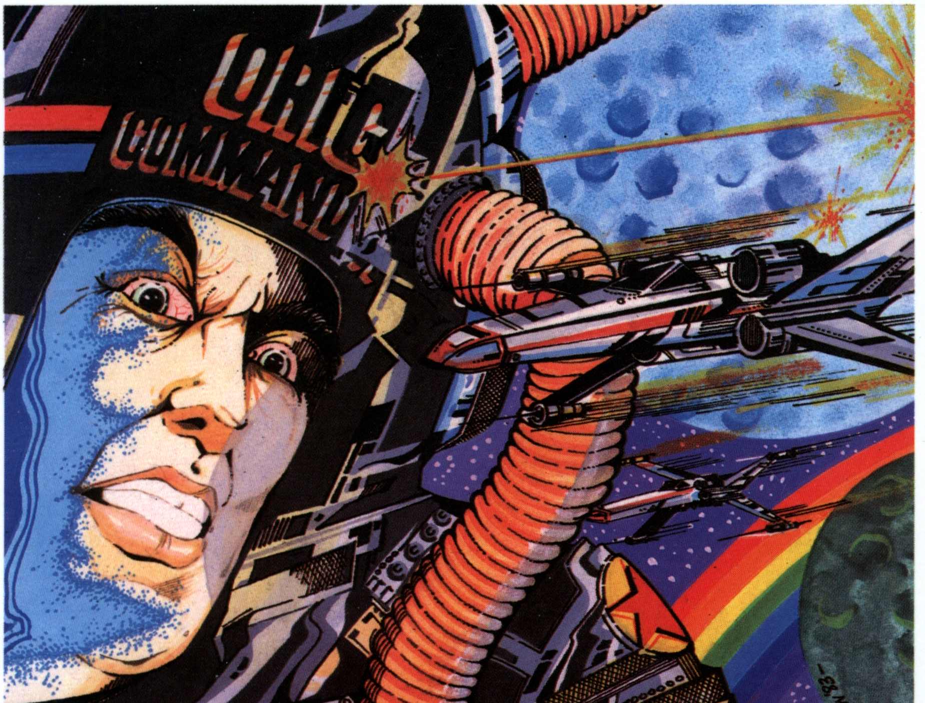
DYNAMIC GAMES FOR YOUR ORIC



CLIVE GIFFORD

DYNAMIC GAMES FOR YOUR ORIC

CLIVE GIFFORD

Best Wishes,

[illegible]

INTERFACE PUBLICATIONS

44-46 Earls Court Road, London W8 6EJ

Interface Publications, London and Melbourne

To all at the Blue Mist.

Published in Great Britain by:
Interface Publications,
44-46 Earls Court Road,
London, W8 6EJ.

ISBN 0 907563 45 7

This volume is part of the *Tim Hartnell 'Success in the Fast Lane'* programming series published by Interface Publications.*

**(reg. TM)*

Cover Illustrator: Keith Gidlow.

CONTENTS

ACKNOWLEDGEMENTS

The author wishes to thank a number of people for their help in writing this book: To my family without whom it would not have been possible; To Scott for his valuable assistance and contributions; To Simon Palmer for his help with equipment; To Alice and Jeff for their friendship; To all at Interface for continual advice; To all my friends at Spelthorne College, especially Caroline, for support; Finally, to Sally, without whom my daily routine would collapse.

AUTHOR'S INTRODUCTION

Your Oric computer has been well designed and is a good investment. Its graphic and sound capabilities, though hard to access, are very powerful and this coupled to its massive memory and easy-to-use keyboard make it a winner.

In this book, you will find the result of hundreds of hours programming and writing. This book offers 36 games and patterns and a couple of routines which, although they are not games, will help you in any future programming that you do. As I have written the programs, I have found the machine's capabilities stretch and stretch and my own programming ability improve. I hope the same thing happens to you when you read and use the book.

Every game is prefixed by a reasonable length introduction. In these introductions, I have not only tried to explain how the game should be played, but have also given some idea of how the program works and where the vital routines are located. Add to this the useful chapters on programming, applications, loading and saving and a comprehensive glossary, and you will see that this book is more than just a programs book.

Here then, is my book – I hope it provides you with much enjoyment and teaches you a few things.

Happy Gamesplaying!

Clive Gifford
Ashford, Middx
September 1983

PROGRAM NOTES

Before you start typing in the programs, there are a few points which should be taken into consideration. Firstly, there are a couple of idiosyncrasies concerning the printouts. The '£' sign on the Oric is given the ASCII code 95, but unfortunately, most of the printers available convert the code 95 into an underline character, '_'. Therefore, wherever you see the underline character, type a pound sign in. Also, the Oric's program lines cannot exceed 78 characters, which can be a bit of a nuisance when dealing with complex calculations. Several lines in the book, therefore, have no spaces between statements. Be careful when you type these lines in.

Many of the programs featured in this book are of some length and it is a good idea to save them on tape before playing them. It only takes a small knock as you battle your way through Laser War to disengage the power lead and lose the program and believe me, it does happen. Saving a program on tape only takes a short time (a couple of minutes) but it can save many hours work. If you have any trouble with your loading and saving reliability, then I suggest you consult the chapter dealing with this, near the back of the book.

EATIE SWEETIE

Once the craft lands, you follow the landing party as they scour the area looking for plant and animal specimens to take back to their planet, millions of miles away. Suddenly, there is a loud noise and all the others run back to the spaceship – but you are too slow and the craft leaves. You are alone on a planet, light years away from your home, very afraid and incredibly hungry! You chance upon a house where, to your surprise, there are tasty sweets laid all over the floor. Your only chance is to build up enough strength to contact the ship (mental telepathy is very exhausting) and explain your situation. You start eating . . . munch, munch, munch . . . but you are being followed by a robot programmed by NASA to catch you. It was the robot which laid the sweets and it is the robot who you must now avoid. There are other hazards too. Time is running out because if you take too long your ship will be too far away to contact. Also, the walls of the house have been electrified and if hit, they will sap your energy. You can withstand being caught or being electrified three times – after all, you're a tough creature!

Not everything is against you, however, as there are Krypton Clusters in the far corners of the house. If eaten they will replenish your strength to the extent that you can survive another onslaught from either the walls or the robot, and you are given a few extra seconds in which to continue munching. You will find a small passage linking the North and South wings of the house. The robot has not been programmed for this so it will not follow you if you choose to use this exit.

This program is the result of many hours playing BASIC 'Pacman' type games. Many of these programs have lacked the speed of machine code and any form of originality. In this program, I have tried to increase the interest by adding a new concept to the game, that of the stranded alien. Many other things have been changed but the basic aim still remains the same: eat the pills/sweets as quickly as possible while avoiding the monsters/ghosts/robots. Adding a novel concept to a traditional program is an excellent way of writing new games. What starts as just a little twist on the old idea can, and often does, develop into a totally separate game, sometimes bearing no resemblance to the original game.

You control the alien using the arrow keys. The figure in the left-hand corner is the time left, while in the opposite corner the score is displayed. After being hit or captured, a green-coloured number will appear in the central yellow block. This depicts the number of lives you have left. There are three skill levels, '1' is very difficult, '2' is far from easy and '3' is fairly gentle and suitable for arcade game beginners. There is a high score feature to keep you amused for some considerable time.

I will just mention a few of the programming features found in the listing. Line 33Ø uses DEEK to read the keyboard, and lines 35Ø and 36Ø convert the reading into the X and Y co-ordinates controlling the alien's position. Lines 37Ø to 436 and lines 7ØØ to 74Ø control the robot's movements. The program uses very simple intelligence for the robot but, as you can see by the robot's positioning, the results are very effective.

The various subroutines are all indicated by REM statements and are all fairly straightforward to follow.

If you succeed in obtaining enough energy then the program flows to lines 4ØØØ onwards, which prints a high resolution picture of the alien's planet and plays a little suitable music.

If you are defeated, the computer goes to line 5ØØØ where a 'defeat' message is printed, the score and high score displayed and the chance of another game is offered.

The routine starting at line 8ØØØ prints the maze, while the routine starting from line 9ØØØ defines the alien, robot, sweet and maze graphics.

```
10 REM*****EATIE SWEETIE******
12 REM******CHECK THAT KEYCLICK IS ON***
****
15 GOSUB 9000:HI=0:PRINT CHR$(6)
20 GOSUB 8000
30 PAPER 0:INK 3
40 X=10:Y=X:R$="@":A=14:B=19
50 SC=0:L=3:XX=X:YY=Y
60 REM*****GHOSTS MOVES*****
65 PLOT 2,0,STR$(N)
70 N=5000:TC=0:AA=A:BB=B
299 TC=0
300 REM****MAIN LOOP****
302 PLOT 18,25," ":PLOT 19,2," ":N=N-4
305 PLOT 1,0,STR$(N)+"  EATIE SWEETIE  "
+CHR$(96)+" 1983  "+STR$(SC
)+"  "
330 C=DEEK(783):TC=TC+1
340 PLOT X,Y,"%":PLOT X,Y+1,"&"
345 PLOT A,B,"'"
347 IF N<0 THEN 5000
350 X=X+(C=48351)-(C=48255)
360 Y=Y+(C=48375)-(C=48319)
362 IF X=18 AND Y=25 THEN X=19:Y=3
364 IF X=19 AND Y=2 THEN X=18:Y=24
365 IF TC<J THEN 440 ELSE TC=0
370 IF A>X THEN A=A-1
380 IF A<X THEN A=A+1
```

```
390 IF B>Y THEN B=B-1
400 IF B<Y THEN B=B+1
410 IF SCRN(A,B)=37 OR SCRN(A,B)=38 THEN
 3000
420 IF SCRN(A,B)=35 THEN GOSUB 700
430 IF B>24 THEN B=24
431 IF B<3 THEN B=3
434 PLOT AA,BB,R$
435 IF A<2 THEN A=2
436 IF A>37 THEN A=37
440 PLOT XX,YY," ":PLOT XX,YY+1," "
445 IF SCRN(X,Y)=111 OR SCRN(X,Y+1)=111
THEN GOSUB 1500
447 IF SCRN(A,B)=37 OR SCRN(A,B)=38 THEN
 3000
450 IF SCRN(X,Y)=35 THEN 2000
460 IF SCRN(X,Y+1)=35 THEN 2000
470 IF SCRN(X,Y)=64 OR SCRN(X,Y+1)=64 TH
EN GOSUB 900
490 XX=X:YY=Y:AA=A:BB=B
500 GOTO 300
700 REM****MOVE FROM WALL****
710 A=A+RND(1)*5-RND(1)*5
720 B=B+RND(1)*4-RND(1)*4
722 IF A>37 THEN A=37
723 IF A<2 THEN A=2
724 IF B>24 THEN B=24
```

```
725 IF B<3 THEN B=3
730 IF SCRN(A,B)=35 THEN 710
740 RETURN
900 REM******MUNCH****
910 SC=SC+13:IF SC>9000-((1000*J)+(1000-
(200*J))) THEN 4000
920 PLAY 1,7,1,70:RETURN
1500 REM****BONUS SWEET****
1510 PING:WAIT 25:PLAY 1,7,1,150
1520 N=N+250:L=L+1
1530 PLOT 22,18,STR$(L)+CHR$(3)
1540 RETURN
2000 REM*****HIT WALL*****
2005 EXPLODE
2010 L=L-1:IF L=0 THEN 5000
2020 INK 0:WAIT 50:INK 3:WAIT 50
2030 PLAY 1,7,1,8
2040 X=10:Y=X:XX=X:YY=X
2050 PLOT 22,18,STR$(L)+CHR$(3)
2060 WAIT 150:GOTO 299
3000 REM*****CAPTURE****
3010 ZAP:WAIT 20:EXPLODE:L=L-1
3012 FOR H=1 TO 6:INK 0:WAIT 12
3015 PLOT XX,YY," ":PLOT XX,YY+1," ":PLO
T AA,BB," ":INK 3:WAIT 12
3017 NEXT H
3020 PLAY 0,0,0,0:X=10:Y=X:XX=X:YY=X
3030 A=19:B=19:PLOT 22,18,STR$(L)+CHR$(3
)
```

```
3040 IF L=0 THEN 5000
3050 GOTO 299
4000 REM*****SUCCESS******
4010 WAIT 100
4020 HIRES:PAPER 0:CURSET 50,50,0
4030 FOR R=1 TO 40:CIRCLE R,1:NEXT R
4050 CURSET 30,55,0:CIRCLE 6,0:CIRCLE 11
,0
4060 CURSET 70,20,0:CIRCLE 9,0:CIRCLE 6,
0
4070 CURSET 100,100,1:CIRCLE 36,0:CIRCLE
 35,0:CIRCLE 34,0:CIRCLE 40
,0
4080 CURSET 18,15,1:CIRCLE 15,0
4090 FOR T=1 TO 200:CURSET INT(RND(1)*23
9)+1,INT(RND(1)*199)+1,1:NE
XT T
4100 FOR T=1 TO 20:CURSET INT(RND(1)*210
)+22,INT(RND(1)*170)+23,1
4110 CIRCLE INT(RND(1)*4)+2,1:NEXT T
4130 PRINT:PRINT"  Well Done!  Eatie is
home now!!!"
4140 PLAY 1,0,2,8:PRINT CHR$(6)
4150 MUSIC 1,4,10,8:WAIT 80
4160 MUSIC 1,4,12,8:WAIT 80
4170 MUSIC 1,4,8,8:WAIT 80
4180 MUSIC 1,3,8,8:WAIT 80
4190 MUSIC 1,4,3,8:WAIT 150
```

```
4200 PLAY 0,0,0,0
4210 IF KEY$="" THEN GOTO 4210
4220 TEXT:PRINT:PRINT:PRINT"
 END OF GAME."
4500 END
5000 REM*****DEFEAT******
5005 PLOT 24,18,CHR$(1)
5010 WAIT 50:INK 1:WAIT 50
5015 FOR R=1 TO 38
5020 FOR T=1 TO 25
5030 PLOT R,T," "
5040 NEXT T
5050 NEXT R
5060 PLOT 2,10,"You did not gain enough
strength"
5070 PLOT 3,12,"to get back to your own
world."
5080 PLOT 2,14,"You are doomed to remain
 on this"
5090 PLOT 13,16,"planet."
5100 WAIT 250:IF SC>HI THEN HI=SC
5120 PLOT 10,6,"HIGH SCORE: "+STR$(HI)
5130 PLOT 4,20,"For another attempt, pre
ss key S"
5140 GET A$:IF A$="S" THEN GOTO 20
5150 GOTO 5140
8000 REM****DRAW MAZE*****
8010 CLS:PRINT:PRINT
```

```
8040 PRINT"#############################
########"
8050 PRINT"#@@@@#@@@@@@@@@@#@@@@#@@@@@@@
@@@@@@##"
8060 PRINT"#@o@@#@@@@@@@@@@#@@@@#@@@@@@@
@@@o@@##"
8070 PRINT"##@@@#@@@@@@@@@@#@@@@#@@@@@@@
@@@@@###"
8080 PRINT"###@@#@@@@@@@@@@#@@@@#@@@@@@@
@@@@####"
8090 PRINT"####@######@@#####@@###@@####
########"
8100 PRINT"###@@#@@@@@@@@@@@@@@@@@@@@@##
@@@@####"
8110 PRINT"##@@@#@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@##"
8120 PRINT"#@@@##@@@#########@@@@@@@@@@@
@@@@@@@#"
8130 PRINT"#@@###@@@@@@###@@@@@@@@@@###@
@@@@@@@#"
8140 PRINT"#@@###@@@@@@###@@@@@@@@@@###@
@@@@@@@#"
8150 PRINT"#@@###@@@@@@###@@@@@@@@@@###@
@@@@@@@#"
8160 PRINT"#@@@@@@@@@@@@@@@@@@@###@@@@@@
@@@@@@@#"
8170 PRINT"#@@@@@@@@@@@@@@@@@@@###@@@@@@
@@@@@@@#"
```

```
8180 PRINT"#######@@@@##@@@@@@@#########
#@@@@@@#"
8190 PRINT"#@@@@@@@@@@##@@@@########@@@@
@@@@@@@#"
8200 PRINT"#@@@@@@@@@@##@@@@########@@@@
@@@@@@@#"
8210 PRINT"#@@##@@######@@@@@@######@@@@
@@@@####"
8220 PRINT"#@@##@@######@@@@@@######@@@@
@@@#####"
8230 PRINT"#@@##@@######@@@@@@######@@@@
@@######"
8240 PRINT"#@@@@@@#####@@@@@@@@@@###@@@@
@###@o@#"
8250 PRINT"#@@@@@######@@@@@@@@@@@@@@@@@
@@@@@@@#"
8260 PRINT"#@@o@@######@@@@@@@@@@@@@@@@@
@@@@@@@#"
8270 PRINT"#############################
########"
8400 RETURN
9000 FOR P=46360 TO 46367
9010 READ N:POKE P,N
9020 NEXT P
9030 DATA 63,63,63,63,63,63,63,63
9050 FOR P=46592 TO 46599
9060 READ N:POKE P,N
9070 NEXT P
```

```
9080 DATA 0,0,0,12,30,12,0,0
9090 FOR P=46376 TO 46398
9100 READ N:POKE P,N
9110 NEXT P
9120 DATA 62,42,42,62,8,8,8,42
9130 DATA 62,28,28,28,62,54,54,0
9140 DATA 12,30,45,45,63,33,63,63
9200 REM*****TITLE******
9210 CLS:PAPER 4:INK 7
9220 FOR P=1 TO 950:PRINT"'";:NEXT P
9230 PLOT 10,10,"    E A T I E    "
9240 PLOT 10,11,"                 "
9250 PLOT 10,12,"  S W E E T I E  "
9260 WAIT 250:FOR T=1 TO 26:PRINT:NEXT T
9270 PLOT 10,10,"WHICH LEVEL? 1,2,3"
9280 GET L$:J=VAL(L$)*2:RETURN
```

EXECUTIVE DECISION MAKER

We all have to make decisions sometime in our lives, some of which we would rather not make. Why not make it easier on yourself, and let your computer decide for you.

In this program, inspired by those little executive dice which have three sides stamped 'NO' and three sides stamped 'YES', you have your mighty Oric ready and waiting to solve your biggest headaches. The range of responses is somewhat more than 'YES' or 'NO' as you will see when you run the program.

The computer switches into CAPS OFF mode and any question must begin with a capital letter and end with a question mark. All inputs apart from the question are single key entries without any need to press the Return key. GET is used to collect your response.

If your question begins with certain words then particularly cheeky responses are thrown back at you. Even the final decision is not as simple as the executive dice as there are seven possible answers, many as ambiguous as the answers we receive in real life.

I have tried to keep the BASIC code at a minimum while trying to retain a flexible response to the player's prompts.

```
10 REM**EXECUTIVE DECISION MAKER**
20 PRINT CHR$(20)
30 CLS:PRINT:PRINT"Hello Sir, May I be o
f help?"
40 GET A$:IF A$<>"y" THEN 40
50 PRINT:PRINT"May I have your name?":PR
INT
60 INPUT N$:PRINT:PRINT
70 PRINT"Thankyou, my name is Eric"
80 PRINT:PRINT"Now then, What is your qu
estion ";N$;"?"
90 PRINT:PRINT:INPUT Q$
100 REM***PHRASE PROCESSING/RESPONSE***
110 L=LEN(Q$)
115 Q$=LEFT$(Q$,L-1)
120 PRINT:PRINT:PRINT
130 IF LEFT$(Q$,4)="What" THEN PRINT"Fra
nkly I don't know ";Q$
140 IF LEFT$(Q$,3)="How" THEN PRINT"How
should I know, I'm only a m
achine"
150 IF LEFT$(Q$,3)="Who" THEN PRINT"I su
ggest that you have a look
in "
160 IF LEFT$(Q$,3)="Who" THEN PRINT:PRIN
T"WHOS WHO. That will tell
you":PRINT,Q$
170 IF LEFT$(Q$,4)="When" THEN PRINT"I a
```

```
m not interested in time sc
ales."
180 IF LEFT$(Q$,3)="Why" THEN PRINT"Ours
 is not to reason why,"
190 IF LEFT$(Q$,3)="Why" THEN PRINT"Ours
 is but to do or  die."
200 IF LEFT$(Q$,1)="W" OR LEFT$(Q$,3)="H
ow" THEN 420
210 REM******DECISION MAKING******
220 WAIT 200
230 R=INT(RND(1)*100)+1
240 IF R<41 THEN P=INT((R/10)+5)
250 IF R>40 AND R<61 THEN P=5
260 IF R>60 AND R<81 THEN P=6
270 IF R>80 THEN P=7
280 FOR X=1 TO P
290 READ L$
300 NEXT X
310 DATA "Possibly with some good luck",
"Definately","Certainly not
!"
320 DATA "Perhaps in the near future","M
aybe...","YES","NO"
330 CLS
340 FOR T=1 TO 8
350 PLOT 10,10,"DECISION MADE"
360 WAIT 35
370 PLOT 10,10,"             "
```

```
380 WAIT 35
390 NEXT T
400 WAIT 150:FOR T=1 TO 14:PRINT:NEXT T
410 PRINT"ANSWER:- ";L$;" ";N$
415 PRINT:PRINT:PRINT"Press my spacebar
";N$:GET A$:CLS
420 PRINT:PRINT:PRINT:PRINT
430 PRINT"Can I be of anymore help?"
440 GET A$:IF A$="y" THEN 80
450 PRINT CHR$(20):END
```

DROIDS

You have been cast into the Labyrinth Of Despair, a fearsome place filled with powerful forcefields which will destroy anything that comes into contact with them. And what is worse, a number of droids have been locked in with you. These creatures have been programmed to home in on you and to destroy you as quickly as possible. However, they have not been programmed for the forcefields, and using skill and cunning you can lure them to their destruction. You move by entering 'N', 'S', 'E' or 'W', and the droids can move one space at a time at the same rate as you but they choose the quickest way towards you and can also move diagonally. By putting a forcefield between yourself and a droid you should be able to lure it into the forcefield.

You are offered a skill level between three and 12 which converts into the number of droids in the labyrinth with you. Your aim, obviously, is to vaporise all of the droids before they get to you, and without walking into a forcefield yourself.

I suggest that you start off with around four or five droids. You will find that the odds of you succeeding are high and when you have nine or 1Ø robots, your task is almost impossible. But I will never forget that in one run, sadly when the printer was disconnected, I managed to clear a labyrinth containing 11 robots. The stuff that dreams are made of...

You are the 'H', the droids are the asterisks and the forcefields are 'O's.

```
10 REM*******DROIDS********
15 PAPER 2:INK 0
17 GOSUB 1010
20 GOTO 250
30 REM****MOVE DROIDS****
40 T=0
50 FOR E=1 TO L
60 IF A(B(E),C(E))=79 THEN T=T+1:GOTO 20
0
70 X=B(E):Y=C(E)
80 IF B(E)<D THEN B(E)=B(E)+1
90 IF B(E)>D AND RND(1)>.3 THEN B(E)=B(E
)-1
100 IF C(E)<F AND RND(1)>.3 THEN C(E)=C(
E)+1
110 IF C(E)>F THEN C(E)=C(E)-1
120 IF B(E)<2 THEN B(E)=2
130 IF B(E)>14 THEN B(E)=14
140 IF C(E)<2 THEN C(E)=2
150 IF C(E)>14 THEN C(E)=14
160 A(X,Y)=46
170 IF A(B(E),C(E))=79 THEN T=T+1:GOTO 2
00
180 IF A(B(E),C(E))=72 THEN A(B(E),C(E))
=36:GOSUB 300:GOTO 910
190 A(B(E),C(E))=42
200 NEXT E
210 IF T<CH THEN T=CH
```

```
220 CH=T
230 IF T=L THEN GOSUB 300:GOTO 970
240 RETURN
250 GOSUB 530
260 GOSUB 300
270 GOSUB 30
280 GOSUB 420
290 GOTO 260
300 REM****PRINT DISPLAY*****
305 PING
310 CLS
315 PRINT:PRINT TAB(24) "D R O I D S":PR
INT
320 IF T>0 THEN PRINT"Vaporised Droid Ta
lly: ";T
330 IF T=0 THEN PRINT:PRINT
340 PRINT:PRINT
350 FOR B=1 TO 15
360 FOR C=1 TO 15
370 PRINT CHR$(A(B,C));" ";
380 NEXT C
390 PRINT
400 NEXT B
410 RETURN
420 REM****PLAYER MOVE****
430 A(D,F)=46
440 GET A$
450 IF A$="N" AND D>2 THEN D=D-1
```

```
470 IF A$="S" AND D<14 THEN D=D+1
480 IF A$="E" AND F<14 THEN F=F+1
490 IF A$="W" AND F>2 THEN F=F-1
500 IF A(D,F)=79 THEN GOSUB 300:GOTO 990
510 A(D,F)=72
520 RETURN
530 REM***INITIALISE***
550 DIM A(15,15),B(12),C(12)
560 CLS
570 PRINT:PRINT"Please stand by for a mo
ment..."
580 CH=0
610 REM****PLACE WALLS****
620 FOR B=1 TO 15
630 FOR C=1 TO 15
640 A(B,C)=46
650 IF B=1 OR B=15 OR C=1 OR C=15 THEN A
(B,C)=88
660 NEXT C:NEXT B
670 REM****PLACE FORCEFIELDS****
680 FOR B=1 TO 20
690 C=INT(RND(1)*13)+1
700 D=INT(RND(1)*13)+1
710 IF A(C,D)=88 THEN 690
720 A(C,D)=79
730 NEXT B
740 DATA 4,4,13,8,8,3,12,7
750 REM****PLACE DROIDS*****
```

```
760 FOR E=1 TO L
770 D=INT(RND(1)*13)+2
780 F=INT(RND(1)*13)+2
790 IF A(D,F)<>46 THEN 770
800 B(E)=D:C(E)=F
810 A(B(E),C(E))=42
820 NEXT E
830 REM****PLACE HUMAN****
840 D=INT(RND(1)*13)+2
850 F=INT(RND(1)*13)+2
860 IF A(D,F)<>46 THEN 830
870 A(D,F)=72
880 RETURN
910 REM*****FINISH GAME*****
920 CLS:PRINT:PRINT"A droid has got you!
!":ZAP:ZAP:ZAP
930 CH=0:T=0:GOSUB 340
940 A$="":PRINT:PRINT"INPUT 'S' to start
.":INPUT S$:IF S$<>"S" THEN
 940
950 GOSUB 570:GOTO 260
970 CLS:PRINT:PRINT"You've defeated the
Droids!!"
975 MUSIC 1,4,6,8:WAIT 80:PLAY 0,0,0,0
980 GOTO 930
990 CLS:PRINT:PRINT"You've run into a fo
rcefield!!":EXPLODE
1000 GOTO 930
1010 CLS:FOR T=1 TO 26:PRINT:NEXT T
```

```
1020 PRINT"$$$$$  $$$$$  $$$$$  $  $$$$$
  $$$$$"
1030 PRINT" $  $  $   $  $   $  $   $  $
  $"
1040 PRINT" $  $  $$$$$  $   $  $   $  $
  $$$$$"
1050 PRINT" $  $  $  $   $   $  $   $  $
      $"
1060 PRINT"$$$$$  $   $  $$$$$  $  $$$$$
  $$$$$"
1070 PRINT:PRINT TAB(27) CHR$(96);" 1983
"
1080 FOR T=1 TO 14:PRINT:NEXT T
1090 PRINT:PRINT:INPUT"WHICH LEVEL (3-12
)";L
1100 IF L>12 OR L<3 THEN 1090
1110 WAIT 50:T=0:RETURN
```

HAWKS AND DOVES

You have the power of life and death over a population of hawks and doves in this game. You must decide how many hawks and how many doves to put inside a large cage. Too many hawks and the doves will not breed fast enough to feed them. Too many doves and the hawks will be overcrowded.

The computer provides you with a month by month read-out of the cage's population.

There is a high score feature which will tell you the best effort so far. This, of course, returns to zero each time the program is run.

The variables used are:

HI	=	High score counter.
FD	=	Random number of doves needed to feed each hawk.
CP	=	Hawk population.
MP	=	Dove population.
DA	=	Month.
X	=	Pause loop counter.

```
10 REM*****HAWKS AND DOVES*****

20 CLS:PAPER 4:INK 7

30 PRINT"WELCOME TO HAWKS AND DOVES."

35 PRINT:PRINT

40 PRINT"THE OBJECT OF THIS GAME IS TO C

REATE"

45 PRINT"A POPULATION OF HAWKS AND DOVES

 WHICH"
```

```
50 PRINT"WILL SURVIVE FOR AS LONG AS POS
SIBLE."
55 PRINT:PRINT:PRINT
60 PRINT"PRESS ANY KEY TO CONTINUE"
65 GET A$
70 HI=0
80 FD=RND(1)
90 CLS
95 PAPER 7:INK 0
100 PRINT:PRINT"HOW MANY HAWKS WILL YOU
START WITH?"
110 INPUT CP:IF CP>99 OR CP<1 THEN PRINT
"TRY AGAIN":GOTO 110
120 MUSIC 1,2,10,6
125 WAIT 120:PLAY 0,0,0,0
130 PRINT:PRINT"HAWK POPULATION: ";CP
140 CP=CP/3
145 PRINT:PRINT
150 PRINT"HOW MANY DOVES WILL YOU START
WITH?"
160 INPUT MP:IF MP>99 OR MP<1 THEN PRINT
"WRONG SIZE":GOTO 160
190 MUSIC 1,3,4,6:WAIT 120
195 PLAY 0,0,0,0
200 PRINT:PRINT"DOVE POPULATION: ";MP
210 FOR X=1 TO 1000:NEXT X:CLS
220 MP=MP/3
250 GOSUB 550
```

```
260 DA=0
270 DA=DA+1
280 PRINT:PRINT:PRINT"MONTH ";DA
295 PRINT"-------"
300 IF CP>MP/FD THEN CP=MP/FD
310 CP=ABS(CP+((8*CP-CP*MP/3)*FD))
320 MP=ABS(MP+((4*MP-MP*CP)*.01))
330 MUSIC 1,3,4,9:WAIT 40
332 MUSIC 1,3,8,9:WAIT 40
334 MUSIC 1,3,6,9:WAIT 50
336 PLAY 0,0,0,0
340 PRINT:PRINT:PRINT INT(CP);"HAWKS",,,
INT(MP);"DOVES"
370 GOSUB 550
380 IF CP<2 OR MP<2 THEN GOTO 400
390 GOTO 270
400 IF MP>2 AND CP>2 THEN GOSUB 550
405 PRINT:PRINT:INK1
410 IF MP<2 AND CP<2 THEN PRINT"WE HAVE
RUN OUT OF HAWKS AND DOVES!
":GOTO 460
420 IF CP<2 AND MP>2 THEN PRINT"WE HAVE
RUN OUT OF HAWKS!":GOTO 460
430 IF MP<2 AND CP>2 THEN PRINT"WE HAVE
RUN OUT OF DOVES!"
460 PRINT:PRINT"THE POPULATION OF HAWKS
AND DOVES"
465 PRINT"SURVIVED FOR ";DA;" MONTHS."
```

```
470 IF DA>HS THEN HS=DA
480 FOR X=1 TO 1000:NEXT X:PRINT:PRINT"T
HE LONGEST SO FAR IS ";HS
485 PRINT:PRINT
490 PRINT"PRESS 'Y' FOR ANOTHER CAGE, OR
 'N'     TO STOP."
500 GET A$
510 IF A$="Y" THEN GOTO 90
515 IF A$="N" THEN GOTO 520
517 GOTO 500
520 CLS
530 PLOT 16,13,"GOODBYE"
540 END
550 FOR X=1 TO 800:NEXT X
560 ZAP
570 FOR X=1 TO 1200:NEXT X
580 CLS
590 RETURN
```

3D CONE

Using this listing, you can form a very complex pattern which takes some time to generate. This program produces, as the title suggests, a cone shape in a three-dimensional image. I have added a burst of sound at the end so that if you cannot wait for it to fully run, then you can leave the computer and go and have a cup of tea. The beep will tell you when to come back.

Try and change the equation in line 7Ø or line 5Ø, or change the size of the step in the loop. I experimented with many values before I came up with this one, but it doesn't mean that there are not better ones—so go and have an experiment.

```
10 REM*****3 D CONE*****
20 HIRES:INK 7
25 PRINT:PRINT"     3 D  C O N E      ";CH
R$(96);" 1983"
30 FOR A=-100 TO 100
40 J=0:K=1:T=10
50 V=T*INT(SQR(10000-A*A)/T)
60 FOR B=V TO -V STEP -T
70 C=INT(80+30*SIN((SQR(A*A+B*B))/12)-0.
7*B)
80 IF C<J THEN 120
90 J=C
100 CURSET A+110,C-15,1
110 K=0
120 NEXT B:NEXT A
130 SOUND 1,200,8:WAIT 150:PLAY 0,0,0,0
```

NOUGHTS AND CROSSES

Everyone has heard and has played noughts and crosses so there is no need for me to explain how to play. Just simply enter the number of the position you would like to fill.

Noughts and crosses is one of the earlier computer games written around the time of the first 'Lunar Lander' and 'Hunt the Wumpus' games.

This program incorporates a small random feature which makes the computer's play at times a little unpredictable. I find this adds to the enjoyment of the game. This program can be beaten—though to write an unbeatable program would have been a lot easier! However, there is little fun generated from the futility of playing a machine knowing that it is impossible to beat.

The pattern displayed at the beginning of the game is generated from line 912 onwards.

Happy OXO'ing!

```
10 REM*****NOUGHTS AND CROSSES*****
20 GOSUB 912
30 CLS
35 PRINT:PRINT"NOUGHTS AND CROSSES"
37 PRINT:PRINT
40 N=0
50 PRINT"Press any key when"
60 PRINT"you're ready to play"
70 N=N+1:IF KEY$="" THEN 70
```

```
80 CLS:DIM A(9)
110 CLS:FOR N=1 TO 9:A(N)=0:NEXT N
120 IF RND(1)>.5 THEN PRINT"I'll have th
e First move":WAIT 200:CLS:
GOTO 170
130 GOSUB 750:GOSUB 480:GOSUB 670
170 GOSUB 750:GOSUB 480
180 IF A(5)=0 THEN A(5)=1:GOTO 130
190 REM*******ROW/BLOCKRELEASECOMPLETION
******
200 D=1
210 B=1
220 IF B=1 THEN X=1:Y=2:Z=3
230 IF B=2 THEN X=1:Y=4:Z=7
240 IF B=3 THEN X=1:Y=5:Z=9
250 IF B=4 THEN X=3:Z=7
260 C=1
270 IF A(X)=D AND A(Y)=D AND A(Z)=0 THEN
 A(Z)=1:GOTO 130
280 IF A(X)=D AND A(Y)=0 AND A(Z)=D THEN
 A(Y)=1:GOTO 130
290 IF A(X)=0 AND A(Y)=D AND A(Z)=D THEN
 A(X)=1:GOTO 130
300 IF B=1 THEN X=X+3:Y=Y+3:Z=Z+3
310 IF B=2 THEN X=X+1:Y=Y+1:Z=Z+1
320 IF C<3 THEN C=C+1:GOTO 270
330 IF B<4 THEN B=B+1:GOTO 230
340 IF D<2 THEN D=D+1:GOTO 210
```

```
350 REM******RANDOM MOVE******
360 B=1
370 D=INT(RND(1)*9)+1
380 IF A(C)=0 THEN A(C)=1:GOTO 130
390 B=B+1
400 IF B<21 THEN 370
410 B=0
420 B=B+1
430 IF A(B)=0 THEN A(B)=1:GOTO 130
440 IF B<9 THEN 420
450 GOSUB 750
460 PRINT:PRINT"It's a draw!"
470 GOTO 650
480 REM*****WIN CHECK*****
490 FOR B=1 TO 4
500 IF B=1 THEN X=1:Y=2:Z=3
510 IF B=2 THEN X=1:Y=4:Z=7
520 IF B=3 THEN X=1:Y=5:Z=9
530 IF B=4 THEN X=3:Z=7
540 FOR C=1 TO 3
550 IF A(X)=A(Y) THEN IF A(Y)=A(Z) THEN
IF A(X)<>0 THEN 610
560 IF B=1 THEN X=X+3:Y=Y+3:Z=Z+3
570 IF B=2 THEN X=X+1:Y=Y+1:Z=Z+1
580 NEXT C
590 NEXT B
600 RETURN
610 REM****WIN****
```

```
620 PRINT:PRINT
630 IF A(X)=1 THEN PRINT "I'm beat you h
uman!"
640 IF A(X)=2 THEN PRINT"Well done, you
beat me, human!"
650 WAIT 300
660 GOTO 110
670 REM****PLAYER MOVE****
680 PRINT:PRINT"Enter your move"
690 A$=KEY$
700 IF A$<"1" OR A$>"9" THEN 690
710 B=VAL (A$)
720 IF A(B)<>0 THEN 690
730 A(B)=2
740 RETURN
750 REM*****DISPLAY*****
760 CLS
770 PRINT:PRINT:PRINT:PRINT
780 PRINT"1 2 3    ";
790 FLAG=0
800 FOR B=1 TO 9
810 IF A(B)=0 THEN FLAG=1
820 IF A(B)=0 THEN PRINT" - ";
830 IF A(B)=1 THEN PRINT" O ";
840 IF A(B)=2 THEN PRINT" X ";
850 IF B=3 THEN PRINT:PRINT:PRINT"4 5 6
  ";
860 IF B=6 THEN PRINT:PRINT:PRINT"7 8 9
  ";
```

```
870 NEXT B
880 PRINT:PRINT
890 IF FLAG=0 THEN 460
895 R=INT(RND(1)*12)+1
900 MUSIC 1,3,R,7:WAIT 30:PLAY 0,0,0,0
910 RETURN
912 REM*******HIRES DISPLAY*******
915 HIRES
920 FOR M=3 TO 6 STEP 3
930 FOR T=1 TO 199 STEP M
935 CURSET T+10,0,1
940 DRAW 199-T,T,1
945 INK INT(RND(1)*7)+1
950 CURSET 209,T,1
960 DRAW -T,199-T,1
970 CURSET(199-T)+10,199,1
980 DRAW -(199-T),-T,1
990 CURSET 10,199-T,1
 1000 DRAW T,-(199-T),1
 1010 NEXT T
 1020 NEXT M
 1022 INK 4
 1025 PRINT"          NOUGHTS AND CROSSES"
 1027 PRINT"                 ";CHR$(96);" 198
3"
 1030 WAIT 300:PING:PING
 1035 TEXT
 1040 RETURN
```

RUSSIAN ROULETTE

Not much of a game when played seriously as many poor soldiers found to their cost when taken prisoner in Vietnam, but thankfully our version is harmless. You must enter a chamber number from one to six, this is the chamber which will be fired at you. If this number equals the computer's number, then the chamber contains a bullet when fired and you suffer. This small program is an ideal way to go through some of the more widely used BASIC commands and simple routines. Below is a line-by-line description of the program which should aid the less experienced programmer.

LINE	DESCRIPTION/ACTION
1Ø	A simple remark indicating the title of the program.
2Ø	R, a variable, is given the value zero. R stands for the number of rounds. At the beginning of the program, this should also be set to zero.
3Ø	The screen is cleared, the background colour set to red and the foreground set to white. Two blank lines are printed.
4Ø	One is added to the number of rounds, the number of round then being displayed on-screen.
45	S is assigned a random number between one and six.
5Ø	The line waits for the user to enter a number.
6Ø	This line checks to see that the number entered is between one and six. If not, a message is displayed and the computer goes back to line 5Ø.
7Ø	Tells the user to press a key and then waits for a single keypress.
8Ø	Waits/pauses for a short time then checks to see if the user's number is the same as the computer's chosen number. If it is, then the user has lost and the computer goes to line 13Ø where the 'lose' routine starts.
9Ø	Plays a deep sound for a short time. PLAY Ø,Ø,Ø,Ø switches all the sound channels off.

100 Displays the 'win' message.
110 "Another Go?" prompt. Also waits for the player's answer.
115 Stops the player from quitting after the first round.
120 The program cycles back to line 30 if the player wants another go. If anything apart from the 'Y' key is pressed, the program goes to the 'end' routine (lines 160 onwards).
130 Generates the gun sound, clears the screen and the background is set to black. Displayed near the centre of the screen is the 'lose' message.
140 This is a small loop which prints the number of rounds that the player survives, at an angle across the screen.
150 This line introduces a pause, then finishes the program.
160 Clears the screen, sets the background to pink (magenta) and the foreground to black.
170 This is a loop which prints the computer's reply to the player giving up. The figure '13' is added to the TAB command due to one of the Oric's bugs in its operating system. The number '13' must be added to any TAB value to get the proper column positioning, eg for column 28, the TAB value must be 41.
180 If the loop has reached halfway then this line makes the ZAP sound, one of the Oric's four pre-defined sounds.
185 The command which continues the loop.
190 A pause, the screen is again cleared and the computer goes back to line 140 to print the number of rounds played.

```
10 REM*****RUSSIAN ROULETTE*****
20 R=0
30 CLS:PAPER 1:INK 7:PRINT:PRINT
40 R=R+1:PRINT"THIS IS ROUND ";R:PRINT
45 S=INT(RND(1)*6)+1
50 INPUT"CHOOSE A CHAMBER (1-6)";C
60 IF C<1 OR C>6 THEN PRINT:PRINT"TRY AG
```

```
AIN, DUMMY":PRINT:ZAP:GOTO
50
70 PRINT:PRINT"PRESS TRIGGER (PRESS ANY
KEY)":GET A$
80 WAIT 60:IF C=S THEN 130
90 SOUND 1,2000,15:WAIT 10:PLAY 0,0,0,0
100 PRINT:PRINT"CLICK....YOU SURVIVED"
110 PRINT:PRINT"AGAIN? (Y/N)":GET A$:PRI
NT
115 IF R=1 AND A$<>"Y" THEN PRINT"YOU MU
ST TRY AGAIN!":WAIT 150:GOT
O 30
120 IF LEFT$(A$,1)="Y" THEN 30 ELSE GOTO
 160
130 SHOOT:CLS:PAPER 0:PLOT 10,10,"BANG..
.YOU ARE DEAD"
140 FOR T=1 TO 14:PRINT:NEXT:PRINT TAB(2
0)"YOU SURVIVED ";R-1;"ROUN
DS"
150 WAIT 120:END
160 CLS:PAPER 5:INK 0
170 FOR A=1 TO 24:PRINT TAB(13+A)"*CHICK
EN*"
180 IF A=12 THEN ZAP
185 NEXT A
190 WAIT 100:CLS:GOTO 140
```

CREATOR

This is your chance to design cell colonies in this semi-serious program. Credit for this program and for the many others which have been written for other computers must go to John Conway who in 1970, invented this simulation. It has been chopped and changed ever since and this program has been developed from a Microsoft BASIC program I wrote some time ago.

Despite the changes made, the basic rules remain the same. Each cell on the grid may have neighbours and if it has no neighbours, then it dies.
Every cell with two or three neighbours survives the next generation.
Every cell with exactly three neighbours gives birth to a new cell.
If a cell has more than three neighbours, it dies from overcrowding.

The program offers a pretty speedy way of setting up your cell colony structure. The co-ordinates of one space on the grid are displayed. If you wish to place a cell there, then you press the spacebar; if not, press any other key except for 'N'. When you have finished designing your colony, press 'N' and the life-cycle will start. As a new generation is ready to appear, a bell sounds and the new colony is printed.

You can get a lot of fun out of seeing how long you can design a colony to last.

The lines 17Ø to 24Ø check each space around the cell to check for neighbours. Lines 3ØØ to 4ØØ print every new generation and lines 5ØØ onwards contain the instructions and the starting routine.

```
10 REM*******CREATOR*********
20 G=1
25 PAPER 0:INK 2
30 CLS
40 DIM M(10,10)
50 DIM N(10,10)
80 GOSUB 500
120 GOSUB 300
130 G=G+1
140 FOR K=2 TO 9
150 FOR Z=2 TO 9
160 C=0
170 IF M(K-1,Z-1)=1 THEN C=C+1
180 IF M(K-1,Z)=1 THEN C=C+1
190 IF M(K-1,Z+1)=1 THEN C=C+1
200 IF M(K,Z-1)=1 THEN C=C+1
210 IF M(K,Z+1)=1 THEN C=C+1
220 IF M(K+1,Z-1)=1 THEN C=C+1
230 IF M(K+1,Z)=1 THEN C=C+1
240 IF M(K+1,Z+1)=1 THEN C=C+1
250 IF M(K,Z)=1 AND C<>3 AND C<>2 THEN N
(K,Z)=0
260 IF M(K,Z)=0 AND C=3 THEN N(K,Z)=1
270 NEXT Z
280 NEXT K
290 GOTO 120
300 PING
305 CLS
```

```
313 PRINT" CREATOR"
314 PRINT
318 PRINT"GENERATION: ";G:PRINT
320 FOR K=1 TO 10
330 PRINT TAB(10);
340 FOR Z=1 TO 10
350 M(K,Z)=N(K,Z)
360 IF M(K,Z)=1 THEN PRINT"O";
365 IF M(K,Z)=0 THEN PRINT" ";
370 NEXT Z
380 PRINT
390 NEXT K
400 RETURN
500 CLS
510 PRINT TAB(27)"CREATOR."
520 PRINT TAB(26)"---------"
530 PRINT:PRINT:PRINT"This is the life g
rid. You must place"
540 PRINT"cells in what you think will b
e the"
550 PRINT"most advantageous pattern. Pre
ss SPACE";
555 PRINT"when you want a cell left in t
hat"
560 PRINT"position. If you don't want a
cell "
562 PRINT"there, press any other key. Wh
en you"
```

```
564 PRINT"have finished, press 'N' to st
art."
570 PRINT:PRINT
575 PRINT"  23456789"
580 FOR H=1 TO 8
590 PRINT H+1;"........ ";H+1
600 NEXT H
605 PRINT
610 PRINT"  23456789"
620 FOR K=2 TO 9
630 FOR Z=2 TO 9
640 PLOT 6,25,"X=         Y=   "
645 PLOT 8,25,STR$(K)
647 PLOT 18,25,STR$(Z)
650 GET A$:IF A$=" " THEN M(K,Z)=1:PLOT
K+1,Z+12,"O"
655 IF A$="N" THEN RETURN
660 IF A$<>" " THEN M(K,Z)=0
665 N(K,Z)=M(K,Z)
670 NEXT Z
680 NEXT K
700 RETURN
```

LASER WAR

This action-packed game shows how a simple idea can be developed into a new program. I had originally intended to write a 'space invader' (yawn) program with continuously moving shields. On perfecting the shields and the laser base, I started to think about adding another shield further up the screen which would protect some non-moving objects. From there on, I knew that the program would never become 'space invaders' but something a lot more original.

The program places you in front of the Phoenix base, and your aim is to score as many points as possible by staying in the game for as long as you can manage. At the bottom of the screen (coloured magenta) is your laser base which you move left and right with the 'lesser than' and 'greater than' arrows. You must attempt to destroy, layer by layer, the multi-coloured defence shield at the top of the screen. Once you have blasted a hole through the wall, you will reap the benefit by sending another laser blast up. This extra shot will attack the heart of the Phoenix base and will earn you a 1,000 point bonus. However, that column will now become filled with anti-matter ('X's) and you will lose the game if you send up another blast to that column. Hitting the defence shield is not that easy, however, as there are the two shields at the top and bottom of the screen protecting the shield from any kind of onslaught. Hitting one of these shields takes a few points off your score. This is important to remember at the beginning of the game for if your score falls below zero, then you lose the game.

The final thing to mention are the cruise vessels which skip along the screen. These must be destroyed every now and then because if one of the cruise vessels reaches the same level as the shields, your game is ended abruptly. Destroying a cruise vessel, apart from gaining you some points, will put a new ship back in the top corner of the screen thus giving you all the extra time gained to continue blasting the shield away.

The shields move at a good speed, especially for a BASIC program, and this is due to lines 11Ø, 115, 111Ø and 112Ø. Instead of moving the shield one space at a time, it is moved four spaces in one go. Moving a single character like that would not give any impression of smoothness, but moving eight characters in line does not lose any smoothness, speeds the game up no end and makes it more difficult for the player to judge when to send up a laser bolt.

When the player fires a laser, the program goes to a separate routine which is actually bigger than the main loop. This loop, starting from line 1ØØØ and continuing to line 123Ø, performs nearly all of the same functions as the main loop but with the addition of subroutines to display the missile and detect what it has hit.

The row of '&'s in line 25 are the bricks which make up the defence shield. To cut down on some extra lines and therefore a slower game, the two shields use the same co-ordinates, the only difference being that 12 is taken off the vertical co-ordinate (F) for the top shield. Doing this saves variable memory, and more importantly cuts down on the number of screen detection lines needed in the 'fire' routine. These detection lines and the lines plotting characters onto the screen are the biggest culprits in slowing the program action down.

The sound is provided by the routine from lines 6ØØ to 65Ø. Every time another sound is used, ie ZAP, PING, etc, the program goes back to this routine so that the sound can be switched back on. (Using one of the Oric's pre-defined sounds also acts as a PLAY Ø,Ø,Ø,Ø command.)

Before playing the game, check that the keyclick sound is *off*. This can be done either by holding down the CTRL key and pressing 'F' or by typing in direct mode, PRINT CHR$ (6).

At the end of the game, the computer will give you your score and tell you the highest score. If you wish to have another game, then press the spacebar.

```
10 REM*******LASER WAR*******
15 GOSUB 9000:HI=0
17 CLS:S=4:PRINT"   LASER WAR  ";CHR$(96
);" 1983 ";
19 PAPER 0:INK 7:SC=0
20 PLOT 0,24,CHR$(5)
22 PLOT 0,25,"--------------------------
-----------"
24 PLOT 0,2,"---------------------------
----------"
25 FOR T=4 TO 7:PLOT 1,T,CHR$(T-3)+"&&&&
&&&&&&&&&&&&&&&&&&&&&&&&&&&
&&&&&"
26 NEXT T
30 A=18:B=24:X=2:Y=9:E=1:F=20
35 GOSUB 600
80 PLOT X,Y,"}0{":PLOT A,B,"#$%":PLOT E,
F,"/\/\/\/\":PLOT E,F-12,"\
/\/\/\/"
90 XX=X:YY=Y:AA=A:EE=E
100 X=X+1:IF X>38 THEN Y=Y+1:X=0
110 E=E+S:IF E>28 THEN S=-4
115 IF E<2 THEN S=4
120 Q$=KEY$
130 A=A+2*((Q$=",")-(Q$="."))
135 IF A>35 THEN A=35
137 IF A<1 THEN A=1
140 IF Q$="Q" THEN GOSUB 1000
```

```
160 IF Y=18 AND X>34 THEN 5000
170 IF SC<0 THEN 5000
340 PLOT XX,YY,"   ":PLOT AA,B,"   "
345 PLOT EE,F,"        ":PLOT EE,F-12,"
        "
350 GOTO 80
600 REM******SOUND*******
605 PLOT C,D+1," "
610 PLOT 30,0,STR$(SC)
615 SOUND 2,1000,0
620 PLAY 2,0,4,500
650 RETURN
800 HT=0
905 PLOT XX,YY,"   ":X=3:Y=9
910 FOR T=1 TO 15:INK 1:WAIT 5
820 INK 3:WAIT 5:INK 7:WAIT 5:NEXT T
830 PLOT 0,24,CHR$(5)
850 RETURN
1000 REM****FIRE****
1005 C=A+1:D=23
1010 PLOT XX,YY,"   ":PLOT EE,F,"
 "
1020 PLOT EE,F-12,"        "
1030 PLOT X,Y,"}O{":PLOT A,B,"#$%":PLOT
C,D,"!"
1032 PLOT E,F,"/\/\/\/\"
1035 PLOT E,F-12,"\/\/\/\/"
1040 XX=X:YY=Y:AA=A:DD=D:EE=E
```

```
1050 D=D-1:K=SCRN(C,D):IF K=38 THEN PLOT
 C,D," ":ZAP:SC=SC+39:GOSUB
600:RETURN
1055 IF K=47 OR K=92 THEN PLAY 1,7,3,350
:SC=SC-23:PLOT C,D," ":GOSU
B600:RETURN
1060 IF K=123 OR K=125 OR K=48 THEN HT=1
1070 IF HT=1 THEN SC=SC+107:EXPLODE:PLOT
C,D+1," ":GOSUB 800:GOSUB 6
00:RETURN
1072 IF K=98 THEN 5000
1075 IF D<4 THEN W=W+1:SC=SC+1000:FOR T=
4 TO 7:PLOT C,T,"X":NEXT T
1077 IF D<4 THEN PLOT 30,0,STR$(SC):RETU
RN
1080 Q$=KEY$:A=A+(Q$=",")-(Q$=".")
1090 A=A+(A=35)-(A=1)
1100 X=X+1:IF X>38 THEN Y=Y+1:X=0
1110 E=E+S:IF E>28 THEN S=-4
1120 IF E<2 THEN S=4
1125 IF B=20 THEN B=21
1130 IF Y=18 AND X>34 THEN 5000
1200 PLOT C,DD," ":PLOT AA,B,"    ":PLOT
XX,YY,"    "
1210 PLOT EE,F,"          ":PLOT EE,F-12,"
      "
1230 GOTO 1030
5000 REM******FAILURE******
```

```
5005 PLAY 0,0,0,0
5010 FOR T=1 TO 46:PRINT:NEXT T
5020 IF SC<0 THEN SC=0
5030 PRINT"         THE GAME IS OVER":PR
INT
5035 PRINT"         YOU SCORED ";SC
5040 FOR T=1 TO 12:PRINT:WAIT 10:NEXT T
5050 IF SC>HI THEN HI=SC
5060 PRINT"   H I G H   S C O R E   ";HI
5070 FOR T=1 TO 8:PRINT:NEXT T
5080 WAIT 100:GET A$:IF A$=" " THEN GOTO
 17
5090 GOTO 5080
9000 REM****** UDG's *******
9010 FOR P=46360 TO 46391
9020 READ N
9030 POKE P,N
9040 NEXT P
9050 DATA 0,0,0,7,15,31,63,63
9060 DATA 12,12,30,63,63,63,63,63
9070 DATA 0,0,0,56,60,62,63,63
9080 DATA 61,61,61,61,55,55,55,55
9090 RETURN
```

HIGHER OR LOWER?

This is a fairly straightforward card game in which you must bet a proportion of your cash on the turn of a card. You are given the simple choice of deciding whether the next card is higher or lower. However, if the next card is equal to the previous card, then you lose. If the card is as you predicted then you can decide to continue that round as the amount of money you receive for every extra go increases quite considerably. When you quit a round (providing you have not lost), the money is added onto your previous winnings and you continue. There is a minimum bet of £5 (this is a high-class establishment!) and all aces count as high.

Have a good time and gamble away, but remember that the casino closes early!

Much of the program is concerned with printing prompts and messages onto the screen. Lines 100 and 224 generate the random number which is converted into the card. In this game, the suit of the card is not important so it is not calculated. Lines 110 to 140 and lines 230 to 236 convert the higher codes into Jack, Queen, King and Ace. The routine from line 800 onwards comes into operation when it's closing time at the casino.

```
10 REM*****HIGHER OR LOWER?*****
20 REM*****YOUR OWN TITLE DISPLAY TO GO
IN HERE*****
70 PAPER 7:INK 4
75 P=1:D=2
80 M=20:G=0
100 R=INT(RND(1)*13)+2:C$="  "+STR$(R)
```

```
110 IF R=11 THEN C$=" JACK"
120 IF R=12 THEN C$=" QUEEN"
130 IF R=13 THEN C$=" KING"
140 IF R=14 THEN C$="N ACE"
150 CLS:PRINT:PRINT:PRINT TAB(27)"HIGH/L
OW"
155 PRINT TAB(27)"------------"
156 P=P+1
157 PRINT:PRINT"YOU NOW HAVE _";M:PRINT
158 IF P=10 THEN GOSUB 800
160 PRINT:PRINT"PRESS A KEY TO DEAL CARD
":GET A$:WAIT 50:PRINT:PRIN
T
170 B=0
200 PRINT"CARD IS A";C$:PRINT:PRINT
202 IF G>0 THEN 210
205 PRINT:PRINT
206 INPUT"BET (MIN _5)";B:PRINT
207 IF B<5 OR B>M THEN PRINT"THINK AGAIN
 BUDDY!":PRINT:GOTO 205
208 H=B
210 PRINT:PRINT"HIGHER OR LOWER? (H/L)"
220 GET H$:PRINT:PRINT
224 N=INT(RND(1)*13)+2
225 PRINT:PRINT"PRESS A KEY TO DEAL NEXT
 CARD":GET A$
227 D$=" "+STR$(N)
230 IF N=12 THEN D$=" QUEEN"
```

```
232 IF N=13 THEN D$=" KING"
234 IF N=14 THEN D$="N ACE"
236 IF N=11 THEN D$=" JACK"
238 PRINT:PRINT"NEXT CARD IS A":D$
240 IF H$="H" AND N>R THEN 500
250 IF H$="L" AND R>N THEN 500
270 PRINT:PRINT:PRINT"THEREFORE YOU LOSE
,"
280 SOUND 1.1500,14:WAIT 50:PLAY 0,0,0,0
285 M=M-H
290 WAIT 100
300 IF M<5 THEN PAPER 0:CLS:PLOT 12,12,"
YOU ARE BROKE!":ZAP:ZAP:END
305 G=0
310 GOTO 100
500 REM***VICTORY***
510 PRINT:PRINT"YOU WIN!"
520 SOUND 1,100,7
530 WAIT 50:PLAY 0,0,0,0
540 R=N:G=G+1:IF G=5 THEN G=0:B=B*3:GOTO
 600
550 B=B*2
560 PRINT"DO YOU WISH TO QUIT THIS ROUND
 (Y/N)":GET A$
570 IF A$="Y" THEN B=INT(B/D):M=M+B:G=0:
GOTO 100
580 PRINT:PRINT"BRAVE SOUL"
585 WAIT 100
```

```
590 GOTO 202
600 PRINT:PRINT"YOU COMPLETED A ROUND, W
ELL DONE!"
610 PRINT:PRINT"YOU ARE NOW _";B-H;" BET
TER OFF"
620 PRINT"AS A RESULT OF YOUR DARING."
630 M=M+B
640 WAIT 200:GOTO 100
800 WAIT 100:CLS
810 FOR T=1 TO 10:PRINT:NEXT T
820 PRINT" IT IS CLOSING TIME AT THE CAS
INO SIR"
825 PRINT
830 PRINT"BUT IF YOU ARE INTERESTED I KN
OW OF A "
840 PRINT"GAME DOWNTOWN WHICH IS ALWAYS
OPEN...":WAIT 700
850 PRINT:PRINT"HOWEVER, IT DOES COST _"
;INT(M/3);" TO GET IN"
860 PRINT:PRINT"AND THE ODDS ARE SLIGHTL
Y HIGHER":D=3
870 PRINT:PRINT"INTERESTED? (Y/N)"
880 INPUT A$:IF A$="Y" THEN 910
890 PRINT:PRINT"OK. YOU LEAVE WITH _";M
900 PRINT:PRINT"WELL DONE!":END
910 PRINT:PRINT"I'LL TAKE YOU THERE...":
WAIT 400:M=M-(M/3):GOTO 100
```

PRAIRIE FARMER

You're out in the mid-west area of the USA in this simulation game. The program contains all the prompts you need to play it, so I will not go into any great depth about the game action. As with most simulation programs, the best way to learn how to play properly is to run them again and again.

You must manage your farm for 10 years by planting acres of cotton, wheat and maize. You must decide how much to pay your workforce, how much of your land to plant and what to plant it with.

I will give you just a couple of hints: be careful how you treat the workforce—if you pay them too little they will desert your farm and you will have no one to help you plant the fields. Also, try to work out which is the most profitable crop and concentrate on that, *but* it is wise not to devote all your resources to just one crop as it could be a bad year for that crop and you will be left with nothing.

As I have said before, practice makes perfect—so have a few attempts before you can expect to succeed.

If you wish to change the length of the game, then change the numbers in lines 11Ø and 76Ø to the number of years you wish to simulate.

```
10 REM******PRAIRIE FARMER******

30 YR=1

40 MO=INT(RND(1)*1000)+7000

50 LA=INT(RND(1)*1000)+100

60 AC=INT(RND(1)*200)+300

70 CS=INT(RND(1)*5)+8:CX=CS
```

```
80 BA=0:WH=0:CO=0
90 GOSUB 170
100 GOSUB 290
110 IF YR=10 THEN 750
120 IF MO<1 THEN 790
130 IF LA<1 THEN 820
140 YR=YR+1
150 CS=CS+INT(12.5*CS/100)
160 GOTO 80
170 REM*******UPDATE INFORMATION******
180 GOSUB 870
190 PRINT"YOU HAVE $";MO;" IN YEAR ";YR
200 GOSUB 880
210 PRINT"YOU ARE EMPLOYING ";LA
220 PRINT"LABOURERS, WORKING FOR"
230 PRINT AC;" ACRES OF LAND"
240 GOSUB 880
250 PRINT"CROPS:- ":PRINT TAB(25) CO;" M
AIZE"
260 PRINT TAB(25) BA;" COTTON"
270 PRINT TAB(25) WH;" WHEAT"
280 RETURN
290 PRINT"IT WILL BE $";CS;" IN GENERAL
COSTS"
300 PRINT"TO WORK EACH ACRE....AND SO TH
E "
310 PRINT"THE MAXIMUM NUMBER OF ACRES YO
U CAN"
```

```
320 PRINT"WORK THIS YEAR IS ";
340 MX=INT(MO/CS):IF MX>AC THEN MX=AC
350 PRINT MX
360 PRINT"HOW MUCH LAND DO YOU WANT TO H
ARVEST"
370 INPUT L
380 IF L>MX THEN 370
390 MO=MO-L*CS
400 GOSUB 170
410 PRINT:PRINT"HOW MUCH WILL YOU PAY EA
CH WORKER"
420 INPUT W
430 IF W*LA>MO THEN 420
440 MO=MO-LA*W
450 GOSUB 170
460 P=10
470 PRINT"WHAT PROPORTION  (OUT OF 10) D
O"
480 PRINT"WISH TO CONCENTRATE ON MAIZE"
490 INPUT CP
500 IF CP>P THEN 490
510 P=P-CP
520 PRINT:PRINT"OF THE REMAINING ";P;" O
UT OF 10,"
530 PRINT"HOW MUCH WHEAT WILL YOU PLANT?
"
540 INPUT WP
550 IF WP>P THEN 540
```

```
560 P=P-WP
570 GOSUB 870
580 PRINT"STAND BY FOR A YEAR..."
590 WAIT 400
600 BA=INT(P*L*LA*W*3/100000)
610 CO=INT(CP*L*LA*W*2.7/17000)
620 WH=INT(WP*L*LA*W*1.4/9300)
630 T=BA+CO+WH
640 GOSUB 170
650 PRINT:PRINT T;" TONS WERE HARVESTED"
660 RT=INT((.5+8.7*BA+5.94*CO+2.2*WH)*(C
S-CX+1))
670 IF BA=0 AND CO=0 AND WH=0 THEN RT=0
680 WAIT 300
690 PRINT:PRINT"AND YOUR TOTAL RETURN"
700 PRINT"WAS $";RT
710 MO=MO+RT
720 LA=INT(LA-LA/(W+.01))
730 WAIT 250
740 RETURN
750 GOSUB 870
760 PRINT"YOU HAVE SURVIVED FOR 10 YEARS
."
770 PRINT:PRINT TAB(25)"CONGRATULATIONS!
"
780 END
790 GOSUB 870
```

```
800 PRINT"BEING A PRAIRIE FARMER IS AN EX
PENSIVE BUSINESS...YOU'RE B
ROKE!"
810 END
820 GOSUB 870
830 PRINT"YOU HAVE NO WORKERS AND HAVE B
EEN"
840 PRINT"FORCED TO SELL YOUR FARM AND G
O BACK  EAST."
860 END
870 CLS
880 PRINT:PRINT
890 RETURN
```

ART NOUVEAUX

This is your chance to become a famous artist with this handy program. The instructions are contained within the program and it's just a matter of typing it in and paint away!

Who knows, your masterpiece may be sold to the Tate Gallery —stranger things have happened, especially there!

```
10 REM*****ART NOUVEAUX*****
20 GOSUB 9000
30 A=19:B=12
60 CLS
70 PRINT:PRINT:PRINT
80 PRINT TAB(24)"ART NOUVEAUX"
90 PRINT TAB(24)"-------------"
100 PRINT:PRINT"      ARROW KEYS TO MOVE
"
110 PRINT:PRINT"         '<' FOR BLACK"
120 PRINT:PRINT"         '>' FOR ERASE"
130 PRINT:PRINT"         'E' TO DISPLAY"
135 PRINT:PRINT"AFTER PRESSING 'E', ENTE
R COLOR CODE"
140 PRINT:PRINT:PRINT
150 PRINT:PRINT"      PRESS ANY KEY TO S
TART"
160 GET R$
```

```
170 CLS
200 PLOT A,B,A$
250 Q$=KEY$
260 A=A+(Q$=CHR$(8))-(Q$=CHR$(9))
270 B=B+(Q$=CHR$(11))-(Q$=CHR$(10))
280 IF A<1 THEN A=1
290 IF A>38 THEN A=38
300 IF B<1 THEN B=1
310 IF B>25 THEN B=25
320 IF Q$="," THEN A$="#"
330 IF Q$="." THEN A$=" "
340 IF Q$="E" THEN 500
400 GOTO 200
500 GET R$:C=VAL(R$)
510 INK 0:WAIT 5
520 INK C:WAIT 5
530 GOTO 510
8000 CLS:PAPER 2:INK 0
8010 A=19:B=7
8020 N=0
8200 RETURN
9000 REM******* UDG's ********
9010 FOR P=46360 TO 46367
9020 READ N
9030 POKE P,N
9040 NEXT P
9050 DATA 63,63,63,63,63,63,63,63
9200 RETURN
```

SQUASH

A versatile program this, which can be converted into a whole host of other games. Squash is a pretty obvious game to understand. You simply have to keep the ball in play for as long as possible by deflecting the ball off your bat. The ball moves quite fast, so the game is quite a good test of your manual dexterity. You use the 'Z' and 'M' keys to control your bat.

At the beginning of the program, you will be asked which level you require, either level one or two. Level one is harder as the bat moves slower, so I suggest that you start off with level two first.

The possibilities for this program are quite varied. With some extra work, it could be converted into a 'Breakout' game where you must remove the bricks in the wall by deflecting the ball off your bat and into the wall. You can quite easily vary the program to include two bats (one for each player) or two balls, or a moving target to try to aim the ball at; as you can see, there are many options open to you.

A good way to improve your programming is to take a programme already written and try to improve/modify it. Why not have a try with this game and add one of the features that I have mentioned or indeed, incorporate a totally new idea. Use your imagination to the full—that is the very essence of games programming.

The 'ball to bat detection' routine is at lines 15Ø and 155 with the score increased by 17, the vertical direction of the ball changed and a PING sounded. There is a sound reason why I have increased the score by that figure instead of, say, one every time. Consider if you have been playing just one game for ten minutes in which you have performed well breaking your previous high score and finally you received a score of 41. Using my scoring system you would have 697 which *sounds* much more like a high score. Choosing an odd number such as '13' or '17' means that the score, when it changes, looks a little more interesting than say '2'...'4'...'6' ...'8', etc. This is just a small feature which is worth including in your own programs.

```
10 REM******SQUASH*******

15 CLS:GOSUB 9000

17 PLOT 14,6,"SQUASH!":PLOT 14,7,"======
="

20 PLOT 10,10,"KNOCK THE BALL":WAIT 200

25 PLOT 12,12,"OFF THE WALL":WAIT 200

27 PLOT 10,16,"USE Z FOR LEFT":PLOT 10,1
8,"& M FOR RIGHT"

28 PLOT 9,22,"WHICH LEVEL 1 OR 2"

29 INPUT LVL

48 PLOT 12,24,"PRESS A KEY":GET R$

50 GOSUB 1000

60 PLOT V,24,"  "

70 A$=KEY$

80 IF A$="" THEN 110

90 IF A$="Z" AND V>0 THEN V=V-LVL

100 IF A$="M" AND V<38 THEN V=V+LVL

110 PLOT V,24,CHR$(127)+CHR$(127)

120 PLOT X,Y," "

130 Y=Y+D:IF Y>23 OR Y<2 THEN SOUND 1,20
0,8:WAIT 15:D=-D:PLAY 0,0,0
,0

140 X=X+C:IF X>36 OR X<2 THEN SOUND 1,40
```

```
0,8:WAIT 15:C=-C:PLAY 0,0,0
,0
145 PLOT X,Y,"o"
150 IF X=V AND Y=23 THEN SC=SC+17:D=-D:P
ING
155 IF X=V+1 AND Y=23 THEN SC=SC+17:D=-D
:PING
160 IF Y=24 THEN GOSUB 500
170 PLOT18,26,CHR$(7)+"SCORE:"+STR$(SC)
180 GOTO 60
500 L=L-1:IF L=0 THEN CLS:PLOT 8,10,"END
 OF GAME: YOU SCORED "+STR$
(SC)
505 IF L=0 THEN PRINT CHR$(17);CHR$(6):E
ND
510 WAIT 50:PLOT 1,26,"LIVES :- "+STR$(L
)
515 PLOT 0,26,CHR$(7)
520 PLOT 1,24,"
           "
525 WAIT 250
530 X=INT(RND(1)*35)+2:Y=2
535 PLOT 1,26,"                    "
540 RETURN
1000 PRINT CHR$(6);CHR$(17)
1010 V=15:SC=0:C=1:D=1:L=3
1020 CLS:PAPER 4:INK 3
1030 PLOT 0,0,"##########################
###############"
```

```
1040 FOR Z=1 TO 24:PLOT 0,Z,"#":PLOT 38,
Z,"#"
1050 NEXT Z
1060 X=INT(RND(1)*35)+2:Y=2
1070 PLOT 0,25,"========================
==============="
1100 RETURN
9000 FOR P=46360 TO 46367
9010 READ N:POKE P,N:NEXT P
9020 DATA 63,63,63,63,63,63,63,63
9030 RETURN
```

ROM BUG

You have a nasty 'bug' in your 65Ø2 ROM. The only way to rid yourself of this menace is to POKE the address in which it is hiding. Your poor, damaged computer can only give an approximate indication of where the evil 'bug' lies. You have six chances to guess its position and the computer will comment on your guess telling you whether to aim higher or lower.

The program is based around the Hi-Res display created in lines 4Ø to 8Ø. The CHAR command is used to put the numbers and the Rom Bug, normally TEXT characters, onto the Hi-Res screen. If when writing your own programs you need to use CHAR and you have more than, say, four or five characters to display on-screen, then it would be wise to use a routine something like the one below.

```
10 REM*******CHAR ROUTINE********
20 HIRES
30 CURSET 10,10,0
40 FOR T=2 TO 12
50 READ N
60 CHAR N,0,1
70 CURSET 10*T,10,0
80 NEXT T
90 DATA 72,69,76,76,79,32,84,72,69,82,69
```

Type in the above routine to see what it displays. The loop is simply reading an ASCII code into N then putting the character on the screen. Finally, the cursor is moved along 1Ø pixels ready for the next character.

Getting back to the main program, lines 38Ø to 41Ø produce the Rom Bug's invasion.

```
10 REM****ROM BUG****
20 T=0:GOSUB 9000:INK 0
25 PAPER 3
30 R=INT(RND(1)*100)+1
40 HIRES:CURSET 80,70,1
50 DRAW 100,0,1:DRAW 0,50,1:DRAW -100,0,
1:DRAW 0,-50,1
60 CURSET 85,75,0:CHAR 35,0,1
70 CURSET 115,90,0:CHAR 54,0,1:CURSET 12
5,90,0:CHAR 53,0,1
80 CURSET 135,90,0:CHAR 48,0,1:CURSET 14
5,90,0:CHAR 50,0,1
200 PRINT"At what address is the Rom Bug
 hiding"
210 INPUT"HINT:Between 1 and 100";N
215 SOUND 1,200,7:WAIT 50:PLAY 0,0,0,0
220 IF N>R THEN PRINT"*****LOWER*****"
230 IF N<R THEN PRINT"*****HIGHER*****"
240 IF N=R THEN 280
250 T=T+1:IF T>5 THEN 350
260 GOTO 200
280 TEXT:CLS
290 FOR A=1 TO 3:FOR B=1 TO 7
300 PLOT 10,12,"You did it! You Poked"
310 PLOT 10,14,"the Rom Bug's address"
320 PAPER B:ZAP
325 WAIT 50:NEXT B
```

```
330 NEXT A:END
350 TEXT:CLS:PRINT:PRINT:PRINT
360 PRINT"He was hiding at :";R
370 PRINT:PRINT"and will now take over y
our machine."
375 WAIT 400:CLS
380 FOR A=1 TO 200
390 IF A/10=INT(A/10) THEN SHOOT
400 PLOT INT(RND(1)*38)+1,INT(RND(1)*25)
,"#"
405 PLOT INT(RND(1)*38)+1,INT(RND(1)*25)
,"#"
408 IF RND(1)>.9 THEN PAPER INT(RND(1)*7
)+1
410 NEXT A
420 CLS:PLOT 19,12,"#":END
9000 REM***UDG****
9010 FOR A=46360 TO 46367
9020 READ N
9030 POKE A,N
9040 NEXT A
9050 DATA 51,51,0,12,12,0,33,30
9100 RETURN
```

THE FORBIDDEN CAVES

Dare you enter the Forbidden Caves? You do? You must be very brave or very foolhardy!

The Forbidden Caves is a dark system, deep underground in which a fortune in gold can be found if you are brave enough to hunt for it. However, you must be careful not to fall into pools of quicksand, trip down deep pot-holes or bump into one of the strange creatures which live in the caves.

You have a random amount of time to wander around the caves and gain as much wealth as possible. The game ends when your time runs out, or sooner if you don't avoid the aforementioned hazards.

You begin your search in cave 55. The computer will then ask you for the direction you wish to move in. Enter 'N' for north, 'S' for south and so on. Occasionally, the computer will give you a look at the map of the caves. It will only be shown for a very short time, so concentrate on it. Your position is marked as an 'H'. I won't tell you what the other symbols stand for as you can work them out by playing the game.

You can also fire arrows. To do this enter 'F' and then the direction which you want the arrow to be fired in. If you kill a monster, you receive some extra game time to continue exploring.

The graphic display starts from line 950, while the lines 260 to 390 handle your direction input and decide what you are sharing your position with.

```
10 REM****THE FORBIDDEN CAVES****
20 DIM P(8)
40 GT=INT(RND(1)*11)+20
45 GOSUB 950:TEXT
50 CLS:PAPER INT(RND(1)*7)+1:INK 0
60 PRINT:PRINT"THE":PRINT"    FORBIDDEN"
:PRINT
62 PRINT"*****   *****   *   *   *****
 *****"
64 PRINT"*       *   *   *   *   *
 *"
66 PRINT"*       *****    * *    ****
 *****"
67 PRINT"*       *   *    * *    *
    *"
68 PRINT"*****   *   *     *     *****
 *****"
70 DIM A(100):H=0:Q=0:L=0:G=0:AR=6
72 PRINT:PRINT:PRINT"PRESS 'S' IF YOU DA
RE TO ENTER"
74 GET A$:IF A$<>"S" THEN 74
76 PAPER INT(RND(1)*7)+1
80 FOR B=1 TO 100
85 A(B)=46
90 IF B<12 OR B>90 OR 10*INT(B/10)=B OR
10*INT(B/10)=B-1 THEN A(B)=
166
100 NEXT
```

```
110 FOR B=1 TO 5:RESTORE:FOR D=1 TO 5
120 Z=INT(RND(1)*76)+12:IF A(Z)=166 THEN
 120
130 READ C:A(Z)=C
140 NEXT:NEXT
150 DATA 166,218,77,81,36
160 FOR B=1 TO 8:READ P(B):NEXT:DATA -11
,-10,-9,-1,1,9,10,11
170 E=55
180 A(E)=72
190 CLS:GOSUB 880
200 Q=INT(RND(1)*7)
210 IF Q=0 AND E<>55 THEN GOSUB 880
220 PRINT:PRINT:PRINT"CAVE ";E
230 IF G>0 THEN PRINT:PRINT TAB(27) G;"G
OLD"
240 GOSUB 610
250 GOSUB 880:PRINT:PRINT
260 INPUT"WHICH DIRECTION ";Z$:U=0
270 IF Z$="N" AND A(E-10)=166 THEN U=1
273 IF Z$="S" AND A(E-10)=166 THEN U=1
276 IF Z$="E" AND A(E+1)=166 THEN U=1
280 IF Z$="W" AND A(E-1)=166 THEN PRINT:
PRINT TAB(24)"BLOCKED CAVE"
:GOTO 250
290 A(E)=46:IF Z$="N" THEN E=E-10
295 IF E>100 OR E<11 THEN 1500
300 IF Z$="S" THEN E=E+10
```

```
310 IF Z$="E" THEN E=E+1
320 IF Z$="W" THEN E=E-1
330 IF Z$="F" THEN GOSUB 730
335 IF A(E)=166 THEN 1300
340 IF A(E)=218 THEN GOSUB 400
350 IF A(E)=77 THEN GOSUB 440
360 IF A(E)=81 THEN GOSUB 510
370 IF A(E)=36 THEN GOSUB 570
380 H=H+1:IF H>GT THEN 850
390 GOTO 200
400 PRINT:PRINT"MAGIC!!"
410 A(E)=46
420 E=INT(RND(1)*76)+12:IF A(E)=166 THEN
 420
430 RETURN
440 CLS:SOUND 1,150,7:WAIT 50:PLAY 0,0,0
,0
442 PRINT:PRINT:PRINT
445 PRINT TAB(25)" MONSTER HERE":PRINT T
AB(25)" ------------"
450 WAIT 400
460 IF RND(1)<.2 THEN PRINT"IT IS RUNNIN
G AWAY":RETURN
470 PRINT"IT HAS SEEN YOU...."
480 WAIT 400
490 IF RND(1)>.85 THEN PRINT TAB(30)"AND
 FLEES":WAIT 200:RETURN
495 PRINT TAB(33);
```

```
500 PRINT"AND EATS YOU!":SOUND 1,1500,7:
WAIT 60:PLAY 0,0,0,0:Q=9:CL
S:GOTO 850
510 CLS:PAPER 1:INK 7
515 FOR J=1 TO 24 STEP 2
517 PRINT TAB((13)+(J/2))"HORRORS...QUIC
KSAND"
519 SOUND 1,20+J*20,8
523 WAIT 2*J:NEXT J
540 SOUND 1,1000,8:WAIT 60:PLAY 0,0,0,0
545 WAIT 200:CLS:PAPER 2:INK 0
550 Q=9
560 GOTO 850
570 CLS:FOR J=4 TO 26
572 PRINT TAB(J+13) "WEALTH!!!"
574 SOUND 2,420-15*J,7:WAIT 30:NEXT J
576 PLAY 0,0,0,0:WAIT 100
580 K=INT(RND(1)*100)+100
590 PRINT:PRINT"YOU HAVE FOUND GOLD!!!"
600 PRINT:PRINT"IT IS WORTH _";K;"!!!":G
=G+K
605 SOUND 1,80,8:WAIT 40:PLAY 0,0,0,0
607 WAIT 300:CLS:PRINT:GOSUB 880:RETURN
610 Y=1
620 L=A(E+P(Y))
630 IF L<>46 THEN 660
640 IF Y<8 THEN Y=Y+1:GOTO 620
650 IF L=46 THEN RETURN
```

```
660 PRINT:PRINT:PRINT"NEARBY IS....";
670 IF L=166 THEN PRINT"NO PATH"
680 IF L=218 THEN PRINT"MAGIC"
690 IF L=77 THEN PRINT"MONSTER"
700 IF L=81 THEN PRINT"QUICKSAND"
710 IF L=36 THEN PRINT"GOLD"
720 WAIT 350:RETURN
730 AR=AR-1:IF AR=0 THEN PRINT:PRINT"NO
ARROWS LEFT":RETURN
740 PRINT:PRINT TAB(20) AR;" ARROWS LEFT
":SS=0
750 PRINT:PRINT:INPUT"WHICH DIRECTION ";
S$:CLS
760 IF S$="N" AND A(E-10)=77 THEN SS=1:Y
T=E-10
770 IF S$="S" AND A(E+10)=77 THEN SS=1:Y
T=E+10
780 IF S$="E" AND A(E+1)=77 THEN SS=1:YT
=E+1
785 IF S$="W" AND A(E-1)=77 THEN SS=1:YT
=E-1
790 IF SS=0 THEN PRINT"NOTHING THERE":GO
TO 840
800 PRINT"******** A HIT ********"
810 WAIT 150:IF RND(1)>.3 THEN 830
820 PRINT:PRINT"****MONSTER IS DEAD****"
:A(YT)=46:G=G+INT(RND(1)*10
0):GOTO 840
```

```
830 PRINT"THE MONSTER IS WOUNDED"
840 WAIT 200:RETURN
850 WAIT 100:IF Q=9 THEN 870
860 CLS:PAPER 0:INK 2:SOUND 1,1500,8:WAI
T 50:PLAY 0,0,0,0
865 PRINT"YOUR TORCH HAS BURNT OUT":PRIN
T
866 PRINT"IN DARKNESS, YOUR CHANCE OF SU
RVIVAL  WAS VERY SMALL"
867 PRINT:PRINT TAB(30);
868 PRINT:PRINT"R";:SOUND 1,1000,8:WAIT
50:PRINT"I";:SOUND 1,1500,8
:WAIT 40
869 PRINT"P":SOUND 1,2000,10:WAIT 80:PLA
Y 0,0,0,0
870 WAIT 120:PRINT:PRINT"YOU SURVIVED FO
R ";H;" HOURS AND FOUND"
874 PRINT"_";G;"OF GOLD":SOUND 1,800,7:W
AIT 50:PLAY 0,0,0,0
876 END
880 A(E)=72:IF RND(1)<.66 THEN 940
890 FOR J=1 TO 100
900 PRINT CHR$(A(J));
910 IF 10*INT(J/10)=J THEN PRINT
920 NEXT:IF Q=9 THEN END
930 WAIT INT(RND(1)*100)+20
935 CLS
940 RETURN
```

```
950 REM****GRAPHIC DISPLAY*****
960 HIRES:CURSET 120,100,0:PATTERN 170
970 FOR K=1 TO 95
980 CIRCLE K,1
990 H=H+1:IF INT(10*H)/10=H THEN INK INT
(RND(1)*7)+1
1000 NEXT K
1020 SOUND 1,150,8:WAIT 50:PLAY 0,0,0,0
1030 WAIT 100
1040 RETURN
1300 WAIT 150:CLS:PAPER 0:INK 3
1310 PRINT:PRINT"YOU FOOLISHLY ATTEMPTED
 TO WALK WHERE THERE WAS NO
 PATH..."
1320 WAIT 100:PRINT:PRINT TAB(25)"YOU BL
EW IT!"
1330 PRINT:PRINT:PRINT
1340 WAIT 150:GOTO 867
1500 REM******POT HOLES*****
1510 WAIT 100:PRINT:PRINT:PRINT
1520 PRINT"YOU FELL INTO A POT HOLE, THE
RE IS NO CHANCE OF RESCUE..
."
1530 PRINT:PRINT:PRINT:PRINT
1540 GOTO 868
```

LOLLIPOP NIM

After the rigours of The Forbidden Caves, here is a friendly, relaxing game. You and the computer are dividing between you a stock of red lollipops. You each take as many as you like every turn, providing that the number you take is within the displayed maximum. Your aim is to leave the computer with the last one—for whoever takes the final sweet must pay for the lot!

This game, a version of the famous 'Matchsticks' puzzle is enlivened by the graphic lollies which appear. The computer's thinking is performed in line 18Ø. It is quite possible to produce a NIM program which is unbeatable but naturally the program just becomes a demonstration and ceases to be a game. To make the game, any game, entertaining there should always be some random element so that the game is not exactly the same every time you play. In Lollipop Nim, there are three such random elements: firstly, in line 6Ø, a random number of lollipops are produced; a random number, being the maximum number of lollipops which can be taken, is produced in line 8Ø; and finally, the computer, when considering how many to take, has its invincibility destroyed by a random element in line 18Ø.

The use of the random element, produced by the pseudo-random RND command, is a popular and almost vital part of BASIC games programming, adding life and interest into what might otherwise be a drab program.

```
 10 REM****LOLLIPOP NIM****
 15 GOSUB 900
 20 CLS:PAPER 7:INK 0
 30 J=20
 50 E=0
```

```
60 Z=INT(RND(1)*13)+15
70 IF 2*(Z/2)=Z THEN Z=Z+1
80 H=2+INT(RND(1)*2)+1
90 PRINT:PRINT TAB(24)"LOLLIPOP NIM":P
RINT TAB(24)"------------":PR
INT:PRINT
93 PRINT:PRINT"MAXIMUM YOU CAN TAKE IS
 ";H
95 PRINT:PRINT
100 IF E>0 THEN PRINT"YOU TOOK ";E;TAB
(28);"I TOOK ";Q
105 WAIT 150
110 GOSUB 400
120 PRINT:PRINT"HOW MANY WILL YOU TAKE
?"
130 INPUT E
140 IF E>H OR E<1 THEN PRINT:PRINT TAB
(26)"IMBECILE!":PRINT:GOTO 12
0
150 CLS
160 Z=Z-E
170 IF Z<1 THEN 250
180 Q=Z-1-INT((Z-1)/(H+1))*(H+1)+INT((
RND(1)*4)+1)-1
190 IF Q>Z OR Q=0 OR Q>H THEN 180
200 Z=Z-Q
210 IF Z<1 THEN 230
220 GOTO 90
230 FOR T=1 TO 20
232 PRINT"*****I TOOK ";Q;", SO YOU WI
N!******"
234 MUSIC 1,4,INT(T/2)+1,8
236 WAIT 40:NEXT T
238 WAIT 50:PLAY 0,0,0,0:GOTO 500
250 PLOT 1,12,">>>>>>>>>>>>>>I WIN!<<<<
<<<<<<<<<<<<<<"
255 MUSIC 1,0,1,8:WAIT 130:PLAY 0,0,0,
0
260 GOTO 500
400 REM****GRAPHICS****
405 J=20
410 PRINT:PRINT
```

```
420 FOR L=1 TO Z
425 K=2*L:IF K>38 THEN K=K-37:J=22
430 PLOT K,J,"#":PLOT K,J+1,"¦"
432 PLOT 0,20,CHR$(1):PLOT 0,22,CHR$(1
)
435 MUSIC 1,3,INT(RND(1)*12)+1,7
440 NEXT L
445 PLOT 14,25,STR$(Z)+CHR$(1)+" LOLLI
POPS"
460 WAIT 80
470 PLAY 0,0,0,0:WAIT 50
480 RETURN
500 REM*****ANOTHER GAME*****
510 WAIT 150:CLS
520 FOR T=1 TO 10:PRINT:NEXT T
530 PRINT"# WOULD YOU LIKE ANOTHER GO?
 #"
540 PRINT"¦
 ¦"
550 GET A$:IF A$="Y" THEN RUN
560 PRINT:PRINT"   O.K.  G O O D B Y E
"
570 PRINT:END
900 REM***** UDG's *****
910 FOR P=46360 TO 46367
920 READ N
930 POKE P,N
940 NEXT P
950 DATA 0,30,63,63,63,63,63,30
960 RETURN
```

JOGGER

My favourite program and certainly one of the highlights of the book, this is a frogger-type game for your Oric. This sort of game without machine code is usually unbearably slow and boring but here by using a few short routines, we have managed to write a game which was fast enough while offering plenty of features for maximum enjoyment. I say 'we' because this program was the joint work of myself and Scott Vincent. Scott is an experienced programmer from Ashford, Middlesex, who has much work published, including an arcade game for a large commercial software house. He developed the string scrolling routines essential to this game.

The aim of this game is to get your little 'jogging character' home. You come across a giant road with many lanes of cars, vans, trucks and trains. You must cross the road without coming to grief—not an easy task. You use the left and right cursor arrows for lateral movement and the up arrow key for forward movement—in this game there is no turning back. The keys are not repeating so they have to be tapped continuously. After a couple of games, you'll get the hang of it. The central reservation in the middle of the screen is the one moment of peace and safety you'll have in the game. You'll quickly have to leave your sanctuary to cross the upper, more difficult, half of the screen to get to the other side. If you succeed in getting a jogger home, then you receive a bonus 1,ØØØ points.

You have three lives which are displayed graphically on the screen. If you score over 5,ØØØ then you receive a bonus life. When you have run out of lives, you are told your score and the high score. You will then be asked if you wish to have another jog. If you do, press 'Y' and then press the Return key.

Every time you are hit, the screen will turn a variety of colours and an explosion will sound. To continue with the game, you must press the spacebar.

As you will see from the game, this program makes extensive use of the Oric's colour and graphics facilities. The cars, lorries, etc, are created using User-Defined Graphics, the routine for this starting at line 9ØØØ. The various shapes created are then combined together in different ways to form a whole host of vehicles to trap you.

The vehicles are held in the strings from line 11Ø to 19Ø and are scrolled across the screen both left and right using string-slicing techniques. Unlike many 'frogger' games, our game does not have logs to jump on over the last part of the course. If you want to, you could convert the whole course to one of trees and timber and change the runner to a lumberjack. Of course, then the game would have to be called 'Logger'!

```
10 REM********JOGGER*********
15 HI=0:DIM C$(7)
17 GOSUB 9000
20 PAPER 0:INK 7:CLS
25 GOSUB 3000
30 PRINT CHR$(17)
35 PRINT CHR$(6)
40 X=1:CR=0:QQ=61:LV=3
45 SC=0
47 RESTORE
110 C$(1)=" de  de   gf    de   gf   gf 
  de  gf"
120 C$(0)=" gfffffffffc  gfffffffffc gff
fffffffc"
```

```
130 C$(3)="   gh   gf    de     gh    de
    gf  "
140 C$(2)="  bbbbbc     fc   abc    de  
abc     "
150 C$(5)="ghhh  de   de  de  gf  gf  de
  gh gh "
160 C$(4)="  abc   de   de   bbbbbc   ac
  ac  ac"
170 C$(7)="de   gh    gff   de   de   gh
   gf   "
180 C$(6)="  de  bbbbc    abc   ac   de 
  abc   "
190 A$="============================
======"
195 H=600
200 PLOT 1,24,A$:PLOT 1,1,A$
210 A$="****************************
******"
212 PLOT 0,5,CHR$(2):PLOT 0,7,CHR$(1)
214 PLOT 0,3,CHR$(4):PLOT 0,9,CHR$(6)
216 PLOT 0,16,CHR$(2):PLOT 0,18,CHR$(4)
218 PLOT 0,20,CHR$(5):PLOT 0,22,CHR$(3)
220 FOR N=11 TO 14
230 PLOT 1,N,A$
240 NEXT N
250 PLOT 10,13," ! JOGGER ! LIVES:
"
255 FOR PF=1 TO LV
```

```
260 PLOT 26+(2*PF),13,"$"
265 NEXT PF
267 A=19:B=24
269 SOUND 1,2000,0:PLAY 1,1,4,2000
270 PLOT A,B,"$"
272 IF A<>19 OR B<24 THEN PLOT 19,24,"="
275 PLAY 2,0,1,13
280 FOR N=0 TO 1
290 D$=C$(N*2):E$=C$(N*2+1)
300 J=N*4+3:K=N*4+5
310 GOSUB 6000
320 NEXT N
321 FOR N=0 TO 1
322 D$=C$(N*2+4):E$=C$(N*2+5)
323 J=N*4+16:K=N*4+18
325 GOSUB 6000
327 NEXT N
328 PLAY 1,0,1,15
330 X=X+1
340 IF X=39 THEN X=1
350 FOR N=97 TO 102
360 IF SCRN(A,B)=N THEN CR=1
370 NEXT N
380 IF CR=1 THEN 7000
390 AA=A:BB=B:Q$=KEY$
400 A=A+(Q$=CHR$(8))-(Q$=CHR$(9))
410 B=B+(Q$=CHR$(11))*2
412 A=A+(A=39)-(A=0)
```

```
415 IF B=12 THEN B=11:SC=SC+INT(RND(1)*5
0)+50
418 IF B=1 THEN 8000
419 H=H-1
420 Q=SCRN(A,B)
430 IF Q=32 OR Q=61 OR Q=36 OR Q=42 THEN
 465
460 GOTO 7000
465 PLOT AA,BB,CHR$(QQ)
468 QQ=SCRN(A,B)
470 PLOT A,B,"$"
480 GOTO 270
3000 REM****TITLE PAGE****
3010 FOR A=1 TO 25:PRINT:NEXT
3050 PRINT"****  ****  ****  ****  ****
 ****"
3060 PRINT"  *   *  *  *     *     *
 *  *"
3070 PRINT"  *   *  *  *     *     ****
 ****"
3080 PRINT"  *   *  *  * **  * **  *
 * *"
3090 PRINT"***   ****  ***   ***   ****
 *  *"
3100 FOR A=1 TO 8:PRINT:WAIT 10:NEXT
3110 PRINT"     "+CHR$(96)+" GIFFORD/VIN
CENT 1983"
3120 FOR A=1 TO 10:PRINT:WAIT 10:NEXT
```

```
3122 FOR T=2 TO 6:PLOT 0,T,CHR$(T)
3124 NEXT
3130 FOR A=1 TO 38:PLOT A,22,"*":NEXTA
3135 PLOT 0,22,CHR$(2)
3140 FOR A=3 TO 38:PLOT A,21,"$":PLOT A-
3,21,"ac":SOUND 1,15*A,8:WA
IT 10
3150 PLOT A,21," ":PLOT A-3,21,"  ":NEXT
3155 WAIT 50
3160 ZAP:ZAP:ZAP
3170 WAIT 100:CLS
3300 RETURN
6000 Y=39-X
6010 PLOT X,J,MID$(D$,1,Y)
6020 PLOT 1,J,MID$(D$,Y+1,X)
6030 PLOT 1,K,MID$(E$,X,Y)
6040 PLOT Y,K,MID$(E$,1,X)
6050 RETURN
7000 PLOT AA,BB,CHR$(QQ)
7005 PLOT 12,11,"PRESS SPACE BAR"
7010 EXPLODE
7015 FOR TT=1 TO 2
7020 FOR T=1 TO 5
7021 INK T
7022 PLOT A-1,B,CHR$(7)
7023 PLOT A+1,B,CHR$(T)
7025 PLOT A,B,"$":WAIT 15
7030 PLOT A,B,"+":WAIT 15
```

```
7031 Q$=KEY$
7035 PLOT A,B,"$":WAIT 15
7040 PLOT A,B,"X":WAIT 15
7060 NEXT:NEXT
7065 CR=0
7067 SC=SC+((25-B)*10)
7068 Q$=KEY$:IF Q$<>" " THEN 7068
7070 LV=LV-1:IF LV=0 THEN 7500
7080 INK 7
7150 GOTO 210
7500 PLAY 0,0,0,0
7505 WAIT 50:CLS
7510 FOR T=1 TO 24:PRINT:NEXT T
7520 PRINT"             YOUR JOG IS OVER"
7530 FOR P=1 TO 23
7540 PRINT:WAIT 10:NEXT P
7550 PLAY 0,0,0,0
7570 PLOT 12,10,"YOU SCORED "+STR$(SC)
7580 GOTO 9500
8000 PLAY 0,0,0,0:WAIT 50:CLS
8005 Q$="L"
8010 PLOT 14,13,"WELL DONE!"
8015 INK 7
8017 SC=SC+H
8018 PING
8020 SC=SC+1000
8030 LV=LV+1
8050 PLOT  6,16,"YOU SCORE A BONUS 1000
```

```
POINTS!"
8060 IF SC>5000 THEN LV=LV+2:PLOT 10,18,
CHR$(5)+"AND AN EXTRA LIFE!
"
8070 PLOT 12,20,"SCORE SO FAR :"+STR$(SC
)
8080 PLOT 12,22,"PRESS 'Q' TO START"
8090 GET A$:IF A$<>"Q" THEN 8090
8100 GOTO 190
9000 REM******UDG*******
9010 FOR P=46856 TO 46919
9020 READ N
9030 POKE P,N
9040 NEXT P
9100 DATA 0,63,63,63,63,63,24,24
9110 DATA 0,62,62,62,63,63,24,24
9120 DATA 56,52,50,49,63,63,12,12
9130 DATA 0,3,7,56,63,63,8,8
9140 DATA 0,48,56,38,63,63,4,4
9150 DATA 63,63,45,45,63,63,6,6
9160 DATA 7,9,17,63,63,63,12,12
9170 DATA 0,0,63,63,63,63,6,6
9200 FOR P=46368 TO 46375
9210 READ N
9220 POKE P,N
9230 NEXT P
9240 DATA 28,28,8,62,8,20,20,20
9300 FOR P=46416 TO 46423
9310 READ N
```

```
9320 POKE P,N
9330 NEXT P
9350 DATA 63,63,63,63,63,63,63,63
9400 RETURN
9500 WAIT 300:CLS
9510 IF SC>HI THEN GOSUB 9800
9520 FOR T=1 TO 10:PRINT:NEXT T
9525 HI=SC:PRINT:PRINT TAB(26)"HIGH SCOR
E: "+STR$(HI)
9527 PRINT:PRINT:PRINT TAB(21)" ";
9530 INPUT"FANCY ANOTHER RUN";C$
9540 IF C$="Y" THEN INK 7:CLS:GOTO 40
9550 END
9800 PRINT:PRINT:PRINT:PRINT
9805 PRINT TAB(18)"THAT IS A NEW HIGH SC
ORE!"
9820 RETURN
```

FOUR-BY-FOUR

In this game, Four-by-four, the aim is to get four of your pieces (the 'H's) in a line in any direction, before the computer (using 'C's) manages to do so.

You indicate your choice of move by specifying the column in which you want to move your piece. The piece then drops to the lowest available position within that column. Any vertical, horizontal or diagonal line will count as a win.

The computer plays a reasonable game, although sometimes it is too attacking and may let you in the 'back door' to create four in a row. Despite the large number of calculations, the computer plays quite fast with only a short pause between moves. The computer, very generously, gives you the first move every game and then bases its move on your decision. Priority is given to stopping you getting four in a row and trying itself to get four pieces in line. If neither is possible, the computer checks the board to see if it can possibly create a row of three for itself or stop you doing the same. If this is not possible, then the computer looks at the board and calculates the most favourable single move.

Naturally, you will be able to beat the computer at this game, but using my family and friends as an example, the score between them and the computer lies, at the time of writing, at 21 to 15 to the computer's advantage so you can see that it will provide you with some opposition. As mentioned earlier, the computer's game is essentially an attacking one which makes the whole program more entertaining. You should receive many hours of fun playing this game.

One point to note is that some of the program lines, due to the Oric's limitations, are 77 and 78 characters long. Take great care when typing these lines in, some of which have no spaces, as one wrong character will stop the program running correctly, if at all. Once you have typed the program in, save it straight away, as an insurance against losing the program and with it, hours of hard work.

```
10 REM****FOUR-BY-FOUR****

15 GOSUB 1500

20 GOSUB 1090:REM INITIALISE

30 GOSUB 860:REM PRINT BOARD

40 GOSUB 680:REM WITH CHECK

50 GOSUB 980:REM HUMAN MOVE

60 GOSUB 860:REM PRINT BOARD

70 GOSUB 680:REM WIN CHECK

80 GOSUB 110:REM COMPUTER MOVE

90 GOTO 30

100 REM ************

110 REM COMPUTER MOVE

120 PRINT:PRINT "STAND BY FOR MY MOVE...
"

125 PING

130 B=10

140 B=B+1

150 IF A(B)= -9 THEN 180

160 IF A(B)= C THEN X=C: GOTO 210

170 IF A(B)= H THEN X=H: GOTO 210

180 IF B<77 THEN 140

190 GOTO 480

200 REM ************************

210 REM FOUR IN A ROW DANGER/CHANCE?

220 REM ACROSS
```

```
230 IF A(B+1)=X AND A(B+2)=X AND A(B+3)=
E AND A(B+13)<>E THEN MOVE=
B+3:GOTO650
240 IF A(B-1)=X AND A(B-2)=X AND A(B-3)=
E AND A(B+7)<>E THEN MOVE=B
-3:GOTO 650
250 IF A(B+1)=X AND A(B+2)=X AND A(B-1)=
E AND A(B+9)<>E THEN MOVE=B
-1:GOTO 650
260 IF A(B-1)=X AND A(B+2)=X AND A(B+1)=
E AND A(B+11)<>E THEN MOVE=
B+1:GOTO 650
270 IF A(B+1)=X AND A(B-1)=X AND A(B+2)=
E AND A(B+12)<>E THEN MOVE=
B+2:GOTO 650
280 IF A(B+1)=X AND A(B-1)=X AND A(B-2)=
E AND A(B+8)<>E THEN MOVE=B-2:GOTO 650
290 IF A(B-1)=X AND A(B-2)=X AND A(B+1)=
E AND A(B+11)<>E THEN MOVE=B+1:GOTO 650
300 REM DOWN
310 IF B>20THENIFA(B-10)=XANDA(B-20)=XAN
DA(B+10)=EANDA(B+20)<>E THEN SDR=1
315 IF SDR=1 THEN MOVE=B+10:GOTO 650
320 REM DIAGONALS
330 IF A(B+11)=X AND A(B+22)=X AND A(B-1
1)=EANDA(B-1)<>E THEN MOVE=B-11:GOTO 650
340 IFA(B+9)=XANDA(B+18)=XANDA(B-9)=EAND
A(B+1)<>ETHENMOVE=B-9:GOTO 650
```

```
350 REM ****************
360 REM MAKE/BLOCK THREE?
370 REM ACROSS
380 IFA(B+1)=XANDA(B+2)=EANDA(B+12)<>E T
HEN MOVE=B+2:GOTO 650
390 IFA(B+1)=XANDA(B-1)=EAND A(B+9)<>E T
HEN MOVE=B-1:GOTO 650
400 IF A(B-1)=XANDA(B-2)=E AND A(B+8)<>E
 THEN MOVE=B-2:GOTO 650
410 REM VERTICAL
420 IF A(B-10)=X AND A(B-9)=E AND A(B)<>
E THEN MOVE=B-10:GOTO 650
430 REM DIAGONAL
440 IF A(B+9)=X AND A(B-9)=E AND A(B+1)<
>E THEN MOVE=B-10:GOTO 650
450 IF B>11THENIFA(B+11)=XANDA(B-11)=EAN
DA(B-1)<>THEN MOVE=B-11:GOTO 650
460 GOTO 180
470 REM ************
480 REM SINGLE MOVES
490 FOR N= 1 TO 3
500 M(N)= 0
510 NEXT N
520 COUNT = 0
530 FOR B = 11 TO 77
540 IF A(B)<>C AND A(B)<>H THEN 600
550 IF A(B+1)=E AND A(B+11)<>ETHENCOUNT=
COUNT+1:M(COUNT)=B+1
```

```
560 IF A(B-1)=E AND A(B+9)<>E THEN COUNT
=COUNT+1:M(COUNT)=B-1
570 IF A(B-10)=E AND A(B)<>E THEN COUNT=
COUNT+1:M(COUNT)=B-10
580 IF A(B-11)=E AND A(B-1)<>E THEN COUN
T=COUNT+1:M(COUNT)=B-11
590 IF A(B-9)=E AND A(B+1)<>E THEN COUNT
=COUNT+1:M(COUNT)=B-9
600 NEXT B
610 IF COUNT <> 0 THEN 640
620 PRINT:PRINT "I THINK WE SHOULD CALL
IT A DRAW"
630 PRINT:PRINT:PRINT:END
640 MOVE=M(INT(RND(1)*COUNT)+1)
650 A(MOVE)=C
655 SDR=0
660 RETURN
670 REM *********
680 REM WIN CHECK
690 X=H
700 B=10
710 B=B+1
720 IF A(B)<>X THEN 770
730 IF A(B+1)=X AND A(B+2)=X AND A(B+3)=
X THEN 800
740 IF B>30 THEN IF A(B-10)=X AND A(B-20
)=X AND A(B-30)=X THEN 800
```

```
750 IF B>33 THEN IF A(B-11)=X AND A(B-22
)=X AND A(B-33)=X THEN 800
760 IF B>27 THEN IF A(B-9)=X AND A(B-18)
=X AND A(B-27)=X THEN 800
770 IF B<77 THEN 710
780 IF X=H THEN X=C:GOTO 700
790 RETURN
800 REM***WIN FOUND***
810 PRINT:PRINT
820 IF X=H THEN PRINT"YOU'VE BEATEN ME,
HUMAN!"
830 IF X=C THEN PRINT"I'VE DEFEATED YOU
HUMAN!"
835 SOUND 1,400,8:WAIT 120:PLAY 0,0,0,0
840 PRINT:PRINT:PRINT:END
850 REM ***********
860 REM PRINT BOARD
870 CLS:PRINT:PRINT
872 PAPER 2:INK 0
875 PRINT:PRINT"*************FOUR-BY-FOU
R************"
880 FOR K=10 TO 70 STEP 10
885 PRINT
890 PRINT TAB(18);
900 FOR J=1 TO 7
910 PRINT CHR$(A(K+J));" ";
920 NEXT J
930 NEXT K
```

```
935 PRINT:PRINT
940 PRINT TAB(18);"1 2 3 4 5 6 7"
950 PRINT:PRINT
960 RETURN
970 REM **********
980 REM HUMAN MOVE
990 PRINT"  YOUR MOVE...":PRINT
1000 PRINT"WHICH COLUMN DO YOU WISH TO"
1010 INPUT"MOVE INTO ";J
1020 Z=J
1030 Z=Z+10
1040 IF A(Z+10)=E THEN 1030
1050 IF A(Z)=E THEN A(Z)=H:RETURN
1060 PRINT "YOU CANT MOVE THERE"
1070 GOTO 1000
1080 REM **********
1090 REM INITIALISE
1100 CLS
1130 DIM A(109),M(30),P(6)
1140 E=ASC(".")
1150 H=ASC("H"):C=ASC("C")
1160 FOR B= 1 TO 109
1170 A(B)=E
1180 D=B-10*INT(B/10)
1190 IF D=0 OR D >7 OR B <11 OR B>77 THE
N A(B)= -9
1200 NEXT B
1210 RETURN
```

```
1500 HIRES
1510 CURSET 40,35,0:CIRCLE 25,1
1520 CURSET 85,80,0:CIRCLE 25,1
1530 CURSET 130,125,0:CIRCLE 25,1
1540 CURSET 175,170,0:CIRCLE 25,1
1550 PRINT"          FOUR-BY-FOUR"
1560 PRINT"          ";CHR$(96);" 1983"
1580 WAIT 300:TEXT:RETURN
```

Here is a sample game below, where the computer got the better of me.

```
*************FOUR-BY-FOUR*************


          . . . . . . .
          . . . . . . .
          . . . . . . .
          . . . . . . .
          . . . . . . .
          . . . . . . .
          . . . . . . .
          1 2 3 4 5 6 7

  YOUR MOVE...
 WHICH COLUMN DO YOU WISH TO
 MOVE INTO

*************FOUR-BY-FOUR*************


          . . . . . . .
          . . . . . . .
          . . . . . . .
          . . . . . . .
          . . . . . . .
          . . . . . . .
          . . . . H . .
          1 2 3 4 5 6 7
```

```
*************FOUR-BY-FOUR*************

     . . . . . . .
     . . . . . . .
     . . . . . . .
     . . . . . . .
     . . . . . . .
     . . . . C . .
     . . . . H . .
     1 2 3 4 5 6 7

 YOUR MOVE...
WHICH COLUMN DO YOU WISH TO
MOVE INTO

*************FOUR-BY-FOUR*************

     . . . . . . .
     . . . . . . .
     . . . . . . .
     . . . . . . .
     . . . . . . .
     . . . . C . .
     . . . H H . .
     1 2 3 4 5 6 7

*************FOUR-BY-FOUR*************

     . . . . . . .
     . . . . . . .
     . . . . . . .
     . . . . . . .
     . . . . . . .
     . . . . C . .
     . . . H H C .
     1 2 3 4 5 6 7

 YOUR MOVE...
WHICH COLUMN DO YOU WISH TO
MOVE INTO
```

At this point, I thought I might have an easy win with the computer not noticing my three in a row...

```
*************FOUR-BY-FOUR*************

        . . . . . . .
        . . . . . . .
        . . . . . . .
        . . . . . . .
        . . . . . . .
        . . . . C . .
        . . H H H C .
        1 2 3 4 5 6 7
```

No such luck, a simple block by the computer spoils my chances.

```
*************FOUR-BY-FOUR*************

        . . . . . . .
        . . . . . . .
        . . . . . . .
        . . . . . . .
        . . . . . . .
        . . . . C . .
        . C H H H C .
        1 2 3 4 5 6 7

 YOUR MOVE...
WHICH COLUMN DO YOU WISH TO
MOVE INTO
```

```
*************FOUR-BY-FOUR*************

      . . . . . . .
      . . . . . . .
      . . . . . . .
      . . . . . . .
      . . . . . . .
      . . . H C . .
      . C H H H C .
      1 2 3 4 5 6 7


*************FOUR-BY-FOUR*************

      . . . . . . .
      . . . . . . .
      . . . . . . .
      . . . . . . .
      . . . . C . .
      . . . H C . .
      . C H H H C .
      1 2 3 4 5 6 7

 YOUR MOVE...
WHICH COLUMN DO YOU WISH TO
MOVE INTO
```

And here is the latter portion of the game.

```
*************FOUR-BY-FOUR*************

      . . . . . . .
      . . . . C . .
      . . . . H . .
      . . . C C . .
      . H C H C . .
      . C H H C . .
      H C H H H C .
      1 2 3 4 5 6 7

 YOUR MOVE...
WHICH COLUMN DO YOU WISH TO
MOVE INTO
```

```
*************FOUR-BY-FOUR*************

      . . . . . . .
      . . . . C . .
      . . . . H . .
      . . . C C . .
      . H C H C . .
      H C H H C . .
      H C H H H C .
      1 2 3 4 5 6 7


*************FOUR-BY-FOUR*************

      . . . . . . .
      . . . . C . .
      . . . . H . .
      . . C C C . .
      . H C H C . .
      H C H H C . .
      H C H H H C .
      1 2 3 4 5 6 7

 YOUR MOVE...
WHICH COLUMN DO YOU WISH TO
MOVE INTO

*************FOUR-BY-FOUR*************

      . . . . . . .
      . . . . C . .
      . . . . H . .
      . H C C C . .
      . H C H C . .
      H C H H C . .
      H C H H H C .
      1 2 3 4 5 6 7
```

```
*************FOUR-BY-FOUR*************

      . . . . . . .
      . . . . C . .
      . . C . H . .
      . H C C C . .
      . H C H C . .
      H C H H C . .
      H C H H H C .
      1 2 3 4 5 6 7

 YOUR MOVE...
WHICH COLUMN DO YOU WISH TO
MOVE INTO

*************FOUR-BY-FOUR*************

      . . . . . . .
      . . H . C . .
      . . C . H . .
      . H C C C . .
      . H C H C . .
      H C H H C . .
      H C H H H C .
      1 2 3 4 5 6 7


*************FOUR-BY-FOUR*************

      . . . . . . .
      . . H . C . .
      . . C . H . .
      . H C C C . .
      . H C H C . .
      H C H H C C .
      H C H H H C .
      1 2 3 4 5 6 7


I'VE DEFEATED YOU,HUMAN!
```

TIME-WARP

This listing uses the DRAW command to create the effect of a 'time warp'. A series of lines are drawn out from the centre of the screen to random points. The overall effect is very effective and interesting. Every now and then, the INK colour is changed to increase the effect.

```
10 REM*****TIME-WARP******
20 HIRES
25 PRINT:PRINT TAB(23)"TIME-WARP ";CHR$(
96);" 1983"
30 C=INT(RND(1)*7)+1
40 A=INT(RND(1)*240)-120
50 B=INT(RND(1)*198)-99
60 CURSET 120,100,1
70 DRAW A,B,1
80 IF RND(1)>.8 THEN INK C
90 WAIT 20
100 GOTO 30
```

BLACKJACK

This is my computer version of the famous card game, which also masquerades under the names of 'Twenty-One' and 'Pontoon'. You must attempt to get as close as possible to a total of 21 without exceeding it. The computer plays aces high unless this would lead to you 'busting', in which case the ace is worth one.

Both you and the computer are dealt one card each. You then have the option of 'sticking' on that one card or trying to get closer to the target of 21 with extra cards. Once you have 'stuck', the computer, as dealer, decides what it is going to do.

The game is very 'user-friendly', a term denoting how easy and straightforward a game is to play.

The game kicks off with a simple picture on the Hi-Res screen and a lovely rendition of a famous piece of gambling music which is played from lines 8ØØØ onwards.

```
10 REM****BLACKJACK****
20 CLS:PAPER 2:INK 0
30 HIRES:CURSET 160,60,0
40 DRAW -60,0,1:DRAW 0,40,1:DRAW 60,0,1:
DRAW 0,40,1
50 DRAW -60,0,1:CURMOV 20,30,0:DRAW 0,-1
40,1:DRAW 20,0,0:DRAW 0,140
,1
55 GOSUB 8000
60 WAIT 100:TEXT:CLS
```

```
70 CLS:R=R+1:IF R/8=INT(R/8) THEN GOSUB
8000
72 PRINT:PRINT"}}}}}}}}}}}}}}BLACKJACK{{
{{{{{{{{{{{{{"
73 PRINT:PRINT"We are into game ";R:PRIN
T
74 PRINT:PRINT"[[[[[[[[[[[[[[[[[[[[[[[[[
[[[[[[[[[[[[["
76 H=0:Z=0:GOSUB 800
80 PRINT:PRINT"Press 'X' for your first
card."
90 GET Z$:IF Z$<>"X" THEN 90
92 CLS
94 B=INT(RND(1)*13)+2
95 B$=STR$(B)
100 IF B=11 THEN B$="J"
110 IF B=12 THEN B$="Q"
120 IF B=13 THEN B$="K"
130 IF B>9 AND B<14 THEN B=10
140 IF B=14 THEN B=11:B$="A"
150 IF B=11 AND H+B>21 THEN B=1
160 PRINT:PRINT TAB(25) B$
170 H=H+B:PRINT:PRINT"Your total is ";H
180 PRINT:PRINT"Enter 'A' for a new card
 or press"
190 PRINT" 'S' to stand.":PRINT:PRINT:IN
PUT T$
```

```
200 CLS:IF T$="A" THEN 94
240 IF H=21 THEN PRINT:PRINT " BLACKJACK
!"
250 IF H>21 THEN PRINT:PRINT"BUST":GOTO
400
260 PRINT:PRINT"OK, you stand on ";H
270 PRINT
280 PRINT:PRINT"And my total was ";Z
290 PRINT:PRINT"I will now deal."
300 GOSUB 800
310 IF Z>21 THEN PRINT:PRINT TAB(30)"BUS
T"
320 IF Z=21 THEN PRINT:PRINT TAB(30)"BLA
CKJACK!"
330 IF Z<18 AND H<22 THEN GOTO 300
340 WAIT 300:CLS
350 PRINT:PRINT"You stood on ";H:PRINT
360 IF Z<22 THEN PRINT"I stand on ";Z:PR
INT
370 IF H=21 THEN PRINT"You have BLACKJAC
K!"
380 IF H=21 AND Z=21 THEN PRINT:PRINT"An
d so have I!"
390 IF Z=21 AND H<>21 THEN PRINT:PRINT"I
 have BLACKJACK!"
400 IF (Z>H OR H>21) AND Z<22 THEN PRINT
:PRINT"I WIN":SOUND 1,1000,
9:WAIT 90
```

```
410 IF (H>Z OR Z>21) AND H<22 THEN PRINT
:PRINT"YOU WIN":SOUND 1,100
,8:WAIT 100
420 IF H<22 AND H=Z THEN PRINT:PRINT"It'
s a draw."
430 IF H>21 AND Z>21 THEN PRINT"We both
BUST"
440 WAIT 500:GOTO 70
800 REM****************************
810 A=INT(RND(1)*13)+2
820 FOR N=1 TO 1000:NEXT N
830 IF A>9 AND A<14 THEN A=10
840 IF A=14 THEN A=11
850 IF A=11 AND Z+A>21 THEN A=1
860 Z=Z+A:PRINT:PRINT"My total is ";Z
870 RETURN
8000 REM*****ENTERTAINER******
8010 O=1
8040 FOR N=1 TO 76
8050 IF N=19 OR N=37 THEN RESTORE
8060 IF N=27 OR N=55 THEN FOR D=1 TO 20:
READ DN:NEXT
8070 READ A,B
8080 IF A>12 THEN O=O+1:A=A-12
8090 IF A<1 THEN O=O-1:A=A-(2*A)
8100 MUSIC 1,O,A,0
8102 MUSIC 2,O+1,A,0
8103 MUSIC 3,O-1,A,0
```

```
8105 PLAY 7,0,1,20000
8110 WAIT B-2
8120 NEXT N
8130 PLAY 0,0,0,0
8200 DATA 8,15,9,15,10,15,18,25,-10,15,1
8,25,-10,15,18,45
8210 DATA 6,15,8,15,9,15,10,15,6,15,8,15
,10,25,5,15,8,25,6,70
8220 DATA 3,15,1,15,-12,15,15,15,6,15,10
,28,8,15,5,15,3,15,8,55
8230 DATA 6,15,8,15,9,15,10,15,6,15,8,15
,10,25,6,15,8,25
8240 DATA 10,15,6,15,8,15,10,25,6,15,8,2
5
8250 DATA 10,15,6,15,8,15,10,25,5,15,8,2
5,6,100
8300 RETURN
```

MISSION APOLLO

This is an unashamedly traditional version of the classic computer game, 'Lunar Lander'. The program follows similar lines to the usual 'Lunar Lander' programs displaying height, velocity and amount of fuel left. A small spacecraft provides a graphic display of your situation.

As by now you must have guessed, the object of the game is to land your craft on the lunar surface as gently as possible. Depending on your success as captain of the mission, you are given a galactic rating. Even once you've managed to land the spaceship, there is still the incentive of obtaining a higher score.

If at some time during the game, you are too generous with the thrust and find yourself going upwards, you can enter 'Ø' as many times as necessary to start bringing your ship down again.

If you would like a more difficult game then change the specified program lines to the following;

```
240 B=B+A+(T-(ABS(A/4)-3)
250 A=A+(T-(ABS(A/4)-3)
270 IF ABS(B)<1Ø AND ABS(A)<6 THEN GOTO 45Ø.
```

If you are feeling very suicidal, you can also change the lines determining the fuel, height and velocity starting values.

```
10 REM****** MISSION APOLLO ******

20 HS=-5000:GOSUB 900

25 SC=0

30 PAPER 0:INK 7
```

```
40 A=-20-INT(RND(1)*60)
50 B=1200+INT(RND(1)*380)
60 C=320+INT(RND(1)*90)
70 CLS
80 PRINT"             MISSION  APOLLO":P
RINT:PRINT
90 B=INT(B):A=INT(A):C=INT(C)
100 PRINT"HEIGHT: ";B,,"VELOCITY: ";A
110 PRINT,,"FUEL: ";C
120 FOR Q=1 TO16-B/100
130 PRINT:PLOT 0,6+Q,CHR$(4):PLOT 0,5+Q,
CHR$(4)
140 NEXT Q
145 R=23+RND(1)*3-INT(RND(1)*3)
150 PRINT TAB(R+1)"&":PRINT TAB(R)"#$%"
160 FOR Q=16-B/100 TO 16
170 PRINT
180 NEXT Q
190 WAIT 150:PRINT"@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@"
200 PRINT:PRINT
202 INPUT"   THRUST";T
205 IF SGN(T)=-1 THEN 202
210 WAIT 100
220 IF T>C THEN T=0
230 C=C-T
240 B=B+A+(T-(ABS(A/6))-2)
250 A=A+(T-(ABS(A/6))-2)
```

```
260 IF C<1 AND B>100 THEN 300
270 IF ABS(B)<15 AND ABS(A)<10 THEN GOTO
 450
280 IF B>14 THEN 70
290 IF C>1 THEN 310
300 PLOT 0,14,CHR$(5)+"YOU HAVE RUN OUT
OF FUEL"
310 EXPLODE:FOR T=1 TO 7:PLOT 0,19,CHR$(
T):PLOT 0,20,CHR$(T)
312 PLOT 0,21,CHR$(T):PLOT 0,22,CHR$(T)
314 WAIT 20:NEXT T:WAIT 100
316 FOR P=1 TO 13:PRINT:NEXT P:PRINT"YOU
R SHIP CRASHED AT ";ABS(A);
" KPH"
320 PRINT:PRINT",CREATING A ";INT(ABS(A)
*45);" METRE DEEP CRATER"
325 FOR T=1 TO 12:PRINT:NEXT T
330 SC=SC-100-ABS(A)
340 WAIT 350
350 FOR T=1 TO 6:PRINT:NEXT T
360 PRINT"YOUR GALATIC RATING IS ";SC
365 PRINT:PRINT:PRINT
370 IF SC>HS THEN HS=SC
380 PRINT:PRINT"BEST RATING SO FAR IS ";
HS
390 PRINT:PRINT"PLEASE STAND BY FOR YOUR
 NEXT MISSION"
400 WAIT 800:GOTO 40
```

```
450 REM***** SUCCESS *****
460 PLOT 4,10,"WELL DONE INTREPID CAPTAI
N"
470 PLOT 3,13,CHR$(1)+"YOU LANDED YOUR S
HIP SAFELY"
480 PLOT 3,16,CHR$(6)+"YOU COMPLETED YOU
R MISSION"
490 SC=C*234:GOSUB 920:GOTO 340
900 REM**** UDG's ****
901 FOR P=46360 TO 46391
902 READ N
903 POKE P,N
904 NEXT P
905 DATA 3,7,15,31,6,6,6,15
906 DATA 63,63,63,63,12,30,0,0
907 DATA 48,56,60,62,24,24,24,60
908 DATA 12,30,63,33,33,63,33,33
910 RETURN
920 FOR J=1 TO 2
930 FOR T=1 TO 10
940 READ A,B
950 MUSIC 1,3,A,0:PLAY 1,0,1,1000
960 WAIT B:NEXT T
970 NEXT J
975 MUSIC 1,4,1,0:PLAY 1,0,1,1000
980 WAIT 60:PLAY 0,0,0,0
990 RETURN
```

```
1000 DATA 1,30,3,30,5,60,3,30,5,30,6,60,
5,30,6,30,8,15,8,100
1010 DATA 6,30,8,30,10,60,8,30,10,30,12,
60,10,30,12,30,12,15,12,15
```

SCRAMBLER

No, not *the* arcade game, but a great anagram solving game which pitches you against your computer and demonstrates the power of BASIC string handling. The computer chooses a word from its fair-sized dictionary/wordbase and then jumbles it up. All you have to do is guess the word. If you are right, the program displays a nice variation on the classic Moire pattern. You are severely reprimanded if you are unfortunate enough to get it wrong.

I decided not to incorporate a time limit feature, but you can if you wish to by adding a timer loop and using KEY$ instead of INPUT. Each letter entered by the player would have to be stored in an array and then each letter of the input array checked with each letter of the computer's word.

The 'word scrambling' routine can be twisted round the other way to aid anyone who regularly tries to solve the anagrams which appear in some of the national and local newspapers. If you have a printer, then you could modify the listing further to get it to print out all the possible combinations of letters.

Anyone seriously interested in this would add some form of discriminating feature which would stop combinations such as 'ZQ' and 'FD' being printed as they cannot help form a genuine word.

Scrambler is split up into five separate, distinct blocks, the main loop (lines 1Ø to 21Ø), the 'win' routine (lines 23Ø to 36Ø), the 'lose' routine (lines 4ØØ to 45Ø), the 'another go' routine (lines 5ØØ to 54Ø) and the 'choose word' routine (lines 9ØØØ to 922Ø). If it is possible to split your own programs up into blocks easily then I recommend that you do so. It is much easier to come back to a program after several weeks absence and modify it if you can see and understand the separate blocks which comprise the whole program. I am not saying that your one program should be lots of little bits and pieces connected loosely, but a sensible thought-out approach compensates for many things.

```
10 REM****SCRAMBLER****
20 GOSUB 9000:F=0
30 CLS:INK 7:PAPER 4
40 FOR M=1 TO 12:PRINT TAB(27)"SCRAMBLER
":PRINT
50 MUSIC 1,3,M,7
60 WAIT 40:NEXT M:PLAY 0,0,0,0
70 FOR A=1 TO 24:PRINT:WAIT 10:NEXT A
80 PRINT:PRINT:PRINT"Unscramble this wor
d: ";O$
90 PRINT:PRINT"Start now...":PRINT:PRINT
110 INPUT K$
170 PING:PRINT:PRINT
180 PRINT"I'll check your answer..."
190 WAIT 100:IF LEN(K$)<6 THEN 400
210 IF K$=W$ THEN 230 ELSE GOTO 400
230 REM****WIN ROUTINE****
240 HIRES
250 FOR B=0 TO 239 STEP 5
260 CURSET 0,199,3
270 DRAW B,-199,1
280 CURSET 239,199,3
290 DRAW -B,-199,1:NEXT
300 RESTORE
```

```
310 FOR A=1 TO 7:READ B
320 CURSET 80+(10*A),170,3
330 CHAR B,0,1
340 NEXT A
350 SOUND 1,100,7:WAIT100:PLAY 0,0,0,0
355 WAIT 300
360 GOTO 500
400 REM****LOSE ROUTINE****
410 CLS:PAPER 0:INK 6
420 PLOT 0,4,"You had plenty of time to
guess right"
430 PLOT 0,6,", but you failed miserably
!"
440 PLOT 4,9,"The word was "+W$
450 SOUND 1,1500,8:WAIT 100:PLAY 0,0,0,0
500 REM****ANOTHER GO****
505 WAIT 100
510 TEXT:CLS:PAPER 2:INK 0
520 PLOT 10,10,"Another word?"
530 GET A$:IF A$<>"Y" THEN PLOT 12,16,"G
OODBYE":END
540 RUN
9000 REM****CHOOSE WORD****
9010 FOR K=1 TO INT(RND(1)*60)+8
9020 READ W$
9030 NEXT K
9040 F=LEN(W$)
9050 DIM A(F)
```

```
9055 FOR T=1 TO F
9060 A(T)=ASC(MID$(W$,T,1))
9065 NEXT T
9070 FOR T=1 TO 40
9075 X=INT(RND(1)*F)+1:Y=INT(RND(1)*F)+1
9080 B=A(X):C=A(Y):A(X)=C:A(Y)=B
9083 NEXT T:O$=""
9087 FOR T=1 TO F
9090 O$=O$+CHR$(A(T)):NEXT T
9095 RETURN
9100 DATA 89,79,85,32,87,79,78
9110 DATA "PERPENDICULAR","VAMPIRE","ANG
ULAR","DIAMOND","UMBRELLA",
"SHREDDING"
9120 DATA "DENIAL","ESTABLISHMENT","MISU
NDERSTANDING","FUTURISTIC",
"GROTESQUE"
9130 DATA "PSYCHOLOGIST","ALGORHYTHM","C
HRYSANTHAMUM","LOGISTICS","
POSITIVE"
9140 DATA "EXAMINATION","TRAVELLER","BAT
CHELOR","DEPRESSION","PASTU
RISED"
9150 DATA "ORGANISATION","SANCTUARY","MA
GNETISM","TACHOGRAPH","JUDI
CIARY"
```

```
9160 DATA "ROUNDABOUT","SYNTHETIC","MEDA
LLION","ALCOHOLIC","CHIROPO
DIST"
9170 DATA "HARLEQUIN","SOCIALIST","CONSE
RVATIVE","HARMONIES","CULIN
ARY"
9180 DATA "PRETENTIOUS","CIVILISATION","
WRINKLED","GARGOYLE","ANTIB
IOTIC"
9190 DATA "TEMPESTUOUS","ARCHITECTURE","
GENETICS","ZOOLOGIST","TOTA
LITARIAN"
9200 DATA "BRUSQUE","INTELLIGENCE","UTOP
IAN","LUDICROUS","VIVACIOUS
","BULBOUS"
9210 DATA "MANDATORY","SPECTRAL","OINTME
NT","FLOUNDER","HYDRAULICS"
9220 DATA "RADIATOR","PREPARATION","TRAN
SITION"
```

IN SEARCH OF THE UNICORN

Your quest to find the last of the Unicorns takes you to a large cave just outside Timbuctoo. In this cave you hope to find the mythical beast that has been hiding for thousands of years. You have been lucky, on your journey down to Timbuctoo, as you were accosted by an elderly witchdoctor who gave you his enchanted rod. This rod will give you an approximate indication of where the unicorn is. As you enter the cave... horrors... miles of twisting tunnels appear before you. Remember, this is not one cave but a series of caves, so your rod will come in handy.

Your computer will display the number of steps that you have taken and will tell you what lies in the direction of the four main compass headings, together with the reading from the enchanted rod. You can either move 'N', 'S', 'E', 'W' or ask for help. Help comes in the form of a quick glance at the map of the whole cave but costs 15 steps.

The speed at which you find the Unicorn is, needless to say, the ultimate aim. When the map is displayed, you are shown as a '+' and the walls as solid blocks. The unicorn's position is not displayed—that would be too easy. To find the unicorn you must learn to interpret the reading from the enchanted rod. I am not going to tell you how—I've probably told you too much already. So off you go into the maze... in search of the Unicorn.

The cave system generated by lines 69Ø to 85Ø is not identical each time you play the game, but certain features dictated by the DATA statements from line 86Ø onwards are the same. The routine from lines 641 to 649 displays the picture of the cave.

```
10 REM****IN SEARCH OF THE UNICORN****
12 INK 7
20 GOSUB 640
25 GOSUB 2000
30 GOSUB 470
40 REM******OPEN/CLOSE*******
50 M=M+1:CLS:PRINT:PRINT
70 PRINT"STEP NUMBER ";M:PRINT
80 PRINT"NORTH: ";
100 IF A(D+1,E)=S THEN PRINT"OPEN" ELSE
PRINT"WALL"
120 PRINT:PRINT"SOUTH: ";
130 IF A(D-1,E)=S THEN PRINT"OPEN" ELSE
PRINT "WALL"
150 PRINT:PRINT"EAST: ";
160 IF A(D,E+1)=S THEN PRINT"OPEN" ELSE
PRINT"WALL"
180 PRINT:PRINT"WEST: ";
190 IF A(D,E-1)=S THEN PRINT"OPEN" ELSE
PRINT "WALL"
200 PRINT:PRINT
210 PRINT"Your enchanted rod reads ";
230 PRINT 100*ABS(Z-D)+10*ABS(Y-E)
240 REM*******MOVEMENT********
250 PRINT:PRINT:PRINT
260 PRINT"Which direction do you want to
 move?"
```

```
270 PRINT:PRINT" N  (north)  S  (south)
 E  (east)    "
280 PRINT"    W  (west)        H  (help)
"
300 INPUT A$
310 IF A$="N" AND A(D+1,E)=X THEN 300
320 IF A$="S" AND A(D-1,E)=X THEN 300
330 IF A$="E" AND A(D,E+1)=X THEN 300
340 IF A$="W" AND A(D,E-1)=X THEN 300
350 IF A$="H" THEN GOSUB 470
360 IF A$="N" THEN D=D+1
370 IF A$="S" THEN D=D-1
380 IF A$="E" THEN E=E+1
390 IF A$="W" THEN E=E-1
400 IF Z=D AND Y=E THEN 430
410 GOTO 50
420 REM********** SUCCESS **********
430 PRINT:PRINT"You have reached your go
al....":WAIT 350
435 PRINT:PRINT"You have just found the
last surviving"
436 INK 1
437 PRINT:FOR T=1 TO 10:PRINT TAB((13)+2
*T)"UNICORN!"
439 NEXT T
443 PRINT:PRINT"It took you just ";M;" d
ays. "
445 MUSIC 1,4,8,7:WAIT 50:MUSIC 1,4,12,7
```

```
:WAIT 70
460 END
470 REM********** MAP DISPLAY *********
490 CLS:PRINT
510 PRINT"North":PRINT
520 FOR B=15 TO 1 STEP-1
530 FOR C=1 TO 15
540 IF A(B,C)=X THEN PRINT"#";
550 IF B=D AND C=E THEN PRINT"+";:GOTO 5
70
560 IF A(B,C)=S THEN PRINT" ";
570 NEXT C:PRINT:NEXT B
575 PRINT
580 PRINT "South"
590 M=M+15
600 WAIT 350
610 CLS:A(D,E)=S
620 RETURN
630 REM********INITIALISATION***********
640 CLS
641 PAPER 7:INK 4
642 HIRES:CURSET 50,199,1:DRAW 40,-140,1
:DRAW 100,-32,1:DRAW 40,30,
1
645 DRAW 5,140,1:DRAW -100,0,0:DRAW 35,-
75,1:DRAW 40,-10,1:DRAW 15,
85,1
647 CURSET 90,59,0:DRAW 80,20,1:DRAW 53,
-23,1
```

```
648 PRINT"A view of the cave from outsid
e just  before you enter to
 find";
649 PRINT" the UNICORN":WAIT 100
650 FOR P=1 TO 2
652 FOR T=2 TO 30
655 CURSET 40,40,1
660 CIRCLE T,1
665 NEXT:NEXT P
680 CLS
690 DIM A(15,15)
700 B=INT(RND(1)*3)+1
710 Z=14:Y=14
720 IF B=2 THEN Y=2
730 IF B=3 THEN Z=2
740 X=1:S=2
750 FOR B=1 TO 15:FOR C=1 TO 15
760 A(B,C)=X:IF RND(1)>.8 THEN A(B,C)=S
770 IF C<2 OR C>14 OR B<2 OR B>14 THEN A
(B,C)=X
780 NEXT C:NEXT B
790 D=2:E=2
800 FOR F=1 TO 68
810 READ B:READ C
820 A(B,C)=2
830 NEXT F
840 M=-15
842 TEXT
```

```
844 PAPER 2:INK 0
850 RETURN
860 REM*******DATA BLOCK*********
870 DATA 2,2,2,3,2,4,2,5,2,6,2,7
880 DATA 3,7,4,7,5,7,5,6,5,5,5,4,5,3,6,3
890 DATA 7,3,7,4,7,5,7,6,7,7,7,8,7,9,9,8
900 DATA 9,9,10,8,10,7,10,6,10,5,10,4,8,
8
910 DATA 10,3,11,3,12,3,13,3,14,3,14,2,7
,10
920 DATA 6,10,5,10,4,10,3,10,2,10,2,11,2
,10
930 DATA 2,13,2,14,6,11,6,12,6,13,6,14,7
,12
940 DATA 14,12,8,12,8,14,9,12,13,9,14,10
,12
950 DATA 11,9,11,10,11,11,11,12,12,9,13,
9,13,10
960 DATA 13,11,13,12,13,13,13,14,14,14
2000 REM***** UDG ******
2010 FOR P=46359 TO 46366
2020 READ N:POKE P,N:NEXT P
2030 DATA 63,63,63,63,63,63,63,63
2040 RETURN
```

FOURRRRR!

A familiar cry to all golfers, this game puts you on the first tee of a putting course which lasts for nine holes. The green is displayed graphically with the hole, flag and golfball plotted onto the screen. You must estimate the strength of shot needed to hole the ball. Any strength figure outside of the limits (1-36) is rejected and a new one asked for. If you hit the ball off the screen then you are out of bounds, your score is increased by three and your ball is placed back on the tee. There is a small random element within the game (any golfer will tell you that two shots hit with the same power do not often end up with the same result). If you get a hole in one, a special congratulations routine is called (lines 5Ø0-53Ø). If you pass the hole with a strong shot, ie the ball is to the right of the hole, then enter a minus sign before the strength figure.

At the end of the game, your score card is displayed together with the number of shots over or under par (par is two shots per hole, 18 for the complete course).

Happy golfing!

```
10 REM*****FOURRRRR!*****
20 CLS:PAPER2:INK 0:GOSUB 900
30 X=1:S=0:G=0:DIM H(9)
90 R=INT(RND(1)*18)+19:G=G+1
95 CLS:IF G>9 THEN GOTO 700
97 PRINT:PRINT
100 PRINT TAB(24)"FOURRRRR!":FOR P=1 TO
 9:PRINT:NEXT P
110 PRINT"------------------------------
-------"
```

```
115 PRINT"------------------------------
-------"
120 PRINT"------------------------------
-------"
125 PLOT R,13," ":PLOT R,12," "
130 PLOT R+1,11,"!":PLOT R+1,10,"!>"
150 PLOT X-1,11,CHR$(7)+"o"+CHR$(0)
160 IF X=R THEN GOTO 400
170 PRINT:PRINT:PRINT"HOLE NO.";G:"  SHO
TS TAKEN SO FAR ";S
175 PRINT:PRINT"SHOTS TAKEN ON THIS HOLE
 ";H(G)
190 PRINT:INPUT"SHOT STRENGTH (1 TO 36)"
;N
190 IF N>36  OR N<-36 THEN 180
200 X=X+N:IF X>37 OR X<1 THEN GOSUB 350
202 IF ABS(R-X)<2 THEN 210
205 X=X+INT(RND(1)*3)-INT(RND(1)*3)
210 H(G)=H(G)+1:GOTO 95
350 REM*****OUT OF BOUNDS*****
360 PRINT:PRINT:PRINT"SORRY SIR, BUT YOU
R BALL IS NOW OUT OF"
370 PRINT"BOUNDS! YOU RECIEVE A 3 SHOT P
ENALTY":H(G)=H(G)+3
380 SOUND 1,2000,8:WAIT 350:PLAY 0,0,0,0
:X=1:RETURN
400 REM*****HOLED SHOT******
405 PLOT R+1,11,"!"
```

```
410 WAIT 50:PLOT X,11," ":WAIT 50:PLOT X
,12,"o":WAIT 50:PLOT X,12,"
 "
420 WAIT 50:PLOT X,13,"o"
430 FOR T=400 TO 100 STEP-30
440 SOUND 1,T,8:WAIT 10:NEXT T
450 FOR T=1 TO 7:PAPER T:WAIT 40:PING:NE
XT T
460 PRINT:PRINT:PRINT TAB(24)"GREAT PUTT
!"
465 IF H(G)=1 THEN GOTO 500
470 PRINT:PRINT"YOU TOOK ";H(G);" ON THA
T HOLE"
480 PAPER 2:WAIT 200:X=1:S=S+H(G):GOTO 9
0
500 REM***HOLE IN ONE***
505 WAIT 150
510 CLS:FOR T=1 TO 22:PRINT TAB(13+T)"HO
LE IN ONE!"
520 MUSIC 1,3,(T/2)+1,0:PLAY 1,0,1,1000:
WAIT 20
530 NEXT T:WAIT 200:GOTO 480
700 REM*****END OF GAME*****
702 CLS
710 PRINT:PRINT TAB(21)"S C O R E    C A
 R D"
715 PRINT:PRINT
720 FOR G=1 TO 9
```

```
730 PRINT"HOLE NO.";G;"  SHOTS TAKEN ";H
(G):PRINT
740 NEXT G
750 PRINT:PRINT"A TOTAL OF ";S;
760 P=18-S:IF SGN(P)=-1 THEN P$="OVER"
770 IF SGN(P)=1 THEN P$="UNDER"
780 IF P=0 THEN GOTO 820
790 PRINT:PRINT:PRINT"THAT IS ";ABS(P);"
 ";P$;" PAR."
800 SOUND 1,150,8:WAIT 100:PLAY 0,0,0,0
810 END
820 PRINT:PRINT:PRINT"THAT IS EQUAL TO P
AR.":SOUND 1,150,8:WAIT 100
830 PLAY 0,0,0,0:END
900 REM**** UDG's ****
910 FOR P=46440 TO 46447
920 READ N:POKE P,N:NEXT P
930 DATA 63,63,63,63,63,63,63,63
940 RETURN
```

BINGO

Everybody has played Bingo at some time or other. Be it at a party or down at the seaside, it is one of Britain's major games. In this program, the computer acts as Bingo caller thus releasing you to play the game. The numbers are displayed on the Hi-Res screen and are large enough for everybody to read. The computer randomly chooses numbers which are then displayed on the screen. Before the number is displayed, the computer checks to see if it has already been called. If this is so, it forgets that number and generates another one.

I've used numbers between one and 9Ø which, as far as I know, are the standard range of figures used in Bingo. However, if your Bingo set is different, you can change line 7Ø to the number of numbers wanted and then change lines 6Ø and 8Ø accordingly.

If you have a Bingo set, then press any key on the keyboard and the computer will print all the numbers called so that you can check your card. It will then offer another game.

All lines up to line 2Ø print the large size 'BINGO' on the screen. Lines 6Ø to 17Ø are the core of the program, producing the random number, checking it, moving the cursor on the Hi-Res screen if the number is a single figure, and providing all the other vital work.

Lines 19Ø to 33Ø come into play when Bingo has been called or when every number has been called.

The bulk of the program (from line 1ØØØ onwards) draws the numbers which appear on screen. The Oric's graphics commands are powerful but they do consume a lot of memory.

Study the program and as a result, why not try to write a program which will display large-scale letters when that letter is pressed on the keyboard. You could then modify this new program into something which could form the basis of a young child's spelling game.

```
10 REM****BINGO****
15 CLS:PRINT:PRINT"*****   *   *   *
*****  *****  *"
16 PRINT"*   *   *   **  *  *      *
 *  *"
17 PRINT"*****   *   * * *  * ***  *
 *  *"
18 PRINT"*   *   *   *  **  *   *  *
 *"
19 PRINT"*****   *   *   *  *****  ***
**  *"
20 PRINT:PRINT:PRINT"        Press a k
ey to start."
30 GET A$:N=-1:DIM R(90)
50 HIRES:PRINT:PRINT"  To call Bingo,
press any key."
60 N=N+1:IF N=88 THEN 190
70 R=INT(RND(1)*90)+1:R(N)=R
72 IF N=0 THEN 80
75 FOR T=0 TO N-1:IF R(T)<>R THEN 77 E
LSE GOTO 70
77 NEXT T
80 R$=STR$(R)
90 RL$=LEFT$(R$,2):RR$=RIGHT$(R$,1)
```

```
95 IF LEN(R$)<3 THEN 120
100 RL=VAL(RIGHT$(RL$,1)):X=30:Y=50:CU
RSET X,Y,0
105 RL=RL+1
110 ON RL GOSUB 1000,1030,1060,1090,11
20,1150,1180,1210,1240,1270
120 RR=VAL(RR$):X=130:Y=50
130 RR=RR+1:CURSET X,Y,0
140 ON RR GOSUB 1000,1030,1060,1090,11
20,1150,1180,1210,1240,1270
150 PING:WAIT 350
170 IF KEY$="" THEN GOTO 50
190 TEXT:PAPER 1:INK 7:CLS
200 PLOT 18,10,"BINGO!"
210 SOUND 1,150,7:WAIT 20
220 PLOT 18,10,"------"
230 SOUND 1,80,8:WAIT 20
240 PLOT 8,25,"PRESS A KEY"
250 IF KEY$="" THEN GOTO 200
260 CLS:INK 7:PRINT:PRINT:PRINT TAB(24
)"NUMBERS CALLED WERE"
270 PRINT:PRINT:PRINT
280 FOR F=0 TO N:PRINT R(F);"    ";:NEX
T F
310 PRINT:PRINT:PRINT TAB(24)"ANOTHER
GAME?"
320 GET A$:IF A$="Y" THEN RUN
330 END
```

```
1000 CURSET X,Y,0:DRAW 0,100,1:DRAW 60
,0,1:DRAW 0,-100,1:DRAW -60,0
,1
1010 CURSET X+10,Y+10,0:DRAW 0,80,1:DR
AW 40,0,1:DRAW 0,-80,1:DRAW -
40,0,1
1020 RETURN
1030 DRAW 0,100,1:DRAW 20,0,1:DRAW 0,-
100,1:DRAW -20,0,1:RETURN
1060 DRAW 60,0,1:DRAW 0,40,1:DRAW -50,
50,1:DRAW 50,0,1:DRAW 0,10,1:
DRAW-60,0,1
1070 DRAW 0,-10,1:DRAW 50,-50,1:DRAW 0
,-30,1:DRAW -50,0,1:DRAW 0,-1
0,1:RETURN
1090 DRAW 60,0,1:DRAW 0,100,1:DRAW -60
,0,1:DRAW 0,-10,1:DRAW 50,0,1
1100 DRAW 0,-35,1:DRAW -40,0,1:DRAW 0,
-10,1:DRAW 40,0,1:DRAW 0,-35,
1
1110 DRAW -50,0,1:DRAW 0,-10,1:RETURN
1120 RESTORE:FOR A=1 TO 14:READ C,D:DR
AW C,D,1:NEXT A
1130 DATA 10,0,0,70,20,0,0,-20,10,0,0,
20,20,0,0,10,-20,0,0,20
1140 DATA -10,0,0,-20,-30,0,0,-80:RETU
RN
1150 DRAW 60,0,1:DRAW 0,10,1:DRAW -50,
0,1:DRAW 0,35,1:DRAW 50,0,1:D
RAW 0,55,1
1160 DRAW -60,0,1:DRAW 0,-10,1:DRAW 50
,0,1:DRAW 0,-35,1:DRAW -50,0,
1
1170 DRAW 0,-55,1:RETURN
1180 DRAW 60,0,1:DRAW 0,10,1:DRAW -50,
0,1:DRAW 0,30,1:DRAW 50,0,1
1185 DRAW 0,60,1:DRAW -60,0,1:DRAW 0,-
100,1:CURMOV 10,50,0:DRAW 40,
0,1
1190 DRAW 0,40,1:DRAW -40,0,1:DRAW 0,-
40,1:RETURN
1210 DRAW 60,0,1:DRAW 0,10,1:DRAW -40,
90,1:DRAW-10,0,1:DRAW 40,-90,
1
1220 DRAW -50,0,1:DRAW 0,-10,1:RETURN
1240 DRAW 0,100,1:DRAW 60,0,1:DRAW 0,-
100,1:DRAW -60,0,1:DRAW 10,10
,0
1250 DRAW 0,35,1:DRAW 40,0,1:DRAW 0,-3
5,1:DRAW -40,0,1:CURMOV 0,45,
0
1260 DRAW 0,35,1:DRAW 40,0,1:DRAW 0,-3
5,1:DRAW -40,0,1:RETURN
1270 DRAW 60,0,1:DRAW 0,100,1:DRAW -10
,0,1:DRAW 0,-50,1:DRAW -50,0,
1
```

```
1280 DRAW 0,-50,1:CURMOV 10,10,0:DRAW
40,0,1:DRAW 0,30,1:DRAW -40,0
,1
1290 DRAW 0,-30,1:RETURN
```

MOPONOLY

A mammoth game this, the longest in the book, which plays a good version of the famous real estate board game. If you have played the game before, just think of all the knowledge, decisions, encounters and happenings in the game and now add to that the strategy of one player, and the result you get is approximately what the computer has to do. You can also add the tasks of processing the other player's response, screen formatting, generating the dice throws, Chance and Community Chest cards and a whole host of other mundane chores. With the computer busy, you are left free to enjoy the game.

You and the computer, then, battle it out for control of the streets of London. Once you have decided upon a site's condition, be it empty, with one house or so on, that is the way it stays in this version. Therefore, you must be that more aware of your strategy as any decisions made in this area cannot be undone.

This game features Go To Jail, Chance and Community Chest cards and a novel twist to Free Parking which you will see if you play the game. As normal, you receive an extra £200 for passing Go which can easily be deducted from your total cash if you land on Income Tax.

Times have changed since Monopoly was first invented in the U.S. in the thirties. Allowing for inflation, the cash you start off with is much higher than the board game's funds. However, the rent you may have to pay if you trespass on the computer's property is far higher than the original costs so one thing evens the other out.

As there are only two of you playing, you throw a single dice to move around the board. The computer checks to see if you have landed on one of the unbuyable properties, eg Super Tax, and if not allows you to consider buying it. If you do buy it, the computer then checks to see that the property has the capability for building on (for example, the stations, Water Works and Electric Company do not). If it does, it asks you for the number of houses you would like to build, and if that is four, would you like a hotel. Answer 'Y' to this if you want to.

The computer then makes its move, deciding whether to buy a site and if it does, how many houses to build on it. If either of you land on one of the unbuyable properties, the computer goes to a separate routine to deal with the situation. Every eight moves you will be shown the properties that you and the computer own. This routine is contained in lines 1600 to 1800. I have left some line space there so that if you want to you can add the facility to be able to change the number of houses on each item of property.

The various sounds made in the game can be attributed to line 4000 and beyond. Lines 1300 to 1420 end the game with either you or the computer winning—in this game, there are no draws only a winner and a loser, so be warned!

Good luck and happy house-hunting!

```
10 REM*****MOPONOLY*********
30 CLS:PAPER 4:INK 7
35 CLS
40 PLOT 15,2,"MOPONOLY"
50 PLOT 14,3,"----------"
55 GOSUB 4130
60 PLOT 0,6,"HELLO THERE!"
65 FOR T=1 TO 8:PRINT"":NEXT T
80 DIM B$(80)
90 FOR A=1 TO 40:READ B$(A):NEXT A
```

```
100 DIM P$(40),CP$(40),V(40),PR(40),CR(
40)
110 GOSUB 5150
120 M=9000:CM=9000:P=1:CP=1
130 REM*******MAIN LOOP*******
132 PAPER 0:CNT=CNT+1:IF CNT=8 THEN CNT
=0:GOSUB 1600:PRINT:PRINT
133 PRINT"Press a key to roll the dice.
"
135 GET A$:PAPER 4:GOSUB 4100
140 D=INT(RND(1)*6)+1:P=P+D
142 PRINT:PRINT:PRINT"DICE ROLLS: ";D
143 GOSUB 4000
147 IF P>40 THEN P=P-40:M=M+200:GOSUB 2
000
150 F=0:IF B$(P)="GO" THEN 450
160 IF B$(P)="COMMUNITY CHEST" THEN 770
170 IF B$(P)="INCOME TAX" THEN 950
180 IF B$(P)="CHANCE" THEN 980
190 IF B$(P)="JAIL" THEN 1160
200 IF B$(P)="FREE PARKING" THEN 1190
210 IF B$(P)="GO TO JAIL" THEN 1240
220 IF B$(P)="SUPER TAX" THEN 1270
230 PRINT:PRINT"You land on ";B$(P);" w
orth"
240 PRINT"_";V(P)
250 IF CP$(P)=B$(P) THEN 420
```

```
260 IF B$(P)=P$(P) THEN PRINT"You own "
;B$(P): GOTO 450
270 PRINT:PRINT:PRINT"Will you buy ";B$
(P);" ?":GET X$
280 IF X$="N" THEN 450
285 PRINT
290 PRINT"Fine, that will be _";V(P)
300 M=M-V(P):PR(P)=V(P)/10:P$(P)=B$(P)
310 IF RIGHT$(B$(P),3)="ION" OR B$(P)="
WATER WORKS" THEN 450
315 IF B$(P)="ELECTRIC COMPANY" THEN 45
0
320 PRINT:PRINT"Will you build on ";B$(
P):GET X$
330 IF X$<>"Y" THEN 450
335 PRINT
340 INPUT"How many houses";N:PRINT
350 IF N<4 THEN GOTO 380
355 IF N>4 THEN PRINT" You must be jok
ing!!!":PRINT:PRINT:GOSUB 420
0:GOTO 340
360 INPUT "An hotel as well";A$
370 IF LEFT$(A$,1)="Y" THEN N=5
380 PR(P)=PR(P)+(V(P)*N)
390 M=M-(200*N)
395 PRINT
400 PRINT"Good":FOR A=1 TO 1000:NEXT A
410 GOTO 450
```

```
420 PRINT:PRINT"You are trespassing on
my property.":PRINT:PRINT"
  PAY : ";
430 PRINT"_";CR(P):M=M-CR(P):CM=CM+CR(P
)
435 PRINT:PRINT
440 PRINT"Thankyou very much!!!":PRINT
450 PRINT:PRINT:PRINT"You now have _";M
:PRINT
460 PRINT"I have _";CM:PRINT
470 PRINT"-------------------------------
---------"
473 GOSUB 4050
475 FOR T=1 TO 1000:NEXT T
480 IF M<0 THEN 1300
490 IF CM<0 THEN 1350
500 PRINT:PRINT:PRINT"MY SHOT"
505 PAPER 1:PRINT:PRINT
510 CP=CP+INT(RND(1)*6)+1
520 F=1:IF CP>40 THEN CP=CP-40:M=M+200:
GOSUB 2000
530 IF B$(CP)="COMMUNITY CHEST" THEN 77
0
540 IF B$(CP)="INCOME TAX" THEN 950
550 IF B$(CP)="CHANCE" THEN 980
560 IF B$(CP)="JAIL" THEN 1160
570 IF B$(CP)="FREE PARKING" THEN 1190
580 IF B$(CP)="GO TO JAIL" THEN 1240
```

```
590 IF B$(CP)="SUPER TAX" THEN 1270
600 IF P$(CP)=B$(CP) THEN CM=CM-PR(CP):
M=M+PR(CP):PRINT"I trespass
on ";B$(CP)
610 IF P$(CP)=B$(CP) THEN 730
620 IF V(CP)=0 THEN 730
630 IF B$(CP)=CP$(CP) THEN 730
640 PRINT"I landed on ";B$(CP)
645 IF B$(CP)="GO" THEN 730
650 IF CM/V(CP)<10 OR CM<1000 THEN 730
660 N=INT(RND(1)*5)+1
665 PRINT:PRINT
670 IF N=5 THEN PRINT"I buy ";B$(CP)
680 IF RIGHT$(B$(CP),1)="N" OR B$(CP)="
WATER WORKS" THEN 710
685 IF B$(CP)="ELECTRIC COMPANY" THEN 7
10
690 IF N=5 THEN PRINT"with 4 houses and
 1 hotel.":GOTO 710
700 PRINT:PRINT"I buy ";B$(CP):PRINT"wi
th ";N;" house/s"
710 CR(CP)=(V(CP)/10):CR(CP)=CR(CP)+(N*
200):CP$(CP)=B$(CP)
720 CM=CM-V(CP):CM=CM-(V(CP)*N
730 PRINT:PRINT"--------------------------
--------------"
735 GOSUB 4050:WAIT 350
740 PRINT:PRINT:PRINT:PAPER 4:GOTO 130
770 IF F=0 THEN A$="You":V=M
```

```
775 PRINT:PRINT
780 IF F=1 THEN A$="I":V=CM
790 PRINT A$;" land on Community Chest.
"
800 C=INT(RND(1)*5)+1
810 IF F=0 THEN M=V
815 IF F=1 THEN CM=V
817 PRINT:PRINT
820 ON C GOSUB 860,880,900,920,940
830 IF F=1 THEN CM=V
835 IF F=0 THEN M=V
840 IF F=1 THEN 730
850 IF F=0 THEN 450
860 PRINT A$;" inherit _100.":V=V+100
870 RETURN
880 PRINT A$;" have won a beauty contes
t!":PRINT TAB(24)"Collect _5
0":V=V+50
890 RETURN
900 PRINT"Bank error. ";A$;" collect _2
00.":V=V+200
910 RETURN
920 PRINT A$;" have to pay _50 for insu
rance!":V=V-50
930 RETURN
940 PRINT"Annuity matures. ";A$;" colle
ct _75.":V=V+75
945 RETURN
```

```
950 PRINT:PRINT
955 IF F=0 THEN PRINT"Ha Ha, pay _200 i
ncome tax.":M=M-200
960 IF F=1 THEN PRINT"Blast! I have to
pay _200 income tax!":CM=CM-
200
970 GOTO 840
980 IF F=0 THEN A$="You":V=M
990 IF F=1 THEN A$="I":V=CM
995 PRINT:PRINT:PRINT A$;" have landed
on CHANCE...."
997 PRINT:PRINT:WAIT 300
998 PRINT:PRINT
1000 C=INT(RND(1)*5)+1:IF C=5 THEN 1140
1010 ON C GOSUB 1040,1060,1090,1120
1015 PRINT:PRINT
1020 IF F=0 THEN M=V
1022 IF F=1 THEN CM=V
1030 GOTO 840
1040 PRINT A$;" won a crossword competi
tion."
1045 PRINT A$;" recieve _100.":V=V+100
1050 RETURN
1060 PRINT A$;" must go back 3 spaces!"
1070 IF F=0 THEN P=P-3 ELSE CP=CP-3
1080 RETURN
1090 PRINT"Advance to Mayfair. Yipee!!!
"
```

```
1095 GOSUB 4100
1100 IF F=0 THEN P=40 ELSE CP=40
1110 RETURN
1120 PRINT A$;" have been speeding agai
n. Lose _15":V=V-15:GOSUB 40
00
1130 RETURN
1140 IF F=0 THEN PRINT"Go to JAIL. Do n
ot pass GO.":GOSUB 4000:GOTO
 1240
1150 IF F=1 THEN PRINT"I must go to JAI
L. Oh dear.":GOSUB 4000:GOTO
 1240
1160 IF F=0 THEN PRINT"You're not in JA
IL, just visiting."
1170 IF F=1 THEN PRINT"I'm not in JAIL,
 only visiting."
1180 GOTO 840
1190 PRINT:PRINT
1195 IF F=0 THEN INPUT"Do you wish to p
ark here?";K$
1200 IF F=1 THEN PRINT"I will not park
here.":GOTO 840
1210 IF LEFT$(K$,1)="N" THEN PRINT"Wise
 choice":GOTO 840
1220 PRINT"After you parked, thieves st
ole your  vehicle and all yo
ur gains."
```

```
1225 PRINT:PRINT" You should not have p
arked there."
1227 GOSUB 4200
1228 FOR T=1 TO 2000:NEXT T
1230 WAIT 500:GOTO 840
1240 IF F=0 THEN PRINT"You're in JAIL.
It costs _50 to be    releas
ed.":M=M-50
1245 IF F=0 THEN P=31
1250 IF F=1 THEN PRINT"I'm in JAIL. Boo
 Hoo!":CM=CM-50:CP=31
1260 GOTO 840
1270 IF F=0 THEN PRINT"You must pay Sup
er Tax at _100":M=M-100
1280 IF F=1 THEN PRINT"I must pay Super
 Tax at _100. Yuk!":CM=CM-10
0
1285 GOSUB 4200
1290 GOTO 840
1300 PAPER 0:INK 1:CLS
1305 PLOT 4,10,"YOU HAVE NO MONEY, YOU
LOSE"
1307 GOSUB 4000
1310 FOR T=1 TO 1000:NEXT T
1320 INPUT"ANOTHER GAME";A$
1330 IF LEFT$(A$,1)="Y" THEN RUN
1340 PING:PING:END
1350 PRINT"BOO! HOO! I LOST."
1355 GOSUB 4000
```

```
1360 CLS:INK 3:PAPER 0
1370 PRINT:PRINT:PRINT"   I just could
not beat you. I tried my bes
t but I"
1380 PRINT"was outclassed by a superior
 player."
1390 PRINT:PRINT:PRINT"   You had _";M;
" at the end of the game."
1400 PRINT:PRINT:PRINT"}}}}}}}}}}}}}}}W
ELL DONE{{{{{{{{{{{{{{"
1420 FOR L=1 TO 5:FOR Z=1 TO 7:INK Z:WA
IT 50:NEXT Z:NEXT L:GOTO 132
0
1600 REM*****PROPERTY DISPLAY*****
1610 CLS:PRINT:PRINT"You own: ";
1620 FOR P=1 TO 40:IF P$(P)<>"" THEN PR
INT TAB(24)P$(P)
1625 NEXT P
1630 PRINT:PRINT"        Press a key to
continue":GET A$
1640 CLS:PRINT:PRINT"I own: ";
1650 FOR CP=1 TO 40:IF CP$(CP)<>"" THEN
 PRINT TAB(23)CP$(CP)
1655 NEXT CP
1660 PRINT:PRINT"Press any key to conti
nue the game":GET A$:CLS:RET
URN
1800 RETURN
```

```
2000 PRINT:PRINT"You passed GO, collect
 _200."
2010 PRINT:RETURN
4000 FOR T=100 TO 400 STEP 20
4010 SOUND 1,T,8
4020 WAIT 20
4030 NEXT T
4035 WAIT 120
4040 PLAY 0,0,0,0
4045 RETURN
4050 FOR T=240 TO 15 STEP-15
4060 SOUND 1,T,8
4070 WAIT T/9
4080 NEXT T
4090 PLAY 0,0,0,0
4095 RETURN
4100 SOUND 1,90,8
4110 WAIT 100
4120 PLAY 0,0,0,0:RETURN
4130 FOR T=1000 TO 40 STEP-20
4140 SOUND 1,T,0
4150 PLAY 7,0,1,2000
4160 WAIT 10
4170 NEXT T
4180 PLAY 0,0,0,0
4200 SOUND 1,2300,15
4215 WAIT 100
4230 PLAY 0,0,0,0
```

```
4240 RETURN
5000 DATA "GO", "OLD KENT ROAD", "COMMU
NITY CHEST", "WHITECHAPEL RO
AD"
5010 DATA "INCOME TAX", "KING'S CROSS S
TATION", "ANGEL, ISLINGTON"
5020 DATA "CHANCE", "EUSTON ROAD", "PEN
TONVILLE ROAD", "JAIL", "PAL
L MALL"
5030 DATA "ELECTRIC COMPANY", "WHITEHAL
L", "NORTHUMBERLAND AVE"
5040 DATA "MARLYEBONE STATION", "BOW ST
REET", "COMMUNITY CHEST"
5050 DATA "MARLBOROUGH STREET", "VINE S
TREET", "FREE PARKING", "STR
AND"
5060 DATA "CHANCE", "FLEET STREET", "TR
AFALGAR SQUARE", "FENCHURCH
STATION"
5070 DATA "LEICESTER SQUARE", "COVENTRY
 STREET", "WATER WORKS", "PI
CCADILLY"
5080 DATA "GO TO JAIL", "REGENT STREET"
, "OXFORD STREET", "COMMUNIT
Y CHEST"
5090 DATA "BOND STREET", "LIVERPOOL ST.
 STATION", "CHANCE", "PARK L
ANE"
```

```
5100 DATA "SUPER TAX", "MAYFAIR"
5150 FOR A=1 TO 40:READ V(A):NEXT A
5170 RETURN
5200 DATA 0,60,0,60,0,200,100,0,100,120
,0,140,150,140,160,200,180,0
,180,200,0
5210 DATA 220,0,220,240,200,260,260,150
,280,0,300,300,0,320,200,0,3
50,0,400
```

TRAVELLING TRIANGLES

This program is a development of Jeremy Ruston's Spinning Triangles program for the Sinclair Spectrum and the BBC micro. The effect produced by this program is quite startling and worth the short time it will take to type in.

I have really just added some frills to it to make it that bit more enjoyable. Once the program is running, it should give the effect of triangles moving around the screen and rotating on one axis. The shapes left behind may give a three-dimensional effect, but what is quite certain is that two screenfuls will never be exactly alike.

Pressing the 'S' key while the program is running pauses the display until you press the 'S' key again. Pressing the less than key, '<', will change the colour downwards from colour code '7' to '1', while pressing the greater than key, '>', will perform the opposite. The effect of the triangles in different colours is quite beautiful. If the screen gets too crowded at any time, by pressing the spacebar, you clear the screen and set the process off again. To quit the program, press the 'Q' key.

The program uses a REPEAT/UNTIL loop for the main drawing.

```
10 REM****TRAVELLING TRIANGLES****
20 REM*******GIFFORD/RUSTON*******
30 G=7:HIRES
35 PRINT:PRINT TAB(17)"TRAVELLING TRIANG
LES ";CHR$(96);" 1983"
40 L=INT(RND(1)*240)
```

```
50 M=INT(RND(1)*200)
60 N=INT(RND(1)*240)
70 O=INT(RND(1)*200)
80 P=INT(RND(1)*240)
90 Q=INT(RND(1)*200)
100 A=INT(RND(1)*8)
110 B=INT(RND(1)*8)
120 C=INT(RND(1)*8)
130 D=INT(RND(1)*8)
140 E=INT(RND(1)*8)
150 F=INT(RND(1)*8)
155 REPEAT
160 CURSET L,M,1
165 INK G
170 DRAW N-L,O-M,1
180 DRAW P-N,Q-O,1
190 DRAW L-P,M-Q,1
210 IF L+A>239 OR L+A<0 THEN A=-A
220 IF M+B>199 OR M+B<0 THEN B=-B
230 IF N+C>239 OR N+C<0 THEN C=-C
240 IF O+D>199 OR O+D<0 THEN D=-D
250 IF P+E>239 OR P+E<0 THEN E=-E
260 IF Q+F>199 OR Q+F<0 THEN F=-F
280 L=L+A:M=M+B
290 N=N+C:O=O+D
300 P=P+E:Q=Q+F
305 Q$=KEY$:IF Q$<>"" THEN 350
310 UNTIL FALSE
```

```
320 GOTO 160
350 REM***OPTIONS****
360 IF Q$="S" THEN 400
370 IF Q$="." AND G<7 THEN G=G+1:GOTO 31
0
375 IF Q$="," AND G>1 THEN G=G-1:GOTO 31
0
380 IF Q$=" " THEN WAIT 100:PING:RUN
390 IF Q$="Q" THEN END
395 GOTO 310
400 Q$=""
405 Q$=KEY$
410 IF Q$="S" THEN 310
420 GOTO 405
```

SEQUENCE

In this program, the computer displays the numbers from zero to nine in a random order. Your task is to unscramble the numbers and put them into the sequence 'Ø 1 2 3 4 5 6 7 8 9' in as few moves as possible. This is done by either reversing the whole sequence or just a part of it. For example, the computer generates '7354Ø98621'. It will then ask you to 'REVERSE NUMBER?'. Entering '1' would reverse the entire sequence to read '12689Ø4537'. Entering '5' would then reverse the numbers from the fifth figure onwards, ie '12687354Ø9'.

Sounds simple, doesn't it? Frankly, I am terrible at this game. My best effort was 13 moves which is pretty awful—I'm sure that you can do much better. Try to aim for a score around 1Ø, anything below that being excellent or at least very lucky!

There is nothing remarkable about the game from a programming point of view. The routine from line 5ØØ onwards plays the small tune. The notes are stored in a DATA statement, the only efficient way of holding enough information to produce a tune. Line 3Ø3 raises the same tune two octaves when you complete the game. This saves memory space compared to having a separate routine to play the higher tune. While your computer has been gifted with a vast quantity of memory, it is always sensible to write a program tidily and orderly in a manner which will save memory. If you do this, you will find that the program will run easier and the techniques learnt will be valuable when you start writing programs which come close to your computer's memory limits.

Getting back to this program, you will find following this piece a sample run of the program in which after many trials and tribulations, I managed to solve the puzzle.

```
MOVE NO.1 :   9263051748


REVERSE NUMBER

7



MOVE NO.2 :   9263058471


REVERSE NUMBER

1



MOVE NO.3 :   1748503629


REVERSE NUMBER

9



MOVE NO.4 :   1748503692


REVERSE NUMBER

2
```

After much consideration (and many wrong moves!) here is the final portion of the game.

```
MOVE NO.11 :   1234508967

REVERSE NUMBER
9


MOVE NO.12 :   1234508976

REVERSE NUMBER
6

MOVE NO.13 :   1234567980

REVERSE NUMBER
8


MOVE NO.14 :   1234567089

REVERSE NUMBER
9


MOVE NO.15 :   1234567098

REVERSE NUMBER
8
```

```
MOVE NO.16 :   1234567890

REVERSE NUMBER
1



MOVE NO.17 :   0987654321

REVERSE NUMBER
2



0123456789
YOU DID IT!!!

     IT TOOK 17  MOVES

10 REM****SEQUENCE****
20 CLS
25 INK 0
30 PAPER3
35 PLOT 16,9,"SEQUENCE"
36 PLOT 15,10,"----------"
40 M=1:X=0:A$=""
50 FOR T=0 TO 9
60 L=INT(RND(1)*10)+48
70 Q=1:S=2
80 IF MID$(A$,Q,1)=CHR$(L) THEN 60
```

```
90 IF Q<T THEN Q=Q+1:GOTO 80
100 A$=A$+CHR$(L)
110 NEXT T
120 GOSUB 500
125 PAPER 3
130 PRINT:PRINT:PRINT TAB(5) "MOVE NO."
;M;":   ";A$
150 PRINT:PRINT"REVERSE NUMBER ";
160 INPUT R:IF R<1 OR R>9 THEN 160
170 B$=""
180 FOR T=10 TO R STEP-1
190 B$=B$+MID$(A$,T,1)
200 NEXT T
210 A$=LEFT$(A$,R-1)+B$
220 IF A$="0123456789" THEN 240
230 M=M+1:GOTO 120
240 FOR T=1 TO 1000:NEXT T
245 CLS
250 PAPER 1:INK 7
260 PRINT:PRINT:PRINT:PRINT
270 PRINT TAB(12) A$
280 PRINT:PRINT:PRINT
285 PRINT"YOU DID IT!!!"
290 PRINT:PRINT
300 PRINT "     IT TOOK ";M;" MOVES"
302 WAIT 400
303 S=4
305 GOSUB 500
```

```
310 PLOT 6,18,"To replay press 'S'"
320 GET S$
330 IF S$<>"S" THEN GOTO 320
340 RUN
500 FOR N=1 TO 8
510 READ D
520 MUSIC 1,S,D,8
530 PLAY 1,0,1,10000
535 PAPER N-1
540 WAIT 25
550 PLAY 0,0,0,0
560 NEXT N
570 PAPER 1
580 DATA 1,3,5,10,8,6,5,5
585 RESTORE
587 CLS
590 RETURN
```

SOLITAIRE

This classic game, at one time played with small pebbles in ancient Greece, is given an up-to-date touch in this computerised version. You must jump over one peg with another, either horizontally or vertically, and the peg that has been jumped over is removed from the board. The object of the game using this movement is to clear the board of pegs leaving one in the middle. In this game you move by entering the co-ordinates of the peg you wish to move followed by the co-ordinates of the position you wish to move to. The computer will reject any illegal moves. You enter the side number and then the top number, eg if you wish to move peg 6 (side co-ordinate), 4, (top co-ordinate) to position 4 (side co-ordinate) 4 (top co-ordinate), you would first enter '64' followed by '44'.

If you succeed in completing the puzzle, the computer will tell you how many moves you took. If you are not so fortunate, the computer will display the number of pegs still left.

If after a number of attempts you still cannot succeed, then try the solution below. Many thanks to George Furlonger of Fareham for his solution.

SOLUTION:

```
46 - 44, 65 - 45, 57 - 55, 37 - 57 45 - 65,
75 - 55, 73 - 75, 63 - 65, 54 - 56 57 - 55,
25 - 45, 55 - 35, 75 - 55, 34 - 54 36 - 34,
43 - 63, 51 - 53, 31 - 51, 32 - 52 63 - 43,
51 - 53, 43 - 63, 55 - 53, 63 - 43 34 - 32,
13 - 33, 15 - 13, 43 - 23, 13 - 33 32 - 34,
24 - 44.
```

```
 10 REM*******SOLITARE*********
 15 GOSUB 400
 20 GOSUB 900
 30 GOSUB 250
40 REM******MOVE******
50 PRINT:PRINT"Which peg do you wish to
move?"
60 INPUT A
70 IF A=99 THEN 240
80 IF A<11 OR A>77 THEN 50
90 IF A(A)<>79 THEN 50
95 PRINT
100 PRINT A;"to where?";
110 INPUT B
120 IF B<11 OR B>77 THEN 110
130 IF A(B)<>E THEN 110
140 A((A+B)/2)=E:A(A)=E:A(B)=79
150 MV=MV+1
160 CO=0
170 FOR F=11 TO 75
180 IF A(F)=79 THEN CO=CO+1
190 NEXT F
200 GOSUB 250
205 PRINT
210 PRINT"There are ";CO;" pegs still on
 the board"
220 IF CO<>1 THEN 40
```

```
230 IF A(44)=79 THEN PRINT:PRINT"You did
 it in just ";mv;" moves!":
END
240 PRINT:PRINT" The game is over, and y
ou've failed!":END
250 REM*****PRINT OUT/DISPLAY*****
260 CLS
270 PRINT"Enter side co-ordinate first"
275 PRINT
280 PRINT TAB(18)"Enter 99 to concede."
285 PRINT:PRINT
290 PRINT"     1 2 3 4 5 6 7"
295 PRINT
300 PRINT TAB(18);
310 FOR D=11 TO 75
320 T=10*(INT(D/10))
330 IF D-T=8 THEN D=D+2:PRINT T/10:PRINT
 TAB(18);:GOTO 350
340 PRINT CHR$(A(D));" ";
350 NEXT D:PRINT "    7"
360 PRINT:PRINT
370 PRINT"Moves so far: ";MV
390 RETURN
400 REM*******INITIALISE*******
410 CLS
420 DIM A(87)
430 E=42
440 FOR D=11 TO 75
```

```
450 T=10*(INT(D/10))
460 IF D-T=8 THEN D=D+3
470 READ A(D)
480 NEXT D
490 MV=0
500 RETURN
510 REM********DATA BLOCK********
520 DATA 32,32,79,79,79,32,32
530 DATA 32,32,79,79,79,32,32
540 DATA 79,79,79,79,79,79,79
550 DATA 79,79,79,42,79,79,79
560 DATA 79,79,79,79,79,79,79
570 DATA 32,32,79,79,79,32,32
580 DATA 32,32,79,79,79
900 REM****** UDG'S ******
910 FOR P=46416 TO 46423
920 READ N:POKE P,N
930 NEXT
940 DATA 0,0,0,12,12,0,0,0
950 RETURN
```

THE AXEMAN COMETH

This program takes you back 6ØØ years to the little-known kingdom of Pendragon. King Edbert III is holding you captive and your last chance is to guess the word he has put before you. If you succeed, freedom, if not . . . well you'll see . . .

This program has quite an extensive dictionary of words. It is worth typing them all in as you will need a good selection of words for a good game of hangman. You are given six chances at the beginning of the game and lose one chance for every wrong letter that you guess. When you think you know the word, enter the whole word. The computer will not accept the complete word if it is only displayed over the dashes, it has to be entered as a separate guess. If you guess the complete word wrongly you lose three guesses.

The program makes extensive use of arrays and the string-handling commands of your computer.

The pattern drawn if you win, is created in between line 1ØØØ and line 1Ø4Ø simply by drawing concentric circles but using the Oric's PATTERN command to give the dotted effect.

```
10 REM*****THE AXEMAN COMETH*****

20 GOSUB 9000

30 L=LEN(W$)

35 DIM D$(20)

40 C=6

45 FOR G=1 TO 20:D$(G)="":NEXT

50 PAPER 6:INK 0

65 PRINT TAB(20)"THE AXEMAN COMETH"

67 PRINT:PRINT:PRINT:PRINT
```

```
70 PRINT TAB(18);
80 FOR X=1 TO L:PRINT"- ";:NEXT X
90 PRINT:PRINT:PRINT"     THE WORD HAS ";
L;" LETTERS"
95 WAIT 100
100 INPUT C$:PRINT
104 MUSIC 1,3,5,8:WAIT 100:PLAY 0,0,0,0
105 N=0
110 IF LEN(C$)=1 THEN 140
120 IF C$=W$ THEN 1000
130 C=C-3
140 FOR V=1 TO L
150 IF C$=MID$(W$,V,1) THEN D$(V)=C$
155 IF C$<>MID$(W$,V,1) THEN N=N+1
160 NEXT V
180 IF N=L THEN C=C-1
190 IF C<1 THEN 2000
200 PRINT:PRINT"------------------------------
---------------"
202 PRINT:PRINT"YOU HAVE ";C;" CHANCE/S
LEFT":PRINT:PRINT
205 PRINT TAB(18);
210 FOR V=1 TO L
215 X$=D$(V)
220 IF D$(V)="" THEN X$="-"
240 PRINT X$;" ";
250 NEXT V
300 GOTO 100
```

```
1000 WAIT 100:HIRES
1005 PATTERN 170
1010 CURSET 120,100,0
1020 FOR R=90 TO 1 STEP-3
1022 SOUND 1,(10*R)+20,7:WAIT 10
1025 CIRCLE R+1,1
1030 CIRCLE R,1
1040 NEXT R
1050 PLAY 0,0,0,0
1060 WAIT 120:TEXT:CLS
1070 PLOT 6, 10,"WELL DONE, YOU EVADED T
HE AXEMAN"
1075 GET R$
1080 IF R$<>"Y" THEN 1000
2000 REM*********LOSE**********
2010 PAPER 0:INK 7:CLS
2020 PRINT:PRINT"##########################
###############":PRINT
2030 PRINT"  ....The drums start to roll
..."
2040 PRINT:PRINT"The crowd quietens down
..."
2050 PRINT:PRINT"You step onto the stage
..."
2060 PRINT:PRINT"But this is no act, thi
s.."
2070 PRINT:PRINT"Is for real. Your heart
..."
```

```
2080 PRINT:PRINT"Beats faster, you put o
n.."
2090 PRINT:PRINT"The blindfold, the axem
an"
2100 PRINT:PRINT"Raises the shining blad
e.."
2110 PRINT:PRINT"You are no more, human.
..."
2112 PRINT:PRINT:PRINT"##################
######################"
2115 WAIT 900
2120 FOR T=1 TO 24:PLOT 0,T,CHR$(1)
2130 WAIT 34:NEXT T
2140 END
9000 REM****CHOOSE WORD****
9010 FOR K=1 TO INT(RND(1)*60)+1
9020 READ W$
9030 NEXT K
9040 CLS
9080 RETURN
9110 DATA "PERPENDICULAR","VAMPIRE","ANG
ULAR","DIAMOND","UMBRELLA",
"SHREDDING"
9120 DATA "DENIAL","ESTABLISHMENT","MISU
NDERSTANDING","FUTURISTIC",
"GROTESQUE"
9130 DATA "PSYCHOLOGIST","ALGORHYTHM","C
HRYSANTHAMUM","LOGISTICS","
```

```
POSITIVE"
9140 DATA "EXAMINATION","TRAVELLER","BAT
CHELOR","DEPRESSION","PASTU
RISED"
9150 DATA "ORGANISATION","SANCTUARY","MA
GNETISM","TACHOGRAPH","JUDI
CIARY"
9160 DATA "ROUNDABOUT","SYNTHETIC","MEDA
LLION","ALCOHOLIC","CHIROPO
DIST"
9170 DATA "HARLEQUIN","SOCIALIST","CONSE
RVATIVE","HARMONIES","CULIN
ARY"
9180 DATA "PRETENTIOUS","CIVILISATION","
WRINKLED","GARGOYLE","ANTIB
IOTIC"
9190 DATA "TEMPESTUOUS","ARCHITECTURE","
GENETICS","ZOOLOGIST","TOTA
LITARIAN"
9200 DATA "BRUSQUE","INTELLIGENCE","UTOP
IAN","LUDICROUS","VIVACIOUS
","BULBOUS"
9210 DATA "MANDATORY","SPECTRAL","OINTME
NT","FLOUNDER","HYDRAULICS"
9220 DATA "RADIATOR","PREPARATION","TRAN
SITION"
```

WOODEN SHOE

This is a version of an ancient English dice game, also known as Sweatcloth, in which three dice are thrown into a wooden shoe—hence the name of my program.

You are given £30 to gamble with. You must enter your bet; any size of bet is allowed between £1 and your total fund. You then choose a number between one and six. The computer rolls the dice and informs you of your situation. If one dice comes up the same as your number, then you get your money back. If two dice are the same as your number then you gain how ever much you bet and if you are lucky enough to find all three dice the same as your choice then you gain twice your bet.

For example, if you bet £5 on number '5', and it comes up three times, then you gain an extra £10 as well as receiving your money back.

Doesn't sound bad, does it? Well if you play the game for some length of time you will see that, as in any other gambling game, the odds are stacked against you. To win, you need to break the £250 barrier. This can be done, despite the odds, as long as you get an early win.

Lines 6Ø and 7Ø ask for your name which is used throughout the game to add a personal touch. The dice are rolled in line 18Ø and your win/loss is calculated in line 2ØØ.

The main subroutine, starting at line 5ØØ displays the amount of cash that you have left.

```
10 REM*********WOODEN SHOE***********
20 PAPER 6:INK 0
30 M=30
40 CLS
50 PRINT:PRINT:PRINT
60 PRINT"WHAT IS YOUR NAME":INPUT N$
70 PRINT:PRINT:PRINT"OK ";N$;" LET'S PLA
Y WOODEN SHOE"
75 WAIT 150
80 REM*******MAIN LOOP*******
85 CLS:GOSUB 500
90 PRINT:PRINT:PRINT
100 PRINT"HOW MUCH WOULD YOU LIKE TO BET
 ";N$
105 INPUT A
110 IF A>M THEN 100
120 M=M-A:PRINT:PRINT
130 PRINT"WHICH NUMBER ARE YOU BETTING O
N ":PRINT TAB(35)N$
135 INPUT B
140 IF B<1 OR B>6 THEN 130
150 FOR C=1 TO 3
160 W=0
170 WAIT 80
180 D=INT(RND(1)*6)+1
190 PRINT:PRINT"DIE ";C;" FELL ";D
195 PRINT
200 IF D=B THEN W=A:PRINT:PRINT"SO YOU W
```

```
IN _";W;" ";N$:PRINT
210 M=M+W
220 GOSUB 500
230 NEXT C
240 WAIT 200
250 IF M>250 THEN 310
260 IF M>0 THEN 85
265 FOR J=1 TO 24
270 PRINT"THE GAME IS OVER, BECAUSE YOUR
 BROKE!"
275 SOUND 1,1600,8:WAIT 20
280 NEXT J:PLAY 0,0,0,0
285 END
310 FOR J=1 TO 25
320 PRINT"YOU'VE TOPPED _250, WELL DONE!
!!"
330 SOUND 1,INT(300/J),0
335 PAPER INT(J/3)-3
340 PLAY 7,0,3,1200
350 WAIT 30
360 NEXT J
370 WAIT 100:PLAY 0,0,0,0
380 END
500 REM********MONEY**********
505 PRINT:PRINT
510 PING:WAIT 50
520 PRINT"*********************************
********"
```

```
530 PRINT"        YOU NOW HAVE _";M
535 PRINT
540 PRINT"******************************
********"
550 RETURN
560 RETURN
```

DIGIT MUNCHER

Your greedy gobbler may be yellow and of a similar shape to that made famous in the arcades, but that is where the resemblance ends. This novel game involves you zooming round the screen munching up any number which appears. You must avoid the ghosts which keep on appearing. Fortunately, they are the non-moving variety and will only capture you if you bump into them. The larger the number you munch, the higher your score. The figure 'Ø' is worth nothing and is only put there to distract you, though they will do you no damage if you eat them. Movement is continuous so care has to be taken as you steer your Muncher around the screen with the four arrow keys.

The program includes on-screen scoring and the movement round the screen is relatively fast for a BASIC program. The main loop has been kept reasonably efficient and this has contributed to the smooth movement the game provides. An enjoyable game, indeed.

The lines 24Ø, 25Ø, 26Ø and 27Ø check the keyboard input and convert it to the change in the Muncher's co-ordinates, X and Y. Line 3ØØ provides the simple sound and the routine from lines 6ØØ to 64Ø adds the extra points to your score when you eat a number.

```
10 REM****DIGIT MUNCHER*****

15 PRINT CHR$(20)

20 GOSUB 9000

30 CLS:SC=0

40 A$="$"

50 X=19:Y=12
```

```
55 A=19:Y=12
60 PAPER 0:INK 3
65 PRINT
70 PRINT"(((((((((((((((((((((((((((((((
((((((" 
80 FOR T=1 TO 22:PRINT"(";TAB(48)"("
90 NEXT T:PRINT"((((((((((((((((((((((((
(((((((((((((" 
100 PLOT 14,0,"DIGIT MUNCHER  "+CHR$(96)
+" 1983"
150 PLOT 13,0,CHR$(1)
200 PLOT X,Y,A$
210 PLOT 3,0,STR$(SC)
220 Q$=KEY$
222 IF Q$="" THEN 235
225 V$=Q$
230 GOTO 237
235 Q$=V$
237 A=X:B=Y
240 IF Q$=CHR$(8) AND X>2 THEN X=X-1:A$=
"%"
250 IF Q$=CHR$(9) AND X<36 THEN X=X+1:A$
="$"
260 IF Q$=CHR$(10) AND Y<23 THEN Y=Y+1:A
$="&"
270 IF Q$=CHR$(11) AND Y>2 THEN Y=Y-1:A$
="#"
300 PLAY 1,0,1,7
```

```
305 D= INT(RND(1)*10):D$=STR$(D)
310 IF RND(1)>.9 THEN PLOT INT(RND(1)*33
)+2,INT(RND(1)*22)+2,D$+CHR
$(3)
320 IF RND(1)<.25 THEN PLOT INT(RND(1)*3
5)+2,INT(RND(1)*22)+2,"'"
350 PLOT A,B," "
490 IF SCRN(X,Y)=39 THEN 1000
495 IF SCRN(X,Y)>47 AND SCRN(X,Y)<58 THE
N 600
510 GOTO 200
600 WAIT 5
605 ZAP:WAIT 5
610 J=SCRN(X,Y)
620 SC=SC+100*(J-48)
640 GOTO 200
1000 WAIT 50:EXPLODE:WAIT 50
1010 FOR A=1 TO 25:PRINT:NEXT A
1020 PRINT"(((((((((((((((((((((((((((((
((((((("
1025 PRINT"(
      ("
1030 PRINT"(
      ("
1040 PRINT"(
      ("
1050 PRINT"(((((((((((((((((((((((((((((
((((((("
1060 FOR T=1 TO 15:PRINT:NEXT T
1070 PLOT 10,8,"YOU SCORED "+STR$(SC)
1080 PLOT 34,8,CHR$(3)
1100 PRINT"   WOULD YOU LIKE ANOTHER GO?
"
1110 INPUT V$
1115 IF V$="Y" OR V$="y" THEN GOTO 30
1120 PRINT CHR$(20)
1130 END
9000 REM***** UDG's ******
9010 FOR P=46360 TO 46406
9020 READ N
9030 POKE P,N
9040 NEXT P
9050 DATA 0,18,51,51,63,63,30,12
9060 DATA 0,30,63,60,56,60,63,30
9070 DATA 0,30,63,15,7,15,63,30
9080 DATA 0,12,30,63,63,51,51,18
9090 DATA 30,63,63,45,45,63,63,63
9100 DATA 63,63,63,63,63,63,63,63
9120 RETURN
```

FLIPPA!

This intriguing game can provide you with a considerable degree of mental calculation. When you run the game, you'll see a mixture of 'X's and asterisks (*) on a three by three grid. You have to end up with an 'X' in the middle and eight asterisks surrounding it.

You enter the number of the place you want to hit with your Flippa. Flipping a corner piece causes those adjoining it to change into their opposites (ie an 'X' becomes a * and a * becomes an 'X'). Hitting a middle piece on a side (numbers '2', '4', '6' and '8') causes the two pieces either side of it to change and hitting the middle piece reverses the middle pieces on each of the sides. The piece that you hit always changes.

This all may sound a bit confusing but play a couple of games and things will fall into their places. This game will provide a challenge for you and your friends for a long time.

From a programming point of view, there is nothing remarkable in this program. The bulk of the work is performed in between lines 385 and 54Ø. The large collection of IF THEN statements perform the flipping functions mentioned above.

Lines 5Ø to 9Ø determine how many 'X's and how many '*'s you start off with.

```
10 REM*******FLIPPA******

15 PAPER 5: INK 0

20 DIM A(10):DIM F(4)

30 M=-1:Q=42:X=88:P=0

50 FOR C=1 TO 9

60 A(C)=Q
```

```
70 IF INT(RND(1)+.5)=0 THEN A(C)=X
80 NEXT C
90 GOSUB 270
100 M=M+1
110 N=0
120 FOR C=1 TO 9
130 IF A(C)=X THEN N=N+1
140 NEXT C
150 IF N=1 AND A(5)=X THEN 350
155 PRINT:PRINT
160 IF M>0 THEN PRINT:PRINT"THAT WAS MOV
E ";M
170 PRINT:PRINT"NUMBER OF X's :";N
180 PRINT
190 PRINT:PRINT"WHICH ONE DO YOU WANT TO
 FLIP.?"
200 GET A$
210 WAIT 40
220 N=VAL(A$):IF N<1 OR N>9 THEN 210
230 P=N
240 GOSUB 380
250 GOTO 90
260 END
270 CLS:PRINT:PRINT TAB(29) "FLIPPA!"
272 PLOT 0,1,CHR$(7)
274 PRINT TAB(28)"---------":PLOT 0,2,CH
R$(3)
280 IF P<>0 THEN PRINT"YOU FLIPPED ";P
```

```
290 PRINT:PRINT:PRINT"1  2  3",CHR$(A(1)
);" ";CHR$(A(2));" ";CHR$(A
(3))
300 PRINT
310 PRINT"4  5  6",CHR$(A(4));" ";CHR$(A
(5));" ";CHR$(A(6))
320 PRINT
330 PRINT"7  8  9",CHR$(A(7));" ";CHR$(A
(8));" ";CHR$(A(9))
340 RETURN
350 PRINT:PRINT
360 PRINT"YOU SOLVED IT IN JUST ";M;" MO
VES!"
361 WAIT 200
362 PAPER 0:INK 1:FOR T=1 TO 24
365 PRINT" ##### WELL DONE, FLIPPA CHAMP
 #####"
367 PLOT 0,T,CHR$(INT((T/4)+1))
369 NEXT T:PING
370 PING
375 GOTO 375
380 MUSIC 1,3,INT(RND(1)*12)+1,8
382 WAIT 50:PLAY 0,0,0,0
385 IF A(N)=X THEN RETURN
390 IF N=1 THEN F(1)=2:F(2)=4:F(3)=5:F(4
)=10
400 IF N=2 THEN F(1)=1:F(2)=3:F(3)=10:F(
4)=10
```

```
410 IF N=3 THEN F(1)=2:F(2)=5:F(3)=6:F(4
)=10
420 IF N=4 THEN F(1)=1:F(2)=7:F(3)=10:F(
4)=10
430 IF N=5 THEN F(1)=2:F(2)=4:F(3)=8:F(4
)=6
440 IF N=6 THEN F(1)=3:F(2)=9:F(3)=10:F(
4)=10
450 IF N=7 THEN F(1)=4:F(2)=5:F(3)=8:F(4
)=10
460 IF N=8 THEN F(1)=7:F(2)=9:F(3)=10:F(
4)=10
470 IF N=9 THEN F(1)=8:F(2)=5:F(3)=6:F(4
)=10
480 FOR G=1 TO 4
490 F=0
500 IF A(F(G))=X THEN F=1
510 IF F=1 THEN A(F(G))=Q
520 IF F=0 AND A(F(G))=Q THEN A(F(G))=X
530 NEXT G
540 A(N)=X
550 RETURN
```

LIFE, A SIMULATION

The title says it all, this is a simulation of Conway's Life, the battle between cells as they struggle for survival. If you wish to know a little more about Life and the rules that apply to it, then read the preamble to Creator.

The computer randomly generates a start pattern for the cells. Each generation then develops according to Conway' principles.

The computer handles every feature—all you have to do is sit back and watch the fascinating display.

The POKE 48Ø42,1 in line 312 is another way of changing the colour of the line by affecting its attribute. The POKE is simply changing the colour of the line to red (colour code 1).

The program is not very long and will only take 20 minutes or so to type in.

```
10 REM*****LIFE, a simulation*****
20 G=1
25 PAPER 0:INK 2
30 CLS
40 DIM M(10,10)
50 DIM N(10,10)
60 FOR K=2 TO 9
70 FOR Z=2 TO 9
80 IF RND(1)<.45 THEN M(K,Z)=1
90 N(K,Z)=M(K,Z)
100 NEXT Z
```

```
110 NEXT K
120 GOSUB 300
130 G=G+1
140 FOR K=2 TO 9
150 FOR Z=2 TO 9
160 C=0
170 IF M(K-1,Z-1)=1 THEN C=C+1
180 IF M(K-1,Z)=1 THEN C=C+1
190 IF M(K-1,Z+1)=1 THEN C=C+1
200 IF M(K,Z-1)=1 THEN C=C+1
210 IF M(K,Z+1)=1 THEN C=C+1
220 IF M(K+1,Z-1)=1 THEN C=C+1
230 IF M(K+1,Z)=1 THEN C=C+1
240 IF M(K+1,Z+1)=1 THEN C=C+1
250 IF M(K,Z)=1 AND C<>3 AND C<>2 THEN N
(K,Z)=0
260 IF M(K,Z)=0 AND C=3 THEN N(K,Z)=1
270 NEXT Z
280 NEXT K
290 GOTO 120
300 PING
305 CLS
310 PRINT
312 POKE 48042,1
313 PLOT 2,0,"LIFE, a simulation."
314 PRINT
318 PRINT"GENERATION: ";G:PRINT
320 FOR K=1 TO 10
```

```
330 PRINT TAB(10);
340 FOR Z=1 TO 10
350 M(K,Z)=N(K,Z)
360 IF M(K,Z)=1 THEN PRINT"O";
365 IF M(K,Z)=0 THEN PRINT" ";
370 NEXT Z
380 PRINT
390 NEXT K
400 RETURN
```

WHIRLPOOL

A relaxing pattern this, just sit back and watch your computer gradually form a spiral shape. All the work is performed in line 6Ø which uses CURSET to plot on the Hi-Res screen.

Why not try and experiment with the equation in line 6Ø or with the two FOR/NEXT loops in lines 4Ø and 5Ø.

```
10 REM*****WHIRLPOOL******
20 HIRES
30 CLS:PRINT:PRINT TAB(22)"WHIRLPOOL ";C
HR$(96);" 1983"
40 FOR J=1 TO 8 STEP.15
50 FOR A=1 TO 12 STEP .5
60 CURSET (A*J*COS(J)+110),(A*J*SIN(J)+8
0),1
70 NEXT A
80 SOUND 1,INT(350-(J*40)),3
90 NEXT J
100 WAIT 80:PLAY 0,0,0,0
110 FOR T=1 TO 7
120 INK T
130 WAIT 50
140 NEXT T
150 GOTO 110
```

HOUSTON, WE HAVE A PROBLEM...

A terrific game this, certainly one of my favourites in this book. It is loosely based on the classic 'Lunar Lander' game, but it is a moving graphics game designed to test your reflexes.

You are flying your NASA spaceship back to Earth where you are expected to land in the Mid-Atlantic ocean—but there is a problem, the pre-programmed flight computer has gone haywire. You contact flight control at Houston, Texas, but they can give you little help. Fortunately, among your personal possessions on the spaceship, is an Oric which can provide you with some assistance as you try to bring your craft back to Earth by hand.

Your small computer will indicate your fuel, velocity and height, and will also provide a graphic picture of your descent. Your craft is swaying from side to side as you descend due to the buffeting it receives from the strong Atlantic winds. To succeed you must have a velocity of less than +/- 10 kph and a height of no more than 15 metres.

At the end of the game, the computer will give you a skill rating which will be positive if you landed your craft and negative if you hit the ocean with so much force that your craft shattered. You control your rate of descent by the amount of thrust you divert to your retro-rockets. These will slow your descent and, if used carefully and correctly, will provide you with a comfortable landing. The rockets are operated by pressing any key from '1' to '9'. The '9' key provides the most thrust but also uses the most fuel. You must be careful to conserve enough fuel for the extra thrust needed at landing. If you hold the rockets on for too long, then your velocity will go from negative to positive and you will shoot up the screen!

Lines 4Ø, 5Ø and 6Ø provide you with random amounts of fuel, height and starting velocity so that no two games are ever the same.

The major routine starting at line 9ØØ generates the User-Defined Graphics necessary for this game. There are five graphics needed: four for the spaceship and one for the sea. User-Defined Graphics are extremely useful when writing games programs. On the normal Text screen you can have all sorts of shapes moving around, after all a custom-designed spaceship looks better on the screen than two 'X's. Your manual explains how to use UDGs (for short) very well. The one essential item for creating UDGs must be graph paper—without this you will find it much more arduous to design your own shapes.

Below are the DATA statements for a few graphic shapes you may want to incorporate in your own programs:

DATA 3Ø, 63, 63, 63, 63, 63, 3Ø, Ø.	A ball for use in breakout, squash, etc.
DATA 18, 51, 51, 63, 63, 3Ø, 12, Ø.	
DATA 3Ø, 63, 6Ø, 56, 6Ø, 63, 3Ø, Ø.	Four 'Pacman' type
DATA 12, 3Ø, 63, 63, 51, 51, 18, Ø.	shapes.
DATA 3Ø, 63, 15, 7, 15, 63, 3Ø, Ø.	
DATA 63, 33, 33, 33, 33, 63, Ø, Ø.	An empty box.
DATA Ø, Ø, Ø, 7, 15, 31, 63, 63.	These three together
DATA 12, 12, 3Ø, 63, 63, 63, 63, 63.	make a perfect laser
DATA Ø, Ø, Ø, 56, 6Ø, 62, 63, 63.	base.
DATA 28, 28, 28, 8, 62, 8, 2Ø, 34.	A little man. If you think
DATA 28, 28, 8, 62, 8, 2Ø, 2Ø, 54.	he is too big-headed, try the next one.
DATA 2Ø, Ø, 28, 34, 34, 34, 28, Ø.	Umlaut over o (ö). Found in Germanic languages.
DATA 1, 3, 7, 15, 31, 63, Ø, Ø.	Triangle pointing to the left.

DATA 32, 48, 56, 60, 62, 63, Ø, Ø.	Triangle pointing to the right.
DATA 8, 28, 62, 28, 8, Ø, Ø, Ø.	Diamond.
DATA 12, 3Ø, 63, 33, 33, 63, 3Ø, 12. DATA Ø, Ø, Ø, 63, 63, Ø, Ø, Ø. DATA Ø, Ø, 12, 12, 12, 12, 12, 12.	These three used together, constitute a head-on aircraft.

As you can see from the sample DATA statements, the range of shapes and figures which can be created is extremely wide and varied.

Back to the Houston game, there are a few other points of note. You may have noticed that when using the Oric with its INK and PAPER commands, that the colour of all which is displayed on screen changes. This does take away the ability to print multi-coloured lines or at least to have one line a different colour from another. There is a way of getting around this, using attributes and their codes. These are mentioned rather sketchily in the Oric manual so here I will show their use simply. Using the command PLOT, it is possible to change one line's colour from the general INK colour of the time. For example, you wish to plot in the middle of the screen at line 12, the words, 'Well Done!' but you would like them in white while the general INK colour is blue. You would use a line something like this:

The CHR$ (7) changes the colour into white. Any colour can be used in this way, simply by changing the number. The codes used for the INK and PAPER commands are the numbers used in the CHR$.

There are problems with this, the rest of the line after the 'Well Done.' will also become the new colour. This can be avoided by plotting the original colour after the message. The main problem with this system is that one space either side of whatever is plotted is blanked out. This makes the use of this technique not really suitable for some arcade games.

You can see that I have used colour plot techniques in lines 75, 3ØØ, 31Ø, 312, 47Ø, 48Ø and 485.

Finally, before you go off and save your damaged spacecraft, line 35Ø provides the scrolling effect at the end of the game and lines 37Ø and 38Ø the high score.

```
10 REM** HOUSTON, WE HAVE A PROBLEM **
15 PRINT CHR$(6);CHR$(17);CHR$(20)
20 HS=-8000:GOSUB 900:SC=0
30 PAPER 0:INK 7
40 A=-20-INT(RND(1)*60):R=36:F=15
50 B=1200+INT(RND(1)*380)
60 C=300+INT(RND(1)*300)
70 CLS
75 PLOT 0,25,CHR$(6)+"''''''''''''''''
''''''''''''''''''''''"
80 PLOT 1,0,"Houston, we have a proble
m..."+CHR$(96)+" 1983"
90 B=INT(B):A=INT(A):C=INT(C)
100 PLOT 1,2,"VELOCITY:"+STR$(A)+"  HE
IGHT:"+STR$(B)+"  FUEL:"+STR$
(C)+"    "
140 RR=R:FF=F:PLOT RR,FF,"   ":PLOT RR
+1,FF-1," "
145 R=23+RND(1)*9-INT(RND(1)*9)
150 F=32-(INT((B/100))+INT((B/300))+8)
155 IF F<4 THEN F=4
160 RR=R:FF=F:PLOT RR,FF,"   ":PLOT RR
+1,FF-1," "
170 PLOT R,F,"#$%":PLOT R+1,F-1,"&"
180 SOUND 1,B,0
190 PLAY 1,1,3,250
```

```
200 T$=KEY$:T=VAL(T$)*1.7
210 WAIT 50
220 IF T>C THEN T=0
230 C=C-T
240 B=B+A+(T-5+RND(1)*2-RND(1)*2)
250 A=A+(T-5+RND(1)*2-RND(1)*2)
260 IF C<1 AND B>100 THEN 300
270 IF ABS(B)<15 AND ABS(A)<10 THEN GO
TO 450
280 IF B>14 THEN 90
290 IF C>1 THEN 305
300 PLOT 0,14,CHR$(5)+"YOU HAVE RUN OU
T OF FUEL"
305 FOR G=1 TO 3
310 EXPLODE:FOR T=1 TO 7:PLOT 0,22,CHR
$(T):PLOT 0,23,CHR$(T)
312 PLOT 0,24,CHR$(T):PLOT 0,25,CHR$(T
)
315 PLOT 4,10,"YOUR CRAFT HIT THE WATE
R"
317 PLOT 8,12,"AT "+STR$(ABS(A))+" KPH
!"
320 WAIT 10:NEXT T
325 NEXT G
330 SC=INT(SC-100-ABS(A))
350 FOR T=1 TO 52:PRINT:NEXT T
360 PRINT"YOUR SKILL LEVEL IS ";SC
365 PRINT:PRINT:PRINT:PRINT:WAIT 200
```

```
370 IF SC>HS THEN HS=SC
380 PRINT:PRINT"BEST RATING SO FAR IS
";HS
385 PRINT:PRINT:PRINT
390 PRINT:PRINT"PLEASE STAND BY FOR YO
UR NEXT MISSION"
395 FOR T=1 TO 8:PRINT:NEXT T
400 WAIT 800:GOTO 40
450 REM***** SUCCESS *****
455 CLS
460 PLOT 9,10,"HOUSTON REPORT....":WAI
T 200
470 PLOT 7,13,CHR$(1)+"Against the odd
s, you"
480 PLOT 7,15,CHR$(6)+"landed the crip
pled "
485 PLOT 5,17,CHR$(2)+"craft safely. W
ell Done!"
490 SC=C*234:GOSUB 920:GOTO 350
900 REM**** UDG's ****
901 FOR P=46360 TO 46399
902 READ N
903 POKE P,N
904 NEXT P
905 DATA 3,7,15,31,6,6,6,15
906 DATA 63,63,63,63,12,30,0,0
907 DATA 48,56,60,62,24,24,24,60
908 DATA 12,30,63,33,33,63,33,33
```

```
 909 DATA 0,33,51,63,63,63,63,63
 910 RETURN
 920 FOR J=1 TO 2
 930 FOR T=1 TO 10
 940 READ A,B
 950 MUSIC 1,3,A,0:PLAY 1,0,1,1000
 960 WAIT B:NEXT T
 970 NEXT J
 975 MUSIC 1,4,1,0:PLAY 1,0,1,1000
 980 WAIT 60:PLAY 0,0,0,0
 990 RETURN
1000 DATA 1,30,3,30,5,60,3,30,5,30,6,6
0,5,30,6,30,8,15,8,130
1010 DATA 6,30,9,30,10,60,8,30,10,30,1
2,60,10,30,12,30,12,15,12,15
1907 ,1,6,10000
```

SOME POSSIBLE APPLICATIONS FOR YOUR ORIC

You have typed in and played many of the 3Ø+ games in this book. I think it is now time just to briefly discuss some of the possible serious applications that you could put your Oric to.

Many ideas which are put forward for personal computers—such as running the heating or for use as a diary or telephone directory are really not very practical. These are the kind of things that a human armed with a pen and a piece of paper, in the case of the latter two, can do much more efficiently.

In the following chapter, I have tried to look fairly critically at a selection of possible uses to put your computer to and have included a few short routines and programs which may be of use.

Below is a very useful program which copies the display of a low resolution screen onto a printer.

```
10 REM*****SCREEN COPIER*****
20 FOR B=0 TO 25
30 FOR A=0 TO 38
40 K=SCRN(A,B)
50 LPRINT CHR$(K);
60 NEXT A
70 LPRINT
80 NEXT B
```

```
**************************************
** This is a demonstration of the   **
** Oric Screen Copy program shown   **
** above. This simple program can   **
** dump the contents of the screen  **
** onto a suitable printer.         **
**************************************




ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

1234567890/\=+-_';:{[}]|)(

!@£$%&*<>?

               THE END
               -------
```

RECORD KEEPING: Your computer is very useful as an adjunct to keeping records, like names of members of your club, or the records or cassettes in your music collection, or the numbers of trains you have spotted IF,and this is a big ‘if’, you want to sort the information into order (such as alphabetical, or highest to lowest, or by some other category such as age), or you wish to extract something out about it (such as the number of club members you have who live in Cambridge, or the number of records you have by The Beatles which were released before 1970). Unless you want to do this sort of manipulation with the information you have, the data is best kept on cards, but it is an invaluable service if you need to work with the data in some way.

INVENTORY MANAGEMENT: The size of the memory you have will obviously dictate the kind of inventory records you can keep. If you, or your parents, run a small store, you may well find that you can divide the stock into types, and keep one type (such as canned goods) in one cassette data file. Alternatively, if you run a pub, you may find the kinds of goods you have to keep in stock (15 types of beer, snacks like crisps and peanuts, various mixers and whatever) can be fairly easily kept within the memory you now have. You can write a program like this fairly easily into which you enter the sales from each day, and at the end of each week, the program tells you how much to order.

MATHS AND SCIENTIFIC WORK

For the student and the professional, there are a lot of useful tasks your computer can be set to do. After all, calculating formulas and number crunching are two of the things which a computer does best. Below is a short program which can be used when dealing with circular sections.

```
10 REM*****CIRCULAR SECTIONS*****
20 CLS:PAPER 7:INK 0
30 PRINT:PRINT TAB(24)"ENTER RADIUS"
40 INPUT R:X=2*R
50 MI=(PI*(X^4))/64
60 AS=(PI*(X*X))/4
70 PRINT:PRINT"THE MOMENT OF INERTIA IS
";MI
80 PRINT:PRINT"THE POLAR MOMENT OF INERT
IA:";2*MI
90 PRINT:PRINT"THE AREA OF THE SECTION I
S ";AS
100 PRINT:INPUT"AGAIN";A$:IF LEFT$(A$,1)
="Y" THEN RUN
110 END
```

Most people only use complex algorithms when they are in education, be it as teachers or at the receiving end of the teacher. This means that there are strong linkages between this area of use for the computer and education. You'll find that you understand a concept much better if you can write a program to solve it. It means that you have a full understanding of the calculation and can present it to the computer in a series of logical straightforward steps.

I remember with some nostalgia, how I borrowed the school's computer and wrote a simple equation solver which proceeded to do my maths homework for me for that night. What I did not think about at the time was that I must have fully understood the work to be able to write the program. Below is a simple quadratic equation solver, of a similar type to the one I wrote all those years ago.

```
10 REM*****QUADRATIC EQUATION SOLVER*****
20 CLS:PRINT:PRINT TAB(25)" AX'+BX+C=0"
30 PRINT:INPUT"ENTER A";X
40 PRINT:PRINT
50 INPUT"ENTER B";Y
60 PRINT:PRINT
70 INPUT"ENTER C";Z
80 CLS
90 Y=-Y/2/X
100 Q=Y*Y-Z/X
110 IF Q=0 THEN 250
120 IF Q>0 THEN 270
130 E=SQR(-Q)
140 PRINT"  IMAGINARY ROOTS:"
150 PRINT:PRINT
160 PRINT"  'REAL' ROOT      ";Y
170 PRINT:PRINT"IMAGINARY ROOT ";E
180 END
190 PRINT"  EQUAL ROOTS: ";Y
200 END
210 PRINT"  'REAL' ROOTS:"
220 PRINT:PRINT
230 PRINT TAB(28) Y+SQR(Q)
240 PRINT:PRINT TAB(28) Y-SQR(Q)
```

STAFF PAYROLL: If you have a small firm, you may well find that the time taken to write a program to work out what you should pay each employee each week will be well rewarded.

Such information as tax could be automatically calculated, plus other deductions, and if you hook up a printer to your Oric there is no reason why you can't get the computer to print out the pay slip for you. Of course, unless you have employees with sufficiently different pay levels and deductions to make this worthwhile, you may prefer to continue with present 'pencil, paper and calculator' method of working out pay.

ROUTINE CORRESPONDENCE: Without a printer, you cannot do this. But you may find, especially if you are organiser of a club which requires you to correspond with a reasonably large membership, that a standard letter—with provision for adding the person's name and address at the beginning, plus some personalised material at the end of the letter—could be used to print out the letters as you require them.

EDUCATION: This is a very big area, where your computer can be used to good effect. No matter which subject you're studying, you'll probably be able to discover an enjoyable way to incorporate the computer into your study, and—as I said—you'll probably learn as much from programming it to help you as you will from running the program.

It is very simple to create a number of simple interactive programs to use with young children, which will help them recognise such things as counting sequences, letters of the alphabet, and simple arithmetic. If you use it with young children, never force them to stay at the computer for a moment longer than they want to. They'll think of it as a game which they'll return to over and over again if they are not forced to do so, and the familiarity they'll gain with working with computers will be worth as much, in terms of their education, as will the material they'll learn from the program.

FOR A TEACHER: The Oric is excellent, not only as an aid to all children, both advanced and backward, but as a machine capable of printing out question sheets (if with a printer) or working out examination marks and finding the class average.

PERSONAL ACCOUNTS: This is another area where your computer can be very useful, especially if you do not use it as the final source of information about the state of your cheque book or whatever, but rather decide to use the computer program as a general indication of the mess (or otherwise) your personal finances are in. Final totting up should be done with the old standby of calculator, pencil and paper.

SIMULATING: This is an area which can make great demands on your programming ability, but which can provide many enjoyable and useful hours at the computer. If, for example, you wanted to find out what you should do with your stocks and shares, you could feed them in, with their current market values, along with the way the prices for these have changed in the past month, and project the trend forward for a month.

This could indicate not only which stocks you should sell, or which you should get rid of, but just what value your portfolio is likely to have at the end of that period.

Of course, this is a simplistic example, and one which would hardly give very useful information on what you should do with that bulging portfolio, but it may give you some idea.

Another idea: You could feed in your annual salary over the past, say, five years, noting how it has changed from year to year, add a counter-indicator on the rates of inflation in those years, and find out not only how your real spending power has changed in the period, but assuming the trends you've observed continue, how it will grow (or, horrors, shrink) in the coming years.

MAKING MONEY: Ah, you say, now we come to something really worthwhile.

I will start with a warning. Many of these ideas will seem impracticable, and will not be applicable by you, but they may start you thinking of the kinds of things you could do with your computer which could enhance your income a little (or even a lot).

ARTICLES: Many people have found that writing articles for computer magazines is a good way to enhance their income.

To write an article which is accepted, and the magazines in Britain tend to have far, far more articles than they can ever print, you must work out three things.

(1) Exactly what one thing will the article teach/demonstrate/discuss?
(2) What sort of computer owners (ie what machine, or price bracket, or user, such as hobbyist, business person or whatever) will likely to be interested in the article?
(3) Which magazine will it be sent to?

Unless you can answer those three questions, you haven't got an article, you've got a vague idea or a concept. And no-one will publish that. Buy the computer magazines, all of them if you can afford to, but certainly buy the ones that you intend to write for. In 1983 the British magazines were all paying between £15 and £60 per article, with more for photographs, programs and diagrams supplied if these were related to the article.

WHERE DO YOU SEND THEM?: The major magazines in this country are:
PRACTICAL COMPUTING IPC, Quadrant House, Sutton, Surrey, SM2 5AS (01-661 3500).
YOUR COMPUTER—(at the same address as PRACTICAL COMPUTING).
PERSONAL COMPUTER WORLD Sportscene Publishers, 14 Rathbone Place, London, W1P 1DE (01-637 7991/2/3).
PERSONAL COMPUTING TODAY—145 Charing Cross Road, London, WC2H 0EE.
COMPUTING TODAY—(at the same address as PERSONAL COMPUTING TODAY).
POPULAR COMPUTING WEEKLY—Hobhouse Court, 19 Whitcomb Street, London, WC2 7HF.
HOME COMPUTING WEEKLY—(at the same address as PERSONAL COMPUTING TODAY).

You'll find there are a number of small publications which are 'machine-specific'. They are often printed by users' groups or by the manufacturer, but welcome, and may pay for, suitable articles.

There are also many other computer magazines in Britain (such as EDUCATIONAL COMPUTING (Robin Bradbeer is the contact there) and WHICH Computer) which deal with more specific markets. If you can write for these markets, pick up a copy of the magazine from your local shop, and see what sort of style the articles are in, and make sure yours fits the pattern.

SOFTWARE: Writing your own software and selling it on a cassette is a risky business unless you are well prepared. It is often better to send your programs to magazines (listed above), preferably dumped from a printer with a cassette copy. Though there is no certainty that they will be published, the losses incurred will be minimal compared with the cost of duplicating cassettes and advertising your software.

IMPROVING YOUR PROGRAMS

After all this discussion, you are probably itching to go off and write your own programs. Here is some further advice about writing programs both games and serious. Some of the comments made earlier are repeated here in a form of summary.

You've probably gone through several stages as you develop your programming skills. After the first, brief struggle with BASIC, you suddenly discovered you could, after a fashion, write programs which ran. They may have looked pretty convoluted when you looked at their listings, and friends may have needed a detailed explanation from you before they knew what to do when running the programs, but at least they worked.

There comes a stage when you decide you're going to have to do better than that. But while you may be vaguely dissatisfied with your programs, you may not have much idea of how to go about becoming a better programmer. Here are a few guidelines which may help.

First, have a look at a printout of your listing. Programs linked by REM statements look better, and are easier to understand when you return to them after a break. Of course, shortage of memory may preclude the luxury of REM statements, but this is rarely any worry with the Oric's memory. REM statements filled just with a line of asterisks can prove quite useful in separating each major section of the program. Examine any unconditional GOTO critically. Too many GOTOs leap-frogging over other parts of the program show a lack of directed thinking, making programs run more slowly, and can make them almost impossible to decipher.

It is very good programming practice, as I have suggested earlier, to have each of the main sections of the program (like the one which assigns the variables at the beginning of a run, the one which prints out the board, the one which works out who has won, and so on) in separate subroutines or procedures. The beginning of your program could well look like this:

```
10         REM *NAME OF PROGRAM*
20         REM ASSIGN VARIABLES
30         GOSUB 9000
40         REM PRINT BOARD
50         GOSUB 8000
60         REM HUMAN'S MOVE
70         GOSUB 7000
80         REM COMPUTER'S MOVE
90         GOSUB 6000
100        REM CHECK IF GAME OVER
110        GOSUB 5000
120        GOTO 50
```

As you can see, this ensures that the program actually cycles through a continuous loop over and over again, until the program terminates within the 'CHECK IF GAME OVER' subroutine. You can actually write a series of lines like these before you start writing anything else, and even before you know how you are going to actually perform some of the tasks within the subroutine.

Then you can write the program module by module, making sure that each module works before going onto the next. It is relatively easy to debug a program like this, and far simpler to keep an image of 'where everything is' when you do this, than when you just allow a program to, more or less, write itself.

The listing should be, then, as transparent as you can make it, both for your own present debugging, and for future understanding of what bit carries out what task. The output of the program should also look good. Again, if memory is not a problem, make sure the display is clear and uncluttered. Use blank PRINT lines to space it out, use rules of graphic symbols or whatever to break the screen up into logical sections and so on. Once you have a program working satisfactorily, it is worth spending extra time on the subroutine which controls the display. Here you'll appreciate again the advantage of having all the display handling in one subroutine, as it will be easy to know where to go to enhance the display.

Of course, as we live in a far from ideal world, it is unlikely that every single display command can be contained within one subroutine, but if you aim towards that end, it will make subsequent working upon the program much easier than it might be otherwise.

The 'structured' approach outlined also helps you realise another aim of a good program—to do what you expected it to, every time you run it. You should write a program so that, even if you are not present when a friend decides to run it for the first time, it performs as expected. This means not only, of course,that it is properly debugged, but that the instructions (which can be contained within the ASSIGN VARIABLES subroutine or procedure) are clear and complete.

The user prompts should be clear, so the human operator knows whether to enter a number, a series of numbers, a word, a date, a mixture of letters and numbers, and so on. The program has to assume that the operator is a complete idiot, and that no matter how clearly the instructions and/or user prompts are stated, he or she will attempt to do things the wrong way. A classic example of this is the entering of dates. 'Mug traps', as the routines to reject erroneous input from the operator are called, should be set up to reject a date being entered in a form which the computer cannot understand (such as the month before the day) or which is clearly wrong (such as entering the 31st of February). You should ensure that, no matter what the operator does, the program does not crash or otherwise misbehave. This can happen if the program was expecting a numerical input, and the operator tried to enter a letter or a word, or hit ENTER without entering anything at all. You can get around this by always allowing a string input, going back for another input if the empty string is entered, and taking the ASC, VAL or CODE of the input to turn it into numerical form.

Documentation is an area of programming which is often neglected. It is virtually essential for a program which is intended for publication, and most advisable for long programs which you've written for yourself. At the least, the documentation should include a list of variables, an explanation of the program structure (which should be easy to do if you've followed the 'modular' approach advised earlier), and brief instructions, especially if the program itself does not contain instructions. A sample run showing the kind of inputs, and the nature and layout of the program outputs, is also useful.

Your program should run as quickly as possible. Every time there is a subroutine or GOTO call, the computer must search through the whole program, line by line, to find the specified line number, so placing often used subroutines near the beginning of the program will speed them up fractionally. That is why the instructions are often placed right at the end. You do not want the computer to have to wade through the initialisation and instruction lines every time it has been told to GOTO or GOSUB looking for the destination, or return line number. Use procedures rather than subroutines if you can as these run more quickly than subroutines. Use variables instead of constants. The computer takes a little more time to change a constant into a number than it does for it to look up a value in an array, for example. Define often-used variables first, so they will occupy the early slots in the variables store. The computer will search the store only until it finds the variable it wants, so there is no point in getting it to look at more entries than absolutely necessary.

Finally, and this is by far the best way to test a program you've written, call in a friend and sit him or her in front of the TV, and tell them to press RUN, without you saying anything, and just sit back and watch. If there is any hesitation, or the problem hiccups, you have more work to do.

In summary, then:

- Use REM statements
- Make program listing neat and logical
- Use structured programming techniques, controlling the program through a loop of subroutine calls, or procedures
- Examine unconditional GOTO commands critically, using REPEAT/UNTIL instead if it is appropriate
- Make output displays attractive and clear
- Ensure all user prompts are clear
- Add 'mugtraps' on all user inputs
- Document your programs, even if you just make a list of variables
- Make your program run as quickly as possible
- Test programs by allowing someone unfamiliar with the program to run it.

Dealing with a specific area, most people find that they write games programs more than any other type of program. This may be because they are writing programs for their children, they have a serious interest in a particular type of game (eg wargaming or casino games), or they are children themselves. Whatever the reason for it, there can be no doubt that many games programs which are written, shelved and forgotten as 'boring' could easily be brought back to life with a little work. I will try and give some indicators now, of what can be done in the form of hints and tips.

The Oric, as I'm sure you know, is a fast machine, Without delay loops, the program would run far too fast. Delay loops give the computer the appearance of 'thinking'. At the other end of the scale, do not let your program slow down too much, particularly on long adventure programs. This is a fault that the dear old ZX81 suffers from but it could happen on the Oric, if proper care is not paid to the program's structuring.

If the instructions are detailed and complex, then let the player have the option to read or not to read them. A simple 'DO YOU

WANT INSTRUCTIONS?' together with an INPUT and an IF/THEN will suffice and will save experienced players from having to go through instructions they already know.

The best way to keep a person's interest in a program is to have a certain element of surprise built into the program. Do not let the program become completely predictable, and try and have some random element(s) within the game.

Try and have within the game a degree of difficulty, for example, a 'Breakout' program could have five ball speeds, '1' being beginners level; '2', '3' and '4' being intermediate levels; and '5' for advanced players. You must make sure that the skill levels really do affect the game and that the hardest one is attainable, although admittedly with a lot of hard work. Another way of maintaining a person's interest in a program is to award points, scores or ratings which are related to the speed and difficulty of the program. Another idea is the usage of a 'rank', ie awarding the player a rank such as, 'best in the universe' or 'novice' or 'blithering idiot'.

Points and ranks ensure that a player remains interested in a game for a longer time as he or she are trying to beat their highest score or ranking.

Good use of graphics can initially hold a person's interest but more than pretty pictures are needed to keep the person there. Graphics should be colourful if possible and should be backed up by a sensible use of sound.

Many of the games programs if not all of them in this book, can act as starter or core programs for bigger and better things.

LOADING AND SAVING WITH YOUR ORIC

It's almost ironic that despite the technology being employed in the design and production of a home computer, manufacturers are still having some difficulty achieving a reliable loading and saving system for their computers. They perhaps may not admit it, but the number of people returning their computers to shops saying, 'I cannot get it to load any programs' is far too great.

What this means is that most cassette interfaces are very sensitive. They do work, but only on certain levels with only a small range either side. I'm afraid that the Oric is no different in this department even with its Schmitt trigger, a device used to achieve better results. Once you have found the optimum recording and playback levels, you should not have any further problems.

The first part of the system to discuss is the cassette recorder. An ordinary mono portable recorder, the kind that can be bought anywhere for between £12 and £25, should work perfectly well. I do not know of any recorders that refuse to work, but naturally some perform better than others. Particularly favoured ones include the Ferguson 3T07, the W.H.Smith CCR800, the National Panasonic range of mono portables and the Prinztronic portable. There are also special certified cassette recorders which are designed for use with computers and cost between £35 and £60. However, I know of no one who owns one of these, claiming a better performance than the owner of a standard portable.

There seem to be two camps split between using battery power and using mains power. While battery is free of mains interruptions, it can get rather expensive and you can be sure, under the rules of Murphy's Law, that your batteries will run down just as you are saving a very important program. If you do get a lot of mains 'spikes' then it may be worth considering a mains suppressor

If you have not yet bought a cassette recorder than I suggest you look for the following features: a tone control; a pause button; a DIN socket as well as the usual Jack sockets; a tape counter; and at least a one year guarantee.

Below are a series of hints which may aid your loading and saving. I've assumed that you know how to load and save as shown in the Oric manual.

(1) The volume and tone levels are vital to good recordings. On a scale of one to 1Ø, the volume should be set at between one and three and the tone at between eight and 1Ø. These figures have been personally tested on more than a dozen recorders with 1ØØ% success. However, before you look at any of the other possible remedies, I suggest that you experiment with slightly different levels.

(2) Check your cassette leads. Undo carefully the casing of the DIN plugs and check for a loose or poor connection. If still in doubt, check with a continuity meter.

(3) You must maintain your recorder in perfect condition paying particular attention to cleaning the record, erase and playback heads. If you are unsure of their position, then consult your cassette instruction manual. These heads, as well as the two rollers that transport the tape, should be cleaned regularly with cleaning fluid *not* with a cassette head cleaner tape. These cassette cleaning tapes scrape the tape heads as they clean them. For cleaning the heads, go to a chemists and ask for Isopropyl Alcohol which is then applied to the recorder by some clean cotton buds. You can buy a tape cleaning kit with fluid, mirrors and applicators but really these do the job no better than the Isopropyl Alcohol and cotton bud combination. Speaking of alcohol, vodka will do just as good a job as Isopropyl Alcohol. Apply a little of the fluid onto a cotton bud and just lightly rub over the heads. Use a dry bud to take off any excess fluid and let the heads dry fully (at least 20 minutes) before using.

(4) Always use good quality cassettes, the computer tapes (C12s and C15s) are ideal and not expensive. Check that there is no tape slack, and if there is, fast forward and rewind the tape backwards and forwards several times. This should remove any slack. If you wish to keep a program on a certain tape forever, then break off the tabs on the top of the tape case. Use a screwdriver to do this. The gaps left stop the record head engaging.

(5) *Never* store your tapes near a magnetic field, eg a TV or where there is an uncommonly high temperature or humidity. Storing tapes in these conditions will cause the tapes to deteriorate quickly.

(6) It is always best to erase the tape before you record another program over it. Always write the loading name on the cassette for reference.

(7) Try to keep safe a tape which always loads perfectly and consistently. If, at some time in the future you have a lot of troubles, then you can attempt to load the tape and ascertain whether it is the computer or not that is causing the trouble.

(8) Check that there is nothing in the memory of the computer before loading in a new program. During the recording or playing process do not adjust the volumes or fiddle with any of the leads as this might cause some disturbance.

(9) The two modes the Oric is equipped with are both quite reliable at the time of loading and saving. However, there have been reports of programs corrupting on loading a couple of weeks after being originally written and recorded. I can back these reports up with my own quite traumatic experiences, I lost the programs on several tapes due to corruption. The tapes contained many of the games in this book in their final form and seemed to convert themselves on loading into meaningless rubbish. Thankfully, this is a fairly rare occurrence which I hope will not affect you.

To summarise this chapter, I would say the one thing that is needed is patience. Rome wasn't built in a day and it's possible that your cassette loading/saving troubles won't be solved in a day either. You must try over and over again with a small test program which is loaded and saved at different levels. If you cannot get it to work after trying all of my suggestions, and a few of your friends no doubt, then take it back to the place where you bought it from and seek help there.

SOME USEFUL BOOKS

These books are not Oric specific but don't let that put you off considering them. Oric BASIC is not far from Microsoft BASIC, the format in which many of the programs are written. After typing in some of these programs, you can make various adjustments in the sound and graphics departments to suit your machine and your tastes. Many of the books contain more than just programs, they include instructive comments, information on various aspects of computing and a whole lot more. They are all worth considering.

THE PEANUT BUTTER AND JELLY GUIDE TO COMPUTERS
JERRY WILLIS with D. Smithy and Brian Hyndman.
Dilithium Press ISBN Ø-918398-13-4.
Despite the ridiculous title, this book is a useful addition to your library. It's an American book written back in 1978 so it's a little out of touch with present situations, but as a starter into the world of computing without too much technical detail, it is very good. Due to its age, have a hunt around for it and don't pay more than a few pounds for a copy—you can often find it in reduced departments.

TIM HARTNELL'S GIANT BOOK OF COMPUTER GAMES
TIM HARTNELL, INTERFACE ISBN Ø-9Ø7563-3Ø-9.
Described as the Barbara Cartland of computer books, Tim's giant volume contains many excellent games just crying out to be converted to the Oric. Highlights include Chess, The Bannochburn Legacy and Chairman of the Board.

1ØØ1 THINGS TO DO WITH YOUR PERSONAL COMPUTER
MARK SAWUSCH, TAB BOOKS. ISBN Ø-83Ø6-116Ø-6
A magnificent book, the best source of ideas that I have ever found. I wouldn't part with my copy for anything. If you really think that you have run out of ideas to put your computer to work then consider buying this.

LEARN PASCAL ON YOUR BASIC MICRO
JEREMY RUSTON INTERFACE ISBN Ø-9Ø7563-27-9
Jeremy has contributed much to the British computer scene particularly for the BBC Micro. But in this book every BASIC computer comes under the familiar Ruston treatment. This is one of the best books to use in learning a new computer language and in this book Jeremy takes us through the commands and structure of Pascal. The real bonus with this book is that it includes a Microsoft BASIC compiler which can be typed into your Oric with only a few changes and you can then run a limited version of the language, Pascal.

1ØØ1 BASIC COMPUTER GAMES
DAVID AHL   CREATIVE COMPUTING PRESS
One of the classic works in the computer field, David Ahl has been around forever as far as computing is concerned. A terrific collection of ideas and BASIC games in this book which when you consider it was originally written in 1973 still holds its own today.

65Ø2 ASSEMBLY LANGUAGE PROGRAMMING
LANCE A. LEVANTHAL   OSBORNE/MCGRAW HILL
ISBN Ø-931988-27-6
This is the machine code book if you want to learn the intricacies of 65Ø2 programming. If you manage to plough through this weighty tome and, more importantly, understand it, you will wield tremendous knowledge of the 65Ø2 processor, the heart of your Oric. With that knowledge, you should be able to start writing your own machine code programs.

A GLOSSARY OF TERMS

The computer industry is full of jargon, so much so that a word describing all the jargon, 'buzzword', has now been coined. In this chapter, an explanation of the more widely-used terms is given. The terms are in alphabetical order and have been cross-referenced for ease of use.

For anyone wanting to obtain a more complete list, then two books can be suggested. There is the huge reference book, 'The Computer Dictionary and Handbook' written by Charles and Roger Sippl. This mammoth work of over 9ØØ small-print pages is a magnificent guide and help but only for someone already interested and having some knowledge of computers. The hardback edition costs over £15 so it is only for the dedicated. For the newer enthusiast, I would recommend Dennis Jarrett's excellent 'The Good Computing Book For Beginners'. The title sums it all up, an excellent book for the first-time user, being very witty with plenty of critical, easy-to-understand explanations.

GLOSSARY OF COMPUTER TERMS

Accumulator — part of the computer's logic unit which stores the intermediate results of computations.

Address — a number which refers to a location, generally in the computer's memory, where information is stored.

Algorithm — the sequence of steps used to solve a problem.

Alphanumeric — generally used to describe a keyboard, and signifying that the keyboard has alphabetical and numerical keys. A numeric keypad, by contrast, only has keys for the digits one to nine, with some additional keys for arithmetic operations, much like a calculator.

APL — this stands for Automatic Programming Language, a language developed by Iverson in the early 1960s, which supports a large set of operators and data structures. It uses a non-standard set of characters.

Application software — these are programs which are tailored for a specific task, such as word processing, or to handle mailing lists.

ASCII — this stands for American Standard Code for Information Exchange. This is an almost universal code for letters, numbers and symbols, which has a number between zero and 255 assigned to each of these, such as 65 for the letter 'A'.

Assembler — this is a program which converts another program written in an assembly language (which is a computer program in which a single instruction, such as ADD, converts into a single instruction for the computer) into the language the computer uses directly.

BASIC — stands for Beginner's All-purpose Symbolic Instruction Code, the most common language used on microcomputers. It is easy to learn, with many of its statements being very close to English.

Batch — a group of transactions which are to be processed by a computer in one lot, without interruption by an operator.

Baud — a measure of the speed of transfer of data. It generally stands for the number of bits (discrete units of information) per second.

Benchmark — a test which is used to measure some aspect of the performance of a computer, which can be compared to the result of running a similar test on a different computer.

Binary — a system of counting in which there are only two symbols, 'Ø' and '1' (as opposed to the ordinary decimal system, in which there are ten symbols, 'Ø', '1', '2', '3', '4', '5', '6', '7', '8' and '9'). Your computer 'thinks' in binary.

Boolean Algebra — the algebra of decision-making and logic, developed by English mathematician George Boole, and at the heart of your computer's ability to make decisions.

Bootstrap — a program, run into the computer when it is first turned on, which puts the computer into the state where it can accept and understand other programs.

Buffer — a storage mechanism which holds input from a device such as keyboard, then releases it at a rate which the computer dictates.

Bug — an error in a program.

Bus — a group of electrical connections used to link a computer with an ancillary device, or another computer.

Byte — the smallest group of bits which makes up a computer word. Generally a computer is described as being 'eight bit' or '16 bit', meaning the word consists of a combination of eight or sixteen zeros or ones.

Central Processing Unit (CPU) — the heart of the computer, where arithmetic, logic and control functions are carried out.

Centronics — a type of interface (see 'Interface'). The Centronics interface allows you to hook up printers, modems and many other peripherals to a computer. Your Oric has a Centronics Interface.

Character code — the number of ASCII (see 'ASCII') which refers to a particular symbol, such as 32 for a space and 65 for the letter 'A'.

COBOL — this stands for COmmon Business Orientated Language, a standard programming language, close to English, which is used primarily for business.

Compiler — a program which translates a program written in a high level (human-like) language into a machine language which the computer understands directly.

CP/M — stands for Control Program/Microcomputer, an almost universal disc operating system developed and marketed by Digital Research, Pacific Grove, California.

Cursor — a marker on the VDU screen which indicates where the next character will be displayed.

Data — a general term for information processed by a computer.

Database — a collection of data, organised to permit rapid access by computer.

Debug — to remove bugs (errors) from a program.

Disc — a magnetic storage medium (further described as a 'hard disc', 'floppy disc' or even 'floppy') used to store computer information and programs. The discs resemble, to a limited extent, 45 rpm sound records, and are generally eight, five and a quarter, or three inches in diameter. Smaller 'micro-discs' are also available for some systems.

Documentation — the written instructions and explanations which accompany a program.

DOS — this stands for Disc Operating System (and generally pronounced 'doss'), the versatile program which allows a computer to control a disc system.

Dot-matrix printer — a printer which forms the letters and symbols by a collection of dots, usually on an eight by eight, or seven by five, grid.

Double-density — adjective used to describe discs when recorded using a special technique which, as the name suggests, doubles the amount of storage the disc can provide.

Dynamic memory — computer memory which requires constant recharging to retain its contents.

EPROM — this stands for Erasable Programmable Read Only Memory, a device which contains computer information in a semi-permanent form, demanding sustained exposure to ultra-violet light to erase its contents.

Error messages — information from the computer to the user, sometimes consisting only of numbers or a few letters, but generally of a phrase (such as 'Out of memory') which points out a programming or operational error which has caused the computer to halt program executions.

Field — a collection of characters which form a distinct group, such as an identifying code, a name or a date; a field is generally part of a record.

File — a group of related records which are processed together, such as an inventory file or a student file.

Firmware — the solid components of a computer system are often called the 'hardware', the programs, in machine-readable form on disc or cassette, are called the 'software', and programs which are hard-wired into a circuit, are called 'firmware'. Firmware can be altered, to a limited extent, by software in some circumstances.

Flag — this is an indicator within a program, with the 'state of the flag' (ie the value it holds) giving information regarding a particular condition.

Floppy disc — see 'Disc'.

Flowchart — a written layout of program structure and flow, using various shapes, such as a rectangle with sloping sides for a computer action, and a diamond for a computer decision, is called a flow chart. A flowchart is generally written before any lines of program are entered into the computer.

FORTRAN — a high level computer language, generally used for scientific work (from FORmula TRANslation).

Gate — a computer 'component' which makes decisions, allowing the circuit to flow in one direction or another, depending on the conditions to be satisfied.

GIGO — acronym for 'Garbage In Garbage Out', suggesting that if rubbish or wrong data is fed into a computer, the result of its processing of such data (the output) must also be rubbish.

Global — a set of conditions which affects the entire program is called 'global', as opposed to 'local'.

Graphics — a term for any output of computer which is not alphanumeric, or symbolic.

Hard copy — information dumped to paper by a printer.

Hardware — the solid parts of the computer (see 'Software' and 'Firmware').

Hexadecimal — a counting system much beloved by machine code programmers because it is closely related to the number storage methods used by computers, based on the number 16 (as opposed to our 'ordinary' number system which is based on 10).

Hex pad — a keyboard, somewhat like a calculator, which is used for direct entry of hexadecimal numbers.

High-level languages — programming languages which are close to English. Low-level languages are closer to those which the computer understands. Because high-level languages have to be compiled into a form which the computer can understand before they are processed, high-level languages run more slowly than do their low-level counterparts.

Input — any information which is fed into a program during execution.

I/O — this stands for Input/Output port, a device the computer uses to communicate with the outside world.

Instruction — an element of programming code, which tells the computer to carry out a specific task. An instruction in assembler language, for example, is ADD which (as you've probably guessed) tells the computer to carry out an addition.

Interface — apart from being the name of my publisher, an interface is a boundary between systems. The most likely use of the term that you will come into contact with is an interface being some form of physical connection between a computer and another item such as a printer. This other item is often known as a peripheral item (see 'Peripheral).

Interpreter — converts the high-level ('human-understandable') program into a form which the computer can understand.

Joystick — an analogue device which feeds signal into a computer which is related to the positon which the joystick is occupying; generally used in games programs.

Kilobyte — the unit of memory measurement; one kilobyte (generally abbreviated as K) equals 1,024 bytes.

Light pen — a handheld light-sensing device, shaped like a pen, which can be used with your computer for menu-selection, games playing, artwork, etc. A light pen detects the brightness of a portion of a screen.

Line printer — a printer which prints a complete line of characters at one time.

Low-level language — a language which is close to that used within the computer (see 'High-level language').

Machine language — the step below a low-level language; the language which the computer understands directly.

Mainframe — the term for ‘giant’ computers such as the IBM 3Ø7. Computers are also classed as minicomputer and microcomputer (such as the computer you own).

Memory — the device or devices used by a computer to hold information and programs being currently processed, and for the instruction set fixed within a computer which tells it how to carry out the demands of the program. There are basically two types of memory (‘RAM’ and ‘ROM’).

Microprocessor — the ‘chip’ which lies at the heart of your computer. This does the ‘thinking’.

Modem — this stands for MOdulator/DEModulator, and is a device which allows one computer to communicate with another via the telephone.

Monitor — (a) a dedicated television-screen for use as a computer display unit, contains no tuning apparatus; (b) the information within a computer which enables it to understand and execute program instructions.

Motherboard — a unit, generally external, which has slots to allow additional ‘boards’ (circuits) to be plugged into the computer to provide facilities (such as high-resolution graphics, or ‘robot control’) which are not provided with the standard machine.

Mouse — a control unit, slightly smaller than a box of cigarettes, which is rolled over the desk, moving an on-screen cursor in parallel to select options and make decisions within a program. ‘Mouses’ work either by sensing the action of their wheels, or by reading a grid pattern on the surface upon which they are moved.

Network — a group of computers working together.

Numeric pad — a device primarily for entering numeric information into a computer, similar to a calculator.

Octal — a numbering system based on eight (using the digits ‘Ø’, ‘1’, ‘2’, ‘3’, ‘4’, ‘5’, ‘6’ and ‘7’).

On-line — device which is under the direct control of the computer.

Operating system — this is the 'big boss' program or series of programs within the computer which controls the computer's operation, doing such things as calling up routines when they are needed and assigning priorities.

Output — any data produced by the computer while it is processing, whether this data is displayed on the screen or dumped to the printer, or is used internally.

Pascal — a high level language, developed in the late 1960s by Niklaus Wirth, which encourages disciplined, structured programming.

Peripheral — the name given to any input or output device which can connect up (usually through an interface) to the CPU and the memory of a computer.

Port — an output or input 'hole' in the computer, through which data is transferred.

Program — the series of instructions which the computer follows to carry out a predetermined task.

PILOT — a high level language, generally used to develop computer programs for education.

RAM — this stands for Random Access Memory, and is the memory on board the computer which holds the current program. The contents of RAM can be changed, while the contents of ROM (Read Only Memory) cannot be changed under software control.

Real-time — when a computer event is progressing in line with time in the 'real world', the event is said to be occurring in real time. An example would be a program which showed the development of a colony of bacteria which developed at the same rate that such a real colony would develop. Many games, which require reactions in real time, have been developed. Most 'arcade action' programs occur in real time.

Refresh — the contents of dynamic memories (see 'Memory') must receive periodic bursts of power in order for them to maintain their contents. The signal which 'reminds' the memory of its contents is called the refresh signal.

Register — a location in computer memory which holds data.

Reset — a signal which returns the computer to the point it was in when first turned on.

ROM — see 'RAM'.

RS-232 — a standard serial interface (defined by the Electronic Industries Association) which connects a modem and associated terminal equipment to a computer.

S-1ØØ bus — this is also a standard interface (see 'RS-232') made up of 1ØØ parallel common communication lines which are used to connect circuit boards within micro-computers.

SNOBOL — a high level language, developed by Bell Laboratories, which uses pattern recognition and string manipulation.

Software — the program which the computer follows (see 'Firmware').

Stack — a temporary store used by the CPU for return addresses, etc. which are accessed on a last in first out (LIFO) basis.

Subroutine — a block of code, or program, which is called up a number of times within another program.

Syntax — as in human languages, the syntax is the structure rules which govern the use of a computer language.

Systems software — sections of code which carry out administrative tasks, or assist with the writing of other programs, but which are not actually used to carry out the computer's final task.

Thermal printer — a device which prints on heat-sensitive paper. Although thermal printers are quieter than other printers, the output is not always easy to read, nor is the used paper easy to store.

Time-sharing — this term is used to refer to a large number of users, on independent terminals, making use of a single computer, which divides its time between the users in such a way that each of them appears to have the 'full attention' of the computer.

Turnkey system — a computer system (generally for busines use) which is ready to run when delivered, needing only the 'turn of a key' to get it working.

VDU — a Visual Display Unit is some form of screen, such as a monitor or the humble family television. The computer outputs onto the screen.

Volatile memory — a memory device which loses its contents when the power supply is cut off (see 'Memory', 'Refresh', 'ROM' and 'RAM').

Word processor — a dedicated computer (or a computer operating a word-processing program) which gives access to an 'intelligent typewriter' with a large range of correction and adjustment features.

If you're tired of drab, old programs, and had enough of paying far too much for cassette software, then this book is for you. Contained within are over 35 exciting, original games specially written for and stretching the capabilities of your Oric computer.

Play 'Houston, We Have A Problem', 'Laser War' and 'The Forbidden Caves' while trying to outwit the computer in 'Four By Four'. Attempt to save your helpless band of athletes in 'Jogger' and then go for some number nibbling in 'Digit Muncher'.

From adventuring to the arcade, from gambling to golf, in fact, whatever your favourite type of game, this book has it. In addition to these programs, there is a chapter to help you write your own software, a collection of routines to aid you and a comprehensive computer glossary.

This then, is a dynamic book that will help you obtain maximum enjoyment from your Oric. After all, who could resist a game of 'Eatie Sweetie'?

Dynamic Games for your Oric

Gifford Interface