Computer Games Series

GIOCHIAMO CON ATARI

GREMESE EDITORE

Computer Games Series

GIOCHIAMO CON ATARI

GIOCHIAMO CON ATARI

Paul Bunn

GREMESE EDITORE

Computer Games Series
Periodico mensile
N. 6 - Agosto 1984
Registrazione Trib. di Roma N. 138/84
del 24 marzo 1984
Direttore responsabile: Grazia Valci

Titolo originale
Games for your Atari

Traduzione dall'inglese
Flavia Pietromarchi

Edizione italiana a cura di
Giancarlo Zagarese

Design
Ray Hyden

Illustrazioni
Sue Walliker

Fotocomposizione
Typo-centro s.n.c. - Roma

Stampa
Litopat - Verona

ISBN 88-7605-121-X

Alla mia famiglia
e agli amici

PAUL BUNN — AUTORE

Paul Bunn è uno studente di sedici anni e ha un Atari da un anno e mezzo. Paul ha anche scritto un manuale di programmazione per l'Atari, intitolato *Total Control,* pubblicato da Interface.

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 l'autore di diversi libri, tra i quali: *Getting Acquainted With Your ZX81, 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 Abbicci".

SUE WALLIKER—ILLUSTRATRICE

Sue Walliker è un'illustratrice free-lance.

RINGRAZIAMENTI

L'autore ringrazia il personale del Micro-C di Fishponds, Bristol, che cortesemente gli ha messo a disposizione una stampante mentre la sua era in riparazione.

INDICE

Introduzione

Il vostro computer è in attesa per sfidarvi. Rapidi «games» di grafica, giochi di concentrazione, di parole e di enigmistica sono tutti qui pronti a farvi divertire.
Nel libro vi sono una notevole varietà di giochi i cui programmi sono stati scritti da alcuni fra i più giovani e abili programmatori che lavorano attualmente in Gran Bretagna. L'esaminare i programmi dei giochi vi dà la possibilità di apprendere raffinate tecniche e sottili metodi di programmazione che voi stessi potete poi applicare. Inoltre, una volta che avete conosciuto a fondo i programmi presentati dal libro, potreste senz'altro provare a migliorarli — un programma non è mai "perfetto" — arricchendo le vostre capacità di programmazione. Adesso voltate pagina e incominciate a "battere" i programmi. Sono certo che il vostro divertimento sarà pari a quello che abbiamo provato durante la stesura di questo volume.

Introduzione dell'autore

In questo libro ho cercato di illustrare il maggior numero possibile di programmi che sappiano sfruttare appieno le capacità degli home computer Atari 400/800. Molti programmi utilizzano insiemi di caratteri ridefinibili, e alcuni usano l'ottima capacità grafica dei computer Atari chiamata "player-missile" (giocatore-missile). Spero che vi divertirete con questi giochi anche se penso che vi sarà molto più utile esaminare i listati per comprenderne il funzionamento. Vi sarà utile, inoltre usare le idee suggerite in questo libro per applicarle ai vostri programmi.

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.

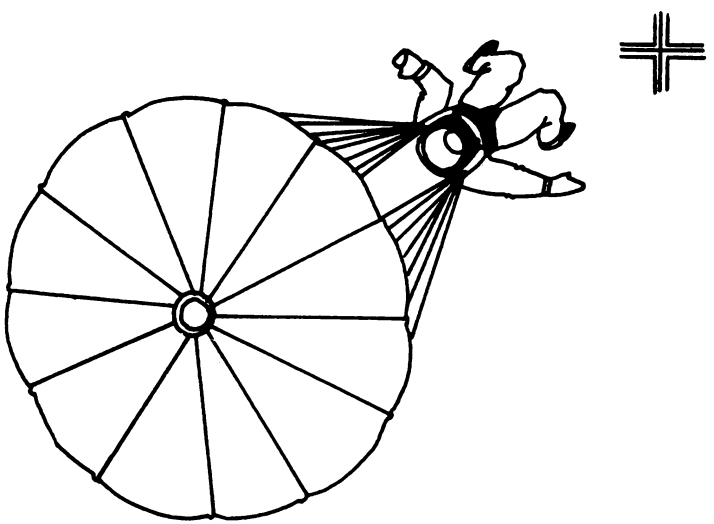
g.z.

SKYDIVER

(Per questo gioco si usa il joystick)

Questo programma utilizza dei caratteri definiti dall'utente e una grafia "player-missile". È opportuno che chi soffre di vertigini non giochi, poiché si deve raggiungere il maggior punteggio possibile con 10 lanci nel vuoto.

Il vostro aeroplano vola attraverso la parte superiore dello schermo: per lanciarvi dovrete premere il pulsante rosso. Vi troverete a scendere a velocità sostenuta. Se desiderate rallentare, "abbassate" il joystic per aprire il vostro paracadute. Guadagnerete dei punti premio scendendo con bravura sulla piattaforma di atterraggio. Questi punti dipendono dall'istante di apertura del paracadute: il vostro punteggio sarà tanto più alto quanto più tardi lo aprirete. Non potete, però, aprire il paracadute al di sotto di una certa quota. Un punteggio attorno a 12.000 è discreto. Buon atterraggio!



```
10 REM %%  SKY DIVER - WRITTEN BY  %%
20 REM %% PAUL BUNN  DECEMBER 1982 %%
25 IF PEEK(203)=PEEK(106)-8 THEN 40
30 GOSUB 10000:REM INITIALIZATION
40 DIM PLANE$(1),MAN$(1),CHUTE$(1),BASE$
(1),LINE$(1)
50 PLANE$=CHR$(ASC("%")+128):MAN$="!":LI
NE$=CHR$(ASC("#")+96)
55 FOR JUMP=1 TO 10
60 GRAPHICS 17:SETCOLOR 2,12,10:POKE 16,
64:POKE 53774,64
61 POKE 53277,3:REM % ENABLE PLAYER MISS
ILE GRAPHICS %
62 POKE 559,46:REM % DOUBLE LINE RESOLUT
ION FOR PLAYERS %
63 POKE 54279,PEEK(204):REM % TELL ANTIC
 WHERE PM GRAPHICS ARE IN RAM %
64 POKE 53256,3:POKE 53257,3:REM % QUADR
OUPLE SIZE FOR CLOUDS %
65 POKE 704,8:POKE 705,14:REM % COLOUR F
OR CLOUDS %
66 X3=INT(RND(0)%100)+75:POKE 53248,X3:R
EM %% RANDOM HORIZONTAL POSITION FOR CLO
UD 1 %%
67 X4=INT(RND(0)%100)+75:POKE 53249,X4:R
EM %% RANDOM HORIZONTAL POSITION FOR CLO
UD 2 %%
68 POKE 623,1:REM % SET PRIORITY BETWEEN
 PLAYERS AND PLAYFIELDS %
69 POKE 53278,255:REM % CLEAR COLLISION
DETECTION REGISTERS %
70 POKE 756,PEEK(203):SETCOLOR 3,3,4:REM
 % POINT TO NEW CHARCTER SET IN RAM INST
EAD OF ROM %
80 X=1:BASE$="#":CHUTE$=CHR$(ASC("%")+12
8)
85 REM %% PRINT BASE LINE %%
90 FOR P=0 TO 19
100 POSITION P,23
110 PRINT #6;LINE$;
120 NEXT P
130 BASE=INT(RND(0)%16)+2
140 POSITION BASE,22
```

```
150 ? #6;BASE$
155 REM ** MAIN PROGRAM LOOP **
160 POSITION X,0:? #6;PLANE$:POSITION X-
1,0:? #6;" "
170 X=X+1:IF X<19 THEN 190
180 POSITION X-1,0:? #6;" ":X=1
185 REM ** JOYSTICK ROUTINE **
190 S1=STICK(0):S2=STRIG(0)
191 X3=X3-3:IF X3<20 THEN X3=220
192 X4=X4+3:IF X4>220 THEN X4=20
193 POKE 53248,X3:POKE 53249,X4
200 IF MAN=0 AND S2=0 THEN GOSUB 1000
205 IF MAN=1 THEN LOCATE X2,Y2,Z
210 IF MAN=1 THEN POSITION X2,Y2:? #6;MA
N$
220 IF CHUTE=1 THEN POSITION X2,Y2-1:? #
6;CHUTE$:FOR P=1 TO 100:NEXT P
222 IF PEEK(53252)<>0 OR PEEK(53253)<>0
THEN 2080
225 IF Y2=22 AND MAN=1 THEN 2000
230 OX=X2:X2=X2+(S1=7)-(X2>18)
240 OY=Y2:X2=X2-(S1=11)+(X2<1):Y2=Y2+1
250 IF S1=13 AND MAN=1 AND CHUTE=0 AND Y
2<19 THEN CHUTE=1:FOR T=0 TO 50:SOUND 0,
T,120,15:NEXT T:SOUND 0,0,0,0:C=Y2
255 IF MAN=0 THEN 160
260 POSITION OX,OY-1:? #6;" "
270 POSITION OX,Y2-1:? #6;" "
280 GOTO 160
999 REM ** MAN OPENS CHUTE **
1000 MAN=1:Y2=2:X2=X
1010 FOR P=30 TO 0 STEP -2:SOUND 0,P,120
,15:NEXT P:SOUND 0,0,0,0:RETURN
2000 IF Z=ASC(BASE$) THEN GOTO 3000
2010 FOR G=0 TO 255
2020 SOUND 0,G,10,15:POKE 708,G
2030 NEXT G
2040 FOR G=255 TO 0 STEP -3
2050 SOUND 0,G,10,15:POKE 708,G
2060 NEXT G
2070 SOUND 0,0,0,0
```

```
2080 POSITION 0,0:POKE 708,104
2090 ? #6;">>>>>SPLAT<<<<<"
2095 ? #6;"SCORE:";SCORE
2100 ? #6:? #6;"PRESS start TO PLAY"
2110 POKE 53279,0
2120 IF PEEK(53279)<>6 THEN 2110
2130 RUN
3000 POSITION 0,0
3010 IF CHUTE=1 THEN 3500
3020 ? #6;"YOU DID NOT HAVE"
3030 ? #6;"your chute open"
3040 ? #6:? #6;"   OHCH ..."
3050 ? #6
3060 FOR P=0 TO 150 STEP 2
3070 SOUND 0,P,12,15
3080 NEXT P
3090 SOUND 0,0,0,0
3100 GOTO 2095
3500 ? #6;"well done ..."
3510 ? #6;"YOU MANAGED TO LAND"
3520 ? #6;"ON THE PAD WITH YOUR"
3530 ? #6;"CHUTE OPEN ..."
3540 POSITION 3,10:? #6;"score:"
3550 BONUS=C*75
3560 FOR P=0 TO BONUS STEP 5
3570 POSITION 9,10:? #6;SCORE+P
3580 SOUND 0,P*25,10,15
3590 NEXT P
3600 SCORE=SCORE+BONUS
3610 SOUND 0,0,0,0
3620 MAN=0:CHUTE=0:NEXT JUMP
3630 GRAPHICS 2:POKE 53248,0:POKE 53249,
0
3640 ? #6;"YOU'VE SUCCESSFULLY"
3650 ? #6;"  LANDED 10 TIMES"
3660 ? #6
3670 ? #6;"your final score :-"
3680 ? #6;"     ";SCORE
3690 SETCOLOR 2,0,0
3700 FOR P=5 TO 200 STEP 3
```

```
3710 FOR G=P-5 TO P+5
3720 SOUND 0,G,10,15
3730 NEXT G
3740 SETCOLOR 1,RND(0)*15,8
3750 NEXT P
3760 SOUND 0,0,0,0
3770 END
10000 RAM=PEEK(106)-8:A=256*RAM
10010 GRAPHICS 0
10020 SETCOLOR 4,9,2:SETCOLOR 2,9,2
10030 PRINT CHR$(127);"PLEASE WAIT A LIT
TLE WHILE"
10040 PRINT CHR$(127);"*****************
*********"
10050 POKE 752,1:PRINT
10055 REM ** COPY CHARACTER SET FROM ROM
INTO RAM **
10060 FOR P=0 TO 1023
10070 POKE A+P,PEEK(57344+P)
10080 SOUND 0,PEEK(53770),10,2
10090 NEXT P:SOUND 0,0,0,0
10100 READ X
10110 IF X<0 THEN POKE 203,RAM:GOTO 1022
0
10120 FOR P=0 TO 7
10130 READ Y
10140 POKE (X*8)+A+P,Y
10150 NEXT P
10160 GOTO 10100
10165 REM ** DATA FOR REDEFINED CHARACTE
RS **
10170 DATA 1,153,189,153,126,60,60,66,12
9
10180 DATA 5,0,0,0,24,60,126,195,129
10190 DATA 3,255,255,255,255,255,255,255
,255
10200 DATA 6,16,8,132,194,255,2,4,8
10210 DATA -1
10220 R=RAM-8
10225 REM ** CLEAR OUT PLAYER MISSILE GR
APHICS MEMORY **
```

```
10230 FOR P=512 TO 768
10240 POKE R%256+P,0:NEXT P
10245 REM ** POKE IN DATA FOR CLOUD PLAY
ERS **
10250 FOR P=0 TO 7
10260 READ X
10270 POKE R%256+70+P+512,X
10280 POKE R%256+50+P+640,X
10290 NEXT P
10300 POKE 204,R
10310 RETURN
10315 REM ** DATA FOR P.M. GRAPHICS **
10320 DATA 16,126,127,255,126,62,28,8
```

3D NOUGHTS AND CROSSES

(Tastiera)

Questo gioco, scritto da Roland Marriott, richiede un computer con una memoria di 24K; potete giocare con il computer o con un amico. Per vincere dovete riuscire a disporre quattro quadrati dello stesso colore su una riga. Vi sono quattro livelli, da sinistra a destra, denominati A, B, C e D. Per fare la vostra mossa, battete innanzitutto una lettera corrispondente a un livello (A-D). Poi introducete due cifre in corrispondenza della posizione sul livello. Le posizioni sono numerate da sinistra verso destra a partire da 01 a 16 in questa maniera:

13	14	15	16
09	10	11	12
05	06	07	08
01	02	03	04

Ecco alcuni esempi di possibili mosse, ma notate che non si deve premere il tasto RETURN:

D15
A05
B10
C02
B16

Quando vi si chiederà quanti giocatori vogliono prender parte al gioco, premete '1' se volete giocare contro il computer, oppure premete '2' se preferite giocare con un amico.

```
0 REM ******************************
1 REM *  3D NOUGHTS AND CROSSES BY  *
2 REM *        ROLAND MARRIOTT      *
3 REM ******************************
4 GOSUB 3000
5 DIM SPACE(64,2):DIM MARK(64):DIM BLOB(
86,3):DIM HZ(40)
6 CL=0
7 GOSUB 6000
10 GRAPHICS 7+16:COLOR 1
17 SETCOLOR 2,12,8:SETCOLOR 4,0,3
18 SETCOLOR 1,4,8:SETCOLOR 0,2,6:SETCOLO
R 3,3,8
20 FOR X=0 TO 129 STEP 43
30 FOR Y=95 TO 30 STEP -1
40 PLOT X,Y:DRAWTO X+30,Y-30
50 NEXT Y
60 NEXT X
61 PLOT 0,28:DRAWTO 28,0:PLOT 0,27:DRAWT
O 27,0
62 FOR RT=0 TO 86 STEP 43
63 PLOT RT+41,95:DRAWTO RT+41,30:DRAWTO
RT+71,0:PLOT RT+40,95:DRAWTO RT+40,30:DR
AWTO RT+70,0
64 NEXT RT
65 COLOR 0:SETCOLOR 4,9,2
66 FOR Z=0 TO 129 STEP 43
70 FOR Y=91 TO 31 STEP -20
72 FOR X=1 TO 31 STEP 9
74 FOR A=0 TO 1
760 PLOT Z+X+A,Y-A-X+3:DRAWTO Z+X+A,Y-A-
X
780 NEXT A
850 NEXT X
900 NEXT Y
950 NEXT Z
1000 FOR P=1 TO 64:READ H,J
1010 SPACE(P,1)=H:SPACE(P,2)=J
1012 NEXT P
1015 IF PEEK(764)=255 THEN 1015
```

```
1016 CLOSE #1:OPEN #1,4,0,"K:":GET #1,W
1017 IF W<65 OR W>68 THEN POKE 764,255:G
OTO 1015
1018 IF W=65 THEN N=0:IF W=66 THEN N=16:
IF W=67 THEN N=32
1019 IF W=66 THEN N=16
1020 IF W=67 THEN N=32
1021 IF W=68 THEN N=48
1023 POKE 764,255
1024 IF PEEK(764)=255 THEN 1024
1025 CLOSE #5:OPEN #5,4,0,"K:":GET #5,Z
1026 IF Z<48 OR Z>49 THEN POKE 764,255:G
OTO 1024
1027 POKE 764,255
1028 CLOSE #4:OPEN #4,4,0,"K:":GET #4,WH
:IF WH<48 OR WH>57 THEN POKE 764,255:GOT
O 1027
1033 REM FOR TR=0 TO 2
1034 SP=N+10*(Z-48)+WH-48:LOCATE SPACE(S
P,1),SPACE(SP,2),GH:IF GH<>0 THEN GOTO 1
015
1035 CL=CL+1:IF CL=2 THEN CL=0
1036 FOR RY=15 TO 0 STEP -1:COLOR (RY+CL
+2):SETCOLOR 1,3,4
1037 FOR TR=0 TO 2
1038 PLOT SPACE(SP,1)+TR,SPACE(SP,2)-TR:
DRAWTO SPACE(SP,1)+TR,SPACE(SP,2)-TR-3
1039 NEXT TR:NEXT RY
1040 CLOSE #1
1042 FOR PQ=1 TO 64
1044 LOCATE SPACE(PQ,1),SPACE(PQ,2),V
1046 MARK(PQ)=V:NEXT PQ
1050 C=CL+2
1100 A=1:B=16:D=16:E=1:GOSUB 1400
1105 A=1:B=61:D=1:E=4:GOSUB 1400
1110 A=1:B=13:D=17:E=4:GOSUB 1400
1115 A=4:B=16:D=15:E=4:GOSUB 1400
1120 A=1:B=4:D=20:E=1:GOSUB 1400
1125 A=13:B=16:D=12:E=1:GOSUB 1400
1130 A=1:B=1:D=21:E=1:GOSUB 1400
1135 A=13:B=13:D=13:E=1:GOSUB 1400
```

```
1140 A=16:B=16:D=11:E=1:GOSUB 1400
1145 A=4:B=4:D=19:E=1:GOSUB 1400
1150 A=1:B=49:D=5:E=16:GOSUB 1400
1155 A=4:B=52:D=3:E=16:GOSUB 1400
1160 A=1:B=4:D=4:E=1:GOSUB 1400
1165 A=17:B=20:D=4:E=1:GOSUB 1400
1170 A=33:B=36:D=4:E=1:GOSUB 1400
1175 A=33:B=36:D=4:E=1:GOSUB 1400
1180 A=49:B=52:D=4:E=1:GOSUB 1400
1299 IF PLAYER=1 AND CL<>0 THEN GOTO 150
0:REM IF CL=0 THEN GOTO 1015
1300 GOTO 1015
1400 FOR PQ=A TO B STEP E
1410 IF MARK(PQ)=C AND MARK(PQ+D)=C AND
MARK(PQ+2*D)=C AND MARK(PQ+3*D)=C THEN G
OSUB 1450
1420 NEXT PQ
1430 RETURN
1450 PLOT SPACE(PQ,1)+1,SPACE(PQ,2)-1:DR
AWTO SPACE(PQ+3*D,1)+1,SPACE(PQ+3*D,2)-1
:RETURN
1500 REM MACHINE LOGIC
1600 FOR PQ=1 TO 64
1605 LOCATE SPACE(PQ,1),SPACE(PQ,2),V
1606 IF V=2 THEN V=10
1608 IF V=3 THEN V=50
1610 MARK(PQ)=V:NEXT PQ
1615 H=0
1620 A=1:B=16:D=16:E=1:GOSUB 1700
1625 A=1:B=61:D=1:E=4:GOSUB 1700
1630 A=1:B=13:D=17:E=4:GOSUB 1700
1635 A=4:B=16:D=15:E=4:GOSUB 1700
1640 A=1:B=4:D=20:E=1:GOSUB 1700
1645 A=13:B=16:D=12:E=1:GOSUB 1700
1650 A=1:B=1:D=21:E=1:GOSUB 1700
1655 A=13:B=13:D=13:E=1:GOSUB 1700
1660 A=16:B=16:D=11:E=1:GOSUB 1700
1665 A=4:B=4:D=19:E=1:GOSUB 1700
1670 A=1:B=49:D=5:E=16:GOSUB 1700
1675 A=4:B=52:D=3:E=16:GOSUB 1700
1680 A=1:B=4:D=4:E=1:GOSUB 1700
```

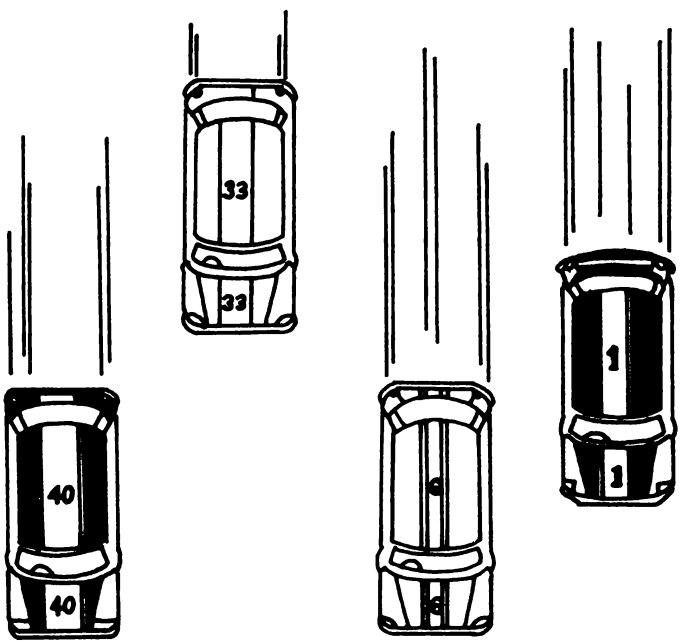
```
1682 A=17:B=20:D=4:E=1:GOSUB 1700
1686 A=33:B=36:D=4:E=1:GOSUB 1700
1688 A=49:B=52:D=4:E=1:GOSUB 1700
1690 GOTO 1740
1700 FOR PQ=A TO B STEP E
1705 H=H+1
1710 SUM=MARK(PQ)+MARK(PQ+D)+MARK(PQ+2%D
)+MARK(PQ+3%D)
1720 IF SUM=30 THEN GOTO 1830
1725 BLOB(H,1)=SUM:BLOB(H,2)=PQ:BLOB(H,3
)=D
1730 NEXT PQ:RETURN
1740 FOR TZ=1 TO H:IF BLOB(TZ,1)=150 THE
N GOTO 1850
1745 NEXT TZ
1746 YZ=0:FOR TZ=1 TO H
1748 IF BLOB(TZ,1)=20 OR BLOB(TZ,1)=100
THEN GOSUB 1860
1750 NEXT TZ:IF YZ<>0 THEN GOTO 1760
1755 GOTO 2050
1760 FOR TZ=1 TO YZ:FOR PZ=2 TO (YZ-1)
1765 IF HZ(TZ)=HZ(PZ) AND TZ<>PZ THEN SP
=HZ(TZ):GOTO 1035
1766 NEXT PZ:NEXT TZ
1770 IF YZ<>0 THEN SP=HZ(YZ):GOTO 1035
1825 GOTO 2050
1830 FOR TK=0 TO 3:IF MARK(PQ+TK%D)=0 TH
EN SP=PQ+TK%D:GOTO 1035
1835 NEXT TK
1850 REM DEFENCE
1855 FOR TK=0 TO 3:IF MARK(BLOB(TZ,2)+TK
%BLOB(TZ,3))=0 THEN SP=BLOB(TZ,2)+TK%BLO
B(TZ,3):GOTO 1035
1857 NEXT TK
1860 FOR P=0 TO 3
1863 IF MARK(BLOB(TZ,2)+P%BLOB(TZ,3))=0
THEN YZ=YZ+1:HZ(YZ)=BLOB(TZ,2)+P%BLOB(TZ
,3)
1865 NEXT P:RETURN
1890 GOTO 2050
1950 GOTO 2050
```

```
2050 SP=INT(64*RND(1)+1):IF MARK(SP)=0 T
HEN GOTO 1035
2055 GOTO 2050
3000 REM COVER PAGE
3020 GRAPHICS 2+16:SETCOLOR 4,13,2
3030 POSITION 3,1:? #6;"################"
3040 POSITION 3,2:? #6;"#  3D NOUGHTS #"
3050 POSITION 3,3:? #6;"# AND CROSSES #"
3052 POSITION 3,4:? #6;"#  BY  ROLAND #"
3054 POSITION 3,5:? #6;"#   MARRIOTT  #"
3060 POSITION 3,6:? #6;"################"
3070 POSITION 1,7:? #6;"PRESS start TO P
LAY"
3080 IF PEEK(53279)=6 THEN RETURN
3090 GOTO 3080
5000 DATA 1,93,10,84,19,75,28,66
5010 DATA 1,73,10,64,19,55,28,46
5020 DATA 1,53,10,44,19,35,28,26
5030 DATA 1,33,10,24,19,15,28,6
5040 DATA 44,93,53,84,62,75,71,66
5050 DATA 44,73,53,64,62,55,71,46
5060 DATA 44,53,53,44,62,35,71,26
5070 DATA 44,33,53,24,62,15,71,6
5080 DATA 87,93,96,84,105,75,114,66
5090 DATA 87,73,96,64,105,55,114,46
5100 DATA 87,53,96,44,105,35,114,26
5110 DATA 87,33,96,24,105,15,114,6
5120 DATA 130,93,139,84,148,75,157,66
5130 DATA 130,73,139,64,148,55,157,46
5140 DATA 130,53,139,44,148,35,157,26
5150 DATA 130,33,139,24,148,15,157,6
6000 GRAPHICS 2+16:? #6;"2 HUMAN PLAYERS
"
6010 ? #6;"OR 1 HUMAN VERSUS me"
6020 ? #6;" (TYPE 1 OR 2 )"
6030 OPEN #1,4,0,"K:":GET #1,M:IF M<49 O
R M>50 THEN POKE 764,255:CLOSE #1:GOTO 6
030
6040 PLAYER=M-48:CLOSE #1
6050 RETURN
```

RACE

(Jostick)

In questo gioco, scritto da Paul Dunning, dovete gareggiare contro altre tre automobili che cercano di coprire il maggior numero possibile di miglia. Usate il joystick per guidare la vostra automobile in corsa. Premendo il pulsante rosso del jostick potete cambiare marcia. Questo gioco impiega il linguaggio macchina, perciò dovete effettuare il "SAVE" prima di far girare il programma.




```
0 REM ** ROAD RACE - WRITTEN BY **
1 REM **          PAUL DUNNING     **
```

```
2 DIM S(4),G$(10),E$(100)
3 GRAPHICS 5:G$="LOW":POKE 752,1
5 GOSUB 1000
6 POKE 765,2:COLOR 2:PLOT 0,0:DRAWTO 79,
0:PLOT 79,47:DRAWTO 79,30:DRAWTO 0,30:PO
SITION 0,47:XIO 18,#6,0,0,"S:"
7 POKE 765,3:COLOR 3:PLOT 79,29:DRAWTO 7
9,1:DRAWTO 0,1:POSITION 0,29:XIO 18,#6,0
,0,"S:"
8 POKE 712,148
10 PP=PEEK(106)-16:POKE 54279,PP
20 PM=PP%256:POKE 559,62
21 FOR Q=53256 TO 53260:POKE Q,3:POKE Q-
8,RND(1)%255:NEXT Q
22 POKE 53248,100:POKE 53249,100:POKE 53
250,100:POKE 53251,50
23 POKE 704,109:POKE 705,89:POKE 706,79:
POKE 707,29:POKE 53277,3:POKE 559,62
24 FOR Q=PM+1024 TO PM+2048:POKE Q,0:NEX
T Q
30 RESTORE 210:FOR Q=PM+1144 TO PM+1280:
READ A:IF A=-1 THEN 32
31 POKE Q,A:NEXT Q
32 RESTORE 210:FOR Q=PM+1360 TO PM+1536:
READ A:IF A=-1 THEN 34
33 POKE Q,A:NEXT Q
34 RESTORE 210:FOR Q=PM+1576 TO PM+1792:
READ A:IF A=-1 THEN 36
35 POKE Q,A:NEXT Q
36 RESTORE 210:FOR Q=PM+1842 TO PM+2048:
READ A:IF A=-1 THEN 40
37 POKE Q,A:NEXT Q
40 POKE 623,1
50 SOUND 0,255,12,5:SOUND 1,145,12,5:SOU
ND 2,200,12,5
60 FOR Q=53256 TO 53260:POKE Q,3:POKE Q-
8,RND(1)%255:NEXT Q
70 POKE 704,109:POKE 705,89:POKE 706,79:
POKE 707,29:POKE 53277,3:POKE 559,62
75 ST=PM+1842:HG=INT(ST/256):LW=ST-256%H
G:POKE 203,LW:POKE 204,HG:POKE 205,10
80 P1=(RND(1)%100)+100:P2=(RND(1)%100)+1
```

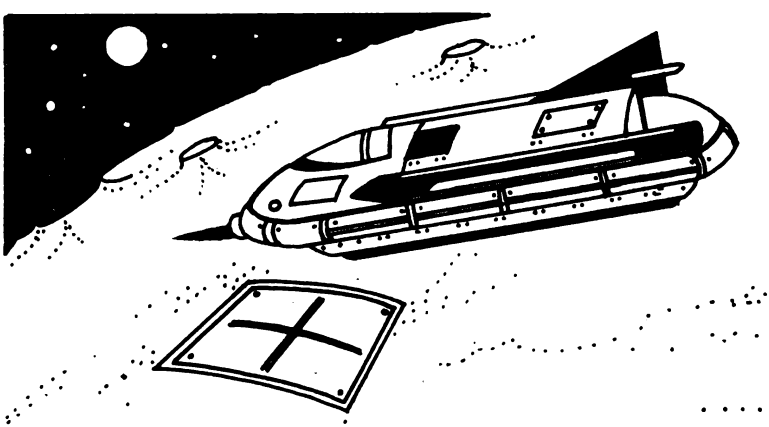
```
00:P0=(RND(1)%100)+100:P3=50
85 FOR Q=0 TO 3:S(Q)=(RND(1)%2)+1:NEXT Q
87 POKE 53248,P0:POKE 53249,P1:POKE 5325
0,P2:POKE 53251,50
88 FOR Q=1 TO 50:X=USR(ADR(E$),7):NEXT Q
90 IF G$="HIGH" THEN GOSUB 400:GOTO 120
92 POKE 53278,1
95 P0=P0+S(0):IF P0>255 THEN P0=0
98 X=USR(ADR(E$),STICK(0))
100 P1=P1+S(1):IF P1>255 THEN P1=0
105 X=USR(ADR(E$),STICK(0))
110 P2=P2+S(2):IF P2>255 THEN P2=0
120 MI=MI+0.01
124 X=USR(ADR(E$),STICK(0))
129 IF STRIG(0)=0 THEN GOSUB 500
130 POKE 53248,P0:POKE 53249,P1:POKE 532
50,P2
135 POKE 656,2:POKE 657,5:? "GEAR ";G$;"
  ":POKE 656,2:POKE 657,20:? "MILES ";MI
;" "
140 CP=PEEK(53263):CS=PEEK(53255)
150 IF CP<>0 THEN 600
160 IF CS<>4 THEN 600
190 GOTO 90
210 DATA 64,228,238,238,238,238,78,68,22
9,245,255,255,233,255,255,233,255,255,24
5,229,68,78,238,238,238
215 DATA 238,238,64,-1
300 C=INT(RND(1)%3):S(C)=(RND(1)%2)+1:RE
TURN
400 P0=P0-(S(0)%2):IF P0<0 THEN P0=255
405 X=USR(ADR(E$),STICK(0))
410 P1=P1-(S(1)%2):IF P1<0 THEN P1=255
415 X=USR(ADR(E$),STICK(0))
420 P2=P2-(S(2)%2):IF P2<0 THEN P2=255
425 X=USR(ADR(E$),STICK(0))
430 MI=MI+0.05:RETURN
500 IF G$="HIGH" THEN G$="LOW":SOUND 0,2
55,12,10:SOUND 1,245,12,10:SOUND 2,235,1
2,10:RETURN
510 G$="HIGH":SOUND 0,100,12,10:SOUND 1,
```

```
110,12,10:SOUND 2,90,12,10:RETURN
600 GOTO 640
605 SOUND 0,0,0,0:SOUND 1,0,0,0:SOUND 2,
0,0,0
610 POKE 53248,100:POKE 53249,100:POKE 5
3250,100
620 TRAP 620:? " ANOTHER GO ";:INPUT G$:
IF G$(1,1)="Y" THEN MI=0:G$="LOW":? CHR$
(125):GOTO 24
630 END
640 J=PEEK(203)+256*PEEK(204)
645 FOR Q=1 TO 250:IF PEEK(J+Q)=0 THEN N
EXT Q
650 ST=J+Q:HG=INT(ST/256):LW=ST-256*HG:P
OKE 203,LW:POKE 204,HG
655 FOR P=1 TO 60
660 X=USR(1536):SOUND 1,PEEK(53770),120,
15
670 POKE 707,PEEK(53770):NEXT P
700 POKE 53251,0:GOTO 605
1000 RESTORE 2000
1010 TRAP 1050:P=0
1020 P=P+1:READ A
1030 E$(P,P)=CHR$(A)
1040 GOTO 1020
1050 RESTORE 3000
1060 TRAP 1090:P=1536
1070 READ A:POKE P,A:P=P+1
1080 GOTO 1070
1090 TRAP 40000
1100 RETURN
2000 DATA 104,104,104,74,72,176,11,160,0
,177,203,136,145,203,200,200,208,247,104
,74,72
2010 DATA 176,11,160,0,177,203,200,145,2
03,136
2020 DATA 136,208,247,104,74,72,176,7,19
8,205,165
2030 DATA 205,141,3,208,104,74,176,7,230
,205,165,205,141,3,208,96,END
3000 DATA 104,160,0,173,10,210,145,203,2
00,192,25,208,246,96,END
```

LUNAR LANDING

(Joystick)

Anche questo gioco è stato scritto da Paul Dunning. Usate il joystick per scendere sulla piattaforma di atterraggio blu: otterrete il massimo punteggio eseguendo un atterraggio lento e morbido (ad esempio, ad una velocità di 5). Manovrando a velocità troppo elevata andrete a fracassarvi. Se riuscite ad atterrare, il vostro punteggio dipenderà da quanto era lenta la vostra velocità di discesa e da quale piattaforma di atterraggio avete usato. Vi saranno fornite allora 500 unità extra di carburante ed il gioco incomincerà di nuovo e terminerà quando si esaurirà il carburante.



```
0 REM ** LUNAR LANDER BY PAUL DUNNING **
1 DATA 108,60,81,40,53,40,60,40,64,40,72
,40,40,40,53,40,60,40,64,40,72,40,40,40,
53,40,60,40,64,40,60,40,72,60
2 DATA 108,60,96,80,60,40,64,40,72,40,81
```

```
,80,72,40,64,40,72,60,96,40,85,40,53,80,
40,40,45,40,50,40,53,40,60,40
3 DATA 68,40,72,40,81,40,53,80,-1
4 CLR :DIM A$(100),H$(11),E$(60):H$="  a
tari":POKE 752,1:HI=500
5 AA=1:FU=4000:TI=100:SC=0:HV=0:VV=HV:PO
KE 53278,1:GOSUB 900:GOSUB 4000
6 DATA 11,12,13,13,14,15,17,19,21,22,23,
23,24
7 DATA 25,26,29,30,31,33,33,33,33,33,33,
31
8 DATA 31,30,29,28,26,25,25,24,23,21,18,
16,15
9 DATA 14,14,14,15,17,19,21,23,24,25,26,
27,29
10 DATA 29,29,29,29,27,26,26,25,26,27,28
,28
11 DATA 27,26,25,25,23,23,23,23,23,23,23
,21,21
12 DATA 20,18,17,17
29 FOR I=1 TO 8:POKE 53247+I,0:NEXT I:GO
TO 495
30 END
120 REM POKE 53248,30
200 REM *****PLAYER SETUP *****
210 POKE 559,0:POKE 710,148:POKE 712,0
215 POKE 752,1:AL=110
230 RESTORE 235:FOR S2=1 TO 60:READ S3:E
$(S2,S2)=CHR$(S3):NEXT S2
235 DATA 104,104,104,74,72,176,11,160,2,
177,203,136,145,203,200,200,208,247,104,
74,72
236 DATA 176,11,160,253,177,203,200,145,
203,136
237 DATA 136,208,247,104,74,72,176,7,198
,205
238 DATA 165,205,141,0,208,104,74,176,7,
230,205,165,205,141,0,208,96,0,0
240 A=PEEK(106)-16:POKE 54279,A:POKE 204
,A+4:POKE 203,0:PMBASE=A*256
250 POKE 53277,3:POKE 704,40:POKE 53248,
150:POKE 205,RND(1)*130+70:POKE 53256,0
260 FOR I=PMBASE+1024 TO PMBASE+1280:POK
```

```
E I,0:NEXT I
270 RESTORE 290
280 FOR I=PMBASE+1155 TO PMBASE+1209:REA
D B:POKE I,B:NEXT I
290 DATA 8,8,60,60,126,126,195,195,126,1
26,60,60,24,24,126,126,165,165
300 DATA 165,165,165,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0
310 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0
320 POKE 559,62:GOTO 390
370 ST=STICK(0)
380 CO=PEEK(53252):IF CO>0 THEN GOSUB 50
0
385 POKE 53278,1
390 IF AA=1 THEN FOR Q=0 TO 100:A=USR(AD
R(E$),14):SOUND 0,255,8,15:NEXT Q:AA=0:A
=USR(ADR(E$),11):SOUND 0,0,0,0
400 POKE 53278,3
410 IF ST=14 AND FU>=30 THEN SOUND 0,255
,8,15:GOTO 420
415 SOUND 0,0,0,0
420 IF ST=7 OR ST=11 AND FU>=5 THEN SOUN
D 1,100,8,3:GOTO 430
425 SOUND 1,0,0,0
430 IF FU<=0 THEN FU=0:GOTO 455
431 IF ST=7 THEN HV=HV+0.05:FU=FU-5
432 IF ST=11 THEN HV=HV-0.05:FU=FU-5
440 HP=HP+HV:IF HP>1 THEN HP=0:SR=7
450 IF HP<-1 THEN HP=0:SR=11
455 VV=VV+0.02:VP=VP+VV:IF VP>1 AND SR=1
5 THEN VP=0:SR=13:AL=AL-1
456 IF VP>1 AND SR=11 THEN VP=0:SR=9:AL=
AL-1
457 IF VP>1 AND SR=7 THEN VP=0:SR=5:AL=A
L-1
458 IF VP<-1 THEN VP=0:SR=14:AL=AL+1
460 IF ST=14 AND FU>29 THEN VV=VV-0.05:F
U=FU-30
480 A=USR(ADR(E$),SR):SR=15
485 ? " SCORE=";SC;" ";CHR$(127);"  VVEL
=";VV*100;"      "
```

```
486 ? " FUEL =";FU;" ";CHR$(127);"  HUEL
=";HU%100;"    "
487 ? " TIME =";INT(TI);" ";CHR$(127);"
 ALT =";AL;" "
488 ? CHR$(28);CHR$(28);CHR$(28);
489 TI=TI-0.1
490 IF TI=0 THEN 720
492 GOTO 370
495 GOTO 210
500 IF CO=4 AND VV<0.5 THEN 600
505 IF CO=4 AND VV>0.4 THEN 700
510 IF CO=2 THEN 700
520 RETURN
600 FOR Q=1 TO 2:FOR W=255 TO 100 STEP -
1
610 SOUND 0,W,10,15:SOUND 1,W-10,10,15:S
OUND 2,W-20,10,15
620 NEXT W:NEXT Q
630 SS=0.5-VV:SO=SC:IF SS=0 THEN SC=SC+5
0:GOTO 650
640 IF AL<5 THEN SC=SC+(SS%100)%10:GOTO
650
642 IF AL<20 THEN SC=SC+(SS%100)%20:GOTO
 650
644 IF AL<60 THEN SC=SC+(SS%100)%5
650 IF FU<2500 THEN FU=FU+(SC-SO%2)
655 FOR Q=0 TO 3:SOUND Q,0,0,0:NEXT Q:AA
=1:POKE 53278,1:GOSUB 800:GOTO 210
700 FOR Q=0 TO 50:POKE 712,RND(1)%255:SO
UND 0,255,8,15:SOUND 1,240,8,15:SOUND 2,
220,8,15:NEXT Q
705 SOUND 0,0,0,0:SOUND 1,0,0,0:SOUND 2,
0,0,0
706 IF FU<30 THEN FU=0:GOTO 720
710 POKE 712,0:AA=1:GOSUB 800:GOTO 210
720 POKE 712,0:GOTO 810
800 VP=0:HP=0:HV=0:VV=0:TI=100:RETURN
810 IF SC>HI THEN GOSUB 1010
820 GRAPHICS 17:SETCOLOR 4,1,0
830 ? #6;"      GAME OVER"
835 ? #6:? #6;"  HIGH SCORE ";HI
836 ? #6:? #6;"     ";H$
```

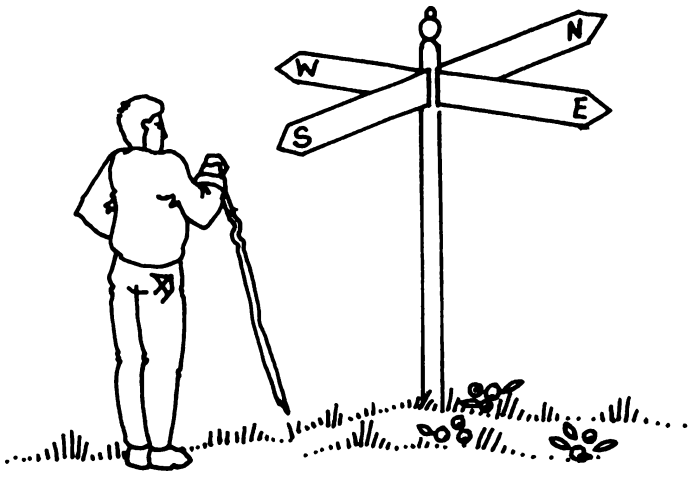
```
840 ? #6:? #6;"    PRESS START "
850 POKE 53248,0
860 IF PEEK(53279)=6 THEN RESTORE :GOTO
5
870 GOTO 860
899 END
900 REM INTRO
910 GRAPHICS 17:SETCOLOR 4,1,0:E=0
920 ? #6;"    LUNAR LANDER "
930 ? #6:? #6;" BY P.DUNNING SEP.82"
940 ? #6:? #6;"  USE JOYSTICK (0)"
950 ? #6:? #6;"         U P        "
960 ? #6:? #6;"          ^ "
970 ? #6;" left           right"
980 ? #6
985 POKE 53248,0
990 READ M:IF M<0 THEN 998
992 READ P
995 SOUND 0,M,10,15:FOR PP=0 TO P:NEXT P
P
997 GOTO 990
998 ? #6;"      good luck"
999 FOR X=1 TO 13:FOR Y=0 TO 255 STEP 20
:SOUND 0,Y,10,15:NEXT Y:NEXT X:SOUND 0,0
,0,0:RETURN
1000 RESTORE 1:GOTO 990
1010 H$="  ":TI=30:GRAPHICS 18:SETCOLOR 4
,1,0:? #6;"    GREAT SCORE ":? #6:X=1:Y=
1:HI=SC
1011 ? #6;"  a b c d e f g h "
1012 ? #6
1013 ? #6;"  i j k l m n o p "
1014 ? #6
1015 ? #6;"  q r s t u v w x "
1016 ? #6
1017 ? #6;"  y z . -      R E "
1018 POKE 53248,0
1020 FOR Q=10 TO 245:SOUND 0,Q-10,8,15:S
OUND 1,Q,8,15:SOUND 2,Q+10,8,15:NEXT Q
1030 POSITION X,Y:? #6;"XXX":POSITION X,
Y+1:? #6;"X":POSITION X+2,Y+1:? #6;"X":P
```

```
OSITION X,Y+2:? #6;"XXX"
1040 IF STICK(0)<>7 THEN 1050
1045 POSITION X,Y:? #6;"   ":POSITION X,
Y+1:? #6;" ":POSITION X+2,Y+1:? #6;" ":P
OSITION X,Y+2:? #6;"   "
1046 IF X>13 AND Y=7 THEN X=1:Y=1:GOTO 1
050
1047 IF X>13 THEN X=1:Y=Y+2:GOTO 1050
1048 X=X+2
1050 IF STICK(0)<>11 THEN 1060
1055 POSITION X,Y:? #6;"   ":POSITION X,
Y+1:? #6;" ":POSITION X+2,Y+1:? #6;" ":P
OSITION X,Y+2:? #6;"   "
1056 IF X<3 AND Y=1 THEN X=15:Y=7:GOTO 1
060
1057 IF X<3 THEN X=15:Y=Y-2:GOTO 1060
1058 X=X-2
1060 POSITION 6,10:? #6;"TIME ";INT(TI);
" ":TI=TI-0.1:IF TI=0 THEN 1100
1065 FOR PP=0 TO 20:NEXT PP
1069 IF STRIG(0)<>0 THEN 1080
1071 LOCATE X+1,Y+1,XX:IF XX=ASC("E") TH
EN 1100
1072 IF XX=ASC("R") AND LEN(H$)>1 THEN H
$=H$(1,LEN(H$)-1):GOTO 1080
1073 IF LEN(H$)>10 THEN 1080
1075 H$(LEN(H$)+1)=CHR$(XX)
1080 POSITION 4,11:? #6;H$;" "
1099 GOTO 1030
1100 FOR Q=0 TO 3:SOUND Q,0,0,0:NEXT Q
1110 RETURN
4000 GRAPHICS 5
4010 RESTORE 6:COLOR 2
4020 FOR X=0 TO 79
4030 READ Y
4040 PLOT X,39:DRAWTO X,Y
4050 NEXT X
4060 COLOR 3
4070 PLOT 18,32:DRAWTO 23,32
4080 PLOT 50,28:DRAWTO 54,28
4090 PLOT 67,22:DRAWTO 73,22
4100 RETURN
```

DECISION MAKER

(Tastiera)

Questo programma vi aiuterà a prendere decisioni su un determinato problema. Innanzitutto vi domanda quante sono le opzioni tra cui non riuscite a decidervi, poi vi chiede quanti sono i fattori che influenzano la vostra decisione (ad esempio, il prezzo dell'uovo, la qualità, la grandezza). Infine vi domanda quale valore attribuite ad un determinato fattore. Risponderete introducendo un numero compreso tra 0 e 100. Dopo aver fornito tutti i dati ed elencate tutte le varie opzioni, sul video in negativo vedrete apparire la decisione che il computer ritiene che voi dovreste prendere.



```
10 REM ** DECISION MAKER II  **
20 REM ** BY CHRIS CALLENDER **
21 REM ** ADAPTED BY P.BUNN  **
22 REM **********************
25 DIM A$(32):GOSUB 30:GOTO 80
30 GRAPHICS 0
40 SETCOLOR 2,8,2:SETCOLOR 4,8,2
50 PRINT ,"DECISION MAKER"
60 PRINT ,"**************"
70 PRINT
75 RETURN
80 PRINT "HOW MANY OPTIONS TO CHOOSE FRO
M"
82 TRAP 82:INPUT A$:O=VAL(A$)
90 PRINT "HOW MANY FACTORS INFLUENCE DEC
ISION"
100 TRAP 100:INPUT A$
110 F=VAL(A$)
120 DIM O$(O*32),F$(F*32)
130 O$(1)=" ":O$(O*32)=" ":O$(2)=O$
140 F$(1)=" ":F$(F*32)=" ":F$(2)=F$
150 GOSUB 30
160 FOR A=1 TO O
170 ? "ENTER NAME OF OPTION ";A
180 TRAP 180:INPUT A$:TRAP 40000
190 O$((A-1)*32+1,(A-1)*32+31)=A$
200 NEXT A
210 GOSUB 30
220 FOR A=1 TO F
230 ? "ENTER NAME OF FACTOR ";A
240 TRAP 240:INPUT A$:TRAP 40000
250 F$((A-1)*32+1,(A-1)*32+31)=A$
260 NEXT A
270 GOSUB 30
280 DIM M(O,F)
290 FOR A=1 TO O
300 FOR B=1 TO F
310 GOSUB 30
320 ? "HOW WOULD YOU RATE ";F$((B-1)*32+
1,(B-1)*32+31)
```

```
330 ? "ON THE ";O$((A-1)%32+1,(A-1)%32+3
1)
340 INPUT V
350 M(A,B)=V
360 NEXT B
370 NEXT A
380 GOSUB 30
390 ? "HERE IS WHAT I THINK YOU SHOULD D
O:-"
400 FOR A=1 TO O
410 C=0
420 FOR B=1 TO F
430 C=C+M(A,B)
440 NEXT B
450 IF C>TOP THEN TOP=C:NO=A
460 NEXT A
470 FOR A=1 TO O
480 IF A=NO THEN FOR P=1 TO 31:L=(A-1)%3
2+P:A$=O$(L,L):K=ASC(A$)+128:O$(L,L)=CHR
$(K):NEXT P
485 IF A=NO THEN FOR P=1 TO 31:L=(A-1)%3
2+P:K=ASC(O$(L,L)):IF K=160 THEN O$(L,L)
=" "
486 IF A=NO THEN NEXT P
490 ? O$((A-1)%32+1,(A-1)%32+31)
500 NEXT A
```

SAFE CRACKER

(Tastiera)

Questo magnifico gioco è stato scritto da Terry Davies e richiede un computer con 24K di memoria.
Lo scopo consiste nell'impossessarsi del maggior numero possibile di monete dalle casseforti di Mrs. Warren Fitzdobody. Per poter aprire una cassaforte dovete indovinare tre numeri della combinazione. Su richiesta del computer inserite il vostro primo numero: se esso risulterà essere entro ± 5 dal numero vero sentirete lo scatto del meccanismo della cassaforte. Se fate più di 10 tentativi per ogni numero, scatterà l'allarme ed arriverà la polizia (dalla finestra potrete vedere lampeggiare la luce intermittente di allarme delle loro automobili). Se riuscite ad indovinare il primo numero, continuate con il secondo e poi con il terzo. Se riuscite ad indovinare i tre numeri (una probabilità su 970.299) la cassaforte si spalancherà e riceverete alcune monete. Andate in un'altra stanza e tentate di nuovo. In questo gioco ci vuole fortuna, ma anche intelligenza. L'intelligenza, infatti, è necessaria per comprendere la logica delle combinazioni di apertura. Dopo aver giocato allo scassinatore ci penserete due volte prima di praticare questa "arte" sul serio!

```
1 REM ********************************
2 REM ** SAFE CRACKER  -            **
3 REM **   WRITTEN BY TERRY DAVIES **
4 REM ********************************
5 GRAPHICS 2:FLASH=0:P=10:DIM A$(5),D$(3
200),C(4),PT$(62):TT=0
10 POKE 708,P+10:POKE 709,145:POKE 752,8
:POKE 712,145:POKE 710,145
15 ? "              press start "
20 POSITION 4,4:? #6;"SAFE CRACKER"
```

```
25 POSITION 4,6:? #6;"safe cracker":GOSU
B 6000
30 IF PEEK(53279)=6 THEN 1000
35 P=P+0.1:FLASH=FLASH+1:IF FLASH>50 THE
N POKE 709,P:IF FLASH>50 THEN POKE 708,1
45:FLASH=-20:POKE 53279,0
40 IF FLASH=0 THEN POKE 709,145:IF FLASH
=0 THEN POKE 708,P:POKE 53279,0
50 IF P<200 THEN 30
1000 GRAPHICS 7:PX=4:LC=40:X=0:Y=0
1005 GOSUB 5100
1007 ? "PLEASE WAIT."
1010 GOSUB 30000
1020 FOR I=0 TO 4:POKE 708+I,C(I):NEXT I
1030 GOSUB 5199
1040 POKE 752,1:POKE 709,0:? :? "     HE
RE IS THE SAFE...SHHHHHH!!!":FOR W=1 TO
1500:NEXT W:? CHR$(125)
1043 ? " YOU BETTER PUT THE SIDE LIGHT O
N":? " THERE'S NO ONE AROUND..PRESS STAR
T":POKE 53279,0
1045 IF PEEK(53279)<>6 THEN 1045
1050 POKE 709,202:? CHR$(125);"THAT'S BE
TTER YOU CAN SEE WHAT YOU":? "ARE DOING
NOW":FOR W=1 TO 2000:NEXT W
1055 ? CHR$(125):GOTO 2000
1060 GOSUB 6000
1070 ? CHR$(125);"PRESS START TO HAVE AN
OTHER CRACK"
1071 POKE 710,30:FOR W=1 TO 200:NEXT W:P
OKE 710,138:FOR W=1 TO 99:NEXT W:POKE 71
0,148:FOR W=1 TO 250:POKE 710,138
1072 FOR W=1 TO 99:NEXT W:POKE 710,148:F
OR W=1 TO 250:NEXT W:POKE 710,138:FOR W=
1 TO 100:NEXT W:POKE 710,148
1073 FOR W=1 TO 250:NEXT W:POKE 710,138:
FOR W=1 TO 100:NEXT W:POKE 710,148
1074 POKE 53279,0:IF PEEK(53279)<>6 THEN
 1072
1075 GOTO 1020
2000 A=INT(RND(1)*81)+10:B=INT(RND(1)*81
)+10:C=INT(RND(1)*81)+10:J=0
2003 J=J+1:? "  WHATS YOUR FIRST NUMBER
```

```
1 TO 99 ":INPUT N:IF N>99 OR N<1 THEN 20
03
2005 IF J=11 THEN GOSUB 3950:GOTO 1060
2006 IF N<>A THEN 2008
2007 GOSUB 3850:GOTO 2499
2008 IF N>A-5 AND N<A+5 THEN GOSUB 3900:
GOTO 2003
2010 IF N>A+5 OR N<A-5 THEN GOSUB 4000:G
OTO 2003
2011 GOTO 2003
2499 J=0
2500 J=J+1:? "WHATS YOUR SECOND NUMBER":
INPUT N:IF N>99 OR N<0 THEN 2500
2510 IF J=11 THEN GOSUB 3950:GOTO 1060
2520 IF N<>B THEN 2530
2525 GOSUB 3850:GOTO 2999
2530 IF N>B-5 AND N<B+5 THEN GOSUB 3900:
GOTO 2500
2535 IF N>B+5 OR N<B-5 THEN GOSUB 4000:G
OTO 2500
2540 GOTO 2500
2999 J=0
3000 ? "  WHATS YOUR THIRD NUMBER":INPUT
 N:IF N>99 OR N<0 THEN 3000
3005 J=J+1
3010 IF J=11 THEN GOSUB 3950:GOTO 1060
3020 IF N<>C THEN 3030
3025 GOSUB 3850:? "YOU HAVE OPENED THE S
AFE":J=0:COLOR 0:PLOT 41,30:DRAWTO 39,16
3026 J=J+1:IF J>9 THEN 3028
3027 DRAWTO 41+J,16:DRAWTO 41+J,30:GOTO
3026
3028 CO=INT(RND(0)*200)+100:? CHR$(125);
"WELL DONE , THERE WAS ";CO;" COINS IN":
? "THAT SAFE !!....."
3029 TT=TT+CO:? "NOW TRY THIS ONE.....":
FOR W=1 TO 1000:NEXT W:GOTO 1020
3030 IF N>C-5 AND N<C+5 THEN GOSUB 3900:
GOTO 3000
3035 IF N>C+5 OR N<C-5 THEN GOSUB 4000:G
OTO 3000
3040 GOTO 3000
```

```
3850 FOR T=1 TO 100:SOUND 0,T,10,8:NEXT
T:? "YOU ARE RIGHT":SOUND 0,0,0,0:RETURN

3900 FOR S=20 TO 15 STEP -4:NEXT S:SOUND
 0,N,10,14:FOR W=1 TO 3:NEXT W:SOUND 0,0
,0,0:RETURN
3950 ? "BLIMEY  !! YOU HAVE SET THE ALAR
M OFF":? "LETS SCARPER....THE POLICE ":R
=0
3960 ? "YOU NICKED ";TT;" COINS FROM THE
 SAFES";:TT=0
3998 FOR N=20 TO 15 STEP -4:NEXT N:SOUND
 0,N,10,14:FOR W=1 TO 33:NEXT W:SOUND 0,
0,0,0:R=R+1:IF R=20 THEN RETURN
3999 GOTO 3998
4000 ? "    YOU ARE WRONG TRY AGAIN !":R
ETURN
5100 DL=PEEK(560)+PEEK(561)*256:MEM=PEEK
(106)*256:DAT=PEEK(DL+4)+256*PEEK(DL+5)
5108 RESTORE 5200:FOR P=1 TO 62:READ A:P
T$(P,P)=CHR$(A):NEXT P
5110 ST=DAT+Y*LC+INT(X/PX):RETURN
5199 GOSUB 5110:A=USR(ADR(PT$),ST,ADR(D$
),BYT,LIN,LC):RETURN
5200 DATA 104,104,133,204,104,133,203,10
4,133,206,104,133,205,104,104,133,207,10
4,104
5210 DATA 170,104,104,133,208,164,207,13
6,177,205
5220 DATA 145,203,136,192,255,208,247,24
,165
5230 DATA 203,101,208,133,203,144,2,230,
204,24,165,205,101,207,133,205,144,2,230
5240 DATA 206,202,208,219,96
6000 FOR J=1 TO 8:SOUND 0,47,10,8:FOR L=
1 TO 100:NEXT L:SOUND 0,64,10,8:FOR L=1
TO 100:NEXT L:NEXT J
6001 SOUND 0,0,0,0:RETURN
20000 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20001 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20002 DATA 0,0,0,12,0,0,0,0,0,0,0,0,0,0,
0
20003 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,195
```

```
,0
20004 DATA 63,255,207,255,63,255,207,255
,63,255,207,255,192,192,0
20005 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20006 DATA 0,0,0,0,0,0,0,0,195,12,63,255
,207,255,63
20007 DATA 255,207,255,63,255,207,255,19
2,192,192,0,0,0,0,0
20008 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20009 DATA 0,0,0,195,12,63,255,207,255,6
3,255,207,255,63,255
20010 DATA 207,255,192,192,192,0,0,0,0,0
,0,0,0,0,0
20011 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,195
,12
20012 DATA 63,255,207,255,63,255,207,255
,63,255,207,255,192,192,192
20013 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20014 DATA 0,0,0,0,0,0,0,0,195,12,63,255
,207,255,63
20015 DATA 255,207,255,63,255,207,255,19
2,192,192,0,0,0,0,0
20016 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20017 DATA 0,0,0,195,12,63,255,207,255,6
3,255,207,255,63,255
20018 DATA 207,255,192,192,192,0,0,0,0,0
,0,0,0,0,0
20019 DATA 0,0,0,0,0,0,63,255,207,255,19
2,0,0,195,12
20020 DATA 63,255,207,255,63,255,207,255
,63,255,207,255,192,192,192
20021 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20022 DATA 0,63,255,207,255,192,0,0,195,
12,63,255,207,255,63
20023 DATA 255,207,255,63,255,207,255,19
2,192,192,0,0,0,0,0
20024 DATA 0,0,0,0,0,0,0,0,0,0,0,63,255,
207,255
20025 DATA 192,0,0,195,12,0,0,0,0,0,0,0,
0,0,0
20026 DATA 0,0,0,192,192,0,0,0,0,0,0,0,0
,0,0
20027 DATA 0,0,0,0,0,0,63,255,207,255,19
```

```
2,0,0,195,12
20028 DATA 63,255,207,255,63,255,207,255
,63,255,207,255,192,192,192
20029 DATA 0,0,0,0,0,0,0,0,0,3,255,255,2
40,0,0
20030 DATA 0,63,255,207,255,192,0,0,195,
12,63,255,207,255,63
20031 DATA 255,207,255,63,255,207,255,19
2,192,192,0,0,0,0,0
20032 DATA 85,85,85,85,92,0,0,12,0,0,0,6
3,255,207,255
20033 DATA 192,0,0,195,12,63,255,207,255
,63,255,207,255,63,255
20034 DATA 207,255,192,192,192,0,0,0,0,0
,85,85,85,85,92
20035 DATA 0,0,12,0,0,0,63,255,207,255,1
92,0,0,195,12
20036 DATA 63,255;207,255,63,255,207,255
,63,255,207,255,192,192,192
20037 DATA 0,0,0,0,0,80,0,0,0,48,0,0,3,0
,0
20038 DATA 0,63,255,207,255,192,0,0,195,
12,63,255,207,255,63
20039 DATA 255,207,255,63,255,207,255,19
2,192,192,0,0,0,0,0
20040 DATA 80,0,0,0,192,0,0,3,0,1,85,85,
85,80,0
20041 DATA 0,0,0,195,12,63,255,207,255,6
3,255,207,255,63,255
20042 DATA 207,255,192,192,192,0,0,0,0,0
,80,3,195,3,0
20043 DATA 0,0,0,192,1,64,63,255,240,255
,192,0,0,195,12
20044 DATA 63,255,207,255,63,255,207,255
,63,255,207,255,192,192,192
20045 DATA 0,0,0,0,0,80,16,51,3,0,0,0,0,
192,1
20046 DATA 69,85,85,240,255,192,0,0,195,
12,0,0,0,0,0
20047 DATA 0,0,0,0,0,0,0,0,192,192,0,0,0
,0,0
20048 DATA 80,4,192,12,0,0,0,0,48,1,69,8
5,85,240,255
```

```
20049 DATA 192,0,0,195,12,63,255,207,255
,63,255,207,255,63,255
20050 DATA 207,255,192,192,192,0,0,0,0,0
,80,85,192,48,0
20051 DATA 0,0,0,12,1,69,81,85,240,255,1
92,0,0,195,12
20052 DATA 63,255,207,255,63,255,207,255
,63,255,207,255,192,192,192
20053 DATA 0,0,0,0,0,88,68,3,192,0,0,0,0
,12,1
20054 DATA 69,85,85,240,255,192,0,0,195,
12,63,255,207,255,63
20055 DATA 255,207,255,63,255,207,255,19
2,192,192,0,0,0,0,0
20056 DATA 88,20,192,192,0,0,0,0,3,1,69,
85,85,240,255
20057 DATA 192,0,0,195,12,63,255,207,255
,63,255,207,255,63,255
20058 DATA 207,255,192,192,192,3,192,0,0
,0,88,16,195,0,0
20059 DATA 0,0,0,3,1,69,85,85,240,255,19
2,0,0,195,12
20060 DATA 63,255,207,255,63,255,207,255
,63,255,207,255,192,192,192
20061 DATA 48,0,0,0,0,88,80,0,170,170,17
0,170,170,170,194
20062 DATA 69,0,85,240,0,0,0,0,195,12,63
,255,207,255,63
20063 DATA 255,207,255,63,255,207,255,19
2,192,195,0,0,0,0,0
20064 DATA 88,64,192,0,88,2,64,0,0,2,69,
85,85,240,255
20065 DATA 192,0,0,195,12,63,255,207,255
,63,255,207,255,63,255
20066 DATA 207,255,192,192,240,0,0,0,0,0
,88,76,12,192,88
20067 DATA 2,64,0,0,2,69,85,85,240,255,1
92,0,0,195,12
20068 DATA 63,255,207,255,63,255,207,255
,63,255,207,255,192,195,0
20069 DATA 0,0,0,0,0,88,0,0,0,88,2,64,0,
0,2
20070 DATA 69,85,85,240,255,192,0,0,195,
```

```
12,0,0,192,0,0
20071 DATA 0,0,0,0,12,0,0,0,240,192,0,0,
0,0,0
20072 DATA 88,0,0,0,88,2,64,0,0,2,69,85,
85,240,255
20073 DATA 192,0,0,195,12,63,255,255,255
,63,255,207,255,60,15
20074 DATA 207,255,207,0,0,0,0,0,0,0,88,
0,0,0,88
20075 DATA 2,64,0,0,2,69,85,85,240,255,1
92,0,0,3,12
20076 DATA 63,255,207,255,63,255,207,255
,56,15,207,255,240,192,0
20077 DATA 0,0,0,0,0,90,170,170,170,88,2
,64,0,0,2
20078 DATA 69,85,85,240,255,192,0,0,3,12
,63,255,207,255,63
20079 DATA 255,207,255,56,15,207,255,0,0
,192,0,0,0,0,0
20080 DATA 85,85,85,85,88,2,64,0,0,2,74,
255,255,240,255
20081 DATA 192,0,0,3,12,63,255,207,255,6
3,255,207,255,56,15
20082 DATA 207,240,192,0,192,0,0,0,0,0,0
,0,0,0,0
20083 DATA 2,64,0,0,2,85,85,85,80,255,19
2,0,0,3,12
20084 DATA 63,255,207,255,63,255,207,255
,56,15,207,12,3,192,192
20085 DATA 0,0,0,0,0,0,0,0,0,0,2,64,0,0,
0
20086 DATA 0,0,0,0,0,0,0,0,3,12,63,255,2
07,255,63
20087 DATA 255,207,255,56,15,192,0,3,192
,192,0,0,0,0,0
20088 DATA 0,0,0,0,0,2,64,0,0,0,0,63,255
,207,255
20089 DATA 192,0,0,3,12,63,255,207,255,6
3,255,207,255,56,12
20090 DATA 0,0,195,192,192,0,0,0,0,0,0,0
,0,0,0
20091 DATA 2,64,0,0,0,0,63,255,207,255,1
92,0,0,3,12
```

```
20092 DATA 0,0,0,0,0,0,0,0,8,0,0,12,3,19
2,192
20093 DATA 0,0,0,0,0,0,0,0,0,0,2,64,0,0,
0
20094 DATA 0,63,255,207,255,192,0,0,3,15
,255,255,255,255,255
20095 DATA 255,255,255,251,12,60,255,195
,192,192,0,0,0,0,0
20096 DATA 0,0,0,0,0,2,64,0,0,0,0,63,255
,207,255
20097 DATA 192,0,0,3,12,0,0,0,0,0,0,0,0,
11,0
20098 DATA 12,12,3,192,192,0,0,0,0,0,0,0
,0,0,0
20099 DATA 2,64,0,0,0,0,63,255,207,255,1
92,0,0,3,12
20100 DATA 0,0,0,0,0,0,0,0,11,0,12,12,3,
192,192
20101 DATA 0,0,0,0,0,0,0,0,0,0,2,64,0,0,
0
20102 DATA 0,63,255,207,255,192,0,0,3,12
,0,0,0,0,0
20103 DATA 0,0,0,11,12,12,12,3,192,192,0
,0,0,0,0
20104 DATA 0,0,0,0,0,2,64,0,0,0,0,63,255
,207,255
20105 DATA 192,0,0,3,12,0,0,0,0,0,0,0,0,
11,12
20106 DATA 12,12,3,192,192,0,0,0,0,0,0,0
,0,0,0
20107 DATA 2,64,0,0,0,0,0,0,0,0,0,0,0,3,
12
20108 DATA 0,0,0,0,0,0,0,0,11,12,12,12,3
,192,192
20109 DATA 0,0,0,0,0,0,0,0,0,0,2,64,0,0,
0
20110 DATA 0,0,0,0,0,0,0,0,3,12,0,0,0,0,
0
20111 DATA 0,0,0,11,12,12,12,3,192,192,0
,0,0,0,0
20112 DATA 0,0,0,0,0,2,64,0,0,0,0,0,0,0,
0
20113 DATA 0,0,0,3,12,0,0,0,0,0,0,0,0,11
```

```
,12
20114 DATA 12,12,3,192,192,0,0,0,0,0,0,0
,0,0,0
20115 DATA 2,64,0,0,0,0,0,0,0,0,0,0,0,3,
12
20116 DATA 0,0,0,0,0,0,0,0,11,12,12,12,3
,192,192
20117 DATA 0,0,0,0,0,0,0,0,0,0,2,64,0,0,
0
20118 DATA 0,0,0,0,0,0,0,0,3,12,0,0,0,0,
0
20119 DATA 0,0,0,11,12,12,12,3,192,192,0
,0,0,0,0
20120 DATA 0,0,0,0,0,2,64,0,0,0,0,0,0,0,
0
20121 DATA 0,0,0,3,12,0,0,0,0,0,0,0,0,11
,12
20122 DATA 12,12,3,192,192,0,0,0,0,0,0,0
,0,0,0
20123 DATA 2,64,0,0,0,0,0,0,0,0,0,0,0,3,
12
20124 DATA 0,0,0,0,0,0,0,0,11,12,12,12,3
,192,192
20125 DATA 0,0,0,0,0,0,0,0,0,0,2,64,0,0,
0
20126 DATA 0,0,0,0,0,0,0,0,3,12,0,0,0,0,
0
20127 DATA 0,0,0,11,12,12,12,3,192,192,0
,0,0,0,0
20128 DATA 0,0,0,0,0,2,64,0,0,0,0,0,0,0,
0
20129 DATA 0,0,0,3,12,0,0,0,0,0,0,0,0,11
,12
20130 DATA 12,63,255,192,192,0,0,0,0,0,0
,0,0,0,0
20131 DATA 2,64,0,0,0,0,0,0,0,0,0,0,0,3,
12
20132 DATA 0,0,0,0,0,0,0,0,11,12,12,51,1
92,252,63
20133 DATA 192,195,195,15,192,0,0,0,0,0,
2,64,0,0,0
20134 DATA 0,0,0,0,0,0,0,0,3,12,0,0,0,0,
0
```

```
20135 DATA 0,0,0,7,12,12,48,252,192,192,
0,0,0,0,0
20136 DATA 0,0,0,0,0,2,64,0,0,0,0,0,0,0,
0
20137 DATA 0,0,0,3,0,63,243,252,255,63,2
07,243,240,247,12
20138 DATA 207,240,15,192,48,0,0,0,0,0,0
,0,0,0,0
20139 DATA 2,64,0,0,0,0,0,0,0,0,0,0,0,0,
0
20140 DATA 0,0,0,0,0,0,0,0,7,12,12,48,3,
192,0
20141 DATA 0,0,0,0,0,0,0,0,0,0,2,64,0,0,
0
20142 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20143 DATA 0,0,0,7,11,255,252,0,240,0,0,
0,0,0,0
20144 DATA 0,0,0,0,0,2,64,0,0,0,0,0,0,0,
0
20145 DATA 0,0,0,0,0,15,255,63,207,207,2
43,252,255,55,2
20146 DATA 192,12,0,63,0,0,0,0,0,0,0,0,0
,0,0
20147 DATA 2,64,0,0,0,0,0,0,0,0,0,0,0,0,
0
20148 DATA 0,0,0,0,0,0,0,0,4,2,176,0,48,
15,192
20149 DATA 0,0,0,0,0,0,0,0,0,0,2,64,0,0,
0
20150 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20151 DATA 0,0,0,7,0,172,0,0,0,240,0,0,0
,0,0
20152 DATA 0,0,0,0,0,2,64,0,0,0,0,0,0,0,
0
20153 DATA 0,0,0,0,0,0,255,243,252,252,2
52,255,63,199,192
20154 DATA 40,0,3,0,12,0,0,0,0,0,42,170,
170,170,170
20155 DATA 170,170,170,165,85,85,85,85,8
5,85,255,255,255,0,0
20156 DATA 0,0,0,0,0,0,0,0,4,51,10,0,0,0
,3
20157 DATA 0,0,0,0,0,0,0,0,0,0,2,64,0,0,
```

```
0
20158 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20159 DATA 0,0,0,4,12,194,128,0,48,0,240
,0,0,0,0
20160 DATA 0,0,0,0,0,2,64,0,0,0,0,0,0,0,
0
20161 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,3
20162 DATA 0,160,0,0,0,12,0,0,0,0,0,0,0,
0,0
20163 DATA 2,64,0,0,0,0,0,0,0,0,0,0,0,0,
0
20164 DATA 0,0,0,0,0,0,0,0,0,0,0,48,40,0,3
,0
20165 DATA 3,255,255,255,252,0,0,0,0,0,2
,64,0,0,0
20166 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20167 DATA 0,0,0,0,0,12,10,0,0,192,3,192
,0,0,0
20168 DATA 0,0,0,0,0,2,64,0,0,0,0,0,0,0,
0
20169 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20170 DATA 51,0,128,0,48,3,192,0,0,0,0,0
,0,0,0
20171 DATA 2,64,0,0,0,0,0,0,0,0,0,0,0,0,
0
20172 DATA 0,0,3,0,0,0,0,0,0,0,12,0,32,0
,15
20173 DATA 3,192,0,0,0,0,0,0,10,170,170,
170,170,160,0
20174 DATA 0,2,168,10,170,130,128,21,72,
70,5,143,197,223,255
20175 DATA 243,255,0,0,0,3,0,8,0,0,195,1
92,0,0,0
20176 DATA 0,0,0,42,170,170,170,170,168,
0,0,42,128,154,106
20177 DATA 0,5,85,69,65,127,0,0,0,0,0,0,
0,0,0
20178 DATA 0,192,2,0,0,51,192,0,0,0,0,0,
2,170,170
20179 DATA 170,170,170,170,128,2,170,130
,170,168,6,138,21,21,71
20180 DATA 213,255,255,223,207,240,255,2
52,0,0,0,48,0,128,0
```

```
20181 DATA 15,192,0,0,0,0,0,0,0,0,0,0,0,
0,0
20182 DATA 43,248,60,42,195,254,63,255,2
55,245,89,81,85,255,204
20183 DATA 243,63,252,0,0,0,12,0,32,0,3,
192,0,0,0
20184 DATA 0,0,0,0,0,0,0,0,0,0,168,32,5,
22,2
20185 DATA 168,42,170,21,79,245,0,0,0,4,
4,1,83,0,0
20186 DATA 0,3,0,11,255,255,192,0,0,0,0,
0,0,0,0
20187 DATA 0,0,0,0,42,170,138,170,170,42
,168,165,85,255,245
20188 DATA 127,241,63,255,207,207,207,25
5,192,0,0,0,192,12,0
20189 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,2,1
70
20190 DATA 168,10,170,168,170,160,1,84,1
65,70,5,81,85,85,207
20191 DATA 207,207,255,240,0,0,0,0,12,0,
0,0,0,0,0
20192 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20193 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20194 DATA 0,0,48,12,0,0,0,0,0,0,0,0,0,0
,0
20195 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20196 DATA 0,0,0,0,0,0,0,0,0,0,0,0,12,12
,0
20197 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20198 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20199 DATA 0,0,0,0,0,0,0,3,12,0,0,0,0,0,
0
20200 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20201 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20202 DATA 0,0,0,204,0,0,0,0,0,0,0,0,0,0
,0
20203 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20204 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,48,
0
20205 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20206 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

```
20207 DATA 0,0,0,0,0,0,0,0,12,0,0,0,0,0,
0
20208 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20209 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
20210 DATA 0,0,0,0,12,0,0,0,0,0,0
30000 BYT=40:LIN=79:GR=7:C(0)=6:C(1)=0:C
(2)=148:C(3)=70:C(4)=0
30010 RESTORE 20000:FOR P=1 TO 3143
30020 READ A:D$(P,P)=CHR$(A)
30030 NEXT P
30040 RETURN
```

TANK BATTLE

(Due joystick)

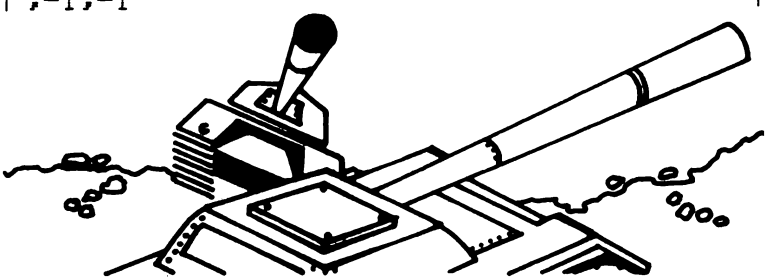
In questo gioco, progettato per due giocatori, muovete i vostri carri armati e fate fuoco per distruggere l'avversario. Se riuscirete a farlo otterrete un punto. Muovendo il joystick verso l'alto si avvicinano i motori del carro armato, il quale comincia ad avanzare. Per girare, muovete il joystick rispettivamente verso sinistra e verso destra. Per sparare premete il pulsante rosso.

```
1 REM ********************************
2 REM *          TANK BATTLE         *
3 REM *          -----------         *
4 REM *     Written by Paul Bunn     *
5 REM ********************************
6 REM
10 DIM F1$(20),F2$(20),X(8),Y(8):F1$(1)=
CHR$(172):F1$(20)=CHR$(172):F1$(2)=F1$:F
2$=",,,,,,,,,,,,,,,,,,,,"
20 RAM=PEEK(106)-16:POKE 756,RAM:IF PEEK
(203)<>200 THEN GOSUB 890:POKE 203,200
30 RESTORE 960:FOR Q=1 TO 8:READ XX,YY:X
(Q)=XX:Y(Q)=YY:NEXT Q:C1=1:C=1
40 GRAPHICS 2
50 ? #6;"      tank battle "
60 ? #6;"     ************"
70 ? :? " ENTER TIME TO PLAY ";:INPUT TI
80 REM
90 GRAPHICS 17:POKE 756,RAM:F1=14:F2=14:
X=5:Y=14:Q=15:W=5:FF1=1:FF2=1
100 S=STICK(0):IF S=7 THEN C=C+1:IF C>8
THEN C=1
110 IF S=11 THEN C=C-1:IF C<1 THEN C=8
120 IF S=14 THEN COLOR 32:PLOT X,Y:X=X+X
(C):Y=Y+Y(C):SOUND 0,200,8,8
130 IF S<>14 THEN SOUND 0,0,0,0
140 IF X<2 OR X>18 THEN X=X+-X(C):X=(X=2
)*18+(X=18)*2
```

```
150 IF Y<2 OR Y>18 THEN Y=Y+-Y(C):Y=(Y=2
)*18+(Y=18)*2
160 COLOR C:PLOT X,Y
170 IF STRIG(0)=0 AND FF1=1 THEN GOSUB 3
40
180 IF STRIG(1)=0 AND FF2=1 THEN GOSUB 5
20
190 S=STICK(1):IF S=7 THEN C1=C1+1:IF C1
>8 THEN C1=1
200 IF S=11 THEN C1=C1-1:IF C1<2 THEN C1
=8
210 IF S=14 THEN COLOR 32:PLOT Q,W:Q=Q+X
(C1):W=W+Y(C1):SOUND 1,100,8,8
220 IF S<>14 THEN SOUND 1,0,0,0
230 IF Q<2 OR Q>18 THEN Q=Q+-X(C1):Q=(Q=
2)*18+(Q=18)*2
240 IF W<2 OR W>18 THEN W=W+-Y(C1):W=(W=
18)*2+(W=2)*18
250 COLOR C1:PLOT Q,W
260 IF F1<1 THEN POSITION 0,21:? #6;"out
 of shots":FF1=0:GOTO 280
270 POSITION 0,21:? #6;"shots1";F1$(1,F1
);" "
280 IF F2<1 THEN POSITION 0,22:? #6;"out
 of shots":FF2=0:GOTO 300
290 POSITION 0,22:? #6;"shots2";F2$(1,F2
);" "
300 POSITION 5,0:? #6;S1:POSITION 15,0:?
 #6;S2:POSITION 8,0:? #6;"TIME":POSITION
 9,1:? #6;INT(TI);" "
310 IF PEEK(53279)=6 THEN GOTO 90
320 TI=TI-0.1:IF TI<0.1 THEN 800
330 GOTO 100
340 X1=X+X(C):Y1=Y+Y(C)
350 IF C=1 OR C=5 THEN FC=10
360 IF C=3 OR C=7 THEN FC=12
370 IF C=2 OR C=6 THEN FC=11
380 IF C=4 OR C=8 THEN FC=13
390 FOR F=1 TO 5:LOCATE X1,Y1,XX:COLOR F
C:PLOT X1,Y1
400 IF XX<>32 AND XX<10 THEN GOSUB 700:S
1=S1+1:HH=1:GOTO 450
410 SOUND 0,255-F*50,8,15:SOUND 1,150-F*
```

```
30,8,155
420 IF X1>18 OR X1<1 THEN 450
430 IF Y1>18 OR Y1<1 THEN 450
440 X1=X1+X(C):Y1=Y1+Y(C):NEXT F
450 X1=X+X(C):Y1=Y+Y(C)
460 FOR E=1 TO 5:COLOR 32:PLOT X1,Y1
470 IF X1>18 OR X1<1 THEN 500
480 IF Y1>19 OR Y1<1 THEN 500
490 X1=X1+X(C):Y1=Y1+Y(C):NEXT E
500 SOUND 0,0,0,0:SOUND 1,0,0,0:F1=F1-1:
IF HH=1 THEN HH=0:GOTO 90
510 RETURN
520 X2=Q+X(C1):Y2=W+Y(C1)
530 IF C1=1 OR C1=5 THEN FC=10
540 IF C1=3 OR C1=7 THEN FC=12
550 IF C1=2 OR C1=6 THEN FC=11
560 IF C1=4 OR C1=8 THEN FC=13
570 FOR F=1 TO 5:LOCATE X2,Y2,XX:COLOR F
C:PLOT X2,Y2
580 SOUND 0,255-F*50,8,15:SOUND 1,150-F*
30,8,15
590 IF XX<>32 AND XX<10 THEN GOSUB 750:S
2=S2+1:HH=1:GOTO 630
600 IF X2>18 OR X2<1 THEN 630
610 IF Y2>18 OR Y2<1 THEN 630
620 X2=X2+X(C1):Y2=Y2+Y(C1):NEXT F
630 X2=Q+X(C1):Y2=W+Y(C1)
640 FOR E=1 TO 5:COLOR 32:PLOT X2,Y2
650 IF X2>18 OR X2<1 THEN 680
660 IF Y2>19 OR Y2<1 THEN 680
670 X2=X2+X(C1):Y2=Y2+Y(C1):NEXT E
680 SOUND 0,0,0,0:SOUND 1,0,0,0:F2=F2-1:
IF HH=1 THEN HH=0:GOTO 90
690 RETURN
700 FOR G=15 TO 0 STEP -3
710 FOR H=0 TO 2
720 SOUND H,RND(1)*50+100,8,G
730 NEXT H
740 NEXT G:FOR G=0 TO 2:SOUND G,0,0,0:NE
XT G:RETURN
750 FOR G=15 TO 0 STEP -3
```

```
760 FOR H=0 TO 2
770 SOUND H,RND(1)*50+100,8,G
780 NEXT H
790 NEXT G:FOR G=0 TO 2:SOUND G,0,0,0:NE
XT G:RETURN
800 POSITION 5,10:? #6;"GAME OVER"
810 S1=0:S2=0:FOR Q=1 TO 5
820 SOUND 0,100+C,10,15:SOUND 1,120+C,10
,15:SOUND 2,90+C,10,15
830 C=C-1:IF C=-30 THEN 850
840 GOTO 820
850 C=0:NEXT Q
860 FOR Q=0 TO 2:SOUND Q,0,0,0:NEXT Q
870 IF PEEK(53279)=6 THEN RUN
880 POKE 53279,0:GOTO 870
890 FOR Q=0 TO 1024:A=PEEK(57344+Q):POKE
 RAM*256+Q,A:NEXT Q:C=8
900 READ A:IF A=-1 THEN RETURN
910 POKE RAM*256+C,A:C=C+1:GOTO 900
920 DATA 24,24,219,219,255,255,255,219,2
5,50,100,249,187,62,12,24,254,254,48,127
,127,48,254,254
925 DATA 24,12,62,187,249,100,50,25
930 DATA 219,255,255,255,219,219,24,24,2
4,50,124,221,159,38,76,152
935 DATA 127,127,12,254,254,12,127,127,1
52,76,38,159,221,124,50,24
940 DATA 112,120,248,254,255,127,60,56,2
4,24,24,24,24,24,24,24,1,2,4,8,16,32,64,
128,0,0,0,255,255,0,0,0
950 DATA 128,64,32,16,8,4,2,1,-1
960 DATA 0,-1,1,-1,1,0,1,1,0,1,-1,1,-1,0
,-1,-1
```



DIGIT DODGE

Usate i tasti 'Z' e 'X' per muovere il vostro mangianumeri. Premete '.' per sparare e "masticare" i numeri che appaiono sul mangia-numeri. Distruggendo l'asterisco ('*'), che nasconde un punteggio misterioso, otterrete un "bonus" da aggiungere ai vostri punti. Il gioco termina quando i numeri invasori raggiungono il vostro cacciatore di numeri.

```
0 GOTO 135:REM ** DIGIT DODGE BY P.BUNN
1 FOR I=50 TO 75 STEP 2:SOUND 0,I,10,15:
NEXT I:I=0:Y=M+48:IF M=10 THEN Y=10
2 Q$=CHR$(Y):IF A$(1,1)=Q$ THEN I=1:S=S+
10
3 IF A$(2,2)=Q$ AND I=0 THEN A$(2,2)=A$(
1,1):I=1:S=S+20
4 IF A$(3,3)=Q$ AND I=0 THEN A$(3,3)=A$(
2,2):A$(2,2)=A$(1,1):I=1:S=S+30
5 IF A$(4,4)=Q$ AND I=0 THEN A$(4,4)=A$(
3,3):A$(3,3)=A$(2,2):A$(2,2)=A$(1,1):I=1
:S=S+40
6 IF A$(5,5)=Q$ AND I=0 THEN A$(5,5)=A$(
4,4):A$(4,4)=A$(3,3):A$(3,3)=A$(2,2):I=1
:S=S+50
7 IF A$(6,6)=Q$ AND I=0 THEN A$(6,6)=A$(
5,5):A$(5,5)=A$(4,4):A$(4,4)=A$(3,3):A$(
3,3)=A$(2,2):I=1:S=S+60
8 IF I=1 THEN A$(1,1)=" ":N=N+1
10 IF I=1 AND Q$=CHR$(10) THEN FOR I=255
 TO 20 STEP -18:SOUND 0,I,10,15:NEXT I:S
=S+(INT(RND(0)*7+3)*100)
11 POSITION 1,0:? #6;SC$;S:? #6;"  high:
";HS
12 SOUND 0,0,0,0:RETURN
20 FOR I=14 TO 0 STEP -2:SOUND 0,M+15*30
,10,I:NEXT I:RETURN
135 DIM A$(6),Q$(1),SC$(6):GOSUB 2000
140 GRAPHICS 2:SETCOLOR 4,2,2:SETCOLOR 2
```

```
,2,4
150 A$="      ":A$(6,6)=CHR$(RND(0)*9+48)
160 S=0:N=5:M=0:U=18:GOSUB 11
165 Y=M+48:IF M=10 THEN Y=10
170 POSITION 5,5:? #6;CHR$(Y);":";A$
180 I=PEEK(764):I2=PEEK(53775)
190 IF I=22 AND I2=251 THEN M=M+1:GOSUB
20:IF M>10 THEN M=0
200 IF I=23 AND I2=251 THEN M=M-1:GOSUB
20:IF M<0 THEN M=10
210 IF I=34 AND I2=251 THEN GOSUB 1
215 I=0
220 TRAP 1000:O=O+1:IF O/U=INT(O/U) THEN
 A$(N,N)=CHR$(RND(0)*10+48):I=1
222 IF I=1 THEN IF A$(N,N)=CHR$(58) THEN
 A$(N,N)=CHR$(10)
224 IF I=1 THEN N=N-1:I=0
225 IF INT(RND(0)*20)=3 THEN U=U-1:IF U<
8 THEN U=8
230 TRAP 40000:GOTO 165
1000 ? #6:? #6;"..invaded..":FOR I=90 TO
 220 STEP 11:FOR H=80 TO 90 STEP 2:SOUND
 0,H,10,15:SOUND 1,I,10,15
1010 SETCOLOR 4,H,4
1020 POKE 709,INT(RND(0)*15)*16+8
1030 NEXT H:NEXT I:SETCOLOR 4,2,4:POKE 7
64,255
1040 SOUND 0,0,0,0:SOUND 1,0,0,0:IF S>HS
 THEN HS=S:POSITION 0,2:? #6;"high score
!!!!"
1050 ? "ANOTHER TRY ";:TRAP 1050:INPUT A
$
1055 IF A$(1,1)="N" THEN GRAPHICS 0:? "G
OODBYE.":? :? :END
1060 IF A$(1,1)="Y" THEN 140
1070 GOTO 1050
2000 FOR P=1 TO 6
2010 READ N
2020 SC$(P,P)=CHR$(N)
2030 NEXT P
2040 DATA 243,227,239,242,229,186
2050 RETURN
```

GRAND PRIX 2

(Joystick)

Usate il joystick per guidare la vettura di formula che corre lungo la pista. Se riuscite ad arrivare al traguardo, vincerete un "bonus" e il vostro punteggio verrà raddoppiato!

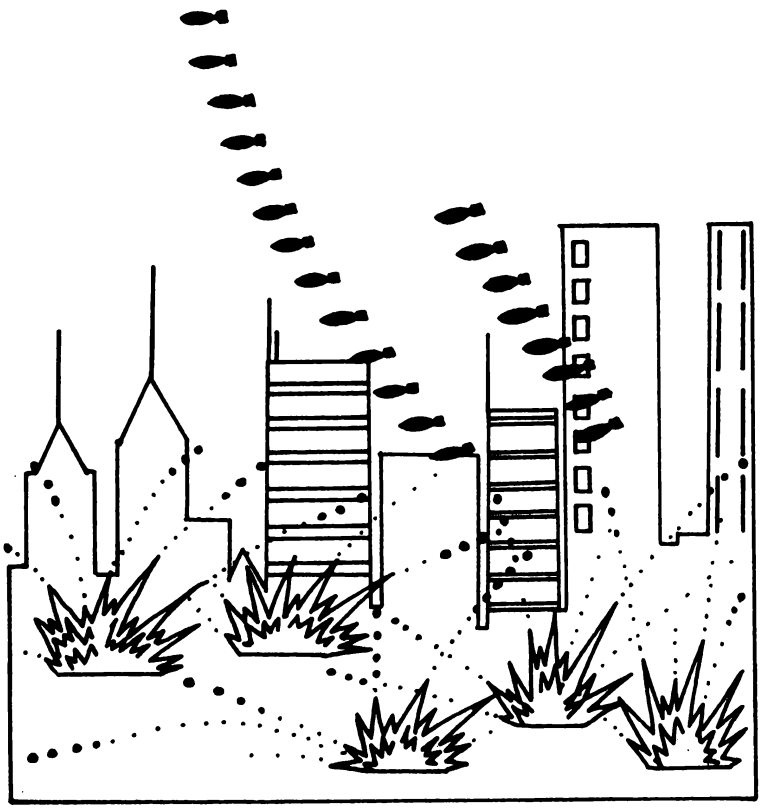
```
10 REM ** GRAND PRIX 2 - P.BUNN **
15 PI=5:GOSUB 3000:CRASH=1000
20 X2=14:X=116:S=PEEK(106)-8:Q=S*256:FOR
 N=Q+512 TO Q+640:POKE N,0:NEXT N:POKE 5
4279,S
30 POKE 559,46:POKE 53248,X:POKE 704,216
:POKE 704,70:POKE 53256,1
40 FOR N=Q+552 TO Q+561:READ A:POKE N,A:
NEXT N
50 DATA 129,195,165,24,24,153,219,165,36
,24
60 GRAPHICS 0:SETCOLOR 2,0,0:POKE 53277,
3:POKE 559,46
65 POKE 752,1:POKE 53278,A
66 FOR P=0 TO 23:POKE 201,X2:? ,S2$:NEXT
 P
70 S=STICK(0):X=X+(S=7)*2:X=X-(S=11)*2:P
OKE 53248,X
72 SOUND 0,40,10,15:SC=SC+PI:SOUND 0,0,0
,0
75 IF O THEN O=0:Z=2:GOTO 110
80 A=PEEK(53770)
90 IF A>85 AND A<170 THEN Z=2
100 IF A>170 AND X2<15 THEN Z=3:O=1
105 IF A<85 AND X2>2 THEN X2=X2-1:Z=1:O=
1
110 POKE 201,X2
115 IF A=192 AND  NOT I THEN ? ,F$:I=1:G
OTO 145
120 IF Z=1 THEN ? ,S1$
```

```
130 IF Z=2 THEN ? ,S2$
140 IF Z=3 THEN ? ,S3$:X2=X2+1
145 Y=PEEK(53252):IF Y<>0 AND I THEN GOT
O 2000
150 IF Y<>0 THEN GOTO CRASH
160 GOTO 70
1000 REM ** YOU CRASHED! **
1010 N=INT(RND(0)*10):POKE Q+552+N,PEEK(
53770):SOUND 0,RND(0)*20+20,80,15:POKE 7
04,PEEK(53770)
1020 IF PEEK(53770)<240 THEN 1010
1030 POKE 53248,0:? :? "YOU SCORED:";SC:
? :? :? "HIT ANY KEY.":POKE 764,255
1035 SOUND 0,0,0,0
1040 IF PEEK(764)=255 THEN 1040
1050 RUN
2000 FOR Y=0 TO 255:POKE 710,Y:NEXT Y
2005 ? CHR$(125);"YOUR SCORE :";SC:POKE
710,0
2010 ? "     BONUS  :1000":B=1000
2020 FOR G=1 TO 1000 STEP 10:B=B-10:SC=S
C+10:POSITION 14,0:? SC:POSITION 14,1:?
B;"  ":SOUND 0,G/4,10,10:NEXT G
2030 SOUND 0,0,0,0
2040 ? :PI=PI%2:? "STEP VALUE NOW IS ";P
I
2045 RESTORE :I=0
2050 GOTO 20
3000 DIM F$(8),S1$(8),S2$(8),S3$(8)
3010 RESTORE 3000:FOR P=1 TO 8
3020 READ A,B,C,D
3030 F$(P,P)=CHR$(A)
3040 S1$(P,P)=CHR$(B)
3050 S2$(P,P)=CHR$(C)
3060 S3$(P,P)=CHR$(D)
3070 NEXT P
3080 RESTORE :RETURN
3090 DATA 160,6,22,7,198,32,32,32
3100 DATA 201,32,32,32,206,32,32,32
3110 DATA 201,32,32,32,211,32,32,32
3120 DATA 200,32,2,32,160,6,32,7
```

CITY BOMB

(Joystick)

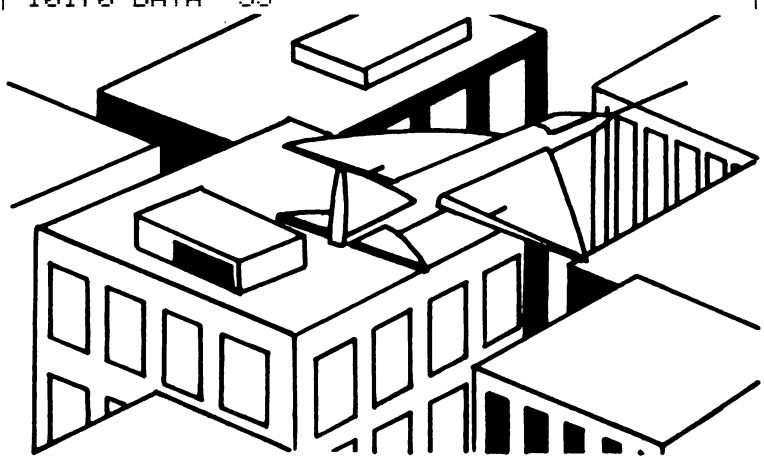
Il vostro aeroplano ha avuto un'avaria ai motori e sta scendendo lentamente verso una città sottostante. Bombardando la città e uccidendo milioni di persone potrete atterrare senza problemi, altrimenti vi fracasserete! Premete il pulsante sul joystick ogni volta che volete sganciare una bomba.



```
10 REM ******************************
20 REM ** CITY BOMB   VERSION II  **
30 REM **  WRITTEN BY PAUL BUNN   **
40 REM **       JANUARY 1983       **
50 REM ******************************
60 REM
65 IF PEEK(203)=PEEK(106)-8 THEN 80
70 GOSUB 10000:REM * INITIALIZATION *
80 GRAPHICS 0:SETCOLOR 2,8,0:SETCOLOR 4,
8,0
90 OPEN #1,4,0,"K:":POKE 752,1
100 ? "WHAT SKILL LEVEL (1-9)"
110 GET #1,A:A=A-48
120 IF A<1 OR A>9 THEN 110
130 A=A+3
135 POKE 756,PEEK(203)
140 ? CHR$(125):FOR I=1 TO 38
150 Y=INT(RND(0)*A)+1
160 FOR P=22 TO 22-Y STEP -1
170 POSITION I,P:? "$":REM *CITY CHARACT
ER *
180 NEXT P
190 FOR P=15 TO 0 STEP -1
200 SOUND 0,I*6+20,10,P
210 NEXT P
220 NEXT I
230 COLOR 94
240 PLOT 1,23:DRAWTO 39,23
250 X=1:Y=0:BM=0:SC=0
260 POSITION X,Y:OX=X:OY=Y
270 PRINT "%";:REM *PLANE CHARACTER*
280 X=X+1
290 IF X=39 THEN X=1:Y=Y+1
300 IF X>25 AND Y=22 THEN 1000:REM *MADE
 IT!*
310 LOCATE X,Y,Z
320 POSITION OX,OY:? " "
330 IF Z=ASC("$") THEN 2000:REM *CRASH*
340 IF STRIG(0)=0 AND BM=0 THEN GOSUB 30
00:REM * BUTTON ROUTINE *
350 IF BM=1 THEN GOSUB 4000:REM * MOVE B
```

```
OMB ROUTINE *
360 GOTO 260
1000 POSITION 13,10:? "LANDED !"
1010 POSITION 13,11:? "********"
1020 FOR T=0 TO 6000 STEP 20
1030 SOUND 0,T,10,15:NEXT T
1035 SOUND 0,0,0,0
1040 ? CHR$(125);"WELL DONE , YOUR SCORE
 WAS ";SC
1050 END
2000 POSITION X-1,Y-1
2010 ? CHR$(7);CHR$(124);CHR$(6)
2020 POSITION X-1,Y
2030 ? "-*-"
2040 POSITION X-1,Y+1
2050 ? CHR$(6);CHR$(124);CHR$(7)
2060 FOR P=200 TO 10 STEP -10
2070 FOR I=P-5 TO P+5
2080 SOUND 0,I,120,15
2090 NEXT I
2100 C=INT(RND(0)*15)
2110 SETCOLOR 2,C,2:SETCOLOR 4,C,2
2120 NEXT P
2130 SOUND 0,0,0,0
2140 POSITION 6,10:? "Another try ?"
2150 GET #1,A
2160 IF CHR$(A)="Y" THEN RUN
2170 IF CHR$(A)<>"N" THEN 2150
2180 GRAPHICS 0:END
3000 BM=1:BX=X:BY=Y+1:K=0:RETURN
4000 POSITION BX,BY:? " "
4010 BY=BY+1:LOCATE BX,BY,Z
4020 POSITION BX,BY
4030 ? CHR$(39):REM *BOMB CHARACTER*
4040 IF Z=ASC("$") THEN 5000
4050 IF BY=22 THEN POSITION BX,22:? " ":
BM=0
4060 RETURN
5000 POSITION BX,BY:? "&":SOUND 0,230,12
0,15:SC=SC+50
5010 POSITION 2,0:? "SCORE:";SC
```

```
5020 K=K+1:IF K=5 THEN BM=0:POSITION BX,
BY:? " "
5030 POSITION BX,BY:? " "
5040 SOUND 0,0,0,0:GOTO 4050
10000 GRAPHICS 0
10010 SETCOLOR 4,9,0:SETCOLOR 2,9,0
10012 ? "CITY BOMB - PLEASE WAIT"
10014 ? "**********************"
10016 POKE 752,1:PRINT
10020 R=PEEK(106)-8
10030 FOR P=0 TO 1023
10040 POKE R*256+P,PEEK(57344+P)
10050 NEXT P
10060 READ I
10070 IF I<0 THEN POKE 203,R:RETURN
10080 FOR P=0 TO 7
10090 READ K
10100 POKE R*256+(8*I)+P,K
10110 NEXT P
10120 GOTO 10060
10130 DATA 4,255,129,129,129,129,129,129
,255
10140 DATA 5,16,8,132,194,255,2,4,8
10150 DATA 6,170,85,170,85,170,85,170,85
10160 DATA 7,0,36,60,24,24,60,24,0
10170 DATA -99
```



ENGULF

(Tastiera)

State guidando la navicella spaziale Battlestar e dovete sconfiggere un alieno distruggendo tutti i settori di spazio che lo circondano. Né voi né l'alieno (raffigurato con la lettera A) potete spostarvi nella parte esterna della zona di spazio visibile (queste aree dall'accesso proibito sono raffigurate come '-'). Ad ogni mossa l'alieno può spostarsi di uno, due o nessuno spazio e voi dovrete inserire la vostra mossa sotto forma di due coordinate, premendo il "RETURN" dopo ognuna di esse. Ogni area distrutta viene indicata con uno spazio vuoto, mentre l'area non colpita viene indicata con degli asterischi.
Sconfiggerete l'alieno distruggendo completamente tutti gli spazi dove potrebbe spostarsi. Non potete atterrare sull'alieno: se lo fate vi autoannienterete. L'alieno tiene sotto controllo lo spazio circostante e segnala al computer il grado di pericolo che rileva, cosicché voi sapete, più o meno, ciò che pensa l'alieno. La strategia migliore consiste nell'intrappolare l'alieno contro uno dei lati, poiché in questa maniera se ne limiteranno le possibili mosse. L'alieno è a conoscenza di questa possibilità e, usando l'"intelligenza" nelle istruzioni 6145 e 6161, tenderà ad allontanarsi dai lati in modo da ridurre la possibilità di restare intrappolato. Sapendo cosa può fare l'alieno fareste bene a costruirgli intorno un «recinto» di aree vuote, tentando di mantenerlo in quelle o obbligandolo a spostarsi verso i lati. Questo gioco prevede un "highest score" (massimo punteggio).

```
1 GRAPHICS 0:SETCOLOR 4,8,0:SETCOLOR 2,8
,0
3 DIM A(10,10)
5 HISCORE=0
10 REM ENGULF
20 REM (C)HARTNELL 1982
30 GOSUB 9000:REM INITIALISE
40 GOSUB 8000:REM PRINT OUT
```

```
50 GOSUB 7000:REM ACCEPT PLAYER DECISION
60 GOSUB 6000:REM UPDATE ALIEN
70 TIME=TIME-1
75 SHOTS=SHOTS+1
80 IF TIME=0 THEN 6570
85 GOSUB 8000
90 GOTO 50
910 TIME=30
5000 REM COLLISION
5010 PRINT "YOU HIT THE ALIEN,CAPTAIN"
5020 PRINT "AND HAVE BEEN DESTROYED"
5030 GOTO 6570
6000 REM UPDATE ALIEN
6010 REM CHECK IF SURROUNDED
6020 H=0
6030 IF A(M-1,N)=2 THEN H=H+1
6040 IF A(M-1,N-1)=2 THEN H=H+1
6050 IF A(M,N-1)=2 THEN H=H+1
6060 IF A(M,N+1)=2 THEN H=H+1
6070 IF A(M-1,N+1)=2 THEN H=H+1
6080 IF A(M+1,N+1)=2 THEN H=H+1
6090 IF A(M+1,N-1)=2 THEN H=H+1
6100 IF A(M+1,N)=2 THEN H=H+1
6110 IF H=8 THEN 6500:REM SURROUNDED
6120 REM MOVE ALIEN
6125 E=M:F=N
6130 M=M-INT(RND(1)*2)+INT(RND(1)*2)
6140 IF M<2 OR M>9 THEN 6130
6145 IF (M<4 OR M>7) AND RND(1)>0.7 THEN
 6130
6150 N=N-INT(RND(1)*2)+INT(RND(1)*2)
6160 IF N<2 OR N>9 THEN 6150
6161 IF (N<4 OR N>7) AND RND(1)>0.7 THEN
 6150
6162 IF A(M,N)=2 THEN 6130
6165 A(E,F)=0
6170 A(M,N)=1
6300 RETURN
6500 REM SURROUND
6505 GOSUB 8000
```

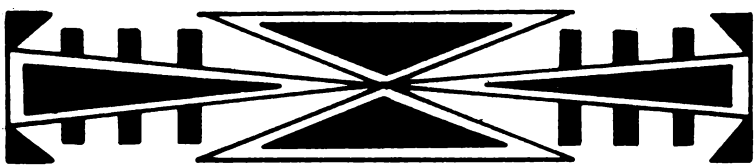
```
6510 PRINT "ENGULFED! WELL DONE"
6520 PRINT "IT TOOK YOU ";SHOTS;" SHOTS"
6530 PRINT "AND YOU DID IT WITH ";TIME;"
 TIME UNITS","LEFT"
6540 Q=TIME*125.67
6550 PRINT "YOUR RATING IS ";Q
6560 IF Q>HISCORE THEN HISCORE=Q
6570 PRINT "HIGHEST SCORE SO FAR IS ";HI
SCORE
6580 PRINT
6590 PRINT "ENTER 1 FOR ANOTHER GAME,2 T
O END"
6600 INPUT A
6610 IF A=1 THEN 10
6620 PRINT "OVER AND OUT,CAPTAIN"
6630 END
7000 REM ACCEPT PLAYER DECISION
7010 PRINT "WHICH SECTOR WILL YOU SHOOT
AT"
7020 PRINT "ACROSS";
7030 TRAP 7030:INPUT S:TRAP 40000
7035 IF S<2 OR S>9 THEN 7030
7040 PRINT S;" AND DOWN";
7050 TRAP 7050:INPUT R:TRAP 40000
7055 IF R<2 OR R>9 THEN 7050
7060 IF A(R,S)=1 THEN 5000:REM HIT ALIEN
,END OF GAME
7070 IF A(R,S)=2 THEN PRINT "SECTOR ALRE
ADY DESTROYED":FOR P=200 TO 255:SOUND 0,
P,10,15:NEXT P:SOUND 0,0,0,0
7090 IF A(R,S)=2 THEN RETURN
7100 A(R,S)=2
7110 RETURN
8000 REM PRINT OUT
8005 PRINT CHR$(125)
8020 PRINT
8030 PRINT "----ALIEN SENSES DANGER OF F
ACTOR-";H;"---"
8050 PRINT "BATTLESTAR COMPUTER REPORTS:
",,"HIGHEST SCORE IS ";HISCORE
8060 PRINT "----------------------------
---------"
```

```
8080 PRINT
8090 PRINT "   ALIEN NOW AT ";N;" ";M
8100 PRINT
8110 PRINT "TIME LEFT: ";TIME,"SHOTS FIR
ED: ";SHOTS
8120 PRINT
8125 PRINT "12345678910"
8130 FOR K=1 TO 10
8140 FOR J=1 TO 10
8145 IF K<2 OR K>9 OR J<2 OR J>9 THEN PR
INT "-";
8146 IF K<2 OR K>9 OR J<2 OR J>9 THEN 81
80
8150 IF A(K,J)=0 THEN PRINT "*";:SOUND 0
,K%8+J%4,10,15
8160 IF A(K,J)=1 THEN PRINT "A";:FOR P=2
0 TO 0 STEP -1:SOUND 0,P,12,15:NEXT P
8170 IF A(K,J)=2 THEN PRINT " ";:FOR P=4
0 TO 0 STEP -2:SOUND 0,P,8,15:NEXT P
8180 SOUND 0,0,0,0:NEXT J
8190 PRINT K
8200 NEXT K
8210 PRINT
8990 RETURN
9000 REM INITIALISE
9010 TIME=30
9020 SHOTS=0
9030 H=0
9050 FOR B=1 TO 10
9055 FOR C=1 TO 10
9060 A(B,C)=0
9065 IF B<2 OR B>9 OR C<2 OR C>9 THEN A(
B,C)=2
9067 NEXT C
9070 NEXT B
9080 M=INT(RND(1)*7)+2
9090 N=INT(RND(1)*7)+2
9100 A(M,N)=1
9900 RETURN
```

COLOUR PATTERN

Questo breve programma presenta figure e creazioni che producono un effetto ipnotizzante con la rotazione continua dei colori.

```
10 GRAPHICS 10
20 POKE 704,0
30 POKE 705,12
40 POKE 706,3%16+4
50 POKE 707,12
55 POKE 708,12
60 FOR X=0 TO 79
70 C=C+1:IF C=5 THEN C=1
80 COLOR C
90 PLOT X,X
100 DRAWTO 79-X,X
110 DRAWTO 79-X,191-X
120 DRAWTO X,191-X
130 DRAWTO X,X
140 NEXT X
145 REM ** ROTATE COLOURS **
150 A=PEEK(705)
160 POKE 705,PEEK(706)
170 POKE 706,PEEK(707)
185 POKE 707,PEEK(708)
186 POKE 708,A
190 A=SIN(A):REM ** DELAY **
200 GOTO 150
```



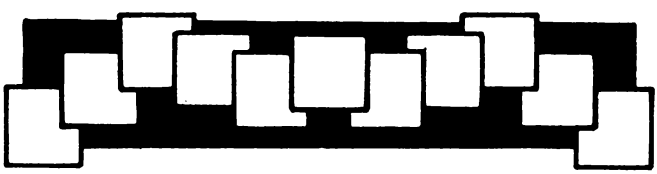
COLOUR PUZZLE

(Tastiera)

La soluzione di questo meraviglioso puzzle dovrebbe rivelarsi una sfida per voi. Ecco le regole:
Le prime quattro mosse devono essere eseguite nei quattro angoli (01, 10, 16 e 25);
Per collocare un quadrato colorato sulla scacchiera è necessario che sia vicino ad un altro quadrato in orizzontale o in verticale, non in diagonale;
Un quadrato colorato non può essere collocato vicino ad un altro quadrato del medesimo colore che si trovi in uno degli otto quadrati che lo circondano. Disponete i quadrati colorati secondo queste regole finchè tutti i 25 spazi saranno riempiti.
I riquadri sono numerati nel modo seguente:

			10			
		05	11	17		
	02	06	12	18	22	
01	03	07	13	19	23	25
	04	08	14	20	24	
		09	15	21		
			16			

Notate che è necessario porre uno zero prima delle cifre singole e che non bisogna premere il tasto RETURN. Se tentate di compiere una mossa che non è permessa il computer vi avvertirà suonando. Quando non riuscite più a muovere, battete '99'.



```
10 REM ** COLOUR PUZZLE -        **
20 REM **          BY PAUL BUNN   **
25 REM ****************************
30 GRAPHICS 10
40 POKE 704,12
50 POKE 705,3%16+4
60 POKE 706,12%16+6
70 POKE 707,7%16+2
80 POKE 708,15%16+4
90 POKE 709,0:POKE 710,15%16+10
100 COLOR 6:POKE 703,4
110 FOR X=0 TO 48 STEP 8
120 READ Y
130 FOR X2=0 TO 7
140 PLOT X+X2+12,90-Y
150 DRAWTO X+X2+12,90+Y
160 NEXT X2
170 NEXT X
180 DATA 10,30,50,70,50,30,10
190 DIM X(25),Y(25)
200 FOR X2=1 TO 25
210 READ X,Y:X(X2)=X:Y(X2)=Y
220 NEXT X2
230 DATA 12,80,20,60,20,80,20,100
240 DATA 28,40,28,60,28,80,28,100
250 DATA 28,120,36,20,36,40,36,60
260 DATA 36,80,36,100,36,120,36,140
270 DATA 44,40,44,60,44,80,44,100
280 DATA 44,120,52,60,52,80,52,100
290 DATA 60,80
300 FOR P=1 TO 25:COLOR 5:PLOT X(P)+3,Y(
P)+10:NEXT P
310 OPEN #1,4,0,"K:"
311 COL=INT(RND(0)%5)+1:COLOR COL
312 B=B+(COL=3):R=R+(COL=1):G=G+(COL=2):
YL=YL+(COL=4):BL=BL+(COL=5)
313 IF COL=1 AND R>5 THEN R=5:GOTO 311
314 IF COL=2 AND G>5 THEN G=5:GOTO 311
315 IF COL=3 AND B>5 THEN B=5:GOTO 311
316 IF COL=4 AND YL>5 THEN YL=5:GOTO 311
```

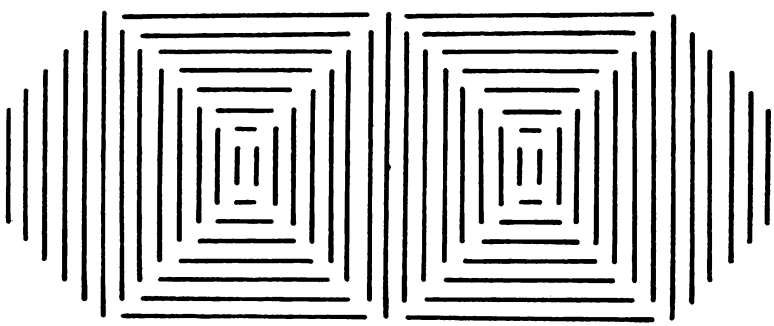
```
317 IF COL=5 AND BL>5 THEN BL=5:GOTO 311
319 FOR P=0 TO 10:PLOT P,0:DRAWTO P,20:N
EXT P
320 GET #1,KEY:KEY=KEY-48:KEY=KEY*10
325 GET #1,KEY2:KEY=KEY+KEY2-48
326 IF KEY=99 THEN 2000
327 IF KEY<1 OR KEY>25 THEN PRINT CHR$(2
53):GOTO 320
330 LOCATE X(KEY),Y(KEY),C
340 IF C<>6 THEN PRINT CHR$(253):GOTO 32
0
350 IF KEY<>1 AND KEY<>10 AND KEY<>16 AN
D KEY<>25 AND K<4 THEN PRINT CHR$(253):G
OTO 320
355 IF J=1 THEN K=5
360 IF KEY=1 OR KEY=10 OR KEY=16 OR KEY=
25 THEN K=K+1:IF K=4 THEN J=1
370 X=X(KEY)+3:Y=Y(KEY)+10
380 LOCATE X-6,Y,C1
390 LOCATE X-6,Y-17,C2
400 LOCATE X,Y-17,C3
410 LOCATE X+6,Y-17,C4
420 LOCATE X+6,Y,C5
430 LOCATE X+6,Y+17,C6
440 LOCATE X,Y+17,C7
450 LOCATE X-6,Y+17,C8
460 IF C1=COL OR C2=COL OR C3=COL OR C4=
COL OR C5=COL OR C6=COL OR C7=COL OR C8=
COL THEN ? CHR$(253):GOTO 320
470 U=(C1=6)+(C3=6)+(C5=6)+(C7=6)
473 I=(C1=0)+(C3=0)+(C5=0)+(C7=0)
475 IF U+I=4 AND K>4 THEN PRINT CHR$(253
):GOTO 320
480 COLOR COL
485 X=X(KEY):Y=Y(KEY)
490 FOR P=0 TO 7
500 PLOT X+P,Y
510 DRAWTO X+P,Y+19
520 NEXT P
525 IF R+G+B+YL+BL=25 THEN 1000
530 NO=NO+1:GOTO 311
```

```
1000 FOR G=205 TO 5 STEP -5
1010 FOR P=G-3 TO G+3
1020 SOUND 0,P,10,15
1030 NEXT P
1040 POKE 704,(INT(RND(0)*15)*16)+8
1050 NEXT G
1060 GRAPHICS 0:SETCOLOR 2,9,2:SETCOLOR
4,9,2
1070 POSITION 4,12
1080 ? "WELL DONE YOU SOLVED THE PUZZLE"
1090 POSITION 4,13
1100 POKE 752,1
1110 ? "**********************************"
1130 FOR P=0 TO 255
1140 SOUND 0,P,10,15
1150 NEXT P
1160 SOUND 0,0,0,0
1170 END
2000 FOR G=0 TO 256
2010 SOUND 0,G,10,15
2020 POKE 704,G/2
2030 NEXT G
2040 SOUND 0,0,0,0:POKE 704,12
2050 GRAPHICS 0:POKE 752,1
2055 POSITION 10,12:POKE 710,3*16+4
2060 ? "BAD LUCK - TRY AGAIN"
2065 POSITION 10,13
2070 ? "********************"
2080 FOR P=0 TO 2000 STEP 20
2090 SOUND 0,P,10,15:NEXT P
2100 SOUND 0,0,0,0
2110 ? :?
2120 ? "YOU HAD ";25-NO;" LEFT TO FILL"
2130 END
```

STRING PATTERN

Questo breve e semplice programma produce disegni e suoni stupendi.

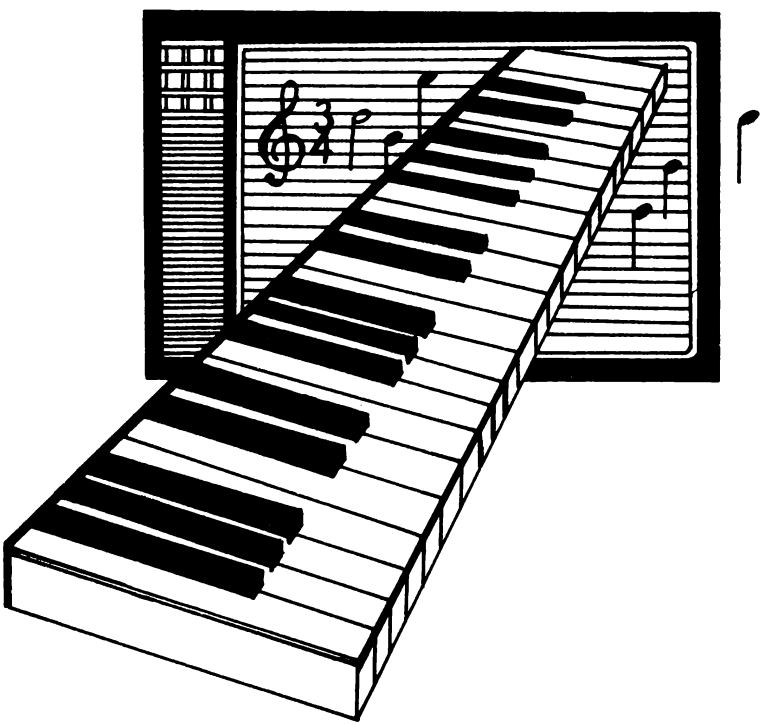
```
10 REM ** STRING PATTERN WRITTEN **
20 REM **    BY PAUL BUNN 1983   **
30 REM ****************************
40 GRAPHICS 8+16:SETCOLOR 2,0,0
45 COLOR 1:SETCOLOR 1,0,14
50 FOR X=0 TO 191 STEP 10
60 PLOT 64,191-X
70 DRAWTO X+64,0
80 PLOT 191-X+64,0
90 DRAWTO 191+64,191-X
100 PLOT 64,X
110 DRAWTO X+64,191
120 PLOT 191+64,X
130 DRAWTO 191-X+64,191
140 NEXT X
145 FOR P=0 TO 14 STEP 0.2
150 SETCOLOR 1,0,P
155 SOUND 0,P%40,10,15
160 NEXT P:GOTO 145
```



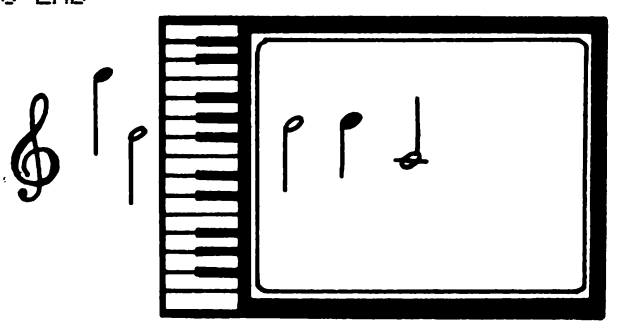
SOUND PROGRAM

(Tastiera)

Questo breve ed interessante programma produrrà suoni simili a quelli emessi da un pianoforte; memorizzerà inoltre le note e, premendo il tasto 'ESC', tornerà a suonarle.



```
10 REM **  SOUND PROGRAM  **
20 REM **        BY        **
30 REM **    PAUL BUNN     **
40 REM **********************
50 GRAPHICS 0
60 SETCOLOR 2,8,0:SETCOLOR 4,8,0
70 TRAP 160
90 DIM T(500)
100 OPEN #1,4,0,"K:"
105 ? CHR$(125):? "PRESS KEYS TO GET DIF
FERENT"
106 ? "NOTES.TO HEAR PLAYED BACK PRESS '
ESC'"
110 GET #1,A:L=L+1
115 IF A=27 THEN 160
120 T(L)=A
130 FOR X=15 TO 0 STEP -1
140 SOUND 0,A,10,X:NEXT X
150 GOTO 110
160 ? CHR$(125)
170 ? "WHAT TEMPO (1-4)"
180 GET #1,A
190 A=A-48:IF A<1 OR A>4 THEN 180
200 FOR J=1 TO L
210 S=T(J)
220 FOR X=15 TO 0 STEP -1
230 SOUND 0,S,10,X:NEXT X
235 FOR G=1 TO 10*(A-1):NEXT G
240 NEXT J
250 END
```



DISPLAY LIST INTERRUPT EXAMPLE

Questo programma mostra soltanto una piccola parte di ciò che una DLI è capace di fare. Vi suggerisco di registrare il programma prima di tentare di eseguirlo, poiché è scritto interamente nel linguaggio macchina.

```
10 REM ******************************
20 REM ** Display List Interrupt  **
30 REM **         example          **
40 REM **  Written by Paul Bunn    **
50 REM ******************************
60 GOSUB 1000
70 X=USR(1536)
1000 FOR A=1536 TO 1636:READ B:POKE A,B:
NEXT A
1001 DATA 162,16,169,3,157,66,3,169,99,1
57,68,3,169,6,157,69,3,169,12,157,74,3,1
69,8,157
1002 DATA 75,3,32,86,228,173,48,2,133,20
3,173,49,2,133,204,160,0,177,203,201,15,
240,8,200,192
1003 DATA 0,208,245,76,63,6,169,143,145,
203,76,48,6,169,81,141,0,2,169,6,141,1,2
,169,192
1004 DATA 141,14,212,76,78,6,72,173,11,2
12,42,73,255,141,10,212,141,24,208,141,2
6,208,104,64,83
1005 DATA 58
1006 RETURN
```

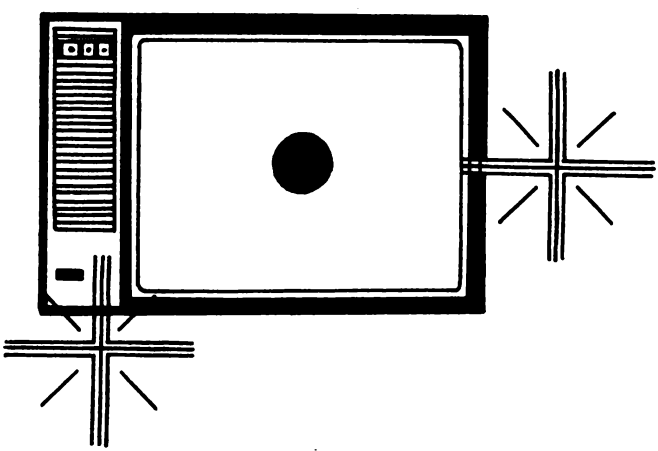
REACTION TIMER

(Due joystick)

Questo è un breve programma per due giocatori ed è molto semplice da giocare. Appena vedete apparire un punto sullo schermo premete il tasto "FIRE". Il giocatore che premerà il tasto per primo guadagnerà un punto. Il primo giocatore che arriva a 10 punti vince il gioco. Questo programma mette alla prova la prontezza di riflessi.

```
10 REM %% REACTION TIMER            %%
20 REM %% BY P.BUNN , NOVEMBER 1981%%
30 REM
31 DIM A$(30),B$(30)
32 ? "PLAYER 1'S NAME ";:INPUT A$
33 ? "PLAYER 2'S NAME ";:INPUT B$
40 GRAPHICS 5
45 SETCOLOR 2,3,4
50 A=INT(RND(0)*200)+200
60 FOR L=1 TO A
70 IF STRIG(0)=0 THEN GOTO 1000
80 IF STRIG(1)=0 THEN GOTO 2000
90 NEXT L
100 COLOR INT(RND(0)*3)+1
110 A=RND(0)*79:B=RND(0)*39
120 PLOT A,B
130 A=0
140 A=A+1
150 IF STRIG(0)=0 THEN 3000
160 IF STRIG(1)=0 THEN 4000
170 GOTO 140
1000 GRAPHICS 0:? "CHEAT! YOU PRESSED TH
E BUTTON TO SOON":SC2=SC2+1:GOTO 5000
2000 GRAPHICS 0:? "CHEAT! YOU PRESSED TH
E BUTTON TO SOON":SC1=SC1+1:GOTO 5000
```

```
3000 GRAPHICS 0:? A$;" WINS!....WITH A T
IME OF:-                ";A/100:SC1=SC1+1:IF
 SC1>=10 THEN 6000
3005 GOTO 5000
4000 GRAPHICS 0:? B$;" WINS!....WITH A T
IME OF:-                ";A/100:SC2=SC2+1:I
F SC2>=10 THEN 7000
4005 GOTO 5000
5000 ? :? :? :?
5005 ? "            ======"
5010 ? "            SCORES"
5020 ? "            ======"
5030 PRINT :PRINT
5040 ? A$;"                              ";B$
5050 ? "--------------------------------
-----"
5060 ?
5070 ? SC1;"
";SC2
5080 ? :? :? "PRESS 'FIRE BUTTON' WHEN R
EADY"
5090 IF STRIG(0)<>0 THEN 5090
5100 FOR L=1 TO 100:NEXT L:GOTO 40
6000 ? :? :? :? A$;" WINS THE GAME !!.."
6005 ? :? :END
7000 ? :? :? :? B$;" WINS THE GAME !!.."
7005 ? :? :END
```

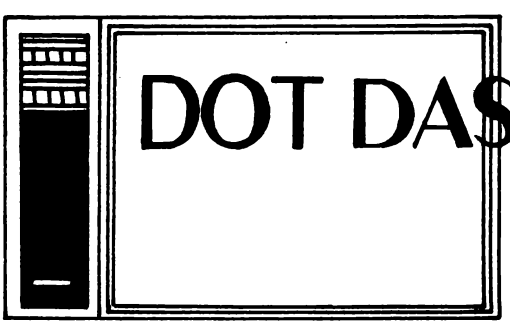


MORSE CODE

Quando il computer ve lo chiede, battete una frase qualsiasi e il programma la tradurrà in codice Morse attraverso l'altoparlante della televisione.

```
10 REM ******************************
20 REM * MORSE CODE CONVERTER BY  *
30 REM *     P.BUNN & J.VINCENT    *
40 REM ******************************
50 GRAPHICS 0
60 SETCOLOR 2,8,0:SETCOLOR 4,8,0
70 DL=PEEK(560)+256*PEEK(561)+4
80 POKE DL+2,7:POKE DL+3,7
90 POSITION 0,1
100 ? "MORSE CODE CONVERTER";
110 ? "**!*****************"
120 ? :? :? "TYPE IN MESSAGE YOU WANT TO
 HEAR :-"
130 DIM A$(100),B$(5)
140 SOUND 0,0,0,0:TRAP 140:INPUT A$
150 FOR X=1 TO LEN(A$)
155 IF A$(X,X)=" " THEN FOR P=1 TO 100:N
EXT P:NEXT X
160 Y=ASC(A$(X,X))-65
170 RESTORE Y+1000
180 B$="      ":READ B$
190 FOR Z=1 TO LEN(B$)
200 IF B$(Z,Z)="." THEN SOUND 0,30,10,15
:FOR P=1 TO 20:NEXT P:SOUND 0,0,0,0
210 IF B$(Z,Z)="-" THEN SOUND 0,30,10,15
:FOR P=1 TO 50:NEXT P:SOUND 0,0,0,0
220 FOR P=1 TO 20:NEXT P
230 NEXT Z
240 NEXT X
250 RUN
1000 DATA .-
1001 DATA -...
```

```
1002 DATA -.-.
1003 DATA -..
1004 DATA .
1005 DATA ..-.
1006 DATA --.
1007 DATA ....
1008 DATA ..
1009 DATA .---
1010 DATA -.-
1011 DATA .-..
1012 DATA --
1013 DATA -.
1014 DATA ---
1015 DATA .--.
1016 DATA --.-
1017 DATA .-.
1018 DATA ...
1019 DATA -
1020 DATA ..-
1021 DATA ...-
1022 DATA .--
1023 DATA -..-
1024 DATA -.--
1025 DATA --..
```



COMPLIMENT GENERATOR

Vi sentite depressi? Basterà eseguire questo programma e la vostra autostima aumenterà immediatamente.

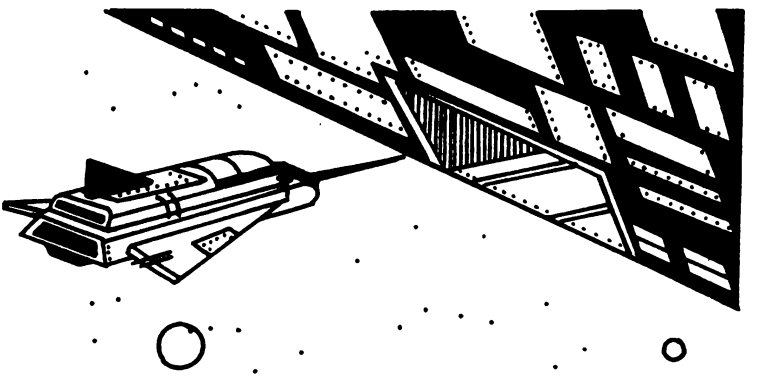
```
10 REM ** COMPLIMENT GENERATOR **
20 DIM A$(15),B$(15)
30 A=INT(RND(1)*10)
40 B=INT(RND(1)*7)
50 RESTORE 1000+A
60 READ A$
70 RESTORE 2000+B
80 READ B$
90 ? A$;" ";B$
95 A=INT(RND(0)*16):SETCOLOR 2,A,4:SETCO
LOR 4,A,4
96 FOR F=1 TO 300:NEXT F
100 RUN
1000 DATA BRILLIANT
1001 DATA LOVELY
1002 DATA GREAT
1003 DATA CLEVER
1004 DATA INTELLIGENT
1005 DATA NICE
1006 DATA WISE
1007 DATA BEAUTIFUL
1008 DATA CHARMING
1009 DATA SMART
2000 DATA PERSON
2001 DATA CHARMER
2002 DATA GENIUS
2003 DATA INDIVIDUAL
2004 DATA OPERATOR
2005 DATA HERO
2006 DATA ACE
```

SPACE DOCKER

(joystick)

Questo è uno dei migliori programmi contenuti in questo libro. È progettato per quattro giocatori e dispone di routine in linguaggio macchina per i movimenti. Il BASIC non può essere usato per le azioni rapide poiché è troppo lento. Combattendo contro invasori stranieri in una galassia lontana, la vostra nave viene improvvisamente colpita in pieno. Entrate nella piccola navetta di salvataggio con il vostro equipaggio, vi staccate dalla nave e correte verso la nave appoggio dove la salvezza è sicura. Ma c'è un problema: state esaurendo rapidamente la scorta di ossigeno e ve ne restano soltanto 300 unità.

Il vostro obiettivo, usando il joystick nel "Port 1", è di attraccare alla nave appoggio prima che la scorta si esaurisca completamente. Per avanzare usate il pulsante del joystick, muovendone la barra a destra o a sinistra per guidare la navetta verso la nave appoggio. Quando riuscite ad attraccare alla nave appoggio, il numero delle unità di aria avanzate verrà aggiunto ai vostri punti. Chi è interessato alla routine del linguaggio macchina che ho impiegato può trovare il listato in assembly dopo il listato del BASIC.



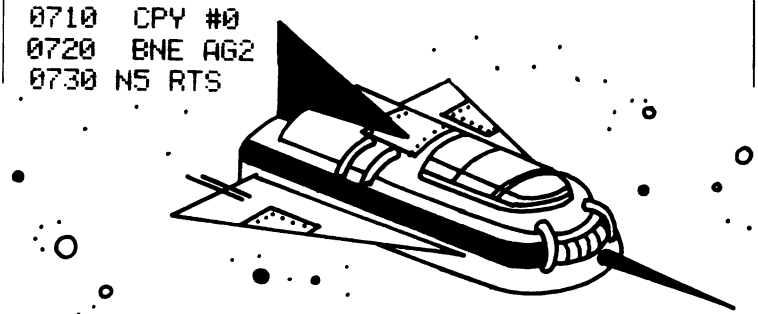
```
10 REM ******************************
20 REM * SPACE DOCKER BY P.BUNN &  *
30 REM *       JONATHAN VINCENT     *
40 REM ******************************
50 GOSUB 10000:REM ** INITIALIZATION **
60 GOSUB 20000
65 SOUND 0,0,8,15:POKE 53278,R:AIR=300
70 X=USR(1536)
80 IF PEEK(781+R*256)=24 THEN 1000
90 IF PEEK(53254)<>0 OR PEEK(53262)<>0 T
HEN GOTO 2000
100 AIR=AIR-1:IF AIR<>0 THEN 70
110 POSITION 4,6:? "out of air":SC=0
115 POKE 53251,0
120 FOR P=0 TO 4000 STEP 12
130 SOUND 0,P,10,15:NEXT P
140 SOUND 0,0,0,0
150 POSITION 4,6:? "press START"
155 POSITION 4,4:? "score:";SC
160 POKE 53279,0
170 IF PEEK(53279)<>6 THEN 160
180 POSITION 4,6:? "            ":GOSUB 1
0100:GOTO 65
1000 IF PEEK(209)>PEEK(203) AND PEEK(209
)<PEEK(204) THEN 1020
1010 GOTO 2000
1020 POSITION 4,6:? "docked"
1025 POKE 53251,0
1030 FOR P=230 TO 10 STEP -10
1040 FOR X=P-5 TO P+5
1050 SOUND 0,X,10,15
1060 NEXT X
1070 SETCOLOR 1,RND(0)*16,10
1080 NEXT P
1090 SOUND 0,0,0,0
1100 POSITION 21,4:? "remaining air:";AI
R
1110 SC=SC+AIR:GOTO 150
2000 FOR P=781 TO 896:IF PEEK(R*256+P)<>
24 THEN NEXT P
2005 POKE 53251,0
```

```
2010 E=53770:POKE P+256*R+RND(0)*8,PEEK(
E)
2020 SOUND 0,PEEK(E),8,15:POKE 706,PEEK(
E)
2030 S=S+1:IF S<50 THEN 2010
2035 POKE 53250,0:POKE 53251,0
2040 S=0:SC=S:GOTO 140
10000 GRAPHICS 0:DL=PEEK(560)+256*PEEK(5
61)+4:POKE 82,0
10010 POKE DL+5,7:POKE DL+6,6:POKE DL+7,
6:POKE DL+8,6:POKE DL+9,6
10020 SETCOLOR 4,9,4:SETCOLOR 1,3,4:POKE
 DL+10,6
10030 SETCOLOR 0,3,8:SETCOLOR 1,13,10
10040 POSITION 0,4:POKE 752,1
10050 ? "    SPACE DOCKER    ";
10055 ? "    ***********    ";
10060 ? " you are losing air ";
10070 ? " and MUST dock with ";
10080 ? " the passing mother ";
10090 ? " ship....GOOD LUCK! ";
10100 R=PEEK(106)-8:RESTORE
10110 FOR P=R*256+512 TO R*256+1024
10120 POKE P,0:NEXT P
10130 FOR P=10 TO 17
10140 READ A,B
10150 POKE R*256+512+P,A
10160 POKE R*256+640+P,B
10170 NEXT P
10180 DATA 7,224,62,124,111,246
10190 DATA 255,255,214,107
10200 DATA 124,62,60,60,4,32
10210 FOR P=0 TO 7
10220 READ A
10230 POKE R*256+768+80+P,A
10240 NEXT P
10250 DATA 24,60,36,60,36,126,90,255
10260 FOR P=0 TO 6
10270 READ A
10280 POKE R*256+896+88+P,A
10290 NEXT P
```

```
10300 ? CHR$(125):COLOR 160:PLOT 0,20:DR
AWTO 39,20:? CHR$(160):POSITION 0,0
10310 POKE 704,8*16
10320 POKE 705,8*16
10330 POKE 706,15*16+10
10340 POKE 707,3*16+4
10350 POKE 53277,3
10360 POKE 559,46
10370 POKE 53248,110
10380 POKE 53249,126
10390 POKE 53250,108
10400 POKE 53256,1
10410 POKE 53257,1
10420 POKE 53258,1
10430 POKE 53259,1
10440 POKE 623,1
10450 POKE 54279,R
10460 DATA 24,60,126,255,255,0,0
10470 POKE 203,110:POKE 204,126
10480 POKE 208,INT(RND(1)*2)+1:POKE 209,
108
10485 K=R*256+768:POKE 206,INT(K/256):PO
KE 205,K-(256*PEEK(206))
10490 RETURN
20000 FOR A=1536 TO 1676:READ B:POKE A,B
:NEXT A
20001 DATA 104,165,208,201,1,240,17,230,
203,230,204,165,204,201,200,208,21,169,1
,133,208,76,38,6,198
20002 DATA 203,198,204,165,203,201,50,20
8,4,169,2,133,208,165,203,141,0,208,165,
204,141,1,208,173,120
20003 DATA 2,201,11,240,10,173,120,2,201
,7,240,8,76,72,6,198,209,76,72,6,230,209
,165,209,141
20004 DATA 2,208,173,132,2,201,0,240,11,
169,0,141,3,208,141,0,210,76,127,6,173,1
0,210,141,195
20005 DATA 2,165,209,141,3,208,169,130,1
41,0,210,160,0,177,205,136,145,205,200,2
00,192,0,208,245,76
20006 DATA 140,6,160,0,177,205,200,145,2
05,136,136,192,0,208,245,96
```

```
20007 RETURN
10  *=$600
20 ;ASSEMBLY ROUTINE USED IN
30 ;SPACE DOCKER WRITTEN BY PAUL BUNN
40 HOR1=$CB
50 HOR2=$CC
60 LF=$D0
70 HOR3=$D1
80  PLA
90  LDA LF
0100  CMP #1
0110  BEQ RIGHT
0120 LEFT  INC HOR1
0130  INC HOR2
0140  LDA HOR2
0150  CMP #200
0160  BNE N1
0170  LDA #1
0180  STA LF
0190  JMP N1
0200 RIGHT DEC HOR1
0210  DEC HOR2
0220  LDA HOR1
0230  CMP #50
0240  BNE N1
0250  LDA #2
0260  STA LF
0270 N1 LDA HOR1
0280  STA 53248
0290  LDA HOR2
0300  STA 53249
0310 STICK LDA $278
0320  CMP #11
0330  BEQ L
0340  LDA $278
0350  CMP #7
0360  BEQ R
0370  JMP N2
0380 L DEC HOR3
0390  JMP N2
0400 R INC HOR3
```

```
0410 N2 LDA HOR3
0420  STA 53250
0430  LDA $284
0440  CMP #0
0450  BEQ ON
0460  LDA #0
0470  STA 53251
0480  STA $D200
0490  JMP N4
0500 ON LDA 53770
0510  STA 707
0520  LDA HOR3
0530  STA 53251
0540  LDA #130
0550  STA $D200
0560  LDY #0
0570 AG1 LDA ($CD),Y
0580  DEY
0590  STA ($CD),Y
0600  INY
0610  INY
0620  CPY #0
0630  BNE AG1
0640  JMP N5
0650 N4 LDY #0
0660 AG2 LDA ($CD),Y
0670  INY
0680  STA ($CD),Y
0690  DEY
0700  DEY
0710  CPY #0
0720  BNE AG2
0730 N5 RTS
```



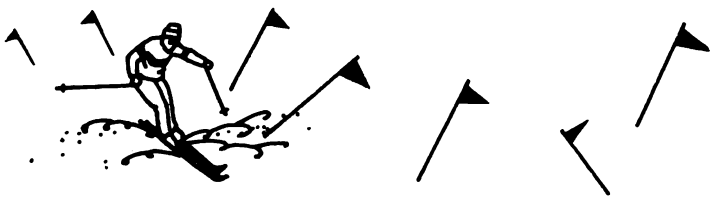
SKI RUN

(joystick)

Note per introdurre il programma:

1. Battete esattamente quello che riporta il listato
2. Registrate il programma su un nastro o un disco prima di eseguirlo.
3. Il programma è designato per un Atari 400/800 dotato di almeno 16K di memoria, con o senza il DOS caricato in macchina.
4. Se avete un computer da 16K, quando eseguirete il programma sullo schermo compariranno strani colori: la cosa è normale e avrà termine dopo poco tempo.
5. Quando state per giocare a Ski Run, spegnete e riaccendete il computer, caricate il programma e quindi eseguitelo. Dopo il comando di inizio il programma richiederà alcuni secondi per essere eseguito. Per cominciare a giocare premete START.

Se seguite le regole appena menzionate, vi ritroverete con un ottimo gioco scritto interamente in codice macchina, qualità caratteristica di un software molto costoso.
Per spostare il vostro sciatore a destra o a sinistra usate il joystick, inserito nel Port 1. Per guadagnare punti dovete farlo sciare lungo le bandierine blu, evitando accuratamente gli alberi. Se siete tanto bravi da terminare un percorso potrete impegnarvi nel livello successivo: vi sposterete, cioè, su una montagna più alta con discese più ripide ed il vostro sciatore si muoverà più rapidamente. Il gioco ha termine quando il vostro sciatore va a sbattere contro un albero. Premete START per sollevarlo da questa frustrante situazione.



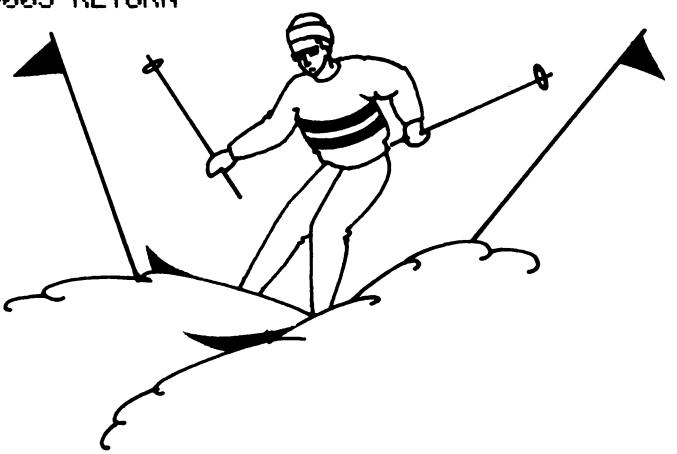
```
10 REM ** SKI RUN BY PAUL DUNNING **
20 REM
30 REM Make sure you SAVE this program
40 REM before you attempt to RUN it !
45 REM
50 GOSUB 1000
60 GOSUB 2000
70 GOSUB 3000
80 GOSUB 4000
90 X=USR(1663)
1000 FOR A=1536 TO 1646:READ B:POKE A,B:
NEXT A
1001 DATA 112,112,112,70,0,60,7,0,6,6,11
8,217,35,54,54,54,54,54,54,54,54,54,54,5
4,54
1002 DATA 54,54,54,54,54,54,54,54,65,0,6
,0,24,24,24,24,60,60,60,60,60,60,126,126
,126
1003 DATA 126,126,126,255,255,255,255,25
5,255,255,255,126,126,24,24,24,24,24,0,2
55,255,255,255,255,255
1004 DATA 255,255,192,192,192,192,192,19
2,192,192,3,3,3,3,3,3,3,3,0,0,0,0,0,0,0,0
1005 DATA 0,7,31,127,31,7,1,1,1,0,0
1006 RETURN
2000 FOR A=13568 TO 14636:READ B:POKE A,
B:NEXT A
2001 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
2002 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
2003 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
2004 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
2005 DATA 0,0,0,101,110,100,0,111,102,0,
114,111,117,110,100,0,0,0,0,0,0,0,0,0,0,0
2006 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
2007 DATA 0,0,110,101,120,116,0,114,111,
117,110,100,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
2008 DATA 0,99,111,109,105,110,103,0,0,1
17,112,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

```
2009 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2010 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2011 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2012 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2013 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2014 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2015 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2016 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2017 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2018 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2019 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2020 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2021 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,8,
0,8,0,0,0,0,0,0,0,0
2022 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2023 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2024 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2025 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2026 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2027 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2028 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2029 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
2030 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
```

```
2031 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0
2032 DATA 0,51,43,41,0,50,53,46,0,0,0,0,0,
0,0,0,0,0,0,0,0,51,35,47,50,37,0
2033 DATA 16,16,16,16,16,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0
2034 DATA 0,0,0,0,63,63,63,63,63,63,63,63,6
3,63,63,63,63,63,63,63,63,63,63,63,63,63
2035 DATA 63,63,63,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0
2036 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0
2037 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0
2038 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0
2039 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0
2040 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0
2041 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,36
2042 DATA 36,36,36,36,36,36,36,36,36,36,36,
36,36,36,36,36,0,0,0,0,0,0,0,0,0,0,0
2043 DATA 0,0,0,0,0,0,0,0,0,0,0,129,129,12
9,90,90,90,60,60,60
2044 RETURN
3000 FOR A=14848 TO 15475:READ B:POKE A,
B:NEXT A
3001 DATA 169,175,141,1,210,169,0,141,11
,6,169,32,141,12,6,169,14,141,200,2,169,
200,141,196,2
3002 DATA 169,32,133,213,169,120,141,197
,2,162,6,169,0,157,0,29,202,208,248,160,
0,185,0,224,153
3003 DATA 0,56,185,0,225,153,0,57,185,0,
226,153,0,58,185,0,227,153,0,59,136,208,
229,169,32
3004 DATA 133,208,162,16,160,0,169,0,145
,207,136,208,249,230,208,202,208,242,169
,56,141,244,2,160,88
3005 DATA 185,37,6,153,8,56,136,208,247,
169,0,141,48,2,169,6,141,49,2,166,213,17
3,10,210,133
```

```
3006 DATA 205,173,10,210,41,14,9,32,133,
206,169,1,160,0,145,205,160,24,169,2,145
,205,160,48,169
3007 DATA 3,145,205,160,72,169,4,145,205
,202,208,215,162,32,173,10,210,133,205,1
73,10,210,41,14,9
3008 DATA 32,133,206,160,0,177,205,201,0
,208,234,169,73,145,205,202,208,227,162,
0,169,0,157,0,32
3009 DATA 202,208,248,32,131,9,32,17,10,
160,0,140,5,212,32,32,9,152,72,104,168,1
73,120,2,201
3010 DATA 7,240,26,201,11,240,3,76,14,9,
165,212,201,56,240,29,198,212,165,212,14
1,0,208,141,1
3011 DATA 208,76,14,9,165,212,201,192,24
0,10,230,212,165,212,141,0,208,141,1,208
,173,4,208,201,0
3012 DATA 240,3,32,235,9,200,192,8,208,1
82,76,52,9,152,72,160,24,162,0,202,208,2
53,169,0,141
3013 DATA 0,210,136,208,243,104,168,96,1
73,11,6,24,105,24,141,11,6,173,12,6,105,
0,141,12,6
3014 DATA 173,12,6,201,45,240,3,76,209,8
,169,0,169,0,141,11,6,169,27,141,12,6,17
3,35,9
3015 DATA 201,7,240,3,206,35,9,162,0,138
,72,32,32,9,104,170,202,208,246,141,11,6
,169,32,141
3016 DATA 12,6,165,213,24,105,10,133,213
,76,44,8,169,58,141,192,2,169,0,141,193,
2,169,128,141
3017 DATA 0,208,133,212,169,128,141,1,20
8,169,0,141,2,208,141,3,208,141,4,208,14
1,5,208,141,6
3018 DATA 208,141,7,208,169,3,141,29,208
,169,48,141,7,212,169,62,141,47,2,162,0,
169,0,157,0
3019 DATA 52,157,0,53,202,208,245,162,17
,189,0,31,157,128,53,202,208,247,162,17,
189,32,31,157,128
3020 DATA 52,202,208,247,162,96,189,0,30
,157,255,59,202,208,247,96,173,4,208,201
,2,240,16,173,31
```

```
3021 DATA 208,201,6,208,249,169,255,141,
30,208,76,0,8,72,32,17,10,169,1,141,30,2
08,173,10,210
3022 DATA 141,0,210,96,32,101,10,238,4,2
9,173,4,29,201,10,240,1,96,169,0,141,4,2
9,238,3
3023 DATA 29,173,3,29,201,10,240,1,96,16
9,0,141,3,29,238,2,29,173,2,29,201,10,24
0,1,96
3024 DATA 169,0,141,2,29,238,1,29,173,1,
29,201,10,240,1,96,169,0,141,1,29,238,0,
29,173
3025 DATA 0,29,201,10,240,1,96,169,0,141
,0,29,0,162,5,189,255,28,24,105,16,157,3
0,60,202
3026 DATA 208,244,96
3027 RETURN
4000 FOR A=1663 TO 1738:READ B:POKE A,B:
NEXT A
4001 DATA 104,169,0,133,203,169,53,133,2
04,160,0,132,205,169,27,133,206,177,203,
145,205,200,208,249,230,204
4002 DATA 230,206,165,206,201,32,208,239
,169,0,133,203,133,205,169,58,133,204,16
9,8,133,206,177,203,145
4003 DATA 205,200,208,249,230,204,230,20
6,165,204,201,61,208,239,169,0,133,12,16
9,8,133,13,76,0,8
4005 RETURN
```



Come scrivere programmi migliori

di Tim Hartnell

Questo libro contiene molti, ottimi programmi, come del resto la maggior parte delle riviste specializzate in computer. D'altra parte, per quanto siano buoni i programmi proposti dalle pubblicazioni, è certo che far "girare" un programma parzialmente e completamente scritto da te ti offre un piacere maggiore. Variando i programmi, dai il tuo tocco personale, rifletti i tuoi desideri, la tua creatività. Questo è un modo eccellente per migliorare i programmi e alla fine, naturalmente, diverrai un programmatore migliore e la tua immaginazione sarà potenziata.

I programmi, presentati dai periodici o da libri come questo, sono un ideale trampolino di lancio per i tuoi progressi in questo settore. Scoprirai che perfino la pubblicità sul software può ispirarti delle idee. Devi solo leggere la descrizione dei programmi commercialmente disponibili e farai il primo passo verso la creazione del "tuo" programma. Naturalmente, non ti è consentito di violare la legge sul diritto d'autore per quanto riguarda i dati e le informazioni illustrati sul video, il nome del programma, o i nomi dei "caratteri" all'interno del programma. Probabilmente scoprirai che il programma, a mano a mano che si sviluppa, prenderà vita propria, crescerà e si evolverà staccandosi dallo schema originale finché il tuo gioco risulterà concepito e realizzato in un modo completamente nuovo.

Qualsiasi cosa tu faccia, non far passare mai come tuo il lavoro di un altro. Adatta e migliora i programmi pubblicati con ogni mezzo, ma poi non presentarli ai periodici come se fossero originali. Ho perduto il conto delle volte in cui mi è stata proposta la pubblicazione di un programma che era stato tratto da uno dei miei libri. Ricordati che i libri e i periodici specializzati in giochi e computer sono tutti ottimi spunti per le tue idee. Troverai sulla tua strada sempre nuove idee per giochi, forme di caratteri, suoni, conclusioni a sfondo drammatico, e così via. È opportuno che tu ne prenda nota. In questo modo non ti troverai mai a corto di idee e sarai in grado di unire il materiale per produrre programmi migliori e capaci di tener desta l'attenzione del giocatore più a lungo.

I giochi si dividono in tre categorie. È utile quindi che tu individui la categoria alla quale appartiene il programma che proponi **prima** di iniziarlo, dato che la categoria del gioco altera materialmente il modo di dare inizio alla programmazione. Ciò non significa che a mano a mano che lo sviluppi, il programma non possa passare da una categoria all'altra, né che un particolare gioco non possa svilupparsi tra due categorie, ma è tuttavia utile considerare i vari gruppi separati per avere idee chiare. Le tre categorie sono:

1) Board games (scacchi, dama, e simili)
2) Arcade games (giochi di movimento veloce, di fantasia, pieni di rumori, in tempo reale)
3) Giochi di fortuna (come Roulette e Snap)

Nei "Board games" la qualità del gioco è più importante della velocità della risposta. Negli "Arcade games" il movimento deve essere mantenuto ad ogni costo, anche se talune "intelligenze" tra i tuoi invasori Marziani devono essere, a tale scopo, sacrificate. I giochi di fortuna si basano sulla facilità di gioco e sulla casualità vera e propria in modo maggiore rispetto alle altre due categorie.

Scoprirai che i programmi dei giochi si dividono in generi, i quali a loro volta sono suddivisioni delle tre categorie di cui sopra. Molti "board" games sono varianti della dama o degli scacchi, molti "arcade" games erano originariamente giochi del tipo "Invasori dello spazio" mentre i giochi di fortuna sono nati nel "mondo reale" dei dati e delle carte. Prestare attenzione alla descrizione di un programma o a una macchina da gioco, per provare ad indovinare la categoria a cui appartiene il gioco che stai osservando, può aiu-

tarti e ispirarti nuove idee che si adattino a quel particolare genere di giochi. Esiste un concetto nell'ambito della programmazione — chiamato generalmente "programmazione strutturata" — secondo il quale la disciplina è essenziale all'inizio del processo di scrittura dei programmi. Pur essendo meno interessante del fatto di sedersi subito davanti al computer, alla fine il programma prodotto sarà migliore. Una volta scrissi un programma chiamato Dome Dweller (Abitante di cupole) un programma a simulazione nel quale il giocatore ha il controllo di una "cupola lunare" e deve decidere quali prodotti produrre e vendere al fine di comprare ossigeno e cibo per gli abitanti del posto (vedi *The book of Listings,* scritto da Jeremy Ruston e edito dalla BBC). Dopo aver deciso lo schema generale da adottare, eleborai i dati e le informazioni da scrivere sul video, nel modo seguente:

Le scorte di ossigeno sono scarse
All'interno della tua cupola vivono 96 persone nell'anno 3
Il credito in denaro è di $ 5.693
Le spese di manutenzione annuali ammontano a $ 226
Le cisterne di ossigeno sono di 811 unità
L'ossigeno costa $ 8 per unità
Ogni abitante di cupola necessita di 5 unità all'anno
Le riserve di cibo ammontano a 2122
Ogni abitante necessita di 3 unità all'anno ($ 6 ognuno, $ 576 per la cupola. Ciò basterà per 7 anni alla popolazione attuale)
Tu puoi commerciare le tue straordinarie sculture lunari con la gente che vive nelle altre cupole. Consumi 2 unità di ossigeno per la costruzione di un'unica scultura, e le vendi a $ 30.

Come hai probabilmente indovinato da questo "printout" l'idea del programma è quella di decidere quante "straordinarie sculture lunari" devi costruire e vendere al fine di comprare ossigeno e cibo e pagare il "mantenimento annuale". Il problema in questo particolare programma è che per costruire ogni scultura si consuma ossigeno cosicché il giocatore si trova combattuto fra il desiderio di guadagnare e la necessità di usare l'ossigeno intelligentemente.
Puoi provare a scrivere un programma come questo da solo. In tal caso potresti riuscire a scrivere un programma divertente e ciò svilupperebbe notevolmente la tua capacità

di programmazione. La prima cosa da fare è un elenco di ciò che il programma deve compiere:

Stabilire le variabili necessarie
Comunicare al giocatore in quali condizioni si trova la cupola
Chiedere quanto ossigeno bisogna comprare
Controllare se si ha il denaro sufficiente. In caso positivo comprarlo, altrimenti ritornare e chiedere di nuovo
Chiedere quanto cibo deve essere comprato
Controllare se il denaro è sufficiente. In caso positivo comprarlo, altrimenti ritornare e chiedere di nuovo
Rinnovare la quantità di ossigeno
Rinnovare la quantità di cibo
Ridurre il totale del denaro lasciato
Domandare quante sculture devono essere costruite
Controllare se c'è l'ossigeno sufficiente per la costruzione di quel numero di statue. In caso negativo ritornare indietro e chiedere di nuovo
Ridurre la quantità di ossigeno per l'ammontare necessario alla costruzione del numero di sculture specificato. Incrementare il totale del denaro per considerare il prezzo delle sculture costruite
Incrementare leggermente l'intera popolazione, aggiungere uno all'"anno attuale"
Controllare se le riserve di cibo sono sufficienti per alimentare l'intera popolazione
Verificare se l'ossigeno è sufficiente per l'intera popolazione
Verificare se c'è un po' di denaro
Se qualcuna fra queste condizioni è negativa (per esempio, il cibo non è sufficiente) trasmettere l'azione ad un sottoprogramma "fine-gioco"
Se tutte le condizioni sono positive, ripetere di nuovo le istruzioni per comunicare al giocatore le condizioni in cui si trova la cupola, e continuare a ripetere le istruzioni.

Probabilmente saresti capace di scrivere un programma "Dome Dweller" usando le voci qui sopra elencate e l'informazione campione "printout". C'è comunque un segreto che vorrei svelarti e che risolve quasi istantaneamente i problemi legati alla programmazione. In realtà, sei in grado

di scrivere tutte le parti vitali di un programma come questo quando ne hai eleborato soltanto un rozzo schema generale, prima quindi di passare ai dettagli. E una volta che hai la struttura, puoi elaborarla per tutto il tempo che vuoi, sapendo che, mentre fai ciò, il programma che stai sviluppando è in ogni momento operante. Non devi aspettare la fine per metterlo in funzione e vedere come tu stesso ti comporti. Il "segreto" è di mantenere l'intero programma all'interno di una serie di sottoprogrammi a disposizione, ognuno dei quali mantenuto in un circuito di istruzioni continuo. Ecco come ciò che ho riferito può adattarsi a questo programma. Le primissime righe che inserirai nel tuo computer saranno le seguenti:

```
10 REM DOME DWELLER
20 GOSUB 1000: REM ASSEGNAZIONE DELLE VA-
RIABILI
30 GOSUB 2000: REM VISUALIZZAZIONE DELLA SI-
TUAZIONE DELLA CUPOLA
40 GOSUB 3000: REM OSSIGENO
50 GOSUB 4000: REM CIBO
60 GOSUB 5000: REM SCULTURA
70 GOSUB 6000: REM AUMENTO DELLA POPOLA-
ZIONE
80 GOSUB 7000: REM CONTROLLO DELLA SITUA-
ZIONE DELLA CUPOLA
90 IF (tutte le condizioni sono positive, da GOSUB
7000) THEN GOTO 30
100 REM Fine del gioco...
```

Come tu stesso puoi vedere, una volta che hai stabilito il "loop principale delle istruzioni", è relativamente facile compilare uno ad uno i sottoprogrammi, e allo stesso tempo sperimentarli ed elaborarli. In questo modo otterrai un ottimo programma. La sola cosa che adesso ti serve è un elenco delle variabili che userai insieme al programma. Sono dell'opinione che il modo migliore per far ciò sia quello di utilizzare nomi espliciti per le variabili. In questo modo, mentre stai programmando, non devi perder tempo nel verificare, per esempio, se AA indica la popolazione o il numero delle unità di ossigeno consumate per costruire ognuna delle sculture. Per facilitare il più possibile i programmi, nella trasmissione di questi fra differenti computer, puoi re-

star fedele ai nomi variabili a due lettere, o puoi usufruire (se il tuo computer lo permette) di nomi lunghi (come OXYUSE per la quantità di ossigeno usata) per le variabili. Di conseguenza non avrai dubbi sul significato di ogni nome di variabile. Per dimostrarti l'efficacia di questo accorgimento e per illustrarti un ulteriore vantaggio dei nomi espliciti per le variabili, ecco le variabili usate nel Dome Dweller:

FOLK — la popolazione della cupola
CASH — denaro in cassa
FOOD — le riserve di cibo a disposizione
FOODCOST — quanto costa ogni unità di cibo
FOODNEED — quante unità di cibo sono consumate da ogni persona nell'arco di un anno
ARTCOST — quanto ossigeno è consumato per costruire ogni pezzo di scultura
ARTPAY — a quanti dollari ogni pezzo è venduto
OXY — le riserve di ossigeno a disposizione
OXYNEED — quante unità di ossigeno sono consumate da ogni persona nell'arco di un anno
OXYCOST — quanto costa ogni unità di ossigeno che deve essere comprata
REPAIR — il costo annuale delle riparazioni concernenti la cupola
YEAR — l'anno di vita della cupola

Sebbene i nomi espliciti di variabile consumino maggiore memoria rispetto ai nomi di variabile a una o a due lettere, il loro uso permette di seguire un programma in modo molto facile, controllando ciò che compie in realtà ogni sezione del programma. Inoltre, e questo è l'ulteriore vantaggio che ho menzionato, al momento di scrivere il programma, è oltremodo facile inserire le formule richieste per operazioni di calcolo. Con questo voglio dire che se, per esempio, desideri includere (come faccio io in questo programma) un'indicazione di quanto ossigeno è necessario ogni anno, moltiplichi semplicemente il numero delle persone presenti nella cupola (FOLK) per il numero di unità di ossigeno di cui ogni persona necessita in un anno (OXYNEED). A questo punto puoi includere entro le "print" le condizioni riguardanti la cupola come, ad esempio:

```
PRINT "CI SONO"; FOLK; "NELLA CUPOLA"
PRINT "NELL'ANNO"; YEAR
PRINT "OGNI PERSONA ABBISOGNA DI"; OXY-
      NEED; "UNITÀ DI"
PRINT "OSSIGENO OGNI ANNO,"; OXYNEED * FOLK;
"FABBISOGNO ANNUALE DI OSSIGENO"
PRINT "PER L'INTERA CUPOLA"
```

In questo modo è anche più facile verificare se è possibile comprare. Per esempio, per comprare cibo, potresti dire:

```
PRINT "QUANTO CIBO VUOI COMPRARE?"
INPUT A
IF A * FOODCOST > CASH THEN GOTO (un'altra A)
```

I consigli che seguono possono migliorare i tuoi programmi facendo uso di una "programmazione strutturata":

— redigere un campione di "printout" o un modello dimostrativo dei dati e delle informazioni illustrati sul video

— redigere una lista di ciò che il programma deve compiere ogni volta che attraversa il "loop principale di istruzioni"

— trasformare questa lista in una serie di sottoprogrammi a disposizione

— se possibile, usare espliciti nomi per le variabili

Se stai progettando dei programmi che poi saranno usati da altri, assicurati che sia estremamente chiaro ciò che il giocatore dovrebbe fare mentre conduce il programma. Includere una lunga serie di istruzioni non è di molta utilità, specialmente quando la memoria è limitata, ma è in ogni caso opportuno prenderne nota. Inoltre, i suggerimenti destinati all'utente dovrebbero essere espliciti (come INSERISCI IL NUMERO DELLE MOSSE CHE VUOI EFFETTUARE) e dovrebbero avvertire riguardo i limiti che saranno posti sull'input (CON QUANTE CARTE VUOI INCOMINCIARE: 1, 2, o 3 ?, per esempio).

Non puoi supporre che sarai sempre presente ogniqualvolta un programma è in funzione, quindi, dovresti fare del tuo meglio per renderne il funzionamento il più possibile facile e sicuro. Se è possibile, aggiungi dei sottoprogrammi che eliminino gli errori.

In tal modo se si è precedentemente inserito un sottoprogramma alternativo e c'è stato un errore, questo non rovinerà il programma e non lo renderà incapace di dare buoni risultati in seguito.
Se leggi questa parte del libro da cima a fondo più di una volta e tenterai di applicare le idee qui proposte al tuo lavoro di programmazione, la qualità del tuo lavoro migliorerà in modo significativo. Ciò ti permetterà anche di dedicare più tempo al miglioramento e al perfezionamento di un programma e minor tempo al semplice lavoro meccanico di renderlo operativo.

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 programml 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

SKYDIVER

2100 — Premi START per giocare
3020 — Non avevi
3030 — il paracadute aperto
3500 — BRAVO...
3510 — sei riuscito ad atterrare
3520 — sulla piattaforma con il
3530 — paracadute aperto...
3540 — punteggio
3640 — sei riuscito
3650 — ad atterrare 10 volte
3670 — il tuo punteggio finale:-
10030 — Per favore aspetta un momento

3D NOUGHTS AND CROSSED

3070 — Premi START per giocare
6000 — oppure un uomo contro di me
6020 — (Batti 1 o 2)

RACE

90 — Alto
3 — Basso
135 — Marcia miglia
620 — Un'altra partenza, Basso

LUNAR LANDING

486 — Carburante
830 — Il gioco è finito
835 — Punteggio elevato
840 — Premi START
970 — Sinistra< >Destra
998 — Buona fortuna
1010 — Ottimo punteggio

DECISION MAKER

80 — Quante sono le opzioni tra cui scegliere
90 — Quanti fattori influenzano la decisione
170 — Inserisci il nome della opzione
230 — Inserisci il nome del fattore
320 — Come valuteresti
330 — Sul
390 — Ecco cosa penso che dovresti fare:

SAFE CRACKER

15 — Premi START
1007 — Per favore aspetta
1040 — Ecco la cassaforte...shhhh!!!
1043 — È meglio accendere la luce laterale. Non c'è nessuno....premi START
1050 — Così va meglio: ora puoi vedere cosa stai facendo
1070 — Premi START per fare un altro tentativo
2003 — Qual è il tuo primo numero da 1 a 99
2500 — Qual è il tuo secondo numero
3000 — Qual è il tuo terzo numero
3025 — Hai aperto la cassaforte
3028 — Bravo, c'erano delle monete in quella cassaforte!!!
3029 — Ora prova con quest'altra...
3850 — Giusto
3950 — Accidenti!! Hai fatto scattare l'allarme—Tagliamo la corda... arriva la polizia
3960 — Hai preso ...monete dalle cassaforti
4000 — Hai sbagliato: prova ancora

TANK BATTLE

70 — Inserisci la durata del gioco
50 — Battaglia di carri armati
260 — Hai finito i colpi
270 — colpi 1
290 — colpi 2
800 — Il gioco è finito

DIGIT DODGE

11 — alto
1000 — ..invaso..
1040 — ottimo punteggio!!!
1050 — Un altro tentativo
1055 — Arrivederci.

GRAND PRIX 2

1000 — sei andato a fracassarti!
1030 — il tuo punteggio è — premi qualsiasi tasto.
2005 — il tuo punteggio
2040 — ora la velocità è

CITY BOMB

100 — Quale livello di difficoltà (1-9)
1000 — Sei atterrato!
1040 — Bravo, il tuo punteggio era
2140 — Vuoi riprovare?
5010 — Punteggio
10012 — City Bomb - Per favore aspetta

ENGULF

5010 — Hai colpito l'alieno, capitano
5020 — e sei stato distrutto
6510 — L'hai sconfitto! Bravo
6520 — Ti ha preso — colpi
6530 — e ce l'hai fatta in — unità di tempo — rimaste
6550 — La tua valutazione è
6570 — Finora il punteggio più alto è
6590 — Premi 1 per giocare ancora e 2 per smettere
6620 — Oltre e fuori, capitano
7010 — Su quale settore sparerai
7020 — Attraverso
7040 — E in basso
7070 — Settore già distrutto
8030 — ---L'alieno rileva il pericolo del fattore-

8050 — Il computer del Battlestar riferisce: — il punteggio massimo è
8090 — Ora l'alieno ha
8110 — Tempo a disposizione — colpi sparati

COLOUR PUZZLE

1080 — Bravo! Hai risolto il puzzle.
2060 — Peccato! Prova ancora.
2110 — Era rimasto — da riempire

SOUND PROGRAM

105 — Premi i tasti per modificare
106 — Note. Per risentirle premi 'ESC'
170 — Quale tempo (1-4)

REACTION TIMER

32 — Nome del giocatore N. 1.
33 — Nome del giocatore N. 2.
1000 — Imbroglione! Hai premuto il pulsante troppo presto
3000 — Vince!... con un tempo di:-
5080 — Premi il pulsante 'FIRE' quando sei pronto
6000 — Vince il gioco!!...

MORSE CODE

100 — Traduttore in codice Morse
120 — Batti il messaggio che vuoi sentire:-

SPACE DOCKER

101 — l'aria è finita
150 — premi START
1020 — hai attraccato
1100 — ossigeno avanzato
10060 — stai perdendo aria
10070 — e devi attraccare alla
10080 — nave appoggio che
10090 — sta passando...Buona fortuna!

SKI RUN

30 — Assicuratevi di registrare questo programma
40 — prima di provare a eseguirlo!

Computer Games

GIOCHIAMO CON ATARI

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: ATTERRAGGIO SULLA LUNA (come atterrare sani e salvi con il modulo lunare); DECISIONI DIFFICILI (il computer vi aiuta a decidere); IL PARACADUTISTA (se soffrite di vertigini non fate questo gioco!); LO SCASSINATORE (10 tentativi prima che arrivi la polizia); IL CACCIATORE DI NUMERI (lotta senza quartiere contro i numeri invasori); IL GENERATORE DI COMPLIMENTI (il tirami-su per i depressi); LO SCIATORE (...e la Coppa del Mondo sarà vostra); CONTRO L'ALIENO (attacco al vostro incrociatore spaziale).

"GIOCHIAMO CON ATARI" vi aiuterà moltissimo, giocando, a migliorare la vostra abilità fornendovi non solo tutte le istruzioni per inserire correttamente i programmi nel vostro 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 006-0121-7 ISBN 88-7605-121-1

L. 9.500 (i.i.)