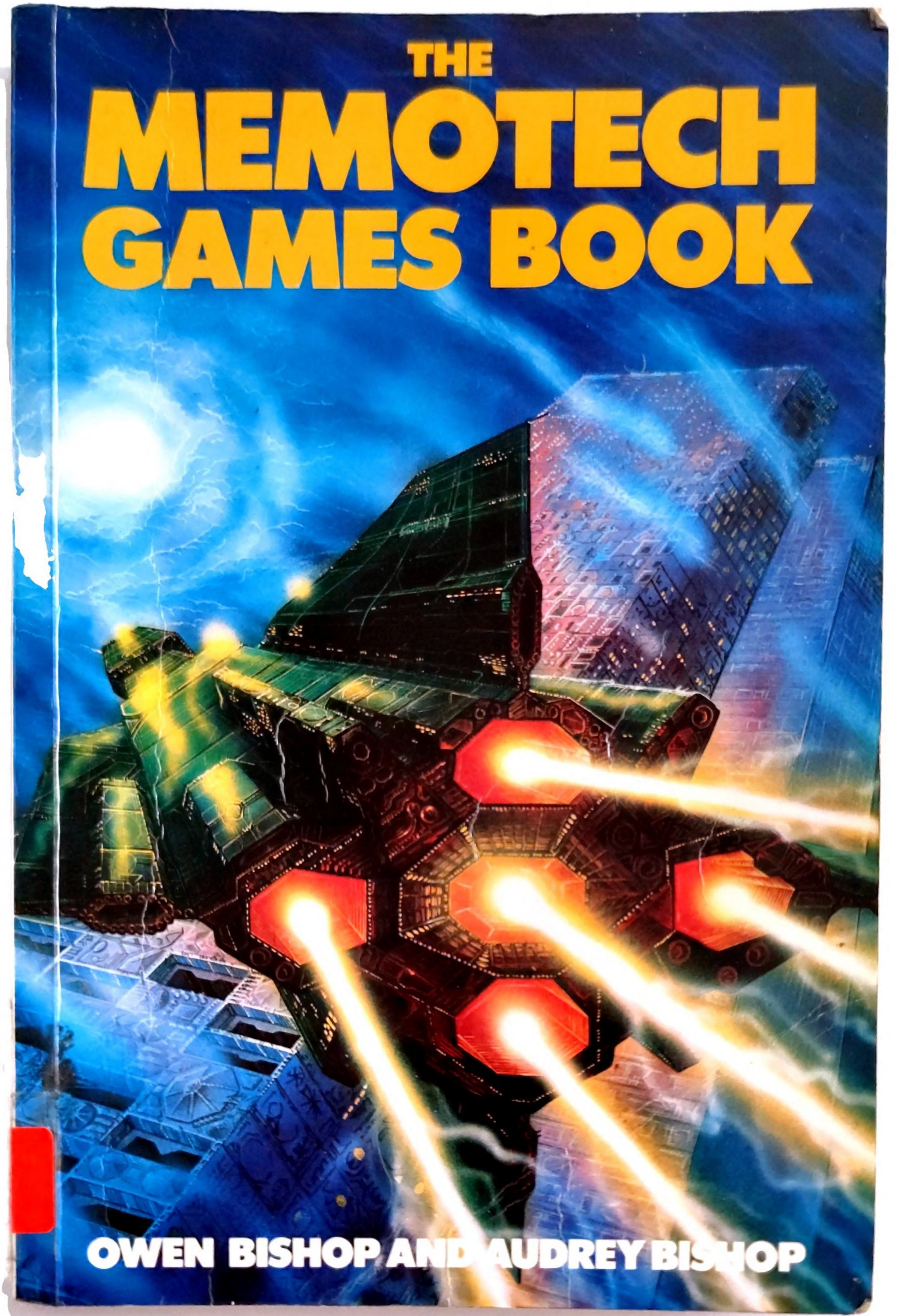
THE MEMOTECH GAMES BOOK



OWEN BISHOP AND AUDREY BISHOP

The Memotech Games Book

Owen Bishop and Audrey Bishop

GRANADA
London Toronto Sydney New York

Granada Technical Books
Granada Publishing Ltd
8 Grafton Street, London W1X 3LA

First published in Great Britain by
Granada Publishing 1984

British Library Cataloguing in Publication Data
Bishop, O. N.
The Memotech games book.
1. Computer games 2. Memotech (Computer)
——Programming
I. Title II. Bishop, Audrey
794.8'028'5425 GV1469.2

ISBN 0-246-12407-5

Typeset by V & M Graphics Ltd, Aylesbury, Bucks
Printed and bound in Great Britain by
Mackays of Chatham, Kent

Contents

Key Points

The BASIC listings with each game have all been made directly from working programs, using a printer attached to the Memotech. This has been done to avoid errors which always seem to appear when listings are independently typeset. All the programs should work perfectly, provided that they are correctly keyed in. The description of each game has a section dealing with keying in. Fortunately the Memotech checks each line for syntax errors, missing brackets and the like as soon as you have typed in each line and pressed RET. This means that many of the common typing errors are detected immediately and can be corrected before the program is run. But some kinds of error can pass unnoticed by the computer. We will describe some of these below.

A mistake that is easy to make is to type 'O' (letter 'oh') instead of '0' (figure zero). The letter 'oh' is used in BASIC words, such as 'FOR' and 'COLOUR', but is never used in this book as a variable. On the screen the 'oh' is squarish in shape, while the zero is shaped like a diamond symbol on a playing card. In the listings, they are similar in shape, but the zero has a stroke across it. Another possible confusion is between 'l' (figure 1) and 'I' (letter 'eye'). The letter is not used alone in any of the listings, but forms a part of certain BASIC words, such as PRINT.

The semicolon ';' is an essential item in many PRINT statements. Keep a sharp eye open for this, especially at the end of program lines. It may be even more difficult to spot in multi-statement lines such as:

```
100 CSR 5,3: PRINT " ";: CSR 8,2: PRINT J$;: CSR Z,3:
PAUSE 500: PRINT T$;
```

If you leave out any of the semicolons, the display on the screen will almost certainly be spoiled.

Many of the programs have GENPAT statements to define the shapes of sprites or graphics characters. It is essential to type in the

numbers exactly. The same applies to the DATA statements at the end of some of the programs. The error-checking of the Memotech insists that GENPAT, COLOUR and SOUND statements have the correct number of parameters (i.e. the row of numbers after the word) when you type in the line. But it does not check that the values themselves are correct. For some parameters a value greater than 255 may not be allowed, for example. If you run the program and get the error message 'Out of range', check the line which is displayed at that stage to make sure that the values are correct. Sometimes a statement has variable names instead of numerical values as parameters. For example, you might have a line:

```
540 COLOUR 2,N
```

This would make the graphics screen change colour. But the value of N must be one of the possible 'colour numbers', 0 to 15. A value less then 0 or greater than 15 causes the 'Out of range message' and the program stops. The offending line is displayed on the screen. Now, the line itself, as checked against the listing may be perfectly correct. If so, what is the error? This is something that beginners often find confusing. What has happened is that the computer has got as far as line 540 and then finds that N is too big or too small to be acceptable as COLOUR parameter. The fault lies not in the line displayed but in some other line in the program in which N was given this unsuitable value. The next step is to search the program for all lines on which the variable N occurs and check that these lines are exactly correct.

Abbreviations

The Memotech's abbreviations for BASIC keywords can remove a lot of the tedium of keying in the listings. They are not explained in the provisional Operator's Manual, but a little experimenting will help you discover them. For example, instead of typing 'GENPAT', type 'GE.'. Note the full-stop. When you have finished typing the whole line and press RET, the full word appears on the listing screen. Similarly, you can type 'CT.' instead of 'CTLSPR' and 'AD.' instead of 'ADJSPR'.

When you are keying in DATA statements it is better always to use the abbreviation 'DA.' instead of the whole word. Then list the various items of the statement without leaving a space between the full-stop and the first item. If you do not do this, the Memotech

inserts an extra space between the word DATA and the first item when the line appears on the list screen. If the DATA statement consists of strings, this results in the addition of an unwanted space to the beginning of the first string. This can ruin the layout of a display when the program is run. Remember to use the abbreviation if you need to edit DATA statements, too.

NODDY programs

Two of the programs include sections that are in NODDY instead of BASIC. The Operator's Guide tells you how to type in Noddy pages, but a few pointers here may be of help.

It is best to key in the Noddy pages before keying in the BASIC program. First type 'NODDY' and press RET. You will see 'Noddy>' at the bottom of the screen. Type (in capitals) the name of the page you want to key in, for example, HORSES. Press RET and the screen clear. On the top line the word HORSES is displayed, which is the name of the page.

Now type in the text, as printed in the program description in this book. Use the arrow keys to move the cursor along the lines or from line to line. Try not to press the RET key out of habit, for this concludes your typing session! If you do press RET by accident, you will see 'Noddy>' again at the bottom of the page. Key in the page name (HORSES) again, and press RET to get back to the same page. Whatever you do, do not press CLS (on the keypad) followed by RET, for this deletes everything you have typed on that page.

When you have finished keying in a page, press RET. When 'Noddy>' appears, key the name of the next page you wish to key in. If you have finished and want to return to BASIC, press CLS, then RET (after 'Noddy>', NOT while you are typing a page!).

The Noddy pages for these games are of two kinds, text pages and program pages. Program pages must be typed exactly as printed. Keep to capital letters, including the page names. With the text pages, you have as much scope as you want in the layout and content, for they make no difference to the actual running of the program. You can alter the text as you please, as long as it still makes sense in connection with the theme of the program.

Having keyed in all the Noddy pages, return to BASIC and key in the BASIC program in the usual way. When you save this BASIC program, the Noddy pages and program are automatically saved too.

Two final tips

(1) If you have been running a program which uses sprites, press the two reset keys (at either end of the space bar) to get rid of the existing sprite definitions before loading the next program.
(2) If a program does not appear to respond when you have pressed a letter key, and it is expected to, it may be that you have accidentally pressed the ALPHALOCK key at some time in the past. The computer will be receiving 'j', for example, when it is expecting 'J'. Try pressing the ALPHALOCK key just once before continuing with the game.

1
Show Jumping

The sun is shining at Memstead as the record crowd of spectators eagerly awaits the semi-finals of the Grand Prix. The course consists of 12 jumps, including three 'doubles' and a water-jump. You are competing, on your favourite horse, and ready for the sound of the judge's bell to send you galloping around the arena.

This is a game of skill and excitement for any number of players from 1 to 4. You have a choice of four horses, each with differing ability and temperament. You also have the option of competing according to one of three different sets of show jumping rules. As will be explained in the 'Variations' section, it is possible to alter the program to include your own horse, so that you can ride it in the Grand Prix.

How to play

When you first run the program you are asked 'How many riders?'. Key in any number between 1 and 4 and press RET. If you are competing by yourself, you may ride more than one or even all four horses in the same event. You are next asked to key in the names of the riders. Key in your name and the names of your friends and press RET after each.

The screen then displays a list of the four horses, and their main features. When you come to ride these horses later, you will find that they act very much as these descriptions indicate. Trooper is easy to handle but really does tend to knock down the fences a little too often! You need precise control when turning the lively Crackerjack. She also occasionally stops in her tracks, which can be most disconcerting if you are just approaching a jump.

Press RET after you have read the list of horses and have decided which one, or ones, you want to ride. If you are competing with

friends or family, you will have to agree among yourselves to take turns with the popular horses. The computer shows you a list of the names of the horses and you are asked to say which horse each competitor wishes to ride. The computer does not allow more than one person to choose the same horse. Press RET after keying in each horses' number.

Now comes a display of the three kinds of competition. These are:

(1) *Foxhunter*: You have unlimited time to complete the course. As in all the competitions, the jumps must be taken in the numbered order. If you knock down a fence or wall or put a foot in the water-jump, you score 4 faults. You will hear the 'clonk' of the hoof on the fence, or the splash of the water. (There are no falls in this program.) You are disqualified if your horse refuses to jump three times during the round. The winner is the competitor with fewest faults.

(2) *Hit and Hurry*: The time limit is 1 minute. During this time you have to attempt as many jumps as you can, in order. If you make a clear jump, you score 3 points. If you knock the jump down or a foot goes into the water-jump you score 1 point. There is no penalty for the first two refusals, but of course they waste time. When the judge's bell goes at the end of the minute, you must go straight to the finishing line and cross it. But if your horse has just refused a jump when the bell goes, you must complete that jump (or be eliminated for three refusals) before going to the finishing line. The winner is the rider with the most points. If riders tie for points, the winner is the rider with the shortest time.

(3) *Scurry*: This is another test of speed. The time limit is 2 minutes and, if you have not completed the first 7 jumps and crossed the finishing line before the bell goes, you are disqualified. You are also disqualified if your horse refuses a jump three times during the round. There are no faults, but 6 seconds is added to your time for each fence knocked down. The winner is the rider taking the least time (after the addition of these penalty times).

Press RET after you have read the list of competitions. Then key 1 to 3 to select which competition you wish to ride in. Press RET.

The Memstead course

The screen now clears to display a plan of the course. The judges' enclosure is just off the bottom of the screen. The dashed lines are the starting and finishing lines. The starting line is on the right and the finishing line is on the left. You will see that the first horse is

already waiting between the two lines, facing the starting line. (NOTE: on certain TV sets the bay and chestnut horses may appear too blurred. If so, you can change their colours, as described later in the 'Variations' section.)

The jumps are numbered from 1 to 12, and must be taken in that order and in the correct direction. These are the rules of the course:

Cross the starting line from left to right. Your horse must be travelling at right-angles to the line as you cross.

Jump 1: a stone wall – jump from left to right (incidentally, at all jumps you must jump across the middle of the jump and at right angles to it).

Jump 2: a hog's back – jump up the screen.

Jump 3 and 4: a double jump consisting of two oxers – jump from top right to bottom left.

Jump 5: a stone wall – jump from right to left.

Jumps 6 and 7: a double jump consisting of two oxers – jump from top left to bottom right. End of *Scurry* course.

Jump 8: a hog's back – jump from left to right.

Jump 9: a water jump – jump from right to left.

Jumps 10 and 11; a double jump consisting of an oxer followed by a hog's back – jump from right to left.

Jump 12: an oxer – jump from left to right.

Cross the finishing line from left to right, travelling at right angles to the direction of the line.

The water of the water-jump is shown as a small blue square. There is also a large blue area slightly to the left of, and below, the centre of the screen. This does not represent water, and horses are allowed to cross this area. The area is a small screen on which you will be able to see your horse later (in close up) as it approaches each jump. The area is also used for displaying short messages. To begin with, it shows the name of rider and horse for the first round. Later it may be used to indicate if a horse has refused or has been disqualified, or if the allowed time has expired.

You are given a few seconds to examine the course, and plan the best route. Then the judge's bell sounds (have you turned up the volume control on the TV?). You are now able to use the keypad at the right-hand side of the keyboard to control your horse. The keys have the following actions.

Up-arrow: makes the horse move forward (i.e. in the direction it is heading, which is not necessarily the same as the direction of the

arrow on the key). A quick tap on the key gets it moving slowly. Holding the key down for a second, or two, brings your horse to full speed.

Down-arrow key: this has the same effect as reining in your horse. The horse slows down and, if you keep your finger on the key, eventually stops. As with a real horse, slowing down takes effect more quickly than gaining speed.

Left-arrow and right-arrow: these are used to turn the horse in the corresponding direction (as seen by a rider on the horse's back). If you hold your finger on this key, the horse turns on the spot. If it was moving forward before it turned, it will move forward in the new direction when the turn is complete. The computer moves the horses like pieces on a draughts-board. A horse moves one row or one column at a time, or it can move diagonally to the next row and column.

After you have heard the bell, ride your horse toward the starting line. At the moment you cross the line, the computer starts to display the time in the top right corner of the screen. Timing starts from when you cross the line, and is displayed in minutes and seconds.

As you approach the first jump, and if you approach it correctly and in the right direction, a side view of the wall or fence appears on the blue 'jump screen'. Forget about the main screen for the moment and look at the 'jump screen' instead. Your horse appears on the left of this screen, heading toward the fence. When you decide that the moment is right to jump, press the HOME key of the keypad. If you press the key at the correct moment, you will see the horse jump the fence. The 'jump screen' then clears. Now turn attention back to the plan view. You will see the horse cross to the other side of the fence and move away. Guide it toward the next jump. The paragraph above describes a clear jump, but you are not always so fortunate!

If you press HOME too soon or too late, the horse will refuse to jump. The picture on the 'jump screen' is replaced with the message 'REFUSED'.

Even with a correctly timed jump, a badly schooled horse may occasionally refuse. Whether it will do this or not on any given occasion is decided at random by the computer. When a horse refuses, its image on the plan turns away from the jump and it usually gallops off in the wrong direction! It is up to you to use your skill to turn it round and let it attempt the jump again.

Whether the horse makes a clear jump or whether it knocks down part of the jump depends on the jumping ability of the horse, the

height of the fence or wall and a random factor decided by the computer. Jumps 3,4,7,8,10,11 and 12 are the more difficult ones. For all jumps, it is important that the horse should be going as fast as you can make it.

You are allowed to break the rules a little when riding from one jump to another, or when circling to re-take a jump. You can ride your horse over the symbols of the jumps (even going straight back over the jump you have just failed to clear). Being able to do this makes it easier to manoeuvre, but these activities are not registered by the computer. If you go too close to the wall of the arena, the horse turns of its own accord and heads toward the centre of the arena.

You will also notice that some of the horses occasionally change direction without warning. Some even stop altogether and face in a direction different from that in which they were originally travelling. These temperamental horses are more difficult to handle but, in this program, such horses have advantages such as speed or jumping ability.

Finishing the course

What you do depends on which competition rules are in force.

In the *Foxhunter competition*, you must take every jump in order from 1 to 12. Jumps that are refused must be re-taken until the horse jumps, or until you are disqualified for 3 faults. After jump 12, you cross the finishing line. You hear the judge's bell as you cross it. the clock shows your course time.

In the *Hit and Hurry competition*, the bell sounds after 1 minute. Do not attempt any more jumps but go straight to the finishing line. If you are on the right-hand side of the course, remember that you have to cross the finishing line from left to right. If your horse has just refused a jump, you must repeat the jump until it is successful (or you are disqualified for 3 refusals). If the jump is successful, you then go to the finishing line.

In the *Scurry competition* you must watch the clock so as to be certain to have completed the course (the first 7 fences, in order) and crossed the finishing line *before* the 2 minutes has expired.

As soon as a round is finished, the horse 'disappears' from the course and the next horse 'appears' ready for the start. After all the riders have completed their rounds, the screen clears and a table of

results is displayed. Times are in seconds and include penalty times in the *Scurry*. The name or names of the winning riders is displayed too. To begin another competition, press the space bar.

Winning tactics

Just as in real show jumping, one of the most important factors is concentration on the part of the rider. A moment's inattention as you approach a jump may lead to a refusal, increasing your chances of being eliminated. Anticipation is important too. As you leave one jump, be ready with your plans for the shortest route to the next. A straight line to the next jump is not necessarily the best, for you must allow for the turns which are needed to align your horse correctly.

The speedier horses are more difficult to control, so much time may be wasted by their running by a jump and the circling needed to bring them back for a second attempt. With such horses it is better to slow down just before the jump (except perhaps the water jump). Give the horse its head on the longer stretches, such as the final run up to jump 12. In the double jumps (especially 3/4 and 6/7) the horse may not take the second fence. Careful control is required. A timely pressure on the 'down-arrow' key may slow it down so that it does not 'run out' and can take the second fence without delay.

Keying in

This is the second longest program in the book, which is to be expected, since programming an eventful show jumping competition is a complicated matter. Part of the program is in the Memotech's second high-level language, NODDY. It is best to begin by keying in the NODDY sections, then follow by keying in the BASIC program.

The way to key in NODDY programs is described in the section called 'Key Points' at the beginning of the book. The program requires four NODDY pages, two being text pages and two being program pages.

There are no particular difficulties in keying in the BASIC program. The most important items to watch out for are the semicolons (;) in PRINT statements. If these are omitted, the display on the 'jump screen' may not appear and the program may come to a halt. Note also the number of blank spaces between quote marks.

The DATA statement of line 4010 looks rather strange. It consists of the characters which have been redefined to make the pictures of the jumps on the 'jump screen'. They use both square brackets, the back-slash, and the letter 'W'. When keying in line 4010, use the abbreviation 'DA.', as explained in 'Key points'.

Points of interest

The program relies on several of the special features of the Memotech. Although it uses only one sprite (the horse on the plan), it needs 8 sprite patterns (lines 70–140). These provide for the eight possible directions in which the horse may travel. It is assigned a new pattern as it changes direction (line 2670 and 2710).

The 'jump screen' is a specially defined virtual screen (VS 2), set up for graphics (line 60). This allows the pictures of the jumping horse, or sometimes messages, to be displayed and cleared independently of displays on the main graphics screen.

NODDY is used for the extensive text displays describing the horses and the competitions.

Program design

20–60	initialising and reading data into arrays; initialising the virtual screen and sprite
70–380	patterns for the sprite and graphics characters
390–600	inputting details of riders, their horses and selection of competition
610–710	displaying the course
720	start of playing loop
730–770	placing the horse at the start
780	passing the starting line?
790	starting the clock; initialising variables for the jumping routines
800–950	jumping routines; including displays, refusal and disqualification
960–980	passing the finishing line?
990	final bell and time
1000–1060	deciding finishing routines for different competitions
1070–1330	displaying results
1340–1350	inviting another competition

2000–2020 subroutine to end sound effects
2300–2310 judge's bell subroutine
2500–2710 subroutine for controlling motion of horse
2800–2810, 2850–2860, 2900–2960, 3000–3010, 3050–3060 subroutines used in calculating scores and finding winners
4000 DATA for locations and features of jumps
4010 DATA for pictures of jumps on 'jump screen'
4020 DATA for characteristics of horses

Text for NODDY

```
HORSES


TROOPER        Bay gelding, 17 hands.
               Reliable performance, but
               not such a strong jumper.

BOOMERANG      Black gelding, 16 hands.
               Strong jumper, but not so
               well schooled.

GIDDYUP        Chestnut gelding, 15 hands
               Good on speed.  Shies at
               water.

CRACKERJACK    Grey mare, 16 hands.
               Quick on turning.  Tends
               to refuse.


     Press RET when you have decided
   which horse you would like to ride
```

Fig. 1.1. NODDY text page, HORSES.

```
COMPETITIONS


FOXHUNTER    4 faults  for knockdowns.
   3 faults for refusals.  Winner
   has fewest faults.


HIT and HURRY  One minute allowed.
   3 points for clear jump; 1 point for
   jump attempted but knocked down.
   Go to finish line as soon as final
   bell sounds, but complete the jump
   if the horse is refusing.  Winner
   has most points.  In a points tie,
   winner has shortest time.

SCURRY  2 minutes for 7 fences. Riders
   taking longer disqualified.  Six
   seconds penalty for each fault, but
   not for up to 2 refusals.  Winner
   has shortest time.        <PRESS RET>
```

Fig. 1.2. NODDY text page, COMPETITIONS.

```
SHOW1


        *DISPLAY HORSES.

        *ENTER

        *RETURN
```

Fig. 1.3. NODDY program page, SHOW 1.

```
SHOW2


        *DISPLAY COMPETITIONS.
        *ENTER
        *RETURN
```

Fig. 1.4. NODDY program page, SHOW 2.

The program

```
10 REM ** SHOW JUMPING **
20 DIM H(4,11),R$(4,15),JU(12,4),JP$(12,
1)
30 FOR J=1 TO 12: FOR K=1 TO 4: READ JU(
J,K): NEXT : NEXT
40 FOR J=1 TO 12: READ JP$(J): NEXT
50 DIM P$(4,11): LET P$(1)="TROOPER": LE
T P$(2)="BOOMERANG": LET P$(3)="GIDDYUP"
: LET P$(4)="CRACKERJACK"
60 CRVS 2,1,4,14,12,4,32: CTLSPR 2,1: CT
LSPR 5,1
70 GENPAT 3,1,0,8,124,255,124,8,0,0
80 GENPAT 3,2,2,68,56,120,120,116,128,0
90 GENPAT 3,3,16,16,56,124,56,56,56,16
100 GENPAT 3,4,128,68,56,60,60,92,2,0
110 GENPAT 3,5,0,4,62,255,62,4,0,0
120 GENPAT 3,6,2,92,60,60,56,68,128,0
130 GENPAT 3,7,16,56,56,56,124,56,16,16
140 GENPAT 3,8,128,116,120,120,56,68,2,0
150 GENPAT 1,129,0,1,17,47,15,15,6,9
160 GENPAT 1,130,132,8,156,254,246,240,4
8,72: GENPAT 1,131,3,6,223,63,62,32,64,1
28
170 GENPAT 1,132,32,240,184,216,32,16,0,
0: GENPAT 1,133,8,4,3,3,19,12,0,0
180 GENPAT 1,134,129,71,231,253,253,60,2
,1
190 GENPAT 1,135,0,0,24,60,126,255,255,1
26
200 GENPAT 1,136,24,24,24,24,24,24,24,24
210 GENPAT 1,137,126,255,255,126,60,24,0
,0
220 GENPAT 1,138,6,15,31,63,63,31,15,6
230 GENPAT 1,139,0,0,0,255,255,0,0,0
240 GENPAT 1,140,96,240,248,252,252,248,
240,96
250 GENPAT 1,141,0,0,248,252,252,252,252
,252
260 GENPAT 1,142,3,6,12,24,48,96,192,128
270 GENPAT 1,143,63,63,63,63,63,31,0,0
280 GENPAT 1,144,0,0,31,63,63,63,63,63
290 GENPAT 1,145,192,96,48,24,12,6,3,1
```

```
300 GENPAT 1,146,252,252,252,252,252,24
8,0,0
310 GENPAT 0,37,0,126,66,90,90,66,126,0
320 GENPAT 0,38,16,0,16,0,16,0,16,0
330 GENPAT 0,91,0,0,0,48,48,48,48,48
340 GENPAT 0,92,0,0,8,32,8,40,8,40
350 GENPAT 0,93,0,8,65,8,93,93,93,73
360 GENPAT 0,123,0,0,0,0,0,0,128,128
370 GENPAT 0,125,0,0,0,0,0,0,0,1
380 LET T$=" "+CHR$(129)+CHR$(130): LET
 J$=CHR$(131)+CHR$(132): LET L$=" "+CHR
$(133)+CHR$(134): LET W$="{"+"}"
390 VS 5: CLS : CSR 2,1: INPUT "How man
y riders? (1-4) ";N$
400 LET N=VAL(N$): IF N<1 OR N>4 THEN
GOTO 380
410 CSR 2,3: PRINT "What are the names
of the riders?"
420 FOR J=1 TO N
430 CSR 2,3+2*J: PRINT J;"  ";: INPUT R
$(J)
440 NEXT
450 PLOD "SHOW1"
460 CLS : CSR 2,2: PRINT "The horses ar
e:"
470 FOR J=1 TO 4: CSR 10,J*2+2: PRINT J
;"  ";P$(J): NEXT
480 CSR 2,12: PRINT "Key a number, 1 to
 4"
490 FOR J=1 TO N
500 CSR 2,J*2+12: PRINT "Rider: ";R$(J)
;: INPUT "   Horse? ";A$
510 LET A=VAL(A$): IF A<1 OR A>4 THEN
GOTO 500
520 LET F=0: FOR K=1 TO 4: IF A=H(K,1)
THEN  LET F=1
530 NEXT : IF F=1 THEN  GOTO 500
540 LET H(J,1)=A: NEXT
550 FOR J=1 TO 4: LET FF=0: FOR K=1 TO
4: IF H(K,1)=J THEN  FOR L=2 TO 7: READ
 H(K,L): NEXT : LET FF=1
560 NEXT : IF FF=0 THEN  FOR L=2 TO 7:
READ FF: NEXT
570 NEXT : PLOD "SHOW2"
580 CLS : CSR 2,2: PRINT "1) FOXHUNTER"
```

```
: CSR 2,4: PRINT "2) HIT AND HURRY": CS
R 2,6: PRINT "3) SCURRY"
590 CSR 2,10: INPUT "Which do you choos
e (1-3) ";A$
600 LET C=VAL(A$)-1: IF C<0 OR C>2 THEN
  GOTO 590
610 VS 4: COLOUR 0,2: COLOUR 1,15: COLO
UR 2,2: COLOUR 4,9: CLS
620 CSR 6,1: PRINT "11";CHR$(135);" 10"
;CHR$(135): CSR 19,1: PRINT "9";CHR$(13
5)
630 CSR 8,2: PRINT CHR$(136): CSR 12,2:
 PRINT CHR$(136): CSR 20,2: PRINT CHR$(
136)
640 CSR 8,3: PRINT CHR$(137): CSR 12,3:
 PRINT CHR$(137): CSR 20,3: PRINT CHR$(
137)
650 CSR 9,4: PRINT "6";CHR$(141): CSR 9
,5: PRINT CHR$(142): CSR 20,5: PRINT "3
";CHR$(144)
660 CSR 8,6: PRINT CHR$(143);"  7";CHR$
(141): CSR 22,6: PRINT CHR$(145): CSR 1
1,7: PRINT CHR$(142): CSR 18,7: PRINT "
4";CHR$(144);"    ";CHR$(146)
670 CSR 10,8: PRINT CHR$(143): CSR 20,8
: PRINT CHR$(145): CSR 21,9: PRINT CHR$
(146): CSR 8,10: PRINT "5";CHR$(135)
680 CSR 9,11: PRINT CHR$(136): CSR 27,1
1: PRINT "2";CHR$(138);CHR$(139);CHR$(1
40): CSR 9,12: PRINT CHR$(137)
690 CSR 22,13: PRINT "8";CHR$(135): CSR
 23,14: PRINT CHR$(136): CSR 23,15: PRI
NT CHR$(137): CSR 11,18: PRINT "%": CSR
 5,19: PRINT "12";CHR$(135);"   &"
700 CSR 20,19: PRINT "%": CSR 7,20: PRI
NT CHR$(136);"   &": CSR 20,20: PRINT "
&  1";CHR$(135): CSR 7,21: PRINT CHR$(1
37);"   %"
710 CSR 20,21: PRINT "&   ";CHR$(136):
CSR 20,22: PRINT "%   ";CHR$(137): COLO
UR 0,4: CSR 19,2: PRINT " "
720 FOR PN=1 TO N
730 VS 2: COLOUR 0,5: COLOUR 2,5: COLOU
R 4,9: CLS
740 SPRITE 1,1,124,19,0,0,H(PN,2)
```

```
750 CSR 1,1: PRINT R$(PN): CSR 1,2: PRI
NT P$(H(PN,1)): PAUSE 8000: CLS
760 GOSUB 2300
770 LET X=15: LET Y=21: LET D=1: LET V=
0
780 GOSUB 2500: IF X<>20 OR  NOT (Y=20
OR Y=21) OR D<>1 THEN  GOTO 780
790 CLOCK "000000": LET J=0: LET FC=0:
LET FD=0: LET FR=0: LET MT=0: LET JK=0
800 LET J=J+1
810 VS 4: COLOUR 0,2: CSR 25,1: PRINT M
ID$(TIME$,3,2);".";RIGHT$(TIME$,2)
820 IF H(PN,11)=3 THEN  VS 2: CSR 0,1:
PRINT "DISQUALIFIED";: PAUSE 5000: CLS
: GOTO 990
830 IF C>0 THEN  LET TU=VAL(MID$(TIME$,
4,1)): IF TU>=C THEN  GOTO 1000
840 GOSUB 2500: IF X<>JU(J,1) OR Y<>JU(
J,2) OR D<>JU(J,3) THEN  GOTO 810
850 VS 2: CLS : COLOUR 0,5: COLOUR 1,15
: CSR 8,3: PRINT JP$(J);: IF J=9 THEN
CSR 8,3: PRINT W$;
860 COLOUR 1,H(PN,2): LET KJ=0: FOR Z=0
 TO 4: IF INKEY$<>"" AND Z<4 THEN  LET
KJ=1
870 CSR Z,3: PAUSE 500: PRINT T$;: NEXT
 : LET JK=0: LET K=1: IF KJ=1 THEN  GOT
O 900
880 IF ASC(INKEY$)=26 THEN  LET JK=1: G
OTO 900
890 LET K=K+1: IF K<50 THEN  GOTO 880
900 LET FR=0: IF RND*150<H(PN,6) OR JK=
0 THEN  CLS : CSR 2,1: PRINT "REFUSED":
 LET H(PN,11)=H(PN,11)+1: LET FR=1: LET
 D=D+4: GOTO 810
910 IF J<>9 AND (V<3 OR RND<.05 OR H(PN
,4)-RND*6+8<JU(J,4)) THEN  LET H(PN,9)=
H(PN,9)+1: SOUND 0,8000,240,5,-50,20,1:
 GOSUB 2000
920 IF J=9 THEN  IF H(PN,7)=1 AND RND<.
5 OR V<3 THEN  LET H(PN,9)=H(PN,9)+1: S
OUND 3,4,14: PAUSE 500: SOUND 3,0,0
930 CSR 5,3: PRINT "  ";: CSR 8,2: PRIN
T J$;: GOSUB 2590: PAUSE 800: CSR 8,2:
PRINT L$;: PAUSE 800: CSR 9,2: PRINT "
```

```
  ";: CSR 9,3: PRINT T$;: LET FR=0
940 PAUSE 1500: CLS : GOSUB 2590
950 IF C<2 AND J<12 OR C=2 AND J<7 THEN
  GOTO 800
960 VS 4: CSR 25,1: PRINT MID$(TIME$,3,
2);".";RIGHT$(TIME$,2)
970 IF C=2 AND VAL(MID$(TIME$,4,1))=2 T
HEN  GOSUB 2300: VS 2: CSR 0,1: PRINT "
DISQUALIFIED": PAUSE 5000: GOTO 990
980 GOSUB 2500: IF X<>10 OR  NOT (Y=19
OR Y=20) OR D<>1 THEN  GOTO 960
990 GOSUB 2300: LET H(PN,8)=VAL(RIGHT$(
TIME$,4)): NEXT : GOTO 1060
1000 IF FC=1 THEN  GOTO 1030
1010 SOUND 0,1000,240,0,-18,50,1: GOSUB
 2000
1020 VS 2: CSR 0,1: PRINT "TIME EXPIRED
": PAUSE 10+5000*(C-1)
1030 IF C=1 AND FR=1 THEN  LET FC=1: GO
TO 840
1040 IF C=1 THEN  LET H(PN,10)=J-1: GOT
O 960
1050 IF C=2 THEN  VS 2: CSR 0,1: PRINT
"DISQUALIFIED": PAUSE 5000: LET H(PN,11
)=3: GOTO 990
1060 VS 4: CLS : VS 2: CLS : VS 5: CLS
1070 CSR 2,1: PRINT "COMPETITION RESULT
S"
1080 CSR 2,3: PRINT "RIDER": CSR 15,3:
PRINT "HORSE"
1090 FOR J=1 TO N
1100 CSR 2,J*3+3: PRINT R$(J): CSR 15,J
*3+3: PRINT P$(H(J,1)): LET MM=INT(H(J,
8)/100): LET SS=H(J,8)-MM*100: LET H(J,
8)=MM*60+SS
1110 NEXT
1120 CSR 26,3: IF C=0 THEN  PRINT "FAUL
TS"
1130 IF C=1 THEN  PRINT "POINTS"
1140 IF C=2 THEN  PRINT "PENALTY"
1150 CSR 34,3: PRINT "TIME"
1160 FOR J=1 TO N: CSR 26,J*3+3: IF H(J
,11)=3 THEN  PRINT "DISQUALIFIED": GOTO
 1200
1170 ON C GOSUB 2800,2850,2900
```

```
1180 PRINT H(J,10)
1190 CSR 34,J*3+3: PRINT H(J,8)
1200 NEXT
1210 CSR 2,20: PRINT "WINNER: ";
1220 LET NS=-1: LET LS=1000: LET BS=0:
FOR J=1 TO N: IF H(J,11)=3 THEN  GOTO 1
240
1230 ON C GOSUB 2950,3000,3050
1240 NEXT : IF C<>1 THEN  GOTO 1300
1250 IF NS<2 THEN  GOTO 1300
1260 LET NS=-1: LET LS=1000: FOR J=1 TO
 N: IF H(J,11)=3 THEN  GOTO 1290
1270 LET H(J,10)=H(J,8)
1280 GOSUB 2950
1290 NEXT
1300 FOR J=1 TO N: IF H(J,11)=3 THEN  G
OTO 1330
1310 IF C=1 AND NS<2 AND H(J,10)=BS THE
N  PRINT R$(J);"  ";: GOTO 1330
1320 IF H(J,10)=LS THEN  PRINT R$(J);"
  ";
1330 NEXT
1340 IF INKEY$<>" " THEN  GOTO 1340
1350 RESTORE 4020: FOR J=1 TO 4: FOR K=
1 TO 11: LET H(J,K)=0: NEXT : NEXT : GO
TO 390
2000 IF PEEK(64082)<>PEEK(64086) THEN
GOTO 2000
2010 SOUND 0,0,0
2020 RETURN
2300 SOUND 0,1000,240,0,-18,50,1
2310 GOSUB 2000: RETURN
2500 LET B=ASC(INKEY$)
2510 IF B=11 AND V<H(PN,3) THEN  LET V=
V+1
2520 IF B=10 THEN  LET V=V-3: IF V<0 TH
EN  LET V=0
2530 IF B=8 THEN  LET D=D+1: GOTO 2690
2540 IF B=25 THEN  LET D=D-1: GOTO 2690
2550 IF RND*300<H(PN,6) THEN  LET V=0:
LET D=D+1: LET FR=1
2560 IF D>8 THEN  LET D=D-8
2570 IF D<1 THEN  LET D=D+8
2580 IF V=0 THEN  GOTO 2670
2590 IF D=1 OR D=2 OR D=8 THEN  LET X=X
+1
```

```
2600 IF D>3 AND D<7 THEN  LET X=X-1
2610 IF D>1 AND D<5 THEN  LET Y=Y-1
2620 IF D>5 THEN  LET Y=Y+1
2630 IF X=0 THEN  LET X=1: LET D=1
2640 IF X=32 THEN  LET X=31: LET D=5
2650 IF Y=0 THEN  LET Y=1: LET D=7
2660 IF Y=23 THEN  LET Y=22: LET D=3
2670 ADJSPR 0,1,D: ADJSPR 2,1,X*8+4: AD
JSPR 3,1,(24-Y)*8-5
2680 PAUSE 600-V*100: RETURN
2690 IF D=9 THEN  LET D=1
2700 IF D=0 THEN  LET D=8
2710 ADJSPR 0,1,D: PAUSE 1000-150*H(PN,
5): RETURN
2800 LET H(J,10)=4*H(J,9)+3*H(J,11)
2810 RETURN
2850 LET H(J,10)=3*H(J,10)-2*H(J,9)
2860 RETURN
2900 LET H(J,9)=6*H(J,9)
2910 LET H(J,10)=H(J,9)
2920 LET H(J,8)=H(J,8)+H(J,10)
2930 RETURN
2950 IF H(J,10)<LS THEN  LET LS=H(J,10)
2960 RETURN
3000 IF H(J,10)>BS THEN  LET BS=H(J,10)
: LET NS=NS+1
3010 RETURN
3050 LET H(J,10)=H(J,8)
3060 GOSUB 2950: RETURN
4000 DATA 23,21,1,4,29,12,3,4,23,5,6,5,
21,7,6,5,10,11,5,4,8,4,8,4,10,6,8,5,22,
14,1,5,21,2,5,2,13,2,5,5,9,2,5,5,6,20,1
,5
4010 DATA [,],\,\,[,\,\,],W,\,],\
4020 DATA 6,3,2,3,0,0,1,3,5,2,3,0,9,5,3
,3,1,1,14,3,3,5,5,0
```

Variations

You may substitute your own horse names and characteristics for those provided by the programs. New names may be typed in at line 50 (maximum 11 letters each). You can rewrite part or all of the Noddy HORSES page. Characteristics are set by the data statements of line 4020. The values are in groups of 6 for each horse:

Colour: one of the Memotech's colour values, 1 to 15. If you find the bay and light bay horses difficult to see on your TV, substitute another number (such as 1, 14, or 15) for the '6' and the '9' in this line.

Speed: any value from 1 to 5, where 5 is the fastest.

Jumping ability: any value from 0 to 5, where 5 is the best jumper.

Turning ability: any value from 0 to 5, where 5 is the best.

Reliability: any value from 0 to 5 where 0 is very reliable and 5 is the least reliable.

Reaction to water: 0 if unaffected, 1 if liable to refuse.

Beginners may find it easier to make successful jumps if the '50' in line 890 is increased to '100'.

2
Black Hole

You are in command of a spacecraft exploring a distant part of the Galaxy. This region is of special attraction to explorers because it teems with nuggets of pure gold! Lots of them are as big as a house, or even larger! Space explorers, such as yourself, find a rich harvest there, just for the taking.

It is thought that these nuggets were produced by nuclear fusion during the last stages of collapse of a supergiant star which once existed in this region. Now all that remains are a few traces of gas, the nuggets – and the Black Hole!

Unfortunately, the last stages of collapse also produced a Black Hole, into which the greater part of the mass of the star disappeared. Black holes need not be large – this one is too small to be seen until you are within a kilometre, or so, of it. Yet the strong pull of the condensed matter of the former star makes itself felt at distances of thousands of kilometres. Any object within this range, including your spacecraft, is drawn forcibly toward the Black Hole at ever increasing velocity, never to be seen again.

Gold hunting in space is a hazardous business but, in readiness for this your craft is equipped with powerful booster rockets which can overcome the enormous gravitational attraction of the Black Hole – provided that you act in time. Since you cannot see the Hole, you can only guess where it is by noting the extent to which your craft is pulled out of its intended course. Act quickly, for a few moments delay will almost certainly lead to disaster.

How to play

When the program is run you are asked to choose at which level of difficulty you wish to play. Level 1 is the easiest; level 10 is the

hardest. The higher the level, the stronger the effect of the Black Hole. Key in your chosen level and press RET.

The screen becomes dark blue with a black border. Scattered over the screen at random you see 20 gold nuggets. Your spacecraft is an impressive vehicle, with flashing lights and pulsating plasma engines. The positions of the nuggets, the Black Hole and your craft are decided at random, so no two games are ever exactly alike.

The top right-hand corner of the screen displays the elapsed time, in seconds. You have 60 seconds in which to collect as many nuggets as possible, preferably without being drawn into the Black Hole. The craft is moved by pressing one of these keys:

Z to move left
X to move right
: to move up
/ to move down

You cannot move the craft completely off the screen. You collect a nugget by moving the craft so that you can just see the nugget through the window in the centre of the craft. A collected nugget disappears from sight. The program keeps score of how many nuggets you have collected.

Sooner or later, you will find that, as well as moving as you intended, the craft moves unexpectedly and sometimes very quickly. Your spacecraft is being attracted by the Black Hole. When you press the control keys now, their effect is automatically boosted to three times their former power. This is enough to overcome the effect of the Black Hole, provided that you press the right key, or keys, at the right times. If you manage to move your craft beyond the range of the Black Hole, power is restored to normal. While you are in the vicinity of the Hole your movements will be erratic, like a frenzied tug-of-war between you and the force of the Hole. But it is still possible to collect nuggets, either by accident or design and, as a bonus, these nuggets score three times as much as those out of range of the Hole. It is worth risking danger to improve your score, but do not get sucked into the Black Hole if you can help it, for this carries a penalty.

One point to remember is that, whereas your commands affect the craft only when you press a key, the gravitational force of the Hole acts all the time. If your craft is within range of the Hole and you do nothing to save it, it will move steadily toward the Hole of its own accord and soon be lost.

Your departure into the Hole is accompanied by suitable sound

effects, after which you are told your score, the best score so far, and the length of time for which the game lasted. If you manage to survive the full 60 seconds, the chimes sound and you are given the same information as above.

When you are ready to play again, press the space bar. You will then be taken back to the beginning of the program and given the opportunity to select the level of play. The Hole will almost certainly be in a different position next time. The best score is retained when you play again, though it is lost if you re-run the program.

Winning tactics

Nuggets score 1 point if they are beyond the range of the hole, or three points if they are within range. The final score is calculated on a 'points per minute' basis. It is therefore best to collect nuggets as quickly as you can, just in case you soon find yourself inside the Hole. If you fall into the Hole, you lose 10 points, so you could end up with a negative score. If time is nearly running out and you have collected most of the nuggets, it might be better not to try to get those which are near the Hole for, although they are worth 3 points each, there is a strong risk of losing 10 points if you fail. Safer to let the 'clock' bring the game to an end. However, you may need that extra point or two, to better your previous score, or that of someone else. In this case the risk may be worthwhile. Or perhaps you just like running risks!

If the Hole is near the top or bottom, or to one side, it is fairly easy to pick up nuggets from distant areas. If the Hole is near the centre, it is a tricky business to circumnavigate it to get the nuggets from the opposite side. Try making a dash for it!

If you think you know where the Hole is, it is well worth venturing closer to it to collect the valuable nuggets in that region. With practise, it is often possible to position yourself with a nugget between you and the Hole, and then let the Hole draw you on to the nugget, so capturing it. Then a well-timed boost of power takes you safely out of range again. Happy hunting!

Keying in

There are few problems with keying in, other than typing the correct values into the GENPAT statements.

Program design

20–30	initialising variables
40	initialising sprite control
50–80	patterns for the sprites and the nuggets
90–110	requesting level of play
120	setting screen colours
130	displaying nuggets at random
140–150	selecting position of Black Hole and initial position of Black Hole and initial position of craft at random
160	repeating the above if craft is placed too near the hole to begin with
170–180	sound effect of craft
190	starting the clock
200	initialising sprites (craft)
210	displaying elapsed time
220	detecting 'time up'
230	change colour of sprites; then read keyboard
240–270	re-calculating craft position as a result of key-press (if any)
280	moving craft under your command; then change colours of sprites
290–350	calculating the effect of the Hole
360–370	moving craft under influence of the Hole, then return to start of playing loop
380–420	changing screen, and sound effects of the Hole
430–470	displaying scores and time
480–500	inviting next game
510–570	changing screen and sound effects when time is up
1000–1060	subroutine to move craft (but not beyond edge of screen) and to detect and collect nuggets

Points of interest

The spacecraft consists of three sprites, one behind the other, which are moved together. These are 16×16 sprites, so they require four GENPAT statements each. Sprite 1 is coloured black and forms the body of the craft. Sprite 2 is alternately transparent and white, and Sprite 3 is alternately dark red and light green. These provide the flashing lights and rocket exhausts.

The program

```
10 REM ** BLACK HOLE **
20 LET N=8: LET BS=-1000: LET SC=0
30 LET SC=0: SBUF 2
40 CTLSPR 2,3: CTLSPR 6,3
50 GENPAT 4,1,3,31,121,255,194,254,254,1
94: GENPAT 5,1,127,35,33,1,1,0,0,0: GENP
AT 6,1,224,248,158,255,67,127,127,67: GE
NPAT 7,1,254,196,132,128,128,0,0,0
60 GENPAT 4,2,0,0,6,0,0,0,0,0: GENPAT 5,
2,0,0,0,32,0,1,1,1: GENPAT 6,2,0,0,96,0,
0,0,0,0: GENPAT 7,3,0,0,0,0,0,14,0,4
70 GENPAT 4,3,0,0,0,0,60,0,0,60: GENPAT
5,3,0,0,0,0,0,112,0,32: GENPAT 6,3,0,0,0
,0,60,0,0,60: GENPAT 7,3,0,0,0,0,0,14,0,
4
80 GENPAT 1,129,28,62,254,252,126,127,24
6,224
90 VS 5: CLS
100 INPUT "LEVEL? (1-10)  ";L$
110 LET L=VAL(L$): IF L<1 OR L>10 THEN
GOTO 100
120 VS 4: COLOUR 0,4: COLOUR 1,10: COLOU
R 2,4: COLOUR 4,1: CLS
130 FOR J=1 TO 20: CSR INT(RND*31),INT(R
ND*23): PRINT CHR$(129): NEXT
140 LET R=INT(RND*192): LET C=INT(RND*25
6)
150 LET RC=INT(RND*192): LET CC=INT(RND*
256)
160 IF ABS(R-RC)<85 OR ABS(C-CC)<85 THEN
  GOTO 140
170 SOUND 1,2500,240,103,0,65535,1
180 SOUND 2,2000,240,200,0,65535,1
190 CLOCK "000000"
200 SPRITE 1,1,CC,RC,0,0,1: SPRITE 2,2,C
C,RC,0,0,0: SPRITE 3,3,CC,RC,0,0,6
210 CSR 30,0: PRINT RIGHT$(TIME$,2)
220 IF VAL(RIGHT$(TIME$,3))>=100 THEN  G
OTO 510
230 ADJSPR 1,2,0: ADJSPR 1,3,6: LET A$=I
NKEY$
240 IF A$="/" THEN  LET RC=RC-N
250 IF A$=":" THEN  LET RC=RC+N
```

```
260 IF A$="Z" THEN  LET CC=CC-N
270 IF A$="X" THEN  LET CC=CC+N
280 GOSUB 1000: ADJSPR 1,2,15: ADJSPR 1,
3,3
290 LET RA=ABS(R-RC): LET CA=ABS(C-CC)
300 LET N=24: IF RA<16 AND CA<16 THEN  L
ET T$=RIGHT$(TIME$,2): GOTO 380
310 IF RA>96 OR CA>160+L*8 THEN  LET N=8
: GOTO 210
320 IF RA<48 THEN  LET RC=RC+8*SGN(R-RC)
330 IF RA<104 THEN  LET RC=RC+8*SGN(R-RC
)
340 IF CA<88 THEN  LET CC=CC+8*SGN(C-CC)
350 IF CA<168+8*L THEN  LET CC=CC+8*SGN(
C-CC)
360 GOSUB 1000
370 GOTO 210
380 CLS : VS 5: PAPER 1: INK 15: CLS
390 SBUF 1: SOUND 1,0,0: SOUND 2,0,0
400 SBUF 2: SOUND 1,6000,0,-50,40,100,1
410 IF PEEK(64092)<>PEEK(64096) THEN  GO
TO 410
420 SOUND 1,0,0: SOUND 3,4,15: PAUSE 250
: SOUND 3,5,10: PAUSE 400: SOUND 3,6,7:
PAUSE 600: SOUND 3,0,0
430 LET SC=INT(SC*60/VAL(T$))-10
440 IF SC>BS THEN  LET BS=SC
450 PRINT : PRINT "Your score = ";SC
460 PRINT : PRINT "Best score = ";BS
470 PRINT : PRINT "Time = ";T$;" seconds
"
480 CSR 2,20: PRINT "Press space bar for
 next mission"
490 IF INKEY$<>" " THEN  GOTO 490
500 CLS : PAPER 5: INK 15: GOTO 30
510 CLS : VS 5: PAPER 9: INK 2: CLS
520 SBUF 1: SOUND 1,0,0: SOUND 2,0,0
530 SBUF 4: SOUND 1,1507,240,0,-15,45,1:
 SOUND 1,1911,240,0,-15,80,1
540 IF PEEK(64092)<>PEEK(64096) THEN  GO
TO 540
550 SOUND 1,0,0
560 PRINT : PRINT "TIME UP!"
570 LET T$="60": GOTO 440
1000 IF RC<4 THEN  LET RC=4
```

```
1010 IF RC>192 THEN  LET RC=192
1020 IF CC<0 THEN  LET CC=0
1030 IF CC>255 THEN  LET CC=255
1040 ADJSPR 2,1,CC: ADJSPR 3,1,RC: ADJSP
R 2,2,CC: ADJSPR 3,2,RC: ADJSPR 2,3,CC:
ADJSPR 3,3,RC
1050 CSR CC/8,24-RC/8: IF SPK$=CHR$(129)
 THEN  CSR CC/8,24-RC/8: PRINT " ": LET
SC=SC+N
1060 RETURN
```

Variations

If you are keen on designing your own graphics, you could have two, or more, kinds of nugget, of different shapes and colours. You will need to add the necessary GENPAT 1 statements to line 80. They could be defined as non-ASCII characters from 130 onward (the existing nuggets are character 129). Insert extra program lines like line 130 to distribute them at random on the screen. They may be detected by adding more lines like 1050 to the subroutine. SPK$ is the statement used to examine the screen and find if a nugget is there. Use statements such as 'IF SPK$=CHR$(130) THEN ...', with appropriate additions to the score.

There is scope for varying the scoring. As an alternative approach, you could dispense with the clock and go for high scores. When a given number of nuggets (say 15) have been collected, the supply could be replaced by returning to line 130. A skilful player could amass a huge score before being sucked in the Black Hole.

3
Magic Jigsaw

Have you heard the story about poor old Humpty Dumpty? This is what happened ...

Humpty Dumpty sat on a wall.
Humpty Dumpty had a big fall.
All the King's horses and all the King's men,
Couldn't put Humpty Dumpty together again.

He was very sad to be broken into so many pieces. The King's horses and the King's men tried so hard to put him together, but they could not.

Could you put him together again? You only have to try!

How to play

The jigsaw has 9 pieces, which appear on the screen when the program is run. They are not in their right places. Some of them are upside down, too.

When all the pieces are showing on the screen, you hear two musical tones – the first two notes of the Humpty Dumpty tune. This tells you that the Memotech is ready for you to begin.

You can choose which of two things you wish to do. One is to turn a piece the other way up. The other thing is to move a piece to another place on the screen.

Find the number of the piece you want to turn or move. The number is there on the screen just below the piece. Find a key in the keypad which has the same number. Press the key once.

Now you will hear four notes – the first four notes of Humpty Dumpty's tune. This tells you that the computer is waiting to know what you want it to do. If you want to move the piece, decide which place you want it to go to. Then press the key which has the same

number as this place. If you just want to turn the piece round, press the '0' key.

Before you begin to 'put Humpty together again', try moving and turning some pieces a few times. Do not worry, it is not possible to lose any of the pieces of this kind of jigsaw.

Now try to work out where each piece should go. Humpty Dumpty is sitting on the wall and there is a plant with flowers on the right of the screen. One at a time, move each piece to its proper place. Make sure that they are all the right way up. When all the pieces are in their right places and are all the right way up, you will see all the pieces of Humpty Dumpty joining together again. Then Humpty will play you the whole of his tune.

If you want to hear the tune again, press the space bar. You can hear the tune as many times as you like.

Winning tactics

The sky is blue, so look for pieces that might have sky on them and move these to the top places on the screen. Turn them so that the sky is at the top of each piece.

Grass is green, so this is likely to be at the bottom of the picture.

Keying in

Some kind parent, or elder brother or sister, will have to key this in for the youngsters. They will also have to read the introduction to them and explain to them how to play. The POKE statements on line 20 enable the keypad and disable the BRK key (so it can be used to key in '9'). Take great care to type in these two statements exactly. It is a good idea to save the program before you run it for the first time, for an error in line 20 could cause the loss of everything you have typed in.

When the program is running, it cannot be stopped by using the BRK key. Wait until after two notes have been heard or until the whole tune had been heard, then press 'S'. This restores the keypad and BRK key to normal and ends the program.

Expressions on lines 200 230, 400, 900, 910 and 3510 must be exactly correct. If your display looks odd, check these lines carefully. Finally, the DATA statements all need care to ensure a proper picture of Humpty and the right tune.

Program design

30–210	initializing, displaying jigsaw pieces
220–250	detecting if all pieces are in their correct positions and the right way up. If so, go to the final routine at line 460
260–320	requesting input, with tunes
330–380	swapping two pieces
390–450	turning a piece the other way up
460–510	displaying the pieces put together to make the final picture
520	playing the whole tune
530–560	inviting repeat of the tune
900–920	subroutine to calculate, the top left corner of each piece
1000–1030	subroutine to display a piece the right way up
2000–2030	subroutine to display a piece upside down
3000–3060	subroutine to play T notes of the tune
3500–3540	subroutine used to find the DATA for a single piece and then call the subroutines needed to display it
4000–4020	subroutine to turn off the sound generator
5000–5170	DATA for the pieces
6000–6030	DATA for the tune

The program

```
10 REM ** MAGIC JIGSAW **
20 DIM F(9,2): SBUF 42: POKE 64145,132:
POKE 64862,13
30 VS 4: COLOUR 0,12: COLOUR 2,12: COLOU
R 4,10: CLS
40 FOR P=1 TO 9
50 LET N=INT(RND*9)+1: LET F=0
60 FOR J=1 TO 9
70 IF F(J,1)=N THEN  LET F=1
80 NEXT
90 IF F=1 THEN  GOTO 50
100 LET F(P,1)=N: GOSUB 900
110 LET Q=INT(RND*2): LET F(P,2)=Q
120 IF N=7 THEN  GOTO 150
130 LET BS=INT(144*(2-2*INT((N-1)/3)+(N-
1)/3)+.5)
```

```
140 FOR J=1 TO BS: READ B: NEXT
150 ON Q GOSUB 1000,2000
160 RESTORE 5000
170 NEXT
180 COLOUR 0,15
190 COLOUR 0,1: COLOUR 1,15: FOR J=1 TO
9
200 CSR INT(6+((J-1)/3-INT((J-1)/3))*27+
.5),INT(22-INT((J-1)/3)*7+.5): PRINT J
210 NEXT
220 LET FF=1: FOR N=1 TO 9
230 IF F(N,1)<>INT(7-INT((N-1)/3)*3+((N-
1)/3-INT((N-1)/3))*3+.5) OR F(N,2)<>0 TH
EN  LET FF=0
240 NEXT
250 IF FF=1 THEN  GOTO 460
260 LET T=2: GOSUB 3000
270 LET A$=INKEY$: IF A$="" THEN  GOTO 2
70
280 IF A$="S" THEN  GOTO 550
290 LET P=VAL(A$): IF P<1 THEN  GOTO 260
300 LET T=4: GOSUB 3000
310 LET R$=INKEY$: IF R$="" THEN  GOTO 3
10
320 LET R=VAL(R$): IF R=0 THEN  GOTO 390
330 LET PT=F(P,1): LET QT=F(P,2)
340 LET F(P,1)=F(R,1): LET F(P,2)=F(R,2)
350 LET F(R,1)=PT: LET F(R,2)=QT
360 GOSUB 3500
370 LET P=R: GOSUB 3500
380 GOTO 220
390 RESTORE 5000: IF F(P,1)=7 THEN  GOTO
 420
400 LET BS=144*(2-2*INT((F(P,1)-1)/3)+(F
(P,1)-1)/3)
410 FOR L=1 TO BS: READ B: NEXT
420 IF F(P,2)=0 THEN  LET F(P,2)=1: GOTO
 440
430 LET F(P,2)=0
440 GOSUB 900: ON F(P,2) GOSUB 1000,2000
450 GOTO 220
460 CLS
470 RESTORE 5000: FOR N=1 TO 9
480 LET X=INT(4+((N-1)/3-INT((N-1)/3))*2
4+.5)
```

```
490 LET Y=INT(15-INT((N-1)/3)*6+.5)
500 GOSUB 1000
510 NEXT
520 LET T=42: GOSUB 3000
530 LET A$=INKEY$: IF A$="" THEN  GOTO 5
30
540 IF A$<>"S" THEN  GOTO 520
550 POKE 64145,160: POKE 64862,15
560 STOP
900 LET X=INT(3+((P-1)/3-INT((P-1)/3))*2
7+.5)
910 LET Y=INT(16-INT((P-1)/3)*7+.5)
920 RETURN
1000 FOR LY=Y TO Y+5: FOR LX=X TO X+7
1010 READ B
1020 CSR LX,LY: COLOUR 0,B: PRINT " "
1030 NEXT : NEXT : RETURN
2000 FOR LY=Y+5 TO Y STEP -1: FOR LX=X+7
 TO X STEP -1
2010 READ B
2020 CSR LX,LY: COLOUR 0,B: PRINT " "
2030 NEXT : NEXT : RETURN
3000 RESTORE 6000
3010 FOR J=1 TO T
3020 READ F,D
3030 IF F=99 THEN  GOSUB 4000: GOTO 3050
3040 SOUND 0,F,160,0,-12,D*15,1
3050 NEXT
3060 GOSUB 4000: RETURN
3500 RESTORE 5000: IF F(P,1)=7 THEN  GOT
O 3530
3510 LET BS=INT(144*(2-2*INT((F(P,1)-1)/
3)+(F(P,1)-1)/3)+.5)
3520 FOR L=1 TO BS: READ B: NEXT
3530 GOSUB 900: ON F(P,2) GOSUB 1000,200
0
3540 RETURN
4000 IF PEEK(64082)<>PEEK(64086) THEN  G
OTO 4000
4010 SOUND 0,0,0
4020 RETURN
5000 DATA 8,8,8,8,8,7,8,8,8,8,8,8,8,7,8,
8,8,8,8,8,1,1,8,8
5010 DATA 2,8,8,8,8,8,8,8,2,2,8,2,8,8,2,
8,2,2,2,2,2,2,2,2
```

```
5020 DATA 7,8,8,8,8,8,8,8,7,8,8,8,8,8,8,
8,1,1,8,8,8,8,8,8
5030 DATA 8,8,8,8,8,8,2,8,8,8,8,2,2,8,2,
8,2,2,2,2,2,2,2,2
5040 DATA 8,8,8,8,2,8,9,8,8,8,9,8,2,9,10
,9,8,9,10,9,2,8,9,8
5050 DATA 8,8,9,8,2,8,8,8,2,8,8,8,2,8,2,
8,2,2,2,2,2,2,2,2
5060 DATA 4,7,4,10,10,15,10,10,4,7,4,10,
10,10,10,10,4,4,7,10,10,10,10,10
5070 DATA 4,4,4,13,13,13,13,13,4,4,4,4,7
,7,7,7,8,8,8,8,8,7,8,8
5080 DATA 15,10,10,4,4,4,4,4,10,10,10,4,
4,4,4,4,10,10,10,4,4,4,4,4
5090 DATA 13,13,13,7,7,4,4,4,7,7,4,4,4,7
,4,4,7,8,8,8,8,11,8,8
5100 DATA 4,4,15,15,15,15,4,4,4,4,15,15,
15,15,4,4,4,4,4,4,4,4,4,4
5110 DATA 4,4,4,4,9,4,4,4,4,4,4,9,10,9,4
,4,8,8,8,8,9,8,8,8
5120 DATA 4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,
4,4,4,4,4,4,4,4,4
5130 DATA 4,4,4,4,4,10,10,10,4,4,4,4,10,
10,10,10,4,10,4,10,10,1,10,10
5140 DATA 4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,
15,4,4,4,4,4,4,4,15
5150 DATA 10,4,4,4,4,4,4,15,10,10,4,4,4,
4,4,4,1,10,10,4,4,4,4,4
5160 DATA 4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,
4,15,15,15,4,4,4,4,4
5170 DATA 15,15,15,4,4,4,4,4,4,4,4,4,4,4
,4,4,4,4,15,4,4,4,4,4
6000 DATA 3822,4,3034,2,3405,4,2863,2,30
34,2,2551,2,2024,2,1911,6,99,0
6010 DATA  3034,4,2551,2,2863,4,2272,2,2
551,2,3034,2,3822,2,3405,6,99,0
6020 DATA 3034,2,2863,2,2551,2,2863,2,25
51,2,2272,2,2551,2,2272,2,2024,2,1911,4,
99,0
6030 DATA 1911,1,1703,1,1517,2,1911,2,19
11,2,1432,2,1432,2,1517,2,1703,2,1911,2,
2024,2,1911,6,99,0
```

Variations

It is easy to replace the picture and tune with entirely new ones. To change the picture, follow these steps:

(1) Draw a grid of squares, 24 across and 18 down.

(2) Subdivide the grid into 3 rows of 3 large squares, each consisting of a grid of 8 × 6 smaller squares.

(3) Lightly sketch your design on this grid.

(4) Colour in the squares, using any or all of the 15 colours:

(1) black	(8) mid red
(2) mid green	(9) light red
(3) light green	(10) dark yellow
(4) dark blue	(11) light yellow
(5) light blue	(12) dark green
(6) dark red	(13) purple
(7) cyan	(14) grey
	(15) white

(5) When you have completed your design, take each of the 9 main squares in turn and work out a pair of DATA lines for each. The pair of lines will contain 48 numbers altogether, one for each of the smaller squares. Run along the top rows of the large squares from left to right, writing down the number of the colour each small square contains. Continue in this way to the second row, and so on down to the bottom row. Repeat this for each of the 9 large squares. In the end, you will have 18 DATA lines to replace lines 5000 to 5170 of this program.

The tune can be changed by altering the values in lines 6000 to 6030. The values are in pairs:

(i) Frequency – as for the 7-parameter SOUND statement, e.g. middle C is 3822.

(ii) Duration.

You can adjust the tempo by altering the 15 in line 3040 to some other value. But, if you make it too small, some of the shorter notes will not be played. You also need to alter the value for T in line 520 and the size of the sound buffer (line 20) to the number of notes in your tune.

4
Cops and Robbers

This is a game for two players. One player takes the part of the cops, while the other takes the part of the robbers. The aim is for the cops to chase the robbers through a city and to try to catch them before they escape to their hideaway.

The streets of the city run north to south and east to west. The cops have the advantage of a faster car, but they must obey the traffic laws. At intersections, they may only go ahead or turn left. No right turns or U-turns are allowed. The robbers ignore all traffic laws so, although their car is slower than the cops' car, they are better able to manoeuvre.

You can choose the speed of the game. At its fastest, it is a hair-raising chase through the streets. Played more slowly, it becomes an interesting game of strategy.

How to play

The player who is taking the part of the robbers sits on the left, the other player on the right. When asked to select the speed of the game, key in any number between '1' (the slowest) and '10' (the fastest). Then press RET. The screen displays the street grid. The intersection which is displayed in yellow is the location of the robber's hideaway. This is randomly placed in the central region of the top half of the screen. The cops' car is blue, and appears at the bottom right corner. The robbers' car is red and is at the bottom left corner. Both cars are pointing north (to top of screen) to begin with.

When you are ready, press key 'G' and the game starts. Moves are paced by the computer. The pacing is indicated by alternate tones, like a police siren. It is the robbers' turn to move when the higher note is sounding. The robbers have first move. A turn lasts for as long as the note is sounding. If you do not make any move during

that turn, your car is automatically moved one square forward. It is part of the game to remember the direction in which your car is facing.

Moves other than one square forward are made by pressing a key:

Robbers	A	turn left and move one square
	S	turn right and move one square
	Z	U-turn and move one square
	'space bar'	stop
Cops	F6	two squares forward
	F7	turn left and move one square
	F8	stop

The effect of these keys lasts only for one turn. You can see that the cops have the option of moving one square forward (no key-press) or two squares forward (press function key F6).

Now the chase begins. The robbers must get to their hideway before they are caught by the cops. To catch the robbers, the cops must come up from behind them and then move on to the same square. Note that both cars must be travelling in the same direction. If the cars are travelling in different directions, there is a head-on crash and neither player wins. A skilful cop may be able to avoid a crash by moving two squares forward, safely by-passing (jumping over) the robbers' car.

The game ends with escape (robbers win), capture (cops win) or collision (a draw). The final display tells you the number of moves for which the game lasted, excluding all moves when the robbers were in the country. The display also shows the greatest and least number of moves in the current series of games. You can score by seeing who is the robber able to evade capture for the greatest number of moves, or who is the cop who makes a capture in the fewest moves. Players take alternate turns at being cops or robbers.

Winning tactics

The balance between the speed advantage of the cops and the freedom of movement of the robbers makes this an interesting game. The cops should avoid having to change direction to the right when they are near the edge of the city. Turning right involves repeated left turns and this may take the car beyond the city limits, where it is all too easy to get lost.

Moving off the screen gives the robbers a chance to escape from view and to make their way stealthily to a point just outside the edge of town which is close to their hideout. Then a quick dash takes them safely home. The danger of this strategy is that a false move may send the car off in the wrong direction. The robbers gain no advantage by deliberately getting themselves lost, for moves are not counted while they are off the board. The cops must avoid becoming lost in the country at all costs, for this leaves the robbers free to make their way to their hideout without opposition.

Keying in

There is a space between the quotes (" ") in lines 200, 250 and 590, but no space in lines 180 and 360.

Program design

20–30	setting variables, etc.
40–50	inputting speed
60–130	displaying street, hideout and cars
140	waiting to start
150	checking if robbers have been off screen for too long
160–170	preparing for robbers' move
180–200	timed wait for robbers' move
210	clear robbers' car from screen
220–250	analysing robbers' input
260–330	making robbers' move and displaying it
340–350	preparing for cops' move
360–380	timed wait for cops' move
390	clear cops' car from screen
400–430	analysing cops' input
440–510	making cops' move and displaying it
520	"robbers escape" message
530–570	calculating and displaying moves
580–590	inviting a repeat game
600–610	"robbers captured" message
620–630	"head-on crash" message
1000–1020	subroutine for preventing RD (robbers' direction) from exceeding 3

1100–1110 subroutine for preventing CD (cops' direction) from exceeding 3
1200 subroutine to display robbers' car
1300 subroutine to display cops' car
1400 subroutine to clear robbers' car
1500 subroutine to clear cops' car

The program

```
10 REM ** COPS AND ROBBERS **
20 GENPAT 0,90,231,231,231,0,0,231,231,2
31: VS 4: CLS : COLOUR 4,3: LET MM=0: LE
T LM=5000
30 LET M=0: LET MC=0
40 VS 5: INK 15: PAPER 5: CLS : INPUT "S
PEED? (1-10) ";A$
50 LET L=11-VAL(A$): IF L<1 OR L>10 THEN
  GOTO 40
60 VS 4: PAPER 15: INK 1: CLS
70 FOR J=1 TO 768: PRINT CHR$(90);: NEXT

80 LET HX=INT(RND*22)+5
90 LET HY=INT(RND*8)+2
100 CSR HX,HY: INK 10: PRINT CHR$(90)
110 LET RM=0: LET RX=2: LET RY=23: LET R
D=0
120 LET CM=0: LET CX=30: LET CY=23: LET
CD=0
130 GOSUB 1200: GOSUB 1300
140 LET A$=INKEY$: IF A$<>"G" THEN  GOTO
 140
150 IF MC=31 THEN  LET MC=0: SOUND 0,0,0
: GOTO 110
160 LET J=0: LET M=M+1
170 SOUND 0,300,15
180 LET A$=INKEY$: IF A$="" AND J<20*L T
HEN  LET J=J+1: GOTO 180
190 IF J=20*L THEN  LET RM=1: GOTO 210
200 IF A$<>"A" AND A$<>"Z" AND A$<>"S" A
ND A$<>" " THEN  GOTO 180
210 IF RX>=0 AND RX<32 AND RY>=0 AND RY<
24 THEN  GOSUB 1400
220 IF A$="A" THEN  LET RM=1: LET RD=RD+
```

```
1: GOSUB 1000
230 IF A$="Z" THEN  LET RM=1: LET RD=RD+
2: GOSUB 1000
240 IF A$="S" THEN  LET RM=1: LET RD=RD-
1: GOSUB 1000
250 IF A$=" " THEN  LET RM=0
260 IF RD=0 THEN  LET RY=RY-RM
270 IF RD=1 THEN  LET RX=RX-RM
280 IF RD=2 THEN  LET RY=RY+RM
290 IF RD=3 THEN  LET RX=RX+RM
300 IF RX=HX AND RY=HY THEN  GOTO 520
310 IF RX=CX AND RY=CY AND CD<>RD THEN
GOTO 620
320 IF RX<0 OR RX>31 OR RY<0 OR RY>23 TH
EN  LET M=M+1: LET MC=MC+1: GOTO 340
330 GOSUB 1200
340 LET J=0
350 SOUND 0,400,15
360 LET A$=INKEY$: IF A$="" AND J<20*L T
HEN  LET J=J+1: GOTO 360
370 IF J=20*L THEN  LET CM=1: GOTO 390
380 IF ASC(A$)<141 AND ASC(A$)>143 THEN
 GOTO 360
390 IF CX>=0 AND CX<32 AND CY>=0 AND CY<
24 THEN  GOSUB 1500
400 IF CX=HX AND CY=HY THEN  INK 10: CSR
 HX,HY: PRINT CHR$(90)
410 IF A$=CHR$(133) THEN  LET CM=2: GOSU
B 1100
420 IF A$=CHR$(134) THEN  LET CM=1: LET
CD=CD+1: GOSUB 1100
430 IF A$=CHR$(135) THEN  LET CM=0
440 IF CD=0 THEN  LET CY=CY-CM
450 IF CD=1 THEN  LET CX=CX-CM
460 IF CD=2 THEN  LET CY=CY+CM
470 IF CD=3 THEN  LET CX=CX+CM
480 IF RX=CX AND RY=CY AND RD=CD THEN  G
OTO 600
490 IF RX=CX AND RY=CY AND RD<>CD THEN
GOTO 620
500 IF CX<0 OR CX>31 OR CY<0 OR CY>23 TH
EN  GOTO 150
510 GOSUB 1300: GOTO 150
520 SOUND 0,0,0: VS 5: PAPER 6: INK 15:
CLS : PRINT : PRINT "ROBBERS ESCAPE"
```

```
530 IF MM<M THEN  LET MM=M
540 IF LM>M THEN  LET LM=M
550 PRINT : PRINT "The game lasted ";M;"
 moves"
560 PRINT : PRINT "Longest game is ";MM;
" moves"
570 PRINT : PRINT "Shortest game is ";LM
;" moves"
580 CSR 6,20: PRINT "Press space bar to
continue"
590 IF INKEY$<>" " THEN  GOTO 590 ELSE
GOTO 30
600 SOUND 0,0,0: VS 4: PAPER 5: INK 15:
CLS : PRINT : PRINT "ROBBERS CAPTURED"
610 GOTO 530
620 SOUND 0,0,0: VS 5: PAPER 1: INK 15:
CLS : PRINT "COPS AND ROBBERS CRASHED HE
AD-ON"
630 GOTO 530
1000 IF RD>3 THEN  LET RD=RD-4
1010 IF RD<0 THEN  LET RD=RD+4
1020 RETURN
1100 IF CD>3 THEN  LET CD=CD-4
1110 RETURN
1200 CSR RX,RY: INK 9: PRINT CHR$(90);:
RETURN
1300 CSR CX,CY: INK 4: PRINT CHR$(90);:
RETURN
1400 CSR RX,RY: INK 1: PRINT CHR$(90);:
RETURN
1500 CSR CX,CY: INK 1: PRINT CHR$(90);:
RETURN
```

Variations

You could add sound-effects, such as squealing tyres when the cars turn corners, gun-shots when the cars are close, and an enormous crash when the cars collide. Lines 110 and 120 could be altered to position the cars differently (or at random) at the beginning of the game. CX, CY, RX and RY are the variables concerned. The limit on the number of moves (MC) allowed to the robbers off screen can be changed at line 150.

5
Mind Over Electrons

Is it possible to communicate with a computer directly, without using the keyboard or a joystick or some other piece of hardware? Can your mind, unaided by material connections, influence what goes on inside the Memotech? Most people would say that this is complete nonsense, but there are others who equally strongly assert that it is possible. Here is a program to help put the idea to the test.

How to play

The program is based on choosing one from among twelve graphics symbols, such as a 'heart', a 'cross' or a 'circle'. These are displayed across the top of the screen. Before this happens, the computer asks if it is to choose. If you answer 'Y', it chooses a symbol and will then ask you to guess which one it has chosen. Can you guess correctly more often than you should by pure chance? If you can, you are either able to detect what has happened inside the computing circuits, or can predict what symbol will appear on the screen before it has been displayed.

If you answer 'N' to the question, it is you who does the choosing. You select a symbol and then try to make the Memotech select the same symbol. If you concentrate hard while it is choosing, you may be able to force it to choose the one you have decided on.

As soon as you have answered 'Y' or 'N', the row of symbols is displayed. A pointer (↑) indicates the one to be selected. You can move this arrow along from symbol to symbol by pressing the function key F8. Assuming it is the computer which is choosing, try to guess which symbol it has chosen, and move the arrow to point to that symbol. The computer has already chosen a symbol and it is displayed on the screen below the message 'Trial no. 1'. You cannot see it because it is coloured grey and the background is grey too.

When you have made your guess, press key F4. The colour of the symbol is now changed to black, so that you can see it. If you have guessed correctly, the computer adds 1 to your score.

After a short pause, the symbol disappears and the message changes to 'Trial no. 2'. Choose another symbol if you like, or stay with the same one; then press F5. There are 120 trials altogether, out of which you would expect to get 10 correct by guesswork. If you can score more than 15, you will be doing very well.

After the 120th trial, your score is displayed. The computer also works out how likely it is that you could have got such a score just by chance. If the score is 13 or less this is not unusual, but a score as high as 15 normally occurs in only 1 in 10 (10%) of games. The chances of scoring 20 or more by guesswork are only 1 in 1000. If you score 20 or more it might be taken to show that you really can influence the computer's inner workings.

If you answer 'N', the order of operations is reversed. The computer waits for you to select a symbol. First you press K8 to move the arrow to the symbol you are selecting. Then, you press K4 to tell the computer that this is the selected symbol. The computer then selects and displays a symbol. Concentrate hard and you may be able to 'force' it to choose the same one as you have already chosen.

Keying in

Note the decimal point before the '1' in line 380. The character between the quotes in line 1020 is the exponentiation symbol (^).

Program design

20	initialising variables
30–90	generating the symbols
100–110	deciding who chooses
120–130	displaying symbols and the pointer
140–150	start of loop
160	the computer chooses a symbol at random; Q is the pattern number of the symbol
170	displaying the symbol invisibly
180–230	the user selects a symbol
240	the symbol made visible
250–260	score increased if symbols are the same

270–280	symbol made invisible again, ready for next trial
290–300	displaying the score
310–390	working out the probability and displaying it
400–430	inviting repeat play
1000–1030	subroutine to move the pointer

Points of interest

If you play this with the computer choosing first, the program runs in the order in which it is listed. If you play it with you choosing first, flag F is zero and several routines are taken out of order. It then becomes a terrible example to programmers, with GOTO's sending the computer jumping backward and forward through the listing in a most unbecoming manner!

The probabilities (or rather, percentages) worked out in lines 320 to 380 are based on a standard statistical test, usually known as the chi-squared test.

The program

```
10 REM ** MIND OVER ELECTRONS **
20 LET F=0: LET P=5: LET N=0
30 GENPAT 1,129,24,24,24,255,255,24,24,2
4: GENPAT 1,130,195,195,102,60,60,102,19
5,195
40 GENPAT 1,131,24,60,126,255,255,90,24,
60: GENPAT 1,132,0,102,255,255,255,126,6
0,24
50 GENPAT 1,133,0,8,28,62,127,62,28,8: G
ENPAT 1,134,0,24,24,231,231,24,24,60
60 GENPAT 1,135,60,126,255,255,255,255,1
26,60: GENPAT 1,136,60,102,195,129,129,1
95,102,60
70 GENPAT 1,137,24,24,24,24,24,24,24,24:
 GENPAT 1,138,0,0,0,255,255,0,0,0
80 GENPAT 1,139,102,102,102,102,102,102,
102,102: GENPAT 1,140,0,255,255,0,0,255,
255,0
90 VS 5: CLS : PRINT
100 INPUT "Computer chooses? (Y/N) ";A$
110 IF A$="Y" THEN  LET F=1
```

```
120 VS 4: COLOUR 0,14: COLOUR 1,1: COLOU
R 2,14: COLOUR 4,14
130 CLS : FOR J=1 TO 12: CSR 3+2*J,6: PR
INT CHR$(128+J): NEXT : CSR P,8: PRINT "
^"
140 FOR J=1 TO 120
150 INK 1: CSR 10,15: PRINT "Trial no. "
;J: IF F=0 THEN  GOTO 180
160 LET Q=INT(RND*12)+129
170 INK 0: CSR 15,19: PRINT CHR$(Q): IF
F=0 THEN  GOTO 240
180 LET A$=INKEY$
190 IF A$="" THEN  GOTO 180
200 IF A$=CHR$(131) AND F=0 THEN  GOTO 1
60
210 IF A$=CHR$(131) THEN  GOTO 240
220 IF A$=CHR$(135) THEN  GOSUB 1000
230 PAUSE 400: GOTO 180
240 INK 1: CSR 15,19: PRINT CHR$(Q)
250 IF Q=(P-5)/2+129 THEN  LET N=N+1
260 PAUSE 2000
270 INK 0: CSR 15,19: PRINT CHR$(Q)
280 NEXT
290 VS 5: PAPER 5: INK 15: CLS
300 CSR 2,4: PRINT "Score: ";N;" out of
100"
310 IF N<14 THEN  CSR 2,7: PRINT "This i
s not unusual": GOTO 400
320 IF N=14 THEN  LET N=25
330 IF N=15 THEN  LET N=10
340 IF N=16 THEN  LET N=5
350 IF N=17 THEN  LET N=2.5
360 IF N=18 THEN  LET N=1
370 IF N=19 OR N=20 THEN  LET N=.5
380 IF N>20 THEN  LET N=.1
390 CSR 2,7: PRINT "The probability is";
N;"%"
400 LET N=0
410 CSR 7,21: PRINT "Press space bar to
repeat"
420 IF INKEY$<>" " THEN  GOTO 420
430 PAPER 4: INK 15: GOTO 90
1000 INK 1: CSR P,8: PRINT " "
1010 LET P=P+2: IF P=29 THEN  LET P=5
1020 CSR P,8: PRINT "^"
1030 RETURN
```

Variations

Do not be tempted to reduce the number of trials. A hundred and twenty trials is the very minimum that could possibly show any real effect. You could reduce the number of symbols to, say, four. Line 130 would need amending. The first numbers in lines 310 to 380 should be altered to: 36,36/37, 38/39,40,41,42,44/45,46. By '36/37' we mean that line 320 should be 'IF N=36 OR N=37 THEN N=25'. Similarly for the other lines.

6
Nimble Thimble

This is a new version of the traditional game of Nim. It is a game for two players. Over the years, the game has been played in a variety of forms. It can be played with piles of stones on the mud floor of a straw hut, with piles of matches on the living-room table, or with counters on a playing board. This version uses the Memotech's sprites to play with coloured glass thimbles on a delicately embroidered cloth.

How to play

First, the screen displays the playing cloth, on which is marked a meandering embroidered path, leading to a 'home' at the bottom left-hand corner. Three thimbles, coloured yellow, green and purple, are placed at random on the path.

The computer tells you whose turn it is to play. The first person plays when the border of the screen has become green. When the border of the screen becomes purple, it is the second player's turn. When it is your turn, you must move any one (but only one) of the thimbles one step along the path in the direction of home. You make your move by pressing one of the three function keys, 'F1', 'F2' or 'F3'. These move the yellow thimble, the green thimble or the purple thimble, respectively. When you press a key, the computer moves the thimble to the next step on the path.

Two thimbles cannot occupy the same position and thimbles cannot jump over one another. The computer does not allow you to make these illegal moves. When a thimble is moved on to 'home', it is removed from the cloth.

The aim of the game is to be the last person to move a thimble on to 'home'. The colour of the winner is displayed at the end of the game.

To play again, press the space bar. The thimbles will reappear, but with new starting positions. It is fairer if players take turns to play first.

Winning tactics

This is a game in which it makes for greater interest if you develop your own tactics. If any more is said on this topic, there is a danger of literally, 'giving the game away'.

Keying in

Take care to type the correct numbers of spaces between pairs of quote marks. There are none on line 510, 1 on line 840 and 19 on line 850.

Program design

20–30	initialising arrays and sprite control
40–160	GENPAT statements for the sprites and user-defined graphics
170	setting display colours
180–340	displaying the cloth
350–430	calculating a table of screen coordinates
440–460	selecting random positions for the thimbles and rejecting the selection if two thimbles are on the same step
470–490	displaying the thimbles in starting positions
500–580	asking for input and checking that it is valid
590–610	selecting sequence of movements
620–650	move a thimble sprite
660–700	detecting a thimble moved to home and preparing for the next turn
710–750	move to 'home', made and registered
760–790	checking if last thimble
800–830	winning display
840–860	preparing for re-play
1000–1020	DATA giving sequences directions for MVSPR (line 630)

The program

```
10 REM ** NIMBLE THIMBLE **
20 DIM B(2,42),T(3,2)
30 CTLSPR 1,4: CTLSPR 2,3: CTLSPR 5,3: C
TLSPR 6,2
40 GENPAT 1,129,195,102,60,24,24,60,102,
195: GENPAT 1,130,255,132,136,144,168,19
7,130,132
50 GENPAT 1,131,255,33,17,9,21,163,65,33
: GENPAT 1,132,132,130,197,168,144,136,1
32,255
60 GENPAT 1,133,33,65,163,21,9,17,33,255
: GENPAT 1,134,255,128,128,128,134,137,1
36,132
70 GENPAT 1,135,255,1,1,1,97,145,17,33:
GENPAT 1,136,132,130,130,129,128,128,128
,255
80 GENPAT 1,137,33,65,65,129,1,1,1,255:
GENPAT 1,138,0,0,0,0,56,68,130,129
90 GENPAT 1,139,0,0,0,0,4,2,1,1: GENPAT
1,140,128,128,64,32,0,0,0,0
100 GENPAT 1,141,129,65,34,28,0,0,0,0: G
ENPAT 1,142,3,4,8,0,0,0,0,0
110 GENPAT 1,143,192,32,16,16,16,32,64,1
28: GENPAT 1,144,1,2,4,8,8,8,4,3
120 GENPAT 1,145,0,0,0,0,0,16,32,192
130 GENPAT 4,1,3,15,31,31,21,51,12,51
140 GENPAT 5,1,12,51,108,123,127,63,15,3
150 GENPAT 6,1,192,240,248,240,204,48,20
4,48
160 GENPAT 7,1,204,48,206,62,254,252,240
,192
170 VS 4: COLOUR 0,9: COLOUR 1,6: COLOUR
 2,9: COLOUR 4,11: CLS
180 FOR J=1 TO 22
190 CSR 1,J: PRINT CHR$(129): CSR 30,J:
PRINT CHR$(129);
200 NEXT
210 FOR J=1 TO 6: FOR K=0 TO 2 STEP 2
220 CSR 3,J*4-3+K/2
230 FOR L=1 TO 6
240 PRINT CHR$(134+K);CHR$(135+K);CHR$(1
38+K);CHR$(139+K);
250 NEXT : PRINT CHR$(134+K);CHR$(135+K)
```

```
260 NEXT : NEXT
270 FOR J=1 TO 3: FOR K=0 TO 2 STEP 2
280 CSR 27,J*8-5+K/2: PRINT CHR$(142+K);
CHR$(143+K)
290 NEXT : NEXT
300 FOR J=1 TO 2: FOR K=0 TO 2 STEP 2
310 CSR 3,J*8-1+K/2: PRINT CHR$(142+K);C
HR$(143+K)
320 NEXT : NEXT
330 CSR 3,21: PRINT CHR$(130);CHR$(131)
340 CSR 3,22: PRINT CHR$(132);CHR$(133)
350 FOR J=0 TO 28 STEP 14: FOR K=1 TO 7
360 LET B(1,J+K)=32+(K-1)*32
370 NEXT : NEXT
380 FOR J=7 TO 35 STEP 14: FOR K=1 TO 7
390 LET B(1,J+K)=224-(K-1)*32
400 NEXT : NEXT
410 FOR J=0 TO 35 STEP 7: FOR K=1 TO 7
420 LET B(2,J+K)=176-J/7*32
430 NEXT : NEXT
440 FOR J=1 TO 3
450 LET T(J,1)=INT(RND*29)+1: LET T(J,2)
=1
460 NEXT : IF T(1,1)=T(2,1) OR T(2,1)=T(
3,1) OR T(3,1)=T(1,1) THEN  GOTO 440
470 FOR J=1 TO 3
480 SPRITE J,1,B(1,T(J,1)),B(2,T(J,1)),0
,0,10+J
490 NEXT
500 COLOUR 4,3: LET P=1
510 LET A$=INKEY$: IF A$="" THEN  GOTO 5
10
520 LET TP=ASC(A$)-127: IF TP<1 OR TP>3
THEN  GOTO 510
530 IF T(TP,2)=0 THEN  GOTO 510
540 LET TN=T(TP,1)+1: LET F=0
550 FOR J=1 TO 3
560 IF T(J,1)=TN AND J<>TP THEN  LET F=1
570 NEXT
580 IF F=1 THEN  GOTO 510
590 LET TT=T(TP,1): IF INT(TT/7)=TT/7 TH
EN  RESTORE 1000: GOTO 620
600 IF TT>0 AND TT<7 OR TT>14 AND TT<21
OR TT>28 AND TT<35 THEN  RESTORE 1010: G
OTO 620
```

```
610 RESTORE 1020
620 FOR J=1 TO 10
630 READ M: MVSPR 1,TP,M
640 PAUSE 100
650 NEXT
660 LET T(TP,1)=TN
670 IF TN=42 THEN  GOTO 710
680 IF P=2 THEN  GOTO 500
690 COLOUR 4,13: LET P=2
700 GOTO 510
710 LET T(TP,1)=0: LET T(TP,2)=0
720 FOR J=1 TO 20
730 ADJSPR 1,TP,0: PAUSE 200
740 ADJSPR 1,TP,10+TP: PAUSE 200
750 NEXT : ADJSPR 1,TP,0
760 LET F=1: FOR J=1 TO 3
770 IF T(J,2)=1 THEN  LET F=0
780 NEXT
790 IF F=0 THEN  GOTO 680
800 CSR 7,7
810 IF P=1 THEN  COLOUR 1,12: PRINT "Gre
en ";
820 IF P=2 THEN  COLOUR 1,13: PRINT "Pur
ple ";
830 PRINT "player wins"
840 IF INKEY$<>" " THEN  GOTO 840
850 CSR 7,7: PRINT "
 "
860 GOTO 440
999 GOTO 999
1000 DATA 6,1,1,2,2,2,2,3,3,2
1010 DATA 6,7,7,0,0,0,0,1,1,2
1020 DATA 6,5,5,4,4,4,4,3,3,2
```

Variations

Let the player who moves the last thimble 'home' be the loser. It is not too difficult to analyse this game and find a winning strategy. Having done this, there is plenty of room in RAM for you to program the Memotech to play against you – and it will win!

7
Sand Castle

This is a game for the younger members of the family. If it is a rainy day and you cannot get to the beach, you can build a sand castle with the computer. Of course, you must finish the castle before the tide comes in.

How to play

When the computer asks "Speed?", press one of the keys '1' to '5'. If you have not played this game before, press '1'. Pressing other numbers makes the computer work more quickly. If you press '5', it goes very fast. You will need plenty of practice to keep up with it at speed 5. When you have chosen the speed, press the RET key.

Now you will see the beach and a child on the beach. This is you, ready to make the sand castle. Look in the sky above the child's head. Very soon, a number appears there. As quickly as you can, find the key which has the same number and press it, once. If you have been quick enough, the child digs up a spadeful of sand and starts to make the sand castle. The amount of sand dug up depends on how big the number is. The bigger the number, the more sand is dug for the castle. But the bigger the number, the quicker you have to be to press the key in time.

As soon as the child has dug up the sand and started to build the castle, the number disappears from the sky. Then you hear another crashing wave, as the tide comes in a bit more. After a short time, another number appears in the sky at the same place. It may be the same number as before, though it is more likely to be different. Find the key which has this number on it, and press it once. If you are in time, the child digs more sand and adds it to the pile.

Each time a number appears and you press the right key, the child digs more sand and the castle gets bigger. Each time the number

appears the tide comes in a little bit more. This game is a race to finish the castle before the tide gets to it.

If many big numbers (such as '7', '8' and '9') appear in the sky, and if you are quick when you see them, the sand castle will soon be finished. Then a brightly coloured flag appears on top of the castle and the child stands proudly beside it. This shows you that you have won.

If the tide reaches the castle before it is finished, it is washed away and the game ends. Press the space bar to start the game again and try to beat the tide. To stop the computer, press 'S'.

Keyboard

The program works with either the number keys along the top of the keyboard or the key-pad to the right of the main key-board. Children may find the keys at the top of the keyboard difficult to reach and too widely spaced out. Also it is illogical that the '0' is to the right of the '9' instead of being to the left of the '1'. For these reasons it is recommended that the key-pad should be used.

Winning tactics

The bigger the number, the more sand is added to the castle. Try hard to press the right key quickly when one of the bigger numbers appears in the sky. Take care not to press the wrong key by mistake.

Keying in

This is a program for parents, or an elder brother or sister, to key in for the younger family members. It is quite a lot of work, but the animated graphics and sound effects provide compensation for the effort required. The POKEs in line 20 enable the numeric key-pad to the right of the main keyboard. They also disable key 9 (BRK) so that it no longer acts as a break key. This means that, if there is a mistake in your program and the computer 'hangs' you will not be able to press BRK to stop the computer. The only way out will be to press the two reset keys on either side of the space bar, or to switch off

the computer. For this reason you are strongly advised to save the program on tape *before* you run it. Note that there are *three* spaces between the quotes in lines 420 to 440.

Program design

20	enabling numeric key-pad and initialising control of sprites
30–180	GENPAT statements for user-defined characters
190–200	initialising variables and requesting speed input
210–290	the initial display routine
300	displaying a random number, in white
310–340	waiting a limited length of time for input, then clear number
350	to subroutine 1200, if answer is correct
360	to end of game, if castle is complete
370	go to subroutine 1300 for next wave
380	to end of game if waves have reached the castle
390	return for next number
400–460	displaying the flag, with child standing beside castle
470–490	inviting replay or end of program
500–530	washing away the castle
1000–1020	subroutine to print N blocks of solid colour
1100–1150	displaying child digging
1200–1290	displaying child throwing sand on castle, then add sand to castle
1300–1360	wave display, with crashing sound effect
2000–2020	DATA for displaying clouds, etc.

Points of interest

This program has 28 user-defined graphics. Two of these are multicolour designs, used for the flag. There are also 6 sprites, but these are never noticeable on the screen. They are used for building up and washing away the castle. The sprites are all square and coloured light blue, the same as the sky. Line 280 places these in 2 rows of 3 sprites, so that they cover the region where the castle is to be. The sprites cannot be seen against the blue sky. Then the castle is displayed (line 290) behind the 'curtain' of sprites, so is invisible. The

'curtain' of sprites is raised (lines 1270 to 1290) as the castle is built. It is lowered (lines 500 to 520) when the castle is washed away.

The program

```
10 REM ** SANDCASTLE **
20 POKE 64145,132: POKE 64862,13: CTLSPR
 1,1: CTLSPR 2,6: CTLSPR 6,1
30 GENPAT 0,35,255,255,255,255,231,219,1
89,126: GENPAT 0,91,231,219,189,126,231,
219,189,126: GENPAT 0,93,225,219,183,119
,251,219,173,118
40 GENPAT 0,35,255,255,255,255,231,219,1
89,126: GENPAT 0,91,231,219,189,126,231,
219,189,126: GENPAT 0,93,225,219,183,119
,251,219,173,118
50 GENPAT 1,129,0,0,28,62,62,62,31,3: GE
NPAT 1,130,0,0,0,0,0,0,248,4
60 GENPAT 1,131,2,2,2,2,2,3,2,5: GENPAT
1,132,132,68,100,104,176,32,96,160
70 GENPAT 1,133,5,6,4,12,116,244,244,108
: GENPAT 1,134,32,32,32,32,32,32,32,48
80 GENPAT 1,135,24,60,60,126,126,62,28,4
: GENPAT 1,136,0,0,0,0,0,0,56,196
90 GENPAT 1,137,0,0,0,96,240,120,240,104
: GENPAT 1,138,7,28,102,130,66,65,33,33
100 GENPAT 1,139,4,4,4,4,4,4,24,96: GENP
AT 1,140,0,0,0,4,31,126,252,48
110 GENPAT 1,141,17,23,25,97,130,4,8,16:
 GENPAT 1,142,120,0,128,128,64,64,64,32
120 GENPAT 1,143,16,8,8,8,4,4,4,12: GENP
AT 1,144,48,48,48,48,48,48,48,48
130 GENPAT 1,145,0,0,1,3,23,39,39,39: GE
NPAT 1,146,0,0,128,192,224,224,224,224
140 GENPAT 0,33,35,32,16,15,0,0,0,0: GEN
PAT 0,37,192,128,128,240,143,129,129,129
150 GENPAT 0,38,25,21,19,16,16,16,16,48:
 GENPAT 0,39,130,130,130,130,66,34,18,18
160 GENPAT 0,64,18,18,18,18,19,19,19,51:
 GENPAT 1,152,31,31,31,31,31,31,31,31
170 GENPAT 1,153,63,63,63,63,63,63,63,63
180 GENPAT 2,152,149,149,165,165,37,37,2
13,213: GENPAT 2,153,41,41,218,218,146,1
```

```
46,173,173: GENPAT 3,1,255,255,255,255,2
55,255,255,255
190 LET R=0: LET SC=22: VS 5: CLS : CSR
3,5: INPUT "Speed? (1-5) ";L$
200 LET L=6-VAL(L$): IF L<1 OR L>5 THEN
 GOTO 190
210 VS 4: COLOUR 0,15: COLOUR 2,5: COLOU
R 4,8: CLS
220 FOR K=1 TO 9: READ X,Y,Z
230 CSR X,Y: LET N=Z: GOSUB 1000: NEXT
240 COLOUR 0,10: FOR K=1 TO 7
250 CSR 15-2*K,14+K: LET N=17+2*K: GOSUB
 1000
260 NEXT
270 COLOUR 0,15: COLOUR 1,4: FOR K=1 TO
32: PRINT "[";: NEXT
280 SPRITE 1,1,168,95,0,0,5: SPRITE 2,1,
184,95,0,0,5: SPRITE 3,1,200,95,0,0,5: S
PRITE 4,1,168,79,0,0,5: SPRITE 5,1,184,7
9,0,0,5: SPRITE 6,1,200,79,0,0,5
290 COLOUR 0,10: FOR K=1 TO 3: READ X,Y,
Z: CSR X,Y: LET N=Z: GOSUB 1000: NEXT
300 GOSUB 1100: COLOUR 0,5: COLOUR 1,15:
 CSR 25,3: LET N=INT(RND*10): PRINT N
310 LET J=0
320 LET F=0
330 LET A$=INKEY$: IF INKEY$="" THEN  LE
T J=J+1: LET F=1: IF J<(1000*L)/(N+10) T
HEN  GOTO 320
340 CSR 26,3: PRINT " "
350 IF VAL(A$)=N AND F=0 THEN  GOSUB 120
0
360 IF R>=24 THEN  GOTO 400
370 GOSUB 1300
380 IF SC=15 THEN  GOTO 500
390 GOTO 300
400 FOR J=1 TO 6: ADJSPR 1,J,0: NEXT
410 COLOUR 1,15: CSR 25,11: PRINT CHR$(1
45);CHR$(146);"    "
420 CSR 25,12: PRINT CHR$(33);CHR$(37);"
   "
430 CSR 25,13: PRINT CHR$(38);CHR$(39);"
   "
440 CSR 26,14: PRINT CHR$(64);"    "
450 COLOUR 1,1: CSR 23,11: PRINT CHR$(14
```

```
4)
460 CSR 23,10: PRINT CHR$(152);CHR$(153)
470 LET A$=INKEY$: IF A$<>"S" AND A$<>"
" THEN  GOTO 470
480 IF A$="S" THEN  POKE 64862,15: POKE
64145,160: STOP
490 CLS : RESTORE 2000: GOTO 190
500 FOR J=1 TO R: FOR S=1 TO 6
510 MVSPR 1,S,2
520 PAUSE 100: NEXT : NEXT
530 GOTO 470
1000 FOR J=1 TO N
1010 PRINT " ";
1020 NEXT : RETURN
1100 COLOUR 0,5: COLOUR 1,15
1110 CSR 28,11: PRINT "   "
1120 CSR 27,12: PRINT " ";CHR$(129);CHR$
(130)
1130 CSR 27,13: PRINT " ";CHR$(131);CHR$
(132)
1140 CSR 28,14: PRINT CHR$(133);CHR$(134
)
1150 RETURN
1200 COLOUR 0,5: COLOUR 1,15
1210 CSR 28,11: PRINT CHR$(135);CHR$(136
)
1220 CSR 27,12: COLOUR 1,10: PRINT CHR$(
137);: COLOUR 1,15: PRINT CHR$(138);CHR$
(139)
1230 CSR 27,13: PRINT CHR$(140);CHR$(141
);CHR$(142)
1240 CSR 28,14: PRINT CHR$(143)
1250 PAUSE 500
1260 GOSUB 1100
1270 LET AD=INT((N+3)/3): LET R=R+AD
1280 FOR M=1 TO AD: FOR S=6 TO 1 STEP -1
: MVSPR 1,S,6: NEXT : NEXT
1290 RETURN
1300 LET SC=SC-.5: LET W=91: IF SC>INT(S
C) THEN  LET W=35
1310 SOUND 3,4,15
1320 COLOUR 0,15: COLOUR 1,4
1330 FOR J=1 TO 32: CSR J-1,INT(SC)
1340 PRINT CHR$(W);CHR$(93): PAUSE 200:
NEXT
```

```
1350 SOUND 3,0,0
1360 RETURN
2000 DATA 13,2,3,9,3,8,8,4,9,9,5,9,13,6,
3
2010 DATA  3,8,3,2,9,6,2,10,7,4,11,5
2020 DATA   23,12,1,22,13,3,21,14,5
```

Variations

This program originated as a game but there are obvious educational applications. Even as it stands, it could help teach a child to match numerals. A simple extension would be to make it display letters of the alphabet too. Or it could easily be adapted to display a mathematical expression for evaluation, such as:

3 + 6

Sand is added to the castle if the child keys in the correct answer.

The range of delays provided by choosing speed at line 190 may not give enough time for a child to work out the answer. In this event, the value '1000' in line 330 can be increased. To make the castle build up more quickly, and so increase the chances of winning, alter the first '3' in line 1270 to '3.5' or '4'.

8
Bombing Run

There is a bright moon and you are on a mission to strike at enemy ground targets. Their territory is heavily defended by anti-aircraft weapons. Your mission continues until you are eventually shot down but, before that happens, you must hit as many targets as possible. Dropping the bombs at just the right moment is difficult enough. To have to weave in and out of bursts of gunfire as well is liable to make you lose both your concentration and your aim.

This game makes use of all 32 of the Memotech's sprites, which provides plenty of movement on the screen. There is a realistic effect of perspective as the enemy country-side passes below the bomber.

How to play

You are first asked at which level you wish to play. The level you choose decides how heavy the anti-aircraft fire will be. At level 1 it is sporadic; you will have ample time to aim your bombs and to take evasive action against most of the bursting shells. At level 5 you will receive an intense barrage of fire, and everything moves faster. You are not likely to survive long unless you are a particularly skilful pilot.

Press 'RETURN'. The screen clears to display the moonlit evening sky. The bomber appears on the left-hand side of the screen. You can gauge your speed by watching the trees and buildings rush past on the ground below. In the sky above you see the flashes and smoke clouds from bursting shells. If you fly into one of these, you are shot down. You change height by using two of the function keys on the right-hand side of the keyboard. Pressing key F7 makes the bomber climb, while pressing key F8 makes it dive. If you climb to the top of the screen, you avoid the gunfire, but bombing is more

difficult from that height. If you go too low, you will crash into the trees or buildings.

When the run first begins you hear a rapid bleeping. This is a sign that no target is in range. Soon the bleeping changes in tone, indicating that you are on a bombing run. At the same time a number appears at the top left corner of the screen. The number is '9' to begin with and counts down to zero, in time with the bleeps. As the number changes to zero, the target comes below the bomber, close to the spot where the bombs will land. The target is a large red factory building.

Your task is to release the bomb at the correct moment, *before* the target is directly below the bomber, so that the bomb falls exactly on the target. To release a bomb, hold the space bar down for about half a second. The higher the bomber is above ground when the bomb is released, the longer it takes to fall. You will hear the whine as it falls and see the flash just below the bomber when it hits the ground. A giant puff of smoke rises from the explosion, and you hear the sound of the explosion at the same time. If it hits the target, the building immediately crumbles to a heap of bricks. To make this happen, your bomb must fall on the left-hand part of the building. At higher playing levels your aim needs to be more precise.

Sooner or later you are bound to be shot down. When this happens, the screen clears and you are told the results of your mission.

Winning tactics

Flying high is one way of avoiding the guns, but it is harder to hit the target from a great height. Not only do they take longer to reach the ground, but air resistance makes them fall further behind the bomber than they would normally do. A low-level attack makes aiming easier, but a quick swerve to avoid a bursting shell may crash you into a tree instead.

You can drop bombs at any time. It is worth while trying a few practice runs, aiming at the passing buildings or trees. You can then gauge at any given altitude of the bomber how far ahead the target should be when you release the bomb. As you become more proficient, try to put every bomb where it belongs – on an enemy target!

Keying in

The dozens of GENPAT and SPRITE statements are due to the program using every one of the Memotech's sprites. Take care to get all the values correct, otherwise strange things may happen on the screen. Note that the sprite generator is not able to handle more than four sprites at a time on any one row of the screen. When buildings or parts of the bomber disappear briefly, it is not your fault or that of the program. Likewise, the high speed on the program sometimes results in the crown of the large tree becoming partly detached from its trunk. Similarly, the tower of the church may perch lopsidedly on the lower part of the building!

Program design

20–130	initialising arrays and the patterns used for the sprites, the most often used variables
140–160	initializing variables and requesting level of play required
170–350	setting up the sprites and the initial screen display
360	ready for the main loop
370–560	the main program loop
570	go back to begin a new loop
580–610	'shot down' routine with mission details displayed
620–630	inviting re-play
640–650	clear array and return for next game
660–680	'crash' routine
2000–2020	subroutine for producing random numbers for placing sprites
3000–3030	subroutine for moving 'shell-burst' sprites and detecting if any have hit the bomber
4000–4100	subroutine for producing visual effects and the sound of the exploding bomb; detects if the target is hit and, if so, makes the building crumble and increments the score

Points of interest

The key to the interest of this game is its speed. It illustrates the high speed with which sprites may be processed from a BASIC program.

To make the operation of the main loop as rapid as possible, yet keeping it in BASIC, frequently-used constants have been assigned to variables. These have been used instead of the constants in the main loop of the program and in the often repeated subroutines beginning at lines 3000 and 4000. Examples are Q and R, the addresses of two memory locations that have to be PEEKed frequently to find out when the sound of the falling bomb has ceased (line 4000). This subroutine is called several times during the main loop, so maximum speed is essential. Two other examples of constant values assigned to variables are U for unity (=1) and Z for zero. It is quicker for the micro to refer to its table of variables to find '1' and '0' and other constants than to convert them from their floating-point equivalents every time they are used.

The sprites which represent the buildings and trees are moved automatically by the sprite generator once they have been set in motion at lines 200 to 340. The sprites which represent the shell-bursts, bomber, target and exploding bomb are under direct program control, using ADJSPR and MVSPR statements.

The program

```
10 REM ** BOMBING RUN **
20 DIM R(12),SX(24),SY(24): LET Q=64092:
 LET R=64096: GENPAT 3,1,56,60,122,254,2
55,127,60,123: GENPAT 3,2,58,28,24,56,24
,24,60,126
30 GENPAT 3,3,8,60,126,255,169,255,173,2
53: GENPAT 3,4,28,60,126,126,60,24,24,60
40 GENPAT 3,5,0,24,60,126,60,24,24,60: G
ENPAT 3,6,64,224,64,224,160,224,224,234
50 GENPAT 3,7,254,213,213,213,213,255,19
1,191: GENPAT 3,8,24,42,40,122,217,40,11
6,10
60 GENPAT 3,9,254,146,146,255,146,255,16
6,231: GENPAT 3,10,36,36,54,255,73,255,7
3,255
70 GENPAT 3,11,0,128,192,192,224,255,255
,0: GENPAT 3,12,3,3,3,3,67,255,255,0
80 GENPAT 3,13,0,128,192,224,240,254,255
,2: GENPAT 3,14,41,170,92,61,30,157,93,7
8
90 GENPAT 3,15,44,157,223,62,156,93,62,2
```

```
55: GENPAT 3,16,0,0,60,30,62,28,8,28
100 GENPAT 3,17,0,0,34,119,93,127,85,119
: GENPAT 3,18,0,0,16,56,84,254,182,246
110 GENPAT 3,19,0,0,32,118,119,119,34,34
: GENPAT 3,20,0,0,0,22,127,127,50,34: GE
NPAT 3,21,0,0,66,255,165,255,173,189
120 GENPAT 3,22,0,0,0,24,126,90,90,126:
GENPAT 3,23,0,0,0,0,16,118,255,82
130 GENPAT 1,129,0,0,0,0,0,3,208,255: GE
NPAT 1,130,16,48,96,96,96,48,59,28: LET
BY=90
140 LET B=0: LET BY=90: LET FB=0: LET S=
0: LET TX=0: LET Z=0: LET U=1: LET T=2:
LET E=8: LET TE=10: LET F=15: LET ST=72:
 LET H=100
150 VS 5: CLS : PRINT "LEVEL?  (1-5)"
160 LET L=ASC(INKEY$)-48: IF L<1 OR L>5
THEN  GOTO 160
170 VS 4: COLOUR 0,4: COLOUR 1,1: COLOUR
 2,4: COLOUR 4,1: CLS
180 GOSUB 2000
190 CTLSPR 0,10-L: CTLSPR 2,32: CTLSPR 3
,32: CTLSPR 4,32: CTLSPR 5,32: CTLSPR 6,
1
200 SPRITE 1,1,R(1),23,-128,0,3: SPRITE
2,2,R(1),7,-128,0,6
210 SPRITE 3,3,R(2),7,-120,0,9: SPRITE 4
,4,R(3),7,-115,0,12
220 SPRITE 5,5,R(4),7,-110,0,2: SPRITE 6
,6,R(5),23,-105,0,14
230 SPRITE 7,7,R(5),7,-105,0,14: SPRITE
8,8,280,100,0,0,15
240 SPRITE 9,8,280,100,0,0,15: SPRITE 10
,8,280,100,0,0,15
250 SPRITE 11,8,280,100,0,0,15: SPRITE 1
2,8,280,100,0,0,15
260 SPRITE 13,9,R(6)+100,7,0,0,6: SPRITE
 14,10,R(6)+116,7,0,0,6: LET TX=R(6)+100
: LET CD=9
270 SPRITE 15,11,72,BY,0,0,1: SPRITE 16,
12,88,BY,0,0,1: SPRITE 17,13,104,BY,0,0,
1
280 SPRITE 18,14,280,23,0,0,14: SPRITE 1
9,15,280,7,0,0,10
290 SPRITE 20,8,280,100,0,0,15: SPRITE 2
```

```
1,8,280,100,0,0,15
300 SPRITE 22,8,280,100,0,0,15: SPRITE 2
3,8,280,100,0,0,15
310 SPRITE 24,8,280,100,0,0,15: SPRITE 2
5,16,R(9),7,-75,0,1
320 SPRITE 26,17,R(8),7,-60,0,1: SPRITE
27,18,R(9),7,-50,0,1
330 SPRITE 28,19,R(10),7,-50,0,1: SPRITE
 29,20,R(11),7,-40,0,1
340 SPRITE 30,21,R(12),7,-30,0,1: SPRITE
 31,22,256,7,-10,0,1: SPRITE 32,23,300,7
,-20,0,1
350 CSR 0,23: FOR J=1 TO 32: PRINT CHR$(
129);: NEXT : CSR 25,6: COLOUR 1,10: PRI
NT CHR$(130)
360 LET J=1
370 SOUND T,40,F: PAUSE H: SOUND T,Z,Z
380 LET RA=INT(RND*20): IF RA>L*3 THEN
GOTO 420
390 LET RB=INT(RND*T)*12: LET RC=INT(RND
*5)+E+RB: IF SX(RC)>F THEN  GOTO 420
400 LET RD=INT(RND*80)+140: LET RE=INT(R
ND*H)+50: ADJSPR U,RC,F: ADJSPR T,RC,RD:
 ADJSPR 3,RC,RE: SOUND 3,4,F: PAUSE H: S
OUND 3,Z,Z
410 ADJSPR U,RC,U: LET SX(RC)=RD: LET SY
(RC)=RE-E
420 LET A$=INKEY$: IF A$=" " AND FB=Z TH
EN  LET FB=U: LET B=B+U: SOUND Z,30,240,
5,Z,T*BY,U: SOUND U,40,240,5,Z,T*BY,U: P
AUSE 20
430 GOSUB 4000
440 LET FH=Z: FOR K=E TO 12: GOSUB 3000:
 NEXT : IF FH=U THEN  GOTO 580
450 FOR K=20 TO 24: GOSUB 3000: NEXT : I
F FH=U THEN  GOTO 580
460 GOSUB 4000
470 LET A$=INKEY$: IF A$=CHR$(134) THEN
 LET BY=BY+TE: IF BY>190 THEN  LET BY=19
0
480 IF A$=CHR$(135) THEN  LET BY=BY-TE:
IF BY<F THEN  GOTO 660
490 ADJSPR 3,F,BY: ADJSPR 3,16,BY: ADJSP
R 3,17,BY
500 IF INKEY$=" " AND FB=Z THEN  LET FB=
```

```
U: LET B=B+U: SOUND Z,30,240,5,Z,T*BY,U:
 SOUND U,40,240,5,Z,T*BY,U: PAUSE F
510 GOSUB 4000
520 LET TX=TX-TE-L: IF TX<Z THEN  SPRITE
 13,9,R(6)+H,7,Z,Z,6: SPRITE 14,10,R(6)+
156,7,0,0,6: LET TX=R(6)+H: LET CD=9
530 IF CD<=Z THEN  GOTO 540 ELSE  IF TX<
350 THEN  CSR T,T: LET CD=INT((TX-50)/30
): PRINT CD: SOUND T,H,F
540 IF TX<239 THEN  ADJSPR T,13,TX: ADJS
PR T,14,TX+16
550 LET J=J+U: IF J<400 THEN  GOTO 370
560 CLS : CTLSPR 2,0: SOUND 2,0,0: VS 5:
 PAPER 8: INK 1: CLS
570 GOTO 170
580 PRINT "You were shot down"
590 IF PEEK(64092)<>PEEK(64096) THEN  GO
TO 590
600 SOUND 0,0,0: SOUND 1,0,0: PRINT : PR
INT "You dropped";B;" bombs and"
610 PRINT : PRINT "scored";S;" hits."
620 CSR 2,20: PRINT "Press Space Bar for
 the next mission"
630 IF INKEY$<>" " THEN  GOTO 620
640 FOR J=1 TO 24: LET SX(J)=0: LET SY(J
)=0: NEXT
650 PAPER 5: INK 15: GOTO 140
660 CLS : CTLSPR 2,0: SOUND 2,0,0: VS 5:
 PAPER 7: INK 12: CLS
670 PRINT : PRINT "You flew too low, and
 crashed."
680 GOTO 590
2000 FOR K=1 TO 12
2010 LET R(K)=INT(RND*1000)+250
2020 NEXT : RETURN
3000 IF SX(K)<F THEN  ADJSPR U,K,Z: RETU
RN
3010 LET SX(K)=SX(K)-TE-L: ADJSPR 2,K,SX
(K): IF ABS(BY-SY(K))>E THEN  RETURN
3020 IF SX(K)>72 AND SX(K)<H THEN  LET F
H=U
3030 RETURN
4000 IF PEEK(Q)<>PEEK(R) OR FB=Z THEN  R
ETURN
4010 LET FB=Z
```

```
4020 SOUND Z,Z,Z: SOUND U,Z,Z
4030 ADJSPR U,18,14: ADJSPR U,19,TE: ADJ
SPR T,18,ST-BY/TE: ADJSPR T,19,ST-BY/TE:
 ADJSPR 3,18,23: ADJSPR 4,18,155: ADJSPR
 4,19,155: ADJSPR 5,18,Z
4040 FOR V=F TO Z STEP -5
4050 SOUND 3,6,J
4060 PAUSE 50
4070 NEXT : SOUND 3,Z,Z
4080 ADJSPR U,18,14: ADJSPR U,19,U: ADJS
PR 5,18,80
4090 IF TX>60-BY/TE AND TX<H-BY/TE-5*L T
HEN  ADJSPR Z,13,E: ADJSPR Z,14,18: LET
S=S+U
4100 RETURN
```

Variations

There is plenty of scope for redesigning the bomber and the trees and buildings. You could also add visual effects, such as a dramatic crash scene when the bomber has been hit. This could be inserted at line 570 (transfer the statements in the existing line 570 to line 580). The sound effect of an enormous explosion could also be included at this point.

9
Trapeze

You are high above the crowds in the Big Top of the circus, poised ready to swing on the trapeze. Across the arena, your partner holds the other trapeze, awaiting your signal to set it swinging. Your task is to jump from one swinging trapeze to the other, and back again as many times as you can, without falling. If you fall, there is a safety net to catch you. You can climb to the top and try again.

This is a game for one player. It exploits some unexpected effects of the Memotech's line graphics facilities to produce a fascinating display of flashing colours on the screen. The display is an abstract composition, reminiscent of the scintillating colours of the circus.

How to play

When you run the program, the screen clears to blue and stays clear for about 10 seconds. The Memotech is calculating the values needed to produce the display. Then the screen becomes dark red. On the left you see a coloured line, which swings regularly from left to right and back again, repeatedly. As it swings it generates a multicoloured pattern of lines, like an open fan. This represents the cords of the trapeze that you are swinging on. You can tell to which position you have swung by watching to see where the colours of the lines are changing.

To the right of the screen is another line but, although it changes colour, it is not moving. This is the other trapeze, being held by your partner. When you are ready and judge the moment to be the right one, press the space bar. This is your signal to your partner to release the other trapeze. The line on the right will also move now, producing a fan-like pattern of lines just like that produced by the left-hand trapeze.

Your score (the number of successful jumps) is displayed in the

centre of the screen below the trapezes. To begin with, this is '0'. Beside the score figure is an arrow which points either to the left or to the right. This indicates which trapeze you are swinging on. To begin with, this arrow is pointing to the left.

When you think the two trapezes are close enough together, press key 'J'. This makes you jump from one trapeze to the other. If your jump is successful, your score is increased by 1. The arrow then changes, to point to the trapeze to which you have jumped. You can press 'J' as often as you like, jumping to and fro between the trapezes as many times as you can.

Note that if you simply hold the 'J' key down, waiting for the trapezes to come close enough together, the Memotech ignores the key-press. Press the key only at the exact moment at which you wish to jump.

If you press the key too long before a new trapeze line is due to be displayed, the computer will ignore your command, as explained above. It is as if you decided to jump, but changed your mind at the last moment. If you press it just before a new line is due, there is no time for changing your mind. You attempt the jump and, if it happens that the two trapezes are too far apart at that instant, you fall. The game then ends. Your score and the best score so far are displayed on the screen.

To play again, press the space bar.

Winning tactics

To begin with, it is essential that the two trapezes should swing toward one another, so as to come close together. Then you can leap the gap safely. Ask your partner (press space bar) to release the right-hand trapeze when the left-hand one is swinging far to the left.

Like a real trapeze, a Memotech trapeze that has no one on it gradually swings less and less widely. The gap between it and the trapeze you are on may then be too wide to cross. To avoid this happening, try to leap from one to the other and back again, keeping both trapezes swinging to their full extent. If a trapeze has lost a lot of its swing but you manage to jump across to it, it will gradually increase its swing as you 'work' on it. Spend a few moments on it to bring it back to maximum swing. But take care that, in the meantime, the other trapeze has not lost too much of its swing.

Note that every time you jump, the distance that you can cross is reduced (you are becoming tired!). The game gets more and more

difficult and you are more likely to fall. If you can make 18 jumps or more, you are doing really well. If you succeed in making 20 successful jumps, you are told to rest and the game ends.

Keying in

This is a short program but there are a few pitfalls for the unwary. Take care with the expressions on lines 40 and 50. For example, note that line 40 ends with two brackets, but line 50 ends with three. In line 310, the first pair of quotes contains four spaces but the second pair contains only two spaces.

Program design

20	initialising arrays and variables
30–140	calculating values needed for producing the swinging trapeze lines
150–180	initialising variables, or resetting them ready for the next game; setting main display colours
190–200	choosing a random colour and plotting lines of trapezes; detecting a key-press that is too early
210–230	setting position of next line; reversing direction of swing, if line has reached either end of its swing
240	releasing right-hand trapeze, if not already released
250	damping the swing of the right-hand trapeze, if you are not on it
260	detecting 'jump' command and testing to see if it is successful; if so, increment score; if not, set 'fail' flag
270–280	'working up' trapezes to maximum swing, if you are on them
290	damping the swing of the left-hand trapeze, if you are not on it
300	resetting loop counter
310	displaying score
320	end game when 20 jumps made
330	repeat loop unless jump has failed
340–380	displays at end of game
390–400	inviting another game

The program

```
10 REM ** TRAPEZE **
20 DIM M(73,2),N(73),S(37,3): LET U=1: L
ET TW=2: LET TS=37: LET SF=64: LET HS=17
5: LET HN=192: LET Z=0: LET TE=3: LET TY
=20: LET BJ=0
30 FOR J=37 TO 73
40 LET M(J,1)=INT(64.5+137*SIN((J-37)/10
0))
50 LET M(J,2)=INT(38.5+137*(1-COS((J-37)
/100)))
60 NEXT
70 FOR J=1 TO 36
80 LET M(J,1)=128-M(74-J,1)
90 LET M(J,2)=M(74-J,2)
100 NEXT
110 FOR J=1 TO 73: LET N(J)=M(J,1)+128:
NEXT
120 FOR J=1 TO 37
130 LET S(J,1)=38-J: LET S(J,2)=J+36
140 LET S(J,3)=INT((J-1)/8+.5)+1: NEXT
150 LET SA=37: LET SB=1: LET NA=1: LET N
B=0: LET FG=0: LET L=0: LET Y=1: LET SC=
0: LET JD=128: LET KF=0: LET FF=0
160 VS 4: COLOUR 0,6: COLOUR 1,15: COLOU
R 2,6: COLOUR 4,10: CLS
170 LET MA=8: LET MB=0
180 LET A=1: LET B=73
190 COLOUR TE,RND*14+U: LET KF=Z: IF INK
EY$<>"" THEN  LET KF=U
200 LINE M(A,U),M(A,TW),SF,HS: LINE N(B)
,M(B,TW),HN,HS
210 LET A=A+MA: LET B=B+MB
220 IF A<S(SA,U)-MA OR A>S(SA,TW)-MA THE
N  LET MA=-MA
230 IF B<S(SB,U)-MB OR B>S(SB,TW)-MB THE
N  LET MB=-MB
240 IF INKEY$=" " AND FG=Z THEN  LET MB=
-8: LET SB=TS: LET FG=U
250 LET L=L+U: IF L>TY AND SB>TW AND Y=U
 THEN  LET SB=SB-U: LET MB=S(SB,TE)*SGN(
MB): LET L=Z
260 IF INKEY$="J" AND KF=U THEN  LET FF=
U: IF N(B)-M(A,U)<JD THEN  LET Y=-Y: LET
```

```
 SC=SC+U: LET JD=JD-TW: LET FF=Z
270 IF Y=U AND SA<TS THEN  LET SA=SA+U:
LET MA=S(SA,TE)*SGN(MA)
280 IF Y=-U AND SB<TS THEN  LET SB=SB+U:
 LET MB=S(SB,TE)*SGN(MB)
290 IF Y=-U AND SA>TW AND L>TY THEN  LET
 SA=SA-U: LET MA=S(SA,TE)*SGN(MA)
300 IF L>TY THEN  LET L=Z
310 CSR 12,TY: IF Y=1 THEN  PRINT "<-";S
C;"     " ELSE  PRINT "  ";SC;" ->"
320 IF JD<90 THEN  GOTO 360
330 IF FF=Z THEN  GOTO 190
340 CSR 2,7: PRINT "You fell after";SC;"
 jumps"
350 GOTO 370
360 CSR 2,7: PRINT "Time to rest, after"
;SC;" jumps"
370 IF SC>BJ THEN  LET BJ=SC
380 CSR 2,10: PRINT "Best score is";BJ;"
 jumps"
390 IF INKEY$<>" " THEN  GOTO 390
400 CLS : GOTO 150
```

10
Snipers

A keen eye and an accurate aim are the prime requirements for beating the snipers. You are in your armoured vehicle stationed beside a row of buildings. Snipers are firing at you from the windows and doorways. There are walls and trees beside the buildings. These provide more hiding places for the snipers. There are snipers in unexpected places, too.

When a sniper fires you see the flash from the rifle. This is your only clue to where the sniper is hiding. If you are quick enough and aim your machine gun at the spot where the flash came from, you can put the sniper out of action. But snipers are cunning and, if you take too long to bring your gun to bear, the sniper will have taken cover, and lives to fire again.

Sooner or later a lucky shot (unlucky for you!) penetrates your armour and you have to give up the fight. Your task is to silence as many of the snipers as you can before that happens.

How to play

The level you choose decides how many snipers there are, and how quick you have to be to put them out of action. As soon as the display has been drawn, the battle begins. The gunsight (a white frame with a cross) appears in the area of sky near the top left corner of the screen. Move it by using the four direction keys in the keypad to the right of the main keyboard.

The keys have repeat action, so hold any one down to sweep the sight across the screen quickly. You will hear the rifle shots and see the flashes. Move your gunsight to aim it at a sniper. When your sight is aimed at the spot where you saw a sniper, press the space bar, just once. You will hear a burst of machine-gun fire. If you fired in time, the sniper will not fire from that spot again. A 'beep' from the

computer indicates that your bullets have found this target. Move quickly on to fire at another sniper, for their shots are being counted by the computer and the next shot may be the one that ends the game. As you eliminate the snipers, those remaining fire less frequently. This gives you more time to discover where the better hidden ones are located. You may prefer to take your time to find and shoot at these. But beware – one of them may still get you. The game ends when there are only five snipers left. These wisely surrender!

When the game ends, you are told how many snipers you hit, and the record score so far.

Winning tactics

Go for the snipers in the windows and doorways first, for these are the easiest to see and locate. It is much harder to see the ones in and around the trees. At levels 8 to 10 you have very little time in which to get your sight trained on a sniper. Unless the sight is very close to a flash, it is hardly worthwhile machine-gunning it. Keep the sight on the building at first, waiting to pounce on any nearby flash you see. Incidentally, the snipers are scattered in a fairly random way, so do not expect to find them in exactly the same places each time you play.

Keying in

As usual, it is the DATA and GENPAT statements which need the most care. Note the essential space between the quotes in line 150. Without this you will get no picture.

Program design

20	initializing array and a variable
30–80	designs for sprites (sight and rifle flash)
90	requesting level
100–160	READing DATA and using it to display the scene
170–210	READing DATA to get details of all sniper positions, then inactivating some at random, to leave a random assortment active

220–240	initializing variables; counting number of inactive snipers and rifle flash initialized
250–260	the gunsight
270–340	a sniper fires
350–490	moving your sight
500–510	updating 'time' factor
520–560	firing your machine gun and registering a hit, if any
570–580	to final display if all but 5 snipers inactivated; otherwise to another shot
590–620	display of score if you are hit
630–650	inviting re-play
660–690	display when all but 5 snipers hit
1000–1170	DATA for display
2000–2040	DATA for sniper locations

Points of interest

The display routine shows an easy way of building up complex pictures on the Memotech. Solid blocks of colour are placed at each screen location by printing a 'space' (line 150). The DATA consists of pairs of numbers. The first gives the number of blocks needed (N) and the second their colour (C). The operation is done in a loop (lines 140–160), so 'scanning' the screen and building up the picture. The COLOUR 0 statement sets the 'paper' colour, for printed characters while in graphics mode. The result is a solid block of 'paper' (background) colour.

The program

```
10 REM ** SNIPERS **
20 DIM S(56,4): LET SM=0
30 CTLSPR 1,64: CTLSPR 2,5: CTLSPR 6,1
40 GENPAT 3,1,224,128,128,2,2,2,31,2
50 GENPAT 3,2,2,2,128,128,224,0,0,0
60 GENPAT 3,3,56,8,8,0,0,0,192,0
70 GENPAT 3,4,0,0,8,8,56,0,0,0
80 GENPAT 3,5,0,0,0,0,24,24,0,0
90 VS 5: CLS : CSR 3,3: INPUT "Level? (1
-10)  ";L$
100 LET L=VAL(L$): IF L<1 OR L>10 THEN
GOTO 90
```

```
110 VS 4: COLOUR 2,10: COLOUR 4,13: CLS
120 CSR 0,0: FOR J=1 TO 176
130 READ N,C
140 FOR K=1 TO N
150 COLOUR 0,C: PRINT " ";
160 NEXT : NEXT
170 FOR J=1 TO 56: READ S(J,1): READ S(J
,2): NEXT
180 FOR K=1 TO 30-L*2
190 LET R=INT(RND*56)+1
200 LET S(R,3)=1
210 NEXT
220 LET SU=0: LET SS=0: LET SN=0: LET T=
0: LET XX=31: LET YY=176
230 FOR J=1 TO 56: IF S(J,3)=0 THEN  LET
 SU=SU+1
240 NEXT
250 SPRITE 1,1,31,176,0,0,15: SPRITE 2,2
,31,160,0,0,15
260 SPRITE 3,3,47,176,0,0,15: SPRITE 4,4
,47,160,0,0,15
270 SPRITE 5,5,0,0,0,0,0
280 LET SP=INT(RND*56)+1
290 IF S(SP,3)=1 THEN  GOTO 350
300 LET S(SP,4)=T
310 ADJSPR 2,5,S(SP,1)+4: ADJSPR 3,5,S(S
P,2)-4: ADJSPR 1,5,15
320 SOUND 3,4,15: PAUSE 100: SOUND 3,0,0
330 ADJSPR 1,5,0
340 LET SS=SS+1
350 FOR R=1 TO 10-INT(L/2): LET A$=INKEY
$
360 LET A=ASC(A$)
370 IF A$=" " THEN  GOTO 520
380 IF A=25 THEN  LET XX=XX+8
390 IF A=11 THEN  LET YY=YY+8
400 IF A=10 THEN  LET YY=YY-8
410 IF A=8 THEN  LET XX=XX-8
420 IF XX<7 THEN  LET XX=7
430 IF XX>239 THEN  LET XX=239
440 IF YY<16 THEN  LET YY=16
450 IF YY>192 THEN  LET YY=192
460 ADJSPR 2,1,XX: ADJSPR 3,1,YY
470 ADJSPR 2,2,XX: ADJSPR 3,2,YY-16
480 ADJSPR 2,3,XX+16: ADJSPR 3,3,YY
```

```
490 ADJSPR 2,4,XX+16: ADJSPR 3,4,YY-16
500 LET T=T+10
510 GOTO 570
520 FOR J=1 TO 56
530 SOUND 3,6,15: LET T=T+1
540 IF S(J,1)=XX+1 AND S(J,2)=YY AND T-S
(J,4)<700-40*L THEN  LET S(J,3)=1: LET S
N=SN+1: SOUND 0,100,15: PAUSE 200: SOUND
 0,0,0
550 PAUSE 10: SOUND 3,6,0: PAUSE 10
560 NEXT
570 NEXT : IF SN=SU-5 THEN  GOTO 660
580 IF SS<150 THEN  GOTO 280
590 VS 5: CLS : CSR 3,3: PRINT "They got
 you at last!!"
600 CSR 3,9: PRINT "But you hit ";SN;" o
f them."
610 IF SM<SN THEN  LET SM=SN
620 CSR 3,9: PRINT "The best score is ";
SM;"."
630 CSR 5,20: PRINT "Key 'Y' to shoot ag
ain"
640 IF INKEY$<>"Y" THEN  GOTO 640
650 RESTORE 1000: GOTO 90
660 VS 5: CLS : CSR 3,3: PRINT "Well don
e!!"
670 CSR 3,6: PRINT "You hit ";SN;" snipe
rs,"
680 PRINT " leaving 5 alive, who surrend
ered."
690 GOTO 610
1000 DATA 72,5,1,2,9,5,2,8,2,5,1,8,16,5,
3,2,7,5,4,8
1010 DATA 1,5,1,8,15,5,4,2,6,5,7,8,16,5,
4,2,3,5,10,8
1020 DATA 3,5,2,3,3,5,4,14,3,5,4,2,4,5,8
,8,2,5,5,3
1030 DATA 2,5,5,14,1,5,5,2,4,5,1,8,2,1,2
,8,2,1,1,8
1040 DATA 2,5,7,3,6,14,5,2,4,5,8,8,2,5,7
,3,5,14,6,2
1050 DATA 4,5,1,8,2,1,2,8,2,1,1,8,2,5,7,
3,5,8,5,2
1060 DATA 2,5,1,8,2,5,8,8,2,5,7,3,2,8,2,
1,2,8,4,2
```

```
1070 DATA 1,5,3,8,1,5,8,8,2,5,7,3,2,8,2,
1,2,8,1,5,3,2
1080 DATA  2,8,1,1,3,8,2,1,2,8,2,1,1,8,3
,5,6,3,6,8
1090 DATA  1,5,3,2,13,8,3,5,5,3,1,5,2,8,
2,1,2,8,2,5
1100 DATA 1,6,1,5,1,8,1,1,1,8,1,1,9,8,5,
5,2,6,2,5
1110 DATA 2,8,2,1,2,8,2,5,1,6,1,5,6,8,2,
1,2,8,2,1
1120 DATA 1,8,5,5,2,6,2,5,6,8,2,5,1,6,1,
5,18,8,2,6
1130 DATA 8,8,2,12,1,6,1,12,18,8,2,6,4,8
,2,1,2,8,2,12
1140 DATA 1,6,1,12,1,8,2,1,5,8,2,1,8,8,2
,6,4,8,2,1
1150 DATA 2,8,2,12,1,6,1,12,1,8,2,1,5,8,
2,1,8,8,2,6
1160 DATA  4,8,2,1,2,8,2,12,1,6,1,12,1,8
,2,1,5,8,2,1
1170 DATA 8,8,2,6,2,8,28,10,2,6
2000 DATA 56,168,168,168,120,160,184,160
,16,152,40,136,72,136,128,136,136,136,16
0,136,168,136
2010 DATA 216,136,240,128,128,120,136,12
0,160,120,168,120,208,120,56,104,112,104
,232,104,16,96
2020 DATA 24,96,96,96,128,96,136,96,160,
96,168,96,216,96,16,80,88,80,104,80,24,7
2
2030 DATA 128,72,136,72,160,72,168,72,20
0,72,240,72,48,64,72,64,216,56,16,48,88,
48
2040 DATA 152,48,240,48,24,40,56,40,144,
40,24,32,72,32,96,32,152,32,216,24,40,16
,176,16
```

Variations

Altering values in lines 2000–2040 places snipers in different positions. The numbers are graphics x and y positions, counting x from 0 to 256 from left to right and y from 0 to 192 from bottom to top. If you would prefer a longer battle, increase the number in line

580 from 150 to, say, 200. If you like the pace of level 10, but can never move your sight quickly enough, alter the 2 in line 350 to 3 or 4.

If you tire of street fighting, why not transfer the skirmish to a jungle or even make it a naval battle? It is easy to redesign the whole display simply by changing DATA lines 1000–1170, and amending lines 2000–2040 to place the snipers in suitable positions.

11
Ball Maze

Steer the ball through the maze from beginning to end, without letting it drop into one of the holes. It needs a steady hand and quick reactions but, given a little practice, it can be done.

How to play

The outline of the maze begins to build up on the screen as soon as you RUN the program. Then the holes appear. These are represented by black disks. The green square is the hole to which you have to guide the ball. The ball appears as a red disk at the top left-hand corner.

The tilt indicators appear along the right side of the screen, and along the bottom edge. The centre of each indicator is marked with a purple star. The idea of the indicators is that they show you which way the maze is 'tilted' and by how much. This helps you control the direction in which the ball rolls and the speed at which it travels.

The maze is tilted by pressing one of the direction keys in the keypad at the right of the Memotech's keyboard:

left-arrow	tilt to left
right-arrow	tilt to right
up-arrow	tilt away from you (toward top of screen)
down-arrow	tilt toward you (toward bottom of screen)

The keys have repeat action so the longer you hold one down, the greater the amount of tilt.

You need to press the right-arrow key to start the ball moving to the right, but do not press it for too long, or the ball may gather speed and fall straight into the first hole. Watch the pointer on the bottom tilt indicator and judge when you should press the left-arrow key. One point to note is that when you are trying to run the ball in

one direction (say, to the right), the maze must not be tilted strongly in a direction at right angles to this (i.e. up or down). If the ball is not free to move in the direction in which the maze has a steep tilt, it 'jams' against the wall and cannot move.

When the ball falls into any of the holes (except the green one), you hear a sound effect. A moment later, the ball reappears at the top left corner, ready for another try. The game is being timed, and when you eventually run the ball into the green hole, you will be told how many minutes you have taken. To play again, press BRK, then re-run the program.

Winning tactics

Anticipation is the key to success in this game. As the ball gathers speed, tilt the maze in the opposite direction to slow it down and prevent it from running into a hole ahead. You will find that, whenever you alter the direction of tilt, a single key-press takes it back to 'zero tilt' immediately and then gives it a slight tilt in the opposite direction.

It is possible to make use of the jamming effect of excessive crossways tilt. Applying this suddenly may often bring the ball to a halt more rapidly than trying to reverse the direction of tilt. It is also useful to be able to make the ball motionless for a few moments while you plan how you intend to negotiate the next awkward part of the course.

Keying in

The design of the maze is contained in lines 1000 to 1230. Make certain every '1' and '0' is correct, otherwise the paths of the maze may be closed across, or there may be gaps in the walls, making it too easy. When keying in these lines it is essential not to type 'DATA'. Use the abbreviation 'DA.' instead, as explained in key points. This also applies if you have to edit any of the DATA lines.

Program design

30	initialising array and sprite control
40–60	patterns for sprites (tilt indicators) and graphics (ball-holes)

70–220	displaying the maze and tilt indicators
230	starting the clock and initialising variables
240–250	re-entry point for each calculation: clearing tilt pointers from screen
260–350	reading keys, calculating and displaying tilt
360–370	calculating the next position for the ball
380–460	finding out if the ball is free to move to this position, and what would happen if it went there
470	moving the ball (unless already down a hole)
480–490	preparing for next move
500–600	ball falls down a hole (sound effect and return ball to start)
610–650	ball falls down final hole; display time
1000–1230	DATA for walls of maze
2000–2010	DATA for positions of holes (except the green one)
3000–3010	subroutine to detect when a sound is finished

The program

```
10 REM ** BALL MAZE **
20 DIM H(20,2)
30 CTLSPR 1,8: CTLSPR 2,2: CTLSPR 6,0
40 GENPAT 1,129,0,60,126,126,126,126,60,
0: GENPAT 1,151,255,255,255,255,255,255,
255,255
50 GENPAT 1,131,0,60,126,102,102,126,60,
0: GENPAT 1,132,0,126,126,126,126,126,12
6,0
60 GENPAT 3,1,24,60,126,219,153,24,24,24
: GENPAT 3,2,24,48,96,255,255,96,48,24
70 VS 4: CLS : COLOUR 0,15: COLOUR 1,4:
COLOUR 2,4: COLOUR 4,4
80 CSR 0,0: FOR J=0 TO 23
90 READ A$
100 FOR K=0 TO 31: IF K=31 AND J=23 THEN
  GOTO 120
110 IF A$(K+1)="1" THEN  PRINT " "; ELSE
  PRINT CHR$(151);
120 NEXT : NEXT
130 COLOUR 1,1: FOR J=1 TO 20
140 READ H(J,1): READ H(J,2): CSR H(J,1)
,H(J,2): PRINT CHR$(129)
```

```
150 NEXT
160 COLOUR 1,8: CSR 1,1: PRINT CHR$(131)
170 LET X=1: LET Y=1: LET NX=1: LET NY=1
180 COLOUR 1,2: CSR 29,21: PRINT CHR$(13
2)
190 COLOUR 1,13: CSR 31,12: PRINT "*"
200 CSR 15,23: PRINT "*";
210 SPRITE 1,1,124,4,0,0,0: LET SX=124
220 SPRITE 2,2,252,92,0,0,0: LET SY=92
230 CLOCK "000000": LET AX=0: LET AY=0:
LET VX=0: LET VY=0
240 IF AX<>0 THEN  ADJSPR 1,1,0
250 IF AY<>0 THEN  ADJSPR 1,2,0
260 LET A=ASC(INKEY$)
270 IF A=25 AND AX<0 OR A=8 AND AX>0 THE
N  LET AX=0
280 IF A=25 AND AX<10 THEN  LET AX=AX+1
290 IF A=8 AND AX>-10 THEN  LET AX=AX-1
300 IF A=10 AND AY<0 OR A=11 AND AY>0 TH
EN  LET AY=0
310 IF A=10 AND AY<10 THEN  LET AY=AY+1
320 IF A=11 AND AY>-10 THEN  LET AY=AY-1
330 IF AX<>0 THEN  ADJSPR 2,1,SX+8*AX: A
DJSPR 1,1,13
340 IF AY<>0 THEN  ADJSPR 3,2,SY-8*AY: A
DJSPR 1,2,13
350 LET VX=VX+AX*2: LET VY=VY+AY*2
360 IF ABS(VX)>10 THEN  LET NX=X+1*SGN(V
X): LET VX=0
370 IF ABS(VY)>10 THEN  LET NY=Y+1*SGN(V
Y): LET VY=0
380 CSR NX,Y
390 LET CX=ASC(SPK$)
400 CSR X,NY: LET CY=ASC(SPK$)
410 CSR NX,NY: LET CZ=ASC(SPK$)
420 IF CX=129 OR CY=129 OR CZ=129 THEN
GOTO 500
430 IF CX=151 THEN  LET VX=0: LET NX=X
440 IF CY=151 THEN  LET VY=0: LET NY=Y
450 IF CZ=151 THEN  LET VX=0: LET VY=0:
LET NX=X: LET NY=Y
460 IF CX=132 OR CY=132 THEN  GOTO 610
470 CSR X,Y: PRINT " ": CSR NX,NY: PRINT
 CHR$(131)
480 LET X=NX: LET Y=NY
```

```
490 GOTO 240
500 CSR X,Y: PRINT " "
510 IF AX<>0 THEN  ADJSPR 1,1,0
520 IF AY<>0 THEN  ADJSPR 1,2,0
530 SOUND 0,400,160,50,0,100,1
540 GOSUB 3000
550 SOUND 0,1000,240,0,-15,70,1
560 GOSUB 3000
570 SOUND 0,0,0
580 LET AX=0: LET AY=0: LET VX=0: LET VY
=0
590 LET X=1: LET Y=1: LET NX=1: LET NY=1
600 GOTO 210
610 CLS : VS 4: CLS
620 LET T=VAL(MID$(TIME$,3,2))
630 CSR 3,3: ADJSPR 1,1,0: ADJSPR 1,2,0
640 PRINT "You won! - in ";T;" minutes"
650 GOTO 650
1000 DATA 000000000000000000000000000000
00
1010 DATA 011111111111111111110111010101
00
1020 DATA 000000010000000000000101010101
01
1030 DATA 011111011111011111111101011101
01
1040 DATA 010001010001010000000001010101
01
1050 DATA 011101111101111101110111010101
01
1060 DATA 000100010101000001010101010101
01
1070 DATA 010111010101011101010101010111
01
1080 DATA 010001010101010101010100010100
01
1090 DATA 010111010100010111010101110101
01
1100 DATA 011100010101010000010101000001
01
1110 DATA 010111010101111111011111011111
01
1120 DATA 010001000100010101000100010000
01
```

```
1130 DATA 0101111111101110101110101010111
01
1140 DATA 0101000000001000101000001010101
01
1150 DATA 0111111111110111010111011111101
01
1160 DATA 0101000000000010001010101000001
01
1170 DATA 0101011111111011111010101110111
01
1180 DATA 0111010000010101000001010101010
01
1190 DATA 0101010111111110111010101010101
01
1200 DATA 0101010000000000101000100010100
01
1210 DATA 0101011111111111101111111110111
01
1220 DATA 0000000000000000000000000000000
01
1230 DATA 0000011111111111111111111110000
00
2000 DATA 19,1,25,1,29,1,15,5,1,7,23,7,1
1,8,27,9,29,9,7,11
2010 DATA 11,11,21,13,23,13,15,15,3,17,2
9,18,7,19,19,19,23,19,1,21
3000 IF PEEK(64082)<>PEEK(64086) THEN G
OTO 3000
3010 RETURN
```

Variations

A new maze can be designed on a sheet of squared paper, 32 squares across and 24 squares deep. This includes the wall around the maze and the tilt indicator area. Shade in squares to define where the dividing walls are to be. The DATA consists of 24 groups of 32 digits, one group for each row of the design. Where there is a shaded square (wall), the digit is a '1'; where there is a blank square (path) there is a '0'. Replace the listed lines 1000 to 1230 with lines containing details of your own maze. Alter lines 2000 and 2010 to give the positions of the holes in your maze. These are given as pairs

of numbers in which the first represents screen column no. (0 to 31) and the second represents screen row (0 to 23, counting down from the top). If you want to start the ball in a different position, or relocate the final hole, amend lines 160 to 180 accordingly.

12
Nine Men Morris

Nine Men Morris is a traditional game. There are two players, each with 9 pieces or 'men' to move around the curiously designed playing board. Although the rules are simple, winning a game is not always as easy as it might seem.

How to play

The two players are represented by two colours, red and blue. Red plays first. The border of the screen shows the colour of the person whose turn it is to play.

The game has two stages. In the first stage, each player puts nine pieces on to the board, one at a time. As soon as the board is displayed you will see a question mark at the top left corner. This is an invitation to place your first piece on that point. It is Red's turn to play. Press the '/' key if you want to put a piece on that point. If not, press the space bar, to move the question mark to the next point on the board. You can press the space bar as many times as you like until the question mark is on the position you select. Then press '/'. When you press '/', a red number appears at the point. This is your piece. The border changes to blue, showing that it is Blue's turn next. Blue places a piece as described for Red.

If you accidentally move the question mark past a point you want to select, carry on pressing the space bar to bring the question mark back to the top of the board again and then on to the point you have selected.

During this stage of the game you should try to place three of your own pieces in line. Three pieces are in line when they are filling the three points on the sides of any of the squares, or on any of the lines which run between the squares. When you place pieces in line, the computer asks you to 'Take one piece'. This means that you must

now take one of your opponent's pieces from the board. Do this by pressing the corresponding number key (1–9) at the top of the keyboard, or on the keypad to the right of the main keyboard.

When each player has placed 9 pieces, the game moves on to its next stage. The border is coloured red or blue, by turns, as before.

You have to move your pieces, in turn, from one point to the next vacant point. Pieces cannot jump over each other, and two pieces cannot be on the same point. When it is your turn to play, the border shows your colour and a message 'Which piece?' appears. Select the piece by pressing one of the number keys. If you select a piece that has already been removed, or one that has no moves available to it (because all adjacent points are occupied), your selection is ignored. Select another piece instead. Then the message 'Which move?' appears. Move your piece in the required direction by pressing the appropriate direction keys of the keypad. If by accident you have tried to make an impossible move, you are asked to select a piece and its move again.

You win the game at this stage in one of two ways:

(1) by blocking your opponent so that no moves are possible.
(2) by reducing the number of your opponent's pieces to 2.

As in the first stage of the game, you should try to move three of your pieces into line, so allowing you to take away any one of your opponent's pieces.

When the game ends, a message shows who has won and why. Press 'S' to end the program or any other key to play again.

Winning tactics

In the first stage of the game, try to get your pieces in line as often as possible. Pieces placed in the middle of the side of a square can be used for three-in-line in more than one direction. At the same time try to prevent your opponent from placing three in a line. When you 'take' a piece, avoid taking one that is already in a line of three. Otherwise, your opponent can replace this with another piece at the next turn, scoring three-in-line and taking one of your pieces.

In the second stage you can often move one of your three-in-line pieces out of line and then back into line again at the next turn. This can be a quick way to reduce your opponent's numbers. If you have few pieces left, keep them close together and try for three-in-line as often as you can. If you have many pieces it is usually better to

spread them widely over the board, so as to reduce the number of moves available to your opponent and eventually block your opponent completely.

There are some situations, particularly when both players have few pieces left, in which there is a 'stalemate' and neither player can force a win. The computer is not able to detect such a situation, so players must agree to a draw. When asked "Which piece?" press key 'S' (for stalemate). Then key in 'RUN' to play again.

Keying in

Take care with DATA statements especially with the POKE statements on lines 150, 670 and 1810. The two POKE statements in line 150 enable the keypad and disable the BRK key. Those in lines 670 and 1810 restore normal action. Omit the second one (POKE 64862, 13) from line 150 of the program when you first type it in. If you include this statement and if there are mistakes in your typing, the program may crash and you will not be able to use the BRK key. You will have to reset the computer (by pressing the two keys on either side of the space bar) or switch it off. In either case you will lose everything you have typed in! When you have finished typing in the program (without the second POKE of line 150), save it to tape and verify it. *Then* test it. Pressing key '9' will break the program instead of moving piece 9. If all seems to be correct, insert the POKE statement at line 150. Then save and verify the program again. It should now work correctly, with key 9 moving piece 9, but not breaking the program.

Program design

20–40	initialising
50–140	displaying the board.
150	enabling the key-pad and disabling BRK
160	re-entry point for each move of 1st stage
170–260	placing a piece
270–290	looking for 3 in-a-line; return for next move
300	re-entry point for each move of 2nd stage
310–390	is player completely blocked?
400–470	which piece to move?
480–600	requesting and executing move

610–620	looking for 3-in-line; return for next move
630–680	final routines with option to play again or to stop the program and restore normal action of keypad
900–910	subroutine to change player at beginning of each move
1000–1200	subroutine looking for 3-in-line, and supervising the 'take'
1500–1580	subroutine to find a 3-in-line
1800–1830	input subroutine with option to stop program and restore normal action of keypad
2000–2010	DATA for array M()

The program

```
10 REM ** NINE MEN MORRIS **
20 DIM M(7,24),P(2,9): LET X=1: LET Y=1:
 LET NX=1: LET NY=1: LET M=2: LET P=0
30 LET F7=0: FOR K=1 TO 24: FOR J=1 TO 7
: READ M(J,K): NEXT : NEXT
40 VS 4: COLOUR 0,2: COLOUR 1,15: COLOUR
 2,11: COLOUR 3,2: COLOUR 4,8: CLS
50 FOR J=0 TO 2
60 PLOT 60+24*J,172-24*J: ANGLE 0
70 FOR K=1 TO 4
80 DRAW 144-48*J: PHI -1.57
90 NEXT : NEXT
100 LINE 132,172,132,124: LINE 132,76,13
2,28
110 LINE 60,100,108,100: LINE 156,100,20
4,100
120 FOR J=1 TO 24
130 CSR M(6,J),M(7,J): PRINT " "
140 NEXT
150 POKE 64145,132: POKE 64862,13: GOSUB
 900
160 IF P=10 THEN  LET P=1
170 LET K=0: COLOUR 0,2
180 LET K=K+1: IF K=25 THEN  LET K=1
190 CSR M(6,K),M(7,K): IF ASC(SPK$)<>32
THEN  GOTO 180
200 CSR M(6,K),M(7,K): PRINT "?"
210 LET A$=INKEY$: IF A$="" THEN  GOTO 2
10
```

```
220 IF A$=" " THEN  CSR M(6,K),M(7,K): P
RINT " ": PAUSE 500: GOTO 180
230 IF A$="/" THEN  GOTO 250
240 GOTO 210
250 COLOUR 0,11-M*3: CSR M(6,K),M(7,K):
PRINT CHR$(P+48)
260 LET M(1,K)=P+10*(M-1): LET P(M,P)=1
270 GOSUB 1000
280 IF F7=7 THEN  GOTO 630
290 IF  NOT (P=9 AND M=2) THEN  GOTO 150
300 GOSUB 900
310 LET FB=1
320 FOR K=1 TO 24
330 LET MM=M(1,K)
340 IF MM=0 OR MM>10 AND M=1 OR MM<10 AN
D M=2 THEN  GOTO 380
350 FOR J=2 TO 5
360 IF M(J,K)>0 THEN  IF M(1,M(J,K))=0 T
HEN  LET FB=0
370 NEXT
380 NEXT
390 COLOUR 4,11-M*3: IF FB=1 THEN  GOTO
650
400 CSR 3,22: PRINT "Which piece?"
410 GOSUB 1800
420 IF P(M,A)<>1 THEN  GOTO 410
430 LET K=1
440 IF M(1,K)<>A+10*(M-1) THEN  LET K=K+
1: GOTO 440
450 LET FB=1: FOR J=2 TO 5: IF M(J,K)<>0
 THEN  IF M(1,M(J,K))=0 THEN  LET FB=0
460 NEXT
470 IF FB=1 THEN  GOTO 400
480 CSR 3,22: PRINT "Which move? "
490 LET A$=INKEY$: IF A$="" THEN  GOTO 4
90
500 IF A$="5" THEN  LET D=2: GOTO 550
510 IF A$="." THEN  LET D=3: GOTO 550
520 IF A$="3" THEN  LET D=4: GOTO 550
530 IF A$="1" THEN  LET D=5: GOTO 550
540 GOTO 490
550 IF M(D,K)=0 THEN  GOTO 400
560 IF M(1,M(D,K))<>0 THEN  GOTO 400
570 COLOUR 0,11: CSR 3,22: PRINT "
      "
```

```
580 COLOUR 0,2: CSR M(6,K),M(7,K): PRINT
 " "
590 COLOUR 0,11-3*M: CSR M(6,M(D,K)),M(7
,M(D,K)): PRINT CHR$(A+48)
600 LET M(1,K)=0: LET M(1,M(D,K))=A+10*(
M-1): LET P=A
610 GOSUB 1000
620 IF F7<7 THEN  GOTO 300
630 CSR 3,22: PRINT "Seven men removed -
 you WIN"
640 GOTO 660
650 CSR 3,22: PRINT "All moves blocked -
 you LOSE"
660 IF INKEY$="" THEN  GOTO 640
670 IF A$="S" OR A$="s" THEN  POKE 64145
,160: POKE 64862,15: STOP
680 RESTORE 2000: LET P=0: GOTO 30
900 IF M=2 THEN  LET M=1: LET MX=2: COLO
UR 0,8: COLOUR 4,8: LET P=P+1: RETURN
910 LET M=2: LET MX=1: COLOUR 0,5: COLOU
R 4,5: RETURN
1000 LET FT=0
1010 FOR K=2 TO 23
1020 IF M(2,K)=0 OR M(3,K)=0 THEN  GOTO
1040
1030 LET PA=1: GOSUB 1500
1040 IF M(4,K)=0 OR M(5,K)=0 THEN  GOTO
1060
1050 LET PA=3: GOSUB 1500
1060 NEXT
1070 IF FT=0 THEN  RETURN
1080 CSR 3,22: PRINT "Take one piece"
1090 GOSUB 1800
1100 IF P(MX,A)=1 THEN  GOTO 1120
1110 GOTO 1090
1120 COLOUR 0,11: CSR 3,22: PRINT "
          "
1130 LET P(MX,A)=2
1140 FOR K=1 TO 24
1150 IF M(1,K)=A+10*(MX-1) THEN  LET M(1
,K)=0: COLOUR 0,2: CSR M(6,K),M(7,K): PR
INT " "
1160 NEXT
1170 LET F7=0: FOR J=1 TO 9
1180 IF P(MX,J)=2 THEN  LET F7=F7+1
```

```
1190 NEXT
1200 RETURN
1500 IF M(1,K)=0 OR M(1,M(PA+1,K))=0 OR
M(1,M(PA+2,K))=0 THEN  RETURN
1510 LET FM=0: IF M(1,K)>10 THEN  LET FM
=FM+1
1520 IF M(1,M(PA+1,K))>10 THEN  LET FM=F
M+1
1530 IF M(1,M(PA+2,K))>10 THEN  LET FM=F
M+1
1540 LET PP=P+10*(M-1)
1550 LET FP=0: IF M(1,K)=PP OR M(1,M(PA+
1,K))=PP OR M(1,M(PA+2,K))=PP THEN  LET
FP=1
1560 IF M=1 AND FP=1 AND FM=0 THEN  LET
FT=1
1570 IF M=2 AND FP=1 AND FM=3 THEN  LET
FT=1
1580 RETURN
1800 LET A$=INKEY$: IF A$="" THEN  GOTO
1800
1810 IF A$="S" OR A$="s" THEN  POKE 6414
5,160: POKE 64862,15: STOP
1820 LET A=VAL(A$): IF A<1 OR A>9 THEN
GOTO 1800
1830 RETURN
2000 DATA 0,0,10,2,0,7,2,0,0,5,3,1,16,2,
0,0,15,0,2,25,2
2010 DATA 0,0,11,5,0,10,5,0,2,8,6,4,16,5
,0,0,14,0,5,22,5
2020 DATA 0,0,12,8,0,13,8,0,5,0,9,7,16,8
,0,0,13,0,8,19,8
2030 DATA 0,1,22,11,0,7,11,0,4,19,12,10,
10,11,0,7,16,0,11,13,11
2040 DATA 0,9,18,14,0,19,11,0,6,21,15,13
,22,11,0,3,24,0,14,25,11
2050 DATA 0,12,0,17,0,13,14,0,0,20,18,16
,16,14,0,13,0,0,17,19,14
2060 DATA 0,11,0,20,0,10,17,0,17,23,21,1
9,16,17,0,14,0,0,20,22,17
2070 DATA 0,10,0,23,0,7,20,0,20,0,24,22,
16,20,0,15,0,0,23,25,20
```

13
Pin Table

With a certain amount of skill, and a large amount of luck you can soon amass a huge score on the Pin table. This is a game for all members of the family, even the youngest. After a few turns at the table you will be compelled to try again to break your own record score.

How to play

When you first run the program, the screen stays blue for a few seconds while the computer transfers the details of the Pin table into memory. Then the Pin table is displayed on the left of the screen and the score panel on the right. The table has circular 'pins', which are red or green. Every time the ball bounces against one of these you score 200 points. The score panel shows you what scores you get. The table also has several pockets into which the ball may fall. The ball may bounce against the outside of the pocket, scoring 200. If it falls into a pocket from above, you score 500, 1000 or 1500, depending on the colour of the pocket.

You have 8 balls to play, the number played being shown on the score panel. When the display appears, the first ball is already bouncing from left to right at the top of the table. When you judge the moment to be right, press any key. The ball is then released and begins to run toward the bottom end of the table. From then on, just watch it bounce on the pins and, with luck, eventually run into a pocket.

As soon as the ball enters a pocket, or reaches the bottom of the table, another ball appears at the top, ready for you to play. When all 8 balls have been played, your record score for the session is displayed. The table is ready for the next game immediately, and the first ball of the next game bounces across the top.

Winning tactics

It is fun to try to release the ball so that it drops straight into the black pocket at the top of the table. But you are likely to get a better score if you let the ball run down the table, bouncing repeatedly against the pins. With practice, you will find that there are certain groups of pins that send the ball bouncing from one to the other many times, giving you a very large score. However, there is a random element built into the game, equivalent to the mechanical variations present in a real Pin table, which ensure that the ball cannot bounce to and fro for ever. Sometimes you may have a lucky break, but at other times the bouncing ceases almost as soon as it has begun.

Keying in

Check the DATA lines carefully after you have typed them in. An error there could cause the ball to shoot off in odd directions, and probably spoil the display. Note that several of the PRINT statements end with a semi-colon(;). If you leave out the semi-colon the display will probably be spoilt during the game.

Program design

20–110	initialising graphics and transferring DATA table to array T
120–380	setting up the display
390–410	beginning of playing loop; initialising variables
420–480	bouncing the ball across the top of the screen, until a key is pressed
490–680	moving the ball
690–730	displaying the score
800–840	subroutine for reading and displaying the score if the ball has fallen into a pocket
850–890	subroutine for displaying the score when the ball bounces against a 'pin' or pocket
900–930	subroutine to make bell sound
1000–1020	subroutine to switch off sound generator when sound has ended
2000–2230	DATA for the table

The program

```
10 REM ** PIN TABLE **
20 DIM T(24,18)
30 FOR J=1 TO 24
40 FOR K=2 TO 18
50 READ T(J,K)
60 NEXT
70 NEXT
80 GENPAT 1,129,0,60,126,126,126,126,60,
0
90 GENPAT 1,130,0,24,60,189,189,189,153,
126
100 GENPAT 1,131,60,66,189,165,165,189,6
6,60
110 VS 4: COLOUR 0,10: COLOUR 2,15: COLO
UR 4,10: CLS
120 FOR J=0 TO 23
130 CSR 0,J: PRINT "  ";: CSR 19,J: PRIN
T "  ";
140 NEXT
150 COLOUR 0,15: FOR J=0 TO 23: FOR K=2
TO 18
160 LET C=T(J+1,K)
170 IF C<100 THEN  GOTO 230
180 IF C=1500 THEN  COLOUR 1,1: CSR K,J:
 PRINT CHR$(130)
190 IF C=1000 THEN  COLOUR 1,13: CSR K,J
: PRINT CHR$(130)
200 IF C=500 THEN  COLOUR 1,4: CSR K,J:
PRINT CHR$(130)
210 IF C=200 THEN  COLOUR 1,8: CSR K,J:
PRINT CHR$(131)
220 IF C=100 THEN  COLOUR 1,2: CSR K,J:
PRINT CHR$(131);
230 NEXT : NEXT
240 LET P$="**********"
250 COLOUR 1,4: CSR 22,1: PRINT "PIN TAB
LE"
260 COLOUR 1,1: CSR 23,3: PRINT CHR$(130
);" 1500"
270 COLOUR 1,13: CSR 23,4: PRINT CHR$(13
0);" 1000"
280 COLOUR 1,4: CSR 23,5: PRINT CHR$(130
);"  500"
```

```
290 COLOUR 1,8: CSR 23,6: PRINT CHR$(131
);"  200"
300 COLOUR 1,2: CSR 23,7: PRINT CHR$(131
);"  200"
310 COLOUR 1,8: CSR 22,9: PRINT P$
320 COLOUR 1,4: CSR 22,11: PRINT "BALL
  1"
330 COLOUR 1,8: CSR 22,13: PRINT P$
340 COLOUR 1,4: CSR 22,15: PRINT "SCORE"
350 CSR 25,17: PRINT "000"
360 COLOUR 1,8: CSR 22,19: PRINT P$
370 COLOUR 1,4: CSR 22,21: PRINT "TOP"
380 CSR 25,22: PRINT "000"
390 LET S=0: LET SM=0: FOR J=1 TO 8
400 GOSUB 1000: COLOUR 1,4: CSR 29,11: P
RINT J
410 LET X=2: LET Y=0: LET VX=1: LET VY=1
: LET NX=3: LET D=0: LET SN=0
420 LET NX=X+VX
430 IF VX=1 THEN  CSR X,0: PRINT " ";CHR
$(129)
440 IF VX=-1 THEN  CSR X,0: PRINT CHR$(1
29);" "
450 LET X=NX: GOSUB 1000
460 IF X=18 AND VX=1 THEN  LET X=17: LET
 VX=-1: GOTO 480
470 IF X=1 THEN  LET X=2: LET VX=1
480 IF INKEY$="" THEN  GOTO 420 ELSE  CS
R X+1,0: PRINT " "
490 LET P=T(Y+1,X)
500 LET RN=INT(RND*6)
510 IF P=12 THEN  GOSUB 800: GOTO 680
520 IF P=15 THEN  CSR X,Y: PRINT " ";: G
OSUB 900: GOTO 680
530 IF P=0 AND VY=-1 AND RN<4 THEN  LET
VY=1: GOTO 620
540 IF P=3 OR P=7 THEN  LET VX=0: GOTO 6
20
550 GOSUB 1000: IF P=4 AND RN>2 AND VX=0
 THEN  LET VX=4-RN
560 IF P=5 OR P=6 THEN  LET D=1
570 IF D=1 AND VY=1 AND RN>3 THEN  LET V
X=0: GOTO 620
580 IF P=1 OR P=5 OR P=9 THEN  LET VX=-1
590 IF P=2 OR P=6 OR P=10 THEN  LET VX=1
```

```
600 IF P>3 AND P<7 THEN  LET VY=-1
610 IF P>7 AND P<11 THEN  LET VY=1
620 GOSUB 850
630 LET NX=X+VX: LET NY=Y+VY
640 CSR X,Y: PRINT " ";
650 CSR NX,NY: PRINT CHR$(129);
660 LET X=NX: LET Y=NY: LET D=0
670 GOTO 490
680 NEXT
690 IF SM<S THEN  LET SM=S
700 CSR 25,22: PRINT SM
710 GOSUB 900: GOSUB 900
720 CSR 22,17: PRINT "   000   "
730 GOTO 390
800 CSR X,Y: PRINT " ";
810 LET SN=T(Y+2,X)
820 LET S=S+SN
830 CSR 25,17: PRINT S
840 GOSUB 900: RETURN
850 IF X=2 OR X=18 OR Y=0 THEN  RETURN
860 LET SN=T(Y+1,X)
870 IF SN=0 THEN  RETURN
880 LET SN=200: GOSUB 820
890 RETURN
900 COLOUR 4,INT(RND*12+2)
910 SBUF 1: SBUF 2: SOUND 0,1000+SN,240,
0,-20,40,1
920 GOSUB 1000
930 RETURN
1000 IF PEEK(64082)<>PEEK(64086) THEN  R
ETURN
1010 SOUND 0,0,0
1020 RETURN
2000 DATA 10,8,8,8,8,8,8,8,8,8,8,8,8,8,8
,8,9
2010 DATA 2,0,0,0,0,0,0,5,12,6,0,0,0,0,0
,0,1
2020 DATA 2,0,0,0,0,0,0,1,1500,2,0,0,0,0
,0,0,1
2030 DATA 2,5,4,6,5,4,6,1,8,2,5,4,6,5,4,
6,1
2040 DATA 2,1,200,2,1,100,2,1,200,2,1,10
0,2,1,200,2,1
2050 DATA 2,9,8,10,9,8,10,9,8,10,9,8,10,
9,8,10,1
```

```
2060 DATA 2,0,0,0,0,5,12,6,0,5,12,6,0,0,
0,0,1
2070 DATA 2,0,0,0,0,1,500,2,0,1,500,2,0,
0,0,0,1
2080 DATA 2,0,0,5,4,0,8,0,4,0,8,0,4,6,0,
0,1
2090 DATA 2,0,0,1,100,2,0,1,200,2,0,1,10
0,2,0,0,1
2100 DATA 2,0,0,9,8,0,12,0,8,0,12,0,8,10
,0,0,1
2110 DATA 2,0,0,0,0,1,500,2,0,1,500,2,0,
0,0,0,1
2120 DATA 2,0,0,0,0,9,8,10,0,9,8,10,0,0,
0,0,1
2130 DATA 2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,1
2140 DATA 2,0,0,5,12,6,0,5,12,6,0,5,12,6
,0,0,1
2150 DATA 2,0,0,1,1500,2,0,1,1000,2,0,1,
1500,2,0,0,1
2160 DATA 2,0,0,9,8,10,0,9,8,10,0,9,8,10
,0,0,1
2170 DATA 2,5,4,6,00,0,5,4,7,4,6,0,0,5,4
,6,1
2180 DATA 2,1,200,2,0,0,1,100,3,100,2,0,
0,1,200,2,1
2190 DATA  2,9,8,10,0,0,9,8,8,8,10,0,0,9
,8,10,1
2200 DATA 2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,1
2210 DATA 3,4,7,12,7,4,7,12,7,12,7,4,7,1
2,7,4,3
2220 DATA 3,200,3,1000,3,100,3,1500,3,15
00,3,100,3,1000,3,200,3
2230 DATA 15,15,15,15,15,15,15,15,15,15,
15,15,15,15,15,15,15
```

14
Singalong

Words and music by the Memotech! To be truthful, you provide the words, but the Memotech puts them together in dozens of different ways. The limericks which the micro invents can be as amusing or as ridiculous as you like. They are even more hilarious if you key in the names of your friends and relatives. Play this for fun on your own, or get the party "singing along with the Memotech".

How to play

All you need to do is to type in the program and RUN it. You will first see the word 'SINGALONG' displayed and hear the theme tune. Then the Memotech displays a rhyming verse, made up from the words and phrases already included in the program. You are then asked 'Singalong?' If you want to sing the words to music, press key Y (for 'yes'). The tune is repeated, with the words still on the screen. After this, you are asked 'Again?'. If you press key Y again, the tune is repeated.

If you want a new verse, press any other key than Y and the computer will compose something new.

Keying in

The words and phrases used in the verses are found in lines 160 and 170, and also in the DATA statements of lines 3000–9000. Some of these have more than one space before and after the words or between the words. Take care to key exactly the right number of spaces, as given in the listing. Otherwise the display will be spoilt and may be difficult to read. The 'o' in line 160 has four spaces in front of it, and the large gaps near the beginnings of lines 6010 to 6050 each

consist of four spaces. If you type the whole word DATA, when keying in lines 3000 to 9000, you may find that the computer inserts an extra unwanted space before the first word or phrase. The way to avoid this is to use the abbreviation for DATA, which is 'DA.'. This is what line 3000 should look like on the Editor Screen, as you type it in:

```
3000 DA.fat,great,sly,bright
```

After you press RET, the line appears in the List Screen exactly as listed in this book. If you make a typing error with these lines and need to edit them, always change the word DATA to DA. to prevent the unwanted space being introduced.

Program design

10–40	initialising virtual screens and colours
50	displaying title
60–180	dimensioning string variables and reading them from the DATA lines
190–250	playing the tune
260–300	inviting repeat of the tune (skipped on first run of program)
310–330	selecting random numbers for picking the words and phrases
340–410	displaying the verse, line by line
420–450	inviting 'Singalong?'
1000–1010	subroutine to detect when each note of the tune has ended
2000–2040	DATA for the tune
3000–9000	DATA for the verse

The program

```
10 REM ** SINGALONG **
20 VS 4: CLS
30 CRVS 2,1,3,3,27,13,32: CRVS 3,1,3,20,
14,1,32
40 VS 2: COLOUR 2,9: CLS : COLOUR 0,9: C
OLOUR 1,4: COLOUR 4,10
50 CSR 9,5: PRINT "SINGALONG": LET Z=0
```

```
60 RESTORE 3000
70 DIM A$(4,6): FOR J=1 TO 4: READ A$(J)
: NEXT
80 DIM B$(4,8): FOR J=1 TO 4: READ B$(J)
: NEXT
90 DIM C$(3,8): FOR J=1 TO 3: READ C$(J)
: NEXT
100 DIM D$(3,8): FOR J=1 TO 3: READ D$(J
): NEXT
110 DIM E$(2,3,30): FOR K=1 TO 3: FOR J=
1 TO 2: READ E$(J,K): NEXT : NEXT
120 DIM F$(2,3,35): FOR K=1 TO 3: FOR J=
1 TO 2: READ F$(J,K): NEXT : NEXT
130 DIM G$(3,2,20): FOR K=1 TO 2: FOR J=
1 TO 3: READ G$(J,K): NEXT : NEXT
140 DIM H$(5,15): FOR J=1 TO 5: READ H$(
J): NEXT
150 DIM J$(6,6): FOR J=1 TO 6: READ J$(J
): NEXT
160 DIM X$(2,8): LET X$(1)="called ": LE
T X$(2)="    of "
170 DIM Y$(3,8): LET Y$(1)=" always ": L
ET Y$(2)=" never  ": LET Y$(3)=" seldom
"
180 DIM Z$(2,7): LET Z$(1)=" day": LET Z
$(2)=" night"
190 RESTORE 2000: SBUF 42
200 FOR J=1 TO 42
210 READ P,D
220 IF P=99 THEN  GOSUB 1000: SOUND 0,0,
0: GOTO 240
230 SOUND 0,P,160,0,-12,D*15,1
240 NEXT
250 GOSUB 1000: SOUND 0,0,0
260 IF Z=0 THEN  LET Z=1: GOTO 310
270 VS 3: COLOUR 2,7: CLS : COLOUR 0,7:
COLOUR 1,15: COLOUR 4,10
280 PRINT "Again?  ";
290 LET K$=INKEY$: IF K$="" THEN  GOTO 2
90
300 IF K$="Y" THEN  GOTO 190
310 LET RA=INT(RND*4)+1: LET RB=INT(RND*
4)+1: LET RC=INT(RND*3)+1: LET RD=INT(RN
D*3)+1
320 LET RE=INT(RND*2)+1: LET RF=INT(RND*
```

```
2)+1: LET RG=INT(RND*3)+1: LET RH=INT(RN
D*5)+1: LET RJ=INT(RND*6)+1
330 LET RX=INT(RND*2)+1: LET RY=INT(RND*
3)+1: LET RZ=INT(RND*2)+1
340 VS 2: CLS
350 PRINT "There was a ";A$(RA);" ";B$(R
B)
360 PRINT X$(RX);: IF RX=1 THEN  PRINT C
$(RC);"," ELSE  PRINT D$(RD);","
370 PRINT : PRINT "Who ";Y$(RY);: IF RX=
1 THEN  PRINT E$(RE,RC);"," ELSE  PRINT
F$(RF,RD);","
380 PRINT : PRINT "Then ";G$(RG,RZ);","
390 PRINT : PRINT "One ";H$(RH);" ";Z$(R
Z);","
400 PRINT : PRINT "That ";J$(RJ);" ";A$(
RA);" ";B$(RB)
410 PRINT X$(RX);: IF RX=1 THEN  PRINT C
$(RC);"!" ELSE  PRINT D$(RD);"!"
420 VS 3: COLOUR 2,7: CLS : COLOUR 0,7:
COLOUR 1,15: COLOUR 4,10
430 PRINT "Singalong?  ";
440 LET K$=INKEY$: IF K$="" THEN  GOTO 4
40
450 IF K$="Y" THEN  GOTO 190 ELSE  GOTO
310
1000 IF PEEK(64082)<>PEEK(64086) THEN  G
OTO 1000
1010 RETURN
2000 DATA 3822,1,2863,2,2863,1,2863,1,28
63,2,3034,1,3405,1,3034,2,2551,3,99,0
2010 DATA 3822,1,2551,2,2551,1,2551,1,25
51,2,2863,1,3034,1,2863,2,2272,3,99,0
2020 DATA 2272,1,1911,2,1911,1,1911,1,19
11,2,99,0
2030 DATA 2272,1,2145,2,2145,1,2145,1,21
45,2,99,0
2040 DATA 2551,1,2272,2,1911,1,2272,1,21
45,2,2272,1,2551,1,2272,2,2863,4,99,0
3000 DATA fat,great,sly,bright
4000 DATA writer,loafer,artist,beauty
5000 DATA Nelly,Parkin,Neville
5010 DATA Luton,Ely,Norwich
6000 DATA   threw bricks at  the telly,a
te ice cream  and jelly
```

```
6010 DATA  set all the    dogs barking,w
as fooling and larking
6020 DATA  said "I'm a    real devil",wa
s quite on   the level
6030 DATA  went huntin'   and shootin',p
layed Chess   with his boots on
6040 DATA  said "Is it    true - really?
",had eyes that  were steely
6050 DATA  ate mustard    with porridge,
lacked vigour  and courage
7000 DATA spent all his pay,fell in the
bay,joined the AA
7010 DATA died of sheer fright,put out t
he light,looked such a sight
8000 DATA terrible,cold frosty,ill fated
,fabulous,memorable
9000 DATA stupid,clever,lazy,crafty,evil
,pretty
```

Variations

The tune can be changed by altering lines 2000 to 2040. Each note is stored there as two numbers, representing pitch and duration, in that order. The duration of each note is indicated by either 1, 2, 3, or 4. These values are multiplied by 15 when used in the SOUND statement of line 230. Silence is indicated by a pitch value of 99. If your tune has a different number of notes (including the 'silent notes' at the end of each line), alter the value in line 90 according to the number of notes in your replacement tune. This makes the right amount of space available in the sound buffer. Also alter the value in line 200, so that the correct number of notes is played.

Most of the fun with this program comes from making up your own words and rhyming phrases. These are typed in as DATA lines, following the scheme adopted in the listing:

Line 3000: A list of words describing a person, each with one syllable, and not beginning with a vowel. The numbers and the words must all be separated by commas. Do not type spaces between the words. If you have more words than can go on one DATA line (4 Editor Screen lines) continue the list by inserting lines 3010, 3020, 3030, etc. The same applies to the other lists described below. The '4' in 'DIM A$(4,6)' in line 70, refers to the number of words in line 3000. If you have a different number of words, alter the '4' to the

number of words you have in line 3000. Similarly, alter the '4' in the 'FOR J=1 TO 4', also in line 70. The '6' in line 70 refers to the maximum number of letters that each word can have. If you want to use longer words, increase the '6'. If, when you run your version of the program, the Memotech stops and displays line 70 with the message 'No space', it probably means that you have included a word that is too long to fit into the A$ array. Just edit the line to increase the second figure in brackets after A$. These remarks also apply to other lines described below: you can alter the number of letters allowed as described above.

Line 4000: A list of 'kinds of person', each consisting of two syllables. It is best for the accent to be on the first syllable, but you can often 'force' a word to fit in, especially when singing along. The maximum number of letters, including spaces allowed is eight. You can increase this by altering the '8' in line 80, to a higher value.

Line 5000: A list of names of people, followed in line 5010 by a list of names of places. You must have the same number of names in both lists. The names should all have two syllables, with the accent on the first. The names of people are stored in array C$(3,8), and the names of places are stored in array D$(3,8). As explained above, alter the values in lines 90 and 100 if you add more names or require longer names.

Line 6000: Phrases which describe what the person did. These rhyme with the names of people and places in line 5000. Their rhythm is 'di-DA-di-di-DA-di'. You can have as many phases to rhyme with each name as you like, but you must have the same number of rhyming lines for each name. Phrases which rhyme with names of people are stored in array E$(2,3,30), while those rhyming with names of places are stored in array F$(2,3,35). The numbers in brackets refer to:

Number of phrases to rhyme with each name (2)
Number of names of people or places in lines 5000 and 5010 (3)
Number of letters and spaces allowed (30 or 35)

Alter lines 110 and 120 as necessary.

Line 7000: Two sets of rhyming lines about what a person did. The first set rhymes with 'day' while the second set rhymes with 'night'. The rhythm for these is 'DA-di-di-DA'. These phrases are stored in array G$(3,2,20). The numbers mean:

Number of phrases rhyming with 'day' or with 'night' (3)
2 – do not alter this
Number of letters and spaces allowed (20)

Line 8000: A list of words describing a day or night: three syllables, with the accent on the first. Stored in array H$(5,15), on line 140.

Line 9000: A list of words describing a person; two syllables, with the accent on the first. Stored in array J$(6,6), on line 150.

Remember that, if you alter the numbers of words or phrases, you *must* alter the values in the DIM statements, and in the FOR...NEXT loops in the same program lines. You must also alter the values used in choosing the random numbers (lines 310 and 320, but not 330). RA, RB, RC ... to RJ are the numbers used in choosing words and phrases from arrays A$, B$, C$ ... to J$. Alter the number in the corresponding RND statement to make it equal the *first* value in brackets in the DIM statement.

If you find the above rather complicated, just try substituting a few words and phrases of your own for those in lines 3000 to 9000. You will soon discover how the program operates and can then attempt more extensive changes.

15
Computer Clues

This is a computer version of what is possibly one of the most popular games ever invented. The computer sets you a problem and you have to find the answer, helped by clues given by the computer. This version allows you to set the level and type of challenge to suit yourself, and to increase and extend it as you become more proficient.

How to play

The idea of the game is that, unseen by you, the computer picks out a set of symbols, or 'pieces'. When the game begins, you are asked to say how many pieces there are to be. These can be from 3 to 6. You are asked how many colours there are to be. Each piece can be in any one of from 3 to 8 colours. You are asked how many shapes. There can be up to 4 different shapes (square, circle, triangle, cross). If you decide to have only one shape, this is displayed as a square. With only 3 pieces of 3 colours and all of them of 1 shape there are 27 possible combinations, but with 6 pieces of 8 colours and 4 shapes the number rises to 1073741800.

The computer now chooses a set of pieces, within the limits you have set, and your task is to guess the colour and shape of each piece.

After this the screen clears and you are asked to make 'GUESS NO. 1'. The first question asked is 'COLOUR?'. In reply to this, press one of the Function keys (keys F1 to F8) on the extreme right of the keyboard. The colours are 1=black;2=green;3=blue;4=cyan (turquoise blue);5=light red(orange);6=yellow;7=purple;8=white. A column of numbers displayed on the right of the screen shows you which colour corresponds to which key. If you have asked for 8 colours, you can press any one of the 8 keys, but if, for example, you have asked for only 4 colours, then only the first 4 keys (F1 to F4)

will operate. As soon as you press a key, a square of the corresponding colour appears near the bottom of the screen. What you do next depends on whether you have asked for 1 shape or more than 1 shape.

If you have asked for 1 shape, the 'COLOUR?' questions remains, and you type in the next colour. Carry on like this until you see the messge 'OK? Y/N'. This comes when you have keyed in all the pieces and your set is complete. If the set is as you want it, press key 'Y'. If not, press 'N' and start again. When you press 'Y', the set of symbols disappears from the bottom of the screen, and reappears near the top, on the left. The computer now compares your set with its own set. This takes a second or two, after which you will see a set of coloured markers at the top right of the screen, level with your set of pieces. The colours of the markers have these meanings:

Blue: 1 piece of correct colour in the correct position
Red: 1 piece of correct colour, but not in the correct position

Note that the markers are always displayed in the order 'blue-red' as listed above. This has nothing to do with the position of the pieces in the set. For example, a blue marker on the extreme left means that one of the pieces (but not necessarily the one on the extreme left of your set) is of the correct colour and is in its correct position in the set. The markers are the clues as to how correct your guesses have been. If you are really unlucky, you may get no markers at all, showing that you have chosen shapes and colours completely different to those chosen by the computer. If you get one or more blues, you are well on the way toward winning.

Now you are asked to make "GUESS NO. 2". This is done the same way as the previous guess. The computer's set of pieces is still the same as before, so you should use the clues from the first guess to make a better guess this time. When you have keyed in the second set and approved it, it is displayed below the first set, and you see a second set of markers to the right. You are allowed 16 guesses altogether. If any of these guesses consists of a set of pieces exactly like that chosen by the computer, you have won. The display informs you of this. If, after 16 turns, you have not managed to find a set exactly like that chosen by the computer, the display tells you that the Memotech has won. As the game ends, the set chosen by the computer is displayed at the top of the screen.

If you decide to have more than 1 shape, the play is a little more complex. At each guess, you are asked to key in the colour, as before. A square of that colour appears at the bottom of the screen.

Then you are asked 'SHAPE?'. Key in any one of the shapes by pressing the appropriate function key. The keys and shapes allowed are listed at the bottom right corner of the screen. When you press the key, the displayed square changes to the corresponding shape. Then the question 'COLOUR?' is repeated, for the second piece of the set. In this way you key in both colour and shape for each piece. When you have the full number of pieces you are asked to confirm them by typing 'Y', as usual.

With more than 1 possible shape, the clue markers can be in one of 6 colours:

Yellow: 1 piece of correct colour and shape in the right position
Blue: 1 piece of the correct colour (but wrong shape) in the right position
Green: 1 piece of the correct shape (but wrong colour), in the right position
Cyan (bluish-green): 1 piece of the correct shape and colour, but in the wrong position
Red: 1 piece of the correct colour, but wrong in shape and position
Purple: 1 piece of the correct shape, but wrong in colour and position

Obviously it takes a lot more thinking to work out how to interpret the clues, but this is all part of the fascination of the colour-plus-shape version. As in the colour-only version, you have 16 guesses, after which the computer displays its chosen set.

Winning tactics

Analysing the clues is a complex task, especially when many shapes and colours are involved. It is too elaborate to go into here. The best advice is to be systematic, keying in sets all of one colour or shape, then varying the next set according to the clues obtained. Change only one or two pieces at each successive guess, then note how the clues alter. That said, there still remains something in favour of letting your inspiration take over occasionally!

Keying in

There are 8 spaces in the string in line 450. Note the space after the question mark in line 370, and the semi-colons in lines 810 and 1010.

Program design

10–80	initialising arrays and patterns
90–150	asking for numbers of pieces, colours and shapes
160–230	displaying lists of colours and shapes available
240–280	making computer's random choice of pieces
290–300	ready to start
310–360	asking for colour
370–420	asking for shape
430–470	asking set to be confirmed
480	displaying completed set at top of screen
490–520	looking for pieces which match in colour and/or shape in correct positions
530–600	looking for pieces which combine correct colour and shape
610–690	looking for pieces which are correct in colour or shape
700–750	looking for a win by player
760–770	clearing arrays and return for next guess
780–820	computer wins
830–860	clearing arrays, ready for next game
1000–1020	subroutine to print set of symbols
2000–2060	subroutine for clearing arrays

The program

```
10 REM ** COMPUTER CLUES **
20 DIM C(6,3)
30 DIM P(6,4)
40 GENPAT 1,129,0,126,126,126,126,126,12
6,0
50 GENPAT 1,130,0,60,126,102,102,126,60,
0
60 GENPAT 1,131,0,64,96,112,120,124,126,
0
70 GENPAT 1,132,0,24,24,126,126,24,24,0
80 GENPAT 1,133,0,126,126,24,24,126,126,
0
90 VS 5: CLS
100 CSR 3,3: INPUT "PIECES? (3-6) ";NP$
110 LET NP=VAL(NP$): IF NP<3 OR NP>6 THE
N  GOTO 100
```

```
120 CSR 3,5: INPUT "COLOURS? (3-8) ";NC$
130 LET NC=VAL(NC$): IF NC<3 OR NC>8 THE
N  GOTO 120
140 CSR 3,7: INPUT "SHAPES? (1-4) ";NS$
150 LET NS=VAL(NS$): IF NS<1 OR NS>4 THE
N  GOTO 140
160 VS 4: COLOUR 0,6: COLOUR 2,6: COLOUR
 4,6: CLS
170 FOR J=0 TO NC-1
180 COLOUR 1,1+J*2: CSR 29,6+J
190 PRINT J+1
200 NEXT
210 COLOUR 1,15: FOR J=1 TO NS
220 CSR 20,18+J: PRINT J;" = ";CHR$(128+
J)
230 NEXT
240 FOR J=1 TO NP
250 LET C(J,1)=INT(RND*NC)*2+1
260 LET C(J,2)=INT(RND*NS)+1
270 LET P(J,3)=0: LET P(J,4)=0
280 NEXT
290 LET J=0
300 COLOUR 1,15: CSR 3,1: PRINT "GUESS N
O. ";J+1
310 FOR K=1 TO NP
320 PAUSE 1000: CSR 3,19: COLOUR 1,15: P
RINT "COLOUR?"
330 LET A$=INKEY$: LET A=ASC(A$)
340 IF A<128 OR A>127+NC THEN  GOTO 330
350 LET P(K,1)=(A-127)*2-1
360 CSR 6,21: GOSUB 1000: IF NS=1 THEN
GOTO 420
370 PAUSE 1000: CSR 3,19: COLOUR 1,15: P
RINT "SHAPE? "
380 LET A$=INKEY$: LET A=ASC(A$)
390 IF A<128 OR A>127+NS THEN  GOTO 380
400 LET P(K,2)=A-127
410 CSR 6,21: GOSUB 1000
420 NEXT K
430 PAUSE 1000: CSR 3,19: COLOUR 1,15: P
RINT "OK? Y/N"
440 LET A$=INKEY$: IF A$="" THEN  GOTO 4
40
450 CSR 6,21: PRINT "          "
460 IF A$="Y" THEN  GOTO 480
```

```
470 FOR K=1 TO NP: LET P(K,2)=0: NEXT :
GOTO 310
480 CSR 6,J+3: LET K=NP: GOSUB 1000
490 FOR K=1 TO NP
500 IF P(K,1)=C(K,1) THEN  LET P(K,3)=1:
 LET P(K,4)=5: LET C(K,3)=1
510 IF P(K,2)=C(K,2) THEN  LET P(K,3)=1:
 LET C(K,3)=1: LET P(K,4)=P(K,4)+3: IF P
(K,4)=8 THEN  LET P(K,4)=11
520 NEXT
530 FOR K=1 TO NP
540 IF P(K,3)=1 THEN  GOTO 600
550 LET PC=P(K,1): LET PS=P(K,2)
560 LET M=1
570 IF C(M,3)=1 THEN  GOTO 590
580 IF PC=C(M,1) AND PS=C(M,2) THEN  LET
 P(K,3)=1: LET P(K,4)=7: LET C(M,3)=1: G
OTO 600
590 LET M=M+1: IF M<NP+1 THEN  GOTO 570
600 NEXT
610 FOR K=1 TO NP
620 IF P(K,3)=1 THEN  GOTO 690
630 LET PC=P(K,1): LET PS=P(K,2)
640 LET M=1
650 IF C(M,3)=1 THEN  GOTO 680
660 IF PC=C(M,1) THEN  LET P(K,3)=1: LET
 P(K,4)=9: LET C(M,3)=1: GOTO 690
670 IF PS=C(M,2) THEN  LET P(K,3)=1: LET
 P(K,4)=13: LET C(M,3)=1: GOTO 690
680 LET M=M+1: IF M<NP+1 THEN  GOTO 650
690 NEXT
700 LET X=0: LET FW=0: FOR K=13 TO 3 STE
P -2
710 FOR M=1 TO NP
720 IF K=P(M,4) THEN  CSR 18+X,3+J: COLO
UR 1,K: PRINT CHR$(133): LET X=X+1: LET
FW=FW+K
730 NEXT : NEXT
740 LET FF=11: IF NS=1 THEN  LET FF=5
750 IF FW=FF*NP THEN  COLOUR 1,15: CSR 3
,19: PRINT "YOU WIN!!": GOTO 790
760 GOSUB 2000
770 LET J=J+1: IF J<16 THEN  GOTO 300
780 COLOUR 1,15: CSR 3,19: PRINT "MEMOTE
CH WINS"
```

```
790 CSR 18,1
800 FOR M=1 TO NP
810 COLOUR 1,C(M,1): PRINT CHR$(C(M,2)+1
28);
820 NEXT
830 IF INKEY$="" THEN  GOTO 830
840 GOSUB 2000
850 FOR K=1 TO NP: LET C(K,2)=0: NEXT
860 GOTO 90
1000 FOR M=1 TO K
1010 COLOUR 1,P(M,1): PRINT CHR$(P(M,2)+
128);
1020 NEXT : RETURN
2000 FOR K=1 TO NP
2010 FOR M=1 TO 4
2020 LET P(K,M)=0
2030 NEXT
2040 LET C(K,3)=0
2050 NEXT
2060 RETURN
```

16
Snorkel

The shallow waters of the tropical bay of Santa Memotecnos are a pearl-driver's dream. The bottom of the bay is almost entirely covered with pearl oysters, their pinkish shells shimmering in the sub-aquatic sunlight. Seagulls soar overhead, crabs scamper among the oysters and shoals of tiny shimmering fish scurry by. But beware, for these tempting waters are hazardous! There are the giant clams, ready to snap shut on the outstretched hand of an unwary diver. The scorpion fish have sharp spines which inflict a painful wound. The Portuguese Man-of-War can paralyse a swimmer who is unlucky enough to come into contact with its dangling tentacles. The octopus darts upward then drifts down, its suckers inflicting painful injuries. Finally, there is the Sea Serpent which appears when least expected, with devastating results should the swimmer happen to be near at the time. As may be guessed from the above account, this program makes good use of the Memotech's sprite graphics.

How to play

When you first RUN the program, the screen displays a view of part of the bay, with you in your cruiser on the water surface. Also on the surface is a Portuguese Man-of-War, with tentacles trailing in the water. At the bottom of the screen you see the oysters, with pinkish shells. Among them are five giant clams. Three scorpion fish swim continuously from right to left at different depths and at different speeds. Although you may not see it at first, the huge Sea Serpent is there too, ready to catch you. The crabs, the shoals of tiny fish and the placid grouper fish (which swim from left to right) will do you no harm.

Figures at the top left corner of the screen display the time, in minutes and seconds. You have 5 minutes in which to collect as

many oysters as you car.. There are only two keys to press. These are the 'up-arrow' and 'down-arrow' keys situated in the keypad to the right of the main keyboard. Press the 'up-arrow' key to swim to the surface or the 'down-arrow' key to dive.

To start, press the 'down-arrow' key. The diver jumps into the water and swims down. There is no need to hold a key down or to press it repeatedly. Once you have pressed the 'down-arrow' key you continue to dive until you press the 'up-arrow' key. Then you rapidly twist around and head for the surface. Sideways motion comes from the tide, which occasionally carries you toward the right. There is wrap-around on the screen both for the fish and diver so, if you drift off screen at the right you re-appear on the left. This is the only way of reaching the oysters which are directly beneath the boat.

If you dive down and touch an oyster, the program automatically picks up the oyster, makes you turn round, swim to the surface and jump into the boat. At the same time the figure at the top right of screen increases by one, keeping count of how many oysters you have collected. The keys have no effect while this is happening. Time your collecting carefully, for, on the way up, you may head directly into the tentacles of a Man-of-War! It inflicts such serious injury that the game ends. While at the sea floor you may accidentally touch a clam instead of an oyster. If you do, the clam snaps shut, gripping your hand. The keys are out of action while you are struggling to escape. After about 8 seconds you manage to release yourself. All this takes time which could be better spent in collecting oysters. The scorpion fish are another hindrance, for contact with them makes it necessary to go back to the boat to remove their painful spines from your skin. If you dive down on to a scorpion fish, you immediately turn and swim back to the boat. This too is automatic. Once there, you have to wait for about 6 seconds (from the time of contact) before being able to dive again. The border of the screen turns grey at this time (as it does when you are held by a clam) to indicate that you cannot move. When the border goes red again, the keys are active once more. The octopus appears from the right. It drifts slowly down, then darts upwards. Contact with the octopus is very serious; you are forced to surface and to wait in the cruiser for 10 seconds. The Sea Serpent makes occasional surprise visits, and may appear almost anywhere at the bottom of the screen. If you are caught by the Serpent, the game ends. A final score of 4 oysters is good for a beginner, and it would take skilful swimming and a certain amount of luck with the tides to score more than 10. Press the space bar to play again.

Winning tactics

Diving as soon as you can is obviously essential, but it is a waste of time to dive straight onto the fish or octopus for you are forced to return to the boat and wait. Try to time your dive so as to go between the fish. Ideally, you should avoid them on the way up, too. By alternately pressing the two keys you can hover in the water waiting for the fish to pass. Be wary when near the sea bed, for a sudden tidal flow could carry you straight on to a clam.

To gain a large score, you will have to drift across the screen until you have reappeared at the left, under the boat.

The main point to consider is whether it is better to avoid the fish and octopus or to ignore them. To avoid them, by swimming up and down until they pass, takes extra time. To ignore them adds the risk of making contact and so having to waste time in the boat.

Keying in

Almost half of the program is taken up with producing the patterns for the sprites and graphics characters. Take care with the values or the picture will look odd.

Program design

20	declaring some of the variables
30–140	defining graphics characters
150–790	defining patterns for sprites
800–860	setting up the display
870–1080	initialising the sprites
1090	starting the clock and setting variables
1100–1110	displaying time and detecting 'time-up'
1120–1130	getting input and setting various flag variables
1140–1150	moving the diver off and on to the boat
1160–1250	moving the diver in the water, then checking for contact with clams and oysters and acting on the results
1260	counting down time diver is inactivated
1270–1370	moving fish and checking for contact with diver
1380–1420	moving octopus and checking for contact with diver
1430–1460	moving Man-of-War and checking for contact with diver

1470–1480	activating serpent and checking for contact with diver
1490	repeat loop
1500–1550	displaying final messages
1560–1570	preparing for replay
3000–3020	subroutine for moving diver down
4000–4020	subroutine for moving diver up

Points of interest

The display uses 22 sprites and several special characters. These are 16×16 sprites and displayed at magnification 2.

The program

```
10 REM ** SNORKEL **
20 LET AF=232: LET BF=232: LET CF=232: L
ET TX=232: LET TY=127: LET PC=-1: LET PX
=204: LET SX=0
30 CTLSPR 2,22: CTLSPR 0,1: CTLSPR 3,22:
 CTLSPR 5,22: CTLSPR 6,2
40 GENPAT 1,129,24,60,189,219,102,60,60,
24
50 GENPAT 1,130,56,56,24,56,123,255,255,
254
60 GENPAT 1,131,16,24,12,198,99,51,31,15
70 GENPAT 1,132,56,124,62,63,31,31,15,15
80 GENPAT 1,133,8,8,28,8,8,8,8,8
90 GENPAT 1,134,8,8,9,9,9,255,127,63
100 GENPAT 1,135,192,128,255,85,255,255,
255,255
110 GENPAT 1,136,0,0,0,0,3,255,255,254
120 GENPAT 1,137,31,15,0,0,0,0,0,0
130 GENPAT 1,138,255,255,0,0,0,0,0,0
140 GENPAT 1,139,253,253,0,0,0,0,0,0
150 GENPAT 4,1,48,48,48,208,208,208,80,1
20
170 GENPAT 5,1,24,24,63,41,45,45,33,16
180 GENPAT 6,1,0,0,0,0,0,0,0,0
190 GENPAT 7,1,0,0,0,0,0,0,0,0
200 GENPAT 4,2,48,48,16,56,92,154,154,15
4
```

```
210 GENPAT 5,2,90,40,68,36,52,54,54,6
220 GENPAT 6,2,0,0,0,0,0,0,0,0
230 GENPAT 7,2,0,0,0,0,0,0,0,0
240 GENPAT 4,3,0,0,00,63,103,247,255,3
250 GENPAT 5,3,63,0,0,0,0,0,0,0
260 GENPAT 6,3,2,6,14,254,255,255,255,25
3
270 GENPAT 7,3,64,96,112,120,28,6,0,0
280 GENPAT 4,4,0,0,0,0,128,255,255,192
290 GENPAT 5,4,192,64,0,0,0,0,0,0
300 GENPAT 6,4,1,3,7,6,14,254,252,62
310 GENPAT 7,4,62,14,2,3,1,1,0,0
320 GENPAT 4,5,79,159,159,159,159,159,12
5,20
330 GENPAT 5,5,36,40,72,72,133,133,132,1
30
340 GENPAT 6,5,198,225,113,114,242,244,2
52,80
350 GENPAT 7,5,72,36,34,145,81,9,210,49
360 GENPAT 4,6,66,66,65,129,129,64,64,12
8
370 GENPAT 5,6,140,82,73,33,34,68,136,11
2
380 GENPAT 6,6,1,9,17,14,0,136,148,66
390 GENPAT 7,6,66,36,24,0,0,0,0,0
400 GENPAT 4,7,1,7,14,26,26,26,26,58
410 GENPAT 5,7,63,63,42,42,84,84,138,138
420 GENPAT 6,7,248,248,184,184,184,176,1
76,176
430 GENPAT 7,7,248,248,168,168,84,84,162
,162
440 GENPAT 4,8,81,81,42,42,69,69,138,138
450 GENPAT 5,8,84,84,42,42,74,74,148,148
460 GENPAT 6,8,84,84,168,168,84,84,138,1
38
470 GENPAT 7,8,69,69,137,137,82,82,137,1
37
480 GENPAT 4,9,32,33,226,28,0,4,8,31
490 GENPAT 5,9,17,21,31,31,29,13,7,3
500 GENPAT 6,9,192,32,17,14,0,64,32,240
510 GENPAT 7,9,16,88,252,254,114,97,193,
128
520 GENPAT 4,10,112,136,4,2,1,0,0,0
530 GENPAT 5,10,0,0,3,4,4,248,224,0
540 GENPAT 6,10,0,0,24,36,66,129,1,98
```

```
550 GENPAT 7,10,148,8,0,0,0,0,0,0
560 GENPAT 4,11,128,192,224,240,240,120,
120,56
570 GENPAT 5,11,63,57,120,240,240,224,19
2,128
580 GENPAT 6,11,31,15,2,21,42,85,170,85
590 GENPAT 7,11,128,255,255,255,127,127,
15,3
600 GENPAT 4,12,240,85,170,85,171,87,169
,85
610 GENPAT 5,12,13,253,254,255,255,255,2
55,248
620 GENPAT 6,12,0,128,224,248,252,62,63,
120
630 GENPAT 7,12,240,248,255,126,252,248,
96,56
640 GENPAT 4,13,0,2,1,2,0,22,15,22
650 GENPAT 5,13,0,176,121,176,1,0,0,0
660 GENPAT 6,13,5,195,229,192,0,0,44,30
670 GENPAT 7,13,44,0,96,240,96,11,7,11
680 GENPAT 4,14,128,203,135,11,0,44,30,4
4
690 GENPAT 5,14,0,176,120,176,5,3,133,0
700 GENPAT 6,14,0,0,128,22,15,22,0,0
710 GENPAT 7,14,88,60,88,0,128,192,128,0
720 GENPAT 4,15,3,7,63,63,125,127,250,32
730 GENPAT 5,15,96,96,192,192,112,120,72
,32
740 GENPAT 6,15,192,224,248,252,190,254,
95,4
750 GENPAT 7,15,6,6,3,3,14,30,18,4
760 GENPAT 4,16,0,0,0,0,0,0,0,1
770 GENPAT 5,16,103,255,63,31,15,7,0,0
780 GENPAT 6,16,6,31,61,56,120,120,240,2
24
790 GENPAT 7,16,192,192,255,255,255,199,
1,0
800 VS 4: COLOUR 0,4: COLOUR 1,1: COLOUR
 2,10: COLOUR 4,8: CLS
810 CSR 0,0: FOR J=1 TO 128: PRINT " ";:
 NEXT
820 COLOUR 0,7: FOR J=1 TO 608: PRINT "
";: NEXT
830 COLOUR 0,4: CSR 4,2: PRINT CHR$(133)
: CSR 4,3: PRINT CHR$(134);CHR$(135);CHR
$(130)
```

```
840 COLOUR 0,7: CSR 4,4: PRINT CHR$(137)
;CHR$(138);CHR$(139)
850 COLOUR 0,7: COLOUR 1,9: CSR 0,22: FO
R J=1 TO 32: PRINT CHR$(129);: NEXT
860 COLOUR 1,1: FOR J=0 TO 5: CSR 7+2^J,
22: PRINT CHR$(131);: NEXT
870 SPRITE 1,1,300,300,0,0,1
880 SPRITE 2,2,300,300,0,0,1
890 SPRITE 3,3,300,111,0,0,8
900 SPRITE 4,4,300,111,0,0,8
910 SPRITE 5,3,300,79,0,0,1
920 SPRITE 6,4,300,79,0,0,1
930 SPRITE 7,3,300,47,0,0,11
940 SPRITE 8,4,300,47,0,0,11
950 SPRITE 9,5,300,300,0,0,12
960 SPRITE 10,6,300,300,0,0,12
970 SPRITE 11,7,204,159,0,0,9
980 SPRITE 12,8,204,143,0,0,9
990 SPRITE 13,9,300,23,0,0,13
1000 SPRITE 14,10,300,23,0,0,13
1010 SPRITE 15,11,-500,127,10,0,3
1020 SPRITE 16,12,-484,127,10,0,3
1030 SPRITE 17,11,-500,39,25,0,10
1040 SPRITE 18,12,-484,39,25,0,10
1050 SPRITE 19,13,-400,95,15,0,15
1060 SPRITE 20,14,-384,95,15,0,15
1070 SPRITE 21,15,50,15,50,0,2
1080 SPRITE 22,16,500,183,-30,0,14
1090 CLOCK "000000": LET DD=0: LET DX=-1
: LET DY=151: LET T=-1: LET FC=0: LET FP
=0: LET P=0: LET FM=0: LET FS=0: LET FF=
0: LET FT=0
1100 COLOUR 0,4: COLOUR 1,15: CSR 2,0: P
RINT MID$(TIME$,4,1);".";RIGHT$(TIME$,2)
1110 IF VAL(MID$(TIME$,3,2))=5 THEN  LET
 FT=1: GOTO 1500
1120 LET A=ASC(INKEY$): IF A=10 AND T<0
AND FP=0 AND FF=0 THEN  LET DD=-8
1130 IF A=11 AND DD=-8 THEN  LET DD=8
1140 IF DD=-8 AND DX=-1 THEN  LET DX=64:
 LET DY=151: GOSUB 3000: COLOUR 0,4: COL
OUR 1,1: CSR 6,3: PRINT CHR$(136): GOTO
1160
1150 IF DD=8 AND DY=151 THEN  LET FP=0:
LET DX=-1: ADJSPR 1,2,0: COLOUR 0,4: COL
```

```
OUR 1,1: CSR 6,3: PRINT CHR$(130)
1160 IF FC=1 THEN  GOTO 1250
1165 IF DX=-1 THEN  GOTO 1210
1170 IF RND>.8 THEN  LET DX=DX+8: IF DX>
255 THEN  LET DX=8
1180 LET DY=DY+DD
1190 IF DD=-8 THEN  GOSUB 3000
1200 IF DD=8 THEN  GOSUB 4000
1210 IF DY>15 THEN  GOTO 1260
1220 CSR (DX-8)/8,22: LET B$=SPK$
1230 IF B$=CHR$(129) THEN  CSR (DX-8)/8,
22: COLOUR 0,7: PRINT " ": LET P=P+1: CS
R 28,0: COLOUR 0,4: COLOUR 1,15: PRINT P
: LET DD=8: LET FP=1
1240 IF B$=CHR$(131) THEN  CSR (DX-1)/8,
22: COLOUR 0,7: COLOUR 1,1: PRINT CHR$(1
32): LET T=30: COLOUR 4,14: LET FC=1
1250 LET DD=8
1260 LET T=T-1
1270 IF T>0 THEN  GOTO 1290
1280 LET FC=0: LET FF=0: COLOUR 4,8
1290 LET AF=AF-8: IF AF=-8 THEN  LET AF=
232
1300 ADJSPR 2,3,AF: ADJSPR 2,4,AF+16
1310 IF ABS(AF-DX)<9 AND ABS(111-DY)<9 T
HEN  LET T=25: LET DD=8: COLOUR 4,14: LE
T FF=1
1320 LET BF=BF-4: IF BF=-4 THEN  LET BF=
232
1330 ADJSPR 2,5,BF: ADJSPR 2,6,BF+16
1340 IF ABS(BF-DX)<9 AND ABS(79-DY)<9 TH
EN  LET T=25: LET DD=8: COLOUR 4,14: LET
 FF=1
1350 LET CF=CF-2: IF CF=-2 THEN  LET CF=
232
1360 ADJSPR 2,7,CF: ADJSPR 2,8,CF+16
1370 IF ABS(CF-DX)<9 AND ABS(47-DY)<9 TH
EN  LET T=25: LET DD=8: COLOUR 4,14: LET
 FF=1
1380 LET TX=TX-1: LET TY=TY-1
1390 IF TX=-1 THEN  LET TX=232
1400 IF TY=59 THEN  LET TY=127
1410 ADJSPR 2,9,TX: ADJSPR 3,9,TY: ADJSP
R 2,10,TX: ADJSPR 3,10,TY-16
1420 IF ABS(TX-DX+8)<9 AND ABS(TY+16-DY)
```

```
<9 THEN  LET T=40: LET DD=8: COLOUR 4,14
: LET FF=1
1430 LET PX=PX+PC: IF PX=8 OR PX=205 THE
N  LET PC=-PC
1440 ADJSPR 2,11,PX: ADJSPR 2,12,PX
1450 IF ABS(PX-DX+8)<9 AND DY>126 AND DY
<152 THEN  LET FM=1: GOTO 1500
1460 IF T=-100 THEN  LET SX=INT(RND*232)
: ADJSPR 2,13,SX: ADJSPR 2,14,SX+16: ADJ
SPR 1,13,13: ADJSPR 1,14,13: LET FS=1
1470 IF T=0 THEN  ADJSPR 1,13,0: ADJSPR
1,14,0: LET FS=0
1480 IF FS=1 AND ABS(SX-DX+8)<9 AND DY<2
4 THEN  LET FS=2: GOTO 1500
1490 GOTO 1100
1500 COLOUR 0,11: COLOUR 1,6: CSR 8,2
1510 IF FC=1 THEN  PRINT "Badly injured
by a clam": GOTO 1550
1520 IF FM=1 THEN  PRINT "Man-of-War stu
ng you": GOTO 1550
1530 IF FS=2 THEN  PRINT "Sea Serpent ca
ught you": GOTO 1550
1540 IF FT=1 THEN  PRINT "Time to rest"
1550 CSR 8,6: PRINT "but you got";P;" pe
arls"
1560 IF INKEY$<>" " THEN  GOTO 1560
1570 CLS : GOTO 800
2000 GOTO 1100
3000 ADJSPR 2,1,DX: ADJSPR 3,1,DY
3010 ADJSPR 1,2,0: ADJSPR 1,1,1
3020 RETURN
4000 ADJSPR 2,2,DX: ADJSPR 3,2,DY
4010 ADJSPR 1,1,0: ADJSPR 1,2,1
4020 RETURN
```

Variations

Alter the length of the game by changing the value '5' in line 1110 to the required length of time in minutes. You could add more sprites to increase the numbers of crabs, gulls or fish. Use the same patterns or design new ones of your own. But remember that the Memotech cannot handle more than 4 sprites on a line, so having too many

sprites may spoil the effect. There is plenty of scope for programming sound effects, such as the splash of the diver entering the water, the gentle puttering of the engines of the cruiser – not to mention the unimaginable sounds of the sea monsters!

17
Rail Runner

This is a game of strategy for 2 to 4 players. Your aim is to be the first to lay a railway line along a route selected by the computer and to run a train along it from start to finish.

How to play

Before running the program the players must decide in which order they are going to play. Each player has a special colour:

1st player – green
2nd player – blue
3rd player – red
4th player – yellow

The computer first asks how many persons are playing. Enter a number from 2 to 4 and press 'RETURN'. Now the computer chooses a route at random for each player and displays these. Points on the routes are referred to by the names of well-known towns. Each route begins at one town, passes through one town, and ends at a third town. Each player should write down the names of the towns on his or her route. When everyone has done this, press any key.

Each town appears on the screen as a coloured square with a cross inside, placed at random within certain areas of the screen:

	LEFT	MIDDLE	RIGHT
TOP	Ayr (black)	Derby (dk blue)	Grimsby (cyan)

MIDDLE	Burnley (dk green)	Everton (lt blue)	Halifax (med red)
BOTTOM	Cardiff (lt green)	Falkirk (dark red)	Ipswich (light red)

The colours of towns have nothing to do with the players' colours. The arrangement of towns on the screen is alphabetical, not geographical! Identify the towns on your route and, when you hear the 'toot', it is the first player's turn. The border of the screen always changes colour to show whose turn it is.

In the first stage of the game you are laying the track. The track begins at the first station and you extend it by pressing one of the direction keys of the keypad to the right of the main keyboard. Press the key which corresponds to the direction in which you wish to extend the track. As soon as you press the key, the track appears on the screen, displayed in your playing colour. The first piece of track laid is straight, coming from one of the four sides of the 'station' block. If you change direction, in a subsequent turn, the computer prints a curved track to take you in the required direction.

Each player builds track by pressing one of the keys when his or her colour is displayed on the border of the screen. Here are the rules for track-laying:

(i) On leaving a station, the track is always a straight piece (no corners): the computer is programmed to do this automatically.

(ii) No U-turns allowed: if you try to turn 'up' when the end of your track is directed 'down' for example, the computer gives you a second chance to press a key. Turn to the left (or right) and make a second turn to the left (or right) when you play again.

(iii) No track can be laid on squares occupied by another player, EXCEPT that if the other player's track is a straight track running across the way you want to go, the computer will print a '+' crossover for you.

(iv) When your track has reached a station, you need to make another key-press in your next turn, to actually enter the station.

(v) Having entered a station, you can leave it in any direction, not already taken by an existing track.

(vi) Not more than one line can go to each of the four sides of a station.

(vii) Do not lay your line through any station except the three on your list. The computer does not prevent you from doing this, but if

you go through a wrong station, then you will not be able to complete the second part of the game.

Depending on where the stations are located, and the routes allotted to the players, it may happen that a player's route is completely blocked by the lines of other players. The situation is so complex that the computer cannot ensure that this is avoided. Players may agree to stop the game and start again, or may prefer to accept that it is just 'hard luck' on the blocked player, who drops out of the game.

If you make a mistake, or for any other reason want to change your layout, you can remove pieces of track one at a time. When it is your turn, press the DEL (delete) key on the keyboard. The last portion of track is then removed and the turn passes to the next player. You may take up as many sections of track as you wish, as far back as the last station visited.

Scoring is done automatically. The running scores are displayed at the bottom of the screen, in each player's colour. Scoring is as follows:

For each straight track laid: 1
For crossing another player's track: −10
For laying track to the second station on the route: 10
For being the first train to arrive at its final station: 10

When a player has completed the line to the final station, the next stage of the game begins for that player. In this stage the players train is automatically moved along the track from start to finish. You will see the picture of a locomotive travelling along the track and hear its 'chuff-chuff-chuff' for each move it makes. A random number (from 1 to 3), displayed in the right bottom corner of the screen, decides how far the train moves in each turn.

The game ends when:

(i) the first train enters its final station, or
(ii) two trains collide at a '+' crossing.

The winner is the player with the highest score.

Winning tactics

These apply to the track-laying stage of the game. Long straight runs build up a good score. Plan ahead, to avoid having to lay non-scoring curves. Taking up track wastes time and carries a heavy

penalty in points. This is another reason for planning, and also for anticipating where other players are likely to lay their track. When approaching a station, remember that someone else may get there before you, forcing you to carry your track around the station to enter on the far side. The shorter the track, the more quickly will your train reach its destination.

Keying in

Beware of confusing brackets with 'greater than' or 'less than' symbols. Lines 220, 240 and 270 are examples where such confusion may arise.

Program design

20–60	initialising
70–140	patterns for station, track and locomotive
150–160	requesting number of players
170–330	setting up arrays and selecting routes
340–390	displaying routes
400	waiting to start
410–450	displaying stations and scores (initially zero)
460	initialising player number
470–510	re-entry point for beginning of each player's turn; whistle sound-effect
520–580	getting and analysing input from players
590–750	analysing intended move and, if valid, making it
760–800	dealing with a move when line reaches a station
810–970	routines for taking up track
980–1160	moving trains in 2nd part of game
1170–1180	end of game; collision
1190–1210	end of game; first train home
2000–2040	subroutine for displaying scores and 'dice'
3000	DATA for deciding which kinds of track are required

The program

```
10 REM ** RAIL RUNNER **
20 DIM P(4,12),S(3,9),D(4,4): LET SC=0:
LET RN=0: LET PN=1
30 DIM S$(9,7): LET U$=""
40 LET S$(1)="AYR": LET S$(2)="BURNLEY":
 LET S$(3)="CARDIFF"
50 LET S$(4)="DERBY": LET S$(5)="EVERTON
": LET S$(6)="FALKIRK"
60 LET S$(7)="GRIMSBY": LET S$(8)="HALIF
AX": LET S$(9)="IPSWICH"
70 GENPAT 1,129,255,153,153,255,255,153,
153,255
80 GENPAT 1,130,0,0,0,255,255,0,0,0
90 GENPAT 1,131,24,24,24,24,24,24,24,24
100 GENPAT 1,132,24,24,24,255,255,24,24,
24
110 GENPAT 1,133,0,0,0,224,240,56,24,24
120 GENPAT 1,134,24,24,56,240,224,0,0,0
130 GENPAT 1,135,24,24,28,15,7,0,0,0
140 GENPAT 1,136,0,0,0,7,15,28,24,24: GE
NPAT 1,137,10,224,34,63,255,254,255,54
150 VS 5: CLS : CSR 3,3: INPUT "How many
 players? (2-4) ";P$
160 LET P=VAL(P$): IF P<2 OR P>4 THEN  G
OTO 150
170 FOR J=1 TO 4: FOR K=1 TO 4: READ D(J
,K): NEXT : NEXT
180 FOR J=1 TO 7 STEP 3: FOR K=0 TO 2
190 LET S(1,J+K)=3+INT((J-1)/3)*10+INT(R
ND*8)
200 LET S(2,J+K)=2+K*7+INT(RND*5): NEXT
: NEXT
210 FOR J=1 TO P
220 LET P(J,1)=INT(RND*8)+1: IF S(3,P(J,
1))>3 THEN  GOTO 220
230 LET S(3,P(J,1))=S(3,P(J,1))+1
240 LET P(J,2)=INT(RND*8)+1: IF S(3,P(J,
2))>3 THEN  GOTO 240
250 IF P(J,1)=P(J,2) THEN  GOTO 240
260 LET S(3,P(J,2))=S(3,P(J,2))+2
270 LET P(J,3)=INT(RND*8)+1: IF S(3,P(J,
3))>3 THEN  GOTO 270
280 IF P(J,3)=P(J,2) OR P(J,3)=P(J,1) TH
```

```
EN  GOTO 270
290 LET S(3,P(J,2))=S(3,P(J,2))+1
300 LET P(J,4)=S(1,P(J,1))
310 LET P(J,5)=S(2,P(J,1))
320 LET P(J,12)=129
330 NEXT
340 CLS : CSR 2,2: PRINT "ROUTES"
350 CSR 2,4: PRINT "No.": CSR 6,4: PRINT
 "From:": CSR 14,4: PRINT "Via:": CSR 22
,4: PRINT "To:"
360 FOR J=1 TO P: CSR 2,J*3+3: PRINT J
370 FOR K=1 TO 3
380 CSR K*8-2,J*3+3: PRINT S$(P(J,K))
390 NEXT : NEXT
400 IF INKEY$="" THEN  GOTO 400
410 VS 4: COLOUR 0,15: COLOUR 2,15: COLO
UR 4,13: CLS
420 FOR J=1 TO 9: COLOUR 1,J
430 CSR S(1,J),S(2,J): PRINT CHR$(129)
440 NEXT
450 GOSUB 2000
460 LET PN=0
470 LET PN=PN+1
480 IF PN=P+1 THEN  LET PN=1
490 COLOUR 4,PN*3-1
500 SOUND 0,100,15: SOUND 3,4,15: PAUSE
1000: SOUND 0,0,0: SOUND 3,0,0
510 IF P(PN,9)=1 THEN  GOTO 980
520 LET A=ASC(INKEY$): IF A=-1 THEN  GOT
O 520
530 IF A=11 THEN  LET D=1: GOTO 590
540 IF A=25 THEN  LET D=2: GOTO 590
550 IF A=10 THEN  LET D=3: GOTO 590
560 IF A=8 THEN  LET D=4: GOTO 590
570 IF A=127 THEN  GOTO 810
580 GOTO 520
590 LET EX=P(PN,4): LET EY=P(PN,5): LET
CD=P(PN,6): CSR EX,EY: IF SPK$=CHR$(129)
 THEN  LET CD=D
600 LET NP=D(D,CD): IF NP=0 THEN  GOTO 5
20
610 IF CD=1 THEN  LET NX=EX: LET NY=EY-1
620 IF CD=2 THEN  LET NX=EX+1: LET NY=EY
630 IF CD=3 THEN  LET NX=EX: LET NY=EY+1
640 IF CD=4 THEN  LET NX=EX-1: LET NY=EY
```

```
650 IF NX<1 OR NX>32 OR NY<0 OR NY>21 TH
EN  GOTO 470
660 CSR NX,NY: LET Q$=SPK$: IF Q$=CHR$(1
29) THEN  GOTO 760
670 IF Q$=CHR$(131) AND NP=130 OR Q$=CHR
$(130) AND NP=131 THEN  LET NP=132: GOTO
 690
680 IF Q$<>" " THEN  GOTO 470
690 CSR NX,NY: COLOUR 0,15: COLOUR 1,PN*
3-1: PRINT CHR$(NP)
700 IF NP=132 THEN  LET SC=-10: GOSUB 20
00
710 IF NP=131 OR NP=130 THEN  LET SC=1:
GOSUB 2000
720 IF EX=S(1,P(PN,1)) AND EY=S(2,P(PN,1
)) THEN  LET P(PN,10)=D
730 IF EX=S(1,P(PN,2)) AND EY=S(2,P(PN,2
)) THEN  LET P(PN,11)=D
740 LET P(PN,4)=NX: LET P(PN,5)=NY: LET
P(PN,6)=D
750 GOTO 470
760 IF NX=S(1,P(PN,2)) AND NY=S(2,P(PN,2
)) AND P(PN,8)=0 THEN  LET P(PN,8)=1: LE
T SC=10: GOSUB 2000
770 IF NX=S(1,P(PN,3)) AND NY=S(2,P(PN,3
)) AND P(PN,8)=1 AND P(PN,9)=0 THEN  GOT
O 790
780 GOTO 740
790 LET P(PN,9)=1: LET P(PN,4)=S(1,P(PN,
1)): LET P(PN,5)=S(2,P(PN,1)): LET SC=0:
 GOSUB 2000
800 GOTO 470
810 LET EX=P(PN,4): LET EY=P(PN,5): CSR
EX,EY: LET U$=SPK$: IF U$=CHR$(129) THEN
  GOTO 470
820 IF U$=CHR$(132) THEN  GOTO 940
830 FOR J=1 TO 4
840 IF D(P(PN,6),J)=ASC(U$) THEN  LET D=
J
850 NEXT
860 LET P(PN,6)=D
870 COLOUR 0,15: CSR EX,EY: PRINT " "
880 IF D=1 THEN  LET PX=EX: LET PY=EY+1
890 IF D=2 THEN  LET PX=EX-1: LET PY=EY
900 IF D=3 THEN  LET PX=EX: LET PY=EY-1
```

```
910 IF D=4 THEN  LET PX=EX+1: LET PY=EY
920 LET P(PN,4)=PX: LET P(PN,5)=PY: LET
SC=-10: GOSUB 2000
930 GOTO 470
940 LET D=P(PN,6): COLOUR 0,15: COLOUR 1
,PN*3-1
950 IF D=1 OR D=3 THEN  CSR EX,EY: PRINT
 CHR$(130)
960 IF D=2 OR D=4 THEN  CSR EX,EY: PRINT
 CHR$(131)
970 GOTO 880
980 LET RN=INT(RND*3)+1: LET SC=0: GOSUB
 2000
990 FOR J=1 TO RN
1000 LET D=P(PN,10): LET EX=P(PN,4): LET
 EY=P(PN,5)
1010 IF EX=S(1,P(PN,2)) AND EY=S(2,P(PN,
2)) THEN  LET D=P(PN,11)
1020 IF D=1 THEN  LET NX=EX: LET NY=EY-1
1030 IF D=2 THEN  LET NX=EX+1: LET NY=EY
1040 IF D=3 THEN  LET NX=EX: LET NY=EY+1
1050 IF D=4 THEN  LET NX=EX-1: LET NY=EY
1060 CSR NX,NY: LET NP=ASC(SPK$): IF NP=
137 THEN  GOTO 1170
1070 COLOUR 0,15: COLOUR 1,PN*3-1: IF P(
PN,12)<>129 THEN  CSR EX,EY: PRINT CHR$(
P(PN,12))
1080 IF NP<>129 THEN  CSR NX,NY: PRINT C
HR$(137)
1090 IF NX=S(1,P(PN,3)) AND NY=S(2,P(PN,
3)) THEN  GOTO 1190
1100 LET P(PN,8)=0: FOR K=1 TO 4
1110 IF D(K,D)=NP THEN  LET P(PN,10)=K
1120 NEXT
1130 LET P(PN,4)=NX: LET P(PN,5)=NY: LET
 P(PN,12)=NP
1140 FOR G=1 TO 3: SOUND 3,6,15: PAUSE 1
50: SOUND 3,6,10: PAUSE 100: SOUND 3,0,0
: PAUSE 200: NEXT
1150 NEXT
1160 GOTO 470
1170 CSR 5,12: COLOUR 0,5: COLOUR 1,15:
PRINT "TRAINS COLLIDE!"
1180 GOTO 1180
1190 CSR 5,12: COLOUR 0,5: COLOUR 1,15:
```

```
PRINT "FIRST TRAIN ARRIVES"
1200 LET SC=10: GOSUB 2000
1210 GOTO 1210
2000 LET P(PN,7)=P(PN,7)+SC
2010 COLOUR 0,15
2020 FOR Q=1 TO P: COLOUR 1,Q*3-1: CSR Q
*6-4,22: PRINT "     ": CSR Q*6-4,22: PRI
NT P(Q,7): NEXT
2030 COLOUR 0,1: COLOUR 1,15: CSR 28,22:
 PRINT RN
2040 RETURN
3000 DATA   131,134,0,135,136,130,135,0,
0,133,131,136,133,0,134,130
```

Variations

If you want to give the game a more local flavour, alter the names of towns (lines 40–60) to ones of your own choosing. If any of the names you choose have more than 7 letters, alter the '7' in line 30 to the number of letters in the longest name.

18
Minefield

You are an escaped prisoner-of-war trying to make your way back to join your own troops. Between you and safety lies an enemy minefield, which you have to cross on foot. It is a sandy barren area, with only a few trees and bushes for cover. The enemy are aware of your escape and, since you are a high-ranking officer, they are out in force to prevent you from getting back alive.

The enemy has sent armoured vehicles to patrol the minefield, with orders to shoot on sight. Although they are a danger to you, they can also be a help. If you can keep out of sight and watch where the vehicles go, you will be able to discover the un-mined paths across the minefield and so reach safety. But put only one foot wrong and you will be blown sky-high by a mine. When you are in sight, the gunners will shoot you, but they cannot see you when you are hiding in a bush. On the other hand, armoured vehicles do not bother to avoid bushes; they simply drive over them. This could be unfortunate for you, if you happen to be hiding in the flattened bush at the time! Apart from that risk, the bushes are a safe place to seek, for there are no mines in or under a bush.

How to play

The game is played at 10 levels, of which the easiest is level 1. The higher the level, the more thickly scattered are the mines. Above level 5 the guns fire further. At the highest levels (8 and over) the vehicles destroy the bushes as they pass over them. If you take too long to cross the minefield, you may find that few places remain in which to hide.

As soon as you have keyed in your chosen level and pressed RET the screen clears, the mines are laid (though you cannot see them) and the green bushes are displayed in unmined locations. Then the

black vehicles of the enemy appear at the top of the screen. The circular red character at the top left corner is your red beret. Your destination is the bottom right corner. The whole of the top row and bottom row of the screen are clear of mines so, once you reach the bottom of the screen, it is safe to move along it (though you may still get shot by a passing vehicle).

With a roar as they accelerate, the vehicles move off into the minefield, travelling down the screen at first. When a vehicle finds a mine ahead, it turns right and then proceeds in a new direction. Vehicles also turn right when one is blocked by another. All the time you will hear the sounds of gunfire, and shells bursting beside you. You move by using the direction keys on the keypad to the right of the main keyboard. To move in any given direction, hold down the appropriate key until your move has been effected. If you press no key, you stay in the same place. You cannot move off the edge of the screen, but the vehicles can. When you are near one edge of the screen, it is essential to watch the opposite edge, for a vehicle may suddenly appear close to you and shoot you before you have had time to get clear. In levels 4 and lower, the range of the guns is 1 row or column. In other words, the vehicle has to be on the next screen location, vertically or horizontally. At levels 5 and over the range is extended to two rows or columns. When you are hiding in a bush, you cannot be shot but, if a vehicle runs over the bush you are hiding in, you are run over.

When the game ends you are told how long you took to get to safety. If you failed to reach your goal, you are told how long you managed to survive the hazards of the minefield.

Press the space-bar to play again.

Winning tactics

Give the vehicles a few moments to head through the minefield and watch where they go. At the lower levels there is a chance that there is a clear path straight across. Even then, watch out for deflected vehicles cutting across your path and shooting you as they pass. Remember that even though the display shows the guns pointing ahead, you will be shot whether you are ahead, to the side of the vehicle or behind it. If there is a bush close by, it is worth deviating from your path to hide until the vehicle has passed. Watch out for pairs of vehicles travelling on parallel paths, two or three rows or columns apart. Between them, their guns cover a wide area of the

minefield and it is difficult to move out of their way quickly. One of the worst dangers is the vehicle which goes off the screen on one side and instantly re-appears on the opposite side. If it goes off the left edge, it re-appears on the same row on the right of the screen. If it leaves the right edge it re-appears on the left. It is essential to take care when you have got to the bottom of the screen and are making your final dash for safety.

Keying in

There are no special problems.

Program design

20–30	initialising arrays
40–90	defining characters for soldier, bushes and tanks
100–110	asking for required level
120–250	setting up minefield, with bushes and soldier
260–270	initialising variables and time
280–310	displaying tanks
320	start of main loop; reading key-press
330–500	processing player's move
510–750	processing moves of vehicles, in turn
760–790	sound effects of gunfire and falling shells
800–820	'escaped' display
830–850	'shot' display
860–880	'run over' display
890–900	'mined' display
910–930	displaying time
940–960	making ready for next game

Points of interest

The mines are displayed on the screen as asterisks (character code 42) but since they are displayed in light yellow (colour code 11, line 120) they are invisible to the player. They are not invisible to the computer, which uses SPK$ to read the video RAM at lines 420 and 600 to find out what lies ahead of the player or a tank. It is useful to have 'visible' mines when checking to see that the program is running

properly. To do this, change the COLOUR 1,11 statement of line 120 to COLOUR 1,4. This makes the mines appear in blue.

The program

```
10 REM ** MINEFIELD **
20 DIM C(2,5),CN(2,5),CC(5),CD(5)
30 DIM M(2,55),S(2,20)
40 GENPAT 1,129,32,118,255,63,254,127,23
8,38
50 GENPAT 1,130,0,60,126,126,126,126,60,
0
60 GENPAT 1,131,40,40,56,124,124,124,124
,56
70 GENPAT 1,132,0,124,255,252,255,124,0,
0
80 GENPAT 1,133,56,124,124,124,124,40,40
,0
90 GENPAT 1,134,0,0,30,255,63,255,30,0
100 VS 5: CLS : CSR 3,3: INPUT "LEVEL? (
1-10) ";A$
110 LET L=VAL(A$): IF L<1 OR L>10 THEN
GOTO 100
120 VS 4: COLOUR 0,11: COLOUR 1,11: COLO
UR 2,11: COLOUR 4,11: CLS
130 FOR J=1 TO 25+3*L
140 LET M(1,J)=INT(RND*31)+1: LET M(2,J)
=INT(RND*21)+1: CSR M(1,J),M(2,J): PRINT
 "*";
150 NEXT
160 COLOUR 1,12: FOR J=1 TO 20
170 LET S(1,J)=INT(RND*31)+1: LET S(2,J)
=INT(RND*21)+1
180 LET FS=0: FOR K=1 TO 25+3*L
190 IF M(1,K)=S(1,J) AND M(2,K)=S(2,J) T
HEN  LET FS=1
200 NEXT
210 IF FS=1 THEN  GOTO 170
220 CSR S(1,J),S(2,J): PRINT CHR$(129)
230 NEXT
240 SOUND 2,1,0,10,0,750,1: SOUND 3,7,15
250 CSR 1,0: COLOUR 1,8: PRINT CHR$(130)
260 LET X=1: LET Y=0: LET NX=1: LET NY=0
: LET FS=1: LET FG=0
```

```
270 CLOCK "000000"
280 FOR J=1 TO 5
290 LET C(1,J)=6*J: LET C(2,J)=0: LET CN
(1,J)=C(1,J): LET CN(2,J)=CN(2,J): LET C
D(J)=3: LET FG=0
300 COLOUR 1,1: CSR C(1,J),C(2,J): PRINT
 CHR$(130+CD(J))
310 NEXT
320 LET A=ASC(INKEY$)
330 IF A=11 THEN  LET NX=X: LET NY=Y-1
340 IF A=25 THEN  LET NX=X+1: LET NY=Y
350 IF A=10 THEN  LET NX=X: LET NY=NY+1
360 IF A=8 THEN  LET NX=X-1: LET NY=Y
370 IF NX<1 THEN  LET NX=1
380 IF NX>31 THEN  LET NX=31
390 IF NY<0 THEN  LET NY=0
400 IF NY>22 THEN  LET NY=22
410 IF NX=31 AND NY=22 THEN  GOTO 800
420 CSR NX,NY: LET PG=ASC(SPK$)
430 IF PG=42 THEN  GOTO 890
440 IF PG>130 AND PG<135 THEN  LET X=NX:
 LET Y=NY
450 IF X=NX AND Y=NY THEN  GOTO 510
460 CSR X,Y: PRINT " "
470 IF FS=0 THEN  CSR X,Y: COLOUR 1,12:
PRINT CHR$(129): CSR NX,NY: COLOUR 1,8:
PRINT CHR$(130): LET FS=1
480 IF FS=1 AND PG=129 THEN  CSR NX,NY:
COLOUR 1,12: PRINT CHR$(130): LET FS=0
490 IF FS=1 AND PG=32 THEN  CSR NX,NY: C
OLOUR 1,8: PRINT CHR$(130)
500 LET X=NX: LET Y=NY
510 LET J=1
520 IF CD(J)=1 THEN  LET CN(1,J)=C(1,J):
 LET CN(2,J)=C(2,J)-1
530 IF CD(J)=2 THEN  LET CN(1,J)=C(1,J)+
1: LET CN(2,J)=C(2,J)
540 IF CD(J)=3 THEN  LET CN(1,J)=C(1,J):
 LET CN(2,J)=C(2,J)+1
550 IF CD(J)=4 THEN  LET CN(1,J)=C(1,J)-
1: LET CN(2,J)=C(2,J)
560 IF CN(1,J)<1 THEN  LET CN(1,J)=31
570 IF CN(1,J)>31 THEN  LET CN(1,J)=1
580 IF CN(2,J)<0 THEN  LET CN(2,J)=21
590 IF CN(2,J)>21 THEN  LET CN(2,J)=0
```

```
600 CSR CN(1,J),CN(2,J): LET PG=ASC(SPK$
)
610 IF PG=42 OR PG>129 AND PG<135 THEN
LET CD(J)=CD(J)+1: LET CN(1,J)=C(1,J): L
ET CN(2,J)=C(2,J)
620 IF CD(J)=5 THEN  LET CD(J)=1
630 IF FS=0 THEN  GOTO 680
640 LET DX=ABS(X-CN(1,J)): LET DY=ABS(Y-
CN(2,J))
650 IF DX<2 AND DY<2 THEN  LET FG=1
660 IF L>5 THEN  IF DX<3 AND DY<3 THEN
LET FG=1
670 IF FG=1 THEN  GOTO 830
680 IF PG=130 THEN  GOTO 860
690 IF CN(1,J)=C(1,J) AND CN(2,J)=C(2,J)
 THEN  GOTO 750
700 CSR C(1,J),C(2,J): PRINT " "
710 IF CC(J)=1 THEN  LET CC(J)=0: IF L<8
 THEN  CSR C(1,J),C(2,J): COLOUR 1,12: P
RINT CHR$(129)
720 IF PG=129 THEN  LET CC(J)=1
730 LET C(1,J)=CN(1,J): LET C(2,J)=CN(2,
J)
740 CSR C(1,J),C(2,J): COLOUR 1,1: PRINT
 CHR$(130+CD(J))
750 LET J=J+1: IF J<6 THEN  GOTO 520
760 SBUF 1: SBUF 10
770 IF RND<.3 THEN  GOTO 320
780 SOUND 3,4,15: PAUSE 100: SOUND 3,4,1
0: PAUSE 100: SOUND 3,7,10
790 SOUND 0,150+RND*100,240,RND*5,0,500,
1: GOTO 320
800 CLS : VS 5: INK 15: PAPER 3: CSR 3,3
810 PRINT "You have escaped!"
820 GOTO 910
830 CLS : VS 5: INK 1: PAPER 8: CSR 3,3
840 PRINT "You were shot by a tank"
850 GOTO 910
860 CLS : VS 5: INK 1: PAPER 11: CSR 3,3
870 PRINT "You were run over by a tank"
880 GOTO 910
890 CLS : VS 5: INK 15: PAPER 1: CSR 3,3
900 PRINT "You were blown up by a mine"
910 CSR 3,7: PRINT "Your time was:"
```

```
920 CSR 15,9: PRINT MID$(TIME$,3,2);" mi
n"
930 CSR 15,11: PRINT RIGHT$(TIME$,2);" s
ec"
940 SBUF 1: SOUND 0,0,0: SOUND 1,0,0: SO
UND 2,0,0: SOUND 3,0,0: SBUF 2
950 IF INKEY$<>" " THEN  GOTO 940
960 GOTO 100
```

Variations

The number of mines planted is 25 plus 3 times the level of play (line 130). To make the game even more difficult, increase the number of mines. In this case line 180 also needs amending to the new value. You can add more vehicles by altering the upper value of J in line 280. Increase the second dimension of all arrays in line 20 to the new upper value of J. Reduce the initial spacing between them by changing the '6' in line 290 to '5' or '4'. The value in line 750 also needs changing to allow for a greater value of J.

Having more vehicles makes the game slower, but more difficult to play. The extra time helps to give the player time to assess the movements of all the vehicles.

19
Sprightly Spooks

Just for once, the Memotech has a chance to be itself. Instead of having to pretend to be a Pin table, an acrobat on a trapeze, or the pieces of a jigsaw puzzle, it can simply be a computer, doing computerish things. SPRIGHTLY SPOOKS is a true computer game, and quite unlike any other game or anything in real life. This is what gives it its fascination. You are pitting your skill and wit against the speed and logic of the computer.

How to play

When the program is run, the screen clears to plain blue and nothing else seems to be happening for almost a minute. This seems a very long time if you spend it staring at a blank TV screen! During this time the Memotech is busy working out all it needs for controlling the complicated movements of the spooks.

Then the screen becomes black and on it you will see lots of spooks. They all have the same shape, but they have many different colours. They start to dance.

The spook at the top left corner of the screen is coloured greenish-blue. This is the colour often referred to as cyan. None of the other spooks are cyan, and this cyan spook is the one on which you, the player, are 'riding' to begin with.

Now for the problem. You are to ride the spooks, leaping from one to another until you reach the dark red spook which is moving up and down at the bottom right corner of the screen. You can only jump from one spook to another when they are touching or when one spook is exactly behind another. If you examine their antics carefully, you will notice that some spooks merely come close to another spook and then they part. Other spooks move in front of or behind other spooks. Two spooks which only come together, but

never come one behind the other must be *exactly* side-by-side or *exactly* one above the other on the screen when you jump. Two spooks which sometimes come one behind the other must be exactly one behind the other when you jump. Sideways jumping or jumping up or jumping down is not allowed with this second kind of spook. So, in effect, this is a 3-D display, and you can jump up, down, left, right and into (or out of) the picture. If you press the correct key at just the right moment, you jump from one spook to the other. When you make a successful jump, the spook on which you *were* riding returns to its original colour. The spook to which you jump becomes cyan.

To jump, press one of the direction keys or the HOME key on the keypad to the right of the main keyboard. To jump up, down or sideways press the direction key which corresponds with the direction is which you wish to jump. If you want to jump between two spooks that are one behind the other, press the HOME key.

It is no good pressing a key and holding it until two spooks are close together. The computer ignores any attempts to 'jump the gun'! Just watch the rhythmic motions of the spooks and press the key once at the instant they come together.

Jumping from one spook to another is only part of the problem. The game can be thought of as a 3-dimensional dynamic maze. Although some pairs of spooks seem at first glance to be coming together regularly, closer observation may show that they rarely or perhaps never line up exactly. A jump from one to the other is never possible. For success, you have to decide your route in advance. Jumping on to certain of the spooks only leads you to a dead end. There is at least one possible passage across the screen, but it makes a bewildering puzzle to find it.

When you reach the spook in the lower right corner, the game ends. If you have become addicted to spook-riding, try to find your way back to the spook you started on.

Keying in

The usual warning about keying in the DATA with the greatest care applies particularly to this program.

Program design

20–30	initialising arrays and variables
40–70	the patterns for the spooks (sprites)
80–100	reading DATA into the array which holds the details of each spook
110–130	initialising the spooks
140–260	filling arrays with the values required to move the spooks
270	setting graphics screen and making the first spook cyan
280	moving those spooks which travel horizontally
290	checking that the keys are not pressed to soon and, if so setting a flag to ignore the key-presses later
300	moving those spooks which travel vertically
310–370	analysing key-press and sending computer to subroutines to determine if a jump is possible
380	repeat the loop indefinitely
500–520	subroutine to check an upward jump and, if allowable, to transfer 'you' from one spook to the next
600–620	subroutine to do the same for rightward jumps
700–720	subroutine to do the same for downward jumps
800–820	subroutine to do the same for leftward jumps
900–930	subroutine to do the same for spooks that are one behind the other
1000–1030	DATA statements for spook details

The program

```
10 REM ** SPRIGHTLY SPOOKS **
20 DIM S(26,4),N(7),H(120,7),V(120,7): C
TLSPR 2,26: CTLSPR 6,2
30 LET Y=1: LET NY=1: LET FC=0
40 GENPAT 4,1,7,15,25,57,63,30,14,7
50 GENPAT 5,1,127,255,199,15,31,63,63,42
60 GENPAT 6,1,224,240,152,156,252,120,11
2,224
70 GENPAT 7,1,254,255,227,240,248,252,25
2,84
80 FOR J=1 TO 26: FOR K=1 TO 4
```

```
90 READ S(J,K)
100 NEXT : NEXT
110 FOR J=1 TO 26
120 SPRITE J,1,S(J,1),S(J,2),0,0,S(J,3)
130 NEXT
140 FOR J=1 TO 7: LET N(J)=0: NEXT
150 FOR J=1 TO 120
160 FOR K=2 TO 7
170 LET N(K)=N(K)+1: IF N(K)=2*K-1 THEN
 LET N(K)=1
180 IF N(K)<K THEN  LET H(J,K)=16 ELSE
LET H(J,K)=-16
190 NEXT : NEXT
200 FOR J=1 TO 7: LET N(J)=0: NEXT
210 FOR J=1 TO 120
220 FOR K=2 TO 7
240 LET N(K)=N(K)+1: IF N(K)=2*K-1 THEN
 LET N(K)=1
250 IF N(K)<K THEN  LET V(J,K)=-16 ELSE
 LET V(J,K)=16
260 NEXT : NEXT
270 VS 4: COLOUR 2,1: COLOUR 4,1: CLS :
ADJSPR 1,1,7
280 FOR J=1 TO 120: FOR H=1 TO 16: LET S
(H,1)=S(H,1)+H(J,S(H,4)): ADJSPR 2,H,S(H
,1): NEXT
290 IF INKEY$="" THEN  LET FC=0 ELSE  LE
T FC=1
300 FOR V=17 TO 26: LET S(V,2)=S(V,2)+V(
J,S(V,4)): ADJSPR 3,V,S(V,2): NEXT
310 IF FC=0 THEN  LET A=ASC(INKEY$)
320 IF A=-1 THEN  GOTO 380
330 IF A=11 THEN  GOSUB 500
340 IF A=25 THEN  GOSUB 600
350 IF A=10 THEN  GOSUB 700
360 IF A=8 THEN  GOSUB 800
370 IF A=26 THEN  GOSUB 900
380 NEXT : GOTO 280
500 FOR L=1 TO 26
510 IF S(Y,1)=S(L,1) AND S(Y,2)=S(L,2)-1
6 THEN  LET NY=L: ADJSPR 1,Y,S(Y,3): ADJ
SPR 1,NY,7: LET Y=NY
520 NEXT : RETURN
600 FOR L=1 TO 26
610 IF S(Y,1)=S(L,1)-16 AND S(Y,2)=S(L,2
```

```
) THEN  LET NY=L: ADJSPR 1,Y,S(Y,3): ADJ
SPR 1,NY,7: LET Y=NY
620 NEXT : RETURN
700 FOR L=1 TO 26
710 IF S(Y,1)=S(L,1) AND S(Y,2)=S(L,2)+1
6 THEN  LET NY=L: ADJSPR 1,Y,S(Y,3): ADJ
SPR 1,NY,7: LET Y=NY
720 NEXT : RETURN
800 FOR L=1 TO 26
810 IF S(Y,1)=S(L,1)+16 AND S(Y,2)=S(L,2
) THEN  LET NY=L: ADJSPR 1,Y,S(Y,3): ADJ
SPR 1,NY,7: LET Y=NY
820 NEXT : RETURN
900 LET L=1
910 IF L<>Y AND S(Y,1)=S(L,1) AND S(Y,2)
=S(L,2) THEN  LET NY=L: ADJSPR 1,Y,S(Y,3
): ADJSPR 1,NY,7: LET Y=NY: RETURN
920 IF L<26 THEN  LET L=L+1: GOTO 910
930 RETURN
1000 DATA  16,183,3,7,176,183,4,5,112,16
7,8,7,32,151,13,6,208,135,6,2,96,119,2,6
1010 DATA 16,103,10,4,208,103,5,3,96,87,
15,4,160,71,9,4,16,55,15,5,144,55,5,3
1020 DATA 32,39,6,6,80,23,11,5,16,7,3,7,
128,7,10,6,128,183,5,4,240,183,12,4,16,1
67,4,6
1030 DATA 80,167,11,5,192,151,15,6,96,11
9,2,5,48,87,8,5,208,87,13,6,240,87,6,6,1
28,55,8,2
```

Variations

It is possible to add sound effects. The combination of sliding tones and sprightly spooks could prove overpowering!

20
Poker-Face

A melancholy kind of a fellow is Poker-Face, who keeps his thoughts to himself for most of the time. He may occasionally smile in a pensive way. Possibly he is holding a sure-fire Poker hand, but probably he is only bluffing. If you think that you can gauge his mood, it could help you to win. If not, you will have to rely on your assessment of the chances of your holding the winning hand. This is a game for one player, based on the rules of five-card Poker, with the computer as your poker-faced opponent.

How to play

As soon as you RUN the program, your hand is displayed on a green screen, reminiscent of a green baize card table. You are given 5 cards, and so is Poker-Face, though of course he will not show you his cards until you say 'See you' later in the game.

The cards have been dealt at random from a normal pack of 52 cards with no jokers, and the pack is shuffled between deals. Aces are high. While Poker-Face is analysing his hand, you have time to examine yours. In Poker, there are 10 kinds of scoring hand, which rank in the order listed below:

Royal straight flush: five consecutive cards of the same suit, from Ace down to 10.

Straight flush: five consecutive cards of the same suit, starting with a card other than the Ace (e.g. 7,6,5,4,3 of hearts).

Fours: four cards of the same value (e.g. 10 of spades, 10 of hearts, 10 of diamonds, 10 of clubs, plus one other card).

Full House: three cards of the same value, plus a pair of cards of another value (e.g. 4 of spades, 4 of hearts, 4 of clubs, plus a pair of Queens).

Flush: all five cards of the same suit but not of consecutive values (e.g. 3,5,6,J,A of clubs).

Straight: five cards of consecutive value, but not of same suit (e.g. 3 of clubs, 4 of spades, 5 of diamonds, 6 of clubs, 7 of clubs).

Threes: three cards of the same value plus two odd cards.

Two pairs: e.g. a pair of threes and a pair of kings with one odd card.

Pair: e.g. a pair of tens with three odd cards.

Top Card: if a player has none of the hands above, the hand is ranked by the value of its highest card.

By the time you have decided how good your hand is, the computer will have analysed both your hand and that held by Poker-Face. Poker-Face's reactions are then described on the screen. How much you can tell from this is a matter for you to decide. Incidentally, although the computer has analysed your hand too, the part of the program dealing with Poker-Face's reactions does not make use of this information.

You are now asked what you want to do. There are two choices:

'Turn in': press key 'T'. You think that your hand is so worthless by comparison with that held by Poker-Face, that you wish to cut your losses and end the round. If you do this, you do not see what hand Poker-Face actually held. It might have been even worse than yours! Whatever the relative rank of the hands, you lose one chip to Poker-Face. The scores are then displayed at the bottom of the screen. By the way, you each begin the game with 100 chips.

'See You': press key 'S'. The screen then displays Poker-Face's hand, and tells you its ranking. The winner is the player with the higher ranking hand. The score is displayed. If you win, Poker-Face pays you 3 chips. If you lose, you pay him 2 chips.

Press the space bar when you are ready to play the next hand.

Poker players will realize that, except when both players hold only a Top Card, this program does not attempt to compare hands that are of the same kind. In this respect a pair of Queens, for example, ranks equally with a pair of twos. In the event of both players holding the same kind of hand, the game is declared a 'null game' and the scores of both players are left unaltered.

Winning tactics

The program is designed so that you cannot reliably guess what hand Poker-Face holds from the reactions he shows. There is a strong random element, which might correspond to a 'change of mood'. There is also an occasional wildly unpredictable response which gives away nothing at all. Poker-Face can truly bluff!

Playing this game gives you insight into the frequency with which the various hands are obtained. Since the 'deal' is a true simulation of the dealing of a pack of cards, the hands turn up with the same frequencies as would be experienced with a real pack. Apart from setting out the various combinations and working out the probabilities mathematically, the best way to success in Poker is to actually play it and get the feel of the game. Gaining experience from Poker-Face could be less expensive than gaining it at the Poker table.

Keying in

Count the spaces carefully in lines 4000 to 5010, otherwise the display will be spoiled. Note that there are no spaces immediately after the commas.

Program design

20–30	initialising arrays and variables
40–70	defining patterns for suit symbols
80–140	'dealing' cards
150–200	displaying your hand
210–230	totalling numbers of cards of same denomination in each hand
240–260	totalling numbers of cards of same suit in each hand
270–400	analysing hands to pick out various kinds of flush
410–540	looking for fours, threes, full house, and pairs
550–610	looking for a straight (not flush)
620–670	finding top card
680–750	displaying Poker-Face's reactions
760–790	asking for your response
800–950	displaying Poker-Face's hand, and scores
960–990	preparing for next deal

1000–1040	subroutine converting values of cards into 'A', 'K', 'Q' etc and printing them
3000	DATA statements holding card denominations
4000–4020	DATA statements holding Poker-Face's reactions
5000–5010	DATA statements holding names of hands

The program

```
10 REM ** POKER-FACE **
20 DIM P(13,4),H(13,8),D(13,2),S(8),C(5,
2),T(2)
30 LET PS=100: LET YS=100: LET M$=""
40 GENPAT 1,129,16,56,124,254,254,146,56
,0
50 GENPAT 1,130,108,238,254,124,124,56,1
6,0
60 GENPAT 1,131,16,56,124,254,124,56,16,
0
70 GENPAT 1,132,16,56,84,238,84,16,56,0
80 FOR J=1 TO 10
90 LET D=INT(RND*13)+1: LET S=INT(RND*4)
+1
100 IF P(D,S)=1 THEN  GOTO 90
110 LET P(D,S)=1
120 IF J>5 THEN  LET S=S+4
130 LET H(D,S)=1
140 NEXT
150 VS 4: COLOUR 0,2: COLOUR 1,15: COLOU
R 2,2: COLOUR 4,2: CLS
160 CSR 3,0: PRINT "Your hand:"
170 COLOUR 1,1: CSR 3,2: PRINT CHR$(129)
;: LET S=5: GOSUB 1000
180 COLOUR 1,6: CSR 3,4: PRINT CHR$(130)
;: LET S=6: GOSUB 1000
190 COLOUR 1,6: CSR 3,6: PRINT CHR$(131)
;: LET S=7: GOSUB 1000
200 COLOUR 1,1: CSR 3,8: PRINT CHR$(132)
;: LET S=8: GOSUB 1000
210 FOR J=1 TO 13: FOR H=0 TO 1: FOR K=1
+H*4 TO 4+H*4
220 IF H(J,K)=1 THEN  LET D(J,H+1)=D(J,H
+1)+1
230 NEXT : NEXT : NEXT
```

```
240 FOR H=0 TO 1: FOR K=1+H*4 TO 4+H*4:
FOR J=1 TO 13
250 IF H(J,K)=1 THEN  LET S(K)=S(K)+1
260 NEXT : NEXT : NEXT
270 LET S=0: FOR H=0 TO 1
280 FOR K=1+4*H TO 4+4*H
290 IF S(K)=5 THEN  LET S=K
300 NEXT : IF S=0 THEN  GOTO 400
310 LET R=0: FOR J=13 TO 5 STEP -1
320 LET F=1: FOR L=J TO J-4 STEP -1
330 IF H(L,S)<>1 THEN  LET F=0
340 NEXT
350 IF F=1 THEN  LET R=J
360 NEXT
370 IF R=13 THEN  LET T(H+1)=1: GOTO 400
380 IF R>0 AND R<13 THEN  LET T(H+1)=2:
GOTO 400
390 LET T(H+1)=5
400 NEXT
410 FOR H=0 TO 1
420 IF T(H+1)>0 THEN  GOTO 540
430 FOR J=1 TO 13
440 LET C(D(J,H+1)+1,H+1)=1
450 NEXT
460 LET F=0: FOR J=1 TO 13
470 IF D(J,H+1)=2 THEN  LET F=F+1
480 NEXT
490 IF F=2 THEN  LET C(3,H+1)=2
500 IF C(5,H+1)=1 THEN  LET T(H+1)=3: GO
TO 540
510 IF C(4,H+1)=1 AND C(3,H+1)=1 THEN  L
ET T(H+1)=4: GOTO 540
520 IF C(4,H+1)=1 THEN  LET T(H+1)=7: GO
TO 540
530 IF C(3,H+1)>0 THEN  LET T(H+1)=10-C(
3,H+1)
540 NEXT
550 FOR H=0 TO 1
560 IF T(H+1)>0 THEN  GOTO 610
570 FOR J=1 TO 9
580 LET F=1: FOR L=J TO J+4: IF D(L,H+1)
=0 THEN  LET F=0
590 NEXT : IF F=1 THEN  LET T(H+1)=6
600 NEXT
610 NEXT
```

```
620 FOR H=0 TO 1
630 IF T(H+1)>0 THEN  GOTO 670
640 LET F=0: FOR J=13 TO 1 STEP -1
650 IF D(J,H+1)=1 AND F=0 THEN  LET T(H+
1)=10+J: LET F=1
660 NEXT
670 NEXT
680 CSR 2,10: PRINT "Poker-Face is ";
690 IF RND>.9 THEN  LET M=INT(RND*11): G
OTO 720
700 LET M=T(1)+INT(RND*3)-2
710 IF M<1 THEN  LET M=1
720 IF M>10 THEN  LET M=10
730 RESTORE 4000
740 FOR J=1 TO M: READ M$: NEXT
750 PRINT M$
760 CSR 2,14: PRINT "What will you do?"
770 LET A$=INKEY$: IF A$="" THEN  GOTO 7
70
780 IF A$="T" THEN  LET PS=PS+1: LET YS=
YS-1: GOTO 940
790 IF A$="S" THEN  GOTO 810
800 GOTO 770
810 RESTORE 5000
820 LET M=T(1): IF M>10 THEN  LET M=10
830 FOR J=1 TO M: READ M$: NEXT
840 CSR 16,0: PRINT "Poker-Face has:"
850 CSR 16,2: COLOUR 1,1: PRINT CHR$(129
);: LET S=1: GOSUB 1000
860 CSR 16,4: COLOUR 1,9: PRINT CHR$(130
);: LET S=2: GOSUB 1000
870 CSR 16,6: COLOUR 1,9: PRINT CHR$(131
);: LET S=3: GOSUB 1000
880 CSR 16,8: COLOUR 1,1: PRINT CHR$(132
);: LET S=4: GOSUB 1000
890 CSR 2,14: PRINT "Poker-face has ";M$
900 IF T(1)>9 AND T(2)>9 THEN  LET T(1)=
-T(1): LET T(2)=-T(2)
910 IF T(1)<T(2) THEN  CSR 2,18: PRINT "
... and WINS.": LET PS=PS+2: LET YS=YS-2
: GOTO 940
920 IF T(1)>T(2) THEN  CSR 2,18: PRINT "
... and LOSES.": LET PS=PS-3: LET YS=YS+
3: GOTO 940
```

```
930 CSR 2,18: PRINT "... so it's a NULL
HAND"
940 CSR 2,22: PRINT "Your score";YS
950 CSR 17,22: PRINT "Poker-Face";PS
960 IF INKEY$<>" " THEN  GOTO 960
970 CLS : FOR J=1 TO 13: FOR K=1 TO 4: L
ET P(J,K)=0: NEXT
980 FOR K=1 TO 8: LET H(J,K)=0: LET S(K)
=0: NEXT : FOR K=1 TO 2: LET D(J,K)=0: N
EXT : NEXT
990 FOR J=1 TO 5: FOR K=1 TO 2: LET C(J,
K)=0: LET T(K)=0: NEXT : NEXT : GOTO 80
1000 RESTORE 3000: COLOUR 1,15: PRINT "
";
1010 FOR J=13 TO 1 STEP -1
1020 READ D$
1030 IF H(J,S)<>0 THEN  PRINT D$;" ";
1040 NEXT : RETURN
3000 DATA A,K,Q,J,10,9,8,7,6,5,4,3,2
4000 DATA laughing his      head off,gas
ping for       breath,smiling serenely,i
nscrutable
4010 DATA obviously         perplexed,wa
tching you,gripping the      table
4020 DATA looking grim,turning pale,gasp
ing for        breath
5000 DATA a Royal           Straight Flus
h,a Straight       flush,Fours,a Full Ho
use
5010 DATA a Flush hand,a Straight hand,T
hrees,Two pairs,a Pair,his Top Card
```

21
Happy Holiday

This is a game for any number of players from 1 to 4. Each has £800 to spend on the holiday of a lifetime. There are 14 days in which to spend it, in any one or more of five different holiday resorts. The aim of the game is to spend as much as possible of the money in the fortnight, while scoring the greatest number of happiness points.

The game is ready for playing as presented here, but there is ample scope for you to adapt it to your favourite holiday places and include the things that you and your family like doing when on holiday. The way to do this is explained in the 'Variations' section.

How to play

The screen remains clear for a few seconds after the game is run, while the Memotech transfers data to other parts of its memory. Then you are asked how many players there are. Key any number from 1 to 4. Next you are asked to type the names of the players. Press RET after typing each name. Incidentally, the names must not be more than 10 letters long.

The screen now displays a list of the features of a happy holiday. You are asked to select those which you believe are important for maximum enjoyment of your holiday. Select the feature that is most important to you and the one which is next most important. You are also asked to select the one which is least important. To make these selections, wait until your name appears at the bottom of the screen, then simply press the correspondingly numbered key. Use the keys along the top edge of the keyboard (the key-pad is not used in this game). Each player in turn is asked to key in their preferences. It is as well to remember which features you have selected, for you will gain more happiness later if you go to those resorts which provide the

features you like best. In short, decide now which kind of holiday you prefer; later, go and have just that kind of holiday.

Each player now has a chance to view the Memotech Travel Brochure, which describes the holidays available. The second page of the brochure lists the five resorts. If you want to find out more about a resort, key its number and press RET. You will then be told what there is to do at the resort and how much it costs. To get back to the main list of resorts, just press RET.

There are three different things to do at each resort. One is cheap, perhaps costing nothing at all. The next is of moderate cost. The third item is a relatively expensive activity. You will note that some activities require half a day, while others require longer. In the game, time is reckoned in half-days and the game ends at half-day number 28. The top right-hand corner of the screen displays the cost of staying at the resort for half a day. This is expressed in local currency. This charge is in addition to the costs (if any) of the various activities.

The charges are all in local currency, which means that you have to get used to using local money, just as you do on a real holiday abroad. Here is a reference table of the exchange rates used in the game:

Resort	Currency	How many to the pound
Norfolk Broads	Pound	1
Loire Valley	French franc	10
Austrian Tyrol	Austrian schilling	25
Dalsland	Swedish kroner	10
Costa Brava	Peseta	200

You begin the game with £800. As you travel from country to country, all of your cash is automatically converted into the currency of that country, using the rates above. You are charged a 2% conversion fee each time you travel to a new country.

At this stage of the game you are looking at the descriptions of activities just to help you plan your holiday. You will be able to see

these descriptions again later, when you are deciding what to do each day.

After you have looked at as many of the resort descriptions as you want, press RET. You will then be asked 'Which resort?'. Press the appropriate number key. This is repeated for each player.

Now the question to be considered is how you intend to travel. You have the option of driving yourself in your own car, going by rail and ferry, or flying. Car travel is the cheapest, but takes longest. Air is the quickest, but is the most expensive. The screen displays the costs for a one-way journey between all the five resorts. Your travel always starts from the Broads. You are asked to key the number 1 to 3 to indicate your choice. This choice operates for the whole game.

The screen displays the costs, but you should also consider the time taken. This is related to the cost, as shown in this table:

Cost of journey (£)	Time taken (half-days)		
	Car	Rail/ferry	Air
10	2	1	1
15	2	1	1
20	3	2	1
25	3	2	1
40	5	3	2

Choosing your means of transport sets you off on your journey to your chosen resort. It may happen that some lucky or unlucky event occurs on the journey. If so, a message describing this is displayed as soon as you have chosen your means of transport. This event may have a cost, and it may cause you to gain or lose happiness. It may also delay your journey or, if you are lucky, make it take less time than usual.

Phrase book

While you are travelling to a new country it is an ideal time to brush up your knowledge of the language. Each time you travel from one country to another you are shown some useful phrases in the local language and in English. You are asked to decide which, if any, of the foreign phrases has the same meaning as each of the English

phrases. Just press the correspondingly numbered key (1 to 5) and press RET. You will then be told if your match was right or wrong. If you have previously decided that 'Trying out the language' is one of the features that you rank important for a good holiday, you will gain happiness for each correct answer. You do not lose anything for wrong answers.

It may easily happen that there is no foreign phrase on the screen to match a given English phrase. If you believe this is so, press the '0' key to score correctly.

Visitors to the Norfolk Broads also need to learn a 'foreign' language! The phrases given are the nearest we have been able to get to spelling the Norfolk dialect. As the form of the phrases may give away their meaning too readily, you are asked to match these against single words in English. Choose the English word that has the closest connection with the dialect word or phrase.

The holiday continues

After you have tried out the Phrase Book, the computer reports on the progress of your travels. It displays your name, the number of half-days that have elapsed and a message to tell you that you are travelling. There may be a random travel event, too. Then the next player is asked to try the Phrase Book until all have finished with it.

The computer then considers half-day 2 and subsequent half days, in order, until one or more of the players has arrived at the chosen resort. Those travelling a short distance or by speedy means may arrive on half-day 2, but others may not arrive until half-day 3 or even half-day 5. Your turn is passed over when you are travelling, except for a brief display to remind you of this fact, and possibly a random travel event.

When you eventually arrive at your chosen resort you will see the message 'Welcome!' on the screen. It is in the local language, so you will know which country you have arrived at. You are then asked if you wish to stay or travel. This choice is allowed each turn, so you may stay as long as you wish in any country and then proceed to another one. If you choose to travel, you are shown the Memo Travel Brochure again, and asked to select a new resort. The procedure is much the same as before, except that you cannot change your means of transport.

If you key 'S' for 'stay', you are shown the page from the brochure which describes what there is to do in the resort. Choose one of these

activities by pressing a number key. A random event may occur while you are engaged in this activity. If so, a message appears and you are told how much cash and how much happiness you have gained or lost. Before your turn ends you are also told how much cash you have left and how much happiness you have gained since the game began. The amount of cash left takes into account the half-daily cost of each resort.

The turns pass from one player to another, half a day at a time. If you select a half-day activity, your turn will come round again on the following half-day. If you select an activity that occupies a whole day, you will miss the next half-day and be given the opportunity of choosing to stay or to travel on the following half-day. If your activity takes longer than a day, several half-days will elapse before you are once again asked to choose.

The game continues in this way until the 28th half-day. On, or before, that day you must return to England (The Broads). You lose a lot of happiness if you are still in Europe or Scandinavia on the last day of the holiday. You also lose happiness if you spend more money than you have. If you spend out, you are left out of the game and take no further turns. On the other hand, do not be too penny-pinching while on holiday! The money has been given to you to enjoy and you must spend as much of it as you possibly can. When the game ends, your happiness score is reduced by an amount in proportion to any money left unspent.

After the 28th half-day, or if all players become spent out, the game ends and the final scores are displayed. The winner is the player with the greatest happiness score.

To play again, press BRK and re-run the program.

Winning tactics

You will gain most happiness if, having picked out those features which you consider most important, you spend most of your holiday in resorts which offer these features. If you like plenty of sunshine, for example, go to Spain. But the Costa Brava is not the only sunny place. Sitting fishing beside the River Bure can also appeal to sun-lovers, as can skiing in Sweden or Austria. Lovers of food and drink should obviously visit France, but there are good meals to be had in the Tyrol, and in Norwich too. If you are fond of travel itself, spend as much as you can afford on travelling from one resort to another.

While you are at a resort, take those excursions, if any, which involve travel.

Towards the end of the game, keep a sharp eye on your cash balance. It is very important to get back to England. Make sure you have enough cash for the fares, and have allowed enough time for the journey.

Most happiness gained by spending freely on activities you enjoy and avoiding those you do not. The ideal to aim at is to arrive back in England on the last half-day, almost broke! Perhaps you might leave time to finish with a quiet half-day fishing on the Broads.

Keying in

The game consists of a BASIC program, which makes use of a number of NODDY programs. The NODDY program pages are called BROCHURE, WHERE, WHICH, HOW, GONORFOLK, GOLOIRE, GOTYROL, GODALSLAND and GOCOSTA. These should be keyed in exactly as listed (though you can use abbreviations, as explained in the Memotech Operator's Manual). The text pages are called MEMO, RESORTS, NORFOLK, LOIRE, TYROL, DALSLAND, COSTA, FEATURES and TRAVEL. You can alter the layout and contents of these pages as you wish. The exception is the FEATURES page, for the layout of this is closely tied to the BASIC program.

It is essential to key in the DATA statements accurately, since the operation of the game depends largely on these. The text in certain lines looks odd because it contains curly brackets and other unexpected symbols. Type these exactly as printed. When the program is running, the Memotech will be made to use its various character sets for different languages. Then these peculiar symbols will appear as ordinary letters but with accents, umlauts and the like. The symbol which is printed as a vertical broken line (see program line 5010, 'Dankeschon') is obtained on the Memotech by using the key marked with a single vertical line (next to the BS key in the top row).

Points of interest

One interesting feature of this program is that it makes use of NODDY for providing the extensive text needed for the game.

Another feature of interest is the use of the Memotech's alternative character fonts, to allow text to be displayed with the correct foreign alphabets and their accents.

Program design

NODDY program

BROCHURE, causes the MEMO page to be displayed, followed after a short pause by the RESORTS page. Pressing a key (1 to 5) calls up one of the resort pages. RET returns the player to the RESORTS page, from which RET returns the user to the BASIC program. RESORTS can be called directly by using WHERE, and the resort pages can be called by using GONORFOLK, GOLOIRE, etc. WHICH displays FEATURES. HOW displays TRAVEL.

BASIC program

20–70	initialise and load arrays
80–100	getting number and names of players
110–150	players indicate their holiday preferences
160–170	players choose first resort
180	player choose means of transport
190	first half-days cost deducted
200–250	beginning a new turn
260	Phrase Book routine on change of country
270–310	deciding if player is travelling, on an excursion, or ready to choose new activity
320–360	happiness calculations while travelling, and random travel events
370–410	displaying 'Welcome!' message and inviting 'stay or travel?'
420–510	displaying list of activities; inviting choice; calculating cost, happiness, and half-days required
520–690	random events, their cost and happiness
700–730	displaying cash left and total happiness, at end of turn
740–760	routine for travel to a new resort
770	return to start of loop
780–890	display of final scores
1000–1010	subroutine for inputting choice of features
1050–1080	subroutine for inputting resort

1100–1270	subroutine for inputting transport, charging transport and selecting random travel events
1300–1350	subroutine for displaying local currency symbol
1400–1570	Phrase Book subroutine
1600–1660	subroutine for finding phrase DATA
1700–1730	'Spent out' subroutine
2000–2070	DATA for arrays
2500	DATA for 'Welcome!' messages
3000–3200	DATA for random travel events
3300–4700	DATA for random events at resorts
4800–5260	DATA for foreign and English phrases

The program

NODDY listings

```
BROCHURE

        *DISPLAY MEMO.
        *PAUSE  *PAUSE
        *PAUSE  *PAUSE
        *GOTO WHERE.
```

Fig. 21.1. NODDY program page, BROCHURE.

```
WHERE

^r *DISPLAY RESORTS.
   *ENTER
   *IF 1,a
   *IF 2,b
   *IF 3,c
   *IF 4,d
   *IF 5,e
   *RETURN
^a *DISPLAY NORFOLK.   *ENTER *BRANCH r
^b *DISPLAY LOIRE.     *ENTER *BRANCH r
^c *DISPLAY TYROL.     *ENTER *BRANCH r
^d *DISPLAY DALSLAND.  *ENTER *BRANCH r
^e *DISPLAY COSTA.     *ENTER *BRANCH r
```

```
GONORFOLK

        *DISPLAY NORFOLK.
        *RETURN
```

Fig. 21.3. NODDY program page, GONORFOLK.

```
GOLOIRE

        *DISPLAY LOIRE.
        *RETURN
```

Fig. 21.4. NODDY program page, GOLOIRE.

```
GOTYROL

        *DISPLAY TYROL.
        *RETURN
```

Fig. 21.5. NODDY program page, GOTYROL.

```
GODALSLAND

        *DISPLAY DALSLAND.
        *RETURN
```

Fig. 21.6. NODDY program page, GODALSLAND.

```
GOCOSTA

        *DISPLAY COSTA.
        *RETURN
```

Fig. 21.7. NODDY program page, GOCUSTA.

```
WHICH

        *DISPLAY FEATURES.
        *RETURN
```

Fig. 21.8. NODDY program page, WHICH.

```
HOW

    *DISPLAY TRAVEL.
    *RETURN
```

Fig. 21.9. NODDY program page, HOW.

```
MEMO TRAVEL UNLIMITED


      5 TIP-TOP RESORTS!


      14 DAYS FOR £800, FULL BOARD,

      EXCURSIONS, AND ALL TRAVEL.


   Have a HAPPY HOLIDAY with MEMO!!
```

Fig. 21.10. NODDY text page, MEMO.

```
RESORTS OF A LIFE-TIME

    Stay at any one or more of these:

    1) The NORFOLK BROADS, ENGLAND

    2) The LOIRE VALLEY, FRANCE

    3) The AUSTRIAN TYROL

    4) DALSLAND, SWEDEN

    5) The COSTA BRAVA, SPAIN

  Key 1 to 5 for more details, key RET
              to continue
```

Fig. 21.11. NODDY text page, RESORTS.

```
NORFOLK BROADS                    (£40)

    1) Fish the waters of the Bure
       or the Ant

          Half-day for £2

    2) Visit Norwich - a fine city!

          Whole-day trip for £6


    3) Sail for half a day at Horning

       and on Wroxham Broad for £10
```

Fig. 21.12. NODDY text page, NORFOLK.

```
LOIRE VALLEY, FRANCE              (FF.300)

    1) Visit the picturesque and
       romantic chateaux.  Day trip
       for FF.100

    2) Enjoy a wine-tasting at the
       chateau, followed by a meal
       at a local cafe. A  half-day
       of happiness for FF.600

    3) A gourmet meal and overnight
       stay at a Hotel Grand Luxe.
       A day and a half for the bon
       viveur - only FF.1500
```

Fig. 21.13. NODDY text page, LOIRE.

```
TYROL                             (OS.750)

    1)  Spend a half-day in the local
        shops, and relax with a beer
        or a glass of wine in the
        cafe, for OS. 100.

    2)  A day's skiing costs only OS.400
        including coaching.

    3)  Join a 2-day climbing expedition
        for OS.1500.
```

Fig. 21.14. NODDY text page, TYROL.

```
DALSLAND                                (Skr.360)

    1)  Follow miles of clearly marked
        foot-paths in the beautiful
        forests.              (No cost)

    2)  A half-day's skiing costs only
        Skr.60, including coaching.


    3)  Fly to Lappland to see the
        Wildlife and the Midnight Sun.
        A 2-day tour by air costs
        SKr.1000
```

Fig. 21.15. NODDY text page, DALSLAND.

```
COSTA BRAVA                             (Pt.4000)

    1)  Relax and enjoy the Sun, sand
        and sea - it costs nothing

    2)  A one-day coach tour of the
        for Pt.1500

    3)  Visit the Casino and the Cafe
        Flamenco. An exotic and
        exciting half-day for Pt.5000
```

Fig. 21.16. NODDY text page, COSTA.

```
FEATURES OF A HAPPY HOLIDAY

    0  Sunshine
    1  Scenery
    2  Local shops
    3  The travelling
    4  Meeting people
    5  Active outdoor life
    6  Trying out the language
    7  Sight-seeing
    8  Food and/or drink
    9  Night-life

  KEY THE NUMBER OF THE FEATURE
      WHICH, FOR YOU IS THE
```

Fig. 21.17. NODDY text page, FEATURES.

```
TRAVEL COSTS  in pounds, one-way:

(1) DRIVE YOURSELF        FROM

          Broads Loire Tyrol Dalld Costa

TO Broads   --     20    25     25     25

   Loire    20     --    10     25     15

   Tyrol    25     10    --     25     25

   Dalld    25     25    25     --     40

   Costa    25     15    25     40     --

      (2) RAIL/FERRY  double above
      (3) AIR         treble above
```

Fig. 21.18. NODDY text page, TRAVEL.

BASIC listing

```
10 REM ** HAPPY HOLIDAY **
20 DIM N$(4,10),P(4,11),C(5,8),T(5,5),H(
10,5,3),RP(2,5): PRINT CHR$(27);"B1"
30 FOR J=1 TO 4: LET P(J,1)=800: LET P(J
,2)=100: LET P(J,4)=1: NEXT
40 FOR J=1 TO 8: FOR K=1 TO 5: READ C(K,
J): NEXT : NEXT
50 FOR J=1 TO 4: FOR K=J+1 TO 5: READ T(
K,J): NEXT : NEXT
60 RESTORE 2020: FOR J=1 TO 4: FOR K=J+1
 TO 5: READ T(J,K): NEXT : NEXT
70 FOR J=1 TO 10: FOR K=1 TO 5: FOR L=1
TO 3: READ H(J,K,L): NEXT : NEXT : NEXT
80 CSR 2,2: PRINT "How many players? (1-
4)";
90 LET PN=ASC(INKEY$)-48: IF PN<1 OR PN>
4 THEN  GOTO 80
100 CSR 2,4: PRINT "Type their names"
110 FOR J=1 TO PN: CSR 4,J*3+4: PRINT "P
```

```
layer ";J;: INPUT "   ";N$(J): NEXT
120 PLOD "WHICH": FOR J=1 TO PN: CSR 10,
21: PRINT N$(J);"'s turn           "
130 CSR 10,15: PRINT "  MOST IMPORTANT
 ": LET F=5: GOSUB 1000
140 CSR 10,15: PRINT "NEXT MOST IMPORTAN
T": LET F=6: GOSUB 1000
150 CSR 10,15: PRINT "  LEAST IMPORTANT
 ": LET F=7: GOSUB 1000: NEXT
160 FOR J=1 TO PN
170 PLOD "BROCHURE": GOSUB 1050: NEXT
180 FOR J=1 TO PN: PLOD "HOW": GOSUB 110
0
190 LET P(J,1)=P(J,1)-P(J,3)*C(P(J,4),2)
: NEXT
200 LET D=0: LET J=0
210 LET D=D+1: IF D=29 THEN  GOTO 780
220 LET FF=1: FOR K=1 TO PN: IF P(K,8)=0
 THEN  LET FF=0
230 NEXT : IF FF=1 THEN  GOTO 780
240 LET J=J+1
250 IF J=PN+1 THEN  LET J=1
260 IF P(J,9)>0 THEN  LET P(J,9)=0: PRIN
T CHR$(27);"B";CHR$(P(J,4)+48): GOSUB 14
00
270 IF P(J,3)>=D AND P(J,11)=0 THEN  GOT
O 770
280 IF P(J,3)<D THEN  LET P(J,11)=0
290 CLS : CSR 2,2: PRINT N$(J);"'s turn.
": CSR 27,2: PRINT "Half-day";D: IF P(J,
1)<0 THEN  CSR 3,10: GOSUB 1700: GOTO 77
0
300 IF P(J,3)<D THEN  GOTO 360
310 CSR 2,4: PRINT "TRAVELLING"
320 IF P(J,5)=4 THEN  LET P(J,2)=P(J,2)+
20
330 IF P(J,6)=4 THEN  LET P(J,2)=P(J,2)+
10
340 LET P(J,2)=P(J,2)+5
350 PAUSE 2000: LET R=P(J,10): LET RD=0:
 GOSUB 1160: GOTO 770
360 PRINT CHR$(27);"B";CHR$(P(J,4))
370 RESTORE 2500: FOR K=1 TO P(J,4): REA
D M$: NEXT
380 CSR 10,5: PRINT M$
```

```
390 CSR 2,8: PRINT "Stay or travel? (S/T
)"
400 LET A$=INKEY$: IF A$<>"S" AND A$<>"T
" THEN  GOTO 400
410 IF A$="T" THEN  GOTO 740
420 IF P(J,4)=1 THEN  PLOD "GONORFOLK"
430 IF P(J,4)=2 THEN  PLOD "GOLOIRE"
440 IF P(J,4)=3 THEN  PLOD "GOTYROL"
450 IF P(J,4)=4 THEN  PLOD "GODALSLAND"
460 IF P(J,4)=5 THEN  PLOD "GOCOSTA"
470 CSR 2,21: PRINT "Key 1 to 3"
480 LET NA=ASC(INKEY$)-48: IF NA<1 OR NA
>3 THEN  GOTO 480
490 LET P(J,1)=P(J,1)-C(P(J,4),1+NA*2)-C
(P(J,4),2)*C(P(J,4),2+NA*2)
500 LET P(J,2)=P(J,2)+H(P(J,5)+1,P(J,4),
NA)*10+H(P(J,6)+1,P(J,4),NA)*5-H(P(J,7)+
1,P(J,4),NA)*10
510 LET P(J,3)=P(J,3)+C(P(J,4),2+NA*2)
520 LET NC=P(J,4): IF NC=1 AND NA=1 THEN
  RESTORE 3300
530 IF NC=1 AND NA=2 THEN  RESTORE 3400
540 IF NC=1 AND NA=3 THEN  RESTORE 3500
550 IF NC=2 AND NA=1 THEN  RESTORE 3600
560 IF NC=2 AND NA=2 THEN  RESTORE 3700
570 IF NC=2 AND NA=3 THEN  RESTORE 3800
580 IF NC=3 AND NA=1 THEN  RESTORE 3900
590 IF NC=3 AND NA=2 THEN  RESTORE 4000
600 IF NC=3 AND NA=3 THEN  RESTORE 4100
610 IF NC=4 AND NA=1 THEN  RESTORE 4200
620 IF NC=4 AND NA=2 THEN  RESTORE 4300
630 IF NC=4 AND NA=3 THEN  RESTORE 4400
640 IF NC=5 AND NA=1 THEN  RESTORE 4500
650 IF NC=5 AND NA=2 THEN  RESTORE 4600
660 IF NC=5 AND NA=3 THEN  RESTORE 4700
670 CLS : READ RN: LET RR=INT(RND*10*RN)
+1: IF RR>RN THEN  GOTO 700
680 FOR RM=1 TO RR: READ RM$,RC,RH: NEXT

690 GOSUB 1210
700 CSR 5,12: PRINT "Cash left:";: GOSUB
 1300: PRINT P(J,1)
710 IF P(J,1)<0 THEN  CSR 5,14: GOSUB 17
00: GOTO 770
720 CSR 5,16: PRINT "Total Happiness:";P
(J,2)
```

```
730 PAUSE 5000: GOTO 770
740 PLOD "WHERE": GOSUB 1050
750 LET R=P(J,10): LET RD=0: GOSUB 1130
760 LET P(J,1)=P(J,1)-INT(((T(P(J,4),P(J
,9))/R)+5)/10+.5)*C(P(J,4),2)
770 IF J=PN THEN  GOTO 210 ELSE  GOTO 24
0
780 PRINT CHR$(27);"B1": LET P(J,1)=INT(
P(J,1)/C(P(J,4),1))
790 CLS : CSR 2,2: PRINT "HOLIDAYS ARE O
VER"
800 CSR 19,4: PRINT "Cash (£)   Happines
s"
810 FOR J=1 TO PN: LET L=J*3+4
820 CSR 2,L: PRINT N$(J);
830 IF P(J,1)<0 THEN  CSR 11,L: PRINT "S
PENT OUT";
840 LET P(J,2)=P(J,2)-INT(ABS(P(J,1))/5)
850 IF P(J,4)<>1 THEN  PRINT " ABROAD";:
 LET P(J,2)=P(J,2)-50: GOTO 870
860 CSR 19,L: PRINT P(J,1)
870 CSR 29,L: PRINT P(J,2)
880 NEXT
890 GOTO 890
1000 LET P(J,F)=ASC(INKEY$)-48: IF P(J,F
)<0 OR P(J,F)>9 THEN  GOTO 1000
1010 PAUSE 500: RETURN
1050 CSR 2,22: PRINT "Which resort, ";N$
(J);"?"
1060 LET CN=ASC(INKEY$)-48
1070 IF CN<1 OR CN>5 THEN  GOTO 1060
1080 LET P(J,9)=CN: RETURN
1100 CSR 2,22: PRINT "Which transport, "
;N$(J);"? (1-3)     "
1110 LET R=ASC(INKEY$)-48: LET P(J,10)=R
1120 IF R<1 OR R>3 THEN  GOTO 1110
1130 LET P(J,1)=P(J,1)-R*T(P(J,4),P(J,9)
)*C(P(J,4),1)
1140 LET P(J,1)=INT(P(J,1)*C(P(J,9),1)/C
(P(J,4),1)*.98)
1150 LET P(J,3)=P(J,3)+INT(((T(P(J,4),P(
J,9))/R)+5)/10+.5): LET P(J,11)=1: LET P
(J,4)=P(J,9)
1160 IF R=1 THEN  RESTORE 3000
1170 IF R=2 THEN  RESTORE 3100
```

```
1180 IF R=3 THEN  RESTORE 3200
1190 READ RN: LET RR=INT(RND*10*RN)+1: I
F RR>RN THEN  RETURN
1200 FOR RM=1 TO RR: READ RM$,RC,RH,RD:
NEXT
1210 CLS : CSR 2,2: PRINT RM$
1220 CSR 5,4: PRINT "Cost:";RC;" (Pounds
)"
1230 CSR 5,6: PRINT "Happiness:";RH
1240 CSR 5,8: PRINT "Delayed:";RD;" half
-days"
1250 LET P(J,1)=P(J,1)-RC*C(P(J,4),1)
1260 LET P(J,2)=P(J,2)+RH
1270 LET P(J,3)=P(J,3)+RD: PAUSE 9000: R
ETURN
1300 IF NC=1 THEN  PRINT " £";
1310 IF NC=2 THEN  PRINT " FF";
1320 IF NC=3 THEN  PRINT " OS";
1330 IF NC=4 THEN  PRINT " Skr";
1340 IF NC=5 THEN  PRINT " £";
1350 RETURN
1400 CLS : CSR 2,1: PRINT "PHRASE BOOK f
or ";N$(J): GOSUB 1600: READ N
1410 FOR K=1 TO 2
1420 FOR L=1 TO 5
1430 LET RN=INT(RND*N+1)
1440 LET FF=0: FOR M=1 TO L-1: IF RP(K,M
)=RN THEN  LET FF=1
1450 NEXT : IF FF=1 THEN  GOTO 1430
1460 LET RP(K,L)=RN: NEXT : NEXT
1470 FOR H=1 TO 2: FOR K=1 TO 5: GOSUB 1
600: READ N: IF H=2 THEN  FOR L=1 TO N:
READ PH$: NEXT
1480 FOR L=1 TO RP(H,K): READ PH$: NEXT
1490 CSR 0,2+K+(H-1)*7: PRINT K+(H-1)*5;
" ";: PRINT PH$: NEXT : NEXT
1500 FOR K=1 TO 5: LET FF=0: FOR L=1 TO
5: IF RP(1,L)=RP(2,K) THEN  LET FF=L
1510 NEXT : LET RP(2,K)=FF
1520 NEXT
1530 FOR K=1 TO 5: CSR 2,20: PRINT "Matc
h      ": CSR 7,20: PRINT K+5;: INPUT A
1540 CSR 2,21: IF A=RP(2,K) THEN  PRINT
"RIGHT" ELSE  PRINT "Wrong"
1550 IF A=RP(2,K) AND P(J,6)=6 THEN  LET
```

```
 P(J,2)=P(J,2)+5
1560 IF A=RP(2,K) AND P(J,5)=6 THEN  LET
 P(J,2)=P(J,2)+10
1570 NEXT : PAUSE 3000: RETURN
1600 LET NC=P(J,4)
1610 IF NC=1 THEN  RESTORE 4800
1620 IF NC=2 THEN  RESTORE 4900
1630 IF NC=3 THEN  RESTORE 5000
1640 IF NC=4 THEN  RESTORE 5100
1650 IF NC=5 THEN  RESTORE 5200
1660 RETURN
1700 PRINT "Spent out": PAUSE 5000
1710 LET P(J,8)=1
1720 LET P(J,2)=P(J,2)-5
1730 RETURN
2000 DATA 1,10,25,10,200,20,150,325,180,
2000,2,100,100,0,0,1,2,1,1,1
2010 DATA 6,70,400,60,1500,2,1,2,1,2,10,
600,1500,1000,5000,1,3,4,4,1
2020 DATA 20,25,25,25,10,25,15,25,25,40
2030 DATA 1,0,1,0,0,0,0,3,2,0,1,4,3,1,0,
2,0,1,2,0,0,0,3,6,3,1,6,1,2,0
2040 DATA 0,4,0,0,0,0,3,0,0,0,0,0,0,1,0,
0,0,1,3,0,0,0,0,6,0,0,8,0,2,0
2050 DATA 0,0,0,0,1,3,1,0,2,0,1,0,0,0,1,
3,0,3,0,0,0,0,5,8,3,3,0,2,0,0
2060 DATA 0,0,0,3,2,3,1,0,0,0,0,0,0,1,0,
0,4,0,4,0,4,0,0,0,0,0,6,0,3,0
2070 DATA 0,3,0,0,3,6,1,0,0,0,0,0,0,2,1,
0,1,0,0,0,2,0,1,0,0,0,0,0,0,4
2500 DATA Welcome!,Bienvenue!,Willkommen
!,V{lkommen!,[bienvenido!
3000 DATA 4,Puncture,-10,-10,0,Minor acc
ident,-5-,-100,1,Good roads,5,20,-1,Rout
e well signposted,0,20,-1
3100 DATA 4,Ferry strike,-5,-20,1,Troubl
e at customs,0,-10,0,Good meal on train,
0,10,0,Caught earlier ferry,0,10,-1
3200 DATA  4,Lost baggage,-10,-10,0,Fog
delays take-off,10,-10,1,Upgraded to fir
st class,0,10,0,Beautiful views from air
,0,10,0
3300 DATA 4,Boat fouls line,-1,-10,Raini
ng,0,-10,Catch pike,0,20,Good pub,0,10
3400 DATA 4,Early closing,1,-10,Lost way
,0,-10,Visit craft museum,0,10,Visit Cat
hedral,0,20
```

```
3500 DATA 4,No wind,0,-30,Capsized,-10,-
30,Ideal conditions,0,30,Watched regatta
,0,10
3600 DATA 4,Chateau ferm',1,-20,Ran out
of film,0,-20,Good local wine,3,20,Excel
lent guide,0,20
3700 DATA 4,Chose wrong wine,0,-5,Could
not translate Menu,0,-5,Pleasant ambienc
e,0,5,superb Menu Touristique,5,10
3800 DATA 4,Ate and drank too much,0,-20
,Central heating broke down,0,-5,Discove
red superb wine,0,50,Met interesting peo
ple,0,20
3900 DATA 4,Arrived late,0,-5,Lost purch
ases,-5,-10,Bargain souvenirs,5,20,Frien
dly service 0,5
4000 DATA 4,Bad fall,0,-10,Chair-lift ou
t of order,0,-20,Charming instructor,0,1
0,On good form,0,20
4100 DATA 4,Blizzard,0,-30,Broke leg,-20
,-50,Views from summit,0,50,Crossed glac
ier,0,20
4200 DATA 4,Lost path,0,-10,Twisted ankl
e,2,-10,Saw rare mosses,0,10,Found wild
raspberries,0,10
4300 DATA 4,Bad fall,0,-10,Chair-lift ou
t of order,0,-20,Charming instructor,0,1
0,On good form,0,20
4400 DATA 4,Cloudy no Sun,0,-50,Blizzard
 delays flight,-10,-20,Saw elk herds,0,2
0,Magnificent scenery,0,40
4500 DATA 4,Sunburnt,0,-10,Lost sunglass
es,-5,-5,Met new friends,0,20,Enjoyed no
vel,0,20
4600 DATA 4,Coach broke down,-15,-20,Dro
pped camera,-10,-10,Found secluded inn,-
5,20,Found deserted beach,0,20
4700 DATA 4,Lost at Casino,-50,-50,Casin
o closed,0,-10,Won at Casino,50,50,Carni
val time!,0,10
4800 DATA 8,Tha's a rum'un,Tha's a gret
boot,Wha'a lo' o' squit!,Come yew up the
 loke,Yew bin mardlin,hollerin',spink,ma
wther
4850 DATA odd,boat,nonsense,lane,gossip,
shout,chaffinch,girl
4900 DATA 10,Puis-je consulter la liste
```

```
des vins?,Ce vin est trop froid,Bon app{
tit!,Le menu s'il vous plait
4910 DATA Avez-vous une carte de la vill
e?,Qu'est-ce que c'est?,Sens interdit,Ph
armacien,Merci,Au revoir
4950 DATA May I see the wine list?,This
wine is too cold,Enjoy your meal!,The me
nu please
4960 DATA Do you have a street-map?,What
 is that?,No entry,Chemist,Thankyou,Good
bye
5000 DATA 10,Geben Sie mir bitte die Wei
nkarte?,Der wein ist zu kalt,Guten Appet
it!,Die Karte bitte,Haben Sie einen Stad
tplan?
5010 DATA Was ist das?,Einfahrt verboten
,Apoteke,Dankesch!n,Auf Wiedersehen
5050 DATA May I see the wine list?,This
wine is too cold,Enjoy your meal!,The me
nu please
5060 DATA Do you have a street-map?,What
 is that?,No entry,Chemist,Thankyou,Good
bye
5100 DATA 11,Hur st}r det till?,God afto
n ,God dag,Var {r det?, Parkering f!rbju
det,Jag !nskar inga m}ltider
5110 DATA Jag skulle vilja ha sm}pengar,
Har du matsedeln?,Ingen adgang,Apotek,Ta
ck
5150 DATA How do you do?,Good evening,Go
od afternoon,What is that?,No parking,I
do not want meals
5160 DATA I should like some small chang
e,Have you got the menu?,No entry,Chemis
t, Thankyou
5200 DATA 10,Por favor ]puedo ver la lis
ta de vinos?,Este vino esta demasiado fr
io,[Que aproveche!,La carta por favor
5210 DATA ]Tiene usted un plano de la ci
udad?,]Que es eso?,Hace buen tiempo,Boti
cario,Gracias,Adios
5250 DATA May I see the wine list?,This
wine is too cold,Enjoy your meal!,The me
nu please
5260 DATA Do you have a street-map?,What
 is that?,It is fine weather,Chemist,Tha
nkyou,Goodbye
```

Variations

Readers who wish to avoid a certain amount of typing may prefer to omit the Phrase Book. If so, the following lines may be omitted:

260
1400–1660
4800–5260

On the other hand, some readers may like to adapt or extend the Phrase Book. The languages are in the order English (dialect), French, German, Swedish and Spanish. Each language begins on an even hundred (line numbers 4800, 4900, 5000, etc) while the corresponding English translations begin on the 'fifties' (4850, 4950, 5050, etc). The first line for each language begins with a number indicating how many phrases there are. Note that if a line is full, the DATA continues on the next line. For example, Swedish phrases begin on line 5100 and are continued on line 5110. Their translations begin on line 5150 and continue on line 5160. Additional Swedish phrases could go on line 5120, with translations on line 5170.

There is plenty of scope for altering all the resorts and the activities, for these are described only on NODDY text pages. You can tailor the whole program to your favourite holiday resorts and experiences. The only requirement is that the languages of the 5 resorts should be English, French, German, a Scandinavian language, and Spanish or Portuguese, in that order. So you could easily provide yourself with a world-wide holiday in (for example), Surfer's Paradise (how about a Phrase Book of Strine?), Tunis, Heidelberg, Reykjavik and Rio de Janiero.

Finally, you can adapt or extend the range of random events. Make them more personal by mentioning members of your family by name. The format for travel events (line 3000 for car, 3100 for rail/ferry, 3200 for air) is to begin the line with the number of events. Then, for each event in order, give: the message, its cost (negative if you lose money), its happiness rating (positive for increased happiness) and the number of half-days delay.

Delays can also be negative, meaning a journey takes less than the normal time. It goes without saying that there should be as many good events as unlucky ones. If you have more than a line-ful of events, continue on to lines 3110, 3210, etc.

Random events at resorts (lines 3300 to 4700) can similarly be extended. Program lines 530 to 660 indicate which DATA lines apply to which resort (NC) and which activity (NA). The format is the same as for travel events, except there is no figure for delay.

APPLE II

APPLE II PROGRAMMER'S HANDBOOK
0 246 12027 4 £10.95

AQUARIUS

THE AQUARIUS AND HOW TO GET THE MOST FROM IT
0 246 12295 1 £5.95

ATARI

GET MORE FROM THE ATARI
0 246 12149 1 £5.95

THE ATARI BOOK OF GAMES
0 246 12277 3 £5.95

BBC MICRO

ADVANCED MACHINE CODE TECHNIQUES FOR THE BBC MICRO
0 246 12227 7 £6.95

ADVANCED PROGRAMMING FOR THE BBC MICRO
0 246 12158 0 £5.95

THE BBC MICRO: AN EXPERT GUIDE
0 246 12014 2 £6.95

BBC MICRO GRAPHICS AND SOUND
0 246 12156 4 £6.95

DISCOVERING BBC MICRO MACHINE CODE
0 246 12160 2 £6.95

DISK SYSTEMS FOR THE BBC MICRO
0 246 12325 7 £7.95

HANDBOOK OF PROCEDURES AND FUNCTIONS FOR THE BBC MICRO
0 246 12415 6 £6.95

INTRODUCING THE BBC MICRO
0 246 12146 7 £5.95

LEARNING IS FUN: 40 EDUCATIONAL GAMES FOR THE BBC MICRO
0 246 12317 6 £5.95

TAKE OFF WITH THE ELECTRON AND BBC MICRO
0 246 12356 7 £5.95

21 GAMES FOR THE BBC MICRO
0 246 12103 3 £5.95

PRACTICAL PROGRAMS FOR THE BBC MICRO
0 246 12405 9 £6.95

THE COLOUR GENIE

MASTERING THE COLOUR GENIE
0 246 12190 4 £5.95

COMMODORE 64

BUSINESS SYSTEMS ON THE COMMODORE 64
0 246 12422 9 £6.95

ADVENTURE GAMES FOR THE COMMODORE 64
0 246 12412 1 £6.95

COMMODORE 64 COMPUTING
0 246 12030 4 £5.95

COMMODORE 64 DISK SYSTEMS AND PRINTERS
0 246 12409 1 £6.95

THE COMMODORE 64 GAMES BOOK
0 246 12258 7 £5.95

COMMODORE 64 GRAPHICS AND SOUND
0 246 12342 7 £6.95

COMMODORE 64 WARGAMING
0 246 12410 5 £6.95

SOFTWARE 64: PRACTICAL PROGRAMS FOR THE COMMODORE 64
0 246 12266 8 £5.95

INTRODUCING COMMODORE 64 MACHINE CODE
0 246 12338 9 £7.95

40 EDUCATIONAL GAMES FOR THE COMMODORE 64
0 246 12318 4 £5.95

DRAGON

THE DRAGON 32 AND HOW TO MAKE THE MOST OF IT
0 246 12114 9 £5.95

THE DRAGON 32 BOOK OF GAMES
0 246 12102 5 £5.95

THE DRAGON PROGRAMMER
0 246 12133 5 £5.95

DRAGON GRAPHICS AND SOUND
0 246 12147 5 £6.95

INTRODUCING DRAGON MACHINE CODE
0 246 12324 9 £7.95

ELECTRON

ADVANCED ELECTRON MACHINE CODE TECHNIQUES
0 246 12403 2 £6.95

ADVANCED PROGRAMMING FOR THE ELECTRON
0 246 12402 4 £5.95

ADVENTURE GAMES FOR THE ELECTRON
0 246 12417 2 £6.95

ELECTRON GRAPHICS AND SOUND
0 246 12411 3 £6.95

ELECTRON MACHINE CODE FOR BEGINNERS
0 246 12152 1 £7.95

THE ELECTRON PROGRAMMER
0 246 12340 0 £5.95

HANDBOOK OF PROCEDURES AND FUNCTIONS FOR THE ELECTRON
0 246 12416 4 £6.95

PRACTICAL PROGRAMS FOR THE ELECTRON
0 246 12362 1 £7.95

21 GAMES FOR THE ELECTRON
0 246 12344 3 £5.95

40 EDUCATIONAL GAMES FOR THE ELECTRON
0 246 12404 0 £5.95

TAKE OFF WITH THE ELECTRON AND BBC MICRO
0 246 12356 7 £5.95

IBM

THE IBM PERSONAL COMPUTER
0 246 12151 3 £6.95

LYNX

LYNX COMPUTING
0 246 12131 9 £6.95

MEMOTECH

MEMOTECH COMPUTING
0 246 12408 3 £5.95

THE MEMOTECH GAMES BOOK
0 246 12407 5 £5.95

ORIC-1

THE ORIC-1 AND HOW TO GET THE MOST FROM IT
0 246 12130 0 £5.95

THE ORIC PROGRAMMER
0 246 12157 2 £6.95

THE ORIC BOOK OF GAMES
0 246 12155 6 £5.95

TI99/4A

GET MORE FROM THE TI99/4A
0 246 12281 1 £5.95

VIC-20

GET MORE FROM THE VIC-20
0 246 12148 3 £5.95

THE VIC-20 GAMES BOOK
0 246 12187 4 £5.95

ZX SPECTRUM

AN EXPERT GUIDE TO THE SPECTRUM
0 246 12278 1 £6.95

INTRODUCING SPECTRUM MACHINE CODE
0 246 12082 7 £7.95

LEARNING IS FUN: 40 EDUCATIONAL GAMES FOR THE SPECTRUM
0 246 12233 1 £5.95

MAKE THE MOST OF YOUR ZX MICRODRIVE
0 246 12406 7 £5.95

THE SPECTRUM BOOK OF GAMES
0 246 12047 9 £5.95

SPECTRUM GRAPHICS AND SOUND
0 246 12192 0 £6.95

THE SPECTRUM PROGRAMMER
0 246 12025 8 £5.95

THE ZX SPECTRUM AND HOW TO GET THE MOST FROM IT
0 246 12018 5 £5.95

WHICH COMPUTER?

CHOOSING A MICROCOMPUTER
0 246 12029 0 £4.95

LANGUAGES

COMPUTER LANGUAGES AND THEIR USES
0 246 12022 3 £5.95

EXPLORING FORTH
0 246 12188 2 £5.95

INTRODUCING LOGO
0 246 12323 0 £5.95

INTRODUCING PASCAL
0 246 12322 2 £5.95

MACHINE CODE

Z80 MACHINE CODE FOR HUMANS
0 246 12031 2 £7.95

6502 MACHINE CODE FOR HUMANS
0 246 12076 2 £7.95

SOFTWARE GUIDES

WORKING WITH dBASE II
0 246 12376 1 £7.95

USING YOUR MICRO

COMPUTING FOR THE HOBBYIST AND SMALL BUSINESS
0 246 12023 1 £6.95

DATABASES FOR FUN AND PROFIT
0 246 12032 0 £5.95

FIGURING OUT FACTS WITH A MICRO
0 246 12221 8 £5.95

INSIDE YOUR COMPUTER
0 246 12235 8 £4.95

SIMPLE INTERFACING PROJECTS
0 246 12026 6 £6.95

PROGRAMMING

COMPLETE GRAPHICS PROGRAMMER
0 246 12280 3 £6.95

THE COMPLETE PROGRAMMER
0 246 12015 0 £5.95

PROGRAMMING WITH GRAPHICS
0 246 12021 5 £5.95

WORD PROCESSING

CHOOSING A WORD PROCESSOR
0 246 12347 8 £7.95

WORD PROCESSING FOR BEGINNERS
0 246 12353 2 £5.95

FOR YOUNGER READERS

BEGINNERS' MICRO GUIDES: ZX SPECTRUM
0 246 12259 5 £2.95

BEGINNERS' MICRO GUIDES: BBC MICRO
0 246 12260 9 £2.95

BEGINNERS' MICRO GUIDES: ACORN ELECTRON
0 246 12381 8 £2.95

MICROMATES

SIMPLE ANIMATION
0 246 12273 0 £1.95

SIMPLE PICTURES
0 246 12269 2 £1.95

SIMPLE SHAPES
0 246 12271 4 £1.95

SIMPLE SOUNDS
0 246 12270 6 £1.95

SIMPLE SPELLING
0 246 12272 2 £1.95

SIMPLE SUMS
0 246 12268 4 £1.95

GRANADA GUIDES: COMPUTERS
0 246 11895 4 £1.95

21 FASCINATING GAMES FOR MEMOTECH MACHINES!

Memotech computers, with their 16-colour high-resolution graphics, 32 mobile sprite graphics and high-quality audio output, have all the facilities for fast and fascinating computer games. Here are 21 exciting, high quality games specifically programmed to make the most of the special features of these microcomputers. All the programs run equally well on all Memotech models.

Each program comes with instructions showing you how to play the game. The way the game works is explained in detail, with tips on winning tactics. Hints are also given on how to change the games creatively to suit your own ideas.

The Authors

Owen and Audrey Bishop are freelance technical writers and programmers. Between them they have written forty books, including a number on popular computing. Owen Bishop is a well known and regular contributor to computing journals.

Front cover illustration by Richard Hopkinson

GRANADA PUBLISHING
Printed in England 0 246 12407 5

£5.95 net