
1200 YY(N)=YY(N)+(YY(N)>Y)-(YY(N)<Y)
1210 XX(N)=XX(N)+(XX(N)>24)-(XX(N)<=0)
1220 YY(N)=YY(N)+(YY(N)>32)-(YY(N)<1)
1230 CALL GCHAR(XX(N),YY(N),HIT)
1240 IF HIT=32 THEN 1270
1250 GOSUB 1820
1260 RETURN
1270 CALL HCHAR(XX(N),YY(N),102)
1280 RETURN
1290 IF (SQR((XX(N)-X)^2)>=3)+
          (SQR((YY(N)-Y)^2)>=3)THEN 1030
1300 T(N)=4
1310 CALL HCHAR(XX(N),YY(N),32)
1320 XX(N)=XX(N)+(XX(N)>(X+XM))-
                  (XX(N)<(X+XM))
1330 YY(N)=YY(N)+(YY(N)>(Y+YM)
                  -(YY(N)<(Y+YM))
1340 XX(N)=XX(N)+(XX(N)>24)-(XX(N)<1)
1350 YY(N)=YY(N)+(YY(N)>32)-(YY(N)<1)
1360 CALL GCHAR(XX(N),YY(N),HIT)
1370 IF HIT=32 THEN 1400
1380 GOSUB 1820
1390 RETURN
1400 CALL HCHAR(XX(N),YY(N),103)
1410 RETURN
```

```
1420 FOR LOOP=1 TO 2
1430 IF INT(RND*10)+1>1 THEN 1460
1440 CALL HCHAR(XX(N),YY(N),104)
1450 GOTO 1470
1460 CALL HCHAR(XX(N),YY(N),32)
1470 XX(N)=XX(N)+(XX(N)>X)-(XX(N)<X)
1480 YY(N)=YY(N)+(YY(N)>Y)-(YY(N)<Y)
1490 XX(N)=XX(N)+(XX(N)>24)-(XX(N)<1)
1500 YY(N)=YY(N)+(YY(N)>32)-(YY(N)<1)
1510 CALL GCHAR(XX(N),YY(N),HIT)
1520 IF HIT=32 THEN 1550
1530 GOSUB 1820
1540 RETURN
1550 IF (SQR((XX(N)-X)^2)<5)*(SQR((YY(N)
     -Y)^2)<5)THEN 1580
1560 CALL HCHAR(XX(N),YY(N),127)
1570 GOTO 1590
1580 CALL HCHAR(XX(N),YY(N),105)
1590 NEXT LOOP
1600 RETURN
1610 CALL SOUND(-200,-7,0)
1620 IF HIT=104 THEN 1790
1630 SCORE=SCORE+SC(T(N))
1640 T(N)=0
1650 GOSUB 1700
1660 KILL=KILL+1
1670 LLLL=SCORE
1680 IF KILL>=A THEN 2330
1690 RETURN
1700 IF LLLL>=3000*FREE THEN 1780
1710 IF SCORE<3000*FREE THEN 1780
1720 FOR IS=1 TO 5
1730 CALL SOUND(150,440,0)
1740 CALL SOUND(150,40000,30)
1750 NEXT IS
1760 LIFE=LIFE-1
1770 FREE=FREE+1
1780 RETURN
1790 CALL HCHAR(XX(N),YY(N),32)
1800 GOTO 1630
1810 HIT=100
1820 IF HIT<>100 THEN 1610
1830 FOR I=1 TO 30
1840 CALL SOUND(-100,-7,0)
1850 CALL HCHAR(X,Y,INT(RND*32)+128)
1860 NEXT I
1870 PRINT "SCORE=";SCORE::::::
1880 XXX=0
1890 FOR III=1 TO A
```

```
1900 IF T(N)=0 THEN 1930
1910 XX(N)=INT(RND*24)+1
1920 YY(N)=INT(RND*32)+1
1930 NEXT III
1940 RETURN
1950 GOSUB 2590
1960 GOTO 100
1970 YB=YM
1980 XB=XM
1990 PX=X
2000 PY=Y
2010 KONST=10-LEVEL
2020 IF KONST>3 THEN 2040
2030 KONST=4
2040 FOR I=1 TO KONST
2050 PX=PX+XB
2060 PY=PY+YB
2070 IF (PX<1)+(PY<1)+(PX>24)+
      (PY>32)THEN 2170
2080 CALL GCHAR(PX,PY,SPLAT)
2090 IF SPLAT=100 THEN 2140
2100 IF SPLAT=104 THEN 2140
2110 IF SPLAT<>32 THEN 2180
2120 CALL HCHAR(PX,PY,104)
2130 CALL HCHAR(PX,PY,32)
2140 NEXT I
2150 XB=0
2160 YB=0
2170 GOTO 760
2180 FOR N=1 TO A
2190 IF T(N)=0 THEN 2210
2200 IF (XX(N)=PX)*(YY(N)=PY)THEN 2230
2210 NEXT N
2220 GOTO 760
2230 LLLL=SCORE
2240 SCORE=SCORE+SC(T(N))
2250 GOSUB 1700
2260 LLLL=SCORE
2270 T(N)=0
2280 KILL=KILL+1
2290 IF KILL>=A THEN 2330
2300 CALL SOUND(-140,-5,0)
2310 CALL HCHAR(PX,PY,32)
2320 GOTO 760
2330 FOR I=1 TO 10
2340 CALL COLOR(1,00,INT(RND*16)+1)
2350 NEXT I
2360 CALL COLOR(1,16,00)
2370 LEVEL=LEVEL+1
```

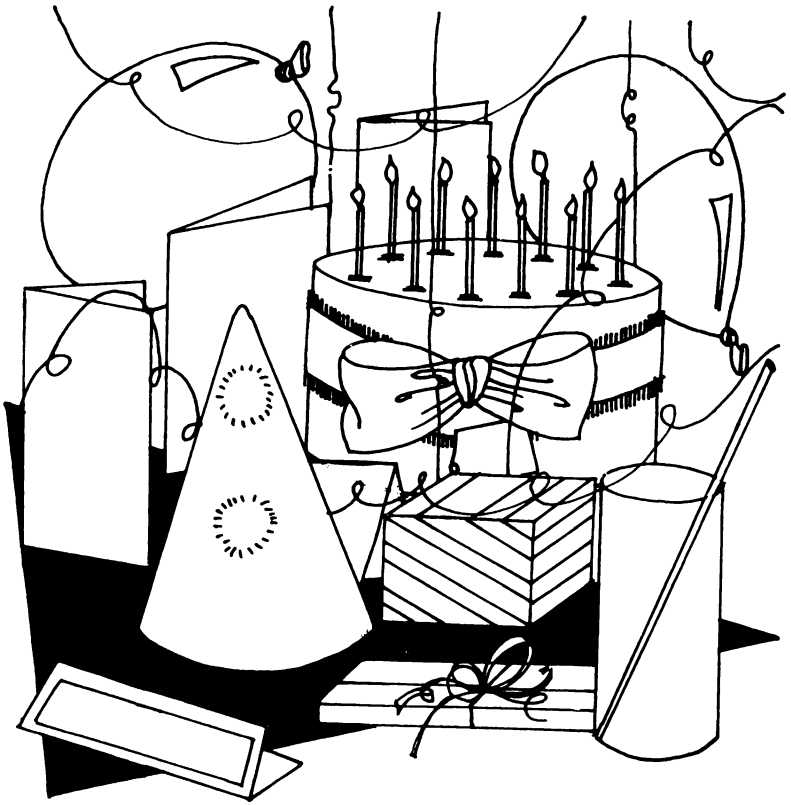
```
2380 FOR I=1 TO A
2390 T(I)=INT((RND*5)+1)
2400 G=G-(T(I)=4)
2410 H=H-(T(I)=5)
2420 XX(I)=INT(RND*24)+1
2430 YY(I)=INT(RND*32)+1
2440 NEXT I
2450 IF (G>LEVEL/3)+(G>A-2)THEN 2490
2460 H=0
2470 G=0
2480 GOTO 2380
2490 X=10
2500 IF ((H+G)>LEVEL/2)+((G+H)>A-2)THEN 2540
2510 G=0
2520 H=0
2530 GOTO 2380
2540 Y=10
2550 XM=0
2560 YM=1
2570 PRINT ::::::::::::::"SCORE="
              ,SCORE::::::::::::
2580 GOTO 450
2590 XX$="      GAME OVER           "
2600 FOR I=1 TO LEN(XX$)
2610 CALL HCHAR(12,(15-INT(.5*LEN(XX$)))+I,
                  ASC(SEG$(XX$,I,1)))
2620 NEXT I
2630 FOR I=1 TO 20
2640 IF SCORE>HS(I)THEN 2730
2650 NEXT I
2660 FOR I=1 TO 20
2670 PRINT HS(I),HS$(I)
2680 NEXT I
2690 FOR I=1 TO 1000
2700 NEXT I
2710 RETURN
2720 REM
2730 FOR J=20 TO I+1 STEP -1
2740 HS(J)=HS(J-1)
2750 HS$(J)=HS$(J-1)
2760 NEXT J
2770 INPUT "NAME PLEASE ?":HS$(I)
2780 CALL CLEAR
2790 IF LEN(HS$(I))<11 THEN 2820
2800 PRINT "TEN CHARACTERS LONG-NO MORE"
2810 GOTO 2770
2820 HS(I)=SCORE
2830 GOTO 2660
```

```
2840 IF HS(1)>0 THEN 2890
2850 FOR I=1 TO 20
2860 HS(I)=5000-15*I-INT(RND*10)
2870 HS$(I)="RAMCHIP"&STR$(I)
2880 NEXT I
2890 SCORE=0
2900 AX=4
2910 XX$="RAMCHIP"
2920 GOSUB 3220
2930 AX=6
2940 XX$="d - YOUR RAMCHIP"
2950 GOSUB 3220
2960 REM  d,h,e,40,f,70,e,100,g,200,i,220
2970 AX=7
2980 XX$="e - RAMCHIPS     40"
2990 GOSUB 3220
3000 AX=8
3010 XX$="f - ROMCHIPS     70"
3020 GOSUB 3220
3030 AX=9
3040 XX$="e - PROMS,V1    100"
3050 GOSUB 3220
3060 AX=10
3070 XX$="g - PROMS,V2    200"
3080 GOSUB 3220
3090 AX=11
3100 XX$="i - EPROMS      220"
3110 GOSUB 3220
3120 AX=12
3130 XX$="HIDDEN EPROMS   220"
3140 GOSUB 3220
3150 AX=13
3160 XX$="h - STATIC ELECTRICITY -BEWARE"
3170 GOSUB 3220
3180 AX=15
3190 XX$="FREE RAMCHIP EACH 3000 POINTS"
3200 GOSUB 3220
3210 RETURN
3220 AY=(16-(.5*LEN(XX$)))
3230 IF XX$<>"GET READY" THEN 3250
3240 CALL CLEAR
3250 FOR IJ=1 TO LEN(XX$)
3260 CALL HCHAR(AX,AY+IJ,
          ASC(SEG$(XX$,IJ,1)))
3270 NEXT IJ
3280 RETURN
```


1 2 3 4 5 6 7 8 9 10 11 12

HAPPY BIRTHDAY

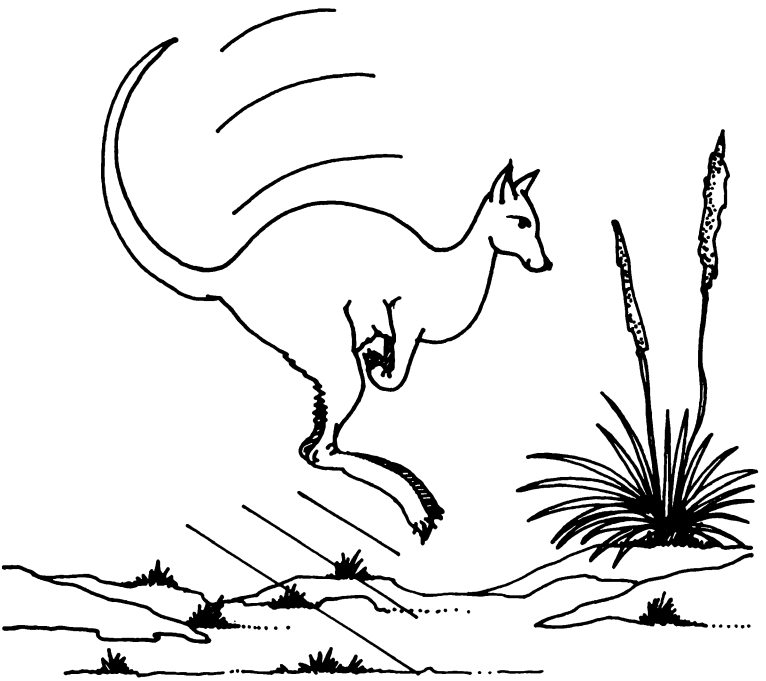
Questo semplice programma ideato da Damon Pillinger e Mark Charlton vi svelerà il giorno della settimana in cui siete nati. Naturalmente lo potete usare anche per scoprire in quale giorno si è verificato un evento storico.



```
10 REM  ** HAPPY BIRTHDAY **
20 REM
30 REM  ** DAMON PILLINGER **
40 REM
50 REM  BASED ON 'DAY OF THE WEEK'
60 REM  BY MARK CHARLTON, IN
70 REM  "THE GATEWAY GUIDE..."
80 DAY$="SUNMONTUEWEDTHUFRISAT"
90 FOR JJ=1 TO 30
100 PRINT
110 NEXT JJ
120 PRINT "PLEASE ENTER THE DAY (AS"
130 PRINT "  A NUMBER, EG 30) ON"
140 INPUT "  WHICH YOU WERE BORN ":DAY
150 IF (DAY<1)+(DAY>31)THEN 120
160 PRINT
170 INPUT "WHAT MONTH ":MO
180 IF (MO<1)+(MO>12)THEN 170
190 PRINT :::
200 INPUT "WHAT YEAR   XXXX ":YEAR
210 REM  IF YEAR>100 THEN 210
220 REM  YEAR=YEAR+1900
230 CALL CLEAR
240 COY=0
250 IF MO>2 THEN 270
260 COY=1
270 LA=YEAR-COY
280 OA=MO+(12*COY)
290 PA=INT(LA/100)
300 DAT=INT((5*LA)/4)+D-1+INT
          (13*(OA+1)/5)-P+INT(P/4)
310 DAT1=(DAT-7*INT(DAT/7))*3+1
320 PRINT ,,,,,,
330 PRINT "THE DATE:"
340 PRINT "          ";DAY;"/";MO;"/";YEAR
350 PRINT
360 PRINT "IS A ";SEG$(DAY$,DAT1,3)
370 PRINT ::::::
380 FOR JJ=1 TO 421
390 NEXT JJ
400 GOTO 90
```

DOWNUNDER

Kevin Burfitt si serve del computer per riprodurre paesaggi australiani. In questo caso dovete far scendere il vostro canguro lungo il video, fino alla porta che vedete in fondo allo schermo. Mentre procedete a rapidi salti, badate di non finire nel fiume o contro il dingo, l'ornitorinco o il vombato.
Disponete di pochissimo tempo per raggiungere la porta; potete però accrescere la vostra forza, e avere quindi qualche chance in più per giungere all'uscita, ingerendo la Vegemite che compare sul vostro video. (Per inciso, la Vegemite è una pasta nera che gli Australiani mangiano praticamente su tutto. Si tratta anche di un ottimo alimento per canguri).
Una volta che avrete guadagnato l'uscita, sullo schermo apparirà un nuovo quadro con percorsi ben più difficili.



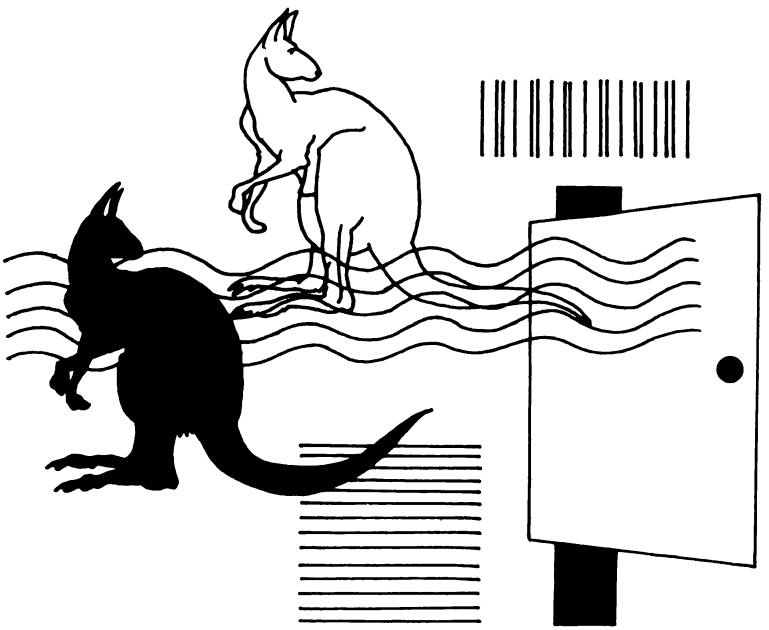
```
10 REM  *************
20 REM  * DOWNUNDER *
30 REM  *************
40 REM  BY Kevin Burfitt
50 REM
60 CALL CLEAR
70 PRINT "DOWNUNDER"::"MOVE YOUR KANGAROO
   WITH THE ARROW KEYS"::"TRY TO REACH TH
   E DOOR (";CHR$(30);")"
80 PRINT "AT THE BOTTOM OF THE SCREEN.DONT
    RUN INTO THE RIVER (f),DINGO (p),PLAT
    YPUS (h)":"OR THE WOMBAT (e)"
90 PRINT ::"GET POINTS FOR HITTING THE
 MAGIC ITEMS (ijklm),BUT MAKESURE YOU DON
 T RUN OUT OF":"TIME BY EATING";
100 PRINT " VEGEMITE (g)"::"GOOD LUCK"::
110 REM
120 RANDOMIZE
130 DEF BIN(X)=((L/2^X)<>INT(L/2^X))*
    (((L/2^X)-INT(L/2^X))>=0.5)
140 REM
150 REM  SETUP
160 LIVES=3
170 SCORE=0
180 CALL CHAR(100,"060704070E1E24CF")
190 CALL CHAR(101,"00037FFEFEFC486C")
200 CALL CHAR(99,"183C7E7E3C18183C")
210 CALL CHAR(102,"04E318C6318C6318")
220 CALL CHAR(103,"007E7E66667E3C18")
230 CALL CHAR(104,"0000003FFF123600")
240 CALL CHAR(105,"00424218184242")
250 CALL CHAR(106,"00544554455445")
260 CALL CHAR(107,"007E425A5A427E")
270 CALL CHAR(108,"0042424242422418")
280 CALL CHAR(109,"001818FFFF1818")
290 CALL CHAR(112,"0000027FFC78486C")
300 INPUT "STARTING LEVEL(1..5)?":BL
310 IF (BL>5)+(BL<1)THEN 300
320 L=BL
330 GOSUB 1730
340 CALL HCHAR(24,1,32,32)
350 X=17
360 CALL HCHAR(24,17,100)
370 GOSUB 1300
380 PRINT :
390 FOR I=2 TO 23
400 IF I=R THEN 550
410 CALL HCHAR(24,1,32,32)
```

```
420 IF I=D THEN 470
430 FOR J=1 TO -L*(L<11)-10*(L>10)
440 CALL HCHAR(24,RND*31+1,99)
450 NEXT J
460 GOTO 570
470 D=INT(RND*29)+1
480 CALL HCHAR(24,1,102,32)
490 CALL HCHAR(24,D,32,2)
500 D=24-INT(RND*I)
510 CALL HCHAR(D,1,102,32)
520 CALL HCHAR(D,D,32,2)
530 D=1
540 GOTO 570
550 CALL HCHAR(24,1,102,32)
560 CALL HCHAR(24,RND*28+1,32,3)
570 PRINT :
580 NEXT I
590 CALL HCHAR(RND*23+1,RND*31+1,32+71*F)
600 CALL HCHAR(24,1,32,32)
610 CALL HCHAR(24,RND*31+1,30)
620 IF W=0 THEN 650
630 WY=INT(RND*23)+1
640 WX=1
650 IF P=0 THEN 680
660 PY=INT(RND*23)+1
670 PX=1
680 GOSUB 1490
690 CALL KEY(1,K,S)
700 XX=X+(K=2)-(K=3)
710 YY=Y+(K=5)-(K+1=1)
720 IF (XX<1)+(YY<1)+(XX>32)+
        (YY>24)+(S=0)THEN 800
730 CALL GCHAR(YY,XX,HIT)
740 IF (HIT=32)+(HIT=100)THEN 770
750 IF HIT=103 THEN 1150
760 IF HIT=99 THEN 800 ELSE 1190
770 CALL HCHAR(Y,X,32)
780 Y=YY
790 X=XX
800 CALL HCHAR(Y,X,100)
810 IF W=0 THEN 870
820 CALL HCHAR(WY,WX,32)
830 WX=WX+1+32*(WX=32)
840 CALL GCHAR(WY,WX,HIT)
850 IF HIT=100 THEN 1080
860 CALL HCHAR(WY,WX,101)
870 IF P=0 THEN 930
880 CALL HCHAR(PY,PX,32)
890 PX=PX-1-32*(PX=1)
```

```
900 CALL GCHAR(PY,PX,HIT)
910 IF HIT=100 THEN 1080
920 CALL HCHAR(PY,PX,104)
930 IF DD<>1 THEN 970
940 DI=DX
950 DJ=DY
960 GOSUB 1800
970 TIME=TIME-1
980 IF TIME<>50 THEN 1000
990 DD=1
1000 DD=DD-1-(DD<2)
1010 IF TIME>20 THEN 690
1020 CALL SOUND(-100,440,0,441,2,442,4)
1030 IF TIME>1 THEN 690
1040 REM
1050 CALL SOUND(1000,440,0,441,2,442,4)
1060 PRINT "NO VEGEMITE"
1070 PRINT "No VEGEMITE !"
1080 LIVES=LIVES-1
1090 X=17
1100 TIME=200
1110 Y=1
1120 PRINT :"NEXT KANGAROO":
1130 IF LIVES>-1 THEN 370
1140 GOTO 1620
1150 CALL SOUND(100,660,0,880,2,1100,4)
1160 SCORE=SCORE+5
1170 TIME=TIME+20
1180 GOTO 770
1190 IF HIT=30 THEN 1220
1200 IF HIT>104 THEN 1560
1210 GOTO 1080
1220 SCORE=SCORE+10
1230 L=L+1
1240 TIME=TIME+10+(140-TIME)*F
1250 CALL HCHAR(Y,X,32)
1260 Y=1
1270 X=XX
1280 CALL HCHAR(YY,XX,100)
1290 GOTO 370
1300 F=BIN(1)
1310 TIME=-150*(L=BL)+TIME
1320 DX=1
1330 DY=24
1340 W=BIN(2)
1350 R=INT(RND*23)*BIN(3)+1
1360 P=BIN(4)
1370 D=BIN(5)
1380 DD=D
```

```
1390 CALL CHAR(32,"55AA55AA55AA55AA")
1400 FG=13-2*D
1410 BG=3+9*D
1420 CALL COLOR(1,FG,BG)
1430 FOR I=2 TO 16
1440 CALL COLOR(I,2,BG)
1450 NEXT I
1460 D=INT(RND*22)*D+1
1470 Y=1
1480 RETURN
1490 REM  SOUND
1500 N=1-(L<10)-(L<5)
1510 FOR I=N TO 5
1520 CALL HCHAR(RND*23+1,RND*31+1,104+I)
1530 NEXT I
1540 REM  CALL SOUND()
1550 RETURN
1560 SCORE=SCORE+110-HIT
1570 CALL HCHAR(YY,XX,48+110-HIT)
1580 CALL SOUND(100,440,1,441,2,442,3)
1590 DD=10+10*(DD=0)
1600 CALL SOUND(80,660,1,661,2,662,3)
1610 GOTO 770
1620 DATA 71,97,109,101,32,111,118,101,114
1630 RESTORE 1620
1640 FOR I=1 TO 9
1650 READ A
1660 CALL HCHAR(9,10+I,A)
1670 NEXT I
1680 PRINT :::::"YOUR SCORE ->";SCORE:::
1690 PRINT "DO YOU WISH TO CONTINUE
      (CON)OR PLAY A NEW GAME(NEW)"
1700 INPUT "CON or NEW ?":A$
1710 IF A$="CON" THEN 32767
1720 GOTO 32767
1730 DATA DOWNUNDER TUNE
1740 RESTORE 1730
1750 FOR I=1 TO []
1760 READ D,N,L
1770 CALL SOUND(D,N,L)
1780 NEXT I
1790 RETURN
1800 DI=DX+(X<DX)-(X>DX)
1810 CALL GCHAR(DY,DI,HIT)
1820 IF HIT=100 THEN 1080
1830 IF (HIT<>102)*(HIT<>30)*
        (HIT<>112)THEN 1900
1840 DI=DX
1850 DJ=DY+(Y<DY)-(Y>DY)
```

```
1860 CALL GCHAR(DJ,DX,HIT)
1870 IF HIT=100 THEN 1080
1880 IF (HIT<>102)*(HIT<>30)*
          (HIT<>112)THEN 1900
1890 DJ=DY
1900 CALL HCHAR(DY,DX,32)
1910 DX=DI
1920 DY=DJ
1930 CALL HCHAR(DY,DX,112)
1940 RETURN
```



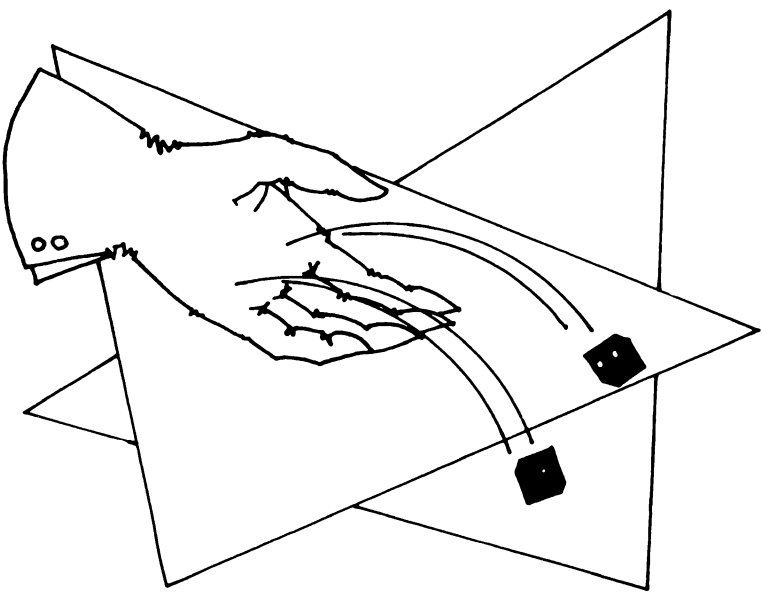
TEXAS TENBY

Texas Tenby è un gioco d'azzardo con i dadi realizzato da Tim Hartnell sulla falsariga di "Craps". Semplice da giocare (ma non altrettanto da vincere), prevede 10 tiri per ogni partita.

Lanciate i due dadi e sommatene i punti. Se al primo tiro ottenete un 7 o un 11 come somma totale, il gioco finisce. Questi punteggi infatti, se ottenuti al primo tentativo, vengono detti « una naturale » e determinano la vittoria.

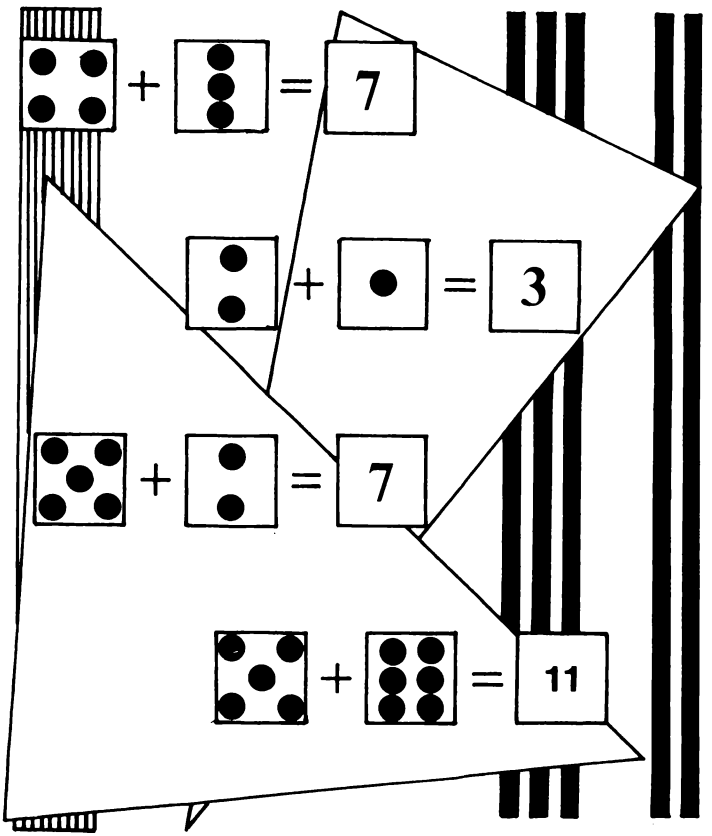
Al contrario, perdete se il risultato del vostro primo lancio ("craps") è un 2, un 3 o un 12. In tutti gli altri casi il risultato del lancio diventa il vostro "punto" e per vincere dovete rifarlo ancora una volta, *prima* di fare un 7.

Vi accorgerete che il TI vi solleva dalla maggior parte delle operazioni: lancia i dadi, controlla vincite e perdite e vi tiene aggiornati sulla situazione. Il gioco è più divertente se lo fate con un amico; tirate i dadi a turno, riservando per voi il 1°, il 3°, il 5°, il 7° e il 9° lancio e lasciando gli altri al vostro partner.



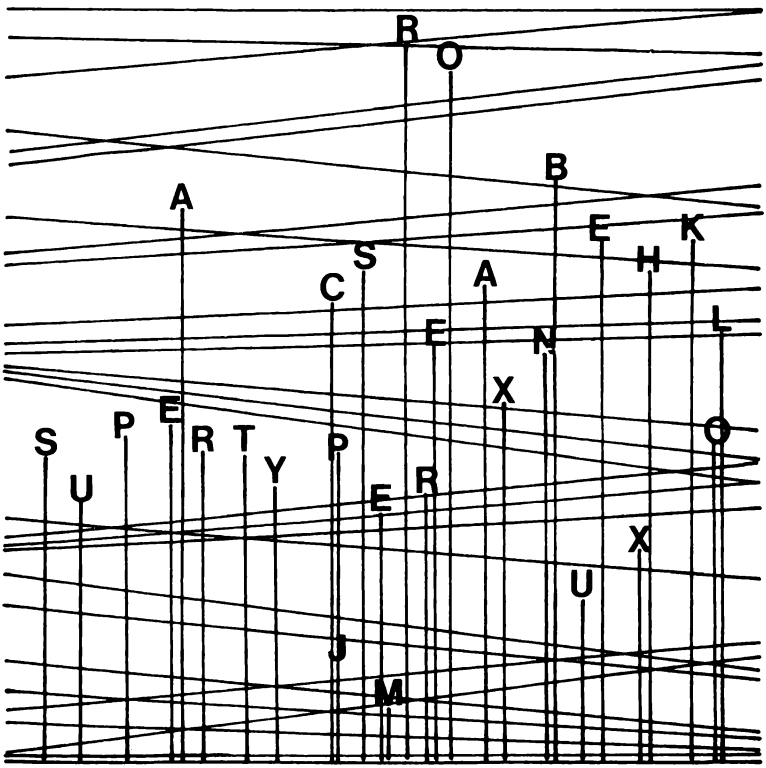
```
10 REM  TEXAS TENBY
20 CALL CLEAR
30 B$="You rolled"
40 RANDOMIZE
50 G=0
60 W=0
70 L=0
80 G=G+1
90 IF G=11 THEN 490
100 PRINT "----------------------------"
110 PRINT "--------NEW GAME------------"
120 PRINT "----------------------------"::
130 PRINT "WINS:";W,"LOSSES:";L
140 PRINT ::"GAME NUMBER";G
150 GOSUB 270
160 IF (A=7)+(A=11)<0 THEN 350
170 IF (A=2)+(A=3)+(A=12)<0 THEN 380
180 P=A
190 PRINT :"Your point is";P
200 GOSUB 270
210 IF A=P THEN 350
220 IF A=7 THEN 380
230 FOR T=1 TO 300
240 NEXT T
250 PRINT :"^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^";
260 GOTO 190
270 A=INT(RND*6+RND*6+2)
280 IF A>12 THEN 270
290 FOR T=1 TO 100
300 NEXT T
310 GOSUB 450
320 GOSUB 450
330 PRINT ::,B$;A
340 RETURN
350 PRINT :"You win!"
360 W=W+1
370 GOTO 400
380 PRINT :"You lose!"
390 L=L+1
400 FOR T=1 TO 400
410 NEXT T
420 GOSUB 450
430 GOSUB 450
440 GOTO 80
450 REM  SOUND ROUTINE
460 NOTE=INT(RND*200)+178
470 CALL SOUND((RND*1000+100),NOTE,0,
    (NOTE+220),2)
```

```
480 RETURN
490 REM  END OF GAME
500 PRINT ::"THAT'S THE END OF THE GAME"
510 PRINT :::"LET'S SEE HOW YOU WENT..."
520 FOR T=1 TO 4
530 GOSUB 450
540 NEXT T
550 IF W>L THEN 590
560 IF L<W THEN 610
570 PRINT ::"THE WINS EQUALLED THE LOSSES"
580 END
590 PRINT :"YOU WON BY";W;"TO";L
600 END
610 PRINT :"YOU LOST BY";L;"TO";W
620 END
```



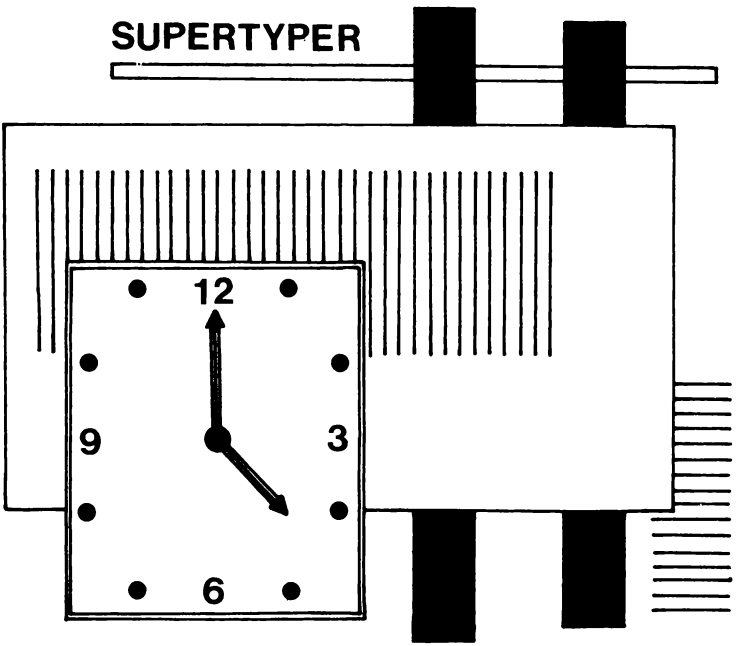
SUPER TYPER

In questo dinamico gioco firmato da K. Burfitt, tenterete di bloccare delle lettere d'alfabeto scelte a caso nella loro corsa verso la sommità dello schermo. Si tratta di una vera sfida tra voi, o meglio le vostre dita, e l'inesorabile avanzare di un orologio.
Le lettere compariranno ad una ad una in fondo al video, cominciando di qui la loro ascesa verso l'alto. Prima che raggiungano il bordo superiore dovete batterne i caratteri corrispondenti sulla tastiera. Vi accorgerete che con l'aumentare del punteggio le lettere si muoveranno a velocità sempre maggiore. Il gioco è dotato di un dispositivo segna-punti.



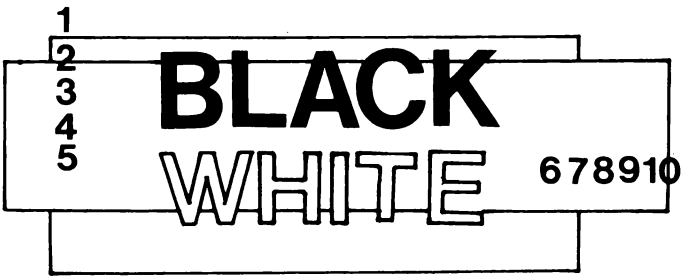
```
10 REM  SUPER TYPER MARK II
20 REM
30 REM  K BURFITT
40 REM
50 HS$="SUPER TYPER MARK "&CHR$(127)
60 HSCORE=400
70 CALL CHAR(127,"FF424242424242FF")
80 CALL CLEAR
90 RANDOMIZE
100 GOTO 540
110 FOR I=0 TO 9
120 A(I)=INT(RND*26)+65
130 B(I)=23
140 NEXT I
150 FOR I=0 TO 9
160 CALL HCHAR(B(I),I*2+5,A(I))
170 NEXT I
180 CALL KEY(0,K,S)
190 IF S<1 THEN 320
200 FOR I=0 TO 9
210 IF A(I)<>K THEN 310
220 CALL VCHAR(1,I*2+5,32,24)
230 SCORE=SCORE+5
240 CALL SOUND(-200,-6,0)
250 A(I)=INT(RND*26)+65
260 B(I)=23
270 FOR II=1 TO LEN(STR$(SCORE))
280 CALL HCHAR(8+II,2,ASC(SEG$
                  (STR$(SCORE),II,1)))
290 NEXT II
300 CALL HCHAR(B(I),I*2+5,A(I))
310 NEXT I
320 X=INT(RND*10)
330 B(X)=B(X)-1-SCORE/100
340 IF B(X)<0.5 THEN 410
350 CALL HCHAR(B(X)+1+SCORE/100,X*2+5,32)
360 FOR I=B(X)+1+SCORE/100 TO B(X)STEP -1
370 CALL HCHAR(I+1,X*2+5,32)
380 CALL HCHAR(I,X*2+5,A(X))
390 NEXT I
400 GOTO 180
410 PRINT "YOU GOT A SCORE OF ";SCORE;
420 FOR I=1 TO 1500
430 NEXT I
440 CALL CLEAR
450 IF SCORE<HSCORE THEN 540
460 PRINT "YOU HAVE TOP SCORE "
470 PRINT "THE PREVIOUS BEST WAS";
    HSCORE;"BY     ";HS$
```

```
480 INPUT "WHAT IS YOUR NAME ?":HS$
490 IF LEN(HS$)<25 THEN 520
500 PRINT "ONLY 24 CHARACTERS LONG,
    NOT";LEN(HS$);" CHARACTERS PLEASE"
510 GOTO 480
520 HSCORE=SCORE
530 CALL CLEAR
540 FOR I=1 TO LEN(HS$)
550 CALL HCHAR(I,26,ASC(SEG$(HS$,I,1)))
560 NEXT I
570 S$=STR$(HSCORE)
580 FOR I=1 TO LEN(S$)
590 CALL HCHAR(I,27,ASC(SEG$(S$,I,1)))
600 NEXT I
610 Q$="PRESS A KEY TO START"
620 FOR I=1 TO LEN(Q$)
630 CALL HCHAR(4,I+2,ASC(SEG$(Q$,I,1)))
640 NEXT I
650 CALL KEY(0,K,S)
660 IF S=0 THEN 650
670 CALL HCHAR(4,2,32,22)
680 SCORE=0
690 GOTO 110
```



TI FASTERMIND

Eccovi la possibilità di sfidare il computer in un famoso gioco, molto spesso conosciuto con il nome di "Code-breaker". Il computer preleva un numero di quattro cifre (senza ripetizioni) che voi v'ingegnerete d'indovinare.
Avete a disposizione un massimo di 10 tentativi. Dopo ogni risposta il computer vi darà un punteggio espresso in "neri" o "bianchi". I neri rappresentano le cifre corrette che occupano la giusta posizione all'interno del codice che avete inserito; i bianchi sono cifre giuste ma in posizione sbagliata. Il gioco si basa su un programma di Tony Baker.

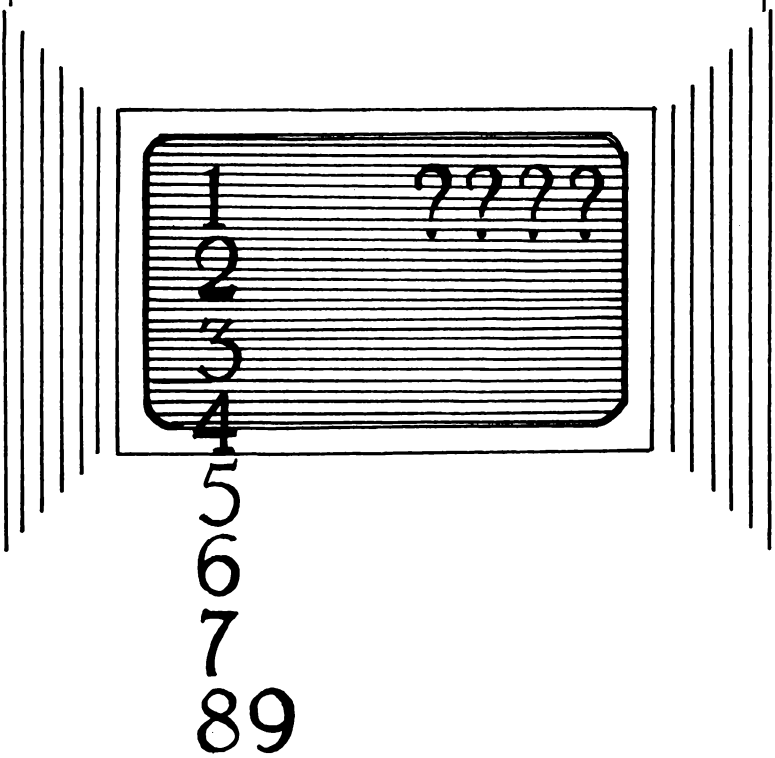



```
10 REM  TI FASTERMIND
20 RANDOMIZE
30 CALL CLEAR
40 C(1)=INT(RND*9)+1
50 Z=2
60 C(Z)=INT(RND*9)+1
70 J=1
80 IF C(J)=C(Z)THEN 50
90 IF J=Z-1 THEN 120
100 J=J+1
110 GOTO 80
120 IF Z=4 THEN 150
130 Z=Z+1
```

```
140 GOTO 60
150 FOR CHANCES=1 TO 10
160 INPUT A
170 IF A<1000 THEN 160
180 IF A>9999 THEN 160
190 A1=A
200 FOR Z=1 TO 4
210 G(Z)=A-10*(INT(A/10))
220 A=INT(A/10)
230 NEXT Z
240 B=0
250 FOR Z=1 TO 4
260 W=0
270 IF C(Z)<>G(Z)THEN 300
280 B=B+1
290 G(Z)=0
300 NEXT Z
310 FOR Z=1 TO 4
320 IF G(Z)=0 THEN 370
330 FOR J=1 TO 4
340 IF C(Z)<>G(J)THEN 360
350 W=W+1
360 NEXT J
370 NEXT Z
380 PRINT B;"black";
390 IF B=1 THEN 470
400 IF B<>1 THEN 490
410 PRINT "  ";W;"white";
420 IF W<>1 THEN 510
430 PRINT
440 IF B=4 THEN 530
450 NEXT CHANCES
460 GOTO 540
470 PRINT " ";
480 GOTO 410
490 PRINT "s";
500 GOTO 410
510 PRINT "s";
520 GOTO 430
530 PRINT ::"YOU GUESSED IT!"
540 PRINT ::"The code was ";
550 FOR Z=4 TO 1 STEP -1
560 PRINT STR$(C(Z));
570 NEXT Z
580 FOR Z=1 TO 3000
590 NEXT Z
600 FOR Z=1 TO 30
610 PRINT
620 NEXT Z
```

SIMON

Si tratta di un programma che mette alla prova la vostra memoria: dovete infatti tentare di ripetere una stringa di cifre in costante aumento. Il gioco inizia sommessamente con una sola cifra. Quando lo schermo si pulisce dovete inserire il numero che è stato appena visualizzato. Se indovinate, il computer ristampa la prima cifra e ve ne affianca una seconda. Pulitosi nuovamente lo schermo, dovete inserire i due numeri, nell'ordine esatto, digitando ENTER dopo ogni cifra. Se non commettete errori, questo stressante processo continua finché non riuscite a ripetere esattamente una sequenza di 10 cifre. Alla fine del gioco vi sarà dato un punteggio sulla vostra *performance*.

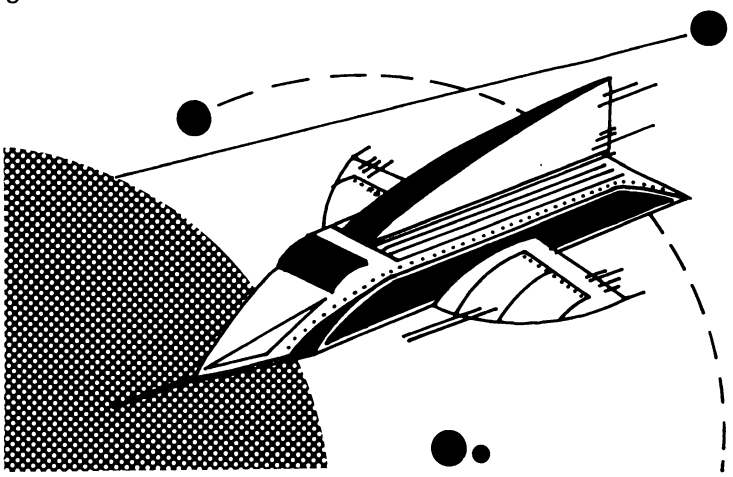


```
10 REM  SIMON
20 CALL CLEAR
30 CALL SCREEN(16)
40 CALL COLOR(2,7,16)
50 FOR T=1 TO 10
60 A(T)=INT(RND*10)+1
70 PRINT "********************":
80 NEXT T
90 FOR Y=1 TO 500
100 NEXT Y
110 FOR T=1 TO 10
120 CALL CLEAR
130 FOR Z=1 TO T
140 PRINT A(Z);
150 RT=500+110*A(Z)
160 CALL SOUND(50,RT,0)
170 NEXT Z
180 PRINT
190 FOR Y=1 TO 100+4*T
200 NEXT Y
210 CALL CLEAR
220 FOR Z=1 TO T
230 INPUT K
240 RT=500+110*K
250 CALL SOUND(-100,RT,0)
260 CALL CLEAR
270 IF K<>A(Z)THEN 380
280 PRINT K;
290 NEXT Z
300 FOR Y=1 TO 500
310 NEXT Y
320 CALL CLEAR
330 FOR Y=1 TO 200
340 NEXT Y
350 NEXT T
360 PRINT "YOU ARE THE CHAMP!"
370 END
380 PRINT :::
390 PRINT "THE NUMBER WAS ";
400 FOR Z=1 TO T
410 PRINT A(Z);
420 RT=500+100*A(Z)
430 CALL SOUND(100,RT,0)
440 NEXT Z
450 PRINT :
460 PRINT :
470 PRINT "YOU SCORED";79*T
```

DIAMOND FEVER

Ecco un altro grande gioco, che sfrutta in modo eccellente le possibilità grafiche del TI, scritto da Kevin Burfitt. Quando il gioco inizia vi trovate da soli su un asteroide sperduto nelle profondità dello spazio. Mentre esplorate il piccolo pianeta notate che la sua struttura rocciosa rivela la presenza di un'alta percentuale di elementi metallici. Essendo un minatore comprenderete immediatamente che estrarre questi metalli dall'asteroide significherebbe per voi la ricchezza.

I vostri sogni di benessere, però, svaniscono quando vi accorgete che all'interno dell'asteroide si cela una terribile astronave Vogon. Il vostro obiettivo ora è di passare da una parte all'altra dell'asteroide (partite da sinistra) per guadagnare l'uscita prima che il Vogon vi raggiunga. Per uscire dalla miniera dovete aprirvi un varco con la trivella: e se il metallo è duro da perforare, il diamante non si lascia neanche scalfire! Solo alla fine del gioco sarà visualizzata la dislocazione dei metalli e dei diamanti, che quindi è ancora un'incognita mentre cercate di eludere il Vogon. Nel programma è contenuta l'esposizone dettagliata di tutte le istruzioni.



```
10 REM  DIAMOND FEVER
20 CALL CLEAR
30 INPUT "DO YOU WANT
         INSTRUCTIONS ?":HUH$
40 IF SEG$(HUH$,1,1)<>"Y" THEN 290
50 PRINT "          DIAMOND FEVER"
60 PRINT "            ~~~~~~~"
70 PRINT :"          written by":
80 PRINT :"         Kevin Burfitt"
90 PRINT "             1983"
100 PRINT ::::::::::
110 FOR I=1 TO 1000
120 NEXT I
130 CALL CLEAR
140 PRINT "        You are alone, on an
    asteroid in deepest space.  While
    scanning the rock"
150 PRINT " you notice it contains an
    abnormaly high amount of metals,
    so you venture"
160 PRINT "down to see if you can mine
   any   WHEN   SUDDENLY   you realise
   that you are not "
170 PRINT "inside an asteroid but an
      evil VOGON spacecraft !!"::
180 PRINT " How can you escape from the
    maze of rock and rubble?  Fate was
    on your side, you got a look at the
    diamond"
190 PRINT "seam in the rock,but can you
    reach the exit before the "
200 PRINT :"   Vogon gets you ????"::
210 PRINT :"PRESS ANY KEY TO CONTINUE"
220 CALL KEY(0,KEY,STATUS)
230 IF STATUS THEN 240 ELSE 220
240 CALL SOUND(100,440,0)
250 PRINT ::"level 1 is easiest  level
 5 for the champions "::"you have your tr
 usty blaster with"
260 PRINT "you and it's running out of
 steam (uraniam actually) but it will c
 ut through rock"::"N.B. YOU ARE RUNNING
 OUT OF"
270 PRINT :"  gasp'  AIR  gasp'"::
280 PRINT "Use the arrow keys to move":
"Your blaster will  'bleep'  when you try
 to cut through rock"
290 HIGHEST_SCORER$="HIGH SCORE"
```

```
300 HIGH_SCORE=50
310 CALL CHAR(132,"995A3C3C3C3C2424")
320 DIAMOND=3
330 ROCK=1
340 SCORE=0
350 CALL CHAR(136,"55AA55AA55AA55AA")
360 DIM A(19,19)
370 FOR I=1 TO 19
380 FOR J=1 TO 19
390 A(I,J)=0
400 NEXT J
410 NEXT I
420 RANDOMIZE
430 INPUT "LEVEL OF DIFFICULTY ?":L
440 SC1=0
450 CALL CLEAR
460 IF (L<0)+(L>5)THEN 430
470 LEVEL=L
480 CALL COLOR(10,3,3)
490 CALL COLOR(11,2,2)
500 CALL COLOR(13,2,3)
510 CALL COLOR(14,2,16)
520 CALL CHAR(133,"0018245A18240000")
530 CALL CLEAR
540 CALL SCREEN(L+4)
550 AX=10
560 PRINT ::::::"  DIAMOND SEAM "::
570 AY=INT(RND*4)+2
580 FOR N=1 TO 30+(LEVEL)
590 DIR=INT(RND*5)+1
600 AX=AX+(DIR=1)-(DIR=3)
610 AY=AY-(DIR=2)-(DIR=4)+(DIR=5)
620 AX=AX-(AX<1)+(AX>19)
630 AY=AY-(AY<1)+(AY>19)
640 CALL HCHAR(AX,AY,136)
650 A(AX,AY)=DIAMOND
660 NEXT N
670 FOR N=1 TO 80+4*LEVEL
680 Y=INT(RND*19)+1
690 X=INT(RND*18)+2
700 IF A(Y,X)=DIAMOND THEN 760
710 IF A(Y,X)=ROCK THEN 770
720 A(Y,X)=0
730 IF ((Y=9)+(Y=10))*((X=18)
          +(X=19))THEN 770
740 A(Y,X)=ROCK
750 GOTO 770
760 A(Y,X)=0
770 NEXT N
```

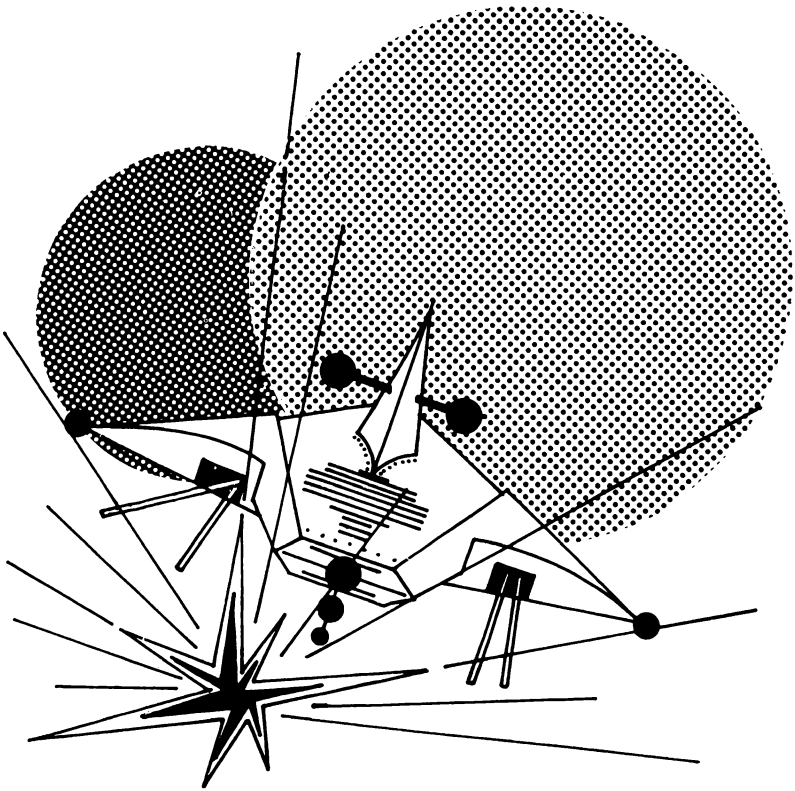
```
780 CALL CLEAR
790 Y=10
800 X=1
810 PRINT "LEVEL -> ";LEVEL:
820 R=Y
830 FOR I=1 TO LEN(STR$(SCORE))
840 CALL HCHAR(2,22+I,ASC(SEG$
          (STR$(SCORE),I,I)))
850 NEXT I
860 V=19
870 FOR I=1 TO LEN(STR$(HIGH_SCORE))
880 CALL HCHAR(6,22+I,ASC(SEG$(STR$
                  (HIGH_SCORE),I,I)))
890 NEXT I
900 C=0
910 FOR I=1 TO LEN(HIGHEST_SCORER$)
920 CALL HCHAR(5,22+I,ASC(SEG$
               (HIGHEST_SCORER$,I,I)))
930 NEXT I
940 CALL HCHAR(21,1,112,21)
950 CALL HCHAR(1,1,112,21)
960 CALL VCHAR(1,21,112,21)
970 CALL VCHAR(1,1,112,21)
980 FOR I=2 TO 20
990 CALL HCHAR(I,2,136,19)
1000 NEXT I
1010 CALL HCHAR(11,21,32)
1020 FOR N=1 TO 8-L
1030 C=C+1
1040 IF C>300+L*100 THEN 1910
1050 IF 50*LEVEL-C>0 THEN 1110
1060 TEST=TEST=0
1070 IF TEST=0 THEN 1090
1080 CALL SCREEN(16)
1090 IF TEST=-1 THEN 1110
1100 CALL SCREEN(L+4)
1110 IF C<20+L*100 THEN 1130
1120 CALL SOUND(-200,770,0,769,0,771,0)
1130 Z=X
1140 T=Y
1150 CALL KEY(0,KEY,STATUS)
1160 Y=Y+(KEY=ASC("E"))-(KEY=ASC("X"))
1170 X=X+(KEY=ASC("S"))-(KEY=ASC("D"))
1180 IF (Y=10)*(X=20)THEN 1560
1190 Y=Y-(Y<1)+(Y>19)
1200 X=X-(X<1)+(X>19)
1210 IF (R=Y)*(V=X)THEN 1910
1220 IF A(Y,X)=0 THEN 1330
1230 A(Y,X)=A(Y,X)-0.1
```

```
1240 IF A(Y,X)>0 THEN 1290
1250 SCORE=SCORE+LEVEL
1260 FOR I=1 TO LEN(STR$(SCORE))
1270 CALL HCHAR(2,22+I,ASC
          (SEG$(STR$(SCORE),I,I)))
1280 NEXT I
1290 X=Z
1300 Y=T
1310 CALL SOUND(-50,880,0)
1320 GOTO 1360
1330 REM
1340 CALL HCHAR(T+1,Z+1,111)
1350 CALL HCHAR(Y+1,X+1,132)
1360 REM
1370 NEXT N
1380 E=R
1390 S=V
1400 V=V-(V<X)+(V>X)
1410 IF A(R,V)=0 THEN 1440
1420 V=S
1430 GOTO 1470
1440 CALL HCHAR(E+1,S+1,111)
1450 CALL HCHAR(R+1,V+1,133)
1460 S=V
1470 R=R-(R<Y)+(R>Y)
1480 IF A(R,V)=0 THEN 1520
1490 R=E
1500 V=S
1510 GOTO 1540
1520 CALL HCHAR(E+1,S+1,111)
1530 CALL HCHAR(R+1,V+1,133)
1540 IF (R=Y)*(V=X)THEN 1910
1550 GOTO 1020
1560 CHECK=50*LEVEL-C
1570 IF CHECK<0 THEN 1640
1580 IF CHECK<300 THEN 1600
1590 CHECK=300
1600 SCORE=SCORE+CHECK+50
1610 IF SCORE>0 THEN 1640
1620 SCORE=0
1630 SC1=0
1640 GOSUB 2130
1650 SC2=SCORE-SC1
1660 SC1=SCORE
1670 PRINT "YOU GOT ";SC2;" THAT TIME";
1680 GOSUB 2010
1690 LEVEL=LEVEL+1
1700 L=L+1
1710 FOR I=1 TO 19
1720 FOR J=1 TO 19
```

```
1730 A(I,J)=0
1740 NEXT J
1750 NEXT I
1760 IF L<8 THEN 530
1770 L=1
1780 DIAMOND=DIAMOND+0.5
1790 ROCK=ROCK+0.2
1800 GOTO 530
1810 IF SCORE>HIGH_SCORE THEN 1850
1820 PRINT "THE HIGH SCORE IS ";HIGH_SCORE;
       " AND YOU GOT ";SCORE
1830 PRINT :::::::
1840 GOTO 310
1850 HIGH_SCORE=SCORE
1860 PRINT "THE HIGH SCORE IS ";
     HIGH_SCORE;" AND YOU GOT IT"
1870 INPUT "WHAT IS YOUR NAME ?"
     :HIGHEST_SCORER$
1880 IF LEN(HIGHEST_SCORER$)<11 THEN 310
1890 PRINT "NOT THAT LONG"
1900 GOTO 1870
1910 GOSUB 1960
1920 FOR I=1 TO 1000
1930 NEXT I
1940 CALL CLEAR
1950 GOTO 1810
1960 FOR I=1 TO 3
1970 CALL SOUND(100,220,0,219,0,218,0)
1980 CALL SOUND(1,110,30)
1990 NEXT I
2000 CALL SOUND(1500,185,0,184,0,183,0)
2010 FOR I=1 TO 19
2020 FOR J=1 TO 19
2030 IF A(I,J)=0 THEN 2090
2040 IF A(I,J)<ROCK+0.1 THEN 2070
2050 CALL HCHAR(I+1,J+1,112)
2060 GOTO 2100
2070 CALL HCHAR(I+1,J+1,136)
2080 GOTO 2100
2090 CALL HCHAR(I+1,J+1,111)
2100 NEXT J
2110 NEXT I
2120 RETURN
2130 FOR I=135 TO 216 STEP 10
2140 CALL SOUND(100,I,0,I+1,0,I+2,0)
2150 NEXT I
2160 CALL SOUND(200,40000,30)
2170 CALL SOUND(300,110,0,111,0,112,0)
2180 RETURN
```

FINAL FRONTIER

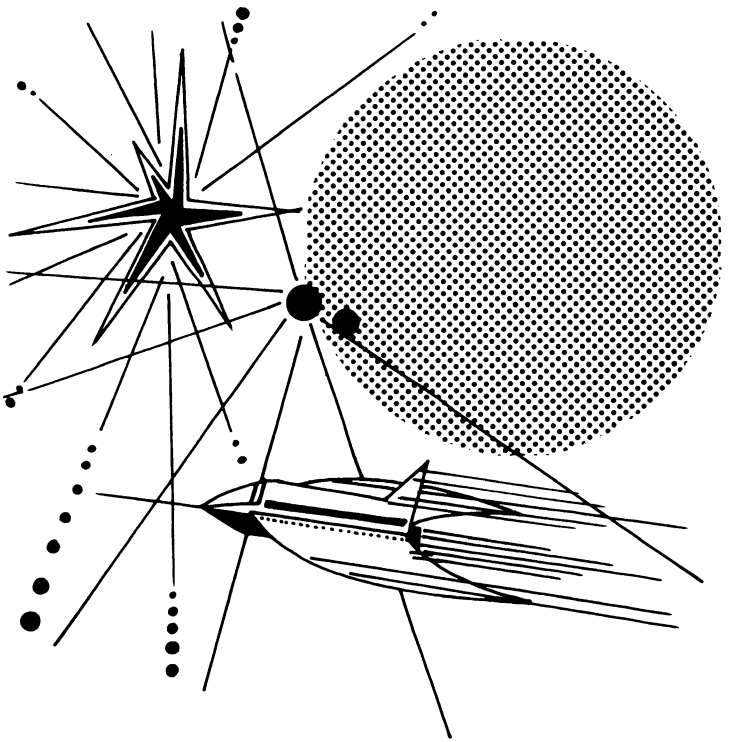
In questo gioco d'azione scritto da Damon Pillinger non avrete neanche il tempo di riprendere fiato!
Ancora una volta l'ultima linea di difesa della terra è nelle vostre mani. Usate i tasti contrassegnati dalle frecce per spostare verticalmente il vostro apparecchio e servitevi della barra spaziatrice per aprire il fuoco. Man mano che andate avanti, il gioco diventa più difficile.



```
10 REM   ** FINAL FRONTIER **
20 REM
30 REM  ** D&D PROGRAMMING *
40 REM
50 REM   ** OF GAMETRONICS **
60 REM  **** MELBOURNE *****
70 REM
80 REM  **** WRITTEN BY ****
90 REM  ****** DAMON *******
100 REM
110 LEVEL=1
120 CALL SCREEN(2)
130 K=28
140 CALL CLEAR
150 LEVEL=LEVEL+3
160 HP=10
170 RANDOMIZE
180 DED=0
190 CALL SCREEN(2)
200 CALL COLOR(9,6,1)
210 CALL COLOR(8,3,3)
220 CALL COLOR(10,4,1)
230 CALL COLOR(3,7,1)
240 CALL COLOR(11,11,1)
250 CALL COLOR(2,14,1)
260 DIM P1(3)
270 P1(1)=10
280 P1(2)=15
290 P1(3)=22
300 CALL CHAR(49,"8151565616ABC566")
310 CALL CHAR(50,"3136380FED9BCA71")
320 CALL CHAR(51,"5554765616FE6154")
330 CALL CHAR(40,"0080C0F08F7C7FF0")
340 CALL CHAR(41,"00000000C030FF00")
350 A$="000000000F3F0100"
360 B$="003F7F8EFCF0F800"
370 CALL CHAR(96,A$)
380 CALL CHAR(97,B$)
390 CALL CHAR(90,"FFFFFFFFFFFFFFFF")
400 CALL CHAR(104,A$)
410 CALL CHAR(48,"00000000000000FF")
420 CALL CHAR(105,B$)
430 CALL CHAR(112,A$)
440 CALL CHAR(113,B$)
450 CALL HCHAR(1,1,90,32)
460 CALL HCHAR(22,1,90,32)
470 FOR T=K TO 1 STEP -(LEVEL/10)
480 FOR Q=1 TO INT(LEVEL/3.3333333)
490 IF P1(Q)=999 THEN 560
```

```
500 CALL HCHAR(P1(Q),T,32,4)
510 P1(Q)=P1(Q)+(INT((RND*2)-.91))
520 IF (P1(Q)>2)*(P1(Q)<22)THEN 540
530 P1(Q)=INT(RND*10)+3
540 CALL HCHAR(P1(Q),T,88+Q*8)
550 CALL HCHAR(P1(Q),T+1,89+Q*8)
560 NEXT Q
570 CALL KEY(3,X,Y)
580 IF Y=0 THEN 690
590 CALL HCHAR(HP,2,32,2)
600 IF (X=69)*(HP>2)THEN 620
610 GOTO 630
620 HP=HP-1
630 IF (X=88)*(HP<21)THEN 650
640 GOTO 660
650 HP=HP+1
660 CALL HCHAR(HP,2,40)
670 CALL HCHAR(HP,3,41)
680 IF X=32 THEN 720
690 REM  END OF YOUR MOVING LOOP
700 NEXT T
710 GOTO 980
720 CALL HCHAR(HP,4,48,28)
730 CALL SOUND(-70,-7,0,300,7)
740 FOR Y=1 TO INT(LEVEL/3.3)
750 IF P1(Y)=HP THEN 800
760 NEXT Y
770 CALL HCHAR(HP,4,32,28)
780 CALL HCHAR(HP,3,41)
790 GOTO 700
800 FOR A=1 TO 5
810 CALL HCHAR(P1(Y),T,49,2)
820 CALL HCHAR(P1(Y),T,50,2)
830 CALL HCHAR(P1(Y),T,51,2)
840 CALL SOUND(-200,-7,0)
850 NEXT A
860 CALL HCHAR(HP,4,32,28)
870 CALL SOUND(-300,-7,0)
880 DED=DED+1
890 SCORE=SCORE+(LEVEL-2)*15
900 IF DED<>INT(LEVEL/3.3)THEN 960
910 LEVEL=LEVEL+1
920 IF LEVEL<>14 THEN 950
930 K=K/1.5
940 LEVEL=4
950 GOTO 170
960 P1(Y)=999
970 GOTO 660
980 FOR H=1 TO 5
```

```
990 FOR T=1 TO 16
1000 CALL SCREEN(T)
1010 NEXT T
1020 NEXT H
1030 CALL COLOR(8,2,16)
1040 DIM X$(4)
1050 X$(1)="YOU HAVE LOST.."
1060 X$(2)="THE ALIENS HAVE WON..."
1070 X$(3)=" "
1080 X$(4)="YOUR SCORE IS "
1090 FOR Y=1 TO 4
1100 FOR T=1 TO LEN(X$(Y))
1110 CALL HCHAR(Y+1,T,ASC(SEG$(X$(Y),T,1)))
1120 NEXT T
1130 NEXT Y
1140 FOR AA=1 TO 300
1150 NEXT AA
1160 CALL CLEAR
1170 PRINT "SCORE=";SCORE:,,,,
     "LEVEL";LEVEL-3,,,,,,
```



FENCED IN

Questo gioco, in cui difficilmente si riesce a vincere, è stato fornito da James Turner. Partendo dall'angolo superiore sinistro del video dovete raggiungere quello inferiore destro passando attraverso un campo di reti elettriche.
Se volete sopravvivere non dovete toccare più di cinque reticolati (ma l'efficace rappresentazione grafica della morte potrebbe indurvi ad un'autodistruzione deliberata!). Ogni volta che urtate contro una di queste reti ne viene automaticamente abbattuta qualche altra, e ciò, naturalmente, agevola il conseguimento della vostra meta.
In questo gioco di strategia non potete tornare sui vostri passi (sono consentiti solo spostamenti a destra e in basso) e quindi dovete calcolare in anticipo le vostre mosse. Servitevi del tasto 'M' per muovervi verso il basso e del tasto 'L' per spostarvi a destra.

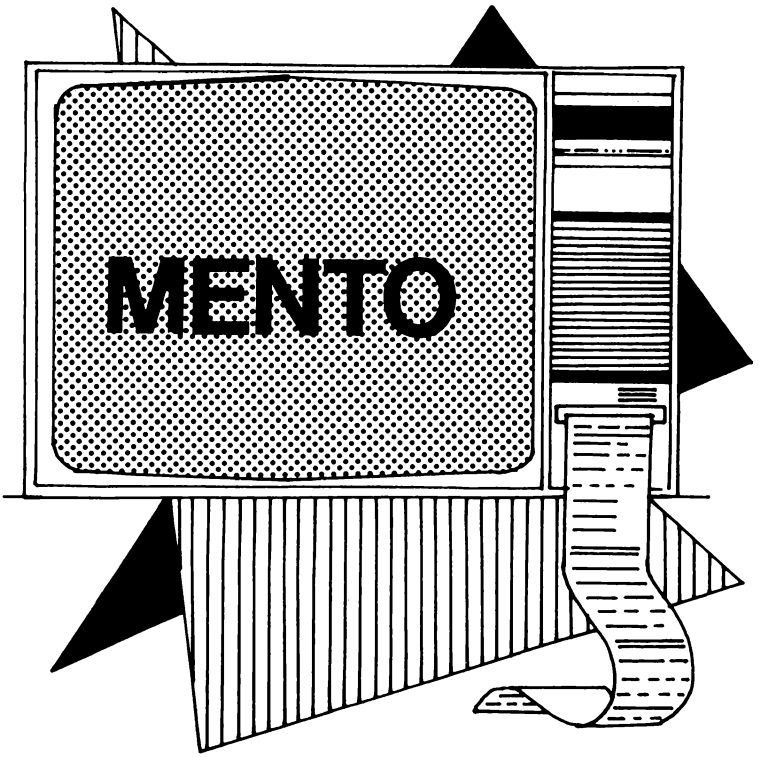

```
10 REM  FENCED IN
20 CALL CLEAR
30 PRINT TAB(9);"FENCED IN!"::
40 PRINT TAB(13);"By"::
50 PRINT TAB(8);"James turner"::::::::
60 FOR I=1 TO 1000
70 NEXT I
80 CALL CLEAR
90 CALL CHAR(129,"00FF00FF00FF00FF")
100 CALL CHAR(130,"AAAAAAAAAAAAAAAA")
```

```
110 CALL CHAR(131,"FF00FF00FF00FF00")
120 CALL CHAR(132,"5555555555555555")
130 CALL CHAR(133,"007F405F50575455")
140 CALL CHAR(134,"00FE02FA0AEA2AAA")
150 CALL CHAR(135,"AA2AEA0AFA02FE00")
160 CALL CHAR(136,"555457505F407F00")
170 CALL CHAR(144,"F0BCDFABD73B0F03")
180 CALL CHAR(152,"1818107C1028286C")
190 CALL CHAR(48,"C3C3C3C3C3C3FFFF")
200 CALL CHAR(112,"995A001818004281")
210 CALL CHAR(96,"99DBBD9999A5C3FF")
220 CALL CHAR(40,"40E04040F048241E")
230 FOR Z=1 TO 5
240 CALL CLEAR
250 CALL HCHAR(1,1,133)
260 CALL HCHAR(1,2,129,30)
270 CALL HCHAR(1,32,134)
280 CALL VCHAR(2,32,130,22)
290 CALL HCHAR(24,32,135)
300 CALL HCHAR(24,2,131,30)
310 CALL VCHAR(2,1,132,22)
320 CALL HCHAR(24,1,136)
330 CALL HCHAR(1,16,53)
340 LIFE=5
350 RANDOMIZE
360 FOR I=1 TO 200
370 R=INT(RND*22)
380 C=INT(RND*29)
390 CALL HCHAR(R+2,C+2,144,2)
400 NEXT I
410 A=2
420 B=2
430 CALL COLOR(3,7,16)
440 CALL COLOR(12,11,4)
450 CALL COLOR(16,16,4)
460 CALL COLOR(13,2,3)
470 CALL COLOR(14,2,3)
480 CALL HCHAR(23,31,48)
490 CALL HCHAR(22,31,32)
500 CALL KEY(0,K,S)
510 CALL HCHAR(A,B,152)
520 IF S=0 THEN 500
530 IF K=76 THEN 540 ELSE 600
540 CALL GCHAR(A,B+1,X)
550 IF X=144 THEN 750
560 IF B+1=32 THEN 500
570 B=B+1
580 CALL HCHAR(A,B-1,32)
590 GOTO 500
```

```
600 IF K=77 THEN 610 ELSE 500
610 CALL GCHAR(A+1,B,X)
620 IF X=144 THEN 750
630 IF A+1=24 THEN 500
640 A=A+1
650 CALL HCHAR(A-1,B,32)
660 IF A=23 THEN 670 ELSE 500
670 IF B=31 THEN 680 ELSE 500
680 CALL HCHAR(23,31,96)
690 PRINT "********YOU DID IT********"
700 CALL SOUND(150,784,0)
710 FOR H=1 TO 200
720 NEXT H
730 NEXT Z
740 GOTO 1040
750 CALL HCHAR(A,B,112)
760 CALL COLOR(11,7,4)
770 CALL SOUND(100,-3,0)
780 FOR I=1 TO 15
790 NEXT I
800 CALL SOUND(100,-3,0)
810 FOR I=1 TO 40
820 NEXT I
830 CALL SOUND(100,-3,0)
840 LIFE=LIFE-1
850 IF LIFE<0 THEN 910
860 CALL HCHAR(A-1,B-1,32,3)
870 CALL HCHAR(A,B-1,32,3)
880 CALL HCHAR(A+1,B-1,32,3)
890 CALL HCHAR(1,16,LIFE+48)
900 GOTO 500
910 CALL HCHAR(A,B,40)
920 FOR I=1 TO 3
930 CALL SOUND(400,262,0)
940 NEXT I
950 CALL SOUND(400,311,0)
960 CALL SOUND(300,294,0)
970 FOR I=1 TO 2
980 CALL SOUND(70,247,0)
990 CALL SOUND(300,262,0)
1000 NEXT I
1010 PRINT "TOO MANY TOUCHES !":"YOU ARE DEAD
1020 END
1030 GOTO 1030
1040 CALL CLEAR
1050 PRINT TAB(6);"CONGRATULATIONS!"::
1060 PRINT TAB(6);"YOU HAVE ESCAPED"::::
1070 GOTO 1070
1080 END
```

MENTO

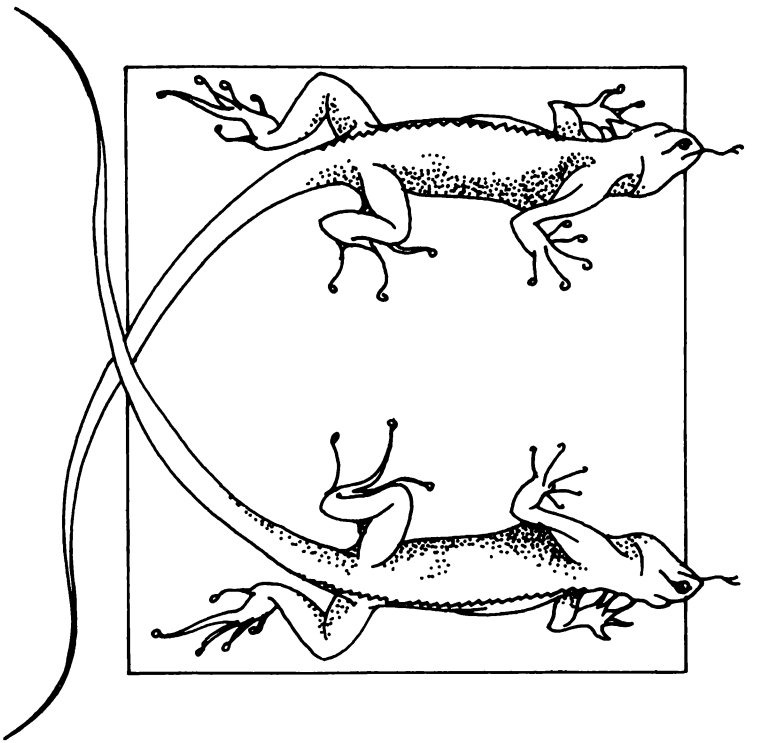
Questo programma mette alla prova la vostra capacità di calcolo mentale, dimostrando al tempo stesso le apparenti facoltà telepatiche del computer.
Sul vostro video comparirà una serie di istruzioni che dovrete eseguire singolarmente. Premete quindi il tasto ENTER e alla fine il computer sarà in grado di dirvi non solo la vostra età ma persino quanti soldi avete in tasca.



```
10 REM  MENTO
20 GOSUB 320
30 PRINT "MULTIPLY YOUR AGE BY"
40 PRINT "TWO, THEN ADD FIVE"
50 PRINT :::::
60 GOSUB 300
70 PRINT "NOW MULTIPLY THAT"
80 PRINT "BY FIFTY, AND"
90 PRINT "SUBTRACT 365"
100 PRINT :::::
110 GOSUB 300
120 PRINT "NOW ADD THE AMOUNT"
130 PRINT "OF CHANGE IN YOUR"
140 PRINT "POCKET"
150 PRINT ::::::
160 GOSUB 300
170 PRINT "NOW GIVE ME THE"
180 PRINT "NUMBER YOU'VE ENDED"
190 PRINT "UP WITH"
200 PRINT ::::::
210 INPUT A
220 A=A+115
230 B=INT(A/100)
240 A=A-100*B
250 GOSUB 320
260 PRINT "YOU HAVE";A;"CHANGE"
270 PRINT
280 PRINT "AND YOU'RE";B;"YEARS OLD"
290 END
300 PRINT
310 INPUT Z$
320 CALL CLEAR
330 FOR T=1 TO 1000
340 NEXT T
350 RETURN
```

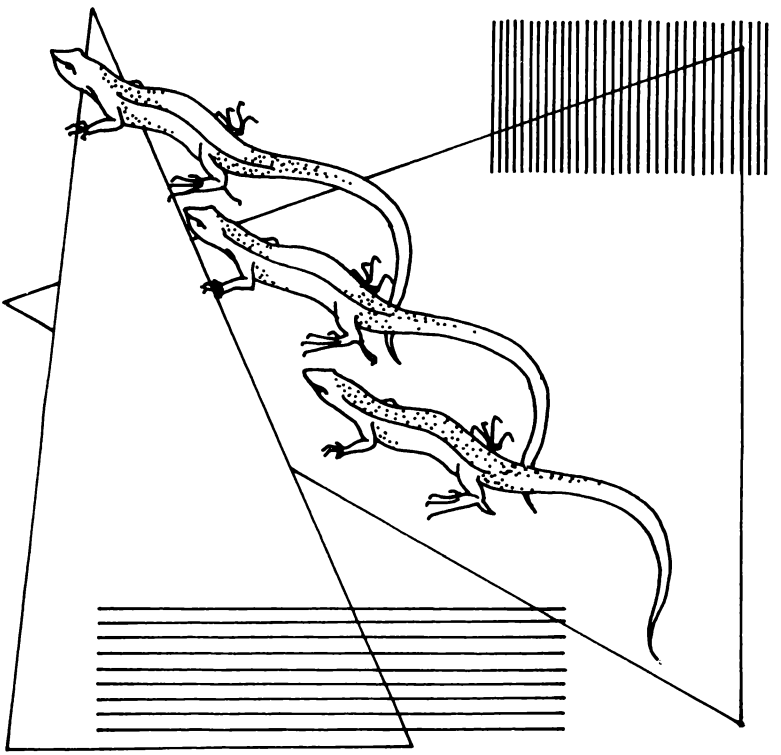
RACE OF THE LIZARD MEN

A mio parere i grafici di Wilton J. Faberge, ideati per essere definiti dal giocatore, somigliano vagamente a degli 'uomini-lucertola' che si arrampicano faticosamente sullo schermo contendendosi il serto d'alloro della vittoria. Dopo aver scommesso state a guardare quale di questi due uomini-lucertola riesce ad attraversare lo schermo per primo.
Se desiderate modificare il loro aspetto, mutate le stringhe A in H, nelle linee da 500 a 570.



```
10 REM  RACE OF THE LIZARD MEN
20 RANDOMIZE
30 CALL CLEAR
40 GOSUB 490
50 A1=6
60 A2=16
70 B1=2
80 C1=2
90 C2=2
100 B2=2
110 J=0
120 IF 2*(INT(J/2))=J THEN 180
130 C=128
140 D=130
150 E=129
160 F=131
170 GOTO 220
180 C=144
190 D=146
200 E=145
210 F=147
220 J=J+1
230 CALL HCHAR(A1,B1,C)
240 CALL HCHAR(A1,C1,32)
250 CALL HCHAR(A1,(B1+1),D)
260 CALL HCHAR((A1+1),C1,32)
270 CALL HCHAR((A1+1),B1,E)
280 CALL HCHAR((A1+1),(B1+1),F)
290 CALL HCHAR(A2,B2,C)
300 CALL HCHAR(A2,C2,32)
310 CALL HCHAR(A2,(B2+1),D)
320 CALL HCHAR((A2+1),C2,32)
330 CALL HCHAR((A2+1),B2,E)
340 CALL HCHAR((A2+1),(B2+1),F)
350 C1=B1
360 C2=B2
370 B1=B1+RND
380 B2=B2+RND
390 IF (B1>30)+(B2>30)<0 THEN 410
400 GOTO 120
410 IF B1<B2 THEN 440
420 PRINT ::"THE WINNER IS NUMBER ONE"
430 GOTO 430
440 PRINT ::"THE WINNER IS NUMBER TWO"
450 GOTO 450
460 J=2
470 GOTO 230
480 END
490 REM  DEFINE CHARACTERS
```

```
500 A$="0000000002060404"
510 B$="0000000001030202"
520 C$="C0E000C4C4C4F8C0"
530 D$="C0C0F88888081000"
540 E$="0000000609090501"
550 F$="0101120600010602"
560 G$="6070008080800B0"
570 H$="0000804080000000"
580 CALL CHAR(128,A$)
590 CALL CHAR(129,B$)
600 CALL CHAR(130,C$)
610 CALL CHAR(131,D$)
620 CALL CHAR(144,E$)
630 CALL CHAR(145,F$)
640 CALL CHAR(146,G$)
650 CALL CHAR(147,H$)
660 RETURN
```



ENCHANTED FOREST

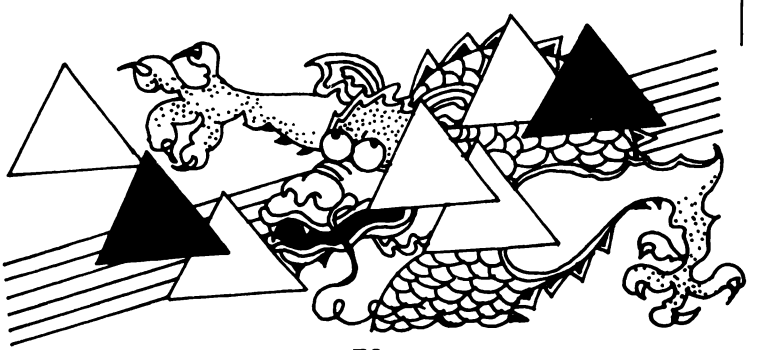
Questa mini-avventura, basata su un programma ideato da Toni Baker e adattato per il TI da Tim Hartnell, vi porta nel cuore di una ben strana foresta... costituita interamente di triangoli numerati. Di seguito vengono elencati i vari abitanti della Foresta Incantata e le conseguenze che dovrete sopportare entrando in uno dei loro triangoli:

FATE - non vi faranno del male, tuttavia verrete trasportati in un altro triangolo, scelto a caso nel bosco.

FOLLETTI - sono veramente maligni e vi uccideranno se solo oserete entrare in uno dei loro settori.

DRAGO - se finite nel suo triangolo siete un uomo morto!

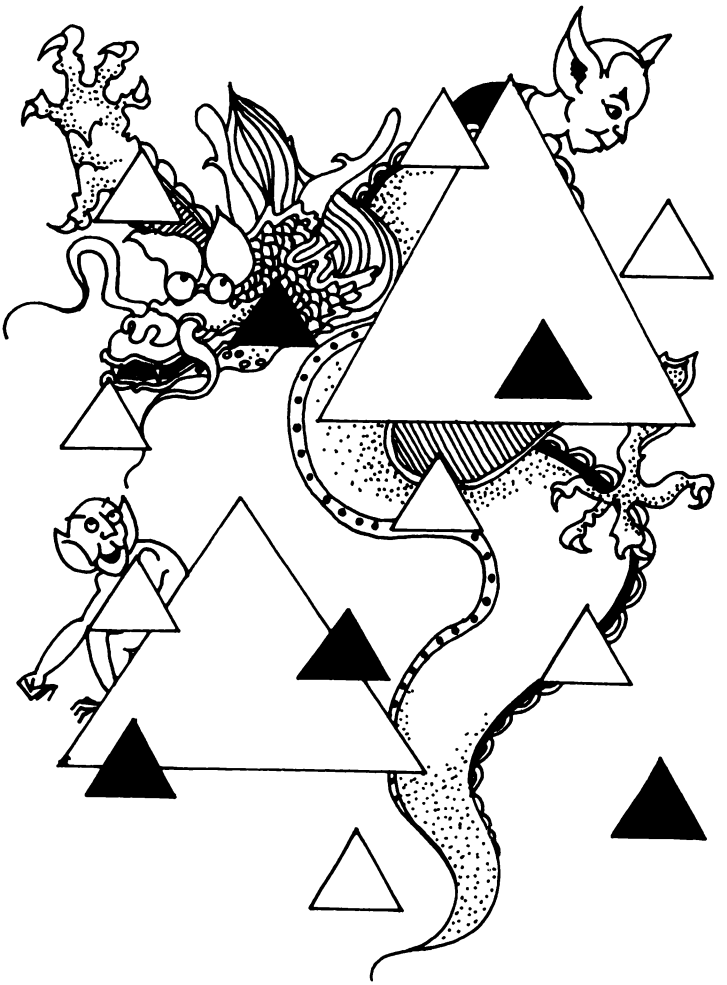
Lo scopo del gioco è l'uccisione del drago. A disposizione avete un numero limitato di frecce (la linea 40 ne prevede 5) che fate scoccare in qualsiasi settore (cioè dove pensate che si nasconda il drago), inserendo il numero del triangolo preceduto dal segno meno (per esempio, se volete tirare la freccia nel triangolo 41, inserite -41). Benché il drago si sposti ad ogni gioco, acquisterete presto una certa destrezza nell'individuare il suo possibile nascondiglio.



```
10 REM  ENCHANTED FOREST
20 DIM A(30),B(30)
30 CALL CLEAR
40 G=5
50 GOSUB 1000
60 PRINT ::
70 FOR Z=0 TO 30
80 A(Z)=0
90 NEXT Z
100 FOR Z=0 TO 10
110 Q=INT(RND*30)+1
120 IF A(Q)=1 THEN 110
130 A(Q)=1
140 B(Z)=Q+35
150 NEXT Z
160 FOR I=1 TO 20
170 PRINT "****************************"
180 NEXT I
190 GOSUB 1000
200 PRINT ::"THERE ARE FAIRIES HERE "
210 FOR Z=1 TO 500
220 NEXT Z
230 X=34+2*(RND*16)+1
240 X=INT(X)
250 Y=7
260 CALL CLEAR
270 FOR I=1 TO 15
280 PRINT
290 NEXT I
300 PRINT :::"YOU ARE NOW IN SECTOR";X
310 FOR J=1 TO 200
320 NEXT J
330 GOSUB 1000
340 Q=-1
350 FOR Z=0 TO 10
360 IF B(Z)<>X THEN 380
370 Q=INT(Z/5)
380 NEXT Z
390 IF Q=0 THEN 200
400 IF Q=1 THEN 850
410 IF Q=2 THEN 880
420 FOR U=1 TO 5
430 PRINT
440 NEXT U
450 PRINT "****************************"
460 PRINT ::
470 PRINT "YOU CAN MOVE TO";X-1;X+1;X+Y
480 GOSUB 1000
490 PRINT ::
```

```
500 FOR Z=0 TO 2
510 A(Z)=0
520 NEXT Z
530 FOR Z=0 TO 10
540 D=B(Z)-X
550 IF ABS(D)=1 THEN 900
560 IF D=Y THEN 900
570 NEXT Z
580 D=ABS(D)
590 IF (D=1)+(D=6)+(D=8)<0 THEN 920
600 PRINT ::"############################"
610 FOR J=1 TO 500
620 NEXT J
630 PRINT ::
640 IF A(0)=1 THEN 940
650 IF A(1)=1 THEN 960
660 GOSUB 1000
670 IF A(2)=1 THEN 980
680 Q=2
690 PRINT :
700 INPUT "YOUR MOVE? ":M
710 FOR J=1 TO 500
720 NEXT J
730 IF M<0 THEN 770
740 X=M
750 Y=-Y
760 GOTO 260
770 IF M=-B(10)THEN 410
780 G=G-1
790 PRINT ::
800 PRINT "YOU HAVE";G;" ARROWS LEFT"
810 PRINT
820 IF G>0 THEN 700
830 PRINT :"SO THE GAME IS OVER"
840 END
850 PRINT :"AND THE GOBLINS HAVE"
860 PRINT "KILLED YOU"
870 END
880 PRINT :"YOU HAVE FOUND THE DRAGON! "
890 END
900 A(INT(Z/5))=1
910 GOTO 570
920 A(2)=1
930 GOTO 600
940 PRINT "FAIRIES NEARBY"
950 GOTO 680
960 PRINT "GOBLINS NEARBY "
970 GOTO 680
980 PRINT ">> DRAGON NEARBY <<"
```

```
990 GOTO 680
1000 FOR MUSIC=1 TO INT(RND*6)+1
1010 NOTE=INT(RND*263)+262
1020 TIME=INT(RND*90)+20
1030 CALL SOUND(TIME,NOTE,0,
     (NOTE+9),0,(NOTE-6),0)
1040 NEXT MUSIC
1050 RETURN
```



FROG PLAGUE

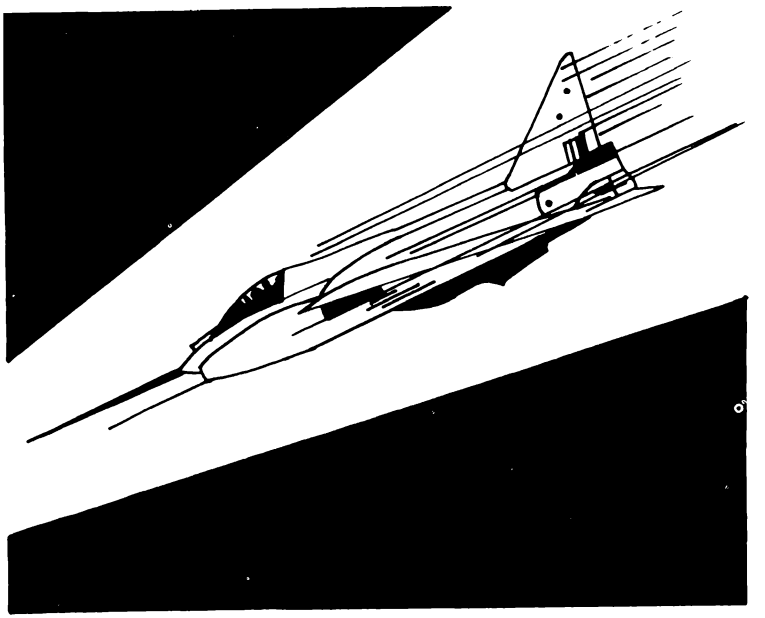
Questo gioco di Damon Pillinger mette nelle vostre mani la vita di una raganella.
Seguendo le indicazioni in REM, usate i tasti 'D' e 'S' per spostarvi rispettivamente a destra e a sinistra: tenterete così di evitare gli oggetti che risalgono lo schermo dirigendosi contro di voi. Dovete stare alla larga da foglie, rocce, pezzi di legno, ponti (estremamente difficile) e da un motoscafo.
Nonostante tutti i vostri sforzi, in questo gioco prima o poi perderete. Le pareti che compaiono sullo schermo, inseguendovi ovunque, rendono la vostra fuga dai pericolosi oggetti ancora più difficile.

```
10 REM   **** FROG PLAGUE ****
20 REM   **** GAMETRONICS ***
30 REM   *** DAMON PILLINGER ***
40 REM
50 REM  USE S AND D TO MOVE
60 REM  LEFT AND RIGHT. THE OBJECTS
70 REM  YOU ARE AVOIDING ARE:
80 REM    LEAVES, MOTORBOAT
90 REM    ROCKS, LOGS AND BRIDGES
100 REM
110 REM  THE WALLS MOVE IN AS
```

```
120 REM  THE GAME CONTINUES...
130 CALL CLEAR
140 CALL SCREEN(2)
150 SCORE=0
160 CALL COLOR(9,14,2)
170 CALL COLOR(10,7,2)
180 CALL COLOR(11,11,2)
190 CALL COLOR(12,8,2)
200 CALL COLOR(4,2,15)
210 CALL COLOR(2,3,2)
220 LEV=2
230 T=09
240 CALL CHAR(40,"0000C33C187E9918")
250 CALL CHAR(104,"0070FE7E1E1E0101")
260 CALL CHAR(96,"1010383838383810")
270 CALL CHAR(112,"303C3C3C783C3C00")
280 CALL CHAR(57,"FF101010101010FF")
290 CALL CHAR(120,"0001501878F8FA20")
300 CALL CHAR(56,"")
310 CALL CLEAR
320 FOR LEV=1 TO 11 STEP .02
330 PRINT TAB(RND*((22-(LEV*2))+10));
    CHR$((INT((RND*4)))*8+96)
340 IF (LEV>0)*(INT(RND*15)=10)THEN 530
350 CALL HCHAR(23,((22-(LEV*2)+10)+1),56)
360 CALL HCHAR(8+LEV,T-1,32,4)
370 CALL HCHAR(9+LEV,T-1,32,4)
380 CALL GCHAR(10+LEV,T,M)
390 SCORE=SCORE+LEV/4
400 IF (M<>32)*(M<>40)THEN 520
410 CALL HCHAR(10+LEV,T,40)
420 CALL KEY(3,X,Y)
430 IF Y=0 THEN 500
440 IF X<>83 THEN 470
450 IF T<2 THEN 470
460 T=T-1
470 IF X<>68 THEN 500
480 IF T>((22-(LEV*2)+10))THEN 500
490 T=T+1
500 NEXT LEV
510 GOTO 310
520 GOTO 550
530 CALL HCHAR(21,1,57,
    (INT(RND*((22-(LEV*2))+10)))+1)
540 GOTO 350
550 CALL CLEAR
560 PRINT "YOU ARE DEAD"
570 PRINT :"YOUR SCORE IS";SCORE
580 PRINT ,,,,,,
```

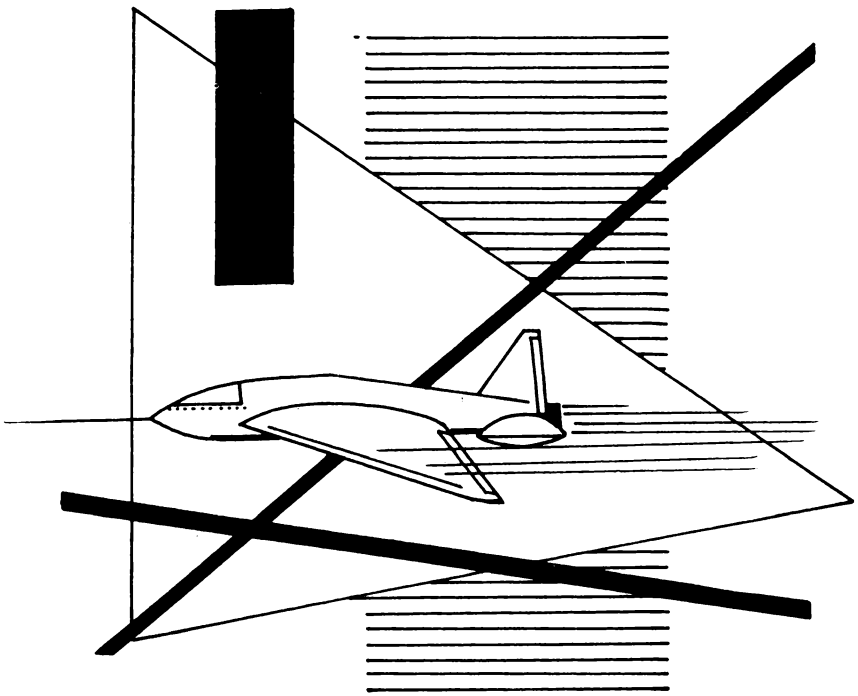
KAMIKAZE PILOT

Prendete in mano il vostro destino e pilotate questo minuscolo aereo giù per il tunnel tortuoso e serpeggiante. Per tutto il corso del gioco il velivolo tenderà a sbandare. Mentre tentate di tenerlo sotto controllo, premete un tasto qualsiasi e correggetene l'inclinazione.
Potete scegliere tra nove livelli di gioco: a quelli contrassegnati dai numeri più alti corrisponde un grado di difficoltà maggiore (sono praticamente impossibili). Quando riterrete di essere sufficientemente pratici del gioco, rendetelo ancora più spinoso riducendo il numero degli spazi (ne sono previsti 7) nelle linee 260 e 270.



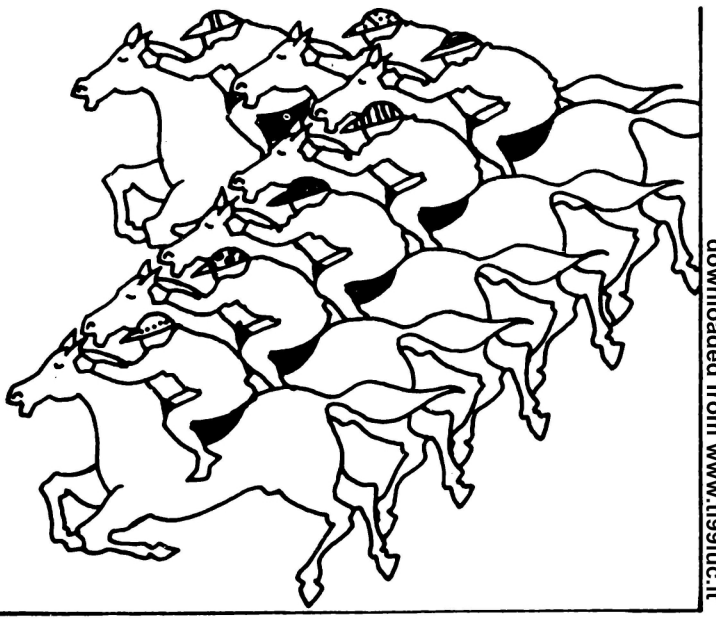
```
10 REM  KAMIKAZE PILOT
20 GOTO 50
30 SIDE=SIDE-1
40 GOTO 320
50 CALL CLEAR
60 PRINT "ENTER A NUMBER FROM 1 TO 9"
70 PRINT :"1 IS EASIEST, 9 IS HARDEST"
80 CALL KEY(3,KY,KK)
90 IF KK=0 THEN 80
100 GOTO 360
110 Y=10*Y
120 CALL CLEAR
130 SC=0
140 GOSUB 480
150 SIDE=16
160 DOWN=12
170 CALL GCHAR(17,SIDE,X)
180 IF X=65 THEN 620
190 CALL HCHAR(DOWN,SIDE,66)
200 FOR T=1 TO Y
210 NEXT T
220 D=INT(D+RND*3-RND*3)
230 IF D<3 THEN 540
240 IF D>29 THEN 560
250 CALL HCHAR(DOWN,SIDE,32)
260 PRINT TAB(D);"A        A"
270 PRINT TAB(D);"A        A"
280 SC=SC+27
290 CALL KEY(3,KY,KK)
300 IF KK=0 THEN 30
310 SIDE=SIDE+2
320 IF SIDE<4 THEN 580
330 IF SIDE>28 THEN 600
340 GOTO 170
350 END
360 Y=-(KY=131)
370 Y=-2*(KY=132)
380 Y=-3*(KY=135)
390 Y=-4*(KY=130)
400 Y=-5*(KY=142)
410 Y=-6*(KY=140)
420 Y=-7*(KY=129)
430 Y=-8*(KY=134)
440 Y=-9*(KY=143)
450 CALL CLEAR
460 CALL SCREEN(5)
470 GOTO 110
480 A$="49221C08493E1C08"
490 B$="0810202018040810"
```

```
500 CALL CHAR(66,A$)
510 CALL CHAR(65,B$)
520 D=10
530 RETURN
540 D=3
550 GOTO 260
560 D=29
570 GOTO 260
580 SIDE=4
590 GOTO 320
600 SIDE=28
610 GOTO 340
620 PRINT ::"THE END!!"
630 CALL HCHAR(DOWN,SIDE,66)
640 CALL SOUND(2000,440,2,659,2,880,2)
650 CALL SOUND(2000,-2,2)
660 PRINT ::
670 PRINT "YOU SCORED";SC
680 FOR T=1 TO 1000
690 NEXT T
```



DARING DAMON

Finalmente potete lasciarvi andare al vostro piccolo vizio segreto: il gioco d'azzardo. Il programma mette in luce la grande efficacia della grafica mobile nel BASIC ordinario.
La gara ippica, base del gioco, ha inizio non appena avete fissato le vostre scommesse. Scendendo lungo il percorso i vostri cavalli (per le cui zampe sono stati usati dei grafici di grande effetto) si lasciano alle spalle tutti gli ostacoli.
Nel programma sono dettagliatamente esposte tutte le istruzioni.
Se volete poi modificare la velocità del galoppo, cambiate il numero casuale generato all'interno del programma. Alterando i DATA, potete anche cambiare il nome dei cavalli.

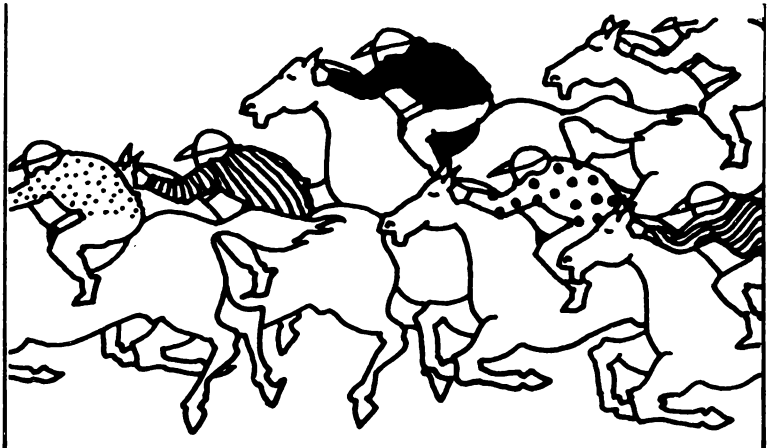


```
10!REM  *** DARING DAMON ***
20 REM
30 REM  * BY D&D
40 DIM ML(10),BET(10),HOR(10)
50 RANDOMIZE
60 CALL CLEAR
70 CALL SCREEN(3)
80 FOR T=1 TO 10
90 ML(T)=2000
100 NEXT T
110 FOR T=9 TO 12
120 CALL COLOR(T,2,4)
130 NEXT T
140 PRINT "welcome to the texan race":
       "meeting ":,,"your credit with
        daring":"damon is set at
    $2000":"there is no maximum bet."
150 GOSUB 1230
160 CALL CLEAR
170 INPUT "HOW MANY PLAYERS  ":PLA
180 IF PLA>8 THEN 170
190 CALL CLEAR
200 PRINT "hi,":"   i am daring damon.
     i am":"called this because of my
     incredibly high odds.",,,," be
     warned i have inside     tips."
210 PRINT "so heed my odds.but i have
     been known to be wrong."
220 GOSUB 1230
230 DIM HNA$(10),ODD(6)
240 HNA$(1)="NAG HEAP     "
250 HNA$(2)="COSMIC RUN   "
260 HNA$(3)="PRINCE       "
270 HNA$(4)="DASSY        "
280 HNA$(5)="FFFLYER      "
290 HNA$(6)="EXTENDER     "
300 HNA$(8)="VECTOR       "
310 HNA$(9)="FLAT FEET    "
320 HNA$(10)="MULE         "
330 HNA$(7)="DOPPY        "
340 CALL CLEAR
350 CALL SCREEN(13)
360 PRINT "      *DARING DAMON*"
370 PRINT ,,,,"NO.NAME          ODDS"
380 CHANCE=-CHANCE
390 RR$="1111111111"
400 PRINT "**********************",,
410 FOR T=1 TO 5
```

```
420 J=INT(RND*10)+1
430 IF SEG$(RR$,J,1)<>"1" THEN 420
440 RR$=SEG$(RR$,1,J)&"2"&SEG$(RR$,J+1,10)
450 ODD(T)=INT(RND*(30))+20
460 DT$="*"&STR$(T)&"**"&(SEG$(HNA$(J),
    1,20))&"*"&STR$(ODD(T))&"/1"
470 PRINT DT$:
480 NEXT T
490 PRINT ,,
500 FOR T=1 TO PLA
510 PRINT "PLAYER ";T;" WHICH HORSE?"
520 INPUT HOR(T)
530 IF HOR(T)<6 THEN 560
540 PRINT "THERE ARE ONLY 5 HORSES
        IN THE RACE.  STUPID....."
550 GOTO 520
560 NEXT T
570 CALL CLEAR
580 FOR T=1 TO PLA
590 PRINT "PLAYER ";T;"WHAT IS YOUR BET"
600 IF ML(T)=0 THEN 650
610 INPUT BET(T)
620 IF BET(T)<=ML(T)THEN 650
630 PRINT ,,"BE VERY LUCKY YOU
              ONLY HAVE $";ML(T)
640 GOTO 610
650 NEXT T
660 GOSUB 1230
670 CALL CLEAR
680 DIM HOR$(2,2),RL$(2)
690 HOR$(1,1)="0000009F7F1F2040"
700 HOR$(1,2)="0040F0C080008040"
710 HOR$(2,1)="000000030F030201"
720 HOR$(2,2)="00081EF8F0E01020"
730 RL$(1)="0000FFAAFF444444"
740 RL$(2)="0000FF55FF111111"
750 GAT1$="0003010101030408"
760 GAT3$="00C0808080C02010"
770 GAT2$="1121C142FF000000"
780 GAT4$="88848342FF000000"
790 FOR T=1 TO 6
800 A(T)=4
810 NEXT T
820 CALL CHAR(104,RL$(1))
830 CALL HCHAR(7,1,104,32)
840 CALL HCHAR(14,1,104,32)
850 CALL CHAR(112,GAT1$)
860 CALL CHAR(114,GAT2$)
```

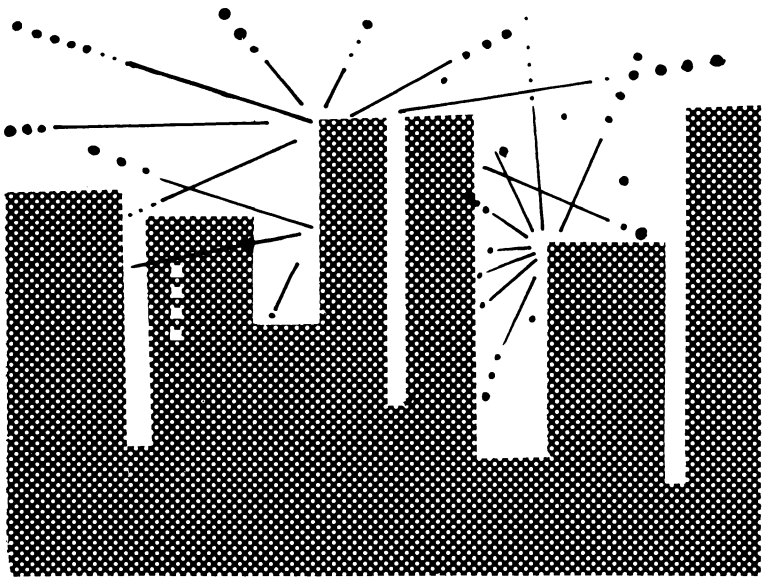
```
870 CALL CHAR(113,GAT3$)
880 CALL CHAR(115,GAT4$)
890 CALL HCHAR(7,5,115)
900 CALL CHAR(96,"00")
910 CALL COLOR(9,2,4)
920 CALL COLOR(10,2,3)
930 CALL HCHAR(8,1,96,192)
940 FOR T=1 TO 50000
950 FOR F=1 TO 5
960 CALL HCHAR(8+(F-1),INT(A(F)),97)
970 CALL HCHAR(8+(F-1),INT(A(F))+1,98)
980 CALL HCHAR(8+(F-1),INT(A(F))-3,96,3)
990 F1=RND*3
1000 OD1=(ODD(F)/60)
1010 CHANCE=F1-OD1
1020 IF CHANCE>0 THEN 1040
1030 CHANCE=-(CHANCE)
1040 A(F)=A(F)+CHANCE
1050 IF T>2 THEN 1100
1060 CALL HCHAR(6,4,112)
1070 CALL HCHAR(6,5,113)
1080 CALL HCHAR(7,4,114)
1090 CALL HCHAR(7,5,115)
1100 IF T<>5 THEN 1150
1110 CALL HCHAR(6,4,32)
1120 CALL HCHAR(7,4,104)
1130 CALL HCHAR(6,5,32)
1140 CALL HCHAR(7,5,104)
1150 GH=INT((SIN(F/6))*2)+1
1160 CALL CHAR(98,HOR$(GH,2))
1170 CALL CHAR(97,HOR$(GH,1))
1180 CALL CHAR(104,RL$(GH))
1190 IF A(F)>20 THEN 1270
1200 IF A(F)>29 THEN 1320
1210 NEXT F
1220 NEXT T
1230 PRINT ,,"PRESS ANY KEY TO CONTINUE"
1240 CALL KEY(3,X,Y)
1250 IF Y=0 THEN 1240
1260 RETURN
1270 CALL HCHAR(6,28,112)
1280 CALL HCHAR(6,29,113)
1290 CALL HCHAR(7,28,114)
1300 CALL HCHAR(7,29,115)
1310 GOTO 1200
1320 FOR J=1 TO 3
1330 HI=0
1340 FOR T=1 TO 5
1350 IF A(T)<A(HI)THEN 1370
```

```
1360 HI=T
1370 NEXT T
1380 WI(J)=HI
1390 A(HI)=0
1400 GOTO 1490
1410 NEXT J
1420 FOR T=1 TO PLA
1430 ML(T)=ML(T)-INT(BET(T))
1440 IF ML(T)>=0 THEN 1470
1450 ML(T)=0
1460 PRINT "PLAYER ";T;" ROTTE
     N EGGS.  YOU HAVE BEEN BUSTED."
1470 NEXT T
1480 GOTO 1590
1490 FOR GG=1 TO 3
1500 FOR T=1 TO PLA
1510 IF HOR(T)<>WI(GG)THEN 1540
1520 PRINT :"PLAYER ";T;"COLLECTS"
1530 ML(T)=ML(T)+(INT(BET(T)*(ODD(T)/(GG+1))))
1540 REM
1550 NEXT T
1560 PRINT ,,
1570 GOTO 1420
1580 NEXT GG
1590 PRINT "NO.  MONEY LEFT",,
1600 PRINT
1610 FOR T=1 TO PLA
1620 PRINT T;"     ";ML(T)
1630 NEXT T
1640 GOTO 220
```



CITY BOMBER

Uno dei più amati video-giochi del passato trova qui un superbo riadattamento. In City Bomber sgancerete ordigni su una schiera di grattacieli per spianare una pista d'atterraggio al vostro aereo.
Il programma adotta una grafica efficacissima e prevede persino un' 'esplosione accidentale' che varia ad ogni gioco. Questo programma è un osso veramente duro e ve ne accorgerete presto.
Per aprire il fuoco basta premere un tasto qualsiasi.



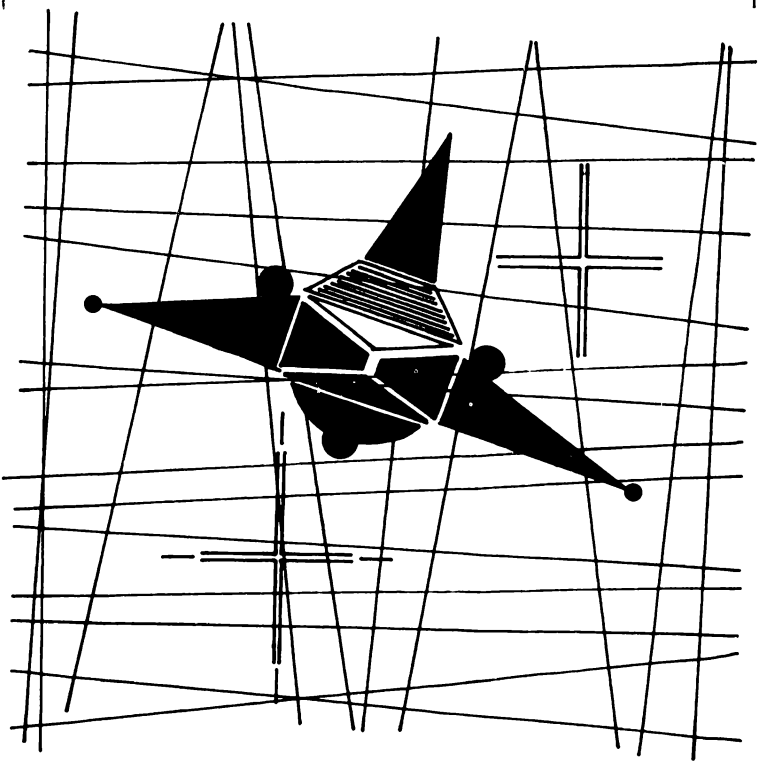
```
10 REM **** CITY BOMBER ****
20 REM
30 REM ***** WRITTEN BY *****
40 REM ** DAMON PILLINGER **
50 REM
60 RANDOMIZE
70 BM$="1A527258DB648F72984D7188305136356
  1376A69AC99E798E7E907BB9851691183175710
  1487006603083068OE707F0C0C070B707A70707
  E707A70E707DED70D8D0"
```

```
80 LEV=1
90 CALL CLEAR
100 FOR T=65 TO 85
110 A$=SEG$(BM$,((RND*100)+1),16)
120 CALL CHAR(T,A$)
130 NEXT T
140 HP=3
150 VP=(LEV)/2+5
160 CALL SCREEN(3)
170 CALL CLEAR
180 FF=32
190 CALL HCHAR(21,1,70,32)
200 CALL COLOR(2,2,14)
210 CALL COLOR(3,2,10)
220 CALL COLOR(4,2,7)
230 CALL COLOR(5,2,9)
240 CALL COLOR(6,2,9)
250 CALL COLOR(7,2,9)
260 CALL COLOR(8,2,9)
270 CALL CHAR(96,"000080C47E7F")
280 CALL CHAR(40,"7F41417F7F414141")
290 CALL CHAR(48,"FF999999FF9999FF")
300 CALL CHAR(56,"18FF3CC3C33CFFFF")
310 CALL CHAR(64,"FF99FF99FF99FF99")
320 FOR T=4 TO 28
330 S=INT(RND*10)
340 CALL VCHAR(20-S,T,
    ((INT(RND*4)*8)+40),S+1)
350 NEXT T
360 CALL KEY(3,X,Y)
370 IF CO<>0 THEN 420
380 IF Y<>0 THEN 400
390 GOTO 520
400 A1=VP+1
410 A2=HP
420 REM
430 CALL HCHAR(A1,A2,32)
440 A1=A1+1
450 IF A1<21 THEN 480
460 CO=0
470 GOTO 360
480 CALL GCHAR(A1,A2,M)
490 CALL HCHAR(A1,A2,111)
500 IF M<>32 THEN 650
510 CO=1
520 CALL HCHAR(VP,HP,96)
530 CALL HCHAR(VP,HP-1,32)
540 CALL SOUND(-100,-6,0)
550 CALL SOUND(-100,-7,1)
```

```
560 IF FF<>32 THEN 740
570 HP=HP+1
580 IF (HP=29)*(VP=20)THEN 800
590 CALL GCHAR(VP,HP,FF)
600 IF HP<=30 THEN 360
610 HP=3
620 VP=VP+1
630 CALL HCHAR(VP-1,28,32,3)
640 GOTO 360
650 FOR T=-9 TO -5
660 CALL HCHAR(A1,A2,T+84)
670 CO=0
680 CALL SOUND(-100,
    (INT(T/3)-3),0,600,5)
690 NEXT T
700 CALL HCHAR(A1,A2,32,1)
710 CALL SOUND(-200,110,0,400,
    0,300,0,-8,3)
720 CALL HCHAR(21,1,70,32)
730 GOTO 360
740 FOR T=1 TO 16 STEP .3
750 CALL SCREEN(T)
760 CALL SOUND(-100,T*10+110,0,-(T/2),1)
770 NEXT T
780 PRINT ,,,,,:"YOU CRASHED ON LEVEL ";LEV
790 END
800 FOR Y=1 TO 40
810 CALL SCREEN((RND*5)+1)
820 CALL SOUND(50,110+(Y*10),0,
    -(((RND*7)+1)),1)
830 NEXT Y
840 PRINT ,,,,LEV+1
850 LEV=LEV+2
860 FOR I=1 TO 400
870 NEXT I
880 GOTO 140
```

ARECIBO ATTACK

Questo gioco semplice, ma estremamente efficace, è stato scritto da James Turner; si tratta di una « sparatoria nello spazio » che tende a misurare la vostra capacità di valutare le distanze. Dovete tentare di abbattere il maledetto alieno che sta attraversando il vostro cielo.
Più l'extraterrestre si abbassa e più aumenta la sua velocità di discesa.
Per vincere il gioco dovete distruggere almeno cinque alieni. Aprite il fuoco servendovi della barra spaziatrice, usate il tasto '.' per spostarvi a destra e il tasto 'Z' per muovervi verso sinistra.

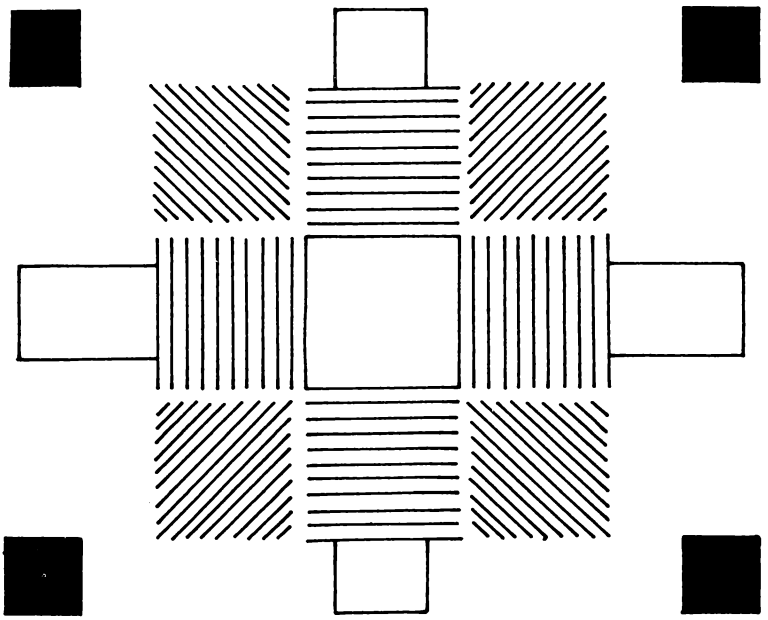


```
10 REM  ARECIBO ATTACK
20 CALL CLEAR
30 PRINT TAB(7);"ARECIBO ATTACK":,,
40 PRINT TAB(13);"By"::
50 PRINT TAB(8);"James Turner"::::::
60 GOSUB 860
70 RANDOMIZE
80 CO=0
90 R=3
100 P=16
110 M=16
120 CALL CLEAR
130 CALL SCREEN(2)
140 S=INT(24*RND)+1
150 ST=INT(32*RND)+1
160 CO=CO+1
170 IF CO=35 THEN 210
180 CALL COLOR(2,16,2)
190 CALL HCHAR(S,ST,46)
200 GOTO 140
210 CALL CHAR(128,"18183C66FFE7BDFF")
220 CALL CHAR(136,"003E49497F491422")
230 CALL COLOR(13,12,2)
240 CALL COLOR(14,9,2)
250 CALL COLOR(12,7,2)
260 CALL KEY(0,K,0)
270 CALL HCHAR(22,M,128)
280 CALL HCHAR(R,P,136)
290 IF K<>90 THEN 350
300 M=M-2
310 IF M<=1 THEN 320 ELSE 330
320 M=1
330 CALL HCHAR(22,M+2,32)
340 GOTO 410
350 IF K<>46 THEN 400
360 M=M+2
370 IF M>=32 THEN 380 ELSE 390
380 M=32
390 CALL HCHAR(22,M-2,32)
400 IF K=32 THEN 490
410 P=P+1
420 CALL HCHAR(R,P-1,32)
430 IF P<>33 THEN 480
440 P=2
450 GH=MEN+2
460 R=R+GH
470 IF R>=22 THEN 760
480 GOTO 260
490 FOR BP=17 TO R STEP -4
```

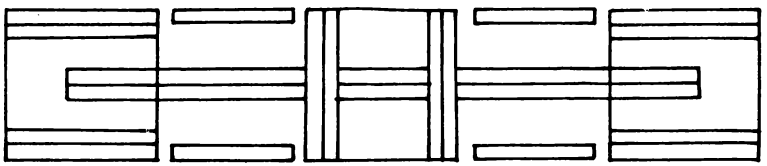
```
500 CALL VCHAR(BP,M,124)
510 CALL HCHAR(BP+4,M,32)
520 P=P+1
530 IF P<>33 THEN 560
540 P=2
550 R=R+GH
560 CALL HCHAR(R,P-1,32)
570 CALL HCHAR(R,P,136)
580 CALL SOUND(150,-4,1)
590 CALL GCHAR(BP,M,X)
600 NEXT BP
610 CALL VCHAR(1,M,32,24)
620 CALL VCHAR(1,32,32,24)
630 IF X=136 THEN 640 ELSE 260
640 CALL SOUND(1000,-5,0)
650 CALL SCREEN(10)
660 CALL SOUND(1000,-6,0)
670 CALL SCREEN(2)
680 CALL SOUND(1000,-7,0)
690 CALL SCREEN(10)
700 PRINT "HIT THE MARTIAN!"
710 GOSUB 860
720 MEN=MEN+1
730 IF MEN=5 THEN 820
740 CO=0
750 GOTO 70
760 CALL SCREEN(4)
770 PRINT :"YOU HAVE BEEN INVADED !"
780 PRINT "BETTER LUCK NEXT SHOT !"
790 CALL SOUND(2000,200,0,110,0)
800 GOSUB 860
810 GOTO 70
820 CALL SCREEN(4)
830 PRINT "YOU HAVE TRIUMPHED !"
840 GOSUB 860
850 GOTO 70
860 FOR I=1 TO 2000
870 NEXT I
880 RETURN
```

GRAPHIC DEMONSTRA-TIONS

Ecco tre grandi programmi di dimostrazione grafica scritti da Damon Pillinger.
Con il primo, 'Ebauche', si delinea sullo schermo un incantevole motivo di linee ricavato da una serie di quadrati. Come indica il nome stesso, "240 Colors" visualizza invece le 240 tinte che possono essere prodotte dal TI. L'ultimo programma, "Graphical Girder", si serve di linee prodotte a caso (e provenienti sia da destra che da sinistra) per generare un disegno in continua evoluzione.

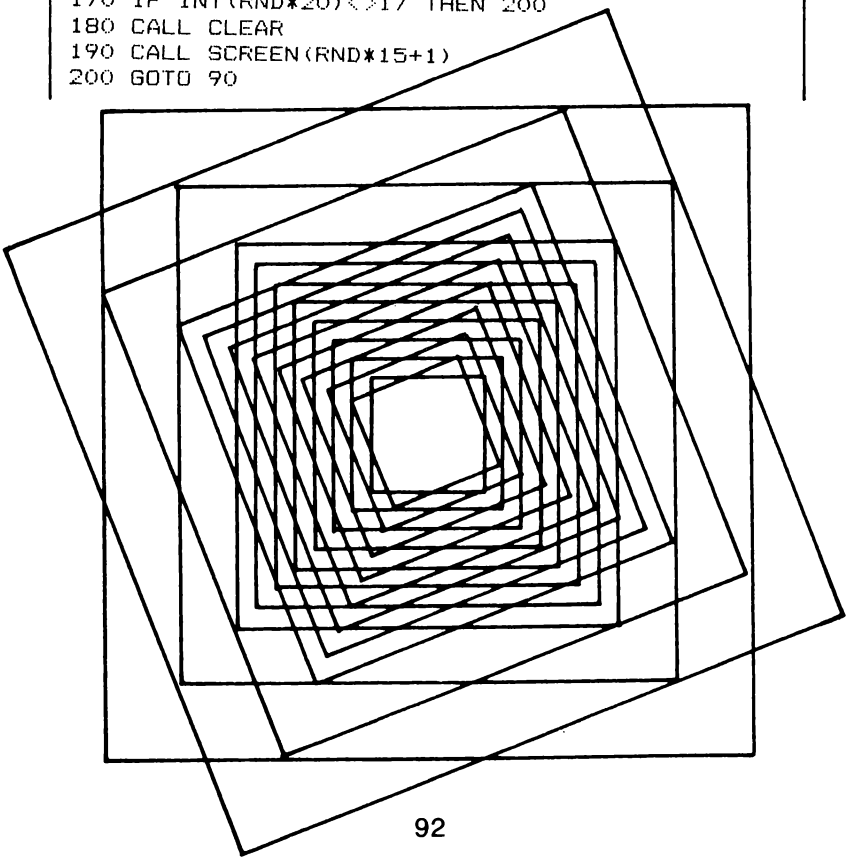


```
10 REM **** EBAUCHE ****
20 CALL SCREEN(4)
30 CALL CLEAR
40 FOR T=2 TO 10
50 CALL CHAR(24+(T*8),"")
60 CALL COLOR(T,T,T)
70 NEXT T
80 FOR T=1 TO 10
90 CALL HCHAR(T,T,32+(T*8),20-(T*2))
100 CALL HCHAR(20-T,T,32+(T*8),20-(T*2))
110 CALL VCHAR(T,T,32+(T*8),20-(T*2))
120 CALL VCHAR(T,19-T,32+(T*8),20-(T*2))
130 NEXT T
140 FOR Y=1 TO 8
150 A(Y)=Y
160 NEXT Y
170 FOR Y=1 TO 8
180 A(Y)=A(Y)+1
190 IF A(Y)<>9 THEN 210
200 A(Y)=1
210 CALL COLOR(Y+1,2,A(Y)+1)
220 NEXT Y
230 GOTO 170
```



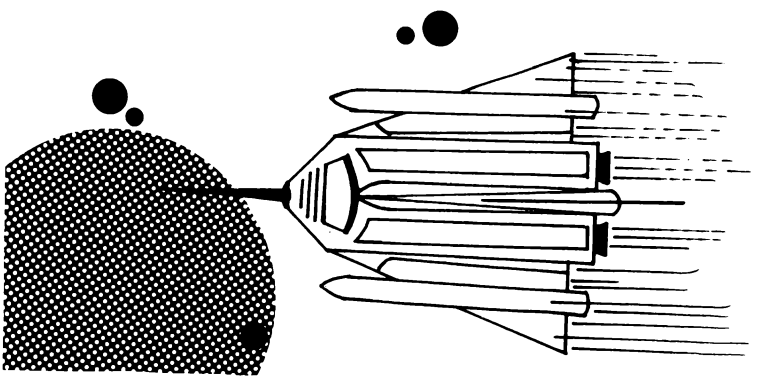
```
10 REM **** 240 COLORS ****
20 REM
30 REM *** BY DAMON ***
40 REM
50 CALL CLEAR
60 INPUT "SPEED  (1-300)":SP
70 CALL CHAR(97,"AA55AA55AA55AA55")
80 CALL HCHAR(1,1,97,(32*24))
90 FOR T=2 TO 16
100 FOR J=1 TO 16
110 FOR G=1 TO SP
120 NEXT G
130 CALL COLOR(9,T,J)
140 NEXT J
150 NEXT T
160 CALL CLEAR
170 PRINT ,,,,,,,,"240 COLORS"
```

```
10 REM  GRAPHICAL GIRDER
20 REM
30 CALL SCREEN(2)
40 CALL CLEAR
50 FOR T=2 TO 12
60 CALL CHAR(40+(T-2)*8,"")
70 CALL COLOR(T,T+2,T+2)
80 NEXT T
90 A=3+INT(RND*(28))
100 B=3+INT(RND*(18))
110 CALL VCHAR(1,A,40+INT(((RND*10)*8)),B)
120 CALL HCHAR(B,31-A,
    40+INT(((RND*10)*8)),A)
130 A=INT(RND*28)+3
140 B=INT(RND*18)+3
150 CALL VCHAR(22-B,A,
     40+INT(((RND*10)*8)),B)
160 CALL HCHAR(B,3,40+INT(((RND*10)*8)),A)
170 IF INT(RND*20)<>17 THEN 200
180 CALL CLEAR
190 CALL SCREEN(RND*15+1)
200 GOTO 90
```



CHALLENGER

Quest'efficacissimo gioco a grafica mobile è stato realizzato da Wayne Southwick. Scritto in *Extended* BASIC, il programma fa largo uso di *sprites*.
All'inizio del gioco vi sarà chiesto di inserire il vostro nome. Quindi la linea 210 vi comunicherà: « Penso che questo programma di tipo spaziale sarà di vostro gradimento », ma ciò soltanto a condizione che disponiate dello "speech unit"; in caso contrario potete cancellare questa linea (assieme alla 710).
Dopo aver inserito il vostro nome ricevete alcune istruzioni. Siete alla guida dell'astronave bianca che si trova vicino al centro dello schermo. Dovete cercare di evitare le numerose navette spaziali colorate che vi passano accanto procedendo a velocità diverse. Se rimanete inattivi il vostro veicolo comincerà a salire. Premete il tasto 'X' per riportarlo in basso. Se urtate contro le barriere rosse che delimitano il margine superiore ed inferiore dello schermo, il gioco finisce.
Il vostro apparecchio è munito di schermi che funzionano ad intermittenza. Se siete veramente alla disperazione, potete tentare di passare attraverso un'astronave degli alieni. Tuttavia se in quel momento i vostri schermi non sono operanti sarete uccisi, ponendo fine al gioco.



```
10 REM CHALLENGER
20 REM
30 REM WAYNE SOUTHWICK
40 REM
50 CALL CHARSET :: CALL MAGNIFY(1)
60 CALL CHAR(128,"191A1C1818385898")
70 CALL COLOR(0,16,1)
80 CALL COLOR(13,7,1)
90 CALL COLOR(1,16,1)
100 CALL COLOR(2,16,1)
110 CALL COLOR(3,16,1)
120 CALL COLOR(4,16,1)
130 CALL COLOR(10,16,1)
140 CALL COLOR(5,16,1)
150 CALL COLOR(6,16,1)
160 CALL COLOR(7,16,1)
170 CALL COLOR(8,16,1)
180 CALL COLOR(12,9,1)
190 CALL CLEAR :: CALL SCREEN(2):: DISPLAY
    AT(10,10):"CHALLENGER" :: CALL HCHAR(
    11,12,128,10)
200 PRINT "    PRESS X TO MOVE YOUR
    SPACE-SHIP DOWN"
210 CALL SAY("I+THINK+YOU+WILL+LIKE+
   THIS+SPACE+TYPE+PROGRAM")
220 CALL CHAR(120,"FF7F3F3F7F3F1E04")
230 CALL CHAR(121,"40E1F3F7F3F3F7FF")
240 CALL CLEAR :: CALL SCREEN(2)
250 DISPLAY AT(1,1):"FOR THE FIRST TWO
  100 POINTS" :: DISPLAY AT(3,1):"   YOU WI
  LL GET DIFFERENT" :: DISPLAY AT(5,1)
  :"          ATTACKERS."
260 DISPLAY AT(10,8):"PLEASE ENTER YOUR NAME: "
270 ACCEPT AT(13,8)SIZE(-7)VALIDATE
             (UALPHA)BEEP:A$
280 CALL CLEAR
290 CALL CHAR(136,"FFFFFFFFFFFFFFFF"):
             : CALL COLOR(14,9,1)
300 CALL HCHAR(2,1,136,32)
310 CALL HCHAR(24,1,136,32)
320 CALL HCHAR(3,1,120,32)
330 CALL HCHAR(23,1,121,32)
340 CALL CHAR(43,"DBE728181828E7DB")
350 CALL CHAR(111,"0F1E3EFAFA3E1E0F")
360 CALL SPRITE(#1,111,16,90,150)
370 FOR F=2 TO 26 STEP 5
380 CALL SPRITE(#F,43,3,20+F*5,20,0,50)
390 NEXT F
```

```
400 CALL CHAR(45,"00183C7EFF7E7E5A")
410 CALL SPRITE(#7,45,8,160,80,0,30)
420 CALL SPRITE(#8,45,8,120,1,0,20)
430 CALL SPRITE(#9,45,11,50,1,0,20)
440 CALL SPRITE(#10,45,11,65,1,0,20)
450 CALL CHAR(61,"0F1E3EFAFA3E1E0F")
460 LET CHEN=CHEN+1 :: IF CHEN=200 THEN
    800 :: IF CHEN>100 THEN 750
470 CALL COINC(#1,3,150,26,G):
              : IF G=-1 THEN 620
480 CALL COINC(ALL,0):: IF 0=-1 THEN 620
490 CALL COINC(#1,162,150,10,P):
              : IF P=-1 THEN 620
500 LET TIME=TIME+1 :: DISPLAY AT(1,1):
    A$&", YOUR SCORE IS: "&STR$(TIME)
510 CALL COINC(#1,3,150,26,G):
                   : IF G=-1 THEN 620
520 CALL COINC(ALL,0):: IF 0=-1 THEN 620
530 CALL COINC(#1,162,150,10,P):
                   : IF P=-1 THEN 620
540 CALL KEY(1,K,S)
550 IF S=0 THEN Y=-15
560 IF K=0 THEN Y=15
570 CALL MOTION(#1,Y,X)
580 CALL COINC(#1,3,150,26,G):
              : IF G=-1 THEN 620
590 CALL COINC(ALL,0):: IF 0=-1 THEN 620
600 CALL COINC(#1,162,150,10,P):
                   : IF P=-1 THEN 620
610 GOTO 460
620 CALL POSITION(#1,R1,C1)
630 CALL DELSPRITE(ALL):: CALL SOUND
       (680,880,0,-5,0):: CALL SCREEN(2)
640 CALL COLOR(10,11,1)
650 CALL HCHAR(5,2,111,31)
660 CALL VCHAR(5,2,111,18)
670 CALL VCHAR(5,32,111,18)
680 CALL HCHAR(18,2,111,31)
690 DISPLAY AT(12,8):"G A M E  O V E R" :
   : DISPLAY AT(16,8):A$&"'S SCORE IS "&ST
   R$(TIME)
700 DISPLAY AT(1,1):"
              " :: DISPLAY AT(20,4):" WANT TO
    PLAY AGAIN Y OR N.Y"
710 CALL SAY("TO+PLAY+AGAIN+PRESS+ENTER+OR+N
    +THEN+ENTER+FOR+NO")
720 ACCEPT AT(21,28)SIZE(-1)BEEP
    VALIDATE("YN"):G$ :: IF G$="Y" THEN 900
```

```
730 FOR D=1 TO 800 :: NEXT D
740 CALL CLEAR :: END
750 CALL CHAR(43,"183C7EFFFF7E3C18")
760 CALL COINC(ALL,Q):: IF Q=-1 THEN 620
770 CALL CHAR(43,"E7C38100008 1C3E7")
780 CALL COINC(ALL,Q):: IF Q=-1 THEN 620
790 GOTO 470
800 CALL DELSPRITE(ALL)
810 CALL SPRITE(#1,61,16,90,150)
820 CALL SPRITE(#7,128,8,150,90,0,40)
830 CALL SPRITE(#9,128,4,32,8,0,20)
840 CALL SPRITE(#11,128,6,48,24,0,40)
850 CALL SPRITE(#13,128,8,64,40,0,60)
860 CALL SPRITE(#15,128,10,80,56,0,80)
870 CALL SPRITE(#17,128,12,96,72,0,20)
880 CALL SPRITE(#19,128,14,112,88,0,40)
890 GOTO 470
900 CALL CLEAR :: RUN 240
```

Come scrivere programmi migliori

Inventando e sviluppando i vostri programmi di gioco
Del Direttore della Collana, Tim Hartnell

Impiegare il proprio tempo battendo i programmi come quelli che vi abbiamo presentato va benissimo, ma verrà certo il momento in cui deciderete di mettere a punto dei programmi di gioco personali. In questa sezione del libro vorrei quindi discutere alcune idee che potranno esservi d'aiuto nella realizzazione dei vostri 'games', rendendo piacevole la fase d'elaborazione e — cosa ancora più importante — divertente il momento del gioco.

CHIAREZZA D'IDEE

Vi accorgerete che in molti casi (per non dire sempre) il programma che state elaborando per il vostro computer comincia a vivere di vita propria già durante la stesura, deviando progressivamente dall'idea che vi eravate prefissi all'inizio della programmazione. Ciò nonostante nelle fasi preliminari di lavoro è importante avere un'idea chiara dei vari elementi che entreranno in gioco.
Questo mio consiglio è meno banale di quanto ptrebbe sembrare. Naturalmente se state pensando ad un gioco in cui « inghiottite alcune pillole iperricostituenti e dovete inseguire dei fantasmi attraverso un labirinto », sapete di non poter usare lo schema di programmazione valido, ad esempio, per un gioco che vi trasporta all'interno di una Quercia Incantata dove vivono elfi e folletti. In ogni caso dovete superare lo stadio elementare dell'« Ora mi scrivo un'avventura » per elaborare invece una serie di punti come: (a) l'oggetto del gioco; (b) l'effetto visivo che si vuole ottenere; (c) le variabili ed i nomi variabili necessari; (d) il tipo di input per il giocatore; (e) le modalità per la derminazione della vittoria e della sconfitta, eccetera.
Esaminiamo queste voci una ad una.

OGGETTO DEL GIOCO

Di norma è possibile sintetizzare questo punto in poche parole: « Trovare il tesoro degli Aztechi », « Distruggere il maggior numero possibile di asteroidi prima di esaurire le navi spaziali » o « Giocare una partita a scacchi ». Pur essendo di rapida esecuzione, questa fase iniziale nella produzione del gioco non deve essere tralasciata. Vi consiglio, anzi, di mettere per iscritto la vostra formula — che può consistere in una sola frase, ma che potrebbe anche raggiungere e superare il paragrafo se fosse necessario mettere a punto più "quadri" con scenari differenti.
Certamente potete accantonare il vostro obbiettivo originale se il corso preso dal programma durante l'elaborazione vi sembra più soddisfacente. Indipendentemente da questo fatto è importante che abbiate qualcosa di concreto cui mirare, se non volete perdere ore ed ore gingillandovi inutilmente con il computer.

VISUALIZZAZIONE

L'esperienza personale mi insegna che è estremamente utile fare qualche schizzo del tipo di visualizzazione grafica che si vuole ottenere a programma inserito. Una volta che avete preparato il disegno (e non importa quanto sia approssimativo purché evidenzi tutti gli elementi che volete avere sul video, con le relative posizioni e dimensioni), l'idea alla base del vostro programma si cristallizzerà più facilmente. Non solo capirete immediatamente come scrivere certe parti del codice in modo da conseguire l'obbiettivo del gioco, ma potrete anche valutare l'opportunità o meno di realizzare il programma nella forma in cui lo avevate ideato. Lasciando sullo schermo tutti gli elementi che vi proponevate, il gioco potrebbe forse risultate troppo complicato, in altri casi potrebbe accadere che il video appaia in gran parte vuoto e che dobbiate quindi escogitare uno scenario di gioco più complesso.
Vi posso assicurare che prima di iniziare la programmazione vi tornerà utile schematizzare l'effetto visivo finale, e specialmente se state ideando 'arcade games' o giochi

di simulazione. Otterrete così delle indicazioni relative alle variabili da adottare, ai grafici definibili dall'utente e al miglior tipo di input da proporre ai giocatori per conseguire una buona interazione tra partners.
Come probabilmente sapete, nei giochi di simulazione il computer riproduce una realtà esterna — ad esempio la conduzione di un'impresa, di una guerra o di un aeroporto — consentendovi di provare (approssimativamente) le esperienze che affrontereste partecipando realmente ad un'impresa del genere. I giochi di simulazione non presentano grandi difficoltà di stesura — non di codificazione, almeno — ma richiedono una metodica e scrupolosa impostazione del programma.
In un mio libro, « The ZX Spectrum Explored » (Sinclair Browne, 1982), è contenuto un programma dal titolo piuttosto inverosimile: « Lavorare per l'Uomo ». Il giocatore è posto a dirigere un'industria la cui forza lavoro, notevolmente variabile, viene impiegata per fabbricare un favoloso prodotto detto 'Zibby'. Due o tre volte a settimana al dirigente viene sottoposta una relazione industriale in base alla quale egli (o ella) deve stabilire quanti operai assumere o licenziare (se ci riesce), la quantità di Zibbies che costituisce l'obbiettivo di produzione della settimana, eccetera.
Questa relazione è la chiave del programma e per definire il gioco ho iniziato col tracciare uno schizzo degli elementi da visualizzare. La mia bozza era all'incirca la seguente:

RELAZIONE INDUSTRIALE: SETTIMANA 5

Il capitale liquido ammonta a $ 2657,92
I magazzini contengono 12 Zibbies per un valore di $ 169,68
Ciascuno viene venduto a $ 14,14 e costa al produttore $ 7,41
La forza lavoro è costituita da 7 persone
Il salario individuale è di $ 41
Questa settimana le spese salariali ammontano a $ 287
Ogni operaio può fabbricare 10 Zibbies alla settimana, per una produzione totale di 70 Zibbies.
Delineato questo schema potevo andare avanti. Come vedete dalla bozza si possono ricavare utili indicazioni circa le variabili da adottare. Tanto per cominciare sapevo

di dover controllare il numero della settimana, il capitale, il contenuto dei magazzini (e il suo valore) e così via.
Ho potuto sperimentare che una volta completato lo schema il resto della programmazione non presentava particolari difficoltà.
Uno schizzo eseguito secondo queste indicazioni vi offrirà una guida immediata alle variabili di cui dovrete servirvi.

VARIABILI UTILI

Generalmente tendo ad usare delle variabili che abbiano una certa attinenza con i vari elementi del gioco; con questo procedimento, infatti, evito di fare una lista delle variabili adottate e dei loro oggetti di riferimento. Per esempio potrei usare SA per settimana, CL per capitale liquido, PZ per il costo di produzione di ogni Zibby e VZ per il suo prezzo di vendita. Così, posto che il numero di Zibbies corrisponda a Z, potrei calcolare il valore globale della merce in mio possesso moltiplicando Z (numero di Zibbies) per VZ (prezzo di vendita), e il totale dei costi di produzione moltiplicando Z per PZ (costo di produzione). Naturalmente se vendessi tutti gli Zibbies di cui dispongo il mio profitto sarebbe dato da Z×VZ - Z×FZ. Scoprirete che seguendo questo tipo di logica è molto più facile tenersi costantemente aggiornati sugli sviluppi del programma.

INPUT DEL GIOCATORE

L'essenziale è scrivere giochi divertenti e facili da eseguire. Non conta nulla disporre del miglior programma che sia mai stato ideato con gli asteroidi, se poi il gioco risulta difficoltoso perché avete sistemato il pulsante per aprire il fuoco proprio accanto al tasto di 'rotazione'.
Per i comandi 'su', 'giù', 'destra' e 'sinistra', in molti programmi si adottano automaticamente i tasti di controllo cursore o quelli contrassegnati dalle frecce, senza considerare che per il giocatore è estremamente scomodo servirsene. Date uno sguardo alla vostra tastiera e cercatene di migliori. Per i programmi che consentono solo spostamenti verso destra e sinistra io adotto spesso

la 'M' e la 'Z', utilizzando la barra spaziatrice per far fuoco. L'uso di questi tasti mi sembra logico, non si perde tempo per impararli ed è estremamente semplice ricordarli quando il gioco è in atto. Allo stesso modo tendo ad usare 'A' (su) e 'Z' (giù) per la mano sinistra ed i tasti 'maggiore di' e 'minore di' per la destra e la sinistra (facendo rilevare ai giocatori che i simboli < e > muovono nelle direzioni in cui sono rispettivamente rivolti)
Quando è possibile usate INKEY$ o GET$, evitando così ai giocatori di adoperare i tasti RETURN o ENTER per avviare il programma.

CONCLUSIONI DEL GIOCO

È necessario definire e illustrare chiaramente ai giocatori le modalità che regolano la vittoria e la sconfitta.
Per vincere dovete far saltare in aria tutti gli alieni. Perdete automaticamente se un extraterrestre atterra mentre disponete ancora di qualche navicella spaziale o, semplicemente, quando avete esaurito le astronavi. Nei giochi a due, esce sconfitto chi per primo perde sette pezzi o viene ucciso per tre volte, in altri casi il gioco si conclude solo quando la differenza tra i due punteggi è tre, sette o un qualsiasi altro numero che vi aggradi.
Elaborate queste disposizioni ed esponetele con linearità ai giocatori. Si tratti di liberare la parte sinistra del video da hippies in protesta o di registrare la bella somma di 7,3 miliardi di dollari, lo scopo del gioco deve essere sempre chiaro allo sfidante e possibile da conseguire. Ai livelli più alti diventa veramente frustrante trovarsi di fronte a delle « condizioni di vittoria » irraggiungibili. Non fate che frodare i giocatori se ponete degli obbiettivi che i vari impedimenti inseriti nel gioco rendono impossibile conseguire; e non importa quante difficoltà incontriate elaborando il programma.
Spero che questi cinque punti vi suggeriscano qualche idea per continuare a scrivere programmi che, pur essendo di semplice realizzazione, possano risvegliare in voi e nei vostri amici il piacere del gioco.

Glossario

A

Accumulatore — Il posto all'interno del computer entro il quale si compiono calcoli aritmetici e dove i risultati di tali calcoli sono immagazzinati.

Algoritmo — La serie di passi compiuti dal computer per risolvere un particolare problema.

Alfanumerico — Questo termine è generalmente usato in relazione ad una tastiera. Esempio: "è una tastiera alfanumerica". Ciò significa che la tastiera ha sia lettere sia numeri. Indica anche "la serie di caratteri" del computer e comprende i numeri e le lettere che il computer può riprodurre sul video.

ALU (Arithmetic/Logic Unit) — La parte del computer adibita ad operazioni aritmetiche (come l'addizione e la sottrazione) e in cui vengono prese le decisioni.

AND — Un'operazione logica "booleana" che il computer usa nei suoi processi decisionali. È basata sull'algebra di "Boole", un sistema sviluppato dal matematico George Boole (1815-64). Nell'algebra di Boole le variabili di un'espressione sono soggette alle operazioni logiche come AND, OR, NOR, ecc.

ASCII — Sta per American Standard Code for Information Exchange (Codice Americano Standard per lo Scambio di Informazioni) ed è il sistema in codice più largamente usato per la lingua inglese con caratteri alfanumerici. Ci sono 128 lettere maiuscole e minuscole, cifre e alcuni caratteri speciali. Il codice ASCII trasforma i simboli e le istruzioni di controllo in combinazioni binarie a sette "bit".

ASSEMBLER — Un programma che trasforma i programmi scritti nel linguaggio Assembly in codice macchina (che il computer può comprendere in modo diretto). Il linguaggio Assembler è un linguaggio di programmazione a basso livello che usa le combinazioni facilmente memorizzabili di due o tre lettere per rappresentare una particolare istruzione che l'Assembler poi trasforma affinché il computer possa comprenderlo. Esempio: ADD (aggiungere) e SUB (sottrarre). Un computer programmatico nel linguaggio Assem-

bler tende a lavorare più velocemente di un altro programmato in un linguaggio di alto livello come il BASIC.

B

BASIC —È un acronimo per Beginners All-Purpose Symbolic Instruction Code. È il linguaggio per computer più largamente usato nel settore dei micro-computer. Sebbene sia stato molto criticato, ha il vantaggio di essere molto facile da imparare. Molte espressioni BASIC sono simili all'inglese comune.

BAUD — Viene da Baudot, un pioniere della comunicazione telegrafica. Baud misura il livello di trasmissione delle informazioni ed è approssimativamente uguale a un bit per secondo.

BCD — Un'abbreviazione per Binary Coded Decimal (Binario Codificato Decimale).

Benchmark — Un test con cui possono essere misurate certe funzioni del computer. Ci sono molti "test standard Benchmark", che generalmente sperimentano soltanto la velocità. Questo è un aspetto raramente importante in un microcomputer, il tipo di computer piú interessante per un potenziale compratore.

Binario — Un sistema numerico che usa soltanto i numeri zero e uno.

Bit — Un'abbreviazione per Binary Digit (cifra binaria). È la piú piccola unità di informazione che un circuito di computer può riconoscere.

Boole, Algebra di— Il sistema algebrico sviluppato dal matematico George Boole che usa numerazioni algebriche per esprimere relazioni logiche (vedi AND).

Bootstrap — Un breve programma o sottoprogramma che viene letto all'interno del computer quando è acceso per la prima volta. Orienta il computer ad accettare i programmi successivi, di lunghezza maggiore.

Bug — Un errore che impedisce il funzionamento del programma. Sebbene sia generalmente usato per indicare solo un difetto o un errore nel programma, il termine bug può anche essere usato per un difetto nell'hardware del computer.

Bus — Un numero di conduttori usati per trasmettere segnali, dati e istruzioni.
Byts — Un gruppo di cifre binarie che compone una parola di computer. Generalmente i bits presenti in un byte sono otto.

C

CAI — Computer Assisted Instruction
CAL — Computer Assisted Learning. Il termine è generalmente usato per descrivere quei programmi che coinvolgono lo studente in processi di apprendimento.
CHIP — Il termine che si usa per indicare l'intero circuito, che è inciso su un piccolo pezzo di silicio. Il chip è, naturalmente, il cuore del computer.
CLOCK — Il congegno di sincronizzazione all'interno del computer che sincronizza le sue operazioni.
COBOL — Un linguaggio ad alto livello che deriva dalle parole Common Business Orientated Language. Il COBOL è usato principalmente per la schedatura e il mantenimento di ciò che è stato registrato.
Comparator — Un congegno che mette a confronto due cose e produce un segnale collegato con la differenza tra i due.
Compiler — Un programma che trasforma i linguaggi di programmazione ad alto livello in codici binari da macchina. In questo modo i programmi scritti in linguaggi ad alto livello possono essere utilizzati dal computer.
Complement — Un numero che è derivato da un altro secondo regole stabilite.
Computer — Un congegno con tre principali capacità o funzioni:
1) accettare i dati
2) risolvere i problemi
3) fornire risultati.
CPU — Sta per Central Processing Unit. È il cuore dell'intelligenza del computer dove si gestiscono i dati e si sviluppano le istruzioni.
Cursore — Un carattere che appare sul video quando il computer sta compiendo le sue operazioni. Esso rivela do-

ve il prossimo carattere sarà stampato. Su un computer ci sono generalmente "tasti di controllo cursore" per permettere all'utente di spostare il cursore sul video.

D

Dati — Informazioni presentate in una forma che il computer può elaborare.
Debug — Il termine che si usa quando si esamina un programma e si correggono eventuali errori, cioè, trovare e rimuovere i bugs, i difetti.
Digital Computer — Un computer che opera su quelle informazioni che si presentano come digitali.
Disk/Disc — È un disco di plastica sensibilizzato magneticamente, un poco più piccolo di un "45 giri". È usato per l'immagazzinamento dei programmi e per ottenere dati. I dischi sono notevolmente più veloci da caricare rispetto ad una cassetta della stessa lunghezza di programma. Si può accedere molto velocemente su un disco mentre un programma sta operando per ottenere ulteriori dati.
Display — L'output visivo del computer, generalmente su un video, o su uno schermo di monitor.
Dot Matrix Printer — Una stampante che stampa i listati di un programma o di ciò che appare sul video. Ogni lettera e ogni carattere sono composti da un certo numero di "dots", cioè di punti. Più alto è il numero di punti per ogni carattere e migliore sarà la qualità operativa della stampante.
Dynamic Memory — Un'unità di memoria all'interno del computer che "dimentica" ciò che contiene quando viene tolta l'alimentazione elettrica.

E

Editor — Questo termine è generalmente usato per designare quella sezione del computer che permette al programmatore di cambiare le istruzioni di un programma mentre lo sta scrivendo.

EPROM — Sta per Erasable Programmable Read-Only Memory. È come il ROM nel computer, con la sola differenza che è abbastanza facile inserire materiale all'interno di una EPROM che non sparisce quando si toglie l'alimentazione. Le EPROM devono essere esposte a forti raggi ultravioletti se si vuole cancellarle.

Error Messages (Messaggi di errore) — L'informazione data da un computer che indica dove è stato commesso un errore nella codificazione durante una parte del programma. L'informazione è trasmessa dal computer che si ferma e stampa una parola, o una parola e dei numeri, o soltanto una combinazione di numeri, in fondo al video. Questo rivela quale errore è stato fatto. Gli errori più comuni includono l'uso della lettera O invece dello zero in una linea, o l'omissione in una espressione di entrambe o di una delle parentesi, o l'errore nella definizione di una variabile.

F

File — Una serie di item d'informazione collegati fra loro e organizzati in modo sistematico.

Floppy Disk — Un disco magnetico relativamente poco costoso, usato per immagazzinare le informazioni del computer, e così chiamato perché molto flessibile (vedi Disk/Disc).

Flow Chart — Un diagramma tracciato prima di scrivere un programma nel quale le principali operazioni sono racchiuse entro rettangoli o altre forme e connesse a frecce attraverso linee per rappresentare "loop" di istruzioni, e le decisioni scritte fra parentesi. Ciò ti aiuta a scrivere un programma in modo molto più semplice perché trappole come loop infiniti o variabili non definite possono essere scoperte in ogni fase. Può risultare utile scrivere questo diagramma per programmi molto brevi, ma è senz'altro conveniente se si vuole creare un programma più lungo.

Firmware — Ci sono tre tipi di "ware" nei computer: cioè programmi software "temporanei"; hardware, come quelli contenuti nelle ROM permanenti; e firmware, nel quale l'informazione è relativamente permanente, come in una EPROM (vedi EPROM).

Flip-Flop — Un circuito che mantiene in memoria una condizione elettrica finché questa non è cambiata nella condizione opposta da un segnale.
FORTRAN — Sta per FORmula TRANslation (traduzione di formula). È un linguaggio di computer ad alto livello, orientato verso problemi matematico-scientifici.

G

Gate — Un circuito elettrico che, sebbene possa captare uno o più segnali in arrivo, manda in uscita soltanto un singolo segnale.
Graphics — Informazione grafica, in opposizione alle informazioni fornite da lettere e numeri.

H

Hard Copy — Uscita di computer il cui supporto è permanente.
Hardware — Le parti fisiche di un computer (vedi anche software e firmware).
Hexadecimal (Hex) — Un sistema numerico con base sedici, Sono usate le cifre da zero a nove e le lettere A, B, C, D, E, F per la rappresentazione dei numeri. A è uguale a 10, B è uguale a 11, C è uguale a 12, e così via. Hex è spesso usato dagli utenti di microcomputer.
Hex Pad — Una tastiera specificamente progettata per inserire numerazioni esadecimali.
High Level Language (Linguaggio ad alto livello) — Un linguaggio di programmazione che permette all'utente di parlare con il computer più o meno in lingua inglese. In generale più è alto il livello del linguaggio (cioè, più è simile all'inglese) più lungo sarà il tempo impiegato dal computer per tradurlo in un linguaggio che esso può utilizzare. Linguaggi a più basso livello sono molto più difficili per l'operatore umano ma generalmente offrono una esecuzione più veloce.

I

Input — L'informazione inserita nel computer attraverso una tastiera, un microfono, una cassetta o un disco.
Input/Output (I/O Device) — Un congegno che accetta le informazioni o le istruzioni dal mondo esterno, le trasmette al computer e, dopo l'elaborazione, le rinvia, o sotto una forma adattabile alla memorizzazione, o una forma comprensibile all'essere umano.
Instruction — Il dato che comanda una sola azione nell'elaborazione delle informazioni operate dal computer (noto anche come comando).
Integrated Circuit (Circuito Integrato) — Un completo circuito elettronico residente sulla superficie di un semiconduttore.
Interface — Il confine fra il computer e un periferico come la stampante.
Interpreter — Un programma che traduce, istruzione per istruzione, il linguaggio ad alto livello inserito da un operatore umano in un linguaggio che la macchina può capire.
Inverter (Invertitore) — Un "gate" logico che cambia nell'opposto il segnale inserito.
Interactive Routine (Sottoprogramma Interattivo)— Parte di un programma che è ripetuto più volte finché non si raggiunge una data condizione.

J

Jump Instruction — Un'istruzione che dice al computer di muoversi verso un'altra parte del programma, quando la destinazione di questo spostamento dipende dal risultato di un calcolo appena compiuto.

K

K — Questa lettera riporta la misura della memoria. La memoria è generalmente misurata in blocchi di "K". Un K contiene 1.024 bytes.

Keyword (parola-chiave) — La parola di inizio in una linea di programmazione, generalmente la prima parola dopo il numero della istruzione. Parole-chiavi sono STOP, PRINT e GOTO.

L

Language — I linguaggi di computer sono divisi in tre gruppi: i linguaggi ad alto livello, come il BASIC, che sono relativamente vicini all'inglese ed abbastanza facili da usare per l'uomo; i linguaggi a basso livello, come l'ASSEMBLER, in cui compaiono brevi frasi che hanno qualche collegamento con l'inglese (ADD per add e RET per return, ad esempio); e il codice macchina, che comunica più o meno direttamente con la macchina.
LCD — Sta per Liquid Crystal Diode. Alcuni computer come il TRS-80 Pocket Computer usano un "display" LCD.
LED — Sta per Light Emitting Diode. I numeri luminosi rossi che sono spesso usati negli orologi da polso o da muro sono composti da LED.
Logic — La formula matematica di uno studio di relazioni fra eventi.
Loop — Una serie di istruzioni all'interno di un programma che sono ripetute finché una particolare condizione viene soddisfatta.

M

Machine Language o Machine Code (Codice macchina) — Un codice che può essere capito e messo in pratica direttamente dal computer.
Magnetic Disk— vedi Disk e Floppy Disk.
Mainframe — I computer sono generalmente divisi in tre gruppi e il fatto di appartenere ad un certo gruppo dipende dalla grandezza del computer. Il computer più venduto è il microcomputer. I computer di media grandezza sono i minicomputer, e i computer giganti che qualche volta si vedono nei film di fantascienza sono computer "mainframe". Fino a 15 anni fa tali computer erano, in termini pratici, i so-

li disponibili.

Memory — Dentro un computer ci sono due tipi di memoria. La prima chiamata ROM, è la memoria che arriva già programmata sul computer e che dice al computer come prendere decisioni e come compiere operazioni aritmetiche. Questa memoria non è cancellata se si spegne il computer. Il secondo tipo è la RAM. Questa memoria mantiene il programma che è stato scritto sulla tastiera o che è trasmesso all'interno tramite una cassetta o un disco. La maggior parte dei computer "dimenticano" ciò che è in RAM quando vengono spenti.

Microprocessor — Il cuore di qualsiasi computer. Richiede le interfacce con le unità periferiche, l'alimentazione di energia e i congegni di input e output. In tal modo può operare come un microcomputer.

MODEM — Sta per Modulatore/Demodulatore. È un apparato che permette a due computer di parlare fra loro per telefono. Generalmente i computer usano un supporto nel quale è posto un ricevitore telefonico.

Monitor — Nel linguaggio dei computer ha due significati. Un significato è uno schermo simile a quello televisivo. Un monitor ha serie difficoltà ad adattarsi ai programmi televisivi e generalmente l'immagine prodotta su un monitor è migliore di quella prodotta da una comune Tv. Il secondo significato di un monitor si rapporta alla ROM. Il monitor di un computer è descritto come il complesso di informazioni incorporate nel computer all'atto dell'acquisto. Queste informazioni permettono di prendere decisioni e di compiere calcoli aritmetici.

Motherboard — Una struttura alla quale possono essere aggiunti circuiti extra. Questi circuiti spesso offrono facilitazioni che non sono incorporate nel computer, come quella di produrre suoni o di controllare una penna ottica.

MPU — Abbreviazione per Microprocessor Unit.

N

Nano-secondo — Un nano-secondo è un millimiliardesimo di secondo, l'unità di tempo con la quale si misura la velocità di un computer o di un microcircuito di memoria.

Non-Volatile Memory — La memoria che non si perde quando il computer è spento. Alcuni computer più piccoli

come il TRS-80 Pocket Computer hanno "non volatile memory". Le batterie mantengono il programma inserito per settecento ore.

Not - Un'operazione booleana che trasforma una cifra binaria nel suo opposto.

Null String — Una "stringa" che non contiene caratteri. Nel programma compare sotto forma di due doppie virgolette, senza niente fra di loro.

Numerico — Concerne i numeri quando sono opposti alle lettere (cioè alfabetico). Molte tastiere sono alfanumeriche, sono cioè provviste sia di numeri sia di lettere.

O

Octal — Un sistema numerico che usa otto come base e quindi le cifre da 0 a 7. Tale sistema non è oggi molto usato nel settore dei microcomputer. Il sistema esadecimale è più comune (vedi Hexadecimale)

Operating System — (Sistema operativo) — Il software e il firmware, generalmente previsti su una macchina che permette di far girare altri programmi.

OR — Un'operazione booleana che ritorna a 1 se uno o più input sono 1.

Oracle (Oracolo) — Un metodo di messaggi invianti test tramite un segnale di trasmissione televisiva. Un set di teletest è richiesto per decodificare i messaggi.

Output — Informazioni o dati trasmessi dal computer a congegni quali uno schermo come quello televisivo, una stampante o una cassetta. L'output generalmente consiste in un'informazione che il computer ha prodotto come risultato della elaborazione di un programma.

Overflow — Un numero troppo grande o troppo piccolo per essere elaborato dal computer.

P

Pad — Vedi Keypad

Pagina — Spesso usata per indicare la quantità di informazioni necessaria per riempire uno schermo televisivo. Così, vedendo una pagina del programma, è possibile analizzare la quantità di informazioni che appaiono sul video tutte in una volta.
PASCAL — Un linguaggio ad alto livello.
Periferico — Qualsiasi cosa che è collegata e controllata dal computer, come un'unità a disco, una stampante o un sintetizzatore vocale.
Port — Un connettore attraverso il quale le informazioni sono trasmesse o inserite nel computer.
Prestel — Il nome inglese per un sistema basato sulla trasmissione di informazioni via telefono da un computer centrale e sulla loro visualizzazione su uno schermo televisivo. Negli Stati Uniti una versione commerciale simile è nota come "The Source".
Program — Nel linguaggio computer può essere una lista di istruzioni che si inseriscono nel computer, oppure può essere un verbo, cioè "programmare un computer".
PROM — Sta per Programmable Read Only Memory. È un sistema che può essere programmato e, una volta che lo è stato, il programma è permanente (vedi anche EPROM e ROM).

R

Random Access Memory (RAM) — La zona di memoria entro il computer che può essere cambiata a comando dalla persona che usa il computer. Il contenuto della RAM è di solito perduto quando un computer è spento. La RAM è l'integrato che memorizza ciò che viene scritto e anche i risultati di calcoli in atto.
Read-Only Memory (ROM) — In contrasto alla RAM, l'informazione qui non può essere cambiata dall'utente e non va perduta quando si spegne il computer. I dati della ROM sono collocati dal produttore e dicono al computer il modo con cui deve prendere decisioni e come compiere calcoli aritmetici. La misura di capacità della ROM e RAM è data in unità K (vedi K).
Recursion — La ripetizione continua di una parte del programma.

Registro — Una specifica sezione della memoria nella quale uno o più parole di computer sono memorizzate nel corso delle operazioni.
Parola Riservata — Una parola che non può essere usata per una variabile in un programma perché il computer la leggerà in modo errato. Un esempio è la parola TO. Poiché TO ha uno specifico significato nel linguaggio dei computer, la maggior parte dei calcolatori respinge questa parola come nome per una variabile. Lo stesso vale per parole come FOR, GOTO e STOP.
Routine — Questa parola può essere usata come sinonimo di programma o può riferirsi a una specifica sezione all'interno del programma (vedi anche Subroutine).

S

Seconda Generazione — Ha due significati. Il primo si applica nei confronti dei computer che usano transistor, in opposizione alla prima generazione di computer che usavano valvole. La seconda generazione può anche indicare la seconda copia di un particolare programma. Susseguenti generazioni sono danneggiate da un disturbo crescente.
Semiconduttore — Un materiale che è generalmente un isolante elettrico, ma sotto specifiche condizioni può diventare un conduttore.
Serial — Informazione che è memorizzata o inviata in una sequenza, un bit alla volta.
Segnale — Un impulso elettrico che trasmette dati.
Silicon Valley — Il nome popolare dato a una zona in California dove si trovano molti produttori di semiconduttori.
SNOBOL — Un linguaggio ad alto livello.
Software — Il programma inserito nel computer dall'utente. Questo programma dice al computer cosa fare.
Software Compatible — Si riferisce a due differenti computer che possono accettare i programmi scritti per l'altro.
Static Memory — Un congegno di memoria non volatile che trattiene le informazioni per tutto il tempo che il computer è acceso. Tuttavia, non richiede ulteriori consumi di energia per mantenere in ordine la memoria.
Subroutine (Sottoprogramma) — Parte di un programma che è spesso inserita molte volte durante l'esecuzione del programma principale. Una subroutine finisce con un'istru-

zione che comanda di ritornare indietro alla istruzione successiva a quella che aveva inviato lo svolgimento del programma alla subroutine.

T

Teletext — Informazione trasmessa nella sezione superiore di un'immagine che appartiene a una trasmissione televisiva. Richiede una struttura speciale per essere decodificata e per riempire il video di informazioni riguardo il test. I messaggi teletext possono anche essere trasmessi tramite un cavo, per esempio, il servizio Prestel in Gran Bretagna o The Sources negli Stati Uniti.
Teletype — Un apparecchio simile ad una macchina da scrivere che può mandare, ricevere e stampare informazioni.
Terminale — Un'unità indipendente dell'unità centrale di elaborazione. Generalmente consiste di una tastiera e di un visore.
Time Sharing — Un processo attraverso il quale molti utenti possono aver accesso a un grande computer che si sposta rapidamente da un utente all'altro in sequenza, cosicché ogni utente ha l'impressione di essere il solo utente del computer.
Truth Table (Tavola della verità) — Una tavola matematica che elenca tutti i possibili risultati di un'operazione booleana. I risultati dipendono dalle varie combinazioni di input.

U

UHF — Ultra High Frequency (300-3000 MegaHertz).
Ultra Violet Erasing — La luce ultravioletta deve essere usata per cancellare le EPROM (vedi EPROM).

V

Variabile — Una lettera o combinazione di lettere e simbo-

li a cui il computer può assegnare un valore o una parola durante il funzionamento di un programma.
VDU — Abbreviazione per Visual Display Unit.
Volatile — Indica la memoria che "dimentica" le informazioni in essa contenute quando il computer è spento.

W

Word (Parola) — Un gruppo di caratteri o una serie di cifre binarie che rappresentano un'unità d'informazione e occupano una singola posizione di memoria. Il computer elabora una parola come singola istruzione.
Word-Processor (Elaboratore di testi) — Una macchina da scrivere altamente intelligente che permette a chi scrive di manipolare il testo, di spostarlo, per giustificare margini e per spostare interi paragrafi, se necessario, su un video prima di mandare l'informazione sulla stampante. Questi elaboratori hanno generalmente memorie, cosicché modelli di lettere e testi di lettere, scritti precedentemente, possono essere nuovamente stampati.

Traduzioni

3-D MAZE

30 — Colore del primo piano e dello sfondo
60 — Sud
70 — Ovest
80 — Nord
90 — Est
680 — Invia il "set up" qui
890 — Invia qui la mappa
930 — Solido anteriore, colore = N
1000 — Porta anteriore, colore = N
1070 — Parete laterale, colore = N
1140 — Porta laterale, colore = N
1210 — Parete laterale, colore = N
1280 — Porta laterale, colore = N
1350 — Parete anteriore, colore = N
1420 — Porta anteriore, colore = N
1490 — Centro - sempre
1680 — Parete centrale
1660 — Grande muro divisorio

SEARCH FOR THE HOLY GRAIL

1490 — Livello V
1730 — Punteggio
1750 — Buon punteggio
1760 — Punteggio medio; da; giochi; totale/giochi
1780 — Giochi ancora?

RAM BLASTER

60 — Tutti i sistemi vanno: tieniti pronto
290 — Livello = 1
300 — Punteggio = 0
590 — Premi un tasto qualsiasi il livello iniziale
640 — Preparati
670 — Pronti!
1870 — Punteggio =

2570 — Punteggio =
2590 — Il gioco è finito
2770 — Il tuo nome, prego?
2800 — Lungo dieci caratteri - non di più
2940 — Il tuo ramchip
3130 — Eproms nascosti 220
3160 — Elettricità statica - cautela
3190 — 3000 punti per ogni Ramchip libero
3230 — Preparati

HAPPY BIRTHDAY

50 — Basato su « Giorno della settimana »
60 — di Mark Charlton, in
70 — 'The Gateway Guide...'
80 — Dom, lun, mar, mer, gio, ven, sab
120 — Per cortesia inserisci il giorno (sotto forma
130 — di numero, p. es. 30) in
140 — cui sei nato
170 — Quale mese
200 — Quale anno
330 — La data
340 — Giorno, mese, anno
360 — È un

DOWNUNDER

70 — Muovi il tuo canguro con i tasti contrassegnati dalle frecce
Cerca di raggiungere la porta
80 — in fondo allo schermo. Non finire nel fiume (f), il dingo (p), l'ornitorinco (h), o il vombato (e)
90 — Ottieni dei punti se colpisci i pezzi magici (ijklm), ma cerca di non esaurire il tempo mangiando
100 — vegemite. Buona fortuna
1060 — Niente vegemite
1070 — Niente vegemite!
1120 — Un altro canguro
1680 — Il tuo punteggio

1690 — Vuoi continuare a giocare (con), o iniziare una nuova partita (new)
1700 — Con o new?

TEXAS TENBY

30 — Hai tirato
120 — Nuova partita
130 — Vincite (W), perdite (L)
140 — Numero della partita (G)
190 — Il tuo punteggio è
350 — Hai vinto!
380 — Hai perso!
490 — Fine del gioco
500 — Questa è la fine del gioco
510 — Vediamo come sei andato...
570 — Le vincite hanno eguagliato le perdite
590 — Hai vinto da a
610 — Hai perso da a

SUPER TYPER

410 — Hai ottenuto un punteggio di
460 — Hai ottenuto un punteggio massimo
470 — Il migliore tra i precedenti era da
490 — Qual'è il tuo nome?
500 — Di soli 24 caratteri, non... caratteri, prego
610 — Premi un tasto qualsiasi per iniziare

TI FASTERMIND

380 — Nero
410 — Bianco
530 — Hai indovinato!
540 — Il codice era

SIMON

360 — Sei il campione!
390 — Il numero era
470 — Hai totalizzato punti

DIAMOND FEVER

30 — Vuoi istruzioni?
140 — Sei solo su un asteroide nel più profondo spazio. Mentre sondi la roccia
150 — ti accorgi che contiene un'eccezionale quantità di metalli, così ti avventuri sotto
160 — per vedere se puoi estrarne qualcuno, quando improvvisamente comprendi di non
170 — trovarti all'interno di un asteroide bensì di una perfida astronave Vogon!!
180 — Come puoi fuggire da questo labirinto di rocce e di macerie? Il fato ti ha assistito, hai dato un'occhiata al filone di diamanti
190 — nella roccia, ma come puoi raggiungere l'uscita prima che il
200 — Vogon ti prenda?
210 — Premi un tasto qualsiasi per continuare
250 — Il livello 1 è il più facile, il livello 5 è per i campioni. Hai la tua fidata trivella con
260 — te, che sta esaurendo il vapore (naturalmente di uranio) ma taglierà la roccia: N.B. Stai finendo ...gasp l'aria ...gasp
280 — Usa i tasti contrassegnati dalle frecce per muoverti. Quando cercherai di tagliare la roccia la tua trivella farà beep
430 — Livello di difficoltà?
1670 — Hai totalizzato quella volta
1820 — Il buon punteggio è , e tu hai ottenuto...
1860 — Il buon punteggio è , e tu l'hai ottenuto
1870 — Qual è il tuo nome?
1890 — Non così lungo

FINAL FRONTIER

690 — Fine del giro della morte
1050 — Hai perso
1060 — Gli alieni hanno vinto
1080 — Il tuo punteggio è
1170 — Punteggio =; Livello

FENCED IN

1010 — Troppi urti! Sei morto
1050 — Congratulazioni!
1060 — Sei fuggito!

MENTO

30 — Moltiplica la tua età per
40 — due, quindi aggiungi cinque
70 — Ora moltiplica questo numero
80 — per cinquanta, e
90 — sottrai 365
120 — Ora aggiungi la somma
130 — di denaro che hai
140 — in tasca
170 — Ora dimmi
180 — il numero finale che
190 — ti risulta
260 — I soldi che hai in tasca ammontano a
280 — Hai ... anni

RACE OF THE LIZARD MEN

420 — Il vincitore è il numero uno
440 — Il vincitore è il numero due
490 — Definisci i caratteri

ENCHANTED FOREST

200 — Qui ci sono le fate
300 — Ora ti trovi nel settore:
470 — Puoi spostarti a
700 — La tua mossa?
800 — Ti rimangono ... frecce
850 — e i folletti ti hanno
860 — ucciso
880 — Hai trovato il drago!
940 — Fate nelle vicinanze
960 — Folletti nelle vicinanze
980 — Drago nelle vicinanze

FROG PLAGUE

50 — Usa S e D per spostarti
60 — a sinistra e a destra. Gli oggetti
70 — che devi evitare sono:
80 — foglie, motoscafo
90 — rocce, pezzi di legno e ponti.
110 — Le pareti compaiono con
120 — il progredire del gioco
560 — Sei morto
570 — Il tuo punteggio è

KAMIKAZE PILOT

60 — Inserisci un numero da 1 a 9
70 — 1 è il più facile, 9 il più difficile
620 — Fine!!
670 — Hai totalizzato

DARING DAMON

140 — Benvenuto alla gara ippica texana, il tuo credito con Daring Damon è fissato a 2000 dollari; non c'è un tetto massimo per le scommesse

170 — Quanti giocatori
200 — Sono Daring Damon, sono chiamato così per le mie quotazioni incredibilmente alte, stai attento ho delle informazioni dall'interno
210 — Quindi tieni conto delle mie quotazioni. Ma è risaputo che sbaglio
370 — N. del nome ... quotazione
510 — Giocatore..., quale cavallo?
540 — Ci sono solo 5 cavalli in corsa. Sciocco...
590 — Giocatore..., quanto scommetti?
630 — Cerca di aver fortuna, hai solo $...
1230 — Premi un tasto qualsiasi per continuare
1460 — Giocatore..., uova fradice. Hai fatto bancarotta
1520 — Il giocatore ... riscuote
1590 — Denaro rimasto

CITY BOMBER

780 — Ti sei schiantato al livello...

ARECIBO ATTACK

700 — Colpisci il marziano!
770 — Hai subito un'invasione!
780 — La prossima volta avrai più fortuna!
830 — Hai trionfato!

CHALLENGER

200 — Digita X per spostare la tua astronave verso il basso
210 — Penso che questo programma di tipo spaziale sarà di tuo gradimento

250 — Per i primi due 100 punti avrai diversi aggressori...
260 — Inserisci il tuo nome, prego
690 — Il gioco è finito. Il punteggio è...
700 — Vuoi giocare ancora Y o N Y
710 — Per giocare ancora digita Enter o N e poi Enter se non vuoi più giocare

-*Thanks to:* ***Gianfranco Mazzarello (www.microatena.it)***

for the Scan of this book.

Computer Games

GIOCHIAMO CON TI99/4A

Tanti fantastici programmi, con la traduzione in italiano, appositamente ideati per questa collana e in grado di garantirvi ore e ore di svago istruttivo e divertente

Tra i giochi spettacolari di questo libro: KAMIKAZE (pilotate il vostro aereo attraverso un tunnel tortuoso e frastagliato); LA SFIDA (misurate i vostri riflessi: evitate gli attacchi degli alieni dallo spazio); FEBBRE DI DIAMANTE (combattete contro il terribile Vogon scavando un passaggio attraverso un asteroide); LABIRINTO A 3-D (fantastici castelli da esplorare); RAM-DISTRUTTORE (sarete capaci di distruggere nello schermo tutte le componenti del computer?).

"GIOCHIAMO CON TI 99/4A" vi aiuterà moltissimo, giocando, a migliorare la vostra abilità fornendovi non solo tutte le istruzioni per inserire correttamente i programmi nel computer, ma anche un utilissimo glossario dei termini essenziali e preziosi consigli e indicazioni su come modificare e migliorare i programmi del libro o realizzarne di nuovi.

Tanti fantastici giochi per tutta la famiglia!

CL 006-0149-7 ISBN 88-7605-149-X

L. 9.500 (i.i.)

Altri giochi per il vostro TI 99/4A