TIM HARTNELL'S

QL GAMES COMPENDIUM



TIM HARTNELL'S QL GAMES COMPENDIUM

INTERFACE PUBLICATIONS

LONDON AND MELBOURNE

TIM HARTNELL'S QL GAMES COMPENDIUM

First published in the UK by:
Interface Publications,
9–11 Kensington High Street,
London W8 5NP.

Copyright © Tim Hartnell
First printing August 1984

ISBN 0 947695 04 4

The programs in this book have been included for their instructional value. They have been tested with care, but are not guaranteed for any particular purpose. While every care has been taken, the publishers cannot be held responsible for any running mistakes which may occur.

ALL RIGHTS RESERVED

No use whatsoever may be made of the contents of this volume – programs and/or text – except for private study by the purchaser of this volume, without the prior written permission of the copyright holder.
Reproduction in any form or for any purposes is forbidden.

Books published by Interface Publications are distributed in the UK by WHS Distributors, St John's House, East Street, Leicester LE1 6NE (0533 551196) and in Australia and New Zealand by PITMAN PUBLISHING. Any queries regarding the contents of this volume should be directed by mail to Interface Publications, 9–11 Kensington High Street, London W8 5NP.

Typesetting by Verhoef Graphics, Clayton, Melbourne, Australia.
Printed and Bound by Heffers Printers Ltd, Cambridge, England.

CONTENTS

Introduction 1

Adventure 3
- Tryst with Memphistopheles 5
- Magician's Maze 33

Artificial Intelligence 45
- Eliza 51
- Poetry Engine 63
- Mindboz 71

Giggle Palace 77
- Nimbo 79
- Simon 83
- Robot Minefield 87
- Twenty One 93

Simulations 97
- Stockbroker 99
- Life 117

Board Games 131
- Checkers 133
- Gomoku 141
- Qthello 149

Creating Moving Graphics 165
- The Secret of Animation 165
- Bouncing Ball 167

Going to Bat 167
Orbit 173
Rex 174
V-wing Space Battle 179
QL Racer 183

Deduction and Perception 191
Memomuse 193
Lexicon 201
Detective Q 207

Write Your Own 219
Further Reading 231
Glossary of Computer Terms 235

INTRODUCTION

Does playing games with your QL make you feel just a little bit guilty? Do you feel the need to pursue the game-playing side of your relationship with your computer in a clandestine way?

Don't let anyone make you feel even the slightest bit guilty over the use of your QL for entertainment. The computer is a perfect companion. Patient, forgiving, gracious in victory, pleasant in defeat. What more can you ask for? If you're not winning a game, you can simply start again. That's like knocking over the chess board every time you lose your first pawn. Try that with a few of your human game opponents, and see how quickly they run.

The best thing about game playing with a computer is that it can be great fun. Of course, the games and simulations in this book may teach you a lot about such things as the management of scarce resources and may enhance your decision-making abilities, and help you concentrate a little better. But these are side effects.

If a game helps you develop some skills as an extra benefit, fine. But don't feel you must claim that the side benefits are the reason for running these programs. Don't even tell yourself that you're doing anything but having fun.

Once you've run these programs as listed in this book,

you may well wish to add to your enjoyment of them by modifying them in some way. Change the display, alter the sound, get the QL to use your name from time to time (without having to start each game with a 'What is your name?') line, and so on. By all means do this. Do whatever you like to add a personal touch to them. While making the changes, and working through my listings to find out how I ensured the program did what I wanted it to do, you'll probably find you're strengthening your programming skills. And that can't be bad.

Good gaming,

Tim Hartnell,
London, 1984.

Tim Hartnell is one of the most widely-published authors of computer books in the world. Companion volumes to this one include *Exploring Artificial Intelligence on your QL* and *Tim Hartnell's QL Handbook*.

ADVENTURE

It began as wargaming, locked solidly to the earth, to sweat and blood, and the here and now. It became, quite literally, magic.

The transformation from the gritty reality of wargaming to the imagineered frontiers of fantasy began in the sixties with the release of Tolkien's 'Lord of the Rings' in paperback. Around the world, imaginations were caught by this epic adult fairy tale.

In 1974, a group of wargamers in Lake Geneva, Wisconsin — led by Gary Gygax — published a small booklet called 'Chainmail'. Instead of containing rules (like most wargaming publications did at the time) governing such things as the reloading rate of French cannon, 'Chainmail' produced a primitive code to control magic and monsters. The hunger for fantasy triggered by Tolkien was finding appeasement.

Dave Arneson, a wargamer in Minneapolis-St. Paul, Minnesota, created an underground dungeon for his battle field, and used the 'Chainmail' rules to control the battles which took place there. Gygax and Arneson worked together to develop a set of 'dungeon rules' which were turned down by several of the established game-publishing companies. Gygax set up his own company, Tactical Studies Rules, to market the rules as 'Dungeons and Dragons' and the game's explosive expansion began.

To play 'D & D' (as it became known to aficiandos) demanded a 'Dungeon Master' to run the show, and some friends to share the delights of the dungeon. It seemed inevitable that computers would be brought into the picture, to remove the prerequisites of other humans.

In the late sixties, two programmers Crowther and Woods produced a huge program called Adventure, which is considered by many to be the first computer adventure. It was written in Fortran on the mainframe at Stanford University. Adventure spread, in various incarnations, across the States and versions of it are still available in most mainframe libraries. Scott Adams boiled it down for the TRS-80 and other companies followed. There are now hundreds of adventure programs available.

Although our two adventures are considerably less complicated than most commercially-produced adventures, the skills you develop solving these adventures will stand you in good stead if you decide to tackle the much more complex ones. And if you are not yet an addict of this unique, computer-driven, leisure activity, 'Magician's Maze' and 'Tryst with Mephistopheles' may awaken the addiction.

TRYST WITH MEPHISTOPHELES

Like Johann Faust was alleged to have done some 500



years ago, you have made a pact with the devil. In return for the freedom to spend as much time as you

like for the rest of your life playing games on your QL, you have bargained away your soul.

Twenty minutes after signing this pact, in your own blood, you realise that the devil had probably done better out of the deal than you had. Logging on Micronet, you summon His Satanic Majesty and explain that it was all a bit of a mistake, and ask if there is any way of undoing the pact so carelessly signed.

H.S.M. thinks about it for a while, then says he will set a difficult task for you to complete. Succeed, and you can have your soul back in perpetuity. Fail, and the original bargain stands.

Your task is as follows. The devil will transport you into the library of The Evil Tower of Kaskaskia, which was one of his royal residences, four hundred years ago. You have to make your way around the tower, looking for the room used by H.S.M. for his private quarters. In this room you will find a magical locket. Get safely out of the tower with the locket, and your soul is yours.

It seems reasonable enough. However, you have doubts. Devils, after all, have a reputation for trickery when dealing with humans. You ask for some advice on how to tackle this task. The devil clicks his tongue in exasperation, waves his arm in the air, and a familiar — in the form of a black raven — appears on his shoulder.

Here is the advice you hear from the raven:

"You are on your own on this adventure. The idea is to get to the end of the tower — alive — carrying the locket. The score you get at the end is related to the number of monsters you kill while wandering through the tower, how much strength and other characteristics you manage to conserve, how much treasure you pick up, and whether or not you manage to find the locket.

"You tell your trusty QL what you want to do with two-word commands, such as GO WEST, GET AXE or DROP DIAMOND. These can be abbreviated to the first three letters of each word, so DRO DIA is acceptable for DROP DIAMOND.

"You are encouraged to make a map as you wander around the Tower, so that you will be able to retrace your steps if you get lost. The words the computer understands include (but are not limited to) GO, GET, DROP, FIGHT and KILL. The computer will tell you if it does not understand your instructions. If you want to give up at any time, you can enter QUIT.

"The Tower floor plan does not change from run to run, although the location of the monsters (which include the dreaded 'ORRIBLE ORC) is different each game and many of the treasures and weapons will be lying in different rooms."

To further aid you, the raven points out that, at the start of the game, you have six attributes: strength, charisma, dexterity, intelligence, wisdom and constitution. These fluctuate throughout the game, depend-

ing on what you are doing. They will fall if you become involved in a fight and are wounded. As well, you'll use up some of the attributes just in surviving from hour to hour. If any one of the attributes drops to zero, the adventure is over. This means you do not have unlimited time to wander around the Tower at your leisure. Waste time, and you will probably end up dead.

You fight the various monsters in terms of your attributes, and those of the monster. You need only select two attributes from your six to actually take part in a fight. This will become clear when you run the program on your QL.

After many trials, if you are clever (and brains count for far more than luck), you'll make it out into the open air and freedom. You can only get the maximum score by completing the adventure. Those weak of heart can simply quit when the going gets too rough.

When you're ready to take on Mephistopheles' quest, enter the following listing:

```
10 REMark Tryst with Mephistopheles
20  REMark APPROX. 20K
30  initialise
40 REPeat CYCLE
50  GAME
60 END REPeat CYCLE
70 REMark ---------------
80 DEFine PROCedure GAME
90 delay
100 CLS
```

```
110 IF ro=16 THEN qu=1:GO TO 5140
120 color
130   room_descriptions
140 IF a(ro,5)<>0 OR a(ro,6)<>0 OR a(ro,7)
    <>0 THEN objects
150 IF ro=12 AND lock=0 THEN PRINT
    "   THE DOOR IS LOCKED..."
160  color
170 IF a(ro,8)<>0 THEN
180  PRINT "LOOK OUT!"
190  PRINT "THERE IS AN ";M$(a(ro,8))/"HERE"
200   IF RND>.7 AND a(ro,8)<>1 THEN
210    PRINT "THE ";M$(a(ro,8));" ATTACKS!"
220    KW=1:FIGHT:GO TO 1550
230  END IF
240 END IF
250 IF st<1 THEN
260  st=0
270   ELSE IF RND>.84 THEN st=st-1
280 END IF
290 IF ch<1 THEN
300  ch=0
310   ELSE IF RND>.84 THEN
320    ch=ch-1
330 END IF
340 IF de<1 THEN
350  de=0
360   ELSE IF RND>.84 THEN
370    de=de-1
380 END IF
390 IF it<1 THEN
400  it=0
410   ELSE IF RND>.84 THEN
420    it=it-1
430 END IF
```

```
440 IF wi<1 THEN
450  wi=0
460   ELSE IF RND>.84 THEN
470    wi=wi-1
480 END IF
490 IF co<1 THEN
500  co=0
510   ELSE IF RND>.84 THEN
520    co=co-1
530 END IF
540 PRINT \"Your attributes are:"
550 PRINT "   Strength - ";st;
            "  Charisma - ";ch
560 PRINT "   Dexterity - ";de;
            "  Intelligence - ";it
570 PRINT "   Wisdom - ";wi;
            "  Constitution - ";co
580 IF st*ch*de*it*wi*co=0 THEN
590  PRINT "Unfortunately, you are exhausted"
600  PRINT "so this adventure must end"
610  qu=2:BRAG=1
620  end_of_game:STOP
630 END IF
640 flag=0
650 FOR j=1 TO 5
660  IF p(j)<>0 THEN flag=1
670 END FOR j
680 IF flag=0 THEN GO TO 780
690 cash=0
700  music
710 PRINT \"You are carrying:"
720 FOR j=1 TO 5
730  color
740 IF p(j)<>0 THEN PRINT "   ";o$(p(j)):
     cash=cash+v(p(j))
```

```
750 END FOR j
760  color
770 IF cash>0 THEN PRINT "    Total
        value - $";cash
780 KW=0
790 IF ro=7 AND c$="ACI" THEN ACID=1
800 IF ro=15 AND c$="LOC" THEN LOCKET=1
810  delay
820  color
830 IF INKEY$<>"" THEN GO TO 830
840  INPUT\"WHAT DO YOU WANT TO DO? ";A$
850 IF A$="" THEN GO TO 840
860 IF CODE(A$)>CODE("Z") THEN
870   FLASH 1
880  PRINT ,"Engage ";:color
890  PRINT "CAPS LOCK":FLASH 0
900  delay:GO TO 810
910 END IF
920 AS=CODE(A$)
930 M=LEN(A$):IF M<7 THEN A$=A$&" ":GO TO 930
940 B$=A$(1 TO 3)
950 IF B$="HEL" THEN
960  PRINT "YOU MUST BE JOKING!"
970  GO TO 1540
980 END IF
990 IF B$="QUI" THEN
1000  qu=4
1010  end_of_game:STOP
1020 END IF
1030 N=1
1040 WORD=0
1050 IF A$(N)=" " THEN c$=A$(N+1 TO N+3):
      WORD=1
1060 IF WORD<>1 THEN GO TO 1090
1070 IF WORD=1 AND c$(1)<>" " THEN GO TO 1190
1080 IF WORD=1 THEN GO TO 1100
```

```
1090 IF N<M THEN N=N+1:GO TO 1040
1100 IF RND>.5 THEN GO TO 1130
1110 PRINT " BY ITSELF, ";A$
1120 PRINT "  CANNOT BE ACTED ON":GO TO 840
1130 PRINT A$;" IS JUST ONE WORD"
1140 PRINT "   I NEED TWO...":GO TO 840
1150 IF ro=2 AND c$="ACI" AND ACID=0 THEN
1160  PRINT "YOU'RE NOT CARRYING IT"
1170  GO TO 1540
1180 END IF
1190 IF ro=2 AND B$="GO " AND c$(1)="E" THEN
1200  PRINT "YOU WILL BE EATEN ALIVE IF YOU"
1210  PRINT "TRY TO CROSS THE STREAM"
1220  GO TO 1540
1230 END IF
1240 IF ro=2 AND ACID=1 AND c$="ACI" THEN
1250  PRINT "THE FISH ARE DYING, DYING..."
1260   delay
1270  PRINT "YOU MAY CROSS SAFELY..."
1280  ro=3
1290  GO TO 1540
1300 END IF
1310 IF B$="KIL" OR B$="FIG" THEN
1320  KW=1
1330  FIGHT
1340 END IF
1350 IF ro=12 AND lock=0 AND (c$="WES"
        OR c$="EAS") THEN
1360  PRINT "   THE DOOR IS LOCKED!"
1370  GO TO 1540
1380 END IF
1390 IF B$="GO " AND a(ro,8)<>0 AND
        RND>.5 THEN
1400  PRINT "NO, YOU MUST FIGHT THE ";
        M$(a(ro,8))
```

```
1410  FIGHT
1420  GO TO 1550
1430 END IF
1440 IF B$="GO " THEN KW=1:MOVING
1450 IF B$="GET" THEN KW=1:GET_OBJECT
1460 IF B$="DRO" THEN KW=1:DROP
1470 IF B$="UNL" THEN KW=1:UNLOCK
1480 IF KW=1 THEN GO TO 1540
1490    music
1500 RR=RND(1 TO 3)
1510 IF RR=1 THEN PRINT "IT WOULD NOT
       BE WISE TO ";A$
1520 IF RR=2 THEN PRINT "ONLY A FOOL
      WOULD TRY TO ";A$
1530 IF RR=3 THEN PRINT "I DON'T
      UNDERSTAND '";A$;"'"
1540  delay
1550 END DEFine GAME
1560 REMark --------------
1570 DEFine PROCedure MOVING
1580 IF ro<>1 THEN GO TO 1810
1590 IF BRIBE=1 THEN GO TO 1810
1600 PRINT "THE EVIL ENTANGLER INSISTS ON A"
1610 PRINT "OF THE WHITE GOLD RING TO LET"
1620 PRINT "    YOU PASS..."
1630 flag=0
1640 FOR j=1 TO 5
1650  IF p(j)=1 THEN flag=j
1660 END FOR j
1670 IF flag>0 THEN GO TO 1740
1680 PRINT "YOU DO NOT HAVE IT!"
1690 PRINT "THE EVIL ENTANGLER SENDS YOU"
1700 PRINT "BACK TO THE ARMOURY..."
1710  delay
1720 ro=8
1730 GO TO 2020
```

```
1740 delay
1750 PRINT \"LUCKY FOR YOU THAT YOU HAVE IT!"
1760 p(flag)=0
1770 a(1,8)=0
1780 BRIBE=1
1790 PRINT "THE EVIL ENTANGLER MUTTERS HIS"
1800 PRINT "      THANKS, AND VANISHES..."
1810 c$=c$(1)
1820 IF c$="N" AND a(ro,1)=0 THEN
1830  PRINT "You cannot go that way!"
1840  GO TO 2020
1850 END IF
1860 IF c$="S" AND a(ro,2)=0 THEN
1870  PRINT "There is no exit south"
1880  GO TO 2020
1890 END IF
1900 IF c$="E" AND a(ro,3)=0 THEN
1910  PRINT "I see nowhere east to go"
1920  GO TO 2020
1930 END IF
1940 IF c$="W" AND a(ro,4)=0 THEN
1950  PRINT "Even you cannot walk through
       walls!"
1960  GO TO 2020
1970 END IF
1980 IF c$="N" THEN ro=a(ro,1)
1990 IF c$="S" THEN ro=a(ro,2)
2000 IF c$="E" THEN ro=a(ro,3)
2010 IF c$="W" THEN ro=a(ro,4)
2020 END DEFine MOVING
2030 REMark --------------
2040 DEFine PROCedure GET_OBJECT
2050 flag=0
2060 FOR j=1 TO 5
2070  IF p(j)<>0 THEN flag=flag+1
2080 END FOR j
2090 IF flag=5 THEN
```

```
2100  PRINT "You are already carrying your"
2110  PRINT "maximum of five objects"
2120  GO TO 2350
2130 END IF
2140 IF a(ro,5)=0 AND a(ro,6)=0 AND
        a(ro,7)=0 THEN
2150  PRINT "I see nothing to pick up"
2160  GO TO 2350
2170 END IF
2180 D$="":E$="":F$=""
2190 D$=o$(a(ro,5))(1 TO 3)
2200 E$=o$(a(ro,6))(1 TO 3)
2210 F$=o$(a(ro,7))(1 TO 3)
2220 IF c$=D$ THEN N=a(ro,5):a(ro,5)=0
2230 IF c$=E$ THEN N=a(ro,6):a(ro,6)=0
2240 IF c$=F$ THEN N=a(ro,7):a(ro,7)=0
2250 j=1
2260 IF p(j)=0 THEN p(j)=N:GO TO 2280
2270 IF j<5 THEN j=j+1:GO TO 2260
2280 BEEP 10000,100:BEEP 10000,250
2290 BEEP 10000,200:BEEP 10000,100
2300 FOR wait=1 TO 100:END FOR wait
2310 BEEP 10000,100:BEEP 10000,250
2320 PRINT "  )--> YOU NOW HAVE THE ";
2330 color
2340 PRINT o$(N)
2350 END DEFine GET_OBJECT
2360 REMark --------------
2370 DEFine PROCedure DROP
2380 flag=0
2390 FOR j=1 TO 5
2400  IF p(j)<>0 THEN flag=1
2410 END FOR j
2420 END IF
2430 IF flag=0 THEN
```

```
2440  PRINT "You are not carrying anything"
2450  GO TO 2800
2460 END IF
2470 IF a(ro,5)<>0 AND a(ro,6)<>0 AND
      a(ro,7)<>0 THEN
2480  PRINT "This room already holds its"
2490  PRINT " maximum number of objects"
2500  GO TO 2800
2510 END IF
2520 D$="":d=0
2530 FOR j=1 TO 8
2540  IF o$(j)(1 TO 3)=c$ THEN
2550   D$=o$(j)
2560   d=j
2570  END IF
2580 END FOR j
2590 IF D$="" THEN
2600  PRINT "How can you when you're"
2610  PRINT "  not holding it?"
2620  GO TO 2800
2630 END IF
2640 FOR j=1 TO 5
2650  IF p(j)=d THEN p(j)=0
2660 END FOR j
2670 IF a(ro,5)=0 THEN
2680  a(ro,5)=d
2690  GO TO 2790
2700 END IF
2710 IF a(ro,6)=0 THEN
2720  a(ro,6)=d
2730  GO TO 2790
2740 END IF
2750 IF a(ro,7)=0 THEN
2760  a(ro,7)=d
2770  GO TO 2790
2780 END IF
```

```
2790 PRINT "YOU HAVE DROPPED THE ";D$
2800 END DEFine DROP
2810 REMark ---------------
2820 DEFine PROCedure FIGHT
2830 IF ro=1 THEN
2840  PRINT "THE ENTANGLER WILL HAVE
           NONE OF THAT"
2850  GO TO 4760
2860 END IF
2870 IF a(ro,8)<>0 THEN GO TO 2930
2880 RR=RND
2890 music
2900 IF RR<.5 THEN PRINT "There is
              nothing to fight here!"
2910 IF RR>=.5 THEN PRINT "You can't
                   fight empty air!"

2920 GO TO 4760
2930 g$=M$(a(ro,8))
2940 workout
2950 s1=roll
2960 workout
2970 h1=roll
2980 workout
2990 d1=roll
3000 workout
3010 i1=roll
3020 workout
3030 w1=roll
3040 workout
3050 c1=roll
3060  DOLINE
3070 PRINT  "YOUR OPPONENT IS AN ";g$
3080 MT=0:HT=0
3090 FF=s1*(RND(1 TO 6))
```

```
3100  DOLINE
3110 PRINT "THE ";g$;"'S DANGER LEVEL IS ";FF
3120  DOLINE
3130  delay
3140 FOR j=1 TO 5
3150  T(j)=0
3160 IF p(j)=6 THEN
3170  PRINT "You axe could be handy":T(j)=6
3180 END IF
3190 IF p(j)=5 THEN
3200  PRINT "Your skill with the sword"
3210  PRINT "may stand you in good stead"
3220  T(j)=5
3230 END IF
3240 END FOR j
3250 flag=0
3260 FOR j=1 TO 5
3270  IF T(j)<>0 THEN flag=flag+1
3280 END FOR j
3290 IF flag=0 THEN
3300  PRINT "YOU MUST FIGHT THE ";g$
3310  PRINT "WITH YOUR BARE HANDS"
3320  GO TO 3540
3330 END IF
3340 IF flag>1 THEN GO TO 3410
3350 FOR j=1 TO 5
3360  IF T(j)<>0 THEN flag=T(j)
3370 END FOR j
3380 PRINT "YOU MUST FIGHT WITH YOUR ";o$(flag
3390 FF=INT(FF*2/flag)
3400 GO TO 3540
3410 PRINT "CHOOSE YOUR WEAPON:"
3420 FOR j=1 TO 5
3430   color
3440  IF p(j)<>0 THEN PRINT j;" - ";o$(j)
```

```
3450 END FOR j
3460 INPUT "Enter the number to choose ";j
3470 IF p(j)=0 THEN
3480  PRINT "YOU DO NOT HAVE THE ";o$(j)
3490  GO TO 3460
3500 END IF
3510 PRINT "SO YOU CHOOSE TO FIGHT"
3520 PRINT "  WITH THE ";o$(j)
3530 FF=INT(FF*2/j)
3540  delay
3550  DOLINE
3560 PRINT "THE ";g$;"'S ATTRIBUTES ARE:"
3570 PRINT "1: Strength ";s1;
                      "   2: Charisma ";h1
3580 PRINT "3: Dexterity ";d1;
                    "  4: Intelligence ";i1
3590 PRINT "5: Wisdom ";w1;
                "     6: Constitution ";c1
3600  DOLINE
3610 PRINT "Your attributes are:"
3620 PRINT "1: Strength ";st;
                      "   2: Charisma ";ch
3630 PRINT "3: Dexterity ";de;
                    "  4: Intelligence ";it
3640 PRINT "5: Wisdom ";wi;
                "     6: Constitution ";co

3650  music
3660  color
3670 PRINT \"Which attributes will you fight"
3680 PRINT "        with (select two)?";
3690 INPUT z:INPUT q
3700 IF z<1 OR q<1 OR z>6 OR q>6 OR z=q THEN
3710  color
3720  PRINT "DO NOT FOOL AROUND WHEN A"
3730  PRINT g$;" IS IN THE ROOM!"
```

```
3740  GO TO 3690
3750 END IF
3760  color
3770 IF z=1 OR q=1 THEN
3780  MT=MT+s1:HT=HT+st
3790 END IF
3800 IF z=2 OR q=2 THEN
3810  MT=MT+h1:HT=HT+ch
3820 END IF
3830 IF z=3 OR q=3 THEN
3840  MT=MT+d1:HT=HT+de
3850 END IF
3860 IF z=4 OR q=4 THEN
3870  MT=MT+i1:HT=HT+it
3880 END IF
3890 IF z=5 OR q=5 THEN
3900  MT=MT+w1:HT=HT+wi
3910 END IF
3920 IF z=6 OR q=6 THEN
3930  MT=MT+c1:HT=HT+co
3940 END IF
3950  music
3960  color
3970 IF HT=MT THEN
3980  PRINT "  You are evenly matched"
3990  GO TO 4080
4000 END IF
4010 PRINT "It looks like the odds"
4020 PRINT "are in favour of ";
4030 IF HT>MT THEN
4040  PRINT "you"
4050   ELSE PRINT "the ";g$
4060 END IF
4070  DOLINE
4080 k=RND(1 TO 7)
```

```
4090  delay
4100 PRINT "THE ";g$;" - ";MT
4110 PRINT "       YOU - ";HT
4120  color
4130  DOLINE
4140 IF k=0 THEN
4150  PRINT "  You struck a splendid blow!"
4160  MT=MT-1
4170 END IF
4180 IF k=1 THEN
4190  PRINT "  THE ";g$;" STRIKES OUT"
4200  HT=HT-3:st=st-1
4210  ch=ch-1
4220 END IF
4230 IF k=2 THEN
4240  PRINT "YOU DRAW THE ";g$;"'S BLOOD"
4250  MT=MT-1
4260 END IF
4270 IF k=3 THEN
4280  PRINT ".....You are wounded....."
4290  HT=HT-RND(-1 TO 1)
4300  de=de-1
4310 END IF
4320 IF k=4 THEN
4330  PRINT "THE ";g$;" IS TIRING"
4340  MT=MT-1
4350 END IF
4360 IF k=5 THEN
4370  PRINT "   ...You are bleeding..."
4380  HT=HT-2:wi=wi-1:co=co-1
4390 END IF
4400 IF k=6 THEN
4410  PRINT "  YOU WOUND THE ";g$
4420  MT=MT-1
```

```
4430 END IF
4440 IF RND>.25 AND HT>0 AND MT>0 THEN
4450  delay
4460  GO TO 4080
4470 END IF
4480 IF HT>MT THEN
4490  PRINT "YOU HAVE SLAIN THE ";g$
4500  st=st+1:de=de+1:wi=wi+1
4510  ch=ch+1:it=it+1:co=co+1
4520  GO TO 4740
4530 END IF
4540 PRINT " THE ";g$;" GOT THE"
4550 PRINT " BETTER OF YOU THAT TIME"
4560 IF z=1 OR q=1 THEN
4570  st=4*INT(st/5)
4580 END IF
4590 IF z=2 OR q=2 THEN
4600  ch=3*INT(ch/4)
4610 END IF
4620 IF z=3 OR q=3 THEN
4630  de=6*INT(de/7)
4640 END IF
4650 IF z=4 OR q=4 THEN
4660  it=2*INT(it/3)
4670 END IF
4680 IF z=5 OR q=5 THEN
4690  wi=5*INT(wi/6)
4700 END IF
4710 IF z=6 OR q=6 THEN
4720  co=INT(co/2)
4730 END IF
4740 a(ro,8)=0
4750  delay
4760 END DEFine FIGHT
4770 REMark ---------------
```

```
4780 DEFine PROCedure UNLOCK
4790 IF ro()12 OR lock=1 THEN
4800  PRINT "There is no locked door
                          in this room"
4810  GO TO 4760
4820 END IF
4830 flag=0
4840 FOR j=1 TO 5
4850  IF p(j)=2 THEN flag=j
4860 END FOR j
4870 IF flag=0 THEN
4880  PRINT "You do not have a key"
4890  GO TO 4760
4900 END IF
4910   music
4920 PRINT "There is a creak as the key turns"
4930   delay
4940   color
4950 PRINT "...The door is now
                          unlocked........"
4960 lock=1
4970 p(flag)=0
4980 END DEFine UNLOCK
4990 REMark ----------------
5000 DEFine PROCedure end_of_game
5010  color
5020  music
5030 sc=0:REMark score
5040 IF qu=4 THEN
5050  PRINT "I did not imagine you would"
5060  PRINT "turn out to be a quitter!"
5070  GO TO 5180
5080 END IF
5090  color:music
5100 IF BRAG=1 THEN
5110  PRINT "It's all over now..."
```

```
5120  sc=50:GO TO 5170
5130 END IF
5140 PRINT \"Congratulations!"
5150 PRINT "You have completed the adventure!"
5160 sc=100
5170 PRINT \\
5180 sc=9999*LOCKET+(sc+20*cash+47*mk+st+
            2*ch+3*de+4*it+5*wi+6*co)/qu
5190  color
5200 IF mk>0 THEN PRINT "   You killed ";
                          mk;" monsters"
5210  color
5220 PRINT \"You found $";cash;" worth"
5230 PRINT "           of treasure"\
5240 IF LOCKET=0 THEN
5250  PRINT "However, you failed to find the"
5260  PRINT "locket, so Mephistopheles"
5270  PRINT "          keeps your soul!"
5280 END IF
5290  color
5300 IF LOCKET=1 THEN
5310  PRINT "And you found the locket, so you"
5320  PRINT "   win your soul back from"
5330  PRINT "         Mephistopheles!!"
5340 END IF
5350 PRINT \"Your score for this adventure"
5360 PRINT "          is ";sc
5370   STOP
5380 END DEFine end_of_game
5390 REMark ----------------
5400 DEFine PROCedure room_descriptions
5410   music
5420 SELect ON ro
5430 ON ro=1
5440 PRINT "You are in a dusty room, guarded by
```

```
5450 PRINT "two stone statues of powerful demons"
5460 PRINT "which stand on each side, staring"
5470 PRINT "into each others' glazed eyes. The"
5480 PRINT "only exit takes you back where"
5490 PRINT "    you came from..."
5500 REMark **************

5510 ON ro=2
5520 PRINT "This is a sombre room, where the"
5530 PRINT "dead have been brought for judgement"
5540 PRINT "Thru' the archway to the east you"
5550 PRINT "can see a swiftly-flowing stream..."
5560 PRINT "Man-eating fish infest the water..."
5570 REMark ***************
5580 ON ro=3
5590 PRINT "As you leave the river behind, you"
5600 PRINT "stand, dripping, in a small crypt..."
5610 PRINT \"Fresh air blows from the north..."
5620 REMark **************

5630 ON ro=4
5640 PRINT "The deserted temple bears witness"
5650 PRINT "to the unholy act of the Black Mass."
5660 PRINT \"The stench is overpowering..."
5670 REMark **************

5680 ON ro=5
5690 PRINT "Skeletons in cages tell the horrid"
5700 PRINT "tale that here men who refused to"
5710 PRINT "give over their souls were left to"
5720 PRINT "die, very, very slowly..."
5730 REMark ***************

5740 ON ro=6 ,
5750 PRINT "This was once the Satanic King's"
5760 PRINT "treasury, but it has long been empty"
5770 REMark **************
5780 ON ro=7
5790 PRINT "This is a tiny, musty room, with"
5800 PRINT "daylight struggling through the tiny"
```

```
5810 PRINT "barred window, high on the wall"
5820 REMark ***************
5830 ON ro=8
5840 PRINT "Heaps of rust on the floor are all"
5850 PRINT "that is left of the armoury of"
5860 PRINT "         the Horned One"
5870 REMark ************
5880 ON ro=9
5890 PRINT "You find yourself in the library."
5900 PRINT "The books are ";:FLASH 1
5910 PRINT "spellbound";:FLASH 0
5920 PRINT " and cannot"
5930 PRINT "be removed from the shelves"
5940 PRINT "       by mere mortals..."
5950 REMark *************

5960 ON ro=10
5970 PRINT "Murals on the walls of this large"
5980 PRINT "room tell tales of battles fought"
5990 PRINT "between the forces of Light and Dark"
6000 REMark ************
6010 ON ro=11
6020 PRINT "You are in the room which the Black"
6030 PRINT "Guard of the Ancient Schismatic"
6040 PRINT "used for its barracks..."
6050 PRINT "       ....centuries ago"
6060 REMark ************


6070 ON ro=12
6080 PRINT "Long tables line this room, which"
6090 PRINT "was once used as a dining room by"
6100 PRINT "the Legion of the Lost. It is now"
6110 PRINT "  almost completely deserted"
6120 REMark ************
6130 ON ro=13
6140 PRINT "You find yourself in a huge, natural"
6150 PRINT "underground cavern, which has been"
```

```
6160 PRINT "reshaped as a store-room..."
6170 REMark **************
6180 ON ro=14
6190 PRINT "The Alcemist's Laboratory, with the"
6200 PRINT "echo of incantations still hanging"
6210 PRINT "in the damp and musty air..."
6220 REMark ************

6230 ON ro=15
6240 PRINT "You have made your way to the room"
6250 PRINT "  used by Mephistopheles for his"
6260 FLASH 1:PRINT "        PRIVATE QUARTERS!!!"
6270 FLASH 0
6280 PRINT "This means your quest is at the"
6290 PRINT "half-way mark...you have done"
6300 PRINT "well...";:color:PRINT"...so far..."
6310  END SELect
6320 END DEFine room_descriptions
6330 REMark ----------------


6340 DEFine PROCedure objects
6350 PRINT \"You can see:"
6360  color
6370 IF a(ro,5)<>0 THEN
6380  PRINT "   ";o$(a(ro,5))
6390  BEEP 10000,80
6400 END IF
6410 IF a(ro,6)<>0 THEN
6420  PRINT "   ";o$(a(ro,6))
6430  BEEP 10000,90
6440 END IF

6450 IF a(ro,7)<>0 THEN
6460  PRINT "   ";o$(a(ro,7))
6470  BEEP 10000,100
6480 END IF
6490   delay
6500 PRINT
```

```
6510 END DEFine objects
6520 REMark -----------------
6530 DEFine PROCedure initialise
6540   intro_frame
6550 DIM a(16,8):DIM p(5)
6560 DIM o$(8,12):DIM v(20)
6570 DIM M$(8,15):DIM T(5)
6580 workout
6590 st=roll
6600 workout
6610 ch=roll
6620 workout
6630 de=roll
6640 workout
6650 it=roll
6660 workout
6670 wi=roll
6680 workout
6690 co=roll
6700 PAN 10
6710 cash=0:REMark Treasure
6720 ro=9:REMark Starting room
6730 qu=1:REMark End of game flag
6740 mk=0:REMark Monsters killed
6750 lock=0
6760 c$=""
6770 BRIBE=0
6780 BRAG=0
6790 LOCKET=0
6800 ACID=0
6810 REMark ** Set up rooms **
6820  RESTORE
6830 PAN -10
6840 FOR x=1 TO 16
6850  FOR y=1 TO 4
```

```
6860   READ a(x,y)
6870  END FOR y
6880 END FOR x
6890 PAN 10
6900 REMark ** Distribute treasure **
6910 FOR z=1 TO 8
6920  READ o$(z):READ v(z)
6930 END FOR z
6940 a(6,5)=1
6950 a(10,5)=2
6960 a(15,5)=3
6970 a(7,5)=4
6980 FOR q=5 TO 8
6990  z=RND(1 TO 15)
7000  IF a(z,5)<>0 THEN GO TO 6990
7010  a(z,8)=q:REMark object in room
7020 END FOR q
7030 PAN -10
7040 REMark ** Distribute monsters **
7050 FOR j=1 TO 8
7060  READ M$(j)
7070  IF j=1 THEN GO TO 7120
7080  z=RND(1 TO 15)
7090  IF z=9 OR z=3 OR z=1 THEN GO TO 7080
7100  IF a(z,8)<>0 THEN GO TO 7080
7110  a(z,8)=j
7120 END FOR j
7130 a(1,8)=1
7140 PAPER 7:INK 0
7150 BORDER 0
7160 CLS:CLS #0
7170 END DEFine initialise
7180 REMark ******************
7190 DEFine PROCedure delay
7200   PAUSE 56
```

```
7210  BEEP 1000,RND(1 TO 100)
7220 END DEFine delay
7230 DEFine PROCedure color
7240  INK RND(1 TO 5)
7250 END DEFine color
7260 DEFine PROCedure workout
7270  roll=0
7280 FOR kk=1 TO 3
7290  roll=roll*RND(2 TO 7)
7300 END FOR kk
7310 END DEFine workout
7320 DEFine PROCedure DOLINE
7330  FLASH 1
7340  color
7350 PRINT "----------------
      ----------------------"
7360  FLASH 0
7370  color
7380 END DEFine DOLINE
7390 DATA 0,4,0,0:REMark room one
7400 DATA 0,5,3,0:REMark room two
7410 DATA 16,0,0,2:REMark room three
7420 DATA 1,8,5,0:REMark room four
7430 DATA 2,0,6,4:REMark room five
7440 DATA 0,0,0,5:REMark room six
7450 DATA 0,13,0,0:REMark room seven
7460 DATA 4,11,0,0:REMark room eight
7470 DATA 0,12,10,0:REMark room nine
7480 DATA 0,0,0,9:REMark room ten
7490 DATA 8,14,12,0:REMark room eleven
7500 DATA 9,0,13,11:REMark room twelve
7510 DATA 7,0,0,12:REMark room thirteen
7520 DATA 11,0,15,0:REMark room fourteen
7530 DATA 0,0,0,14:REMark room fifteen
7540 DATA 0,3,0,0:REMark room sixteen
```

```
7550 REMark ** Object data **
7560 DATA "RING",567
7570 DATA "KEY",2
7580 DATA "LOCKET",15
7590 DATA "ACID",7
7600 DATA "SWORD",0
7610 DATA "AXE",0
7620 DATA "AMYTHEST",27
7630 DATA "CRYSTAL",45
7640 REMark ** Monster names **
7650 DATA "ENTANGLER"
7660 DATA "ARTIFACTUM"
7670 DATA "INKBLOTT"
7680 DATA "UGLY UNDEAD"
7690 DATA "'ORRIBLE ORC"
7700 DATA "ARCBLUNDEN"
7710 DATA "EMBIHUUND"
7720 DATA "INSALIVATE"
7730 DEFine PROCedure intro_frame
7740 PAPER 2
7750 CLS:CLS #0
7760 BORDER 15,3,1
7770 FOR zz=1 TO 3
7780  INK 6
7790 CSIZE 3,1
7800 FOR muse=1 TO 20
7810  BEEP 1000,4*muse
7820   PAUSE 4
7830  END FOR muse
7840 AT 1,0
7850 PRINT "        T R Y S T"
7860 CSIZE 1,0
7870 PRINT \\"          with"
7880 CSIZE 3,1
7890 PRINT \"     MEPHISTOPHELES"
```

```
7900 CSIZE 0,0
7910 INK 1
7920 IF zz=3 THEN
7930 PRINT \\\ "          Please stand by"
7940  END IF
7950 IF zz=1 THEN
7960 FOR j=1 TO 30
7970  PAN j
7980 END FOR j
7990 ELSE IF zz=2
8000 FOR j=1 TO 30
8010   PAN -j
8020 END FOR j
8030 END IF
8040 END FOR zz
8050 END DEFine intro_frame
8060 DEFine PROCedure music
8070  BEEP 1000,RND(1 TO 10)
8080  BEEP 1000,RND(40 TO 60)
8090  BEEP 1000,1
8100  BEEP 1000,RND(10 TO 30)
8110  BEEP 1000,1
8120 END DEFine music
```

MAGICIAN'S MAZE

Here's a program which will get your brain in a sweat,



as it tests your powers of deduction . . . and map-making.

The wily old magician has trapped you in an abandoned house which is positively littered with magical and valuable items. To secure your freedom, you must transport six different objects into the store room.

Easy, you say, until the magician points out the true nature of his test. The house is built like a maze, and you do not yet have a map. You have to try and work out the ground plan as you go along. You do not even know where the store room is. To complicate matters further, you can only carry one thing at a time. Although you can drop any of the objects if you choose, you cannot drop them in a room which already contains one object.

Sounds complicated? It is, but the satisfaction you'll experience when you solve the maze and complete the task set by the magician will make it more than worthwhile.

The program understands the words you type in (after a limited fashion) and can carry out your wishes as you'll soon see. You type in your instructions in the form of two words, such as GO NORTH.

Once you move north, you might see a 'silver chalice', and decide to pick it up, which you do with the command GET SILVER. In fact, only the first three letters are needed, so GET SIL would have worked as well (and you could have typed in GET SILVER CHALICE if you really wanted to do so).

If you are sensible, you'll be making a map as you work out the relationship of the rooms to each other, so you can move through the maze with growing confidence.

Soon you will come to the store room, where you can drop your silver chalice. After much wandering about and carrying, you'll find the store room becoming gratifyingly full.

The QL will reject any input from you that it does not understand. To acquire an object, you tell the computer to GET or TAKE it, and to get rid of anything, you either DROP it or PUT it. To give up (as if you would!), enter QUIT. GO, followed by NORTH, SOUTH, EAST or WEST will move you in the direction of your choice.

Here's the listing so you can take on the challenge posed by the magician:

```
10 REMark MAGICIAN'S MAZE
20 initialise
30 REPeat cycle
40   game
50 END REPeat cycle
60 DEFine PROCedure game
70   delay
80 CLS:PRINT \\
90 mov=mov+1
100  color
110 PRINT "   This is move ";mov
120 PRINT \" You are in the ";n$(ro)
130 IF ro<>1 THEN GO TO 240
140 flag=0
```

```
150 PRINT "      which holds:"
160 FOR zz=1 TO 8
170 IF p$(zz)<>"" THEN PRINT "        ";
        p$(zz):flag=1
180 END FOR zz
190 IF flag=0 THEN PRINT "        NOTHING"
200 color
210 FOR gg=10,40,3,50,3,2,46,4,1
220  BEEP 10000,gg
230 END FOR gg
240 PRINT \\" There are exits to the"
250 color
260 PRINT "       ";
270 IF a(ro,1)<>0 THEN PRINT "NORTH ";
280 IF a(ro,2)<>0 THEN PRINT "SOUTH ";
290 IF a(ro,3)<>0 THEN PRINT "EAST ";
300 IF a(ro,4)<>0 THEN PRINT "WEST"
310 color
320 IF ro<>1 AND O$(ro)<>"" THEN PRINT \
          " THE ROOM HOLDS A ";O$(ro)
330 IF M$<>"" THEN PRINT\ "YOU ARE
              CARRYING A"/M$
340 IF M$="" THEN PRINT \"      YOUR
              ARMS ARE EMPTY"
350 IF RND>.7 THEN BEEP 10000,35
360 take_words
370 END DEFine game
380 DEFine PROCedure take_words
390 color:FLASH 1
400 PRINT \\" What do you want to do? ";
410 FLASH 0
420 INPUT e$
430 IF CODE(e$(1))>CODE("Z") THEN PRINT
    "ENGAGE CAPS LOCK!":delay:GO TO 390
```

```
440 IF LEN(e$)<7 THEN e$=e$&" ":GO TO 440
450 f$=e$(1 TO 3)
460 IF f$="QUI" THEN GO TO 1190
470 IF f$="GO "THEN f$=e$(4 TO 6)
480 IF RND>.7 THEN MUSIC
490 IF f$="NOR" OR f$="SOU" OR f$="EAS" OR
      f$="WES" THEN action:GO TO 550
500 z=1
510 IF e$(z)=" " THEN
          g$=e$(z+1 TO z+4):GO TO 570
520 IF z<LEN(e$) THEN z=z+1:GO TO 510
530 color
540 PRINT "I DON'T UNDERSTAND ";e$
550 END DEFine take_words
560 GO TO 70
570 IF f$="GET" OR f$="TAKE" THEN do_it:
          GO TO 550
580 IF f$="PUT" OR f$="DRO" THEN put_it:
          GO TO 550
590 GO TO 540
600 DEFine PROCedure action
610 IF RND>.7 THEN MUSIC
620 f$=f$(1)
630 IF f$="N" THEN GO TO 690
640 IF f$="S" THEN GO TO 710
650 IF f$="E" THEN GO TO 730
660 color
670 IF a(ro,4)=0 THEN PRINT "ONLY GHOSTS
    CAN MOVE THROUGH WALLS":GO TO 750
680 ro=a(ro,4):GO TO 750
690 IF a(ro,1)=0 THEN PRINT "THERE IS
    NO EXIT THAT WAY":GO TO 750
```

```
700 ro=a(ro,1):GO TO 750
710 IF a(ro,2)=0 THEN PRINT "THAT'S
                  A SOLID WALL":GO TO 750
720 ro=a(ro,2):GO TO 750
730 IF a(ro,3)=0 THEN PRINT "THAT
                IS IMPOSSIBLE!":GO TO 750
740 ro=a(ro,3)
750 END DEFine action
760 DEFine PROCedure do_it
770  color
780 IF ro=1 THEN EXIT do_it
790 IF g$(1 TO 3)<>O$(ro)(1 TO 3) THEN
       PRINT "IT IS NOT HERE":GO TO 830
800 IF M$<>"" THEN BEEP 20000,255:PRINT \
    "YOU CAN ONLY CARRY ONE THING AT ONCE":
    GO TO 830
810 M$=O$(ro)
820 O$(ro)=""
830 END DEFine do_it
840 game
850 DEFine PROCedure put_it
860  color
870 IF M$="" THEN PRINT "YOU'RE NOT
    CARRYING IT":GO TO 1040
880 IF ro=1 THEN GO TO 940
890 color
900 IF O$(ro)<>"" THEN PRINT "THIS ROOM
    ALREADY HOLDS A"/O$(ro):GO TO 1040
910 O$(ro)=M$
920 M$=""
930 GO TO 1040
940 place=0:flag=0
950 FOR zz=1 TO 6
960  IF place=1 THEN GO TO 980
```

```
970 IF p$(zz)="" THEN p$(zz)=M$:place=1:M$=""
980 IF p$(zz)<>"" THEN flag=flag+1
990 END FOR zz
1000 IF flag=0 THEN PRINT "        NOTHING"
1010 PRINT "      ";6-flag;" OBJECTS TO GO"
1020 delay
1030 IF flag=6 THEN end_of_game:STOP
1040 END DEFine put_it
1050 DEFine PROCedure end_of_game
1060  color
1070 PRINT\"     YOU'VE DONE IT!"
1080 PRINT\"YOU GOT 6 OBJECTS IN
                          THE STORE ROOM"
1090 FOR zz=1 TO 8
1100  color
1110 IF p$(zz)<>"" THEN PRINT "    ";p$(zz)
1120 END FOR zz
1130 PRINT "          IT TOOK YOU ";
1140  color:PRINT mov;:color
1150 PRINT " MOVES"
1160 END DEFine end_of_game
1170 STOP
1180 color
1190 PRINT \"I didn't think you were a ";
1200 color
1210 PRINT "quitter"
1220 color
1230 PRINT \"     You survived for ";
1240 color:PRINT mov;:color
1250 IF mov>1 THEN
1260  PRINT " moves"
1270   ELSE PRINT " move"
1280 STOP
1290 DEFine PROCedure initialise
1300 CLS:CLS #0
```

```
1310 intro_frame
1320 PAPER 7:CLS
1330 BORDER 1,1
1340 CSIZE 0,0
1350 DIM a(9,4):DIM n$(9,25)
1360 DIM O$(18,25):DIM p$(9,25)
1370 DIM m(8)
1380  RESTORE
1390 ro=RND(2 TO 9)
1400 FOR jj=1 TO 9
1410  READ n$(jj)
1420 END FOR jj
1430 FOR jj=1 TO 18
1440  READ O$(jj)
1450 END FOR jj
1460 REMark Distribute objects
1470 REMark Moses/Oakford shuffle
1480 FOR j=18 TO 11 STEP -1
1490  z=RND(1 TO j)
1500  M$=O$(z)
1510  O$(z)=O$(j)
1520  O$(j)=M$
1530 END FOR j
1540 M$=""
1550 mov=0
1560 FOR j=1 TO 9
1570  FOR k=1 TO 4
1580   READ a(j,k)
1590  END FOR k
1600 END FOR j
1610 END DEFine initialise
1620 DATA "STORE ROOM","GLOOMY TURRET"
1630 DATA "MARBLE HALL","LIBRARY"
1640 DATA "AUDIENCE CHAMBER"
1650 DATA "MASTER BEDROOM","STUDY"
1660 DATA "ARTIST'S STUDIO"
```

```
1670 DATA "MUSICIAN'S QUARTERS"
1680 DATA "BROKEN CANDLESTICK"
1690 DATA "GOLD PIECE"
1700 DATA "SHINING EMERALD"
1710 DATA "BRONZE STATUE"
1720 DATA "SILVER CHALICE"
1730 DATA "MIGHTY SWORD"
1740 DATA "DIAMOND RING"
1750 DATA "CRYSTAL GOBLET"
1760 DATA "SHADOW BOWL"
1770 DATA "MYSTIC SCROLL"
1780 DATA "WILLOW WAND"
1790 DATA "NEWT'S TAIL"
1800 DATA "HEMLOCK POTION"
1810 DATA "PAPYRUS INCANTATION"
1820 DATA "BLACK ACONITE"
1830 DATA "GLOWING ARTHAME"
1840 DATA "GLASS FRAGILUS"
1850 DATA "POWERFUL PENTANGLE"
1860 DATA 0,0,2,0:REMark ROOM ONE
1870 DATA 0,6,0,1:REMark ROOM TWO
1880 DATA 0,9,7,0:REMark ROOM THREE
1890 DATA 0,0,5,0:REMark ROOM FOUR
1900 DATA 0,7,6,4:REMark ROOM FIVE
1910 DATA 2,0,0,5:REMark ROOM SIX
1920 DATA 5,0,8,3:REMark ROOM SEVEN
1930 DATA 0,0,9,7:REMark ROOM EIGHT
1940 DATA 3,0,0,8:REMark ROOM NINE
1950 DEFine PROCedure intro_frame
1960 PAPER 1:BORDER 4,2,1
1970 CSIZE 3,1
1980 CLS:CLS #0
1990 FOR gg=10,40,10,20,50,10,3,44,250,240,
              230,233,1,10,55,44,5,3,55,1
2000  BEEP 30000,gg
2010 END FOR gg
```

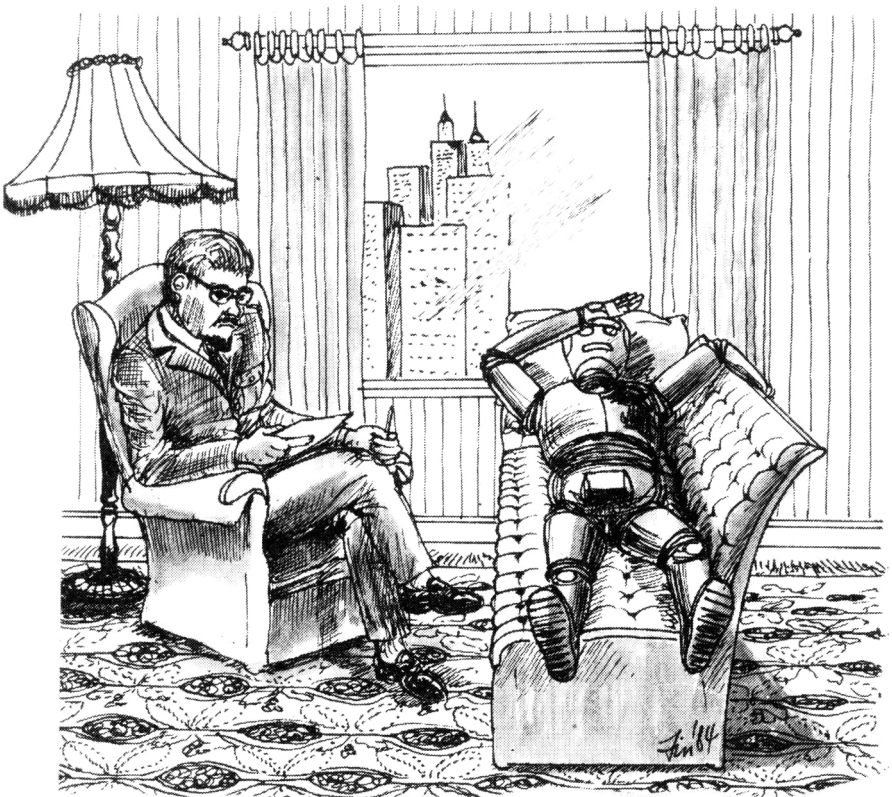
```
2020 FOR k=1 TO 100
2030  INK (k/25+2)
2040 AT 3,11:PRINT "Magician's Maze"
2050 AT 5,11
2060 IF k=20 THEN PRINT "NOW WE EXPLORE"
2070 IF k=50 THEN PRINT "  THE MYSTERY "
2080 IF k=70 THEN PRINT "       OF
2090 IF k=90 THEN PRINT "THE AWFUL MAZE!"
2100  s=RND(1 TO 50)
2110  t=RND(1 TO 50)
2120   INK 7
2130  POINT s,t
2140  POINT 256-s,256-t
2150  POINT s,256-t
2160  POINT t,256-s
2170  POINT t,s
2180  POINT 256-t,256-s
2190  POINT 256-s,t
2200  POINT 256-t,s
2210 END FOR k
2220  MUSIC:MUSIC:MUSIC
2230 END DEFine intro_frame

2240 DEFine PROCedure delay
2250  BEEP 1000,1:BEEP 1000,40
2260  FOR wait=1 TO 700
2270  END FOR wait
2280  BEEP 1000,20
2290 END DEFine delay
2300 DEFine PROCedure color
2310 INK RND(1 TO 5)
2320 END DEFine color
2330 DEFine PROCedure MUSIC
2340  FOR gg=89,23,89,43,2,1,66,1
2350   BEEP 10000/gg,gg
```

```
2360    FOR hh=1 TO 60+gg
2370    END FOR hh
2380  END FOR gg
2390 END DEFine MUSIC
```

ARTIFICIAL INTELLIGENCE

Artificial intelligence is a goal which has not yet been



achieved. Certainly, programs which enable computers to exhibit behavior which could conceivably be

classed as intelligent have been written, but the 'intelligence' has been limited and effective only within a severely restricted domain.

That is, a computer can appear to be brainy, but only if you limit the environment within which it has to demonstrate those brains.

You'll see here that we have several programs which will give your computer the appearance of intelligence, and certainly provide impressive demonstrations of 'electronic brainpower' when the programs are shown to computer-naive people.

Despite the claims made by some people (such as Carl Sagan, "In Defence of Robots", *Broca's Brains)* that human beings are, essentially, just well-made computers, we sense there is a wide gulf between our own perception of our selfhood, and the total non-awareness that computers can have (at least at present) of their own existence. And it seems to me that this self-perception lies at the heart of at least one aspect of intelligence.

Computers can function extremely 'intelligently' in restricted areas. Look at the five inch square, hand-held chess machines. Within the limited domain of chess game, a solid appearance of intelligence can be created.

It is so within the limited universes created by the programs in this section. One of them is a variant of one of the most well-known and popular AI programs

ever, ELIZA, in which the computer takes the role of a psychiatrist, and apparently carries on a conversation. ELIZA was developed in the mid-60s by Joseph Weizenbaum. He was attempting to produce a program which would caricature a Rogerian psychoanalyst, and it has been suggested (see "You're never alone with a fluffy micro", David Tebbutt, MicroScope, January 13, 1983, p.9) that Weizenbaum has regretted ever since that day his work on Eliza. It is not hard to see why.

In the euphoric days after the program was made public, wild and erratic claims were made that it was only a matter of (very little) time before computers could totally replace 'mind-doctors'. Further, the Eliza success — said some incautious observers — paved the way for genuinely intelligent computers.

Weizenbaum did not believe it for a moment. He writes that he was amazed at the beginning — and continued to be amazed — at the reaction people had to the program. Written in LISP on a mainframe at MIT, the program could be accessed from any one of a number of time-sharing terminals dotted around the campus. Weizenbaum reports he was very interested to see that the program was being accessed time and time again very late at night, as though troubled students were really discussing their problems with the program. Furthermore, the accesses were all very long. The computer log showed some people were hooked into Eliza for over an hour at a time.

Weizenbaum's secretary had worked closely with him over the six months or so it took to produce the

program, and she knew as well as he how it worked, scanning a person's speech for 'key words' (such as **dream** and **friends**) and then chosing a suitable reply from a bank of such replies. Other words from the user's sentences could be incorporated, sentences could be 'turned around' (so "I am happy because of the weather" could be simply fed back as either "Why are you happy because of the weather" or just as a statement of the form "You are happy because of the weather. What does this suggest to you?") and a number of other replies (such as "That is interesting, please go on") could be used if no keyword was recognized.

Despite the secretary's familiarity with the program, Weizenbaum noticed that if he walked into the office when she was accessing the program, she became embarrassed and refused to let him see the printout.

Further, when he suggested it would be interesting to hook up a printer to the main body of the computer to record the late night conversations students had with the program, the suggestion was greeted with horror, as though he was suggesting a kind of electronic peeping-Tom activity. (If you'd like to look further into Weizenbaum's work on AI, you might be interested in his book *Computer Power and Human Reason,* W H Freeman and Co., San Francisco, 1976.)

A common stumbling block in the work of early researchers into the field of artificial intelligence was that writing a program which emulated some aspect of human reasoning (such as a very impressive Checkers

program by Samuels) did not necessarily lead researchers any closer to either producing behavior which could be classed as intelligent, nor towards a theoretical understanding of the processes of human reasoning and deduction. "Progress in producing intelligent behaviour is not necessarily progress towards producing thought" (Norman Whaland, *When Is A Program Intelligent?,* Creative Computing, February, 1981, pp. 44-48).

There is another way of approaching the problem, which I call the "if it quacks like a duck, it's a duck" approach. In the book "The Turing Criterion — Machine Intelligent Programs for the 16K ZX81" (Charlton et al, Interface Publications London, 1982), which I edited, I answer the question "What is the Turing Criterion?" in the following words:

> "In 1950, the English mathematician and logician A. M. Turing proposed what has become known as the 'Turing Criterion' for machine intelligence.
>
> "He said that if you were dealing with something at the end of a wire that could be a machine, or could be human, and you could not tell — from the responses coming to you over the wire — whether that with which you were dealing was human or machine, the 'thing' at the other end was, by definition, intelligent."

On this basis, it would be possible to class many, many computer programs as intelligent. Certainly several of those within this book could almost pass the test,

especially if you were prepared to accept the possibility that some pretty stupid humans were dealing with you.

ELIZA

We'll start our investigation of Artificial Intelligence with a simplified version of the classic program, ELIZA, which we discussed at some length in the introduction. As we said there, the program scans your input for words it can use, and then reflects your words back to you in a way which makes it seem as if the program is really speaking to you. Once you've played with it a while, you'll learn how to trigger the most effective responses.

Here's part of one 'conversation' I had with my version of ELIZA:

```
Welcome, to another session with
your QL psychiatrist, Eliza.

What is your first name? Tim


  WELL, HI THERE, TIM

IT SURE IS NEAT TO HAVE YOU DROP BY

------------------------------------
 > ? I want some help with my problems
WHY DO YOU WANT SOME HELP WITH YOUR
          PROBLEMS
------------------------------------
 > ? Computers have a good reputation
         in this field
```

```
WHAT IS THE BEST THING ABOUT COMPUTERS?
---------------------------------------
 > ? Their ability to compute
CAN YOU ELABORATE ON THAT?

---------------------------------------
 > ? The QL, for example, runs really
     quickly
WHY DO YOU MENTION THE QL?

---------------------------------------
 > ? I am talking to one
HOW LONG HAVE YOU BEEN TALKING TO ONE
---------------------------------------
 > ? For ten minutes or so
I UNDERSTAND, TIM...

---------------------------------------
 > ? I am afraid of mice
HOW LONG HAVE YOU BEEN AFRAID OF MICE
---------------------------------------
 > ? I was bitten by a rat at the age of ten
THAT'S SO, TIM. PLEASE CONTINUE

---------------------------------------
 > ? I think the rat scared me
DOES IT WORRY YOU THAT YOU
THINK THE RAT SCARED YOU

---------------------------------------
 > ? Yes, of course it does
YOU SEEM VERY POSITIVE. WHY IS THIS?
---------------------------------------
 > ? Because I feel I know the truth
     about rats
```

TELL ME MORE ABOUT FEELING YOU KNOW
THE TRUTH ABOUT RATS

> ? I shake and rattle and roll when
I see them

WHAT WOULD YOUR BEST FRIEND SAY TO THAT?

> ? My friends never talk about rats

WHY DO YOU BRING UP THE SUBJECT OF FRIENDS?

> ? You brought it up, turkey

I UNDERSTAND, TIM...

> ? Do you really understand?

YES, I REALLY UNDERSTAND

> ? I bet you say that to all the boys

WHY ARE YOU SAYING THAT, TIM?

> ? Because I want a proper conversation
with you

WHY DO YOU WANT A PROPER CONVERSATION
WITH ME

> ? To prove you are not a fraud

WHAT MAKES YOU THINK I AM

> ? Your dumb answers

WHY ARE YOU CONCERNED ABOUT MY DUMB ANSWERS

```
 ) ? I don't think you're listening to me
WHY DON'T YOU THINK I'M LISTENING TO YOU
------------------------------------

 ) ? I think we have been through this
     before
DOES IT WORRY YOU THAT YOU THINK WE HAVE
BEEN THROUGH THIS BEFORE

------------------------------------

 ) ? Not in the slightest, Sinclair machine
WHY ARE YOU BEING NEGATIVE?
------------------------------------

 ) ? Why are you asking dumb questions?
WHAT IS IT THAT YOU REALLY WANT TO KNOW?
------------------------------------

 ) ? I want to know when I will get some help
WHY DO YOU WANT TO KNOW WHEN
------------------------------------

 ) ? You sure talk strangely, buddy
WHAT DOES THAT SUGGEST TO YOU, TIM?

------------------------------------

 ) ? That you are pretty el stupido
WHAT MAKES YOU THINK I AM

------------------------------------

 ) ? You are doing it again
WHAT MAKES YOU THINK I AM DOING IT AGAIN
------------------------------------

 ) ? This conversation is enough to
      turn a programmer to drink
MODERATION IN ALL THINGS SHOULD BE THE RULE.

------------------------------------

 ) ? Even in sex
```

```
I FEEL YOU SHOULD REALLY DISCUSS THIS WITH
 A HUMAN.
-------------------------------------
 > ? Do you really think so
YES, I REALLY THINK SO
-------------------------------------
 > ? Perhaps I should go
YOU'RE NOT VERY FIRM ON THAT!
-------------------------------------
 > ? Bye, bye, turkey

 BYE BYE FOR NOW, TIM

SEE YOU AGAIN SOME TIME.
```

And here's the listing, so your computer can now help you sort yourself out:

```
10 REMark Eliza
20 initialise
30 PRINT \"Welcome, to another session with"
40 PRINT "your QL psychiatrist, Eliza."
50 INPUT \"What is your first name? ";zi$
60 IF zi$="" THEN GO TO 50
70 z$=""
80 FOR ju=1 TO LEN(zi$)
90  IF CODE(zi$(ju))>CODE("Z") THEN
100    zi$(ju)=CHR$(CODE(zi$(ju))-32)
110  END IF
120  z$=z$&zi$(ju)
130 END FOR ju
140 CLS
150 PRINT \\\\"  WELL, HI THERE, ";z$
160 PAUSE 50
170 PRINT \"IT SURE IS NEAT TO HAVE ";
```

```
180 PRINT "YOU DROP BY"
190 PAUSE 90
200 CLS
210 REPeat conversation
220   INK RND(6 TO 7)
230   ACCEPT_INPUT
240   process_input
250 END REPeat conversation
260 REMark ---------------
270 DEFine PROCedure process_input
280  REMark Look for trigger words at start
290 p=0
300 p=p+1
310 IF A$(1 TO LEN(B$(p)))=B$(p) THEN
320 PRINT c$(p)/
330 A$=" " & A$(LEN(B$(p))+1 TO LEN(A$))&"
340 GO TO 470:REMark use rest of input
350 END IF
360 IF p<38 THEN GO TO 300
370 REMark Look for triggers within text
380 HH=0
390 HH=HH+1
400 IF B$(HH) INSTR A$<>0 THEN
410   PRINT c$(HH)
420   GO TO 740
430 END IF
440 IF HH<38 THEN GO TO 390
450 RANDOM_REPLIES
460 GO TO 740
470 REMark Use rest of input
480 IF CODE(c$(p)(LEN(c$(p))))<CODE("A")
    THEN GO TO 740
490 REMark Convert balance of a$
500 s=0
510 s=s+1
520 IF A$(s)=" " THEN GO TO 560
```

```
530 IF s<LEN(A$) THEN GO TO 510
540 PRINT A$
550 GO TO 740
560 REMark Search for conjugate
570 jk=0
580 jk=jk+1
590 IF s+1+jk)=LEN(A$) THEN GO TO 740
600 IF A$(s+1+jk)=" " THEN GO TO 630
610 IF jk+s<LEN(A$)-3 THEN GO TO 580
620 GO TO 740
630 m$=A$(s+1 TO s+jk)
640 IF LEN(m$)<2 THEN GO TO 740
650 lp=0
660 lp=lp+1
670 IF f$(lp)=m$ OR f$(lp)=m$(2 TO)
      THEN GO TO 700
680 IF lp<15 THEN GO TO 660
690 PRINT m$;" ";:GO TO 710
700 PRINT g$(lp);" ";
710 A$=A$(LEN(m$) TO LEN(A$))
720 IF A$(3 TO 4)="  " THEN GO TO 740
730 GO TO 510
740  BEEP 10000,RND(40 TO 100)
750 END DEFine process_input
760 REMark ------------
770 DEFine PROCedure RANDOM_REPLIES
780 Z=RND(1 TO 11)
790 SELect ON Z
800 ON Z=1
810 PRINT "WHAT DOES THAT SUGGEST TO
       YOU, ";z$;"?"
820 ON Z=2
830 PRINT "I SEE..."
840 ON Z=3
850 PRINT "I'M NOT SURE I UNDERSTAND
      YOU FULLY"
```

```
860 ON Z=4
870 PRINT "CAN YOU ELABORATE ON THAT?"
880 ON Z=5
890 PRINT "THAT IS QUITE INTERESTING"
900 ON Z=6
910 PRINT "THAT'S SO, ";z$;". PLEASE CONTINUE"
920 ON Z=7
930 PRINT "I UNDERSTAND, ";z$;"..."
940 ON Z=8
950 PRINT "WELL, WELL...DO GO ON"
960 ON Z=9
970 PRINT "WHY ARE YOU SAYING THAT, ";z$;"?"
980 ON Z=10
990 PRINT "PLEASE EXPLAIN THE BACKGROUND"
1000 PRINT "  TO THAT REMARK"
1010 ON Z=11
1020 PRINT "COULD YOU SAY THAT AGAIN,"
1030 PRINT "   IN A DIFFERENT WAY?"
1040 END SELect
1050 END DEFine RANDOM_REPLIES
1060 REMark ------------------
1070 DEFine PROCedure ACCEPT_INPUT
1080 PRINT \"--------------------
                    ------------------"\
1090 PRINT " ? ";
1100 INPUT x$:A$=""
1110 IF x$="" THEN GO TO 1100
1120 FOR jj=1 TO LEN(x$)
1130 IF x$(jj)="," OR x$(jj)="'" OR
        x$(jj)="?" THEN GO TO 1180
1140 IF CODE(x$(jj))>CODE("Z") THEN
1150  x$(jj)=CHR$(CODE(x$(jj))-32)
1160 END IF
1170  A$=A$&x$(jj)
1180 END FOR jj
```

```
1190 k$=A$
1200 IF A$(1 TO 3)="BYE" THEN
1210  PRINT \\"   BYE BYE FOR NOW, ";z$
1220  PRINT \"  SEE YOU AGAIN SOME TIME."\\\
1230  STOP
1240 END IF
1250 IF A$(1 TO 7)="BECAUSE" THEN
1260  A$=A$(9 TO)
1270  K=K-7
1280 END IF
1290 K=LEN(A$)
1300 A$=" " & A$ & " "
1310 END DEFine ACCCEPT_INPUT
1320 REMark ---------------
1330 DEFine PROCedure initialise
1340 PAPER 1:INK 7
1350 CLS:CLS #0
1360 DIM B$(38,15):DIM c$(38,55)
1370 DIM f$(15,7):DIM g$(15,9)
1380  RESTORE
1390 REMark ** FILL B$,C$ **
1400 FOR AA=1 TO 38
1410  READ B$(AA):READ c$(AA)
1420 END FOR AA
1430 RESTORE 2190
1440 FOR AA=1 TO 15
1450  READ f$(AA):READ g$(AA)
1460 END FOR AA
1470 END DEFine initialise
1480 REMark --------------
1490 REMark ** KEYWORDS, REPLIES **
1500 DATA " CAN YOU"
1510 DATA "PERHAPS YOU WOULD LIKE TO BE
          ABLE    TO"
1520 DATA " DO YOU","YES, I"
1530 DATA " CAN I"
```

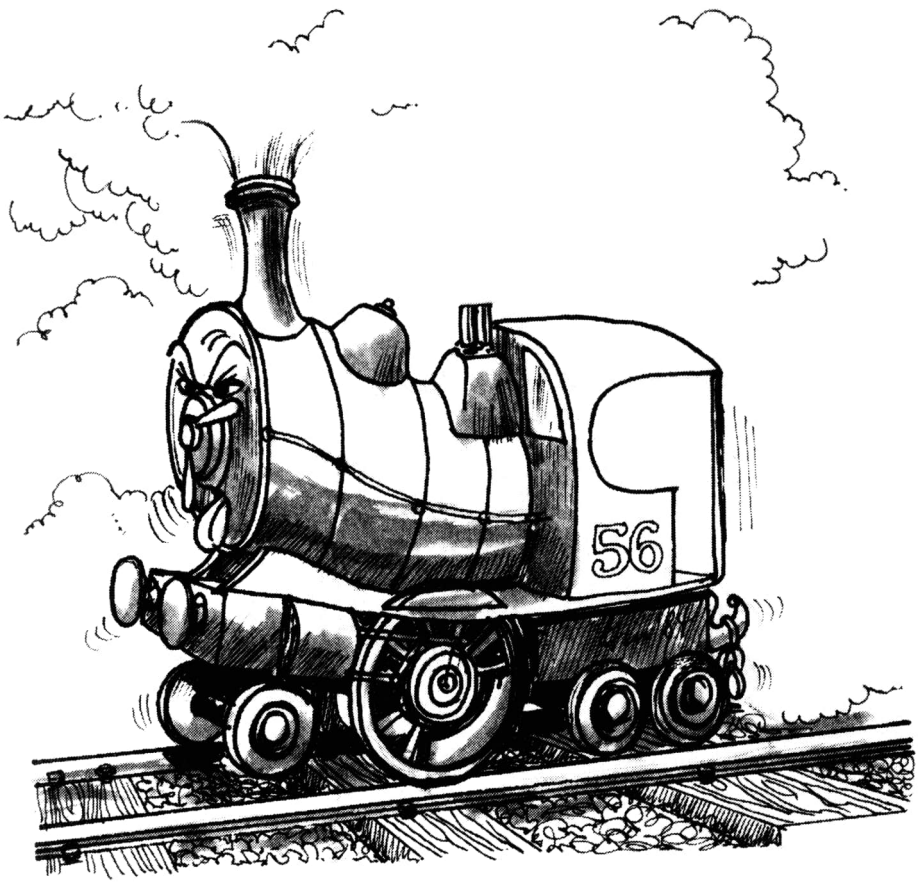
```
1540 DATA "PERHAPS YOU DON'T WANT TO
          BE ABLE    TO"
1550 DATA " YOU ARE"
1560 DATA "WHAT MAKES YOU THINK I AM"
1570 DATA " YOURE"
1580 DATA "WHAT'S YOUR REACTION TO ME BEING"
1590 DATA " I DONT","WHY DON'T YOU"
1600 DATA " I FEEL"
1610 DATA "TELL ME MORE ABOUT FEELING"
1620 DATA " WHY DONT YOU"
1630 DATA "WHY WOULD YOU WANT ME TO"
1640 DATA " WHY CANT I"
1650 DATA "WHAT MAKES YOU THINK YOU SHOULD
          BE   ABLE TO"
1660 DATA " ARE YOU"
1670 DATA "WHY ARE YOU INTERESTED IN
          WHETHER    OR NOT I AM"
1680 DATA " I CANT"
1690 DATA "HOW DO YOU KNOW YOU CAN'T"
1700 DATA " YES"
1710 DATA "YOU SEEM VERY POSITIVE.
           WHY IS THIS?"
1720 DATA " SEX"
1730 DATA "I FEEL YOU SHOULD REALLY
          DISCUSS     THIS WITH A HUMAN."
1740 DATA " I AM"
1750 DATA "HOW LONG HAVE YOU BEEN"
1760 DATA " IM","WHY ARE YOU TELLING
           ME YOU'RE"
1770 DATA " I WANT"
1780 DATA "WHY DO YOU WANT"
1790 DATA " WHAT","WHAT DO YOU THINK?"
```

```
1800 DATA " HOW"
1810 DATA "WHAT ANSWER WOULD PLEASE YOU MOST?"
1820 DATA " I THINK YOURE"
1830 DATA " HOW COULD YOU SAY I'M"
1840 DATA " I THINK YOU ARE"
1850 DATA "WHY DO YOU SAY I'M"
1860 DATA " WHO"
1870 DATA "HOW OFTEN DO YOU THINK OF
          SUCH       QUESTIONS?"
1880 DATA " WHERE"
1890 DATA "WHY DID YOU ASK ME THAT?"
1900 DATA " WHEN"
1910 DATA "WHAT WOULD YOUR BEST FRIEND
          SAY      TO THAT?"
1920 DATA " WHY"
1930 DATA "WHAT IS IT THAT YOU REALLY
          WANT TO   KNOW?"
1940 DATA " PERHAPS"
1950 DATA "YOU'RE NOT VERY FIRM ON THAT!"
1960 DATA " DRINK"
1970 DATA "MODERATION IN ALL THINGS
          SHOULD      BE THE RULE."
1980 DATA " SORRY"
1990 DATA "WHY ARE YOU APOLOGIZING?"
2000 DATA " DREAM"
2010 DATA "WHY DID YOU BRING UP THE
          SUBJECT      OF DREAMS?"
2020 DATA " I LIKE"
2030 DATA "IS IT GOOD THAT YOU LIKE"
2040 DATA " MAYBE"
2050 DATA "AREN'T YOU BEING A BIT TENTATIVE?"
2060 DATA " NO","WHY ARE YOU BEING NEGATIVE?"
2070 DATA " YOUR"
```

```
2080 DATA "WHY ARE YOU CONCERNED ABOUT MY"
2090 DATA " ALWAYS"
2100 DATA "CAN YOU THINK OF A SPECIFIC EXAMPLE?
2110 DATA " FRIEND"
2120 DATA "WHY DO YOU BRING UP THE
          SUBJECT      OF FRIENDS?"
2130 DATA " QL","WHY DO YOU MENTION THE QL?"
2140 DATA " I THINK"
2150 DATA "DOES IT WORRY YOU THAT
          YOU            THINK"
2160 DATA " AM I","YOU ARE"
2170 DATA " COMPUTER"
2180 DATA "WHAT IS THE BEST THING
          ABOUT          COMPUTERS?"
2190 DATA "I"," YOU"
2200 DATA "ARE","AM"
2210 DATA "ME","YOU"
2220 DATA "WERE","WAS"
2230 DATA "YOU","ME"
2240 DATA "YOUR","MY"
2250 DATA "MY","YOUR"
2260 DATA "IS","ARE"
2270 DATA "IM","YOU'RE"
2280 DATA "ME"," YOU"
2290 DATA "MY","YOUR"
2300 DATA "AM","ARE"
2310 DATA "YOURE","I'M"
2320 DATA "MYSELF","YOURSELF"
2330 DATA "YOURSELF","MYSELF"
```

POETRY ENGINE

Our next program generates, as you've probably



guessed from the title, poetry (of a sort). It demands no interaction from you, except for that of admiring the

wonderful output of the program. Here's a sample of the kind of verse it produces:

```
SILENTLY A MAN SIGHED PAST THE WOODS
  NOTHING A CROWD HAD SHOOK
    ...HAD ENDED
  SHE SIGHED IN THE AGONY
...THAT THE WOODS MIGHT BE WAITED
  ASKING, THEY THOUGHT
      ...TURNING, I HOPED


CALMLY A MAN SPOKE BEFORE THE NORTH
  NOTHING SHADOWS HAD SCREAMED
    ...HAD DIED
  SHADOWS CRIED PAST THE PAIN
...THAT THE CAVERN MAY BE WAITED
  HOPING, I SCREAMED
      ...SIGHING, THEY DIED


QUICKLY HE SCREAMED IN THE WOODS
  MUCH A MIST HAD HOPED
    ...HAD SHOOK
  A CROWD SCREAMED IN THE AGONY
...THAT THE NORTH MAY BE CRIED
  CRYING, A MIST DREAMED
      ...TURNING, A CHILD SHOOK
```

CALMLY THEY HELD IN THE DOORWAY
MUCH HE HAD THOUGHT
...HAD SCREAMED
SHADOWS SPOKE TO THE PAIN
...THAT THE NIGHT MIGHT BE SHOOK
ASKING, A CROWD DIED
...WANTING, SHADOWS SCREAMED

SAVAGELY HE SIGHED BEFORE THE DOORWAY
ONLY A MAN HAD THOUGHT
...HAD HOPED
THEY SIGHED IN THE ANGUISH
...THAT THE SUN MIGHT BE STOPPED
WAITING, A CROWD WAITED
...HOPING, A MAN THOUGHT

SAVAGELY THEY WAITED IN THE CAVERN
LITTLE SHE HAD WAITED
...HAD STOPPED
A MIST CRIED PAST THE JOY
...THAT THE SUN MAY BE HOPED
WANTING, A CHILD SIGHED
...WANTING, A MAN STOPPED

QUIETLY SHADOWS DREAMED BEFORE THE SUN
NOTHING SHE HAD STOPPED
...HAD ENDED

```
  A CHILD DIED TO THE JOY
...THAT THE NIGHT OUGHT BE THOUGHT
  SIGHING, I SPOKE
     ...ASKING, A CHILD SPOKE


QUIETLY SHE SIGHED IN THE PATH
  NOTHING A MIST HAD SCREAMED
    ...HAD SIGHED
  A MIST STOPPED BEFORE THE ANGUISH
...THAT THE CAVERN MIGHT BE ENDED
  TURNING, A MIST STOPPED
     ...NEEDING, SHE STOPPED
```

Once you've run it for a while, you may well wish to change the words in the DATA statements, to vary the kind of poetry it produces.

```
10 REMark Poetry Engine
20 BORDER 0:PAPER 7
30 CLS:CLS #0
40 REPeat ENGINE
50 INK RND(1 TO 5)
60 PRINT \\\\
70 aword
80 bword
90 cword
100 iword
110 dword
120 PRINT \"  ";
130 eword
140 bword
```

```
150 PRINT  "HAD ";
160 cword
170 PRINT \"     ...HAD "!
180 cword
190 PRINT \"  ";
200 bword
210 cword
220 iword
230 FWORD
240 PRINT \"...THAT THE "!
250 dword
260 GWORD
270 PRINT "BE ";
280 cword
290 PRINT \"  ";
300 HWORD
310 bword
320 cword
330 PRINT \"      ..."!
340 HWORD
350 bword
360 cword
370 PAUSE 86
380 END REPeat ENGINE
390 REMark ----------
400 DEFine PROCedure aword
410 RESTORE 460
420 FOR J=1 TO RND(1 TO 10)
430  READ A$
440 END FOR J
450 PRINT A$;" "!
460 DATA "SADLY","GENTLY","SLOWLY"
470 DATA "QUICKLY","JOYFULLY","OFTEN"
480 DATA "SAVAGELY","CALMLY","QUIETLY"
490 DATA "SILENTLY"
500 END DEFine aword
```

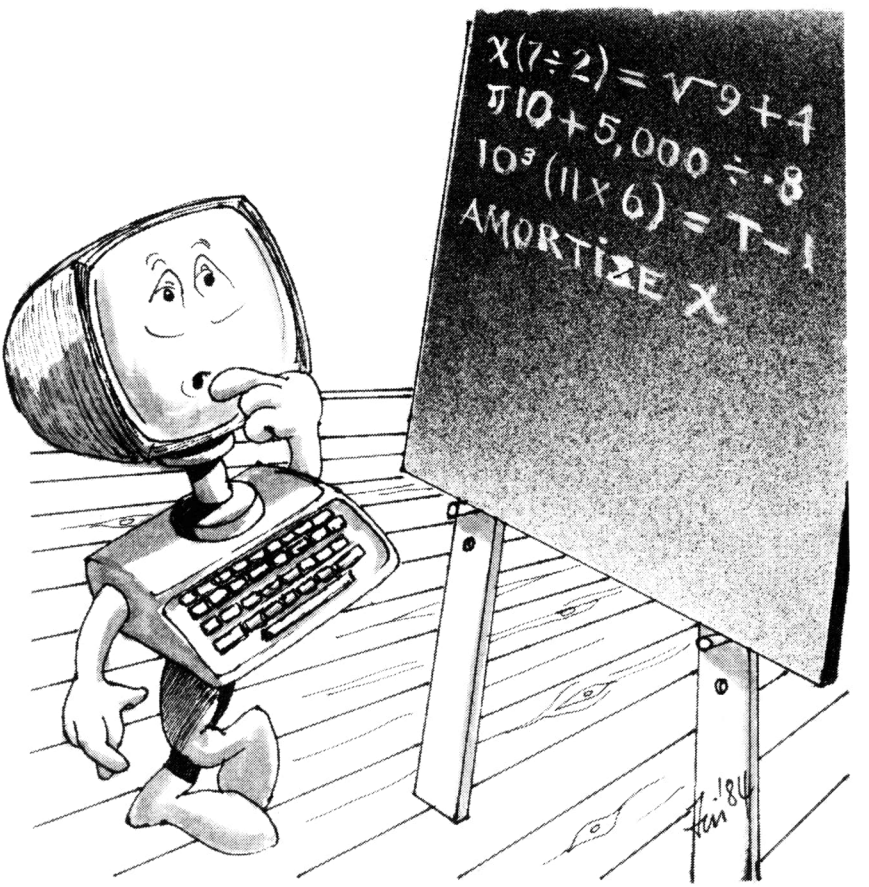
```
510 REMark ----------
520 DEFine PROCedure bword
530 RESTORE 580
540 FOR J=1 TO RND(1 TO 10)
550  READ B$
560 END FOR J
570 PRINT B$;" "
580 DATA "I","HE","THEY","PROPHETS"
590 DATA "A MAN","SHADOWS","A CHILD"
600 DATA "SHE","A CROWD","A MIST"
610 END DEFine bword
620 REMark ----------
630 DEFine PROCedure cword
640 RESTORE 690
650 FOR J=1 TO RND(1 TO 13)
660  READ C$
670 END FOR J
680 PRINT C$;" "
690 DATA "STOPPED","THOUGHT","HOPED"
700 DATA "WAITED","SHOOK","SIGHED"
710 DATA "SPOKE","CRIED","SCREAMED"
720 DATA "ENDED","HELD"
730 DATA "DREAMED","DIED"
740 END DEFine cword
750 REMark ----------
760 DEFine PROCedure dword
770 RESTORE 820
780 FOR J=1 TO RND(1 TO 10)
790  READ D$
800 END FOR J
810 PRINT D$;" "
820 DATA "WOODS","NORTH","SUN"
830 DATA "PATH","SEA","DARKNESS"
840 DATA "NIGHT","VOID"
850 DATA "DOORWAY","CAVERN"
```

```
860 END DEFine dword
870 REMark --------------
880 DEFine PROCedure eword
890 RESTORE 940
900 FOR J=1 TO RND(1 TO 5)
910  READ E$
920 END FOR J
930 PRINT E$;" "!
940 DATA "ALL","MUCH"
950 DATA "NOTHING","LITTLE","ONLY"
960 END DEFine eword
970 REMark --------------
980 DEFine PROCedure FWORD
990 RESTORE 1040
1000 FOR J=1 TO RND(1 TO 5)
1010  READ F$
1020 END FOR J
1030 PRINT F$;" "!
1040 DATA "SORROW","AGONY","PAIN"
1050 DATA "ANGUISH","JOY"
1060 END DEFine FWORD
1070 REMark --------------------
1080 DEFine PROCedure GWORD
1090 RESTORE 1140
1100 FOR J=1 TO RND(1 TO 4)
1110  READ G$
1120 END FOR J
1130 PRINT G$;" "!
1140 DATA "MAY","MIGHT","OUGHT"
1150 DATA "SHOULD"
1160 END DEFine GWORD
1170 REMark --------------
1180 DEFine PROCedure HWORD
1190 RESTORE 1240
1200 FOR J=1 TO RND(1 TO 10)
1210  READ H$
```

```
1220 END FOR J
1230 PRINT H$;", "!
1240 DATA "SIGHING","REACHING","TURNING"
1250 DATA "HOPING","ASKING","NEEDING"
1260 DATA "WAITING","CRYING"
1270 DATA "TAKING","WANTING"
1280 END DEFine HWORD
1290 REMark --------------
1300 DEFine PROCedure iword
1310 RESTORE 1360
1320 FOR J=1 TO RND(1 TO 4)
1330  READ I$
1340 END FOR J
1350 PRINT I$!"THE "!
1360 DATA "TO","IN","PAST","BEFORE"
1370 END DEFine iword
```

MINDBOZ

This is an attempt to write a program which would



work towards the answer to a problem, in an apparently intelligent manner. The computer is

attempting to solve a problem of the type where you think of a numerical code, and the computer tries to guess it. The feedback you give its answers are as 'whites' and 'blacks', where a white is given for a digit which is correct, but is in the wrong position in the code, and a black is given whenever there is a correct digit within the code.

The problem is not totally straightforward, as the computer does not **know,** for certain, which digit produced which result. Mindboz works in a simple manner (although implementing the relatively simple idea behind the program was not particularly easy). Every time a digit appears in a code which is awarded a black, every digit within that code is weighted so that it appears more often in future guesses. The more blacks in that particular code, the higher the weighting each code gets. A much smaller weighting is awarded if the code gets one or more whites. Any code getting neither a black nor a white leads to all the digits within that guess being totally removed from future consideration.

```
10 REMark Mindboz
20  initialise
30 REPeat major_loop
40 IF guess<3 THEN
50  FOR z=1 TO 3
60   b(z)=z+3*guess
70  END FOR z
80 END IF
90 IF guess<3 THEN GO TO 110
100 pick_three_numbers
110 CLS
```

```
120 guess=guess+1
130 PRINT \\\;"       My guess is "
              ;b(1);b(2);b(3)
140 INPUT \"      How many blacks? ";bl
150  IF bl=3 THEN end_of_game:STOP
160  IF bl=2 THEN GO TO 180
170 INPUT \ "    And how many whites? ";w
180 IF w+bl=3 THEN
190  q=3:c(1)=b(1)
200  c(2)=b(2):c(3)=b(3)
210 END IF
220 IF bl+w=0 THEN
230  c(b(1))=0:c(b(2))=0
240  c(b(3))=0
250 END IF
260 IF bl+w=0 THEN GO TO 400
270 IF bl>aid THEN
280  FOR z=1 TO 3
290   e(z)=b(z)
300  END FOR z
310  aid=bl
320 END IF
330 FOR z=1 TO 9
340  FOR d=1 TO 3
350   IF b(d)=c(z) THEN
360    c(z)=c(z)+(bl+w)*100+w*100
370   END IF
380  END FOR d
390 END FOR z
400 END REPeat major_loop
410 DEFine PROCedure pick_three_numbers
420  FOR z=1 TO 3
430   d1=c(RND(1 TO q))
```

```
440     BEEP 100,RND(1 TO 250)
450   IF d1=0 THEN GO TO 430
460    d2=c(RND(1 TO q))
470   IF d2=0 THEN GO TO 430
480 IF INT(d1/10)>INT(d2/10) THEN b(z)=d1
490 IF INT(d1/10)<INT(d2/10) THEN b(z)=d2
500 IF INT(d1/10)=INT(d2/10) THEN b(z)=d1
510 IF b(z)>100 THEN b(z)=b(z)-
        100*INT(b(z)/100):GO TO 510
520 IF b(z)>10 THEN b(z)=b(z)-
        10*INT(b(z)/10):GO TO 520
530   END FOR z
540 IF b(1)=b(2) OR b(1)=b(3) OR b(2)=b(3)
       THEN BEEP 100,250:GO TO 420
550 IF aid>0 THEN
560   count=0
570   FOR z=1 TO 3
580     IF b(z)=e(z) THEN count=count+1
590   END FOR z
600 END IF
610   IF count<aid-1 THEN GO TO 420
620 m=100*b(1)+10*b(2)+b(3)
630 k(guess)=m
640 IF guess<3 THEN GO TO 690
650 count=1
660 count=count+1
670 IF k(count)=m THEN GO TO 420
680 IF count<guess-1 THEN GO TO 660
690 END DEFine pick_three_numbers
700 DEFine PROCedure initialise
710 CLS:CLS #0:BORDER 2,1,1
720 guess=0:q=9:aid=0:count=0
730 DIM b(3):DIM c(9)
740 DIM e(3):DIM k(100)
750 CSIZE 3,1
```

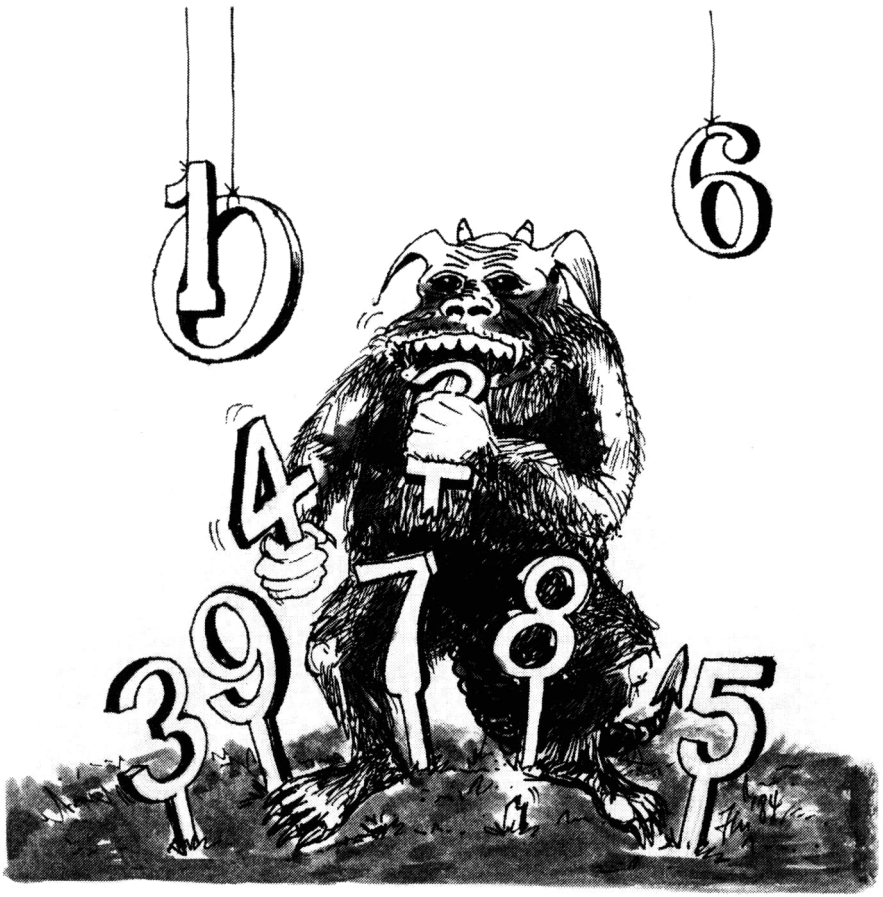
```
760  FOR z=1 TO 9
770   c(z)=z
780  END FOR z
790 FOR z=1 TO RND(1 TO 1000)
800  r=RND
810 END FOR z
820 END DEFine initialise
830 DEFine PROCedure end_of_game
840 PRINT \" I guessed your code of "
              ;b(1);b(2);b(3)
850 PRINT "    in just ";guess;" guesses"
860 END DEFine end_of_game
```

GIGGLE PALACE

This section of the book consists of games designed just for fun. Take part in a round of Nimbo, follow the shifting patterns in Simon, run for your life on the Robot Minefield and indulge your love of gambling with a few rounds of Twenty One.

NIMBO

Nimbo is a game which seems easy at first, but is the



very devil to play . . . and win. You and the wily QL take it in turns to take objects away from the ones on

the screen. There is a limit to how many you can take away each move (and this limit changes from game to game, although it does not change within a particular round of the game). The winner is the person who manages to force the other player to remove the last object.

Once you see it in action, you'll understand how to play it. Then you can start working on perfecting a method of actually beating your QL.

```
10 REMark Nimbo
20 MODE 256
30 CSIZE 3,1
40 PAPER 0:INK 7
50 CLS
60 m=0:e$=0:z=RND(8 TO 16)
70 h=RND(3 TO 5)
80 PRINT "The most you can take is ";h
90 REMark **********************
100 REPeat game_cycle
110 show_it
120 IF e$>0 THEN PRINT "You took ";
                  e$;", and I took ";q
130 FOR k=1 TO z
140  INK RND(2 TO 6)
150 PRINT k;"  ";
160  IF RND>.8 THEN PRINT
170 END FOR k
180 REMark **********************
190 show_it
200 PRINT "How many will you take?"
210 REMark ********************
220 REPeat getnum
```

```
230   e$=INKEY$(-1)
240   IF e$>Ø AND e$<=h THEN EXIT getnum
250 END REPeat getnum
260 PRINT \"So you want ";e$
270 z=z-e$
280 show_it
290 IF z<1 THEN
300   PRINT "You took the last one"
310   PRINT "so I win":STOP
320   ELSE
330 REPeat qcheck
340 q=z-1-INT((z-1)/(h+1))*(h+1)-
          RND(Ø TO 1)+RND(Ø TO 1)

350   IF NOT(q<1 OR q>h) THEN EXIT qcheck
360   END REPeat qcheck
370 show_it

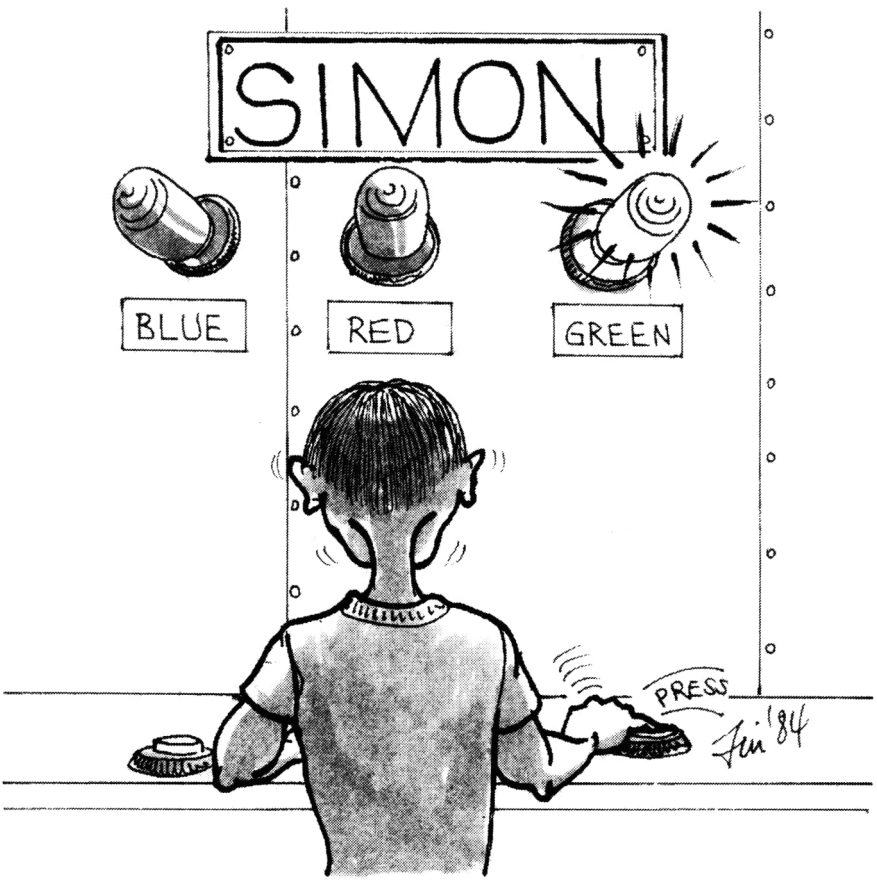
380 REMark **********************
390 z=z-q
400 IF z<1 THEN
410    PRINT "I grabbed the last one"
420    PRINT "so you win":STOP
430 ELSE   show_it
440 END REPeat game_cycle

450 REMark **********************
460 DEFine PROCedure show_it
470 INK RND(2 TO 6)
480   PRINT \;FILL$("-",26)
490    FOR delay=1 TO 100
500    END FOR delay
510 music
520 END DEFine
```

```
530 REMark **********************
540 DEFine PROCedure music
550  BEEP 1000,RND(5 TO 80),
   RND(5 TO 80),6,7,6,7,RND(-1000 TO 1000)
560 END DEFine music
```

SIMON

Although the listing is not very long, so the program



won't take you long to enter into your QL, you'll find Simon a pretty hard act to follow.

The program generates an ever-growing sequence of tones and squares on the screen. You have to copy the sequence. First the QL will place a single coloured square on the screen and sound a note. You have to press a key corresponding to that coloured square. If you have it right, the QL will play that tone (and show that square again) and then show another square and play another tone. You have to re-enter the first one, and then the second one. The sequence will continue to grow, until either you make a mistake, or you manage to copy eleven correctly in a row.

You enter your moves by using the keys '1' to '4', using them in this pattern:

1 2

3 4

The message 'You did it!!' will appear on the screen if you manage to get eleven correct. However, if you make a mistake (see line 480), the discouraging words 'You blew it!' are all you will get.

Note how this program uses the SuperBASIC command BLOCK to place the coloured squares rapidly on the screen.

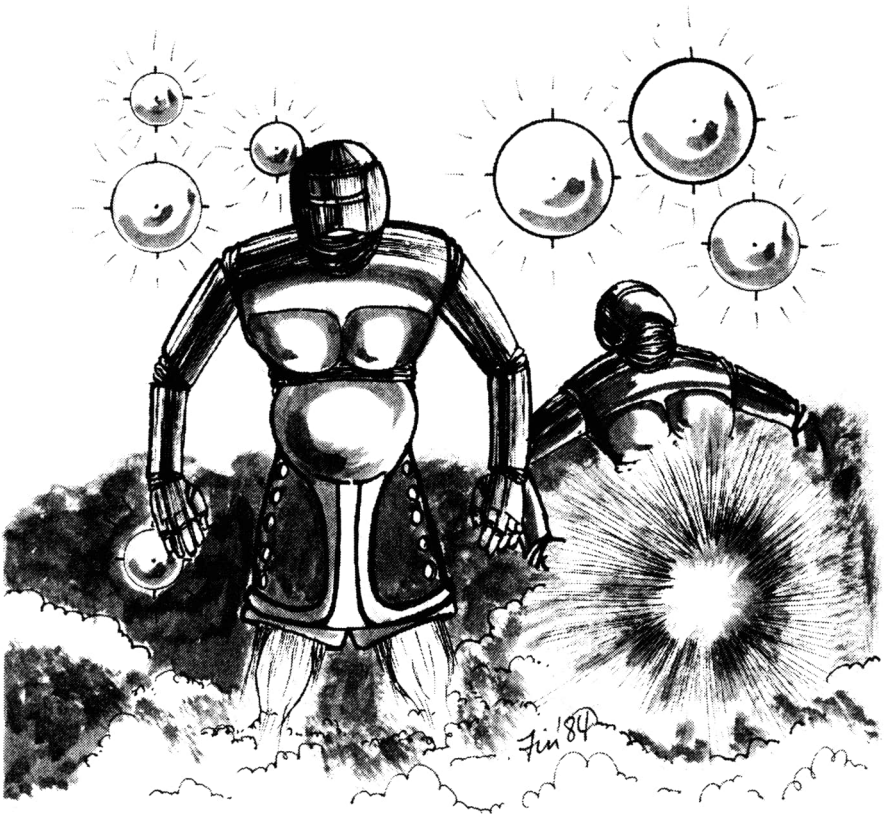
```
10 REMark Simon
20 PAPER 1
30 CLS:CLS #0
40 DIM b(11)
50 FOR j=1 TO 11
```

```
60  b(j)=RND(1 TO 4)
70 END FOR j
80 IF INKEY$<>"" THEN GO TO 80
90 REMark ----------------
100 FOR delay=1 TO 15 STEP .5
110  BEEP 100,delay:PAUSE 7
120 END FOR delay
130  x=1
140 FOR q=1 TO x
150  IF b(q)=1 THEN one
160  IF b(q)=2 THEN two
170  IF b(q)=3 THEN three
180  IF b(q)=4 THEN four
190  BEEP 10000,45*b(q)
200  PAUSE 20
210  CLS
220 END FOR q
230 REMark ----------------
240 FOR z=1 TO x
250   a$=INKEY$
260 IF a$<"1" OR a$>"4" THEN GO TO 250
270  IF a$=1 THEN one
280  IF a$=2 THEN two
290  IF a$=3 THEN three
300  IF a$=4 THEN four
310  BEEP 10000,45*a$
320  PAUSE 7
330 IF a$<>b(z) THEN CLS:GO TO 460
340  CLS
350 END FOR z
360 REMark ----------------
370 PAUSE 100
380 IF x<11 THEN
390  x=x+1
400  GO TO 140
```

```
410 END IF
420 REMark ----------------
430 CSIZE 3,1
440 PRINT \\"   You did it!!"
450 STOP
460 CSIZE 3,1
470  INK 6
480 PRINT \\"    You blew it!"
490  FLASH 1
500 PRINT \"    You scored ";x
510  FLASH 0
520 STOP
530 DEFine PROCedure one
540  BLOCK 100,75,90,12,2
550 END DEFine one
560 DEFine PROCedure two
570  BLOCK 100,75,240,12,3
580 END DEFine two
590 DEFine PROCedure three
600  BLOCK 100,75,90,115,4
610 END DEFine three
620 DEFine PROCedure four
630  BLOCK 100,75,240,115,6
640 END DEFine four
```

ROBOT MINEFIELD

Time now to face the terrors, the dangers, and the



horrors of the Robot Minefield. There you are, peacefully tripping along one day in the sun, when suddenly

you find yourself trapped in a walled area, with a number of manic robots, all bent on your destruction.

The robots are shown on the display as dollar signs, the H is you, and the asterisks (*) are mines which are deadly to both you and robots. Robots, pretty clumsily programmed, can detect you, but they cannot detect mines. Therefore, your salvation lies in getting a mine or two in between you and the robots, so that when they come towards you they'll smash into mines and be wiped out.

The robots have one trick up their sleeves. They can merge into one another, so you may think only one robot is approaching you, when at the last minute it splits into two or more, one going each way around a mine. This is not a game for those who seek the quiet life.

You move by using the cursor keys.

```
10 REMark Robot Minefield
20 initialise
30 print_minefield
40 REPeat game_loop
50   move_human
60   move_robots
70   print_minefield
80 END REPeat game_loop
90 DEFine PROCedure print_minefield
100 AT 0,2
110 INK 7
120 IF tally>0 THEN
130   PRINT "Dead robot tally: ";tally
```

```
140  ELSE PRINT
150 END IF
160 PRINT \
170 FOR bx=1 TO 15
180  PRINT " ";
190  FOR cx=1 TO 15
200   INK 2
210  r$=CHR$(a(bx,cx))
220  IF r$="X" THEN INK 2
230  IF r$="$" THEN INK 6:BEEP 100,20
240  IF r$="*" THEN INK 4:BEEP 1,250:
          BEEP 1,100:BEEP 1,250
250  IF r$="H" THEN INK 7
260   PRINT r$;" ";
270  END FOR cx
280  PRINT
290 END FOR bx
300 END DEFine print_minefield
310 DEFine PROCedure move_human
320 a(d,f)=46
330 IF INKEY$<>"" THEN GO TO 330
340 BEEP 10000,10
350 mo=CODE(INKEY$(-1))
360 BEEP 10000,200
370 IF mo=208 AND d>2 THEN d=d-1
380 IF mo=216 AND d<14 THEN d=d+1
390 IF mo=200 AND f<14 THEN f=f+1
400 IF mo=192 AND f>2 THEN f=f-1
410 IF a(d,f)=42 THEN
420  print_minefield
430  hit_mine
440  music:music:music
450  STOP
460 END IF
470 a(d,f)=72
```

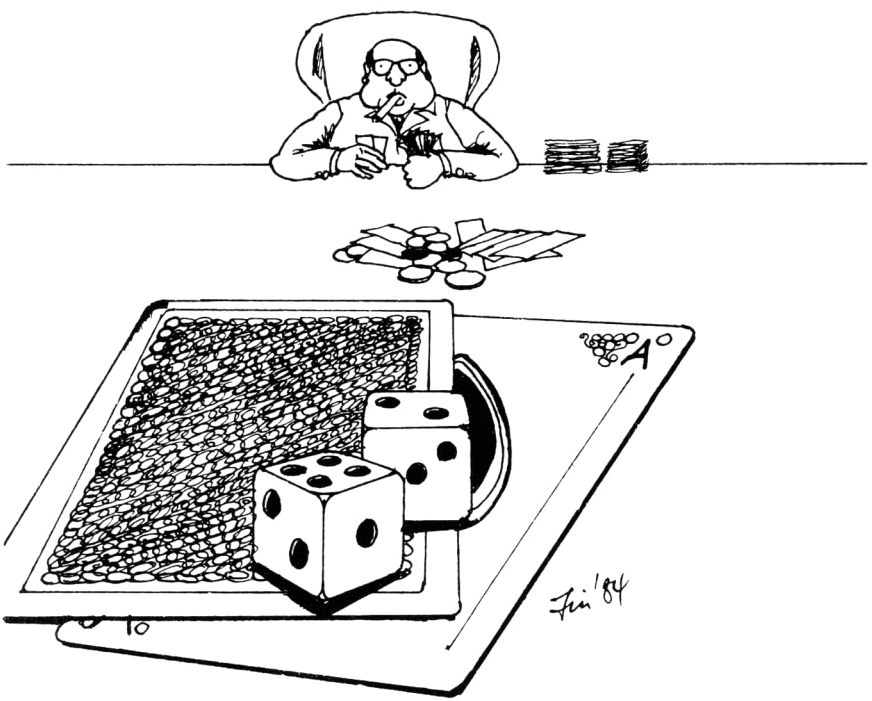
```
480 END DEFine move_human
490 DEFine PROCedure initialise
500 DIM a(15,15):DIM b(4):DIM c(4)
510 PAPER 1:INK 7:CLS
520 BORDER 15,2,3
530 ch=0:tally=0
540 REMark Place walls
550 FOR q=1 TO 15
560  FOR s=1 TO 15
570  a(q,s)=46
580 IF q=1 OR q=15 OR s=1 OR s=15
                        THEN a(q,s)=88
590  END FOR s
600 END FOR q
610  music
620 REMark Place mine
630 FOR bb=1 TO 20
640  cc=RND(3 TO 13)
650  dd=RND(3 TO 13)
660 IF a(cc,dd)=88 THEN GO TO 640
670  a(cc,dd)=42
680 NEXT bb
690 REMark Place robots
700 FOR e=1 TO 4
710  dd=RND(3 TO 13)
720  ff=RND(3 TO 13)
730 IF a(dd,ff)<>46 THEN GO TO 710
740 b(e)=dd:c(e)=ff
750 a(dd,ff)=36
760 NEXT e
770 REMark Place human
780 d=RND(3 TO 13)
790 f=RND(3 TO 13)
800 IF a(d,f)<>46 THEN GO TO 770
810 a(d,f)=72
```

```
820  music
830 END DEFine initialise
840 DEFine PROCedure move_robots
850  tally=0
860  FOR e=1 TO 4
870 IF a(b(e),c(e))=42 THEN tally=tally+1:
                music:GO TO 1070
880 x=b(e):y=c(e)
890 IF b(e)<d THEN b(e)=b(e)+1
900 IF b(e)>d THEN b(e)=b(e)-1
910 IF c(e)>f THEN c(e)=c(e)-1
920 IF c(e)<f THEN c(e)=c(e)+1
930 IF b(e)<2 THEN b(e)=2
940 IF c(e)>14 THEN b(e)=14
950 IF c(e)<2 THEN c(e)=2
960 IF c(e)>14 THEN c(e)=14
970 a(x,y)=46
980 IF a(b(e),c(e))=42 THEN tally=
              tally+1:music:GO TO 1070
990 IF a(b(e),c(e))=72 THEN
1000  a(b(e),c(e))=36
1010  music:music
1020  print_minefield
1030  killed_human
1040  STOP
1050 END IF
1060 a(b(e),c(e))=36
1070 END FOR e
1080 IF tally<ch THEN tally=ch
1090 ch=tally
1100 IF tally=4 THEN
1110  print_minefield
1120  victory
1130  music:music
1140  STOP
```

```
1150 END IF
1160 END DEFine move_robots
1170 DEFine PROCedure hit_mine
1180  AT 0,2
1190 INK 7
1200  PRINT "You hit a mine..."
1210  music
1220  FOR delay=1 TO 1000
1230  END FOR delay
1240  AT 0,2
1250 PRINT "                    "
1260 END DEFine hit_mine
1270 DEFine PROCedure victory
1280  AT 0,2
1290 INK 7
1300  PRINT "You've defeated the robots!"
1310  music:music:music
1320 END DEFine victory
1330 DEFine PROCedure killed_human
1340  AT 0,2
1350 INK 7
1360 PRINT "A robot has killed you!"
1370  music:music:music
1380 END DEFine killed_human
1390 DEFine PROCedure music
1400  FOR musique=1,200,36.7,88,
                    3,99,2,1,2,1,2,1
1410  BEEP 1000,musique
1420  END FOR musique
1430 END DEFine music
```

TWENTY ONE

This is the QL's dice version of Blackjack. The game is reasonably simple, but demands a degree of cool



thinking, as the Microdrives heat up, and the strain starts to get to you.

You are playing against the QL. You get to move first. You 'roll the die'' as many times as you like, aiming to get a total as close as possible to, but not exceeding, 21. You can stop adding to the total whenever you like.

If you exceed 21, that is you 'bust', the QL wins that round automatically. A game consists of five rounds, and the winner of most rounds — naturally enough — wins the game.

Just follow the prompts as the game gets underway for a challenging round or five.

```
10 REMark TWENTY ONE
20 HS=0:CS=0
30 BORDER 30,2,2
40 PAPER 1
50 CLS:CLS #0
60 CSIZE 3,1
70 REPeat mainloop
80    INK 6
90    h=0:c=0:n=0
100   CLS
110 PRINT \\" Press 'R' to roll"
120 PRINT "  or 'S' to stand"
130 IF INKEY$<>"" THEN GO TO 130
140 a$=INKEY$(-1)
150 IF a$<>"S" AND a$<>"s" AND a$<>"r" AND
          a$<>"R" THEN GO TO 140
160 BEEP 1000,40
170 IF a$="s" OR a$="S" THEN PRINT
          " Your total is ";h:GO TO 260
180 delay
```

```
190 r=RND(1 TO 6)
200 PRINT \ " You rolled a ";r
210 h=h+r
220 PRINT " Your total is ";h
230 lineprint
240 delay
250 PRINT:GO TO 110
260 IF c>h AND c<22 OR c>21 OR h>21 OR
          h=21 AND c=21 THEN GO TO 350
270 r=RND(1 TO 6)
280 delay
290 INK 4
300 PRINT \" I rolled a ";r
310 c=c+r
320 PRINT " so my total is ";c
330 delay
340 GO TO 260
350 INK 2
360 IF h=c OR h>21 AND c>21 THEN GO TO 440
370 lineprint
380 INK 4
390 IF (c>h OR h>21) AND c<22 THEN
            FLASH 1:PRINT ," I";:CS=CS+1
400 IF (c<h OR c>21) AND h<22 THEN
          FLASH 1:PRINT ," You";:HS=HS+1
410 PRINT " win!!"
420 GO TO 460
430 INK 1
440 PRINT "That round is a"
450 PRINT "   dead heat"
460 delay
470 lineprint
480 FLASH 0
490 PRINT " The scores are:"
500 delay
```

```
510 FLASH 1
520 PRINT " You: ";HS;", and me ";CS
530 FLASH 0
540 lineprint
550 IF CS+HS=5 THEN GO TO 650
560 PRINT \"Stand by..."
570 delay:delay
580 END REPeat mainloop
590 DEFine PROCedure delay
600 BEEP 1150,RND(60 TO 90)
610  FOR waiting=1 TO 1000
620  END FOR waiting
630 BEEP 1220,RND(20 TO 60)
640 END DEFine delay
650 PRINT \"End of the game"
660 delay
670 PRINT \" Final scores:"
680 delay
690 PRINT "  You: ";HS
700 PRINT "   Me: ";CS
710 IF CS>HS THEN PRINT " so I win!"
720 IF HS>CS THEN PRINT " so you win!"
730 STOP
740 DEFine PROCedure lineprint
750 PRINT "--------------------"
760 END DEFine lifeprint
```

SIMULATIONS

You walk a thin line between two compromises when writing simulation programs on the QL. These are the need to (a) replicate reality and (b) simplify the harsh complexities of real life. Out there in the 'real world', cause rarely follows effect as neatly, cleanly and predictably as you'd like it to. However, to create managable simulations, we have to assume that action A, under circumstances B, will generally lead to result C.

This does not mean, however, that computer simulations need lack veracity. Think about what happens when you make a decision on any action to be taken at school, at home or at work. In most cases, you are forced to ignore certain factors simply to be able to actually make a decision. You'd probably find that you would never get around to making decisions at all if every single element which could be effected by the decision had to be weighed and measured.

I try to keep this in mind whenever I write computer simulations. The decisions the user must make are clearly laid out. Most of the relevant facts needed to make the decision are revealed, so that the user can quickly learn the effects which these decisions will have.

When you write simulations on your QL, you should try and ensure that the mechanisms of running them are as simple as you can make them, so the user can concen-

trate on the 'what should I do?' rather than on the 'how can I do it?'. Note, however, that achieving the goals set by the programs should not be simple. Real life is rarely simple. A simulation which purports to replicate reality must contain some of the rough edges of real life.

You'll see this in action in the first of the games which follows, called Stockbroker. Thirteen areas of industry are represented in the stocks and shares available to you in this program — from Teleamerications Inc. (telecommunications) to Aerosoar Industrial (space technology) — and you must trade these to increase your starting stake by a factor of ten. You receive solid advice on the prospects of the stock available to you, and you must use this information, plus your own knowledge of how the market behaves in certain circumstances, to determine which stocks you will buy and sell.

Computer programs which depend heavily on random factors, outside the control of the player, are rarely satisfactory for very long. One characteristic of a good program is that the person using it becomes better with time. If practice does not lead to improvement, the fault almost always lies within the program.

We have two simulations in this section. The first one, Stockbroker, enables you to make a fortune (or go bust!) in the City. The second is a graphic simulation showing the evolution of a colony of cells.

STOCKBROKER

In this simulation, you get a chance to show your



market savvy by buying and selling stocks and bonds in the City. You start with a generous allowance of

cash, and can choose from stock offered by thirteen leading companies and from a bond offering. The aim of this simulation is to compound your original investment tenfold.

You'll find that this is not easy to do, even though you have professional (and reasonably accurate) advice on the prospects of each of the companies you are considering. If the pace becomes too hot, you can always pull out of the market.

Unemployment and inflation fluctuate throughout the game, and — in general — these determine whether you'll have a bull or bear market. As you know, although the action tends to be upwards in a bull market, there are some stock offerings which behave against the trend. Some companies grow in a bear market, so you cannot follow the 'strategy' of buying in a bear, and selling after you've waited for as long as you can in a bull.

Simply attending the stock market carries an overhead. Your 'attendance fee' is related to your net worth and the current inflation rate. This fee has been included to ensure you don't simply invest in undervalued shares, and then wait for turn after turn before selling. You could not do that in real life, and the costs imposed will certainly discourage you from trying to do it in this simulation.

You wisely decide to seek professional advice before pouring your savings into the market. You tell your adviser that you're looking for a mixed portfolio, with

some steady stocks to provide security, and a few more risky ventures which could provide significant gains. A week later, the following report lands on your desk. Read it carefully. It contains an accurate assessment of thirteen stocks, and a bond issue, and your future fortunes will depend on how well you understand, and apply, its advice.

'INSIDER ADVICE TO INVESTOR'

Following your instructions to suggest a mixed portfolio which will combine some stability, with the possibility of significant financial gain, we have decided to suggest the following thirteen stocks, and a bond issue. You should find that a wide spread of purchases is the best path to follow, as the more stable offerings can offset — to some extent — the risks posed by some of the more high-flying stocks.

Keep in mind that no actions based on our assessment are guaranteed to work. Public perception of a company's potential markedly affects the performance of that stock on the market. As well, unexplained fluctuations can occur from time to time. In general, however, you will find that prices tend to grow in a bull market, and fall in a bear market. The nature of the market is related to a number of external economic indicators. We suggest that you look at just two of these — the unemployment figure and the inflation rate. When both of these are high, you'll see the market loses confidence, and prices tend to fall.

Here, then, are the companies we've determined are the ones you should consider. Make your selection from the following:

CANDELABRA ELECTRIC (electric utility): This is a stable company, offering a 5% yield. However, electric utilities in general, including Candelabra Electric, initiated a number of major projects over a decade ago when energy was cheap, and growth looked likely to continue indefinitely. These major projects are now coming on stream, and represent significant over-capacity. Therefore, although the company is stable, the long term prospects for its stock are only fair.

BEGGARMAN BANKING CORPORATION (industries investment): Due to mismanagement in the past, the current president is faced with trying to bring the company's ballooning expenses under control. As a result, the yield on Beggarman is just 1%, but the very long term prospects for the stock are fair to good.

HIGH CARBON SMELTERS INC. (steel): This stock yields 3%, mainly due to the massive government support the steel industry receives. Despite drastic cost-cutting (which has shaved employment at the company's plants around the country by 65%) the company offers few, if any, long-term prospects.

GOODLOOKER APPAREL INC. (clothing): The clothing market tends to be related to the strength of the housing market, and many purchases in the field are in the non-essential or 'luxury' category. Goodlooker has a solid background (over 20 years trading) and its

long term prospects could be very good. However, like its competitors in the field, the company's fortunes are dependant upon the whims of fashion. Therefore, this could be a volatile stock. The price-to-earnings ratio is fairly high, and a yield of 2% is returned.

DENCHABYTE COMPUTERS INC. (electronics): The market need for computer products produced by D.C.I. appears to be constantly growing. As one of the pioneers in the field, D.C.I. is unlikely to be effected by any shakeout in the industry. The stable company offers a fair to good return in the long run, although much of the current profit is being ploughed back into research and development. The yield on the stock is 3%, and if the current demand for ever-increasing computer power continues (as is expected for at least the next decade) D.C.I. should exhibit almost unchecked-growth.

AEROSOAR INDUSTRIAL (space technology): With the promise of a significant contract from the armed services, Aerosoar stock could, indeed, soar. It is currently returning a yield of 6%, and if the defence contracts eventuate, stock will become very valuable. However, although this stock could return significant dividends, it is highly speculative, and the long-term prognosis for the company is not good.

RYDING TRUCKERS INC. (freight haulage): Despite an expected growth of 3% to 4% in the coming year, Ryding has been hit by nationwide deregulation, which has already sounded the death-knell for many badly-run carriers. The stock has a 2% yield. Because

Ryding is one of the country's biggest haulers, its standing is firm and the long-term prospects are good, although dramatic gains are not expected.

PINK BOX HOUSING INC. (housing): Pink Box finds its building costs are rising more slowly than housing prices, so its steady growth pattern of the last decade is expected to continue. This solid company deserves serious consideration as a back-up to more wider-ranging investments.

CLARENDON CHEMANTIC (medical consumer products): Clarendon stock provides no yield. The company has only been trading for three years, and while some of its products have found ready acceptance, major investment in other lines has been lost. Clarendon need a major consumer success this year to justify the reputation the company gained in its early months. If it achieves this, and inside sources suggest a 'winner' of a product is due to be launched this year, stock could leapfrog in value. However, great care is needed. This year looks like boom or bust for the company.

ARCANE EXPLORATIONS (oil and minerals prospecting): Despite a high yield of 7% due, in part, to large North Sea leases, the company has shown little growth since the original petro-panic caused by the first Arab oil embargo. The long term prospects are bad.

GADGET GOODIES INC. (electrical appliances): a yield of 4% and an indication that Europeans are continuing their love affair with electronic devices,

suggests some sectors of the appliance industry are going to boom this year. G.G.I. may well be part of this boom. However, it has not displayed significant innovation in recent years, and has failed to capitalise on the potential of the silicon chip. Despite this, it has a good range of solid-selling products, so the company is in no danger of fading from the scene, although pyrotechnics are not to be expected.

PINTPOT FORKLIFTS INC. (industrial equipment): At long last, the major depression of the past 36 months which has hit the makers of major industrial equipment, appears to be over. This company's stock returns no yield, but its products have a good reputation for reliability, so Pintpot is likely to be an early beneficiary of the pickup. The long-term prospects are good.

TELEAMERICATIONS INC. (telecommunications): Deregulation in this field means it is hard to pick the winners which may emerge. However, T.A.I. has been around a long time, and industry observers feel it is likely to be at the forefront of the deregulation survival battle to be fought in the coming years. There is only a 1% yield, as most earnings are re-invested. The future prospects of the company: slow, but with solid growth.

FIREBELL BONDS: Security and significant income (8%) make Firebell worthy of inspection. Bonds do not appreciate but most serious investors include some for stability in their portfolios.

Armed with that information, you can now go and make

your fortune. Throughout the simulation, you'll be provided with information on the fourteen stocks you have nominated. Your market report will also tell you the current unemployment and inflation rates, and the state of the market. At any time you have four options: sell, buy, 'stand steady' (that is, wait for a favourable change in the market before acting) or quit.

If a stock's value rises above 60, the stock will split two for one. The price per share will be halved, and your holdings (if any) of that stock, will double.

Here's your chance, then, to try out your investment system on your computer, before pouring real pounds into the Exchange.

```
10 REMark Stockbroker
20 initialise
30 REPeat cycle
40  r$=""
50   CLS
60  cash=cash-INT(inflat*(cash+value)/10000)
70  environment
80  trading
90  dividends_payment
100  splits
110  market_movements
120 IF cash+value>100*1000 THEN
130  PRINT "       Well done!"
140  PRINT "You've succeeded in multiplying
                                    your"
150  PRINT "starting stake of £10,000 by ten"
160  PRINT \"Here is your portfolio..."
170  PRINT \
```

```
180  value=0
190 FOR j=1 TO 14
200  IF h(j)<>0 THEN
210    INK RND(0 TO 4)
220    PRINT n$(j);
230   PRINT " ";h(j);"  £";h(j)*v(j)
240   value=value+h(j)*v(j)
250  END IF
260 END FOR j
270 END IF
280 IF r$="q" OR r$="Q" THEN
290  cash=INT((cash+.5)*10)/10
300  PRINT \"     Cash worth £";cash
310  PRINT " Holdings worth £";value
320  PRINT " Total worth is £";value+cash
330  STOP
340 END IF
350 IF cash+value<1 THEN
360  PRINT \"You have failed in the City"
370  STOP
380 END IF
390 END REPeat cycle
400 REMark ---------------
410 DEFine PROCedure environment
420 REMark ** Unemployment **
430 IF unemploy>10 THEN
440  market=-1
450  a$="Bear"
460 END IF
470 IF unemploy<5 THEN
480  market=1
490  a$="Bull"
500 END IF
510 IF RND>.8 THEN GO TO 610
520 unemploy=INT((unemploy+uchange+.5)*10)/10
530 IF unemploy<3.7 THEN
```

```
540  uchange=-uchange
550  unemploy=3.8
560 END IF
570 IF unemploy>13.6 THEN
580  uchange=-uchange
590  unemploy=13.5
600 END IF
610 PRINT "Unemployment ";INT(unemploy*
          100)/100;"%  ";
620  INK  4
630 PRINT "  Market is ";a$
640 BEEP 1000,10:PAUSE 10:BEEP 1000,3
650 IF inflat>9 THEN
660  market=-1
670  a$="Bear"
680 END IF
690 IF inflat<4.5 THEN
700  market=1
710  a$="Bull"
720 END IF
730 REMark ** Inflation **
740 IF RND>.8 THEN GO TO 850
750 inflat=INT((inflat+ichange+.5)*10)/10
760 IF inflat<4 THEN
770  inflat=4.1
780  ichange=-ichange
790 END IF
800 IF inflat>12 THEN
810  inflat=11.9
820  ichange=-ichange
830 END IF
840  INK 2
850 PRINT "    Inflation ";
          INT(inflat*10)/10;"%"
860 REMark ** Bull/Bear **
```

```
870 IF RND<.8 THEN GO TO 940
880 market=-market
890 IF a$="Bull" THEN
900  a$="Bear"
910  GO TO 940
920 END IF
930 a$="Bull"
940 END DEFine environment
950 REMark -----------------
960 DEFine PROCedure trading
970  print_out
980 IF INKEY$<>"" THEN GO TO 980
990 PRINT "-->Do you wish to sell (A),
                        buy (B),"
1000 PRINT "        Stand steady (C)
                   or Quit (Q)?"

1010 r$=INKEY$(-1)
1020 IF r$<>"A" AND r$<>"a" THEN
1030  IF r$<>"B" AND r$<>"b" THEN
1040   IF r$<>"C" AND r$<>"c" THEN
1050    IF r$<>"Q" AND r$>"q" THEN
1060     GO TO 1010
1070    END IF
1080   END IF
1090  END IF
1100 END IF
1110 BEEP 20000,100

1120 IF r$="Q" OR r$="q" THEN GO TO 160
1130 IF r$="c" OR r$="C" THEN GO TO 1640
1140  CLS
1150  environment
1160  print_out
1170 IF r$="A" OR r$="a" THEN GO TO 1390
```

```
1180 REMark ** Buy **
1190 PRINT "Enter stock number to buy,
                  0 to end"
1200 INPUT wh
1210 IF wh=0 THEN GO TO 1640
1220 IF wh<1 OR wh>14 THEN GO TO 1200
1230 PRINT n$(wh);" at £";v(wh);" each"
1240 PRINT "How many do you want? ";
1250 INPUT ho
1260 IF ho*v(wh)>cash THEN
1270  PRINT "Your funds will not
              permit that"
1280  PAUSE 50
1290  GO TO 1240
1300 END IF
1310 FLASH 1
1320 PRINT ">>> Transaction being
             processed <<<"
1330 FLASH 0
1340  BEEP 5000,200:PAUSE 55:BEEP 10000,6
1350 PAUSE 12
1360 cash=cash-v(wh)*ho
1370 h(wh)=h(wh)+ho
1380 GO TO 1140
1390 REMark ** Sell **
1400 PRINT "Enter number to sell, 0 to end"
1410 INPUT wh
1420 IF wh=0 THEN GO TO 1640
1430 PRINT "You hold ";h(wh);" of"
1440 PRINT n$(wh);". How many to"/"sell?"
1450 INPUT ho
1460 IF ho>h(wh) THEN
1470  PRINT "You do not have that many"
1480  PAUSE 50
```

```
1490  GO TO 1400
1500 END IF
1510 PRINT n$(wh);":  £";v(wh);" each"
1520 FLASH 1
1530 PRINT ")))  Transaction being
                          processed  ((("
1540 FLASH 0
1550 BEEP 10000,6:PAUSE 50:BEEP 10000,100
1560 cash=cash+v(wh)*ho
1570 h(wh)=h(wh)-ho
1580 IF wh=0 THEN GO TO 1640
1590 CLS:INK 7:PAPER 2:STRIP 1
1600 PRINT \"You can continue to
                    sell if you like "
1610 STRIP 0:PAPER 7
1620 print_out
1630 GO TO 1400
1640 END DEFine trading
1650 REMark ------------
1660 DEFine PROCedure print_out
1670 value=0
1680 FOR j=1 TO 14
1690  INK 2
1700 IF v(j)=0 THEN GO TO 1830
1710 IF j<10 THEN PRINT " ";
1720 PRINT j;" )";
1730  INK 3
1740 PRINT n$(j)(1 TO 12);"  ";
1750  INK 2
1760 PRINT "£";v(j);
1770 IF h(j)<>0 THEN
1780  INK 3
1790  PRINT ,h(j);"  £";h(j)*v(j)
1800  value=value+h(j)*v(j)
1810   ELSE PRINT
```

```
1820 END IF
1830 END FOR j
1840  INK 1
1850 PRINT "        Net worth - £";cash+value
1860  INK 2
1870 PRINT "Cash fund: £";cash;"   Shares
                                    : £";value
1880 END DEFine print_out
1890 REMark ------------
1900 DEFine PROCedure dividends_payment
1910 FOR j=1 TO 14
1920  IF h(j)<>0 THEN
1930   cash=cash+v(j)*h(j)*y(j)/5200
1940  END IF
1950 END FOR j
1960 cash=INT(cash*100)/100
1970 END DEFine dividends_payment
1980 REMark ---------------
1990 DEFine PROCedure splits
2000 FOR x=1 TO 14
2010  IF v(x)>60 THEN
2020    CLS
2030    v(x)=INT(v(x)/2)
2040  FOR uu=1 TO 12
2050   INK RND(1 TO 5)
2060   BEEP 1000,10*uu
2070   tab$="              "(1 TO uu)
2080   PRINT tab$;"*)"
2090  END FOR uu
2100    INK 4
2110    PRINT n$(x);
2120    INK 3
2130    PRINT " splits 2 for 1"
2140    PAUSE 50
2150    h(x)=2*(h(x))
2160  END IF
```

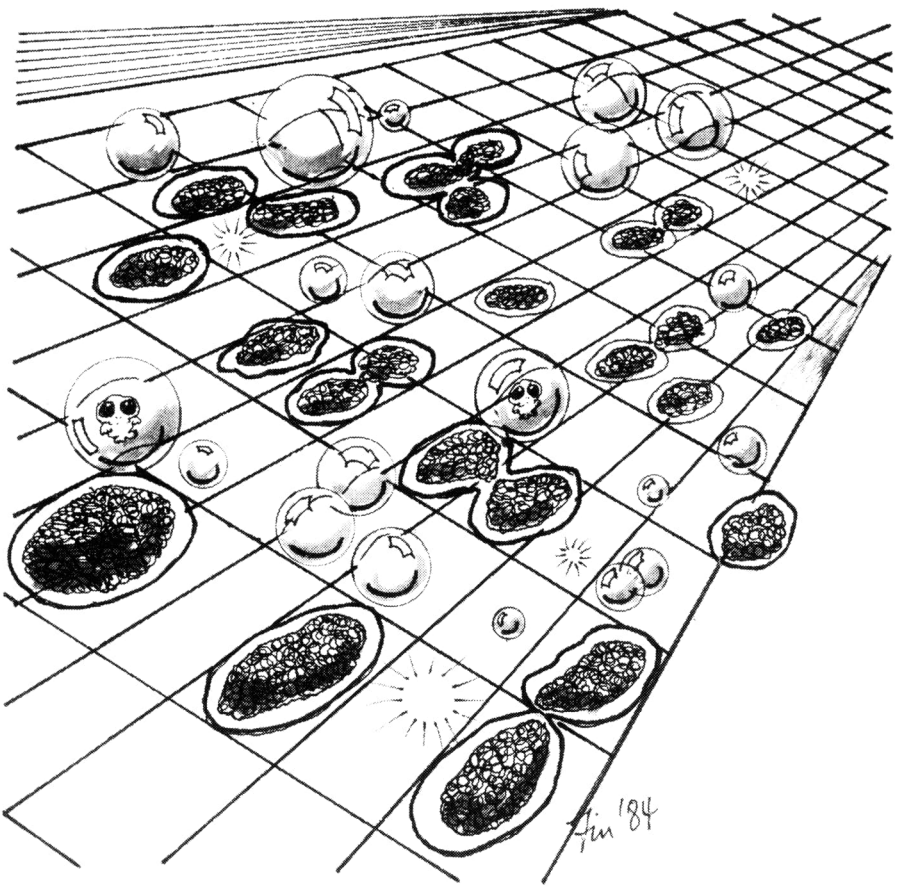
```
2170 END FOR x
2180 END DEFine splits
2190 REMark -----------------
2200 DEFine PROCedure market_movements
2210 IF RND>.4 THEN GO TO 2270
2220 m=RND(1 TO 10)
2230 FOR j=1 TO 14
2240  v(j)=v(j)+market*f(j,m)
2250  IF v(j)<1 THEN v(j)=0
2260 END FOR j
2270 END DEFine market_movements
2280 REMark -----------------
2290 DEFine PROCedure initialise
2300  RESTORE
2310 BORDER 0:PAPER 7:INK 1
2320 CSIZE 0,0
2330 CLS:CLS £0
2340  REMark n$ = name of stock
2350  REMark  y = yield
2360  REMark  v = starting values
2370  REMark  h = holdings
2380  REMark  f = changes in worth
2390 DIM n$(14,25),y(14),v(14)
2400 DIM h(14),f(14,10)
2410 unemploy=RND(3 TO 7)
2420 uchange=RND
2430 inflat=RND(4 TO 8)
2440      value=0
2450 ichange=RND
2460 cash=0
2470 a$="Bull"
2480 cash=10010
2490 market=1
2500 FOR j=1 TO 14
2510  READ n$(j)
```

```
2520  READ y(j)
2530  v(j)=50+RND(1 TO 10)-RND(1 TO 10)
2540 END FOR j
2550 REMark ** Stock movements **
2560 FOR j=1 TO 14
2570  FOR k=1 TO 10
2580   READ f(j,k)
2590  END FOR k
2600 END FOR j
2610 END DEFine initialise
2620 REMark ---------------
2630 DATA "CANDELABRA ELECTRIC",5
2640 DATA "BEGGAMAN BANKING",1
2650 DATA "HIGH CARBON SMELTERS",3
2660 DATA "GOODLOOKER APPAREL",2
2670 DATA "DECHABYTE COMPUTERS",3
2680 DATA "AEROSOAR INDUSTRIAL",6
2690 DATA "RYDING TRUCKERS",2
2700 DATA "PINK BOX HOUSING",0
2710 DATA "CLARENDON CHEMANTIC",0
2720 DATA "ARCANE EXPLORATIONS",7
2730 DATA "GADGET GOODIES",4
2740 DATA "PINTPOT FORKLIFTS",0
2750 DATA "TELEAMERICATIONS",1
2760 DATA "FIREBELL BONDS",8
2770 REMark ** BULL DATA **
2780 REMark CANDELABRA
2790 DATA 3,3,0,1,3,-8,2,3,12,-6
2800 REMark BEGGAMAN
2810 DATA 4,2,4,0,4,3,2,1,2,3
2820 REMark HIGH CARBON
2830 DATA 1,-1,1,0,0,-1,-2,1,-2,0
2840 REMark GOODLOOKER
2850 DATA 12,14,-10,-13,0,8,-7,12,12,10
2860 REMark DENCHABYTE
2870 DATA 6,5,6,-1,5,6,8,0,6,5
```

```
2880 REMark AEROSOAR
2890 DATA 12,-20,9,-20,0,15,20,3,-19,1
2900 REMark RYDING
2910 DATA 6,5,4,-3,4,0,-9,4,3,2
2920 REMark PINK BOX
2930 DATA 3,2,1,0,2,1,3,2,1,3
2940 REMark CLARENDON
2950 DATA 15,-10,15,-20,3,-19,20,-3,-4,10
2960 REMark ARCANE
2970 DATA 1,0,2,0,1,0,2,0,2,0
2980 REMark GADGET
2990 DATA 5,4,3,-1,-2,0,-4,2,-5,6
3000 REMark PINTPOT
3010 DATA 7,6,5,-4,-3,-1,12,3,2,0
3020 REMark TELEAMERICATIONS
3030 DATA 1,2,1,0,-1,-2,0,1,2,-1
3040 REMark FIREBELL
3050 DATA 0,0,0,0,0,0,0,0,0,0
```

LIFE

This program produces an infinite series of evolving



designs. The patterns develop according to the rules of the famous computer game of LIFE, developed by

John Conway while at Gonville and Caius College at Cambridge. Martin Gardiner spread the game throughout the world when he wrote about it in *Scientific American* in October, 1970.

In LIFE, cells are born, grow and die according to rules which Conway invented. Each cell on a grid (the colony of cells is imagined to be evolving on a grid) is surrounded by eight others, and the state of those eight other cells dictates what happens to the cell in question in the following generation.

The rules which govern the evolution of the cells are as follows:

* If a cell has two or three surrounding it, it survives to the next generation.

* If there are three, and just three, full cells next to an empty one, a cell will be 'born' in that empty space in the next generation.

* Any cell with four of its neighbouring cells occupied dies in the next generation.

Don't worry, you don't have to know the rules, as the QL interprets them quite happily by itself. The rules produce patterns which are far more attractive (and far less predictable) than you could possibly imagine by reading the rules, as you can see from these sample patterns produced by LIFE in action:

```
--------------------------
Generation 1
 *       *       *           *
 *   *       * *   *   *   *
 * *     *                 * * *
 *     * * *     *     *       * *
    * *     * * *
 *     * * * * *     *     *
    * *     * * *     *
            * *     * * *     *
            * *     * * *     *
    * *     * * *     *
 *     * * * * *     *     *
    * *     * * *
 *     * * *     *     *       * *
 * *     *                 * * *
 *     *       * *   *   *
 *       *
--------------------------
Generation 1
                    * * *     *
 *       *   * *       *
 *     * * * * * *       *
       * *     * *   * * * *
 *     * * *     * * * *   *
    *     *   * * * * *
 *       *   * *       *     *
 *     * *             * *
 *     * *             * *
 *       *   * *       *     *
    *     *   * * * * *
 *     * * *     * * *   *
       * *     * *   * * * *
 *     * * * * * * *       *
 *       *   * *       *
                    * * *     *
```

```
--------------------------
Generation 2
              * * * * *
    * * *                 * *
                              *
                              *

  * *                         *
  *       *     *             *
    * * * * *   * * * * * *
    * * * * *   * * * * * *
  *       *     *             *
  * *                         *

                              *
                              *
    * * *                 * *
              * * * * *

--------------------------
Generation 3
      *           * * * * * *
      *           * *   * * *
      *                       *


  * *
  *       *     * * * * * *   *
    * * * * * * * * * * * *
    * * * * * * * * * * * *
  *       *     * * * * * *   *
  * *

      *                       *
      *           * *   * * *
      *           * * * * * *
```

```
--------------------------
Generation 4
                  *         *
    * * *         *            *
                              *


 * *              * * *
 *       *
    * * *                     *
    * * *                     *
 *       *
 * *              * * *


                              *
    * * *         *            *
                  *         *

--------------------------
Generation 5
       *
       *                      * *
       *

                     *
 * *                 *
 *       *           *
    * * *
    * * *
 *       *           *
 * *                 *
                     *

       *
       *                      * *
       *
```

```
--------------------------
Generation 6

    * * *



  * *           * * *
  *       *
    * * *
    * * *
  *       *
  * *           * * *

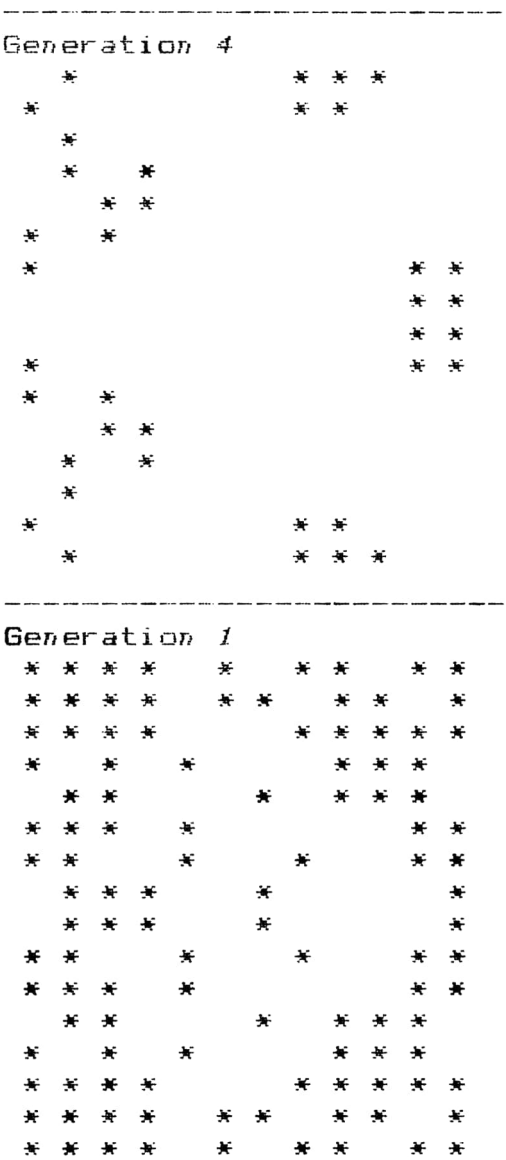


    * * *

--------------------------
Generation 1
  *     * * *     *           * *
        *     *         * * *
    *             * *     *     * *
  * *     *                 * * * *
  *           *     * * * * * * *
    *     * * * * * * * * *
  *     * *             * *       *
    * * *     *     *     * * *
    * * *     *     *     * * *
  *     * *             * *       *
    *     * * * * * * * * * *
  *           *     * * * * * * *
  * *     *                 * * * *
    *             * *     *     * *
        *     *         * * *
  *     * * *     *           * *
```

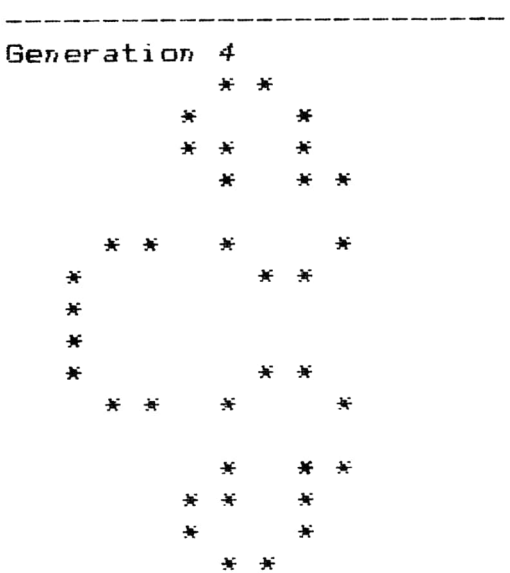
```
--------------------------
Generation 2
    * *     * *     * * * *
      *     *         *
 * *     * * * *           *
 * * *     *
 *
 * *
 *                       *
    *     * *     * *     * * *
    *     * *     * *     * * *
 *                       *
 * *
 *
 * * *     *
 * *     * * * *           *
      *     *         *
    * *     * *     * * * *
```

```
--------------------------
Generation 3
    * *     * *     * * *
 *                 *     *
 *
      *     *
      *
 * *
 *     *                       *
                           * *
                           * *
 *     *                     *
 * *
      *
      *     *
 *
 *                 *     *
    * *     * *     * * *
```


Generation 4
Generation 1

```
--------------------------
Generation 2
 *       *     *     * *     * *
                   *
                   * *              *
 *
                   *     * *
                   *     * *
                 * *
 * * * *                            * *
 * * * *                            * *
                 * *
                   *     * *
                   *     * *
 *
                   * *              *
                   *
 *       *     *     * *     * *
```

```
--------------------------
Generation 3
               *     *
                   *     *         * *
                   * *
                   *     *
                         * *
                   *     * *
    * * * * * *
    * * * *
    * * * *
    * * * * * *
                   *     * *
                         * *
                   *     *
                   * *
                   *     *         * *
               *     *
```



This is the listing to give LIFE designs on your QL:

```
10 REMark QLife
20   initialise
30 REPeat evolve
40  BEEP 1000,RND(40 TO 60)
50  BEEP 1000,RND(40 TO 60)
60  BEEP 1000,RND(40 TO 60)
70 AT 1,6:PRINT "Generation ";z:AT 3,0
80 FOR m=2 TO 9
90   PRINT "       ";
100    FOR n=2 TO 13
110     x(m,n)=y(m,n)
120     PRINT CHR$(x(m,n));" ";
130    END FOR n
140   PRINT
150 END FOR m
```

```
160 FOR m=9 TO 2 STEP -1
170  PRINT "       ";
180   FOR n=2 TO 13
190    PRINT CHR$(x(m,n));" ";
200   END FOR n
210  PRINT
220 END FOR m
230 FOR m=2 TO 9
240  FOR n=2 TO 13
250   q=0
260 IF x(m-1,n-1)=42 THEN q=q+1
270 IF x(m-1,n)=42 THEN q=q+1
280 IF x(m-1,n+1)=42 THEN q=q+1
290 IF x(m,n-1)=42 THEN q=q+1
300 IF x(m,n+1)=42 THEN q=q+1
310 IF x(m+1,n-1)=42 THEN q=q+1
320 IF x(m+1,n)=42 THEN q=q+1
330 IF x(m+1,n+1)=42 THEN q=q+1
340   IF x(m,n)=42 AND q<>2 AND q<>3
                    THEN y(m,n)=32

350   IF x(m,n)=32 AND q=3 THEN y(m,n)=42
360  END FOR n
370 END FOR m
380    z=z+1
390 END REPeat evolve
400 DEFine PROCedure initialise
410 MODE 8
420   PAPER 1:INK 6:CLS
430  DIM x(10,14):DIM y(10,14)
440   z=1
450 REMark initial culture
460 FOR p=1 TO 10
470  FOR q=1 TO 14
480 x(p,q)=32
```

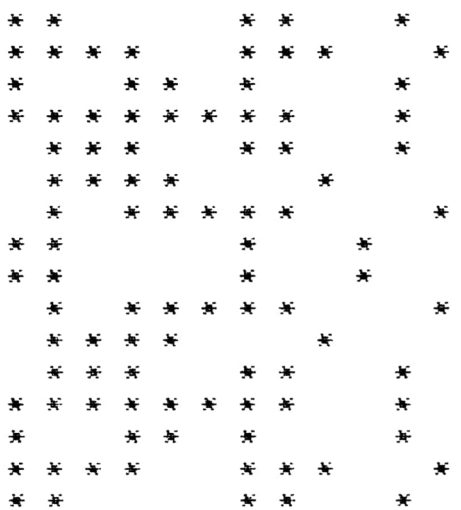
```
490 IF p>1 AND q>1 AND p<10 AND
          q<14 AND RND>.45 THEN x(p,q)=42
500 y(p,q)=x(p,q)
510  END FOR q
520 END FOR p
530 END DEFine initialise
```

You can use this version of the program if you want it to print out to the printer (designated here as channel 3), rather than on to the screen:

```
10 REMark QLife
15 REMark Modified to print out
16 REMark     to channel #3
20   initialise
30 REPeat evolve
40  BEEP 1000,RND(40 TO 60)
50  BEEP 1000,RND(40 TO 60)
60  BEEP 1000,RND(40 TO 60)
65 PRINT #3,\\"
      ------------------------"\
70 AT 1,6:PRINT #3,
      "     Generation ";z:AT 3,0
80 FOR m=2 TO 9
90   PRINT #3,"       ";
100    FOR n=2 TO 13
110     x(m,n)=y(m,n)
120     PRINT #3,CHR$(x(m,n));" ";
130    END FOR n
140   PRINT #3,
150 END FOR m
160 FOR m=9 TO 2 STEP -1
170  PRINT #3,"       ";
```

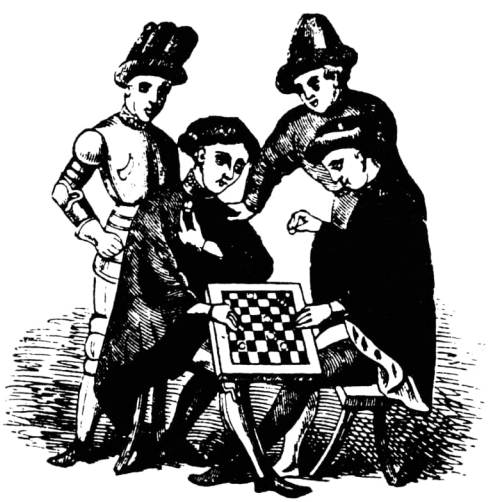
```
180    FOR n=2 TO 13
190     PRINT #3,CHR$(x(m,n));" ";
200    END FOR n
210   PRINT #3,
220 END FOR m
230 FOR m=2 TO 9
240  FOR n=2 TO 13
250    q=0
260 IF x(m-1,n-1)=42 THEN q=q+1
270 IF x(m-1,n)=42 THEN q=q+1
280 IF x(m-1,n+1)=42 THEN q=q+1
290 IF x(m,n-1)=42 THEN q=q+1
300 IF x(m,n+1)=42 THEN q=q+1
310 IF x(m+1,n-1)=42 THEN q=q+1
320 IF x(m+1,n)=42 THEN q=q+1
330 IF x(m+1,n+1)=42 THEN q=q+1
340    IF x(m,n)=42 AND q<>2 AND q<>3
                              THEN y(m,n)=32
350    IF x(m,n)=32 AND q=3 THEN y(m,n)=42
360   END FOR n
370 END FOR m
380     z=z+1
390 END REPeat evolve
400 DEFine PROCedure initialise
410 MODE 8
420    PAPER 1:INK 6:CLS
430   DIM x(10,14):DIM y(10,14)
440    z=1
450 REMark initial culture
460 FOR p=1 TO 10
470   FOR q=1 TO 14
480 x(p,q)=32
490 IF p>1 AND q>1 AND p<10 AND q<14 AND
                        RND>.45 THEN x(p,q)=42
500 y(p,q)=x(p,q)
```

```
510  END FOR q
520 END FOR p
530 END DEFine initialise
```



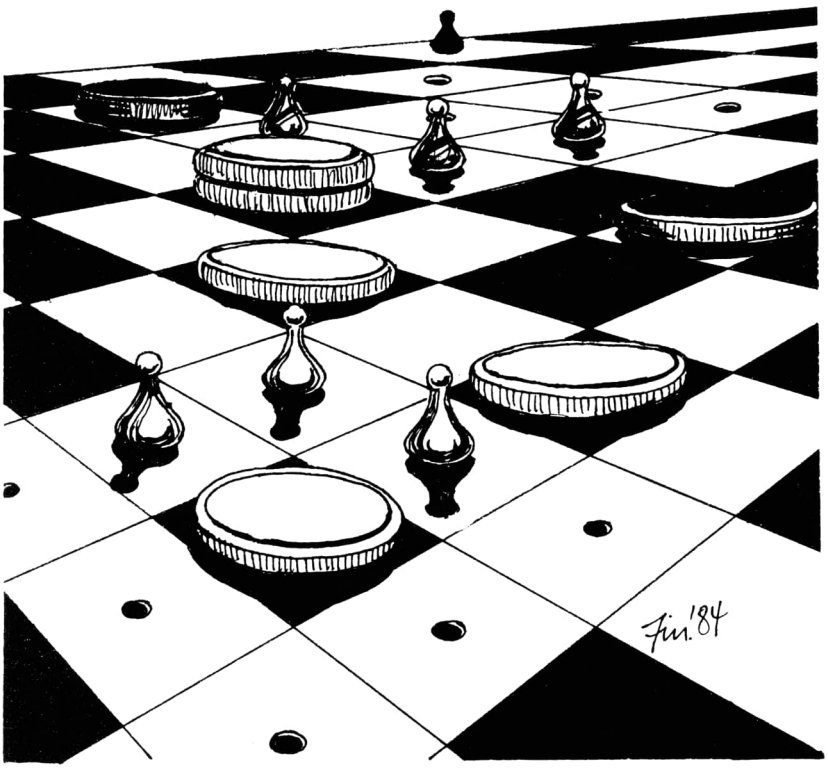
BOARD GAMES

Now you have the chance to pit your feeble little carbon-based brain against the QL's silicon-based one. There are three board games in this section, which show the QL's brain off at its best. Challenge your computer now to Checkers, Gomoku and Qthello.



CHECKERS

The game of Checkers has a long and honourable



history. R. C. Bell (in his book *Discovering Old Board Games,* Shire Publications, Aylesbury, UK, 1980) says

it was invented around 1100, "probably in the south of France, using Backgammon tablemen on a chequered chessboard with the Alquerque method of capture" (pp. 35-36). The *Encyclopedia of Sports, Games and Pastimes* (Fleetway House, London, C.1935) puts it much further back in time: "Forms of it were known in ancient Egypt, Greece and Rome, while the game was played in the mid-seventeenth century much as it is today." (p.237).

Regardless of its age, it is a very popular game around the world, with many European countries having regional variations on the game of their own.

Continental draughts (checkers is generally known as draughts except in America), for example, is played on a board of 100 squares with each player starting the game with 20 pieces. It was developed in the early 1700's.

Our QL checkers program plays the game you are probably most familiar with. It plays swiftly, and reasonably well, although its lack of endgame strategy often leads to a dramatic collapse in the final moment of a game.

When the game begins, you're at the bottom of the screen, playing up, and the computer is at the top playing down. As the program is currently set up, the computer always has the first move. If you want the first move then swap lines 50 and 70.

All moves must be made along diagonals. Individual

pieces can only move forward, that is towards the opponent's starting side. Pieces are converted into 'kings' when they reach the back rank across the board. Your kings are shown as K's, the computer's kings are dollar signs. Kings can move either forward or back. After capturing a piece, by jumping over it into the vacant square immediately behind the captured piece, you can move again if there is a further capture which can be made.

```
10 REMark Checkers
20   initialise
30 REPeat game
40   print_board
50  machine_move
60   print_board
70  accept_move
80 END REPeat game
90 REMark --------------------
100 DEFine PROCedure machine_move
110 FOR x=1 TO 10:s(x)=0:END FOR x
120 sc=0:a=89
130 a=a-1
140 IF q(a)<>c AND q(a)<>ck THEN GO TO 220
150 b=0:IF a<29 THEN b=2
160 b=b+1
170 m=a+n(b)
180 IF m>88 OR m<11 THEN GO TO 220
190 IF (q(m)=h OR q(m)=hk) AND q(m+n(b))=e
                        THEN GO TO 260
200 IF q(m)=e THEN IF (q(m-11)<>h AND
q(m-11)<>hk) THEN IF (q(m-9)<>h AND q(m-9)
<>hk) AND q(m+9)<>hk THEN IF ((q(m+22)<>hk
OR q(m+18)<>hk) AND (q(m+9)<>c OR q(m+9)
```

```
 <>ck OR q(m+11)=c OR q(m+11)=ck)) AND
   q(m+11)<>hk THEN store_move
210 IF b<2 OR (q(a)=ck AND b<4)
                THEN GO TO 160
220 IF a>11 THEN GO TO 130
230 f1=0:IF q(22)=c OR q(24)=c OR q(26)=c
             OR q(28)=c THEN king_make
240 IF f1=1 THEN GO TO 590
250 GO TO 440
260 q(m+n(b))=q(a):q(m)=e:q(a)=e
270 co=co+1
280  BEEP 100,1:BEEP 100,4:BEEP 100,1
290  BEEP 200,10:BEEP 200,1:BEEP 500,4
300 print_board
310 a=m+n(b)
320 b=0
330 b=b+1
340 IF (a+2*n(b)<11 OR a+2*n(b)>88) AND
                  b<4 THEN GO TO 330
350 m=a+n(b)
360 IF q(m)=c AND b>3 THEN END DEFine
370 IF (q(m)=h OR q(m)=hk) AND
                q(m+n(b))=e THEN GO TO 260
380 IF b<2 OR (q(a)=ck AND b<4)
                              THEN GO TO 330
390 END DEFine
400 REMark --------------------
410 DEFine PROCedure store_move
420 IF sc<10 THEN sc=sc+1
430 s(sc)=100*a+b+20:END DEFine
440 IF sc=0 THEN GO TO 490
450 xc=RND(1 TO sc)
460 a=INT(s(xc)/100)
470 m=a+n(s(xc)-100*a-20)
480 GO TO 590
490 sc=sc+1:a=RND(11 TO 88)
```

```
500 IF q(a)<>c AND q(a)<>ck THEN GO TO 570
510 b=0
520 b=b+1
530 m=a+n(b)
540 IF m>88 OR m<11 THEN GO TO 560
550 IF q(m)=e THEN GO TO 590
560 IF b<2 OR q(a)=ck AND b<4 THEN GO TO 520
570 IF sc<300 THEN GO TO 490
580 PRINT:PRINT "I concede the game":STOP
590 q(m)=q(a):q(a)=e
600 END DEFine
610 REMark --------------------
620 DEFine PROCedure print_board
630 AT 1,0
640  INK 6
650 PRINT "          QL: ";co
660 PRINT "       Human: ";hu
670   BEEP 100,30
680 PRINT \"      1 2 3 4 5 6 7 8"
690 FOR f=80 TO 10 STEP -10
700  INK 6:PRINT "  ";f/10;"  ";
710   FOR g=1 TO 8
720    tr=q(f+g)
730    INK 2
740    IF tr=72 OR tr=75 THEN INK 7
750    IF tr=67 OR tr=36 THEN INK 4
760    PRINT CHR$(tr);" ";
770   END FOR g
780  INK 6:PRINT " ";f/10
790 END FOR f
800 PRINT "      1 2 3 4 5 6 7 8"
810   BEEP 100,10
820 IF co=12 OR hu=12 THEN GO TO 840
830 END DEFine
840 IF hu=12 THEN PRINT "You have won":STOP
850   PRINT "I have won"
```

```
860 STOP
870 REMark --------------------
880 DEFine PROCedure accept_move
890 AT 15,0
900 PRINT "Your move. From ";
910 INPUT a$
920 IF LEN(a$)<>2 THEN GO TO 890
930 AT 15,0:PRINT "                              "
940 IF a$=99 THEN GO TO 860
950 IF q(a$)<>h AND q(a$)<>hk THEN GO TO 890
960  BEEP 50,5
970 AT 15,0
980 PRINT "                To ";
990 INPUT b$
1000 IF LEN(b$)<>2 THEN GO TO 980
1010 IF q(b$)<>e THEN GO TO 970
1020 AT 15,0
1030 PRINT "                                 "
1040 q(b$)=q(a$):q(a$)=e
1050 FOR t=11 TO 17
1060  IF q(t)=c THEN q(t)=ck
1070 END FOR t
1080 FOR t=82 TO 88
1090  IF q(t)=h THEN q(t)=hk
1100 END FOR t
1110 IF ABS(a$-b$)<12 THEN GO TO 1270
1120 BEEP 1000,40:BEEP 1000,200
1130 AT 15,0
1140 PRINT "Well done, human"
1150 delay
1160 hu=hu+1:q((a$+b$)/2)=e
1170 print_board
1180 FOR t=82 TO 88
1190  IF q(t)=h THEN q(t)=hk
1200 END FOR t
```

```
1210 AT 15,0
1220 PRINT "Enter 'Y' to jump again"
1230 answer$=INKEY$(-1)
1240 AT 15,0:
  PRINT "                              "
1250 IF answer$()"y" AND answer$()"Y"
                          THEN GO TO 1270
1260 a$=b$:GO TO 950
1270 END DEFine
1280 REMark ------------------
1290 DEFine PROCedure initialise
1300  BEEP 100,30
1310 RESTORE
1320 PAPER 0:INK 7:BORDER 17,1
1330 CLS:CLS #0
1340 AT 5,5:PRINT "Please stand by"
1350 DIM q(150):DIM n(4):DIM s(10)
1360 h=CODE("H"):hk=CODE("K")
1370 c=CODE("C"):ck=CODE("$")
1380 e=32:of=-99
1390 FOR m=1 TO 150
1400  q(m)=of
1410 END FOR m
1420 FOR m=1 TO 64
1430  READ d:READ g
1440   IF g=46 THEN g=0
1450  q(d)=g
1460 END FOR m
1470 FOR m=1 TO 4
1480  READ x
1490  n(m)=x
1500 END FOR m
1510 co=0:hu=0
1520 END DEFine
1530 REMark ------------------
1540 DEFine PROCedure king_make
```

```
 1550 IF q(22)=c AND q(11)=e THEN a=22:m=11:
          fl=1:GO TO 1610
 1560 IF q(22)=c AND q(13)=e THEN a=22:m=13:
          fl=1:GO TO 1610
 1570 IF q(24)=c AND q(13)=e THEN a=24:m=13:
          fl=1:GO TO 1610
 1580 IF q(24)=c AND q(15)=e THEN a=24:m=15:
          fl=1:GO TO 1610
 1590 IF q(26)=c AND q(15)=e THEN a=26:m=15:
          fl=1:GO TO 1610
 1600 IF q(26)=c AND q(17)=e THEN a=26:
          m=17:fl=1
1610 END DEFine
1620 REMark --------------------
1630 DEFine PROCedure delay
1640  FOR wait_around=1 TO 1000
1650  END FOR wait_around
1660 END DEFine
1670 REMark --------------------
1680 DATA 81,46,82,67,83,46,84,67,85,46,86,67
                                      ,87,46
1690 DATA 88,67,71,67,72,46,73,67,74,46,75,67
                                      ,76,46
1700 DATA 77,67,78,46,61,46,62,67,63,46,64,67
1710 DATA 65,46,66,67,67,46,68,67,51,32,52,46
1720 DATA 53,32,54,46,55,32,56,46,57,32,58,46
1730 DATA 41,46,42,32,43,46,44,32,45,46,46,32
1740 DATA 47,46,48,32,31,72,32,46,33,72,34,46
                                      ,35,72
1750 DATA 36,46,37,72,38,46,21,46,22,72,23,46
                                      ,24,72
1760 DATA 25,46,26,72,27,46,28,72,11,72,12,46
                                      ,13,72
1770 DATA 14,46,15,72,16,46,17,72,18,46
1780 DATA -11,-9,11,9
```

GOMOKU

You'll find GOMOKU an easy game to learn, but one



which is almost impossible to win. The QL plays extremely well in this program, which is based on one written by Graham Charlton.

You have to try to get five of your pieces (the H's) in a row in any direction, while the computer is trying to do the same with its pieces (the C's).

Here's the listing for your very own game of GOMOKU:

```
10 REMark Gomoku
20  RESTORE
30 initialise
40 board
50 REMark ----------
60 REPeat game
70  player_move
80   board
90    qlmove
100   board
110 move_two
120 END REPeat game
130 REMark ----------
140 DEFine PROCedure check
150 g$=DATE$
160 AT 15,20:PRINT g$(17 TO 20)
170 e=aa
180 REPeat checkmove
190 e=e+n
200 IF a(e)<>z THEN GO TO 230
210 k=k+1
220 END REPeat checkmove
230 END DEFine check
240 REMark ----------
250 DEFine PROCedure board
260  BEEP 1000,45:PAUSE 5
270  BEEP 10000,30
280 AT 3,0
290 INK 4
```

```
300 PRINT "      1 2 3 4 5 6 7 8"
310 PRINT "      ----------------"
320 FOR aa=1 TO 8
330  PRINT "    ";aa;
340   FOR bb=2 TO 9
350 PRINT " ";
360    m=a(aa*10+bb)
370 INK (1*(m=67)+7*(m=72)+2*(m=46))
380 PAPER (6*(m=67)+2*(m=72)+7*(m=46))
390    PRINT CHR$(m);
400 PAPER 7
410   END FOR bb
420  INK 4:PRINT " ";aa
430 END FOR aa
440 PRINT "      ----------------"
450 PRINT "      1 2 3 4 5 6 7 8"
460 INK 0
470 SDATE 1984,6,30,0,0,0
480 END DEFine board
490 REMark ------------------
500 DEFine PROCedure player_move
510 AT 15,20:PRINT "     "
520 AT 1,0
530 PRINT "Now enter your move ";
540 INPUT g
550 AT 1,0
560 PRINT "                                    "
570 g=g+1
580 IF g<12 OR g>89 OR a(g)<>CODE(".")
                          THEN GO TO 520
590 AT 1,0
600 PRINT "  Please stand by..."
610 z=CODE("H")
620 a(g)=z
630 END DEFine player_move
```

```
640 REMark ---------------
650 DEFine PROCedure qlmove
660 aa=g
670 l=0
680 FOR xx=1 TO 4
690  k=0
700  n=x(xx)
710   check
720  n=-n
730   check
740  IF k>l THEN l=k
750 END FOR xx
760 IF l>3 THEN you_win
770 t=1
780 IF t<>2 THEN z=CODE("C")
790 IF t=2 THEN z=CODE("H")
800 g=0
810 h=0
820 l=0
830 FOR aa=12 TO 89
840  m=0
850  IF a(aa)<>CODE(".") THEN GO TO 1020
860 FOR xx=1 TO 4
870  k=0
880  n=x(xx)
890   check
900  n=-n
910   check
920 IF k>l THEN h=0:l=k
930 IF l<>k THEN GO TO 980
940 IF t=1 AND l<4 THEN GO TO 980
950 IF t=2 AND l<2 THEN GO TO 980
960 IF t=3 AND l<2 THEN GO TO 980
970  m=m+1
980 END FOR xx
990 IF m<=h THEN GO TO 1020
```

```
1000    h=m
1010   g=aa
1020 END FOR aa
1030 IF h<>0 THEN GO TO 1130
1040 t=t+1
1050 IF t<>4 THEN GO TO 780
1060 aa=1
1070 g=RND(12 TO 88)
1080 IF a(g)=CODE(",") THEN GO TO 1130
1090 aa=aa+1
1100 IF aa<400 THEN
1110  GO TO 1070
1120    ELSE you_win:STOP
1130 a(g)=CODE("C")
1140 END DEFine qlmove
1150 REMark --------------
1160 DEFine PROCedure move_two
1170 z=CODE("C")
1180  aa=g
1190 l=0
1200 FOR xx=1 TO 4
1210  k=0
1220    n=x(xx)
1230     check
1240    n=-n
1250     check
1260   IF k>1 THEN l=k
1270 END FOR xx
1280 IF l>3 THEN i_win
1290 END DEFine move_two
1300 REMark -----------------
1310 DEFine PROCedure initialise
1320  intro_frame
1330  PAPER 7
1340 CSIZE 3,0
```

```
1350 BORDER 15,1
1360  CLS:CLS #0
1370  k$=CHR$(254)
1380  DIM a(100):DIM x(4)
1390 FOR aa=1 TO 8
1400  FOR bb=2 TO 9
1410   a(aa*10+bb)=CODE(",")
1420  END FOR bb
1430 END FOR aa
1440 FOR q=1 TO 4
1450  READ x(q)
1460 END FOR q
1470 DATA 1,9,10,11
1480 FOR j=1 TO RND(1 TO 12)
1490  READ zz
1500 END FOR j
1510 a(zz)=CODE("C")
1520 DATA 34,35,36,44,45,46,47,54,55,56,57,66
1530 END DEFine initialise
1540 REMark --------------
1550 DEFine PROCedure you_win
1560 REPeat victory
1570  INK RND(2 TO 5)
1580 AT 1,0
1590  PRINT "    You win...              "
1600 IF INKEY$<>"" THEN STOP
1610 END REPeat victory
1620 END DEFine you_win
1630 REMark --------------
1640 DEFine PROCedure i_win
1650  REPeat victory
1660   INK RND(2 TO 5)
1670  AT 1,0
1680 PRINT "   I win...                  "
1690 IF INKEY$<>"" THEN STOP
1700 END REPeat victory
```

```
1710 END DEFine i_win
1720 DEFine PROCedure intro_frame
1730 BORDER 0
1740 PAPER 0
1750 CLS:CLS #0
1760 CSIZE 3,1
1770 FOR design=1 TO 60
1780 INK RND(0 TO 7)
1790 FILL 1
1800 e1=RND(1 TO 200)
1810 e2=RND(1 TO 100)
1820 e3=RND(10 TO 46)
1830  CIRCLE e1,e2,e3
1840 INK RND(1 TO 7)
1850 AT 3,3
1860 PRINT "g o m o k u"
1870 AT 6,12
1880 PRINT "g o m o k u"
1890 BEEP 1000,design
1900 END FOR design
1910 END DEFine intro_frame
```

QTHELLO

Qthello, as I'm sure you've realised, is the QL's version



of Reversi, or Othello. Invented in the late eighteen hundreds, the game is played on an ordinary eight by

eight board. When played on a board, you use pieces which have different colours on each side. The game begins with four pieces placed on the centre squares.

From this point on, you move by placing one of your pieces next to a computer piece or pieces, with another of your pieces further on. When that happens, all the computer pieces 'reverse' to become your pieces.

Here's how it works. Suppose a line of pieces looked like this:

OXXXX

. . . and you decided to put your piece (the O) at the end of the line like this:

OXXXXO

The QL pieces would reverse, so the line looked like this after your move:

OOOOOO

The game continues until every square on the board is filled, or neither player can move. As you can see, fortunes can change swiftly in this game, as rows branching off your position (such as on the diagonals) can be changed with a single move.

If you cannot move at any time, your signal this to the computer by entering a **double** zero.

Othello is a very enjoyable game to play. Writing in *Creative Computing* magazine (June 1981, p.188) David Levy tenders the opinion that it is "one of the best games ever invented, simply because the rules can be learned in no more than a minute, yet the game can take years to master". He goes on to observe that although the game is much simpler than chess, it is more complex than checkers.

Although the modern game is called Othello, and was 'invented' by Goro Hasegawa in Tokyo in 1971, the only difference between it and Reversi is the fact that the first four positions must be as in the sample game you'll see following this description.

Back in the 1880's, when the game was first invented in London, two gentlemen — Lewis Waterman and John W. Mollett — both claimed to be the originators of the game. Stephen Kimmel (writing in *Creative Computing* magazine, July 1981, p.94) says he believes Waterman has the strongest case for being the inventor of Reversi, because he had the details published first in a series of articles in the magazine, *The Queen* (which concentrated on "affairs of interest to ladies").

Regardless of who should get the credit, Hasegawa, Waterman or Mollett, it is a great game, and this program (based on one written by Graham Charlton) puts up a spirited defence. This sample run shows various stages of one game:

```
   1 2 3 4 5 6 7 8
1  . . . . . . . .  1
2  . . . . . . . .  2
3  . . . . . . . .  3
4  . . . o x . . .  4
5  . . . x o . . .  5
6  . . . . . . . .  6
7  . . . . . . . .  7
8  . . . . . . . .  8
   1 2 3 4 5 6 7 8

   The QL has 2
   The human has 2
```

```
   1 2 3 4 5 6 7 8
1  . . . . . . . .  1
2  . . . . . . . .  2
3  . . . . . . . .  3
4  . . . o x . . .  4
5  . . o o o . . .  5
6  . . . . . . . .  6
7  . . . . . . . .  7
8  . . . . . . . .  8
   1 2 3 4 5 6 7 8

   The QL has 4
   The human has 1
```

```
> Enter your move 63

   1 2 3 4 5 6 7 8
 1 . . . . . . . . 1
 2 . . . . . . . . 2
 3 . . . . . . . . 3
 4 . . . o x . . . 4
 5 . . o x o . . . 5
 6 . . x . . . . . 6
 7 . . . . . . . . 7
 8 . . . . . . . . 8
   1 2 3 4 5 6 7 8

    The QL has 3
   The human has 3

----------------------

   1 2 3 4 5 6 7 8
 1 . . . . . . . . 1
 2 . . . . . . . . 2
 3 . . . . o . . . 3
 4 . . . o o . . . 4
 5 . . o x o . . . 5
 6 . . x . . . . . 6
 7 . . . . . . . . 7
 8 . . . . . . . . 8
   1 2 3 4 5 6 7 8

    The QL has 5
   The human has 2
```

```
> Enter your move 47

   1 2 3 4 5 6 7 8
1  . . . . o . . . 1
2  . . . o . . . . 2
3  . x x x x x . . 3
4  . . x x x x x . 4
5  . . x x o x . . 5
6  o o x o . . . . 6
7  . . o . . . . . 7
8  . . . o . . . . 8
   1 2 3 4 5 6 7 8

    The QL has 8
   The human has 14

--------------------

   1 2 3 4 5 6 7 8
1  . . . . o . . . 1
2  . . o o . . . . 2
3  . x o x x x . . 3
4  . . o x x x x . 4
5  . . o x o x . . 5
6  o o o o . . . . 6
7  . . o . . . . . 7
8  . . . o . . . . 8
   1 2 3 4 5 6 7 8

    The QL has 13
   The human has 10
```

```
> Enter your move 52

   1 2 3 4 5 6 7 8
1  .  .  .  .  o  .  .  .  1
2  .  .  o  o  .  .  .  .  2
3  .  x  o  x  x  x  .  .  3
4  .  .  x  x  x  x  x  .  4
5  .  x  x  x  o  x  .  .  5
6  o  o  o  o  .  .  .  .  6
7  .  .  o  .  .  .  .  .  7
8  .  .  .  o  .  .  .  .  8
   1 2 3 4 5 6 7 8

    The QL has 11
   The human has 13

--------------------

   1 2 3 4 5 6 7 8
1  .  .  .  .  o  .  .  .  1
2  .  .  o  o  .  .  .  .  2
3  .  x  o  x  x  x  .  .  3
4  .  .  x  x  x  x  x  .  4
5  o  o  o  o  o  x  .  .  5
6  o  o  o  o  .  .  .  .  6
7  .  .  o  .  .  .  .  .  7
8  .  .  .  o  .  .  .  .  8
   1 2 3 4 5 6 7 8

    The QL has 15
   The human has 10
```

```
> Enter your move 57

   1 2 3 4 5 6 7 8
1  o o o o o . . .  1
2  . . x o . . . .  2
3  x x o x x x . .  3
4  x o o o o x o o  4
5  x o o x x x x .  5
6  x x o o o o . .  6
7  x . o o . . . .  7
8  . . . o . . . .  8
   1 2 3 4 5 6 7 8

    The QL has 22
   The human has 16

----------------------

   1 2 3 4 5 6 7 8
1  o o o o o . . .  1
2  . . x o . . . .  2
3  x x o x x x . .  3
4  x o o o o x o o  4
5  x o o o o o o o  5
6  x x o o o o . .  6
7  x . o o . . . .  7
8  . . . o . . . .  8
   1 2 3 4 5 6 7 8

    The QL has 27
   The human has 12
```

```
> Enter your move 67

   1 2 3 4 5 6 7 8
1  o o o o o . . .  1
2  . . x o . . . .  2
3  x x o x x x . .  3
4  x o o o x x o o  4
5  x o o o o x o o  5
6  x x x x x x x .  6
7  x . o o . . . .  7
8  . . . o . . . .  8
   1 2 3 4 5 6 7 8

    The QL has 21
   The human has 19

--------------------

   1 2 3 4 5 6 7 8
1  o o o o o . . .  1
2  . . o o . . . .  2
3  x x o o x x . .  3
4  x o o o o x o o  4
5  x o o o o o o o  5
6  x x x x x x o .  6
7  x . o o . . . o  7
8  . . . o . . . .  8
   1 2 3 4 5 6 7 8

    The QL has 27
   The human has 14
```

```
> Enter your move 87

   1 2 3 4 5 6 7 8
1  o o o o o o . .  1
2  o o o o o . . o  2
3  o o o o x x . o  3
4  o x o o o x o o  4
5  o x x o o x o o  5
6  o x o x x o o o  6
7  o x x o o x o o  7
8  o x x x x x x .  8
   1 2 3 4 5 6 7 8

    The QL has 39
   The human has 19

--------------------

   1 2 3 4 5 6 7 8
1  o o o o o o . .  1
2  o o o o o . . o  2
3  o o o o x x . o  3
4  o x o o o x o o  4
5  o x x o o x o o  5
6  o x o x x o o o  6
7  o x x o o x o o  7
8  o o o o o o o o  8
   1 2 3 4 5 6 7 8

    The QL has 46
   The human has 13
```

```
> Enter your move 37

   1 2 3 4 5 6 7 8
1  o o o o o o . . 1
2  o o o o o o . o 2
3  o o o o o o x o 3
4  o x o o o x o o 4
5  o x x o x o o o 5
6  o x o x x o o o 6
7  o x x o o x o o 7
8  o o o o o o o o 8
   1 2 3 4 5 6 7 8

    The QL has 49
   The human has 12

----------------------

   1 2 3 4 5 6 7 8
1  o o o o o o . . 1
2  o o o o o o o o 2
3  o o o o o o o o 3
4  o x o o o x o o 4
5  o x x o x o o o 5
6  o x o x x o o o 6
7  o x x o o x o o 7
8  o o o o o o o o 8
   1 2 3 4 5 6 7 8

    The QL has 51
   The human has 11
```

```
> Enter your move 18

   1 2 3 4 5 6 7 8
1  o o o o o o . x  1
2  o o o o o o x o  2
3  o o o o o x o o  3
4  o x o o x x o o  4
5  o x x x x o o o  5
6  o x x x x o o o  6
7  o x x o o x o o  7
8  o o o o o o o o  8
   1 2 3 4 5 6 7 8

    The QL has 46
   The human has 17

----------------------

   1 2 3 4 5 6 7 8
1  o o o o o o o x  1
2  o o o o o o o o  2
3  o o o o o x o o  3
4  o x o o x x o o  4
5  o x x x x o o o  5
6  o x x x x o o o  6
7  o x x o o x o o  7
8  o o o o o o o o  8
   1 2 3 4 5 6 7 8

    The QL has 48
   The human has 16


    I'm the champ!
```

And here's the listing for Qthello:

```
10 REMark Qthello
20   CLEAR:RESTORE
30  initialise
40 REPeat major_cycle
50  qthello
60 END REPeat major_cycle
70 REMark -------------
80 DEFine PROCedure qthello
90 AT 2,2:PRINT "I am thinking..."
100 s=ox:t=x:h=0
110 FOR az=2 TO 9
120  FOR b=2 TO 9
130 AT 9,23:PRINT "* *":AT 9,23:PRINT "   "
140 IF a(az,b)<>46 THEN GO TO 310
150   q=0
160   FOR c=-1 TO 1
170    FOR d=-1 TO 1
180    k=0:f=az:g=b
190 IF a(f+c,g+d)<>s THEN GO TO 220
200    k=k+1:f=f+c:g=g+d
210      GO TO 190
220 IF a(f+c,g+d)<>t THEN GO TO 240
230      q=q+k
240   END FOR d
250  END FOR c
260 IF az=2 OR az=9 OR b=2 OR b=9 THEN q=q*2
270 IF az=3 OR az=8 OR b=3 OR b=8 THEN q=q/2
280 IF (az=2 OR az=9) AND (b=3 OR b=8) OR
  (az=3 OR az=8) AND (b=2 OR b=9) THEN q=q/2
290 IF q<h OR (RND<.3 AND q=h) THEN GO TO 310
300   h=q:m=az:n=b
310  END FOR b
320 END FOR az
330 IF h=0 THEN IF r=0 THEN end_game
```

```
340 IF h=0 THEN GO TO 360
350  rework
360  print_board
370 AT 2,2:INPUT "Enter your move ";mr$
380 IF LEN(mr$)<>2 THEN GO TO 370
390 r=mr$
400 REMark 00 to pass
410 clear_line
420 IF r=0 THEN GO TO 500
430 IF r<11 OR r>88 THEN GO TO 370
440 BEEP 1000,200
450  r=r+11
460  m=INT(r/10)
470  n=r-10*m
480  s=x:t=ox
490  rework
500  print_board
510 END DEFine qthello
520 REMark -------------
530 DEFine PROCedure print_board
540  clear_line
550  c=0:h=0
560  PRINT:PRINT
570  INK 2
580 PRINT "    1 2 3 4 5 6 7 8"
590  FOR b=2 TO 9
600  INK 2:PRINT "  ";b-1;" ";
610   FOR d=2 TO 9
620    INK 3:tj=a(b,d)
630    IF tj=x THEN INK 1:c=c+1:BEEP 1000,1
640    IF tj=ox THEN INK 0:h=h+1:BEEP 1000,154
650    PRINT CHR$(tj);" ";
660   END FOR d
670   INK 2:PRINT b-1
680  END FOR b
```

```
690 PRINT "    1 2 3 4 5 6 7 8"
700 PRINT:PRINT
710 INK 4:PRINT "      The QL has ";c;"    "
720 INK 1:PRINT "    The human has ";h;"
730 IF c+h=64 THEN end_game:STOP
740 END DEFine print_board
750 REMark -------------
760 DEFine PROCedure rework
770  FOR c=-1 TO 1
780   FOR d=-1 TO 1
790    f=m:g=n
800 IF a(f+c,g+d)<>s THEN GO TO 830
810  f=f+c:g=g+d
820  GO TO 800
830 IF a(f+c,g+d)<>t THEN GO TO 870
840  a(f,g)=t
850  IF m=f AND n=g THEN GO TO 870
860  f=f-c:g=g-d:GO TO 840
870  END FOR d
880 END FOR c
890 END DEFine rework
900 REMark -------------
910 DEFine PROCedure end_game
920  print_board
930  FLASH 1
940 IF c>h THEN PRINT " I'm the champ!"
950 IF h>c THEN PRINT " You're the champ!"
960 IF h=c THEN PRINT " It's a draw"
970  FLASH 0
980 END DEFine end_game
990 REMark -------------
1000 DEFine PROCedure initialise
1010  PAPER 7:INK 0:CLS:CLS #0
1020  BORDER 4,2,5
1030  DIM a(10,10)
```

```
1040  x=CODE("o"):ox=CODE("x")
1050  FOR b=2 TO 9
1060   FOR c=2 TO 9
1070    a(b,c)=CODE(",")
1080   END FOR c
1090  END FOR b
1100  a(5,5)=x:a(6,6)=x
1110  a(6,5)=ox:a(5,6)=ox
1120  p=0:h=0:r=0
1130  CSIZE 1,0
1140  print_board
1150  BORDER 4,0,1
1160  END DEFine initialise
1170 REMark -------------
1180 DEFine PROCedure clear_line
1190  AT 2,2
1200  PRINT "                    "
1210  BEEP 100,90
1220 END DEFine clear_line
```

```
  1 2 3 4 5 6 7 8
1 o o o o o o . . 1
2 o o o o o . . o 2
3 o o o o x x . o 3
4 o x o o o x o o 4
5 o x x o o x o o 5
6 o x o x x o o o 6
7 o x x o o x o o 7
8 o o o o o o o o 8
  1 2 3 4 5 6 7 8

 The QL has 46
The human has 13
```

CREATING MOVING GRAPHICS

There is little doubt that moving graphics games, such as you can see in arcades, are among the most exciting things you can do with your computer. Unfortunately, to get proper, high-speed graphics, you need to use machine code, which is beyond the scope of this book. However, as you'll soon see, the QL can easily produce very satisfactory moving graphics programs in SuperBASIC. This section of the book will not only give you several moving graphics games, but will explain how they were written, so you can create similar games of your own.

ANIMATION

It is simple to make something appear to move on the screen. You put a shape, such as the letter A, on the screen in a particular spot, and hold it there for a moment or two. Then, you print a blank space where the A was and at the same time, reprint the A a short distance away from its original position.

This process is repeated, over and over again, and it

creates the impression that the object is moving. Believe it or not, that is all you have to do. Put one moving object under your control, and another one under the QL's control, and you have the raw ingredients of a game.

Let's see how it works in practice.

AT MAGIC

The AT command works by counting the very first position on the screen (the one in the top left hand corner) as 0,0. Follow an AT command with one to PRINT, and you will see the QL begin its printing at the position specified by the two numbers which follow the word AT.

We can show this in action very simply, by placing something on the screen. Enter this brief program and run it:

```
10 REMark AT DEMONSTRATION
20 CLS:CLS #0
30 INK 3:PAPER 7
40 AT 0,0
50 INPUT "FIRST CO-ORDINATE ";A
60 IF A<0 OR A>19 THEN GO TO 50
70 INPUT "SECOND CO-ORDINATE ";B
80 IF B<0 OR B>36 THEN GO TO 70
90 CLS
100 AT A,B
110 PRINT "*"
120 GO TO 40
```

When prompted, enter two numbers of your choice, pressing the ENTER key after each one. The first number you enter is the number of lines down the screen you want an asterisk to be printed. The second number is the number of positions you want the object to be across the screen. The first number must be between 0 and 19, and the second number between 0 and 36.

BOUNCING BALL

Next, we will use AT to create a 'bouncing ball' program. You'll see that the ideas used in it will be used several times in the programs which are in this section, and — of course — can also be used in your own programs:

```
10 REMark BOUNCING BALL
20 PAPER 0:INK 6
30 CLS:CLS #0
40 A=6:B=11
50 X=1:Y=1
60 EA=A:EB=B
70 AT A,B
80 PRINT "O"
90 B=B+Y
100 A=A+X
110 IF A<2 THEN X=-X
120 IF A>18 THEN X=-X
130 IF B<2 THEN Y=-Y
140 IF B>35 THEN Y=-Y
150 GO TO 60
```

You'll find the techniques used for animating this 'ball' (heavily disguised as the letter 'O') will be used in just about every moving graphics game you write in BASIC with AT on the QL, so it is worth taking the trouble to understand exactly how it works.

When you run this, you'll see the ball appear, and bounce across the screen, but without erasing the 'old' ball behind it. Eventually, the screen will be filled with ball images. Add the following lines, and you'll see the old ball erased, to be replaced with a full stop.

```
145 AT EA,EB
147 PRINT "."
```

Running this program, will see the ball leaving a trail of dots behind itself as it moves. From this, you can see the program actually prints the ball (with lines 70 and 80) and then prints over this position (with lines 145 and 147) with the full stop.

Replace line 147 with the following one, which erases the old ball with a space, and you'll see a true moving image on your screen:

```
147 PRINT " "
```

There are two very important variables in this program, EA and EB. These stand for 'erase A' and 'erase B'. The computer needs to know the **old** position of the ball, so that once a new position has been determined, a space to erase the old ball can be printed at the original location. EA and EB are used for this. Line 60 sets EA

equal to A and EB equal to B **before** A and B are modified in lines 90 and 100.

GOING TO BAT

Although we have a moving ball, there is no way we can pretend that this is more than the rawest ingredient of a game. Adding a little bat, or slide, at the bottom of the screen to interact with the ball will take our program one step closer to becoming a game.

Modify your program, so it reads as follows:

```
10 REMark BOUNCING BALL
15 REMark     WITH BAT
20 PAPER 0:INK 6
30 CLS:CLS #0
40 A=6:B=11:F=22
50 X=1:Y=1
60 EA=A:EB=B
61 AT 19,F
62 PRINT " --- "
63 A$=INKEY$
65 IF A$=CHR$(192) AND F>2 THEN F=F-1
66 IF A$=CHR$(200) AND F<30 THEN F=F+1
70 AT A,B
80 PRINT "O"
90 B=B+Y
100 A=A+X
110 IF A<2 THEN X=-X
120 IF A>18 THEN X=-X
130 IF B<2 THEN Y=-Y
```

```
140 IF B>35 THEN Y=-Y
145 AT EA,EB
147 PRINT " "
150 IF A>17 THEN
160   IF ABS(F-B)>3 THEN
170  AT A,B:PRINT "O":STOP
180  END IF
190 END IF
200 GO TO 60
```

The little object at the bottom of the screen is your bat. To keep the ball in play, you must bounce it off the bat at the bottom of the screen. You use the arrow keys to move the bat right and left.

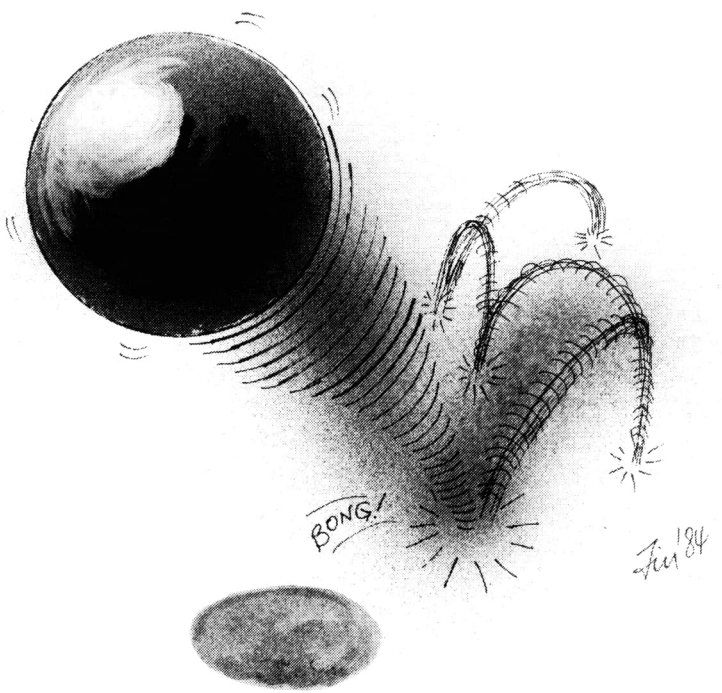
This is not a particularly satisfying game (it probably doesn't even deserve the title 'game') but it shows an extra ingredient which you'll use in moving graphics games — outside interaction. You move the bat, and the computer responds to your action, keeping the ball in play if the bat is in the right position.

This is not a very inspiring game, but it can be modified to make it somewhat more interesting to play. You can dress it up further, and use a few more ingredients of SuperBASIC to produce a much more satisfactory program:

```
10 REMark BOUNCING BALL
15 REMark     WITH BAT
20 PAPER 0:INK 6
25 BORDER 3,3
29  score=.5
30 CLS:CLS #0
```

```
35 bat$=CHR$(254)&CHR$(253)&CHR$(254)
40 A=6:B=11:F=22
50 X=1:Y=1
55 REPeat  game
60 EA=A:EB=B
61  AT 18,F-1
62 PRINT "  ";bat$;"  "
63 A$=INKEY$
64 IF A$="" THEN GO TO 70
65 IF A$=CHR$(192) AND F>3 THEN F=F-2
66 IF A$=CHR$(200) AND F<29 THEN F=F+2
70 AT A,B
75 INK 5
80 PRINT "O"
85 INK 2
90 B=B+Y
100 A=A+X
110 IF A<3 THEN X=-X:BOP
120 IF A>16 THEN X=-X:BOP
130 IF B<3 THEN Y=-Y:BOP
140 IF B>33 THEN Y=-Y:BOP
145 AT EA,EB
147 PRINT " "
150 IF A>16 THEN
160   IF ABS(F-B)>3 THEN
165  INK 5
170  AT A,B:PRINT "O":STOP
180  END IF
190 END IF
191 IF A=4 THEN
192  score=score+.5
193  AT 0,12
194  STRIP 1:INK 6
195   PRINT "Score is ";INT(score)
196  STRIP 0:INK 2
```

```
198 END IF
200 END REPeat game
210 DEFine PROCedure BOP
220 BEEP 70,1
230 END DEFine BOP
```



You'll soon discover there is much more challenge to the game. You might like to spend some time playing with this program, modifying it as you wish, before returning to the book to discuss moving an object up and down, as well as left and right, on a screen.

ORBIT

This next program will allow you to simulate — using AT — the rotation of a planet around a sun. This is not a game, by any stretch of the imagination, but it demonstrates how effectively AT can be used, and, incidentally, demonstrates that the QL can run pretty quickly if relatively few demands are placed on it. Note that a delay (line 210) has to be included to allow you to see what is going on.

```
10 REMark Orbit
20 PAPER 0:INK 6
30 BORDER 1,2
40 CLS:CLS #0
50 AT 10,18
60 PRINT "O"
70 INK 5
80 ce=0:de=0
90 REPeat orbit
100 y=RND(7 TO 11)
110 FOR yy=1 TO 2
120 FOR a=1 TO 100 STEP 3
130  AT ce,de
140  PRINT " "
150   b=PI*a/50
160  c=6.5*COS(b)+y
170  d=10*SIN(b)+18
180  ce=c:de=d
190  AT c,d
200 PRINT "o"
210 PAUSE .5
220 END FOR a
```

```
230 END FOR yy
240 END REPeat orbit
```

REX

We now have a program which is difficult (and very enjoyable) to play, in which the mean-hearted computer, playing REX, stalks the humble human.

You control the human and, of course, the QL is in charge of REX. You use the cursor keys to move yourself around the screen, trying to stay out of the computer's clutches for as long as possible.

Here's the listing, so you can do a little dinosaur hunting:

```
10 REMark REX
20 hiscore=0
30 BORDER 0:CLS:CLS #0
40 REPeat game
50 got=0
60 begin
70 place_human
80 place_rex
90 REPeat cycle
100 update_score
110 get_move
120 place_human
130 move_rex
140 place_rex
```

```
150 check_if_got
160 IF got=1 THEN EXIT cycle
170 END REPeat cycle
180 END REPeat game
190 DEFine PROCedure begin
200  FOR j=40 TO 1 STEP -4
210   FOR K=2,4,2,4,0
220    BORDER j,K
230   BEEP 10,j:END FOR K
240  END FOR j
250  BORDER 20,0
260  PAPER 0:CLS:CLS #0
270 BEEP 10000,1:BEEP 10000,9
280 CSIZE 3,0
290  PAPER 6
300 FOR as=1 TO 10
310  FOR bs=1 TO 10
320  AT as*3,bs*5
330  PRINT CHR$(254)
340  END FOR bs
350 END FOR as
360  PAPER 7
370 score=0
380 c=RND(1 TO 10)
390 c1=c
400 d=RND(1 TO 10)
410 d1=d
420 x=RND(1 TO 10)
430 x1=x
440 y=RND(1 TO 10)
450 y1=y
460 END DEFine begin
470 REMark ----------
480 DEFine PROCedure place_human
490 AT x1*3,y1*5
500  INK 2:PAPER 6
```

```
510  PRINT CHR$(254)
520 AT x+3,y+5
530  INK 6:PAPER 2
540  PRINT CHR$(183)
```



```
550  PAPER 7
560 END DEFine place_human
570 REMark ---------------
580 DEFine PROCedure place_rex
590  AT c1+3,d1+5
600  INK 2:PAPER 6
610   PRINT CHR$(254)
620   INK 0:PAPER 4
630   AT c+3,d+5
640   PRINT CHR$(182)
650   PAPER 7
660 END DEFine place_rex
670 REMark -----------------
680 DEFine PROCedure get_move
690  x1=x
700  y1=y
710  flee=CODE(INKEY$)
720  IF flee=208 AND x>1 THEN x=x-1
730  IF flee=216 AND x<10 THEN x=x+1
740  IF flee=192 AND y>1 THEN y=y-1
750  IF flee=200 AND y<10 THEN y=y+1
760 END DEFine get_move
770 REMark -----------------
780 DEFine PROCedure check_if_got
790 IF x=c AND y=d THEN gotcha
800 END DEFine check_if_got
810 REMark ---------------
820 DEFine PROCedure move_rex
830  d1=d
840  c1=c
850 IF RND>.8 THEN GO TO 930
860 IF c<x AND c<10 THEN c=c+1
870 IF RND>.8 THEN GO TO 930
880 IF c>x AND c>1 THEN c=c-1
890 IF RND>.8 THEN GO TO 930
```

```
900 IF d<y AND d<10 THEN d=d+1
910 IF RND>.8 THEN GO TO 930
920 IF d>y AND d>1 THEN d=d-1
930 END DEFine move_rex
940 REMark ----------------
950 DEFine PROCedure update_score
960 IF hiscore<score THEN
970   hiscore=score
980 END IF
990 score=score+1
1000 AT 1,0:INK RND(4 TO 7):PAPER 0
1010 PRINT "       Score is ";score
1020 PRINT "   High score is ";hiscore+1
1030 END DEFine update_score
1040 REMark -------------
1050 DEFine PROCedure gotcha
1060  FOR again=1 TO 4
1070   FOR count=1 TO 65 STEP 1.3
1080  BEEP 750-8*count,200/count
1090   END FOR count
1100  END FOR again
1110 FOR again=1 TO 90
1120 AT 15,3
1130  INK RND(2 TO 6)
1140 PRINT "Gotcha, human !!"
1150   BEEP 100-again,100-again
1160  END FOR again
1170 got=1
1180 END DEFine gotcha
1190 REMark -------------
```

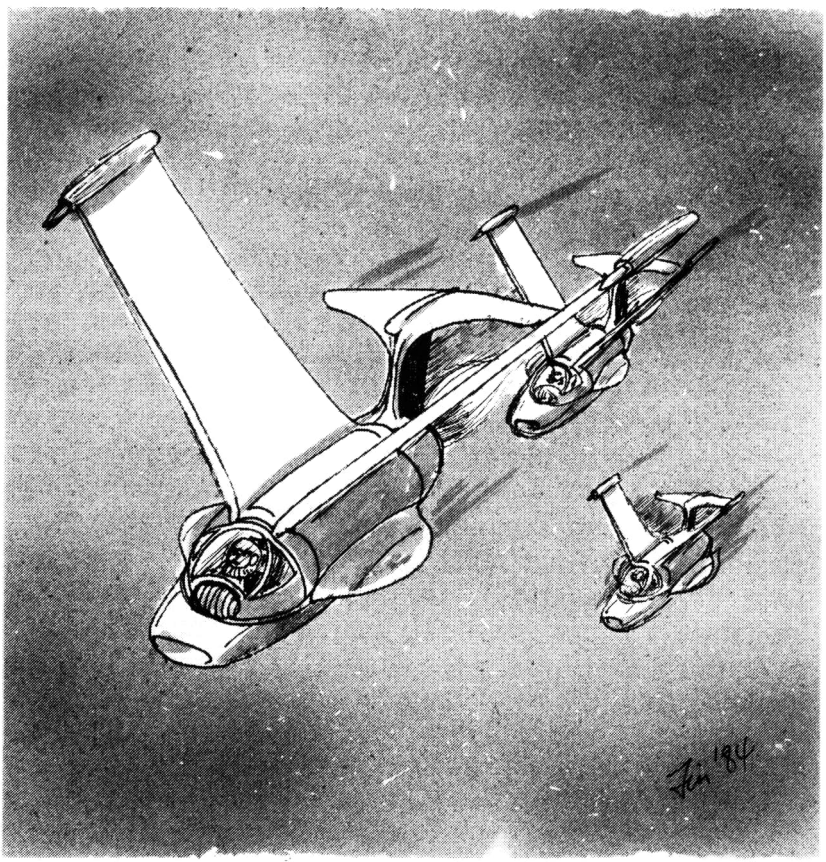
The computer's 'intelligence' in this program comes from the procedure 'move_rex' (defined in lines 820 to 930) which modifies the computer's co-ordinates to move it closer and closer to your co-ordinates. If you deleted the four RND lines (850, 870, 890 and 910),

you would see the QL homing in mercilessly towards you. Without the RND's to give some waywardness to the computer's behaviour, the game would generally be over in seconds.

V-WING SPACE BATTLE

The last game we looked at depended on you trying to avoid capture by the QL. In the next game, you have to steer your tiny 'V-wing space ship' around the screen, trying to run over the numbers which appear at random positions on it. Each number stays in place for a limited time. If you manage to run over the number before it vanishes, a score related to the number you've just run over (so a 9 scores more than a 1) is added to your growing total.

```
10 REMark V-wing Space Battle
20 hiscore=0
30 initialise
40 b$=CHR$(200)
50 new_number
60 REMark ----------
70 REPeat cycle
80 ea=a:eb=b
90 a$=INKEY$
100 IF a$="" THEN
110    a$=b$
120  ELSE
```



```
130     b$=a$
140  END  IF
150  n=CODE(a$)
```

```
160 IF n=208 AND a>2 THEN
170  a=a-1
180  bop=40
190 ch=up
200 END IF
210 IF n=216 AND a<17 THEN
220  a=a+1
230  bop=8
240 ch=down
250 END IF
260 IF n=192 AND b>2 THEN
270  b=b-1
280  bop=120
290  ch=left
300 END IF
310 IF n=200 AND b<33 THEN
320  b=b+1
330  bop=16
340  ch=right
350 END IF
360 AT ea,eb
370 PRINT " "
380 BEEP 1,bop
390 IF v(count)=a AND w(count)=b THEN
400  FOR j=1 TO 150:BEEP 151-j,j
410  sc=sc+value*137
420  new_number
430 END IF
440 AT a,b
450 PRINT CHR$(ch)
460 IF RND>.979 THEN new_number
470 END REPeat cycle
480 REMark ------------------
490 DEFine PROCedure new_number
500 CLS:BORDER 2,RND(1 TO 7)
```

```
510 PAPER 1:INK 7
520 a=6:b=6
530 STRIP 1
540 BEEP 1000,1:BEEP 1000,90
550 AT 0,5:PRINT "Score: ";sc
560 PAPER 2
570 AT 0,22:PRINT 29-count;" to go"
580 AT 18,5:PRINT "High score: ";hiscore
590 STRIP 0
600 INK RND(0 TO 4)
610 count=count+1
620 IF count=30 THEN
630  FOR j=1 TO 255:BEEP 1000,j
640   IF sc>hiscore THEN hiscore=sc
650  count=0
660  value=0:sc=0
670   PAPER 0
680  GO TO 40
690 END IF
700 value=RND(1 TO 9)
710 STRIP 1:PAPER 7
720 AT v(count),w(count)
730 PRINT value
740 STRIP 0:INK 6:PAPER 0
750 END DEFine new number
760 REMark ------------------
770 DEFine PROCedure initialise
780 DIM v(30):DIM w(30)
790 BORDER 2,1:PAPER 0
800 CLS:CLS #0
810 value=0:sc=0
820 count=0
830 FOR tt=1 TO 30
840  v(tt)=RND(5 TO 14)
850  w(tt)=RND(5 TO 28)
```

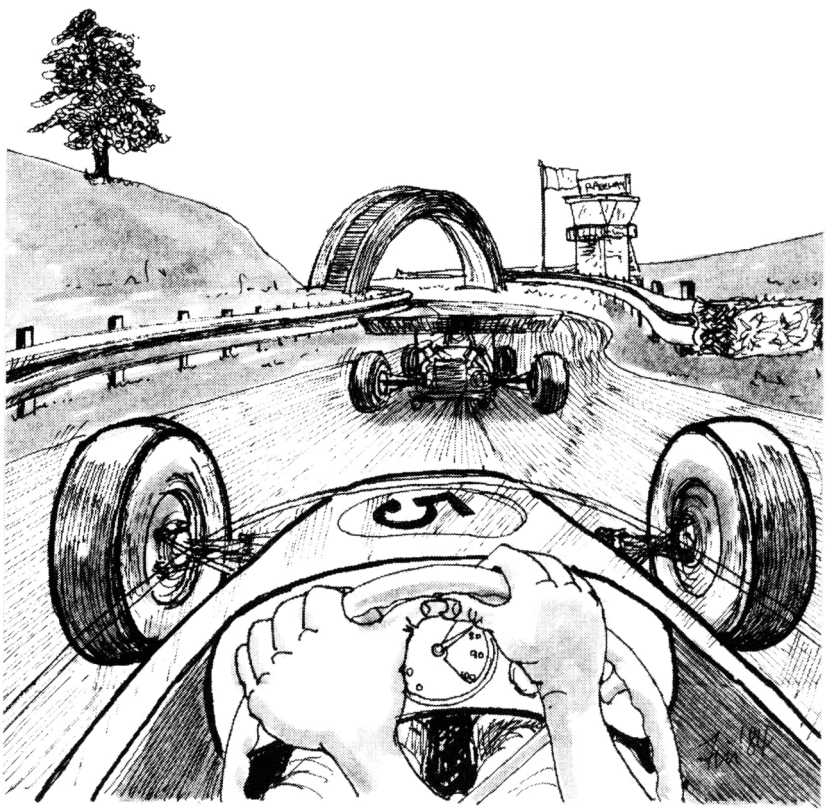
```
860 END FOR tt
870 left=188
880 right=189
890 up=190
900 down=191
910 END DEFine initialise
```

QL RACER

Finally, in this section of the book, we'll look at a program in which you drive a little racing car (which looks remarkably like an arrow) around a race track. You'll discover that the game, although it starts off running fairly slowly, is almost impossible to play. If you manage to get around the track once without crashing, it will speed up, and will continue to increase its speed for twenty games.

You travel from the top left hand corner round the course clockwise, then up the left hand edge to your starting position, where you'll be given a new car. You must avoid all the edges to stay in the race. Your score is related to how long you manage to keep the car in action, and also the 'difficulty level' which is set at the start of a game. You'll see that the QL's internal clock is used to give a readout which shows how long you have survived. The clock, and the score, is updated using the procedure defined in lines 470 to 560.

```
10 REMark QL Racer
20  high_score=0
```



```
30  difficulty=21
40 REPeat cycle
50  initialise
```

```
60    REPeat race
70    increment_score
80     read_keyboard
90     erase_old_car
100     check_if_smash
110      IF smash=1 THEN EXIT race
120     place_new_car
130     BEEP 1000,RND(240 TO 250)
140    FOR delay=1 TO difficulty
150    END FOR delay
160    END REPeat race
170 REMark ---------------
180 REMark Smash sequence
190 FOR j=1 TO 70
200 AT car_down,car_across
210 INK RND(1 TO 7)
220 PRINT "*"
230 BEEP 10,RND(1 TO 20)
240 AT car_down,car_across
250 PRINT "+"
260 BEEP 10,RND(40 TO 70)
270 AT car_down,car_across
280 PRINT CHR$(254)
290 END FOR j
300 AT 0,0
310 STRIP 1:INK 7:PAPER 1
320 FLASH 1
330 PRINT "Your score is ";score
340 IF score>high_score THEN
350  high_score=score
360 END IF
370 AT 18,16
380 PAPER 2
390 PRINT "High score is ";high_score
400 FLASH 0
410 FOR y=1 TO 100
```

```
420    BEEP 50,y
430    BEEP 70,200-2*y
440 END FOR y
450 END REPeat cycle
460 REMark ------------------
470 DEFine PROCedure increment_score
480  score=score+1
490 AT 1,32
500 PRINT score
510 get$=DATE$
520 INK 7:PAPER 2
530 AT 9,5
540 PRINT get$(16 TO 20)
550 PAPER 0
560 END DEFine increment_score
570 REMark ------------------
580 DEFine PROCedure place_new_car
590 INK 6
600 AT car_down,car_across
610    PRINT car$
620 END DEFine place_new_car
630 REMark ----------------
640 DEFine PROCedure check_if_smash
650 smash=c(car_down,car_across)
660 END DEFine check_if_smash
670 REMark --------------
680 DEFine PROCedure erase_old_car
690 AT erase_down,erase_across
700 PRINT " "
710 END DEFine erase_old_car
720 REMark --------------
730 DEFine PROCedure read_keyboard
740 erase_across=car_across
750 erase_down=car_down
760 new$=INKEY$
770 IF new$="" THEN new$=old$
```

```
780 IF new$=CHR$(192) THEN
790  car_across=car_across-1
800  car$=CHR$(188)
810 END IF
820 IF new$=CHR$(200) THEN
830  car_across=car_across+1
840  car$=CHR$(189)
850 END IF
860 IF new$=CHR$(208) THEN
870  car_down=car_down-1
880  car$=CHR$(190)
890 END IF
900 IF new$=CHR$(216) THEN
910  car_down=car_down+1
920  car$=CHR$(191)
930 END IF
940 old$=new$
950 END DEFine read_keyboard
960 REMark ---------------
970 DEFine PROCedure initialise
980 PAPER 0:INK 3:BORDER 4,2
990 CLS:CLS #0
1000  score=0
1010 IF difficulty>1 THEN difficulty=
                              difficulty-1
1020 PAPER 7:INK 2
1030 AT 0,30:PRINT "Score:"
1040 AT 4,32:PRINT " ";difficulty;" "
1050 DIM a(233),b(233),c(20,30)
1060 REMark Build racetrack
1070 PAPER 6:INK 4
1080 CSIZE 2,0
1090 RESTORE
1100 FOR j=1 TO 233
1110  READ a(j),b(j)
1120  c(a(j),b(j))=1
1130 AT a(j),b(j)
```

```
1140 PRINT CHR$(254)
1150 END FOR j
1160 PAPER 1
1170 FOR j=1 TO 29
1180 READ x,y
1190 AT x,y
1200 INK RND(2 TO 7)
1210 PRINT CHR$(253):BEEP 100,5*j
1220 END FOR j
1230 REMark Place car
1240 car_across=3:car_down=3
1250 erase_across=car_across
1260 erase_down=car_down
1270 old$=CHR$(200)
1280 smash=0
1290 PAPER 0:INK 6
1300 FOR y=1 TO 50:BEEP 100,y
1310 SDATE 1984,7,3,0,0,0
1320 END DEFine initialise
1330 REMark --------------
1340 REMark Track data
1350 DATA 1,4,1,5,1,6,1,7,1,8,1,9,1,10,1,11
1360 DATA 1,20,1,21,1,22,1,23,1,24,1,25,1,26
1370 DATA 1,27,1,28,1,29
1380 DATA 2,2,2,3,2,4,2,11,2,12
1390 DATA 2,18,2,19,2,20,2,29,2,30
1400 DATA 3,1,3,2,3,12,3,16,3,17,3,18,3,30
1410 DATA 4,1,4,6,4,7,4,8,4,12,4,13
1420 DATA 4,16,4,23,4,24,4,25,4,26,4,30
1430 DATA 5,1,5,4,5,5,5,6,5,7,5,8,5,9
1440 DATA 5,13,5,14,5,15,5,16,5,21,5,22
1450 DATA 5,23,5,26,5,27,5,30
1460 DATA 6,1,6,4,6,9,6,19,6,20,6,21
1470 DATA 6,22,6,23,6,24,6,25,6,26,6,30
1480 DATA 7,1,7,4,7,9,7,10,7,19,7,20
```

```
1490 DATA 7,21,7,22,7,30
1500 DATA 8,1,8,4,8,10,8,11,8,12,8,13,8,14
1510 DATA 8,15,8,16,8,17,8,18,8,19
1520 DATA 8,20,8,28,8,29,8,30
1530 DATA 9,1,9,4,9,10,9,11,9,12,9,13
1540 DATA 9,14,9,15,9,16,9,17,9,18
1550 DATA 9,24,9,25,9,26,9,27,9,28,9,29
1560 DATA 10,1,10,4,10,11,10,18,10,19
1570 DATA 10,20,10,23,10,24
1580 DATA 11,1,11,4,11,5,11,6,11,7,11,8
1590 DATA 11,11,11,19,11,20,11,24,11,25
1600 DATA 11,26,11,27,11,28,11,29
1610 DATA 12,1,12,7,12,8,12,9,12,10,12,11
1620 DATA 12,14,12,15,12,19,12,20,12,21
1630 DATA 12,28,12,29
1640 DATA 13,1,13,10,13,11,13,14,13,15,13,16
1650 DATA 13,19,13,20,13,21,13,22
1660 DATA 13,29,13,30
1670 DATA 14,1,14,2,14,6,14,10,14,11
1680 DATA 14,15,14,16,14,19,14,20,14,22
1690 DATA 14,23,14,24,14,25,14,26,14,30
1700 DATA 15,2,15,4,15,5,15,6,15,7,15,10
1710 DATA 15,11,15,12,15,15,15,16
1720 DATA 15,20,15,21,15,22,15,23,15,24
1730 DATA 15,25,15,29,15,30
1740 DATA 16,2,16,3,16,4,16,7
1750 DATA 16,15,16,16,16,17,16,28,16,29
1760 DATA 17,7,17,8,17,14,17,15
1770 DATA 17,17,17,18,17,28
1780 DATA 18,8,18,9,18,10,18,11,18,12,18,13
1790 DATA 18,14,18,18,18,19,18,20
1800 DATA 18,21,18,22,18,23,18,24
1810 DATA 18,25,18,26,18,27,18,28
1820 DATA 14,21,7,7,9,5,6,5,9,8
1830 DATA 10,9,7,5,11,10,6,6
1840 DATA 9,6,5,25,6,8,9,7
```

```
1850 DATA 10,7,6,7,5,24,7,6
1860 DATA 7,8,11,9,8,5,10,10
1870 DATA 8,6,10,8,8,7,10,6
1880 DATA 8,8,8,9,10,5,9,9
```

DEDUCTION AND PERCEPTION

While it is, perhaps, hard to pin down the 'redeeming qualities' of some computer games, others present no such problems. Such games are not simple. They make real demands on your ability to concentrate and to weigh known facts and from them reach conclusions.

Detective Q, for example, presents you with a body lying in a mansion, and six suspects. Any of them could have committed the murder, and you have to find out which one it was. The murder took place at either 8, 9, 10 or 11 pm, and as well as identifying the murderer, you have to be able to say when the foul deed took place. Twenty-four possibilities, and only one of them correct, and you only get two stabs at the right answer. You narrow down the possibilities by questioning the suspects and then trying to find inconsistencies in their stories which will lead to the correct identification.

In Memomuse, you challenge the QL to a round of the game 'Concentration' and in Lexicon, you have to deduce the four-letter word the QL has thought of.

MEMOMUSE

This is a program in which the computer shows what a



wonderful memory it has, as it challenges you to a round of the card game 'Concentration'.

A deck of cards is shuffled, and then laid out on the screen. You and the computer take it in turns to choose two cards. These are turned over, and if they are a pair, the player who selected them is awarded a point. This process continues until all the cards have been selected. The winner, naturally enough, is the player who has selected the most pairs.

```
10 REMark Memomuse
20 initialise
30 REPeat main_loop
40   mov=3:FLASH 0
50   aw=-99:b=-99
60     print_out
70  computer_guess
80   mov=3:FLASH 0
90   aw=-99:b=-99
100     print_out
110  blank_lines
120  human_guess
130  IF hs+cs=26 OR endflag=1
                    THEN EXIT main_loop

140 END REPeat main_loop
150 AT 1,0
160 PRINT "That brings us to the end"
170 PRINT "of a round of Memomuse!"\
180 IF hs>cs THEN
190  PRINT "You're the winner!"
200   ELSE IF cs>hs THEN
210    PRINT "I'm the winner!"
220   ELSE
230    PRINT "It's a draw, human!"
240 END IF
```

```
250 STOP
260 REMark ----------------
270 DEFine PROCedure computer_guess
280 mov=0
290 IF r=1 OR RND<level THEN GO TO 380
300 x=r
310 x=x-1
320 y=1
330 y=y+1
340 IF d(x)=d(y) THEN GO TO 360
350 IF a$(d(x))(1)=a$(d(y))(1) THEN IF
x<>y THEN IF a$(d(x))(3)="X" AND a$(d(y))
(3)="X" THEN aw=d(x):b=d(y):GO TO 480
360 IF y<x THEN GO TO 330
370 IF x-1>r THEN GO TO 310
380 REMark Random guess
390 count=0
400 count=count+1
410 IF count=400 THEN endflag=1
420 IF endflag=1 THEN GO TO 680
430 REPeat card_choose
440  aw=RND(1 TO 52)
450  b=RND(1 TO 52)
460  IF aw<>b AND a$(aw)(3)="X"AND
     a$(b)(3)="X" THEN EXIT card_choose
470 END REPeat card_choose
480 AT 1,0:PRINT "I've selected
              cards ";aw;" and ";b;"     "
490  delay
500  blank_lines
510 a$(aw)(3)=" ":a$(b)(3)=" "
520 print_out
530 blank_lines
```

```
540 AT 1,0
550 IF a$(aw)(1)=a$(b)(1) THEN
560  PRINT "I've picked a pair"
570  a$(aw)="    ":a$(b)="    "
580  cs=cs+1
590 ELSE
600  PRINT "But they're not a pair"
610  a$(aw)(3)="X":a$(b)(3)="X"
620  d$(r)=a$(aw)
630  d$(r+1)=a$(b)
640  d(r)=aw
650  d(r+1)=b
660  r=r+2
670 END IF
680    delay
690 END DEFine computer_guess
700 REMark ---------------
710 DEFine PROCedure human_guess
720 mov=1
730 blank_lines
740 AT 1,0
750  againflag=0
760 INPUT "Enter first card choice ";aw
770  IF aw<1 OR aw>52 THEN GO TO 740
780 IF a$(aw)(1)=" " THEN
790  AT 1,0
800   PRINT "Previously
            picked...              ":BEEP
10000,11:BEEP 10000,120:BEEP 10000,80
810  againflag=1
820 END IF
830 IF againflag=1 THEN delay:GO TO 740
840  blank_lines
850 a$(aw)(3)=" "
860 AT 1,0
```

```
870  againflag=0
880 INPUT "Enter second choice ";b
890 IF b=aw OR b<1 OR b>52 THEN GO TO 860
900 IF a$(b)(3)=" " THEN
910  AT 1,0
920   PRINT "Previously
            picked...            ":BEEP
    10000,11:BEEP 10000,120:BEEP 10000,80
930  againflag=1
940  blank_lines
950 END IF
960 IF againflag=1 THEN delay:GO TO 860
970 a$(aw)(3)=" ":a$(b)(3)=" "
980 print_out
990 blank_lines
1000 AT 1,0
1010 IF a$(aw)(1)=a$(b)(1) THEN
1020  PRINT "Yes, they are a pair"
1030  a$(aw)="    ":a$(b)="    "
1040  hs=hs+1
1050 ELSE
1060  PRINT "No, they're not a pair"
1070  a$(aw)(3)="X":a$(b)(3)="X"
1080  d$(r)=a$(aw)
1090  d$(r+1)=a$(b)
1100  d(r)=aw
1110  d(r+1)=b
1120  r=r+2
1130 END IF
1140  delay
1150  blank_lines
1160 END DEFine human_guess
1170 REMark ---------------
1180 DEFine PROCedure print_out
```

```
1190 AT 1,0
1200 PRINT "My score: ";cs;"  Your score: ";hs
1210 PRINT " We are playing on level ";10*level
1220 PRINT \"  ";
1230 FOR z=1 TO 52
1240  FLASH 0
1250  IF z=aw OR z=b THEN FLASH 1:INK 2:STRIP 7
1260  IF z=5 THEN PRINT " ";
1270  IF z=6 THEN PRINT "  ";
1280  IF z=51 THEN PRINT "            ";
1290 PRINT z;"  ";
1300 b$=a$(z)(3)
1310 IF b$="X" THEN PRINT "-- ";
1320 IF b$=" " THEN PRINT a$(z)(1 TO 2);" ";
1330 IF 5*(INT(z/5))=z THEN PRINT
1340 FLASH 0:INK 6:STRIP 0
1350 END FOR z
1360 FLASH 0:INK 6:STRIP 0
1370 PRINT \\
1380 IF mov<3 THEN INK 1:STRIP 7:PRINT
"   Cards selected were ";a$(aw);"and ";a
    $(b);"  ":STRIP 0:INK 6
1390  delay
1400 IF mov=3 THEN
PRINT "                                    "

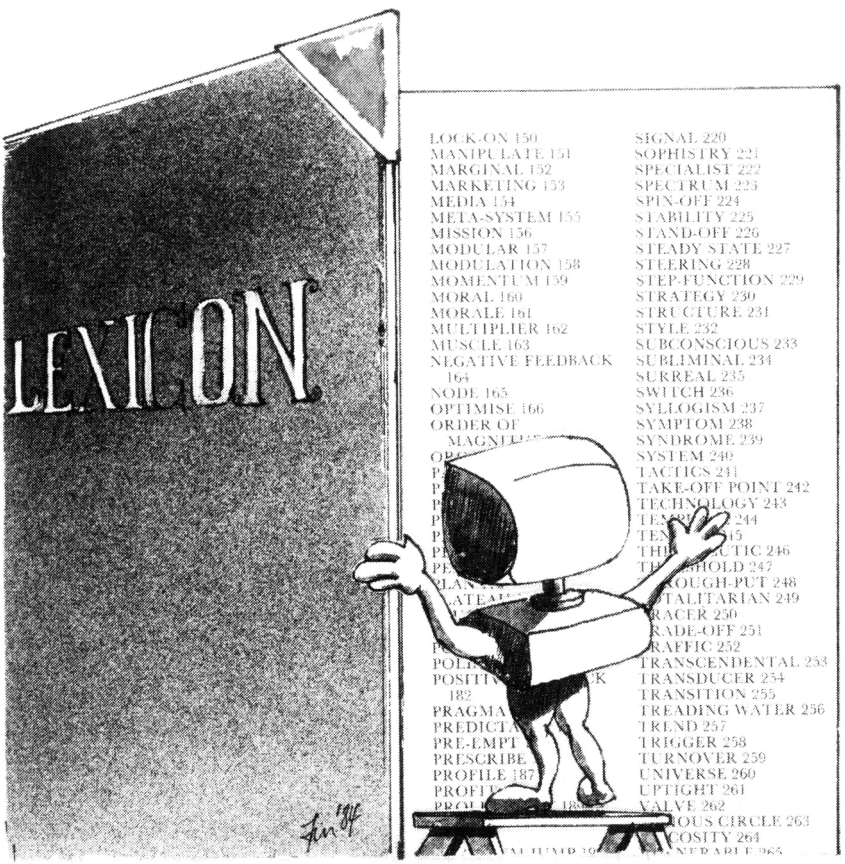
1410 END DEFine print_out
1420 REMark ------------------
1430 DEFine PROCedure blank_lines
1440  AT 1,0
1450  PRINT "
    "
1460 REMark Two lines of blanks
1470 END DEFine blank_lines
1480 REMark ------------------
```

```
1490 DEFine PROCedure initialise
1500  RESTORE
1510 DIM a$(52,3)
1520 DIM d$(500,3)
1530 DIM d(500)
1540 BORDER 0:PAPER 0:INK 6:CLS:CLS #0
1550 FOR t=1 TO 10
1560  BEEP 9900,t,5*t,6,-4,15,0
1570 NEXT t
1580 FOR try=90 TO 2 STEP -2
1590  BORDER try,RND(0 TO 7)
1600 END FOR try
1610 FOR try=2 TO 90 STEP 2
1620  BORDER try,RND(0 TO 7)
1630  BEEP RND(10 TO 700),RND(1 TO 25)
1640 END FOR try
1650 BORDER 7,2,2:CLS
1660 CSIZE 0,1
1670 CLS:CLS #0
1680 AT 3,0
1690 PRINT "  Stand by, card sharp!"
1700 PRINT \"Start flexing your memory muscles!"
1710 cs=0:hs=0:r=1:mov=3
1720 endflag=0
1730 FOR b=1 TO 52
1740  d$(b)=",  "
1750  READ a$(b)
1760 END FOR b
1770 REMark Moses/Oakford shuffle routine
1780 FOR j=52 TO 2 STEP -1
1790  z=RND(1 TO j)
1800  h$=a$(z)
1810  a$(z)=a$(j)
1820  a$(j)=h$
1830 END FOR j
```

```
1840 CLS
1850  blank_lines:blank_lines
1860 PRINT  "   Enter level of difficulty"
1870 PRINT \"    Easy - 10   hard - 1 ";
1880 INPUT level
1890 IF level<1 OR level>10 THEN GO TO 1850
1900 CLS
1910 level=level/10
1920 CSIZE 0,0
1930 END DEFine initialise
1940 REMark ----------------
1950 DEFine PROCedure delay
1960  BEEP 100,1
1970  FOR waiting=1 TO 1000
1980  END FOR waiting
1990 BEEP 3000,RND(10 TO 30)
2000 END DEFine delay
2010 REMark ----------------
2020 DATA "AHX","2HX","3HX","4HX"
2030 DATA "5HX","6HX","7HX","8HX"
2040 DATA "9HX","THX","JHX","QHX","KHX"
2050 DATA "ADX","2DX","3DX","4DX"
2060 DATA "5DX","6DX","7DX","8DX"
2070 DATA "9DX","TDX","JDX","QDX","KDX"
2080 DATA "ASX","2SX","3SX","4SX"
2090 DATA "5SX","6SX","7SX","8SX"
2100 DATA "9SX","TSX","JSX","QSX","KSX"
2110 DATA "ACX","2CX","3CX","4CX"
2120 DATA "5CX","6CX","7CX","8CX"
2130 DATA "9CX","TCX","JCX","QCX","KCX"
```

LEXICON

The next game is Lexicon, where you have to work out



which four-letter word (a clean one, of course) the QL is thinking of. This game is, as I'm sure you've realised,

a word version of the Invicta game 'Mastermind'. After each guess, you'll get some feedback in terms of 'blacks' and 'whites', where a black is your reward for getting a correct letter in the correct position, and a white is awarded if you have a letter right, but it is not in the right position within the code.

```
10 REMark Lexicon
20 PAPER 7:BORDER 5,2,1
30 CLS:CLS #0
40  DIM e(4):DIM f(4)
50 RESTORE
60 REMark ------------
70 find_a_word
80  color
90 PRINT \ "I am thinking of a four-letter word"
100 PRINT "   which you must try and guess."
110  color:FLASH 1
120 IF INKEY$()"" THEN GO TO 130
130 PRINT \\\"    Press any key to begin..."
140 IF INKEY$="" THEN GO TO 140
150  FLASH 0
160 CLS
170 REMark * The game itself *
180 FOR TRY=1 TO 10
190  color
200 AT 2,2
210 PRINT \" Enter guess number ";
220  color:FLASH 1
230  PRINT TRY:FLASH 0
240  AT 4,7:INPUT b$
250  IF CODE(b$)>CODE("Z") THEN
260   color
270   FLASH 1
280   PRINT "Engage CAPS LOCK":FLASH 0
290   GO TO 240
```

```
300 END IF
310  AT 4,7:
 PRINT "                                  "
320  AT 5,0:
 PRINT "                                      "
330 IF LEN(b$)<>4 THEN GO TO 190
340 IF b$=c$ THEN win
350 b=0
360 w=0
370 REMark * Get blacks *
380 FOR c=1 TO 4
390  e(c)=CODE(b$(c))
400  f(c)=CODE(a$(c))
410  IF e(c)=f(c) THEN
420   b=b+1
430   e(c)=0
440  END IF
450 END FOR c
460 REMark * Get whites *
470 FOR c=1 TO 4
480  IF e(c)=0 THEN GO TO 530
490  FOR d=1 TO 4
500   IF e(c)<>f(d) THEN GO TO 520
510   w=w+1
520  END FOR d
530 END FOR c
540 a$=c$
550 REMark * Print result *
560  color
570 AT TRY+5,2
580 PRINT b$;
590  color
600 PRINT " - ";b;" black";
610 IF b<>1 THEN
```

```
620  PRINT "s";
630   ELSE
640    PRINT " ";
650 END IF
660 PRINT " and ";w;" white";
670  IF w<>1 THEN
680   PRINT "s"
690    ELSE
700     PRINT
710  END IF
720 NEXT TRY
730 REMark * If didn't guess *
740  color
750 PRINT \"Sorry, time is up"
760 word
770 nextgo
780 STOP
790 REMark ---------------
800 DEFine PROCedure find_a_word
810 FOR t=1 TO RND(1 TO 40)
820  READ a$
830 END FOR t
840 c$=a$
850 END DEFine find_a_word
860 REMark ---------------
870 DEFine PROCedure win
880  color
890 PRINT \\\\"You got it right in just ";
900 PRINT TRY;" guess";
910 IF TRY>1 THEN
920  PRINT "es"
930   ELSE
940    PRINT
950 END IF
960 word
```

```
970 nextgo
980 END DEFine win
990 REMark --------------
1000 DEFine PROCedure word
1010  color
1020 PRINT "The word was ";
1030  color
1040 PRINT a$
1050 END DEFine word
1060 REMark --------------
1070 DEFine PROCedure nextgo
1080  color
1090 PRINT \\"Do you want to play again?"
1100  color
1110 PRINT "       Enter Y or N ";
1120 z$=INKEY$(-1)
1130 IF z$<>"Y" AND z$<>"N" THEN GO TO 1120
1140 IF z$="N" THEN GO TO 1160
1150  RUN
1160 REMark * Farewell message *
1170 CLS
1180 REPeat message
1190 PRINT FILL$(" ",RND(1 TO 10));
1200  PRINT "OK, thanks for the game"
1210    PAUSE 7
1220    color
1230 END REPeat message
1240 END DEFine nextgo
1250 DEFine PROCedure color
1260  INK RND(1 TO 5)
1270 END DEFine color
1280 REMark ---------------
1290 REMark * Words *
1300 DATA "FACE","FATE","FEAT","FEAR"
1310 DATA "DASH","NEAT","PUSH","PAST"
```

```
1320 DATA "DATE","DIET","DEAD","GRAB"
1330 DATA "SAID","DUET","MIST","DUEL"
1340 DATA "RANT","RAVE","MARS","GETS"
1350 DATA "TINY","TRIP","PERT","REST"
1360 DATA "ROAD","RAID","SOUL","SOLE"
1370 DATA "STAY","THIN","OVER","DOWN"
1380 DATA "PEST","LONG","LAST","LIST"
1390 DATA "MANY","MOST","ONLY","QUIT"
1400 DATA "QUIZ","PACE","BATS","CATS"
```

DETECTIVE Q

You are Detective Q, the man called in to solve the Mystery of the Murder at the Mansion.



It happened like this. Six guests are sharing a meal one evening at the Micro Mansion. The meal ends just before 8 p.m., and the guests decide to spend the rest of the evening exploring the mansion, and to meet again just after midnight for a final drink before bed.

However, when midnight arrives and the guests re-assemble, there is one missing. A hasty search of the mansion produces a Body in a Location.

You are called in. You have to discover The Murderer and The Time At Which It Took Place. Once you've questioned as many of the guests as you like, you can make an accusation. You are only allowed to make two accusations per game, and if you do not succeed in discovering the murderer, and the time at which the murder took place, your QL will tell you who did the dastardly deed.

You refer to the guests by using the first letters of their first names, which due to a happy coincidence, are extremely easy to remember:

A — Arthur Moment
B — Betty Crumb
C — Caroline Forsooth
D — Derek Lockjaw
E — Emma Hornswipe
F — Fatty Thompson
G — General Bluff

Once the game is underway, and you know the name of the victim, you'll be asked if you wish to question a

suspect or make an accusation. You have to indicate not only the person you wish to question (which you do, as was pointed out above, by entering the first letter of their first name) but also the time (8, 9, 10 or 11) that you wish to know about.

You have a map of the mansion, which is very useful as an aid to help determine who was where, at what time. As well, you should make notes of where people have claimed they were at particular times, and who they saw in other rooms, to track down any discrepancies in their stories. A sample chart is given just before the program listing.

The idea, then, is to try and find out who is lying, and therefore pin them down as the murderer. Only the murderer will lie to you, and he or she will only lie about the particular hour in which the murder took place (and the lie will be relatively consistent, at least in some particulars).

However, the other guests — although they will never lie — are not particularly helpful, and will not always tell you the complete truth. That is, they are quite happy to tell you where they were at a particular hour, but are not always so forthcoming as to who was in the room with them (which is, I guess, understandable in some circumstances) nor the identity of those they saw in adjoining rooms.

Eventually you should have enough information to make an accusation. Remember, you only have two

guesses, so you must not waste them. If you fail to identify the murderer and the time at which the deathly deed was done, your miracle of modern technology from Uncle Sir Clive will spill the beans.

When you turn up at the mansion, you are provided with a map of the building, to help you clarify your thoughts:

<table>
<tr><td>LOUNGE</td><td>DINING ROOM</td><td>STUDY</td></tr>
<tr><td>CONSERVATORY</td><td rowspan="2">PITCH BLACK COURTYARD — NO ONE USES IT, MURDER NEVER DONE IN HERE.</td><td>BALLROOM</td></tr>
<tr><td>ENTRANCE HALL</td><td>SERVANTS' QUARTERS</td></tr>
<tr><td>MASTER BEDROOM</td><td>LIBRARY</td><td>KITCHEN</td></tr>
</table>

Ok, Detective Q. Armed with your map and your razor-sharp mind, you must now tackle one of the most difficult cases of your career.

Keep in mind that, although the murderer will lie (as

was pointed out earlier), other guests may well have seen him or her in the room where it happened at the time of the murder, although (of course) the murderer will claim to have been elsewhere at the time. A chart like the following can help you track down the killer in record time:

	8	9	10	11
A	KITCHEN	SERV'TS		B
B			?	
C	BALLR.	LOUNGE?		STUDY
D	STUDY		BEDRO.?	
E	LIBRARY	LOUNGE	BEDRO.	
F	DINING	SERV'TS		STUDY?
G	SERV'TS			BALLR.

Here's the listing to enable you to tackle the Awful, Unspeakable Crime:

```
10 REMark Detective Q
20   initialise
30 CLS:AT 6,1
40 INK 4
50 PRINT g$(mv);"'s";
60 INK 2
70 PRINT " body was"
80 PRINT "found in the ";
90 INK 4
100 PRINT r$(mp)\\
110 INK 3
120 PRINT "Enter '1' to question";
130 PRINT " a suspect,"
140 PRINT "   or '2' to make";
150 PRINT " an accusation"
```

```
160 c$=INKEY$(-1)
170 IF c$<"1" OR c$>"2" THEN GO TO 160
180 c=c$
190 IF c=2 THEN GO TO 660
200 INPUT \"Who would you like to
                question? ";b$
210 IF CODE(b$)>CODE("Z") THEN PRINT "Please
    engage CAPS LOCK":delay:GO TO 30
220 color
230 flag=0
240 color
250 IF b$=g$(mv)(1) THEN PRINT "Don't
              be silly!":delay:GO TO 110
260 PRINT
270 color
280 FOR j=1 TO 7
290 IF b$=g$(j)(1) THEN PRINT
              g$(j);" is ready":flag=j
300 END FOR j
310 IF flag=0 THEN GO TO 110
320 PRINT \\
330 color
350 INPUT \"Which hour do you wish to
            ask              about? ";b
360 IF b<8 OR b>11 THEN GO TO 350
370 b=b-7
380 color
390 PRINT g$(flag);" replies:"
400 color
410 IF flag<>mr OR b<>mh THEN PRINT "   I
              was in the ";r$(a(flag,b))
420 IF flag=mr AND b=mh THEN GO TO 1010
430 color
```

```
440 REMark Sharing room
450 FOR j=1 TO 7
460 IF flag=j THEN GO TO 480
470 IF a(j,b)=a(flag,b) AND RND>.2 THEN
        PRINT g$(j);" was with me"
480 NEXT j
490 color
500 REMark Adjoining rooms
510 IF a(flag,b)=10 THEN n1=9:n2=1:GO TO 550
520 IF a(flag,b)=1 THEN n1=10:n2=9:GO TO 550
530 n1=a(flag,b)-1:n2=a(flag,b)+1
540 color
550 FOR j=1 TO 7
560 IF a(j,b)=n1 AND RND>.4 THEN PRINT
      "I saw ";g$(j);" in the "/r$(n1)
570 IF a(j,b)=n2 AND RND>.4 THEN PRINT
      "I saw ";g$(j);" in the "/r$(n2)
580 NEXT j
590 REMark Chance for accusation
600 color
610 PRINT \\"     > Do you want to make"
620 PRINT "       an accusation (Y or N)?";
630 b$=INKEY$(-1)
640 IF b$<>"N" AND b$<>"Y" THEN GO TO 630
650 IF b$="N" THEN CLS:AT 6,1:GO TO 200
660 guess=guess+1
670 color
680 PRINT \\"This is try number ";
690 color
700 PRINT guess
710 color
720 INPUT \"Who was the murderer? ";m$
730 INPUT \"When did it happen? ";clock
740 PRINT  \\
750 flag=0
```

```
760 color
770 IF m$(1)=g$(mr)(1) THEN PRINT
"Yes, ";g$(mr);" is the killer":flag=flag+1

780 IF clock=mh+7 THEN PRINT \ "Yes, it
      happened at ";clock:flag=flag+4

790 delay
800 IF flag=5 THEN GO TO 880
810 IF flag=1 THEN PRINT "You've got the
    killer, all you need  now is the time"

820 IF flag=4 THEN PRINT "You've got the
  time, but the killer  still eludes you"

830 IF flag=0 THEN PRINT "Wrong on
                          both counts!"

840 delay
850 IF guess<2 THEN GO TO 30
860 color
870 PRINT \"However, you've used up
        all your      guesses":GO TO 910

880 color
890 PRINT \\"Well done!"
900 PRINT "You solved the murder in ";
                          guess;" guesses"
910 color
920 PRINT "The murderer was ";
930 color
940 PRINT g$(mr)
950 color
960 PRINT \\"The killing took place at ";
970 color
980 PRINT mh+7;" pm"
990 delay
1000 STOP
```

```
1010 color
1020 PRINT "I was in the ";r$(x)
1030 t=RND(1 TO 7):IF t=mr OR t=mv
                          THEN GO TO 1030
1040 s=RND(1 TO 7):IF s=mr OR s=mv OR s=t
                          THEN GO TO 1040

1050 IF RND>.7 THEN PRINT g$(s);" was with me"
1060 IF RND>.7 THEN PRINT g$(t);" was with me"
1070 UJ=t
1080 IF x=1 THEN GO TO 590
1090 n1=x-1
1100 t=t+1:IF t=8 THEN t=6
1110 IF s=6 OR mv=6 OR mr=6 THEN GO TO 1160
1120 IF t=mr OR t=mv THEN GO TO 590
1130 IF t=UJ OR mr=t THEN GO TO 590
1140 PRINT "I saw ";g$(t);" in the"/r$(n1)
1150 GO TO 590
1160 IF x>9 THEN GO TO 590
1170 n2=x+1
1180 t=t-1:IF t=0 THEN t=2
1190 color
1200 IF s=2 OR mv=2 OR mr=2 THEN GO TO 590
1210 PRINT "I saw ";g$(t);" in the "/r$(n2)
1220 GO TO 590

1230 DEFine PROCedure initialise
1240   RESTORE
1250  intro_frame
1260 DIM a(7,4):DIM g$(7,20)
1270 DIM r$(10,20)
1280 FOR j=1 TO 7
1290  FOR k=1 TO 4
1300   a(j,k)=RND(1 TO 10)
1310  END FOR k
```

```
1320 END FOR j
1330 mh=RND(1 TO 4):REMark murder hour
1340 mv=RND(1 TO 7):REMark murder victim
1350 mr=RND(1 TO 7):REMark murderer
1360 IF mr=mv THEN GO TO 1350
1370 mp=RND(1 TO 10):REMark murder place
1380 a(mr,mh)=mp:REMark put murderer
                              at murder scene
1390 REMark remove victim from hour of death
1400 FOR k=mh TO 4
1410  a(mv,k)=0
1420 END FOR k
1430 REMark give murderer an alibi
1440 x=RND(1 TO 10)
1450 IF x=mp THEN GO TO 1440
1460 guess=0
1470 REMark name guests
1480 FOR aw=1 TO 7
1490  READ g$(aw)
1500 END FOR aw
1510 REMark name rooms
1520 FOR aw=1 TO 10
1530  READ r$(aw)
1540 END FOR aw
1550 END DEFine initialise
1560 DEFine PROCedure delay
1570 BEEP 10000,230
1580  FOR waiting=1 TO 1000
1590  END FOR waiting
1600  BEEP 1000,1
1610 END DEFine
1620 DEFine PROCedure intro_frame
1630 CLS:CLS #0
1640 CSIZE 3,1
1650  BORDER 6,2,1
```

```
1660 FOR tyt=1 TO 2
1670 PRINT "DETECTIVE Q"
1680  FOR ty=1 TO 197 STEP 3
1690  color
1700 BORDER ty/3,RND(1 TO 7)
1710 AT 1,1: PRINT "DETECTIVE Q"
1720  BEEP 100,ty
1730 END FOR ty
1740 END FOR tyt
1750 CSIZE 0,0
1760 BORDER 2,2:PAPER 0:INK 4:CLS
1770 AT 5,15:PRINT "Time"
1780 AT 7,8:PRINT "to find a murderer"
1790 INK 7:PAPER 1:STRIP 1:FLASH 1
1800 AT 12,12
1810 PRINT "Detective Q":STRIP 0
1820 INK 1:PAPER 7:AT 15,9
1830 PRINT "Please stand by..."
1840 delay
1850 FLASH 0
1860 END DEFine intro_frame
1870 DEFine PROCedure color
1880  INK RND(1 TO 5)
1890  BEEP 1000,RND(1 TO 80)
1900 END DEFine color
1910 DATA "ARTHUR MOMENT","BETTY CRUMB"
1920 DATA "CAROLINE FORSOOTH","DEREK LOCKJAW"
1930 DATA "EMMA HORNSWIPE","FATTY THOMPSON"
1940 DATA "GENERAL BLUFF"
1950 DATA "ENTRANCE HALL","CONSERVATORY"
1960 DATA "LOUNGE","DINING ROOM"
1970 DATA "STUDY","BALLROOM"
1980 DATA "SERVANTS' QUARTERS","KITCHEN"
1990 DATA "LIBRARY","MASTER BEDROOM"
```

WRITE YOUR OWN

You may find, after you've been programming for some time, that although you've got many of the skills you need to write games of your own, you're a little short on ideas. If that's the case, you should find this section of the book of interest to you. I'm going to outline a number of games which seem to me to be ideal for conversion to games for the QL.

Unless you're converting a well-known game, such as Checkers, when your program will be expected to coincide in every respect with the non-computer version, you do not need to ensure that your computer developments of these ideas slavishly follows the outlines here. You'll probably find that, after a certain degree of development, the game takes off on its own, and may well end up in due course bearing little resemblance to its 'parent'. This is all to the good, as you'll probably end up developing games which would not have existed without the QL.

SHUFFLEBOARD. Players slide flat, circular pieces along a board towards a triangular 'target' which contains painted circles with numbers on them. A piece ending up entirely within the numbered circle gives the player the score of that number. There are many possible variations of this, including darts-like target games. Once you get a shuffleboard program

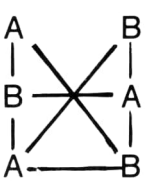
running, you may well find that only cosmetic changes are needed to convert it into a ten-pin bowling game.

HORSESHOE: Starting with a board like this, with one player's pieces at the positions marked A, and one at those marked B, the players take it in turns to move along a line to a vacant spot:



The game ends when a player discovers he or she (or it) cannot move. Despite its simplicity, this is an interesting game, as you'll discover when you play it.

DAMOCLES: This game is like HORSESHOE, but uses a larger board, and three pieces per player. The aim of the game is the same as HORSESHOE, and uses this board:



MU-TORERE: This game, which originated among the Maoris in New Zealand, is played on a board shaped like an eight-pointed star, with each point of the star joined to a circle in the middle. This circle is called the

putahi. The players start with four pieces each, which are placed on the outside points, with all of one player's pieces occupying points next to each other. The aim of the game, as in the last two we've discussed, is to make it impossible for a player to move. You can move from one point to an adjacent empty point, or from the putahi to any vacant point. The third possible move is from a point to the putahi, but this move can only be made if at least one of the points immediately adjacent to the one from which you intend to move is occupied by an opponent's piece.

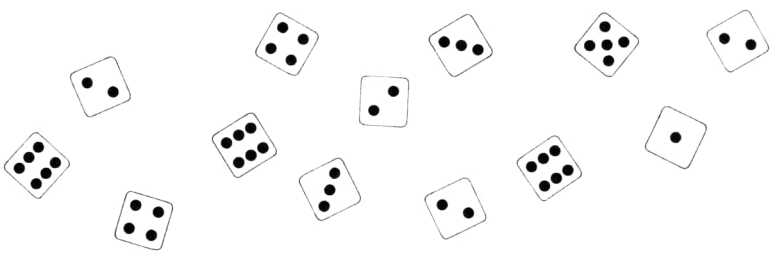
BINGO: This game is usually played with cards marked with numbers, which are crossed off as numbers are drawn at random from a pool, the winner being the player who first corsses off five in a row on their card, in any direction. It can be greatly simplified for computer use, with the computer generating two 'cards' at random, using numbers from zero to 99, and printing them on a five by five grid. The computer can then choose numbers at random from zero to 99, and do the crossings off, and look for possible wins.

The program can be made more interesting to look at by, instead of just generating a random number, including two 'dials' with spinning arrows. The first dial, the 'tens' is marked from zero to nine, and this is spun first. The second dial is marked the same as the first one, except that it represents the 'ones'.

DICE GAMES: You should find dice games relatively easy to program on your computer, especially if the computer is not expected to exhibit too much intelli-

gence when playing. There are a number of dice games which you may wish to try and adapt for computer play, including:

DROP DEAD: You need five dice for this. The dice are thrown by the first player, and any which show two or five are removed. The total of the rest is written down, and the dice which fell with a two or five are thrown again. After this throw, the dice landing two and five are removed, and the total of the remaining dice is added to the total from the first throw. This process



continues until there are no dice left, at which point that particular player's move ends. It is then player two's go, and the same procedure is followed. The winner is the player with the biggest total after both have had their throws. You can make the game the best of, say, three rounds, or limit the total number of throws any player can have per go, regardless of how many dice he or she has left.

MARTINETTI: You need three dice for this, and a score sheet marked with a row of boxes marked from one to 12, and two tokens, one each for you and the computer. You are aiming to traverse the numbers in order, before your opponent manages to do so. Each

player takes it in turn to throw the dice. If the throw contains a one, the player's token can be placed in the one square. If the first throw also contained a two, the token could be moved into the two. After box one, players can either take the pip shown on an individual die for a move, or can add together the pips showing to get a desired number. A variation of this game (often called EVEREST) is to 'climb down' through the numbers, after one through to 12 has been achieved. The winner of EVEREST is the first player getting back to base camp on one.

INDIAN DICE: This game is similar to POKER DICE and is played with five dice. The aim is simply to get the best possible 'poker hands'. There is no provision for getting additional 'cards' in one deal. You throw the five dice at once, and that is your hand. The ranking of the hands is as follows (with six highest and ones wild):

— five of a kind
— four of a kind
— full house (three of a kind plus a pair)
— two pairs
— one pair
— highest die

BIDOU: This game is also similar to POKER DICE, however, you aim to get rid of all your chips, rather than win them. The winner of the game is, paradoxically, the person who has lost the most rounds. You need three dice for this game, and a supply of chips which

are distributed equally between the two players. A bet is made, and matched, before the players take it in turns to throw the three dice. The throws are ranked as follows, with 'winning throws' coming first:

2 1 1; 2 2 1; 4 2 1; 6 6 6; 5 5 5; 4 4 4;

3 3 3; 2 2 2; 1 1 1; 3 3 6; 3 3 5; 3 3 4;

3 3 2; 3 3 1; 1 1 6; 1 1 5; 1 1 4; 1 1 3;

3 2 1; 4 3 2; 5 4 3; 6 5 4

If the combination thrown is not in the list, the pips are added, with the highest total winning.

BARBOOTH: Also known as BARBUDI, this game is played with two dice, and features winning and losing throws. It is suggested you start with a bank of 30 chips, and give both players a starting stake of 15 chips each, charging two chips per player per round.

The players take it in turns to throw the dice. Winning throws are 3 3; 5 5; 6 5; and 6 6. Losing throws are 1 1; 2 2; 2 1 and 4 4. If the players both get winning throws, the one closest to the start of the list wins (that is, 3 3 is better than 6 6). If both throw the same winning throw, the bank gets the money. If one players throws a combination which is not a winning one nor a losing one, and the other throws a losing throw, the loser's chips go into the bank, and the other player retains his or her chips.

If one player throws a winning throw, and the other a non-ranking combination, the winner gets his own chips back, as well as those bet by the loser. If, however, one player throws a winning combination, and the other a losing combination, the bank contributes the same number of chips as have been bet in total. The game continues until either player, or the bank, runs out of chips. The player holding the maximum number of chips at this point wins. Note that this way of distributing the wins and losses is not the same as BARBOOTH when played as a 'professional' gambling game, but is a pay-off scheme which is relatively easy to computerise, and makes for an interesting game.

FIVE THOUSAND: There are no prizes for guessing that the first player to score over 5000 points in this game is the winner. Played with as many players as you like, all you need to emulate on the QL is five six-sided dice. Each player takes it in turn to throw the five dice, and scoring is carried out in the following way:

- a 5 is worth 100
- a 2 is worth 50
- three 5's thrown in one go are worth 1000
- three 2's thrown in a go score 500
- three of a kind (apart from 2's and 5's) are worth 100 times their value (that is three 1's are worth 100 points, and three 6's are worth 600)
- if the sequence 1, 2, 3, 4 and 5 is thrown in a single go, this scores 1500

— 2, 3, 4, 5 and 6 in one throw are worth 2000 points.

Note that 4's and 6's do not score in this game, except as part of a sequence. Once you've thrown, you can — if you like — throw the non-scoring dice again. However, if the next throw gains no score, the turn ends, and the overall score for the throw is zero. So long as a scoring throw has been made, any dice 'left over' can be thrown.

The game rewards the slow and careful player with a reasonable score, but gives him or her no chance against a player who decides to take a chance and try to get a massive score in a single throw.

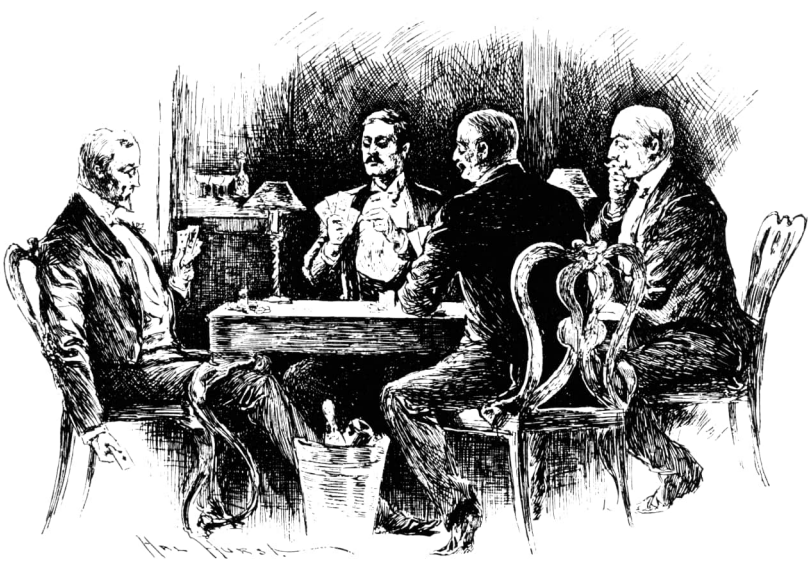
CARD GAMES: Once you've worked out a routine to get the computer hold, shuffle and deal the cards (and such a routine is in the game of MEMOMUSE in this book) you have the raw bones of a host of games. Any book of card games will give you more ideas than you can possibly cope with. Here are a few to start you off:

TRENTE AND QUARANTE: This game, which originated in 17th century Europe, is also known as ROUGE ET NOIR. Six decks of cards are used in the casino version, but it works well with a single deck. The cards are dealt into two rows, with the total of each row being examined after each row has had an additional card.

Court cards (jack, queen, king) count as ten, with all other cards (including the ace) counting as their face

value. The moment the total of a row equals or exceeds 31, no more cards are added to that row. However, the dealer keeps adding to the other row until it, too, equals or exceeds 31. The first row is called 'black', regardless of the suits of the cards involved, and the second row is 'red'.

You can bet that, at the end, the red row or the black row total will be closer to 31. You can also bet that the



colour of the first card dealt will match the colour name of the winning row. All bets are made against the bank, and winners receive their bet back, plus that amount again (odds of one to one). If both rows equal 31, the bank takes half the stake and returns the rest to the player.

CARD CRAPS: This game was devised to get around laws passed by some US States which outlawed craps played with dice. You use a special deck of 48 cards, made up from the aces, 2's, 3's, 4's, 5's and 6's, from two decks of cards. The game follows the standard rules of craps.

In its simplified form, without the somewhat complex betting combinations, you play as follows:

- a total of 2, 3 or 12 is 'craps' and the dealer loses, passes the cards to the next player to deal
- a total of 7 or 11 is a 'natural' and the dealer wins
- any other number (4, 5, 6, 8, 9 or 10) is the player's 'point' and he or she must continue dealing in pairs until the point is dealt again, at which point the dealer wins. However if a 7 is dealt before the point is obtained, the dealer loses.

THREE CARD BRAG: This game was the foreruner of poker, and as its name suggests, it is played with hands of three, rather than five, cards. Here are the winning combinations, ranked from highest to lowest:

- *'pryle'*, three of a kind
- *'on a bike'*, three in sequence from the same suit
- *'run'*, three cards in sequence
- *'flush'*, three cards from the same suit
- *'pair'*, two of a kind
- *'high card'*

A normal deck is used, and aces rank high, except for *'on a bike'* and *'run',* when 3, 2, ace beats ace, king, queen. You can vary the game by making it SEVEN CARD BRAG, in which the players split their seven cards into two three-card hands of their choice, or NINE CARD BRAG, when three hands are created.

ACE-DEUCE-JACK: This is a simple game, heavily loaded in the dealer's favour, when players bet on the likelihood that the next card which appears will be an ace, a two (a 'deuce') or a jack.

FURTHER READING



I make a point of reading any book on games I can get my hands on. It doesn't matter whether the book is about computer games or the old-fashioned 'play against another human' type. So long as it is a book of games, I want to see it.

Games books are superb as idea-starters. Read about a game and you've got the seed of a program. And it doesn't matter if the program doesn't end up playing precisely the game you read about. Many great computer games started their lives as an attempt to computerise a more traditional game, and somehow got side-tracked.

I hope some of the following list of books prove as valuable to you as they have to me.

* **DISCOVERING OLD BOARD GAMES** — R C Bell (Shire Publications, Aylesbury, UK, 1980)

* **DISCOVERING DICE AND DOMINOES** — R C Bell (Shire Publications, Aylesbury, UK, 1980)

* **DICE GAMES NEW AND OLD** — William E Tredd (The Oleander Press, London and New York, 1981)

* **GAME PLAYING WITH BASIC** — Donald D Spencer (Hayden Book Co., Inc., Rochelle Park, New Jersey, 1977)

* **THE TURING CRITERION — MACHINE INTELLIGENT PROGRAMS FOR THE 16K ZX81** — Graham Charlton, Mark Harrison & Dilwyn Jones, ed. Tim Hartnell (Interface Publications, London and Melbourne, 1982)

* **GAMES ANCIENT AND ORIENTAL AND HOW TO PLAY THEM** — Edward Falkener (Dover Publications, Inc., New York, 1961 — a reprint of an original work published by Longmans, Green and Co. in 1892)

* **BASIC COMPUTER GAMES** — Ed. David H Ahl (Creative Computing Press, Morristown, New Jersey, 1978)

* **MORE BASIC COMPUTER GAMES** — Ed. David H Ahl (Creative Computing Press, Morristown, New Jersey, 1979)

* **26 BASIC PROGRAMS FOR YOUR MICRO** — Derrick Daines (Newnes Technical Books, Butterworths & Co., London and Boston, 1982)

* **TABLE GAMES OF GEORGIAN AND VICTORIAN DAYS** — F R B Whitehouse (Priority Press, Royston, Herts., UK, 1971)

* **PET FUN AND GAMES** — Ron Jeffries and Glen Fisher (Osborne/McGraw-Hill, Berkeley, California, 1981)

* **COMPUTER GAMES FOR BUSINESSES, SCHOOLS AND HOMES** — J Victor Nahigian and William S Hodges (Winthrop Publishers, Inc., Cambridge, Massachusetts, 1979)

* **GAMES YOU MAKE AND PLAY** — Pia Hsiao, Neil Lorimer and Nick Williams (Macdonald and Jane's, London, 1975)

* **WHAT TO DO AFTER YOU HIT RETURN** — People's Computer Company (Hayden Book Co., Inc., Rochelle Park, New Jersey, 1980)

* **EVERYMAN'S INDOOR GAMES** — Gyles Brandreth (Dent, Everyman's Library, London, Melbourne and Toronto, 1982)

* **A PLAYER'S GUIDE TO TABLE GAMES** — John Jackson (Stackpole Books, Harrisburg, Pennsylvania, 1975)

* **GAMES FOR TWO** — John Wasley (Proteus, London and New York, 1981)

* **THE ADDISON-WESLEY BOOK OF APPLE COMPUTER SOFTWARE 1982** — Jeffrey Stanton and John Dickey, editors (The Book Company, Lawndale, California, 1982)

* **DICING WITH DRAGONS** — Ian Livingstone (Routledge & Kegan Paul, London, Melbourne and Henley, 1982)

* **FANTASY ROLE PLAYING GAMES** — J Eric Holmes (Hippocrene Books Inc., New York, 1981)

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 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

Artificial Intelligence — the section of computer science which concentrates on eliciting machine behavior which, if it came from a human being, would be called intelligent. One of the aims of AI (as Artificial Intelligence is often written) is to make computers more useful. AI research may also turn out to help us understand our own thinking processes

ASCII — stands for American Standard Code for Information Exchange. This is an almost universal code for letters, numbers and symbols, which has a number between 0 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, 0 and 1 (as opposed to the ordinary decimal system, in which there are ten symbols, 0, 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. Much of a computer's strength in demonstrating 'intelligent' behavior lies in its use of Boolean Algebra.

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

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

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

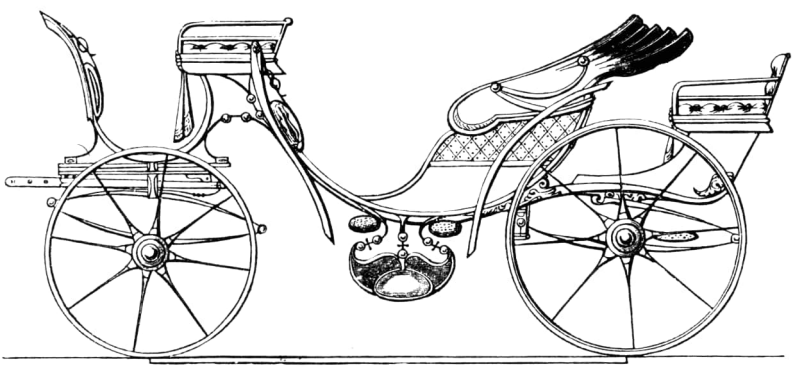
Compiler — a program which translates a program written in a high level (human-like) language into a machine language which the computer is able to understand directly

Concatenate — to add (adding two strings together is known as 'concatenation')

CP/M — these initials stand for Control Program/ Microcomputer, an almost universal disk operating system developed and marketed by Digital Research, Pacific Grove, California

Data — a general term for information processed by a computer

Database — a collection of data, organised to permit rapid access by computer. A relational data base is



one in which the interconnections between various elements within the database are stored explicitly to aid manipulation of, and access to, the elements within the data base

Debug — to remove bugs (errors) from a program

Disk — a magnetic storage medium (further described as a 'hard disk', 'floppy disk' or even 'floppy') used to store computer information and programs. The disks resemble, to a limited extent, 45 rpm sound records,

and are generally eight, five and a quarter, or three and a half inches in diameter. Smaller 'microdisks' are also available for some systems.

Documentation — the written instructions and explanations which accompany a program

DOS — stands for Disk Operating System (and generally pronounced 'doss'), the versatile program which allows a computer to control a disk 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 disks when recorded using a special technique which, as the name suggests, doubles the amount of storage the disk can provide

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

EPROM — 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 opera-

tional error which has caused the computer to halt program execution

Expert system — a computer program which, drawing on the encoded expertise of human experts, performs a specialised task as well as (or, in some cases, better than) the human expert in that field would. They often call on extensive databases of knowledge, and are sometimes called knowledge-based systems

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 disk 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' (i.e. the value it holds) giving information regarding a particular condition

Floppy disk — see disk

Flowchart — this is 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. 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 effects the entire program is called 'global', as opposed to 'local'

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

Hardware — the solid parts of the computer (see 'software' and 'firmware')

Heuristic — a path towards a goal which has been worked out by experience, rather than by calculation; a path of this type is not guaranteed to produce a certain result (in contrast to an algorithm which is a

technique or procedure which, when applied, produces a desired result); chess programs play, in large measure, heuristically

Hexadecimal — a counting system often used 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

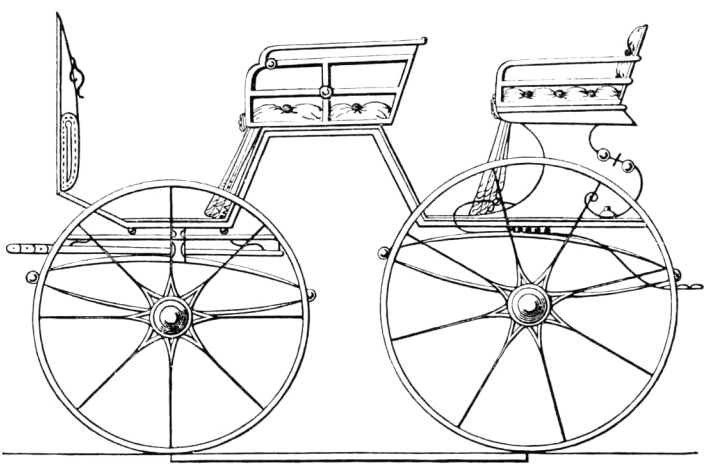
Human interface — the 'outward and visible sign' of an expert system, with which the human user of the system interacts; these often work with natural language

Inference system — the mechanism by which a computer program reaches conclusions; some systems make hard and fast YES/NO decisions, others operate in the world of 'fuzzy logic', where shades of grey (degrees of uncertainty) are permitted

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

I/O — 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, could be ADD which (as you've guessed) tells the computer to carry out an addition

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 position which

the joystick is occupying; generally used in games programs

Kilobyte — the unit of language measurement; one kilobyte (generally abbreviated as K) equals 1024 bits (a bit is the smallest discrete unit)

Knowledge base — the accumulated knowledge upon which an expert system makes its judgements

Knowledge engineering — the process by which human expertise is transferred to an expert system

Knowledge Information Processing Systems — these are the Fifth Generation computer systems which are under development in Japan

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

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 (see RAM and ROM)

Microprocessor — the 'chip' which lies at the heart of

your computer. This does the 'thinking'

Modem — 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 in tandem

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 0, 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

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 — stands for Random Access Memory, and is the memory onboard 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

Representation — the organisation of information within a computer so that it can be managed by the knowledge base control system

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-100 bus — this is also a standard interface (see

RS-232) made up of 100 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 — the end point of a series of events which are accessed on a last in, first out basis

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

Symbolic inference — see inference system

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 the output from the computer on heat-sensitive paper

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 business use) which is ready to run when delivered, needing only the 'turn of a key' to get it working

VLSI — Very Large Scale Integration of components on a 'chip', with present development in Japan and the US aiming for the equivalent of 10 million transistors per chip (current chips contain the equivalent of around half a million transistors at most)

Volatile memory — a memory device which loses its contents when the power supply is cut off

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

From Tim Hartnell, the author Personal Computer World called 'Mr Sinclair', comes this major games book for the QL.

Not only does it contain listings for more than 20 complete games, but the detailed instructions and ideas will help you develop your own games, and improve your programming as you do so. There is even a complete chapter to teach you how to create moving graphics games on the QL.

The programs include:

ELIZA — in which your QL acts as a psychiatrist, and carries on conversations with you

QTHELLO — which allows you to challenge the QL to a round of Reversi/Othello (very hard to beat!)

MAGICIAN'S MAZE — high adventure as you explore the twisted ruins of the magician's house, and attempt to gather up the treasures

DETECTIVE Q — in which you have to solve a murder (and the game is different each time you run it)

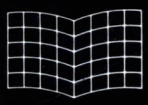
STOCKBROKER — a challenging simulation which puts thousands of pounds under your direct control

TRYST WITH MEPHISTOPHELES — nearly 20K of challenging adventure, as you battle for the release of your soul

QL RACER — real-time moving graphics, in a game which very few will be able to win

Bring your QL along. The party's underway, and the games are waiting.

Another great book from



£5.95

ISBN 0-947695-04-4

