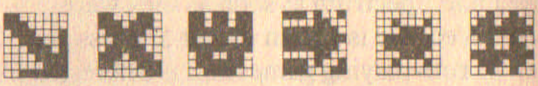
A FONTANA ORIGINAL

NEW ADVENTURE SYSTEMS FOR THE SPECTRUM

S. ROBERT SPEEL

FONTANA

Robert Speel was born in 1964. One of the first people to acquire a ZX80, he started writing software immediately, and moved on to the ZX81 and Spectrum as soon as they were introduced. He is a keen player of chess and fantasy role-playing games and is interested in natural history. He is also the author of *Better Programming for Your Spectrum and ZX81* (available in Fontana).



S. Robert Speel

New Adventure Systems for the Spectrum

Fontana Paperbacks

First published by Fontana Paperbacks 1984

Copyright © in text and programs S. Robert Speel 1984

Set in Linotron Plantin
Reproduced, printed and bound in Great Britain by
Hazell Watson and Viney Limited
Member of the BPCC Group,
Aylesbury, Bucks.

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

Conditions of Sale
This book is sold subject to the condition
that it shall not, by way of trade or otherwise,
be lent, re-sold, hired out or otherwise circulated
without the publisher's prior consent in any form of
binding or cover other than that in which it is
published and without a similar condition
including this condition being imposed
on the subsequent purchaser.

Contents

Foreword

An adventure system is much more than just a single program. The emphasis is on expandability and diversity, to give the player a wide range of possibilities:

The Places – tunnels, castles, pits, marshes, craters, wide prairies, rocky islands and forest clearings. . . .

The Characters – warriors, explorers, hunters, wizards, rulers, castaways, bounty hunters, treasure seekers. . . .

The Aims – finding treasure, escaping from seemingly endless mazes, collecting useful articles, fighting monsters, staying alive, keeping in power, or just fleeing from oversize monsters. . . .

The Challenges – man-eating plants, poisoned food, slippery cliffpaths, magic that gets used up too quickly, weapons that break, treasure that is hidden, marshes to sink in, unlabelled potions to drink, precipices to fall over, holes to slide into, cliffs, rocks, rivers, seas and thick forests to confuse you, labyrinths to get lost in, and tough monsters such as stonebeasts, fangmoles, dragons, snakes, sabretooths, griffins, and the weaker armadillos, penguins, tarantulas and bats, or human adversaries – guards, priests, guildsmen and the wizard. There are monsters that are stronger than you, faster than you, magic-resistant, poison-resistant, too big to kill or too numerous, have poisonous bites or that put out your light. . . .

The Solutions – you can find those!

all
g in co
n sce

n the
into a
e par
offer
prob
ntro
or n
chanc
fames

Introduction

This book contains six major adventure systems, all of which are designed to be expandable, either increasing in complexity and variety, or changing to completely different scenarios as the systems grow.

In an adventure game, you, the player, take on the role of a warrior, explorer or mage, and venture forth into a different world presented by the computer. Adventure games differ from most other computer games because they offer a choice of strategies. The way in which you tackle the problems and achieve your aim is up to you, and every adventure player develops his or her own style. There is scope for improving technique, and learning how to increase your chance of winning is one of the main attractions of adventure games.

The chapters for each system are divided up into a number of sections.

First, there is an overall introduction describing the type of system, its scope and the way in which it expands.

Next comes the initial program – the *Core Program* – which is at the centre of the system. Several headings are used here. '*To play*' describes your actual aim, and the commands and options you will use on running the program.

Typing in the listing gives any special points to look out for in the listing. This may include details or graphic characters to look out for, as well as which programs to MERGE onto which when additions to the core are made. This section may be short, but it is important that you read it.

Explanation of listing gives details of the program construction, and the purposes of some of the important variables. Interesting programming techniques which have been used are discussed here. It is not necessary to read this section before typing in and using the program, but when you want to alter the system, or write your own, the explanations of listings will be very useful.

The *listing* follows, and every few lines (normally at the end of subroutines) there is a gap of one line to make it easier to read. Line numbers are normally in tens, but in the early parts of a system there may be odd gaps which allow for lines that are included in later additions.

Finally, *Hints on play* gives some useful tips for tackling the game so far. Reading this is, of course, optional, but it is helpful if you find that your early attempts at the adventures are rather brief and unsuccessful!

Each addition to the core program is detailed in the same way, although the explanations of listings may be very brief – if the scenario alone is changed, providing too many details may give the game away! For the addition it is more important than ever to read *Typing in the listing*, especially when MERGEing is required.

Even though the systems are split up into short sections wherever possible, some of the listings are still very long, and typing them in will take a long time. The listings have one-line gaps between each group of lines, and after every twenty minutes or so it is a good idea to type up to the next gap and then SAVE. In this way, if the power lead is pulled out, or you accidentally wipe the memory, you cannot lose more than 20 minutes' typing. At the end of a session, save again, and this will be the version you reload to carry on next time. Finally, you will have a tape with lots of partly completed versions, and one complete version at the end. Save this complete version onto another cassette, and wipe out the development one.

As the final programs are very long, it is vital that you have a back-up tape with all of them saved on it, so you can be sure that you won't have to type them again.

If you have microdrives, I suggest that you save only the completed programs onto the microdrive cartridges. Always make sure the programs are saved on cassette first – as if a microdrive fails to save, it usually becomes stuck in an endless loop which you have to unplug the computer to stop. If you have more than one microdrive, always save the programs using drive 1, as this tends to give slightly more capacity on a cartridge.

The first system is **Preset Fantasy** and this includes three separate scenarios: *Statue* is a short scenario which is a good introduction to adventuring; *Temple* is more complicated, and may take quite a while to solve; *Wizard's Tower* has a magical bias – your rather limited magic versus the powerful wizard's spells and guardian monsters. Preset Fantasy System has its room and action descriptions in a special code so you won't spoil all the surprises by reading the listings.

Warrior Mage System is a fighting adventure. Fairly short and simple to begin with, the game expands to include high-res pictures, special rooms, lots of interesting objects (including potions), and a lower level with extra-tough monsters. The system is random – i.e., different each time you run the program – but you can make several consecutive attempts in any one maze. Both fighting skills and magical powers are utilized in this game, but you must not waste your spells indiscriminately.

Tribe puts you in charge of a small tribe, elected on the promise that you will bring prosperity to the people. Again, this system expands stage by stage from a relatively small, 16K version into a very large program, giving you the full pleasures of bribery, stealing from the treasury, controlling employment and stockpiling food. Your only problems are droughts, falling standards of living, riots, clashing with one of the guilds and, of course, elections.

Preset Future System uses a similar type of programming method to Preset Fantasy System, but develops in a different direction to give two scenarios set in the future. You are a space explorer, and 'Crater on Archelon One' finds you stranded in an inhospitable, misty hole out of which you must climb before you can call for help. In 'Mountains of Sirius

Two', your planet-hopper takes you out of communicator range before running out of fuel, and you must walk back to your spaceship. All you can remember is that the spaceship is to your west, and that you flew over some rather impressive mountains. . . .

The **Fangmole Tunnels System** is a graphic system. You control your character with the cursor keys, moving around a large maze of tunnels and rooms looking for treasure. The system expands to give various objects, such as exploders to blast your way through the walls (blasting yourself all over the walls if you are careless), ladders to bridge pits with, and torches to increase your vision. Monsters include bats which take your torches, snakes which bite if stepped on, and, in the largest version of the game, the fangmoles. These huge, indestructible, voracious monsters can tunnel after you as fast as you can run, creating new corridors in the process. Your only hope is that they will give up pursuing you before you reach a dead end. . . .

Finally, **Anarchic System** provides four separate scenarios, each of which can be slotted in to the core program. This system is designed to let the player exercise his initiative, perhaps using objects to do things they were not intended for. With multiple aims and a highly advanced fighting system, the four scenarios are very different from one another. As a special bonus, the program used to write the scenarios is also included, so you can invent your own. Work out the details of your scenario and input them, and the computer will tell you which lines to type in!

S. Robert Speel
Eastcote, Middlesex

1. The Structure of an Adventure Program

Most of the adventures in this book have the following main sections.

1. *Setting up the game.*
2. *Description of locations.*
3. *Entering commands.*
4. *Moving.*
5. *Interaction with monsters.*
6. *Interaction with objects.*
7. *Special commands.*
8. *Introduction, end, and final score.*

The length and importance of each of these sections varies between adventures and in some adventures one or more of the above may be absent altogether.

1. Setting up the game
This includes initializing the variables, DIMensioning the arrays and string arrays and often READing data into strings, arrays and string arrays. The adventure usually has a map, which consists of string arrays holding all the locations of the adventure, the characteristics of these locations and their positions in relation to one another.

In Random adventures, the whole map is usually set up at the beginning of the game, often with characteristics for each location. For this reason, there may be a wait of several minutes while running the program before play can commence.

Preset adventures, due to their locations and characteristics being within the listing, do not need to build up large arrays holding the whole map. The characteristics of rooms can be extracted from DATA statements as needed. This can save memory, as only small arrays are needed. However, due to the large number of special characteristics of various rooms, many single variables may be needed.

2. *Description of locations*
This can take the form of text or a picture of your surroundings. In Preset adventures, as each location is individual, there can be a very large description section to the program. This can often be reduced by coding the text into a shorter form, translating the code as needed.

In a Random adventure, the description section can be fairly short, as all descriptions have the same format. If graphics are used, then this section of any adventure expands rapidly.

3. *Entering commands*
This is the routine which acts on what you type into the keyboard. If the inputs are single key-presses, INKEY$ can be used. This is useful when a time factor is important, such as when fighting or travelling at speed. For general purposes, a whole string is entered – one- or two-word commands – and this is more complicated. A program will often contain several 'Enter command' routines, for different types of action.

4. *Moving*
This section deals with moving you around the adventure map. Generally, this consists of checking if you can move, then altering your X and Y coordinates within an array. If there is a third dimension, a Z coordinate may be used as well. An alternative to this can be used in the Preset adventures, where each location has a separate routine. A record of the program line number where the routine starts is kept, and this is changed when you move to a new location.

5. Interactions with monsters
This includes fighting, bargaining, bribing, questioning and other ways of dealing with living creatures you meet in the adventure. Random adventures may confine themselves to fighting or bargaining, while Preset adventures tend to feature diverse ways of overcoming life forms. Either way, this is likely to be a large section of most adventures.

6. Interactions with objects
The two most important commands for dealing with objects are 'take' and 'put'. Generally, an object cannot be used until it has been taken and, once put down somewhere, should remain there until the time when (and if) you come back for it, unless there is a special reason why it should be removed (objects may be eaten or stolen).

Objects tend to have special uses for fighting, changing your characteristics or, as wealth, helping your score – e.g., money and valuable objects. In Preset adventures, each object may have just one specific purpose, invoked by the right command at the right time. The other extreme is reached in Anarchic Adventure where any object can be used in any way: you may want to use a plank for fighting, or throw your food at a monster – it's up to you.

7. Special commands
These are very important in Preset adventures and refer to commands that are used only occasionally, in special locations or with certain objects. These tend to invoke several lines of text and a few variables for each command.

8. Introduction, end and score
At the beginning, an introduction with background details of the adventure, your commands, and some advice may be given. As it commonly takes a long time to play a single game, it is usual to have some sort of 'reward' at the end – a picture, or a tune, and maybe a score to show how well you did. These

can add greatly to the length of listing and amount of memory used, so I have reduced them as much as possible, giving explanations of how to play, etc. in the text before each listing. You can expand the introduction to the limits of your RAM, as line numbers have been left for this.

How the sections fit together

I construct adventure programs within a fairly loose framework. Setting up the game is usually done at lines 9000 onwards, to keep it out of the way of the main program, and the program starts with GOTO 9000 or GOSUB 9000. The introduction may also be here, or at the beginning of the program, before line 1000.

The main routines tend to be between lines 1000 and 4990. Manipulating objects, moving, commands and descriptions are all held in this section of the program. Fighting is usually at line 5000 or later. Extras tend to come after line 7000, and may include some special commands, subroutines, scoring and similar short sections. For some adventures, this format is not suitable, and another is chosen, but in general it is a good idea to keep to a particular system. This helps when changing a program or trying to follow through a listing.

Other programming conventions are used for variables and strings. FOR-NEXT loops start with f, and nested loops continue g, h, i, etc. If a loop is needed outside the f loop, e is used. Variables have two-letter names (except for coordinates, x, y, z), particularly important variables (e.g., cash), and variables very closely related (e.g., variables dependent on variable ab might be called abx and aby). To save on RAM, when a routine is continually GOSUBed or GOTOed, the line number is given a single letter variable, the letter being towards the end of the alphabet. When variables are used just for convenience and then discarded (dummy variables), the letters a, b, c are used.

Inputted strings are nearly always called a$, and dummy strings are b$, c$, etc. Monsters are usually held in m$, and the map is held in string arrays z$, y$, etc., and numeric arrays of similar names.

The advantage of these sort of conventions is that, when writing additions to an adventure, it is easier to know how the program works and what each subroutine does without the need to refer to written tables saying what each variable stands for.

2. Preset Fantasy System

Each time a preset adventure is played, its locations, objects, monsters and traps remain the same. The player has a small set of commands for moving through the locations and manipulating objects, while the other commands are discovered by experimentation. A particular object's use is often not immediately apparent – for instance, a metal pipe may seem useless until you find yourself in a collapsing room, where the pipe can wedge apart the approaching walls. The obvious playing technique is to take anything you find, even if you are not sure why, and for this reason the player is generally restricted to carrying just a few objects at a time.

When the player first enters a preset adventure, he is usually killed off fairly quickly. Next time he will avoid the particular trap or monster that originally killed him, and progress a little further until a subsequent trap gets him. Gradually the player will progress through the adventure, learning how to overcome each difficulty, until finally the adventure is solved.

If the player was able to read the room descriptions straight from the listing, as well as the special commands, monsters and objects, the surprise element of the adventure would be spoiled. To avoid this a special code is used. The room descriptions are held as a string of characters, each character representing one word. When a description is to be printed, the character string is decoded, READing the correct words from a vocabulary held as DATA. This has the added bonus of saving RAM. Special commands, monster names and object names are also held in DATA statements, but here the actual words are in code. They are decoded as the program runs and put into string arrays for ease of use. No attempts have been made to make this code complicated and indecipherable, as the only purpose of the code (which could easily be deciphered by using the decoder which is in each program) is to increase the surprise element for the player.

The same wordlists and basic command set have been used for several different adventures. To avoid the need to type in a monstrously long program in one go, the adventures are split into several stages, allowing progressive stages of play.

The Foundation program is essential to the whole series. Foundation 1 is typed in, and a short scenario program (mini adventure) is MERGEd with this. This allows you to play a complete game. When you are ready for more, Foundation 1 can be MERGEd with Foundation 2, and this can in turn be MERGEd with firstly Scenario 2, and then Scenario 3. Each scenario gives a totally separate game which means, in effect, that you have several games in one.

FOUNDATION 1 and SCENARIO 1

Statue (fits into 16K)

Statue is a very simple, short adventure, but it is a very good introduction to adventuring. There are no complicated traps: it is just a matter of finding the right order in which to do things.

On looking through your attic, you find that you have the deeds to a small cottage on the coast, where a golden statue of a bull used to be kept. You decide to investigate. The cottage is derelict and the front door has fallen in, but you decide to look for the statue anyway. You go in. . . .

TO PLAY

You start in the hall of the cottage and your aim is to find the statue. Commands are: go north, south, east or west (these can be abbreviated to: n, s, e, w); go up; go down; take; put; look and search. Plus other commands which you have to think of for yourself. You can only carry three objects at a time. As this is a simple adventure, with just ten locations, it should not take very long to solve.

TYPING IN THE LISTINGS

Type in **Foundation 1** and SAVE it.
Type in *Statue,* and MERGE it with **Foundation 1,** and SAVE it.
The *Statue* scenario can then be played.
Keep the separate **Foundation 1** (saved independently on a cassette or microdrive) for use later.

NOTES ON LISTINGS

Foundation 1
Foundation 1 contains the simplest possible vocabulary together with the translator. It also contains the basic command set and the main program loop which allows you to enter commands, act on them and print the result, together with the associated variables and arrays.

100–330 Basic vocabulary, including spaces. The word "xxx" (line 330) is a dummy word, as in the code used it corresponds with the graphic space character (character code 128), indistinguishable in a listing from a normal space (character code 32).

700–710 Removes leading spaces from inputted words. Note that the NEXT f is only carried out until there are no more leading spaces.

750–760 Converts a single character into the word it represents.

800–830 Decoder. Calls the routine at 750 repeatedly for a whole room description, also dealing with space deletion before commas, capital letters after full stops, etc.

6000–6050 Calls decoder subroutine and prints names of any objects or monsters present, if the location has just been entered.
6100–6580 Input command routine. All the basic commands are held there.
6600–6610 Move routine.
6650–6670 Search routine.
6700–6840 Put routine.
6900–6990 You win.
9100–9110 Variables.
9650–9820 READs data for names of objects, special commands and monsters, decodes it and puts it into string arrays.

Scenario 1: *Statue*
340–350 Extra data for this particular scenario.
1100–2050 The locations. Z$ contains the code for the room description, d$ holds the exits. Note that when the game is played, after you input your command, program control returns to the location subroutine to consider special commands. This means that the special commands are considered *before* the normal commands.
6910–6930 You win. Picture DRAWn.
7000–7120 Fighting system.
8000–8210 Deals with special commands and responses.
9200 O$ contains object locations, q$ indicates their visibility.
9500–9600 Coded data for objects, commands and monsters.
9640–9680 Information for size of arrays, and to test whether lines 9500–9600 have been entered correctly.
9910–9920 Introduction. Expand as required.
9930–9980 Picture.

Foundation 1
LISTING

```
   1 REM Foundation 1
   2 REM © S.Robert Speel 1984
  10 GO TO 9000

 100 DATA "You are","entrance"
 110 DATA "there","basement"
 120 DATA "and","single"
 130 DATA "the","stairway"
 140 DATA "room","down"
 150 DATA "door","up"
 160 DATA "north","dark"
 170 DATA "south","pit"
 180 DATA "west","out"
 190 DATA "east","then"

 200 DATA "with"
 210 DATA "going"
 220 DATA "in"
 230 DATA "a "
 240 DATA "s "
 250 DATA "to"
 260 DATA "is"
 270 DATA "no"
 280 DATA ", "
 290 DATA ". "

 300 DATA "     "
 310 DATA "    "
 320 DATA "  "
 330 DATA "it","xxx","kill","fal
l"

 700 FOR f=1 TO 6: IF a$(1)=" "
THEN LET a$=a$(2 TO ): NEXT f
 710 RETURN

 750 LET b=CODE v$: LET c=INT ((
b+1)/4)*10+10: LET d=2+b-INT ((b
+1)/4)*4
 760 RESTORE c: FOR g=1 TO d: RE
AD b$: NEXT g: RETURN

 800 LET ro=0: LET a=0: FOR f=1
TO LEN z$: LET v$=z$(f): IF v$="
l" OR v$="o" OR v$="s" OR v$="w"
 OR v$="i" OR v$="k" THEN PRINT
CHR$ 8;
```

```
 810 GO SUB 750: PRINT CHR$ (COD
E b$(1)-a);b$(2 TO );" ";: LET a
=(v$="o")*32
 820 IF v$="W" OR v$="k" OR v$="
o" OR v$="I" THEN PRINT CHR$ 8;
 830 NEXT f: RETURN

6000 POKE 23692,255: IF NOT ro T
HEN GO TO 6100
6010 GO SUB 800: PRINT '': LET r
o=0
6020 LET a=0: FOR f=1 TO LEN o$:
 IF q$(f)="1" THEN IF (CODE o$(f
)-96)*100+1000=ru THEN LET a=a+1
: PRINT INK 2;("  There is a" AN
D a=1);TAB 13;(CHR$ 8+CHR$ 8+CHR
$ 8+CHR$ 8+"& a " AND a>1);r$(f)
6030 NEXT f

6050 IF mn THEN PRINT PAPER 5;"T
here is a ";m$(mn)

6100 IF at AND mn THEN GO TO 700
0

6200 INPUT "What next? "; LINE a
$: LET c$=a$: PRINT 'TAB 2; BRIG
HT 1; PAPER 7;a$''
6210 LET a$=a$+"
 "
6220 GO SUB 700: RETURN

6300 LET at=0: IF mn THEN LET at
=1
6310 DATA "north","east","south"
,"west","up","down"
6320 RESTORE 6310: FOR f=1 TO 6:
 READ b$: IF a$( TO 2)=b$(1)+" "
 OR a$(4 TO 3+LEN b$)=b$ THEN LE
T rt=CODE d$(f): GO TO 6600
6330 NEXT f

6350 IF a$( TO 4)="look" THEN LE
T ro=1: GO TO ru
6360 IF a$( TO 4)="sear" THEN GO
 SUB 6650
6370 IF a$( TO 4)="take" THEN GO
 SUB 6700
6380 IF a$( TO 3)="put" THEN GO
SUB 6800
6580 PRINT "You cannot do that."
: GO TO ru
```

```
6600 LET rt=(rt-96)*100+1000: IF
 rt<1 THEN PRINT "You cannot go
that way.": LET ro=0: GO TO ru
6610 LET ru=rt: LET ro=1: GO TO
ru

6650 LET a=0: FOR f=1 TO LEN o$:
 IF q$(f)="2" THEN IF (CODE o$(f
)-96)*100+1000=ru THEN PRINT INK
 3;"You find a ";r$(f): LET a=a+
1: LET q$(f)="1": NEXT f
6660 NEXT f: IF NOT a THEN PRINT
 INK 3;"You find nothing."
6670 GO TO ru

6700 LET a$=a$(5 TO ): GO SUB 70
0
6710 FOR f=1 TO LEN o$: IF a$( T
O 3)=r$(f, TO 3) AND ((CODE o$(f
)-86)*100=ru OR q$(f)="£") THEN
GO TO 6740
6720 NEXT f

6730 PRINT TAB 6;"You cannot tak
e that.": GO TO ru
6740 IF o$(f)="£" THEN PRINT TAB
 6;"You already have that!": GO
TO ru
6750 IF q$(f)>"1" THEN GO TO 673
0
6760 IF ob>2 THEN PRINT TAB 4;"Y
ou cannot carry any more.": GO T
O ru
6770 PRINT TAB 8;"Okay": LET ob=
ob+1: LET o$(f)="£": GO TO ru

6800 LET a$=a$(4 TO ): GO SUB 70
0
6810 FOR f=1 TO LEN o$: IF a$( T
O 3)=r$(f, TO 3) AND o$(f)="£" T
HEN GO TO 6840
6820 NEXT f

6830 PRINT TAB 4;"You can't put
down something you don't have!":
 GO TO ru
6840 PRINT TAB 8;"Okay": LET q$(
f)="1": LET o$(f)=CHR$ (ru/100+8
6): LET ob=ob-1: GO TO ru

6900 PRINT "  You have succeeded
 in your    mission.  Well done.
"
```

```
6950 INK 0: FOR f=5 TO 10 STEP 2
: FOR g=1 TO 5 STEP 2: FOR h=1 T
O 5 STEP 2: BEEP .1,f*2: BEEP .1
,f-g*h: NEXT h: NEXT g: NEXT f
6990 STOP

9100 LET mn=0: LET at=0: LET ru=
1100: LET ro=1: LET ob=0
9110 LET lt=0: LET lu=0

9650 RESTORE 9500: LET e=0: GO S
UB 9800: DIM r$(b,c): FOR f=1 TO
 b: LET r$(f)=t$(f): NEXT f
9670 GO SUB 9800: DIM s$(b,c): F
OR f=1 TO b: LET s$(f)=t$(f): NE
XT f
9690 GO SUB 9800: DIM m$(b,c): F
OR f=1 TO b: LET m$(f)=t$(f): NE
XT f

9800 LET a=0: DIM t$(b,c): FOR f
=1 TO b: READ b$: FOR g=1 TO c:
LET c$=CHR$ ((CODE b$(g)+120+g*2
)/2): LET t$(f,g)=CHR$ (CODE c$*
(c$<>"£")+32*(c$="£")): LET a=a+
CODE t$(f,g): NEXT g: NEXT f
9810 IF a<>d THEN PRINT "error i
n code, line ";9500+e*50: STOP
9820 LET e=e+1: RETURN

9900 INK 1: PAPER 6: BORDER 4: C
LS
9990 PRINT '''': GO TO 1100
```

Scenario 1: *Statue*
LISTING

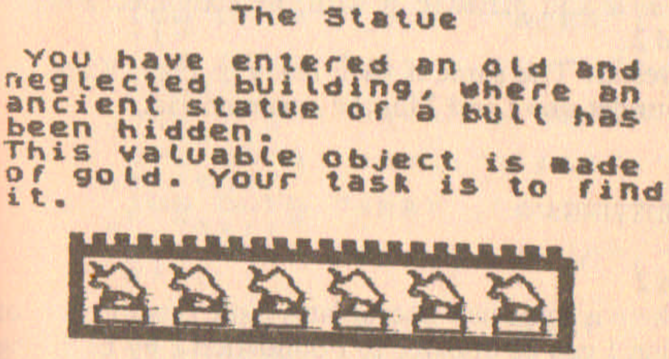
```
   1 REM The Statue
   2 REM © S.Robert Speel 1984

 340 DATA "die","locked","you","
burn"
 350 DATA "goes"

1100 LET z$="£#5/$30'cW7_/?o"
1120 LET d$="00c000": GO SUB 600
0
1150 GO TO 6300

1200 LET z$="#SW3KW,7_/;o"
```

```
1220 LET d$="c00000": GO SUB 600
0
1250 GO TO 6300

1300 LET z$="#SW3K7£_;k?kC+Go"
1320 LET d$="aebg00": GO SUB 600
0
1330 IF a$( TO 2)="s " OR a$( TO
 7)="go sout" THEN IF NOT lu THE
N LET z$="/7c ao": GO SUB 800: GO
 TO ru
1340 IF a$( TO 4)=s$(3, TO 4) TH
EN GO SUB 8200
1350 GO TO 6300

1400 LET z$="#SW3KW7_/?+W0D4o"
1420 LET d$="00e00j": GO SUB 600
0
1450 GO TO 6300

1500 LET z$="#SW3K7£_;kG+Co"
1520 LET d$="df0c00": GO SUB 600
0
1550 GO TO 6300

1600 LET z$="#SW3KW7_/Co"
1620 LET d$="000e00": GO SUB 600
0
1650 GO TO 6300

1700 LET z$="{#SW3KW{w008k+W7_/G
o"
1720 LET d$="0c00h0": GO SUB 600
0
1750 GO TO 6300

1800 LET z$="#SW3KW,7_/;o'cW£0D4
o"
1820 IF t$(1)="1" THEN LET mn=1
1830 LET d$="i0000g": GO SUB 600
0
1840 IF a$( TO 4)=s$(1, TO 4) AN
D mn THEN GO SUB 7100
1850 GO TO 6300

1900 LET z$="#SW3KW7_/?o"
1920 LET d$="00h000": GO SUB 600
0
1950 GO TO 6300

2000 LET z$="#S/(o'{cW008o": REM
 #S/(o'{cW008o"
2010 LET z$=z$+("©cs<o" AND (NOT
 lt OR o$(1)<>"£"))+("´{cW<@o" A
ND lt AND o$(1)="£")
2020 LET d$="0000d0": GO SUB 600
0
2030 IF a$( TO 4)=s$(4) THEN GO
SUB 8000
2050 GO TO 6300

6500 IF a$( TO 4)=s$(2) THEN GO
SUB 8100

6910 POKE 23692,255: FOR f=1 TO
18: PRINT PAPER 0;TAB 31;" ": NE
XT f:
6920 DATA "0305-1-5030506-2-1-30
20301-300030103-809-6-4-1010002-
30102-100-3-2-301-1030101-2-1-51
50000-5-900-80000050200"
6930 INK 6: RESTORE 6910: READ b
$: PLOT 100,50: FOR f=1 TO 108 S
TEP 4: LET a=VAL b$(f TO f+1): L
ET b=VAL b$(f+2 TO f+3): DRAW a*
4,b*4: NEXT f: PLOT 96,90: DRAW
2,0: DRAW -1,1: DRAW -1,-1

7000 PRINT "The ";m$(1);" ";: LE
T z$=" ▝[█o": GO SUB 800: STOP

7100 IF o$(3)<>"£" THEN PRINT "Y
ou can't": GO TO 7000
7110 LET z$="█▀©K/o": GO SUB 800
: PRINT r$(3)
7120 LET mn=0: LET at=0: LET t$(
1)="0": GO TO ru

8000 IF NOT lt OR o$(1)<>"£" THE
N LET z$="S/<k█▀4W@+▀o": GO SUB
800: STOP
8010 IF lt THEN PRINT "You find
the statue.": GO TO 6900

8100 LET a$=a$(6 TO ): GO SUB 70
0
8110 IF o$(2)<>"£" THEN PRINT "Y
ou have no ";r$(2): GO TO ru
8120 FOR f=1 TO LEN o$: IF o$(f)
="£" AND a$( TO 3)=r$(f, TO 3) T
HEN GO TO 8140
8130 NEXT f: PRINT TAB 4;"You ca
n't.": GO TO ru
```

```
8140 IF f=2 THEN LET z$="©a£oH©█
Do": GO SUB 800: LET ob=ob-1: LE
T o$(2)="1": GO TO ru
8150 IF f=1 THEN PRINT TAB 2;"Ok
ay.": LET lt=1: LET ob=ob-1: LET
 o$(2)="1": GO TO ru
8160 PRINT "It won't burn.": GO
TO ru

8200 IF o$(4)="£" THEN PRINT TAB
 2;"Okay.": LET lu=1: GO TO ru
8210 PRINT TAB 2;"You have no ";
r$(4): GO TO ru

9200 LET o$="befijg": LET q$="11
1221"

9500 DATA "LF↑HVF","£FJFN<","1r£
dF<","\Nt@><","llDhhF","£Vfd\£"

9550 DATA "\UZX><","↑UPPf<","p£Z
↑DR","lNDdDL"

9600 DATA "l£DVH"

9640 LET b=6: LET c=6: LET d=351
7
9660 LET b=4: LET c=4: LET d=172
1
9680 LET b=1: LET c=5: LET d=530

9720 DIM t$(1): LET t$="1": GO T
O 9850

9910 PRINT TAB 10;"The Statue"''
" You have entered an old and
 neglected building, where an
 ancient statue of a bull has
 been hidden."
9920 PRINT "This valuable object
 is made     of gold. Your task i
s to find    it."

9930 PRINT '': FOR f=3 TO 24: PR
INT AT 10,f;"▙";AT 14,f;"▀▀": NEX
T f: FOR f=1 TO 3: PRINT AT 10+f
,3;"▌";TAB 24;"▐": NEXT f

9950 RESTORE 6910: READ b$: FOR
e=1 TO 6: PLOT e*28+7,70: FOR f=
1 TO 108 STEP 4: LET a=VAL b$(f
TO f+1): LET b=VAL b$(f+2 TO f+3
): DRAW a,b: NEXT f: NEXT e
```

HINTS ON PLAY

This scenario is very easy to win, and as long as you make a good map of where the objects are, there should be few difficulties. The map was originally planned on a square grid, so mapping is uncomplicated, even though there is more than one level. Remember that to use an object you have to take it first.



```
             The Statue
 You have entered an old and
neglected building, where an
ancient statue of a bull has
been hidden.
This valuable object is made
of gold. Your task is to find
it.
```

FOUNDATION 2 and SCENARIO 2

Temple

Temple is more complicated than *Statue*. There are more locations, more objects and more enemies. In this scenario you are after the hoard of offerings collected in a temple. It is known that there is a tunnel leading from the treasure chamber to the outside world, but at its end there is a door which can only be opened from the inside. So, once you find the treasure chamber it's easy to escape. The only problem is finding it.

You arrive at the temple entrance to find the guard absent, and the door invitingly open. . . .

TO PLAY

You start at the entrance to the temple and your aim is to get to the treasure room. Your basic commands are the same as for the previous scenario, but this time there are more special commands which you have to think of.

TYPING IN THE LISTINGS

MERGE **Foundation 2** with **Foundation 1** and SAVE it.
Type in *Temple* and MERGE it with the combined **Foundations 1 and 2**.
SAVE the result. The game is then ready to play. Keep the separate **Foundations 1 and 2** for the next scenario.

NOTES ON LISTINGS

Foundation 2
100–250 There is some new vocabulary. Note that the lines of Foundation 1 can be updated to Foundation 2 by EDITing and adding the new words, rather than typing in the whole lines again.

Scenario 2: ***Temple***
340–370 Extra vocabulary for this scenario.
1100–2520 The locations.
7000–7900 You meet your doom.
8000–8540 Special commands and responses. The greater complexity of this scenario means there are more special situations than in *Statue*.
9200–9680 The new object, command and monster set.
9900 No introduction has been included, as the program nearly fills the 16K machine. A very short introduction could be added, or a longer one if you have more than 16K.

Foundation 2
LISTING

```
   1 REM Foundation 2
   2 REM © S.Robert Speel 1984
 100 DATA "You are","entrance","
small","you"
 110 DATA "there","basement","lo
ck","which"
 120 DATA "and","single","high",
"are"
 130 DATA "the","stairway","wide
","narrow"
 140 DATA "room","down","table",
"at"
 150 DATA "door","up","centre","
ladder"
 160 DATA "north","dark","single
","wall"
 170 DATA "south","pit","pile","
happens"
 180 DATA "west","out","furnitur
e","exit"
 190 DATA "east","then","floor",
"of"

 200 DATA "with","your","plank",
"nothing"
 210 DATA "going","passage","'s"
,"ed"
 220 DATA "in","on","lead","but"
 230 DATA "a ","large"

6150 IF tt THEN GO SUB 8500

7000 PRINT "the ";m$(mn);" kills
 you. ": GO TO w
7900 PRINT TAB 6;"You are dead."
: STOP

8800 RESTORE 8900: READ a,b,c,n$
: PLOT a,b
8810 FOR f=1 TO LEN n$: LET a1=C
ODE n$(f): DRAW (INT (a1/10)-8)*
c,(a1-(INT (a1/10)*10)-4)*c: NEX
T f
8820 RETURN

9000 LET z=800: LET y=6300: LET
x=6000: LET w=7900

9120 LET ul=0: LET dd=0: LET de=
0: LET df=0: LET tt=0
```

Scenario 2: *Temple*
LISTING

```
   1 REM Scenario: Temple
   2 REM © S.Robert Speel 1984

 290 DATA ". ","way","cloth","fi
replace"
 300 DATA "    ","treasure","sev
eral","crossroads"
 310 DATA "    ","turn","block","
lever"
 320 DATA "    ","end","stop","gol
d"
 330 DATA "it","xxx","kill","fal
l"
 340 DATA "bed","bolt","temple",
"altar"
 350 DATA "priest","sleeping","s
toreroom","pursuit"
 360 DATA "dog","key","guards","
secret"
 370 DATA "wake"

1100 LET z$="#6/$_/{▊o_LGc/s7o"
1120 LET d$="0b00": GO SUB x: GO
 TO y

1200 LET z$="{#SWXPwO_Wvo'.p[_;k
?kG+Co_/;"
1220 GO SUB 8050: LET d$="fcma":
 GO SUB x
1230 IF a$( TO 2)="n " OR a$( TO
 4)="go n" THEN GO SUB 8000
1240 IF a$( TO 3)=s$(2, TO 3) TH
EN GO SUB 8100
1250 GO TO y

1300 LET z$="#SW1POC+GkKWX36/G!o
'.s7[S/;+?s>[J/Po"
1320 LET d$="edib": GO SUB x
1330 IF (a$( TO 2)="n " OR a$( T
O 4)="go n") AND NOT ul THEN LET
 z$="/7c)Ro": GO SUB z: GO TO ru
1340 IF a$( TO 4)=s$(1) THEN PRI
NT ("you have no key" AND o$(1)<
>"£");("  okay" AND o$(1)="£"):
LET ul=ul+(o$(1)="£"): GO TO ru
1350 GO TO y

1400 LET z$="{#SWX3oWs-▚cS/9J/3o
'cW%zT/>o'cW,F_/Co"
```

```
1410 LET z$=(z$( TO LEN z$-1)+"k
V©csyRKW7o" AND dd)+(z$ AND NOT
dd)+("'cWw↑7_/?o" AND de)
1420 LET d$="000c": IF de THEN L
ET d$(3)="g"
1430 IF dd THEN LET d$(4)="0"
1440 GO SUB x: IF a$( TO 4)=s$(7
) THEN LET z$=("WX7▀[4+sy[/C7_/S
3o" AND NOT dd)+("NBo" AND dd):
GO SUB z: LET dd=1: LET tt=0: GO
 TO ru
1450 IF a$( TO 4)="sear" AND NOT
 de THEN LET z$="'cW▙7S/w?>o": G
O SUB z: LET de=1: GO TO ru
1460 GO TO y

1500 LET z$="{#SW%▖▘kKW,F_/?o'.A[
JEk{+X▖[Jq'o"
1510 IF o$(5)="e" THEN LET z$=z$
( TO LEN z$-1)+"k+W:T/Io": LET q
$(5)="0"
1520 LET d$="00c0": GO SUB x: GO
 TO y


1600 LET z$="#SWXk13k{Ku▀[o'.s▟
T/▀[oW,kXFc_/{?o"
1610 IF o$(3)="f" THEN LET z$=z$
+"'cWMT/{Io": LET q$(3)="0"
1620 LET d$="00b0": GO SUB x
1630 IF a$( TO 2)="s " OR a$( TO
 4)="go s" THEN GO SUB 8000
1650 GO TO y

1700 LET z$="#5W2PKW76/;!o_/?k©x
[Co"
1720 LET d$="d00h": GO SUB x
1730 IF NOT df THEN IF a$( TO 2)
="w " OR a$( TO 4)="go w" THEN L
ET z$="&xCk+H▘s4W@o": GO SUB z:
GO TO 7900
1750 GO TO y

1800 LET z$="{#SW2PsOGk#x[;o/CF!
[SW7o"
1820 LET d$="g00j": GO SUB x: GO
 TO y

1900 LET z$="#SW23K{7[_;+?o'cWzS
/G>o"
1910 LET mn=4: IF o$(6)="i" THEN
 LET mn=0
1920 LET d$="c0j0": GO SUB x
```

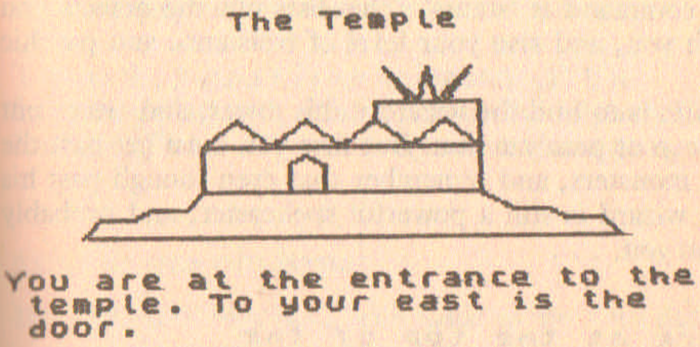
```
1930 IF a$( TO 4)=s$(4) THEN IF
o$(6)="£" THEN LET o$(6)="i": LE
T x$="the wolfhound grabs the bo
ne and settles down to eat. ": L
ET mn=0: LET at=0: PRINT x$: GO
TO ru
1940 IF a$( TO 4)=s$(7) THEN LET
 df=1: PRINT "there is a low rum
ble. ": LET tt=tt+(tt=0): GO TO
ru
1950 GO TO y

2000 LET z$="%S/-▜@w3o'£UX▀xW{5+
Wro7[._;+Co"
2010 IF NOT dd THEN LET z$=z$+"/
-w▜c'ʎ+▘[&o": GO SUB z: GO TO 79
00
2020 LET z$=z$+"'cg-▜'o/-s▜cK/▟k
S{L█o"
2030 LET d$="ih0k": GO SUB x: GO
 TO y

2100 LET z$="{%S/t3oA[J"y[.T/Ik+
W▖T/7{} [▌oW2PU[8_W}R7kKW▄S/}o"
2110 GO SUB z: GO TO 6900

2200 LET z$="{#SWX3KW,▀kT*W▜{c▖o
'£W7_/Co"
2220 LET d$="000m": GO SUB x: GO
 TO y

2300 LET z$="#SW%PK7[_CkG+?oW1PU
[;o"
2320 LET d$="blno": GO SUB x: GO
 TO y

2400 LET z$="{#SWX3KW▀oW▜c▖S{©o'
cW7_/w{;o"
2420 LET d$="m000": GO SUB x: GO
 TO y

2500 LET z$="{#SW3KWs{rxW2▀k+W,7
_/GoW{s▜c▖T/▀o"
2520 LET d$="0m00": GO SUB x: GO
 TO y

6500 IF a$( TO 4)=s$(8) OR a$( T
O 4)=s$(9) THEN GO SUB 8200

7900 PRINT TAB 6;"You are dead."
: STOP
```

```
8000 IF o$(3)<>"b" AND o$(5)<>"b
" THEN LET z$="&▀4/@o": GO SUB z
: GO TO w
8010 PRINT " you cross the pit o
n the ";(r$(3) AND o$(3)="b" AND
 q$(3)="0");(r$(5) AND o$(5)="b"
 AND q$(5)="0")
8020 PRINT : RETURN

8050 IF o$(3)<>"b" AND o$(5)<>"b
" THEN LET z$=z$+"W1@£y[/po": RE
TURN
8060 LET z$=z$+"cW@ksKW"
8070 LET a=3*(q$(3)="0" AND o$(3
)="b")+5*(q$(5)="0" AND o$(5)="b
")
8080 IF NOT a THEN LET a=3*(o$(3
)="b")+5*(o$(5)="b"): LET q$(a)=
"0"
8090 LET b=77*(a=3)+58*(a=5): LE
T z$=z$+CHR$ b+"T©o": LET q$(3)=
"1": LET q$(5)="1": LET q$(a)="0
": RETURN

8100 FOR f=1 TO 4: IF a$(4)=" "
THEN LET a$=a$( TO 3)+a$(5 TO ):
 NEXT f
8110 IF a$(4 TO 6)<>r$(3, TO 3)
AND a$(4 TO 6)<>r$(5, TO 3) THEN
 RETURN
8120 LET a=3*(a$(4)=r$(3,1))+5*(
a$(4)=r$(5,1))
8130 IF o$(a)<>"£" THEN RETURN
8140 IF o$(8-a)<>"b" THEN LET q$
(a)="0"
8150 RETURN

8200 LET a=ru/100: IF a=22 OR a=
24 OR a=25 THEN GO TO 8300+(a$(
TO 4)=s$(9))*100
8210 IF a=16 THEN GO TO 8350
8220 RETURN

8300 LET z$="&▀/▜kVHt/▟▀&o": GO
SUB z: GO TO w

8350 LET z$="/██8+▀&o": GO SUB z
: GO TO w

8400 LET z$="/▜█£+▀£&o": GO SUB
z: GO TO w
```

```
8500 DATA "the guards have waked
","the guards are trying to
cross the pit","the guards have
crossed the pit","the guards are
 running up the passage"
8510 IF tt>5 THEN RETURN
8520 IF tt=5 THEN PRINT PAPER 5;
"The guards catch you and kill
 you.": GO TO w
8530 RESTORE 8500: FOR f=1 TO tt
: READ b$: NEXT f: PRINT PAPER 5
;"***";b$;"***"
8540 LET tt=tt+1+(tt=1 AND (o$(3
)="b" OR o$(3)="f" OR o$(5)="b"
OR o$(5)="f"))*2: RETURN

8900 DATA 70,120,3,"©y©y©y©yY,v(
NF0n,RV[illegible]hQR6@6,,,,,,,QR[illegible]wig06@
||||hYPSz|g|f@,,,,,,,,,,,,,,,,j|i
|j|"

9100 LET t=0: LET mn=0: LET at=0
: LET ru=1100: LET ro=1: LET ob=
0: LET lt=0
9200 LET o$="odflecm": LET q$="2
201010"

9500 DATA "\Nt@><","JbPX><","f\O
\T<","Jb£V><","↑FJHH£","Jb↑J><",
"Jh£↑X<"
9550 DATA "p£Z↑","fnj@","jNDH","
RNLH","lrLJ","f\DF","fnZX","\VZX
","tFXJ"
9600 DATA "fhTJdd:864","VUPP↑£LB
\\","TnDdF<:864","tbZLNZdT>4"

9640 LET b=7: LET c=6: LET d=366
1
9660 LET b=9: LET c=4: LET d=375
8
9680 LET b=4: LET c=10: LET d=35
75
9710 GO TO 9850

9990 PRINT 'TAB 10;"The Temple":
 GO SUB 8800: PRINT '''''''''':
GO TO 1100
```

HINTS ON PLAY

You may get frustrated in this scenario, faced with a no-win situation. Remember that a lever is only installed for a purpose, which must be beneficial to whoever is intended to use it. Also remember that whatever obstacles are in your way are likely to be obstacles to your enemies as well. Incidentally, as the temple represents a cult which uses many martial arts, you are likely to be outclassed in fair fights.



```
            The Temple
```

```
You are at the entrance to the
 temple. To your east is the
 door.
```

FOUNDATION 2 and SCENARIO 3

Wizard's Tower

For many years a wizard with vast riches has ruled over the land from his tower, and wielded great power. In recent times, however, he has done little, living a hermit-like existence in his tower, guarded by various monsters. You feel it is high time that the old wizard was removed from office, to make way for a more dynamic, enterprising person – yourself. To this end you have decided to enter the tower, find the magician and dispose of him.

TO PLAY

You have two potent spells – a sleeping spell and a petrifying spell which paralyzes an enemy. You use these with the commands 'cast petrifyspell' and 'cast sleepspell'. Each time you cast a spell you lose psychic energy, 1 for a sleepspell, 2 for the more powerful petrifyspell. As you start with only 4 energy points, you cannot afford to waste spells. Physically you are not up to very much, so monsters will be able to destroy you relatively easily. Your spells should take care of them, but there are ways other than spells to get past a monster. . . . Another command is 'status'. This lists out the objects you have with you, and also your level of resistance and psychic energy.

Your aim is to find the wizard in his tower, and either put him to sleep or paralyze him. But first you must get past the guardian monsters, and remember that even though past his best, the wizard is still a powerful spellcaster, and probably faster than you. . . .

```
You are at the top of the
stairway in a room with doors
to north, south and west.

   search

You find nothing.

   status


Objects carried:
                   jug
              Energy left: 4
              Resistance:4

   go up

You cannot go that way.
```

TYPING IN THE LISTING

There are more coded words than normal characters available, so some user defined characters have been included. To make these appear different in the listings, the characters have bars above and below. When you type in the listing, first type line 9910 and run it before typing in the rest of the listing. This gives UDGs bars. Then when you see a barred letter in the listing, go into graphics mode to type it, and you should get the barred character. A few of these letters look a little odd, for example in line 7710 the barred letters are N, I, H and U. Most of the words which have barred character codes are used infrequently, so there are not many to type in.

Type in the *Wizard's Tower* listing and SAVE it.
Load **Foundation 1**, MERGE with **Foundation 2**, and then MERGE *Wizard's Tower* to play the complete game.

EXPLANATION OF LISTING

100–300 Additions have been made to the word data list, and slight alterations made in order.
700–830 Due to more words being added, the decoder has had to be modified.
1100–2770 Locations. Note use of barred capital letters – these are user defined graphics.
6100 Check if monsters are able to attack you.
6380–6410 New commands recognized here.
6900–6930 You win. Picture drawn.
7000–7010 Monster attacks. You can sustain several levels of being damaged before you collapse, so can try different methods of overcoming a monster. Note that if you flee, the monster will follow you.
7100–7180 You cast a spell. sp is the psychic energy you have left.
7500–7530 Status command.
7700–8310 Special circumstances. There are rather a large number of these now.
8900 Data for pictures.
9010–9120 Changed variables.
9200 Object positions.
9200–9710 Objects, monsters, special actions.

9910 Poke UDGs with bars to make it easier to type in listings. Note this line can be deleted when game fully operational.

Scenario 3: *Wizard's Tower*
LISTING

```
   1 REM Scenario:
     Wizard's Tower
   2 REM © S.Robert Speel 1984

 110 DATA "there","basement","lo
cked","which"

 210 DATA "going","has","'s","an
"

 230 DATA "a ","large","too","on
e"

 240 DATA "s ","empty","it","oil
"

 250 DATA "to","jug","say","can"
 260 DATA "is","not","top","eat"
 270 DATA "no","box","see","bell
"

 280 DATA ", ","open","burn","la
mp"
 290 DATA ". ","fall","will","ta
nk"

 300 DATA "      ","have","book","
cast"
 310 DATA "     ","help","full","g
oes"
 320 DATA "  ","when","leak","da
mp"
 330 DATA "tower","xxx","spire",
"crack"
 340 DATA "sting","chair","shelf
","tough"
 350 DATA "spell","straw","broke
n","thick smoke"
 360 DATA "waste","mirror","barr
el","match"
 370 DATA "manage","appear","pre
pare","giving"
 380 DATA "strewn","cuttlefish b
ones","propped against","through
 the ceiling"
 390 DATA "goldfish","surrounded
","petrified","filthy"

 400 DATA m$(1),m$(2, TO 7),m$(3
, TO 6),m$(4, TO 5)
 410 DATA "other","passage","for
","put"
 420 DATA "turn","do"

 800 LET ro=0: LET a=0: FOR f=1
TO LEN z$: LET v$=z$(f): IF v$="
[" OR v$="o" OR v$="s" OR v$="w"
 OR v$="{" OR v$="k" OR v$="@" T
HEN PRINT CHR$ 8;

1100 LET z$="{#6/7_/s▙@©k*PWs,▀o
'cWj{T/>o/7c_s/;o"
1120 LET d$="g00000": GO SUB 600
0
1130 IF a$( TO 4)=s$(1) THEN LET
 z$="W{OR[o": GO SUB 800: LET mn
=4: GO SUB 6000
1150 GO TO 6300

1200 LET z$="#SWX3o'{cWh'o7[._s;
+Go"
1220 LET d$="dg0000": GO SUB 600
0
1230 IF a$( TO 4)=s$(9) THEN PRI
NT "There is a ";m$(1);" ";: LET
 z$="S]o]{▀[&o": GO SUB 800: GO
TO 7900
1241 IF a$( TO 4)=s$(2) AND q$(9
)="1" THEN IF a$(6 TO 8)=r$(9, T
) 3) THEN LET z$="]cYXo": GO SUB
 800: GO TO ru
1250 GO TO 6300


1300 LET z$="#SWX3o'{.W5+▄[o/s,▛
c_/Co"
1310 IF o$(5)="c" THEN LET z$=z$
+"R\k▟▌c{E/>KWno"
1320 IF o$(5)<>"c" THEN LET z$=z
$+"T/IcW▟▙o"
1330 LET d$="000g00": GO SUB 600
0
1340 IF o$(5)="c" THEN IF a$( TO
 4)=s$(2) THEN IF a$(6 TO 9)=r$(
5, TO 4) THEN LET o$(5)="£": LET
 z$="/▙p[k+WD▀H[o": GO SUB 800:
LET mn=4: GO SUB 6000
1350 GO TO 6300
```

```
1400 LET z$="{#SWX3K{7[_/?+{Go'.
A[J\▟[' k+Z#cyJ↑o"
1410 IF t$(5)>"0" THEN LET mn=5
1420 LET d$="0eb000": GO SUB 600
0
1430 IF a$( TO 4)=s$(2) THEN GO
TO 8000
1450 GO TO 6300

1500 LET z$="#SWX13wK7[_GkC+s?oW
X5cS/9J/3k+]cwIK▄[o"
1520 LET d$="0fgd00": GO SUB 600
0
1550 GO TO 6300

1600 LET z$="{#SWX\3o'cW0#U[Gk+P
{FI._;+Co"
1620 LET d$="h00ei0": GO SUB 600
0
1650 GO TO 6300

1700 LET z$="#SW1Qo/{F_/©c_/s?k+
'.7[_w/GkC+;o"
1720 LET d$="ecab00": GO SUB 600
0
1750 GO TO 6300

1800 LET z$="#SWXV2{3KWrJHs'o/Z7
c_/w?o"
1820 LET d$="00f000": GO SUB 600
0
1830 IF a$( TO 4)=s$(2) THEN IF
a$(6 TO 9)=r$(3, TO 4) THEN IF o
$(3)="£" THEN PRINT "Don't be gr
eedy, you have one  already.":
GO TO ru
1850 GO TO 6300

1900 LET z$="#6/eJ/s{0SW3K7[{_;k
?+Co"
1920 LET d$="j0ml0f": GO SUB 600
0
1950 GO TO 6300

2000 LET z$="#5/N03oW▌Ju[cT/>k{▜
u[.D{T/Ik+▄[.T/{5o'.7[_/w?+Co"
2020 LET d$="00ik00": LET mn=3:
GO SUB 6000
2050 GO TO 6300

2100 LET z$="#SW%k"3wKEDT/sIo'.7
[_C+G"
2120 LET d$="0j0p00": GO SUB 600
0
2150 GO TO 6300

2200 LET z$="£#SWik-3o'cWX\rS£/9
o'cZFk*C_/Go"
2220 LET d$="0i0000": GO SUB 600
0
2250 GO TO 6300

2300 LET z$="w#SW2▚o7£U;+Co"
2320 LET d$="i00n00": GO SUB 600
0
2350 GO TO 6300


2400 LET z$="£&.SWK3k£sIK▖+s£EoF
£.s_;+Go"
2410 IF t$(2)="1" THEN LET mn=2
2420 LET d$="om0000": GO SUB 600
0
2430 IF t$(2)>"0" THEN IF a$( TO
 4)=s$(7) AND a$(6 TO 9)=m$(2, T
O 4) THEN GO TO 7300
2450 GO TO 6300

2500 LET z$="£#SWX3K£R\u▌o'.7£_;
+?o"
2510 IF q$(4)="0" THEN LET z$=z$
+"'cWX£▄sE/C>o"
2520 IF q$(4)="2" THEN LET z$=z$
+"W££X▟▄cT/£>o"
2530 LET d$="p0n000": GO SUB 600
0
2540 IF q$(4)="0" THEN IF a$( TO
 4)=s$(2) AND a$(6 TO 9)=r$(4, T
O 4) THEN LET z$="]p£+C▞k+W£OR£o
": GO SUB 800: PRINT : LET q$(4)
="2": LET mn=4: GO TO ru
2550 GO TO 6300

2600 LET z$="£#SW3o'c£N'VW:*£U££
Go'.7£_?+££Go"
2610 LET y$=" the door is locked
."
2620 LET d$="0ko0q0": GO SUB 600
0
2630 IF a$( TO 2)="e " OR a$( TO
 7)="go east" THEN IF NOT ul THE
N LET z$="/7c)o": GO SUB 800: GO
 TO ru
```

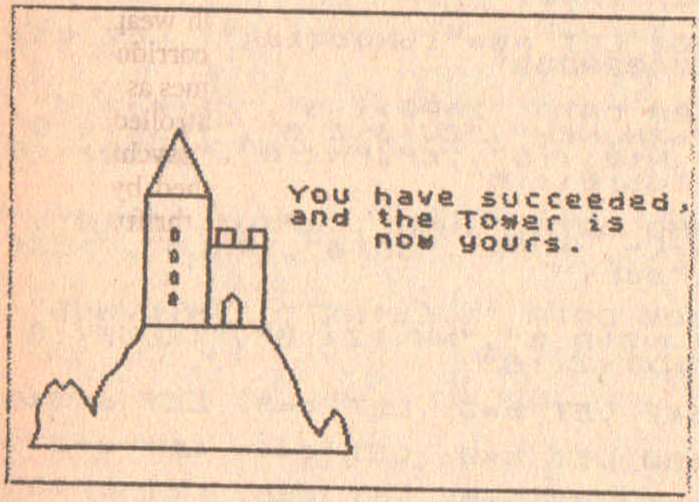
```
2640 IF a$( TO 4)=s$(8) THEN PRI
NT ("you have no "+r$(6, TO 3) A
ND o$(6)<>"£");("okay" AND o$(6)
="£"): IF o$(6)="£" THEN LET ul=
1: GO TO ru
2650 GO TO 6300

2700 LET z$="#S/▀o'cWAJ▞▄{o/▀,Fc
/:#UI4o"
2720 LET d$="00000p": GO SUB 600
0
2730 IF a$( TO 4)=s$(9) AND q$(9
)="1" AND o$(6)="q" AND NOT cr T
HEN LET z$="&l/ho'cW": GO SUB 80
0: PRINT r$(6, TO 4);"in it.": L
ET q$(6)="1": GO TO ru
2740 IF q$(9)="2" THEN IF a$( TO
 4)="sear" THEN LET z$="&iWho":
GO SUB 800: LET q$(9)="1": GO TO
 ru
2750 IF a$( TO 4)=s$(2) AND q$(9
)="1" THEN IF a$(6 TO 8)=r$(9, T
O 3) THEN LET z$="]cYXo": GO SUB
 800: GO TO ru
2760 IF a$( TO 4)=s$(2) AND cr=0
 AND q$(6)="1" THEN IF a$(6 TO 8
)=r$(6, TO 3) THEN LET z$="]p[5W
▪S/s{Io": GO SUB 800: LET cr=1:
LET q$(6)="3": GO TO ru
2770 GO TO 6300

6100 IF at AND mn THEN GO TO 700
0+(mn=3)*710

6390 IF a$( TO 4)="stat" THEN GO
 TO 7500
6400 IF a$( TO 10)="cast sleep"
THEN GO TO 7100
6410 IF a$( TO 10)="cast petri"
THEN GO TO 7150

6420 IF a$( TO 4)=s$(4) OR a$( T
O 4)=s$(5) THEN GO TO 8100
6430 IF a$( TO 4)=s$(3) OR a$( T
O 4)=s$(6) THEN GO TO 8200
6440 IF a$( TO 6)=m$(6, TO 6) AN
D o$(5)="£" THEN GO TO 8300

6840 PRINT TAB 8;"Okay": LET o$(
f)=CHR$ (ru/100+86): LET ob=ob-1
: IF f=1 AND o$(8)="£" THEN LET
ob=ob-1: LET o$(8)="d"
6850 GO TO ru
6900 FOR f=1 TO 16: PRINT : NEXT
 f: PRINT AT 10,13;"You have suc
ceeded,";AT 11,13;"and the Tower
 is";AT 12,17;"now yours."
6910 PRINT '''''''''
6920 GO SUB 8800
6930 FOR f=1 TO 4: PLOT 60,50+f*
8: DRAW 0,3: DRAW 1,1: DRAW 1,-1
: DRAW 0,-3: DRAW -2,0: NEXT f

7000 PRINT "the ";m$(mn);" attac
ks you. ": LET rs=rs-INT (RND*2)
-1: IF rs<1 THEN GO TO 7900
7010 PRINT " You are ";("badly"
AND (rs=3 OR rs=2));("grievously
" AND rs=1);" wounded.": GO TO 5
200

7100 LET sp=sp-1: IF sp<0 THEN P
RINT "You have no energy left to
 cast spells!": GO TO ru
7110 IF NOT mn THEN PRINT "Okay.
": GO TO ru
7120 IF mn<>3 THEN PRINT "The ";
m$(mn);" falls asleep.": LET t$(
mn)="0": LET mn=0: GO TO ru
7130 LET z$="NBK/NcεY▀o": GO SUB
 800: PRINT : GO TO 7710

7150 LET sp=sp-2: IF sp<0 THEN G
O TO 7100
7160 IF NOT mn THEN PRINT "Okay.
": GO TO ru
7170 IF mn=3 THEN GO TO 7700+100
*lo
7180 PRINT "the ";m$(mn);" is pe
trified.": LET t$(mn)="0": LET m
n=0: GO TO ru

7200 LET z$="/NcJo": GO SUB 600:
 GO TO 6900

7300 LET z$=("]f[/H+zo" AND o$(3
)="£")+("&tNB]_fo" AND o$(3)<>"£
"): GO SUB 800: IF o$(3)="£" THE
N LET t$(2)="0": LET at=0: LET m
n=0: LET o$(3)="h"
7310 GO TO ru

7500 PRINT ''"Objects carried:";
: LET a=0: FOR f=1 TO LEN o$: IF
 o$(f)="£" THEN PRINT TAB 15;r$(
f): LET a=a+1
```

```
7510 NEXT f: IF NOT a THEN PRINT
 " none."
7520 PRINT TAB 10;"Energy left:
";sp;TAB 10;"Resistance:";rs
7530 GO TO ru

7700 LET z$="W[&▣L▛kw{": GO SUB
800
7710 LET z$="/▣i [&kv [Ww▛k +&T5_Ws
▣o&.SW▣ro&biDo'cN&b▣o"
7720 GO SUB 800: STOP

7800 LET z$="/▣kS/▘kbdi&o&vL▛o"
7810 LET z$="/▣c▣o": PRINT : GO
SUB 800: GO TO 6900

8000 LET b$=a$(5 TO ): FOR f=1 T
O 5: IF b$(1)=" " THEN LET b$=b$
(2 TO ): NEXT f
8010 IF b$( TO 3)<>r$(8, TO 3) T
HEN GO TO 6300
8020 IF o$(1)="£" THEN LET z$="&
▣↑SL£o": LET o$(8)="£": GO SUB 8
00: GO TO ru
8030 IF o$(5)="£" THEN LET z$="&
▣↑SLnkVJ}[Do": GO SUB 800: GO TO
 ru
8040 LET z$="&tN_▣]So": GO SUB 8
00: GO TO ru

8100 IF o$(7)<>"£" THEN LET z$="
&tN_m/{": GO SUB 800: PRINT a$(6
 TO LEN a$-12);"with.": GO TO ru
8110 LET a$=a$(6 TO ): FOR f=1 T
O 3: IF a$(1)=" " THEN LET a$=a$
(2 TO ): NEXT f
8120 LET o$(7)="z": LET v$="▖":
GO SUB 750: IF a$( TO 5)=b$ THEN
 LET z$="]cY"_mo": GO SUB 800: G
O TO ru
8130 LET o$(7)="0": IF a$( TO 3)
=r$(8, TO 3) AND o$(8)="£" THEN
LET z$="]m[k▣D▌o": GO SUB 800: L
ET at=0: LET lo=(ru=2000): LET o
$(8)="d": GO TO ru
8140 LET z$="]qdmo&▛/▙o": GO SUB
 800: GO TO ru

8200 PRINT " A ";m$(6, TO 6);: L
ET z$="▣[o]a[]wqx&|&asss": GO SU
B 800: PRINT "*";m$(6, TO 6);: L
ET z$="xo": GO SUB 800: PRINT ""
": GO TO ru
8300 IF ru=2700 AND cr=1 THEN PR
INT "the ";m$(6, TO 6);" gets th
e ";r$(6, TO 3): LET z$="DJ/" +5[
]T/Io": GO SUB 800: LET q$(6)="1
": LET cr=2: GO TO ru
8310 PRINT " The ";m$(6, TO 6);:
 LET z$="a[": LET a=INT (RND*2):
 LET z$=z$+("go" AND a=0)+("&b█o
" AND a=1): GO SUB 800: GO TO ru

8900 DATA 50,50,3,"XXXXlldd,,||P
R|rV@R@V@RJPPR,,|lhW_]Q@l↑XXVPPR
ed[ek]Z,,,,,,,,,,,£W£_]_]eV£j£_£"

9010 LET t=0: LET cr=0: LET lo=0
9100 LET mn=0: LET at=0: LET ru=
1100: LET ro=1: LET ob=0: LET lt
=0
9120 LET ul=0: LET dd=0: LET de=
0: LET df=0: LET tt=0: LET rs=4:
 LET sp=4

9200 LET o$="lbhocqkdq": LET q$=
"100022202"

9500 DATA "ZnP@><:8","Jbr@><:8",
"TbZHJN£H","£Vfd\£:8","↑F\£><:8"
,"\Nt@><:8","£FJFN<:6","dVZ@><:8
","Jbr@><:8"

9550 DATA "jV↑N","nFXJ","fbZR","
Jnf\","↑VPP","jnF@","RNLH","p£Z↑
","ddL\"

9600 DATA "lJ£d↑NXT","ThTLJNV8",
"tUvBbD:8","NN\↑Z<:8","l£DVH<:8"
,"NXT\Z<:8"

9640 LET b=9: LET c=8: LET d=519
1
9660 LET b=9: LET c=4: LET d=373
3
9680 LET b=6: LET c=8: LET d=425
1
9710 GO TO 9850

9910 FOR f=144 TO 164: POKE USR
CHR$ f,255: POKE USR CHR$ f+7,25
5: NEXT f
9920 DIM t$(8): LET t$="11111111
"

9990 PRINT TAB 10;"Wizard's Towe
r"'''': GO TO 1100
```

HINTS ON PLAY

It is a good idea to explore as much as you can, and it may be necessary to retrace your steps at some points. Do not become frustrated if you get completely stuck at some point – take a review of your various objects and see if one of these can help. Shortage of psychic energy is an acute problem, but if you do not waste it, you will have enough spells to get through and win the game.

Finally, it is not necessary to carry out certain actions – find objects, kill monsters – in the order in which you come across them, and by choosing a different route to the 'natural' one you may pass through more quickly, or use up less spells.



```
You have succeeded,
and the Tower is
    now yours.
```

3. Warrior Mage System

Warrior Mage is a Random adventure system. The setting is an underground maze, called the 'upper reaches', which is inhabited by various monsters. Your aim, initially, is to get a certain amount of gold. Gold is found in the monsters' lairs, so if you meet a monster just wandering through the maze, it will not have treasure.

You are the warrior mage, and can use both weapons and magic. Weapons lie around in the rooms and corridors of the maze, and you may use any weapon as many times as you like, unless it breaks. Magic spells, however, are controlled by your psychic energy. Each time you use a spell, your psychic energy (initially 5) decreases. It can only be replenished by finding psi-stones, and as these are rare, you must be thrifty in your use of magic.

The system expands to include high-res pictures of the monsters. Then carnivorous plants and magical potions are introduced. Pits, traps and special pool rooms are also added in later stages, as extra hazards. Finally, a lower level is added to the maze, with new, much tougher monsters, and much greater treasures. . . . The aims also change, with a final range of three alternative aims.

CORE PROGRAM – WARRIOR MAGE 1

Monsters and Spells

This program contains all the essential parts of the system. As the maze set-up is complex, and the fighting routines very comprehensive, it has not been possible to reduce this program to a shorter first stage listing. Note that on running the program you have to wait for a couple of minutes while the maze is set up.

TO PLAY

At the beginning of the game, you are asked which level of difficulty you wish to play at. In level 1, you must collect 30 gold coins; you need 60 in level 2; 120 in level 3; and 160 in level 4.

The maze is different each time the program is run, but you always start off in a room with four exits and a stairway, and you are strongly advised to draw a map as you travel around. To move around the maze, use go north, south, east and west, which can be abbreviated in the usual way. For the various weapons lying around (you can carry a maximum of 10 objects), use take or put, followed by the object's name, e.g., 'take spike'. Fighting involves hitting the monster with a chosen weapon and the monster biting back in turn until one of you dies. The computer asks which weapon you use, and you type in the name of the weapon. If you have no weapons, typing 'fist' will let you punch the monster, and typing 'spell' will let you use magic.

There are six type of weapons. Swords are the easiest to hit with, and are therefore the most dependable weapons. Daggers, also, do not miss often, but neither do they cause much damage. Axes are fairly useful all-round weapons. Clubs and maces can inflict the most damage, but are very unwieldly and difficult to hit with. They have the advantage that they cannot be blunted (although they can be broken) in combat with thickly armoured monsters. A spike, very unreliable, is a last-ditch weapon.

There are six monsters, of which the sabretooth is easily the toughest. Dire wolves and hellcats are fairly strong. Snakes are weak, but just a couple of snakebites can be lethal. Armadillos have tough shells which are able to blunt edged weapons, making them hard to kill, but they are feeble at biting you. Giant rats are the weakest monsters.

During a fight your resistance is reduced, and you can recover afterwards with the command 'rest'. This increases your resistance by one or two points, up to a maximum of 10. You also gradually recuperate as you wander through the maze, by 0.1 point at a time. You are not killed unless your resistance drops to zero, so you can be alive with a resistance of 0.1.

When you find gold, it is automatically added to your collection. When you have enough, return to the room where you entered the maze, and use the stairway by typing 'escape'. You are awarded the title of warrior mage class A, B, C or D, depending on the difficulty level you chose (the highest, most difficult grading to achieve being A).

You have a 'status' command, which gives your current resistance, gold found, psychic energy, and a list of objects taken. You start the game with a randomly chosen weapon: typing 'status' as your first command will tell you what that weapon is.

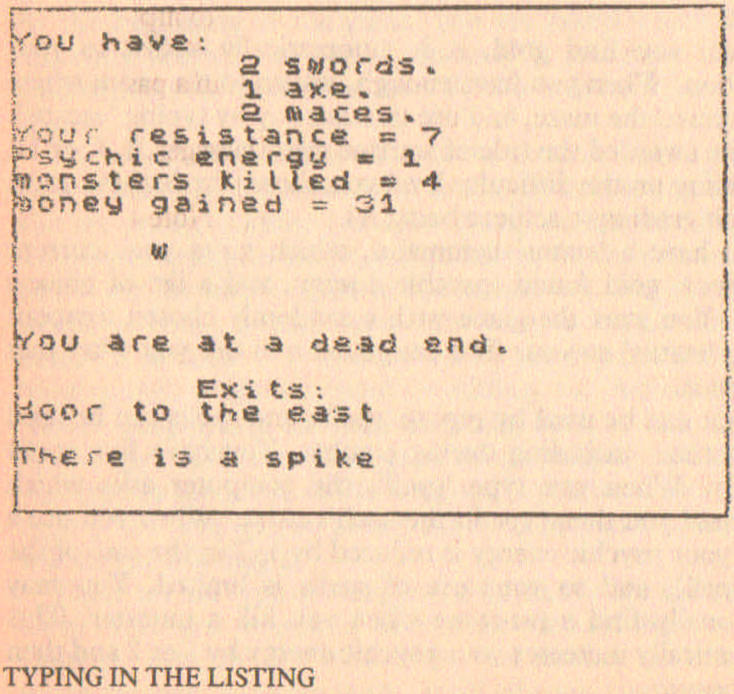
Magic can be used by typing 'spell', and spells can be used at any time, including during combat. You have five spells initially. When you type 'spell', the computer asks which spell, and you then type in the spell's name. When you use a spell, your psychic energy is reduced by 1 (2 in the case of the fear spell), and so your use of spells is limited. You may occasionally find a psi-stone when you kill a monster. This automatically increases your psychic energy by 1 or 2 and then disappears.

The strength spell puts your resistance up to its maximum value of 10 without you resting. This is important if you're getting very battered in a fight.

The teleport spell puts you back in the room where you entered the maze. This is useful for escaping from monsters, and for getting out quickly once you have enough gold to win.

The summonsword spell magically conjures a sword, and you may well need this if your last weapon has just broken.

There are two purely fighting spells: 'disarm' makes the monster lose its defensive powers – helpful versus armadillos – and 'fear' makes a monster run away, leaving behind any treasure it had. The fear spell is a very powerful one, which is why it costs 2 psychic energy points to cast, but it will not work against a sabretooth.



```
You have:
              2 swords.
              1 axe.
              2 maces.
Your resistance = 7
Psychic energy = 1
monsters killed = 4
money gained = 31

      w


You are at a dead end.

         Exits:
Door to the east

There is a spike
```

TYPING IN THE LISTING

As explained, this type of complex adventure starts with a long listing. It may be a good plan to type it in over two or more sessions.

EXPLANATION OF LISTING

This first listing consists of five main parts.
Lines **1000–1520** are descriptions of your surroundings.
2000–2920 are the input and execute commands routines.
3000–3990 include the fighting system.
4000–4630 are the magic routines.
4800–9990 are the routines setting up, beginning and ending the game.

Taking a more detailed look at the listing:

1000–1020 w$ contains a miniature 'map' of your surroundings, and e1 and e2 are the numbers of doors and open passageways around you. x3 and y3 are a record of where you are before moving, and mp is set to mp = 1 when there is a monster present.

1100–1160 Describes your position if you are in a passage.

1200–1220 You are in a room.

1400–1440 Exits, and if any are doors or passages. Note that doors in a passage usually lead to rooms.

1500–1520 Print list of any objects in room. Note that there can be no more than five objects in a room.

1600–1630 Check for monster in room, and also wandering monsters, which do not usually carry treasure.

2000–2390 Input command and go to correct routine.

2400–2450 You move in a direction. If you succeed, then the routine jumps to line **1000** to print new location description, etc.

2500–2570 You take something. tc = total number of objects carried.

2600–2680 You put something down. There are five places in y$, the object array, where the object can be put. This routine finds an empty one and puts the object there. If there is no space (i.e., there are five objects in the room), one object is eliminated so that the new one can be put down.

2700–2750 Status command.

2900–2920 You try to escape.

3000–3070 Find monster name and characteristics and announce its presence.

3500–3570 You select a weapon. You can also type 'fist' to punch a monster, or 'spell' to use magic. If you mistype, or choose a weapon you do not have, you miss your chance.

3580–3590 You attack.

3600–3660 You hit. h$(12) is the monster's resistance. da is the damage your blow causes. Note that there is a chance of your weapon breaking, and that if the animal has a tough shell, swords, axes and daggers can be blunted.

3700–3790 The monster has a 60 per cent chance of getting you, and its maximum attack strength is h$(13).

3900–3990 The monster dies, treasure is checked for, and there is a 20 per cent chance that you will find a psi-stone if the monster has treasure.

4000–4120 You cast a spell, and if you have enough psychic energy, it takes effect.

4200 Strength spell.

4250 Teleport spell.

4300 Non-existent spell – used in next expansion of program.

4350 Summonsword spell.

4400–4430 Disarm spell. h$(15) is the monster's defence value, and is set to a minimum by this spell.

4450–4490 Fear spell. mt is the 'monster number', and if the monster is a sabretooth, nothing happens. When further monsters are added in a later expansion, only two will be immune to this spell.

4600–4630 You find a psi-stone – en is your psychic energy.

4800–4840 You die and a new game in present maze is offered. If accepted, the maze structure will not change, and nearly all the monsters and objects will be as they were. Some of your objects may be left in the location where the monster got you, and in any case fresh monsters and treasure will be added all over.

6300–6430 Check if you have succeeded in your aim, and print a congratulatory message if you have. Again, another go in the same maze is offered. For a new maze, press 'n' in reply to the question 'Another game?' and rerun the program.

8000–8510 Set up for a particular game in a maze. This is used each time you get killed or escape, and is fairly quick in execution as a new maze does not need to be formed.

9000–9040 Select your initial position, near one of the four corners of the maze.

9100–9180 Set up maze. There are two alternative 'patterns' for each 5 × 5 location block of the maze, and these can be rotated, reflected and otherwise manipulated to give many different variations. The array is twice the dimension of the actual maze, as every second space in the array is used to denote a door, open passageway or wall. These are not locations as you cannot enter them.

9300–9310 y$ is the object array, and is filled with objects at random.

9400–9450 x$ contains the names of objects and their characteristics when used for fighting. Objects added in later additions to the system will not have these characteristics, as they are not used in fighting.

9500 List of monsters and their characteristics. When read into h$, h$(12) is the monster's resistance, h$(13) is its attack strength, h$(14) refers to how much treasure it will have (if not wandering) and h$(15) is its defence value. h$(11) is the number of spaces after the name, and is used when printing.

9600–9660 Monster array.

9800–9830 Starting variables. sp is the number of spells, tt the time you have spent in a room. x2, y2 are your starting coordinates, rop is set if you have a rope, and is not used until the expansions are added. t$ holds the number of each object that you are carrying.

Warrior Mage 1: *Monsters and Spells*
LISTING

```
   1 REM Warrior Mage 1
   2 REM  © S.Robert Speel 1984

  10 GO TO 9000
```

```
1000 LET w$=z$(x,y)+z$(x-1,y)+z$
(x,y+1)+z$(x+1,y)+z$(x,y-1)
1010 LET e1=0: LET e2=0: FOR f=2
 TO 5: LET e1=e1+(w$(f)="3"): LE
T e2=e2+(w$(f)="2" OR w$(f)="B")
: NEXT f
1020 LET x3=x: LET y3=y: LET mp=
0

1090 GO TO 1100+100*(w$(1)="0")+
110*(w$(1)="4")

1100 BORDER 4: PRINT ''"You are
";: IF e1+e2=1 THEN PRINT "at a
dead end.": GO TO 1400
1110 IF e2<>2 THEN PRINT ("in a
short passage       between rooms
." AND NOT e2);("at the end of a
 passage." AND e2=1);("at a T-ju
nction." AND e2=3);("at a crossr
oads." AND e2=4): GO TO 1400

1150 IF (w$(2)="2" OR w$(4)="2")
 AND (w$(3)="2" OR w$(5)="2") TH
EN PRINT "at a corner in the"'"p
assage.": GO TO 1400
1160 PRINT "in a ";("north-south
" AND w$(2)="2");("east-west" AN
D w$(3)="2");" passage.": GO TO
1400

1200 BORDER 5: PRINT ''"You are
in a room."
1220 IF x=x2 AND y=y2 THEN PRINT
 "A stairway leads up to the
  Outside."

1400 PRINT 'TAB 8;"Exits:"'
1410 DATA "north","east","south"
,"west"
1420 RESTORE 1410: FOR f=2 TO 5:
 READ b$: IF w$(f)="2" OR w$(f)=
"3" OR w$(f)="B" THEN PRINT ("do
or" AND w$(f)="3");("passage" AN
D (w$(f)="2" OR w$(f)="B"));" to
 the ";b$
1440 NEXT f

1500 LET b$="": FOR f=1 TO 5: IF
 y$(x/2,y/2,f)>"0" THEN LET b$=b
$+y$(x/2,y/2,f)
1510 NEXT f: IF b$="" THEN GO TO
 1600
```

```
1520 PRINT '"There is a ";: FOR
f=1 TO LEN b$: PRINT (CHR$ 8+CHR
$ 8+"some " AND b$(f)=">");x$(CO
DE b$(f)-48, TO VAL x$(CODE b$(f
)-48,7));'TAB 7;("& a " AND f<LEN
 b$);: NEXT f

1600 IF g$(x/2,y/2)>" " AND RND>
.3 THEN GO TO 3000
1610 IF RND>.08 THEN GO TO 2000
1620 IF y>21 AND RND<.7 THEN GO
TO 2000
1630 RESTORE 9500: LET a=INT (RN
D*6)+1: FOR f=1 TO a: READ b$: N
EXT f: LET h$=b$: LET mt=a: LET
a=(RND<.1): LET h$(14)=(b$(14) A
ND a)+("0" AND NOT a): PRINT "Yo
u meet a ";h$( TO 10);"...": GO
TO 3060

2000 INPUT "What do you do? "; L
INE a$: BEEP .1,-10: BEEP .2,-20
: PRINT 'TAB 6; BRIGHT 1; PAPER
7;a$''
2010 LET a$=a$+"          "

2100 RESTORE 1410: FOR f=1 TO 4:
 READ b$: IF a$( TO 2)=b$(1)+" "
 OR a$(4 TO 7)=b$( TO 4) THEN GO
 TO 2400
2110 NEXT f

2120 IF a$( TO 4)="take" THEN GO
 TO 2500
2130 IF a$( TO 4)="stat" THEN GO
 SUB 2700: GO TO 2000
2140 IF a$( TO 4)="spel" THEN GO
 SUB 4000: GO TO 2000
2150 IF a$( TO 4)="look" THEN GO
 TO 1000
2160 IF a$( TO 3)="put" THEN GO
TO 2600
2180 IF a$( TO 4)="rest" THEN GO
 SUB 4700: GO TO 1610
2190 IF a$( TO 6)="escape" THEN
GO SUB 2900: GO TO 2000

2390 PRINT "You cannot do that."
: GO TO 2000

2400 LET f=f+2: LET x1=INT (SIN
(PI/2*f)): LET y1=INT (COS (PI/2
*f)): LET b$=z$(x+x1,y+y1)
```

```
2410 LET e$=a$(1+3*(a$(1)="g")):
 IF b$<>"2" AND b$<>"3" THEN GO
TO 2440
2430 LET x=x+2*x1: LET y=y+2*y1:
 LET rs=rs+(rs<10)*.1: GO TO 100
0
2450 PRINT INK 2;"You cannot go
that way.": GO TO 2000

2500 LET a$=a$(5 TO ): IF a$(1)=
" " THEN LET a$=a$(2 TO )
2510 FOR f=1 TO ob: IF a$( TO 3)
=x$(f, TO 3) THEN GO TO 2530
2520 NEXT f: PRINT INK 2;"You ca
n't take that.": GO TO 2000
2530 FOR g=1 TO 5: IF y$(x/2,y/2
,g)=CHR$ (f+48) THEN GO TO 2550
2540 NEXT g: PRINT INK 2;"It is
not here": GO TO 2000
2550 IF tc>=10 THEN PRINT "You c
annot carry any more.": GO TO 20
00
2560 PRINT INK 2,"Okay": LET t$(
f)=CHR$ (CODE t$(f)+1): LET y$(x
/2,y/2,g)=" "
2570 LET tc=tc+1: GO TO 2000

2600 LET a$=a$(4 TO ): IF a$(1)=
" " THEN LET a$=a$(2 TO )
2610 FOR f=1 TO ob: IF a$( TO 3)
=x$(f, TO 3) THEN GO TO 2640
2620 NEXT f
2630 PRINT "You can't put down s
omething you don't have!": GO TO
 2000
2640 IF t$(f)=" " THEN GO TO 263
0
2650 PRINT INK 2,"Okay": LET t$(
f)=CHR$ (CODE t$(f)-1)
2660 FOR g=1 TO 5: IF y$(x/2,y/2
,g)>" " THEN NEXT g
2670 LET y$(x/2,y/2,g-(g>5)*5)=C
HR$ (f+48)
2680 LET tc=tc-1: GO TO 2000

2700 PRINT TAB 8; BRIGHT 1;"STAT
US"''"You have:"
2710 LET a=0: FOR f=1 TO ob: IF
t$(f)>" " THEN LET a=a+1: PRINT
TAB 10;CODE t$(f)-32;" ";x$(f, T
O VAL x$(f,7));("s" AND t$(f)>"!
");"."
```

```
2720 NEXT f: IF NOT a THEN PRINT
 TAB 6;"nothing."
2730 PRINT "Your resistance = ";
rs'"Psychic energy = ";en'"monst
ers killed = ";mn'"money gained
= ";cash
2750 RETURN

2900 IF x<>x2 OR y<>y2 THEN PRIN
T "You need to find a way out
   before escaping! ": RETURN
2910 PRINT " You escape to the O
utside..."
2920 GO TO 6300+aim*200

3000 RESTORE 9500: LET mt=CODE g
$(x/2,y/2)-32: FOR f=1 TO mt: RE
AD h$: NEXT f
3050 PRINT ' PAPER 7;"There is a
 ";h$( TO 10-VAL h$(11))
3060 LET mp=1
3070 PRINT "You must fight it."

3500 INPUT "What weapon do you u
se? "; LINE a$: LET a$=a$+"     "
3510 LET a=0: RESTORE 9400: FOR
f=1 TO 6: READ b$: IF a$( TO 3)=
b$( TO 3) THEN LET a=VAL b$(8):
LET c=VAL b$(9): LET d=VAL b$(10
): GO TO 3570
3520 NEXT f: LET a=1: LET c=2: L
ET d=6: IF a$( TO 3)="fis" THEN
PRINT INK 2;"You punch it.": GO
TO 3590
3530 IF a$( TO 4)="spel" THEN GO
 SUB 4000: GO TO 3700-2700*(e=2)
3560 PRINT INK 2;"You pause in i
ndecision.": GO TO 3700

3570 IF t$(f)<=" " THEN PRINT IN
K 2;"You look for a weapon": GO
TO 3700
3580 PRINT INK 2;"You attack wit
h your ";b$( TO VAL b$(7))
3590 IF RND>d/10 THEN PRINT "You
 miss": GO TO 3700

3600 LET da=INT (RND*(c-a+1)+a+3
-VAL h$(15)): IF mt=7 AND b$( TO
 3)="spi" THEN LET da=da+6
```

```
3610 PRINT "You hit ";("weakly"
AND da<2);("quite hard" AND da>1
 AND da<4);("very hard" AND da=4
);("with perfect strength" AND d
a>=5)
3620 LET h$(12)=CHR$ (CODE h$(12
)-da): IF h$(12)<"1" THEN GO TO
3900

3650 IF RND<.1 AND a$( TO 3)<>"f
is" THEN PRINT "Your ";b$( TO VA
L b$(7));" breaks!": LET t$(f)=C
HR$ (CODE t$(f)-1): BEEP .1,0: B
EEP .2,-20: GO TO 3700
3660 IF h$(15)>"4" AND RND<.4 AN
D f<4 THEN PRINT "Your ";b$( TO
VAL b$(7));" is blunted by"'"the
 blow, and you discard it.": LET
 t$(f)=CHR$ (CODE t$(f)-1): BEEP
 .1,2: BEEP .1,-4

3700 PRINT PAPER 7;"The ";h$( TO
 10-VAL h$(11));" tries to bite
you"
3710 LET a=RND: PRINT PAPER 7;("
and misses." AND a<.4);("success
fully" AND a>=.4): IF a<.4 THEN
GO TO 3500
3720 LET rs=rs-INT (RND*(CODE h$
(13)-48)+1): IF rs<.1 THEN GO TO
 4600
3790 GO TO 3500

3900 LET mp=0: PRINT PAPER 7;"Th
e ";h$( TO 10-VAL h$(11));" is d
ead."
3910 LET mn=mn+1: LET g$(x/2,y/2
)=" "
3920 IF h$(14)<"1" THEN PRINT "I
t has no treasure.": GO TO 2000
3930 LET a=INT (RND*5+2+(1+RND*5
)*(h$(14)>"1")+(5+RND*6)*(h$(14)
>"2"))+INT (RND*(CODE h$(14)-48)
)*(h$(14)>"3")
3940 PRINT "You take its hoard o
f ";a;" crowns."
3950 LET cash=cash+a
3970 BEEP .1,0: IF RND<.2+.1*(y>
20) THEN GO SUB 4600
3990 GO TO 2000

4000 INPUT "Which spell do you c
ast"' LINE a$: LET a$=a$+"     "
```

```
4010 DATA "strength",1,"teleport
",1,"weedkill",1,"summonsword",1
,"disarm",1,"fear",2
4050 RESTORE 4010: FOR e=1 TO sp
: READ b$,b: IF a$( TO 3)=b$( TO
 3) THEN GO TO 4100
4060 NEXT e: PRINT INK 2; BRIGHT
 1;"You try to cast a spell, but
    fail.": RETURN

4100 PRINT INK 2; BRIGHT 1;" You
 cast a ";b$;" spell."''
4110 IF en<b THEN PRINT INK 2; B
RIGHT 1;"You do not have enough
psychic  energy for your spell t
o work.": RETURN
4120 LET en=en-b: GO TO 4150+e*5
0

4200 PRINT "You are back to full
 strength.": FOR f=1 TO 10: BEEP
 .05,0: BEEP .05,5: NEXT f: LET
rs=10: RETURN

4250 PRINT "You feel a rush of a
ir, feel    momentarily dizzy, a
nd suddenly you find yourself in
 the room   you first entered."
4260 FOR f=1 TO 20: BEEP .05,f:
NEXT f: FOR f=1 TO 20: BEEP .05,
20-f: NEXT f
4270 LET x=x2: LET y=y2: GO TO 1
000

4300 GO TO 2000

4350 PRINT " There is a high-pit
ched humming sound, and a sword
appears in  your hand."
4360 LET t$(1)=CHR$ (CODE t$(1)+
1)
4370 BEEP 2,50: RETURN

4400 IF NOT mp THEN PRINT "You w
aste your spell.": RETURN
4410 FOR f=1 TO 20: BEEP .05,f:
BEEP .05,0: NEXT f
4420 PRINT "The ";h$( TO 10-VAL
h$(11));" looks weaker."
4430 LET h$(15)="0": RETURN

4450 IF NOT mp THEN GO TO 4400
```

```
4460 FOR f=1 TO 16: BEEP .01,0:
BEEP .05,-40: BEEP f/100,0: NEXT
 f: BEEP .4,0
4470 IF mt=5 OR mt>6 THEN PRINT
"nothing happens.": RETURN
4480 PRINT "The ";h$( TO 10-VAL
h$(11));" panics and flees!"
4490 LET mp=0: LET mn=mn-1: GO T
O 3910

4600 FOR f=1 TO 20: BEEP .02,20:
 BEEP .01,60: NEXT f
4610 PRINT " You find a Psi-Ston
e!": LET en=en+INT (RND*3)+1
4620 PRINT "Your psychic energy
is now ";en
4630 RETURN

4700 PRINT INK 2;TAB 6;"OK"
4710 IF rs<10 THEN LET rs=rs+INT
 (RND*2)+1: IF rs>10 THEN LET rs
=10
4740 RETURN

4800 BORDER 0: PRINT "You are de
ad."''"You found ";cash;" crowns
,"''"& killed ";mn;" monsters."
4810 PRINT ''"Another game in th
is system?"
4820 LET a$=INKEY$: IF a$<="" TH
EN GO TO 4820
4830 IF a$="y" THEN GO TO 8000
4840 BEEP 1,0: INK 0: PAPER 7: B
ORDER 7: CLS : STOP

6300 PRINT "You have ";cash;" cr
owns.": IF cash>=lev*30 THEN GO
TO 6330
6310 PRINT " You wanted ";lev*30
;" crowns."'" You have failed."
6320 PRINT " That was your ";go;
("st" AND go=1);("nd" AND go=2);
("rd" AND go=3);("th" AND go>3);
" try in this"'"system...": GO T
O 4810

6330 PRINT '' BRIGHT 1;" You hav
e won!"''"You can return to your
 home cityas a warrior-mage grad
e ";CHR$ (69-lev)
6340 FOR e=1 TO 10: FOR f=1 TO 1
0: BEEP .05/f,f+e*4: NEXT f: NEX
T e
```

```
6430 GO TO 4810

8000 LET go=go+1: BORDER 6: CLS
8010 LET a=5: FOR f=1 TO ob: IF
t$(f)>" " THEN LET y$(x/2,y/2,a)
=CHR$ (48+f): LET a=a-1: IF NOT
a THEN GO TO 8030
8020 NEXT f
8030 FOR f=1 TO 10: FOR g=1 TO 1
0: IF RND<.1 THEN LET y$(f,g,5)=
CHR$ INT (48+RND*ob): BEEP .1,0:
8040 NEXT g: NEXT f
8050 FOR f=2 TO 20: FOR g=2 TO 2
0: IF z$(f,g)<>"0" THEN GO TO 96
50
8060 IF RND<.5 THEN GO TO 9620
8080 GO TO 9660

8100 CLS : PRINT TAB 8;"Warrior-
Mage"''
8110 INPUT "Level of difficulty?
 (1-4) ";a$: IF a$<"1" OR a$>"4"
 THEN LET a$="4"
8120 LET l$(INT (RND*6)+1)="!"
8130 LET lev=VAL a$: LET a$="123
4567890": LET aim=0

8500 PRINT " Your aim is to find
 ";lev*30;" gold      coins, by d
estroying creatures  and taking
their treasures."
8510 RETURN

9000 BORDER 4: INK 1: PAPER 6: C
LS : RANDOMIZE
9010 LET x2=6+(RND<.5)*10: LET y
2=6+INT (RND<.5)*10
9040 LET ob=6: LET go=1: PRINT A
T 10,10;"Please Wait"

9100 DIM z$(21,21): FOR f=1 TO 2
1: FOR g=1 TO 21 STEP 10: LET z$
(f-(f-21)*(f>21),g)="A": LET z$(
g-10*(g=31),f)="A": NEXT g: NEX
T f
9110 LET b$="BKJKKBKJKLKTLKTKKVK
ZK↑": FOR f=1 TO 16 STEP 2: LET
z$(CODE b$(f)-64,CODE b$(f+1)-64
)="2": NEXT f
9120 DATA "031303031","3AAA3AAA2
","1AAA1AAA1","2AAA3AAA2","12130
3121","2AAA3AAA2","0A1A1A031","3
A3A2A3AA","030312130"
```

```
9130 DATA "03121A130","3AAA2A2A3
","1AAA1A1A1","2AAA3A3A2","03130
3031","2AAA3A3A2","1A0A12021","3
A3A3A2A3","031A0A130"

9170 FOR e=0 TO 1: FOR f=0 TO 1:
 RESTORE 9120+10*(RND<.5): LET a
=(RND<.5)*8+1: LET c=(RND<.5): F
OR g=a TO 10-a STEP 1-2*(a=9)
9180 READ b$: LET b=(RND<.5)*8+1
: FOR h=b TO 10-b STEP 1-2*(b=9)
: LET z$((g*c+h*NOT c)+e*10+1,(h
*c+g*NOT c)+f*10+1)=b$(1): LET b
$=b$(2 TO ): NEXT h: NEXT g: NEX
T f: NEXT e

9300 DIM y$(10,10,5): FOR f=1 TO
 10: FOR g=1 TO 10: IF RND<.75 T
HEN LET y$(f,g)=CHR$ INT (49+RND
*ob)
9310 NEXT g: NEXT f

9400 DATA "sword.5238","axe...31
46","dagger6127","mace..4355"
9410 DATA "club..4155","spike.51
44"
9450 RESTORE 9400: DIM x$(ob,10)
: FOR f=1 TO ob: READ x$(f): NEX
T f

9500 DATA "viper.....53812","dir
e wolf.15523","hellcat...36433",
"armadillo.13316","sabretooth087
23","giant rat.14413"

9600 DIM g$(10,10): FOR f=2 TO 2
0: FOR g=2 TO 20: IF z$(f,g)<>"0
" THEN GO TO 9650
9620 IF RND<.7 THEN LET g$(f/2,g
/2)=CHR$ INT (33+RND*6)
9660 NEXT g: NEXT f

9800 LET sp=6: LET tt=0: LET mp=
0: LET x=x2: LET y=y2: LET rs=10
: LET a$="look": LET tc=0
9810: LET en=4: LET cash=0: LET
mn=0: LET rop=0
9820 DIM t$(ob): LET ymp=0
9830 LET g$(x2/2,y2/2)=" ": LET
z$(x2,y2)="0"

9980 GO SUB 8100
9990 BEEP 1,10: GO TO 1000
```

HINTS ON PLAY

Due to the random factor in this system, you may find that you do not meet monsters for quite a while, or you may meet a sabretooth almost immediately. Although things will even out over a long period, you may get some very untypical distributions of monsters and objects. It is important that you make a map of where you go, so that you can retrace your steps and, if killed off, do not have to find your way around all over again.

You start with just one weapon, and you may need to use up most of your psychic energy against the first few monsters you meet, to avoid being killed. This is not as bad as it sounds, for although you will lose out on energy, you should gain a wide range of weapons and so be better at fighting. You may also find a psi-stone, which will replenish your psychic energy. If you are killed off, it is worth going to the site of your demise next time round, as some of your objects from your previous explorings may still be there.

Do not become discouraged if you get killed off rather quickly in the first few games – as you become more familiar with the game, and improve your skills, your success rate will increase.

ADDITION 1
gives **WARRIOR MAGE 2**
High-res Pictures

TO PLAY

This addition adds high-resolution pictures to the program. Each of the monsters is now accompanied by a picture when you first meet it, and when you are successful in your aim and escape, you are given a picture of your home city.

```
You have won!

You can return to your home city
as a warrior-mage grade D
```

```
Another game in this system?
```

TYPING IN THE LISTING

MERGE **Addition 1** onto the program so far.
This listing is awkward to type in due to the code in lines 7800–7850. It is easy to make mistakes in these lines, and for this reason a self-check routine is included at line 7950. When you have typed in this listing, type GOTO 7950. The routine will detect if there is an error in any of the lines, but not exactly where in the line it occurs. Note that if there are two or more errors, it is possible, although unlikely, that they will cancel out. Errors in lines 7800–7850 during play show up by monster pictures which do not join up at the ends.

EXPLANATION OF LISTING

3200 This line translates the code in line 7800–7850 into high-res pictures on screen. See pages 174–9 on 'Adding Graphics to Adventures'.

6350–6420 Draw picture of city at end of successful game. 6350 draws the wall, 6360–6390 draws the towers, using

POINT to stop them overlapping. 6400–6420 draws the windows.

7800–7850 Data for pictures of monsters. They are, in order, the viper, dire wolf, hellcat, armadillo, sabretooth and giant rat. The numbers in each DATA statement are: the ink colour of picture (the background is always black), the number of lines that need to be left to draw the picture on, the magnification, and the x and y coords of the initial plotted point.

7950–7980 Routine to check you have put in the right characters in lines 7800–7850. It does this by adding up the character codes, and checking if their sums are equal to the correct sums in line 7950. Note that two or more errors may cancel out, and this routine will not then detect a fault – in which case, the error will be obvious when the picture is drawn.

Warrior Mage 2: *High-res pictures*
LISTING

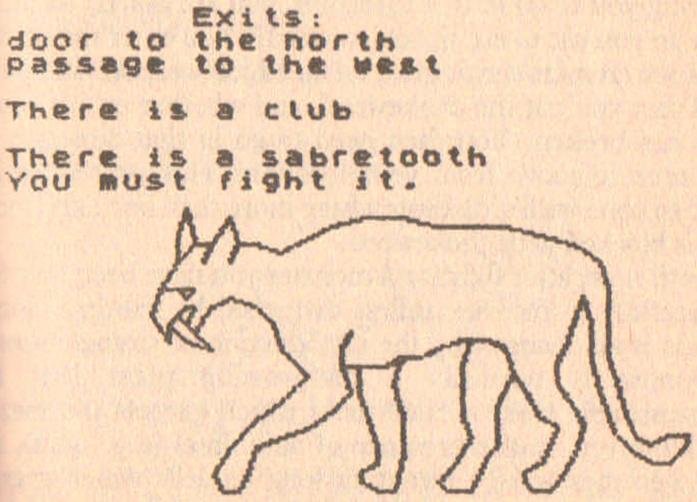
```
   1 REM Warrior Mage 2
   2 REM  © S.Robert Speel 1984

3200 RESTORE 7790+mt*10: READ l,
i,a,b,c,n$: POKE 23692,255: FOR
f=1 TO l: PRINT PAPER 0;TAB 31;"
 ": NEXT f: PLOT b,c: FOR f=1 TO
 LEN n$: LET a1=CODE n$(f): DRAW
 INK i;(INT (a1/10)-8)*a,(a1-(IN
T (a1/10)*10)-4)*a: NEXT f

6350 FOR f=1 TO 15: PRINT : POKE
 23692,255: NEXT f: PLOT 10,10:
DRAW 0,30: FOR f=1 TO 10: DRAW 5
,0: DRAW 0,-4: DRAW 5,0: DRAW 0,
4: NEXT f: DRAW 5,0: DRAW 0,-30:
 DRAW -105,0

6360 LET a$="020,10,20,10,040,20
,10,08,052,05,15,06,065,05,10,30
,103,30,10,12,015,40,03,06,051,2
5,25,12,033,60,12,10,065,10,08,1
4,101,40,10,04,099,20,03,-9"
```

```
6370 FOR f=1 TO LEN a$ STEP 13:
 INK INT (RND*4): LET c=VAL a$(f+
7 TO f+8): LET d=VAL a$(f+10 TO
f+11): PLOT VAL a$(f TO f+2),40:
 DRAW 0,VAL a$(f+4 TO f+5): DRAW
 d/2,c: DRAW d/2,-c: DRAW -d,0:
DRAW d,0
6380 FOR g=39+VAL a$(f+4 TO f+5)
 TO 35 STEP -1: IF POINT (VAL a$
(f TO f+2)+d,g)=0 THEN DRAW 0,-1
: NEXT g
6390 NEXT f

6400 PLOT 60,10: DRAW 0,15: DRAW
 10,0,-2: DRAW 0,-15
6410 DATA 20,30,40,30,80,30,100,
30,37,65,37,80,37,95,108,50,108,
65
6420 RESTORE 6410: FOR f=1 TO 9:
 READ a,b: INK INT (RND*4): PLOT
 a,b: FOR g=1 TO 4: DRAW 2,0: DR
AW 0,-1: DRAW -2,0: DRAW 0,-1: N
EXT g: DRAW 2,0: NEXT f: INK 0

7800 DATA 9,7,3,80,50,"_iVi_RJ_U
h\epgsszrs|p[=47?4.6>Á@ĪS]jR̄?AV̄j
l_sgq_£iqiVA54@AK557ABLD+?5"

7810 DATA 11,4,3,50,55,"thKIVsC_
y5UJsq|}(hqqefP\fgiACD8fPQRfeP,J
L87MLDdG[=G8K_↑aMLM@?MW0[[ZF+KjW
MCCe<==5@K_skakB7,40_i47V"

7820 DATA 11,2,3,100,40,"isqrhsq
↑W[SIhKU↑fSI,=ĪVRHR]S6VWMK][,@Aa
USGG5]S6XCCNal(Jiss}{f\IA86555>=Q
↑_Lf_Lf↑V\]V\↑Lf↑V\_Kg_S"

7830 DATA 9,4,3,50,0,"}tuuVaktss
||qpehNkR\R(sĪQ?>↑UUyq5-f,7£WQH4
Fh?g@57NU}+J9Maki?=G[o,+?Hh?g@59
Wq7A9o=*555"

7840 DATA 13,7,3,50,62,"a|IK@kj>
WaZhH£k[H£hqi"}|t||{{p[PP[Z\gih5
>@ABLMNXMGGHRe\G=I@KUi(ALCMA@hg[
<PI<6KjkLWWaJ5?@?6riGZPF@AUiVMMW
@V£HR[>455GH@Lkutu9MB7?=K£J>fq9K
IRC"

7850 DATA 7,3,2,40,2,"|}}"ti£k£a
£ktisqpefgW£ZRakFPAgJ6J_↑feA7@g7
A@IVPHhIJ@]ABMVRS6A@jMLAg\[3Hgrs
IJ@\A,AAMBfLV£HR\A>?455+6@K"
```

```
7950 LET c=6: DATA 5722,8148,813
2,7150,11676,8484

7960 RESTORE 7950: DIM t(c): FOR
 f=1 TO c: READ t(f): NEXT f
7970 RESTORE 7800: FOR f=1 TO 6:
 READ a,a,a,a,a,a$: LET b=0: FOR
 g=1 TO LEN a$: LET b=b+CODE a$(
g): NEXT g: IF b<>t(f) THEN PRIN
T b,t(f),"Error in line ";f*10+7
790: STOP
7980 NEXT f: PRINT "All code oka
y.": STOP
```

HINTS ON PLAY

There is nothing in this listing which should make you change your tactics – unless it gives you a healthier respect for the sabretooth!



```
          Exits:
door to the north
passage to the west

There is a club

There is a sabretooth
You must fight it.
```

ADDITION 2
gives **WARRIOR MAGE 3**

Carnivorous Plants

TO PLAY

In this addition, a new aim is introduced. Instead of trying to get gold, you are trying to kill monsters. The computer chooses your aim at random from the two possibilities.

A plant, chokeweed, has been introduced into the maze. This plant is a nuisance, as it literally chokes off passages and doors. The stems are thick and rubbery, and cannot be pushed or crushed out of your way – they have to be cut. This means that you must use a sword, axe or dagger to get past the chokeweed. You can always cut it down with one of these weapons, but you may break the weapon while doing so. To cut chokeweed, go in the direction it blocks you and, instead of allowing you to go in that direction, you are asked, 'Which weapon do you use to cut the chokeweed?' You enter the name of a weapon (remember, it must be an edged weapon) and you are told that you cut the chokeweed, and whether or not your weapon has broken. You then need to go in that direction a second time to move from your location. This procedure is adopted to cope with situations where more than one exit from a room is blocked with chokeweed.

Up until now, after fighting a monster you have been able to rest peacefully to recover, unless disturbed by another monster. This is all changed by the introduction of stranglemoss. Stranglemoss is normally a flat-growing plant that is indistinguishable from normal moss which carpets the maze floors. However, in the presence of an animal (e.g., you), it quickly becomes active, sprouting long tendrils which grasp its prey. The victim is then pulled slowly into the centre of the stranglemoss, and is ingested.

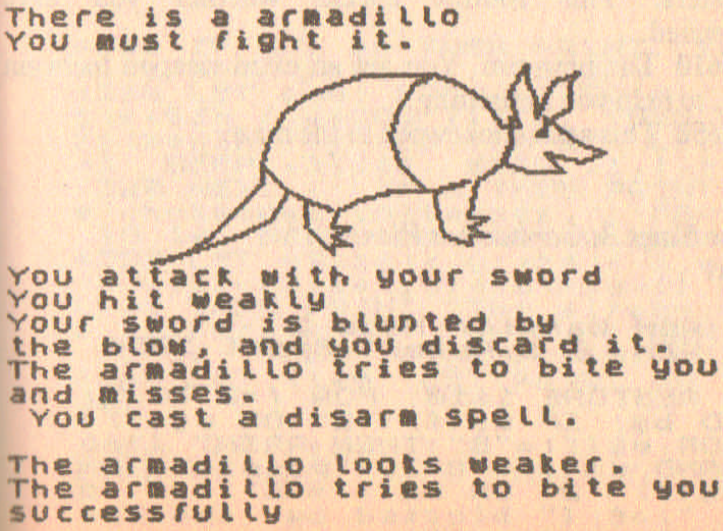
Whenever you rest, there is a risk that the stranglemoss will get you, and the longer you rest in one place, the more likely it is to attack. The active stranglemoss is far too strong to escape from, or cut down, so you need another way to deal with it. There are two ways you can escape.

1. *Magic*. A new spell – weedkill – has been added to your repertoire. Casting a weedkill spell instantly destroys stranglemoss and nearby chokeweed. You may also teleport out of the weed's grasp.

2. *Overfeeding*. The stranglemoss is gluttonous, and once it has started to feed it will ingest anything. If it ingests non-digestible material too fast, it will die. By feeding it objects you may be able to kill off the moss. As an extra bonus, stranglemoss is passionately fond of wood, so feeding it with wooden objects (e.g., spikes, clubs) is more likely to make it die from indigestion! To feed the stranglemoss, use the command 'feed'. You will then be asked what you feed it with, and you enter the name of one of your objects. Stranglemoss takes a while to ingest a whole person, so you have quite a bit of time to feed it. . . .

TYPING IN THE LISTING

MERGE **Addition 2** onto the program so far.



```
There is a armadillo
You must fight it.

You attack with your sword
You hit weakly
Your sword is blunted by
the blow, and you discard it.
The armadillo tries to bite you
and misses.
 You cast a disarm spell.

The armadillo looks weaker.
The armadillo tries to bite you
successfully
```

EXPLANATION OF LISTING

1420–1430 Passages can be blocked by chokeweed.
2440 b$ = "B" when you are trying to cut the chokeweed.
2800–2890 You cut the chokeweed. Only the first three weapon-types can be used. There is a 30 per cent chance that your weapon breaks.
4300–4340 Weedkill spell. The four squares surrounding your location are examined, and if chokeweed is there it is removed. Stranglemoss is also destroyed, and mp, which denotes the presence of a monster or plant, is set to zero.
4720–4730 tt is a measure of how long you have rested, and this determines the chance of stranglemoss growing.
5300–5390 Input routine for stranglemoss. The stranglemoss only decreases your resistance by 1 or 2 each turn, so you have plenty of time to try to kill it.
5800–5860 You feed the moss. In line 5840 objects 5 and 6 are the club and spike and have a greater chance of killing the moss. Object 8 does not yet exist, but will appear in later scenarios as the paddle – another wooden item with which to overcome stranglemoss.
6500–6520 You leave the maze, and your aim was to kill monsters. This routine checks whether you have succeeded.
8600–8610 The new aim. You get an extra weapon to begin with, to help you in fighting.
9640–9650 This adds chokeweed to the maze.

Warrior Mage 3: *Carnivorous Plants*

LISTING

```
   1 REM Warrior Mage 3
   2 REM  © S.Robert Speel 1984

1420 RESTORE 1410: FOR f=2 TO 5:
 READ b$: IF w$(f)="2" OR w$(f)=
"3" OR w$(f)="B" THEN PRINT ("do
or" AND w$(f)="3");("passage" AN
D (w$(f)="2" OR w$(f)="B"));" to
 the ";b$;(" blocked by chokewe
ed" AND w$(f)="B")
```

```
2430 IF w$(f)="6" THEN LET mp=1

2440 IF b$="B" THEN GO TO 2800

2800 PRINT INK 2;"You decide to
cut the chokeweed."
2810 INPUT "Which weapon do you
use? "; LINE a$: LET a$=a$+"###"
2820 RESTORE 9400: FOR f=1 TO 4:
 READ b$: IF a$( TO 3)=b$( TO 3)
 THEN GO TO 2840
2830 NEXT f: PRINT INK 2;"You ch
ange your mind.": GO TO 2000
2840 IF t$(f)<=" " THEN PRINT IN
K 2;"You look for a cutting tool
.": GO TO 2000
2850 IF f>3 THEN PRINT INK 2;"Yo
u try to use an unsuitable    to
ol.": GO TO 2000


2860 PRINT INK 2;"You hack down
the weed with your";b$( TO VAL b
$(7));: IF RND<.3 THEN PRINT INK
 2;", which unluckily breaks";:
LET t$(f)=CHR$ (CODE t$(f)-1)
2870 PRINT "."'"The way is clear
.": LET z$(x+x1,y+y1)="2"
2890 LET w$=z$(x,y)+z$(x-1,y)+z$
(x,y+1)+z$(x+1,y)+z$(x,y-1): GO
TO 2000


4300 IF NOT mp THEN PRINT "You w
aste your spell.": RETURN
4310 LET a=0: FOR f=1 TO 4: IF z
$(x+SIN f,y+COS f)="B" THEN LET
z$(x+SIN f,y+COS f)="2": LET a=a
+1: BEEP .1,f*a: BEEP .1,f-a
4320 NEXT f: IF a THEN PRINT "Th
e chokeweed collapses.": FOR f=1
 TO 10: BEEP .01,RND*40: NEXT f
4330 LET w$=z$(x,y)+z$(x-1,y)+z$
(x,y+1)+z$(x+1,y)+z$(x,y-1): IF
mp=1 THEN PRINT "The stranglemos
s withers and    dies.": FOR f=1
 TO 10: BEEP .01,RND*-40: NEXT f
4340 LET tt=0: LET mp=0: GO TO 2
000

4720 LET tt=tt+1
4730 IF RND<tt/10 THEN GO TO 530
0
```

```
5300 LET mp=1: PRINT "While you
have been here,"'"stranglemoss h
as grown, and now it grabs you..
.."
5310 INPUT "What do you do? "; L
INE a$: LET a$=a$+"      "
5320 IF a$( TO 4)="spel" THEN GO
 SUB 4000
5340 IF a$( TO 4)="stat" THEN GO
 SUB 2700
5350 IF a$( TO 4)="feed" THEN GO
 SUB 5800
5380 PRINT "The stranglemoss tig
htens its   grip...": LET rs=rs-
1-INT (RND*2): IF rs<=0 THEN GO
TO 4800
5390 GO TO 5310

5800 INPUT "What do you feed the
 moss with? "; LINE a$: LET a$=a
$+"     "
5810 FOR f=1 TO ob: IF a$( TO 3)
=x$(f, TO 3) AND t$(f)>" " THEN
GO TO 5830
5820 NEXT f: PRINT BRIGHT 1;"You
 look for something to give  the
 stranglemoss.": RETURN

5830 PRINT BRIGHT 1;"You feed th
e stranglemoss with  a ";x$(f, T
O VAL x$(f,7));"."
5840 LET t$(f)=CHR$ (CODE t$(f)-
1): PRINT " It ingests your offe
ring...."'': IF RND>.3+.2*(f=5 O
R f=6 OR f=8) THEN RETURN

5850 PRINT " The moss has over-i
ngested,     collapses and dies!"
'': LET tt=0: LET mp=0
5860 FOR f=1 TO 20: BEEP .1,-RND
*20-30: NEXT f: GO TO 2000

6500 PRINT "You have killed ";mn
;" monsters."
6510 IF mn<lev*5 THEN PRINT "Tha
t is not enough. You have    fai
led.": GO TO 6320
6520 GO TO 6330

8130 LET lev=VAL a$: LET a$="123
4567890": LET aim=INT (RND*2): G
O TO 8500+100*aim
```

```
8600 PRINT " You must kill ";lev
*5;" monsters."'" You must then 
escape."'" You start with an ext
ra          weapon..."
8610 LET a=RND*6+1: LET t$(a)=CH
R$ (CODE t$(a)+1)
8620 RETURN

9640 GO TO 9660
9650 IF z$(f,g)="2" AND RND<.1 T
HEN LET z$(f,g)="B"
```

HINTS ON PLAY

The addition of chokeweed can deplete your stock of edged weapons or limit your explorations. If you play several games in one system the amount of chokeweed tends to increase, thus compensating for the increase in the number of objects lying around, which in turn accumulates.

Stranglemoss is a much more dangerous problem. It is a good idea to collect spikes and clubs just to deal with stranglemoss. Do not rest for too long at a time, and you may avoid the problem altogether.

The monster-killing aim reduces the use of the fear spell, as scaring off a monster does not count as killing it.

ADDITION 3 gives WARRIOR MAGE 4

Magic Potions

TO PLAY

A third aim of collecting bounty points has been introduced. Each time you kill a monster, you get a certain number of points, depending on how tough that monster was. This means that you should actually look forward to meeting a sabretooth as it represents a lot of bounty points!

A very important element in this addition is the potion. Potions, like weapons, are randomly distributed around the maze, and are also often found in a monster's lair. You use a potion first by taking it, then by using the command 'drink potion'. Potions may be healing, replenishing your strength, or poisonous, damaging (even killing) you if you are weak. Alternatively, a potion may contain scent, which is harmless to drink but invariably lures the nearest monster to investigate. . . . Naturally, there is no way you can tell the effects of a potion before drinking it. Note that you can drink a potion during a fight.

Potions also provide a new method of fighting stranglemoss. Using the command 'pour potion' you can tip the potion over the moss. If the potion is a poison, the moss will die. If it is a scent, nothing will happen – no monster will want to disturb the stranglemoss. Of course, if you pour a healing potion on to stranglemoss, it will get stronger and therefore hurt you more.

TYPING IN THE LISTING

MERGE **Addition 3** on to the program so far.

```
There is a hellcat
You must fight it.
```

```
You attack with your sword
You hit quite hard
The hellcat tries to bite you
and misses.
 You cast a fear spell.

The hellcat panics and flees!
You take its hoard of 14 crowns.
You find a potion.
```

EXPLANATION OF LISTING

2170–3540 The new 'drink potion' command is put in the main input routine, line 2170, and also in the fighting input routine at 3540. aim = 2 is 'bounty point' aim, and pmt is the bounty points for a particular monster, calculated from its attack, defence and resistance values. Note this is done *before* the fight, as its resistance decreases during the fight. ymp is your total bounty points so far.

3960–4520 For monsters with treasure, you may find a potion.

5330–5450 You pour a potion on to stranglemoss. There is a 30 per cent chance of you killing it, and a 30 per cent chance that it will grow faster and attack more.

5500–5660 You drink a potion. Note that if it is a scent potion, no monster is attracted if you are already fighting one when you drink.

6700–6720 Your bounty point score is given.

8130 There are now three possible aims.

8700–8710 The new aim. You need forty times the difficulty level bounty points to win.

9040–9420 The new object (potion) is added.

```
      OK

      rest

      OK
While you have been here,
stranglemoss has grown, and now
it grabs you....
You feed the stranglemoss with
a sword.
 It ingests your offering....

The stranglemoss tightens its
grip...
You pour a potion on the moss.
Nothing happens.
The stranglemoss tightens its
grip...
 You cast a weedkill spell.

The stranglemoss withers and
dies.
```

Warrior Mage 4: *Magic Potions*
LISTING

```
   1 REM Warrior Mage 4
   2 REM  © S.Robert Speel 1984

2170 IF a$( TO 4)="drin" THEN GO
 SUB 5500: GO TO 2000

2740 IF aim=2 THEN PRINT "Bounty
 points = ";ymp

3060 LET mp=1: LET pmt=CODE h$(1
2)+CODE h$(13)+CODE h$(15)-150

3540 IF a$( TO 4)="drin" THEN GO
 SUB 5500: GO TO 3700

3910 LET mn=mn+1: LET ymp=ymp+pm
t: LET g$(x/2,y/2)=" "

3960 IF h$(14)>"2" OR RND<.1 THE
N GO SUB 4500

4500 FOR f=1 TO 10: BEEP .02,20:
 BEEP .01,60: NEXT f
4510 PRINT "You find a potion."
4520 LET f=7: LET tc=tc+1: GO TO
 2660

5330 IF a$( TO 4)="pour" THEN GO
 SUB 5400

5400 IF t$(7)<"!" THEN PRINT "Yo
u have nothing to pour.": RETURN

5410 LET t$(7)=CHR$ (CODE t$(7)-
1): LET tc=tc-1
5420 PRINT INK 2;"You pour a pot
ion on the moss."
5430 IF RND<.4 THEN PRINT "Nothi
ng happens.": RETURN
5440 IF RND<.7 THEN PRINT "The m
oss withers and dies.": GO TO 20
00
5450 PRINT "The moss thrives, gr
ows faster  and is eager for mor
e potion.": LET rs=rs-3: RETURN

5500 IF t$(7)=" " THEN PRINT INK
 2;"You have nothing to drink.":
 RETURN
```

```
5510 LET t$(7)=CHR$ (CODE t$(7)-
1): LET tc=tc-1
5520 PRINT INK 2;"You swallow th
e potion..."
5530 GO TO 5550+50*INT (RND*3)


5550 PRINT "You feel sick...it w
as poison!": LET rs=rs-1-INT (RN
D*5): IF rs<1 THEN PRINT "You co
llapse and die.": GO TO 4800
5560 RETURN

5570 PRINT "It appears to have n
o effect.": RETURN


5600 IF mp THEN GO TO 5570
5610 RESTORE 9500: LET a=INT (RN
D*6)+1: FOR f=1 TO a: READ b$: N
EXT f: LET h$=b$: LET a=(RND<.1)
: LET h$(14)=(b$(14) AND a)+("0"
 AND NOT a)
5620 PRINT "There is a pungent s
mell, and a passing ";h$( TO 10-
VAL h$(11));" comes to"'"investi
gate."
5630 GO TO 3060

5650 PRINT "You instantly feel r
ested and  strong.": LET rs=10+
(rs=10)
5660 RETURN


6700 PRINT "You have gained ";ym
p;" bounty"'"points."
6710 IF ymp<40*lev THEN PRINT "Y
ou needed ";lev*40;" points,"'"s
o you have failed.": GO TO 6320
6720 GO TO 6330

8130 LET lev=VAL a$: LET a$="123
4567890": LET aim=INT (RND*3): G
O TO 8500+100*aim
8700 PRINT " You are a bounty hu
nter. You  have been commission
ed to do a  monster cull, and fo
r each one  you kill, you are aw
arded bountypoints. You want to
get ";lev*40;" of"'"these points
."
8710 RETURN
```

```
9040 LET ob=7: LET go=1: PRINT A
T 10,10;"Please Wait"

9420 DATA "potion6"
```

HINTS ON PLAY

The use of potions decreases the threat from stranglemoss. Potions also help when you are losing a fight and have no magic, as they give you a chance to recover from your wounds. Do not feel compelled to drink potions as soon as you find them, but keep them in reserve.

With three different possible aims, you should develop slightly different tactics for each. For instance, potions are more beneficial if you are aiming to kill monsters, as they may attract monsters for you to attack. But if you are looking for gold, remember that wandering monsters do not have treasure.

ADDITION 4 gives **WARRIOR MAGE 5**

Traps and Pits

TO PLAY

This addition adds pits, traps and a special type of room called the pool room. A room may now contain a pit, which drops you into a hole and damages you, or kills you if you are weak. It may also have a trap from which rocks drop upon your head.

Pits and traps may work every time you enter a room, or only 50 per cent of the time. There is no way you can avoid these perils, but there is a new object, the helmet, which protects you from the falling-rock traps. When you take a helmet, you are assumed to be wearing it, but this will break if hit by rocks, so you will need a new one for the next rockfall.

The pool room consists of a room with a small lake in the centre which blocks all passages through it. If you go into a pool room and wish to exit by a way other than how you came in, type in your instruction as usual. You are then asked how you get across the pool. You can swim, or go along a ledge against the wall. Alternatively, you can go over a bridge across the pool, or use one of the rafts there. To use a raft, you need a new object – the paddle.

There are various things that can happen: you may lose objects, money, or be attacked by carnivorous fish.

TYPING IN THE LISTING

MERGE **Addition 4** on to the program so far.

EXPLANATION OF LISTING

1210 Check for pool room.
1300–1330 If you have just walked into a room, check for pits and traps.
5000 The pit. This may do considerable damage when you fall in: 1–5 resistance points.
5010 Falling-rock trap. A 50 per cent chance they hit you, and you lose 0–5 resistance unless you have a helmet.
6000–6270 The pool room. **6000–6010** You enter the pool room, description printed. **6020–6070** If you decide to go across the room, determine which way you try to cross and go to appropriate subroutine.
6080–6090 You cross the bridge. A 20 per cent chance you fall, or more if you are carrying lots of objects.
6120–6140 You go along the edge of the pool. A 25 per cent chance you fall and lose gold, unless you have daggers to drive into the wall.
6160–6190 You try a raft. You need a paddle and only have a 10 per cent chance of sinking.
6200–6220 You swim. A 25 per cent chance fishes attack, and this routine is used if you fall in the water while using other methods of crossing the pool.

6240–6270 Miscellaneous routines. Line 6260 updates the number of objects carried after you lose things in the water.

9040–9630 Two extra objects, the helmet and the paddle, are included. There is an 8 per cent chance of a pool room occurring.

Warrior Mage 5: *Traps and Pits*

LISTING

```
   1 REM Warrior Mage 5
   2 REM  © S.Robert Speel 1984

1210 IF z$(x,y)="4" THEN GO SUB
6000

1300 IF g$(x/2,y/2)>=" " OR a$(
TO 4)="look" THEN GO TO 1400
1310 LET d$="000000": LET a=CODE
 g$(x/2,y/2): FOR f=1 TO 6: IF a
>=2↑(6-f) THEN LET a=a-2↑(6-f):
LET d$(f)="1"
1320 NEXT f
1330 IF d$(4)="0" AND (d$(6)="0"
 OR (d$(6)="1" AND RND<.5)) THEN
 GO SUB 5000

2420 LET tt=0: IF z$(x,y)="4" TH
EN GO TO 6020

5000 IF d$( TO 2)="00" THEN PRIN
T "A pit opens in the floor, and
   you fall in.": LET rs=rs-INT
 (RND*5)-1: PRINT ("The fall kill
s you" AND rs<=0);("You are brui
sed, but able to    climb out."
AND rs>0)

5010 IF d$( TO 2)="01" THEN PRIN
T "Rocks fall from a hole in the
   ceiling.": LET a=INT (RND*6)*
 (RND<.5): PRINT "They ";("miss"
 AND NOT a);("hit" AND a);" you."
;: LET b=(t$(9)>" " AND a): PRIN
T (CHR$ 8+", but you are       p
rotected by your helmet, which b
reaks." AND b): LET a=a*(NOT b):
 LET rs=rs-a: LET t$(9)=CHR$ (CO
DE t$(9)-b): PRINT ("You are hur
t." AND rs>0 AND a)
```

```
5020 IF d$( TO 2)="10" AND tt TH
EN GO TO 5400
5030 IF rs<=0 THEN GO TO 4800
5040 RETURN

6000 LET f$=e$: PRINT "You are i
n a cavern with a wide pool occu
pying the floorspace.  There is 
a narrow ledge along  the walls
, a shaky bridge, and  several d
riftwood rafts at each exit."
5010 POKE 23692,255: RETURN

6020 FOR f=1 TO 2: RESTORE 1410:
 FOR g=1 TO f: READ c$: NEXT g:
LET c$=c$(1): IF e$=c$ OR f$=c$
THEN READ c$: READ c$: LET c$=c$
(1): IF e$=c$ OR f$=c$ THEN PRIN
T "You return the way you came."
: GO TO 2430
6030 NEXT f: PRINT "How do you c
ross the cavern?"
5040 PRINT "You may:"'" 1)use th
e bridge"'" 2)use the ledge"'" 3
) try a raft"'" 4)swim"'"or 5)tu
rn back.'

6050 LET a$=INKEY$: IF a$<"1" OR
 a$>"5" THEN GO TO 6050
6060 LET a=VAL a$: LET a$="     "
: FOR f=21 TO 15 STEP -1: PRINT
AT f,0;TAB 31;" ": NEXT f
6070 GO TO 6040+40*a

6080 PRINT "You cross the bridge
.": IF RND<.5 AND tc>4 OR RND<.2
 THEN PRINT "A rotten plank give
s way, and  you fall into the w
ater.": LET rs=rs-INT (RND*2): F
OR f=1 TO ob: LET t$(f)=CHR$ (CO
DE t$(f)-INT (RND*2)*(t$(f)>" ")
): NEXT f: LET a=.5: GO TO 6210
6090 PRINT "You get across the p
ool and exitthe room.": GO TO 24
30

6120 PRINT "You inch your way al
ong the"'"ledge.": IF t$(3)>"!"
THEN PRINT "You drive daggers in
to the wall to help you keep ste
ady.": LET t$(3)=" ": GO TO 6250
5130 IF RND<.25 THEN PRINT "You
slip and fall, tearing your pouc
```

```
h and losing all your gold  coin
s.": LET cash=0: LET a=.3: GO TO
 6210
6140 GO TO 6090

6160 PRINT "You try a raft.": IF
 t$(8)=" " THEN PRINT "You have
nothing to paddle with.": GO TO
6030
6170 PRINT "You start to paddle.
": IF RND<.1 THEN PRINT "The raf
t sinks, and you fall in the wat
er.": LET t$(8)=CHR$ (CODE t$(8)
-1): LET a=.2: GO TO 6210
6180 IF RND<.5 THEN PRINT "As yo
u reach the other side, youlose
your paddle.": LET t$(8)=CHR$ (C
ODE t$(8)-1)
6190 GO TO 6250

6200 LET a=.25: PRINT "You swim
across.": IF t$(4 TO 5)>"  " THE
N PRINT "Clubs and maces are to
bulky to swim with, so you drop
them.": LET t$(4 TO 5)="  "
6210 IF RND<a THEN PRINT "The sp
lash you make attracts    biting
 fishes, and these attack.": LET
 rs=rs-INT (RND*4)
6220 GO TO 6250

6240 PRINT "You go back.": GO TO
 2430
6250 IF rs<=0 THEN PRINT "You pe
rish from your wounds.": GO TO 4
800
6260 LET tc=0: FOR f=1 TO ob: LE
T tc=tc+CODE t$(f)-32: NEXT f
6270 GO TO 6090

9040 LET ob=9: LET go=1: PRINT A
T 10,10;"Please Wait"
9420 DATA "potion6","paddle6","h
elmet6"
9610 IF RND<.08 THEN LET z$(f,g)
="4"
9630 IF RND<.25 THEN LET b$="0"+
STR$ INT (RND*2)+"0"+STR$ INT (R
ND*2)+"0"+STR$ INT (RND*2): LET
g$(f/2,g/2)=CHR$ (VAL ("BIN "+b$
))
```

HINTS ON PLAY

You are unlikely to die from the inclusions in this addition, unless you are already in bad shape. However, once you know where the pits and traps are, it is a good idea to try to find diversions around them.

If you need to cross a pool room, consider your aim and status. You can lose your gold by going via the ledge, so avoid this if your aim is to get gold. Similarly, if you want to keep your clubs and maces, avoid swimming.

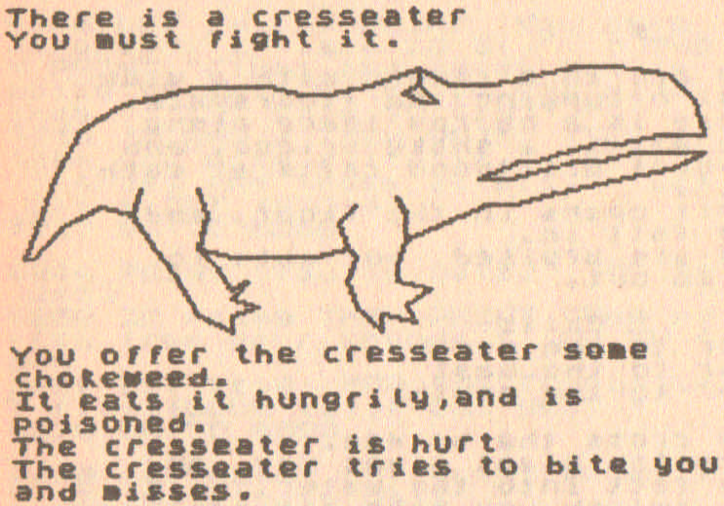
```
door to the south
passage to the west

There is a club

      s

You are in a cavern with a wide
pool occupying the floorspace.
There is a narrow ledge along
the walls, a shaky bridge, and
several driftwood rafts at each
exit.
A pit opens in the floor, and
you fall in.
You are bruised, but able to
climb out.

         Exits:
door to the north
door to the east
door to the south

You cross the bridge.
A rotten plank gives way, and
you fall into the water.
The splash you make attracts
biting fishes, and these attack.
You get across the pool and exit
the room.

You are at a dead end.

         Exits:
door to the west
```

ADDITION 5
gives **WARRIOR MAGE 6**
(The Complete System)

The Lower Reaches

TO PLAY

You may have experienced great difficulty in finding enough monsters to achieve your level 4 difficulty aim. With this addition, a new lower level to the maze – the *Lower Reaches* – is introduced. Here the monsters are much tougher but the treasures much greater.



```
There is a cresseater
You must fight it.
```

```
You offer the cresseater some
chokeweed.
It eats it hungrily,and is
poisoned.
The cresseater is hurt
The cresseater tries to bite you
and misses.
```

To get to the Lower Reaches, you will have to find a hole leading down. You also need a rope. The command 'tie rope' will attach the rope to the top of the hole, and you can go down using the command 'descend'.

There are 5 new types of monster in the Lower Reaches. The vampire has a lot of treasure, but is very tough to kill, and strong in attack. You will need to protect yourself with a cross, and can attack best using a spike.

The cresseater is a huge beast with a passion for green leaves. Feeding it with poisonous strangleweed is the best way of damaging it. To do this, type 'weed' as your weapon during a fight. When you cut strangleweed, you are given the option of taking some for this purpose. However, strangleweed is common only in the Upper Reaches, so you will need to collect some before going down. Cresseaters have no treasure.

Fanged bats attack viciously from above, but if you have a helmet they cannot fight well.

Tarantulas need to be killed with your first blow, or they can destroy you. Tarantulas have venom, and you can take this to fight with. During a fight, entering 'venom' as your weapon will throw it at a monster.

The worst creature with the most treasure is the dragon. You can protect yourself to some extent with a shield, but the dragon will still be very difficult to kill.

TYPING IN THE LISTING

Add **Addition 5** to **Warrior Mage 5** to get the complete, fully expanded **Warrior Mage 6**. Finally, delete the checking routine, lines 7950–7980, once you are sure everything is working.

EXPLANATION OF LISTING

1230 If there is a hole down to the Lower Reaches, describe it.

2200–2210 The two new commands to get you down the hole.

2880 You can take some weed just after cutting down a patch of strangleweed.

3100–3120 Special effects of various objects on some Lower Reaches monsters.

3550–3880 Venom and weed. You have a 70 per cent chance of hitting with venom, which affects all creatures except tarantulas and vampires. Offering weed to any other monster than a cresseater is useless, but the cresseater loses 1–6 resistance if it eats.

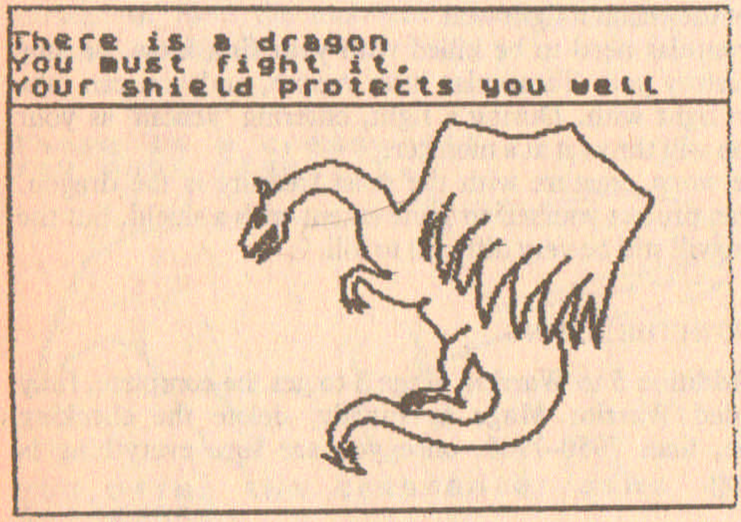
3910 If you kill a tarantula, you can take some venom.

5100–5220 You try to go down to the Lower Reaches.

7860–7890 DATA for new monster pictures.

7950 The self-test routine is enlarged. Type GOTO 7950 to carry out test after new DATA lines have been entered.

8030–9300 There are more objects now. The maze is dimensioned bigger to include the Lower Reaches.



```
There is a dragon
You must fight it.
Your shield protects you well
```

Warrior Mage 6: *The Lower Reaches*

LISTING

```
   1 REM Warrior Mage 6
   2 REM  © S.Robert Speel 1984

1230 IF x=x4 AND y=y4 THEN PRINT
 "There is a hole leading down t
o the Lower Level.": IF rop THEN
 PRINT "A rope hangs down the ho
le."

2200 IF a$( TO 3)="tie" THEN GO
SUB 5100: GO TO 2000
```

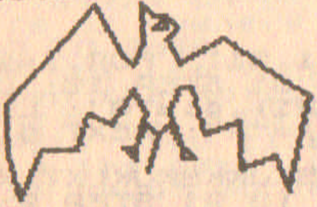
```
2210 IF a$( TO 4)="desc" THEN GO
 TO 5200

2880 PRINT "You can take some we
ed.": LET y$(x/2,y/2,5)="="

3100 IF mt=7 AND t$(11)>" " THEN
 PRINT "It is weakened by your c
ross.": LET h$(12)="5": LET h$(1
5)="3"
3110 IF mt=8 AND t$(12)>" " THEN
 PRINT "Your shield protects you
 well": LET h$(13)="2"
3120 IF mt=11 AND t$(9)>" " THEN
 PRINT " It cannot attack well,
due to  your helmet.": LET h$(13
)="1"

3550 IF a$( TO 4)="veno" OR a$(
TO 4)="weed" THEN GO TO 3800+50*
(a$(1)="w")

3800 IF t$(14)<=" " THEN GO TO 3
560
3810 LET t$(14)=CHR$ (CODE t$(14
)-1): PRINT "You throw some spid
ervenom": FOR f=1 TO 20: BEEP f/
200,40-f: NEXT f: IF RND<.3 THEN
 PRINT "You miss.": GO TO 3700
3820 PRINT "You hit!": LET da=IN
T (RND*8): IF mt=7 OR mt=10 THEN
 LET da=0: PRINT "The venom has
no effect": GO TO 3700
3830 LET h$(12)=CHR$ (CODE h$(12
)-da): IF h$(12)<"1" THEN GO TO
3900
3840 PRINT "The ";h$( TO 10-VAL
h$(11));" is hurt": GO TO 3700

3850 IF t$(13)<" " THEN GO TO 35
60
3860 LET t$(13)=CHR$ (CODE t$(13
)-1): PRINT "You offer the ";h$(
 TO 10-VAL h$(11));" some"'"chok
eweed.": IF mt<>9 THEN PRINT "It
 is not interested.": GO TO 3700
3870 PRINT "It eats it hungrily,
and is     poisoned."
3880 LET da=1+INT (RND*6): GO TO
 3830
```

```
3910 LET mn=mn+1: LET ymp=ymp+pm
t: LET g$(x/2,y/2)=" ": IF mt=10
 THEN PRINT "You can take some v
enom.": LET y$(x/2,y/2,5)=">"

5100 IF t$(10)<=" " THEN PRINT "
You have no rope.": RETURN
5110 IF x<>x4 OR y<>y4 THEN PRIN
T "You tie your rope in a knot.
   Then you untie it.": RETURN
5120 PRINT "You tie your rope at
 the top of the hole.": LET rop=
1: LET t$(10)=CHR$ (CODE t$(10)-
1): RETURN

5200 IF x<>x4 OR y<>y4 THEN PRIN
T "There is nowhere to descend t
o!": GO TO 2000
5210 IF NOT rop THEN PRINT "You
jump down the hole."'" You break
 your neck...": GO TO 4800
5220 PRINT "You climb down the r
ope into theLower Level.": LET x
=x5: LET y=y5: GO TO 1000

7860 DATA 13,4,4,150,80,"A,K_|IK
IK_r@aA@3=QRZf[HrR?|V£5sh@qLV@hq
s£LH7LMV>HUIUJ99zoI_v:qd>"

7870 DATA 17,5,2,150,95,"HG[Wajk
=GQ@[aaj©]GF{lkt4F@RZb£kj©}F@@[b
ajju3HG0Z((jj>FGZba£RQZWabaakkkU
8BB8BC+5,7--=GGG>>D88-+?4>VA[RJQ
_UJUH<\↑U↑U↑U↑U↑U3>juR2tjVa©rpfP
[f]↑S7A4P\VfLgBV_fpqhi?\fe©)F[I6
5Ht>tH_szLU_a£[g[G?57-68754=Re7P
{BWHLks{z{{|r}tjkbXN"

7880 DATA 13,7,3,5,30,"mmw▄▀▀▜rrr
{r▜ssRf@AiVz▚▛▛▛▙▇oZ6@!6+5@ |}}r}
r▚*!!!+6  WL\QGR{4\-R9MaKMj>[+66
,-bM[Ffei>q5R.;:CaG-*(J"

7890 DATA 10,2,3,100,15,"Xs{r6He
\£WIJK_↑]_↑]IJK@\wL6r}f<vD4_gs\G
RmD4?£▄fQ@aW:4?Uv▚▄I\G77=HN763:C4
>Gk"eeI/55GRa£▚▄zR9*4QRc}io\7A4Z5
"

7900 DATA 9,3,4,50,5,"NXuujd]£W]
↑I↑U]I\"ypFFM5X5NJH{]JULIGaKHSK↑
aL=AG6<"
```

```
7950 LET c=11: DATA 5722,6146,61
32,7150,11676,6484,5369,18595,80
64,8652,4116

8030 FOR f=1 TO 10: FOR g=1 TO 1
5: IF RND<.1 THEN LET y$(f,g,5)=
CHR$ INT (48+RND*(ob-(g(11)-2)):
 BEEP .1,0

9020 LET x4=22-x2: IF RND<.5 THE
N LET x4=x2
9030 LET y4=22-y2: IF RND<.5 AND
 x2<>x4 THEN LET y4=y2
9040 LET ob=14: LET go=1: PRINT
AT 10,10;"Please Wait"
9050 LET x5=6+(RND<.5)*10: LET y
5=26

9100 DIM z$(21,31): FOR f=1 TO 3
1: FOR g=1 TO 31 STEP 10: LET z$
(f-(f-21)*(f>21),g)="A": LET z$(
g-10*(g=31),f)="A": NEXT g: NEX
T f
9110 LET b$="BKJKKBKJKLKTLKTKKVK
ZK↑": FOR f=1 TO 21 STEP 2: LET
z$(CODE b$(f)-64,CODE b$(f+1)-64
)="2": NEXT f

9170 FOR e=0 TO 1: FOR f=0 TO 2:
 RESTORE 9120+10*(RND<.5): LET a
=(RND<.5)*8+1: LET c=(RND<.5): F
OR g=a TO 10-a STEP 1-2*(a=9)

9300 DIM y$(10,15,5): FOR f=1 TO
 10: FOR g=1 TO 15: IF RND<.75 T
HEN LET y$(f,g)=CHR$ INT (49+RND
*(ob-2-(g<11)))

9420 DATA "potion6","paddle6","h
elmet6","rope..4","cross.5","shi
eld6","weed..4","venom.5"

9510 DATA "vampire...3=8X5","dra
gon....4A>█3","cresseater0A501",
"tarantula.12A00","fanged bat049
02"

9600 DIM g$(10,15): FOR f=2 TO 2
0: FOR g=2 TO 20: IF z$(f,g)<>"0
" THEN GO TO 9650

9700 FOR f=2 TO 20: FOR g=21 TO
30: IF z$(f,g)>"1" THEN GO TO 97
30
```

```
9710 IF z$(f,g)="0" THEN IF RND<
.6 THEN LET g$(f/2,g/2)=CHR$ (37
+INT (RND*5)): GO TO 9730
9720 IF z$(f,g)="1" THEN IF RND<
.4 THEN LET g$(f/2,g/2)=CHR$ (42
+INT (RND*2))
9730 NEXT g: NEXT f
```

```
There is a fanged bat
You must fight it.
 It cannot attack well, due to
your helmet.
```



```
You attack with your axe
You hit quite hard
The fanged bat tries to bite you
and misses.
You attack with your sword
You hit quite hard
The fanged bat is dead.
 It has no treasure.
```

HINTS ON PLAY

The Lower Reaches literally add a whole new dimension to the system. You can now use the Upper Reaches merely to find the various objects needed to conduct more daring exploits in the lower level passages.

The objects you will want to have are:

rope, spike, helmet, weed, cross, shield, a couple of good weapons and, ideally, some potions.

As this system has expanded so much, the various details on play are rather dispersed. Here is a full list of commands:

```
go  north
    south
    east
    west
take (object)
put (object)
look
rest
status
tie rope
descend
escape
spell   strength
        summonsword
        teleport
        disarm
        weedkill
        fear (does not work versus sabretooth or Lower
          Reaches monsters)
feed stranglemoss
drink potion
pour potion
```

4. Mapping for Programmers and Players

Apart from **Tribe**, all the systems in this book contain a map. When going around a maze, island or forest, for example, it is useful to draw a map of where you have been.

If the locations are the same shape, the map can easily be drawn out on a grid, and this is the best technique to use for systems such as **Warrior Mage**. If the locations vary in size and shape, as in the **Preset Systems,** then mapping cannot be done in this way. In this case, just draw locations as you come to them, and be aware if two different routes lead to the same place.

When there is more than one level to the game, start a map for each level on a different piece of paper – marking clearly which locations lead from one level to another. Each location drawing should be large enough to contain several words – details about the type of location, monsters and any special objects. Exit directions are, of course, very important, and you should also note which directions are blocked off.

In **Warrior Mage**, one of the first things you should do after an unsuccessful mission is to go to the place of your previous character's demise, and you will probably find at least some of his objects there. In **Fangmole Tunnels**, it is not practical to draw a map as you go, but do keep a mental note of which direction the map extends in.

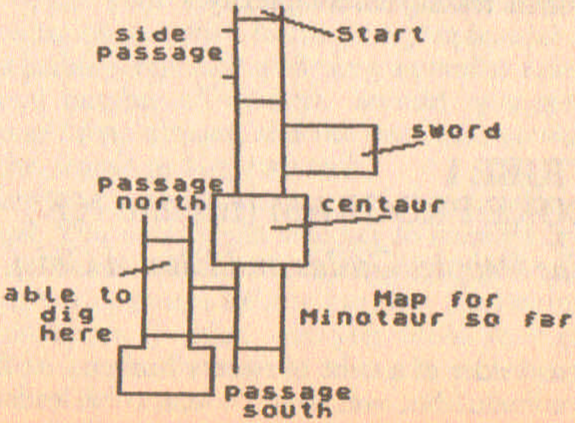
For programming purposes, there are several ways of storing maps. The **Preset** adventures have a different set of line numbers for each location, and a variable (ru) holds the line number where your location begins. A string holds the numbers of the locations in each direction from a location, or zero if a direction is blocked. This method uses very little memory for variables and arrays, but is inflexible.

For random-maze adventures, the map is stored in one or more two-dimensional arrays. Each array-coordinate holds one location, and the map is made up of small blocks – each block containing, say, a room and a passage, or a T-junction. These blocks are selected randomly, turned in a random direction or reflected, and placed in the array, giving many different configurations.

Another method is to make a room consist of many locations, as is done in **Fangmole Tunnels**. In both these methods, there is no need to have a list of which locations are connected to which – the computer can tell if you can go that way just by examining the next location in the array. The advantage of this is the tremendous flexibility of arrays – locations can be easily altered and randomized, but at the cost of using up huge amounts of RAM, and often a long setting-up time, before the game can be played.

Anarchic System is midway between the Preset and Random systems. The map is pre-defined, but the details of locations are held in arrays so that they can be easily altered, e.g., holes being dug. The use of strings to hold the orientations of the locations in relationship to one another means that very irregular maps can be formed.

Here is a typical example of the way in which you might build up your map.



5. Tribe System

Tribe is different from the other adventures in this book. You are the elected leader of a band of hunters and their families – the Tribe. Your aim is to survive in power for as many years as you can, or until a maximum of twenty-five years has passed at which point you retire.

The most important features of an adventure are all present: attributes which are altered (foodstocks, treasury, private savings, people); interactions with the voters and their guilds; a range of commands; 'treasure' (good harvests); and a score at the end. However, rather than fighting monsters you fight elections, instead of taking objects you help yourself from the treasury, and instead of a status command there is an opinion poll.

The game is in four parts, consisting of a core program which fits into 16K and three additions, each adding new features and increasing the complexity.

TRIBE 1
CORE PROGRAM (fits into 16K)

The People, Guilds and Elected Chief

TO PLAY

You start as leader of a tribe of twenty hunters. At first they are fairly content, but you were chosen to be leader on the condition that you would give them a better life, and unless you do this you will soon be removed from office. Each year you make several important decisions, and for each type of decision you are shown a different screen of information. The first screen shows how many there are of each type of worker (initially only twenty hunters), and any retirements and new recruits to the workforce. (Note that the workforce numbers are given as they were *before* retirements.)

Your first decision concerns the occupation of the new recruits – hunters, gatherers, fishers or farmers. Unfortunately, your choice is restricted due to the workers' guilds. There is a guild for each type of worker, and their aim is to increase the numbers of their particular workers. Initially, there is only the hunters' guild, and a confrontation between it and you is inevitable. If you do not increase the membership of each guild – or worse still, you reduce it – they will try to force an election.

The next screen shows the situation after the harvest. This depends considerably on the weather: hunters do fairly well in warm weather; gatherers prefer dry weather but always collect something; fishers do well in wet weather, and farmers achieve huge crops in warm weather, but miserable crops in the cold. Out of the total crop, you must feed the people, sell some produce abroad for money, and maybe store some for poor-harvest years. You start with fifty sacks of food stored from the year before your leadership commenced. First, you are asked how much you wish to feed the people, and the minimum acceptable is one sack each, i.e., a number at least as big as the total number of workers. Second, you are given a chance to sell food abroad, and the price offered varies from year to year, from 1 to 4 ingots a sack.

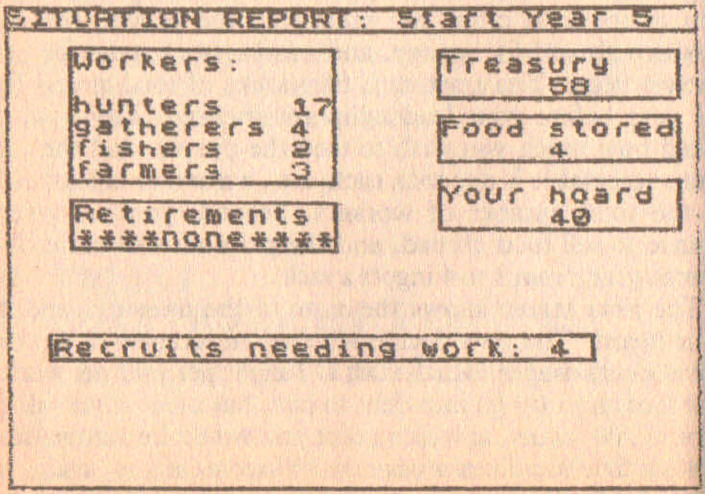
The next screen shows the state of the treasury, and also your hoard. You now decide whether to increase or decrease the workers' wages, which start at 1 ingot per year per worker. The treasury may go into debt to pay, but must never fall to a debt of 200 ingots, at which point you would be automatically sacked. You are given a wage of 20 ingots (unless, again, this would lead to the treasury dropping to a debt of 200 ingots).

Finally, at the end of the year, the living standards (food sacks per worker and wages per worker) are announced, and the degree of contentment among the people and the guilds. The people like to have constantly increasing living standards, and the guilds want constantly increasing memberships.

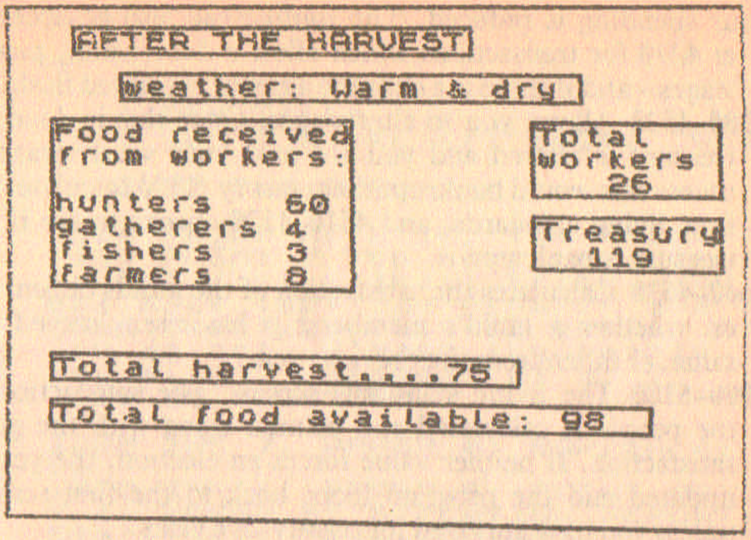
If the people riot, due to an insufficient rise in their living standards, or having less than the minimum of one food sack per year per worker, an election is called. The guilds can also force an election if their support for the government falls below 50 per cent. If no election is called, you proceed with the next year.

When an election is called, you have to pay 50 ingots from your hoard (not the treasury) to stand for re-election. The way the people vote depends only partly on their guilds' recommendations, so the guilds may force elections even if you get 100 per cent of the vote! People care mainly about rising living standards (i.e., 'are we better off than last year?' and 'are living standards rising at a fast or slow rate?'). For you to lose an election, there must be a majority against you, so a 50/50 split means that you are re-elected.

Once you are defeated, thrown out of office, too poor to



```
SITUATION REPORT: Start Year 5

Workers:                    Treasury
                               58
hunters        17
gatherers      4            Food stored
fishers        2               4
farmers        3
                            Your hoard
Retirements                    40
****none****

Recruits needing work: 4
```



```
AFTER THE HARVEST
  weather: Warm & dry
Food received            Total
from workers:            workers
                           26
hunters      60
gatherers    4           Treasury
fishers      3             119
farmers      8

Total harvest....75
Total food available: 98
```

fight for re-election, or you have survived for twentyfive years, your final score is given, which depends on how long you survived, the final population, and how much money you received.

EXPLANATION OF LISTING

100–110 Introduction.

1000–1340 First screen. This shows types of workers, new recruits, retirements, food stored and the money available. It allows you to input the number of recruits which turn to each profession. Note that if when inputting the recruits' jobs the recruits are not all placed, the routine repeats until all the recruits do have work.

3000–3550 Deals with the harvest, works out how many sacks of food are gathered, and gets you to input how much food is fed to the workers and how much sold.

4200–4230 Deals with reduction of stored food, preventing the amount of food dropping to less than zero.

4250–4280 This routine is accessed at 4250 when using money. If you try to spend more than the treasury holds, your spending is reduced. The routine can also be accessed at 4270 for transactions which allow a debt – e.g., paying wages – and if there is a debt the amount is printed flashing.

4300–4520 Allows you to alter wages. Once this is done the treasury is altered and you are paid your yearly wages – unless this would bankrupt the treasury. 4500 calculates the final living standards, and 4510–4520 checks to see if the treasury is bankrupt.

4600–4630 Calculates the satisfaction of the guilds depending on whether a guild's membership has risen, stayed the same, or fallen from the previous year.

5000–6180 The living standards screen. The satisfaction of the people is calculated and printed along with the guild satisfaction. If neither value forces an election, the year is updated and the program loops back to the first screen, unless you have survived for twenty-five years.

7000–7020 Announces election and decides if you can afford to stand for re-election.

7030–7300 Election fought. Each trade is taken separately, since the guild satisfaction is a factor in deciding how likely a worker is to vote for you. For each worker in a trade the calculated chance of voting for you is changed by up to 10 per cent. Finally, the worker's actual vote is decided by checking if RND is smaller than his chance of voting for you. This is repeated for all workers in all trades.

7320–7360 Tells you how long you lasted in power, how much money you gained and your final achievement score.

7500–7520 At the end of each screen this subroutine is called, which causes a small sound and instructs you to press ENTER to continue.

8240–8280 You survive twenty-five years. The final score is calculated in the same way as when you lose.

9000–9940 Starting variables. Ye = year, st = food stored, cash = treasury, rs = your hoard, ls and ls1 are the total living standards of the current and the previous years. Arrays a and b show how many people are in each trade this year and last, and array s gives the actual living standard in terms of food and wage standards.

Tribe 1: *Core Program*
LISTING

```
   1 REM Tribe
   2 REM © S. Robert Speel 1984

  10 GO SUB 9000

 100 PRINT AT 2,10;"The Tribe"''
 110 PRINT " You have been elect
ed Chief of a small tribe. You m
ust make thetribe prosper and be
come wealthywithin 25 years. If
you do not  please the people, t
hey will    dispose of you."
 120 GO SUB 7500

1000 INK 7: PAPER 0: BORDER 4: C
LS
1010 INK 1: PAPER 5: PRINT BRIGH
T 1; PAPER 6;"SITUATION REPORT:
Start Year ";ye;TAB 32
1020 LET ls1=ls
1030 FOR f=1 TO 4: LET b(f)=a(f)
: NEXT f
1050 PRINT AT 2,3;"Workers:";TAB
 15;AT 3,3;TAB 15
1060 DATA "hunters","gatherers",
"fishers","farmers"
1070 RESTORE 1060: FOR f=1 TO 4:
 READ b$: PRINT PAPER 0;TAB 3; P
APER 5;b$;TAB 13;a(f);TAB 15: NE
XT f
1080 PRINT AT 2,20; PAPER 5;"Tre
asury  ";AT 3,20;TAB 31;AT 3,25
;cash

1100 PRINT AT 9,3; PAPER 5;"Reti
rements "': IF ye=1 THEN LET c=0
: GO TO 1120
1110 LET c=0: RESTORE 1060: FOR
f=1 TO 4: READ b$: LET a=RND: IF
 a<a(f)/30 THEN LET b=INT (RND*a
(f)/(4-3*(a(f)<5)))+1: LET b=b-(
b>4)*(b-4)-(b>a(f))*(a(f)-b): PR
INT PAPER 0;TAB 3; PAPER 5;b;" "
;b$( TO LEN b$-(b=1));TAB 15: LE
T a(f)=a(f)-b: LET c=1
1120 NEXT f

1130 IF NOT c THEN PRINT PAPER 5
;AT 10,3;"****none****"
```

```
1140 PRINT AT 5,20; PAPER 5;"Foo
d stored";AT 6,20;TAB 31;AT 6,25
;st
1150 PRINT AT 8,20; PAPER 5;"You
r hoard ";AT 9,20;TAB 31;AT 9,25
;rs

1200 LET tp=0: FOR f=1 TO 4: LET
 tp=tp+a(f): NEXT f
1210 LET np=INT (RND*tp/9)+2
1220 PRINT AT 15,2;"Recruits nee
ding work: ";np;" "
1230 LET tp=tp+np

1300 RESTORE 1060: FOR f=1 TO 4:
 READ b$
1310 INPUT ("new ";b$;"? ");a: L
ET a=INT ABS a: IF a>np THEN BEE
P 1,0: BEEP 1,0: BEEP 1,0: GO TO
 1310
1320 LET np=np-a: LET a(f)=a(f)+
a: PRINT AT f+3,13;a(f);TAB 15;A
T 15,25;np: IF np>0 THEN NEXT f
1330 IF np THEN GO TO 1300
1340 PRINT AT 15,2; FLASH 1;"ALL
 RECRUITS PLACED"; FLASH 0; PAPE
R 0;TAB 30
1350 GO SUB 7500

3000 PAPER 2: CLS : INK 1: PAPER
 6
3010 PRINT AT 1,3;"AFTER THE HAR
VEST"
3020 PRINT AT 3,5;"weather: ";
3030 LET we=INT (RND*4)+1

3050 DATA "Warm & dry","Warm & w
et","cold & dry","cold & wet"
3060 RESTORE 3050: FOR f=1 TO we
: READ b$: NEXT f: PRINT b$;" "

3100 PRINT AT 5,2;"Food received
";AT 6,2;"from workers:" ;AT 7,2
;TAB 15

3200 DATA "2514030203","46242423
14","0236022500","3647130201"
3210 LET ha=0: FOR f=1 TO 4: IF
a(f)=0 THEN GO TO 3250
3220 RESTORE 1060: FOR g=1 TO f:
 READ b$: NEXT g: PRINT PAPER 2;
TAB 2; PAPER 6;b$;TAB 12;
```

```
3230 RESTORE 3200: FOR g=1 TO f:
 READ b$: NEXT g: LET a=0: FOR g
=1 TO a(f): LET a=a+INT (RND*(VA
L b$(we*2)-VAL b$(we*2-1)))+VAL
b$(we*2-1): NEXT g
3240 PRINT a;TAB 15;TAB 15: LET
ha=ha+a
3250 NEXT f

3300 PRINT AT 15,2;"Total harves
t....";ha;" "
3310 LET st=st+ha
3320 PRINT ' PAPER 2;TAB 2; PAPE
R 6;"Total food available: ";st;
" "''
3330 PRINT AT 5,23;"Total   ";AT
 6,23;"workers  ";AT 7,23;TAB 26;
tp;TAB 31
3340 PRINT AT 9,23;"Treasury";AT
 10,23;"  ";cash;TAB 31

3500 INPUT "How much food for wo
rkers? ";a: GO SUB 4200
3510 LET s(1)=INT (a/tp)
3520 LET sp=INT (RND*4)+1: INPUT
 "Selling price for food  is"',(
sp);" ingots/sack"'"How many sac
ks food sold? ";a
3530 GO SUB 4200
3540 LET cash=cash+a*sp
3550 PRINT AT 10,25;cash;TAB 31

4110 GO TO 4300

4200 IF a>st THEN LET a=st
4210 LET st=st-a
4220 PRINT AT 17,24;st;TAB 28
4230 RETURN

4250 IF a>cash THEN LET a=cash
4260 LET cash=cash-a
4270 PRINT AT 4,4; FLASH (cash<0
);cash; FLASH 0;TAB 10;AT 4,23;r
s;TAB 30
4280 RETURN

4300 GO SUB 7500: INK 7: PAPER 5
: CLS : PAPER 1: PRINT AT 1,12;"
WAGES"
4310 PRINT AT 3,2;"Treasury";AT
4,2;"  ";cash;TAB 10;AT 3,20;"Yo
ur hoard";AT 4,20;"  ";rs;TAB 3
0
```

```
4320 LET wi=0
4330 PRINT AT 6,2;"Wages current
ly cost ";s(3)*tp;TAB 28
4340 PRINT AT 9,2;"A wage increa
se would make";AT 10,2;"this ";t
p*(s(3)+1);TAB 28

4350 PRINT AT 20,0; INK 0; PAPER
 4;"Do you 1)increase, 2)decreas
e or3)not change the wages?";TA
B 32: LET a$=INKEY$: IF a$<"1" O
R a$>"3" THEN GO TO 4350
4360 PRINT AT 20,0; PAPER 5;TAB
31;" ";TAB 31;" "
4370 LET a=VAL a$: PRINT AT 12,4
;"Wages ";("increased" AND a=1);
("decreased" AND a=2);("unchange
d" AND a=3)
4380 LET s(3)=s(3)+(a=1)-(a=2)
4390 LET cash=cash-s(3)*tp

4400 GO SUB 4270
4410 LET a=20-(cash<-179)*(-179-
cash): IF a<1 THEN LET a=0
4420 LET cash=cash-a
4430 PRINT AT 14,2;"You take you
r tithe.": LET rs=rs+a: GO SUB 4
270

4500 LET ls=0: FOR f=1 TO 3: LET
 ls=ls+s(f): NEXT f
4510 IF cash>-200 THEN GO TO 460
0
4520 PRINT AT 16,2;"The treasury
 owes too much";AT 17,2;"money t
o continue, and you";AT 18,2;"ar
e thrown out of office!": GO TO
7300

4600 LET us=100: FOR f=1 TO 4: I
F NOT a(f) THEN GO TO 4620
4610 LET us=us-INT (a(f)/tp*100*
((b(f)>a(f))*.75+(b(f)=a(f))*.2)
)
4620 NEXT f: LET sb=0
4630 IF us>55 THEN GO TO 5000

5000 GO SUB 7500
5010 INK 2: PAPER 1: CLS : PAPER
 7
5020 PRINT AT 1,2;"LIVING STANDA
RDS"
5030 DATA "food","wages"
```

```
5040 LET a=1: FOR f=1 TO 3 STEP
2: IF s(f)>a THEN LET a=s(f)
5050 NEXT f
5060 RESTORE 5030: FOR f=1 TO 3
STEP 2: READ b$: PRINT AT 3+f*2,
5;b$;TAB 13;s(f): NEXT f

6000 LET sat=(ls-ls1)*2+(s+s(3)-
(s(1)=0)*50-wi+(ls-ye)/2
6010 LET sat=sat+(sat<-1)*(-1-sa
t)-(sat>3)*(sat-3)-sb
6020 IF us<0 THEN LET us=0

6100 PRINT AT 12,2;" Guild suppo
rt = ";us;"%"
6110 DATA "rioting","annoyed","r
estless","satisfied","happy"
6120 RESTORE 6110: FOR f=-1 TO s
at: READ b$: NEXT f
6130 PRINT AT 14,2;"The people a
re ";b$
6140 IF sat<0 OR us<50 THEN GO T
O 7000
6150 LET ye=ye+1
6160 GO SUB 7500
6170 IF ye>25 THEN GO TO 8000
6180 GO TO 1000

7000 PRINT AT 16,2;"The ";("guil
d" AND us<50);("people" AND sat<
0 AND us>=50);" force you to hol
d";TAB 30;AT 17,2;"an election!"
;TAB 30
7010 IF rs<50 THEN PRINT AT 18,2
;"You do not have enough";TAB 30
;AT 19,2;"ingots to fight an ele
ction.": GO TO 7300
7020 PRINT AT 18,2;"You have to
spend 50 ingots ";AT 19,2;"to st
and."
7030 LET rs=rs-50: GO SUB 7500
7040 LET vty=0: LET vtn=0

7050 INK 1: PAPER 4: CLS : PAPER
 7: PRINT INK 2;AT 1,5;"▞▞▞ ELEC
TION ▚▚▚"
7060 PRINT AT 4,2;"PEOPLE    FOR
YOU  AGAINST YOU"

7100 RESTORE 1060: FOR f=1 TO 4:
 READ b$: IF a(f)=0 THEN GO TO 7
190
7110 PRINT AT f*2+4,2;b$;TAB 13
```

```
7120 LET vt=50-40*(s(1)=0)-25*(s
(3)=0)+5*s(1)+8*s(3)+20*(ls>ls1)
-25*(ls<ls1)-5*(ls=ls1)-40*(a(f)
<b(f))-10*(a(f)=b(f))+20*(a(f)>b
(f))
7130 LET vt=vt+(ls-ye*4/5)*4
7140 LET vo=0: FOR g=1 TO a(f)
7150 LET vu=vt+INT (RND*10)-INT
(RND*10)-wi*10
7160 LET vo=vo+(RND<vu/100): PRI
NT AT f*2+4,13;vo;TAB 25;g-vo;TA
B 30: NEXT g
7170 LET vty=vty+vo: LET vtn=vtn
+a(f)-vo
7180 PRINT AT f*2+5,2;TAB 30
7190 NEXT f

7200 PRINT AT 15,2;"Total votes
for you = ";vty;" "
7210 PRINT AT 17,2;"Total votes
against you = ";vtn;" "
7220 IF vty>=vtn THEN PRINT AT 1
9,2;"You have been re-elected!."
: GO TO 6150

7300 PRINT AT 20,2;"You have bee
n sacked.";TAB 30
7310 GO SUB 7500
7320 INK 1: PAPER 5: BORDER 2: C
LS
7330 PRINT AT 1,5; PAPER 6;"THE
END OF YOUR REIGN"
7340 PRINT AT 4,2;"You were Rule
r for ";ye;" years,"
7350 PRINT AT 6,2;"during which
time you";TAB 30;AT 7,2;"amassed
 ";rs;" ingots.";TAB 30
7360 PRINT AT 10,10;"Score:";50*
ye+(tp-20)*5+rs*2: STOP

7500 BEEP .5,-40: PRINT AT 21,2;
 INK 2; PAPER 7;"Press "; FLASH
1;"ENTER"; FLASH 0;" to continue
"
7510 IF INKEY$<CHR$ 1 THEN GO TO
 7510
7520 RETURN

8240 INK 4: PAPER 0: BORDER 3: C
LS
8250 PRINT AT 2,6;"25 YEARS OF P
OWER"
8260 PRINT AT 4,2;"You have succ
essfully ruled";AT 5,2;"the Trib
e for 25 years!"
```

```
8270 PRINT AT 7,2;"You are rewar
ded with lands";AT 8,2;"and 500
ingots by the grateful";AT 9,2;"
people."
8280 GO TO 7360

9000 RANDOMIZE
9010 LET ye=1: LET st=50: LET ca
sh=0: LET rs=10: LET ls=0: LET l
s1=0
9020 DIM a(4): DIM b(4): LET a(1
)=20
9030 DIM s(3): LET s(1)=1: LET s
(3)=1

9990 RETURN
```

HINTS ON PLAY

The main problem initially is the dominance of the hunters' guild. As soon as you reduce the hunters they will force an election, therefore it is in your interest to build up counterforces of other workers as quickly as possible. The ideal situation is when no single guild can force an election on its own.

Increasing living standards is not necessary at first, but has to be done at some time. Increasing the food standard (i.e., to 2, 3 or more sacks per worker – no fractions) reduces your income as there is less food to sell. Increasing the wages increases your expenditure. As reducing living standards often forces an election, don't be too nice to the people too quickly as you may regret it later!

ADDITION 1 gives TRIBE 2

Developments

This addition expands the program beyond 16K, giving you **Tribe 2**. To make it easier for you to improve the people's living standards, a new 'housing standard' has been introduced. This has the advantage over other living standards of being a one-off expense. Unfortunately, builders do not work on credit, and so the treasury must hold enough money to pay them to work, or they get angry. A new screen, 'developments', has been added to cope with the builders.

After wages have been paid you now have the option of requisitioning money from the treasury, which is morally quite fair as your wages are now only 10 crowns per year. Again, this only works when the treasury is solvent. The guild leaders now give you warning when the guilds look like forcing an election, and for a suitable bribe they can swing the guilds into a less destructive attitude. Be warned, though; occasionally the guild leaders try to get you to bribe them even if there is no chance of an election being forced!

The other major change is in voting. People now take into account not just the current and the immediate preceding years, but also the year before last. Although this is not as important to people as events nearer the present, it is still a significant factor in their chances of voting for you. This also affects the attitude of the guilds.

TYPING IN THE LISTING

Load **Tribe** and type in **Addition 1**, which adds new lines and replaces a few existing lines. SAVE the result.

EXPLANATION OF LISTING

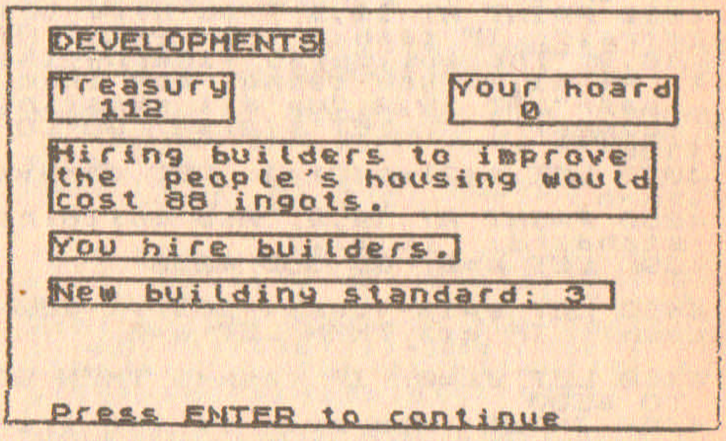
1020 RS2 is the total living standard the year before last.

4000–4090 The new 'developments' screen. This allows you to hire builders if the treasury has sufficient money available. The more the people, the greater the expense of improving their houses, but there is a considerable random factor in the cost.

4450–4470 You take money from the treasury: di is set 50 per cent of the time if you take more than 20 per cent of the treasury, and represents the people learning about your greed. This affects their satisfaction with you as leader.

4640–4680 The guild fathers (leaders) say they will force an election and you can bribe them. Note that line 4630 lets them try for bribes even if fifty-five per cent of the guilds support you. Bribes are taken out of your personal hoard, not the treasury.

7120 The voting system is more complex now, taking into account living standards and guild size both last year and the year before.



```
DEVELOPMENTS

Treasury                 Your hoard
  112                        0

Hiring builders to improve
the people's housing would
cost 88 ingots.

You hire builders.

New building standard: 3


Press ENTER to continue
```

Tribe 2: *Developments*

LISTING

```
   1 REM Tribe 2

1020 LET ls2=ls1: LET ls1=ls
1030 FOR f=1 TO 4: LET c(f)=b(f)
 : LET b(f)=a(f): NEXT f

3580 GO SUB 7500

4000 PAPER 6: INK 1: CLS : PAPER
 7
4010 PRINT AT 1,2;"DEVELOPMENTS"
4020 PRINT AT 3,2;"Treasury";AT
4,2;"   ";cash;TAB 10;AT 3,20;"Yo
ur hoard";AT 4,20;"   ";rs;TAB 3
0
```

```
4030 LET b=(s(2)+2+INT (RND*3))*
tp: PRINT AT 6,2;"Hiring builder
s to improve";AT 7,2;"the peopl
e's housing would";AT 8,2;"cost
";b;" ingots."
4040 PRINT AT 21,0;"Do you hire
builders? (y/n) ": LET a$=INKEY$
: IF a$<>"y" AND a$<>"n" THEN GO
 TO 4040

4050 PRINT AT 21,0; PAPER 6;TAB
31: IF a$="n" THEN GO TO 4100
4060 PRINT AT 10,2;"You hire bui
lders.": IF cash<b THEN PRINT AT
 10,2;"The builders, finding tha
t";AT 11,2;"you cannot pay them,
 smash";AT 12,2;"up all existing
 houses!   ": LET s(2)=0: GO TO
4100
4070 LET s(2)=s(2)+1: LET cash=c
ash-b
4080 PRINT AT 12,2;"New building
 standard: ";s(2);" "
4090 LET a=0: GO SUB 4250

4410 LET a=10-(cash<-189)*(-189-
cash): IF a<1 THEN LET a=0

4440 LET di=0: IF cash<1 THEN GO
 TO 4500
4450 INPUT "How much of the trea
sury do you divert to your own f
unds? ";a: IF a>cash THEN LET a=
cash
4460 IF a THEN LET cash=cash-a:
LET rs=rs+a: IF a>(cash+a)/5 AND
 RND<.5 THEN LET di=2
4470 GO SUB 4270

4610 LET us=us-INT (a(f)/tp*100*
((b(f)>a(f))+(c(f)>b(f))*.75+(b(
f)=a(f))*.2))
4640 PRINT AT 16,2;"The guild fa
thers say that";AT 17,2;"they wi
ll force an election."
4650 PRINT AT 21,0;"Do you try t
o bribe them? (y/n) ": LET a$=IN
KEY$: IF a$<>"y" AND a$<>"n" THE
N GO TO 4650
4660 PRINT AT 21,0;TAB 31: IF a$
<>"y" THEN GO TO 5000
```

```
4670 INPUT "How much? ";a: IF a>
rs THEN LET a=rs
4680 PRINT AT 18,2;"You bribe th
em.": LET rs=rs-a: LET us=us+INT
 (a*10/tp): IF RND<.3 THEN LET s
b=1

5030 DATA "food","housing","wage
s"
5040 LET a=1: FOR f=1 TO 3: IF s
(f)>a THEN LET a=s(f)
5060 RESTORE 5030: FOR f=1 TO 3:
 READ b$: PRINT AT 3+f*2,5;b$;TA
B 13;s(f): NEXT f

6000 LET sat=(ls-ls1)*2+ls-ls2+s
(3)-(s(1)=0)*50-wi-di+(ls-ye)/2

7050 INK 1: PAPER 4: CLS : PAPER
 7: PRINT INK 2;AT 1,5;"▞▞▞ ELEC
TION ▞▞▞"
7060 PRINT AT 4,2;"PEOPLE   FOR
YOU  AGAINST YOU"

7120 LET vt=50-40*(s(1)=0)-5*(s(
2)=0)-25*(s(3)=0)+5*s(1)+5*s(2)+
8*s(3)+20*(ls>ls1)+10*(ls1>ls2)-
25*(ls<ls1)-5*(ls1<ls2)-5*(ls=ls
1)-40*(a(f)<b(f))-30*(b(f)<c(f))
-10*(a(f)=b(f))-5*(b(f)=c(f))+20
*(a(f)>b(f))+10*(b(f)>c(f))

9020 DIM a(4): DIM b(4): DIM c(4
): LET a(1)=20
```

HINTS ON PLAY

The addition of a 'building standard' is very useful, as raising it means a single payment, not a permanent expense. Taking money from the treasury is useful for remaining rich enough to fight an election, but do not take too much as this can ruin the whole economy.

Bribing the guilds often needs a large amount (more than 20 ingots) to be effective, and is worth doing only if you want to avoid an election. Remember that bribes come from your own hoard, so be careful not to put yourself in the position where bribing the guilds means you cannot afford to fight an election.

ADDITION 2
gives TRIBE 3
Opinion Polls and Inflation

Your main new option now is polling. If you want to know how people would vote if an election were called, you can have a poll. This takes place after food has been harvested and allocated, but before developments, so you can still do something to placate the people if they are against you. Polls are paid for out of your personal hoard, and come in three types. The quick poll is the cheapest, and considers about one-third of the voters. The large poll costs twice as much, but polls two-thirds of the population, and the in-depth poll, which costs three times as much as the quick poll, takes all the workers into account to give a fairly accurate idea of your popularity.

A new type of weather, drought, has been introduced, but it occurs only rarely. When it does, the results can be calamitous if you have insufficient stores. A new bonus, however, is increased food prices. This means that, due to increased population of neighbouring countries, the average food price occasionally increases by 1 crown per sack.

The presentation has been improved by altering the display of living standards from single numbers to a row of user defined symbols.

TYPING IN THE LISTING

MERGE with **Tribe 2**. SAVE the result.

EXPLANATION OF LISTING

3040–3050 Drought has been incorporated. Note that on average this occurs only once in ten years.
3520 ic is the provision for inflation, i.e., prices can increase from a 1–4 range to 2–5, 3–6, etc.
3560–3570 A poll option is offered.

4100 Food prices have a 10 per cent chance of increasing from the fifth year onwards.

5060 Display of user defined graphics alters the printing routine slightly.

7600–7930 Polls. Note that prices vary (for the cheapest poll) by up to 11 ingots. You have the option of declining the poll if you think it is too expensive.

9500–9540 User defined graphics data.

```
    25 YEARS OF POWER

You have successfully ruled
the Tribe for 25 years!

You are rewarded with lands
and 500 ingots by the grateful
people.
          Score:

For time in power.......1300

For living standards....346

For population growth...210

For your hoard..........540

    TOTAL POINTS.....2396
```

Tribe 3: *Opinion Polls and Inflation*
LISTING

```
   1 REM Tribe 3
   2 REM © S. Robert Speel 1984

3040 IF RND<.1 THEN LET we=5: BE
EP .5,15: BEEP .5,10
3050 DATA "Warm & dry","Warm & w
et","cold & dry","cold & wet","d
rought"

3520 LET sp=INT (RND*4)+1+ic: IN
PUT "Selling price for food is"
',(sp);" ingots/sack"'"How many
sacks food sold? ";a
```

```
3560 PRINT AT 21,0;"Would you li
ke a poll? (y/n) ": LET a$=INKEY
$: IF a$<>"y" AND a$<>"n" THEN G
O TO 3560
3570 PRINT AT 21,0;TAB 31: IF a$
="y" THEN GO TO 7600

4100 IF ye>4 AND RND<.1 AND ic<4
 THEN BEEP .5,15: BEEP .5,10: PR
INT AT 16,2; FLASH 1;"BONUS";: P
RINT " food prices increase!": L
ET ic=ic+1

4500 LET ls=0: FOR f=1 TO 3: LET
 ls=ls+s(f): NEXT f: LET z(ye)=l
s

5060 RESTORE 5030: FOR f=1 TO 3:
 READ b$: PRINT AT 3+f*2,5;b$;TA
B 13;: FOR g=1 TO s(f): PRINT IN
K 0;CHR$ (143+f);: NEXT g: PRINT
 TAB 14+a;AT f*2+4,5;TAB 14+a: N
EXT f

7600 INK 7: PAPER 3: BORDER 1: C
LS : PAPER 2
7610 PRINT AT 1,7;"ELECTORAL POL
L"
7620 LET a=INT (RND*11)+10
7630 PRINT AT 4,2;"Prices:";AT 6
,2;"1) Quick poll.....";a;AT 7,2
;"2) Large poll.....";a*2;AT 8,2
;"3) In-depth poll..";a*3;" ingo
ts"
7640 PRINT AT 3,20;"Your Hoard";
AT 4,20;"  ";r$;TAB 30

7650 PRINT AT 21,2;"Press 1, 2,
3 or 0 for no poll"
7660 LET a$=INKEY$: IF a$<"0" OR
 a$>"3" THEN GO TO 7660
7670 PRINT AT 21,0; PAPER 3;TAB
31;" ": IF a$="0" THEN GO TO 400
0
7680 FOR f=1 TO 3: PRINT AT 5+f,
2; PAPER 3;TAB 4+(VAL a$<>f)*27:
 NEXT f
7690 IF rs<a*VAL a$ THEN PRINT A
T 18,2; PAPER 0;"You cannot affo
rd this poll!": FOR f=1 TO 5: BE
EP .5,20: NEXT f: GO SUB 7500: G
O TO 4000

7700 LET rs=rs-a*VAL a$
```

```
7710 PRINT AT 4,23;rs;TAB 30
7720 LET vty=0: LET vtn=0

7800 RESTORE 1060: FOR f=1 TO 4:
 READ b$: IF a(f)=0 THEN GO TO 7
890
7810 PRINT AT f*2+8,2;b$;
7820 LET vt=50-40*(s(1)=0)-30*(s
(2)=0)-5*(s(3)=0)+5*s(1)+5*s(2)+
8*s(3)+20*((ls>ls1)+10*((ls1>ls2)-
25*((ls<ls1)-5*((ls1<ls2)-5*((ls=ls
1)-40*(a(f)<b(f))-30*(b(f)<c(f))
-10*(a(f)=b(f))-5*(b(f)=c(f))+20
*(a(f)>b(f))+10*(b(f)>c(f))
7830 LET vo=0: FOR g=1 TO a(f)/(
4-VAL a$)
7840 LET vu=vt+INT (RND*10)-INT
(RND*10)-wi

7850 LET vo=vo+(RND<vu/100): NEX
T g: LET vo=INT (vo*(4-VAL a$))
7860 LET vty=vty+vo
7870 LET vtn=vtn+a(f)-vo
7880 PRINT TAB 13;vo;TAB 25;a(f)
-vo;TAB 30;AT f*2+9,2;TAB 30
7890 NEXT f

7900 PRINT AT 18,2;"Estimated vo
tes for you=";vty
7910 PRINT AT 19,2;"Est. votes a
gainst you=";vtn
7920 GO SUB 7500
7930 GO TO 4000

9010 LET ye=1: LET st=50: LET ca
sh=0: LET rs=0: LET ls=0: LET ls
1=0: LET ic=0
9020 DIM z(30): DIM a(4): DIM b(
4): DIM c(4): LET a(1)=20

9500 DATA 0,60,24,60,126,126,60,
0
9510 DATA 16,40,68,124,68,84,124
,0
9520 DATA 0,0,30,38,74,244,152,2
40
9530 DATA 255,129,129,129,129,12
9,129,255
9540 DATA 0,0,60,60,60,60,0,0

9600 RESTORE 9500: FOR f=144 TO
148: FOR g=0 TO 7: READ a: POKE
USR CHR$ f+g,a: NEXT g: NEXT f
```

HINTS ON PLAY

The introduction of polls makes it possible to 'fine-tune' the economy. The quick poll is suitable for getting a rough idea of how things are going, but neglects minorities. The large poll gives a good idea of how particular worker-types will vote and the in-depth poll is useful when you think an election might give a very close result. The cheaper polls often miscalculate the total number of voters, which is what you should expect if you opt for second best!

Droughts mean you have to exercise a little more prudence in storing food, or risk riots if the crops fail totally. Gatherers, being somewhat drought-resistant, are a useful failsafe.

An increase in food export prices may provide a welcome boost to an ailing economy but cannot be relied upon to occur.

ADDITION 3
gives **TRIBE 4** (The Full System)

Progress Graph

The main new feature is the progress graph. This screen comes up at the end of the first year and every third year thereafter, and gives an indication of your position regarding total living standards. It is also to help you improve your play, as the prominent peaks and troughs in the graph can usually be identified with particular decisions you have taken.

Another addition is an occasional demand by the guilds for wage increases. Although this can be ignored, it makes the people more dissatisfied and prone to vote you out of office. Like droughts, wage demands are nuisances which can sometimes disrupt a tottering economy, but they can be brushed off if the situation is healthy.

A detailed breakdown of your final score is also provided, and this can help you decide on ways of improving your play.

TYPING IN THE LISTING

MERGE with **Tribe 3**. SAVE the result.

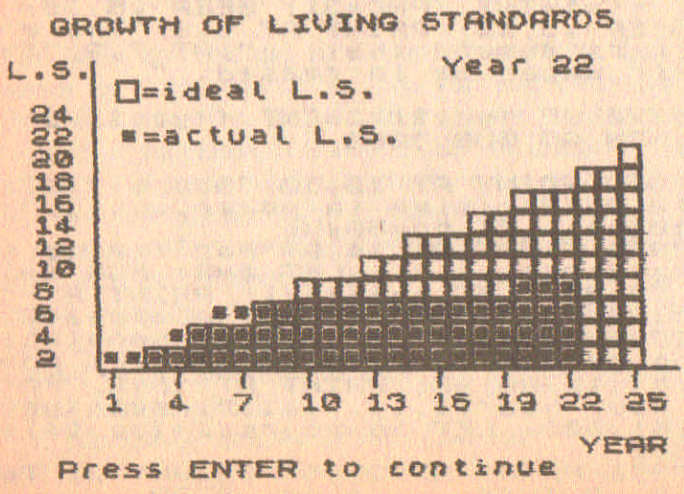
EXPLANATION OF LISTING

4320 A 10 per cent chance of workers demanding wage increases. wi is a measure of how important they think it is that you agree.
6170 Every third year, living standards subroutine is called.
7360–7390 Breakdown of your final score. Note that your hoard is not as important as the time you spend in power and the living standards achieved.
8000–8230 Living standards screen. Shows ideal and actual living standards total for the years you have been in power.

Tribe 4 (The Full System): *Progress Graph*
LISTING

```
   1 REM Tribe 4
   2 REM © S. Robert Speel 1984

4320 LET wi=0: IF RND<.1 THEN LE
T wi=1+INT (RND*3): BEEP .5,15:
BEEP .5,10: PRINT AT 6,2;"The wo
rkers demand that   ";AT 7,2;"th
eir wages be increased. "

6170 IF (ye+1)/3=INT ((ye+1)/3)
THEN GO SUB 8000

7360 PRINT AT 10,10;"Score:";AT
12,2;"For time in power........";
50*ye: LET po=50*ye
7370 PRINT AT 14,2;"For living s
tandards.....";: LET a=0: FOR f=1
 TO 30: LET a=a+z(f): PRINT AT 1
4,26;a*2: NEXT f: LET po=po+a*2
7380 PRINT AT 16,2;"For populati
on growth...";(tp-20)*5: LET po=
po+(tp-20)*5: PRINT AT 18,2;"For
 your heard...........";rs*2+(ye>
24)*500: LET po=po+rs*2+(ye>24)*
500
7390 PRINT AT 20,6; BRIGHT 1;"TO
TAL POINTS.....";po: STOP
```

```
8000 BORDER 5:: PAPER 5: CLS
8010 PRINT AT 1,2; INK 6; PAPER
1;"GROWTH OF LIVING STANDARDS"
8020 PRINT AT 3,20; PAPER 7;"Yea
r ";ye-1

8100 INK 0: PLOT 31,150: DRAW 0,
-111: DRAW 212,0
8110 FOR f=1 TO 12: PRINT AT 17-
f,1;f*2: NEXT f
8120 FOR f=1 TO 25 STEP 3: PRINT
 AT 18,f+3;f;AT 17,f+3; OVER 1;"
|": NEXT f
8130 BRIGHT 1: PRINT AT 4,5; INK
 3; PAPER 7;CHR$ 147;: PRINT "=i
deal L.S.";AT 6,5; INK 2;CHR$ 14
8;: PRINT "=actual L.S."
8140 PRINT OVER 1;AT 3,0;"L.S."
8150 PRINT AT 20,26;"YEAR": BRIG
HT 0

8200 FOR f=1 TO 25: FOR g=1 TO f
*2/5: PRINT AT 17-g,f+3; INK 3;
PAPER 7;CHR$ 147: NEXT g: NEXT f
8210 FOR f=1 TO ye: FOR g=1 TO z
(f)/2: PRINT OVER 1; INK 2;AT 17
-g,f+3; PAPER 8;CHR$ 148: NEXT g
: NEXT f
8220 GO SUB 7500
8230 IF ye<25 THEN GO TO 1000
```



```
    GROWTH OF LIVING STANDARDS

L.S.                Year 22
     □=ideal L.S.
24
22   ■=actual L.S.
20
18
16
14
12
10
8
6
4
2

     1  4  7  10 13 16 19 22 25

                          YEAR
  Press ENTER to continue
```

HINTS ON PLAY

The addition of the living standards screen is a useful aid to improving your play. If you find that what you've achieved in living standards lags behind the ideal level all the time, you will probably be facing riots and the sacking fairly soon. If you are constantly ahead, you may be expanding too fast and could find later that you cannot afford to keep going. You will then be forced to reduce your economy.

Judging the gravity of wage demands is not easy, and if a wage increase is demanded you would do well to check recent polls and the expansion rate of the economy before refusing.

In the complete game, you must be willing to take risks to improve your position and achieve your aim of lasting twenty-five years. There are several distinct stages in a typical game. The first stage is demolishing the superiority of the hunters' guild. Periods of rapid growth are ended by years of non-expansion, often with guild-bribing to forestall elections. At some point there is generally a cash crisis, when you are in danger either of bankruptcy, or of not being able to feed the people due to having sold your stores.

There is a fair amount of luck involved in this game, but it needs a skilled player to take advantage of good fortune and survive bad luck, and complete a twenty-five-year reign.

6. Saving Memory in Adventures

On the 16K Spectrum, the screen uses up about 7K, leaving only 9K for programs. Even with 48K, an adventure may easily expand to fill all available space. Here are a few ways in which memory can be conserved.

1. *Use string arrays rather than numeric arrays*. String arrays consume much less memory than numeric arrays because they store each element as 1 byte, without the need of a mantissa, or numbers outside the range 0–255.
A 20 by 20 numeric array occupies 2034 bytes, whereas a 20 by 20 string array only occupies 435 bytes. It is well worth putting large amounts of information into string arrays, and keeping numeric arrays small.

2. *Avoid long lists of numeric data*. These use a surprisingly large amount of memory. It takes less memory to enclose each piece of data with quotes than to leave them as numbers, and take these strings' VALues. It is even better to take all the strings and put them together in one long string, saving on all the commas and quotes, e.g., instead of

```
DATA 1, 2, 18, 0, 36, 92
```

less space is occupied by

```
DATA "1", "2", "18", "0", "36", "92"
```

and even less by

```
DATA "010218003692"
```

Notice that extra zeros have been added so that a piece of DATA can be used by READing the string, then extracting 2 bytes at a time from it.

3. *Use small strings with* DATA, *rather than multidimensional string arrays*. When a multidimensional string array is filled by information in the listing, e.g.,

```
DIM A$ (2Ø,12)
LET A$(1) = "13A416B6PQR5"
LET A$(2) = "LMNOPQRSTU12"
```

etc.

the information is stored when the program runs, once in the listing, and once in array A$. It is often better to use DATA statements instead, i.e.,

```
DATA"13A416B6PQR5"
DATA"LMNOPQRSTU12"
```

etc.

and then READ the desired one of these into a short string when needed. This only applies when the information does not change as the game is played.

4. *User defined graphics, and* DRAW*n pictures can be efficiently coded into forms where they take up little memory*. This is discussed in the chapter on 'Adding Graphics to Adventures' (pages 174–9).

5. *In a large adventure, there may be dozens of lines* GOTO*ing or* GOSUB*ing particular routines*. By replacing all destination numbers with a variable, hundreds of bytes may be saved, e.g., if there are fifty GOSUB 1000 statements, replacing each by GOSUB a, and by having one initial LET a = 1000, approximately 200 bytes are saved.

6. *Do not create new variables and arrays for every task, but try to use the same ones again and again.* Most of the FOR-NEXT loops in my programs are f loops, and nested loops will be g, h and i loops. Similarly, a, b, c, d are used in many parts of a program.

7. *Avoid making arrays and string arrays larger than you need to.* A map of a Random adventure may be, for example, 40 by 40 locations, with the border ones being a wall. The array of objects will therefore only need to be 38 by 38 locations, as there will be no objects in a wall location. This leaves 156 more bytes than if a 40 by 40 object array were used.

8. *Tidy up finished programs.* In a long program like an adventure, many variables and strings are initialized and dimensioned. Some of these will be made redundant as the program is built up, and spaces at the ends of strings may be left for data which is never used in the end. Odd REM statements may be scattered through the program. Looking at the finished listing and trying to remove all the redundant material can be a large space-saver.

7. Preset Future System

The format of this adventure system is similar to **Preset Fantasy System**. The initial foundation program is derived from Foundation 1 from **Preset Fantasy System** and the same program structure is used throughout. However, the vocabulary is very different and the game develops in new directions.

The system is set in the future and you are an explorer of new worlds who gets into various difficult situations. There is no magic in this system, but you are aided by some of the best futuristic technology – spacecraft that cannot fly, communicators that won't communicate, blasters that get used up, and many others!

FOUNDATION F1 and SCENARIO 1:

Crater on Archelon One (fits into 16K)

While flying on a reconnaissance mission, on an unmapped planet, you eject from your planet hopper before it crashes into a crater. Although you have a communicator it will not work through the side of the crater and so you must get away from the crater before you can use it. Another problem is that your communicator has a flat powercell, so you will have to find the wreck of your hopper to get a spare.

TO PLAY

Your aim is to get out of the crater with your communicator, its antenna and a powercell. The basic command set is: go north, south, east, west, up and down, which can be abbreviated to n, s, e, w, u and d. Look repeats the description of the place you are in, take and put are as normal, and status gives a list of which objects you have. Other commands have to be worked out by yourself, as in the Preset Fantasy System.

TYPING IN THE LISTINGS

Type in **Foundation F1** and SAVE it for use with later scenarios. Then type in the *Crater on Archelon One* listing and SAVE it. LOAD **Foundation F1** and MERGE *Crater on Archelon One* to get **Scenario 1**. The complete scenario can then be SAVEd. Note that I have used inverse stops for the picture in the *Crater on Archelon One* listing, which makes it easier to count the graphic characters. When you type these in, use CHR$ 143 (the all-black graphic) instead.

EXPLANATION OF LISTINGS

Foundation F1
This listing is based on **Foundation 1** from chapter 2, and lines 6000–6840 are similar to those of **Foundation 1**. However, there are many small but significant alterations, and it is easier to type in the new listing from scratch than to alter **Foundation 1**.

100–320 The simplest vocabulary set. This is very different from Preset Fantasy, as the emphasis is less on rooms and corridors, and more on outdoor locations.

700–710 Removes leading spaces from inputted words.

750–840 Vocabulary decoder. Note that provision has been made for a larger vocabulary than that of Preset Fantasy, by using the character for a zero (CHR$ 48) to access a further list of words. Although not needed for **Foundation F1**, this will be used in the next scenario.

6000–6840 The input and act on command routines. The treatment of monsters has been changed from that in Preset Fantasy, and this means that many lines have GOTO 6580, instead of GOTO ru (ru is your location). Although line 6580 does not exist yet, this will be important in the other scenario.

6900–6990 You win.

7900–7050 Status routine.

7900–7940 You die, and a new game is offered, without having to re-run the program.

9010 The setting-up routines are now subroutines, so that a new game can be played without re-dimensioning all the arrays.

9650–9820 The object list, special action list and monster list are taken from code, the same as in **Preset Fantasy**.

Scenario 1: *Crater on Archelon One*

330–370 Words specific to this scenario.

1100–2450 Locations. As in Preset Fantasy, z$ is the coded description of a location, and d$ is the list of exits.

6900–6930 You win. A picture is drawn of you on the top of the crater using low-res printed landscape and a high-res DRAWing of a little man.

8000–8330 Special circumstances which occur in the various locations.

9110–9680 Set up variables. Note that there are no monsters in this scenario, so line 9680 skips the 'set up monster array' line!

9910 Introduction. Expand as desired.

Foundation F1

LISTING

```
 1 REM  Foundation F1
 2 REM © S.Robert Speel 1984

10 GO TO 9000
```

```
 100 DATA "You are","wall","abov
e","top"
 110 DATA "there","all","too","m
uch"
 120 DATA "and","ground","fall",
"here"
 130 DATA "west","xxx","crack","
side"
 140 DATA "east","which","break"
,"lower"
 150 DATA "north","centre","exce
pt","rising"
 160 DATA "south","around","high
er","surface"
 170 DATA "the","your","thick","
climb"
 180 DATA "on","by","through","s
catter"
 190 DATA "at","standing","of","
rocky"

 200 DATA "with","it","you","cov
ered"
 230 DATA "in","ed","are"
 240 DATA "s  "
 250 DATA "to"
 260 DATA "is"
 270 DATA "a  "
 280 DATA ",  "
 290 DATA ".  "

 300 DATA "        "
 310 DATA "     "
 320 DATA "    "

 700 FOR f=1 TO 6: IF a$(1)=" "
THEN LET a$=a$(2 TO ): NEXT f
 710 RETURN

 750 LET b=CODE v$: IF b=0 THEN
LET b=b+120
 760 LET c=INT ((b+1)/4)*10+10+(
k=1)*270: LET d=2+b-INT ((b+1)/4
)*4
 770 LET k=0: RESTORE c: FOR g=1
 TO d: READ b$:: NEXT g: RETURN

 800 LET z$=z$+CHR$ 255: LET ro=
0: LET a=0: LET k=0: FOR f=1 TO
LEN z$: LET v$=z$(f): IF v$="["
OR v$="X" OR v$="o" OR v$="s" OR
 v$="w" OR v$="{" OR v$="k" OR v
$="V" THEN PRINT CHR$ 8;
```

```
 810 IF v$="0" THEN LET k=1: NEX
T f
 820 GO SUB 750: PRINT CHR$ (COD
E b$(1)-a);b$(2 TO );" ";: LET a
=(v$="o")*32
 830 IF v$="g" OR v$="k" OR v$="
o" OR v$="[" THEN PRINT CHR$ 8;
 840 IF z$(f+1)<CHR$ 255 THEN NE
XT f
 850 RETURN

6000 POKE 23692,255: IF NOT ro T
HEN GO TO 6100
6010 GO SUB 800: PRINT '': LET r
o=0
6020 LET a=0: FOR f=1 TO LEN o$:
 IF q$(f)="1" THEN IF (CODE o$(f
)-96)*100+1000=ru THEN LET a=a+1
: PRINT INK 2;(" There is a" AN
D a=1);TAB 13;(CHR$ 8+CHR$ 8+CHR
$ 8+CHR$ 8+"& a " AND a>1);r$(f)
6030 NEXT f

6200 INPUT "What next? "; LINE a
$: LET c$=a$: PRINT 'TAB 2; BRIG
HT 1; PAPER 7;a$''
6210 LET a$=a$+"
   "
6220 GO SUB 700: RETURN

6300 LET at=0: IF mn THEN LET at
=1
6310 DATA "north","east","south"
,"west","up","down"
6320 RESTORE 6310: FOR f=1 TO 6:
 READ b$: IF a$( TO 2)=b$(1)+" "
 OR a$(4 TO 3+LEN b$)=b$ THEN LE
T rt=CODE d$(f): GO TO 6600
6330 NEXT f

6350 IF a$( TO 4)="look" THEN LE
T ro=1: GO TO ru
6360 IF a$( TO 4)="sear" THEN GO
 TO 6650
6370 IF a$( TO 4)="take" THEN GO
 TO 6700
6380 IF a$( TO 3)="put" THEN GO
TO 6800
6390 IF a$( TO 4)="stat" THEN GO
 TO 7000

6570 PRINT "You cannot do that."
6590 GO TO ru
```

```
6600 LET rt=(rt-96)*100+1000: IF
 rt<1 THEN PRINT "You cannot go
that way.": LET ro=0: GO TO ru
6610 LET ru=rt: LET ro=1: GO TO
6580

6650 LET a=0: FOR f=1 TO LEN o$:
 IF q$(f)="2" THEN IF (CODE o$(f
)-96)*100+1000=ru THEN PRINT INK
 3;"You find a ";r$(f): LET a=a+
1: LET q$(f)="1": NEXT f
6660 NEXT f: IF NOT a THEN PRINT
 INK 3;"You find nothing."
6670 GO TO 6580

6700 LET a$=a$(5 TO ): GO SUB 70
0
6710 FOR f=1 TO LEN o$: IF a$( T
O 3)=r$(f, TO 3) AND ((CODE o$(f
)-86)*100=ru OR q$(f)="£") THEN
GO TO 6740
6720 NEXT f

6730 PRINT TAB 6;"You cannot tak
e that.": GO TO 6580
6740 IF o$(f)="£" THEN PRINT TAB
 6;"You already have that!": GO
TO 6580
6750 IF q$(f)>"1" THEN GO TO 673
0
6760 IF ob>obm THEN PRINT TAB 4;
"You cannot carry any more.": GO
 TO 6580
6770 PRINT TAB 8;"Okay": LET ob=
ob+1: LET o$(f)="£": GO TO 6580

6800 LET a$=a$(4 TO ): GO SUB 70
0
6810 FOR f=1 TO LEN o$: IF a$( T
O 3)=r$(f, TO 3) AND o$(f)="£" T
HEN GO TO 6840
6820 NEXT f
6830 PRINT TAB 4;"You can't put
down something you don't have!":
 GO TO 6580
6840 PRINT TAB 8;"Okay": LET q$(
f)="1": LET o$(f)=CHR$ (ru/100+8
6): LET ob=ob-1: GO TO 6580

6900 PRINT ''"  You have succeed
ed in your    mission.  Well don
e."
```

```
6950 INK 0: FOR f=5 TO 10 STEP 2
: FOR g=1 TO 5 STEP 2: FOR h=1 T
O 5 STEP 2: BEEP .1,f*2: BEEP .1
,f-g*h: NEXT h: NEXT g: NEXT f
6990 STOP

7000 PRINT " Objects carried:";
7010 LET a=0: FOR f=1 TO LEN o$:
 IF o$(f)="£" THEN PRINT TAB 15;
r$(f): LET a=a+1
7020 NEXT f: IF NOT a THEN PRINT
 TAB 15;"none."
7050 GO TO 6580

7900 PRINT '" You are dead. "
7910 PRINT ''" Another try? (y/n
) "
7920 IF INKEY$="y" THEN BEEP 1,0
: GO SUB 9100: GO TO 9850
7930 IF INKEY$="n" THEN INK 0: P
APER 7: BORDER 7: CLS : STOP
7940 GO TO 7920

9000 RANDOMIZE
9010 GO SUB 9100: GO SUB 9600: G
O TO 9850

9100 LET mn=0: LET at=0: LET ru=
1100: LET ro=1: LET ob=0: LET ob
m=3
9210 RETURN

9650 RESTORE 9500: LET e=0: GO S
UB 9800: DIM r$(b,c): FOR f=1 TO
 b: LET r$(f)=t$(f): NEXT f
9670 GO SUB 9800: DIM s$(b,c): F
OR f=1 TO b: LET s$(f)=t$(f): NE
XT f
9690 GO SUB 9800: DIM m$(b,c): F
OR f=1 TO b: LET m$(f)=t$(f): NE
XT f
9700 RETURN

9800 LET a=0: DIM t$(b,c): FOR f
=1 TO b: READ b$: FOR g=1 TO c:
LET c$=CHR$ ((CODE b$(g)+120+g*2
)/2): LET t$(f,g)=CHR$ (CODE c$*
(c$<>"£")+32*(c$="£")): LET a=a+
CODE t$(f,g): NEXT g: NEXT f
9810 IF a<>d THEN PRINT "error i
n code, line ";9500+e*50: STOP
9820 LET e=e+1: RETURN
```

```
9900 INK 1: PAPER 6: BORDER 4: C
LS
9990 PRINT '': GO TO ru
```

Scenario 1: *Crater on Archelon One*

```
   1 REM    Scenario 1
     Crater on Archelon One
   2 REM © S.Robert Speel 1984

 330 DATA "mud","xxx","mist","fa
ce"
 340 DATA "crater","cliff","drow
n","ledge"
 350 DATA "weight","wreck","hard
ened","planet hopper"
 360 DATA "foot","creeper","powe
r cell","from"
 370 DATA "hangs"

1100 LET z$="#HCJs{,kK?▀$_/+;o'c
A{▀(<o"
1120 LET d$="bd0000": GO SUB 600
0
1150 GO TO 6300

1200 LET z$="#G?▛I?{s▀$k4c_@w/+7
o?▞cNWA▄[o"
1220 LET d$="0ca0n0": GO SUB 600
0
1230 IF a$( TO 3)=s$(2, TO 3) TH
EN GO TO 8000
1250 GO TO 6300

1300 LET z$="#HK?s{▀$_@7kK6,_;+3
k+=,_?/o"
1320 LET d$="0idb00": GO SUB 600
0
1350 GO TO 6300

1400 LET z$="#Cg1Xk▞{©>kKJ,:C(2[
o"
1420 LET d$="chea00": GO SUB 600
0
1430 GO SUB 8200
1450 GO TO 6300

1500 LET z$="#G?▛I?s{▀$4c_@{;+/o
_@7{c6,o"
1520 LET d$="df0000": GO SUB 600
0
```

```
1550 GO TO 6300

1600 LET z$="#CJ,kKw?▀$:_@{3o"
1620 LET d$="h0ge00": GO SUB 600
0
1650 GO TO 6300

1700 LET z$="'Y▀$[C({2[IM9?7k{+?
$[Y6.o"
1710 IF cu THEN LET z$=z$+"g▄█▄?
▖{&o"
1720 LET d$="f000m0": GO SUB 600
0
1730 IF a$( TO 2)=CHR$ 117+" " O
R a$(4 TO 5)=s$(4, TO 2) THEN GO
 TO 8100
1740 IF a$( TO 4)=s$(1, TO 4) AN
D a$(7 TO 11)=r$(6, TO 5) THEN G
O TO 8150
1750 GO TO 6300

1800 LET z$="#W?8I?w▀kK=,_w3+;k6
,s_/+7o"
1820 LET d$="ilfd00": GO SUB 600
0
1850 GO TO 6300

1900 LET z$="#HC▖▘w©kK=,C({2[o"
1920 LET d$="jkhc00": GO SUB 600
0
1930 GO SUB 8200
1950 GO TO 6300

2000 LET z$="#CJ,Kw?▀$[:_@7k/+3o
"
2020 LET d$="00i000": GO SUB 600
0
2050 GO TO 6300

2100 LET z$="#D?▖I@w▌kK▖[_w7+3o"
2120 LET d$="00li00": GO SUB 600
0
2150 GO TO 6300

2200 LET z$="#CJ,Kw▖[_@;+3o"
2220 LET d$="k00h00": GO SUB 600
0
2250 GO TO 6300

2300 LET z$="#C?▖&k%?▘o"
2320 LET d$="00000g": GO SUB 600
0
2330 GO SUB 8300
2350 GO TO 6300
```

```
2400 LET z$="#Cg▚o'Y{▄[4B_?&I?▗O
"
2420 LET d$="00000b": GO SUB 600
0
2430 IF a$( TO 3)=s$(2, TO 3) TH
EN GO TO 8000
2440 IF c$=CHR$ 117 OR a$(4 TO 5
)=CHR$ 117+CHR$ 112 THEN LET z$=
"MB=k+?s{▄5[oM-o": GO SUB 800: P
RINT : GO TO 8050
2450 GO TO 6300

6900 PRINT ''"You have escaped f
rom the         crater. All you ne
ed to do is     call for help and
 wait..."
6910 INK 0: PAPER 3: FOR f=1 TO
5: PRINT TAB 31;" ": NEXT f: PRI
NT "██▄▄▄▄▄ ▄▄▄";TAB 27;"▄▄█▄":
 FOR f=1 TO 2: PRINT "██████████
██";TAB 27;"█████": NEXT f: PRIN
T "███████████████▄    ▄  ▄███
███"
6920 FOR f=1 TO 32: PRINT "█";:
NEXT f
6930 INK 1: PLOT 234,40: LET a$=
"020402-4-104000302-3-2040202-30
002-2-3-3020200-4": FOR f=1 TO 4
8 STEP 4: DRAW VAL a$(f TO f+1),
VAL a$(f+2 TO f+3): NEXT f

8000 IF o$(5)<>"£" THEN PRINT "y
ou have no ";r$(5): GO TO ru
8010 IF ru=1200 THEN LET z$="?▄[
Y)A_{": GO SUB 800: PRINT " ";s$
(2): GO TO ru
8020 IF q$(6)<"3" THEN PRINT "Do
 not be a vandal.": GO TO ru
8030 PRINT " You ";s$(2, TO 3);"
 a ";r$(6): LET q$(6)="1": LET o
$(6)="£": LET ob=ob+1
8040 GO TO ru
8050 GO TO 7900

8100 IF NOT cu THEN GO TO 6580
8110 LET z$="MB?▄o": GO SUB 800:
 PRINT : LET ru=2300: LET ro=1:
GO TO ru
8150 IF o$(6)<>"£" THEN GO TO 63
00

8160 PRINT " You ";s$(1);" ";: L
ET z$="?▄G?s ▗&oL█'o": GO SUB 800
```

```
8170 LET cu=1: LET o$(6)="3": LE
T ob=ob-1: GO TO ru

8200 IF ob<4 THEN RETURN
8210 LET z$="@▜c}*k?w▗©5[kM-wE+▌
ɒ": GO SUB 800: PRINT : GO TO 80
50

8300 IF o$(1)<>"£" AND o$(1)<>"m
" THEN RETURN
8310 FOR f=1 TO 3: IF o$(f)<>"£"
 AND o$(f)<>"m" THEN RETURN
8320 NEXT f
8330 GO TO 6900

9110 LET lt=0: LET lu=0
9200 LET o$="££k£kn": LET q$="11
1123"
9210 LET ob=3: LET obm=4: LET cu
=0

9500 DATA "H£jJZX<86420","Lb\ZhX
L>8\PT","fbpJb<@BNL20","R\Dh>BDP
N420","\£TLH<:86420","LhLJ↑F↑864
20"
9510 DATA "nTf↑L","Lnj@>","nFXJ>
","pdB@>"

9640 LET b=6: LET c=12: LET d=58
85
9660 LET b=4: LET c=5: LET d=173
8
9680 GO TO 9700

9910 PRINT TAB 4;"Crater on Arch
elon One"'"' Your planet hopper
has crashed into a crater on a p
lanet you    are exploring. You n
eed to call for help, but your c
ommunicator battery is flat, and
 it won't    work in the mist tha
t fills the crater. Can you repa
ir your      communicator and get
 out of the crater so you can ca
ll for help?"
```

HINTS ON PLAY

This first scenario is fairly simple to solve. Remember to use 'go up' and 'go down' or 'u' and 'd', as 'up' and 'down' will not be accepted. There is no need to try and fit the powercell into the communicator, or to attach the antenna – just getting the three parts to the top of the crater will be sufficient.

FOUNDATION F2 and SCENARIO 2

Mountains of Sirius Two

TO PLAY

This scenario is much more difficult to win than *Crater on Archelon One*. You have landed your spaceship and gone exploring in your planet hopper. After landing on the shore of a large ocean, you find that your hopper does not have enough fuel left to take off again – its fuel tank was leaking. Your task is to get back to your ship which is somewhere to the west of you. You have a small dinghy – not much use on the rough ocean, but handy if you find a river. You also have a blaster, which can be shot twice, and a powercell to recharge your blaster once.

There are no special commands needed to use the dinghy – if you go on water and have the dinghy with you, the computer assumes you are in it. To use the blaster, type 'shoot', followed by the target name; e.g., 'shoot icebeast' (a good idea) or 'shoot powercell' (not such a good idea). To recharge the blaster you must be holding both the blaster and the powercell, and type 'recharge'. The charges in your blaster and powercell are now given when you type 'status'.

There are monsters in this scenario and once you meet one it must be dealt with immediately. Running away won't work, nor will using your blaster solve everything.

TYPING IN THE LISTINGS

The **Foundation F2** program consists of **Foundation F1** plus **Foundation F2**). SAVE this. Type in the listing for *Mountains of Sirius Two* and SAVE it.

Starting with **Foundation F2** in the computer, MERGE *Mountains of Sirius Two* to get the complete **Scenario 2**. This in itself is playable and should be SAVEd separately.

Foundation F2

```
   go south

You are at the foot of the
 crater wall which is to your
south and west. To your north
is lower ground.

   search

You find nothing.
   go east

You are on rocky ground, with
 the crater wall rising to your
   east.
```

210–310 Extra words.
6040 If a monster actually attacks, goto 'you get killed' routine.
6050–6060 Check if you meet a monster.
6300 If a monster is there, set at = 1 allows you to attack a monster only once before it kills you.
6400–6410 The new commands are 'shoot' and 'recharge'.
7030–7040 If you have blaster and powercell with you, they are now included in the stat command.
7100–7190 You fire your blaster. cha is your current blaster charge.
7200–7260 You recharge your blaster. Note that you can recharge a partially charged blaster.
7300–7310 You meet a monster.
7320–7340 The rather simple fighting system does not result in wounds – just sudden death for you or the monster!
9110 New variable associated with the blaster and recharger.

EXPLANATION OF LISTINGS

Scenario 2: *Mountains of Sirius Two*

330–390 There are more scenario-specific words now.

1100–3210 There are 32 locations now and this is a much more complex scenario than Scenario 1. Location 32 is different from all the others, as it includes the whole sea.

7180 Modification to 'shooting blaster' routine used against a particularly tough monster.

8000–8110 Special circumstances routines. The routine at 8050 checks if you are going west, as this can introduce many special circumstances.

9120–9910 Variables and array lists. t$ in line 9910 gives the positions of the monsters. Note it has to be dimensioned, as t$ is earlier used as a two-dimensional array while setting up.

Foundation F2

LISTING

```
   1 REM  Foundation F2
   2 REM © S.Robert Speel 1984

 210 DATA "valley","cavern","tra
il","dead"
 220 DATA "mountain","passage","
tunnel","ing"
 230 DATA "in","ed","are","goes"
 240 DATA "s ","up","see","has"

 250 DATA "to","go","from","can"
 260 DATA "is","lead","deep","ou
t"
 270 DATA "a ","end","wide","fil
l"
 280 DATA ", ","narrow","down","
hole"
 290 DATA ". ","large","plain","
river"

 300 DATA "    ","high","roof","
block"
 310 DATA "   ","have","exits","
no"
```

```
6040 IF at AND mn THEN GO TO 733
0
6050 FOR f=1 TO LEN t$: IF (CODE
 t$(f)-86)*100=ru THEN GO TO 730
0
6060 NEXT f

6300 LET at=0: IF mn THEN LET at
=1

6400 IF a$( TO 5)="shoot" THEN G
O TO 7100
6410 IF a$( TO 5)="recha" THEN G
O TO 7200
6580 IF mn THEN GO TO 7330

7030 IF bl THEN IF o$(bl)="£" TH
EN PRINT "blaster charge: ";cha
7040 IF bm THEN IF o$(bm)="£" TH
EN PRINT "Powercell charge: ";ch
b

7100 IF NOT bl THEN GO TO 6570
7110 LET fi=0: IF o$(bl)<>"£" TH
EN PRINT "You have no blaster.":
 GO TO 6580
7120 PRINT "You shoot your blast
er.": IF NOT cha THEN PRINT "Not
hing happens - it needs      rec
harging.": GO TO 6580
7130 FOR f=1 TO 20: BEEP .02,-20
: BEEP .01,10: NEXT f: LET cha=c
ha-1: LET fi=1
7140 IF mn THEN GO TO 7320

7150 LET a$=a$(6 TO ): GO SUB 70
0: FOR f=1 TO LEN o$: IF a$( TO
3)=r$(f, TO 3) AND (o$(f)="£" OR
 CODE o$(f)-86=ru/100) THEN GO T
O 7170
7160 BEEP .01,0: NEXT f: GO TO 6
580
7170 PRINT " You blow up the ";r
$(f): LET o$(f)=" "
7190 GO TO 6580

7200 IF bm THEN IF o$(bm)="£" TH
EN GO TO 7220
7210 PRINT "You have no powercel
l.": GO TO 6580
7220 IF NOT chb THEN PRINT " You
r powercell is flat.": GO TO 658
0
7230 IF o$(bl)<>"£" THEN GO TO 7
110
```

```
7240 PRINT "You recharge your bl
aster."
7250 LET a=1+(chb>1)-cha: LET a=
a*(a>0): LET cha=cha+a: LET chb=
chb-a: FOR f=1 TO a*3: FOR g=1 T
O 10: BEEP .01,g: NEXT g: NEXT f
7260 GO TO 6580

7300 LET mn=f: PRINT INK 2;" You
 meet a ";m$(mn)
7310 GO TO ru

7320 PRINT " You kill the ";m$(m
n): LET t$(mn)=" ": LET mn=0: GO
 TO ru
7330 LET n$=m$(mn): FOR f=LEN n$
 TO 3 STEP -1: IF n$(LEN n$)=" "
 THEN LET n$=n$( TO LEN n$-1): N
EXT f
7340 PRINT '"The ";n$;" kills yo
u.": GO TO 7900

9110 LET bl=0: LET cha=2: LET bm
=0: LET chb=2: LET fi=0
```

Scenario 2: *Mountains of Sirius Two*

LISTING

```
   1 REM  Future Scenario 2
        Mountains of Sirius Two

 330 DATA "headland","xxx","sea"
,"flow"
 340 DATA "soon","slide","face",
"water"
 350 DATA "beach","plateau","gla
cier","cliff"
 360 DATA "drown","slippery","sp
aceship","planet hopper"
 370 DATA "fire", "blaster","put
","take"
 380 DATA "offering","ignores",r
$(4 , TO 3),m$(5, TO 10)
 390 DATA r$(3, TO 6),"nest","ba
y"

1100 LET z$="?r© [EgU+fC_giqoMb]@
▇D?{r2O"
1120 LET d$="0b0000": GO SUB 800
: GO TO 6900
```

```
1200 LET z$="#W@HkCgslr4▘ [aw3_/o
"
1220 LET d$="0c0a00": GO SUB 600
0
1250 GO TO 6300

1300 LET z$="#WgPjXKe▜ogr▘ [f{_?/
k+gUd[\_?3o"
1320 LET d$="0d0b00": GO SUB 600
0
1330 IF o$(3)<>"£" THEN GO SUB 8
050: IF a THEN LET z$="MxzHk+Ms▛
o": GO TO 7900
1350 GO TO 6300

1400 LET z$="#WglU4d[/o'cgnG?3h4
d[mo"
1420 LET d$="000c0f": GO SUB 600
0
1430 IF a$( TO 4)=s$(1) AND mn T
HEN LET z$="?0*0(@s0$██o": GO SUB
 800: PRINT : GO TO 7330
1440 IF mn THEN IF a$( TO 3)=s$(
3, TO 3) OR a$( TO 4)=s$(9) THEN
 GO TO 8100
1450 GO TO 6300

1500 LET z$="#WgOKt{SIC(2[ogUd[3
o"
1520 LET d$="0f0000": GO SUB 600
0
1550 GO TO 6300

1600 LET z$="#WglTs4↑y_;+s/o'cgn
W?sUI?To"
1620 LET d$="00ged0": GO SUB 600
0
1650 GO TO 6300

1700 LET z$="#WgU4d[3+/o'cglT_?7
o"
1720 LET d$="fh0000": GO SUB 600
0
1730 GO SUB 8050: IF a THEN LET
z$="?U-[mC&IwMo": GO SUB 800: GO
 TO 7900
1750 GO TO 6300

1800 LET z$="#WgiUKgnW?uG?3who?U
Z/kw{dVW_?█ [o"
1810 IF NOT fo THEN LET z$=z$+"'
cg█vVM_?{/o"
```

```
1820 LET d$="000gq0": GO SUB 600
0
1830 IF NOT fo THEN GO SUB 8050:
 IF a THEN LET z$="M-W_?█o": GO
SUB 800: GO TO 7900
1840 FOR f=3 TO 5: IF a$( TO 3)=
s$(f, TO 3) THEN GO TO 8000
1850 NEXT f
1860 GO TO 6300

1900 LET z$="#GgRhkKg{█_@7k+twS[
_/+;o"
1920 LET d$="0j0000": GO SUB 600
0
1950 GO TO 6300

2000 LET z$="#Ct,o_7+;YS[k+_?3g▄
o"
2020 LET d$="0k0i00": GO SUB 600
0
2050 GO TO 6300

2100 LET z$="#C?/hIgw▄kKt,_?/o'Y
█[_7+;o"
2120 LET d$="0p0j00": GO SUB 600
0
2150 GO TO 6300

2200 LET z$="#C?3hIgw▄kKS[_?{;+g
█_?3o"
2220 LET d$="m00k00": GO SUB 600
0
2250 GO TO 6300

2300 LET z$="#C?/2Ig{0-kK?▀_@3k+
S[_?/o"
2320 LET d$="pvl000": GO SUB 600
0
2350 GO TO 6300

2400 LET z$="#Cg▄▀Ks{S[C(2[o'cgTE
?S[_@3k+glS▄▄QdV/o"
2420 LET d$="0o0000": GO SUB 600
0
2430 GO SUB 8050: IF a THEN LET
z$="M£/k+▀▄+-o": GO SUB 800: GO
TO 7900
2440 IF a$( TO 4)=s$(9) AND o$(4
)="£" THEN LET z$="LC[?F+Zo": GO
 SUB 800: LET o$(4)=" ": LET t$(
1)="0": LET mn=0: GO TO ru
2450 GO TO 6300
```

```
2500 LET z$="{#Cg0E?S[oLd[3+s/o"
2520 LET d$="0p0n00": GO SUB 600
0
2550 GO TO 6300

2600 LET z$="#CgJqkK{S[_@/og0d[E
?S[o"
2620 LET d$="qtmo00": GO SUB 600
0
2650 GO TO 6300

2700 LET z$="#Ct,kKwS[_@/o"
2710 IF o$(5)=" " THEN LET z$=z$
+"gn{d[mo"
2720 LET d$="rsp00h": GO SUB 600
0
2730 IF a$( TO 4)=s$(7) THEN LET
 a$=a$(5 TO ): GO SUB 700: LET d
$=a$: LET a$=c$: IF d$( TO 4)=r$
(5, TO 4) THEN LET z$="Lc}po": G
O SUB 800: GO TO ru
2740 IF a$( TO 2)=s$(8,1)+" " AN
D o$(5)<>" " THEN GO TO 6580
2750 IF a$( TO 2)="go" THEN LET
a$=a$(3 TO ): GO SUB 700: LET a$
=a$(1)+"          ": GO TO 2740
2760 GO TO 6300

2800 LET z$="#Ct,kKwS[_/+7k+g█_@
3£Vm_?▀ogpIcs.C?,o"
2820 LET d$="00q000": GO SUB 600
0
2850 GO TO 6300

2900 LET z$="#CgJ█K{?▀_7+3o"
2920 LET d$="vvtq00": GO SUB 600
0
2950 GO TO 6300

3000 LET z$="{#C?72Ig{0-kK?▀_@w;
o"
3020 LET d$="suvp00": GO SUB 600
0
3050 GO TO 6300

3100 LET z$="#CgJ©D{@▄o?▀c{_?7k;
+30"
3120 LET d$="vvvt00": GO SUB 600
0
3150 GO TO 6300

3200 IF o$(3)="£" THEN LET z$="M
£f_▀W@": GO SUB 800: PRINT r$(3,
 TO 6);: LET z$="k+▀-f+█o": GO S
```

```
UB 800: GO TO 7900
3210 LET z$="M£f_ªk+▛o": GO SUB
800: GO TO 7900

7180 IF f=5 THEN LET z$="M]gpndV
wmo": GO SUB 800: LET ro=0: GO T
O ru

8000 LET b$=a$(3 TO ): FOR f=1 T
O 5: IF b$(1)<>"s" THEN LET b$=b
$(2 TO ): NEXT f
8010 IF b$( TO 4)<>r$(6, TO 4) T
HEN GO TO 6300
8020 IF o$(6)<>"£" THEN PRINT "
you do not have a ";r$(6): GO TO
 ru
8030 LET o$(5)=" ": LET fo=1: PR
INT "The ";r$(6, TO 9);: LET z$=
"â[f?█o": GO SUB 800: LET o$(6)=
"¦": GO TO ru

8050 LET a=0: IF a$( TO 2)=s$(6,
1)+" " THEN LET a=1
8060 IF a$( TO 2)<>"go" THEN RET
URN
8070 LET b$=a$(3 TO ): FOR f=1 T
O 6: IF b$(1)=" " THEN LET b$=b$
(2 TO ): NEXT f
8080 IF b$(1)=s$(6,1) THEN LET a
=1
8090 RETURN

8100 IF o$(4)="£" THEN LET z$="?
0#0&[@0)w+Zo": GO SUB 800: LET o
$(4)=" ": LET t$(5)=" ": LET mn=
0: GO TO ru
8110 LET z$="?G▪es□o": GO SUB 80
0: GO TO 7330

9120 LET ru=3100: LET obm=2
9130 LET bl=1: LET bm=2: LET fo=
0

9200 LET o$="uuurqni": LET q$="1
111111
    "

9500 DATA "J\DffF↑86","fbPJbBDPN
","NU↑NNn:86","PRP@><:86","JblXF
F↑86","l££nB>RP6","Lhtff>R86"
9510 DATA "lT£↑","lNDd","fnj@","
£NZh","nTf↑","tNhh","nFXJ","NbP\
","RNLH"
9520 DATA "XJLDH>££6420","l££nBF
<↑↑420","jbHUBF<↑↑420","RNDhNF↑<
@6XX","ll£\H@D:\\20"
```

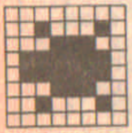
```
9640 LET b=7: LET c=9: LET d=556
9
9660 LET b=9: LET c=4: LET d=384
0
9680 LET b=5: LET c=12: LET d=54
98

9910 DIM t$(5): LET t$="njprd"
9920 PRINT "  THE MOUNTAINS OF S
IRIUS TWO"
9930 PRINT ''" You are on a scou
ting mission  on the planet Siri
us 2, when youfind that your pla
net hopper hasrun out of fuel. Y
ou recall     flying east from y
our spaceship,and going over som
e mountains,  before you landed 
on the coast  of an ocean. Can y
ou find your  way back, past the
 mountains, toyour spaceship? "
```

HINTS ON PLAY

You will find several false trails, and will need to draw a map. The locations are on more than one level, so the map you draw may have to be in various parts.

It is important to take advantage of the objects you find, but do not feel that there must be a use for every object.

8. Fangmole Tunnels System



This moving graphic system is based on an underground city – a maze of passages and rooms beneath the earth. The city was abandoned by its inhabitants when it was invaded by huge carnivorous burrowing creatures – the fangmoles. The original name of the city being forgotten, it is now known as the Fangmole Tunnels.

The adventure system builds up in several sections, from the initial 16K version where you are just exploring the maze, through to larger programs which involve you avoiding pits, snakes and bats, and the final version where you meet the fangmoles themselves.

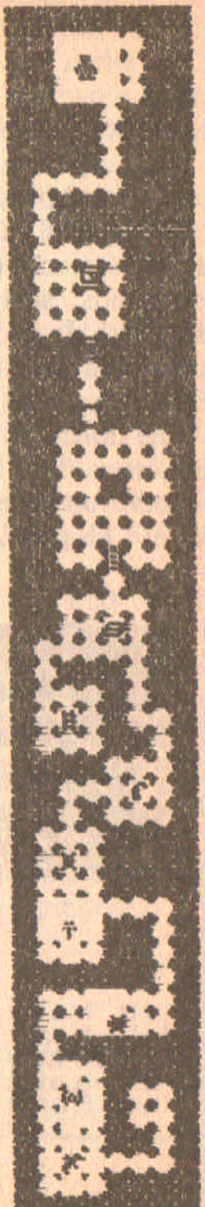
TO PLAY

The screen is divided into three parts. The top left shows a picture of your present location, and a small part of the surrounding area. You can see three locations in each direction, although this can alter in the more advanced versions. You can also see if there are any objects in the locations directly adjacent to your position, i.e., in the eight squares around you. You move around the maze by pressing the cursor control keys, which are held down to move you in a desired direction. As you move, the picture is scrolled left, right, up or down, so that your figure stays in the centre.

There are a number of objects, monsters and location types which you will come across, and these are shown below for easy identification.

Key to Symbols

♣ =	you
▓ =	wall
⁚ =	passage
▣ =	stairway
■ =	closed door
□ =	open door
■ =	pit
▦ =	ladder over pit
▰ =	gold bar
‖ =	ladder
ʃ =	exploder
✖ =	food sack
† =	torch
✱ =	fangmole
ѡ =	snake
✗=	bat

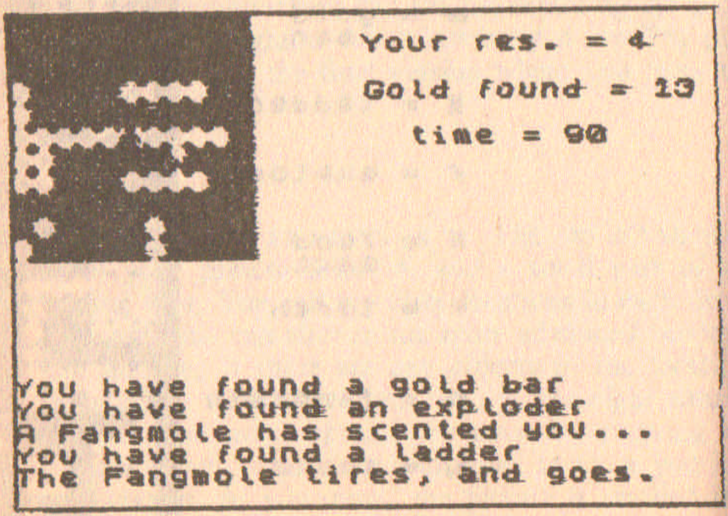


You open a closed door by pressing the 'O' key. However, you must be facing it – i.e., you must have been moving towards it. So when you are next to a door, hold down the key that would move you on to it for a second or two, then hold down the 'O' key to open the door.

To escape from the maze, you need to go on to a stairway location and press the 'U' key for up.

The second part of the screen is the panel in the lower third of the screen. This gives accompanying messages to your commands, such as 'You open the door' or 'You find an exploder'. It also gives details of any approaching monsters. After each message it scrolls upwards, so that only the latest few messages are displayed.

The third part of the screen is the panel in the top right of the screen. This gives your resistance, the number of gold bars (ingots) you have found and the time that you have been in the maze. This panel is updated continually and does not scroll.



```
Your res. = 4
Gold found = 13
time = 90

You have found a gold bar
You have found an exploder
A Fangmole has scented you...
You have found a ladder
The Fangmole tires, and goes.
```

FANGMOLE TUNNELS — CORE PROGRAM

The Maze (fits into 16K)

TO PLAY

The people who built the underground city fled when the fangmoles attacked, leaving vast quantities of gold lying around. As the fangmoles returned to lower depths, the tunnels were left unguarded for a while.

You are an explorer in search of gold. You have made your way into the tunnels, and your aim is to pick up as many gold ingots as you can find. Movement is with the cursor keys. Note that you have to hold the keys down for a moment before there is a response. The only objects are gold ingots, and you pick up an ingot by moving into the location containing it, then holding down the 'T' key (take) until you hear a short blip.

You must exit the tunnels within 300 seconds, and you do this by moving on to a stairway symbol and pressing 'U' (for up). Doors can be opened by moving towards the door and pressing 'O' (for open) when you are adjacent. Note that just being next to a door is not enough, you have to move towards it. Pressing the correct movement key for a second, as if to move on to the door, ensures that it will open when you press the 'O' key.

You score 50 points for each gold bar you find, but if you fail to get out within the time limit, you lose 300 points.

TYPING IN THE LISTING

The **Fangmole Tunnels** listing should be typed and SAVEd *before* testing, because if the machine code routine data in line 971Ø is incorrectly entered you could crash the system.

The lines 91ØØ–9145 and line 92ØØ contain user-defined graphic characters, *not* normal capital letters. When you run the program, the listing will alter to leave these lines as solid graphics.

EXPLANATION OF LISTING

10 CLEAR space for the machine code scrolling routine and goto initialize variables.

1000 Print man symbol and surrounding part of maze. To stop you going too near the edge of the maze and crashing the routine, there is a three-location thick wall around the perimeter of the map.

1010 Print any objects in nearest eight locations.

1050 Calls the machine code routine to scroll the lower third of the screen, and prints on the lowest line.

1200–1290 You take a gold bar. y$ is the object array, and (y,x) your position. K is 1050 and so GOSUB K scrolls the lower third of the screen and prints on the bottom line.

1800–1830 You move. This routine stops you walking into walls and closed doors. b$ is the direction you last moved in.

2000–2150 Enter command routine. 2050 deals with opening doors, which are held in z$, the map array.

2200–2230 You try to exit the maze. If you are on a stairway, you succeed. aim = 0, but in later expansions it can change its value.

4000–4090 The main routine. All other subroutines are called from this when play begins. ta is the time since you entered. If ta > tb, the maximum time allowed (tb = 300), then the game is ended with you losing 300 points.

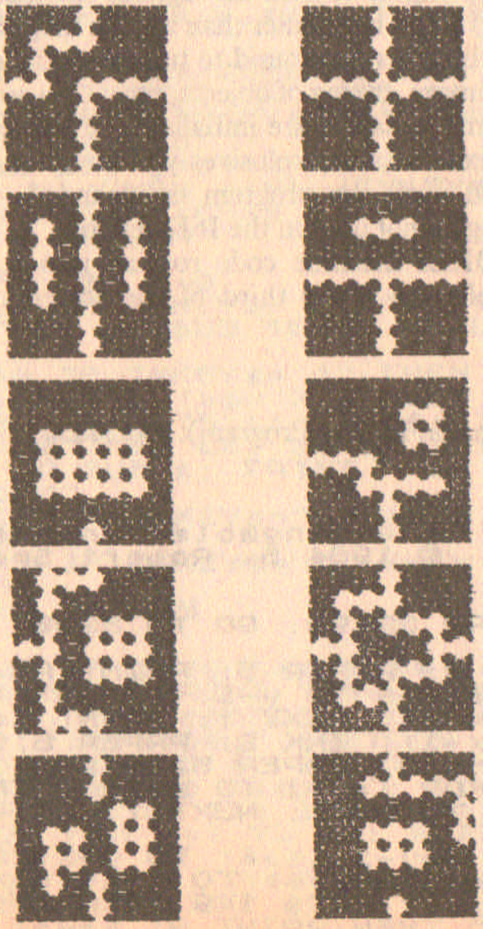
7000–7050 Set up screen and assign starting values to variables.

7500–7690 Print introduction and gosub set up maze procedures. a and b are the length and width of the maze in 'blocks'. See explanation of lines 9000–9220.

8000–8010 Setting up control routine. This gosubs all the routines before the game starts.

8200–8500 Game ended, your points calculated, and a deduction of 300 points is made if you were over the time limit.

9000–9220 Set up maze routine. The maze is constructed of square blocks, held in lines 9100–9140. Each of these contains a short section of corridor, or perhaps a room, and they are planned so that the corridors tend to join up when two blocks are placed next to each other. Note that the lines 9100–9140 contain UDG letters, not normal capitals. Lines 9000–9080 set up a maze using a x b blocks. Each block can be rotated in any direction, and reflected in any direction, so that it has a maximum of eight orientations. Blocks with lines of symmetry have fewer


Maze Building Sections

orientations. With sixteen block spaces used, each filled with any one of eight blocks with up to eight possible orientations, there is a very large number of possible variations in the maze. Lines 9200–9220 put a border around the maze of three characters thick to prevent you escaping.

9300–9370 Data for UDGs and routine to POKE them into position. Some memory could be saved by removing the commas, but it is easier to read the data in the format used.

9400–9450 Puts ingots of gold throughout the maze. y$ (the object array) could be smaller than z$ (the map) to miss out the border, but this was found to unnecessarily complicate printing the maze, taking of objects, etc.

9500–9610 Some variables are initialized. ld and exp are the number of ladders and explosives you have collected, and are not used until the program is expanded. rs is your resistance, again not used in the 16K version.

9700–9720 POKEs machine code routine into RAM. This routine scrolls the lower third of the screen up by one character.

Fangmole Tunnels 1 (Core Program): *The Maze*
LISTING

```
   1 REM  The Fangmole Tunnels
   2 REM  © 1984 S. Robert Speel

  10 CLEAR 32499: GO TO 8000

1000 INK 0: PAPER 5: PRINT AT 0,
0;: FOR f=y-s TO y-1: PRINT z$(f
,x-s TO x+s): NEXT f: PRINT z$(y
,x-s TO x-1); INK 2; PAPER 6;CHR
$ 148; INK 0; PAPER 5;z$(y,x+1 T
O x+s): FOR f=y+1 TO y+s: PRINT
z$(f,x-s TO x+s): NEXT f: RETURN

1010 FOR f=y-1-(s>4) TO y+1+(s>4
): FOR g=x-1-(s>4) TO x+1+(s>4):
 IF z$(f,g)<>CHR$ 149 THEN IF y$
(f,g)>" " THEN PRINT AT s-y+f,s-
```

```
x+g; PAPER 7; INK 1+(y$(f,g)<"█"
); BRIGHT (y$(f,g)<"█");CHR$ (CO
DE y$(f,g)+100*(y$(f,g)<"█"))
1020 NEXT g: NEXT f: RETURN

1050 LET a=USR 32500: PRINT AT 2
1,0;: RETURN

1200 LET jk=CODE y$(y,x): IF jk<
151 THEN RETURN
1210 GO SUB k: PRINT "You have f
ound a gold bar.": BEEP .1,10
1220 LET gold=gold+1: PRINT AT 3
,29;gold
1290 LET y$(y,x)=" ": RETURN

1800 LET b$=INKEY$: LET m=1: LET
 x=x+(INKEY$="8" AND z$(y,x+1)<C
HR$ 148)-(INKEY$="5" AND z$(y,x-
1)<CHR$ 148)
1810 LET y=y+(INKEY$="6" AND z$(
y+1,x)<CHR$ 148)-(INKEY$="7" AND
 z$(y-1,x)<CHR$ 148)
1830 RETURN

2000 LET m=0
2010 IF INKEY$>"4" AND INKEY$<"9
" THEN GO SUB 1800: RETURN

2050 IF INKEY$="o" THEN LET y1=y
+(b$="6")-(b$="7"): LET x1=x+(b$
="8")-(b$="5"): IF z$(y1,x1)=CHR
$ 150 THEN LET z$(y1,x1)=CHR$ 14
6: GO SUB k: PRINT "You open the
 door"
2110 IF INKEY$="u" THEN GO SUB 2
200
2120 IF INKEY$="t" THEN GO SUB 1
200
2150 RETURN

2200 IF z$(y,x)<>CHR$ 147 THEN G
O SUB k: PRINT "You look for an
exit, but cannot": GO SUB k: PRI
NT "find one.": RETURN
2210 GO SUB k: PRINT "You use a
stairway to escape": GO SUB k: P
RINT "from the subsystem."
2220 FOR f=1 TO 20: BEEP .05,f:
NEXT f
2230 INK 4: PAPER 0: CLS : GO TO
 8500+aim*100
```

```
4000 LET ma=0: LET rs=4: LET s=3
: GO SUB 7000
4010 LET ta=ta+1: GO SUB 1000: I
F ta>tb THEN GO TO 8200+100*aim
4040 GO SUB 1010: PRINT AT 5,25;
ta
4060 INK 0: GO SUB 2000
4090 GO TO 4010

7000 LET tte=0: LET k=1050
7010 INK 0: PAPER 0: BORDER 0: C
LS
7020 FOR f=16 TO 21: PRINT AT f,
0; PAPER 5;TAB 31;" ": NEXT f
7030 FOR f=0 TO 6: PRINT PAPER 4
;AT f,15;TAB 31: NEXT f
7040 PRINT PAPER 5; INK 1;AT 1,1
6;"Your res. = ";rs;AT 3,16;"Gol
d found = ";gold;AT 5,18;"time =
 ";ta
7050 RETURN

7500 PRINT TAB 6;"FANGMOLE"''" Y
our aim:"'
7510 LET aim=0

7600 PRINT "You must collect as
many gold   ingots as you can, a
nd escape   before dawn, in 300
secs."
7610 LET a=4: LET b=4: GO SUB 90
00
7620 GO SUB 9400
7630 GO SUB 9500
7640 LET tb=300

7660 FOR f=1 TO 8: LET z$(15+RND
*15,15+RND*15)=CHR$ 147: NEXT f
7670 LET x=23: LET y=23: FOR f=1
 TO 5: IF z$(y,x)>CHR$ 147 THEN
LET x=x+1: NEXT f
7690 BEEP .01,20: NEXT f: RETURN

8000 RANDOMIZE : INK 6: PAPER 0:
 BORDER 2: CLS : GO SUB 9300: GO
 SUB 9700
8010 LET ta=0: GO SUB 7500: GO T
O 4000

8200 FOR f=1 TO 20: BEEP .05,f:
BEEP .05,-f: NEXT f: PAPER 0: IN
K 3: CLS
8210 PRINT " Your time is up, an
d you did   not escape."
```

```
8220 PRINT "You get -300 points
for not    escaping.": LET pt=-
300
8230 PRINT '"You found ";gold;"
gold bars"'"and get ";gold*50;"
points": LET pt=pt+gold*50
8240 PRINT "Total Points = ";pt:
 STOP

8500 PRINT " You escaped from th
e subsystem.": LET pt=0: GO TO 8
230

9000 RANDOMIZE : DIM z$(a*7+6,b*
7+6): PRINT AT 8,10;"Setting up
maze"
9010 FOR f=0 TO a-1: FOR g=0 TO
b-1: PRINT AT 10,0;f;g: GO SUB 9
100+5*INT (RND*4)+30*(RND<.5)
9020 LET x=1+6*(RND>.5): LET y=1
+6*(RND>.5)
9030 LET to=1: LET rf=(RND>.5)

9040 FOR h=x TO x+6*SGN (2-x) ST
EP SGN (2-x): FOR i=y TO y+6*SGN
 (2-y) STEP SGN (2-y): IF rf THE
N LET sw=h: LET h=i: LET i=sw
9050 LET z$(f*7+h+3,g*7+i+3)=x$(
to): LET to=to+1: IF rf THEN LET
 sw=h: LET h=i: LET i=sw
9060 NEXT i: NEXT h
9070 BEEP .05,10
9080 NEXT g: NEXT f
9090 GO TO 9200

9100 LET x$="FFFAFFFFFFAFFFFFFAF
FFAAAAAAAFFFAFFFFFFAFFFFFFAFFF":
 RETURN
9105 LET x$="FFFAFFFFAAAFFFFAFFF
FFAAFAAAAFFFAFFFFFFAFFFFFFAFFF":
 RETURN
9110 LET x$="FFFGFFFFFAAAFFFFFFF
FFAAAAAAAFFFAFFFFFFAFFFFFFAFFF":
 RETURN
9115 LET x$="FFFAFFFFAFAFAFFAGAF
AFFAFAGAFFAFAFAFFFFAFFFFFFAFFF":
 RETURN

9130 LET x$="FFFAFFFFAAAFFFFAFAA
AFAAFFFAAFAFAAAFFAAAFFFFFFAFFF":
 RETURN
9135 LET x$="FAAAFFFFAFFFFFFAFAA
AFAAGAAAFFAFAAAFFAFFFFFFAAAFFF":
 RETURN
```

```
9140 LET x$="FFFAFFFFFAAAAAFFFGF
FAFFAAAFAFFAAAFFFFFGFFFFFFAFFF":
 RETURN
9145 LET x$="FFFAFFFFFFAGFFFFFFA
AFGAAFAAFFAAFAAFFAGAGFFFFFAFFF":
 RETURN

9200 LET a$="FFFFF": LET b$=a$:
FOR f=1 TO b*2: LET b$=b$+a$: NE
XT f: FOR f=1 TO a*7+3: LET z$(f
, TO 3)=a$: LET z$(f,b*7+3 TO )=
a$: NEXT f: FOR g=1 TO 3: LET z$
(g)=b$: LET z$(a*7+7-g)=b$: NEXT
 g
9220 BEEP .5,5: RETURN

9300 RESTORE 9300: DATA "195,129
,000,000,000,000,129,195,219,219
,195,090,066,219,195,219,255,195
,129,129,129,129,195,255,127,065
,093,075,069,125,001,255,"
9310 DATA "056,056,016,124,186,1
86,040,108,219,255,118,223,251,1
10,255,219,"
9320 DATA "255,195,189,189,189,1
89,195,255,000,000,063,067,253,1
34,252,000,000,036,060,036,060,0
36,060,036,"

9350 LET a$="": FOR f=1 TO 3: RE
AD b$: LET a$=a$+b$: NEXT f: LET
 b$=""
9360 LET h=1: FOR f=144 TO 152:
FOR g=0 TO 7: LET a=VAL a$(h TO
h+2): POKE USR CHR$ f+g,a: LET h
=h+4: NEXT g: NEXT f
9370 BEEP 1,10: RETURN

9400 PRINT AT 10,4;"Filling maze
 with objects": DIM y$(a*7+6,b*7
+6): FOR f=4 TO a*7+3: FOR g=4 T
O b*7+3: IF RND<.8 THEN NEXT g:
NEXT f
9430 LET y$(f,g)=CHR$ 151: NEXT
g: BEEP .01,0: NEXT f
9450 BEEP 1,30: RETURN

9500 PRINT "Initial variables":
LET d$="": LET gold=0:
9520 LET ld=0: LET rt=0
9530 LET exp=0
9550 LET rs=4: LET ma=0
9600 PAPER 0: BORDER 0: CLS
9610 RETURN
```

```
9700 RESTORE 9710: LET h=1: READ
 a$: FOR f=32500 TO 32530: POKE
f,VAL a$(h TO h+2): LET h=h+4: N
EXT f
9710 DATA "033,032,080,017,000,0
80,062,008,001,224,000,237,176,0
14,032,237,074,079,062,000,006,0
32,018,019,016,0-4,121,061,032,-
22,201"
9720 RETURN
```

HINTS ON PLAY

It may take you a little while to get used to the movement system, and to opening doors. Remember to keep an eye on the time, and remember also in which direction the nearest exit is. When exploring rooms, you do not have to look all around for objects, but can stay one location away from the walls, and thus save time.

ADDITION 1
gives FANGMOLE TUNNELS 2
The Miners

Since the tunnels were deserted, many people have been there to collect and mine the gold. The new situation is that there are fewer gold ingots around, but quite a lot of exploders being used by the miners to excavate.

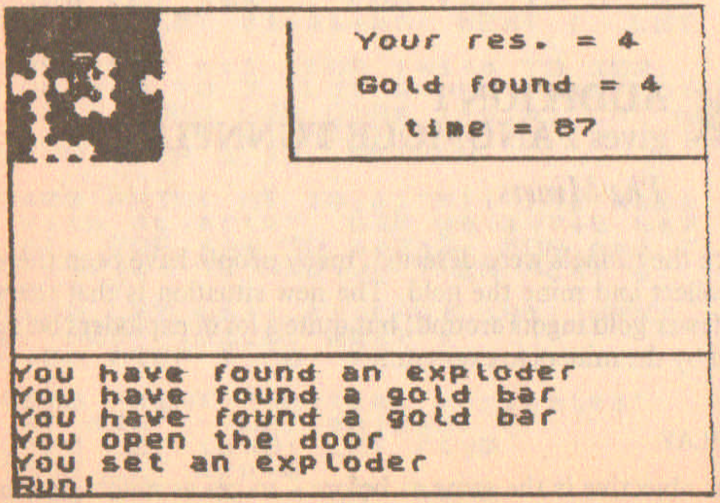
TO PLAY

Your objective is the same as before – to get as much gold as possible and exit within 300 seconds. However, exploders are now lying around, and these can be taken in the same way as gold. Exploders are used to get you to places which would otherwise be difficult or impossible to reach.

To use an exploder, you put it down by pressing the 'P' key. There will be a message telling you to run, and you will hear a high-pitched bleeping. This is the warning for you to get away. You may have between two and ten seconds to escape before the exploder blows up. When the explosion occurs, a random blast will be produced, destroying walls and corridors and objects alike. If you are hit, you die, and you not only end the game early, but also lose 400 points. However, with a bit of luck and skill, you can blast your way through to anywhere you want to go. But do not try to use explosives near the edge of the maze – if you could go into the border the program would crash, and if the exploder destroys any of the border, the blast kills you too!

By pressing the 'I' key for inventory, you are told how many exploders you have. You are also told that you have 0 ladders, which are used in the next version of the game.

You may notice that the mazes have even more variety now, as an extra two possible blocks have been added.



```
Your res. = 4
Gold found = 4
time = 87
```

```
You have found an exploder
You have found a gold bar
You have found a gold bar
You open the door
You set an exploder
Run!
```

TYPING IN THE LISTING

Type in **Addition 1** and SAVE it. LOAD original program, **Fangmole Tunnels**, and MERGE **Addition 1**. SAVE the complete program. Note that you must *not* RUN the original program before MERGEing **Addition 1**, as the original program CLEARs the memory to an area too small to run the expanded program.

Remember that lines 9120 and 9125 contain UDGs.

EXPLANATION OF LISTING

10 This program now needs a 48K Spectrum, so CLEAR is no longer used.
1050 The machine code is now stored at address **60000**, to prevent it being overwritten by the basic program.
1210–1240 You may find explosives as well as gold.
1500–1570 You set an exploder. This routine takes over from the one at **4000** while there is an exploder getting ready to blow up, so you can still collect objects. If you are destroyed in the blast, you lose 400 points.
1700–1720 Inventory. Tells you how many exploders and ladders you have.
2100–2130 Extra commands 'P' for put exploder, 'I' for inventory.
7620–7690 Addition to starting routine to include exploder in the maze, and extra gold.
8910–8950 Another game offered. This means you do not have to wait while the machine code is re-poked and the UDGs defined.
9000–9200 A thicker border is included as a preparation for the next addition. Two new maze block types are included, and the block selector modified. Remember to type graphic characters for lines 9120, 9125 and 9200 statement 1.
9330–9360 New UDGs are needed.
9400–9440 Exploders as well as gold are put in the maze.
9700 The machine code has been moved to address **60000**.

Fangmole Tunnels 2: *The Miners*
LISTING

```
   1 REM  The Fangmole Tunnels
         Addn 1
   2 REM  © 1984 S. Robert Speel
  10 GO TO 8000

1050 LET a=USR 60000: PRINT AT 2
1,0;: RETURN

1210 GO SUB k: PRINT "You have f
ound a";("n" AND c$(jk-150,1)="e
");" ";c$(jk-150): BEEP .1,10
1220 IF jk=151 THEN LET gold=gol
d+1: PRINT AT 3,29;gold: GO TO 1
290
1240 IF jk=153 THEN LET exp=exp+
1

1500 IF exp=0 THEN GO SUB k: GO
SUB k: PRINT "You can't do that.
": GO SUB k: RETURN
1510 GO SUB k: PRINT "You set an
 exploder": LET exp=exp-1: LET e
px=x: LET epy=y: LET tte=2+INT (
RND*8): GO SUB 1050: PRINT "Run!
"
1520 GO SUB 1000: GO SUB 1010: G
O SUB 2000: BEEP .1,40: LET tte=
tte-1: LET ta=ta+1: IF tte>0 THE
N GO TO 1520
1530 GO SUB k: PRINT AT 21,0;"BO
OM!!!": FOR f=1 TO 5: LET a=USR
60000: NEXT f: FOR f=1 TO 20: BE
EP .05,-20: BEEP .05,-30: NEXT f
1540 FOR f=epx-1 TO epx+1: FOR g
=epy-1 TO epy+1: IF RND<.5 THEN
LET z$(g,f)=" ": LET y$(g,f)=" "

1550 IF f=x AND g=y OR f<6 OR f>
b1*7+5 OR g<6 OR g>a1*7+5 THEN F
OR f=y-1 TO y+1: PRINT AT s-y+f,
s-1;"   ": NEXT f: GO SUB k: PRI
NT "You are destroyed in the bla
st.": BEEP 2,0: LET pt=-400: INK
 2: PAPER 0: CLS : PRINT '''"Blo
wn up...-400 points": GO TO 8230
1570 NEXT g: NEXT f: RETURN

1700 FOR f=1 TO 5: LET a=USR 600
00: NEXT f
```

```
1710 PRINT AT 16,2;"INVENTORY"'"
 You have:  ";exp;" explosives"
;TAB 15;ld;" ladders"
1720 GO TO 1000

2100 IF INKEY$="i" THEN GO SUB 1
700
2130 IF INKEY$="p" AND NOT ma TH
EN GO TO 1300

7620 LET c=.25: LET j1=151: LET
j2=2: GO SUB 9400
7640 LET tb=300: PRINT AT 10,10;
"Wait a bit more."
7680 FOR f=6 TO 30: FOR g=6 TO 3
0: IF RND<.05 THEN LET y$(f,g)=C
HR$ 151
7690 NEXT g: BEEP .01,20: NEXT f
: RETURN

8240 GO TO 8900

8910 PRINT ''"TOTAL POINTS: ";pt
8920 PRINT ''"Another game? (y/n
)"
8930 LET a$=INKEY$: IF a$="y" TH
EN CLEAR : GO TO 8010
8940 IF a$="n" THEN PAPER 7: INK
 0: BORDER 7: CLS : STOP
8950 GO TO 8930

9000 RANDOMIZE : DIM z$(a*7+10,b
*7+10): LET a1=a: LET b1=b: PRIN
T AT 8,10;"Setting up maze"
9010 FOR f=0 TO a-1: FOR g=0 TO
b-1: PRINT AT 10,0;f;g: GO SUB 9
100+5*INT (RND*10)
9020 LET x=1+6*(RND>.5): LET y=1
+6*(RND>.5)
9050 LET z$(f*7+h+5,g*7+i+5)=x$(
to): LET to=to+1: IF rf THEN LET
 sw=h: LET h=i: LET i=sw

9120 LET x$="FFFFFFFFFFFAAFFFFFA
AFAAAFFGFFFAAAAFFFFAFFFFFAFFFF":
 RETURN
9125 LET x$="FFFFFFFFFFFFFFFAAAA
AFGAAAAAFFAAAAAFFFFGFFFFFAFFFF":
 RETURN

9200 LET a$="FFFFF": LET b$=a$:
FOR f=1 TO b*2: LET b$=b$+a$: NE
XT f: FOR f=1 TO a*7+5: LET z$(f
, TO 5)=a$: LET z$(f,b*7+6 TO )=
a$: NEXT f: FOR g=1 TO 5: LET z$
(g)=b$: LET z$(a*7+11-g)=b$: NEX
T g

9330 DATA "000,012,018,048,048,0
48,048,000,000,068,056,056,056,0
56,068,000,000,040,000,084,016,0
16,016,000,"

9350 LET a$="": FOR f=1 TO 4: RE
AD b$: LET a$=a$+b$: NEXT f: LET
 b$=""
9360 LET h=1: FOR f=144 TO 155:
FOR g=0 TO 7: LET a=VAL a$(h TO
h+2): POKE USR CHR$ f+g,a: LET h
=h+4: NEXT g: NEXT f

9400 PRINT AT 10,4;"Filling maze
 with objects": DIM y$(a*7+10,b*
7+10): FOR f=6 TO a*7+5: FOR g=8
 TO b*7+5: IF RND<.8 THEN NEXT g
: NEXT f
9430 LET y$(f,g)=CHR$ (2*INT (RN
D*j2)+j1): NEXT g: BEEP .01,0: N
EXT f
9440 DIM c$(5,8): LET c$(1)="gol
d bar": LET c$(3)="exploder"

9700 RESTORE 9710: LET h=1: READ
 a$: FOR f=60000 TO 60030: POKE
f,VAL a$(h TO h+2): LET h=h+4: N
EXT f
```

ADDITION 2
gives FANGMOLE TUNNELS 3

Pits and Snakes

Time has passed and the fangmole tunnels are becoming less hospitable. The ground is less steady, and using explosives is more dangerous. There are bottomless pits in the corridors. Rattlesnakes have moved in and lurk in wait of prey.

TO PLAY

There are pits now, and if you try to move on top of one you fall down it and die. To get over a pit, you need a ladder. Ladders are now scattered around the mazes, and can be taken as normal. The put command has altered slightly. Type 'P' and you are asked 'Put down what?' Then type 'E' for exploder, or 'L' for ladder. If you put down a ladder while adjacent to a pit, the ladder will automatically be placed over the hole. If not adjacent to a hole, the ladder will be put down in the location that you are in. If you are next to more than one hole, north has priority over east in the placing of ladders, east over south and south over west.

Using an exploder can now lead to an earthquake, which produces pits randomly over a fairly wide area. These pits may be filled in by more explosions, but be wary of getting trapped!

A new hazard is the rattlesnake. These can be seen at the same distance as objects, and unless stepped on they will ignore you. They tend to block your way down a passage or across a room so that you cannot avoid them. If you step on a rattlesnake, your resistance (shown on the top right of the screen) decreases by 1. If your resistance reaches 0, you die. Fortunately, sacks of food left by other adventurers are distributed around the maze, and when you take one of these (same command as for gold, etc.), your resistance is increased by 1.

Another change is your ability now to carry on even after being killed. If you die, either from an explosion, falling down a pit, or from a snakebite, you are resurrected. Your resistance goes back to 4, but things are otherwise unchanged. However, you are penalized by the loss of 160 points for each resurrection.

TYPING IN THE LISTING

Type in the listing **Addition 2** and SAVE it. LOAD **Fangmole Tunnels 2**, and MERGE **Addition 2**, to get the new version. SAVE it.

EXPLANATION OF LISTING

123Ø–125Ø Taking ladders and food. ld is the number of ladders you have, rs is your resistance.
13ØØ–134Ø Put routine modified to cope with both ladders and exploders.
14ØØ–142Ø You put down a ladder. If there is a hole next to you, put a ladder over it. If not, put the ladder down where you are.
155Ø–159Ø If you are killed by blast, go to resurrection routine. 20 per cent chance exploder starts earthquake.
182Ø You fall into a pit.
3ØØØ–3Ø3Ø You walk on a snake. Your resistance is reduced by 1, and if Ø, you die.
4Ø5Ø Check for presence of snake.
423Ø Resurrection.
76ØØ–768Ø Various variables changed to give larger maze, and longer playing time.
89ØØ Points lost for resurrection.
934Ø–936Ø Extra UDGs.
941Ø–943Ø Alterations to put different objects in y$, and pits in z$. Also puts snakes in y$.

Fangmole Tunnels 3: *Pits and Snakes*

LISTING

```
   1 REM  The Fangmole Tunnels
            Addn 2
   2 REM  © 1984 S. Robert Speel

1230 IF jk=152 THEN LET ld=ld+1:
 GO TO 1290
1250 IF jk=154 THEN LET rs=rs+1:
 GO SUB k: PRINT "You eat, and g
et stronger.";AT 1,28;rs

1300 GO SUB k: PRINT "Put down w
hat?"
1310 IF INKEY$="l" AND ld>0 THEN
 GO TO 1400
1320 IF INKEY$="l" AND ld=0 THEN
 GO SUB k: PRINT "You can't do t
hat.": GO TO 1000
```

```
1330 IF INKEY$="e" THEN GO TO 15
00
1340 GO TO 1310

1400 LET ld=ld-1: BEEP .1,-5: GO
 SUB k: PRINT "You put down a la
dder.": GO SUB k
1410 LET a$="12211001": FOR g=1
TO 7 STEP 2: IF z$(y+VAL a$(g+1)
-1,x+VAL a$(g)-1)<>"■" THEN NEXT
 g: LET y$(y,x)=CHR$ 152: RETURN

1420 LET z$(y+VAL a$(g+1)-1,x+VA
L a$(g)-1)=CHR$ 145: LET y$(y+VA
L a$(g+1)-1,x+VAL a$(g)-1)=" ":
RETURN

1550 IF f=x AND g=y OR f<6 OR f>
b1*7+5 OR g<6 OR g>a1*7+5 THEN F
OR f=y-1 TO y+1: PRINT AT s-y+f,
s-1;"   ": NEXT f: GO SUB k: PRI
NT "You are destroyed in the bla
st.": BEEP 1,0: GO TO 4230
1570 NEXT g: NEXT f: IF RND<.8 T
HEN RETURN

1580 FOR f=1 TO 3: LET a=USR 600
00: NEXT f: PRINT AT 18,0;"You h
ave started a small earth- quake
! Chasms appear in the    groun
d..."
1590 FOR f=1 TO 2+RND*10: LET a=
INT (RND*7): LET b=INT (RND*7):
LET z$(epy-3+a,epx-3+b)="■": LET
 y$(epy-3+a,epx-3+b)=" ": NEXT f
: RETURN

1820 IF z$(y,x)="■" THEN GO SUB
k: PRINT AT 20,0;"You fall into
a chasm and plunge to your death
.": BEEP 1,0: GO TO 4230

3000 BEEP .5,-50
3010 GO SUB k: PRINT "You step o
n a rattlesnake!": GO SUB k: PRI
NT "It bites you..."
3020 LET rs=rs-1: PRINT AT 1,28;
rs: IF rs<1 THEN GO SUB 1050: PR
INT "You die.": GO TO 4230
3030 RETURN

4050 IF y$(y,x)="9" THEN GO SUB
3000
```

```
4230 GO SUB k: FOR f=1 TO 50: BE
EP .01,f: NEXT f: PRINT "Press E
NTER for ressurection": INPUT a$
: LET rt=rt+1: LET z$(y,x)=CHR$
144: LET y$(y,x)=" ": GO TO 4000

7600 PRINT "You must collect as
many gold  nuggets as you can,
and escape  before dawn, in 500
secs."
7610 LET a=5: LET b=5: GO SUB 90
00
7620 LET c=.25: LET j1=151: LET
j2=4: GO SUB 9400

7640 LET tb=500: PRINT AT 10,10;
"Wait a bit more."
7680 FOR f=6 TO 40: FOR g=6 TO 4
0: IF RND<.05 THEN LET y$(f,g)=C
HR$ 151

8900 PRINT '"  ";rt;" resurrect
ions...";-160*rt;"pts.": LET pt=
pt-160*rt

9340 DATA "000,034,028,126,126,0
28,034,000,000,064,192,033,073,0
85,085,034,000,038,109,024,056,1
00,064,032,000,"
9350 LET a$="": FOR f=1 TO 5: RE
AD b$: LET a$=a$+b$: NEXT f: LET
 b$=""
9360 LET h=1: FOR f=144 TO 158:
FOR g=0 TO 7: LET a=VAL a$(h TO
h+2): POKE USR CHR$ f+g,a: LET h
=h+4: NEXT g: NEXT f

9410 IF RND<.05 THEN LET z$(f,g)
="█": NEXT g: NEXT f
9420 IF RND<c THEN LET y$(f,g)="
g": NEXT g: BEEP .01,1: NEXT f
9430 LET y$(f,g)=CHR$ (INT (RND*
j2)+j1): NEXT g: BEEP .01,0: NEX
T f
9440 DIM c$(5,8): LET c$(1)="gol
d bar": LET c$(2)="ladder": LET
c$(3)="exploder": LET c$(4)="foo
dsack"
```

HINTS ON PLAY

The easiest mistake in this version is to step on a rattlesnake and watch while your resistance goes 4 . . . 3 . . . 2 . . .

1 . . .0. If you have to step on a rattlesnake, do get off it quickly. The addition of earthquakes should make you more cautious in your use of exploders.

With 500 seconds, you have plenty of time to move into most parts of the maze, and it is a good idea to try to move in a wide circle, never retracing your path. This means you will come across more objects.

ADDITION 3 gives FANGMOLE TUNNELS 4

Bats and Torches

Gradually, bats have colonized the Fangmole Tunnels. These bats hate light and will attack anyone carrying torches. Without a torch, you can only see two locations in any direction.

TO PLAY

Torches are now strewn around the maze, and when you find a torch your view of the maze increases by one location in each direction. The maximum seeing distance is five locations, when you can also see objects and snakes at a distance of two locations away.

Bats pursue you in order to capture your torches, one at a time. A bat pursues for ten seconds, and emits a high-pitched squeak. If the bat fails to catch you in ten seconds, it will fly off. Bats have a distressing ability to fly through cracks in the walls, but as they have difficulty in turning corners they may just hover a little way from you without attacking.

The new aim now is to get out of the maze via a single exit somewhere to your southeast. You must do this in as short a time as possible. You start with five exploders to help you get through, and you get extra points for each exploder you escape with – including those which you find on the way. The computer chooses randomly between the two possible aims, and alters the size of the map and the objects on it accordingly.

TYPING IN THE LISTING

Type in **Addition 3** and SAVE it.
LOAD **Fangmole Tunnels 3** and MERGE **Addition 3** to get the new program.
SAVE it.

EXPLANATION OF LISTING

1260 You pick up a torch. s is the number of locations that you can see away.
4030 Check if bat attacks.
4300–4390 A bat attacks. The bat is initially placed some distance away from you and its attack path is calculated so that it will follow a straight path without hesitation but be poor at turning corners.
4400–4450 The bat grabs a torch, the top left of the screen is cleared, and redrawn one size smaller. The minimum distance of sight is two locations in each direction, as this corresponds to unaided vision.
7510 Select one of two possible aims.
7620–7650 m2 is the possibility of a bat attacking. j2 is the range of objects, and j1 is the character code of the first object.
7700–7790 Alternative aim. A single exit is in the southeast of the map, which is extended in the east-west direction, and you start in the northwest corner.
8810–8620 Your points are calculated. You get 2 points for every second less than 500 you took to escape (negative if you took longer), and 20 for each exploder you are carrying.
9440 Complete list of objects.

Fangmole Tunnels 4: *Bats and Torches*

LISTING

```
   1 REM  The Fangmole Tunnels
        Addn 3
   2 REM  © 1984 S. Robert Speel

1260 IF jk=155 THEN IF s<5 THEN
LET s=s+1: GO SUB k: PRINT "You
can see further now."

4030 IF RND<m2 THEN GO TO 4300

4300 FOR f=1 TO 10: LET x4=x+INT
 (2+RND*2)*SGN (RND-.5): LET y4=
y+INT (2+RND*2)*SGN (RND-.5): IF
 z$(y4,x4)>CHR$ 147 THEN NEXT f:
 GO TO 4040
4310 GO SUB k: PRINT "A bat sees
 your light & attacks"
4320 FOR f=1 TO 10: BEEP .05,40+
RND*20: NEXT f
4330 FOR e=1 TO 10: LET d$=z$(y4
,x4): LET z$(y4,x4)=CHR$ 158: GO
 SUB 1000: GO SUB 1010
4340 PRINT AT 5,25;ta: INK 0: GO
 SUB 2000

4350 LET z$(y4,x4)=d$
4360 LET y5=y4+(y4<y)-(y4>y)+(RN
D<.1)-(RND<.1): LET x5=x4+(x4<x)
-(x4>x)+(RND<.1)-(RND<.1): IF z$
(y5,x5)<CHR$ 149 THEN LET y4=INT
 y5: LET x4=INT x5: IF x=x4 AND
y=y4 THEN GO TO 4400
4370 BEEP .05,50+e: NEXT e
4380 GO SUB k: PRINT "The bat fi
nally flies off."
4390 LET z$(y4,x4)=d$: GO TO 401
0

4400 GO SUB k: PRINT "The bat gr
abs a torch & flies": GO SUB k:
PRINT "off triumphantly"
4410 LET s=s-1*(s>2): GO SUB 445
0: GO TO 4010

4450 FOR f=0 TO 10: PRINT AT f,0
; PAPER 0;"                ": NEX
T f: GO SUB 1000: GO SUB 1010: R
ETURN

7510 LET aim=INT (RND*2): GO TO
7600+aim*100
```

```
7620 LET c=.25: LET j1=151: LET
j2=5: GO SUB 9400

7650 LET m1=.01: LET m2=.02

7700 PRINT "You must reach the e
xit, to the southeast, as quickl
y as you can with as many explod
ers as you  can."
7710 LET a=4: LET b=7: GO SUB 90
00
7720 LET c=.25: LET j1=152: LET
j2=4: GO SUB 9400
7730 GO SUB 9500: LET exp=5: LET
 tb=1e6

7740 LET m1=.01: LET m2=.01
7750 LET z$(30,51)=CHR$ 147
7760 LET x=9: LET y=9
7790 RETURN

8600 PRINT " You have finally es
caped."''" You took ";ta;" secs.
..";1000-ta*2;" pts.": LET pt=10
00-ta*2
8610 PRINT "You have ";exp;" exp
loders...";exp*20;"pts.": LET pt
=pt+exp*20
8620 GO TO 8900

9440 DIM c$(5,8): LET c$(1)="gol
d bar": LET c$(2)="ladder": LET
c$(3)="exploder": LET c$(4)="foo
dsack": LET c$(5)="torch"
```

HINTS ON PLAY

The best way to avoid bats is by keeping a thick wall between you and them, so they will just flutter in one spot. The game slows down a bit when you see further, because more of the arrays z$ and y$ have to be printed each turn, but the ability to see over long distances gives you a much greater scope for intelligently planning your explorations.

With the new aim, you may find yourself in a cut-off portion in one corner of the maze. You will need to use your exploders to blast your way through to the main section of the maze. Seeing long distances is especially useful in deciding which paths will take you furthest east.

ADDITION 4
gives **FANGMOLE TUNNELS 5** (The Complete System)

Fangmoles Attack

Finally, the fangmoles return to the maze to feed on the rich crop of adventurers who wander in search of treasure. There is no known defence against a fangmole; flight is your only chance. The fangmole has a short attention-span, and often gives up a chase fairly soon. If it does not give up, you are doomed, for a fangmole can dig a tunnel through solid rock as fast as you can run. . . .

TO PLAY

This is the complete Fangmole Tunnels program, and at last you will meet the fangmoles themselves. Fangmoles start to pursue you from a distance away, and you hear them before you see them. A fangmole may give up pursuing you almost immediately, or it can follow you for more than thirty seconds. If you flee up a dead end – tough luck. You may stop to take objects as you flee, but this is not usually worthwhile.

The computer can now select three possible aims for you. The first two are as for the game so far. The new one gives you 800 seconds in an extra-large maze to collect as many gold bars, ladders and exploders as you can lay your hands on.

One other addition is a help command. Pressing the 'H' key gives you a list of all the commands. This is useful when you return to the game after a long absence!

TYPING IN THE LISTING

Type in the listing **Addition 4**, SAVE it and LOAD **Fangmole Tunnels 4**. MERGE **Addition 4** and finally SAVE the complete program.

List the program and see how long it has become – you may be surprised!

EXPLANATION OF LISTING

1600–1610 Help command.
2140 If 'H' key pressed, do help routine.
4020 Checks if fangmole attacks.
4100–4180 A fangmole pursues you. The fangmole gives a warning beep each go which is useful as it starts off-screen. Otherwise you might not notice it before being pounced on! The fangmole routine takes over from the routine at line 4000. Although the time is updated, it is not printed, so that when a fangmole gives up the chase, the time display will jump several seconds while the routine at line 4000comes back into control. The fangmole has a 5 per cent chance of giving up the chase each go, so there is roughly a 40 per cent chance that it will depart within the first ten seconds.
4200–4220 The fangmole gets you. A message is printed, a nasty noise made – the victory cry of the hunting fangmole – and you go straight to resurrection.
7510 There are three possible aims now.
7800–7890 The new aim. A 36-block maze – the largest used – and extra gold is put in. There are less snakes than usual, but there is a good possibility of being attacked by fangmoles and bats.
8400–8710 Your points score for the new aim. You get 40 points per gold bar, 15 points per ladder and 25 points per exploder found.

Fangmole Tunnels 5 (Complete System): *Fangmoles Attack*
LISTING

```
   1 REM  The Fangmole Tunnels
           Addn 4
   2 REM  © 1984 S. Robert Speel

1600 FOR g=1 TO 6: GO SUB 1050:
NEXT g: PRINT AT 16,0;"Commands:
  5,6,7,8 to move"'"o to open a
door, t to take"'"p to put, foll
owed by the first letter of the
```

```
object's name"'"u for up, & i fo
r inven."
1610 GO TO 1000

2140 IF INKEY$="h" THEN GO SUB 1
600

4020 IF RND<m1 THEN GO TO 4100

4100 LET ma=1: GO SUB k: PRINT "
A Fangmole has scented you...":
FOR f=1 TO 5: BEEP .05,10: NEXT
f
4110 LET x3=x+4*SGN (27.5-x): LE
T y3=y+4*SGN (27.5-y)
4120 LET z$(y3,x3)=CHR$ 156: GO
SUB 1000: LET ta=ta+1
4130 GO SUB 1010: PRINT AT 5,25;
ta
4140 INK 0: GO SUB 2000

4150 LET z$(y3,x3)=CHR$ 144
4160 IF RND<.9 THEN LET y3=y3+(y
3<y AND x=x3)-(y3>y AND x=x3): L
ET x3=x3+(x3<x)-(x3>x): IF x=x3
AND y=y3 THEN GO TO 4200
4170 BEEP .01,10: IF RND<.95 THE
N GO TO 4120+40*(RND<.1)
4180 GO SUB k: PRINT "The Fangmo
le tires, and goes.": LET ma=0:
LET a=USR 60000: LET z$(y3,x3)=C
HR$ 144: GO TO 4010

4200 LET z$(y3,x3)=CHR$ 144: LET
 y$(y,x)=CHR$ 156: GO SUB 1000:
GO SUB 1010
4210 GO SUB k: GO SUB k: PRINT A
T 20,0;"The Fangmole springs on
you &  rips you in two."
4220 FOR f=1 TO 5: FOR g=1 TO 10
: BEEP .05,-50: BEEP .05,30-g: N
EXT g: NEXT f

7510 LET aim=INT (RND*3): GO TO
7600+aim*100

7600 PRINT "You must find as man
y objects as you can, and escape
 within 800 secs."
7610 LET a=6: LET b=6: GO SUB 90
00
7620 LET c=.15: LET j1=151: LET
j2=5: GO SUB 9400
7630 GO SUB 9500
```

```
7840 LET tb=800: LET m1=.02: LET
 m2=.02

7850 FOR f=1 TO 10: LET z$(21+RN
D*21,21+RND*21)=CHR$ 147: NEXT f
7860 LET x=21+INT (RND*21): LET
y=21+INT (RND*21): IF z$(y,x)>CH
R$ 148 THEN GO TO 7860
7890 RETURN

8400 PRINT " You did not escape
in time, & you get ";: LET pt=-
100-(ta+tb)*5: PRINT pt;"pts."
8410 PRINT ''"You found ";gold;"
 gold bars...";gold*40;"pts": LE
T pt=pt+gold*40
8420 PRINT 'TAB 5;ld;" ladders..
.";ld*15;"pts": LET pt=pt+ld*15
8430 PRINT 'TAB 5;exp;" exploder
s...";exp*25;"pts": LET pt=pt+ex
p*25
8440 GO TO 8900

8700 PRINT " You have exited the
 Tunnels    within the required
time."
8710 LET pt=0: GO TO 8410
```

HINTS ON PLAY

Fangmoles are easily your greatest threat and it is sensible to know where to flee to if you are suddenly attacked. Having to stop to open doors when trying to run can be fatal. One good point is that fangmoles eat snakes like normal moles eat worms, so the snakes will not bite you if there is a fangmole nearby.

The fangmoles often expose treasures as they dig, and it is worthwhile backtracking along a fangmole's freshly dug tunnel after it has given up the chase.

The new aim of collecting as many objects as you can means that you should be less eager to 'waste' ladders on holes and exploders in order to reach tiny, isolated rooms. As the maze is very large, you will need to keep a rough idea of the direction in which exits are found, so that as the end of the 800 seconds approaches, you can make a dash for a stairway.

Rather than the game just stopping when your time limit expires, you lose 100 points immediately, and 5 points per second thereafter until you reach a stairway. Be very sure not to strand yourself 10 passages away from the nearest exit with only 5 seconds to go!

9. Adding Graphics to Adventure Programs

Adding graphics to a text adventure makes it more enjoyable to play, and they can be used to make winning a special occasion. Graphics may be informative – 'You meet a glitchcreature' does not tell you much about it, but a picture of it may give some clue as to whether you want to attack it or not. Graphics can also enhance numerical data, as in Tribe, where the addition of graphics to show living standards and a graph of progress look better than just lines of figures.

There are three main types of graphics – user-defined graphics, low-resolution graphics (block graphics) and high-resolution graphics. They each have their own particularities and are good for certain types of pictures.

1. User-defined Graphics

UDGs are used as symbols, small pictures, and in large blocks to give a textured appearance. In the Fangmole Tunnels, the picture is made entirely of UDGs, with different symbols for objects, monsters and passages.

When using numerous UDGs, there are simple ways of conserving memory. Never store UDG data as BIN statements, as this is extremely wasteful.

Convert the binary forms of a UDG into decimal – just use PRINT BIN 1Ø111Ø to give its decimal value. Even so, each number in a DATA statement will occupy 5 bytes, not including the commas, and therefore remains expensive in memory. If you increase the length of each number to 3 digits by adding leading zeros where necessary, and adding quotes at the beginning and end of the list, the data becomes a string. This string consists of groups of 4 bytes (3 character digits and a comma) which can be easily manipulated, and there is a substantial saving of memory.

A further saving could be made by removing the commas, but the listing becomes rather difficult to follow and to check for errors, so this is usually avoided.

2. Low-resolution Graphics

Low-res or block graphics have the advantage of being easy to form, easy to colour, and very suitable for landscapes, buildings and backgrounds. The disadvantages are their 'chunky' appearance and lack of fine detail, which makes them unsuitable for drawing monsters. They consume a significant amount of memory – a picture covering half the screen typically occupies between 250–350 bytes.

Low-res pictures are appropriate for use at the end of adventures – as a 'reward' picture when you win. They can be mixed with UDGs or high-res DRAWing. However, having dozens of low-res pictures in one adventure would be costly in memory and lack the fine detail of high-res drawings.

3. High-resolution Drawings

Using high-res drawings, accurate pictures can be drawn of monsters, buildings and objects with all the detail required. It is time-consuming to colour in such drawings, and so I limit myself to outline drawings – it hardly seems worth while to wait three quarters of a minute for a picture to colour itself in for the tenth time in one adventure.

The usual form of a line drawing consists of lines of DATA, which is read in pairs of figures by a FOR-NEXT loop, e.g.,

```
DATA 1,2,3,−2,1,4,0,2,−3,1
      .
      .
      etc.
LET a = 1: FOR f = 1 TO 30: READ x,y: DRAW a * x, a
* y: NEXT f
```

In this example, the variable a is 1 for the normal drawing. If Let a = 2 is used, the drawing is doubled in size, if a = 3, it is tripled, and so on. This is a very useful facility. Note however, that non-integral values of a will not work, e.g., a = 3/2, as the odd values of x and y will be rounded down in length.

The pictures can be moved around on screen simply by changing the initial plotted position where the first line starts. If necessary, using DRAW a * x, b * y with differing values of a and b can elongate or compress a picture in one dimension. Negative values, a = −1, b = −1, give reflected pictures. This versatility of high-res drawings is one of their main attractions.

One problem is the memory space occupied by a drawing. A typical picture of a monster might involve 60–90 vector lines, i.e., 120–180 numbers. If stored as numeric data, this means each picture will take 500 bytes or more! This can be reduced as for UDGs by enclosing the data in quotes. However, it would be useful if a whole drawing could be represented as a string just a few lines long.

For every picture, each line drawn is stored as x vector, y vector. With most pictures, these lines are very short to give the curves and details of the monster, landscape or object being drawn. It would considerably reduce the memory consumption if each pair of coordinates could be represented by a single character. The following routines are a coder – converting DATA x,y pairs into single characters – and a decoder, converting these characters into lines on the screen. The routine has the advantage of being very short, so occupying little memory itself. A typical picture becomes a string of about three lines of characters – about 100 bytes. A more complex picture may be five or six lines of characters, but this still represents a considerable saving of memory.

The coder converts a DATA list starting at line 1000 into characters in a string, n$, which is then printed. The DATA list contains x and y vector pairs which may be numbers between −4 and +5. The restriction is done to keep away from unprintable characters and UDGs. In practice, there are few straight lines longer than this in a drawing of a typical monster, but where necessary, two or more shorter lines have to be drawn.

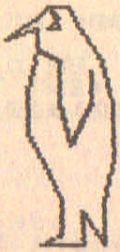
```
8000 REM Graphic Decoder
8001 REM  © S.Robert Speel 1984

8010 RESTORE 2000: READ a,b,c,n$
: PLOT a,b
8020 FOR f=1 TO LEN n$: LET a1=C
ODE n$(f): DRAW (INT (a1/10)-8)*
c,(a1-(INT (a1/10)*10)-4)*c: NEX
T f



9000 REM Graphic Coder
9001 REM  © S.Robert Speel 1984

9010 RESTORE 1000: LET n$="": RE
AD a
9020 FOR f=1 TO a: BEEP .01,1: R
EAD a1,a2: LET n$=n$+CHR$ ((a1+8
)*10+a2+4): NEXT f

9030 PRINT "n$=""";n$;""""
```

My decoder converts the string n$ into a drawing on the screen. The parameters, a, b, c, are respectively the x and y initial plotting position, and the magnification.

As an example, here is some data for a picture of a penguin, shown below (perhaps suitable for the *Island of the Penguins* scenario). 1010 is the uncoded data, and 2000 the coded data. The string which is the fourth piece of data in line 2000 is the one outputted by the coder as n$. The saving in memory is obvious, and is even more than it looks due to changing from numbers to string characters.

```
1000 REM  Penguin data

1010 DATA 34,4,0,0,3,2,-3,0,4,1,
4,0,5,-1,4,-2,3,-1,2,-2,0,1,-2,-
1,0,1,1,-1,1,-1,-1,-3,-2,3,1,0,-
2,2,-1,0,-4,1,-4,2,4,0,3,0,-3,-2
,-4,-1,4,0,4,-2,1,-1,-4,0,-4,2,-
4,1,-2,0,-3,-2,-1
```

```
2000 DATA 100,80,4,"{WeXbYNCL@\J
_KI4sRgPZlWQ<NXAFPd\Q?"
```



```
  n$="|WeXbYNCL@\J_KI4sRgPZlWQ<N
XAFPd\Q?"
```

If you work out a picture, code it and then use a decoder, but do not get your picture back, this will be because you have inadvertently used a vector pair outside the −4 to +5 limit. The character produced by this vector pair will be decoded as a different vector pair, and hence the drawing will be wrong.

When making your own drawing, it is easier to draw the picture out first on blank paper, and then, when it looks right, draw it on suitable scaled graph paper. A fairly rough outlined drawing can be done on paper with ten or twelve lines to the inch, more finely detailed ones on paper with ten lines to the centimetre.

When drawing on graph paper, it is not necessary to try to fit the contours of the picture to the edges of squares – just draw it naturally. When finished, make dots on the picture wherever a drawn line crosses from one square to another. These will be the points you use as your coordinates. Finally, start at an easily recognized dot (typically the snout of a monster drawing) and write down the x, y vectors to get you from point to point on your picture.

These are the data for line 1010 (for a large drawing the data will continue over several lines). Note that the first number in line 1010 should be the total number of (x,y) vectors in the drawing. Run the coder to get n$ for line 2000, and put suitable values at the beginning of line 2000 for a, b and c, e.g., 100, 100, 2. It is worth checking the decoding before wiping out line 1010 and the coder to make sure you have copied n$ correctly, and your vectors are in the −4 to +5 parameters.

The last thing to do is to adjust the first three numbers in line 2000 to get the picture to the desired size and position, and to make sure it does not go off-screen.

Typically, a simple picture such as the penguin shown takes 10–20 minutes to design, draw, input and code, making it easy for you to add line drawings to any adventure.

10. Anarchic System

This system consists of three types of programs. The first is the **Player Program**, which deals with your inputted commands, fighting monsters and describing your locations. The second is the **Scenario**, of which there are four types, each containing the data to set up the map, objects, and creatures for a particular game. By MERGEing a scenario on to the **Player Program**, you get a complete game. The final program is the **Scenario Maker**. This lets you create your own scenarios, which can then be used with the **Player Program**.

The **Player Program** is very long and complex. This is due to the emphasis being on the adaptability of objects. All objects can be used for fighting with, throwing at monsters, digging with and even eating. This allows great freedom to the player in the way in which he tackles the adventure.

Once you have typed in the **Player Program**, the scenarios are very short and quick to type in. Various descriptions are given in uncoded form in the scenarios, as these, being out of context, will not in any way spoil the novelty of the adventure. The important part is mainly in the DATA in lines 8020–8310, which gives very little away except for the names.

The **Scenario Maker** need not be used until you want to write your own scenarios. It asks a series of questions about the places, objects and monsters in your scenario, and ends with a list of the lines which you then have to type in.

ANARCHIC SYSTEM PLAYER PROGRAM

TO PLAY

The Player Program cannot be used on its own, but has to be MERGEd with one of the scenarios which follow or one of your own scenarios. However, once MERGEd, all scenarios will use the same command set.

Go north, south, east and west are as normal and can be abbreviated. On entering a location, you are given its description – e.g., grassland, mountaintop – and a list of the possible directions in which you may go. Sometimes there is a special description for that location, which is also printed, and you can repeat the description of where you are by using the command 'look'. Any objects found there are also listed.

The 'take' and 'put' commands are more complex than usual. A maximum weight and volume of objects is given, so you cannot take an object if it will overload you.

You can look at an object using 'look' followed by the name of that object. This will tell you its height and depth, together with any special description there might be. The command 'open' lets you see what is inside an object, but this only works with a few objects: 'Open box' has a good chance of success, 'open sword' does not. There is no close command since there is no practical reason why you should want to do this.

You can dig using 'dig with [object]', e.g., 'dig with axe'. Some types of location – e.g., rocky ground – are too hard to dig in, but where there is soft ground you can dig in search of buried objects. There are three depths of hole you can dig. A poorly shaped digger will only go a few centimetres down, which will not uncover deeply buried objects. A better instrument, typically a plank or sword, may dig ½ metre deep, and an ideal object, such as a spade, can uncover every buried object in a location, going 1 metre deep. Widely excavating holes is also a good way of knowing where you have been before.

To regain resistance lost while fighting, you can eat small objects. If they are classed as edible, you will recover. Eating objects like spades, small rocks or similar will merely waste them, and some may actually be poisonous. . . .

You have a status command which lists the objects you are carrying, the amount of gold you have, and roughly how strong you are at present.

Fighting occurs whenever you meet a monster: you usually see it at a distance, and as it charges you can throw things at it. In this way you can damage a monster considerably before it gets close enough to retaliate. Once thrown, an object cannot be used again until the monster is dead.

At close quarters, you can use your weapons in two ways – hitting and stabbing. The commands are 'hit with [object]' and 'stab with [object]'. Stabbing works best with long, thin objects, whereas blunt instruments are more effective for hitting. If an object is actually a weapon, it will do more damage than one not designed to be a weapon. Thus, a club will be better than a stick of the same dimension, as it will have been designed to have a grip and proper balance, making it less clumsy to use. Typing 'hit' on its own allows you to punch the creature with your fist.

SUMMARY COMMANDS

go	north, south, west, east	
take	object	
put	object	
look,	look at object	
dig,	dig with object	
eat	object	
status		
when fighting	throw	object
	hit, hit with	object
	stab with	object

TYPING IN THE LISTING

This is a very long listing, and it is suggested that you input it in several stages. SAVE the complete program on tape at least twice.

EXPLANATION OF LISTING

The program consists of numerous subroutines, which are mainly accessed from lines **5000–5290**, although some of them access each other.

1000 Input command, add extra spaces and change to lower-case letters if necessary.

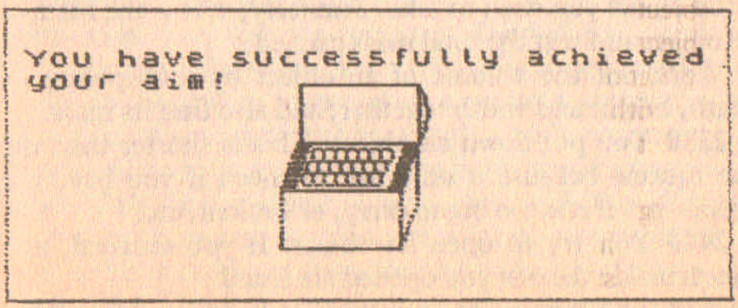
1010–1020 Check for movement in a direction, and if successful print message.

1030–1050 Remove first word from a multiword command, and go to beginning of next word.

1060–1070 Cycle through a$ until the command entered has been deciphered.

1100–1110 Check for object name being used.

1120–1180 Check if object you want to use is in your hand or in your location. b$ = "1" when the object is in the location, and b$ = "2" if you have it. This means, for instance, that you can dig with an object if it is in the same location as you, but in order to fight with it, you must actually possess it.



```
You have successfully achieved
your aim!
```

1200 No correct command found.

1220–1230 Decode length, height or depth of object from a character to a number. These two routines are very useful, allowing the size of an object to vary between 1cm and 12.7m in each dimension, and to be stored in one character. The mass, also stored as one character, can vary between 0.1kg and 127kg. This is done by giving less accuracy to the larger figures, which is reasonable since it does not matter if an object weighs 30kg or 30.2kg, but there is a significant difference if it weighs .1kg or .2kg.

1300–1340 Print where you are, and if any holes are visible.

1350–1360 Print list of directions you may go in.

1370–1380 Print list of objects you can see.

1400–1490 Main command list. Where a command involves an object, a routine to check if the object is available, is first GOSUBed, before going to the 'carry out command' routine.

1500–1520 Check for monsters.

1600–1630 Check for 'sudden death' location, e.g., if you walk over a cliff. Special message printed, and then GOTO end of game routine.

2000–2110 Take routine. Note that most of this routine is concerned with when, for one reason or another, you cannot take an object. This is typical of most such routines – the program has to check for several conditions to be fulfilled before you can do something. In this routine, the actual 'taking' is done in two lines – 2100 and 2110!
tc is the total volume you are carrying, and tu is the volume of the object you wish to take. Similarly, wht is the mass of the object and wgt the total mass carried.

2160 Work out the volume of an object by multiplying its length, height and width together, and also find its mass.

2200–2230 You put down an object. This is shorter than the take routine because it only has to check if you have the object – not if it is too big to carry, or broken, etc.

2300–2440 You try to open an object. If you succeed, any objects inside the one you opened are listed.

2500–2570 You look at an object. If it is open, the routine continues at the 'open object' routine to tell you what is inside.

2600–2750 You dig. If you dig with no object, you are assumed to dig with your hands. Any objects dug up are listed out, and they are 'brought to the surface' – which is important to stop them burying themselves when you put them down elsewhere!

2800–2810 This routine checks if you are using a legitimate fighting command.

2850–2970 You eat something. You cannot eat something too big or heavy. Objects may be poisonous or merely inedible, wasting the object, or edible, restoring your resistance.

3000–4110 The fighting routine.

3000–3010 A monster is there.

3020 Check if it has reached you, or is coming towards you.

3100–3140 Check if you throw an object, or hit or stab a monster.

3300–3350 Check if you can actually hit or stab a monster. You may be using a broken or non-existent weapon, or you may be trying to hit a monster 10 metres away, or not be using an object at all.

3360–3390 You hit a monster. ya = your attack strength, dependent on the attack value of the object and its mass, and a random factor.

3400–3440 Check if a monster has treasure. The GOSUB **7000** lines found here and elsewhere are to check if you have won the game.

3450–3460 If this type of monster is common, resurrect it after death in a new location.

3500–3540 You stab the monster. Good if you attack with a long, thin object, but not necessarily a dedicated weapon. So stabbing with a sword or a pointed spike will have a similar effect while stabbing with a wide object such as an axe will do little good even though the axe is a weapon.

3600–3710 The monster attacks you and the effect of its attack is commented upon. att is the monster's attack value, and this is deducted from your resistance rs if it hits.

3800–3830 Check if you hit. There is a 70 per cent chance of you hitting, and this is increased if you use a dedicated weapon.

3850–3860 The resistance of your attack object is reduced, and if zero, the object breaks.

3900–3940 Check for effect of your attack on monster, and comment.

4000–4110 You throw an object. The object may fall short if too heavy to throw far. Whether it hits depends on the range and a random factor. If it hits, the damage caused is affected by the weight, and if the object is a weapon, it is increased. If you are throwing from right next to the monster, the impact is reduced, as the object will have less momentum – you are almost 'battering' the monster rather than throwing an object at it.

4500–4560 Status.

5000–5290 Main routines, which gosub the rest of the program. This format makes it easy to access routines from different places, and modification of a routine is simpler to control – you do not have to worry about affecting the rest of the program.

6000–6020 You fail in your mission. If your task required several objects and achievements, and you acquired some, then the percentage score will show you are on the right track.

7000–7060 Check if you win, by seeing if you have the money required, the objects required, and that you are in the appropriate place.

7100–7190 You have won. A small picture of a treasure chest is drawn, but this routine can be replaced by the picture of your choice to suit the scenario.

8000–9990 The scenario DATA fits in here. All the variables are read and the arrays filled.

Player Program

LISTING

```
  1 REM  Anarchic System:
           Player
  2 REM  © S.Robert Speel 1984

 10 GO TO 8000
```

```
1000 POKE 23692,255: INPUT "what
 do you do? ";a$: BEEP .1,0: LET
 a=LEN a$: LET a$=a$+"     ": FO
R f=1 TO LEN a$-5: LET a$(f)=CHR
$ (CODE a$(f)+32*(CODE a$(f)<91
AND CODE a$(f)>64)): NEXT f: RET
URN

1010 DATA "north","east","south"
,"west": RESTORE 1010: FOR g=1 T
O 4: READ b$: IF (b$(1)+" "=a$(f
 TO f+1) OR b$=a$(f TO f+LEN b$-
1)) AND u$(g)>"0" THEN LET pl=CO
DE u$(g)-48: PRINT "You go ";b$:
 LET t=1: RETURN
1020 NEXT g: LET t=0: RETURN

1030 LET a$=a$(2 TO ): FOR g=1 T
O LEN a$: IF a$(1)<>" " THEN LET
 a$=a$(2 TO ): NEXT g
1040 FOR g=1 TO LEN a$-5: IF a$(
1)=" " THEN LET a$=a$(2 TO ): NE
XT g
1050 RETURN

1060 FOR f=1 TO LEN a$-5: POKE 2
3692,255: GO SUB 1010: GO SUB 14
00: IF t THEN RETURN
1070 NEXT f: RETURN

1100 GO SUB 1030: FOR f=1 TO LEN
 a$-4: LET k=f: FOR g=1 TO ob: L
ET b$="": IF o$(g, TO 4)=a$(k TO
 k+3) THEN GO TO 1120
1110 NEXT g: NEXT f: LET b$="":
LET n$="12345678901234567890 1██"
: RETURN

1120 LET i=g: LET q$=o$(g): LET
n$=q$: IF q$(24)>"0" THEN LET f=
LEN a$: GO TO 1110
1130 IF q$(22)="1" AND q$(23)=CH
R$ (pl+48) THEN LET b$="2": LET
n$=o$(i): RETURN
1140 IF q$(22)="2" AND q$(23)<>"
0" THEN GO TO 1170
1150 IF q$(23)="0" THEN LET b$="
1": LET n$=o$(i): RETURN
1160 LET f=LEN a$: GO TO 1110

1170 IF o$(CODE q$(23)-48,21)<>"
Y" THEN LET f=LEN a$: GO TO 1110
1180 LET q$=o$(CODE q$(23)-48):
GO TO 1130
```

```
1200 PRINT "You try to do someth
ing, and     fail.": FOR g=1 TO 5
: BEEP .3,20: PAUSE 10: NEXT g:
RETURN

1220 LET a=CODE c$/100: IF a>1.2
7 THEN LET a=a-1.27: LET a=a*10
1230 RETURN

1240 LET a=CODE c$/10: IF a>12.7
 THEN LET a=a-12.7: LET a=a*10
1250 RETURN

1300 LET u$=r$(pl): LET v$=t$(CO
DE u$(5)-48): PRINT '"You are ";
("on" AND v$(13)="1");("in" AND
v$(13)="2");" a ";v$( TO 12)
1310 GO SUB 7000: IF v$(15)>"0"
THEN GO SUB 1600
1320 IF u$(6)>" " THEN PRINT s$(
CODE u$(6)-48)
1330 IF u$(7)>"0" AND u$(7)<"4"
THEN PRINT "Someone has been dig
ging "'("shallow" AND u$(7)="1")
;("50cm deep" AND u$(7)="2");("m
etre deep" AND u$(7)="3");" hole
s here."
1340 RETURN

1350 RESTORE 1010: PRINT '"You m
ay go:": FOR f=1 TO 4: READ b$:
IF u$(f)>"0" THEN PRINT b$
1360 NEXT f: RETURN

1370 FOR f=1 TO ob: IF o$(f,24)=
"0" THEN IF o$(f,22)="1" AND o$(
f,23)=CHR$ (pl+48) THEN PRINT "T
here is a ";o$(f, TO 10): PRINT
("It is broken" AND o$(f,17)<"1"
): IF o$(f,21)<"1" THEN GO SUB 2
400
1380 NEXT f: RETURN

1400 IF a$(f TO f+3)="take" THEN
 GO SUB 1100: GO TO 2000
1410 IF a$(f TO f+2)="put" OR a$
(f TO f+3)="drop" THEN GO SUB 11
00: GO TO 2200
1420 IF a$(f TO f+5)="look  " TH
EN LET t=1: RETURN
1430 IF a$(f TO f+3)="look" THEN
 GO SUB 1100: GO TO 2500
1440 IF a$(f TO f+3)="open" THEN
 GO SUB 1100: GO TO 2300
```

```
1450 IF a$(f TO f+2)="dig" THEN
GO SUB 1100: GO TO 2600
1460 IF a$(f TO f+2)="eat" THEN
GO SUB 1100: GO TO 2850
1470 IF a$(f TO f+3)="stat" THEN
 GO SUB 4500: RETURN
1490 RETURN

1500 LET a=0: IF NOT mo THEN RET
URN
1510 FOR f=1 TO mo: IF m$(f,16)=
CHR$ (pl+48) AND m$(f,17)>"0" TH
EN LET a=1: RETURN
1520 NEXT f: RETURN

1600 IF CODE v$(15)-48<=ob THEN
IF o$(CODE v$(15)-48,22 TO 23)="
10" THEN RETURN
1610 IF v$(16)>"0" THEN PRINT IN
K 3;f$(CODE v$(16)-48)
1620 FOR f=1 TO 30: BEEP .01,10:
 BEEP .01,20: BEEP .02,-f: NEXT
f
1630 GO TO 6000

2000 LET t=2: IF b$="" THEN PRIN
T "You try and take something th
at isn't there.": RETURN
2010 LET tu=1: FOR f=12 TO 14: L
ET c$=n$(f): GO SUB 1220: LET tu
=tu*a: NEXT f
2020 LET c$=n$(15): GO SUB 1240:
 LET wht=a

2030 IF n$(23)="0" THEN PRINT "Y
ou already have the ";n$( TO 10)
: RETURN
2040 IF tu+tc>1 OR wht+wgt>25 OR
 wht>15 OR tu>.8 THEN PRINT "You
 try to take the ";n$( TO 10)
2050 IF wht>15 THEN PRINT "You c
annot carry something this heavy
.": RETURN
2060 IF wht+wgt>25 THEN PRINT "Y
ou cannot carry this without   d
ropping something else.": RETURN

2070 IF tu>.8 THEN PRINT "You ca
nnot carry something this big.":
 RETURN
2080 IF tu+tc>1 THEN PRINT "You
cannot carry something this bulk
y without dropping something els
e.": RETURN
```

```
2090 IF n$(17)<"1" THEN PRINT "Y
ou try with no success to pick u
p the hundreds of pieces of    b
roken ";n$( TO 10): RETURN

2100 PRINT "You take the ";n$( T
O 10): LET tc=tc+tu: LET wgt=wgt
+wht
2110 LET o$(i,22 TO 23)="10": GO
 SUB 7000: RETURN

2150 LET o$(i,22)="1": LET o$(i,
23)=CHR$ (pl+48): RETURN

2160 LET tu=1: FOR f=12 TO 14: L
ET c$=n$(f): GO SUB 1220: LET tu
=tu*a: NEXT f: LET c$=n$(15): GO
 SUB 1240: LET wgt=wgt-a: LET tc
=tc-tu: RETURN

2200 LET t=2: IF b$="" OR n$(23)
<>"0" THEN PRINT "You try to put
 down something  that you do no
t have.": RETURN
2210 PRINT "You put down the ";n
$( TO 10)
2220 GO SUB 2160
2230 GO SUB 2150: RETURN

2300 LET t=2: IF b$="" THEN PRIN
T "You try to open something tha
t  isn't there.": RETURN
2310 IF n$(21)="n" THEN PRINT PA
PER 7;"You try to open the ";n$(
 TO 10)'"but fail..": RETURN
2320 IF n$(21)="Y" THEN PRINT "I
t is already open.": RETURN
2330 PRINT PAPER 7;"You open the
 ";n$( TO 10): LET o$(i,21)="Y"
2340 GO SUB 2400: RETURN

2400 LET a$="": FOR f=1 TO ob: I
F o$(f,22)="2" THEN IF o$(f,23)=
CHR$ (i+48) THEN LET a$=a$+"a "+
o$(f, TO 10)+"
"

2410 NEXT f
2420 IF LEN a$>1 THEN PRINT "Ins
ide is ";a$
2430 IF a$="" THEN PRINT "It is
empty."
2440 PRINT '': RETURN
```

```
2500 LET t=2: IF b$="" THEN PRIN
T "You try to look at something
  that you cannot see.": RETURN

2510 PRINT "You look at the ";n$
( TO 10)''" It is:"
2520 IF n$(17)="Y" THEN PRINT "b
roken into many pieces.": RETURN

2530 DATA "long","wide","thick",
"high": RESTORE 2530: FOR f=12 T
O 14: LET c$=n$(f): GO SUB 1220:
: PRINT TAB 10;a;"m ";: READ a$:
 IF f=14 AND n$(14)>" " THEN REA
D a$
2540 PRINT TAB 16;a$: NEXT f

2550 IF n$(11)>"0" THEN PRINT 'p
$(CODE n$(11)-48)
2560 IF n$(21)="Y" THEN PRINT "I
t is open.": GO SUB 2400
2570 PRINT '': RETURN

2600 LET t=2: IF v$(14)="n" THEN
 PRINT "You try to dig, but the
ground is too hard.": RETURN
2610 IF b$="" THEN PRINT "You di
g with your hands": LET hol=1: G
O TO 2700
2620 PRINT "You dig with your ";
n$( TO 10)
2630 IF n$(17)="0" THEN PRINT "I
t is broken, and you cannot digw
ith it.": RETURN
2640 LET n$(17)=CHR$ (CODE n$(17
)-1): LET o$(i,17)=n$(17): IF n$
(17)="0" THEN PRINT "It breaks..
.": GO SUB 2160: GO SUB 2150: RE
TURN

2650 IF n$(16)="n" THEN PRINT "I
t is useless for digging!": RETU
RN
2660 IF n$(12)>CHR$ 148 OR n$(13
)>"-" OR n$(14)>CHR$ 2 THEN PRIN
T "It is too big to dig with..."
: RETURN
2670 IF n$(15)>"(" THEN PRINT "I
t is too heavy to dig with...";
RETURN
2680 LET hol=2: IF n$(12)<CHR$ 1
0 OR n$(13)<CHR$ 5 OR n$(14)>CHR
$ 2 OR n$(15)>"#" OR n$(15)<CHR$
 2 THEN LET hol=1
```

```
2690 IF n$(12)>"E" AND n$(12)<"▟
" AND n$(13)>CHR$ 7 AND n$(13)<C
HR$ 31 AND n$(14)<CHR$ 3 AND n$(
15)>CHR$ 4 AND n$(15)<" " THEN L
ET hol=3

2700 PRINT "You dig ";("a few cm
" AND hol=1);("1/2 a metre" AND 
hol=2);("a metre" AND hol=3);" d
eep."
2710 FOR f=1 TO 5: FOR g=1 TO 5:
 BEEP .1/g,-20-g*2: NEXT g: FOR 
g=1 TO 20: BEEP .0001*g,g: NEXT 
g: NEXT f
2720 IF r$(pl,7)<CHR$ (hol+48) T
HEN LET r$(pl,7)=CHR$ (hol+48)

2730 FOR f=1 TO ob: IF o$(f,22)=
"1" AND o$(f,23)=CHR$ (pl+48) AN
D o$(f,24)<=STR$ hol AND o$(f,24
)>"0" THEN FOR g=1 TO 10: BEEP .
01,10: BEEP .01,20: NEXT g: PRIN
T "You find a ";o$(f, TO 10): LE
T o$(f,24)="0"
2740 NEXT f
2750 RETURN

2800 LET b$="":: FOR f=1 TO LEN 
a$-4: POKE 23692,255: GO SUB 310
0: IF b$>"" THEN RETURN
2810 NEXT f: RETURN

2850 LET t=2: IF b$="" THEN PRIN
T "You try and eat something tha
t  is not there.": RETURN
2860 LET tu=1: FOR f=12 TO 14: L
ET b$=n$(f): GO SUB 1220: LET tu
=tu*a: NEXT f
2870 IF tu>.02 THEN PRINT "You c
annot eat something that   big."
: RETURN
2880 PRINT "You eat the ";n$( TO
 10): LET o$(i,22 TO 23)="3 ": G
O SUB 2160
2890 IF n$(18)="1" THEN RETURN

2900 IF n$(18)="0" THEN GO TO 29
50
2910 PRINT "You feel sick...it w
as poisonous": FOR f=1 TO 10: BE
EP .1,RND*50-RND*50: NEXT f
2920 LET rs=rs-INT (RND*5)-1: IF
 rs<1 THEN PRINT "You collapse a
nd die.": GO TO 6000
```

```
2930 IF rs<4 THEN PRINT "You fee
l very weak."
2940 RETURN

2950 PRINT "It tastes very good.
"

2960 IF rs<10 THEN LET rs=10: PR
INT "You feel stronger and refre
shed."
2970 RETURN

3000 LET l$=m$(f): LET m1=f: LET
 k$=l$( TO 10): FOR g=10 TO 1 ST
EP -1: IF l$(g)=" " THEN LET k$=
k$( TO g): NEXT g
3010 LET dis=1+INT (RND*3): PRIN
T "There is a ";k$: RETURN

3020 PRINT ("It is "+STR$ (dis*1
0)+"m away from" AND dis>0);("It
 has reached" AND dis<1);" you..
.": RETURN

3030 LET a=1: IF RND<.5 THEN PRI
NT "It has not seen you...": LET
 a=0
3040 RETURN

3100 LET ya=0: LET b$="": IF a$(
f TO f+2)="hit" THEN GO SUB 1100
: GO SUB 3300: GO TO (dis<=0 AND
 t>0)*(3360+10*(b$=""))+(dis>0 O
R t<=0)*3130
3110 IF a$(f TO f+3)="stab" THEN
 GO SUB 1100: GO SUB 3300: GO TO
 (dis<=0 AND t>0)*(3500+10*(b$="
"))+(dis>0 OR t<=0)*3130
3120 IF a$(f TO f+3)="thro" THEN
 GO SUB 1100: GO TO 4000
3130 IF a$>"a" THEN LET b$="": G
O SUB 1100: GO SUB 3330: LET t=-
1: RETURN
3140 RETURN

3300 LET ya=0: LET t=-1: IF dis>
0 AND (b$="" OR n$(23)>"0") THEN
 PRINT "You wave your fist at it
.": RETURN
3310 IF dis>0 THEN PRINT "You wa
ve your ";n$( TO 10)'"  in the a
ir": RETURN
3320 IF b$="" AND a<5 THEN PRINT
 "You punch it.": LET t=1: LET y
a=1+(RND>.9)+(RND>.9): LET rs=rs
```

```
-(RND<.2)*(rs>1): LET n$=".......
....01112ynn00..1": RETURN
3330 IF b$<>"1" THEN PRINT "You
try to use a weapon that youare
not holding.": RETURN
3340 IF n$(17)<"1" THEN PRINT "Y
ou try to use a broken weapon.":
 RETURN
3350 LET t=1: RETURN

3360 PRINT "You try to hit the "
;k$'"with your ";n$( TO 10)
3370 LET c$=n$(15): GO SUB 1240:
 LET ya=a*.6+1+VAL n$(19): IF n$
(16)="n" THEN LET ya=INT (ya/3)
3380 BEEP .2,30: LET ya=INT (RND
*ya/2+ya/2)
3390 RETURN

3400 LET t=3: IF l$(15)>"0" THEN
 PRINT "You find its hoard of ";
CODE l$(15)-48;" crowns."
3410 FOR f=1 TO 10 STEP 3: FOR g
=10 TO 1 STEP -1: BEEP .1,f: BEE
P .05,g: NEXT g: NEXT f
3420 LET cash=cash+CODE l$(15)-4
8
3430 IF v4 THEN IF v4=m1 THEN LE
T v4=-1
3440 GO SUB 7000

3450 LET m$(m1,17)=CHR$ (CODE m$
(m1,17)-1): IF m$(m1,17)>"0" THE
N LET m$(m1,16)=CHR$ (48+INT (RN
D*loc)): RETURN
3460 LET m$(m1,16)="0": RETURN

3500 PRINT "You try to stab the
";k$'"with your ";n$( TO 10)
3510 LET c$=n$(15): GO SUB 1240:
 LET ya=a*.2+1+VAL n$(19): IF n$
(16)="n" THEN LET ya=INT (ya/4)
3520 LET ya=ya-ya/2*(n$(12)<CHR$
 10 OR n$(13)>CHR$ 5 OR n$(14)>C
HR$ 2)+ya/3*(n$(12)>"2" AND n$(1
3)<CHR$ 3 AND n$(14)<CHR$ 3)
3530 BEEP .4,20: LET ya=INT (RND
*ya/2+ya/2)
3540 RETURN

3600 LET dis=dis-1: IF dis>=0 TH
EN PRINT "The ";k$;" approaches.
..": GO SUB 3020: RETURN
```

```
3610 LET att=0: PRINT "The ";k$;
" attacks ";
3620 LET a=RND: IF a<.3 THEN PRI
NT "but you"'"avoid it.": RETURN

3630 IF a<.3+VAL n$(20)/20 THEN
PRINT "but you"'"parry with your
 ";n$( TO 10): RETURN


3640 LET att=INT (RND*VAL l$(11)
)-VAL n$(20)
3650 IF att<1 THEN PRINT "but on
ly"'"bruises you.": RETURN
3660 IF att<3 THEN PRINT "and"'"
wounds you."
3670 IF att>2 THEN PRINT "and"'"
damages you badly."
3680 LET rs=rs-att: IF rs<1 THEN
 PRINT "You die from your wounds
.": GO TO 6000
3690 IF rs<2 THEN PRINT "You are
 very weak indeed.": RETURN
3700 IF rs<5 THEN PRINT "You are
 rather battered.": RETURN
3710 RETURN

3800 IF RND<.3-VAL n$(19)/20 THE
N PRINT "Your bad aim causes you
 to miss.": LET ya=0: RETURN
3810 IF ya<1 THEN PRINT "You mer
ely graze it.": RETURN
3820 IF ya<3 THEN PRINT "It is h
urt.": RETURN
3830 IF ya>2 THEN PRINT "It is g
rieviously wounded.": RETURN

3850 LET o$(i,17)=CHR$ (CODE o$(
i,17)-1): LET n$=o$(i): IF n$(17
)<"1" THEN PRINT "Your ";n$( TO
10);" breaks!"'" You discard it.
": GO SUB 2160: GO SUB 2150
3860 RETURN

3900 LET l$(13)=CHR$ (CODE l$(13
)-ya)
3910 IF l$(13)<="0" THEN PRINT "
You have killed the ";k$: GO TO
3400
3920 IF l$(13)<"3" THEN PRINT "I
t is very weak.": RETURN
3930 IF l$(13)<"5" THEN PRINT "I
t is in bad shape.": RETURN
3940 RETURN
```

```
4000 LET t=-1: IF b$="" THEN PRI
NT "You look for something to th
row.": RETURN
4010 IF n$(23)>"0" THEN PRINT "Y
ou try to throw something that y
ou are not  holding.": RETURN
4020 PRINT "You throw your ";n$(
 TO 10)
4030 FOR f=1 TO 20: BEEP .005*f,
20-f: NEXT f: BEEP .2,-1
4040 GO SUB 2160: GO SUB 2150: L
ET ya=0: IF (dis>0 AND a>10) OR
(dis>1 AND a>5) OR (dis>2 AND a>
2) THEN PRINT "It falls short.":
 GO TO 4080
4050 LET g=.6-ABS (dis/10): IF d
is<1 THEN LET g=.8
4060 IF RND<g THEN PRINT "You hi
t!": GO TO 4100

4070 PRINT "You miss."
4080 LET o$(i,17)=CHR$ (CODE o$(
i,17)-1): IF o$(i,17)<"1" THEN P
RINT "On impact your ";n$( TO 10
)'" breaks up."
4090 RETURN

4100 LET ya=a+(a<1)*5+VAL n$(19)
/2-dis-2*(dis=0): LET ya=INT (RN
D*ya): LET ya=ya+(ya<1)
4110 GO SUB 3820: GO SUB 3900: L
ET t=-1: RETURN

4500 LET t=2: PRINT ''TAB 10; IN
K 7; PAPER 1;"STATUS"'
4510 PRINT TAB 5;"Strength: ";("
above " AND rs>10);("below " AND
 rs<10 AND rs>5);("half" AND (rs
=5 OR rs=4));("normal" AND rs>3)
;("feeble" AND rs<4)
4520 PRINT 'TAB 5;"Objects:";
4530 LET g=0: FOR f=1 TO ob: IF
o$(f,23)="0" THEN PRINT TAB 15;o
$(f, TO 10): LET g=g+1
4540 NEXT f: IF NOT g THEN PRINT
 TAB 15;"none."
4550 PRINT 'TAB 5;"Money ";cash
4560 RETURN

5000 RANDOMIZE : INK 1: PAPER 6:
 BORDER 4: CLS : PRINT z$
5010 INK 2: GO SUB 1300: INK 1
5020 GO SUB 1500: IF a THEN GO T
O 5100
```

```
5030 GO SUB 1370
5040 GO SUB 1350
5050 GO TO 5200

5100 GO SUB 3000
5110 GO SUB 3020
5120 GO SUB 1000
5130 INK 3: GO SUB 2800: INK 1:
IF dis>0 OR t=-1 THEN GO TO 5180
5140 INK 3: GO SUB 3800: INK 1:
IF b$="" THEN GO TO 5160
5150 GO SUB 3850
5160 GO SUB 3900: IF t=3 THEN GO
 TO 5010
5180 GO SUB 3600
5190 GO TO 5120

5200 LET t=0: LET b$="": GO SUB
1000
5210 INK 3: GO SUB 1060: INK 1:
IF t THEN GO TO 5010+190*(t=2)
5290 INK 3: GO SUB 1200: INK 1:
GO TO 5200

6000 GO SUB 7000: PRINT "You ach
ieved ";ys;"% of your"'" mission
."
6010 PRINT " Better luck next ti
me!"
6020 STOP

7000 LET pv=0: IF v1 AND cash>=v
1 THEN LET pv=pv+1
7010 IF v2 THEN FOR g=1 TO v2: L
ET pv=pv+(o$(v(g),23)="0"): NEXT
 g
7020 IF v3 THEN LET pv=pv+(pl=v3
)
7030 IF v4 THEN LET pv=pv+(v4=-1
)
7040 LET ys=INT (pv/(v1+v2+(v3>0
)+(v4=-1))*100)
7050 IF ys=100 THEN GO TO 7100
7060 RETURN

7100 PRINT ''"You have successfu
lly achieved  your aim!"''''''''
'
7110 FOR f=1 TO 5: PLOT 96+f,30:
 DRAW 38,0: DRAW 6,16: DRAW -38,
0: DRAW -6,-16: NEXT f
7120 DRAW -4,0: DRAW 0,-20: DRAW
 42,0: DRAW 0,20: DRAW 6,16: DRA
W 0,-20: DRAW -6,-16
```

```
7130 PLOT 104,46: DRAW 0,20: DRA
W 42,0: DRAW 0,-20: DRAW 0,20,1
7150 FOR f=1 TO 3: FOR g=1 TO 7:
 CIRCLE INK 2;100+4*g+f*2,30+4*f
,2: NEXT g: NEXT f
7190 FOR f=1 TO 5: FOR g=1 TO 10
: BEEP .02,g+f*4: NEXT g: FOR g=
11 TO 2 STEP -1: BEEP .02,g+f*4:
 NEXT g: NEXT f: FOR f=1 TO 10:
BEEP .02,20: NEXT f: STOP
7990 STOP

8000 REM  Scenario
8030 RESTORE 8000: READ loc,ob,m
o,pl,dt,do,dd,v1,v2,v3,v4,wgt,tc
,rs,cash,lt

8100 DIM o$(ob,24): RESTORE 8100
: FOR f=1 TO ob: READ o$(f): NEX
T f: FOR f=1 TO ob: READ a$: FOR
 g=12 TO 15: LET o$(f,g)=CHR$ VA
L a$((g-12)*3+1 TO (g-12)*3+3):
NEXT g: NEXT f
8200 DIM r$(loc,7): RESTORE 8200
: FOR f=1 TO loc: READ r$(f): NE
XT f

8300 IF mo THEN DIM m$(mo,17): R
ESTORE 8300: FOR f=1 TO mo: READ
 m$(f): NEXT f
8400 IF dt THEN RESTORE 8400: DI
M s$(dt,50): FOR f=1 TO dt: READ
 s$(f): NEXT f
8500 IF do THEN RESTORE 8500: DI
M p$(do,50): FOR f=1 TO do: READ
 p$(f): NEXT f
8600 RESTORE 8600: DIM t$(lt,17)
: FOR f=1 TO lt: READ t$(f): NEX
T f

8700 IF dd THEN RESTORE 8700: DI
M f$(dd,50): FOR f=1 TO dd: READ
 f$(f): NEXT f
8800 IF v2 THEN RESTORE 8800: DI
M v(v2): FOR f=1 TO v2: READ v(f
): NEXT f
9000 LET c$=""
9990 GO TO 5000
```

HINTS ON PLAY

Every time an object is used to fight or to dig, its resistance is decreased by 1. When the resistance reaches 0, the object breaks. It is therefore not a good idea to use a spade to dig everywhere in sight, or to waste your best weapon on feeble monsters.

Digging is a useful way of registering where you have been, but do not expect to find objects buried all over the place – clues to where objects are buried are given. Remember, too, that digging shallow holes will not uncover deeply buried objects.

You will probably get killed or break up the vital objects the first few times you play, but do not get discouraged – you will get a little further each time, and learn which objects you must conserve, until you finally succeed.

The per cent success when you get killed off is only a very rough guide – if the only aim is to kill a monster, and you solve all the problems apart from killing it, then you may get 0 per cent. To overcome this, where possible, the absolutely necessary items have been added to the victory conditions. For instance, if a special plank is needed to get over a deep chasm, then possession of this object would be one of the required victory conditions. The only problem here is that you might cross the chasm, put down the plank, and then find the treasure. Should this happen, the computer would not count you as having won, as you do not have the plank! So if you find an object is absolutely essential for one part of the adventure, keep hold of it just in case.

ANARCHIC SYSTEM SCENARIO 1

Valley of the Swampbeast

TO PLAY

You have come to a secret valley where, it is rumoured, lies a great treasure, guarded by a huge monster – the swampbeast.

Your aim is to find the swampbeast, overcome it, and take the treasure. You are somewhat ill-equipped, having mislaid your weapons, but as the valley was once the scene of a great battle, there should be plenty lying around. . . .

To win, you must actually take the treasure – just finding it is not enough. There are only a couple of monsters that you will have to fight and a few simple objects, so this is a fairly simple introductory scenario.

TYPING IN THE LISTING

Type in the **scenario** listing and SAVE it. LOAD the **Player Program** and MERGE the **scenario**. The system is then ready to play. Be careful when typing in the spaces in lines 8110 and 8610. Also note that the < > in line 8210 consists of *two separate characters*.

EXPLANATION OF LISTING

8010 Introduction
8020 DATA for various variables used in the program – where you start, your resistance, the aims, etc.
8110 Object list. The various monsters and letters after the name give the dimensions of the object, whether it is rigid or not, if it is a weapon, if it is edible, and where it is.
8120 Location list. The characters give the location in each direction, what type the location is, and any special message numbers.
8310 Monster list.
8410 Description list. Each description may be given at several locations.
8510 Description of objects.
8610 Types of terrain, with details of whether it can be dug, and whether you will be killed if you go there without the correct object – e.g., going over a cliff without a rope.
8810 Relates to the victory conditions.

Scenario 1: *Valley of the Swampbeast*
LISTING

```
8000 REM          Scenario:
     Valley of the Swampbeast
8001 REM  © S.Robert Speel 1984

8010 LET z$="Valley of the Swamp
beast        There are two legen
ds about the valley which you ha
ve entered.  One is that a great
 treasure is hidden there. The o
ther is that a huge monster - th
e Swampbeast - guards it. Can yo
u overcome   the Swampbeast and
take the     treasure?"
8020 DATA 25,8,2,1,6,3,1,0,3,9,1
,0,0,10,0,3

8110 DATA "treasure  00000yE100n
193","spade      00000y6100n160",
"cudgel     00000y5140n120","raft
      10000y:100n1;0","spear
 20000y:192n1H0","plank      0000
@yD142n1@0","sword      30000y116
4n1E0","marrow     00000y2000n110
"

8120 DATA "050020020020","120008
001013","142020030025","14705001
0030","147005002020","1450200100
35","100002001012","020010010010
"

8210 DATA "4@B2110","31C0160","0
420130","57131 0","0640230","7=?
51 0","0860330","09=7130","0:<81
30","00;9350",":0I<150","9;G=320
","8<>63 0","=GE?320","6>@4320",
"?EA11 0","@D0B140","1A0C140","2
B00160","E00A250",">FD@2 0","GH0
E240","<IF>1 0","I00F250",";0HG2
50"

8310 DATA "Swampbeast75D 591","G
rizzly    52< 5H3"

8410 DATA "All about you are pla
ins.","You have to use your raft
 here.","To the north are high m
ountains.","Southwards is a high
 ridge","There are mountains to
```

```
the east","There is a cliff to t
he west."

8510 DATA "It looks sturdy yet l
ightweight.","It is sharp, and l
ooks vicious","It is rusty and b
lunt."

8610 DATA "grassy plain1y0","pla
teau      1n0","swamp        2y41"
8710 DATA "You slowly sink into
the mud and drown."
8810 DATA 1,2,4
9990 GO TO 5000
```

HINTS ON PLAY

You may find it difficult to locate the swampbeast at first, and you may easily get killed by going into the wrong places unprepared. It is important to draw a map as you go.

ANARCHIC SYSTEM SCENARIO 2

Hoard of the Exile

TO PLAY

You have translated some ancient documents, which reveal that an exiled king from a small country fled into the forests with his hoard of treasure. He left word in these documents that he intended to bury his treasure by a conspicuous landmark – a deep gorge – to be recovered by his relations at some later date. Neither king, hoard, nor relations were ever seen again. . . .

You have discovered a likely looking gorge from your aeroplane and, as there is nowhere for it to land, you parachute down – intending to find the hoard and rejoin your pilot when he can borrow a helicopter. Your objective is to dig up and take the hoard. Note that in the excitement of your parachute jump, you left your metal detector and digging tools in the plane.

There are no monsters in this scenario, so all you have to worry about is digging up the hoard of the exiled king, wherever it is. . . .

TYPING IN THE LISTING

As for the previous scenario, be careful with the spaces in the DATA lines. Again, this is MERGEd on to the **Player Program**. If you have microdrives you can save the **Player Program** and all the unMERGEd scenarios on to one cartridge, and use LOAD*"m";1;"Player":MERGE*"m";1;"Scenario name".

NOTES ON LISTING

It is worth noting that the **Player Program** can cope with there being no monster DATA – line 831Ø, as long as the DATA in line 8Ø2Ø is correctly altered.

Scenario 2: *Hoard of the Exile*

LISTING

```
8ØØØ REM         Scenario:
           Hoard of the Exile
8ØØ1 REM  © S.Robert Speel 1984

8Ø1Ø LET z$="     Hoard of the Ex
ile             You have parachute
d into a       clearing of a large
 forest,       where in the distan
t past, an     exiled king fled wi
th his goldenhoard, and was neve
r seen again. You have learned t
hat he had    intended to bury it
 by the side of a deep gorge. Yo
ur aim is to find the hoard."

8Ø2Ø DATA 3Ø,7,Ø,28,5,3,1,Ø,2,Ø,
Ø,Ø,Ø,1Ø,Ø,5
```

```
8110 DATA "shovel     10000y8100n
270","spade      20000y2100n1G0",
"hoard      00000yE100n240","ches
t     00000y5101y1C3","box
 00000yG111y1H1","plank     0000
0y3123n1<0","sack      30000n120
0y112"

8120 DATA "120009001015","080004
001004","020030010030","10005005
0020","147100020001","1410150020
20","030030030001"

8210 DATA "6200411","730112","84
0222","950313",":0045",";71011",
"<82615","=9371",">:4813","?0595
","@<6011","A=7;1","B>8<2","C?9=
23","D0:>5","EA;011","F8<@2","GC
=A2","HD>B23","I0?C5","JF@031","
KGAE3","LHBF1","MICG23","N0DH51"
,"0KE014","0LFJ14","0MGK44","0NH
L13","00IM5"

8410 DATA "To the west are inpen
etrable    thorn trees.","To the
 south is a lake","There is a de
ep gorge to your   east.","An un
climbable cliff is to the  north
.","All around you is forest."

8510 DATA "It is strong and stur
dy.","It is a little on the smal
l side","It is old and dirty."

8610 DATA "dark forest 2y0","ope
n heath  1y0","grassy hill 1y0",
"clearing     1y0","deep gorge  2
nz1"
8710 DATA "You fall down to your
 doom on   the rocks below."
8810 DATA 1,3
9000 LET c$=""
9990 GO TO 5000
```

HINTS ON PLAY

As there are no monsters, you do not need to worry about finding anything to eat. You may be surprised at one of the descriptions when you enter a particular location, but this is not a bug in the program! Remember that objects may be buried at more than one depth.

ANARCHIC SYSTEM SCENARIO 3

Island of the Penguins

TO PLAY

The only survivor of a shipwreck in the far south of the Pacific Ocean, you are swept on to the shore of a snow-covered island. Looking around you can see to the northeast one of the ship's lifeboats tossed up on a rock. Water and slippery icefloes lie between you and the boat, but you think you could scramble from floe to floe to the boat. A slip into the freezing, choppy sea would be fatal.

To your west is the main part of the island, and at least the ground is firm there. Apart from some large penguins staring hungrily at you, and the distant howl of a wolf, the island looks more inviting than the icefloes for the time being. Your aim is to get hold of a paddle somehow, and get yourself to the lifeboat.

TYPING IN THE LISTING

There are spaces in lines 8110, 8310, and 8510 which are important. SAVE this scenario along with the others, and MERGE as before with **Player Program** to use.

EXPLANATION OF LISTING

There are a larger number of descriptions and location types than in the previous scenarios, and also a longer list of 'sudden death' messages.

Scenario 3: *Island of the Penguins*
LISTING

```
8000 REM  Scenario:
   Island of the Penguins
8001 REM  © S.Robert Speel 1984
```

```
8010 LET z$=" Island of the Peng
uins           After a terrific
storm, during which your ship si
nks, you are washed ashore on a
barren, snow-covered island. To
the northwest you see a boat on
the rocks.    There is water in
the way, but you think that by s
crambling on the icefloes, you c
an reach the boat. Your task is
 to find a  paddle and get to t
he boat."

8020 DATA 25,9,5,16,9,2,4,0,1,23
,0,0,0,10,0,11

8110 DATA "paddle      00000y:100n
180","stick       10000y2144n130",
"axe        00000y:140n130","icep
ick   00000y2131n110","rock
 00000yX100n1;0","sledge      0000
0y5100n160","harpoon     00000y<17
0n180","bread        20000y1200n110
","egg          00000n1000n140"

8120 DATA "15803001 0030","147010
010030","14002000102 0","10004000
2016","02002000200 80","149050015 0
70","15903000305 0","030010010001
","006004004001"

8210 DATA "20007","34109","05208
","5<025","6;439","7050:7","9000
48","H9009","E:789","H>H934","0@
<59",";0=46","<0006","HI?:3",">D
H026","0A0;6","BHH@23","HCAH32",
"HHHH31","HFH?24","H09H62","IG0D
5","HHHF;","00001","H0F>35"

8310 DATA "penguin   426 045","p
enguin    537 063","penguin    338
 0D1","wolf           748 093","seali
on    55< 0I6"

8410 DATA "Your icefloe is drift
ing out to sea...","The sea is t
o west and north.","There is wat
er to east and south","The sea i
s to your north and    south.","
There is deep water to the north
","The sea lies to the east","A
steep slope goes down to the  no
rth.","The slope to your south i
s too  steep to climb.","The boa
t is to your east."
```

```
8510 DATA "It is long and heavy"
,"It is green and powdery."
8610 DATA CHR$ 8+"float        4 A
1","icesheet      1y42","icefloe
   1y43","steep slope 1n64","roc
ky crag  1n0","rocky shore 1n0",
"igloo         2y0","pine forest 2
y0","rocky plain 1n0","small hut
   2n0","boat           2n0"

8710 DATA "You drown in the icy
water.","You lose your footing a
nd slide to your doom.","You sli
p, fall into the sea   and drown
.","You fall down the slope and
     break your neck."
8810 DATA 1
```

HINTS ON PLAY

The footing is treacherous, and sudden death may come without warning. For this reason it is more important than ever to keep an accurate map. In your battered condition, even a penguin is not afraid to tackle you, so finding a weapon should be one of your priorities. Next you should look for an aid to crossing the icefloes.

ANARCHIC SYSTEM SCENARIO 4

Labyrinth of the Minotaur

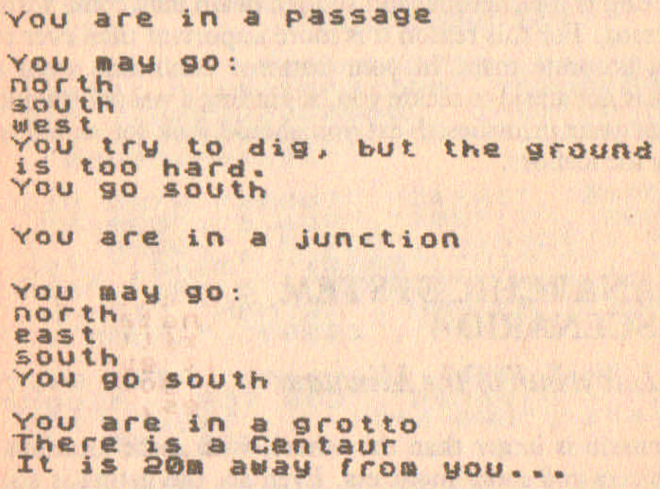
This scenario is larger than the others, with more locations, more objects and more monsters. Even so, the listing is still comparatively short, and there is room for much bigger scenarios than this.

TO PLAY

The minotaur lives in its traditional home – a labyrinth – guarding a huge diamond. You have gathered a small band of followers and arrive at the entrance of the maze. Your followers have decided not to go in with you, and are grumbling that you had better find them some gold to pay for their services up until now. So you venture into the tunnels, on your own, to hunt the minotaur.

The minotaur leads a good social life, and you are likely to meet some of its friends, but not yours – e.g., a centaur, toadbeast and hydra. The labyrinth is also reputed to be the last home of the nearly extinct dodo. Note that one monster you may meet – the drago – is so-called because it is not quite as tough as a small dragon.

Your aim is to kill the minotaur, take its diamond and escape with some extra gold for your followers.



```
You are in a passage

You may go:
north
south
west
You try to dig, but the ground
is too hard.
You go south

You are in a junction

You may go:
north
east
south
You go south

You are in a grotto
There is a Centaur
It is 20m away from you...
```

TYPING IN THE LISTING

Type in the listing and SAVE it. MERGE with the **Player Program** to use as normal. Some of the lines in the listing are very long, and you may make mistakes when typing them in. One useful check which detects many errors is to check down the right-hand column of your typed in program against the listing in this book. So for example, in line 811Ø the check would read n,h1ØØØhnØ41"o. This way you can spot where odd characters have been left out, leaving only those actually mistyped, which are usually much fewer.

EXPLANATION OF LISTING

The listing is made longer mainly by the lines 811Ø (objects) and 821Ø (locations). You may have noticed the regular patterns in the data lists, due to all the strings being the same length. This pattern breakdown in line 821Ø occurs because the last character in each piece of data is actually a normal space, so does not need to be included – but it is put there anyway. In the positions where there are special characters, the strings are longer and so the symmetry is lost.

Scenario 4: *Labyrinth of the Minotaur*
LISTING

```
8ØØØ REM Labyrinth of the
         Minotaur
8ØØ1 REM  © S.Robert Speel 1984

8Ø1Ø LET z$="  Labyrinth of the
 Minotaur    You have entered t
he Labyrinth of the Minotaur, wh
ich you have sworn to kill, to g
et the huge  diamond it guards.
You must alsofind some gold to g
ive to your  accomplices, who aw
ait you      outside."
8Ø2Ø DATA 56,17,14,1,4,4,2,5Ø,1,
56,1,Ø,Ø,1Ø,Ø,9

811Ø DATA "dagger      ØØØØØy6142n
13Ø","axe         ØØØØØy6151n15Ø",
"melon       ØØØØØn12ØØn1:Ø","mush
room  ØØØØØn12ØØn1;Ø","toadstool
 ØØØØØn12ØØn1;Ø","casket      ØØØØ
Øy11ØØy1:Ø","club         1ØØØØyb1Ø
Øn26Ø","loaf        ØØØØØn2ØØØn1ØØ
","boot        ØØØØØy21ØØ 26Ø","ch
est      ØØØØØy21ØØy1↑Ø","javelin
```

```
   20000y2190n1I0","scythe    00
000y6153n1d1","paddle    30000y4
100n1e0","egg          00000n2200n1
=0","diamond    40000yE100n2:0","
sword       00000y<154n1g0","poiso
ndart00000y2290n1_0"

8120 DATA "03000600 1001","100012
001050","0100100 10001","00600500
5001","00800501 0001","1001000501
62","142020002 0060","0200090009001
","0200080007001","147100100100",
"157003003020","141045001016","1
47005009035","010006006001","020
010010004","120010001032","01000
1001001"

8210 DATA "h02011","10431","0200
2","2g504","40608","507L4","6080
1","709;4","80:01","90005","080<
5","0;0=1","0<0>5","0=0?1","@>00
1","A0?08","C0@B4","eA001","D0A0
12","0EC06","0F0D13","£aGE3","F0
H01","GJI04","H0005","0K0H1","00
0J2","060M1","OLN04","M0008","P0
M094","RO0094","00dP94","50P094"
,"T0RX4","U0S01","0WTU4","0U002"
,"00fU1","050Y1","ZX\[3","00Y05"
,"0Y002","Y]£04","0↑0\1","0_0]5"
,"000↑8","\0F01","0b0F1","00ca1"
,"b0002","Q00094","00B02","W0002
","00042","hh1h7"

8310 DATA "Minotaur  84@ 0↑1","S
pirit     72< DU1","Hydra       548
 :Z1","Demon        526 0\2","Fishm
an    31< 0D1","Centipede 822 0R3
","Toadbeast 51< 6d1","Serpent
 728 8I1","Centaur    62= 151","P
lantman   75: :L1","Cyclops    52?
 4:1","Dodo         319 0=9","Trogl
odyte507 304","Drago      54: 199
"

8410 DATA "The exit is to the no
rth.","To your north is a pool."
,"To your west lies a pool.","Th
e ground is muddy here."

8510 DATA "It looks rather unwie
ldly.","It is sharp and made of
glass.","It looks unused.","It l
ooks very valuable."
```

```
8610 DATA "passage      2n0","dea
d end     2n0","four-way     2n0",
"junction     2n0","cavern      2
n0","acid lake   2n91","the open
 air2y72","grotto       2n0","pas
sage      2y0"

8700 IF dd THEN RESTORE 8700: DI
M f$(dd,52): FOR f=1 TO dd: READ
 f$(f): NEXT f
8710 DATA "As you wade in, your
shoes       dissolve, and you per
ish.","Your accomplices, seeing
you    without the diamond, kill
 you."

8810 DATA 15
```

HINTS ON PLAY

This is another scenario where your map is very important. There are several subtle traps which may not kill you outright, but make your life more difficult. Try to see if you actually need to take certain diversions, or if the rewards they bring are worth the trouble. The minotaur is very strong, and you will need good weapons to kill it. These weapons tend to get used up quickly, so do not waste them on weaker creatures or they may break when you really need them.

Once you have killed the minotaur, it is probably advantageous to carry on, as you will need more gold for your followers and there are more ways than one to the exit.

ANARCHIC SYSTEM SCENARIO MAKER

This program allows you to input your own scenario.

First you design the map, and invent the various monsters and objects you want to include. Then RUN **Scenario Maker** and you will be asked a series of questions: where the locations are in relationship to one another, where the monsters and objects are, the characteristics of the objects, and so on. When you have answered these questions, the computer will display a list of the lines which you will have to type in. This list will be longer than one screenful, and the computer stops with the 'scroll?' message. At this point you can write down the lines, or BREAK and COPY if you have a printer, before typing CONT to get the next screen. If you COPY the whole output, you will see that it corresponds to the listings for the various scenarios, as this was the program I used to design them!

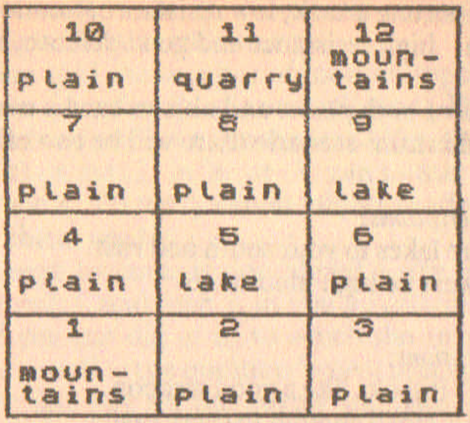
TO USE

This is best shown by an example. Before RUNning the program, the scenario has to be designed. First the map must be drawn, and this consists of locations arranged in any way you like. It is easiest to make a simple grid of locations, which is what will be done for the example. The map is just twelve locations, arranged as opposite.

Note the different terrain types, and the locations numbered from the bottom left-hand corner, line by line upwards. This is important when using grid-type locations.

To make the map less 'square', the lower left and top right corners are removed – this is done by making them impassable locations. You need to note down the different location types, which in this case are (1) mountains (impassable), (2) plain, (3) quarry, and (4) lake.

Next, the objects and monsters need to be planned out. The aim of this mini-scenario will be to find a fossil which is buried



Map for Mini-Scenario

in the quarry. To get this fossil, a pickaxe is needed, and this will be in location 3. The lakes block the way between the quarry and the pickaxe, and there is a boat in location 4 for crossing the water. A spade in location 6 is a decoy – it will not dig deep enough to get the fossil. In the quarry itself there is a crowbar, and this can be used as a weapon.

A list can be made like this:

Objects:

(1) Fossil	buried 1 metre deep, location 11.
(2) Boat	location 4, low resistance.
(3) Pickaxe	location 3, can dig 1 metre deep, low resistance.
(4) Spade	location 6, low resistance, can dig ½ metre deep.
(5) Crowbar	location 11, high resistance, weapon.

The monsters come next. There will be two of these, in locations 2 and 6, so that the player will have to fight at least one of them. As the setting is fairly 'modern', present-day creatures will be used. Again, a list is made:

Monsters:
(1) Jaguar strong attack, low resistance, location 2.
(2) Cayman high resistance and good defence, location 6.

Descriptions of both places and objects can be made, and for the purpose of this mini-scenario there will be two of each.

Location descriptions:
(1) There are lakes to your south and east.
(2) To the west lie high mountains.

Object descriptions:
(1) (Pickaxe) It looks like a good weapon.
(2) (Boat) It is small enough to carry easily.

The player must therefore do the following: get the crowbar from the quarry, find the boat, cross the lake and kill either the jaguar or the cayman, collect the pickaxe, return to the quarry and dig up the fossil.

Note that there are several ploys to mislead the player: the spade seems the ideal object to dig for the fossil, but is actually no good; the pickaxe is described as a weapon, but if broken in a fight, the game cannot then be won; the boat will also break if used in a fight, leaving the player stranded. Players do not like retracing their steps, and the chances are that, after killing one monster and finding the pickaxe, the player will then move on to the other monster which may finish him off. The other details of the scenario can be thought up when the computer asks for them, so it is now time to run **Scenario Maker**.

First you are asked for the name of the scenario and a short introduction. Next you are asked 'How many sudden death location types?' This refers to the type of locations that kill you when you enter them, unless you have a particular object – the type that abound in *Island of the Penguins*. In this scenario the lakes cannot be entered unless you have the boat, so type '1' to answer the question. You will then be asked for a special message – printed up if you enter that type of location without the necessary object. A suitable message would be 'You cannot swim, and quickly drown'.

Next you are told the number of location types – 4 in this case – and then for a few details on each type. The computer prints 'type 1: ' and you enter 'mountain'. You are then asked 'Do you go (1) on it? (2) in it? or (3) can't enter it?' For the mountains, type 3. To the question 'object needed to enter?', type Ø for no object needed.

Then the next location is considered – the plain. Enter 'plain' for 'type 2: ', and then 1, as you go on a plain. You are then asked if you can dig in it, to which the answer is y, for yes. Again, answer Ø to the question 'object needed to enter'.

Continue in this way for the quarry and lake. For the lake location, there is an object needed to enter – object 2, the boat. Hence, type 2 to the question 'object needed to enter'. The last message number you are asked for will need the answer 1, as this was the only last message you used.

After this, the map is designed. First you are asked 'How many descriptions?' In this case the answer is two, and these are then asked for. Input these as in the list of location descriptions given earlier.

The next question is '1) grid? or 2) non-grid locations?' Your answer to this depends on how you designed the map. If it is a maze, or has irregular locations, type 2 for non-grid. You will then be asked for the location number of all the locations surrounding location 1, then for location 2 and so on. Some questions are missed out – for instance, if you tell the computer that north of location 3 is location 4 it will not ask you what is south of location 4, as it already has this information.

Putting in all this data is laborious, so for simple maps (as in the mini-scenario) the grid option is chosen. You input 1 to the question 'Grid, or no grid, locations?' and will then be asked how many locs east/west?' and 'how many north/south?' to which you answer 4 and 3 respectively.

The computer will then work out all the location numbers and orientations by itself. Note that impenetrable locations are used to get rid of the rectangularity of the map.

You are now asked questions about each location. For location 1 input 1 for 'what type is location 1?' As this is inpenetrable mountains, you then go immediately to location 2. This is type 2 (plain), and to the question 'Special description?' answer n. Continue in this way for all twelve locations. Locations 4, 7 and 10 have special descriptions. On typing y to the appropriate question, you are asked for the description number, which is 2 for 'To the west lie high mountains'. Location 8 will have description 1.

That ends the mapping and the objects are next. These are fairly easy to input, and you provide suitable dimensions and answers to other questions as you come to them. One special point: to dig a full metre deep, an object must be between about 0.8 and 1.2 metres long, 0.08–0.30 metre wide, 0.01–0.02 metre in depth, and weigh not more than 1kg or so, and not less than 0.4kg. Make sure that the pickaxe conforms to this and that the spade doesn't! Suitable answers to the object questions could be:

Fossil	n, 1, .2, .04, 3, y, 1, 1, Ø, Ø, n, 1, 11, 3
Boat	y. "It is small enough to carry easily", 1.5, .5, .2, 3, y, 2, 1, Ø, Ø, n,1, 4, Ø
Pickaxe	"It looks like a good weapon", 1, .08, .01, 1, y, 3, 1, 5, Ø, n, 1, 3, Ø
Spade	n, 1, .1, .04, .2, y, 3, 1, Ø, Ø, n, 1, 6, Ø
Crowbar	n, .5, .05, .05, 2, y, 12, 1, 5, 2, 1, 11, Ø

After the objects have been inputted, there are the monsters. Each of these has an attack value, a defence value and a resistance. For the jaguar, these could be 7, 2 and 8, and 5, 4, 6 for the cayman. Both have zero treasure, and their locations are 2 and 6 respectively. To the question 'how many are there?' input 1 in each case. If higher figures are put, a new one appears when a monster is killed, in a random location. This is the technique used to get so many penquins in 'Island of the Penguins'.

To the question 'Where do you start?', input 1Ø, the top left-hand corner of the map. The victory conditions are the last things to input. For 'how many crowns needed?' enter 0. There are two objects to be found – the fossil and the pickaxe – which are object number 3 and number 1. The location to be found is 11, and as you do not have to kill any specific monster, enter zero for 'monster to be killed?'.

The correct lines for you to type in will then appear on the screen, and these can be COPYed on the printer, jotted down by hand, or typed in directly. They can then be MERGEd with the **Player Program** exactly as with the four previous scenarios in this chapter.

The above routine may look a bit complicated, but if you RUN the program and follow it through step by step it is very simple. After you successfully input this mini-scenario, you will be ready to write your own scenarios.

EXPLANATION OF LISTING

This program is not at all complicated, merely assigning a long list of variables and forming various string arrays, and then printing out a list of the lines you must type in.

100–480 Asks questions and sets up scenario variables and strings accordingly. Lines 180–190, which set up the location orientations for a grid, are worth noting. This use of variables as DATA can be useful on many occasions. Lines 210–220 are for non-grid locations. Note that when you input the location, say to the east of location 1, the computer also puts location 1 to the west of the second location. This means that you are asked the minimum of questions.

500–640 These lines print out a simulated listing of the actual lines that you must type in. This is the reason for so many quotes, and the COPYed screen when you type it in should look just like the real listing for easy checking.

Scenario Maker

LISTING

```
   1 REM   Scenario Maker
   2 REM  © S.Robert Speel 1984

 100 INPUT "Name of adventure? "
;z$: FOR f=LEN z$ TO 31: LET z$=
z$+" ": NEXT f: INPUT "Introduct
ion? ";a$: LET z$=z$+a$

 120 INPUT "How many sudden deat
h location  types?";dd: IF dd TH
EN DIM f$(dd,50): FOR f=1 TO dd:
 INPUT "Special message ";(f);"?
 ";f$(f): NEXT f
 130 INPUT "How many location ty
pes? ";lt: DIM t$(lt,16): FOR f=
1 TO lt: INPUT ("type ";f;": ");
t$(f)'"Do you go 1)on it, 2)in i
t or   3)can't enter it? (type 1
, 2 or 3) ";a: LET t$(f,13)=STR$
 a: IF a<3 THEN INPUT "Can you d
ig in it? (y/n) ";t$(f,14)
 140 INPUT "Object needed to ent
er? (0 if  none) ";a: LET t$(f,
15)=CHR$ (a+48): IF a THEN INPUT
 "last message no? ";a: LET t$(f
,16)=CHR$ (a+48)

 150 NEXT f: INPUT "How many des
criptions? ";dt: IF dt THEN DIM
s$(dt,50): FOR f=1 TO dt: INPUT
("Description ";f;"? ");s$(f): N
EXT f

 160 INPUT "1)grid or 2)non-grid
 locations? ";a: IF a=2 THEN GO
TO 200

 170 INPUT "How many locs. east/
west? ";a'"how many north/south?
 ";b: IF a*b>200 THEN BEEP 1,10:
 BEEP 1,10: GO TO 170
 180 DATA g*a+f-200*(g=b),(g-1)*
a+f+1-200*(f=a),(g-2)*a+f-200*(g
=1),(g-1)*a+f-1-200*(f=1): DIM r
$(a*b,6): FOR f=1 TO a: FOR g=1
TO b: RESTORE 150: FOR h=1 TO 4:
 READ c: IF c<1 THEN LET c=0
 190 LET r$((g-1)*a+f,h)=CHR$ (c
+48): NEXT h: NEXT g: NEXT f: LE
T loc=a*b: GO TO 250
```

```
 200 INPUT "How many locations?
";loc: DIM r$(loc,8): FOR f=1 TO
 loc: LET r$(f)="████0 0": NEXT
f
 210 DATA "north","east","south"
,"west": FOR f=1 TO loc: RESTORE
 210: FOR g=1 TO 4: READ a$: IF
r$(f,g)="█" THEN INPUT "Which lo
c. to ";(a$;" of place ";f);"? "
;a: LET r$(f,g)=CHR$ (a+48): IF
a AND a<>f THEN LET r$(a,g+2-4*(
g>2))=CHR$ (f+48)
 220 NEXT g: NEXT f

 250 FOR f=1 TO loc: INPUT "What
 type is loc. ";(f);"? ";g: LET
r$(f,5)=CHR$ (g+48): IF t$(g,13)
="3" THEN GO TO 270
 260 IF dt THEN INPUT "Special d
escription? (y/n) ";a$: IF a$="y
" THEN INPUT "Which number? (1-"
;(dm);") ";g: IF g<=dt THEN LET
r$(f,6)=CHR$ (g+48)
 270 NEXT f

 300 INPUT "How many objects? ";
ob: DIM o$(ob,24): LET a=1: INPU
T "How many object descriptions?
 ";do: IF do THEN DIM p$(do,50)
 310 FOR f=1 TO ob: DIM m$(25):
INPUT "Name of object ";(f);"? "
;m$( TO 10): LET m$(11)="0": IF
a<=do THEN INPUT "Description? (
y/n) ";a$: IF a$="y" THEN INPUT
"Type in description: ";p$(a): L
ET m$(11)=CHR$ (a+48): LET a=a+1
 320 DATA "Length","Width","Dept
h"
 330 RESTORE 320: FOR g=12 TO 14
: READ a$: INPUT (a$);"? (.01 to
 12.7m) ";b: LET b=b+(b>1.27)*(b
/10+1.27-b): LET m$(g)=CHR$ (b*1
00): NEXT g
 340 INPUT "Weight? (.1 to 127kg
) ";b: LET b=b+(b>12.7)*(b/10+12
.7-b): LET m$(15)=CHR$ (b*10)

 350 INPUT "Is it rigid? (y/n) "
;m$(16)'"Resistance? (0-100) ";b
: LET m$(17)=CHR$ (b+48): INPUT
"Is it 0)edible, 1)inedible or
 2)poisonous? (type 0, 1 or 2) "
;m$(18): IF m$(18)="0" THEN LET
m$(19 TO 21)="00n": GO TO 370
```

```
 360 INPUT "Attack value? (0-9)
";m$(19)'"Defence value? (0-9) "
;m$(20): LET m$(21)="n": IF m$(1
9 TO 20)="00" THEN INPUT "Can th
ings go in it? (y/n) ";m$(21)
 370 INPUT "Is it in 1)a place,
or 2)an       object? (type 1 or 2
) ";m$(22)'("What ";("place" AND
 m$(22)="1");("object" AND m$(22
)="2");" is it in? ");g: LET m$(
23)=CHR$ (g+48): IF m$(22)="2" T
HEN LET m$(24)="0": GO TO 390
 380 INPUT "Is it 0)on surface,
1)buried a  few cm, 2)buried 1/2
 m or        3)buried 1m deep? (t
ype 0-3) ";m$(24)
 390 LET o$(f)=m$: NEXT f

 400 INPUT "How many monsters? "
;mo: IF NOT mo THEN GO TO 450
 410 DIM m$(mo,17): FOR f=1 TO m
o: DIM n$(17): INPUT "Monster ";
(f);": name? ";n$( TO 10)'"attac
k? (1-8) ";n$(11)'"defence? (1-5
) ";n$(12)'"resistance? (1-20) "
;g: LET n$(13)=CHR$ (g+48)
 420 INPUT "treasure? (0-100) ";
g: LET n$(15)=CHR$ (g+48)
 430 INPUT "Location? (0 if any)
 ";g: LET n$(16)=CHR$ (48+g): LE
T n$(17)="1": INPUT "How many ar
e there? (1-9) ";n$(17): LET m$(
f)=n$: NEXT f

 450 INPUT "Where do you start?
";lo: LET pl=lo

 460 INPUT "Victory conditions:"
'"How many crowns needed? ";v1'"
How many objects needed? ";v2: I
F v2 THEN DIM v(v2): DATA "st","
nd","rd","th": FOR f=1 TO v2: RE
STORE 460: FOR g=1 TO f-(f>4)*(f
-5): READ a$: NEXT g: INPUT "Num
ber of ";(f;a$);" object: ";v(f)
: NEXT f
 470 INPUT "Which location to be
found? ";v3
 480 INPUT "Which monster to be
killed? (0  if none) ";v4

 500 PRINT "8000 REM Scenario ";
z$( TO 32)''
 510 PRINT "8010 LET z$="""";z$;"
""""'
```

```
 520 PRINT "8020 DATA ";loc;",";
ob;",";mo;",";pl;",";dt;",";do;"
,";dd;",";v1;",";v2;",";v3;",";v
4;",";0;",";0;",";10;",";0;",";l
t''

 530 PRINT "8110 DATA ";: FOR f=
1 TO ob: PRINT """";o$(f, TO 11)
;"0000";o$(f,16 TO );""",";: NEX
T f: PRINT CHR$ 8;" "''
 540 PRINT "8120 DATA ";: FOR f=
1 TO ob: PRINT """";: FOR g=12 T
O 15: PRINT ("0" AND CODE o$(f,g
)<10);("0" AND CODE o$(f,g)<100)
;CODE o$(f,g);: NEXT g: PRINT ""
",";: NEXT f: PRINT CHR$ 8;" "''

 550 PRINT "8210 DATA """;: FOR
f=1 TO loc: PRINT r$(f, TO 7);""
",""";: NEXT f: PRINT CHR$ 8;CHR
$ 8;" "''
 560 IF mo THEN PRINT "8310 DATA
 ";: FOR f=1 TO mo: PRINT """";m
$(f);""",";: NEXT f: PRINT CHR$
8;" "''

 600 IF dt THEN PRINT "8410 DATA
 ";: FOR f=1 TO dt: PRINT """";:
 FOR g=1 TO 49: PRINT (s$(f,g) A
ND s$(f,g TO g+1)<>"  ");: NEXT
g: PRINT """,";: NEXT f: PRINT C
HR$ 8;" "''
 610 IF do THEN PRINT "8510 DATA
 ";: FOR f=1 TO do: PRINT """";:
 FOR g=1 TO 49: PRINT (p$(f,g) A
ND p$(f,g TO g+1)<>"  ");: NEXT
g: PRINT """,";: NEXT f: PRINT C
HR$ 8;" "''

 620 PRINT "8610 DATA ";: FOR f=
1 TO lt: PRINT """";t$(f);""",";
: NEXT f: PRINT CHR$ 8;" "''
 630 IF dd THEN PRINT "8710 DATA
 ";: FOR f=1 TO dd: PRINT """";:
 FOR g=1 TO 49: PRINT (f$(f,g) A
ND f$(f,g TO g+1)<>"  ");: NEXT
g: PRINT """,";: NEXT f: PRINT C
HR$ 8;" "''
 640 IF v2 THEN PRINT "8810 DATA
 ";: FOR f=1 TO v2: PRINT v(f);"
,";: NEXT f: PRINT CHR$ 8;" "

7990 STOP
```

HINTS ON USE

When you design your own scenarios, it is easy to get over-ambitious and try to do a 100-location scenario which ends up failing. Start with very simple scenarios, based on a grid map with ten or twelve locations. Gradually progress to larger maps, non-grid systems, and lots of monsters, treasure and objects.

Pay careful attention to aims, and make sure that these are possible to achieve. Getting a suitable balance to the game is difficult, and you may have to alter the monster characteristics several times before they are about the right strength – neither pushovers nor invincible. For this reason, and to aid in altering your scenarios in general, the system has been designed so that the 'Player Program', scenario and 'Scenario Maker' can all be MERGEd enabling you to check alterations immediately they are made.

Try looking at the listings for the scenarios I have included to get rough ideas on object data and monster strengths. 'Valley of the Swampbeast' and 'Hoard of the Exile' were designed on grids, the other two scenarios are non-grid.

Index

Fontana Paperbacks: Non-fiction

Fontana is a leading paperback publisher of non-fiction, both popular and academic. Below are some recent titles.

- ☐ WHAT DO WOMEN WANT? Luise Eichenbaum and Susie Orbach £1.75
- ☐ AVALONIAN QUEST Geoffrey Ashe £2.50
- ☐ WAR AND SOCIETY IN EUROPE, 1870–1970 Brian Bond £3.50
- ☐ MARRIAGE Maureen Green £2.75
- ☐ AMERICA AND THE AMERICANS Edmund Fawcett and Tony Thomas £2.95
- ☐ FRANCE, 1815–1914: THE BOURGEOIS CENTURY Roger Magraw £4.95
- ☐ RULES OF THE GAME Nicholas Mosley £2.50
- ☐ THIS IS THE SAS Tony Geraghty £4.95
- ☐ THE IMPENDING GLEAM Glen Baxter £2.95
- ☐ A BOOK OF AIR JOURNEYS Ludovic Kennedy (ed.) £3.95
- ☐ TRUE LOVE Posy Simmonds £2.95
- ☐ THE SUPPER BOOK Elizabeth Kent £2.95
- ☐ THE FONTANA BIOGRAPHICAL COMPANION TO MODERN THOUGHT Alan Bullock and R. B. Woodings (eds.) £6.95
- ☐ LIVING WITH LOSS Liz McNeill Taylor £1.75
- ☐ RASPUTIN Alex de Jonge £2.95
- ☐ THE PRIVATE EYE STORY Patrick Marnham £4.95
- ☐ BARTHES: SELECTED WRITINGS Susan Sontag (ed.) £4.95
- ☐ A POCKET POPPER David Miller (ed.) £4.95
- ☐ THE WRITINGS OF GANDHI Ronald Duncan (ed.) £2.50

You can buy Fontana paperbacks at your local bookshop or newsagent. Or you can order them from Fontana Paperbacks, Cash Sales Department, Box 29, Douglas, Isle of Man. Please send a cheque, postal or money order (not currency) worth the purchase price plus 15p per book for postage (maximum postage required is £3).

NAME (Block letters) ____________________

ADDRESS ____________________

____

While every effort is made to keep prices low, it is sometimes necessary to increase prices at short notice. Fontana Paperbacks reserve the right to show new retail prices on covers which may differ from those previously advertised in the text or elsewhere.

NEW ADVENTURE SYSTEMS FOR THE SPECTRUM

A system is much more than a single program. Starting with a basic core program – a complete adventure game in its own right – you can build on to it in short stages, adding a more sophisticated program, a new aim or a different scenario. For example:

THE PRESET FANTASY GAME A simple scenario in which you are besieged with problems searching for a statue – then expanding to an adventure in a strange temple, before finally bringing you into conflict with a cunning wizard....

THE FANGMOLE TUNNELS Starting with a 16K program, you are hunting for gold in a maze. Then the computer selects one of three possible aims for you – while you are also fleeing from charging Fangmoles....

Secrecy is preserved by the innovative use of special codes, which maintain all the surprise elements and challenges as you type in and read the listings.

NEW ADVENTURE SYSTEMS FOR THE SPECTRUM makes it easy and fun to build up the games, and concentrates on a wide range of scenarios, lots of graphics, and the expandability of each game.

ISBN 0-00-636699-6
90000
UK £3.95
NZ $14.95*
* recommended price
9 780006 366997
0 00 636699 6