The Virgin Computer Games Series

GAMES FOR YOUR AMSTRAD

£££££'s of entertaining games for only £2.95

Series Editor: Tim Hartnell

Edward Way

GAMES FOR YOUR AMSTRAD

By Edward Way

Best Wishes,

C. Gifford.

GAMES FOR YOUR AMSTRAD

By
Edward Way

Virgin Books

First published in Great Britain in 1984 by Virgin Books Ltd, 61-63 Portobello Road, London W11 3DD.

ISBN 0 86369 077 7

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 — THE EDITOR

Tim Hartnell is the most widely-published computer author in the world. Founder of the National ZX Users' Club, and founding editor of *ZX Computing* magazine, Tim has been involved over the years in a wide variety of computer activities. His published works include *The Personal Computer Guide* (Virgin Books) and *Tim Hartnell's QL Handbook* (Interface Publications).

EDWARD WAY — THE AUTHOR

Edward Way is in his mid thirties and works as a Sales Manager with a well-known retailing firm. He has a degree in commercial technology and it was while studying for his degree that he first came into contact with computers, albeit the very primitive models of a decade and a half ago. Since he caught the computer bug some while back, it is surprising that the Amstrad is only Edward's second machine (the first being a Colour Genie). Most of his leisure time is spent with his wife and two daughters.

SUE WALLIKER — THE ILLUSTRATOR

Sue Walliker is a freelance illustrator.

ACKNOWLEDGEMENTS

The author wishes to acknowledge the tremendous advice and assistance that he received from Clive Gifford in the creation of this book. The author would also like to thank Tim Hartnell and Peter Shaw for their help and, of course, his family for putting up with his new-found author status.

TO CLIVE, WHO GAVE ME MY FIRST BIG BREAK

CONTENTS

Author's Introduction

The Amstrad is a fantastic machine with all the features you might hope to find in a machine costing double the price. I have tried, with the help of other contributors, to provide a book that will give you a lot of fun and enjoyment. The programs cover all aspects of games playing — arcade games, adventure games, games of chance and games of skill are all well-represented in this book. I have also added a couple of graphics and sound demonstrations to show you what can be achieved when you have an Amstrad for a friend.

I hope you find this book a lot of fun and that, in enjoyment terms, it pays for itself in a few hours of action-packed excitement. And finally, congratulations on choosing such a great machine!

Edward Way
London
July 1984

Editor's Introduction

Typing in a computer program is like opening an unknown door. You do not know until you actually open the door — or, in our case, run the program — what experience is waiting for you. Of course, the sign on the door has given you some indication, but nothing can equal first-hand experience.

You do not know precisely what experiences are waiting for you in the great programs in this book. Of course, if the introduction says you're entering a space game, it's very likely the program won't play 'Guess My Number' when you get it up and running. But the listing rarely hints at the computer's game-playing strategy, or the screen display, or the fun that is waiting for you.

This book has a number of unknown doors — doors leading into outer space and into the fiendish worlds of computer intelligence, wizards and Adventure.

We've provided the doors . . . and the keys. All you have to do to turn the lock is type in the program, and run it. Whatever you find behind each door, I guarantee you won't be disappointed.

Tim Hartnell
Series editor
London
March 1984

Program Notes

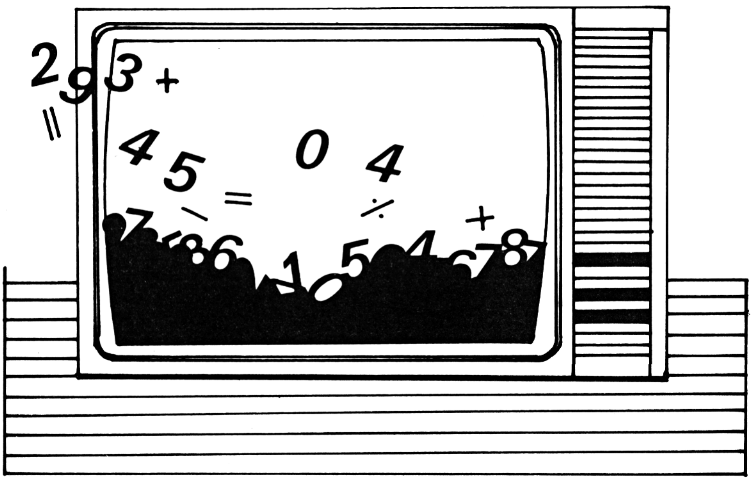
Before using each of these programs the machine needs to be reset. To reset the machine, hold down the CTRL key, the SHIFT key and the ESC key simultaneously. Unless specified, keep the machine in Upper Case characters.

To ensure that the games are totally random, type in RANDOMIZE TIME before using the game. You will find that some of the games have this command in the program already.

NUMBER CRUNCHER

In this game, you take the role of a hungry little fellow who has escaped from that all-too-familiar maze. You have been to school and have learnt about maths and now you wish to try your hand at gobbling numbers. You control your character using the ',' and '.' keys for left and right, and the 'Q' and 'A' keys for up and down. You must eat the red numbers as they appear. Each number adds its own value to your score, so if you have a choice, then go for the higher numbers.

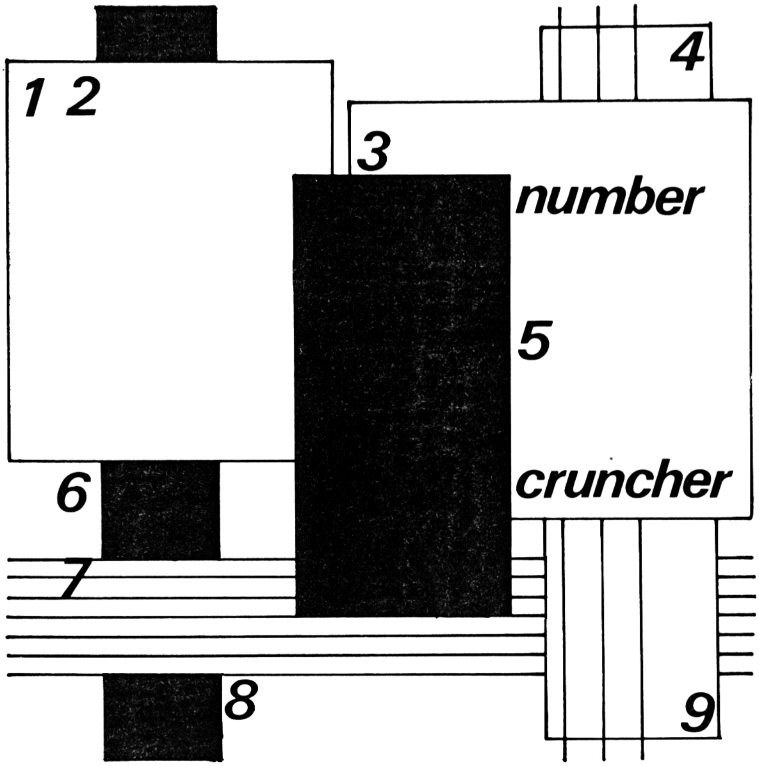
So far the game may have sounded easy, but there are two factors that make it much more difficult (as well as extremely addictive!). Firstly, many ghosts appear which you must avoid bumping into. Secondly, once you press a direction key, movement is continuous; your creature won't stop moving until it hits a wall, or until another direction key is pressed . . . or until it bumps into a hungry ghost.



```
10 REM***************************
20 REM*****NUMBER CRUNCHER*****
30 REM***************************
40 GOSUB 480
50 GOSUB 390
60 REM****MAIN LOOP****
70 A$=INKEY$
80 IF A$="Q" AND X>3 THEN X=X-1:C$=CHR
$(198):P=1
90 IF A$="A" AND X<24 THEN X=X+1:C$=CH
R$(196):P=2
100 IF A$="," AND Y>2 THEN Y=Y-1:C$=CH
R$(197):P=3
110 IF A$="." AND Y<38 THEN Y=Y+1:C$=C
HR$(199):P=4
120 IF A$="" AND P=1 AND X>3 THEN X=X-
1
130 IF A$="" AND P=2 AND X<24 THEN X=X
+1
140 IF A$="" AND P=3 AND Y>2 THEN Y=Y-
1
150 IF A$="" AND P=4 AND Y<38 THEN Y=Y
+1
160 LOCATE YY,XX:PRINT" "
170 IF S(Y,X)>0 AND S(Y,X)<10 THEN SC=
SC+S(Y,X):SOUND 1,150,4,4:S(Y,X)=0
180 IF S(Y,X)>9 THEN GOTO 260:REM***LO
SE***
190 LOCATE Y,X:PRINT C$
200 LOCATE 7,1:PRINT SC
```

```
210 XX=X:YY=Y
220 IF RND>0.96 THEN A=INT(RND*36)+2:B
=INT(RND*21)+3:LOCATE A-1,B:C=INT(RND*
9)+1:S(A,B)=C:PEN 3:PRINT C:PEN 1
230 IF RND>0.84 THEN F=INT(RND*36)+2:G
=INT(RND*21)+3:LOCATE F,G:PRINT CHR$(2
24):S(F,G)=10
240 SOUND 1,600,1,4
250 GOTO 60
260 REM*****LOSE*****
270 SOUND 2,600,16,7,1,12,2
280 SOUND 4,600,80,7,9,13,6
290 SOUND 1,600,30,7,4,1,15
300 CLS
310 LOCATE 17,12
320 PRINT"OUCH!!!"
330 PRINT:PRINT"YOUR SCORE WAS ";SC
340 PRINT:PRINT:PRINT"ANOTHER GAME? (Y
/N)"
350 D$=INKEY$:IF D$="" THEN 350
360 IF D$="Y" THEN RUN
370 IF D$="N" THEN PRINT"OK>":END
380 IF D$<>"" THEN 350
390 REM***SCREEN***
400 MODE 1
410 CLS
420 PRINT"SCORE:";SC;
430 PRINT TAB(12);"NUMBER CRUNCHER"
440 PRINT"################################
###########"
```

```
450 FOR T=1 TO 22:PRINT"#";TAB(39);"#"
:NEXT T
460 PRINT"################################
###########"
470 RETURN
480 REM***INITIALISATION***
490 DIM S(40,25)
500 X=20:Y=13:SC=0
510 XX=X:YY=Y
520 F=0:G=0:A=0:B=0
530 RETURN
```



MEPHISTO THE MAGICIAN

This program incorporates a classic magician's trick that is fun for everyone, as well as some realistic high resolution graphics. Just follow the instructions in the program.

```
10 REM************************
20 REM******MEPHISTO********
30 REM************************
40 CLS
50 REM******GRAPHICS******
60 INK 1,26
70 INK 0,0
80 BORDER 0
90 MODE 1
100 PLOT 40,40
110 DRAW 20,60
120 DRAW 360,400
130 DRAW 380,380
140 DRAW 40,40
150 PLOT 80,80
160 DRAW 60,100
170 PLOT 340,340
180 DRAW 320,360
190 FOR T=80 TO 40 STEP-2
```

```
200 PLOT T,T
210 DRAW T-20,T+20
220 NEXT T
230 FOR T=380 TO 340 STEP-2
240 PLOT T,T
250 DRAW T-20,T+20
260 NEXT T
270 PLOT 380,40
280 DRAW 550,40
290 DRAW 550,250
300 DRAW 640,250
310 DRAW 290,250
320 DRAW 380,250
330 DRAW 380,40
340 PLOT 380,100
350 DRAW 550,100
360 DRAW 550,140
370 DRAW 380,140
380 FOR T=140 TO 100 STEP-2
390 PLOT 380,T
400 DRAW 550,T
410 NEXT T
420 LOCATE 3,3
430 PRINT "MEPHISTO"
440 PRINT
450 PRINT "the Magician"
460 WHILE INKEY$=""
470 SOUND 1,INT(RND*650)+25
480 FOR T=1 TO 600:NEXT T
490 WEND
```

```
500 REM******THE MAIN PROGRAM******
510 LOCATE 1,25
520 INPUT"THE MAGICIAN'S NAME PLEASE";
P$
530 SOUND 1,100,100
540 FOR T=1 TO 25:PRINT
550 FOR P=1 TO 200:NEXT P
560 NEXT T
570 LOCATE 3,3
580 PRINT "I PERFORM THIS TRICK WITH H
ELP FROM MY  COLLEAGUE ";P$
590 PRINT:PRINT:PRINT"A MEMBER OF THE
AUDIENCE MUST BE CHOSEN AND MUST WRITE
```

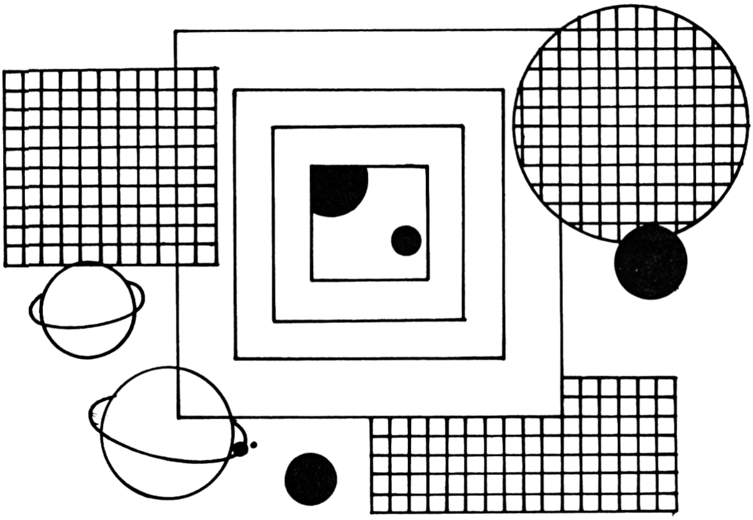
```
 THEIR AGE DOWN ON PAPER  WITHOUT SHOW
ING ";P$;"."
600 WHILE INKEY$=""
610 WEND
620 CLS:PRINT:PRINT
630 PRINT "THE PERSON MUST THEN SUBTRA
CT FROM      THEIR AGE THEIR FAVOURITE
 SINGLE DIGIT  NUMBER"
640 PRINT:PRINT"THIRDLY THE PERSON MUS
T MULTIPLY THE    ANSWER BY 9 AND ONCE
 COMPLETED THEIR    AGE SHOULD BE ADDE
D"
650 PRINT:PRINT:PRINT"PRESS A KEY ONCE
 THIS IS DONE"
660 WHILE INKEY$=""
670 WEND
680 CLS
690 SOUND 1,INT(RND*100)+40,90
700 SOUND 2,INT(RND*100)+140,120
710 FOR T=1 TO 10
720 LOCATE 12,12
730 PRINT "                  "
740 FOR P=1 TO 200:NEXT P
750 LOCATE 12,12
760 PRINT "PLEASE ENTER TOTAL"
770 FOR P=1 TO 200:NEXT P
780 NEXT T
790 INPUT N$
800 A=VAL(LEFT$(N$,2))
810 B=VAL(RIGHT$(N$,1))
```

```
820 CLS
830 LOCATE 1,11
840 PRINT TAB(6);"######################
########"
850 PRINT TAB(6);"#";TAB(33);"#"
860 PRINT TAB(6);"######################
########"
870 LOCATE 11,12
880 PRINT "THE ANSWER IS ";A+B;
890 FOR T=300 TO 40 STEP -20
900 SOUND 1,T,35
910 SOUND 2,T+5,35
920 SOUND 3,T-5,35
930 NEXT T
940 PRINT:PRINT:PRINT:PRINT
950 PRINT "WOW!!! AMAZING!!! FANTASTIC
!!!"
960 PRINT "HOW DOES MEPHISTO DO IT? (W
hat modesty)"
970 PRINT:PRINT:PRINT
980 PRINT"Would you like ";P$;" and my
self"
990 PRINT "to perform the trick again?
  (Y/N)"
1000 A$=INKEY$
1010 A$=UPPER$(A$)
1020 IF A$="Y" THEN RUN
1030 IF A$="N" THEN PRINT:PRINT"O.K.":
END
1040 IF A$="" THEN 1000
```

LASER LIGHT

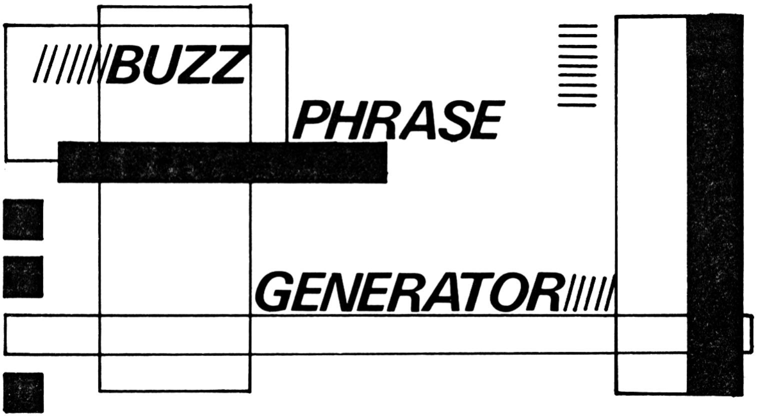
This simple program produces an effective display that can be used in your own programs for hyperspace or time travel.

```
10 REM******LASER LIGHT******
20 MODE 2
30 INK 1,24,3
40 PAPER 0
50 CLS
60 CLG
70 BORDER INT(RND*14)+1
80 PLOT 320,200
90 DRAW RND*640,RND*400
95 SOUND 1,INT(RND*500)+10
100 FOR T=1 TO 200:NEXT T
110 GOTO 70
```



BUZZ-PHRASE GENERATOR

I'm sure you have often heard high-blown phrases which sound very impressive but often mean nothing. This has been termed 'buzz-phrase'. This program produces some excellent examples of buzz-phrase which would apply to an office or computer situation.



```
10 REM**********************************
20 REM*****BUZZ-PHRASE GENERATOR*****
30 REM**********************************
40 DIM A$(10,3)
50 FOR P=1 TO 3:FOR T=1 TO 10
60 READ A$(T,P)
70 NEXT T
80 NEXT P
90 CLS
100 FOR G=1 TO 3
```

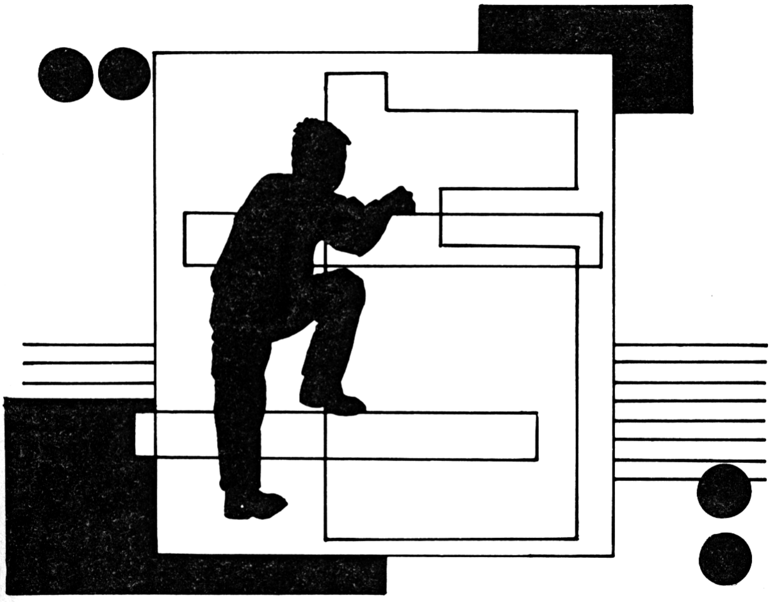
```
110 T=INT(RND*10)+1
120 SOUND 1,(400-(100*G))
130 PRINT A$(T,G);" ";
140 NEXT G
150 PRINT:PRINT
160 G$=INKEY$
170 IF G$="" THEN 160 ELSE GOTO 100
180 DATA ONGOING,TEMPORARY,INTERMITTEN
T
190 DATA UNFORSEEN,SHORT-TERM,LONG-TER
M
200 DATA DREADFUL,UNBELIEVABLE,UNEXPEC
TED
210 DATA PERMANENT,MAN-POWER,MACHINE T
IME
220 DATA COMPUTER,SOFTWARE,HARDWARE
230 DATA FIRMWARE,CASH-FLOW,RESOURCES
240 DATA PAPER CLIP,COFFEE,SHORTAGE
250 DATA SURPLUS,CRISIS,FREEZE
260 DATA STOPPAGE,BREAKDOWN,ASSESSMENT
270 DATA REASSESSMENT,CHECK,PROBLEM
```

BUZZ PHRASE GENERATOR

ALL SYSTEMS GO

Here you are playing a game of Zappo Invaders when suddenly your Amstrad switches off. You have suffered from a systems crash. Unfortunately, it is Sunday and there are no computer engineers available so you will have to mend it yourself. However, you do have a special shrink beam that reduces you to the size of an ant. Once you are this small, you can enter the computer and find out what is wrong with it.

There are a number of problems with your machine (it must have been a pre-production model) and they must be put to rights. I am not going to give anything else away except to say that the command 'H' is for help. Good luck, you'll need it!



```
10 REM***************************
20 REM******ALL SYSTEMS GO*******
30 REM***************************
40 GOSUB 1070
50 REM****MAIN LOOP****
60 IF PF=12 THEN GOTO 1020
70 CLS
80 BORDER INT(RND*27)
90 SOUND 1,INT(RND*700)+50
100 PRINT:PRINT
110 PRINT"  YOUR POSITION:"
120 PRINT
130 IF L(P)=1 THEN PRINT"ON PRINTED CI
RCUIT BOARD NUMBER ";INT(RND*30)
140 IF L(P)=2 THEN PRINT "BALANCING AL
ONG A WIRE"
150 IF L(P)=3 THEN PRINT "IN THE MIDDL
E OF SOME RESISTORS"
160 IF L(P)=4 THEN PRINT "IN THE RAM C
HIPS"
170 IF L(P)=5 THEN PRINT "STANDING ON
ONE OF THE MICRO'S          INTERFACES
"
180 IF L(P)=6 THEN PRINT "IN THE ROM"
190 IF L(P)=7 THEN PRINT "ENTANGLED IN
 A MASS OF CIRCUITRY"
200 IF L(P)=8 THEN PRINT "BY A CIRCUIT
 BREAKER"
210 IF L(P)=9 THEN INK 1,15,6 :PRINT"G
ASP!":PRINT"ON A POWER LINE":GOTO 590
```

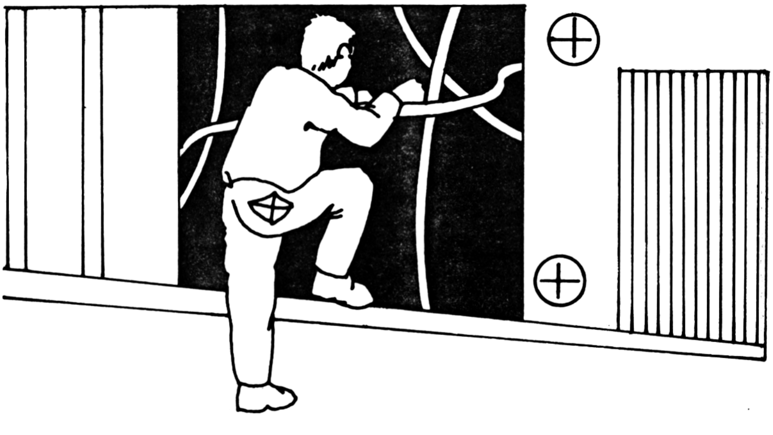
```
220 IF L(P)=10 THEN PRINT "NEXT TO A D
IRTY CONNECTION"
230 IF L(P)=11 THEN PRINT "WITHIN A FA
ULT-RIDDEN ROM CHIP"
240 IF L(P)=12 THEN PRINT "AT THE CORN
ER OF A PCB WHEN SUDDENLY, A BUG ATTAC
KS..."
250 IF L(P)=13 THEN PRINT "A DISCOLOUR
ED HEAT SINK"
260 PRINT:PRINT:PRINT
270 PRINT:PRINT"What do you want to do
 now?"
280 PRINT
290 INPUT "  N,S,E,W,SC,H,C,D";A$
300 PRINT:PRINT
310 A$=UPPER$(A$)
320 IF A$<>"N" AND A$<>"S" AND A$<>"E"
 AND A$<>"W" AND A$<>"SC" AND A$<>"H"
AND A$<>"C" AND A$<>"D" THEN PRINT" NO
 SUCH CHOICE, FOOL
":GOTO 410
330 IF A$="H" THEN GOSUB 460
340 IF A$="N" AND P>10 THEN P=P-10
350 IF A$="S" AND P<91 THEN P=P+10
360 IF A$="E" AND P<100 AND P/10<>INT(
P/10) THEN P=P+1
370 IF A$="W" AND P>1 AND P/1<>INT(P/1
0)+1 THEN P=P-1
380 IF A$="SC" THEN WINDOW #1,10,30,20
,23:PAPER #1,3:CLS #1:PRINT #1,"MOVES
```

```
";M:PRINT #1,"PARTS FIXED ";PF:PRINT #
1,"SCORE THUS...";
INT((PF*25)-M)
390 IF A$="C" THEN GOSUB 730
400 IF A$="D" THEN GOSUB 820
410 LOCATE 12,25
420 PRINT "PRESS A KEY"
430 WHILE INKEY$=""
440 WEND
450 M=M+1:GOTO 50
460 REM***HELP FEATURE****
470 CLS
480 MODE 0
490 PRINT TAB(8);"HELP"
500 PRINT:PRINT"N,S,E,W = DIRECTIONS"
510 PRINT:PRINT"SC      = SCORE"
520 PRINT:PRINT"H       = HELP"
530 PRINT:PRINT"C       = CLEAN"
540 PRINT:PRINT"D       = DEBUG"
550 PRINT:PRINT:PRINT"PRESS ANY KEY"
560 WHILE INKEY$="":WEND
570 MODE 1
580 RETURN
590 REM***HIGH VOLTAGE****
600 SV=0
610 B$=CHR$(INT(RND*26)+65)
620 PRINT:PRINT"YOU MUST PRESS ";B$;"
QUICKLY"
630 PRINT TAB(16);CHR$(208)
640 FOR T=1 TO 1000
```

```
650 IF INKEY$=B$ THEN SV=1:T=1000
660 NEXT T
670 IF SV=1 THEN PRINT:PRINT"YOU MADE
IT, BUT ONLY JUST IN TIME!":INK 1,18:G
OTO 220
680 PRINT:PRINT:PRINT
690 PRINT "YOU HAVE BEEN ELECTROCUTED!
"
700 SOUND 1,25,250,7
710 PRINT:PRINT:PRINT
720 END
730 REM****CLEAN****
740 IF L(P)=13 THEN PRINT "THE HEAT SI
NK BURNS A HOLE RIGHT THROUGHYOU!!!"
750 IF L(P)=13 THEN FOR T=600 TO 1500
STEP 20:SOUND 1,T,6,7:NEXT T
760 IF L(P)=13 THEN SOUND 1,1512,500,7
770 IF L(P)=13 THEN END
780 IF L(P)=10 THEN PRINT "GOOD WORK!
THE CONNECTION IS NOW FULLY  WORKING."
790 IF L(P)=10 THEN PF=PF+1:SOUND 1,50
:L(P)=INT(RND*3)
800 M=M+5
810 RETURN
820 REM****DEBUG****
830 IF L(P)=11 THEN PRINT "IT TOOK SOM
E TIME BUT YOU EVENTUALLY    SORTED OU
T THE BUGS":PF=PF+1
840 IF L(P)=11 THEN L(P)=6
850 IF L(P)=12 THEN GOTO 890
```

```
860 M=M+7
870 RETURN
880 REM*****THE GREAT BATTLE*****
890 B=INT(RND*7)+1
900 Y=INT(M/7)
910 YY=Y:BB=B
920 WHILE B>0 AND Y>0
930 IF INT(RND*9)>INT(RND*6) THEN PRIN
T "YOU SCORE A HIT":B=B-1
940 IF INT(RND*9)<INT(RND*8) THEN PRIN
T "IT FIGHTS BACK...":Y=Y-1
950 FOR T=1 TO 1000:NEXT T
960 WEND
970 PRINT:PRINT
980 IF B=0 THEN PRINT "YOU HAVE DEFEAT
ED IT.":PF=PF+1
990 IF B=0 THEN M=M+(10*(YY-BB)):L(P)=
1:RETURN
1000 IF Y=0 THEN PRINT "IT'S ALL OVER
FOR YOU,BEATEN BY A ROM   BUG!!!"
1010 IF Y=0 THEN FOR T=1 TO 12:SOUND 1
,300:SOUND 1,700:NEXT T:END
1020 REM****WIN****
1030 INK 1,2,26
1040 CLS
1050 FOR T=1 TO 24:PRINT TAB(T);"}}WEL
L DONE{{":SOUND 2,400-(12*T),50:SOUND
1,395-(12*T),50:NEXT T
1060 GOTO 1060
1070 REM****INITIALISATION****
```

```
1080 P=34
1090 INK 1,18
1100 INK 0,0
1110 BORDER 6
1120 M=0:PF=0:B$="":A$=""
1130 DIM L(100)
1140 FOR T=1 TO 100
1150 READ L(T)
1160 NEXT T
1170 DATA 10,5,5,5,5,5,1,3,7,2
1180 DATA 8,1,12,4,1,2,9,7,8,5
1190 DATA 7,2,4,3,13,7,1,9,6,5
1200 DATA 3,10,6,1,1,9,7,1,13,1
1210 DATA 2,7,8,1,2,3,10,7,3,6
1220 DATA 8,9,1,12,1,6,6,6,1,12
1230 DATA 4,4,13,8,2,1,11,6,4,4
1240 DATA 4,4,3,11,9,6,7,1,4,1
1250 DATA 10,4,1,2,2,7,3,9,6,6
1260 DATA 3,1,12,2,10,5,5,5,12,6
1270 RETURN
```

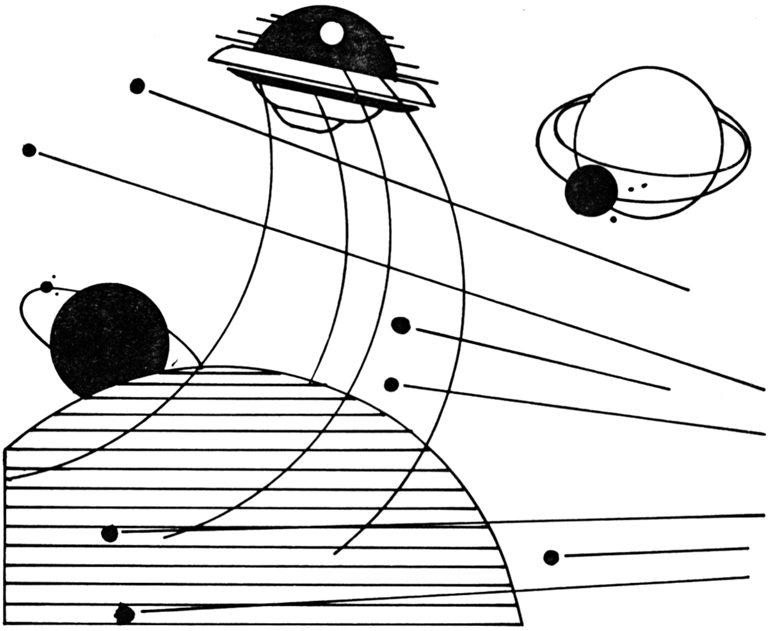


PROTECTOR

Once again, you must save Earth from a hoard of marauding aliens. Yes, I know you might rather play football but, believe me, this is more important. The infamous aliens are back and with a new plan. They have a 50-Megaton Nuclear Fission Laser Blasting Wonder Bomb that can blow the planet to pieces.

The aliens have cleverly manufactured enough bombs to rain down on planet Earth and so put the lives of you and your family in danger. Armed only with a High Velocity, Multi-Function Projectile Receptacle (a basket to you or me), you must catch the bombs as they fall. The character on the screen which runs very charmingly in both directions along the base of the screen is controlled by the right and left cursor keys.

Good luck, your planet needs you . . .



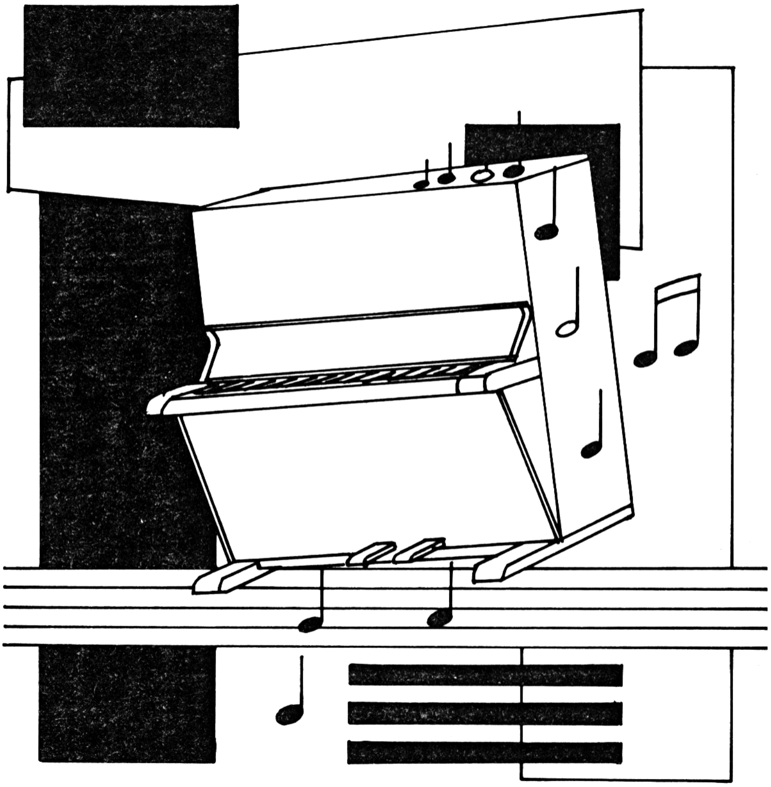
```
10 REM**********************
20 REM*****PROTECTOR*******
30 REM**********************
40 RANDOMIZE TIME
50 GOSUB 590
60 MODE 1
70 KEY DEF 8,1,55
80 KEY DEF 1,1,57
90 SPEED KEY 5,3
100 BORDER 0
110 PAPER 12:PEN 1
120 X=INT(RND*38)+1:Y=2
130 F=248
140 CLS
150 LOCATE 14,1
160 PEN 3
170 PRINT"PROTECTOR"
180 FOR H=1 TO 1200:NEXT H
190 PEN 1
200 LOCATE 1,25
210 FOR Z=1 TO 40:PRINT CHR$(207);:NEX
T Z
220 T=20:SC=0
230 REM*********GAME ACTION*********
240 LOCATE T,24
250 PRINT CHR$(198);CHR$(F)
260 Y=Y+1
270 IF RND>0.8 AND Y<16 AND X>1 AND X<
39  THEN LOCATE X,Y-1:PRINT" ":X=X+INT
(RND*3)-1
```

```
280 LOCATE X,Y
290 PRINT CHR$(252)
300 LOCATE X,Y-1
310 PRINT" "
320 SOUND 2,Y*20,1
330 FOR D=1 TO (55-(SC/3)):NEXT D
340 IF Y>23 AND X<>T THEN GOTO 510
350 IF X=T AND Y=24 THEN GOSUB 440
360 A$=INKEY$
370 LOCATE T,24
380 PRINT"  "
390 IF A$="7" THEN T=T-1:F=251
400 IF A$="9" THEN T=T+1:F=250
410 IF T>39 THEN T=39
420 IF T<1 THEN T=1
430 GOTO 230
440 SC=SC+INT(RND*4)+7
450 SOUND 1,20,18,5
460 Y=2:X=X+INT(RND*40)-20
470 IF X<1 THEN 460
480 IF X>39 THEN 460
490 LOCATE T,24
500 RETURN
510 PRINT"  "
520 CLS
530 SOUND 1,600:SOUND 1,640
540 LOCATE 12,12
550 PRINT"You prevented ";SC
560 PRINT TAB(8);"Tonnes of bombs land
ing"
```

```
570 SPEED KEY 20,3
580 GOTO 530
590 REM*****INSTRUCTIONS*****
600 MODE 0
610 CLS
620 PRINT:PRINT
630 PEN 3
640 PRINT TAB(7);"PROTECTOR"
650 PEN 1
660 PRINT:PRINT:PRINT
670 PRINT"  ONLY YOU CAN SAVE"
680 PRINT"  YOUR BELOVED AREA"
690 PRINT"  FROM THE FEARSOME"
700 PRINT"  SHELLS DROPPED BY"
710 PRINT"  YOUR ENEMY."
720 PRINT:PRINT
730 PRINT" YOU HAVE A BASKET"
740 PRINT"  TO CATCH SHELLS"
750 PRINT" THAT IS MOVED BY"
760 PRINT"  USING THE CURSOR"
770 PRINT" KEYS,LEFT & RIGHT"
780 PEN 3
790 PRINT:PRINT:PRINT
800 PRINT"     GOOD LUCK"
810 PEN 1
820 FOR T=1 TO 15000:IF INKEY$<>"" THE
N T=15000
830 NEXT T
840 RETURN
```

IVORY TUNES

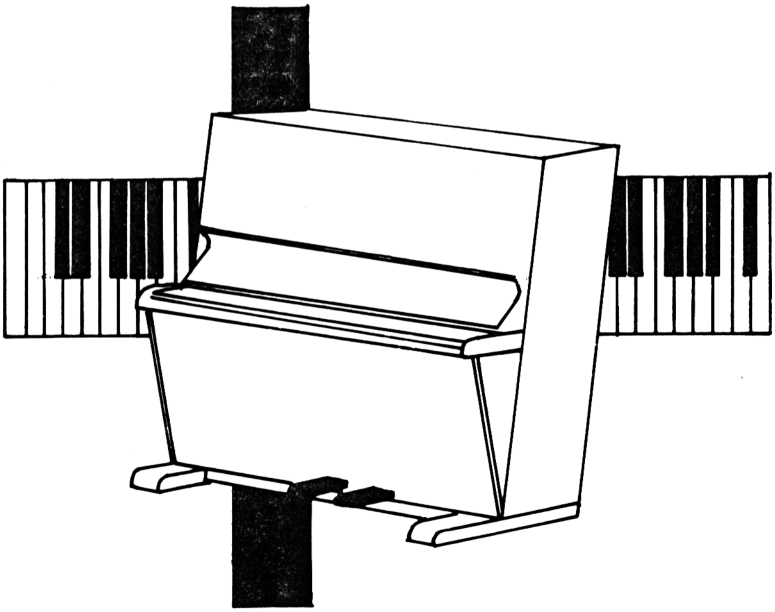
Oscar Peterson, Richard Clayderman and Elton John move over — here is your chance to show them all up with this fun program from Clive Gifford. When the program is RUN a remarkably life-like picture of a piano appears on the screen; you can then use the top row of the keyboard as a piano (it is, in fact, closer to an electronic organ). All the keys — including the delete key — can be used, while the ESC key, if played once, pauses the sounds, giving the effect of a rest.



```
10 REM****************************
20 REM*****IVORY TUNES***********
30 REM****************************
40 GOSUB 180
50 T=0
60 KEY DEF 79,1,17
70 GOSUB 250
80 A$=INKEY$
90 IF A$="" THEN GOTO 160
100 IF A$="0" THEN T=M(10):GOTO 160
110 IF A$="-" THEN T=M(11):GOTO 160
120 IF A$="^" THEN T=M(12):GOTO 160
130 IF A$=CHR$(16) THEN T=M(13):GOTO 1
60
140 IF A$=CHR$(17) THEN T=M(14):GOTO 1
60
150 T=M(VAL(A$))
160 SOUND 1,T,1,7
170 GOTO 80
180 REM***INITIALISATION****
190 DIM M(14)
200 FOR T=1 TO 14
210 READ M(T)
220 NEXT T
230 DATA 568,506,478,426,379,358,319,2
84,253,239,213,190,179,159
240 RETURN
250 REM*******SCREEN*******
260 CLG
270 INK 0,26
```

```
280 INK 1,0
290 CLS
300 PLOT 0,250
310 DRAW 640,250
320 LOCATE 15,14
330 PRINT "Steinway"
340 PLOT 0,150
350 DRAW 640,150
360 FOR T=1 TO 16
370 PLOT T*40,150
380 DRAW T*40,0
390 NEXT T
400 FOR T=1 TO 26
410 PLOT 66+T,150
420 DRAW 66+T,50
430 NEXT T
440 FOR T=1 TO 26
450 PLOT 106+T,150
460 DRAW 106+T,50
470 NEXT T
480 FOR T=1 TO 26
490 PLOT 146+T,150
500 DRAW 146+T,50
510 NEXT T
520 FOR T=1 TO 26
530 PLOT 306+T,150
540 DRAW 306+T,50
550 NEXT T
560 FOR T=1 TO 26
570 PLOT 266+T,150
```

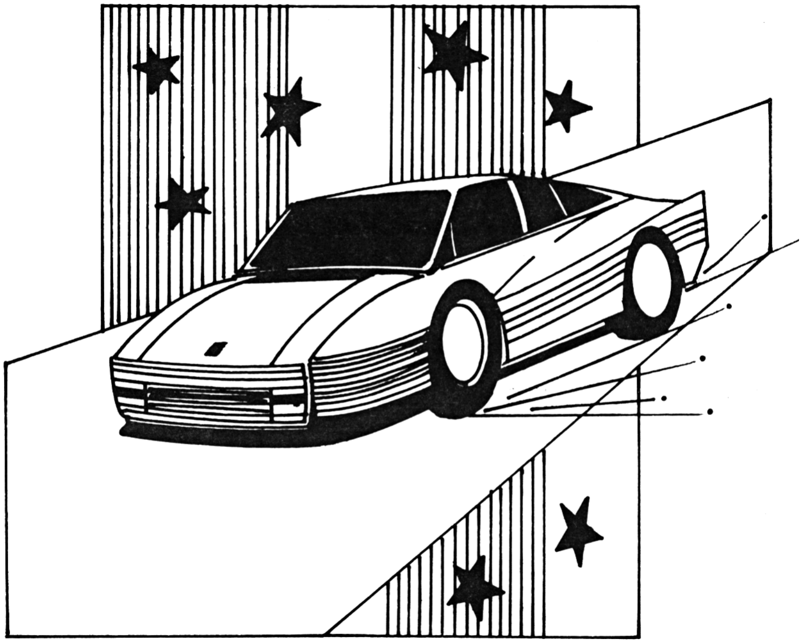
```
580 DRAW 266+T,50
590 NEXT T
600 FOR T=1 TO 26
610 PLOT 426+T,150
620 DRAW 426+T,50
630 NEXT T
640 FOR T=1 TO 26
650 PLOT 466+T,150
660 DRAW 466+T,50
670 NEXT T
680 FOR T=1 TO 26
690 PLOT 506+T,150
700 DRAW 506+T,50
710 NEXT T
720 RETURN
```



DEMON DRIVER

You are cruising along, the sun goes down and out comes the moon and the stars. Yet something seems wrong, the road appears to be deserted and your humble Citroen 2CV is suddenly transformed into a red Ferrari; the road, once a humble dual-carriageway, becomes a twisting, turning track and your timid driving is replaced with a new sense of daring and confidence. Gone are the days of poodling along, letting the snails overtake you; now you are a 'Demon Driver' fighting for your life . . .

In this fast-moving game you use the 'Z' and 'M' keys to steer the car and keep it on the track for as long as possible.

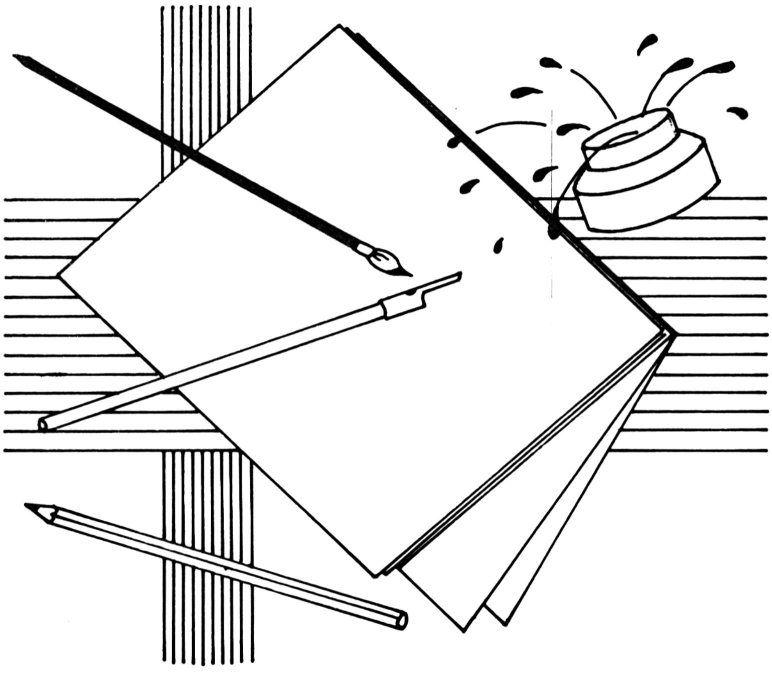


```
10 REM**************************
20 REM******DEMON DRIVER******
30 REM**************************
40 MODE 1
50 RANDOMIZE TIME
60 INK 0,0
70 CLS
80 M=0
90 INK 1,26
100 FOR T=1 TO 100
110 PLOT INT(RND*639)+1,INT(RND*100)+3
00,1
120 NEXT T
130 FOR T=1 TO 360 STEP 3
140 PLOT 600,380
150 DRAW 600+30*COS(T),380+30*SIN(T)
160 NEXT T
170 P=310
180 PEN 3
190 LOCATE 19,24
200 PRINT CHR$(244)
210 LOCATE 18,25
220 PRINT CHR$(151);CHR$(155);CHR$(157
)
230 REM*****MAIN LOOP*****
240 PLOT 0,300
250 DRAW 640,300,1
260 PLOT P,300,1
270 DRAW P-120,0,1
280 PLOT P+20,300,1
```

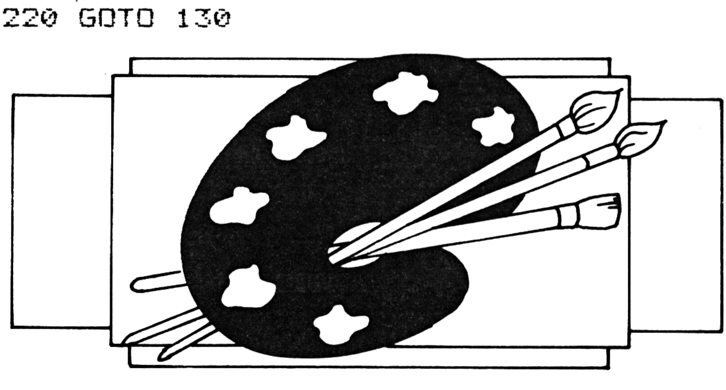
```
290 DRAW P+140,0,1
300 PLOT P,300,0
310 DRAW P-120,0,0
320 PLOT P+20,300,0
330 DRAW P+140,0,0
340 IF P>382 OR P<200 THEN GOTO 410
350 SOUND 4,500,2
360 M$=INKEY$
370 IF M$="M" THEN P=P-20
380 IF M$="Z" THEN P=P+20
390 P=P+INT(RND*50)-25:M=M+0.2
400 GOTO 240
410 REM****CRASH ROUTINE*****
420 INK 1,7,15
430 FOR T=500 TO 1000
440 SOUND 1,T,1
450 SOUND 2,T,1
460 SOUND 3,T,1
470 SOUND 4,T,1
480 NEXT T
490 LOCATE 4,9
500 M=INT((M*100))/100
510 PRINT " YOU TRAVELLED ";M;" MILES
"
520 LOCATE 1,25
530 PRINT "ANOTHER GO? (Y/N)
          "
540 INPUT A$
550 IF A$="Y" THEN RUN ELSE END
```

PICASSO

Let your artistic feelings flow out onto the Amstrad screen with this artist program. The program has been designed so that it is simple to use, with the four cursor keys controlling the direction of movement of your ink line. You can alter the colour (at the beginning it is the same as the background) by pressing the COPY key in the middle of the cursor key cluster. Pressing COPY adds one to the colour value (this lies between 1 and 26 — have a look in your Amstrad manual). Pressing the space bar brings the colour value back to zero — which is equivalent to the background — thus acting in the same way as a rub out key.



```
10 REM***********************
20 REM*******PICASSO*******
30 REM***********************
40 P=0
50 MODE 0
60 X=320:Y=200
70 CLS
80 KEY DEF 2,1,56
90 KEY DEF 0,1,55
100 KEY DEF 8,1,54
110 KEY DEF 1,1,57
120 KEY DEF 9,1,53
130 A$=INKEY$:IF A$="" THEN 130
140 IF A$="7" AND Y<398 THEN Y=Y+1
150 IF A$="8" AND Y>1 THEN Y=Y-1
160 IF A$="9" AND X<638 THEN X=X+1
170 IF A$="6" AND X>1 THEN X=X-1
180 IF A$="5" THEN P=P+1
190 IF A$=" " THEN P=0
200 IF P=16 THEN P=0:SOUND 1,180,10
210 PLOT X,Y,P
220 GOTO 130
```



PLAY IT AGAIN SAM

This superb version of the memory game 'Simon' was written by Clive Gifford. You have to remember the colour and sound sequence played to you by the computer, then type it back into the computer. The colour squares and their corresponding numbers are:

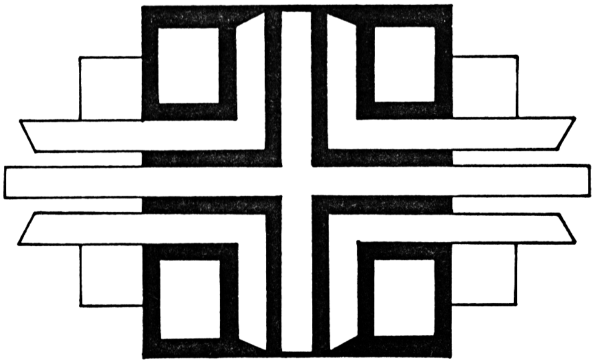
BLUE	1
RED	2
GREEN	3
YELLOW	4

Unlike inferior versions, 'Play It Again Sam' has no limit to the number of moves you can remember and play. So those of you with fantastic memories can continue for ever! Anyone who can gain a score of over 25 moves (without using paper, I hasten to add) is a better man than me.

```
10 REM************************
20 REM***PLAY IT AGAIN SAM***
30 REM************************
40 GOSUB 510
50 GOSUB 370
60 REM****MAIN GAME ACTION*****
70 FOR G=1 TO 40
80 FOR N=1 TO G
90 PAPER #T(N),0
100 CLS #T(N)
110 SOUND 1,T(N)*100,40
```

```
120 FOR T=1 TO 300:NEXT T
130 PAPER #T(N),C(T(N))
140 CLS #T(N)
150 NEXT N
160 FOR D=1 TO G
170 A$=INKEY$:IF A$="" THEN 170
180 IF VAL(A$)<>T(D) THEN 290
190 PAPER #T(D),0
200 CLS #T(D)
210 SOUND 1,T(D)*100,50
220 FOR T=1 TO 300:NEXT
230 PAPER #T(D),C(T(D))
240 CLS #T(D)
250 NEXT D
260 FOR T=1 TO 1000:NEXT T
270 NEXT G
280 GOTO 280
290 REM***WRONG GUESS****
300 BORDER 0
310 MODE 1
320 INK 0,0
330 LOCATE 12,12
340 PRINT "YOU FAILED BUDDY"
350 PRINT TAB(6);"YOU GOT ";N-1;"NUMBE
RS CORRECT"
360 GOTO 360
370 REM****SCREEN DRAW****
380 MODE 0
390 CLS
400 PRINT " PLAY IT AGAIN SAM"
```

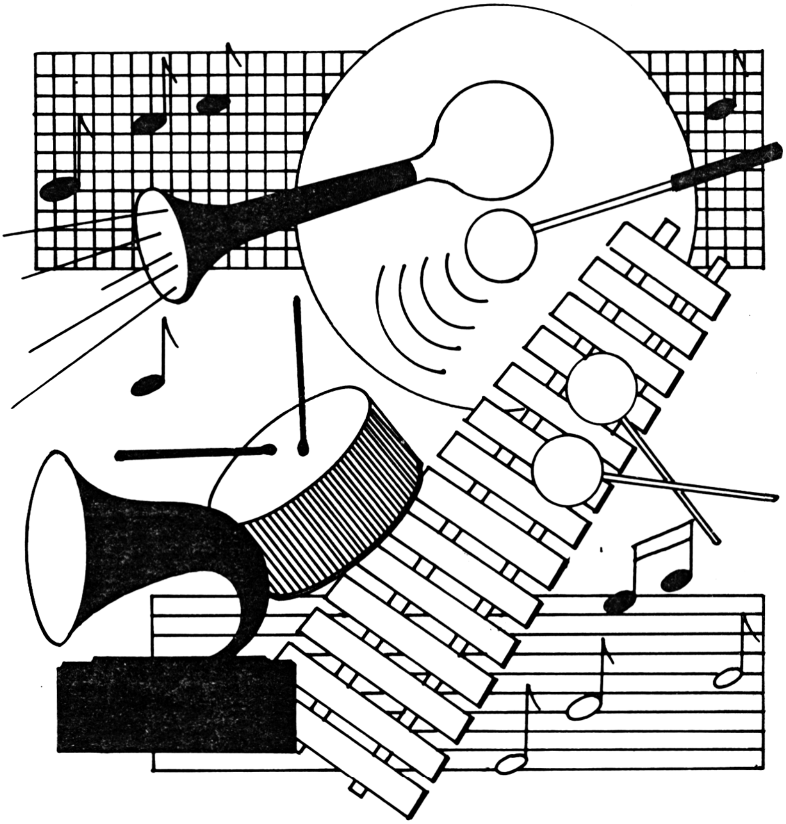
```
410 WINDOW #1,1,8,5,14
420 WINDOW #2,12,19,5,14
430 WINDOW #3,1,8,16,25
440 WINDOW #4,12,19,16,25
450 PAPER #1,10
460 PAPER #2,3
470 PAPER #3,9
480 PAPER #4,1
490 CLS#1:CLS#2:CLS#3:CLS#4
500 RETURN
510 REM****INITIALISATION*****
520 DIM T(40)
530 FOR X=1 TO 40
540 T(X)=INT(RND*4)+1
550 NEXT X
560 DIM C(4)
570 C(1)=10
580 C(2)=3
590 C(3)=9
600 C(4)=1
610 RETURN
```



SONIC FORCE

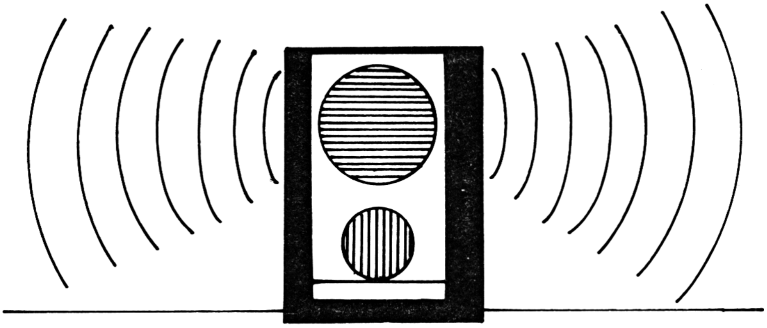
Pit your wits against the Amstrad in this classic game of sound recognition, written by Miss P. Tration of Birmingham. You must guess the value of the sound played to you by the computer. Raise your guess for the higher-pitched notes and lower it for the lower-pitched ones. Penny has incorporated a useful aid for the player in the form of a number of stars: fewer stars will be printed if your guess is close.

Good luck and happy guessing!



```
10 REM***************************
20 REM******SONIC FORCE*********
30 REM***************************
40 CLS
50 MODE 0
60 PRINT:PRINT:PRINT
70 PRINT"YOU MUST GUESS THE  PITCH OF
THE SOUND  ONCE THE NOTE IS     PLAYED"
80 PRINT:PRINT"ENTER YOUR GUESS AND TH
E COMPUTER WILL  GIVE YOU A CLUE"
90 PRINT:PRINT:PEN 6
100 PRINT TAB(15);"GOOD LUCK"
110 R=INT(RND*120)
120 MV=0
130 FOR T=1 TO 4000:NEXT T
140 F=INT(RND*14)+1
150 PAPER F+1:PEN 1
160 CLS
170 SOUND 1,(R*3)+100,150,5
180 INPUT"*GUESS (1-120)*";N
190 IF N<1 OR N>120 THEN PRINT:GOTO 18
0
200 IF N=R THEN 270
210 IF N-20>R THEN PRINT"FAR, FAR TOO
BIG"
220 IF N-20<=R AND N>R THEN PRINT"TOO
LARGE BUT CLOSE"
230 IF N+20<R THEN PRINT"MUCH TOO SMAL
L"
240 IF N+20>=R AND N<R THEN PRINT"TOO
```

```
SMALL, BUT NEAR"
250 FOR T=1 TO 5000:NEXT T
260 MV=MV+1:GOTO 140 RUN
270 REM*****YOU GUESSED IT!******
280 MODE 1:PAPER 4:PEN 1
290 CLG
300 PLOT 70,200
310 DRAW 320,50
320 DRAW 570,200
330 DRAW 320,350
340 DRAW 70,200
350 LOCATE 10,12
360 PRINT"CORRECT,NUMBER WAS";R
370 LOCATE 10,13
380 PRINT"IT TOOK YOU";MV;" MOVES"
390 FOR T=300 TO 100 STEP-1
400 SOUND 2,T,5
410 NEXT T
420 SOUND 2,95,180
430 LOCATE 4,25
440 PRINT"ANOTHER GO:- PRESS 'P'"
450 IF INKEY$="P" THEN RUN ELSE 450
```



BLASTER MIND

A more difficult version of the classic game, all instructions are included within the game . . . what more can I say, except good luck!

```
10 REM*************************
20 REM*******BLASTER MIND*******
30 REM*************************
40 GOSUB 500
50 PAPER 0
60 PEN 1
70 MV=0
80 MODE 1
90 RANDOMIZE TIME
100 CLS
110 DIM R(4)
120 FOR H=1 TO 4
130 R(H)=INT(RND*10)
140 NEXT
150 DIM A(4)
160 DIM A$(4)
170 REM*****MAIN LOOP*****
180 BORDER INT(RND*27)
190 MV=MV+1
200 FOR T=1 TO 4
210 A$(T)="_"
```

```
220 NEXT T
230 PRINT "ENTER EACH NUMBER, PRESSING
 'ENTER'      AFTER EACH"
240 PRINT:PRINT:PRINT
250 FOR T=1 TO 4
260 INPUT A(T)
270 SOUND 1,T*60
280 NEXT T
290 IF R(1)=A(1) THEN A$(1)="*"
300 IF R(2)=A(2) THEN A$(2)="*"
310 IF A(3)=R(3) THEN A$(3)="*"
320 IF A(4)=R(4) THEN A$(4)="*"
330 PRINT:PRINT:PRINT
340 FOR T=1 TO 4
350 PRINT A$(T);
360 NEXT T
370 IF A$(1)="*" AND A$(2)="*" AND A$(
3)="*" AND A$(4)="*" THEN GOTO 400
380 PRINT:PRINT
390 GOTO 170
400 REM****WIN*******
410 PAPER 3
420 PEN 2
430 SOUND 1,100,200
440 CLS
450 LOCATE 10,10
460 PRINT "WELL DONE"
470 PRINT:PRINT TAB(8);"YOU DID IT IN
";MV;" MOVES"
480 PRINT:PRINT:PRINT
```

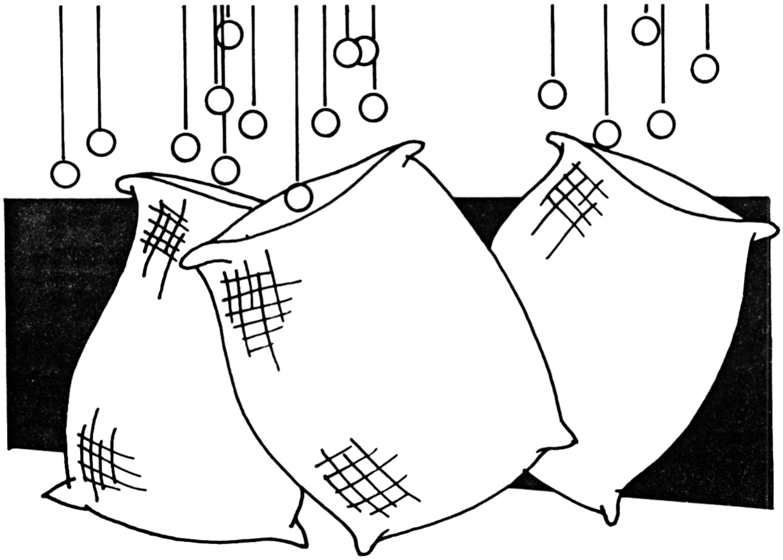
```
490 END
500 REM****INSTRUCTIONS****
510 CLS
520 PRINT:PRINT "   In this modified v
ersion of this      classic game, you m
ust guess the digit  combination (4 di
gits)."
530 PRINT:PRINT "   Your only help is
when a correct      number is in the co
rrect position, then a '*' will appear
 else a '_' will b
e    displayed"
540 PRINT:PRINT "   '0' is included an
d there can be morethan one digit bein
g the same number,    for example, 7887
 is valid"
550 PRINT:PRINT:PRINT"************ GOO
D LUCK **************"
552 PRINT:PRINT:PRINT "Press any key t
o continue"
553 WHILE INKEY$=""
554 WEND
560 RETURN
```

GOLD BULLION RAID

You and your robber friends have just held up a train carrying the British Museum's collection of gold pieces. Speed is essential if you are to make a quick getaway and your associates are throwing down the pieces as quickly as possible.

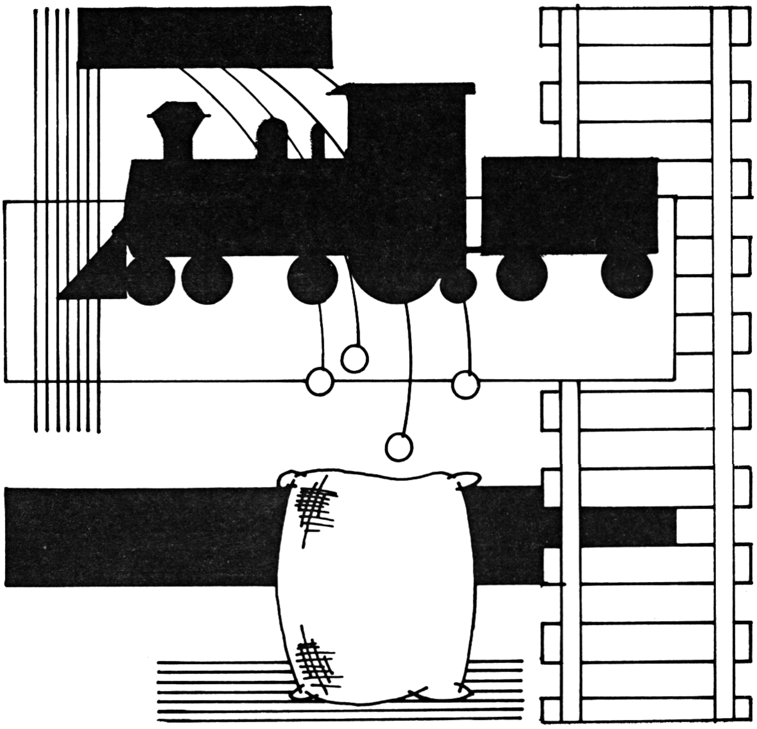
You control a mechanical sack system whereby all the coins are categorised from 1 to 8. In order for the coin to fall into the sack and so increase the total of your haul, you must press the appropriate number for each coin as the pieces rain down.

You will need extremely fast reflexes in this fast-moving game written by William E. Warmer of Sussex.



```
10 REM*******************************
20 REM*****GOLD BULLION RAID**********
30 REM*******************************
40 N=0
50 RANDOMIZE TIME
60 CLS
70 LOCATE 1,24
80 PRINT "\_/  \_/  \_/  \_/  \_/  \_/
  \_/  \_/"
90 PRINT
100 PRINT " 1    2    3    4    5    6
    7    8
110 A=INT(RND*8)+1
120 FOR T=3 TO 23
130 LOCATE (A*5)-3,T
140 LOCATE (A*5)-3,T-1
150 PRINT "  ";
160 LOCATE (A*5)-3,T
170 PRINT CHR$(231)
180 A$=INKEY$
190 IF VAL(A$)=A THEN GOTO 250
200 SOUND 1,T*20,1
210 NEXT T
220 N=N+1:IF N>17 THEN 300
230 SOUND 1,800,10,7
240 GOTO 60
250 REM****CAUGHT ONE****
260 SOUND 1,40,50
270 N=N+1
280 SC=SC+(A*10)
```

```
290 GOTO 60
300 REM***END OF GAME****
310 LOCATE 1,25
320 FOR T=1 TO 26
330 PRINT
340 SOUND 1,400-(6*T)
350 NEXT T
360 LOCATE 10,12
370 PRINT "SCORE EQUALS ";SC
380 LOCATE 8,20
390 PRINT "ANOTHER TRY BUDDY?"
400 IF INKEY$="Y" THEN RUN
410 GOTO 400
```



MADAME ROSA

From the mysterious East, Madame Rosa has come to advise you on your present status in life. She is particularly concerned about your health and the amount of stress you are under at present. Answer her questions with a 'Y' or a 'N' and see what her conclusion is at the end of the interview.

This program was created by Richard Donn of Crawley, Surrey.

```
10 REM********************************
20 REM*******MADAME ROSA***********
30 REM********************************
40 GOTO 130
50 INK 3,26,3
60 PAPER 0
70 SPEED INK 100,100
80 CLG
90 PLOT 320,200
100 DRAW RND*640,RND*400
110 FOR T=1 TO 200:NEXT T
120 GOTO 90
130 INK 0,7
140 INK 1,26
150 CLS
160 BORDER 1,9
170 SPEED INK 60,60
180 S$="9973716563635350474545444040039
```

```
3939383736353331302929292928262625242З
212020201919181716151513121 1"
190 I=1
200 PRINT:PRINT
210 PRINT"LET MADAME ROSA, THE FAMOUS
EASTERN     MYSTIC (WELL, EAST-END, AC
TUALLY) ADVISEYOU ON YOUR STATE OF LIF
E AT THE MOMENT"
220 L$="IS YOUR HUSBAND OR WIFE DEAD"
230 GOSUB 1320
240 L$="ARE YOU DIVORCED"
250 GOSUB 1320
260 L$="ARE YOU GIVING UP DOPE/DRUGS"
270 GOSUB 1320
280 L$="IS YOUR MARRIAGE ON THE ROCKS"
290 GOSUB 1320
300 L$="HAVE YOU RECENTLY BEEN IMPRISO
NED"
310 GOSUB 1320
320 L$="HAS A RELATIVE DIED RECENTLY"
330 GOSUB 1320
340 L$="HAVE YOU BEEN INJURED LATELY"
350 GOSUB 1320
360 L$="ARE YOU MARRIED"
370 GOSUB 1320
380 L$="ARE YOU UNEMPLOYED"
390 GOSUB 1320
400 L$="HAVE YOU ANY CHILDREN"
410 GOSUB 1320
420 L$="ARE YOU RETIRED"
```

```
430 GOSUB 1320
440 L$="IS YOUR FAMILY IN POOR HEALTH"
450 GOSUB 1320
460 L$="ARE YOU PREGNANT"
470 GOSUB 1320
480 L$="ARE YOU QUITTING SMOKING"
490 GOSUB 1320
500 L$="HAVE YOU ANY SEX PROBLEMS"
510 GOSUB 1320
520 L$="HAVE THERE BEEN ANY BIRTHS IN
THE          FAMILY"
530 GOSUB 1320
540 L$="ANY BUSINESS RE-ADJUSTMENT AT
WORK"
550 GOSUB 1320
560 L$="ARE YOU BROKE"
570 GOSUB 1320
580 L$="HAVE ANY CLOSE FRIENDS DIED RE
CENTLY"
590 GOSUB 1320
600 L$="HAS YOUR JOB CHANGED"
610 GOSUB 1320
620 L$="DO YOU ARGUE MUCH WITH YOUR SP
OUSE"
630 GOSUB 1320
640 L$="ARE YOU SUFFERING FROM PRE-MEN
STRUAL      TENSION"
650 GOSUB 1320
660 L$="HAVE YOU HAD A FORECLOSURE ON
YOUR         MORTGAGE OR PERSONAL LOAN"
```

```
670 GOSUB 1320
680 L$="IS YOUR MORTGAGE OVER #20,000"
690 GOSUB 1320
700 L$="HAS THERE BEEN A CHANGE IN YOU
R WORK     RESPONSIBILITY"
710 GOSUB 1320
720 L$="ARE YOU SUFFERING FROM JET LAG
"
730 GOSUB 1320
740 L$="ARE YOUR CHILDREN LEAVING HOME
"
750 GOSUB 1320
760 L$="ARE YOUR IN-LAWS CAUSING PROBL
EMS"
770 GOSUB 1320
780 L$="HAVE YOU REACHED ANY OUTSTANDI
NG          PERSONAL ACHIEVEMENTS"
790 GOSUB 1320
800 L$="IS YOUR SPOUSE STARTING OR STO
PPING     WORK"
810 GOSUB 1320
820 L$="ARE YOUR CHILDREN STARTING OR
STOPPING  SCHOOL"
830 GOSUB 1320
840 L$="HAVE YOU BEEN BULLIED RECENTLY
"
850 GOSUB 1320
860 L$="HAVE YOU REVISED YOUR PERSONAL
 HABITS"
870 GOSUB 1320
```

```
880 L$="ARE YOU IN TROUBLE WITH YOUR B
OSS"
890 GOSUB 1320
900 L$="ARE YOU GIVING UP DRUGS OR HEA
VY          DRINKING"
910 GOSUB 1320
920 L$="HAS THERE BEEN A CHANGE IN YOU
R WORKING HOURS"
930 GOSUB 1320
940 L$="HAVE YOU CHANGED YOUR RESIDENC
E"
950 GOSUB 1320
960 L$="HAVE YOU CHANGED YOUR SCHOOL"
970 GOSUB 1320
980 L$="HAVE YOU CHANGED YOUR HOBBIES"
990 GOSUB 1320
1000 L$="ARE YOU CHANGING YOUR CHURCH
          ACTIVITIES"
1010 GOSUB 1320
1020 L$="ARE YOU CHANGING YOUR SOCIAL
          ACTIVITIES"
1030 GOSUB 1320
1040 L$="IS YOUR MORTGAGE OR LOAN UNDE
R #20,000"
1050 GOSUB 1320
1060 L$="HAVE YOUR SLEEPING HABITS CHA
NGED"
1070 GOSUB 1320
1080 L$="ARE THERE FEWER FAMILY GET-TO
GETHERS"
```

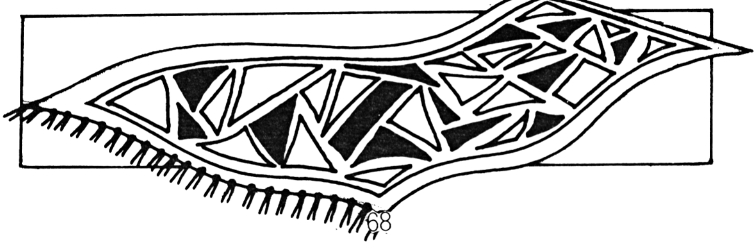
```
1090 GOSUB 1320
1100 L$="HAVE YOUR EATING HABITS CHANG
ED"
1110 GOSUB 1320
1120 L$="ARE YOU GOING ON HOLIDAY SOON
"
1130 GOSUB 1320
1140 L$="IS IT CHRISTMAS"
1150 GOSUB 1320
1160 L$="ARE YOU VIOLATING THE LAW, E.
G. BY      PIRATING SOFTWARE"
1170 GOSUB 1320
1180 CLS
1190 PRINT:PRINT
1200 PRINT TAB(4);"YOUR TOTAL IS ";L
1210 PRINT:PRINT:PRINT
1220 PRINT TAB(14);"THE VERDICT"
1230 IF L<150 THEN PRINT"YOU LEAD A ST
ABLE WAY OF LIFE AND ARE   LESS LIKELY
 TO INJURE YOURSELF..."
1240 IF L<150 THEN FOR T=1 TO 6000:NEX
T T:PRINT"UNLESS YOU DIE OF BOREDOM!"
1250 IF L>149 AND L<200 THEN PRINT "TH
ERE IS A FAIR CHANCE OF YOU SUFFERING
HEALTH OR SAFETY PROBLEMS"
1260 IF L>199 AND L<300 THEN PRINT "CA
LM DOWN. AT THIS RATE YOU WILL NOT
LIVE ANOTHER DAY"
1270 IF L>299 THEN PEN 0:PRINT"STOP AL
L WORK AND WAIT UNTIL YOUR TOTAL DECRE
```

```
ASES SIGNIFICANTLY."
1280 FOR T=1 TO 1000:NEXT T
1290 WHILE INKEY$=""
1300 WEND
1310 GOTO 50
1320 REM***ANSWER CONVERSION***
1330 PRINT:PRINT
1340 PRINT L$;"?"
1350 PRINT:PRINT
1360 INPUT "Y or N";B$
1370 B$=UPPER$(B$)
1380 IF B$<>"Y" AND B$<>"N" THEN 1350
1390 IF B$="Y" THEN L=L+VAL(MID$(S$,I*
2-1,2))
1400 I=I+1
1410 SOUND 1,INT(RND*600)+50,60:SOUND
1, INT(RND*600)+50,50
1420 RETURN
```

PERSIAN CARPET

An effective, high-resolution graphic display which after a bit of 're-jigging' does look like the item in the title.

```
10 REM****************************
20 REM******PERSIAN CARPET******
30 REM****************************
40 BORDER 1
50 INK 0,6
60 INK 1,0
65 MODE 2
70 CLS
80 RANDOMIZE TIME
90 C=640
100 D=400
110 A=C*RND
120 B=D*RND
130 PLOT A,B,1
140 PLOT A,D-B,1
150 PLOT C-A,B,1
160 PLOT C-A,D-B,1
170 GOTO 110
```

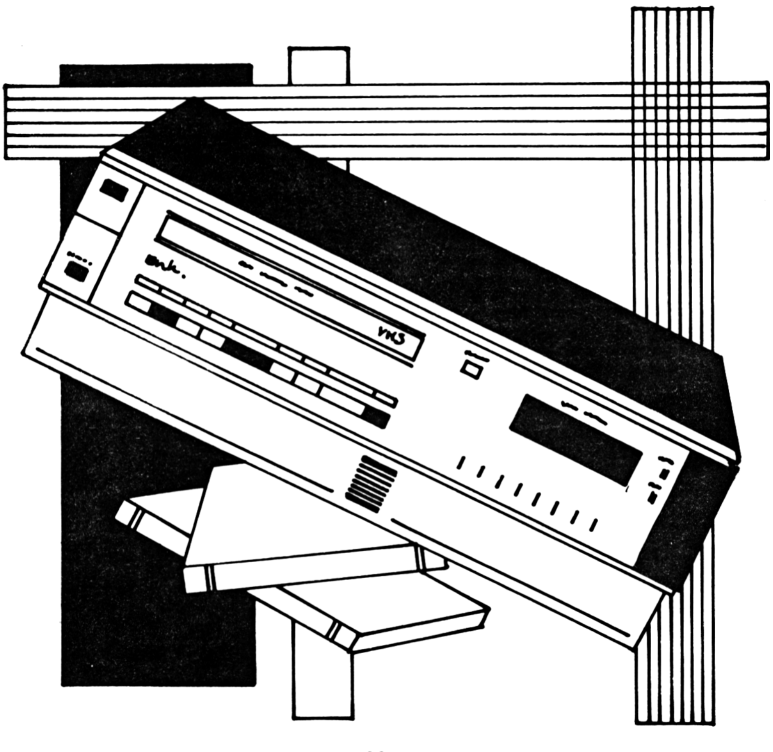


VIDEO SALES

If you think you've got the business acumen to make it in the video business, why not put your money where your mouth is in an attempt to buy and sell video recorders to the public. The game was written by Peter Shaw of Chiswick and converted to the Amstrad.

The computer will allow up to four players to partake in this game. Events, from the release of an infamous 'video nasty' to the increase in popularity of Cable Television, all affect the market and your likely success.

At the end of six weeks of intensive action, the computer calculates who the winner is and rewards that person accordingly.

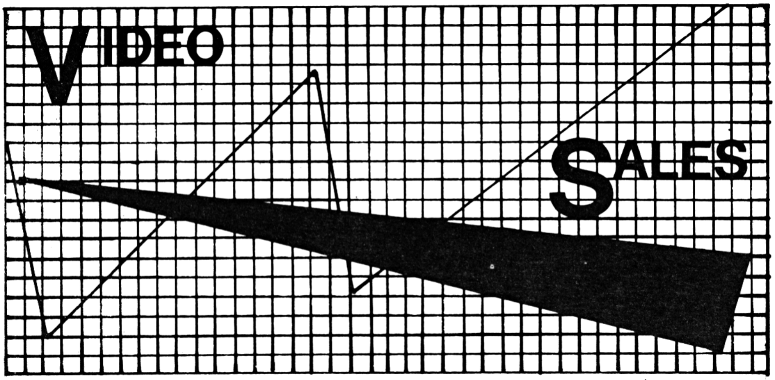


```
10 REM***************************
20 REM********VIDEO SALES********
30 REM***************************
40 REM********SHAW / WAY*********
50 REM***************************
60 GOSUB 770
70 CLS:MODE 1:PEN 3
80 WK=1
90 LOCATE 12,10:PRINT"How many players
?"
100 INPUT P:IF P>4 THEN GOTO 100
110 PRINT"Press ENTER to start."
120 INPUT DUM$
130 DIM S(4),C(4)
140 FOR A=1 TO P:C(A)=1000:NEXT A
150 L=0
160 L=L+1:IF L=P+1 THEN WK=WK+1:L=1
170 IF C(L)<200 THEN PRINT"Salesman ";
L,,"You do not have enough money to bu
y any stock":GOTO 160:GOTO 430
180 CLS
190 PRINT"Salesman ";L;" Sales ";S(L)
200 PRINT"Cash in hand ";C(L)
210 GOSUB 440
220 PRINT:PRINT:PRINT"Local and Nation
al News...."
230 PRINT H$
240 PRINT:PRINT:INPUT"How many recorde
rs will you stock this  week. Your cos
t (#200 ea.)";RE
```

```
250 PRINT"How much will you sell them
for?":INPUT CH
260 PRINT:PRINT:PRINT"Press ENTER to c
ontinue"
270 INPUT DUM$
280 IF RE*200>C(L) THEN GOTO 240
290 CLS
300 PRINT"Salesman ";L
310 SA=(H*(10)/(CH/10))
320 IF SA>RE THEN SA=RE
330 PRINT"Cash in hand ";C(L)
340 S(L)=INT(SA)
350 PRINT"Sales this week ";S(L)
360 PR=(S(L)*CH)-(RE*200)
370 PRINT"Profit ";PR
380 C(L)=C(L)+PR
390 PRINT"New Balance ";C(L)
400 INPUT"Press ENTER to continue ";DU
M$
410 IF WK=6 THEN GOTO 600
420 GOTO 160
430 END
440 h=INT(RND*20)+1
450 IF H>10 THEN GOTO 530
460 RESTORE 440
470 FOR A=1 TO INT(RND*7)+1
480 READ H$
490 NEXT A
500 RETURN
510 DATA "The Olympics are on","It's g
```

```
etting near to Christmas","Video recor
ders drop in price","The worst Video N
asty yet, 'I Spit
On   Your QL' is made available despit
e       strong protests particularly fr
om  the  Cambridge area!"
520 DATA "The latest Spielberg film is
 available  on video officially!","A n
ew tape hire shop has opened in town",
"National price of
fer on Video tapes by  major tape comp
any"
530 RESTORE 530
540 FOR A=1 TO INT(RND*7)+1
550 READ H$
560 NEXT A
570 RETURN
580 DATA "Inflation hits video industr
y","Tougher Laws passed on Video Pirac
y","Video Nasties banned by local Coun
cil"
590 DATA "Cinema increases in populari
ty","Cable television catches on very
fast","Summer television very poor","L
ocal 'Pirate' is c
aught and his         complete stock is
 burned"
600 REM********THE WINNER!********
610 CLS
620 MODE 0
```

```
630 FOR X=1 TO P
640 PRINT:PRINT"Salesman ";X
650 PRINT"Cash in hand";C(X)
660 PRINT:NEXT X
670 IF C(1)>C(2) AND C(1)>C(3) AND C(1
)>C(4) THEN W=1:GOTO 710
680 IF C(2)>C(1) AND C(2)>C(3) AND C(2
)>C(4) THEN W=2:GOTO 710
690 IF C(3)>C(1) AND C(3)>C(2) AND C(3
)>C(4) THEN W=3:GOTO 710
700 W=4
710 PRINT:PRINT:PRINT"Therefore..."
720 PRINT
730 PEN 15
740 PRINT"Salesman ";W;" Wins!"
750 SOUND 1,100,200
760 GOTO 760
770 INK 1,24
780 BORDER 9
790 RETURN
```



SHOP WINDOW

This program allows you to create slogans and sentences, and the resultant effect is one that could be used in a shop window, hence the title. They are scrolled across the screen by your Amstrad, and you have a choice of screen sizes, positions and colours.

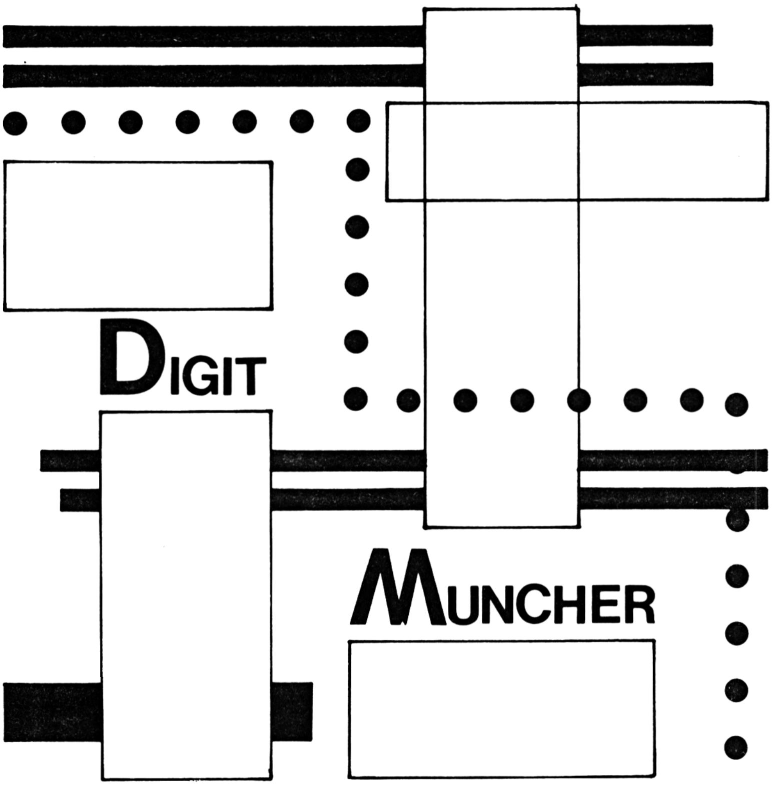
```
10 REM********************************
20 REM****S H O P    W I N D O W****
30 REM********************************
40 CLS
50 LOCATE 8,2
60 PRINT"SHOP WINDOW"
70 PRINT:PRINT
80 PRINT:PRINT
90 INPUT"SIZE: SMALL (2): MEDIUM (1):
BIG (0)";N
100 IF N>2 OR N<0 THEN PRINT:GOTO 90
110 F=N
120 IF N=0 THEN C=20
130 IF N=1 THEN C=40
140 IF N=2 THEN C=80
150 PRINT:PRINT:PRINT
```

AMSTRAD

```
160 INPUT"MESSAGE";A$
170 IF LEN(A$)>C-4 OR LEN(A$)<2 THEN 1
60
180 PRINT:PRINT:INPUT"WHAT ROW ON THE
SCREEN (1-25)";N
190 IF N>25 OR N<1 THEN 180
200 PRINT:PRINT
210 INPUT"SPEED OF MOVEMENT (1-300)";D
LAY
220 IF DLAY>300 OR DLAY<1 THEN PRINT:G
OTO 210
230 INPUT"DIFFERENT BORDER (Y/N)";G$
240 IF G$="Y" THEN BORDER INT(RND*26)
250 REM*****SCROLLING TECHNIQUE******
260 MODE F
270 CLS
280 A$=" "+A$+" "
290 L=LEN(A$)
300 WHILE INKEY$=""
310 FOR T=1 TO L
320 IF INT((C-L)/2)<1 THEN LOCATE 1,N:
GOTO 340
330 LOCATE INT((C-L)/2),N
340 PRINT MID$(A$,T,L-T);
350 LOCATE INT((((C-L)/2)+L)-T),N
360 PRINT LEFT$(A$,T)
370 CALL &BD19
380 FOR G=1 TO DLAY:NEXT G
390 NEXT T
400 WEND
```

DIGIT MUNCHER

This is a modified and improved version of an earlier program, 'Number Cruncher'. The new program is controlled by a joystick and there is now a high-score feature as well. The joystick is programmed for the four compass directions used by the game. When using the joystick, make sure that you push the stick directly into one of the four positions, as the computer does not allow for diagonal movement.



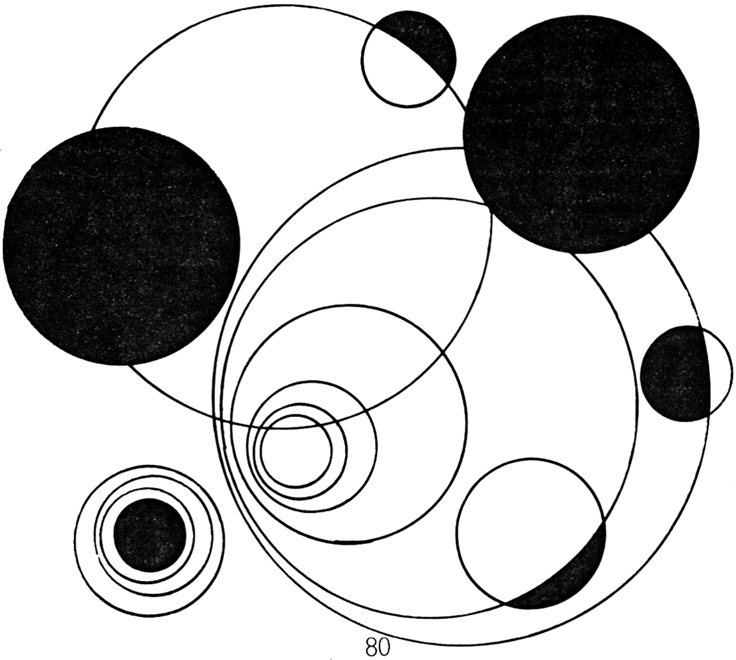
```
10 REM************************
20 REM*****DIGIT MUNCHER*****
30 REM************************
40 DIM S(40,25):HS=0
50 GOSUB 600
60 GOSUB 480
70 REM****MAIN LOOP****
80 J=JOY(0)
90 IF J=1 AND X>3 THEN X=X-1:C$=CHR$(1
98):P=1
100 IF J=2  AND X<24 THEN X=X+1:C$=CHR
$(196):P=2
110 IF J=4 AND Y>2 THEN Y=Y-1:C$=CHR$(
197):P=3
120 IF J=8 AND Y<38 THEN Y=Y+1:C$=CHR$
(199):P=4
130 IF J=0 AND P=1 AND X>3 THEN X=X-1
140 IF J=0 AND P=2 AND X<24 THEN X=X+1
150 IF J=0 AND P=3 AND Y>2 THEN Y=Y-1
160 IF J=0 AND P=4 AND Y<38 THEN Y=Y+1
170 LOCATE YY,XX:PRINT" "
180 IF S(Y,X)>0 AND S(Y,X)<10 THEN SC=
SC+S(Y,X):SOUND 1,150,4,4:S(Y,X)=0
190 IF S(Y,X)>9 THEN GOTO 260:REM***LO
SE***
200 LOCATE Y,X:PRINT C$
210 LOCATE 7,1:PRINT SC
220 XX=X:YY=Y
230 IF RND>0.96 THEN A=INT(RND*36)+2:B
=INT(RND*21)+3:LOCATE A-1,B:C=INT(RND*
9)+1:S(A,B)=C:PEN 3:PRINT C:PEN 1:SOUN
D 1,INT(RND*50)+80
,4,5
240 IF RND>0.84 THEN F=INT(RND*36)+2:G
=INT(RND*21)+3:LOCATE F,G:PRINT CHR$(2
24):S(F,G)=10
250 GOTO 70
260 REM*****LOSE*****
270 SOUND 2,600,16,7,1,12,2
280 SOUND 4,600,80,7,9,13,6
290 SOUND 1,600,30,7,4,1,15
300 CLS
310 LOCATE 17,8
320 PRINT"OUCH!!!"
330 PRINT:PRINT"YOUR SCORE WAS ";SC
340 IF HS>SC THEN PRINT"THAT IS NOT UP
 TO THE HIGH SCORE ";HS
350 IF HS<SC THEN HS=SC:PRINT"WELL DON
E, YOU NOW HAVE THE HIGH SCORE"
360 PRINT:PRINT:PRINT
370 PEN 3
380 PRINT"        ^^^^^^^^^^^^^^^^^^^^"
390 PRINT"        ^HIGH SCORE       ^"
400 PRINT"        ^^^^^^^^^^^^^^^^^^^^"
410 LOCATE 21,16:PRINT HS
420 PEN 1
430 PRINT:PRINT:PRINT"ANOTHER GAME? (Y
/N)"
440 D$=INKEY$:IF D$="" THEN 440
450 IF D$="Y" THEN GOTO 50
```

```
460 IF D$="N" THEN PRINT"OK>":END
470 IF D$<>"" THEN 440
480 REM***SCREEN***
490 MODE 1
500 CLS
510 PRINT"SCORE:";SC;
520 PEN 2
530 PRINT TAB(15);"DIGIT MUNCHER";
540 PEN 1
550 PRINT TAB(35);CHR$(164);"1984"
560 PRINT"###########################
###########"
570 FOR T=1 TO 22:PRINT"#";TAB(39);"#"
:NEXT T
580 PRINT"###########################
###########"
590 RETURN
600 REM****INITIALISATION****
610 FOR H=1 TO 40
620 FOR V=1 TO 25
630 S(H,V)=0
640 NEXT:NEXT
650 X=20:Y=13:SC=0
660 XX=X:YY=Y
670 F=0:G=0:A=0:B=0
680 RETURN
```

WHIRLS

The title sums up this program: whirls are plotted on the Amstrad's screen, with a dash of sound for added effect.

```
10 REM*******************
20 REM******WHIRLS******
30 REM*******************
40 MODE 1
50 CLG
60 FOR A=1 TO 35
70 FOR J=1 TO 7 STEP 0.1
80 PLOT A*J*COS(J)+270,A*J*SIN(J)+250
90 SOUND 1,INT(RND*500),1,3
100 NEXT J
110 NEXT A
```



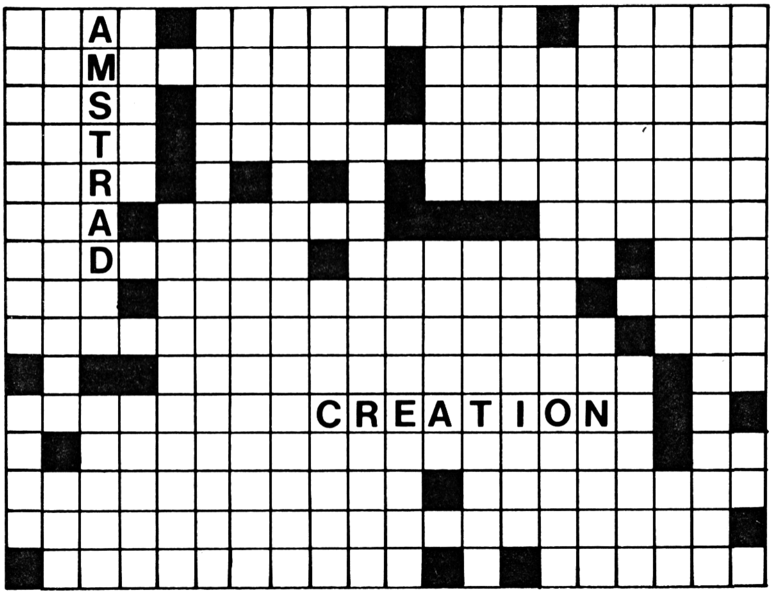
CREATION

Here is an offer you can't refuse — a chance to be God. This program is an Amstradised version of John Conway's classic game, 'Life'. You must construct a cell colony that will last for the maximum number of generations.

To create a colony, type in the co-ordinates of where you would like a cell to be placed. Continue doing this until you have your required cell pattern, then type 'O' in response to the computer's prompt for the next set of co-ordinates.

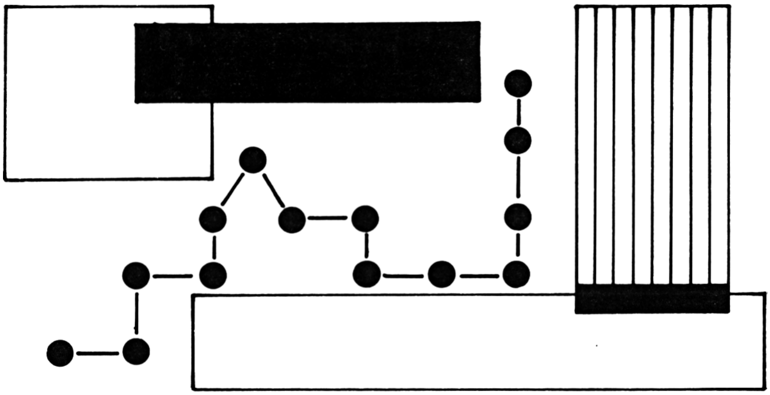
The computer will then calculate each cell generation using Conway's rules.

To construct a long-lasting colony you must not overcrowd the cells, but on the other hand you must not leave the cells too far away from each other. Several weeks ago I think I found the optimum combination, and my cell colony lasted for 35 generations. So beat that if you can!



```
10 REM******************
20 REM*****CREATION*****
30 REM******************
40 MODE 1
50 CLS
60 DIM M(1,10,10)
70 G=0
80 GOSUB 250
90 LOCATE 6,19
100 INPUT"ACROSS";X:IF X=0 THEN 160
110 IF X<1 OR X>10 THEN 100
120 LOCATE 6,21
130 INPUT"DOWN";Y
140 IF Y<1 OR Y>10 THEN 130
150 M(1,X,Y)=ABS(M(1,X,Y)-1):GOTO 90
160 FOR Y=1 TO 9
170 FOR X=1 TO 9
180 C=M(0,X-1,Y-1)+M(0,X-1,Y)+M(0,X-1,
Y+1)+M(0,X,Y-1)+M(0,X,Y+1)+M(0,X+1,Y-1
)+M(0,X+1,Y)+M(0,X+1,Y+1)
190 IF C=3 THEN M(1,X,Y)=1
200 IF C<2 OR C>3 THEN M(1,X,Y)=0
210 NEXT X:NEXT Y
220 G=G+1:GOSUB 250
230 GOTO 160
240 END
250 IF G<2 THEN CLS
260 LOCATE 16,2
270 PRINT"CREATION"
280 IF G>0 THEN PRINT:PRINT TAB(14);"G
```

```
ENERATION";G
290 PRINT:PRINT:PRINT "1234567890"
300 FOR Y=1 TO 9:X$=RIGHT$(STR$(Y),1)
310 FOR X=1 TO 9
320 IF M(1,X,Y)=1 THEN PRINT"O";:GOTO
340
330 PRINT".";
340 M(0,X,Y)=M(1,X,Y):NEXT:PRINT X$:NE
XT
350 PRINT "1234567890"
360 FOR T=140 TO 60 STEP 20
370 SOUND 1,T:NEXT T
380 IF G<1 THEN RETURN
390 LOCATE 4,23
400 PRINT "S TO STOP, C TO CONTINUE"
410 A$=INKEY$
420 IF A$="S" OR A$="s" THEN END
430 IF A$="C" OR A$="c" THEN SOUND 2,1
80:RETURN
440 GOTO 410
```



LAS VEGAS

An excellent simulation of a one-armed bandit concludes this collection of games for your Amstrad computer. Peter Nessbreth has added little touches, such as authentic spinning reels that slow down and a gambling display, to produce quite a realistic version of the old gambling weapon.

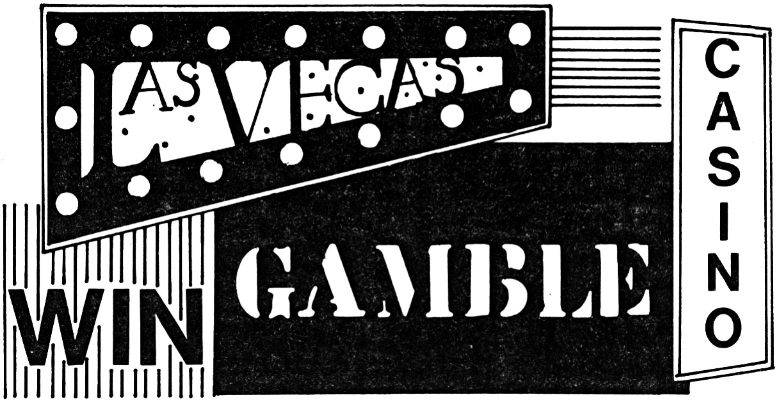
You must press the space bar to start the reels spinning. Allow a little time at the beginning of the program for the computer to set up the whole game.

All the instructions, including details of how to win, are included within the program.

```
10 REM*********************
20 REM*****LAS VEGAS*****
30 REM*********************
40 MODE 0
50 GOSUB 1080:REM**instructions***
60 AM=10
70 JK=100
80 CLS
90 LOCATE 5,2
100 PEN 1
110 PRINT"LAS VEGAS"
120 PEN 3
130 LOCATE 4,4
140 PRINT"###########"
150 FOR M=4 TO 14
160 LOCATE 4,M
170 PRINT"#"
180 LOCATE 14,M
```

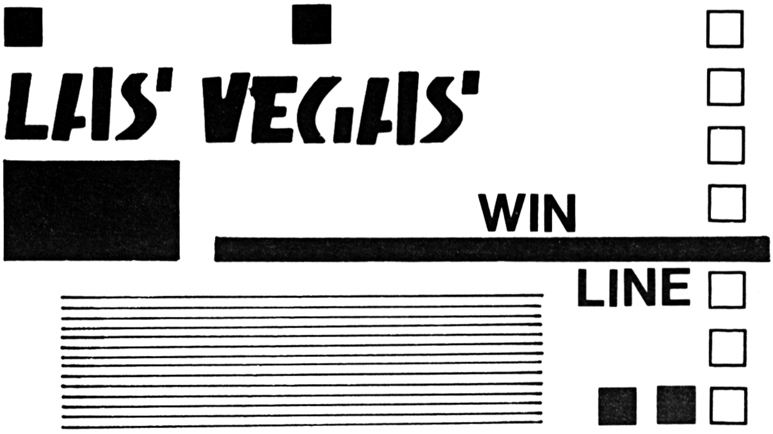
```
190 PRINT"#"
200 NEXT
210 LOCATE 4,14
220 PRINT"##########"
230 LOCATE 5,15
240 PRINT"\       /"
250 LOCATE 5,16
260 PRINT" \_____/"
270 WINDOW #1,6,6,6,11
280 WINDOW #2,9,9,6,11
290 WINDOW #3,12,12,6,11
300 WINDOW #4,18,20,4,6
310 WINDOW #5,18,20,9,11
320 WINDOW #6,18,20,14,16
330 PAPER #4,2
340 PAPER #5,13
350 PAPER #6,1
360 CLS #4
370 CLS #5
380 CLS #6
390 PEN #4,0:PEN #5,0:PEN #6,0
400 PRINT #4,"     x 4"
410 PRINT #5,"     x 3"
420 PRINT #6,"     x 2"
430 PAPER#1,4
440 PAPER#2,4
450 PAPER#3,4
460 PEN#1,8
470 PEN#2,8
480 PEN#3,8
```

```
490 GOSUB 760
500 X=INT(RND*170)
510 AM=AM-1:JK=JK+1
520 LOCATE 3,25
530 PRINT PT$;"                    "
540 LOCATE 1,20
550 PRINT" Money held:$";AM
560 IF AM<1 THEN GOTO 1230
570 PRINT
580 PRINT "  Jackpot:$";JK
590 FOR T=X TO X+INT(RND*20)+10
600 PRINT #1,MID$(A$,T,1);
610 PRINT#1," ";
620 PRINT #2,MID$(B$,T,1);
630 PRINT#2," ";
640 PRINT #3,MID$(C$,T,1);
650 PRINT#3," ";
660 FOR G=1 TO 12*(T-X):NEXT G
670 NEXT T
680 WN=0
```



```
690 GOSUB 870
700 LOCATE 3,25
710 PRINT PT$
720 PT$=""
730 P$=INKEY$:IF P$="" THEN 730
740 IF P$=" " THEN GOTO 500
750 GOTO 730
760 REM***REEL DEFINE****
770 A$="":B$="":C$=""
780 FOR T=1 TO 255
790 AA$=STR$(INT(RND*9))
800 A$=A$+RIGHT$(AA$,1)
810 BB$=STR$(INT(RND*9))
820 B$=B$+RIGHT$(BB$,1)
830 CC$=STR$(INT(RND*9))
840 C$=C$+RIGHT$(CC$,1)
850 NEXT T
860 RETURN
870 REM***WINNING CHECK***
880 R1=VAL(MID$(A$,T-2,1))
890 R2=VAL(MID$(B$,T-2,1))
900 R3=VAL(MID$(C$,T-2,1))
910 IF R1+R2+R3<7 THEN AM=AM+5:SOUND 1
,90,50:SOUND 1,60,40:PT$="YOU WIN $5":
RETURN
920 IF R1=R2 AND R2=R3 THEN AM=AM+R1+R
2+R3:WN=1
930 IF R1=R2 AND R2=R3 AND R1=0 THEN G
OTO 990
940 IF R1=R2 AND R2=R3 THEN SOUND 1,40
```

```
,150,4:PT$="YOU WIN!!!":RETURN
950 IF R1=R2 OR R1=R3 OR R2=R3 THEN SO
UND 1,240,150,4:AM=AM+1:PT$="MONEY BAC
K":RETURN
960 SOUND 2,800,75
970 PT$="YOU LOSE"
980 RETURN
990 AM=AM+JK
1000 BORDER 0,13
1010 FOR T=100 TO 30 STEP-2
1020 SOUND 1,T,3,7
1030 NEXT T
1040 PT$="JACKPOT!"
1050 WN=1
1060 RETURN
1070 RETURN
1080 CLS
1090 PEN 1
1100 PRINT TAB(6);"LAS VEGAS"
1110 PRINT TAB(4);"By P.NESSBRETH"
```



```
1120 PRINT
1130 PEN 3
1140 PRINT"   As a visitor to  Las Veg
as, you must gamble on the fruit machi
ne"
1150 PRINT
1160 PRINT"   You start with   $10. St
art the reelsspinning by pressingthe s
pacebar"
1170 PRINT:PRINT"     If your reels tot
al less than 9, you win $4"
1180 PRINT:PRINT"     If you get two  n
umbers the same,   then you get your
 money back. All 3 & you receive 3 x t
he number that cam
e up";
1190 PRINT"3 x '0' and you win the jac
kpot"
1200 WHILE INKEY$=""
1210 WEND
1220 RETURN
1230 REM***GAME OVER***
1240 FOR T=1 TO 26
1250 PRINT
1260 SOUND 1,20*T
1270 NEXT T
1280 LOCATE 5,12
1290 PRINT "GAME OVER"
1300 LOCATE 1,25
1310 END
```

Inventing and Developing Your Own Games Programs

By Series Editor, Tim Hartnell

It's all very well spending your time typing in programs like those in this book, but there is sure to come a time when you decide you'd like to develop some games programs of your own. In this section of the book, I'd like to discuss a few ideas which may help you write games which you'll both enjoy developing and — more importantly — you and your friends will enjoy playing.

HAVE A CLEAR GOAL IN MIND

Although in many (perhaps most) cases, your computer program will take on a life of its own as you write it, developing away from the concept you had in mind when you started programming, it is important at the outset to have a pretty good idea of what your game will involve.

This is not as obvious a suggestion as you might think. Of course, you'll know if you're developing a 'chase the ghosts around the maze as you eat the power pills' program that you are going to need a different sort of program layout to one which places you inside a Haunted Oak, peopled with gremlins and halflings. But you have to go beyond the basic "I'm going to write me an Adventure" stage to work out such things as (a) what the object of the game will be; (b) what the screen display will look like; (c) what variables, and variable names, you'll need;

(d) the nature of the player input; (e) how 'winning' or 'losing' will be determined; and so on.

Let's look at these one by one.

THE OBJECT OF THE GAME

This can usually be stated very succinctly: "To find the lost treasure of the Aztecs"; "To destroy as many asteroids as possible before running out of ships"; or "To play a game of chess". But even though this stage of the game production can be accomplished very quickly, it should not be overlooked. Get this statement — which might be just a sentence, or may run to a paragraph length or more, if there is more than one 'screen' to be worked through, with a different scenario for each screen — down in writing.

You may well discard the original aim as the program develops, and it looks like the direction it is taking is better than the one you first thought of. Despite this, it is important to have something concrete to aim at, to stop you wasting hour after hour doodling aimlessly.

THE SCREEN DISPLAY

I've found that making a sketch, or sketches, of what the display will look like once the program is up and running, is of tremendous benefit. Once you have your drawing, and it doesn't matter how rough it is so long as it shows all the important things you want on the screen, and their relative positions and size, you'll discover the program concept is likely to crystalize.

As well as seeing immediately how you will write parts of the code to achieve the game's aim, you'll get an idea of whether or not the game is even worth writing in the form you had considered. Perhaps the game will be too complex if you leave everything on the screen you were intending to; or maybe most of the screen will be wasted space, so that a more complicated game scenario should be devised.

I've discovered that sketching the proposed screen display before starting to program is particularly useful, especially when creating arcade and simulation games. You get an indication of the variables you'll need, the user-defined graphics, the kind of player inputs which will be most conducive to good player interaction, and so on.

Simulation games, as you probably know, are those in which the computer models an external reality — such as running a cake shop, a war, or an airport — and allows you to experience (after a fashion) what it would be like to take part in such an activity in real life. Simulation games are not particularly difficult to write — in terms of needing clever coding — but instead demand a methodical, painstaking approach to the program.

In my book *The ZX Spectrum Explored* (Sinclair Browne, 1982), there is a program with the unlikely name of 'Workin' for the Man', in which you are running a factory, staffed with a highly-erratic workforce, involved in the manufacture of some mythical product called 'The Zibby'. The player gets a factory report two or three times a week, and from this report has to decide how many staff he or she will hire or (attempt to) fire, how many Zibbies will be the production target for the week, and so on.

This report is the key to the program, and when I wrote the game, I started by making a sketch of how the screen would look. It was a bit like this:

```
FACTORY REPORT: WEEK 5

Capital in hand is $2,657.92


Your stores hold 12 Zibbies worth $169.68
They sell for $14.14 each and cost $7.41
each to make


Workforce is 7 people
Their wages are $41 each and the wage bill
this week is $287
```

Each person can make 10 Zibbies a week,
a total output of 70

Once I had this sketch drawn up, I was ready to go. As you can see, it gives a very good indication of the variables which will be needed. For a start, I know I'll have to control the number of the week, the capital, the contents of the stores (and their value) and so on.

I found that once I'd completed the screen display sketch, the rest of the program was relatively easy to write. Doing a sketch in this way gives you an instant guide to the main variables you'll need.

USE HELPFUL VARIABLE NAMES

I also tend to use variable names which relate in some way to that which they are representing, as it saves having to keep a list of the variables which have been assigned, and what they've been assigned to. For example, I could use WK for week, CH for capital in hand, MZ for the cost of making each Zibby and SZ for the selling price. If Z was the number of Zibbies, I would know that the total value of Zibbies I had was Z (the number of them) multiplied by SZ (their selling price) and it cost me Z multiplied by MZ (their price of manufacture) to make them. My profit, if I sold them all, would then be Z*SZ minus Z*MZ.

If you follow a similar idea, you'll find it is much easier to keep track of what is happening in your program than might otherwise be the case.

THE NATURE OF THE PLAYER INPUT

It's important to make games *easy* and fun to play. It's not good having the best Asteroids-derivative program in the world if players have trouble hitting the fire button because you've placed it right next door to the 'rotate' control.

Many programs which provide 'up', 'down', 'right' and 'left' controls, automatically use arrow or cursor keys, even though these might be most inconvenient for the player to use. Have a look at your keyboard, and see if you can find better ones. I often use "Z" and "M" for programs which need just left and right movement, with the space bar for fire. These keys seem logical to me, and no player time is wasted in learning them, or trying to remember them when the game is underway. In a similar way, I tend to use "A" (for up) and "Z" (for down) for the left hand, and the "greater than" and "less than" keys for left and right (pointing out to the player that the < and > symbols point in the relevant directions).

Use INKEY$ or GET$ whenever you can, to prevent the player from having to use the RETURN or ENTER keys to get the program underway.

HOW THE GAME WILL END

The way the game will be won and lost needs to be defined, and clear to the player. Do you need to blast all the aliens to win, and will you lose automatically if one alien lands, and you've still got ships left, or only if you have no ships left. In a two-player game, is the loser the first player to lose three lives, or seven pieces, or does the game only end when the *difference* between the two scores is three or seven or whatever.

Work this out, and make it very clear to the player. Whether the goal of the game is to clear the left-hand side of the screen of the Screaming Widgies, or to clock up a fortune of $7.3 billion, it must be both clear to the player, and *possible to achieve*. A 'win condition' which can never be achieved on the higher levels of play is most unsatisfactory. No matter how difficult it is to do, you are only defrauding players if you set goals whose achievement is not possible within the constrictions you've put into the game.

I hope these five points may give you a few ideas on how you can go ahead and write programs which will be relatively easy to write, and which will be satisfying for you and your friends to play.

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

By Series Editor, Tim Hartnell

Usborne have released a number of very attractive books in their Usborne Computer Books series. Drawing on their vast experience in the field of producing low-priced, highly-coloured, attractive books for young readers, they've produced some books which will enlighten both young and not-so-young readers.

I'll look at three of their titles, three which cover just about the whole field of computer interests:

Information Revolution
(Lynn Myring and Ian Graham, Rigby).
Presenting an eminently readable introduction to the 'revolution' which covers such fields as computers (of course), text information services via the television screen, word processing, 'future phones' and satellite communications, *Information Revolution* is an ideal guide for the person who wants an easy-to-read introduction to the field.

Computer Jargon
(Corinne Stockley and Lisa Watts).
The tone of this book is set by the frontispiece, which has a number of odd little coloured robots sitting around a table laden with computer junk, pointing at each piece saying "This is a disk drive", "This is a digital tracer" (!) and "This is a printer".

Robotics — What Robots Can Do and How They Work
(Tony Potter and Ivor Guild).
This is definitely a candidate for the award of 'the longest

title of the year'. But it is very accurate. Don't be put off by the pretty pictures, as you'll soon discover this book has a lot of solid information. Topics covered include "What robots can and cannot do", "How arm robots work", "How to teach a robot" and "Build your own micro-robot"; this last section actually includes nine pages of circuit diagrams and all to build a little two-motor robot which, following a program typed into your micro, will run about the floor. Robotics is a field of the near future (with personal robots certain to be a bigger craze — when 'real robots' finally arrive — than computers will ever be).

Practise Your BASIC
(Gaby Waters and Nick Cutler).
You'll find this book — which predictably contains a number of exercises, puzzles and problems to solve by writing programs — should be useful in giving you a number of 'core problems' which will run on your computer and which can then be modified to take advantage of your system's special features. Program listings include 'Pattern Puzzles', 'Jumping Man', 'Horse Race', 'Word Editor' and 'Treasure Hunt', a mini-Adventure.

Help With Computer Literacy
(June St Clair Atkinson, Houghton Mifflin).
This is a large format book with an attractive cover, fairly priced for its 122 pages. It appears to be aimed at the early to middle years of secondary education, but contains a lot of material which those teaching younger children could easily adapt. Although it avoids the 'Gee Whiz' approach of the Usborne texts, it uses cartoons and diagrams to get its message across in an inviting manner.

The Interface Computer Encyclopedia
(Ken Ozanne, Interface Publications).
Compiled by a lecturer in mathematics at the NSW Institute of Technology, this work could perhaps be more accurately called 'The Computer Book of Lists', rather

than an encyclopedia. It contains annotated references to 'all' microprocessors, 'all' microcomputers, and 'most' microcomputing magazines. The inverted commas are there because — as the author admits candidly in his introduction — any such work is likely to be out of date even before it is published. Fat (445 pages) with minimalist presentation (the whole book is dumped directly from a word processor onto a dot-matrix printer) you'll find this a useful work if you want a ready reference to chips, computers and the ever-growing field of specialist magazines.

Computer Resource Book — Algebra
(Thomas Dwyer and Margot Critchfield, Houghton Mifflin).
Dwyer and Critchfield have clocked up an enviable string of successful computer books, and this one, part of a series, shows why. With simple, but valuable programs, the authors lead the reader (who can be a secondary student, or an instructor) through most of the phrases of the BASIC programming language which are common to all low-priced computers, and most educational time-sharing systems.

Apple II BASIC
(David Goodfellow, Tab Books Inc.).
Attractively packaged, this book is clearly laid out, with an abundance of example programs; it takes a commendable approach to the business of teaching programming, with the qualities of 'programming style' introduced without fanfare. In the crowded field of 'how to program your Apple' books, this one stands out. Much of the material presented is applicable to any microcomputer.

Pre-Computer Activities
(Dorothy Diamond, Hulton Educational).
This practical guide for teachers and parents can help make children familiar with essential computer processes and language before they have hands-on ex-

perience. The book contains a number of interesting activities, including investigating binary numbers using little lights, and working with cardboard 'calculators' before getting to the real thing. The discussion on computer graphics is enlivened by reference to the solid blocks which make up a 'Pacman' figure.

Word Processing Experience
(Janet Pigott and Roger Atkins-Green, Stanley Thornes Publishers Ltd.).
Designed for schools, but ideal for adapting if you'd like to increase your skill with a word processor (or simply because you'd like to see what word processors can do so you can write one for your own microcomputer), this book looks at the mechanics of word-processing, while passing on a great deal of useful information about word-processing techniques.

An Introduction to Micro-electronics and Micro-processor Systems
(G H Curtis and P G Wilks, Stanley Thornes Publishers Ltd.).
This work was written for junior college students and older school pupils, as well as for non-specialists who wanted a comprehensive — if dry — technical introduction to the subject. The going is not easy, but it's worth the effort. Topics covered include 'Logic', 'Programming the Microcomputer' and 'Analogue, Binary and Digital Systems'.

Computer Images — State of the Art
(Joseph Deken, Thames and Hudson).
This is a beautiful book, large and glossy, and packed with quality full-colour computer-generated (or, in some cases, computer-modified) images. The whole fascinating field of modern computer graphics is discussed — from television programme introductions using photographs which are colour-modified, twisted and tweeked, to the use of incredible high-resolution images in

simulators for flight training and tank manoeuvring. You'll read (and see) how computers are used to produce images, how these are used for education and communication, why 'art for art's sake' is a goal worth pursuing, and how computer images can evolve using processes uncannily akin to the processes by which groups of cells multiply and divide. If you want to see what can be done with high resolution graphics and when time, money and skill abound, you should get this book.

Computer Bluff
(Stephen Castell, Quartermaine House Ltd.).
A much more valuable book than its title indicates, it contains a lot of information on the what and how of computers, along with a generous dollop of computer jargon (or 'How to Cheat in Computer-Speak'). The style is gentle and amusing, with no appalling puns or excessive asides (such as 'didja get that joke, buster?'). A pleasant, painless book which you can digest, then give to a parent.

Penguin Books has moved into the computer field with enthusiasm. As well as a 'Getting the Most Out of Your...' series, they have a number of games books. Two which stand out are **The Penguin Book of VIC 20 Games** (Paul Copeland) and **The Penguin Book of Commodore 64 Games** (Robert Young and Paul Copeland). Priced at £4.95 each, these large format books include such programs as 'Space Venture', 'Oil Rig' and 'Red Alert'. Worth buying, even if you do not have a VIC or a Commodore 64, simply as a source of ideas for new programs to create on your own microcomputer.

Arcade Games for Your VIC 20 and **Arcade Games for Your Commodore 64** (Brett Hale, Corgi/Addison-Wesley) by contrast, are definitely only for those who have the machine specified. The programs are locked irrevocably to the computer named. Taking advantage of a number of machine-specific features (such as sprite graphics on the 64), Brett has produced a selection of around 20 programs for each machine. Each one is listed twice, the first time for the joystick and the second time for the keyboard. Titles include 'Galaxy Robbers', 'Bullet Heads' and 'Yackman'.

CREATING ADVENTURE PROGRAMS

There are a number of books, some of which are aimed at computer owners, which will help you if you are one of the many, many computer games players who are interested in developing 'Adventure' and 'Dungeons' type programs. The place to start is with TRS Hobbies' **Dungeons and Dragons** (TM) Basic Set, which comes with the introductory rule book, Dungeon Dice (tm) and an instruction module, along with a sample scenario 'The Keep on the Borderlands'. If you're new to the field, you should start with this set to give you an idea how 'real life' Adventure programs are built up.

Additional information is provided by **Fantasy Role-Playing Games** (J. Eric Holmes, Hippocrene Books Inc.) which looks at the whole field and, despite some disparaging things to say on computer versions of such games, is worth looking for. Another overview of the field — with more sympathetic comments on the use of computers — is provided by **Dicing With Dragons — An Introduction to Role-Playing Games** (Ian Livingstone, Routledge and Kegan Paul), which includes a full 'solo Adventure', a review of the major games on the market, and a fascinating chapter on the pleasures and perils of being Dungeon Master in 'Playing God'.

Fantasy Wargaming (compiled Bruce Galloway, published Patrick Stephens) provides a complete unified system for 'historically accurate' (or at least in tune with the beliefs and circumstances of individuals in the peasant, feudal-economy times in which many Adventures are set) games. The fight, weapon and monster tables alone are worth the book, as many of their ideas can easily be incorporated into your Adventures.

There are two computer Adventure books which you could get to help you in the fascinating area of producing Adventure games on your machine.
Creating Adventure Programs on Your Computer (Andrew Nelson, Interface Publications).
Written by the author of *More Games for Your VIC20* and *Games for Your TI 99/4A*, in the Virgin Books games series, this book takes you through the task of developing an Adventure program of your own, concentrating more on the 'Loot and Pillage' school of gaming than the Scott Adams' 'solve this puzzle to advance' field. Three complete Adventure programs are included.
Write Your Own Adventure Programs for Your Microcomputer (Jenny Tyler and Les Howarth, Usborne) is a much quicker introduction to the field than Nelson's, but nevertheless packs a lot of valuable information into its 48 pages. Step-by-step instructions are provided for creating an Adventure from scratch. A complete program — 'Haunted House' — is included in the book.

The Age of Computers is the general title of four fine books produced by Wayland Publisher Limited. Each priced at £4.95, the books present a careful, but inviting, view of four aspects of the computer field, one on the history of computers and the others looking at specific areas of modern computer application. Each book is by Ian Litterick and Chris Smithers. The four titles are **The Story of Computers,** with Charles Babbage and Uncle Sir Clive Sinclair just inside the cover (and these two pictures accurately sum up the historical period covered by the book); **How Computers Work** (with chapter headings including 'Bits, Bytes and Binary', 'Decision-making by Transistor', and 'Talking With Computers'); **Computers in Everyday Life** (such things as 'Robots in the Home', 'Magnetic Money' and 'Medicine and the Disabled');

and **Computers and You** ('Computopia', 'Big Brother', 'War and Peace' and — a fascinating final chapter — 'Will Computers Need Us?').

Inside BASIC Games
(Richard Mateosian, Sybex).
This book is a slightly overwritten guide to understanding computer games. You'll learn how to write interactive programs in BASIC and how the principles of system development are applied to small computers. The book also looks at how the features of specific small computer systems have been supported in BASIC. If you can contend with the verbiage, you'll find this book well worthwhile.

1001 Things to Do With Your Personal Computer
(Mark Sawush, Tab Books).
Big and fat, and full of ideas, you'll find much here of interest to enlarge your computer horizons. The book tells you about writing music and stories with your computer, aiding a mechanic or a carpenter, solving simultaneous equations, astrology and much, much more.

Stimulating Simulations
(C.W. Engel, Hayden Book Company).
Here are 12 unique programs written in a good, general version of BASIC. The fascinating programs include 'Forest Fire', 'Rare Birds' and 'The Devil's Dungeon'. You're sure to enjoy playing those three, along with 'Diamond Thief', in which the computer decides who has committed the crime, then challenges you to discover which of the suspects is guilty. The material in this book is generally tightly programmed, and can be a helpful source of ideas to improve your own computer work.

The BASIC Handbook
(David A. Lien, Compusoft Publishing).
This is an encyclopedia of the BASIC language. It comes into its own when you find a program in a magazine or

book which you'd love to try, but are frustrated because it is written for another version of BASIC. Every BASIC word you've ever heard of (and many you may not have, such as NE, GOTO-OF and LE) is in here, along with a number of variations, one of which will almost certainly be on your machine.

BASIC Computer Games
(David Ahl, Creative Computing Press).
This is a classic work, still selling well despite the fact it was one of the first such books — if not *the* first — on the market. David Ahl has been in personal computers even before there were such things. Although several of the games are overly-dependent on the random number generator, you'll find there are many, many games you'll want to adapt and improve for your own computer.

How to Buy (and Survive) Your First Computer
(Carolee Nance Kolve, McGraw-Hill Book Company).
When is a business ready for a computer? How do you make an intelligent, informed choice among the hundreds of computers available? Will a computer improve a company's operations? Answers to these and a score of similar questions are in this book, which explains in detail what to consider before buying, how to select the right computer, and what to do after ordering the computer to ensure a successful installation. Ms Kolve has over 15 years computer experience (including a stint with IBM) and brings her experience to bear in a relatively easily-digestible guide.

Your First BASIC Program
(Rodnay Zaks, Sybex).
This book, liberally illustrated with large red dinosaurs in a variety of situations vaguely related to the text (one, for instance, as a cowboy getting tangled up in his ropes with the caption 'Be careful when looping'), is a gentle and worthwhile introduction to the not-so-secret secrets of programming in BASIC. When you want to move

beyond just typing in other people's programs from books and magazines, this may be a good place to start.

This bibliography was compiled by the series editor, Tim Hartnell, who has felt constrained not to recommend any of his own books. However, he asked us to mention two which could be of use and interest to you.

The first is **The Personal Computer Guide** (Virgin Books) which explains what a personal computer is, and answers questions like "Will it help my kids?", "What sort of games can we play on it?" and "What can I use it for in the home?" The book describes many of the most popular computers available today, with illustrations, technical specifications and other information to help you to choose the equipment best suited to your requirements. Also included is an introduction to BASIC programming, with details of programs suitable for use in the home, a list of suppliers and user clubs, and a guide to further reading. There are also chapters covering the personal computer's history and its future. When you're ready to upgrade, you'll find this book a good, unbiased, reference work which looks at the choices facing you.

Tim Hartnell's Giant Book of Computer Games.
Described by *Personal Computer News* as 'a good source of ideas', this 386-page book, published by Fontana, for £3.95, contains over 40 programs which will run with minimum modifications on most popular microcomputers. The games include chess (of a sort!), a 17K Adventure and 'Hyperwar'.

The Virgin Computer Games Series

GAMES FOR YOUR AMSTRAD

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

The games include PROTECTOR (save the Earth from marauding bombers); GOLD BULLION RAID (rob the train and hide the loot!); LAS VEGAS (try your luck on your own animated one-armed bandit); DIGIT MUNCHER (an exciting new version of the old pill-eating arcade game); ALL SYSTEMS GO (get your computer operational again in this novel and brain-taxing adventure); VIDEO SALES (can you survive the cut and thrust of the video business?); DEMON DRIVER (control your fast-moving red rider on the twisting, turning roadway).

GAMES FOR YOUR AMSTRAD 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 extend the programs in the book.

Programs of originality and quality for all the family.

Virgin

ISBN 0 86369 077 7

United Kingdom £2.95
Australia $9.95 (recommended)

THE VIRGIN COMPUTER
GAMES SERIES

GAMES FOR YOUR AMSTRAD

AMSTRAD CPC

MÉMOIRE ÉCRITE
MEMORY ENGRAVED
MEMORIA ESCRITA

https://acpc.me/