The Virgin Computer Games Series

Series Editor: Tim Hartnell

More GAMES FOR YOUR ZX SPECTRUM

£££££'s of entertaining games for only £3.50

Graham Carter

MORE GAMES FOR YOUR ZX SPECTRUM

By Graham Carter

Virgin Books

First published in Great Britain in 1983 by Virgin Books Ltd, 61-63 Portobello Road, London W11 3DD.

ISBN 0 907080 99 5

Printed and bound in Great Britain by Richard Clay (The Chaucer Press) Ltd, Suffolk.

Production services by Book Production Consultants, Cambridge.

Designed by Ray Hyden.

Illustrated by Sue Walliker.

Typeset by QV Typesetting.

Distributed by Arrow Books.

TIM HARTNELL — SERIES EDITOR

Tim Hartnell is a leading computer expert and journalist, who has contributed extensively to the Technical Consumer Press. He is also the author of several books including *Getting Acquainted With Your ZX81*, *Let Your BBC Micro Teach You to Program* and *Programming Your ZX Spectrum*.

GRAHAM CARTER — THE AUTHOR

Graham Carter is an 18-year-old student, currently studying for a National Diploma in a BEC/TEC Computer Studies Course. He has been working with computers for over three years. His other interests include music and photography.

SUE WALLIKER — THE ILLUSTRATOR

Sue Walliker is a freelance illustrator.

ACKNOWLEDGEMENTS

The author wishes to thank Scott Vincent, Clive Gifford, Peter Shaw, Andrew Sweetland and Simon Gould.

TO PAUL F.

CONTENTS

Editor's Introduction

Your computer is waiting to challenge you. Moving graphics games, brain stretchers, word games and puzzles are all here and ready to entertain you.

A wide variety of games are included in this book. The programs have been written by some of the most talented young programmers working in this country at the moment, and represent a variety of approaches to solving programming problems.

An examination of the listings should teach you many tricks and techniques to apply to your own programming. And once you have mastered the programs in their present form, you might want to try your hand at improving them. There is no such thing as a 'perfect program', so these games are sure to benefit from your programming skill.

All that now remains is for you to turn the page and enter the programs. I can only hope that you enjoy playing the games as much as we did when preparing this volume.

Tim Hartnell, series editor
London
March 1983

Author's Introduction

The Sinclair ZX Spectrum is a relatively cheap personal computer with some advanced, professional computer features.

The programs in this book all use the Spectrum's colour facilities, and many use the high resolution plotting features to add realism to the games.

The book includes arcade games, strategy games, adventure games and gambling games. In games against the computer, a balance has been maintained to equalise out the computer's and the player's chances of winning. Many of the programs use useful routines which you can incorporate into your own programs.

Here, then is a collection of games to suit programmers of all ages.

Graham Carter
Stanwell, Middlesex
August 1983

Program Notes

Space Cavern
Line 1 contains the machine code routine and must contain 151 zeros.
N.B. Save a copy of the program on cassette before attempting to RUN for the first time.

Rescue
Line 380 graphic A B
Line 390 graphic C D E F
Line 400 graphic G H I J K L M
Line 410 graphic N O P Q
Line 420 graphic R S T U

Bazooka
Line 60 graphic A B B C, 7 shifted, D E E F
Line 100 graphic 8 shifted twice
Line 160 graphic 3 shifted twice, 3 shifted, 3 unshifted, 6 shifted, 5 shifted, 5 unshifted.
Line 480 graphic 8 shifted, 8 shifted.

Towering Inferno
Line 53 graphic A

21
Line 10 graphic A B repeated.

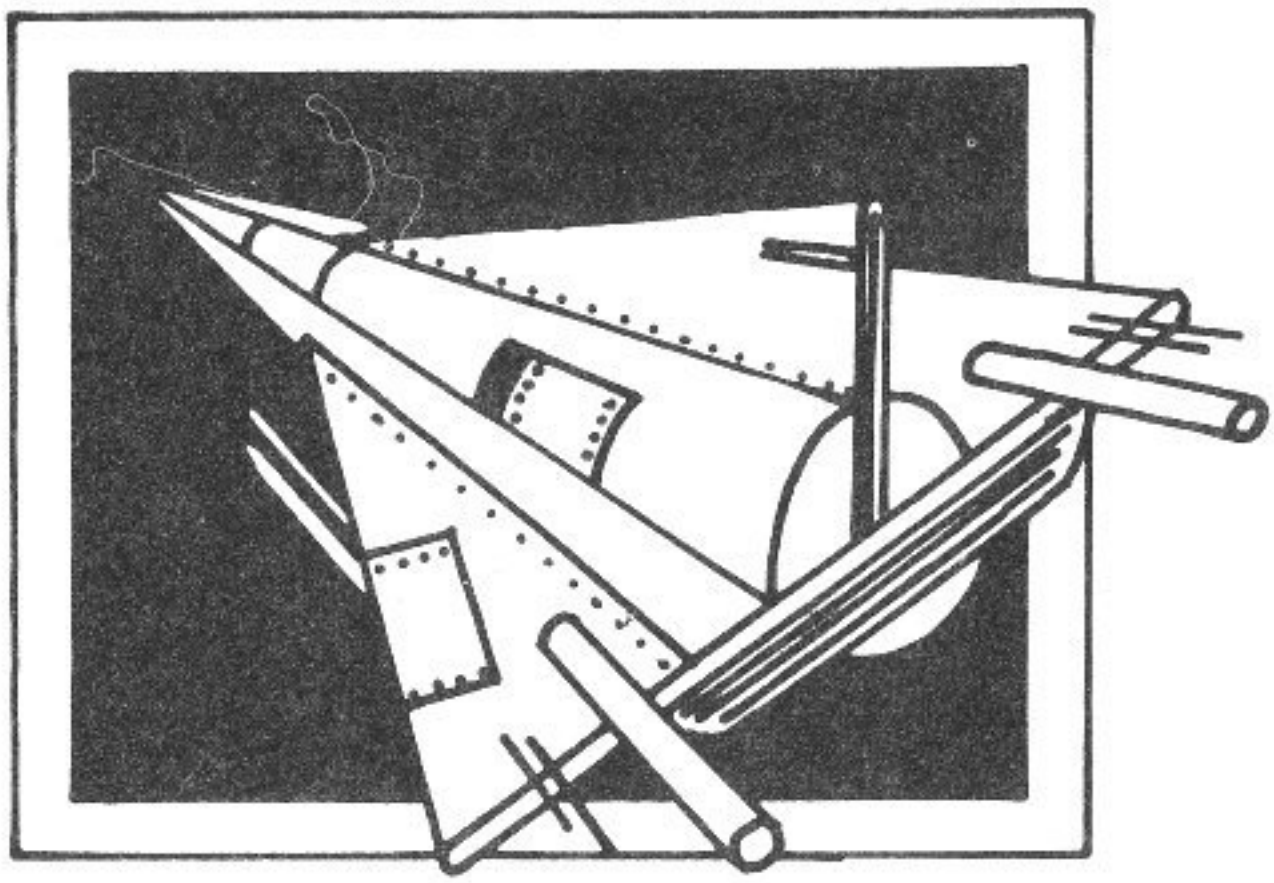
Stunt Motorcycle
Lines 120, 140, 150, 180, 200, 210, 1070 graphic E
Line 1010 graphic D
Line 1020 graphic A
Line 1040 graphic B
Line 1060 graphic C

SPACE CAVERN

This game will keep arcade game fans happy for a long, long time. This is a very fast, challenging game written partly in machine code to achieve the extra speed necessary for a good action game.

You control a space cruiser and must try to travel through the dangerous Space Cavern. The path is twisting and turning; the tunnel into the cavern gets narrower and narrower.

You control your craft with the up and down cursor keys. A game of 'Space Cavern' is a real challenge for your reflexes.



```
    1 REM 000000000000000000000000000
000000000000000000000000000000000000
000000000000000000000000000000000000
000000000000000000000000000000000000
000000000000000000000000000000000000
    5 IF PEEK 23909=201 THEN GO T
O 60
   10 LET a$="042 209 092 062 002
 050 060 092 062 017 215 062 005
 215 062 016 215 062 001 215 062
 022 215 058 208 092 215 062 006
 215 062 032 215 034 208 "
   15 LET A$=A$+"092 033 000 086
014 024 013 006 031 035 126 043
119 035 016 249 035 121 254 000
032 240 042 208 092 077 033 008
086 121 254 000 "
   20 LET a$=a$+"040 007 017 032
000 065 025 016 253 126 254 000
032 005 062 001 050 210 092 062
022 215 121 215 062 006 215 062
144 215 237 091 "
   25 LET a$=a$+"211 092 033 031
086 014 024 067 054 000 013 197
006 032 035 016 253 193 016 244
066 054 045 013 197 006 032 035
016 253 193 016 "
   30 LET a$=a$+"244 065 054 000
197 006 032 035 016 253 193 016
245 201 "
   40 FOR a=23765 TO 23909
   50 POKE a,VAL a$( TO 3)
   60 LET a$=a$(5 TO )
   70 NEXT a
   80 LET a$="224 048 024 124 127
 024 048 224"
   90 FOR n=0 TO 7
  100 POKE USR "a"+n,VAL a$( TO 3
)
  110 LET a$=a$(5 TO )
  120 NEXT n
  130 RANDOMIZE
  135 LET hi=0
  140 POKE 23760,10: POKE 23761,1
0: POKE 23762,0
  150 PAPER 5: INK 0: BORDER 0: C
LS
  160 LET sc=0: LET g=6: LET n=10
  162 LET a$="
                "
  165 PRINT INK 0; PAPER 0;AT 21,
0;a$;AT 0,0;a$
  170 FOR a=1 TO 25
  180 IF PEEK 23762=1 THEN GO TO
2000
```

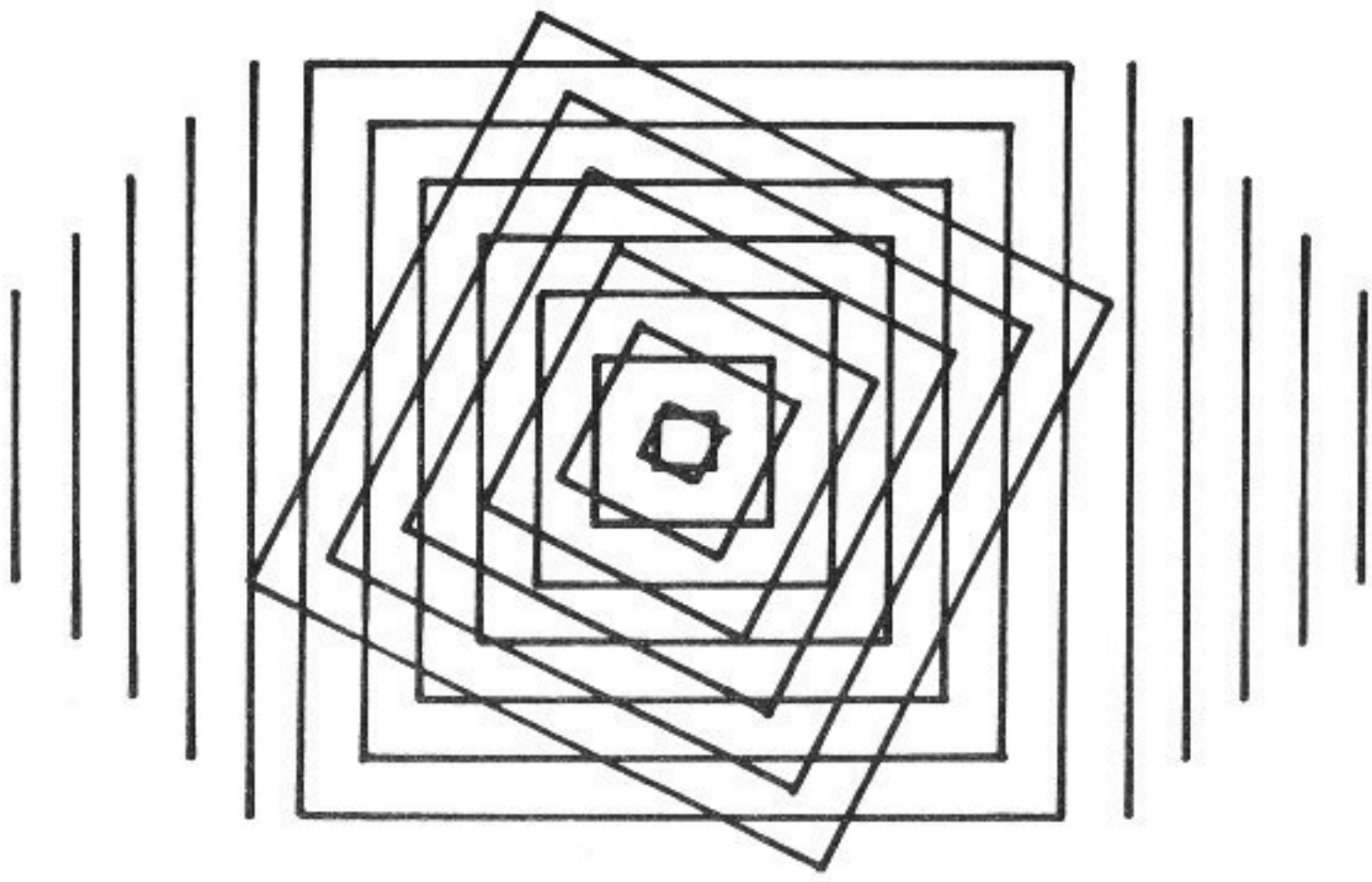
Checksum = 12840

```
  100  LET  L=a/2
  200  LET  g=21-a
  210  IF   a>14 THEN LET L=6: LET
g=6
  220  GO SUB 1000
  230  NEXT a
  240  LET a=6
  250  LET b=a-INT (RND*10)
  260  IF b<1 THEN LET b=1
  270  FOR L=a TO b STEP -1
  280  IF PEEK 23762=1 THEN GO TO
2000
  290  GO SUB 1000
  300  NEXT L
  310  LET a=b+INT (RND*10)
  320  IF a>21-g THEN LET a=21-g
  330  FOR L=b TO a
  340  IF PEEK 23762=1 THEN GO TO
2000
  350  GO SUB 1000
  360  NEXT L
  370  LET sc=sc+13
  380  IF sc/117=INT (sc/117) THEN
 LET g=g-1
  390  IF g<2 THEN LET g=2
  400  GO TO 250
 1000  IF INKEY$="6" THEN LET n=n+
1
 1010  IF INKEY$="7" THEN LET n=n-
1
 1020  POKE 23761,n
 1030  POKE 23763,L
 1040  POKE 23764,g
 1050  LET z=USR 23765
 1060  RETURN
 2000  LET p=PEEK 23760
 2010  FOR n=1 TO 10
 2020  FOR L=0 TO 7
 2030  PRINT INK L;AT p,6;CHR$ 143
 2035  BEEP .005,n*2+L*5
 2040  NEXT L: NEXT n
 2050  PRINT INK 0;AT p,6;CHR$ 143
 2060  PRINT PAPER 7; INK 0;AT 9,1
0;"score = ";sc
 2070  IF sc<=hi THEN PRINT PAPER
7; INK 0;AT 13,6;"high score = "
;hi
 2080  IF sc>hi THEN LET hi=sc: PR
INT FLASH 1; PAPER 0; INK 7;AT 1
3,6;"HIGH SCORE = ";hi
 2090  FOR a=1 TO 500: NEXT a
 2100  GO TO 140
```

MOIRE

This program generates high-resolution colour patterns on the screen, and is a very good demonstration of the Spectrum's plotting capabilities.

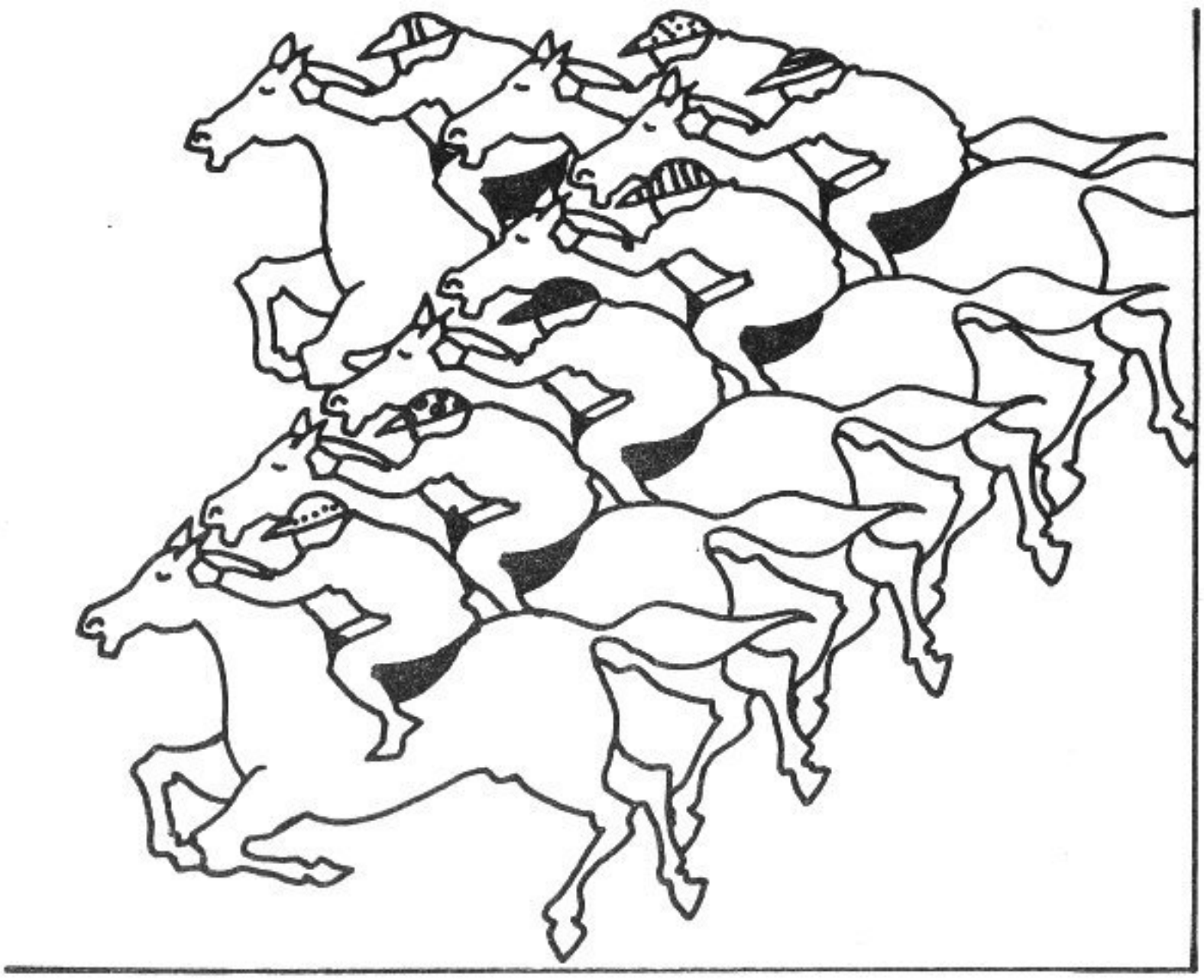
```
   10  REM  Moire
   20  REM  Martyn  Latimer
   30  REM  Daniel  Mills
   40  LET  i=6:  OVER  1:  BORDER  0:
PAPER  0:  CLS  :  INK  9:  FOR  a=1 TO
  255
   45  IF  RND>.9  THEN  LET  i=INT  (R
ND*7)+1
   50  INK  i:  PLOT  a,0:  DRAW  255-(
a*2),175
   60  NEXT  a
   70  FOR  a=0  TO  175
   75  IF  RND>.9  THEN  LET  i=INT  (R
ND*7)+1
   80  INK  i:  PLOT  0,a:  DRAW  255,1
75-(a*2)
   90  NEXT  a
  100  PAUSE  100:  GO TO 40
```



A DAY AT THE CHASES

Why bother to travel to the racetrack and lose all your money when you can gamble to your heart's content in the comfort of your own home? This game puts you on a course of your choice, betting a portion of your £200 on one of eight horses. There are different odds, a pay-out for winning and a graphic display of the race in progress with the horses jumping the fences and galloping to the finishing post.

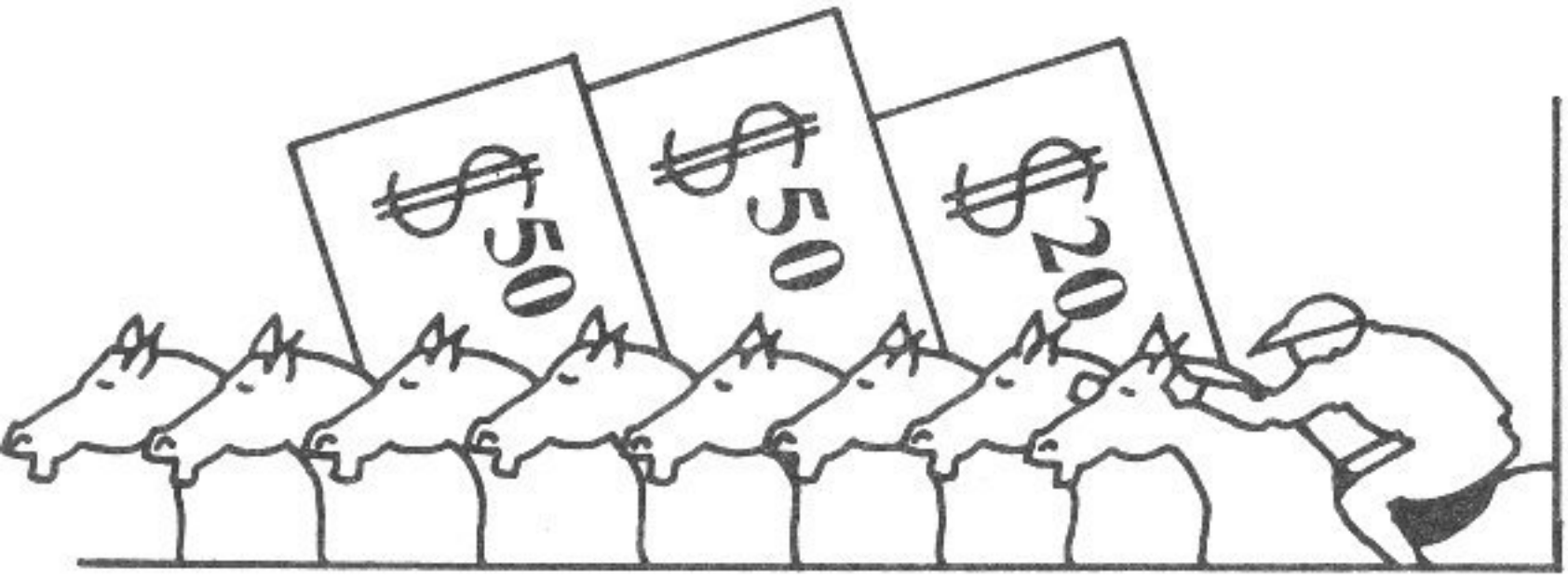
The game is simple to follow. To place a bet, just enter the number of the required horse and the amount you wish to bet and gamble away!



```
    5 INVERSE 0: PAPER 4: INK 0:
BORDER 1
   10 BRIGHT 0: CLS
   20 GO SUB 1470
   30 DIM h(8)
   35 LET m=200
   36 DIM d(8)
   38 GO SUB 1000
   40 FOR p=1 TO 8
   41 LET h(p)=0
   42 LET d(p)=0
   43 NEXT p
   44 CLS
   45 FOR L=4 TO 20
   50 PRINT AT L,29;CHR$ 134;AT L
,0;"/";AT L,16;"/";AT L,24;"/"
   55 NEXT L
   60 INK 1: PAPER 7: INVERSE 1
   62 PRINT AT 0,0;"
                    "
   64 PRINT "        A Day At The
Chases          "
   66 PRINT "
                    "
   68 FOR p=1 TO 16: PRINT CHR$ 9
6;CHR$ 36;: NEXT p
   70 INVERSE 0: INK 0: PAPER 4
   80 FOR p=1 TO 8
   82 IF h(p)=1 THEN PRINT AT p*2
+3,d(p);CHR$ 142;CHR$ 142
   84 IF h(p)=1 THEN GO TO 150
   85 LET r=RND
   87 PRINT AT p*2+3,d(p);"  "
   90 IF r>.9 THEN LET d(p)=d(p)+
2
   95 IF r>.45 AND r<.9 THEN LET
d(p)=d(p)+1
   98 IF r<.35 THEN LET d(p)=d(p)
+3
   99 IF p<4 AND RND<.5 THEN LET
d(p)=d(p)+INT (RND*3)
  100 IF p<7 AND p>3 AND RND<.34
THEN LET d(p)=d(p)+1
  108 IF d(p)=8 OR d(p)=16 OR d(p
)=24 THEN GO SUB 350
  110 IF d(p)>=29 THEN GO TO 490
  120 PRINT AT p*2+3,d(p);p
  150 NEXT p
  152 LET n=0
  155 FOR p=1 TO 8
  156 IF h(p)=1 THEN LET n=n+1
  158 NEXT p
  170 IF n<>8 THEN GO TO 80
  180 INVERSE 1: PRINT AT 10,10;"
***NO RACE***": INVERSE 0
  190 PAUSE 150
```

```
  195 CLS
  200 GO TO 640
  350 LET j=RND
  360 IF j>a THEN PRINT AT p*2+3,
d(p);CHR$ 141;CHR$ 141
  370 IF j>a THEN LET h(p)=1
  380 RETURN
  490 CLS
  500 PAUSE 50
  520 FOR s=1 TO 21
  530 FOR L=0 TO 31: PRINT CHR$ 1
34;: NEXT L
  540 NEXT s
  545 PAPER 4: INK 0: INVERSE 1:
PRINT AT 7,6;"
"
  550 PRINT AT 8,6;" The Winner W
as ";p;" "
  560 PRINT AT 9,6;"
      ": INVERSE 0
  570 PAUSE 100
  580 CLS
  600 IF p=z AND z>4 THEN LET m=m
+(mo*2)
  610 IF p=z AND z>6 THEN LET m=m
+(mo*8)
  620 IF p=z AND z<7 AND z>3 THEN
  LET m=m+(mo*4)
  630 IF p=z THEN PRINT "WELL DON
E"
  640 IF z<>p THEN PRINT "Hard Lu
ck,Try Again."
  650 IF m<1 THEN GO TO 800
  700 PAUSE 100
  710 GO SUB 1000
  720 GO TO 40
  800 CLS
  810 PRINT "You have run out of
money."
  820 PRINT "Good-Bye!"
1000 REM **All betting instruct-
ions here**
1005 CLS
1010 PRINT "    ";CHR$ 142;"  A D
AY AT THE 'CHASES   ";CHR$ 141
1015 PRINT " ";CHR$ 139;CHR$ 139
;"     **********************  ";CH
R$ 135;CHR$ 135
1020 PRINT '"Welcome,punter."'
1025 PRINT ,,"Choose your horse:
"
1030 PRINT AT 8,4;"1  Waywood So
n    2 to 1"
1040 PRINT AT 9,4;"2  Little Gen
ie   2 to 1"
1050 PRINT AT 10,4;"3  Blue Mist
```

```
      2 to 1"
1060 PRINT AT 11,4;"4  Mill Roug
h     4 to 1"
1070 PRINT AT 12,4;"5  Crystal C
lear  4 to 1"
1075 PRINT AT 13,4;"6  Frisky Fi
lly   4 to 1"
1080 PRINT AT 14,4;"7  Brown Bea
uty   8 to 1"
1090 INPUT z
1095 IF z<1 OR z>8 THEN GO TO 10
90
1120 CLS
1130 PRINT '"How much do you wan
t to bet ?"
1140 PRINT '"You have ";CHR$ 96;
m
1150 INPUT mo
1160 IF mo<1 OR mo>m THEN GO TO
1150
1170 LET m=m-mo
1180 PRINT '"You Now Have ";CHR$
 96;m
1200 PAUSE 100
1300 RETURN
1470 CLS
1480 PAPER 1: INK 7
1482 PRINT "Choose course: Aintr
ee=1
1484 PRINT TAB 15;"Leopardstown=
2"
1486 PRINT TAB 15;"Ascot=3
"
1488 PRINT TAB 15;"Cheltenham=4
"
1490 PRINT TAB 15;"Newbury=5
"
1500 INPUT a
1510 LET a=((a/10)*2)-0.05
1520 PAPER 4: INK 0
1790 CLS
1800 RETURN
```



AIR SEA RESCUE

While on holiday one year, Tim Hartnell capsizes his pedal-boat! You must fly to his rescue in your helicopter, or be haunted by his ghost for the rest of your life.

Use keys 5, 6 and 7 to manoeuvre the helicopter in the direction indicated by the arrows above these keys.

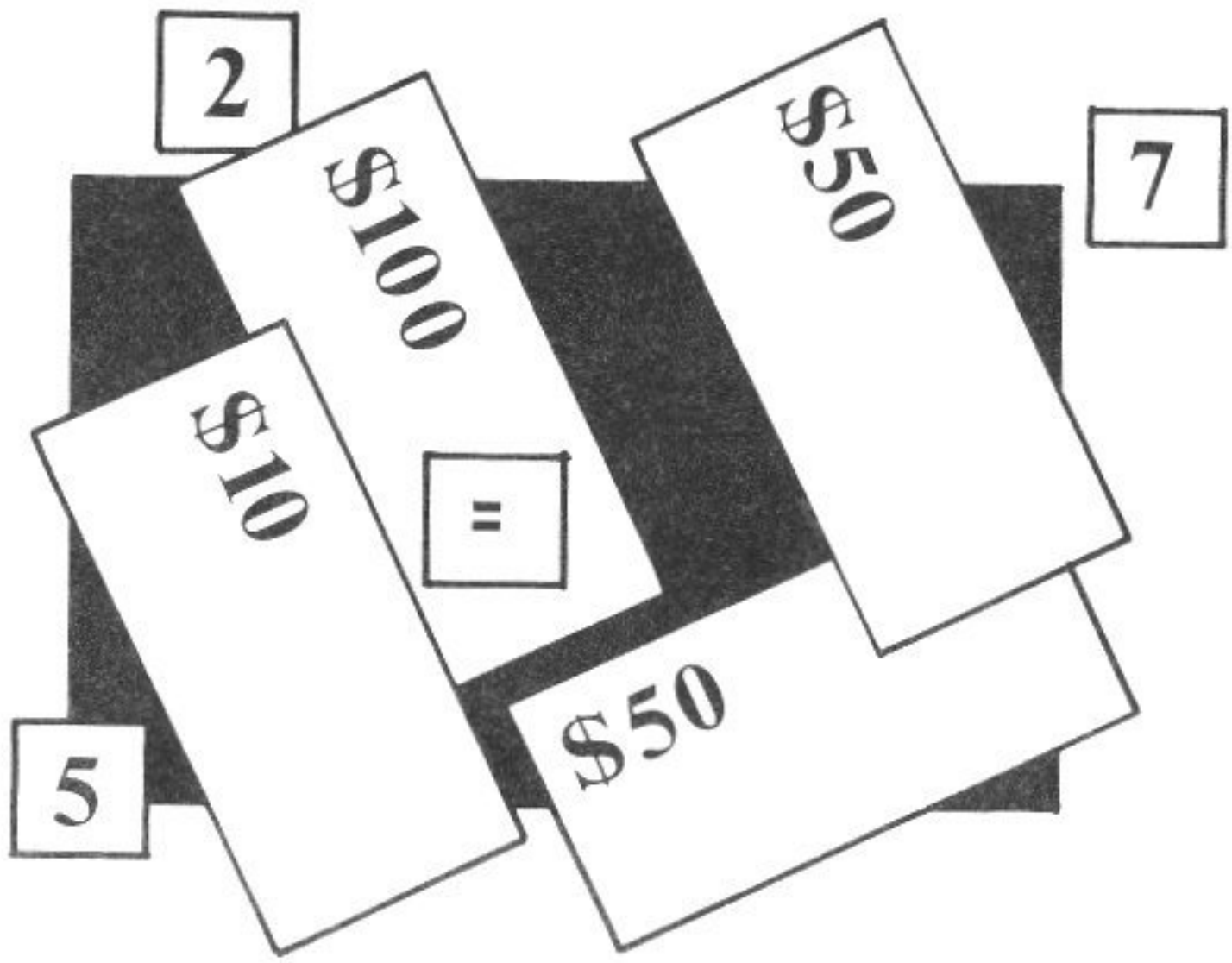
```
  10 INVERSE 0: PAPER 0: INK 7:
BORDER 0: BRIGHT 0
  20 GO TO 460
  30 LET a=0
  40 LET dx=INT (RND*29)
  50 LET y=20
  60 LET x=0
  70 IF dx<2 THEN GO TO 40
  80 IF dx>24 THEN GO TO 40
  90 CLS
 100 FOR b=1 TO 25
 110 LET y=y+(INKEY$="8")-(INKEY
$="5")
 120 LET x=x+(INKEY$="6")-(INKEY
$="7")
 130 IF x=13 AND y+4=dx OR x=13
AND y+3=dx OR x=13 AND y+2=dx TH
EN GO TO 230
 140 GO SUB 290
 150 IF x=13 AND y+4=dx THEN GO
TO 230
 160 NEXT b
 170 CLS
 180 PRINT AT 1,0;"You have fail
ed miserably. Tim Hartnell will
 haunt you forever."
 190 PRINT AT 6,0;"Again?"
 200 INPUT a$
 210 IF CODE a$=89 OR CODE a$=12
1 THEN GO TO 30
 220 STOP
 230 LET a=a+1
 240 PRINT AT 0,0; FLASH 1;"Cong
ratulations!"
 250 PAUSE 75
 260 CLS
 270 PRINT "You have saved Tim H
artnell"
 280 GO TO 190
 290 IF a=1 THEN GO TO 350
 300 LET z=INT (RND*10)
 310 IF z=1 OR z=2 THEN LET dx=d
x+2
 320 IF z=8 OR z=9 THEN LET dx=d
x-2
 330 IF dx>24 THEN LET dx=24
 340 IF dx<2 THEN LET dx=2
 350 PRINT AT 19,0; PAPER 5;"
                             "
 360 IF x>1 THEN PRINT AT x-1,y-
1;"
                             "
 370 IF x>=1 THEN PRINT AT 0,0;"
                             "
 380 PRINT AT x,y-1; INK 6;"  AB
"
```

```
 390 PRINT AT x+1,y; INK 6;"   CD
EF
 400 PRINT AT x+2,y; INK 6;"   GH
IJKLM
 410 PRINT AT x+3,y; INK 6;"   NO
PQ
 420 PRINT AT x+4,y; INK 6;"   RS
TU
 430 IF x<13 THEN PRINT AT x+6,y
;"
 440 PRINT AT 18,dx;"    ";CHR$ 9
2;CHR$ 111;CHR$ 47;"    "
 450 RETURN
 480 FOR n=USR "a" TO USR "u"+7:
 READ r
 490 POKE n,r: NEXT n
 880 GO TO 30
 890 DATA 24,6,3,0,0,0,0,0,0,0,0,0
,192,96,4,2,1,0,0,0,0,0,63,224,0
,192,24,6,3,62,224,1,15,0,0,3,24
8,224,96,144,250
 900 DATA 0,0,127,128,0,0,0,0,0,
7,8,24,32,64,128,126,48,255,32,3
2,32,32,32,32,8,255,0,0,0,0,0,0,
64,224,47,22,9,8,7,2,0,0,255,0,1
28,255,48,0,0,0,224,15,0,224,15,
0,36,56,76,156,36,34,193,129
 910 DATA 128,128,128,128,128,12
9,135,122,16,48,32,64,128,0,0,12
8,0,0,0,0,0,3,1,5,3,1,1,3,2,4,4,
142
 920 DATA 154,175,74,244,0,255,2
55,0,0,255,0,0,255,0,255,0,3,25
,0,0,255,0,255,0,80,160,160,80,2
55,0,255,0
```

007 DICEMASTER

In this multi-player game you must try to throw a seven with your two dice. You win double your bet if you throw a seven and receive your money back if you throw an odd number, but if you throw an even number you lose your money. You have a choice of six game lengths and as many players as you like. The computer updates the situation every round and skilfully calculates its bet according to its financial situation, and how recent games have gone.

The game stops after one go, but if you require a repeat option, then add the following line: 490 GO TO 5



```
    1  INVERSE 0: BORDER 1: PAPER
6:  INK 1
    2  BRIGHT 1: CLS
    5  GO SUB 500
    9  LET Z=0
   10  LET V=0
   15  CLS
   16  INPUT "Length of game?(1-6;
 6=Long)";L
   18  IF L<1 OR L>6 THEN GO TO 16
   19  LET L=L*5
   20  INPUT "How many players?";P
   30  DIM A$(P+1,12): DIM M(P+1)
   35  FOR K=1 TO P: INPUT "Name?"
;A$(K)
   45  LET M(K)=40
   50  NEXT K
   55  LET A$(P+1)="Computer"
   57  LET M(P+1)=40
   60  PRINT
   65  FOR K=1 TO P+1
   66  IF M(K)<=0 THEN NEXT K
   67  CLS
   68  PRINT "Enter your bet, ";A$
(K)
   69  IF K=P+1 THEN LET C=INT (.2
5*M(K))
   70  IF K=P+1 AND Z>2 THEN LET C
=C+INT (.1*M(K)*(2*Z/3))
   71  IF K=P+1 AND RND>.8 THEN LE
T C=C+INT (M(K)/6)
   72  IF K=P+1 AND RND<.2 THEN LE
T C=C-INT (M(K)/6)
   73  PRINT
   75  IF K=P+1 THEN PRINT "Comput
er bets ";C
   77  IF K=P+1 THEN GO TO 80
   78  INPUT C
   79  IF C>M(K) THEN GO TO 68
   80  PRINT
   82  INPUT "Press ""ENTER"" to t
hrow dice";N$
   90  LET DD=INT (RND*6)+1
  100  LET D=INT (RND*6)+1
  105  PRINT
  110  PRINT D,DD
  115  LET T=D+DD
  117  PRINT
  120  PRINT "Total of ";T
  135  IF T/2=INT (T/2) THEN GO SU
B 420
  140  IF T=7 THEN GO SUB 300: REM
 T=7???
  142  PRINT ,,
  145  INPUT "Press ""ENTER"" to c
ontinue";N$
```

```
 147 IF K=P+1 AND T<>7 THEN LET
Z=Z+1
 150 CLS
 160 NEXT K
 162 LET V=V+1
 165 PRINT "End of round, ";V
 167 FOR K=1 TO P+1: IF M(K)<=0
THEN NEXT K
 170 PRINT 'A$(K);" has ";M(K)
 172 PRINT
 174 NEXT K
 175 IF V=L THEN GO TO 460
 176 INPUT "Press ""ENTER""";N$
 180 GO TO 65
 300 PRINT '
 310 IF K=P+1 THEN PRINT "******
****** I win!  **************"
 320 IF K<>P+1 THEN PRINT "*****
****** You win!  ************"
 330 LET M(K)=M(K)+2*C
 400 RETURN
 420 PRINT '
 430 IF K<>P+1 THEN PRINT "You l
ose, Buddy!"
 432 IF K=P+1 THEN PRINT "I lost
!"
 435 LET M(K)=M(K)-C
 440 RETURN
 460 PAUSE 300
 462 CLS
 464 FOR T=1 TO 20
 466 PRINT "++++++++++++++++++++
++++++++++++"
 468 NEXT T
 470 PRINT AT 8,8;"End of game"
 480 PAUSE 1000
 490 GO TO 5
 500 CLS
 505 PRINT "          007 - Dicemas
ter"
 510 PRINT '"You have £40 to bet
 on the"
 520 PRINT "two dice. You win if
 the dice"
 530 PRINT "come up seven.  You
lose  if the"
 540 PRINT "dice total an even n
umber.  Any"
 550 PRINT "other score does not
 count"
 560 PRINT "-----Press any key t
o start-----"
 570 IF INKEY$="" THEN GO TO 570
 580 PRINT '"          Good luck!"
 590 PRINT 10000
 600 RETURN
```

REPEAT

Repeat is a simple game where you must repeat the selection of characters that appear on the screen. Concentrate now, they only appear for a very short time. Every time you guess the correct combination of symbols, an extra character is added on. When you have guessed all the characters correctly, you win the game.

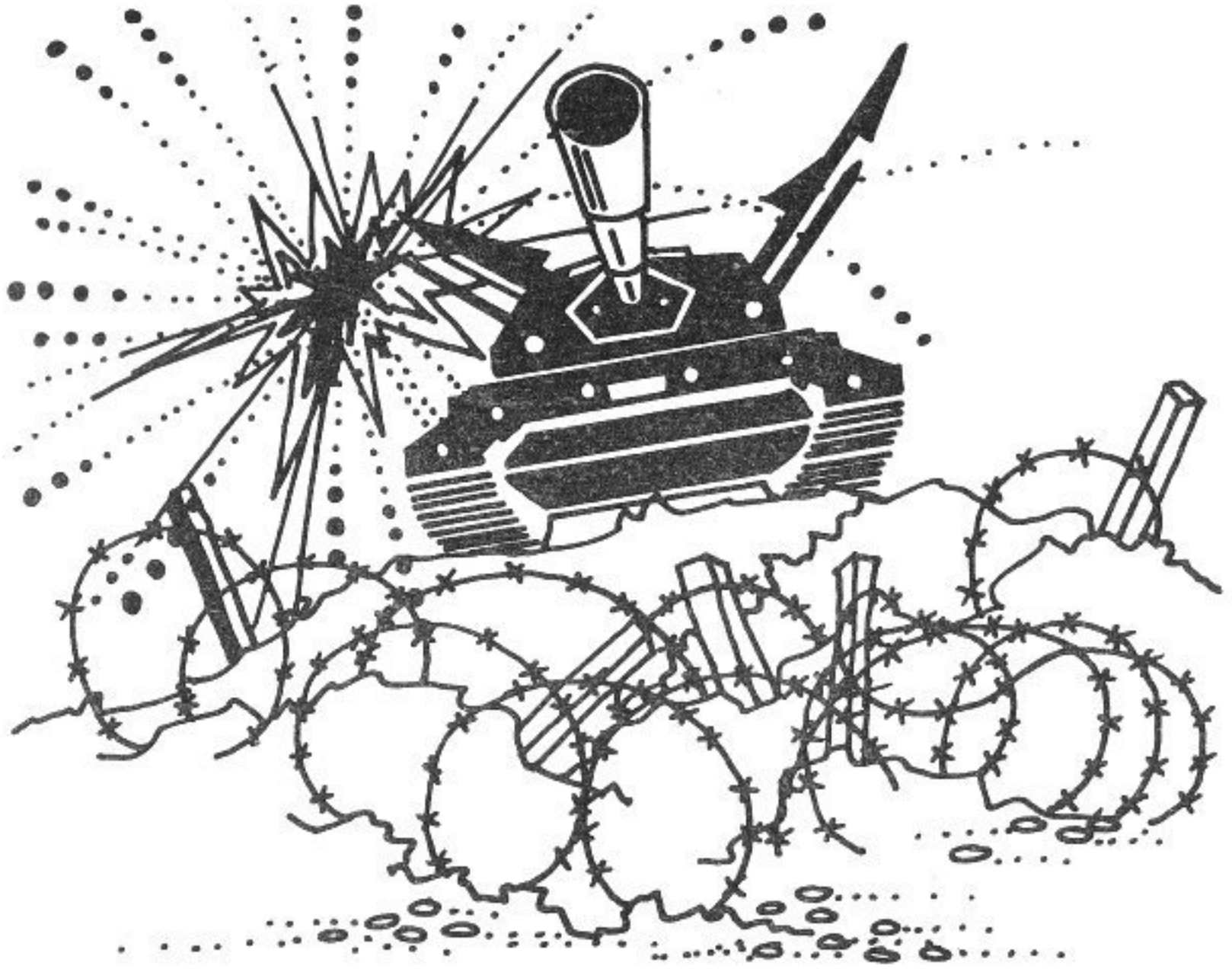
```
   60 INVERSE 0: PAPER 0: BORDER
0:  INK 6
   90 BRIGHT 1: CLS
  100 LET x=6+INT (RND*10)
  110 DIM f$(x)
  120 FOR n=1 TO x
  130 LET f$(n)=CHR$ (48+INT (RND
*10))
  140 NEXT n
  150 PRINT "This number has ";LE
N (f$);" figures."
  160 PAUSE 25
  170 FOR t=1 TO LEN (f$)
  180 PRINT AT 10,10; INK 7; BRIG
HT 0;f$(1 TO t)
  190 PAUSE 12
  200 CLS
  210 PRINT "Enter guess"
  220 INPUT a$
  230 IF a$<>f$(1 TO t) THEN GO T
O 300
  240 PRINT AT 14,7; INK 4;CHR$ 1
34;CHR$ 134;CHR$ 134;"Correct!";
CHR$ 137;CHR$ 137;CHR$ 137
  250 PAUSE 40
  260 CLS
  270 NEXT t
  280 PRINT AT 10,10;"You win."
  290 STOP
  300 PRINT ',,,: INVERSE 1: PRINT
  INK 3;"...Wrong...": INVERSE 0
  310 PRINT "End of game."
  320 PRINT 'f$
```

BAZOOKA!

You lie in wait behind a pile of sandbags, with your anti-tank gun loaded and ready for use. An enemy tank appears on the horizon, but there is a large wall in your line of fire. You raise the turret of your gun and fire! The shell arcs over the wall and penetrates the tank's armour, turning it into a mass of mangled metal. There is little time for praise, however, as another tank comes into view.

In this game you must judge your angle and velocity for the shell to clear the wall and hit the tank.

The shell's journey is plotted according to genuine formulae as it hurtles towards the tank.



```
  5 BORDER 0: PAPER 0: INK 7
  7 CLS
 11 GO SUB 530
 13 RANDOMIZE
 15 CLS
 18 LET try=0
 20 DIM a$(32)
 30 LET wal=5+INT (RND*18)
 40 LET hgt=3+INT (RND*6)
 50 LET tank=wal+2+INT (RND*(2
7-wal))
 60 PRINT AT 19,tank; INK 5;"~~
";AT 20,tank; INK 5;"ABBC";AT 2
1,tank; INK 5;"DEEF"
 70 FOR a=1 TO hgt
 80 PRINT AT 22-a,wal; INK 6;"
G"
 90 NEXT a
100 PRINT AT 20,1; INK 4;"▄▄▄";A
T 21,0; INK 4;"▀▀▀"
110 PRINT AT 6,3;"Enter angle o
f projection"
120 INPUT ang
130 IF ang<1 OR ang>90 THEN GO
TO 120
140 PRINT AT 6,0;a$
150 LET x=1+(ang>10)+(ang>33)+(
ang>55)+(ang>77)
160 PRINT AT 20,1; INK 4;("▄▄▄"
AND x=1)+("▀▄▄" AND x=2)+("▀▄ " AN
D x=3)+("▀▀ " AND x>3)
170 PRINT AT 19,1; INK 4;(" ▄▀"
AND x=3)+(" ▀▀" AND x=4)+("▀▀ " AND
x=5)
180 PRINT AT 8,6;"Enter shell v
elocity"
190 INPUT vel
200 IF vel<10 THEN GO TO 180
202 LET try=try+1
205 LET vel=vel/4
210 PRINT AT 8,0;a$
220 LET ang=ang*PI/180
230 LET hori=COS ang*vel
240 LET vert=SIN ang*vel
243 LET v=0
246 LET w=0
250 LET t=0
260 LET y=2+hori*t
270 LET x=18-INT (vert*t-4.9*(t
*t))
273 IF x<0 THEN PRINT AT v,w;"
"
275 IF x<0 THEN GO TO 320
277 IF y>31 THEN GO TO 410
278 IF x>21 THEN LET x=21
280 PRINT AT x,y; INK 3;"*"
```

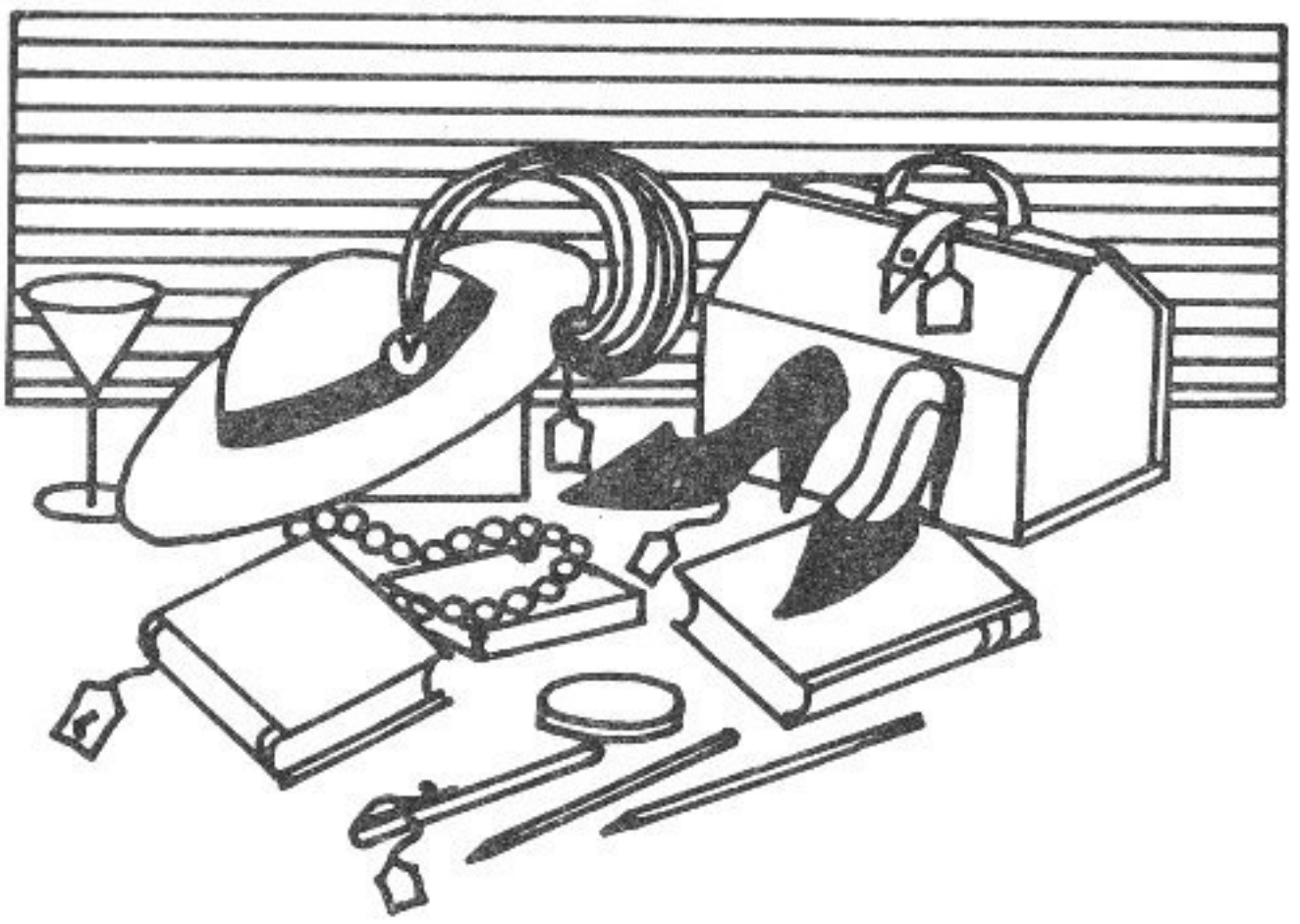
```
 290 PRINT AT v,w;" "
 300 LET v=x: LET w=y
 320 LET t=t+.2
 330 IF x>20 THEN GO TO 360
 340 IF y>=wall-1 AND y<=wall+1
AND x>=21-hgt THEN GO TO 360
 350 GO TO 260
 360 PRINT AT v,w;" "
 370 IF x>20 THEN GO TO 400
 380 PRINT AT 8,5;"Oops, you hit
 the wall"
 390 GO TO 420
 400 IF y>=tank AND y<=tank+4 TH
EN GO TO 470
 410 PRINT AT v,w;" "
 415 PRINT AT 8,9;"You missed it
"
 420 PRINT AT 10,4;"Hit any key
to try again"
 430 IF INKEY$="" THEN GO TO 430
 440 PRINT AT 19,1;"  "
 450 PRINT AT 8,0;a$;AT 10,0;a$
 460 GO TO 100
 470 FOR n=1 TO 15
 480 PRINT AT 20,tank+1;"▒ ";AT
20,tank+1;" ▒"
 490 NEXT n
 495 CLS
 500 PRINT AT 8,4;"Great shot -
that took you"
 505 PRINT AT 10,12;try;: IF try
=1 THEN PRINT " go."
 506 IF try<>1 THEN PRINT " goes
."
 508 PRINT AT 12,9;"Press any ke
y."
 510 IF INKEY$="" THEN GO TO 510
 520 RUN
 530 FOR n=USR "a" TO USR "g"+7:
 READ r
 540 POKE n,r: NEXT n
 690 RETURN
 700 DATA 1,3,7,15,31,63,127,255
: DATA 255,255,255,255,255,255,2
55,255: DATA 128,192,224,240,248
,252,254,255: DATA 63,64,156,162
,162,156,64,63: DATA 255,0,60,12
6,126,60,0,255: DATA 252,2,57,69
,69,57,2,252: DATA 0,126,255,126
,0,126,255,126
```

JUNKSHOP

Why not grab anything that you don't want and take a stroll down to your local junk shop?

This program allows you to try to sell anything which you wish to get rid of to the owner of the junk shop. You must haggle over the price with him; he is a very stubborn man and will not accept anything valued at more than £200.

```
  10 RANDOMIZE
  20 CLS
  30 LET d=0
  40 INPUT "What do you want to
s e l l ? ";a$
  50 IF RND<.8 THEN GO TO 90
  60 PRINT ,,"Sorry,I am not int
e r e s t e d  i n  ";a$+("s" AND a$(L
EN  a$)<>"s")
  70 PAUSE 75
  80 RUN
  90 CLS
 100 INPUT "How much are you ask
i n g ? ";a
 110 IF  a>100+RND*100 THEN GO TO
 60
 120 LET  c=a
 130 LET  b=d+INT  (RND*(a-d))+2
 140 IF  b>=a  THEN  GO TO  1000
 150 PRINT "I  w i l l  g i v e  y o u  £";b
 160 PRINT "What do you say?"
 170 INPUT  b$
 180 CLS
 190 IF  b$="YES"  OR  b$="yes"  OR
b$="Yes"  THEN  GO TO  1010
 200 IF  VAL  b$>c-RND*c/3  OR  VAL
b$>a  THEN  GO TO  60
 210 LET  a=VAL  b$
 220 LET  d=b
 230 GO TO  130
1000 LET  b=a
1010 CLS
1020 PRINT "I will take your ";a
$
1040 PAUSE 50
1050 RUN
```



ESCAPE!

You are in a P.O.W. camp in Germany, and are trying to escape through a tunnel beneath your hut.

There are two types of soil to dig through, but one type is much harder than the other, and digging through it uses up more of your valuable time.

Use the cursor keys to move in any direction, as you work towards the exit at the top left-hand corner of the screen.

```
   10  INVERSE  0:  CLEAR
   15  PRINT  "Enter a number for t
he  maze        pattern."
   20  INPUT  z
   25  IF  z<=0  OR  z>65535  THEN  GO
TO  20
   30  RANDOMIZE  z
   35  LET  sc=0
   40  DIM  x(608)
   45  FOR  a=1  TO  608
   50  LET  b=RND*1
   55  IF  b<.3  THEN  LET  x(a)=61
   60  IF  b>=.3  THEN  LET  x(a)=48
   70  NEXT  a
   80  CLS
   85  FOR  a=1  TO  608
   90  PRINT  CHR$  x(a);
  100  NEXT  a
  110  LET  c=27+RND*100
  120  FOR  n=22527  TO  23135
  130  POKE  n,c
  140  NEXT  n
  160  LET  y=0:  LET  x=19
  200  PRINT  AT  x,y;CHR$  94
  210  LET  ox=x:  LET  oy=y
  230  LET  y=y+(INKEY$="6")-(INKEY
$="5")
  235  PAUSE  15
  240  LET  x=x+(INKEY$="6")-(INKEY
$="7")
  243  GO  SUB  1000
  245  IF  y<=0  THEN  LET  y=0
  247  IF  y>=31  THEN  LET  y=31
  249  IF  x>=20  THEN  LET  x=20
  251  IF  x<=0  THEN  LET  x=0
  260  PRINT  AT  x,y;CHR$  94
```

```
 270 PRINT AT ox,oy;" "
 280 IF x=0 AND y=0 THEN GO TO 3
00
 290 GO TO 200
 300 LET sec=INT (sec/4)
 310 LET min=INT (sec/60)
 350 PRINT AT 20,0; INK 7;"It to
ok you ";min;" minutes and";sec-
(min*60);" seconds to escape"
 360 INPUT '"Another go? (Y or N
)";w$
 370 IF w$="N" OR w$="n" THEN ST
OP
 380 INPUT '"Press A for a new m
aze, or B forthe one just played
.";w$
 390 IF w$="B" OR w$="b" THEN GO
 TO 80
 400 RUN
1000 LET q$=SCREEN$ (x,y)
1010 IF CODE q$=61 THEN LET sec=
sec+1
1020 IF CODE q$=48 THEN LET sec=
sec+25
1050 RETURN
```

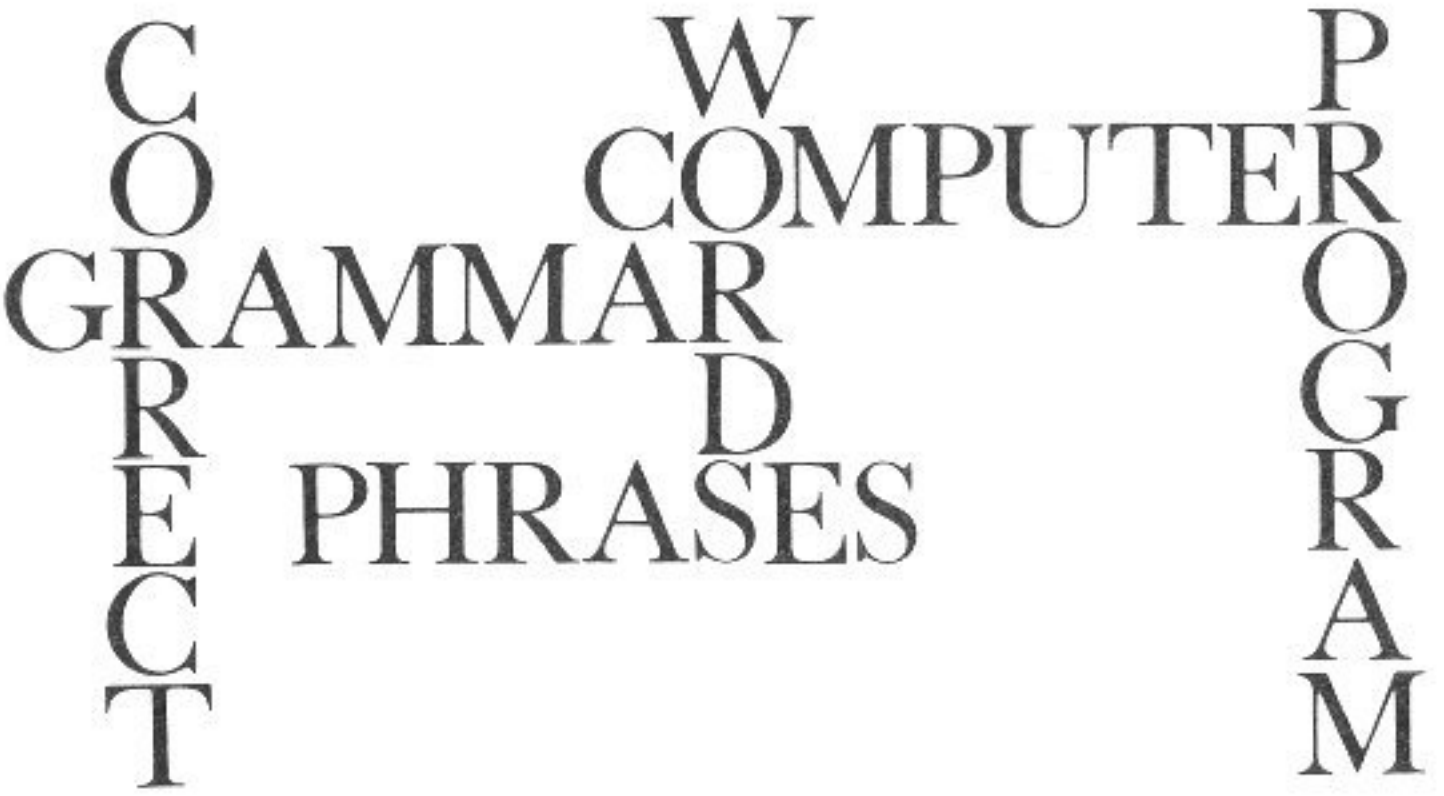
QUIZ

This program stores questions and answers in DATA statements, and presents them in a random order.

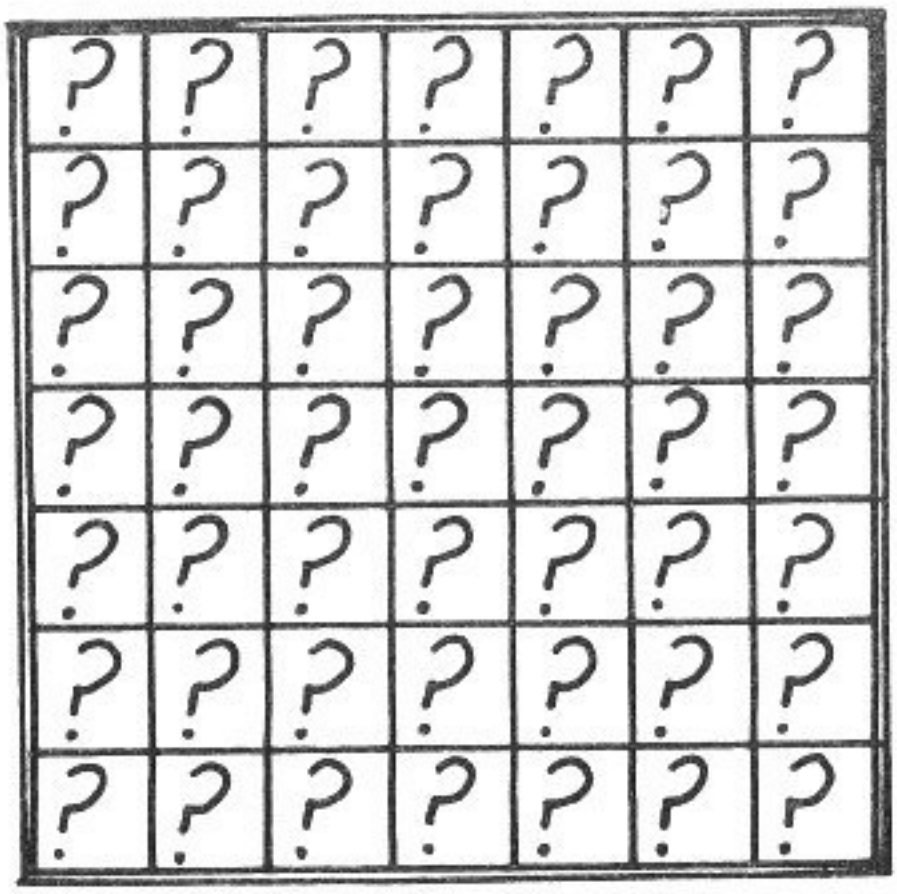
It is easy to add more questions, or to alter the ones provided.

```
    5 INVERSE 0: PAPER 3: BORDER
3: INK 7: CLS
   10 LET R=0
   20 LET SC=0
   50 GO TO 500
   55 LET R=R+1
   57 IF R>6 THEN GO TO 350
   60 PRINT "Question number ";R
   70 PRINT "----------------"
   80 PRINT '''"The answer is a n
umber."
   90 PRINT '0$;"?"
  100 INPUT N
  120 IF N=0 THEN GO TO 300
  130 FOR T=1 TO 30: PRINT INK 0;
"Wrong!";
  160 NEXT T
  170 PAUSE 50
  180 GO TO 50
  300 LET SC=SC+1
  320 PRINT "Correct!"
  330 PAUSE 50
  340 GO TO 50
  350 PAPER 4: BORDER 4: INK 0: C
LS
  360 PRINT AT 10,10;"Your score
was ";SC: PAUSE 50
  450 STOP
  500 FOR N=1 TO INT (RND*11)+5
  510 READ Q$,Q
  520 NEXT N: CLS : RESTORE
  530 GO TO 55
  550 DATA "Year of Tchaikovsky'
Overture",1812
  560 DATA "No. of wings on a bi
plane",2
  570 DATA "Prime Minister's resi
dence",10
  580 DATA "Start of World War II
",1939
  590 DATA "Virginia Wade won Wim
bledon",1977
```

```
  600 DATA "Atmospheric Pressure
 in pounds",44
  610 DATA "27 x 16",402
  620 DATA "Baker's Dozen",13
  630 DATA "John Buchan's 1- Ste
p",39
  640 DATA "Cross",144
  650 DATA "Degrees in a triangl
e",180
  660 DATA "Year of Churchill's d
eath",1965
  670 DATA "How many points for b
ue in sno-ker",5
  680 DATA "Orwell's year of des
tiny",1984
  690 DATA "Percentage U.R.T.",45
  700 DATA "Britain's newest TV c
hannel",4
  710 DATA "Number of MP's in Par
liament",650
  720 DATA "Price of a 1st class
 stamp",16
  730 DATA "Year of first VW Bee
tle",1939
  740 DATA "1729-956",783
  750 DATA "No. of countries in E
EC",11
  760 DATA "Year of Queen's coron
ation",1953
```



```
  770 DATA "Pounds in a stone",14
  780 DATA "No. of legs on a spid
er",6
  790 DATA "Number of states in U
.S.A.",50
  800 DATA "Number of stripes on
U.S. flag",13
  810 DATA "ASCII code for *",42
  820 DATA "Decimal of binary 11
11110",254
```



TOWERING INFERNO

A 30-storey skyscraper is on fire and you, as Fire Chief, must save the ten people who are still trapped in the burning building.

As they jump from the windows, you must catch them in your blanket.

To move the blanket, press 2 for left and 5 for right.

TOWERING INFERNO

```
    1 INVERSE 0: BORDER 0: PAPER
1
    2 INK 7: CLS
    3 GO SUB 900
    4 GO SUB 960
    5 LET P=0: LET S=P
    6 LET F=1: LET G=9
   10 LET E=10: CLS
   12 FOR X=2 TO 20
   15 INVERSE 1: PRINT AT X,0;"
      "
   18 NEXT X: INVERSE 0
   19 FOR x=0 TO 31: PRINT AT 21,
x;CHR$ 131;: NEXT x
   23 FOR X=3 TO 16 STEP 2
   24 PRINT AT X,1;" ";AT X,3;" "
;AT X,5;" ";AT X,7;" "
   28 NEXT X
   50 PRINT AT 20,E-1;CHR$ 131;CH
R$ 131
   51 FOR T=1 TO 8: NEXT T
   53 PRINT AT F,G;"A"
   59 PRINT AT 20,E-1;"   "
   60 IF INKEY$="5" THEN LET E=E+
1
   62 IF E>31 THEN LET E=31
   70 IF INKEY$="2" THEN LET E=E-
1
   72 IF E<10 THEN LET E=10
   85 PRINT AT F,G;" "
   90 LET F=F+1
   94 IF F>18 THEN GO TO 105
  100 LET G=G+INT (RND*4)-1
  102 IF G<10 THEN LET G=10
  105 IF F=21 THEN GO TO 165
  110 GO TO 50
  165 LET P=P+1
  170 IF G=E OR G=E-1 THEN GO TO
300
  179 PRINT AT 21,G;CHR$ 140
  180 FOR R=1 TO 40: INVERSE 1: P
RINT AT 7,16;"Splat!": INVERSE 0
  182 PRINT AT 7,16;"Splat!": NEX
T R
  185 PRINT AT 9,12;"Score= ";S;"
 out of ";P
  186 PAUSE 200
  187 IF P=10 THEN GO TO 360
  190 CLS : GO TO 6
  300 LET S=S+1
  305 FOR Q=1 TO 50
  310 INVERSE 1: PRINT AT 7,14;"W
ell caught!": INVERSE 0
  315 PRINT AT 7,14;"Well caught!
"
  320 NEXT Q
```

```
  340  GO TO 185
  360  CLS
  370  FOR X=1 TO 21
  380  PRINT "**********************
*********"
  390  NEXT X
  395  FOR K=7 TO 13
  400  PRINT AT K,4;"
                "
  410  NEXT K
  460  PRINT AT 8,5;"Your final sc
ore is ";S
  470  PRINT AT 10,1;"You will be
demoted unless you""                do bet
ter next time."
  490  GO TO 500
  491  FOR K=1 TO 60
  493  INVERSE 1: PRINT AT 10,0;"
                Well done!": INVERSE 0
  495  PRINT AT 10,0;"                 0
ell done!"
  498  NEXT K
  500  PAUSE 250
  510  RUN
  900  CLS
  910  PRINT AT 6,6;"The Towering
Inferno"
  920  PRINT AT 8,9;"2=Left, 5=Rig
ht"
  930  PRINT AT 14,5;"Press any ke
y  to  start"
  940  IF INKEY$="" THEN GO TO 940
  950  RETURN
  960  FOR N=0 TO 7
  970  READ A: POKE USR "a"+N,A
  980  NEXT N
  990  DATA 60,66,66,60,255,24,36,
36
 1000  RETURN
```

RULER OF LOS SINCLAIROS

No, not the 12-inch kind, the kind that rules a country. You are the dictator of a Banana Republic in Central America. The peasants have demanded that you rule for the next 12 years.

The incentives to stay the course are enormous — a peaceful retirement with a large pension! However, managing the country is not as easy as it sounds. Only your grovelling friend, the Spectrum, gives you any information whatsoever.

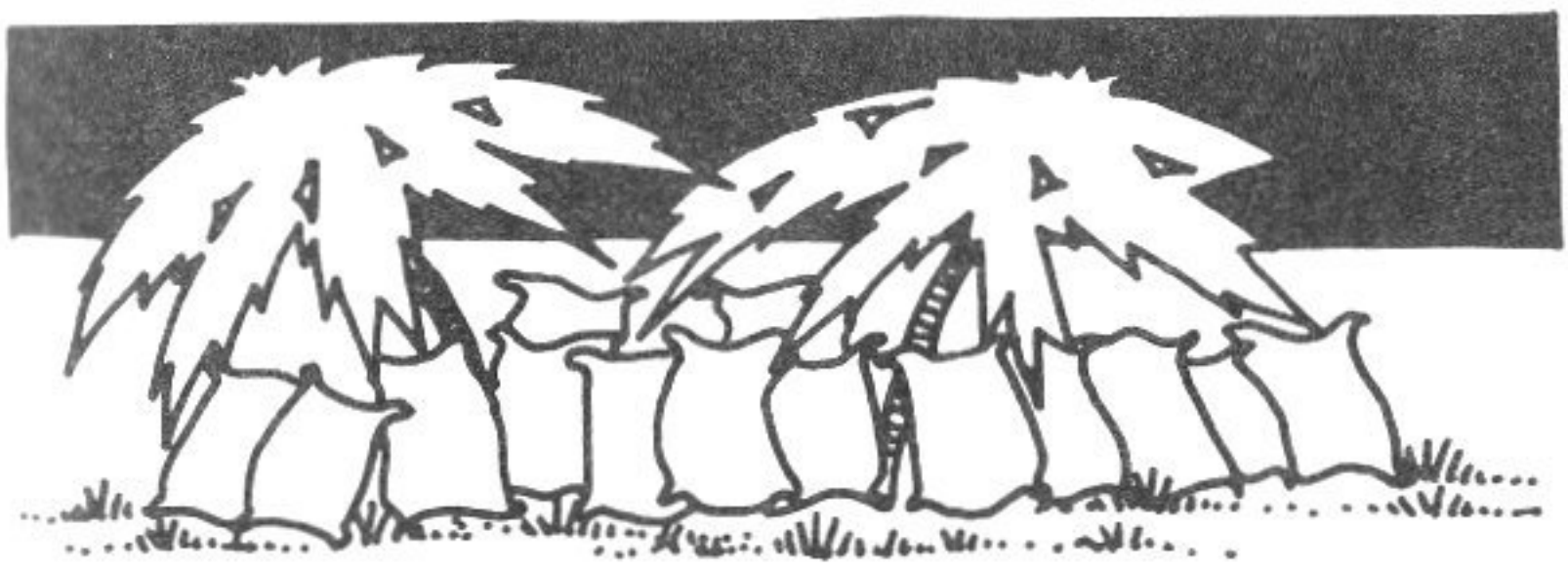
You must try to increase your lands without feeding the peasants poorly; if you starve them, they will rise up in revolution.

Each peasant requires two sacks of corn for nourishment a year. To buy an extra acre of land you need ten sacks of corn.

Only enter the number of extra acres of land needed, not the amount of corn that the extra land will cost.

I will leave the rest of the strategy up to you; there is still a lot to find out. Happy Ruling!

N.B. When the program pauses, press a key to continue.



```
    5 REM *When program pauses
press  anykey *
   10 INVERSE 0: PAPER 1: INK 7:
BORDER 1
   15 CLS
   30 LET c=1000
   35 LET a=100
   40 LET r=500
   45 LET n=1
   50 LET l$="**** *   *   *    ****
   ****          *  * *   *   *
   *   *         *** *   *   *   ***
   ***           *  * *   *   *
   *   *         *  *  **** ***  ****
   *   *"
   52 CLS
   55 PRINT l$
   60 PAUSE 4e4
   70 CLS
  100 PRINT "In year ";n;" of ou
  glorious rul-  er's reign"
  120 PRINT '"You have oh Master,
"
  130 PRINT ,"   a population of
  ";r;" loyal    subjects,"
  140 PRINT ,"   a corn store of
  ";c;" sacks"
  150 PRINT ,,"   and ";a;" acres
  of land."
  190 PAUSE 300
  200 CLS
  210 PRINT "How much extra land,
  Master?"
  220 INPUT L
  225 LET a=a+L
  230 LET c=-L*10
  240 IF a<0 THEN GO TO 1000
  250 PRINT '"How many sacks as f
ood, Master?"
  260 INPUT f
  262 IF L=0 AND f=r THEN PRINT '
"Do something constructive.": GO
  TO 190
  265 IF f>r+100 THEN PRINT ''"Yo
ur peasants will be very fat.":
PAUSE 80
  270 LET c=c-2*f
  280 IF a<0 THEN GO TO 1000
  290 IF r-f>r/4 THEN GO TO 1500
  300 LET r=r+(f-r)*RND
  310 IF r<50 THEN GO TO 1000
  315 LET r=INT (r)
  320 LET c=c+INT (r*a/50)
  330 LET n=n+1
  340 IF n=12 THEN GO TO 500
  345 IF RND>.7 THEN GO SUB 705
```

```
 350  GO TO 52
 500  CLS
 520  FOR g=1 TO 91
 530  PRINT "******You did it oh
Master******"
 540  IF g/18=INT (g/18) THEN CLS

 550  NEXT g
 555  PRINT '\$
 560  PRINT '''"Well done! The loy
al subjects ofLos  Sinclairos ar
e very happy."
 570  PRINT '"They will give you
a peaceful   retirement,leaving
you alone"
 580  PRINT "with you Spectrum.
I hope you  are happy."
 590  PAUSE 4e4
 600  CLS
 610  PRINT '"Oh, before you go,
I thought you would like to know
 that"
 620  PRINT "You and your peasant
s harvested a total of ";c;" sac
ks of corn."
 630  PRINT AT 10,4;"Press any ke
y to start."
 635  IF INKEY$<>"" THEN RUN
 640  PRINT AT 17,8;"another game
"
 650  PAUSE 5
 660  INVERSE 1: PRINT AT 17,8;"a
nother game": INVERSE 0
 670  PAUSE 5
 680  PRINT AT 17,8;"another game
"
 690  PAUSE 5
 700  GO TO 635
 705  PRINT '''"        Bad harvest,
 oh Master..."
 710  LET c=c-INT (RND*(c/4))
 720  LET c=c+a
 730  PRINT '"Press any key to re
turn"
 740  IF INKEY$="" THEN GO TO 740
 750  RETURN
1000  CLS
1010  PRINT "       You faited,you
 greedy  barbarian"
1020  PRINT "       and were overt
hrown in a  violent revolution."
1030  PRINT "       You will be e
xecuted at  sunrise tomorrow."
1050  PAUSE 100
1070  PRINT '''"The people of Los
 Sinclairos    have decided that
```

```
 leaving their "
1080 PRINT "country's economy in
 the hands  of ignorant fools is
 a bad thing"
1090 PRINT "and they have theref
ore decided to let  computer loo
k after"
1100 PRINT "their affairs..."
1110 PAUSE 4e4
1120 PRINT "A ZX Spectrum, of co
urse!"
1130 STOP
1500 CLS
1505 FOR t=1 TO 30
1510 PRINT AT 10,11; INK 5; PAPE
R 7;"REVOLUTION"
1515 BORDER 5
1520 PRINT AT 10,11;"revolution"
1525 BORDER 1
1530 NEXT t
1540 PRINT '''"Press a key to sur
vive."
1542 IF INKEY$="" THEN GO TO 154
2
1545 PRINT ''
1550 FOR t=1 TO 16
1560 PRINT ".";
1570 PAUSE 20
1580 NEXT t
1600 IF RND<.25 THEN GO TO 1000
1605 LET d=INT (RND*(3*n))+10
1615 LET e=INT (RND*(8*n))+40
1620 CLS
1630 PRINT "%%%%%%%Revolution st
opped%%%%%%%"
1640 PRINT '''"But let that be a
lesson to you."
1650 PRINT '"You also lost ";d;"
 acres in"
1660 PRINT "the uprising, as wel
l as ";e
1670 PRINT "sacks of corn burnt
in a fire."
1680 LET r=r-d
1690 LET a=a-e
1700 PAUSE 250
1710 GO TO 300
```

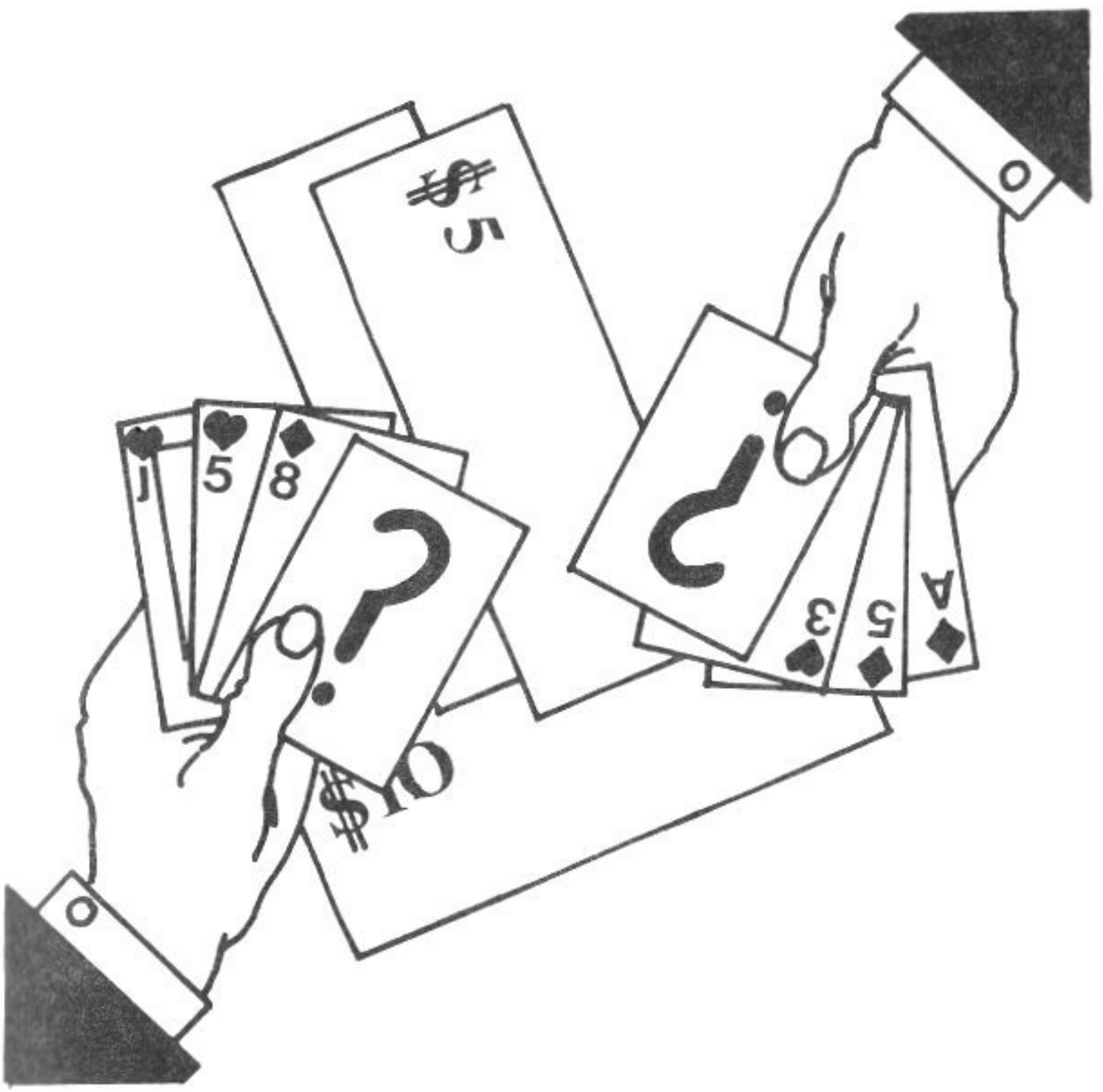
21

This is the Spectrum version of the classic card game played all over the world.

You and the computer take turns in trying to get as close to 21 as possible without exceeding that figure. You can also achieve five-card tricks in this version.

The computer is a difficult, but not unbeatable, opponent who weighs up the situation carefully before making a decision.

This game is extremely friendly to play, with all the instructions contained in the program.



```
    1 GO SUB 5000: INK 1: PAPER 7
    5 INVERSE 0: BRIGHT 1: BORDER
 2: CLS
   10 LET p$="BABABABABABABAB21ABAB
ABABABABABABA"
   20 PRINT INK 2;p$
   25 RANDOMIZE
   26 LET b=0
   28 LET c=0
   30 PRINT '"Please enter your n
ame"
   32 INPUT n$
   33 CLS
   34 PRINT ,,n$
   35 PRINT ,," I will toss a co
in to see who goes first.
                Do you want heads
 or tails?      (press H or T)"
   40 LET a$=INKEY$
   50 IF NOT (a$="h" OR a$="t" OR
 a$="H" OR a$="T") THEN GO TO 40
   60 PRINT ,,a$
   70 LET x=INT (RND*2)
   80 IF x=0 THEN LET b$="Heads"
   90 IF x=1 THEN LET b$="tails"
  100 PRINT 'b$;", ";
  110 LET a=1
  120 IF x=1 AND a$="H" OR x=1 AN
D a$="h" OR x=0 AND a$="T" OR x=
0 AND a$="t" THEN LET a=0
  130 IF a=0 THEN PRINT "I win"
  140 IF a=1 THEN PRINT "You win
";n$;"."
  141 PAUSE 50
  142 LET x=0
  144 LET y=0
  145 FOR f=1 TO 10
  146 PRINT '''"Press any key for
game ";f
  147 PAUSE 190
  148 IF INKEY$="" THEN GO TO 148
  150 IF a=0 THEN GO SUB 2000
  160 IF a=1 OR a=0 AND x>0 THEN
GO SUB 1000
  170 IF a=1 AND y<>0 THEN GO SUB
 2000
  175 IF y=0 THEN LET x=1
  177 IF x=0 THEN LET y=1
  180 IF x>y THEN LET b=b+1
  190 IF x<y THEN LET c=c+1
  200 PRINT '''" Computer: ";n$;"
"
  210 PRINT TAB 5;b;TAB 15;c
  215 LET a=a+(a=0)-(a=1)
  220 NEXT f
  225 PAUSE 100
```

```
  230  PRINT '''"I enjoyed that."
  240  IF b>c THEN PRINT "Maybe yo
u will beat me next time."
  250  IF b<=c THEN PRINT '''"I wi
ll beat you next time."
  260  PRINT '''"Another game? (Y/N
)"
  270  INPUT a$
  280  IF a$="y" OR a$="Y" THEN RU
N
  290  PRINT '''"Bye then."
  300  STOP
 1000  PRINT '''  Your cards have
been dealt.    Press any key to t
urn them over."
 1010  IF INKEY$="" THEN GO TO 101
0
 1020  GO SUB 3000
 1025  PAUSE 75: CLS
 1026  IF p=1 THEN GO SUB 3500
 1030  PRINT '''"     ";p'
 1040  LET y=p
 1045  LET L=1
 1050  PAUSE 25: GO SUB 3000
 1051  IF p=1 THEN GO SUB 3500
 1055  IF L=3 THEN CLS
 1060  PRINT '''"     ";p
 1070  LET y=y+p
 1075  LET L=L+1
 1080  PRINT ,,L;" cards.  Total="
;y
 1090  POKE 23692,255
 1105  IF L=5 AND y<=21 THEN GO TO
 1200
 1106  IF y>21 THEN GO TO 1400
 1108  IF y=21 THEN GO TO 1600
 1110  PRINT ,,"Twist or stick?  (
Press T or S)"
 1120  LET a$=INKEY$
 1125  IF NOT (a$="T" OR a$="t" OR
 a$="S" OR a$="s") THEN GO TO 11
20
 1130  PRINT ,,"    ";a$
 1135  PAUSE 75
 1140  IF a$="t" OR a$="T" THEN GO
 TO 1050
 1150  CLS
 1160  RETURN
 1200  PAUSE 50: CLS
 1205  PRINT "You have a five-card
 trick.      This can only be bea
ten by Pon- toon."
 1208  LET y=22
 1210  RETURN
 1400  PAUSE 50: CLS
 1405  PRINT "Sorry - you have bus
```

```
t.  I  win."
1410 LET y=0
1420 RETURN
1600 CLS
1605 PRINT "Well done - you have
 scored 21."
1608 LET y=21
1610 IF l=2 THEN PRINT ,,p$: LET
 y=23
1620 RETURN
2000 PRINT '"Press any key to de
al me two     cards."
2010 IF INKEY$="" THEN GO TO 201
0
2020 GO SUB 3000
2030 CLS
2040 IF p=1 THEN GO SUB 4000
2050 PRINT ,,"      ";p
2060 LET x=p
2070 LET m=1
2080 GO SUB 3000
2090 IF p=1 THEN GO SUB 4000: PA
USE 50
2100 IF m=3 THEN CLS
2110 PRINT ,,"      ";p
2120 LET x=x+p
2130 LET m=m+1
2140 PRINT ,,m;" cards.  Total="
;x
2145 PRINT "Press any key."
2147 IF INKEY$="" THEN GO TO 214
7
2150 IF m=5 AND x<=21 THEN GO TO
 2400
2160 IF x>21 THEN GO TO 2500
2170 IF x=21 THEN GO TO 2600
2180 IF x>14+INT (RND*3) AND a=0
 THEN GO TO 2700
2190 IF x>14+INT (RND*3) AND RND
>.3 THEN GO TO 2700
2200 PAUSE 75: PRINT ,,"I think
I will twist."
2210 GO TO 2080
2400 CLS
2410 PRINT "I have a five-card t
rick."
2420 IF a=0 THEN PRINT ,,"You mu
st get Pontoon to beat me."
2430 IF a=1 AND y=22 THEN PRINT
"You still beat me though, as yo
ugot Pontoon."
2440 IF a=1 AND y<22 THEN PRINT
"That beats you. I win."
2450 LET x=22
2460 RETURN
2500 CLS
```

```
2510 PRINT "Oh dear - I have bus
t. You win."
2520 LET x=0
2530 RETURN
2600 CLS
2610 PRINT "I have scored 21"
2620 IF m=2 THEN PRINT ,,p$
2630 LET x=21
2640 IF m=2 THEN LET x=23
2650 RETURN
2700 CLS
2710 PRINT " I think I will stic
k."
2720 RETURN
3000 LET p=INT (RND*13)+1
3010 IF p>10 THEN LET p=10
3015 PAUSE 75
3020 RETURN
3500 IF y>10 THEN GO TO 4100
3505 PRINT ,," An Ace. Do you
want it to be high or low? (Pres
s H OR L)"
3510 LET a$=INKEY$
3520 IF NOT (a$="H" OR a$="L" OR
 a$="h" OR a$="l") THEN GO TO 35
10
3525 PRINT ;a$
3530 IF a$="H" OR a$="h" THEN LE
T p=11
3535 PAUSE 100
3540 RETURN
4000 IF x>=7 OR x<=10 THEN LET p
=11
4010 IF x<7 OR x>10 THEN LET p=1
4020 PRINT ;" I have an Ace."
4030 IF p=1 THEN PRINT "I think
I will make it low."
4040 IF p=11 THEN PRINT "I think
 I will make it high."
4050 RETURN
4100 PRINT ;" An Ace. It had bet
ter be low or you will bust."
4120 RETURN
5000 FOR n=USR "a" TO USR "b"+7:
 READ r
5010 POKE n,r: NEXT n
5040 RETURN
5050 DATA 0,36,126,126,60,60,24,
24,0,24,60,126,255,126,60,24
```

MINEFIELD

You must try to escape from the minefield in this strategy game written for the Spectrum by Simon Gould.

Use the cursor keys to move carefully around the screen, but take note of the warning sign.

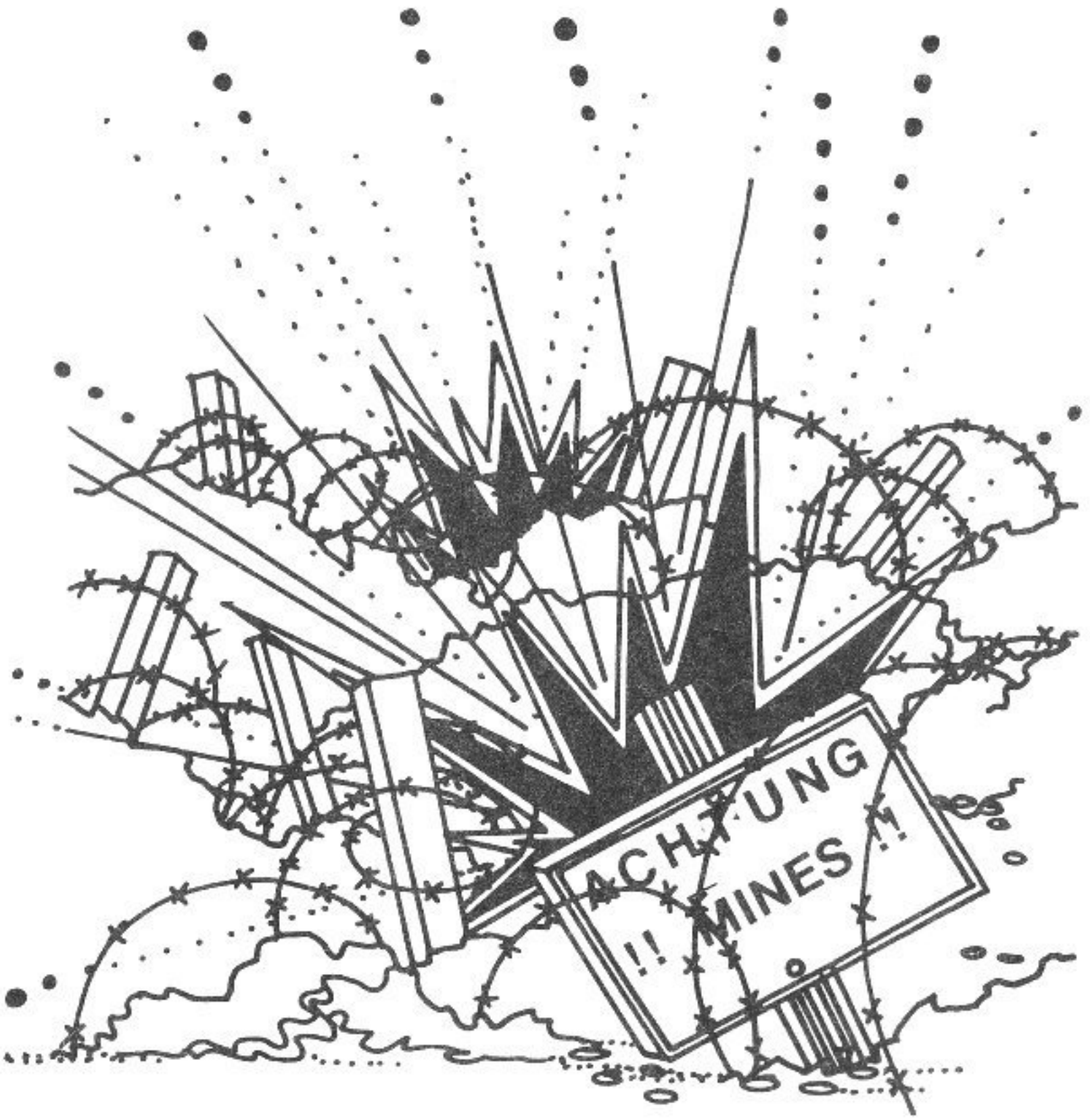
At the beginning of the game, you must enter the number of mines that will be planted in the minefield. Any number less than 35 can be used.

If you tread on a mine you will explode and the locations of the other mines will be shown.

There is a timer and an on-screen scoring feature.

```
  5 INVERSE 0: INK 7: PAPER 2
 10 BORDER 2
 20 LET sc=0
 25 LET A$="o
                  "
 30 RANDOMIZE
 40 CLS
 45 PRINT "Number of mines (2 -
35)";
 50 INPUT mns
 54 IF mns<2 OR mns>35 THEN GO
TO 40
 55 PRINT "=";mns
 60 DIM m(mns,2)
 70 FOR n=1 TO mns
 80 LET m(n,1)=INT (RND*20)+1
 90 LET m(n,2)=INT (RND*32)
100 NEXT n
105 CLS
110 PRINT AT 21,0; INK 5;"Time
=                    SCORE=";sc
120 LET t=0
130 LET x=19
140 LET y=INT (RND*28)
150 PRINT AT x-2,y-1;a$
160 LET t=t+1
170 PRINT AT 21,6;t
180 IF t=100 THEN GO TO 1000
190 GO SUB 300
191 LET a=x
192 LET b=y
195 LET b$=INKEY$
196 IF b$="" THEN GO TO 150
200 LET x=x+(b$="6" AND x<23)-(
b$="7" AND x>3)
210 LET y=y+(b$="8" AND y<30)-(
b$="5" AND y>0)
220 PRINT AT a-2,b-1;"
                  "
230 GO TO 150
300 PRINT AT 0,11; BRIGHT 1; IN
K 6;"     EXIT     "
310 IF x=3 AND y>11 AND y<20 TH
EN GO TO 2000
320 LET n=1
325 IF x>=20 THEN LET x=x-1: PR
INT AT x+1,y-1;"     "
330 LET xc=m(n,1)-x
340 LET yc=m(n,2)-y
350 IF ABS xc<3 AND ABS yc<3 TH
EN PRINT AT 0,11;"WARNING"
360 IF xc=0 AND yc=0 THEN GO TO
1000
365 LET n=n+1
370 IF n=mns THEN RETURN
```

```
  380 GO TO 330
 1000 PRINT AT x-2,y-1; FLASH 1;"
    ";AT x-2,y;"    ";AT x,y+1;"
"
 1010 FOR n=1 TO mns
 1020 PRINT AT m(n,1),m(n,2);"*"
 1030 NEXT n
 1040 FOR n=0 TO 100
 1050 NEXT n
 1060 RUN
 2000 LET sc=sc+100
 2010 PRINT AT 10,5;"Mission Succ
essful"
 2020 PRINT AT 12,5;"Stand by for
  the next field"
 2030 FOR N=0 TO 100: NEXT N
 2050 GO TO 105
```



DROID

Here is your chance to program a real robot, a small droid, to reach the bottom of the maze. You must enter the correct commands for the droid.

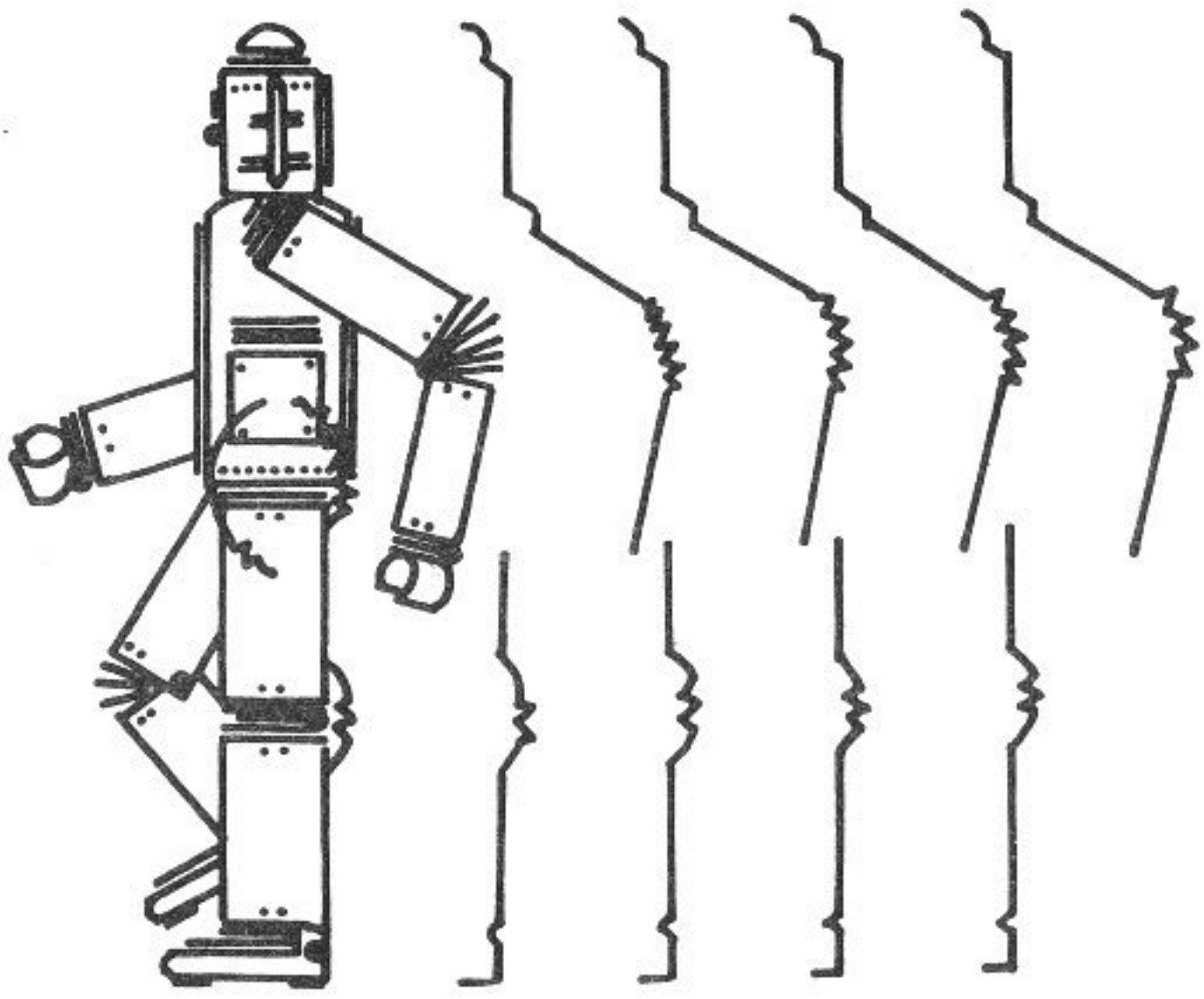
Use l for left, r for right, u for up and d for down. Every instruction you give the droid will move him one space in the appropriate direction. If you make any mistakes in the initial entry, just type Ø to rub out.

You can have a maximum of 60 instructions. The droid's memory is very small and after every 20 instructions the screen clears so you cannot see what you have previously entered.

You must rely on your mental dexterity and memory to get the droid out of the maze safely.

When you have finished typing in your instructions, press the 9 key and watch him move. If he hits a wall along the way, he will be destroyed.

Happy programming!



```
  5 DIM s$(60): GO SUB 9000
 10 INVERSE 0: CLS
 15 RANDOMIZE
 20 GO SUB 1000
 22 LET sc=0
 24 LET q=0
 25 PRINT AT 19,5;">("
 30 FOR n=1 TO 20
 40 IF INKEY$<>"" THEN GO TO 40
 42 LET a$=INKEY$
 43 IF a$="9" THEN GO TO 90
 44 IF a$="0" THEN GO TO 500
 46 IF a$<>"l" AND a$<>"r" AND
a$<>"u" AND a$<>"d" THEN GO TO 4
00
 50 LET s$(q+n)=a$
 60 PRINT AT 19,5+n;s$(q+n);"("
 70 NEXT n
 80 PRINT AT 19,5;"
                  "
 85 LET q=q+21
 86 IF q=60 THEN GO TO 2150
 88 GO TO 25
 90 PRINT AT 19,5;" ";AT 19,5+n
;" "
 92 LET q=0
 93 PRINT AT 21,11;"Score= 0"
 94 PRINT AT 19,6;s$(q+1 TO q+2
0)
 100 LET n=1
 105 IF s$(q+n)=" " THEN GO TO 2
00
 110 LET x=x+(s$(n+q)="r")-(s$(n
+q)="l")-32*(s$(n+q)="u")+32*(s$
(n+q)="d")
 120 IF s$(n+q)="d" THEN LET sc=
sc+6
 122 LET sc=sc-(sc<>0)
 125 PRINT AT 21,19;sc
 130 PRINT AT 18,4+n;CHR$ 131;CH
R$ 143;AT 20,4+n;CHR$ 32;CHR$ 13
1
 140 LET c=INT (x/32): LET v=x-(
c*32)
 145 IF SCREEN$ (c,v)<>" " THEN
GO TO 2000
 150 PRINT AT c,v;"o"
 155 IF c=17 THEN GO TO 3000
 160 LET n=n+1
 165 IF INKEY$<>"" THEN GO TO 16
5
 170 IF n<>20 THEN GO TO 105
 175 LET q=q+21
 180 PRINT AT 18,25;CHR$ 131;AT
20,25;" "
 190 GO TO 94
```

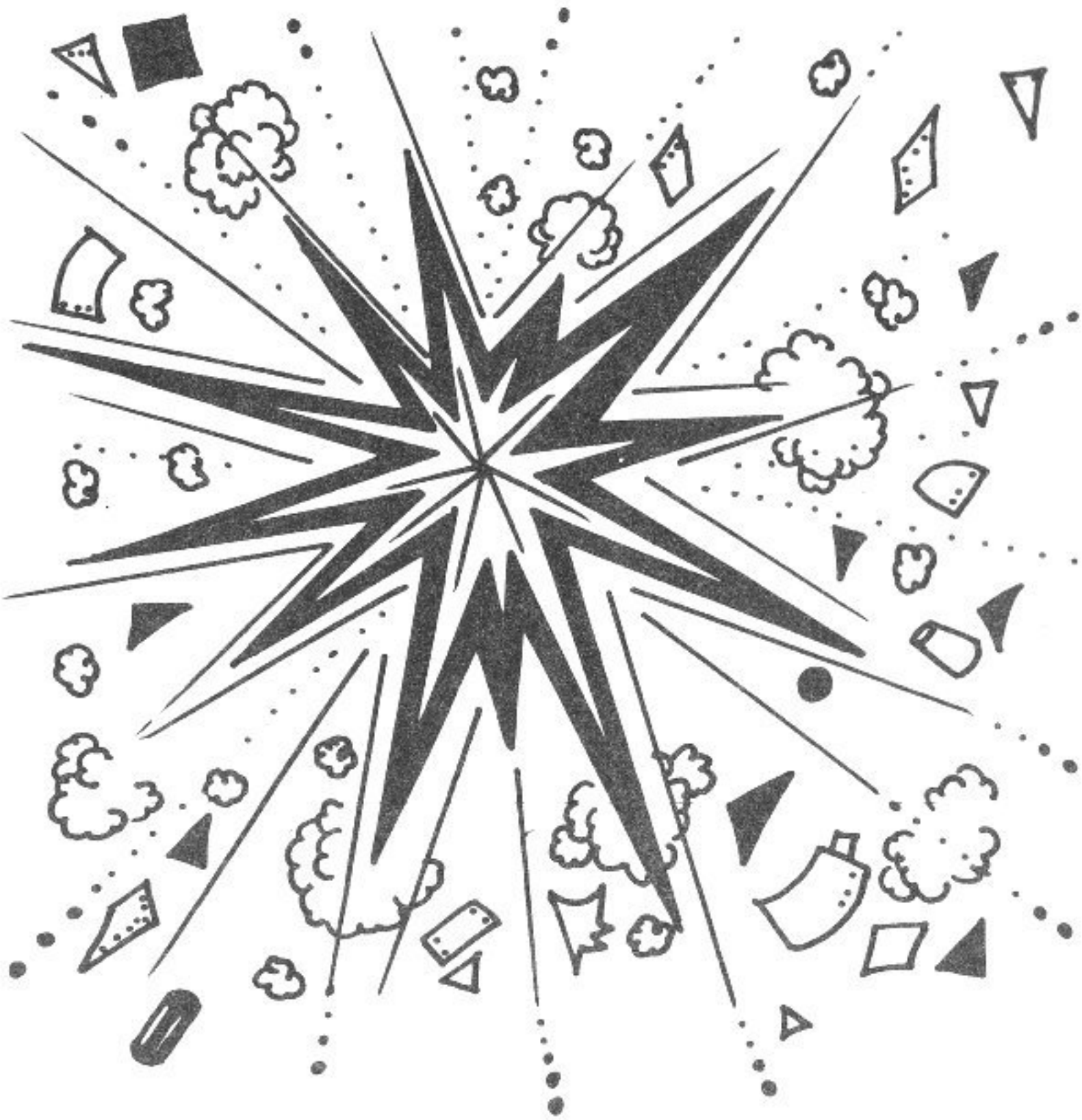
```
  200 PAUSE 100
  220 CLS
  230 PRINT AT 8,2;"You did not g
ive the robot"
  240 PRINT AT 10,2;"enough infor
mation to reach"
  250 PRINT AT 12,5;"the base of
the screen."
  255 PRINT AT 14,7;"Your score w
as ";sc
  260 PRINT AT 16,5;"Another game
? (y/n)"
  270 LET a$=INKEY$
  280 IF a$="" OR a$<>"y" AND a$<
>"n" THEN GO TO 270
  290 IF a$="y" THEN RUN
  300 STOP
  500 IF n=1 THEN GO TO 40
  502 LET n=n-1
  505 PRINT AT 19,5+n;"<"
  510 GO TO 40
 1000 FOR n=0 TO 31: PRINT AT 0,n
;CHR$ 140;: NEXT n
 1010 BRIGHT 1: PRINT AT 20,0; PA
PER 0; INK 6;CHR$ 144;: FOR a=2
TO 31: PRINT PAPER 0; INK 6;CHR$
 145;: NEXT a: PRINT PAPER 0; IN
K 6;CHR$ 146: BRIGHT 0
 1020 FOR n=1 TO 15 STEP 2
 1030 PRINT AT n,0;CHR$ 133;AT n,
31;CHR$ 138
 1040 PRINT AT n+1,0; PAPER 1; IN
K 1;"xxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxx"
 1050 NEXT n
 1060 PRINT AT 17,0;CHR$ 133;AT 1
7,31;CHR$ 138
 1070 FOR n=2 TO 16 STEP 2
 1080 LET y=1
 1090 LET y=y+INT (RND*15)+3*(y<>
1)
 1100 LET z=INT (RND*2)
 1110 IF y>29 THEN GO TO 1140
 1120 PRINT AT n,y;(" " AND z=0)+
("  " AND z=1)
 1130 GO TO 1090
 1140 NEXT n
 1150 LET x=80
 1160 LET c=INT (x/32): LET v=x-(
c*32)
 1170 PRINT AT c,v;"o"
 1180 RETURN
 2000 LET c=INT (x/32): LET v=x-(
c*32)
 2002 IF SCREEN$ (c,v)=CHR$ 140 T
HEN GO TO 210
```

```
2005 FOR n=1 TO 10
2010 PRINT AT c,v;CHR$ 133
2015 PAUSE 5
2020 PRINT AT c,v;CHR$ 131
2025 PAUSE 5
2030 PRINT AT c,v;CHR$ 138
2035 PAUSE 5
2040 PRINT AT c,v;CHR$ 140
2045 PAUSE 5
2050 NEXT n
2060 CLS
2070 PRINT AT 8,2;"The robot has
 hit a wall and"
2080 PRINT AT 10,9;"is now in bi
ts."
2090 GO TO 255
2100 LET L=q+n
2120 PRINT AT 8,3;"Your robot go
t through the"
2130 PRINT AT 10,8;"maze in ";L;
" moves."
2140 GO TO 255
2150 CLS
2160 PRINT AT 8,1;"You may burn
out the robot's"
2170 PRINT AT 10,13;"memory."
2180 GO TO 260
3000 CLS : PRINT '''TAB 11;"WEL
L DONE"
3010 PRINT 'TAB 5;"YOUR ROBOT HA
S ESCAPED"
3020 GO TO 2100
9000 FOR a=USR "a" TO USR "c"+7
9010 READ us: POKE a,us: NEXT a
9020 RETURN
9030 DATA 63,127,106,245,234,117
,127,63
9040 DATA 255,255,170,85,170,85,
255,255
9050 DATA 252,254,174,87,175,86,
254,252
```

APOLLO

In this program, supplied by Peter Shaw, you must take control of the Apollo 24 as it falls to Earth. To do this you must apply thrust using the 1-9 keys. Key 1 will apply the least amount of thrust, whereas 9 will give you the maximum power.

When you hit the ground you will be informed of your performance. (It is sometimes better to crash just to see the animation!)



```
  10 REM APOLLO
  20 REM
  30 PRINT AT 10,10;"PLEASE WAIT
"
  40 GO SUB 2130
  50 GO SUB 2370
  60 BORDER 0: PAPER 0: INK 5: C
LS
  70 LET ch=1
  80 CLS
  90 FOR f=1 TO 25: PRINT AT 8,1
2;"APOLLO";: BEEP .05,F: PRINT A
T 8,12; INVERSE 1;"APOLLO": BEEP
 .05,-F: NEXT f
 100 PRINT AT 8,12; FLASH 1;"APO
LLO"
 110 PRINT AT 3,0;"Do you want i
nstructions ? (y/n)"
 120 POKE 23658,8
 130 LET i$=INKEY$: IF i$="" THE
N GO TO 120
 140 IF i$="Y" THEN GO TO 1800
 150 IF I$<>"N" THEN GO TO 130
 160 CLS
 170 RANDOMIZE : LET E=0: LET G=
0
 180 LET R=INT (SQR (RND*5000))
 190 LET X=31: LET S=0: LET V=10
0: LET U=500: LET H=1000: LET M=
2.5
 200 LET M$=CHR$ 130: LET M$=M$+
M$+M$+M$+M$
 210 LET B$=CHR$ 138
 220 LET A$=B$+M$+B$+M$+B$+M$+B$
+M$+B$+M$+B$
 230 PRINT AT 0,1; INK 7;"Time=0
00";AT 3,5; INK 4;"0";TAB 16;"FU
EL";TAB 29;"500";TAB 5; INK 6;A$
;AT 8,5; INK 5;"0";TAB 15;"HEIGH
T";TAB 26;"1000";TAB 5; INK 6;A$
;AT 13,5; INK 3;"0";TAB 13;"FINE
 HEIGHT";TAB 29;"100";AT 14,5; I
NK 6;A$;AT 18,16; INK 2;"SPEED";
TAB 5; INVERSE 1;"DOWN"; INVERSE
 0;" ";A$(2 TO 19);" "; INVERSE
1;"UP"
 240 FOR I=40 TO 240 STEP 4
 250 PLOT I,32*4
 260 BEEP .005,I/5
 270 PRINT AT 0,14; FLASH 1;"RE-
FUELING"
 280 NEXT I
 290 PRINT AT 0,14;"                "
 300 PRINT AT 0,0;#1;"HIT ""0""
TO DROP OUT OF ORBIT...."
 310 PAUSE 0
```

```
 320  INPUT ""
 330  IF H<=1000 THEN PLOT (H/20+
10)*4,22*4
 340  IF H>1000 THEN PRINT AT 10,
31; INK 6; INVERSE 1;">"
 350  IF H<=100 THEN PLOT (H/2+10
)*4,12*4
 360  IF S=R THEN GO SUB 2040
 370  IF E=1 THEN LET Q=Q-1
 380  IF Q=0 AND E=1 THEN GO SUB
2000
 390  LET F$=INKEY$: IF F$<"0" OR
 F$>"9" OR E=1 THEN LET F$="0"
 400  LET F=2*VAL F$
 410  IF F>U THEN LET F=U
 420  BEEP .1,U/15: BEEP .005,F*2
 430  PRINT AT 0,7; INK 7;S;" "
 440  IF F<>0 THEN GO SUB 2110
 450  LET U1=INT (U/10+10): LET U
=U-F
 460  LET U2=INT (U/10+11)
 470  FOR P=U1 TO U2 STEP -1
 480  PLOT OVER 1;P*4,32*4
 490  NEXT P
 500  PRINT AT 3,21;("LOW" AND U<
=50);AT 3,21;("OUT" AND U=0)
 510  LET V=V-F/M+1
 520  LET S=S+1
 530  PLOT OVER 1;X*4,8
 540  LET X=36-V/LN (2+ABS V)
 550  PLOT X*4,8
 560  PRINT AT 10,31;" "
 570  IF H<=1000 THEN PLOT OVER 1
;(H/20+10)*4,22*4
 580  IF H<=100 THEN PLOT OVER 1;
(H/2+10)*4,12*4
 590  LET H=H-V/2
 600  IF H>0 THEN GO TO 330
 610  CLS
 620  PRINT "Speed ";.1*INT (10*V
);" m/s, Time ";S;" secs"
 630  PRINT '"Fuel remaining ";.
1*INT (2*U);"%"''''
 640  LET D=INT ((V-0.1)/5)
 650  IF D<0 THEN LET D=0
 660  IF D=0 THEN PRINT AT 5,5;"S
AFE LANDING.";AT 6,5;"_________
_____"
 670  IF D=1 THEN PRINT AT 5,5;"B
UMPY LANDING.";AT 6,5;"_________
_____"
 680  IF D=2 THEN PRINT AT 5,5;"D
ANGEROUS LANDING.";AT 6,5;"_____
_____________"
 690  IF D=3 THEN PRINT AT 5,5;"C
RASH LANDING.";AT 6,5;"_________
```

```
        "
 700 IF D=4 THEN PRINT AT 5,5;"C
ATASTROPHIC LANDING.";AT 6,5;"___
                              "
 710 IF D>10 THEN GO TO 1190
 720 IF D>6 THEN PRINT AT 7,1;"Y
OU HAVE MADE A ";INT (V/2);" ft
CRATER";TAB 5;"IN THE MOON'S SUR
FACE."
 730 GO SUB 1690
 740 IF D=0 THEN GO TO 1380
 750 PRINT AT 12,14;" ";TAB 14;"
 "
 760 FOR Z=12 TO 20
 770 PRINT AT Z-1,12;"  "
 780 PRINT AT Z,12;"}/"
 790 NEXT Z
 800 IF D<>1 AND D<>2 THEN GO TO
 830
 810 PRINT AT 15,5;"**OUCH**"
 820 FOR Z=1 TO 20: NEXT Z
 830 PRINT AT 15,5;"          "
 840 IF D=1 THEN GO TO 1780
 850 LET Y=22
 860 FOR Z=13 TO 20
 870 PRINT AT Z-1,Y-.3;"   ";AT
Z,Y;CHR$ 142;" <"
 880 BEEP .05,-Z
 890 LET Y=Y+.3
 900 NEXT Z
 910 IF D=2 THEN GO TO 1780
 920 LET Z$="                "
 930 LET K=14
 940 PRINT AT 11,K; INK 6;Z$;TAB
 K;"    ";CHR$ 144;CHR$ 140;CHR$
140;CHR$ 145;AT 13,K;"    ";CHR$
146;"  ___ ";CHR$ 147;TAB K;"    ";CH
R$ 148;CHR$ 133;CHR$ 138;CHR$ 14
9;TAB K;"    ";CHR$ 150;CHR$ 140;
CHR$ 140;CHR$ 151;;TAB K;"    ";CH
R$ 152;CHR$ 153;"    ";CHR$ 154;CH
R$ 155;TAB K;"  ";CHR$ 156;CHR$ 1
57;CHR$ 158;CHR$ 140;CHR$ 140;CH
R$ 159;CHR$ 160;CHR$ 161;TAB K;"
  ";CHR$ 162;"    ";CHR$ 163;CHR$ 1
64;"    ";CHR$ 162
 950 BEEP 1,-9: BEEP .5,-3: BEEP
 1,-9
 960 INK 6
 970 IF D=3 THEN GO TO 1780
 980 PRINT AT 18,K+1;CHR$ 151
 990 BEEP .5,-4
1000 PRINT AT 18,K+4;CHR$ 152;CH
R$ 155
1010 BEEP .5,-6
1020 PRINT AT 18,K+8;CHR$ 150
```

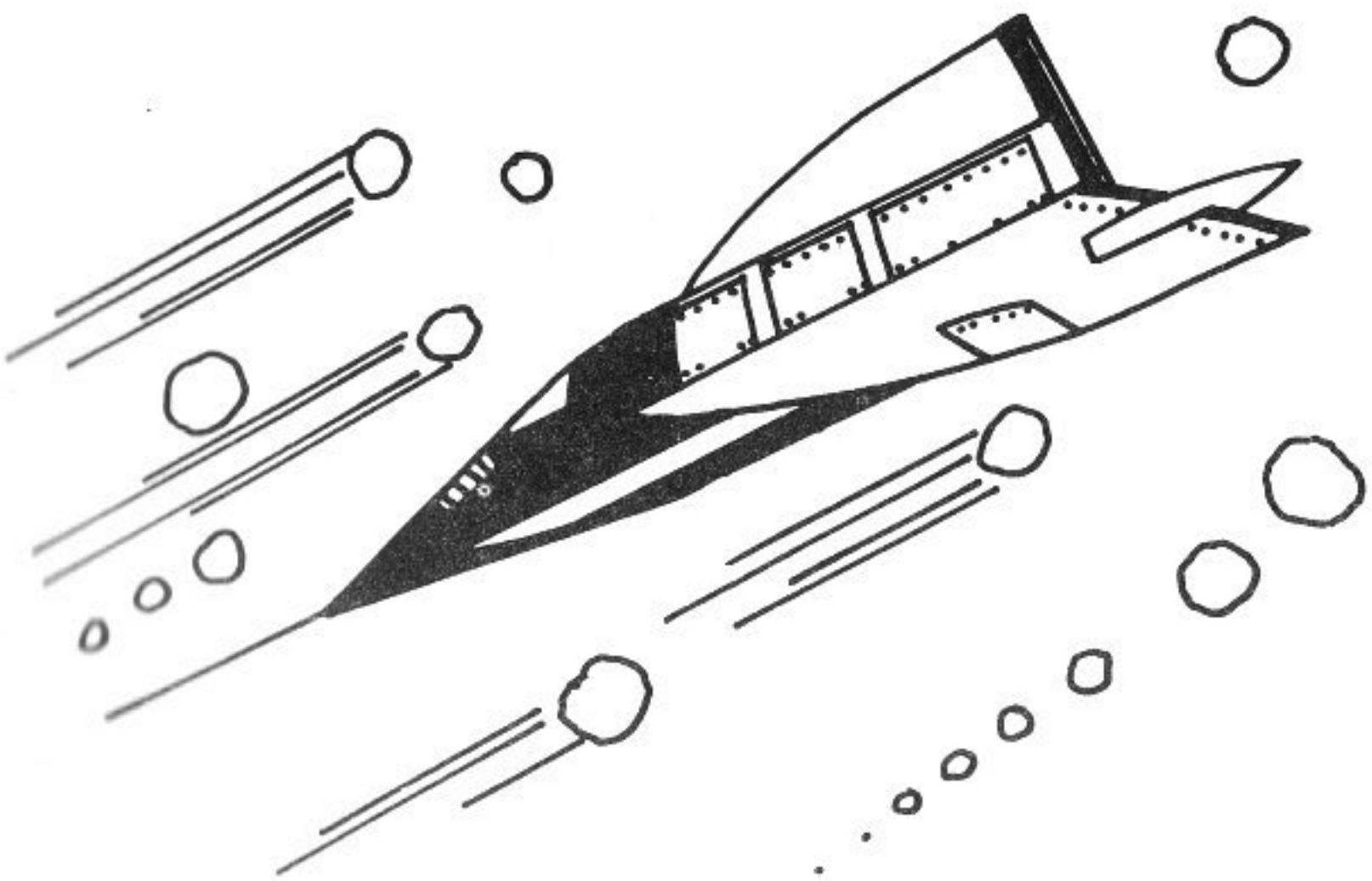
```
1030 BEEP .5,-10
1040 IF D<3 THEN INK 5: GO TO 17
80
1050 FOR O=12 TO 18: LET NO=0
1060 PRINT AT NO,K;Z$;TAB K;(S$(
1) AND NO+1<18);AT NO+2,K;(S$(2)
 AND NO+2<18);TAB K;(S$(3) AND N
O+3<18);TAB K;(S$(4) AND NO+4<18
);TAB K;(S$(5) AND NO+5<18);TAB
K;(S$(6) AND NO+6<18)
1070 BEEP O/10,-O
1080 PRINT AT 18,10;CHR$ 151;CHR
$ 153;AT 18,30;(CHR$ 144+CHR$ 14
9 AND NO>13);AT 18,14;(CHR$ 151+
CHR$ 154+CHR$ 155 AND NO>14);AT
18,4;(CHR$ 147+CHR$ 145+CHR$ 151
 AND NO>15);AT 18,17;(CHR$ 163+C
HR$ 151+CHR$ 153+CHR$ 154+CHR$ 1
56+CHR$ 151+CHR$ 145 AND NO>16)
1090 NEXT O
1100 INK 5
1110 PAUSE 40
1120 FOR Z=1 TO 5
1130 PRINT AT 16,16;"R.I.P."
1140 PAUSE 10
1150 PRINT AT 16,16;"          "
1160 PAUSE 10
1170 NEXT Z
1180 GO TO 160
1190 PRINT AT 21,4;CHR$ 141;  INV
ERSE 1;"                ";  INV
ERSE 0;CHR$ 142
1200 PRINT AT 20,5;CHR$ 141;  INV
ERSE 1;"              ";  INVER
SE 0;CHR$ 142
1210 PRINT AT 19,6;CHR$ 141;  INV
ERSE 1;"            ";  INVERSE
 0;CHR$ 142
1220 PRINT AT 18,8;CHR$ 141;  INV
ERSE 1;"          ";  INVERSE 0;C
HR$ 142
1230 FOR Z=13 TO 18
1240 PRINT AT Z,13;":"
1250 PAUSE 6
1260 NEXT Z
1270 FOR Z=1 TO 10
1280 PRINT AT 19,13;  INVERSE 1;"
*";AT 19,13;  INVERSE 0;"*"
1290 NEXT Z
1300 PRINT AT 21,4;CHR$ 141;  INV
ERSE 1;"                ";  INV
ERSE 0;CHR$ 142
1310 PRINT AT 20,5;CHR$ 141;  INV
ERSE 1;"              ";  INVER
SE 0;CHR$ 142
1320 PRINT AT 19,6;CHR$ 141;  INV
```

```
ERSE 1;"      ";CHR$ 131;CHR$ 143;
CHR$ 131;"          ";CHR$ 129
1330 PRINT AT 18,8;CHR$ 141; INV
ERSE 1;"   ";CHR$ 131; INVERSE 0;
"      "; INVERSE 1;CHA$ 131;"   ";
CHR$ 129
1340 PRINT AT 7,1;"YOU HAVE MADE
 A ";INT (V/2);"ft CRATER IN THE
 MOON'S SURFACE."
1350 FOR Z=18 TO 13 STEP -1: PRI
NT AT Z,13;" ": PAUSE 10: NEXT Z
1360 PRINT AT 16,11;"R.I.P."
1370 GO TO 1780
1380 FOR Z=1 TO 60: NEXT Z
1390 PRINT AT 13,16;"O"
1400 PRINT AT 13,16; INVERSE 1;"
 ";CHR$ 140
1410 PRINT AT 13,16;CHR$ 131;CHR
$ 138;AT 11,15;" O "
1420 PRINT AT 10,15;" O ";AT 11,
15;CHR$ 131;CHR$ 143;CHR$ 131;AT
 13,16;"  "
1430 PRINT AT 9,15;" O ";AT 10,1
5;CHR$ 131;CHR$ 143;CHR$ 131;AT
11,15;CHR$ 133;CHR$ 131;CHR$ 138
1440 FOR Z=1 TO 6
1450 PRINT AT 9,15;" O ";AT 10,1
5;CHR$ 131;CHR$ 143;CHR$ 131;AT
11,15;CHR$ 133;CHR$ 131;CHR$ 138
1460 PAUSE 4
1470 PRINT AT 9,15;CHR$ 134;"O";
CHR$ 137;AT 10,15;" ";CHR$ 143;"
 ";AT 11,15;CHR$ 137;CHR$ 131;CH
R$ 134
1480 PAUSE 4
1490 NEXT Z
1500 PRINT AT 9,15;" O ";AT 10,1
5;CHR$ 131;CHR$ 143;CHR$ 131;AT
11,15;CHR$ 133;CHR$ 131;CHR$ 138
1510 FOR Z=1 TO 20: NEXT Z
1520 PRINT AT 10,15;CHR$ 129: PA
USE 1: PRINT AT 10,15;" "
1530 FOR Z=11 TO 9 STEP -1
1540 PRINT AT Z,21;CHR$ 133
1550 PAUSE 8
1560 NEXT Z
1570 PAUSE 10
1580 PRINT AT 9,22; INVERSE 1;"
";AT 10,22;" "
1590 PAUSE 8
1600 PRINT AT 9,23; INVERSE 1;"
";AT 10,23;" "
1610 PAUSE 8
1620 PRINT AT 9,24; INVERSE 1;"
";AT 10,24;" "
1630 IF RND>.5 THEN GO TO 1870
```

```
1640 FOR Z=1 TO 10
1650 PRINT AT 9,15;CHR$ 134;AT 1
0,15;" "
1660 PAUSE 8
1670 NEXT Z
1680 GO TO 160
1690 INK 6: PRINT AT 12,14;S$(1)
1700 PRINT AT 13,14;S$(2)
1710 PRINT AT 14,14;S$(3)
1720 PRINT AT 15,14;S$(4)
1730 PRINT AT 16,14;S$(5)
1740 PRINT AT 17,14;S$(6)
1750 PRINT AT 18,14;S$(7)
1760 PRINT AT 19,10; INK 2;"_____
                "
_____
1770 INK 5: RETURN
1780 FOR A=1 TO 100: NEXT A
1790 GO TO 160
1800 CLS
1810 PRINT INK 6;"                APO
LLO"
1820 PRINT INK 5;"                _____
    "
_____
1830 PRINT INK 6;'"You are the c
ommander of Apollo 24. You must
try to land your    craft on the
lunar surface at a speed of less
 than 5 mps.
                    Skillful mano
uvring is required but computer
reports indicate    that your lan
ding site is good."
1840 PRINT INK 7;'"Use keys 1-9
to give thrust.",,,"At any time
your fuel line couldblock."''"
Press any key to start.."
1850 PAUSE 0
1860 GO TO 160
1870 LET X=22: LET Y=9
1880 FOR Z=1 TO 27
1890 PRINT AT 10,Z;" (**)"
1900 IF Z=3 THEN PRINT AT 8,16;"
?"
1910 IF Z=5 THEN PRINT AT 8,15;"
GULP"
1920 IF Z=6 THEN PRINT AT 8,15;"
    ";AT 9,15;"   ";AT 10,15;" O
  "
1930 IF Z=9 THEN PRINT AT 10,15;
"   ";AT 11,15;" O ";AT 13,16;CH
R$ 131;CHR$ 138
1940 IF Z=16 THEN PRINT AT 8,15;
"PHEW"
1950 IF Z=17 THEN PRINT AT 8,15;
"    ";AT 10,15;" O ";AT 11,15;C
HR$ 131;CHR$ 143:CHR$ 131;AT 13,
16;"  "
```

```
1960 IF Z=18 THEN PRINT AT 9,15;
"  O  ";AT 10,15;CHR$ 131;CHR$ 143
;CHR$ 131;AT 11,15;CHR$ 133;CHR$
 131;CHR$ 138
1970 IF Z=24 THEN PRINT AT 9,21;
"     "
1980 IF Z>24 THEN PRINT AT Y,X;"
   ": LET X=X+0.3: LET Y=Y+1.3
1990 PRINT AT Y,X; INVERSE 1;"
  "
2000 NEXT Z
2010 CLS
2020 PAUSE 40
2030 GO TO 160
2040 LET Q=INT (RND*6)+3
2050 PRINT AT 0,11; INVERSE 1;"F
UEL BLOCK FOR ";Q;" secs"
2060 LET E=1
2070 RETURN
2080 PRINT AT 0,11;"
          ";AT 1,11;"
        "
2090 LET E=0
2100 RETURN
2110 PRINT AT 4,1;"T";AT 6,1;"H"
;AT 8,1;"R";AT 10,1;"U";AT 12,1;
"S";AT 14,1;"T": FOR A=1 TO 4: N
EXT A: PRINT AT 4,1;"  ";AT 6,1;"
  ";AT 8,1;"  ";AT 10,1;"  ";AT 12,
1;"  ";AT 14,1;"  "
2120 RETURN
2130 FOR A=USR "A" TO USR "U"+7
2140 READ US: POKE A,US
2150 NEXT A: RETURN
2160 DATA 32,168,251,32,63,17,15
,4
2170 DATA 0,0,0,14,14,254,204,40
2180 DATA 8,24,40,68,68,66,65,64
2190 DATA 32,24,20,34,34,66,130,
2
2200 DATA 64,64,65,198,198,68,68
,72
2210 DATA 2,2,130,67,67,34,34,18
2220 DATA 104,104,24,14,7,63,193
,193
2230 DATA 22,22,24,113,196,253,1
31,131
2240 DATA 7,7,14,20,20,28,42,42
2250 DATA 193,65,65,65,65,65,65,
65
2260 DATA 131,129,129,129,129,12
9,129,129
2270 DATA 224,240,40,40,40,52,84
,84
2280 DATA 0,0,0,1,1,3,15,62
2290 DATA 102,70,133,133,133,258
```

```
,255,3
2300 DATA 65,65,65,65,65,255,192
,128
2310 DATA 129,129,129,129,129,25
5,3,1
2320 DATA 102,99,163,161,161,255
,255,224
2330 DATA 0,0,0,128,128,192,240,
120
2340 DATA 24,24,24,24,24,24,255,
126
2350 DATA 63,127,255,1,1,1,15,7
2360 DATA 253,254,255,128,128,12
8,240,224
2370 DIM S$(7,9)
2380 LET S$(1)="   "+CHR$ 144+CHR
$ 140+CHR$ 140+CHR$ 145
2390 LET S$(2)="   "+CHR$ 146+"___
"+CHR$ 147
2400 LET S$(3)="   "+CHR$ 148+CHR
$ 143+CHR$ 143+CHR$ 149
2410 LET S$(4)="   "+CHR$ 150+CHR
$ 140+CHR$ 140+CHR$ 151
2420 LET S$(5)="  "+CHR$ 152+CHR$
 153+"   "+CHR$ 154+CHR$ 155
2430 LET S$(6)=CHR$ 156+CHR$ 157
+CHR$ 158+CHR$ 140+CHR$ 140+CHR$
 159+CHR$ 160+CHR$ 161
2440 LET S$(7)=CHR$ 162+"   "+CHR
$ 163+CHR$ 164+"   "+CHR$ 162
2450 RETURN
```



CHARACTER

In this program, written by Peter Shaw, you can redefine your graphics with ease. There are many 'character designers' on the market at the moment, but you now have your very own fast and easy-to-use definer.

The commands at your disposal are:

Cursor keys (5-8) move flashing cursor in direction of arrows.

S — Set block

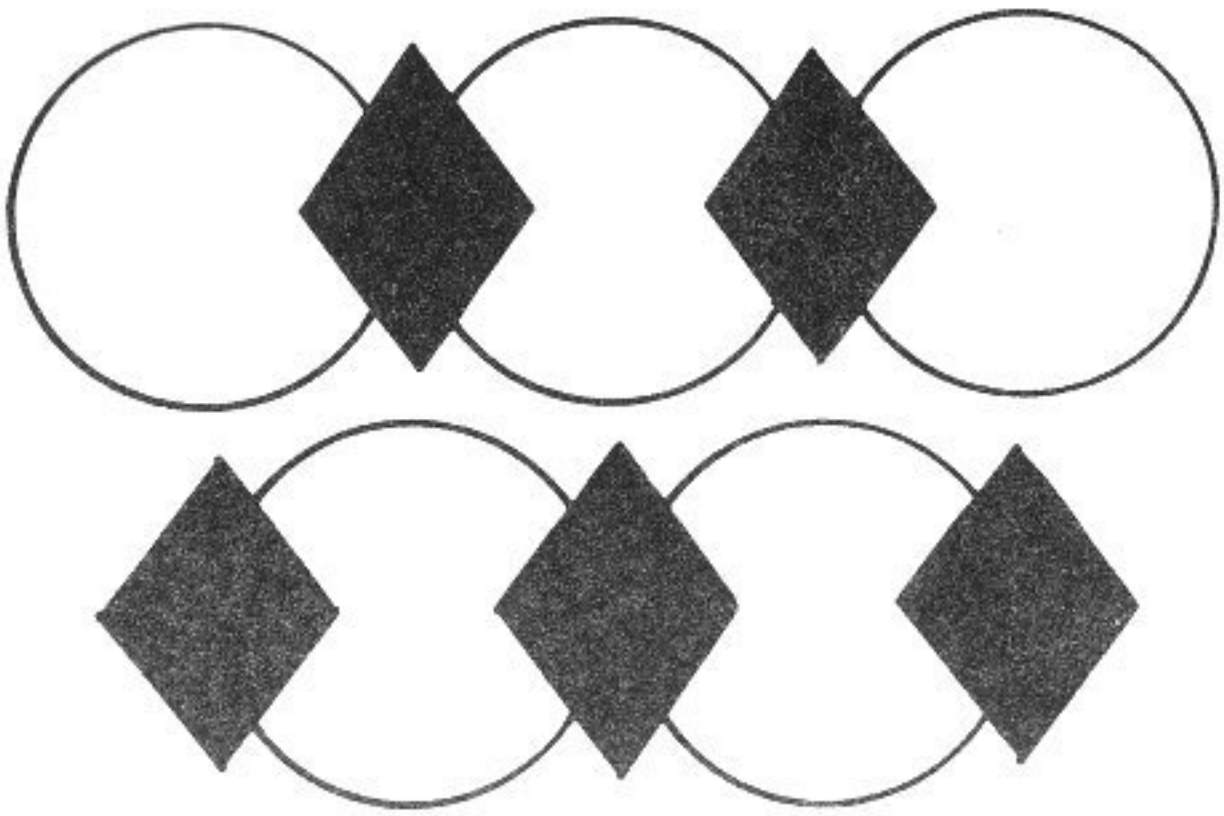
R — Reset (erase) block

P — Put (add to set) present character in 'display square'

G — Get character from set and display in 'display square'

M — Display Menu 2

Menu 2 includes all save and load commands necessary to get the most out of your user definable graphics.

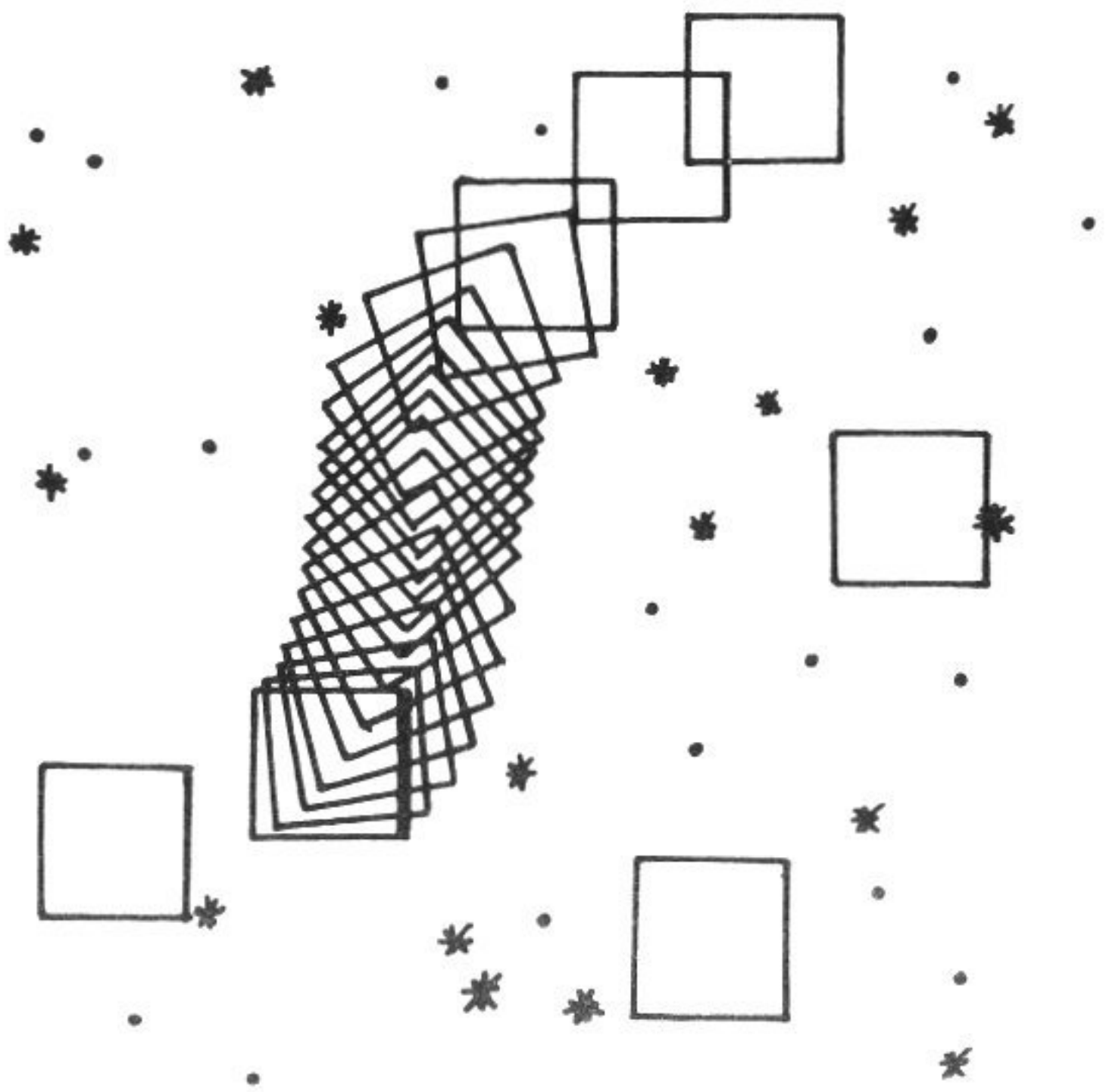


```
  10 REM CHARACTER DESIGNER
  20 REM   PETER SHAW   1983
  30 REM
  40 POKE 23658,8: GO SUB 990
  50 GO SUB 860: REM SQUARE
  60 GO SUB 730: REM MENU
  70 PRINT AT 2,13; PAPER 7;"PRE
SENT GRAPHICS:"
  80 PRINT AT 4,11;"A B C D E F
G H I J K"
  90 PRINT AT 5,11;P$( TO 21)
 100 PRINT AT 7,11;" L M N O P Q
 R S T U"
 110 PRINT AT 8,11;P$(22 TO )
 120 REM
 130 LET A=1: LET B=1
 140 PRINT AT P(A,B,1),P(A,B,2);
 OVER 1; BRIGHT 1; FLASH 1; INK
 (1 AND C(A,B)=0)+(0 AND C(A,B));
" "
 150 IF INKEY$="8" AND B<8 THEN
PRINT AT P(A,B,1),P(A,B,2); OVER
 1;" ": LET B=B+1
 160 IF INKEY$="5" AND B>1 THEN
PRINT AT P(A,B,1),P(A,B,2); OVER
 1;" ": LET B=B-1
 170 IF INKEY$="6" AND A<8 THEN
PRINT AT P(A,B,1),P(A,B,2); OVER
 1;" ": LET A=A+1
 180 IF INKEY$="7" AND A>1 THEN
PRINT AT P(A,B,1),P(A,B,2); OVER
 1;" ": LET A=A-1
 190 IF INKEY$="R" AND C(A,B)=0
THEN GO TO 210
 200 IF INKEY$="R" AND C(A,B)=1
THEN LET C(A,B)=0: PRINT AT P(A,
B,1),P(A,B,2); OVER 1; INVERSE 1
;" "
 210 IF INKEY$="S" AND C(A,B)=1
THEN GO TO 240
 220 IF INKEY$="S" AND C(A,B)=0
THEN LET C(A,B)=1: PRINT AT P(A,
B,1),P(A,B,2); OVER 1; INVERSE 1
;" "
 230 IF INKEY$="P" THEN GO TO 20
0
 240 IF INKEY$="G" THEN GO TO 35
0
 250 IF INKEY$="M" THEN GO SUB 5
00
 260 GO TO 140
 270 STOP
 280 PRINT AT P(A,B,1),P(A,B,2);
 OVER 1;" "
 290 INPUT "WHICH CHARACTER (A-U
)?";A$
```

```
  300 IF A$<"A" OR A$>"U" THEN GO
  TO 290
  310 FOR A=1 TO 8
  320 POKE USR A$+A-1,(128 AND C(
A,1))+(64 AND C(A,2))+(32 AND C(
A,3))+(16 AND C(A,4))+(8 AND C(A
,5))+(4 AND C(A,6))+(2 AND C(A,7
))+(C(A,8))
  330 NEXT A
  340 GO TO 70
  350 INPUT "WHICH CHARACTER (A-U
)?";A$
  360 IF A$<"A" OR A$>"U" THEN GO
  TO 350
  370 DIM C(8,8): FOR A=1 TO 8: L
ET U=PEEK ((USR A$)+A-1)
  380 IF U>=128 THEN LET C(A,1)=1
: LET U=U-128
  390 IF U>=64 THEN LET C(A,2)=1:
  LET U=U-64
  400 IF U>=32 THEN LET C(A,3)=1:
  LET U=U-32
  410 IF U>=16 THEN LET C(A,4)=1:
  LET U=U-16
  420 IF U>=8 THEN LET C(A,5)=1:
LET U=U-8
  430 IF U>=4 THEN LET C(A,6)=1:
LET U=U-4
  440 IF U>=2 THEN LET C(A,7)=1:
LET U=U-2
  450 IF U=1 THEN LET C(A,8)=1: L
ET U=U-1
  460 NEXT A
  470 GO SUB 860: FOR A=1 TO 8: F
OR B=1 TO 8
  480 IF C(A,B)=1 THEN PRINT AT P
(A,B,1),P(A,B,2); OVER 1; INVERS
E 1;" "
  490 NEXT B: NEXT A: GO TO 70
  500 CLS
  510 PRINT PAPER 1;; INK 9;"
            MENU 2";;;
  520 PRINT ''''TAB 3;"S - SAVE W
HOLE SET
  530 PRINT 'TAB 3;"D - SAVE BLOC
K"
  540 PRINT 'TAB 3;"F - SAVE SING
LE CHARACTER"
  550 PRINT 'TAB 3;"J - LOAD SET"
  560 PRINT 'TAB 3;"K - LOAD BLOC
K"
  570 PRINT 'TAB 3;"L - LOAD SING
LE CHARACTER"
  580 PRINT 'TAB 3;"R - RETURN TO
  MAIN PROGRAM"
  590 INPUT "YOUR CHOICE ? "; LIN
```

```
E A$
  600 LET S=0: LET F=0
  610 IF A$="R" THEN CLS : GO SUB
  730: GO TO 470
  620 IF A$="S" THEN LET S=USR "A
": LET F=USR "U"+8
  630 IF A$="D" THEN INPUT "START
  FROM (A-U) "; LINE S$: LET S=US
R S$: INPUT "FINISH ("; (S$); "-U)
"; LINE K$: LET F=USR K$+8
  640 IF A$="F" THEN INPUT "CHARA
CTER (A-U) "; LINE S$: LET S=USR
  S$: LET F=USR S$+8
  650 IF A$="J" OR A$="K" OR A$="
L" THEN GO TO 700
  660 IF S=0 AND F=0 THEN GO TO 6
00
  670 INPUT "NAME OF FILE ? ";N$:
  IF LEN N$<1 OR LEN N$>9 THEN GO
  TO 670
  680 SAVE N$CODE S,F-S
  690 LET A$="R": GO TO 610
  700 INPUT "FILE NAME ? (JUST PR
ESS ENTER TOLOAD THE FIRST FILE
ON TAPE) ";N$
  710 LOAD N$CODE
  720 GO TO 690
  730 PLOT 31,80: DRAW 193,0
  740 DRAW 0,-73: DRAW -193,0: DR
AW 0,73
  750 FOR A=15 TO 20
  760 PRINT AT A,4; PAPER 5;"
                            "
  770 NEXT A
  780 PRINT AT 12,4; BRIGHT 1; PA
PER 5;"                         "
  790 PRINT AT 13,4; BRIGHT 1; PA
PER 5;"          MENU           "
  800 PRINT AT 14,4; BRIGHT 1; PA
PER 5;"                         "
  810 PRINT AT 16,5; PAPER 8;"CUR
SOR (ARROW) KEYS:"
  820 PRINT AT 17,5; PAPER 8;"MOV
E FLASHING CURSOR"
  830 PRINT AT 18,5; PAPER 8;"S-S
ET:R-RESET:M-MENU 2"
  840 PRINT AT 19,5; PAPER 8;"G-G
ET CHAR:P-PUT CHAR"
  850 RETURN
  860 PRINT AT 1,2;"              "
  870 PRINT TAB 2;"              "
  880 PRINT TAB 2;"              "
  890 PRINT TAB 2;"              "
  900 PRINT TAB 2;"              "
  910 PRINT TAB 2;"              "
  920 PRINT TAB 2;"              "
```

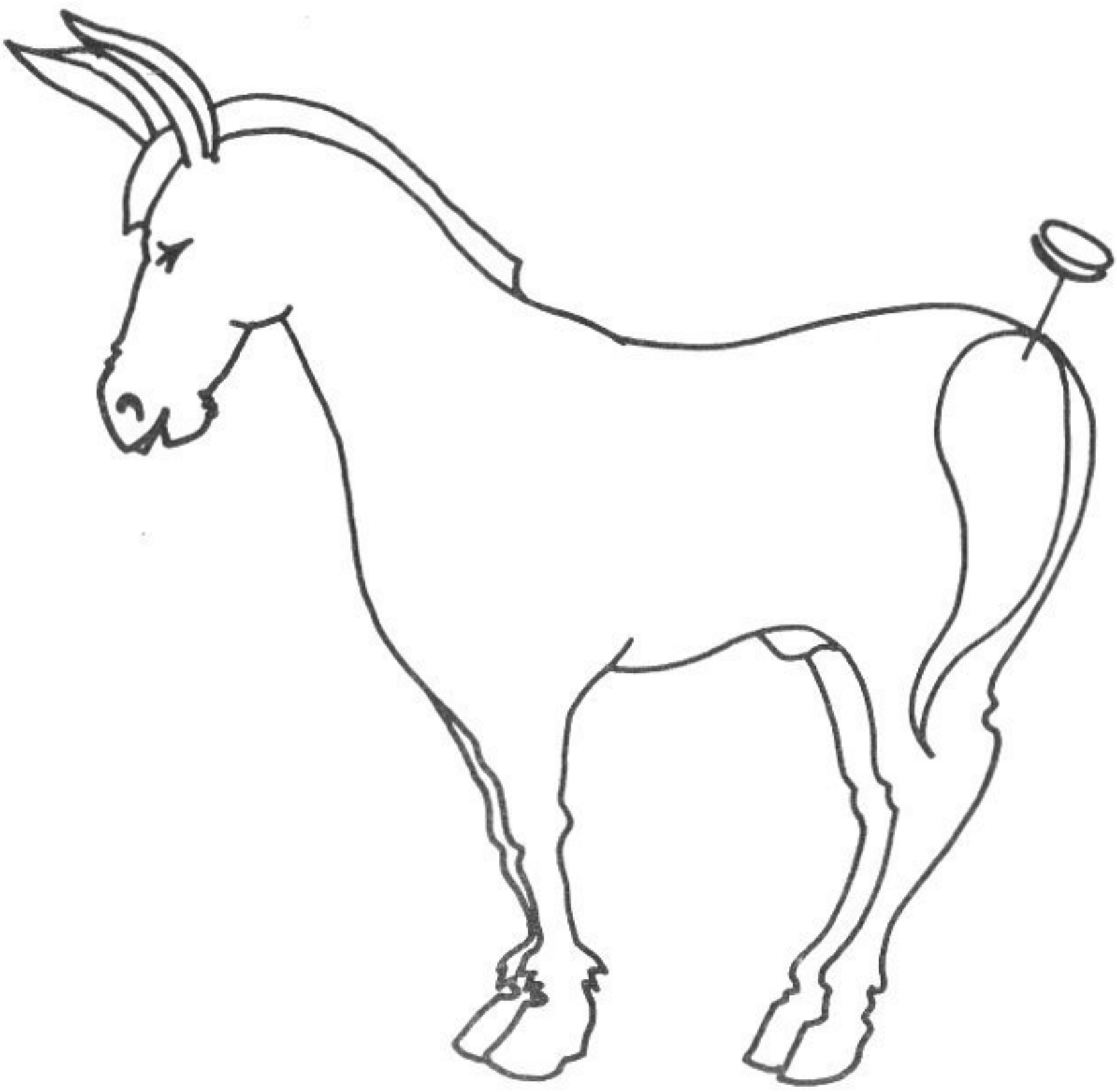
```
  930 PRINT TAB 2;"_____"
  940 PRINT TAB 2;"_____"
  950 FOR A=16 TO 80 STEP 8
  960 PLOT A,96: DRAW 0,64
  970 NEXT A
  980 RETURN
  990 BORDER 6: PAPER 6: INK 9: C
LS
1000 LET P$=""
1010 FOR A=144 TO 164: LET P$=P$
+CHR$ A+" ": NEXT A
1020 DIM C(8,8): DIM P(8,8,2)
1030 FOR A=1 TO 8: FOR B=1 TO 8
1040 LET C(A,B)=0
1050 LET P(A,B,1)=A+1: LET P(A,B
,2)=B+1
1060 NEXT B: NEXT A
1070 RETURN
```



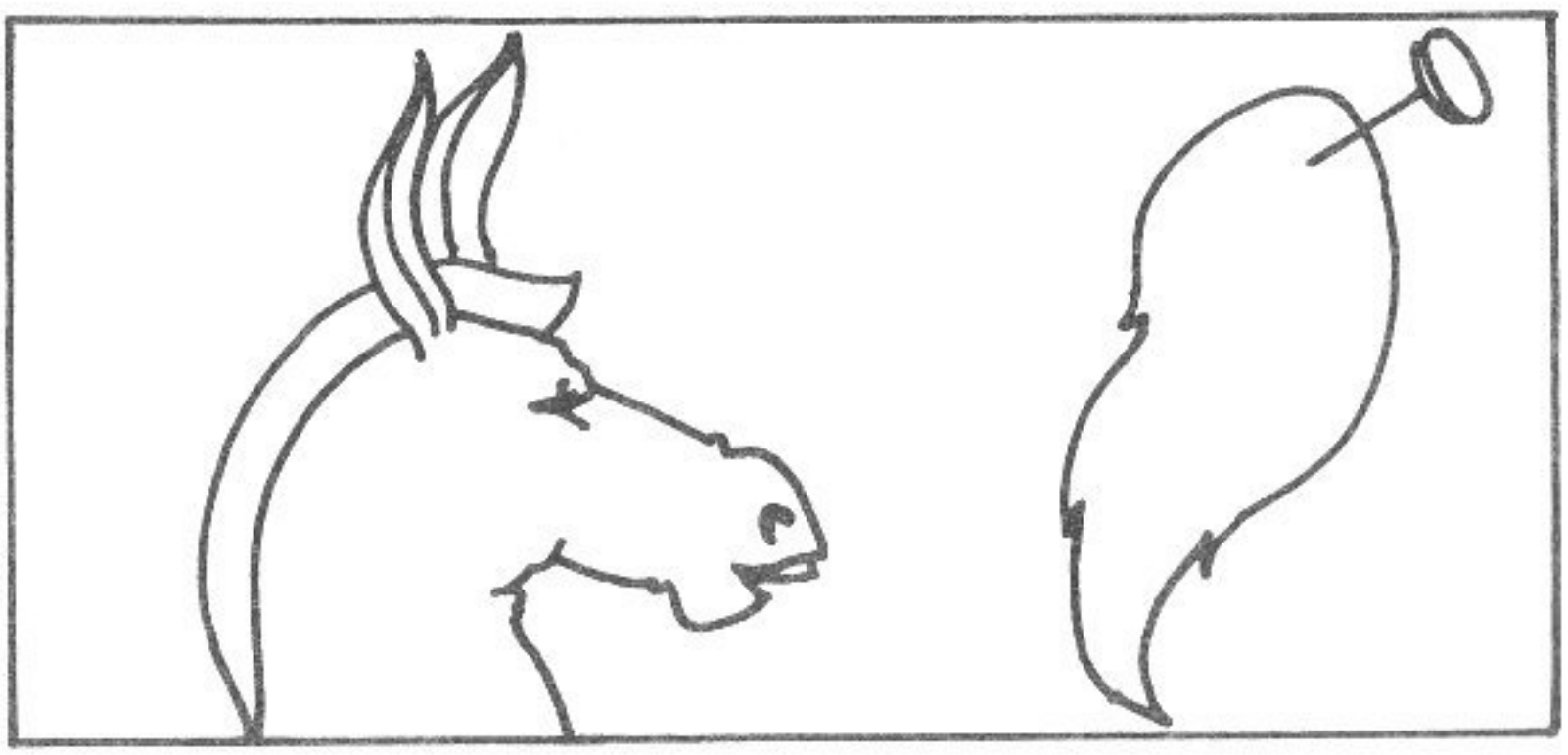
DONKEY

This is a simple, fun game. You must guess the position of the donkey's tail.

The donkey appears on the screen for a short length of time and then disappears. You will be asked to enter the horizontal and vertical co-ordinates of the point where you think the donkey's tail should be. You can have up to four guesses.



```
    5 INVERSE 0: BORDER 0: PAPER
0: INK 7: CLS
   10 RANDOMIZE
   20 LET a=0: LET b=a: LET n=a
   30 LET y=INT (RND*25)+1
   40 LET x=INT (RND*18)+2
   50 CLS
   60 LET a$=CHR$ 32+CHR$ 141
   70 LET b$=CHR$ 32+CHR$ 133+CHR
$ 131+CHR$ 138
   80 PRINT AT x,y; INK 3;a$;AT x
+1,y; INK 3;b$;AT a,b; INK 5;CHR
$ 47
   90 IF a=x AND b=y+4 THEN GO TO
 230
  100 IF n=4 THEN GO TO 200
  110 PAUSE 20
  120 CLS
  130 PRINT "Tail position"
  140 PRINT '"Vertical position 2
-19"
  150 INPUT a
  160 PRINT "Horizontal position
5-30"
  170 INPUT b
  180 LET n=n+1
  190 GO TO 50
  200 PRINT AT x-2,2;"You lose"
  210 PRINT 'x;"   ";y+4
  220 STOP
  230 INVERSE 1: PRINT AT x-2,2;"
Correct, you took ";n;" goes"
  235 INVERSE 0: PRINT AT x-2,2;"
Correct";
  240 IF INKEY$="" THEN GO TO 230
  250 RUN
```



BUTTERFLY

This is another high-resolution plotting program, showing the advanced graphic capabilities of the computer.

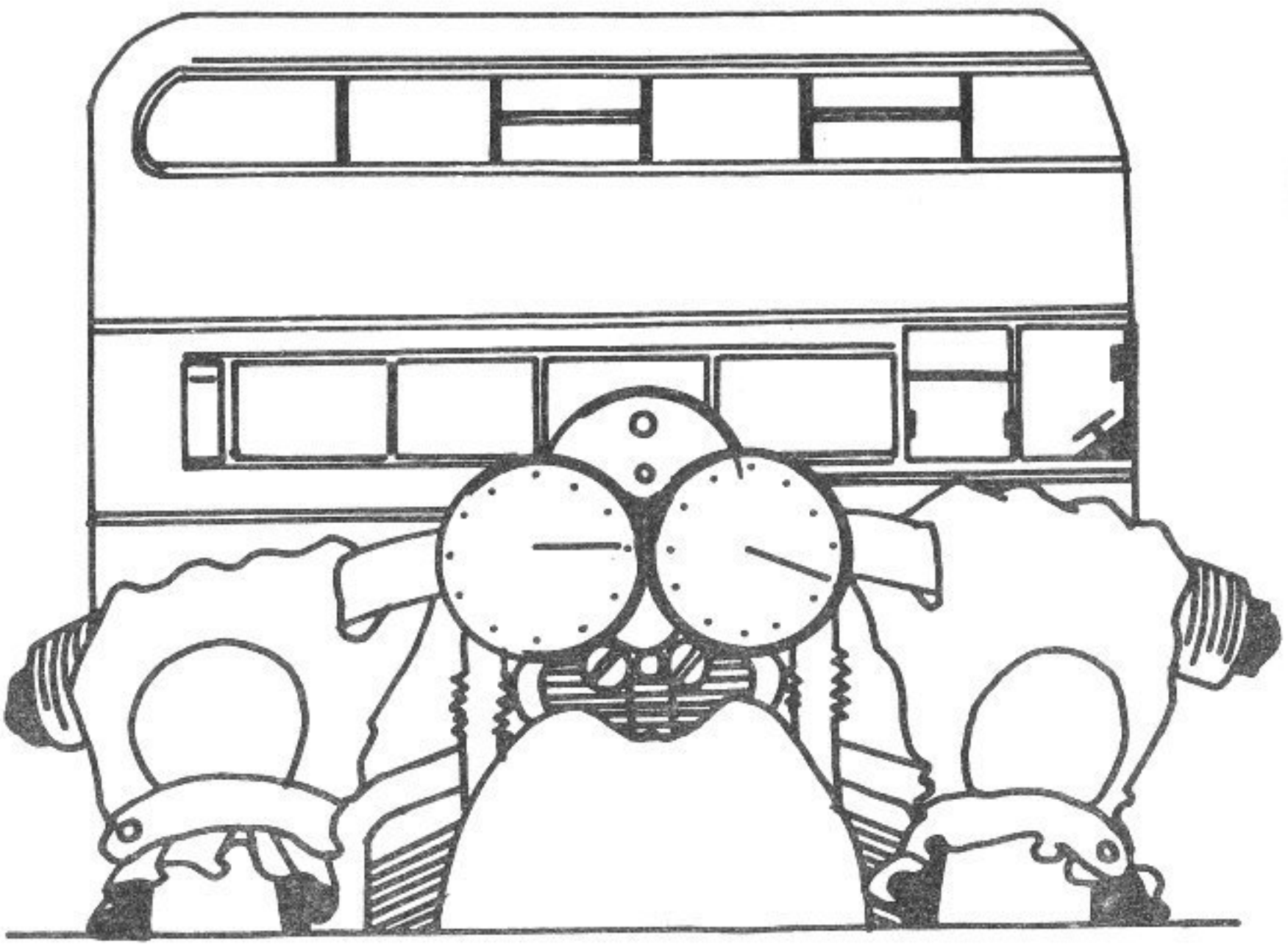
```
  10 REM BUTTERFLY
  20 REM DANIEL MILLS
  30 REM MARTYN LATIMER
  40 BORDER 0: PAPER 0: INK 9: B
RIGHT 1
  50 CLS
  60 LET A=INT (RND*120)
  70 LET B=INT (RND*80)
  80 LET C=INT (RND*7)+1
  90 INK C
 100 PLOT 128-A,88-B
 110 PLOT 128+A,88-B
 120 PLOT 128-A,88+B
 130 PLOT 128+A,88+B
 140 IF RND>.99 THEN GO TO 40
 150 GO TO 60
```

MOTORCYCLE STUNTMAN

In this game, it's on with your crash helmet and away you go. Your object is to jump over a line of buses, which change in number randomly every game.

Hold down any key and watch your speed increase. Release the brakes (by taking your finger off the key) and watch your motorcycle fly over the ramps.

This game uses accurate formulae for thrust and velocity, and is excellent fun.



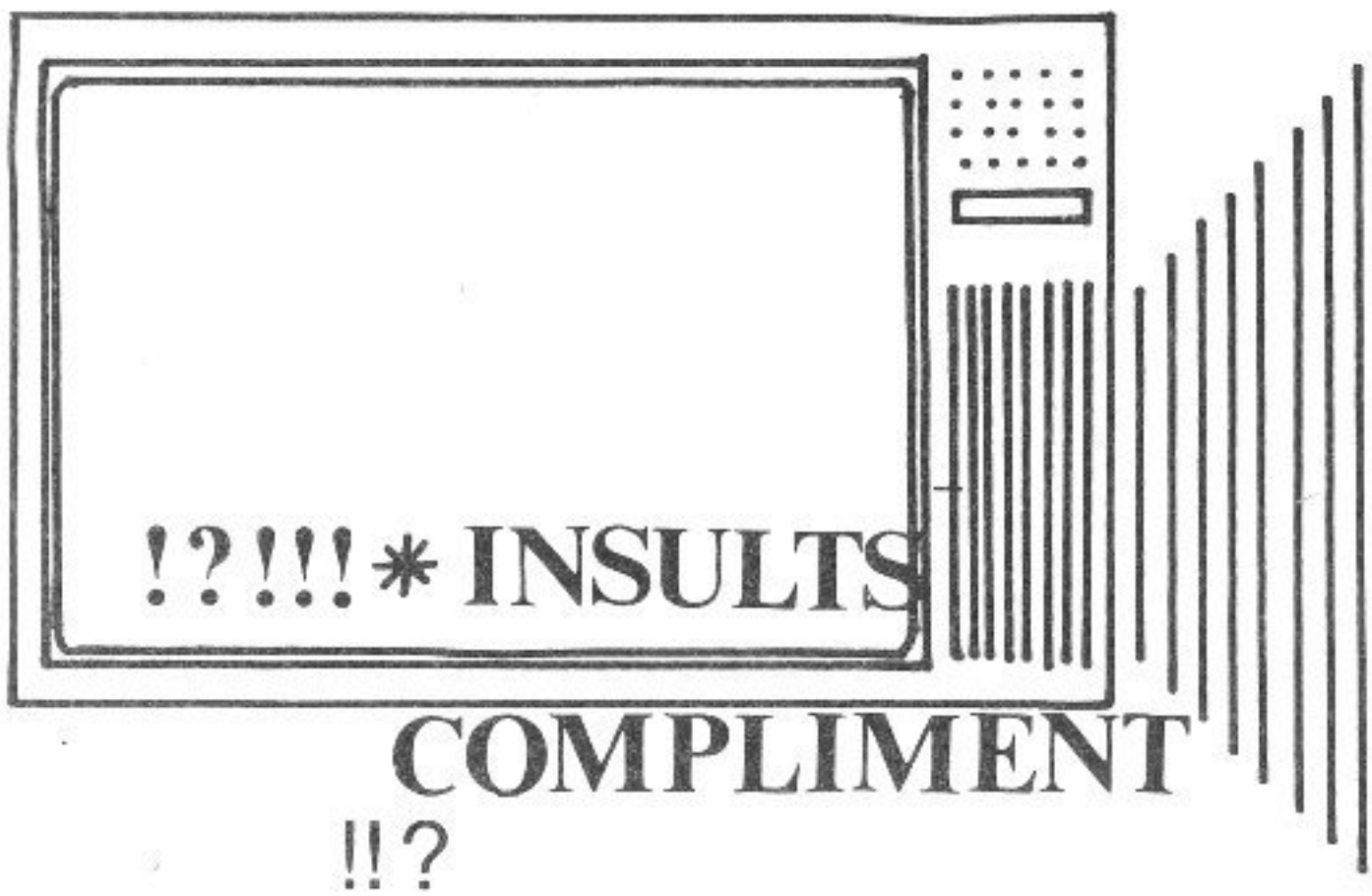
```
  5 INVERSE 0: GO SUB 3000
 10 RANDOMIZE
 15 LET bus=INT (RND*17)+2
 20 GO SUB 1000
 30 LET m=0
 32 PRINT AT 1,11;"0 M.P.H."
 35 IF INKEY$="" THEN GO TO 35
 40 LET m=m+5+INT (RND*6)
 42 PAUSE 3
 45 IF m>170 THEN LET m=170
 50 PRINT AT 1,11;m;" M.P.H."
 60 IF INKEY$<>"" THEN GO TO 40
 65 LET mp=m
 70 LET z=INT (RND*11)
 80 LET m=m+(z-5)-40
 90 LET j=9+INT (m/7.5+.5)
100 IF m<-5 THEN LET j=6
110 FOR n=1 TO 4
120 PRINT AT 10,n-1;" ";AT 10,n
;"E"
130 NEXT n
140 PRINT AT 10,4;" ";AT 9,5;"E
"
150 PRINT AT 9,5;" ";AT 8,6;"E"
160 IF j=6 THEN GO TO 200
170 FOR n=7 TO j
180 PRINT AT 8,n-1;" ";AT 8,n;"
E"
185 PAUSE 2
190 NEXT n
200 PRINT AT 8,j;" ";AT 9,j+1;"
E"
210 PRINT AT 9,j+1;" ";AT 10,j+
2;"E"
220 IF j<>bus+7 THEN GO TO 2000
230 FOR n=j+3 TO 31
240 PRINT AT 10,n-1;" ";AT 10,n
;"E"
250 NEXT n
255 PAUSE 25
260 PRINT AT 10,31;" "
270 CLS
280 PRINT AT 4,11;"Great jump!"
290 PRINT AT 5,2;"You cleared a
l ";bus;" buses at"
300 PRINT AT 8,6;"a speed of ";
mp;" M.P.H."
310 PAUSE 120
320 NEXT n
330 RUN
1000 CLS
1010 PRINT AT 11,0;"DDDDDDDDDDDD
DDDDDDDDDDDDDDDDDDDDD"
1020 PRINT AT 10,5;"A ";AT 9,6;"
"
1030 FOR n=7 TO bus+6
```

```
1040 PRINT AT 10,n; INK 2;"B"
1050 NEXT n
1060 PRINT AT 10,bus+7;" C";AT 9
,bus+7;" "
1070 PRINT AT 10,0;"E"
1080 RETURN
2000 FOR f=1 TO 10
2010 PRINT AT 10,j+2;" ";AT 10,j
+2,;" "
2020 NEXT f
2030 PAUSE 40
2040 CLS
2045 IF j<bus+7 THEN GO TO 2100
2050 PRINT AT 4,3;"You went too
fast and you"
2060 PRINT AT 6,4;"missed the la
nding ramp."
2070 PRINT AT 8,7;"Try a slower
speed."
2080 PAUSE 120
2090 GO TO 20
2100 PRINT AT 4,1;"You went too
slow and your bike"
2110 PRINT AT 6,9;"is a write-of
f."
2120 PRINT AT 8,3;"Try again on
a new bike"
2130 PAUSE 120
2150 GO TO 20
3000 FOR n=USR "a" TO USR "e"+7:
 READ r
3010 POKE n,r: NEXT n
5000 DATA 0,1,3,7,15,31,63,255
5010 DATA 0,60,66,124,66,60,90,6
5
5020 DATA 0,128,192,224,240,248,
252,255
5030 DATA 0,85,219,85,219,85,219
,85
5040 DATA 32,96,80,110,50,82,181
,66
5050 GO TO 10
```

INSULTS/ COMPLIMENTS

These two programs use the useful routine of extracting information stored in strings to generate compliments or insults.

This technique has many uses in games programs.

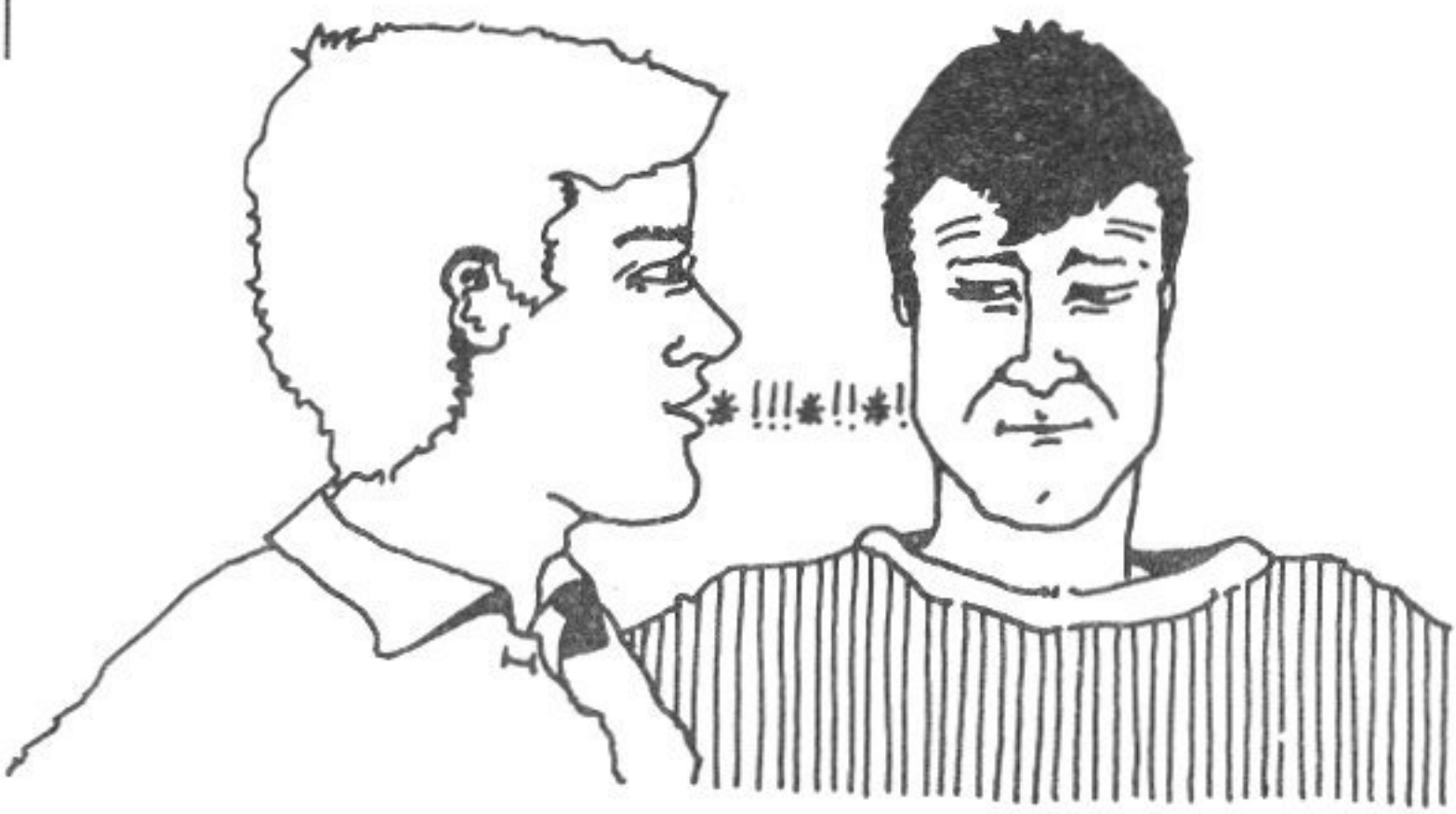



```
  10 RANDOMIZE
  20 LET A$="SILLYUGLYFOOLISHSTU
PIDTIRESOMEBARMYTHICKGORMLESSDOZ
EYSENSELESS"
  30 DIM B(10): DIM D(10): RESTO
RE 30: FOR A=1 TO 10: READ B(A),
D(A): NEXT A
  40 LET C$="FOOLIDIOTMANIACTWIT
IMBECILEBLOCKHEADMORONDOLTCOWARD
LUNATIC"
  50 DATA 1,1,6,5,10,10,17,16,23
,20,31,28,36,37,41,42,49,46,54,5
2,63,59
  60 LET A=INT (RND*9)+1
  70 BORDER (A/2)+2: LET B=INT (
RND*9)+1
```

```
   80 BEEP .1,INT (RND*20)+10: PO
KE 23692,-1: PRINT AT 21,0;''
   90 PRINT PAPER INT (RND*7)+1;
INK 9;A$(B(A) TO B(A+1)-1);" ";C
$(D(B) TO D(B+1)-1)
  100 GO TO 60
```

COMPLIMENTS

```
   10 RANDOMIZE
   20 LET a$="NICEWISEBEAUTIFULCL
EVERCHARMINGWITTYFRIENDLYBRILLIA
NTSMARTLOVELY"
   30 DIM B(10): DIM D(10): RESTO
RE 30: FOR A=1 TO 10: READ B(A),
D(A): NEXT A
   40 LET C$="PERSONPROGRAMMERCHA
RMERGENIUSWITOPERATORSAGEINDIVID
UALHEROACE"
   50 DATA 1,1,5,7,9,17,18,24,24,
30,32,33,37,41,45,45,54,55,59,59
,65,62
   60 LET A=INT (RND*9)+1
   70 BORDER (A/2)+2: LET B=INT (
RND*9)+1
   80 BEEP .1,INT (RND*20)+10: PO
KE 23692,-1: PRINT AT 21,0;''
   90 PRINT PAPER INT (RND*7)+1;
INK 9;A$(B(A) TO B(A+1)-1);" ";C
$(D(B) TO D(B+1)-1)
  100 GO TO 60
```



How To Write Better Programs

By Tim Hartnell, series editor

There are a number of fine programs in this book, and many of the regular computer magazines contain other such ones. But no matter how good the programs from published sources are, you are certain to get more pleasure from running them if they have been partially or completely written by you. Putting your personal stamp on programs, altering them to reflect your wishes and creativity, is an excellent way to improve the programs, and eventually, of course, you'll become a better and more imaginative programmer.

Programs in magazines, and in books like this one, are ideal as starting points for your own developments. You may also find that advertisements for software packages can be fruitful 'idea-starters'. You only need to read the description of what the commercially available program does, and you will have the first step towards creating your own program. You have to be careful, of course, not to infringe copyright either in the screen displays, in the name of the program, or the names of the 'characters' within the program. However, you will probably find that at a certain point in its development the program will take on a life of its own, growing and evolving away from the original scenario, until you eventually have a completely new game concept and implementation.

Whatever you do, be careful not to pass off other people's work as your own. By all means adapt and im-

prove published programs, but do not then present them to magazines as if they were originals. I have lost count of the number of times one of my own programs, from one of my books, has been submitted to me for publication.

Always watch out for new ideas as you look through books, game and computer magazines, or wander through video game arcades. It may be worth keeping notes of ideas you come across for games, for character shapes, for sounds, for dramatic endings and so on. Thus you will never be short of ideas, and you will also be able to merge the material together to produce better games which hold the player's attention for longer.

Games tend to fall into one of three categories, and it is worth making sure of the category into which your proposed program will fall *before* you start to program, since the category of game materially alters the programming approach. This is not to say that, as you develop a program, it will not move from one category into another, nor that a particular game might not extend across two categories, but it is nevertheless useful to keep the various groups separate in your mind, just to clarify your thoughts. The three categories are:

1. Board games
2. 'Arcade' (that is, highly visual, fast moving, noisy, real time) games
3. Games of chance (such as Roulette and Snap).

In board games, the quality of play is more important than lightning-fast response, while the arcade-type programs must be kept moving at all costs, even if some 'intelligence' from your Martian intruders must be sacrificed to achieve this. Games of chance depend more on their ease of play ('user-friendly' inputs), and an approach to true randomness, than do either of the other categories.

You will find that games programs tend to fall into types, which are subdivisions of the three above mentioned categories. Many board games are variants of chess or checkers; many arcade games started off life as Space Invader-type games; and games of chance

started off in the 'real world' of dice and cards. Looking at a program description, or a games machine, and trying to categorise the game you see can help trigger new ideas which fit within that particular game's genre.

There is a school of thought within programming — generally called 'structured programming' — which believes that discipline at the beginning of the games-writing process is essential. While less interesting than sitting down at the computer right away, a much better program is produced in the end. I once wrote a program called Dome Dweller, a simulation program in which the player is in charge of a 'lunar dome' and must decide which products to manufacture and sell in order to buy oxygen and food for the station's inhabitants. (This program was used in my book *The Book of Listings*, written with Jeremy Ruston, and published by the BBC.)Once I had decided the overall scenario, I worked out the screen display, and came up with an idea as follows:

Oxygen supplies are low
There are 96 people living within your dome in year 3
Money credit is $5,693
Annual maintenance charge is $226
Oxygen tanks hold 811 units
Oxygen costs $8 per unit
Each dome dweller needs 5 units a year
Food stocks stand at 2122
Each dweller needs 3 units a year ($6 each, $576 for dome. This will last 7 years at present population.)
You can trade your unique lunar sculptures with the people who live in other domes. You use up 2 units of oxygen making each one, and sell them for $30.

As you can probably guess from this 'sample printout', the idea of the program is to decide how many 'unique lunar sculptures' you must make and sell in order to buy oxygen and food, and to pay the 'annual maintenance' charge. The problem with this particular program is that

making each sculpture uses up oxygen, so you must balance your wish to make money against the need to use the oxygen intelligently.

You may well wish to try writing such a program yourself. You should end up with an enjoyable program, and writing it will do much to help you develop your programming skills. The first thing to do is to make a list of what the program has to do:

Set up the needed variables
Tell the player the 'state of the dome'
Ask how much oxygen to be bought
Check if can afford this, if so buy it, if not go back and ask again
Ask how much food to be bought
Check if can afford this, if so buy it, if not go back and ask again
Update oxygen quantity
Update food quantity
Reduce money left total
Ask how many items of sculpture to be made
Check if there is enough oxygen to make this many, if not go back and ask again
Reduce oxygen quantity by amount needed to make the number of sculptures specified, increase money total to reflect value of sculptures made
Increase the population total slightly, add one to the 'current year'
Check if there is enough food in stocks to feed whole population
Check if there is enough oxygen for whole population
Check if there is any money
If any of these conditions are negative (eg not enough food) send action to an 'end of game' routine
If all are positive, loop back to tell the player the state of the dome, and continue to circle

You could probably write a Dome Dweller program

using the list above, together with the 'sample printout' information. There is, however, a secret I should like to share with you which unlocks programming problems almost instantly. You can actually write all the vital parts of a program in minutes, so you can see the raw framework of a program like this running long before you fill in the details. And once you have a framework you can work on it for as long as you like, knowing as you do so that — at every moment in program development — you have a working program. You do not have to wait until the end until you can run it to see how you are going. The 'secret' is to hold the entire program within a series of subroutine calls, all held within a perpetual loop. Here's how it could work with this program. The very first lines you enter in your computer are as follows:

```
10 REM DOME DWELLER
20 GOSUB 1000: REM ASSIGN VARIABLES
30 GOSUB 2000: REM PRINT OUT STATE OF
DOME
40 GOSUB 3000: REM OXYGEN
50 GOSUB 4000: REM FOOD
60 GOSUB 5000: REM SCULPTURE
70 GOSUB 6000: REM UPDATE POPULATION
80 GOSUB 7000: REM CHECK ON STATE OF
 DOME
90 IF (all conditions positive, from GOSUB 7000)
THEN GOTO 30
100 REM End of game ...
```

As you can see once you have the 'master loop' set up in this way, it is relatively simple to fill in each of the subroutines one by one, testing each as you do so, and elaborating each one so that you end up eventually with a very good program. The only thing you need now is a list of the variables which you will use with the program.

I find the best way to do this is to use explicit names for variables so that when you are programming you do not have to spend time checking, for example, whether AA stands for the population, or the number of units of oxygen used up in making each item of sculpture. To make

programs as easy as possible to transfer between different computers you can stick to two letter variable names, or you can take advantage (if your computer allows it) of long names (such as OXYUSE for the amount of oxygen used) for variables. Then you have no doubts whatsoever as to the meaning of each variable name. To show how this can work, and to illustrate a further advantage of explicit variable names, here are the variables used in Dome Dweller:

FOLK — population of dome
CASH — money in treasury
FOOD — food stocks on hand
FOODCOST — how much each unit of food costs
FOODNEED — how many units of food were consumed per person per year
ARTCOST — how much oxygen was used up making each piece of sculpture
ARTPAY — how many dollars each piece of sculpture was sold for
OXY — oxygen stocks on hand
OXYNEED — how many units of oxygen were consumed per person per year
OXYCOST — how much each unit of oxygen cost to buy
REPAIR — the cost of annual repairs to the dome
YEAR — the year of the dome's life

Using explicit variable names in this way — although they use up more memory than do single or double-letter variable names — makes it very simple to follow through a program, working out what each section of the program actually does. Moreover, and this is the further advantage mentioned, it is very easy when writing the program to insert the formulae required for calculations. By this I mean that if, for example, you wished to include (as I do in this program) an indication of how much oxygen is needed for each year, you simply multiply the number of people in the dome (FOLK) by the number of oxygen units each person needs each year (OXYNEED). You can then

include within the printouts for the state of the dome a line like:

```
PRINT "THERE ARE ";FOLK;" IN THE DOME"
PRINT "IN YEAR ";YEAR
PRINT "EACH PERSON NEEDS ";OXYNEED;"
      UNITS OF"
PRINT "OXYGEN EACH YEAR,";
      OXYNEED*FOLK;" NEEDED"
PRINT "FOR THE WHOLE DOME"
```

It also makes it very easy to check on whether purchases are possible. For example, to buy food, you could say:

```
PRINT "HOW MUCH FOOD WILL YOU BUY?"
INPUT A
IF A*FOODCOST > CASH THEN GOTO (get
another A)
```

So the suggestions given here for improving your programs by the use of 'structured programming' include the following:

- draw up a sample printout, or mock-up of the final screen display
- draw up a list of what the program has to do each time through a 'master control loop'
- change this list to a series of subroutine calls
- use explicit variable names if possible

It is useful if you are designing programs for others to use to ensure that it is quite clear what the player should do when running the program. There is little point, especially when memory is limited, in including a long set of instructions within the program, but you should certainly write such instructions down. In addition, user prompts should be explicit (such as ENTER THE NUMBER OF GOES YOU WANT) and should include warnings of the limits which will be placed on the input (HOW MANY CARDS WILL YOU START WITH: 1, 2 OR 3 ?, for instance).

You cannot assume that you will be present every time a program is run, so you should do your best to make it as foolproof as possible. If you can, add error-trapping routines to the program to ensure that a mistake in enter-

ing a choice earlier on in the program will not cause it to crash or come up with stupid results later on.

If you read through this section of the book several times and try to apply the ideas to your own programming work, you should find your work quality improves significantly, and also that you can spend more time improving and embellishing a program and less in the raw mechanical task of getting the thing running.

GLOSSARY

A

Accumulator — the place within the computer in which arithmetic computations are performed and where the results of these computations are stored.

Algorithm — the series of steps the computer follows to solve a particular problem.

Alphanumeric — this term is usually used in relation to a keyboard, as in 'it is an alphanumeric keyboard', which means that the keyboard has letters as well as numbers. It is also used to refer to the 'character set' of the computer. The character set comprises the numbers and letters the computer can print on the screen.

ALU (Arithmetic/Logic Unit) — the part of the computer which does arithmetic (such as addition, subtraction) and where decisions are made.

AND — a Boolean logic operation that the computer uses in its decision-making process. It is based on Boolean algebra, a system developed by mathematician George Boole (1815-64). In Boolean algebra the variables of an expression represent a logical operation such as OR and NOR.

ASCII — stands for American Standard Code for Information Exchange, the most widely used encoding system for English language alphanumerics. There are 128 upper and lower case letters, digits and some special characters. ASCII converts the symbols and control instructions into seven-bit binary combinations.

Assembler — a program which converts other programs written in assembly language into machine code

(which the computer can understand directly). Assembly language is a low level programming language which uses easily memorised combinations of two or three letters to represent a particular instruction which the assembler then converts so the machine can understand it. Examples of these are ADD (add), and SUB (subtract). A computer programmed in assembly language tends to work more quickly than one programmed in a higher level language such as BASIC.

B

BASIC — an acronym for Beginners All-Purpose Symbolic Instruction Code. It is the most widely used computer language in the microcomputer field. Although it has been criticised by many people, it has the virtue of being very easy to learn. A great number of BASIC statements resemble ordinary English.

Baud — named after Baudot, a pioneer of telegraphic communications. Baud measures the rate of transfer of information and is approximately equal to one bit per second.

BCD — an abbreviation for Binary Coded Decimal.

Benchmark — a test against which certain functions of the computer can be measured. There are a number of so-called 'standard Benchmark tests', but generally these only test speed. This is rarely the aspect of a microcomputer that is most of interest to the potential buyer.

Binary — a numbering system that uses only zeros and ones.

Bit — an abbreviation for Binary Digit. This is the smallest unit of information a computer circuit can recognise.

Boolean Algebra — the system of algebra developed by mathematician George Boole which uses algebraic notation to express logical relationships (see AND).

Bootstrap — a short program or routine which is read into the computer when it is first turned on. It orients the computer to accept the longer, following program.

Bug — an error in a computer program which stops the program from running properly. Although it is generally used to mean only a fault or an error in a program, the term bug can also be used for a fault in the computer hardware.

Bus — a number of conductors used for transmitting signals such as data instructions, or power in and out of a computer.

Byte — a group of binary digits which make up a computer word. Eight is the most usual number of bits in a byte.

C

CAI — Computer Assisted Instruction.

CAL — Computer Assisted Learning. The term is generally used to describe programs which involve the learner with the learning process.

Chip — the general term for the entire circuit which is etched onto a small piece of silicon. The chip is, of course, at the heart of the microcomputer.

Clock — the timing device within the computer that synchronises its operations.

COBOL — a high level language derived from the words Common Business Orientated Language. COBOL is designed primarily for filing and record-keeping.

Comparator — a device which compares two things and produces a signal related to the difference between the two.

Compiler — a computer program that converts high level programming language into binary machine code so the computer can handle it.

Complement — a number which is derived from another according to specified rules.

Computer — a device with three main abilities or functions:
1) to accept data
2) to solve problems
3) to supply results

CPU — stands for Central Processing Unit. This is the heart of the computer's intelligence, where data is handled and instructions are carried out.

Cursor — a character which appears on the TV screen when the computer is operating. It shows where the next character will be printed. On a computer there are usually 'cursor control keys' to allow the user to move the cursor around the screen.

D

Data — information in a form which the computer can process.

Debug — the general term for going through a program and correcting any errors in it, that is, chasing down and removing bugs (see Bug).

Digital Computer —a computer which operates on information which is in a discrete form.

Disk/Disc — this is a magnetically sensitised plastic disk, a little smaller than a single play record. This is used for storing programs and for obtaining data. Disks are considerably faster to load than a cassette of the same length program. The disk can be searched very quickly while a program is running for additional data.

Display — the visual output of the computer, generally on a TV or monitor screen.

Dot Matrix Printer — a printer which prints either the listing of a program or that which is displayed on the TV screen. Each letter and character is made up of a number of dots. The higher the number of dots per character the finer the resolution of the printer.

Dynamic Memory — a memory unit within the computer which 'forgets' its contents when the power is turned off.

E

Editor — this term is generally used for the routine within the computer which allows you to change lines of a program while you are writing it.

EPROM — stands for Erasable Programmable Read-Only Memory. This is like the ROM in the computer, except that it is fairly easy to load material into an EPROM and it doesn't disappear when you turn the power off. EPROMs must be placed in a strong ultra violet light to erase them.

Error Messages — the information given by a computer where there is a fault in the coding during a part of a program, usually shown by the computer stopping, and printing a word, or a word and numbers, or a combination of numbers only, at the bottom of the screen. This tells you what mistake has been made. Common mistakes include using the letter O instead of zero in a line, or leaving out a pair of brackets, or one of the brackets, in an expression, or failing to define a variable.

F

File — a collection of related items of information organised in a systematic way.

Floppy Disk — a relatively cheap form of magnetic disk used for storing computer information, and so named because it is quite flexible (see Disk/Disc).

Flow Chart — a diagram drawn up before writing a program, in which the main operations are enclosed within rectangles or other shapes and connected by

lines, with arrows to represent loops, and decisions written at the branches. It makes writing a program much easier because traps such as infinite loops, or non-defined variables can be caught at an early stage. It may not be worth writing a flow chart for very short programs, but generally a flow chart aids in creating programs.

Firmware — there are three kinds of 'ware' in computers: software 'temporary' programs; hardware like the ROM which contains permanent information; and firmware in which the information is relatively permanent, as in an EPROM (see EPROM).

Flip-Flop — a circuit which maintains one electrical condition until changed to the opposite condition by an input signal.

FORTRAN — an acronym for FORmula TRANslation, this is a high level, problem orientated computer language for scientific and mathematical use.

G

Gate — an electrical circuit which, although it may accept one or more incoming signals, only sends out a single signal.

Graphics — pictorial information as opposed to letters and numbers.

H

Hard Copy — computer output which is in permanent form.

Hardware — the physical parts of the computer (also see software and firmware).

Hexadecimal (Hex) — a numbering system to the base sixteen. The digits zero to nine are used, as well as the letters A, B, C, D, E and F to represent numbers. A

equals 10, B equals 11, C equals 12, and so on. Hex is often used by microprocessor users.

Hex Pad — a keyboard designed specifically for entering hexadecimal notation.

High Level Language — a programming language which allows the user to talk to the computer more or less in English. In general, the higher the level of the language (that is, the closer it is to English), the longer it takes for the computer to translate it into a language it can use. Lower level languages are far more difficult for human operators but are generally executed far more quickly.

I

Input — the information fed into the computer via a keyboard, a microphone, a cassette or a disk.

Input/Output (I/O Device) — a device which accepts information or instructions from the outside world, relays it to the computer, and then, after processing, sends the information out in a form suitable for storing, or in a form which could be understood by a human being.

Instruction — data which directs a single step in the processing of information by the computer (also known as a command).

Integrated Circuit — a complete electronic circuit imprinted on a semiconductor surface.

Interface — the boundary between the computer and a peripheral such as a printer.

Interpreter — a program which translates the high level language fed in by the human operator, into a language which the machine can understand.

Inverter — a logic gate that changes the signal being fed in, to the opposite one.

Interactive Routine — part of a program which is repeated over and over again until a specified condition is reached.

J

Jump Instruction — an instruction which tells the computer to go to another part of the program, when the destination of this move depends on the result of a calculation just performed.

K

K — this relates to the size of the memory. Memory is usually measured in 4K blocks. 1K contains 1,024 bytes.

Keyword — the trigger word in a line of programming, usually the first word after the line number. Keywords include STOP, PRINT and GOTO.

L

Language — computer languages are divided into three sections: high level languages, such as BASIC, which are reasonably close to English and fairly easy for humans to use; low level languages, such as Assembler, that use short phrases which have some connection with English (ADD for add and RET for return, for instance); and machine code which communicates more or less directly with the machine.

LCD — this stands for Liquid Crystal Diode. Some computers such as the TRS-80 Pocket Computer use an LCD display.

LED — this stands for Light Emitting Diode. The bright

red numbers which are often used on watch or clock displays are made up of LEDs.

Logic — the mathematical form of a study of relationships between events.

Loop — a sequence of instructions within a program which is performed over and over again until a particular condition is satisfied.

M

Machine Language or Machine Code — an operation code which can be understood and acted upon directly by the computer.

Magnetic Disk — see Disk and Floppy Disk.

Mainframe — computers are generally divided into three groups, and the group a computer falls into depends more or less on its size. The computer you are thinking of buying is a microcomputer; medium sized computers are known as minicomputers; and the giant computers that you sometimes see in science fiction movies are mainframe computers. Until 15 years ago mainframe computers were, in practical terms, the only ones available.

Memory — there are two types of memory within a computer. The first is called ROM (read-only memory); this is the memory that comes already programmed on the computer, which tells the computer how to make decisions and how to carry out arithmetic operations. This memory is unaffected when you turn the computer off. The second type is RAM (random access memory). This memory holds the program you type in at the keyboard or send in via a cassette or disk. In most computers the computer 'forgets' what is in RAM when you turn the power off.

Microprocessor — the heart of any computer. It requires peripheral unit interfaces, such as a power supply and input and output devices, to act as a microcomputer.

MODEM — stands for Modulator Demodulator. This is a device which allows two computers to talk to each other over the telephone. The computers usually use a cradle in which a telephone receiver is placed.

Monitor — this has two meanings in computer terms. One meaning is a television-like display. A monitor has no facility for tuning television programs, and usually the picture produced on a monitor is superior to that produced by an ordinary television. The second meaning of a monitor relates to ROM. The monitor of a computer is described as the information it has built in when you buy it. This information allows it to make decisions and carry out arithmetic computations.

Motherboard — a framework to which extra circuits can be added. These extra circuits often give the computer facilities which are not built-in, such as that of producing sound or of controlling a light pen.

MPU — an abbreviation for Microprocessor Unit.

N

Nano-second — a nano-second is one thousand billionth of a second, the unit of speed in which a computer or a memory chip is often rated.

Non-Volatile Memory — memory which is not lost when the computer is turned off. Some of the smaller computers such as the TRS-80 Pocket Computer have non-volatile memory. The batteries hold the program you enter for several hundred hours.

Not — a Boolean logic operation that changes a binary digit into its opposite.

Null String — a string which contains no characters. It is shown in the program as two double quote marks, without anything between them.

Numeric — pertaining to numbers as opposed to letters (that is, alphabetic). Many keyboards are described

as being alphanumeric which means both numbers and letters are provided.

O

Octal — a numbering system which uses eight as the base, and the digits 0, 1, 2, 3, 4, 5, 6 and 7. The Octal system is not used very much nowadays in microcomputer fields. The Hexadecimal system is more common (see Hexadecimal).

Operating System — the software or firmware generally provided with the machine that allows you to run other programs.

OR — an arithmetic operation that returns a 1, if one or more inputs are 1.

Oracle — a method of sending text messages with a broadcast television signal. A teletext set is required to decode the messages. Oracle is run by Independent Television Service in the UK, and a similar service — Ceefax — is provided by the BBC.

Output — information or data fed out by the computer to such devices as a TV-like screen, a printer or a cassette tape. The output usually consists of the information which the computer has produced as a result of running a program.

Overflow — a number too large or too small for the computer to handle.

P

Pad — see Keypad.

Page — often used to refer to the amount of information needed to fill one TV screen, so you can talk about seeing a page of a program, the amount of the listing that will appear on the screen at one time.

PASCAL — a high level language.

Peripheral — anything which is hooked onto a computer, for control by the computer, such as a disk unit, a printer or a voice synthesiser.

Port — a socket through which information can be fed out of or in to a computer.

Prestel — the British telecom name for a system of calling up pages of information from a central computer via the telephone and displaying them on a television screen. A similar commercial version in the United States is known as The Source.

Program — in computer terms program has two meanings. One is the list of instructions that you feed into a computer, and the second is used as a verb, as in 'to program a computer'.

PROM — stands for Programmable Read Only Memory. This is a device which can be programmed, and once it is then the program is permanent (also see EPROM and ROM).

R

Random Access Memory (RAM) — the memory within a computer which can be changed at will by the person using the computer. The contents of RAM are usually lost when a computer is turned off. RAM is the memory device that stores the program that you type in and also stores the results of calculations in progress.

Read-Only Memory (ROM) — in contrast to RAM, information in ROM cannot be changed by the user of the computer, and the information is not lost when the computer is turned off. The data in ROM is put there by the manufacturers and tells the computer how to make decisions and how to carry out arithmetic computations. The size of ROM and RAM is given in the unit K (see K).

Recursion — the continuous repetition of a part of the program.

Register — a specific place in the memory where one or more computer words are stored during operations.

Reserved Word — a word that you cannot use for a variable in a program because the computer will read it as something else. An example is the word TO. Because TO has a specific computer meaning, most computers will reject it as a name for a variable. The same goes for words like FOR, GOTO and STOP.

Routine — this word can be used as a synonym for program, or can refer to a specific section within a program (also see Subroutine).

S

Second Generation — this has two meanings. The first applies to computers using transistors, as opposed to first generation computers which used valves. Second generation can also mean the second copy of a particular program; subsequent generations are degraded by more and more noise.

Semiconductor — a material that is usually an electrical insulator but under specific conditions can become a conductor.

Serial — information which is stored or sent in a sequence, one bit at a time.

Signal — an electrical pulse which is a conveyor of data.

Silicon Valley — the popular name given to an area in California where many semiconductor manufacturers are located.

SNOBOL — a high level language.

Software — the program which is entered into the computer by a user which tells the computer what to do.

Software Compatible — this refers to two different computers which can accept programs written for the other.

Static Memory — a non-volatile memory device which retains information so long as the power is turned on, but does not require additional boosts of power to keep the memory in place.

Subroutine — part of a program which is often accessed many times during the execution of the main program. A subroutine ends with an instruction to go back to the line after the one which sent it to the subroutine.

T

Teletext — information transmitted in the top section of a broadcast television picture. It requires a special set to decode it to fill the screen with text information. The BBC service is known as Ceefax, the ITV service as Oracle. Teletext messages can also be transmitted by cable, for example the Prestel service in Britain or The Source in the United States.

Teletype — a device like a typewriter which can send information and also receive and print it.

Terminal — a unit independent of the central processing unit. It generally consists of a keyboard and a cathode ray display.

Time Sharing — a process by which a number of users may have access to a large computer which switches rapidly from one user to another in sequence, so each user is under the impression that he or she is the sole user of the computer at that time.

Truth Table — a mathematical table which lists all the possible results of a Boolean logic operation, showing the results you get from various combinations of inputs.

U

UHF — Ultra High Frequency (300-3000 megaHertz).

Ultra Violet Erasing — Ultra violet light must be used to erase EPROMs (see EPROM).

V

Variable — a letter or combination of letters and symbols which the computer can assign to a value or a word during the run of a program.

VDU — an abbreviation for Visual Display Unit.

Volatile — refers to memory which 'forgets' its contents when the power is turned off.

W

Word — a group of characters, or a series of binary digits, which represent a unit of information and occupy a single storage location. The computer processes a word as a single instruction.

Word-Processor — a highly intelligent typewriter which allows the typist to manipulate text, to move it around, to justify margins and to shift whole paragraphs if necessary on a screen before outputting the information onto a printer. Word-processors usually have memories; so that standard letters and the text of letters, written earlier, can be stored.

BIBLIOGRAPHY

Compiled by Tim Hartnell

The A to Z Book of Computer Games (McIntire, Thomas C, Tab Books, Blue Ridge Summit, Pa.).

This is a fine Tab book to give you program ideas and ready-to-run programs, although some of the games are a disappointment, such as the overly long Othello program which does not even play, but simply records the moves made by two human players. Others, however, such as Fivecard and Hotshot, are well written, and well worth entering into your microcomputer.

BASIC Computer Games (ed. Ahl, David, Creative Computing Press, Morristown, New Jersey).

This is a classic work, the source of more programming ideas than any other computer games book ever published. I had a meal with David Ahl one night in London after a PCW show and discussed the book. He said that he'd been in the personal computer field almost before there were personal computers, and while many of the games in this book do not seem startling now, the fact that people could write and play games for computer interaction at all seemed quite incredible in the late seventies. The Checkers program, and Life for Two are just a couple of the treasures you will find in this splendid program and idea source book.

BASIC Computer Programs for the Home (Sternberg, Charles D, Hayden Book Company, Inc., Rochelle Park, New Jersey).

Traditionally, home computers (when first purchased) have been used for playing games. One reason why they have not been used for more serious applications

stems from the lack of a readily available, comprehensive set of home applications programs that were easy to use and understand and that satisfied the practical requirements of the home. This book provides a set of programs to make your computer start earning its keep. The programs provide a good cross-section of practical applications; these have been designed so as not to rely upon the availability of tape or disk-storage devices. The programs cover a wide field, and are divided into a number of sections: home financial programs (including household expenses and income tax recording); car related programs (including fuel use and trip planning); 'Kitchen Helpmates' (including diet and meal planning programs); scheduling programs for home use (including a reminder calendar and a couple of programs which I imagine are designed to short circuit arguments about which television programs will be watched); and 'List programs for every purpose' (including Christmas cards, music collections and three versions of an addresses program).

The BASIC Handbook (Lien, David A, Compusoft Publishing, San Diego, California).

This is an encyclopedia of the BASIC language. Now that BASIC is so firmly established throughout the microcomputer world, it is necessary to make its many dialects understandable so that programs can be transported between different computers. When you have found exactly the program you've been looking for, it is very frustrating to be unable to run it on your computer. This book addresses that problem by discussing in detail just about every commonly used BASIC statement, function, operator and command. For the most part, BASIC words mean the same thing to every computer which recognises them. If a computer does not possess the capabilities of a needed or specified word, there are often ways to accomplish the same function by using another word, or combination of words. Although the handbook requires some

application to transform the information into usable form, it is a very valuable reference work indeed. Every BASIC word you have ever heard of (and many you may not have heard of, such as LE, NE, GOTO-OF, RES and TIME) is probably in the book. It may be of limited use to you in your early days of computing, but it should become an indispensable handbook once you get more involved in the subject.

Beat the Odds, Microcomputer Simulations of Casino Games (Sagan, Hans, Hayden Book Company, Inc., Rochelle Park, New Jersey).

The book explains how to play certain casino games (trente-et-quarante, roulette, chemin-de-fer, craps and blackjack) and gives complete program listings in BASIC with commentaries on systems and optimal strategies. Professor Sagan (Professor of Mathematics at North Carolina State University) says he wrote the book in an attempt to convince people that, in the long run, they could not win — except possibly at blackjack — and to explain some popular systems and their pitfalls, and above all to provide very realistic computer simulations of the games themselves. He has succeeded in his attempt. The listings are possibly longer than other computer versions of the same games, but this is because the Sagan versions strictly duplicate the odds involved in playing the game 'in real life', and cover all the eventualities that a real game can produce. The programs are well-structured, and an examination of the listings should give you ideas for improving your own programming.

Beginner's Guide to Chess (Keene, Raymond, Pelham Books Ltd, London).

An ideal guide to simple chess-playing techniques which you can turn into algorithms if you intend to write a chess program of your own.

The Calculator Game Book for Kids of All Ages (Hartman, Arlene, Signet Books, New York).

The book's title says it all, and the names of the games

(which include Fibonacci Follies, Stretch to Sixty and Casting Out 9s) suggest the book's contents. There are some worthwhile brain-stretching puzzles, and 15 or so ideas definitely calling for conversion to computer games.

33 **Challenging Computer Games for TRS-80/Apple/PET** (Chance, David, Tab Books, Blue Ridge Summit, Pa.).

Even if you don't have any of the three computers named in the title, you will still find the book a goldmine of ideas for your own development, and many of them will run, with minimal alteration, on any BASIC-using computer. Particularly commendable programs are Life Support, Scrambled Eggs and Tank Assault.

Communicating with Microcomputers (Witten, Ian H., Academic Press, London).

This is an introduction to the technology of man/computer communications for the non-specialist. By placing particular emphasis on low-cost techniques associated with small systems and personal computers, the reader's attention is focused on the positive nature of the 'microprocessor revolution' — how machines can help people — rather than the negative aspects which are often highlighted in the non-technical press. The level of the book is suitable for the layman with some acquaintance with electronics. The final section, on speech communication, provides the most fascinating reading.

Computer Appreciation (Fry, T.F., Newnes, Butterworths, 1975).

A fairly 'straight' but useful overview of computer operation, and business applications. Designed to be used as a text for a course of business studies, the book covers a wide range of topics from a short account of the historical development of calculating devices, through computer hardware and programming, to the

organisation of a modern data-processing department. It concludes with a brief consideration of the applications of computers and a discussion on the effects of computers upon management matters. It is surprisingly undated, despite the extraordinary increase in hardware availability and capability since the book was written.

The Computer Book: An Introduction to Computers and Computing (Bradbeer, Robin; De Bono, Peter; Laurie, Peter; BBC Publications).

This book was published in conjunction with the BBC television series 'The Computer Programme', first transmitted on BBC2 from January 1982, and produced by Paul Kriwaczek. I discussed this book with Robin Bradbeer while it was being written, and he told me that the BBC editors were ruthless in pointing out any use of jargon. They insisted, said Robin, that nothing could be taken for granted. This insistence has resulted in a book which anyone can understand. It assumes nothing, not even the knowledge of how to use a shift key — or the effect of using it — on a typewriter. The many illustrations and photographs break up the text, which gives a detailed introduction to computers, especially micros, and their possible applications.

Computer Games for Businesses, Schools and Homes (Nahigian, J Victor and Hodges, William S. Winthrop Publishers Inc., Cambridge, Mass.).

Some of the programs are a little thin for the size and price of the book, but the best ones are well worth adapting to run on your computer. The inclusion of long, clear sample run printouts ensures that you know exactly what the programs will do before you run them. The Tennis and Star Trek programs are especially good.

Dice Games Old and New (Tredd, William E., Oleander Press, Cambridge).

This will give you enough clearly written games

explanations to keep you creating games programs on your microcomputer for a long time to come.

The Electronic Calculator in Business, Home and School (Birtwistle, Claude, Elliot Right Way Books, Kingswood, Surrey).

To get the best out of a calculator, you need to understand the mathematics which lies behind the operations. That is the purpose of the book, and in general it succeeds in this aim. The maths involved is, however, fairly simple and basic, since the book was written with a wide range of people in mind — the pupil at school, the student at college, the business person and the householder. It is a practical book which should be read and worked through with a calculator to hand.

Everyman's Indoor Games (Brandreth, Gyles, J M Dent and Sons Ltd, London).

If you're looking for games to convert into computer programs, ignore the chapters entitled Parlour Games and Children's Party Games and stick to the rest of the book, a treasure trove of games concepts which are certainly worth using as a starting point. Fox and Geese, Poker Dice and Billiards, as described in the book, are only a few of the programs you might write after reading it.

Games and Puzzles for Addicts (Millington, Roger, M and J Hobbs, Walton-on-Thames).

These games and puzzles first appeared in the weekly computer news-magazine 'Datalink', so they are especially likely to appeal to computer buffs. There are many ideas here that can be converted into games to be played with the computer.

Games for Home, Travel and Parties (Jensen, Helen, Western Publishing Company Inc., Racine, Wisconsin).

Aimed squarely at children, this book gives some games which are simple to program (these include Snakes, Lift-Off and Fish), and contains a complete chapter on how to play chess.

Home Computers, Questions and Answers, Hardware (Didday, Rich, dilithium Press).

The book has two main purposes. Firstly, it is intended to give readers a real feeling for what is involved in home computing, so that they can make rational decisions before buying equipment. Secondly, it is intended to give people who have no specialised knowledge of computing a general background to the subject, and specifically to microcomputers. The book succeeds in imparting enough information to ensure you will have little trouble understanding articles about advanced projects in computer hobbyist magazines, advertisements for home computing equipment, or other people who do have advanced computer knowledge.

Inside BASIC Games (Mateosian, Richard, Sybex).

This book is a guide, albeit a slightly overwritten one, for anyone who wishes to understand computer games. You will learn how to write interactive programs in BASIC and how the principles of systems development are applied to small computers. The book also looks at how the features of specific small computer systems have been supported in BASIC. The sections of the book include: Arithmetic Games, Guessing Games, Time Games, Date Games, Taxman, and programming in 'Free BASIC', a structured BASIC that is translated manually into the actual BASIC instructions to be entered into the computer. Free BASIC is not a language; it is a program description medium (like flowcharts) that has no line numbers, and uses symbolic names for subroutines. Additional chapters look at The Match-Up Game, Craps and Alien Life. If you can contend with the verbiage, you will find this book well worthwhile.

An Introduction to Personal and Business Computing (Zaks, Rodnay, Sybex).

I had lunch with Rodnay in London during a PCW show and he told me that he thought current American predictions on the growth of the personal computer

field were grossly pessimistic. He pointed out that the predictions current in 1978, when he wrote this book, have been proved so inaccurate that would-be prophets should take warning and assume that whatever they say will be wrong by a factor of 10 or 100. Despite its age — and computer books do age uncommonly quickly — this book is a good introduction to the field, explaining in clear, snappy English the fundamentals of computer operation. Dr Zaks also gives suggestions on what to look for when buying a computer.

Microsoft BASIC (Knecht, Ken, dilithium Press, Forest Grove, Oregon).

This book presents a complete introduction to programming in Microsoft BASIC. The concepts presented are illustrated with short, working programs. By starting with the simplest and most commonly used commands, and then progressing on to the more complex BASIC commands, Mr Knecht shows how the more powerful versions of the language can save valuable programming time and effort.

The Personal Computer Book (Bradbeer, Robin, Input Two-Nine).

The title says it all. Robin is deeply involved in the microfield in Great Britain. He started the North London Polytechnic Computer Fairs, assisted the BBC with their microcomputer television show (and co-authored *The Computer Book*, published by the BBC), and edited the monthly publication *Educational Computing*. This gave him a strong background from which to write the book. It explains what a computer is and how it works; it elucidates the mysteries inside the 'black boxes' which make up a computer; and it gives a number of very useful appendices, including bus standards, manufacturers and distributors, magazines, a selected bibliography (compiled by Richard Ross-Langley, of *Mine of Information*) and a glossary. But perhaps the most interesting and useful section of the book is the part which describes, in some

detail, the majority of computer systems available on the British market, their price and their capabilities. Overall, this is a very impressive source book.

Personal Computers: What They Are and How to Use Them (Wels, Byron G., Trafalgar House Publishing).

A great deal has happened in the computer world since this book was written in 1978, but there is still a great deal of value and interest in it. The book details some of the personal computers available, and the improvements that are likely to be made in the future. It explains, in layman terms, how a computer works, and how to make it work for you. There is also material on the construction and maintenance of small computer systems.

Play the Game (Love, Brian, Michael Joseph and Ebury Press, London).

This is a splendid book, containing 40 or so full-size reproductions of Victorian (and pre-Victorian) board games, many suitable for playing against a computer. It is even possible to use the boards in the book (the computer then tells you where it is moving on this external board) rather than write a routine within the program to display a board.

The Pocket Calculator Games Book (Schlossberg, Edwin and Brockman, John, Wilton House Publications Ltd., London).

There are many ideas here suitable for conversion into computer games.

57 Practical Programs and Games in BASIC (Tracton, Ken, Tab Books, Blue Ridge Summit, Pa.).

There are more serious programs than games (of the Chi-Square Evaluation and Fibonacci Numbers variety) in this book. They are well-programmed, and supported by adequate (if brief) documentation, and by flow-charts. The Space Wars programs (versions one and two) at the end of the book are particularly good.

Problems for Computer Solution (Rogowski, Stephen J, Creative Computing Press, Morristown, New Jersey).

This outlines over 50 simple (and a few not-so-simple) problems which can be solved by writing a program. There are both teacher and student editions of this book; the teacher edition has a suggested program and sample run printout to solve the difficulty. It is an excellent source for educational ideas.

Stimulating Simulations (Engel, C.W., Hayden Book Company, Inc., New Jersey).

Here, according to the cover, are '12 unique programs in BASIC for the computer hobbyist'. Inside you will find some fascinating programs: Forest Fire, Rare Birds and The Devil's Dungeon are three you are sure to enjoy playing, while Diamond Thief (the computer decides who has committed the crime, then challenges you to discover which of the suspects is guilty) is both well written and tightly programmed.

TAKE TWO! 32 Board Games for 2 Players (Tapson, Frank, A & C Black, London).

This book is aimed at children, but it does give many fascinating ideas that could be transformed into computer games (even if some of them are duplicated elsewhere in the book).

24 Tested, Ready-to-Run Game Programs in BASIC (Tracton, Ken, Tab Books, Blue Ridge Summit, Pa.).

Tab Books are prolific publishers in the microcomputer program field, and their books are deservedly successful. If nothing else, reading a book such as this one will give you ideas for structuring programs neatly, and for writing them to ensure the maximum compatibility between different versions of BASIC. Many of the games, such as Auto Rally and Capture the Alien, are (despite their weak titles) well thought out, carefully constructed programs.

1001 Things to Do with Your Personal Computer (Sawush, Mark, Tab Books, Blue Ridge Summit, Pa.).

I bought this book at a computer fair in Atlanta, and

read it (making notes, and turning down page corners) on the flight to London. And I still hadn't finished it on arrival. If you feel you have come to the end of possible applications for your computer, buy this book and discover that you have barely scratched the surface. It tells you about writing music and stories, aiding a mechanic or a carpenter, solving simultaneous equations, astrology, and much, much more.

The World Computer Chess Championship (Hayes, Jean E., and Levy, David N.L., Edinburgh University Press, Edinburgh).

This is a fascinating account of the world's first machine versus machine chess championship, held in 1974, when the dozen or so computer programs taking part were the only chess programs in existence. The games are analysed in detail, and the final section of the book outlines a board-numbering system which you could use if you're considering writing your own chess program. The book makes you realise how far the computer world has come in only a few years.

The Virgin Computer Games Series

More GAMES FOR YOUR ZX SPECTRUM

More than 20 challenging programs, each one especially written for the series and guaranteed to provide hours of entertainment.

TOWERING INFERNO (the fate of people trapped in a blazing skyscraper depends on you); BAZOOKA! (take aim and destroy the enemy's armoured tanks); DROID (help the robot escape from a maze — or risk an android's wrath!); A DAY AT THE CHASES (make your fortune on the racecourse); and AIR SEA RESCUE (save a drowning man or his ghost will haunt you forever).

MORE GAMES FOR YOUR ZX SPECTRUM will improve your programming skills as you follow the instructions to put each of the programs into your machine, and comes complete with a brief dictionary of computer terms, a selective bibliography and some hints on how to improve and extend the programs in the book.

Programs of originality and quality for all the family.

Virgin

0 907080 99 5

£3.50