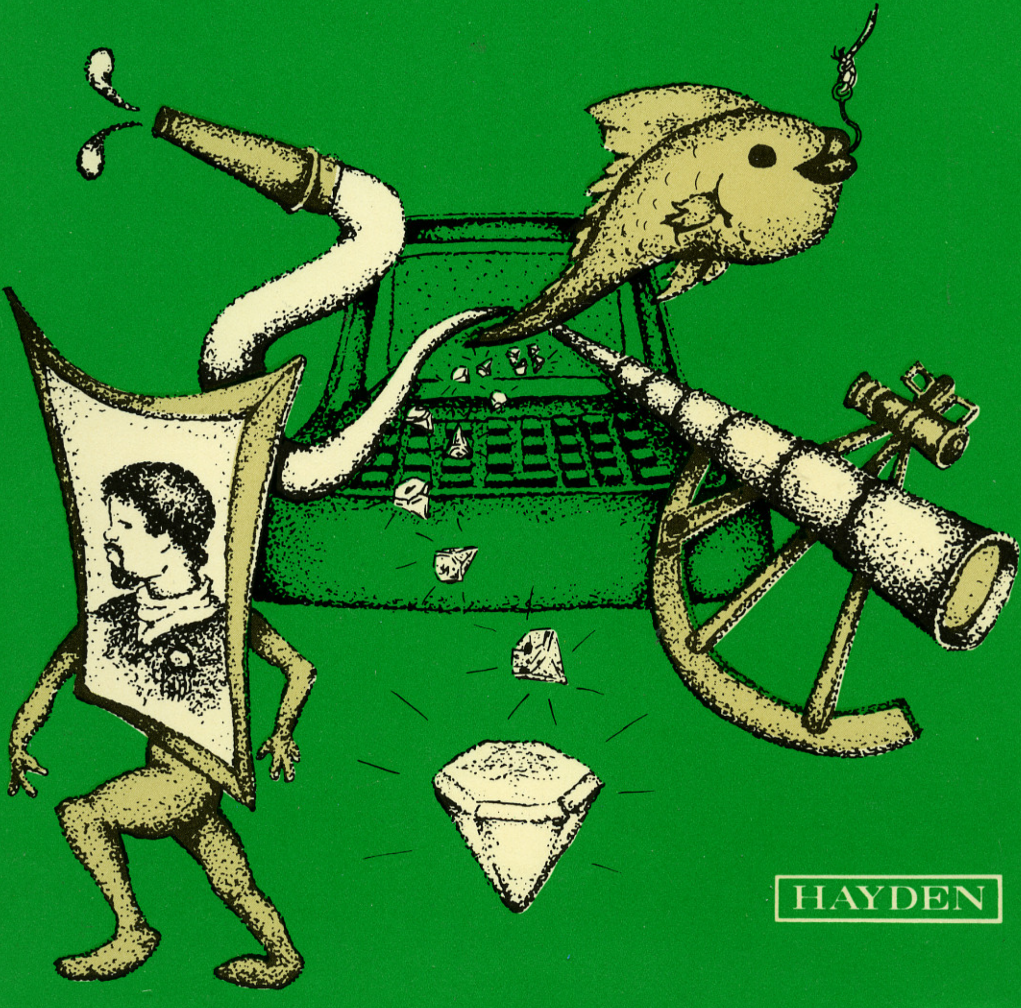
Stimulating Simulations for the TI-99/4A™

C. W. Engel

13 unique programs in BASIC for the computer hobbyist

Soccer • Art Auction • Monster Chase • Lost Treasure • Gone Fishing • Space Flight • Forest Fire • Nautical Navigation • Business Management • Rare Birds • Diamond Thief • The Devil's Dungeon • Life



HAYDEN

Stimulating Simulations

for the TI-99/4A™

Stimulating Simulations

for the TI-99/4A™

C. W. Engel

HAYDEN BOOK COMPANY, INC.
Hasbrouck Heights, New Jersey

EQUIPMENT NEEDED

To use the programs in this book on a TI-99/4A, you will need the following equipment:

- □ TI-99/4A, a monitor or TV
- □ Cassette tape player (optional)
- □ Disk drive and disk controller (optional)
- □ TI Extended BASIC Cartridge
  (for the Soccer and Life programs only)

Acquisitions Editor: Les Purificación
Production Editor: Lori Williams
Developmental Editor: Karen Pastuzyn
Art Director: Jim Bernard
Conversion by: Karl Smith
Production Service: The Bookmakers, Incorporated

TI-99/4A is a trademark of Texas Instruments Incorporated, which is not affiliated with Hayden Book Company, Inc.

Printed in the United States of America

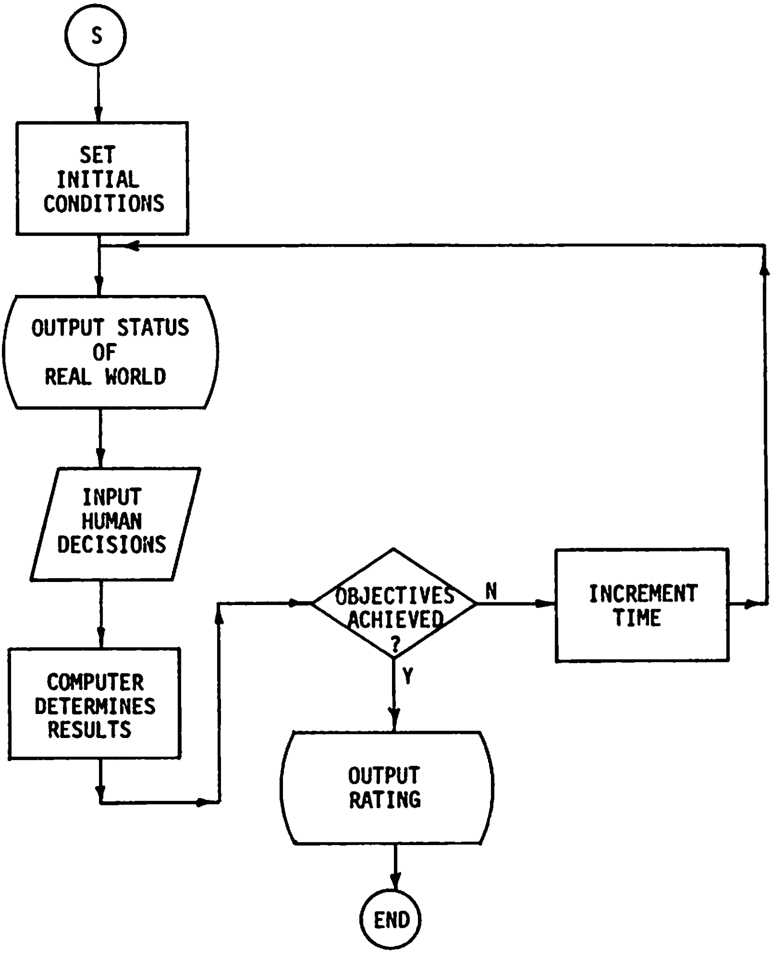
1 2 3 4 5 6 7 8 9 PRINTING
84 85 86 87 88 89 90 91 92 YEAR

Contents

INTRODUCTION

Simple number games and puzzles are frequently developed by beginning computer hobbyists. While some enthusiasts develop computer systems that monitor environmental conditions, compute income tax, or serve as expensive burglar alarms, most continue to use their computers primarily for recreation. This book is designed for the person who is beyond the simple number-game stage of software development and would like to develop some interesting simulations. It is assumed that the reader is familiar with most of the BASIC commands and has written some simple programs.

Most of the programs in this book are written so that the computer does not do all of the "thinking" but forces the player to develop strategies for achieving the objectives. A general overview of a simulation is illustrated in the flowchart below.



The simulations presented in this book are written in BASIC and can be easily adapted to almost any system.* The programs vary from 1,000 to 3,500 bytes.

Each simulation begins with a scenario describing the rules, conditions and objectives to be achieved. The rules have been written in second person, because some programmers like to condense the rules and place them in a subroutine for access by the operator. A sample run and a general flowchart with line numbers provide additional information about each program. A description of the variables precedes the program listing. Some program modifications are suggested. The minor modifications require only adjustments of variables in specific lines, while major modifications require additional programming. In some cases, supplemental playing boards, graphs, and charts are supplied for recording information on the progress of the simulation.

A brief description of each program is given below.

1. ART AUCTION (63) lines)
   One buys and sells paintings to make a maximum profit. This is a fast simulation and does not require extra materials.

2. MONSTER CHASE (101 lines)
   A monster is chasing a victim in a cage. The victim must elude the monster for ten moves to survive. This is a fairly quick simulation that doesn't require too much thinking.

3. LOST TREASURE (100 lines)
   A map of an island that contains treasure is presented. The adventurer travels over different terrain with a compass that isn't very accurate in an attempt to find the treasure. This is a short simulation that requires about 15 moves. A map is provided.

4. GONE FISHING (96 lines)
   The objective is to catch a lot of fish during a fishing trip. Half of the catch spoils if the time limit is exceeded, time is lost in a storm, and the boat sinks if it is guided off of the map. There are also sea gulls and sharks to watch. A chart is needed to keep track of good fishing spots.

5. SPACE FLIGHT (93 lines)
   The task is to deliver medical supplies to a distant planet while trying to stay on course without running out of fuel. Graph paper is required to plot the course.

6. FOREST FIRE (120 lines)
   The objective is to subdue a forest fire with chemicals and backfires. The success of a firefighter is based on the time needed to control the fire and to completely extinguish it.

*The soccer programs are in extended BASIC.

7. NAUTICAL NAVIGATION (98 lines)
   This simulation requires the navigation of a sailboat to three different islands, using a radio direction finder. The wind direction is an important variable. Graph paper, protractor and ruler are needed to plot the course.

8. BUSINESS MANAGEMENT (110 lines)
   In this simulation, raw materials are bought and finished products are produced and sold. The cost of materials and production and the selling price vary each month. The objective is to maximize the profits. No extra materials are required.

9. RARE BIRDS (103 lines)
   This is a bird watching simulation. The objective is to identify as many different birds as possible. A record of those identified is helpful and a bird watching chart is provided.

10. DIAMOND THIEF (110 lines)
    One assumes the role of a detective is this simulation. A thief has just stolen a diamond from a museum. Five suspects must be questioned to determine the thief. A floor plan of the museum and a chart indicating suspects and times are provided.

11. THE DEVIL'S DUNGEON (225 lines)
    A fantasy adventure into a bottomless cave. The player must chart his way, fight monsters, poisonous gas and demons to escape with the gold.

12. LIFE (155 lines)
    Beginning as an uneducated bum whose only source of income is mugging, the player advances through education and luck to become an executive who earns lots of money.

The SOCCER program developed in the last section of the Introduction is designed for two players, although it could be modified so that the computer is one of the players. In this simulation, each player controls a team of five soccer players whose objective is to kick the ball across the opponent's goal line. This program is written in three stages to illustrate the procedure for modifying and expanding already existing simulations.

In addition to extending the simulations in this book, the reader might try combining some of them. For example, one could use the money earned in Art Auction to start the Business Management simulation. After twelve months of business, the profits could be used to buy a boat to use in the Gone Fishing simulation. A larger boat could survive more storms, hold more fish, and allow fishing in deeper water. The ultimate objective could be to catch the most fish.

The computer hobbyist is limited only by the imagination in simulating real events. It is the author's desire that this book provide some fun and, at the same time, stimulate further development of creative simulations. Some additional ideas for simulations are suggested below.

1. Hunt Big Foot
2. Race a Sailboat
3. Inhibit the Andromeda Strain
4. Stop the African Bee Invasion
5. Climb Mountains
6. Survive in the Wilderness
7. Find Gold or Oil
8. Swim from Sharks
9. Dispatch Airplanes, Trains, or Trucks
10. Herd Sheep
11. Explore Caves
12. Catch Butterflys

The next section offers some guidelines for developing simulation activities.

DEVELOPING SIMULATIONS

A Creative Process

If one has a mathematical problem for computer solution, the programming process can be approached in the following manner: 1) Develop the flowchart. 2) Define the variables. 3) Write the initial program. 4) Debug. 5) Run. In developing a simulation activity, however, there is a great deal more creative effort involved; and the steps listed above are not necessarily implemented in sequence. One can compare the development of a simulation program to that of a creative artist such as a painter. The blank computer memory is the canvas and BASIC language represents the paint and brushes. An artist continually retouches and reworks the painting until the final product meets the artist's criteria for success.

Most technological advances, such as television and radio, are "one-way streets" -- one observes what takes place. The observer seldom creates, composes or interacts with such devices. Developing simulation programs for computers can provide intelligent people with an opportunity to react with their environment in a problem-solving mode.

Selecting a Topic

The first task in developing a computer simulation is to select a topic. Almost any idea could serve as a starting point; however, the reader's own interests and hobbies are usually the best resource for ideas. The possibilities are unlimited. One could develop simulations on cooking, stamp collecting, gardening, racing cars, dating, jogging or dreaming. With a little research, a long-desired ambition could become material for an exciting simulation -- a safari across Africa, a trip around the world, or a walk on the moon. The creative programmer can be transported to any time or any place in the universe via the computer simulation.

Once a topic for the simulation is selected, the next step is to write down a fairly detailed description of what the program will accomplish. This narration will become the scenario. To illustrate this process, the author has chosen "survival in a jungle" as a topic.

Jungle Survival Scenario

You have crashed somewhere in the middle of an uninhabited jungle island in the Pacific. You will have to select a limited quantity from the provisions on the plane. The more provisions you carry, the slower you will travel. As you travel across the island, you will encounter various hazards with which you must deal. The terrain will consist of mountains, rivers, plains, swamps and lakes. Crossing a mountain range will be slow, but it will provide a more direct route. Traveling down a river will be easy, but a variety of unpredictable hazards will occur. Your objective is to hike to the perimeter of the island in as few days as possible.

The scenario should provide answers to the following questions.

1. What will the operator do?
2. What feedback will the computer provide?
3. What surprise elements will produce fun and excitement?
4. What are the winning conditions?
5. How will the success of the simulation be measured?

The writer must realize that the first scenario is only an approximation to the final product. As the program is developed and field tested, the scenario will probably change considerably.

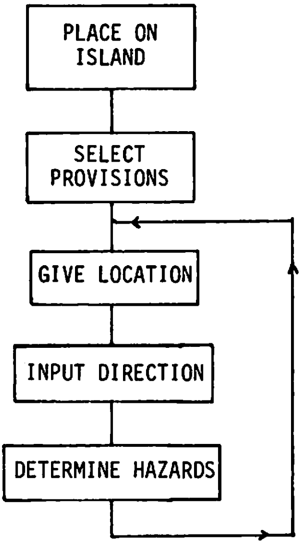
While developing the scenario, the writer should begin to visualize a sample run. In the case of the jungle survival program, a sample run might look something like the following.

```
CHOOSE YOUR PROVISIONS:  1  FOOD
                         2  WATER
                         .
                         .
                         N  XXXXXXX

READY TO START JOURNEY?
YOU ARE AT POSITION 42,43.  IN THE CLEAR
CHOOSE THE DIRECTION OF YOUR NEXT MOVE?  N
HOW FAR WOULD YOU LIKE TO GO?  32 MILES

YOU ARE AT POSITION 42,42.  IN THE MOUNTAINS
CHOOSE THE DIRECTION OF YOUR NEXT MOVE?  E
HOW FAR WOULD YOU LIKE TO GO?  10 MILES

YOU FELL INTO THE RIVER!
```

The sample run listed above has several problems. First, the distance the player can travel in a given time-interval should be limited. Also, one should probably be able to see mountains ahead. At this point in the development of the program, however, the writer should have decided that the output of the computer will include the location of the player, the type of terrain, and a request for the player to select the direction of travel.

Flowchart

The next step in developing a simulation is to construct a general flowchart. In the case of the jungle survival simulation, the first flowchart might take the following form.



It is not necessary to provide all of the details in the flowchart in the beginning. It is better to start writing the program and develop the flowchart along with the program. The flowchart should provide a graphic aid to the programming and need only be developed to the extent that the programmer feels it is necessary to keep track of the flow of ideas.

Selecting the Variables

It is a good idea to keep a list of the variables used in the program. If such a list is not referred to and continually updated, the same variable might be used to represent two different things. Usually the letters, I, J, K, are used for indexing loops; and the first one or two letters of a word are selected for major variables in the program, e.g., T for time. It is also useful to designate a range for the variables.

In the jungle survival program, a list of the variables might be as follows.

		Range
X,Y	position on island	0 - 100
T	time on island	0 - 100
E	energy of survivor	0 - 100
W	weight of provisions	0 - 50
MX,MY	location of mountains	
LX,LY	location of lakes	

CX.CY	location of clearings
M	direction of movement

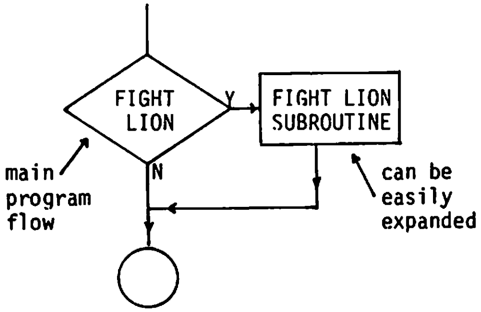
The list of variables should be expanded as needed during the writing of the program.

Subroutines

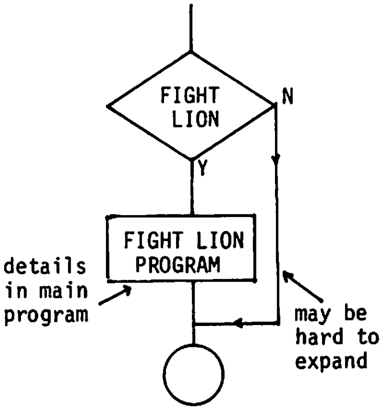
One of the reasons given for using subroutines is to limit the amount of repetition in a program. Another use of subroutines is to provide flexibility in developing a program. The main parts of a program can be written first and subroutines can be used to add the details later. The use of subroutines frees the writer from having to determine in advance how many lines are needed between main parts of the program. Also, the main parts of the program can be more easily identified if subroutines are used to handle the details.

The use of subroutines, as described above, is illustrated below.

FLOWCHART WITH SUBROUTINES



FLOWCHART WITHOUT SUBROUTINES



Writing the Program

After developing a rough flowchart, one can start to write and test the first part of the program. It is not usually a good idea to type in and test a long, complicated program in its entirety. The writer should make sure that the first part of the program works independently. Usually after some experimentation with the initial part of the program, one will think of new ideas; and the flowchart and/or scenario will be revised. The programmer should not forget to keep an updated version of the program on a disk or tape to avoid a second typing of the program due to an accidental loss of memory.

Sometimes the writer may find a particular objective very difficult to program. Rather than spend considerable time trying to achieve what may be impossible, it would be advisable to change the scenario. Quite often such "open-mindedness" leads to a more interesting or more elegant simulation than was originally anticipated. The writer, on the other hand, should not hesitate to program what might seem like a complex idea. Many times complex ideas are easy to program, while simple ideas are very difficult to program. The programmer should not strive for perfection. Most programs could probably be "neater" or more elegant with the investment of a few more hours of programming time; however, the only accomplishment might be to save a few milliseconds during the run.

The simulation should be fairly simple at first, until it is running. Then the programmer can add the "bells and whistles" if desirable. Sometimes too much complexity distracts from the enjoyment of the simulation, especially if it takes another computer to operate the simulation.

When writing a program, one should keep all program statements involving a similar idea together. Such a practice will make debugging a program much easier. A brief summary of the instructions for the simulation is also worthwhile if memory capacity is sufficient.

It is sometimes difficult to provide an appropriate balance between skill and luck. The chance factors provide interest, excitement and intrigue; however, too much luck does not provide sufficient challenge. Also, with too many chance factors, it would be difficult to compare different runs of the program. An interesting possibility would be to provide a variety of options at the beginning of a program that determines the balance of luck and skill.

Field Testing

When the program is in a "playable" form, it should be tested by several different players. An unanticipated method for achieving the objective may be discovered or the objective may be almost impossible to achieve. Most likely, one will find that many new ideas will result from feedback from these players, and some will be easily incorporated into the program.

The writer will find that the simulation will never reach, but only approximate, the ideal. The fun and excitement of creating, modifying, and expanding your simulation will never end.

In the next section of this book are fourteen simulations that are in a playable form; however, they are only the beginning for the person with a creative mind.

MODIFYING AND EXPANDING SIMULATIONS

Each program in this book concludes with a list of suggested modifications. This section illustrates how to modify and expand a simple program, SOCCER I, to the more sophisticated SOCCER II and SOCCER III. These three programs require two people to operate the computer, where each person controls five players on a playing field.

The objective in SOCCER I is to eliminate the opponent's players. SOCCER I is the least sophisticated of the three programs and does not provide for incorrect inputs from the keyboard.

In SOCCER II, the objective is to be the first team to pick up a ball that is resting in the middle of the field. Sidelines are drawn in this program, and a player's movement can be stopped by pressing the space bar. Incorrect key entries are ignored.

In the last version presented here, SOCCER III, one must kick the ball across the opponent's goal line. When a player touches the ball, it moves in one of three random directions toward the goal, unless it is blocked by an opponent. Injured players appear on the sidelines.

The technique of modifying and/or expanding existing programs is very valuable. It would be a good exercise for the student to continue expanding this program by using the suggestions listed at the end of the SOCCER III section.

SOCCER I

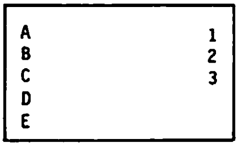
Scenario

This simulation requires two people to play. One person controls the five letters, A, B, C, D and E; another person controls the five numerals, 1, 2, 3, 4 and 5. In the beginning, the letters appear on the left side of the screen and the numerals appear on the right side of the screen. A small dot will appear on either the left or right side of the screen to indicate which player can take a turn.

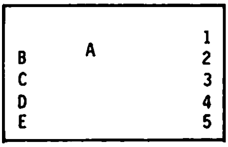
A turn consists of moving one of the five players by entering the appropriate numeral or letter, followed by an arrow entry to indicate the general direction of movement. A player moves ten spaces each turn. If a player lands on an opponent, the game is over. Incorrect key entries must be avoided in this program or the program will halt.

Sample Run

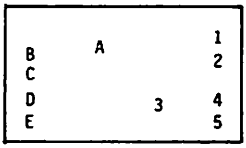
1



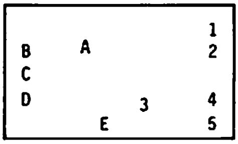
2



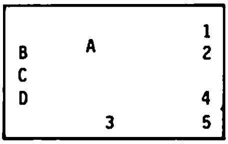
3



4



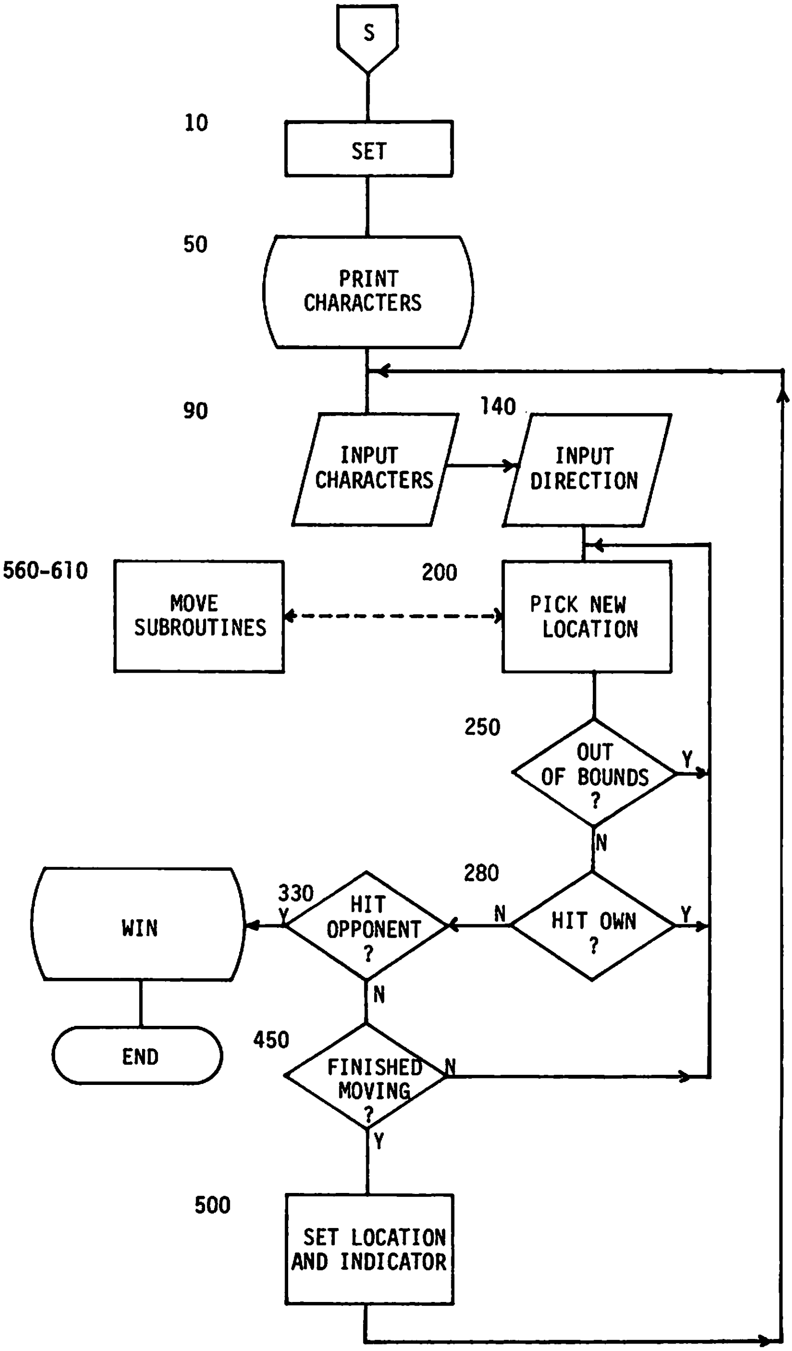
5



6

NUMBERS WIN!

SOCCER I FLOWCHART



SOCCER I PROGRAM

Variables

I,J	Index variables
P	Player (1 or 2)
CL	Location of prompt
K	Keyboard input
S	Status of keyboard input
SP	Sprite number
DE	Delay
X,Y	Position of player
R,C	Row, Column
DI	Direction input
E	Energy
CK	Check

Program Listing

```
1 REM ** SOCCER I **
2 REM SET CHARACTERS
3 RANDOMIZE
5 CALL CLEAR
7 CALL SCREEN(4)
10 FOR I=1 TO 5
12 CALL SPRITE(#I,64+I,2,15+25*I,15)
13 CALL SPRITE(#I+5,48+I,2,15+25*I,235)
15 P=1
20 NEXT I
50 IF P=1 THEN CL=3
52 IF P=2 THEN CL=30
54 CALL HCHAR(24,3,32)::CALL HCHAR(24,30,32)
56 CALL HCHAR(24,CL,94)
59 REM INPUT CHARACTER
60 CALL KEY(0,K,S)::IF S=0 THEN 60
62 IF P=1 THEN SP=K-64 ELSE SP=K-43
65 CALL PATTERN(#SP,42)
67 FOR DE=1 TO 50::NEXT DE
70 CALL POSITION(#SP,X,Y)
99 REM INPUT DIRECTION
100 CALL KEY(0,DI,S)::IF S=0 THEN 100
110 D$=CHR$(DI)::E=5::R=0::C=0
120  IF D$="S" AND  Y>30 THEN  R=E*(RND-.5)::
     C=-E
130  IF D$="D"  AND Y<130 THEN R=E*(RND-.5)::
     C=E
140  IF D$="E"  AND X>30  THEN C=E*(RND-.5)::
     R=-E
150  IF  D$="X" AND X<140 THEN C=E*(RND-.5)::
     R=E
199 REM MOVE
200 CALL MOTION(#SP,R,C)
210 FOR DE=1 TO 25
220 NEXT DE
230 CALL MOTION(#SP,0,0)
```

```
280 IF P=1 THEN J=6 ELSE J=1
290 FOR I=J TO J+4
295 IF I=SP THEN 340
300 CALL COINC(#SP,#I,10,CK)
320 IF  CK=-1 AND P=1 THEN CALL  CLEAR::PRINT
     "LETTERS WIN"
330 IF CK=-1  AND P=2  THEN CALL CLEAR::PRINT
     "NUMBERS WIN"
340 NEXT I
400 IF P=1 THEN P=2 ELSE P=1
410 CALL PATTERN(#SP,K)
420 GOTO 40
```

Soccer II

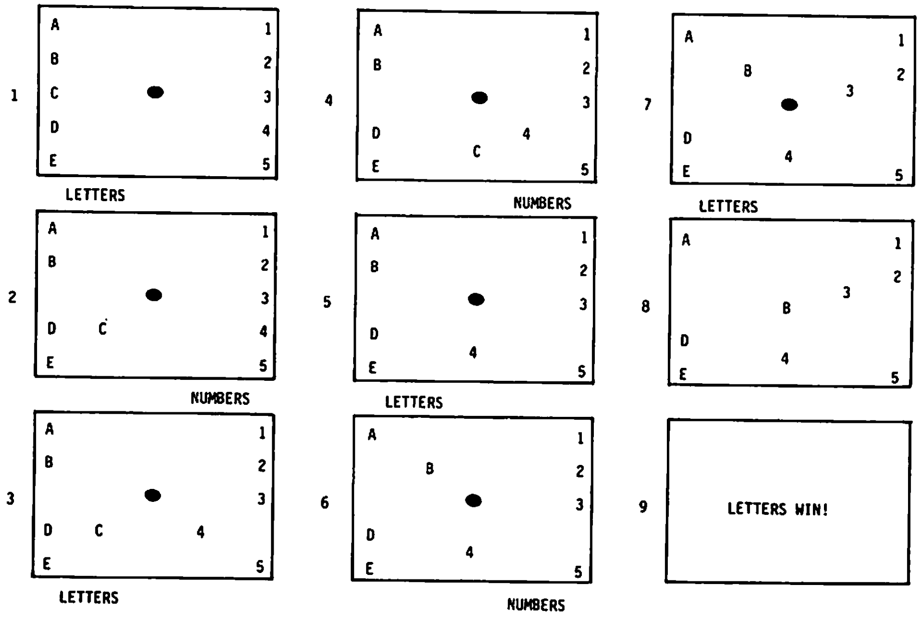
This program is an extension of the previous program, SOCCER I. It is a good idea to have SOCCER I running before proceeding with the modifications and additions suggested in this section.

Scenario

In this simulation, as in SOCCER I, two people control five players each. The major difference is the objective -- to be the first to land on a ball resting in the middle of the field. You can eliminate more than one of your opponent's players. Also, you can stop your own player's movement by pressing the space bar.

A border is drawn around the field, and prompts are printed at the bottom of the field to indicate each player's turn and the character that has been entered. Inappropriate entries from the keyboard are not accepted. The strength of the players, which diminishes with each move and increases when resting, determines the players' ability to move and eliminate opponents.

Sample Run



SOCCER II PROGRAM

Variables

The same as for SOCCER I with the following additons:

ST(SP)	Strength of player
FI	Finish move
W	Win check

Program Listing

The same as for SOCCER I with the following changes:

To change location of characters and ball, add line 8 and change lines 12 and 13:

```
8 CALL SPRITE(#11,42,7,80,124)
12 CALL SPRITE(#I,64+I,2,10+25*I,15)
13 CALL SPRITE(#I+5,48+I,2,10+25*I,235)
```

To add a border, add lines 22, 24, and 26:

```
22 CALL COLOR (14,14,1)
24 CALL CHAR(136,"FFFFFFFFFFFFFFFF")
26 CALL HCHAR(1,1,136,32)
```

To replace dot indicators with the words "LETTERS" and "NUMBERS" eliminate lines 54 and 56 and make the following additions to lines 50 and 52:

```
50 ..... ::DISPLAY AT(24,1)BEEP:"LETTERS"
52 ..... ::DISPLAY AT(24,20)BEEP:"NUMBERS"::
           DISPLAY AT(24,1):"     "
```

To make sure that correct characters are entered, add line 61 and change line 62:

```
61 IF P=1 AND (K<65 OR K>69) THEN 60
62 IF P=2 AND (K<49 OR K>53) THEN 60
```

To make sure that correct directions are entered, add line 155:

```
155 IF R*C=0 THEN 100
```

To stop a player with any key, add lines 212 and 214:

```
212 CALL KEY(0,FI,S)
214 IF S<>0 THEN 230
```

To eliminate opponents, change line 320 to

```
320 IF CK=-1 THEN CALL DELSPRITE(#1)
```

and delete line 330.

To add strength, change line 210 and add lines 411, 415, 417, and 418:

```
210 FOR DE=1 TO 50
411 ST(SP)=ST(SP)-4::
      IF ST(SP)<0 THEN ST(SP)=1
415 FOR I=1 TO 10::ST(I)=ST(I)+1
417 IF ST(I)>10 THEN ST(I)=10
418 NEXT I
```

To determine winner, change line 420 and add lines 430, 440, and 500:

```
420 IF W=-1 THEN CALL CLEAR ELSE 40
430 IF P=1 THEN PRINT "LETTERS WIN"::END
440 IF P=2 THEN PRINT "NUMBERS WIN"::END
500 GOTO 40
```

Program Listing

```
1 REM ** SOCCER II **
2 REM SET CHARACTERS
3 RANDOMIZE
5 CALL CLEAR
7 CALL SCREEN(4)
8 CALL SPRITE(#11,42,7,80,124)
10 FOR I=1 TO 5
12 CALL SPRITE(#I,64+I,2,10+25*I,15)
13 CALL SPRITE(#I+5,48+I,2,10+25*I,235)
15 P=1
20 NEXT I
22 CALL COLOR(14,14,1)
24 CALL CHAR(136,"FFFFFFFFFFFFFFFF")
26 CALL HCHAR(1,1,136,32)::
      CALL HCHAR(22,1,136,32)
27 CALL VCHAR(1,1,136,22)::
      CALL VCHAR(1,32,136,22)
30 FOR I=1 TO 10::ST(I)=10::NEXT I
50 IF P=1 THEN CL=3::
     DISPLAY AT(24,1)BEEP:"LETTERS"
52 IF P=2 THEN CL=30::
     DISPLAY AT(24,1)BEEP:"     "::
     DISPLAY AT(24,22):"NUMBERS"
59 REM INPUT CHARACTER
60 CALL KEY(0,K,S)::IF S=0 THEN 60
61 IF P=1 AND(K<65 OR K>69)THEN 60
62 IF P=2 AND(K<49 OR K>53)THEN 60
63 IF P=1 THEN SP=K-64 ELSE SP=K-43
65 CALL PATTERN(#SP,42)
67 FOR DE=1 TO 50::NEXT DE
70 CALL POSITION(#SP,X,Y)
99 REM INPUT DIRECTION
100 CALL KEY(0,DI,S)::IF S=0 THEN 100
110 D$=CHR$(DI)::E=ST(SP)::R=0::C=0
120 IF D$="S" AND  Y>30  THEN  R=E*(RND-.5)::
     C=-E
130 IF D$="D"  AND Y<220 THEN  R=E*(RND-.5)::
     C=E
140  IF D$="E"  AND X>30  THEN C=E*(RND-.5)::
     R=-E
150  IF D$="X" AND X<140 THEN  C=E*(RND-.5)::
     R=E
155 IF R*C=0 THEN 100
199 REM MOVE
200 CALL MOTION(#SP,R,C)
```

```
210 FOR DE=1 TO 50
212 CALL KEY(0,FI,S)
214 IF S<>0 THEN 230
220 NEXT DE
230 CALL MOTION(#SP,0,0)
280 IF P=1 THEN J=6 ELSE J=1
290 FOR I=J TO J+4
295 IF I=SP THEN 340
300 CALL COINC(#SP,#I,10,CK)
320 IF CK=-1 AND ST(SP)>ST(I)THEN
      CALL DELSPRITE(#I)
340 NEXT I
400 IF P=1 THEN P=2 ELSE P=1
410 CALL PATTERN(#SP,K)
411 ST(SP)=ST(SP)-4::IF ST(SP)<=0 THEN
     ST(SP)=1
415 FOR I=1 TO 10::ST(I)=ST(I)+1
417 IF ST(I)>10 THEN ST(I)=10
418 NEXT I
420 CALL COINC(#SP,#11,10,W)
425 IF W=-1 THEN CALL CLEAR ELSE 40
430 IF P=1 THEN PRINT "LETTERS WIN"::END
440 IF P=2 THEN PRINT "NUMBERS WIN"::END
```

Soccer III

This program is an expansion of the previous program, SOCCER II. SOCCER II should be working well before one begins to develop SOCCER III.

Scenario

The movement of the players in SOCCER III is the same as in the previous program, SOCCER II. In order to win in SOCCER III, however, one of your players must kick the ball across the opponent's goal line. The distance the ball is kicked will depend on the strength of the player. When eliminated, a player appears on the sideline.

SOCCER III PROGRAM

Variables

The same as SOCCER II with the following additon:

SL	Sideline position

Program Listing

The same as for SOCCER II with the following changes:

So that strength is not a factor in eliminating players, and to place an eliminated player on the sideline, change line 320:

```
320 IF CK=-1 THEN CALL LOCATE(#1,180,120+SL)::
     SL=SL+10
```

To move ball, add the following:

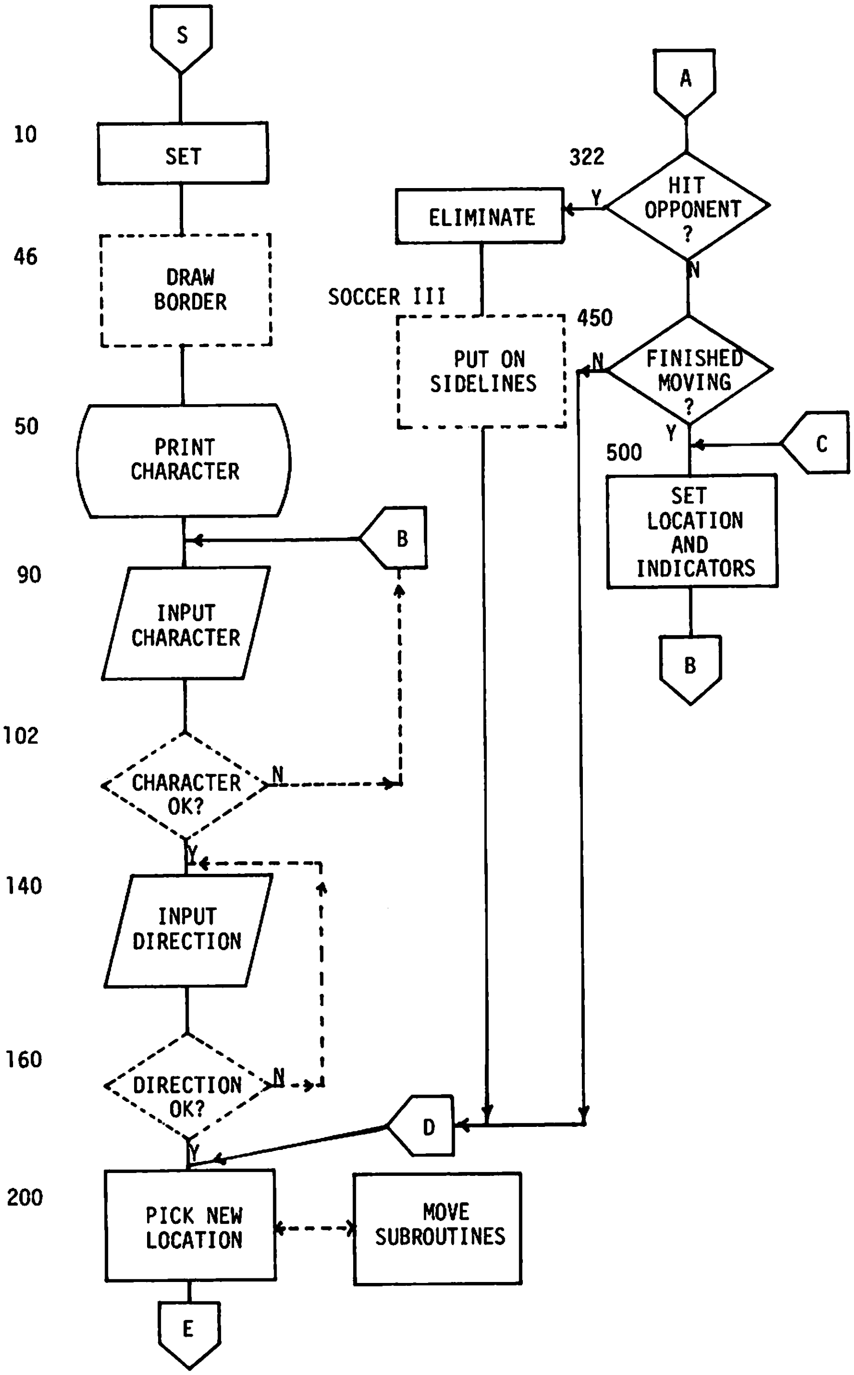
```
425 IF W<>-1 THEN 40
440 CALL POSITION(#11,X,Y)
442 IF D$="S" AND Y<40 THEN CALL CLEAR::
     PRINT "NUMBERS WIN"::END
444 IF D$="D" AND Y>210 THEN CALL CLEAR::
     PRINT "LETTERS WIN"::END
446 IF D$="E" AND X<40 THEN 40
448 IF D$="X" AND X>120 THEN 40
450 CALL MOTION(#11,10*R,10*C)
455 FOR DE=1 TO 5*ST(SP)::NEXT DE
460 CALL MOTION(#11,0,0)
500 GOTO 40
```

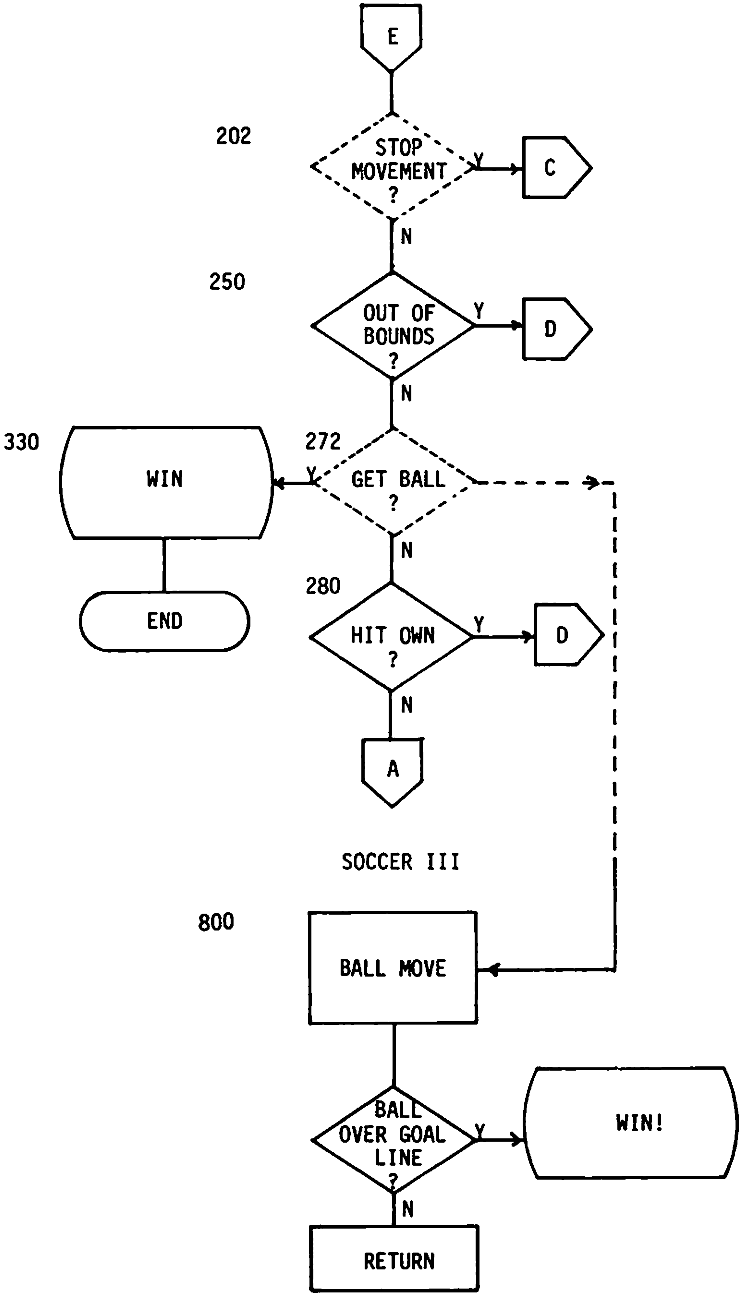
Program Listing

```
1 REM ** SOCCER III **
2 REM SET CHARACTERS
3 RANDOMIZE
5 CALL CLEAR
7 CALL SCREEN(4)
8 CALL SPRITE(#11,42,7,80,124)
10 FOR I=1 TO 5
12 CALL SPRITE(#I,64+I,2,10+25*I,15)
13 CALL SPRITE(#I+5,48+I,2,10+25*I,235)
15 P=1
20 NEXT I
22 CALL COLOR(14,14,1)
24 CALL CHAR(136,"FFFFFFFFFFFFFFFF")
26 CALL HCHAR(1,1,136,32)::
     CALL HCHAR(22,1,136,32)
27 CALL VCHAR(1,1,136,22)::
     CALL VCHAR(1,32,136,22)
30 FOR I=1 TO 10::ST(I)=10::NEXT I
50 IF P=1 THEN CL=3::
     DISPLAY AT(24,1)BEEP:"LETTERS"
52 IF P=2 THEN CL=30::
     DISPLAY AT(24,1)BEEP:"       "::
     DISPLAY AT(24,22):"NUMBERS"
59 REM INPUT CHARACTER
60 CALL KEY(0,K,S)::IF S=0 THEN 60
61 IF P=1 AND(K<65 OR K>69)THEN 60
62 IF P=2 AND(K<49 OR K>53)THEN 60
63 IF P=1 THEN SP=K-64 ELSE SP=K-43
```

```
65 CALL PATTERN(#SP,42)
67 FOR DE=1 TO 50::NEXT DE
70 CALL POSITION(#SP,X,Y)
99 REM INPUT DIRECTION
100 CALL KEY(0,DI,S)::IF S=0 THEN 100
110 D$=CHR$(DI)::E=ST(SP)::R=0::C=0
120 IF D$="S" AND  Y>30  THEN  R=E*(RND-.5)::
     C=-E
130 IF D$="D"  AND Y<220 THEN  R=E*(RND-.5)::
     C=E
140  IF D$="E"  AND X>30  THEN C=E*(RND-.5)::
     R=-E
150  IF D$="X" AND X<140 THEN  C=E*(RND-.5)::
     R=E
155 IF R*C=0 THEN 100
199 REM MOVE
200 CALL MOTION(#SP,R,C)
210 FOR DE=1 TO 50
212 CALL KEY(0,FI,S)
214 IF S<>0 THEN 230
220 NEXT DE
230 CALL MOTION(#SP,0,0)
280 IF P=1 THEN J=6 ELSE J=1
290 FOR I=J TO J+4
295 IF I=SP THEN 340
300 CALL COINC(#SP,#I,10,CK)
320 IF CK=-1 AND ST(SP)>ST(I)THEN
      CALL LOCATE(#I,180,120+SL)::
      SL=SL+10
340 NEXT I
400 IF P=1 THEN P=2 ELSE P=1
410 CALL PATTERN(#SP,K)
411 ST(SP)=ST(SP)-4::IF ST(SP)<=0 THEN
     ST(SP)=1
415 FOR I=1 TO 10::ST(I)=ST(I)+1
417 IF ST(I)>10 THEN ST(I)=10
418 NEXT I
420 CALL COINC(#SP,#11,10,W)
425 IF W<>-1 THEN 40
440 CALL POSITION(#11,X,Y)
442 IF D$="S" AND Y<40 THEN CALL CLEAR::
     PRINT "NUMBERS WIN"::END
444 IF D$="D" AND Y>210 THEN CALL CLEAR::
     PRINT "LETTERS WIN"::END
446 IF D$="E" AND X<40 THEN 40
448 IF D$="X" AND X>120 THEN 40
450 CALL MOTION(#11,10*R,10*C)
455 FOR DE=1 TO 5*ST(SP)::NEXT DE
460 CALL MOTION(#11,0,0)
500 GOTO 40
```

SOCCER II AND III FLOWCHART




E
202
STOP MOVEMENT ?
Y
C
N
250
OUT OF BOUNDS ?
Y
D
N
330
WIN
272
Y
GET BALL ?
END
N
280
HIT OWN ?
Y
D
N
A
SOCCER III
800
BALL MOVE
BALL OVER GOAL LINE ?
Y
WIN!
N
RETURN

SOCCER III MODIFICATIONS

Modifications

The following modifications provide the reader with a variety of interesting options.

To make the ball easier to hit, change line 420:

```
420 CALL COINC(#SP,#11,15,W)
```

To keep score, add lines 442 and 444:

```
442 IF D$="S" AND Y<40 THEN GOTO 600
444 IF D$="D" AND Y>120 THEN GOTO 700
```

and the following subroutines:

```
600 REM WIN SUBROUTINE
610 CALL CLEAR
620 DISPLAY AT (10,8)BEEP:"NUMBERS WIN!"
630 NU=NU+1:GOTO 800

700 REM WIN SUBROUTINE
710 CALL CLEAR
720 DISPLAY AT (10,8)BEEP:"LETTERS WIN!"
730 LE=LE+1:GOTO 800

800 DISPLAY AT(12,5):"LETTER'S SCORE:";LE
810 DISPLAY AT(14,5):"NUMBER'S SCORE:";NU
812 DISPLAY AT(24,8):"PRESS ANY KEY"
814 FOR DE=1 TO 200:NEXT DE
816 CALL DELSPRITE(ALL)
820 CALL KEY(0,XX,S)::IF S=0 THEN 820
830 GOTO 2
```

To block the ball, add

```
270 BB=0
305 CALL COINC(#SP,#1,30,BL)
307 CALL POSITION(#I,X1,Y1)

324 IF BL<>-1 OR ST(I)<5 OR RND<.5 THEN 340
325 IF P=1 AND D$="D" AND X<X1 THEN BB=1
327 IF P=2 AND D$="S" AND X>X1 THEN BB=1

425 IF W<>=1 OR BB=1 THEN 40
```

Program Listing

```
1 REM ** SOCCER III MODIFICATION **
2 REM SET CHARACTERS
3 RANDOMIZE
5 CALL CLEAR
7 CALL SCREEN(4)
8 CALL SPRITE(#11,42,7,80,124)
10 FOR I=1 TO 5
12 CALL SPRITE(#I,64+I,2,10+25*I,15)
```

```
13 CALL SPRITE(#I+5,48+I,2,10+25*I,235)
15 P=1
20 NEXT I
22 CALL COLOR(14,14,1)
24 CALL CHAR(136,"FFFFFFFFFFFFFFFF")
26 CALL HCHAR(1,1,136,32)::
     CALL HCHAR(22,1,136,32)
27 CALL VCHAR(1,1,136,22)::
     CALL VCHAR(1,32,136,22)
30 FOR I=1 TO 10::ST(I)=10::NEXT I
50 IF P=1 THEN CL=3::
    DISPLAY AT(24,1)BEEP:"LETTERS"
52 IF P=2 THEN CL=30::
    DISPLAY AT(24,1)BEEP:"       "::
    DISPLAY AT(24,22):"NUMBERS"
59 REM INPUT CHARACTER
60 CALL KEY(0,K,S)::IF S=0 THEN 60
61 IF P=1 AND(K<65 OR K>69)THEN 60
62 IF P=2 AND(K<49 OR K>53)THEN 60
63 IF P=1 THEN SP=K-64 ELSE SP=K-43
65 CALL PATTERN(#SP,42)
67 FOR DE=1 TO 50::NEXT DE
70 CALL POSITION(#SP,X,Y)
99 REM INPUT DIRECTION
100 CALL KEY(0,DI,S)::IF S=0 THEN 100
110 D$=CHR$(DI)::E=ST(SP)::R=0::C=0
120 IF D$="S" AND  Y>30  THEN  R=E*(RND-.5)::
     C=-E
130 IF D$="D"  AND Y<220  THEN R=E*(RND-.5)::
     C=E
140 IF  D$="E"  AND X>30  THEN C=E*(RND-.5)::
     R=-E
150 IF D$="X"  AND X<140 THEN  C=E*(RND-.5)::
     R=E
155 IF R*C=0 THEN 100
199 REM MOVE
200 CALL MOTION(#SP,R,C)
210 FOR DE=1 TO 50
212 CALL KEY(0,FI,S)
214 IF S<>0 THEN 230
220 NEXT DE
230 CALL MOTION(#SP,0,0)
280 IF P=1 THEN J=6 ELSE J=1
270 BB=0
290 FOR I=J TO J+4
295 IF I=SP THEN 340
300 CALL COINC(#SP,#I,10,CK)
305 CALL COINC(#SP,#1,30,BL)
307 CALL POSITION(#I,X1,Y1)
320 IF CK=-1 AND ST(SP)>ST(I)THEN
      CALL LOCATE(#I,180,120+SL)::
      SL=SL+10
324 IF BL<>-1 OR ST(I)<5 OR RND<.5 THEN 340
```

```
325 IF P=1 AND D$="D" AND X<X1 THEN BB=1
327 IF P=2 AND D$="S" AND X>X1 THEN BB=1
340 NEXT I
400 IF P=1 THEN P=2 ELSE P=1
410 CALL PATTERN(#SP,K)
411 ST(SP)=ST(SP)-4::IF ST(SP)<=0 THEN
     ST(SP)=1
415 FOR I=1 TO 10::ST(I)=ST(I)+1
417 IF ST(I)>10 THEN ST(I)=10
418 NEXT I
420 CALL COINC(#SP,#11,10,W)
425 IF W<>-1 OR BB=1 THEN 40
440 CALL POSITION(#11,X,Y)
442 IF D$="S" AND Y<40 THEN GOTO 600
444 IF D$="D" AND Y>210 THEN GOTO 700
446 IF D$="E" AND X<40 THEN 40
448 IF D$="X" AND X>120 THEN 40
450 CALL MOTION(#11,10*R,10*C)
455 FOR DE=1 TO 5*ST(SP)::NEXT DE
460 CALL MOTION(#11,0,0)
500 GOTO 40
600 REM WIN SUBROUTINE
610 CALL CLEAR
620 DISPLAY AT (10,8)BEEP:"NUMBERS WIN!"
630 NU=NU+1:GOTO 800
700 REM WIN SUBROUTINE
710 CALL CLEAR
720 DISPLAY AT (10,8)BEEP:"LETTERS WIN!"
730 LE=LE+1:GOTO 800
800 DISPLAY AT(12,5):"LETTER'S SCORE:";LE
810 DISPLAY AT(14,5):"NUMBER'S SCORE:";NU
812 DISPLAY AT(24,8):"PRESS ANY KEY"
814 FOR DE=1 TO 200:NEXT DE
816 CALL DELSPRITE(ALL)
820 CALL KEY(0,XX,S)::IF S=0 THEN 820
830 GOTO 2
```

Modifications

1. Injured players on the sideline return after three or four moves.
2. Provide a goal keeper.
3. Use a timer and scoring device.
4. Add more players.
5. Implement regulation soccer rules.
6. Allow passing to teammates.

ART AUCTION

Scenario

In this simulation, you will be given an opportunity to buy and sell up to five paintings. The objective is to make a large profit by buying the paintings for as little as possible and selling them for as much as possible.

In order to buy a painting, you must bid against a secret bid made by another buyer (the computer). When a painting is offered for sale, three numbers will be given that represent the mean and range of bids for this particular painting. For example, "200 300 400" indicates that the mean bid price for the painting is 300, and about 70% of the time the price will be between 200 and 400. (Note that higher priced paintings tend to have a larger range of prices.)

After you buy your paintings, you will be given an opportunity to sell them. You will receive from one to five offers, but you do not know in advance how many offers will be made. The offers will be, on the average, 50 higher than the bids made during the buying phase. If you do not accept an offer, and it is the last one, then the offer will be automatically processed. Sometimes it will be wise to accept an offer that is less than the purchase price rather than gamble on a higher offer that does not materialize.

When all of the paintings that you have bought have been sold, you will be given your total profit for all of the transactions.

Sample Run

```
BUY PAINTING  1
PRICES:  546  553  560
YOUR BID?  560
OPPONENT'S BID  565.
YOU WERE OUT BID.

BUY PAINTING  2
PRICES:  336  449  562
YOUR BID?  400
OPPONENT'S BID  440.
YOU WERE OUT BID.

BUY PAINTING  3
PRICES:  213  288  363
YOUR BID?  300
OPPONENT'S BID  324
YOU WERE OUT BID.

BUY PAINTING  4
PRICES:  403  514  625
YOUR BID?  600
OPPONENT'S BID  497.
YOU BOUGHT IT.
```

```
BUY PAINTING  5
PRICES:  274  346  417
YOUR BID?  350
OPPONENT'S BID  311.
YOU BOUGHT IT.

SELL PAINTING  4
YOU BOUGHT IT FOR  600.
AVERAGE OFFER IS  564.
OFFER 1 IS  649.
ACCEPT?  Y

SELL PAINTING  5
YOU BOUGHT IT FOR  350.
AVERAGE OFFER IS  396.
OFFER 1 IS 365.
ACCEPT?  N

YOUR PROFIT IS  64.
PLAY AGAIN?
```

ART AUCTION PROGRAM

Variables

P(5)	Prices
S(5)	Price range
F(5)	Set flag if painting is bought
CB	Opponent's bid
YB	Your bid
I,J,K	Indices
P	Profit
N	Number
D	Dividend
Q	Quotient

Program Listing

```
1 REM *** ART AUCTION ***
5 REM  SET PRICES AND RANGES
10 DIM P(5),S(5),F(5)
15 PR=0
20 FOR I=1 TO 5
25 RANDOMIZE
30 P(I)=100+INT(900*RND)
40 S(I)=INT(P(I)*RND)
50 IF P(I)<500 THEN 55 ELSE 60
55 S(I)=INT(P(I)*.7*RND)
60 F(I)=0
70 NEXT I
95 REM  BUY PAINTINGS
96 CALL CLEAR
100 FOR I=1 TO 5
110 GOSUB 500
115 PRINT ::
120 PRINT "BUY PAINTING ";I
125 PRINT
130 PRINT "PRICES";
     INT(P(I)-.5*S(I));P(I);INT(P(I)+.5*S(I))
140 INPUT "YOUR BID? ":YB
150 PRINT "OPPONENT'S BID";CB;
160 IF YB>CB THEN 161 ELSE 170
161 PRINT "YOU BOUGHT IT."
162 F(I)=YB
163 GOTO 180
170 PRINT "YOU WERE OUT BID."
180 NEXT I
195 REM  SELL PAINTINGS
196 PRINT :::
200 FOR I=1 TO 5
210 IF F(I)=0 THEN 310
220 FOR K=1 TO INT(5*RND+1)
230 GOSUB 500
235 PRINT
```

```
240 PRINT "SELL PAINTING";I
250 PRINT "YOU BOUGHT IT FOR";F(I)
252 PRINT "AVERAGE OFFER IS";P(I)+50
260 PRINT "OFFER";K;"IS";CB
270 INPUT "ACCEPT? ":Y$
280 IF Y$="Y" THEN 300
290 NEXT K
300 PR=PR+CB-F(I)
305 PRINT
310 NEXT I
315 PRINT
320 PRINT "YOUR PROFIT IS";PR;
325 PRINT
330 INPUT "PLAY AGAIN? ":Y$
340 IF Y$="Y" THEN 15
350 END
495 REM  NORMAL DISTRIBUTION SUBROUTINE
500 D=0
510 N=INT(65536*RND)
520 FOR J=1 TO 16
530 Q=INT(N/2)
540 D=D+2*(N/2-Q)
550 N=Q
560 NEXT J
570 CB=P(I)+S(I)*(D-8)/8
580 CB=CB+20*RND
590 CB=INT(CB)
600 RETURN
```

ART AUCTION MODIFICATIONS

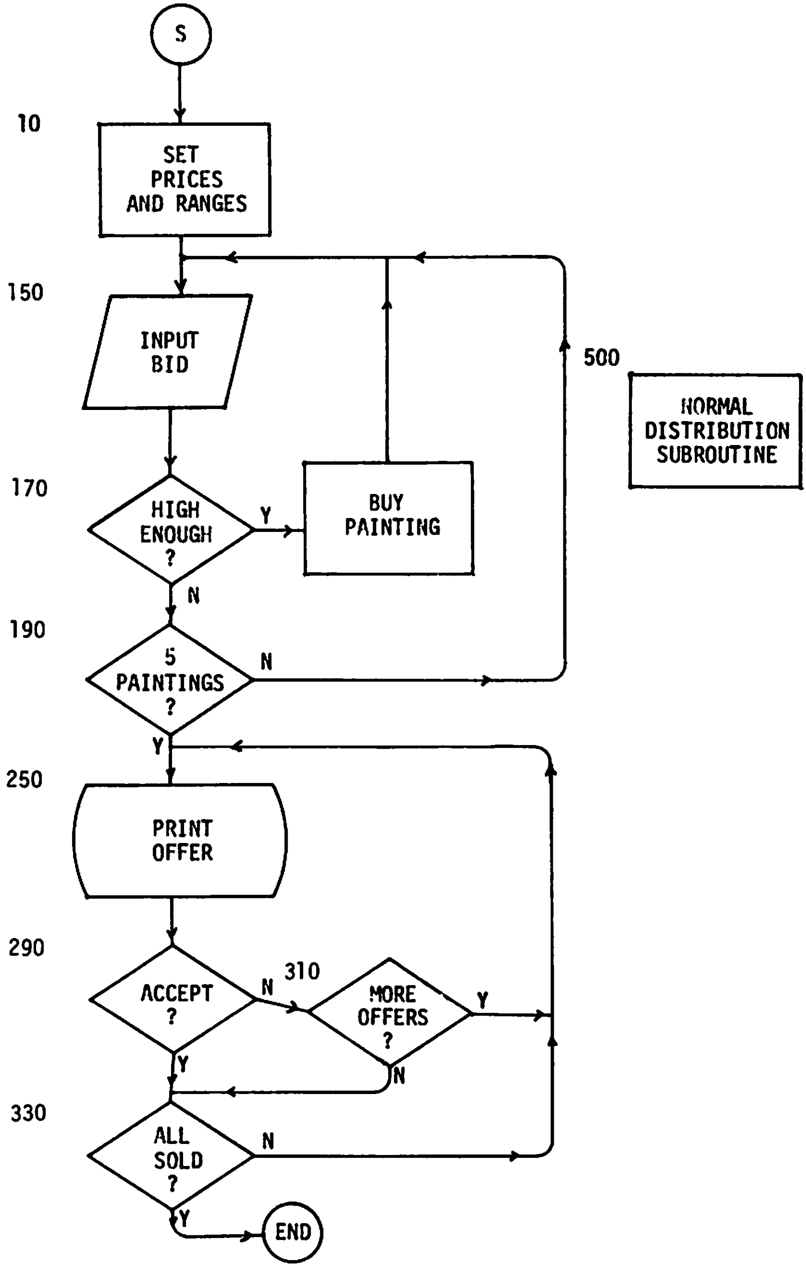
Minor

1. Number of paintings -- lines 10, 20, 100, 200
2. Starting prices -- line 30
3. Price spread -- lines 40, 50
4. Built-in profit -- lines 252, 300
5. Error in price range -- line 580
6. Number of offers -- line 220

Major

1. Have one or more of the paintings a forgery that is worth nothing.
2. Have one or more of the paintings that have a low purchase price be very valuable.
3. Have more opponents bid against you.

ART AUCTION FLOWCHART

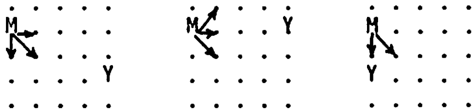


MONSTER CHASE

Scenario

In this simulation you are locked in a cage with a hungry monster who has a life span of ten turns. Your movement and that of the monster takes place on a 5X5 grid. You may move north, east, south, or west by entering N, E, S, or W. If you enter any other letter, you will remain in the same place.

The monster is programmed to move along one of the arrows toward you as shown below :



Your only means of survival is to outwit the monster for ten turns.

Sample Run

```
M . . . .
. . . . .
. . . . .
. . . . .
. . . . Y
MOVE NUMBER 1
DIRECTION
(NESWO)? W

. . . . .
. M . . .
. . . . .
. . . . .
. . . Y .
MOVE NUMBER 2
DIRECTION
(NESWO)? N

. . . . .
. . . . .
. . M . .
. . . Y .
. . . . .
MOVE NUMBER 3
DIRECTION
(NESWO)? S
```

```
. . . . .
. . . . .
. . . M .
. . . . .
. . . Y .
MOVE NUMBER 4
DIRECTION
(NESWO)? W

. . . . .
. . . . .
. . . . .
. . . M .
. . Y . .
MOVE NUMBER 5
DIRECTION
(NESWO)? W

. . . . .
. . . . .
. . . . .
. . . . .
. Y . M .
MOVE NUMBER 6
DIRECTION
NESWO)? N
```

```
. . . . .
. . . . .
. . . . .
. Y M . .
. . . . .
MOVE NUMBER 7
DIRECTION
(NESWO)? W

. . . . .
. . . . .
Y M . . .
. . . . .
. . . . .
MOVE NUMBER 8
DIRECTION
(NESWO)? N

EATEN
PLAY AGAIN?
```

MONSTER CHASE PROGRAM

Variables

R,C	Your row and column
X,Y	Monster's row and column
L,M	Temporary variables
M$	Your move (N,E,S,W,0)
D	Direction of the monster (1-8)
T	Turns (1-10)

Listing

```
1 REM *** MONSTER CHASE ***
5 REM  SET CONDITIOONS
6 CALL CHAR(144,"1818FFBDBD2424E7")
7 CALL COLOR(15,3,12)
8 CALL CHAR(152,"00240018007E4200")
9 CALL COLOR(16,2,10)
10 X=1
15 Y=1
20 R=5
25 C=5
35 REM  GRID DISPLAY
37 CALL CLEAR
38 CALL SCREEN(12)
40 FOR I=1 TO 5
50 FOR J=1 TO 5
60 CALL HCHAR(5*I-4,5*J+2,46)
70 NEXT J
80 NEXT I
90 PRINT "MOVE:";TAB(10);"DIRECTION NESWO:";
100 CALL HCHAR(5*R-4,5*C+2,144)
110 CALL HCHAR(5*X-4,5*Y+2,152)
200 FOR T=49 TO 58
202 CALL SOUND(300,R*120,1,C*120,1)
205 CALL HCHAR(24,8,T)
210 CALL KEY(3,KEY,ST)
220 IF ST=0 THEN 210
225 CALL HCHAR(24,28,KEY)
230 CALL HCHAR(5*R-4,5*C+2,46)
235 M$=CHR$(KEY)
240 IF M$="N" THEN 242 ELSE 250
242 R=R-1
250 IF M$="E" THEN 252 ELSE 260
252 C=C+1
260 IF M$="S" THEN 262 ELSE 270
262 R=R+1
270 IF M$="W" THEN 272 ELSE 280
272 C=C-1
280 IF (R*C=0)+(R>5)+(C>5)THEN 282 ELSE 285
282 CALL CLEAR
283 PRINT "OUT OF BOUNDS"
```

```
284 GOTO 520
285 CALL HCHAR(5*R-4,5*C+2,144)
287 CALL HCHAR(5*X-4,5*Y+2,46)
290 IF (R=X)*(Y=C)THEN 292 ELSE 300
292 CALL CLEAR
293 PRINT "EATEN"
294 GOTO 520
300 IF (X=R)*(Y<C)THEN 302 ELSE 310
302 D=1
310 IF (X>R)*(Y<C)THEN 312 ELSE 320
312 D=2
320 IF (X>R)*(Y=C)THEN 322 ELSE 330
322 D=3
330 IF (X>R)*(Y>C)THEN 332 ELSE 340
332 D=4
340 IF (X=R)*(Y>C)THEN 342 ELSE 350
342 D=5
350 IF (X<R)*(Y>C)THEN 352 ELSE 360
352 D=6
360 IF (X<R)*(Y=C)THEN 362 ELSE 370
362 D=7
370 IF (X<R)*(Y<C)THEN 372 ELSE 380
372 D=8
380 D=D+INT(3*RND-1)
390 IF D=0 THEN 392 ELSE 400
392 D=8
394 GOTO 410
400 IF D=9 THEN 402 ELSE 410
402 D=1
410 IF (D>1)*(D<5)THEN 412 ELSE 420
412 X=X-1
420 IF D>5 THEN 422 ELSE 430
422 X=X+1
430 IF (D>3)*(D<7)THEN 432 ELSE 440
432 Y=Y-1
440 IF (D<3)+(D=8)THEN 442 ELSE 450
442 Y=Y+1
450 IF X=0 THEN 452 ELSE 460
452 X=X+1
460 IF Y=0 THEN 462 ELSE 470
462 Y=Y+1
470 IF X=6 THEN 472 ELSE 480
472 X=X-1
480 IF Y=6 THEN 482 ELSE 490
482 Y=Y-1
484 IF (X<1)+(X>5)+(Y<1)+(Y>5)THEN 486 ELSE 490
486 CALL CLEAR
488 PRINT "MONSTER OUT OF BOUNDS"
489 GOTO 520
490 IF (X=R)*(Y=C)THEN 492 ELSE 500
492 CALL CLEAR
494 PRINT "EATEN"
496 GOTO 520
500 CALL HCHAR(5*X-4,5*Y+2,152)
```

```
501 CALL SOUND(104,120*X,5,Y*120,5)
502 NEXT T
505 CALL CLEAR
510 PRINT "YOU SURVIVED! ";
520 INPUT "PLAY AGAIN? ":Y$
530 IF Y$="Y" THEN 5
540 END
```

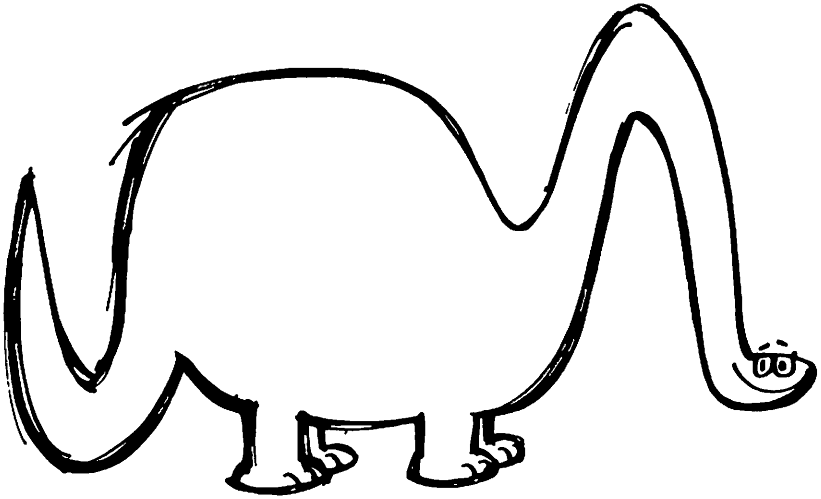
MONSTER CHASE MODIFICATIONS

Minor

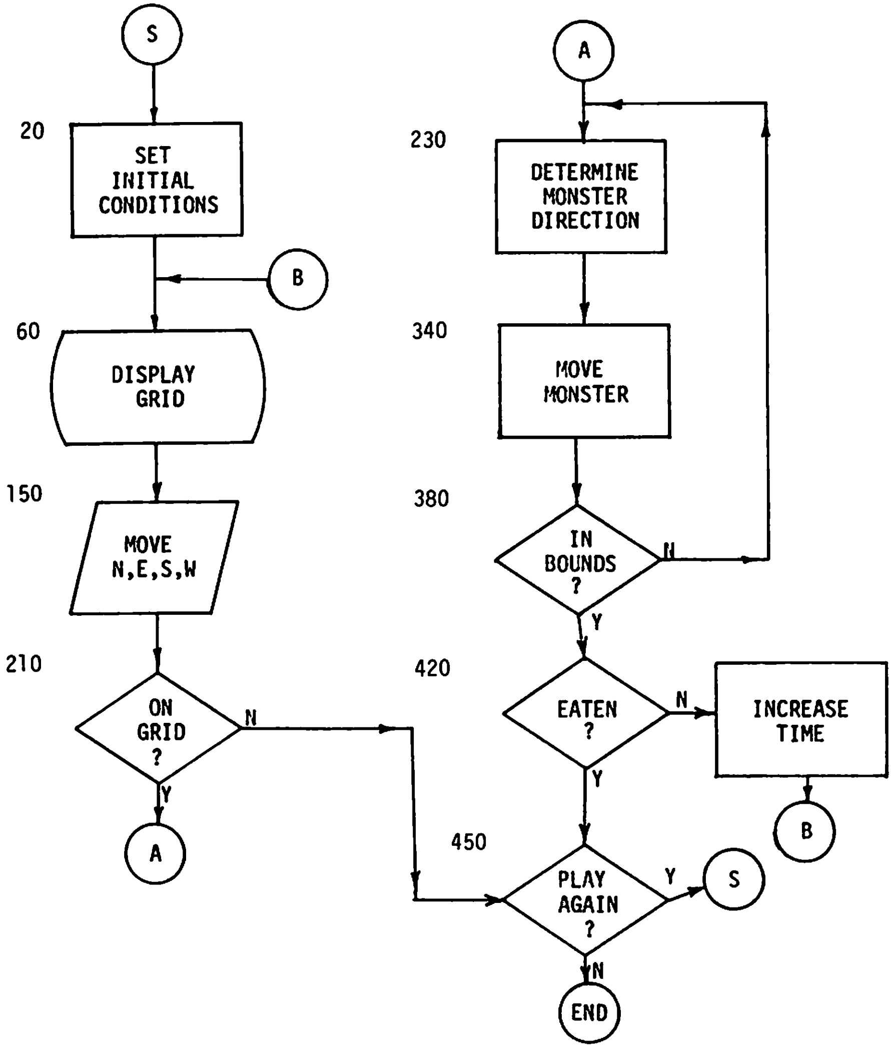
1. Grid size -- lines 20, 25, 40, 50
2. Turns to win -- line 200
3. Change colors -- lines 7, 9
4. Add RANDOMIZE in line 2

Major

1. Have more than one monster.
2. Chase a little monster while a big monster tries to get you.
3. Have the monster fall into quicksand.
4. Require food in order to maintain energy.



MONSTER CHASE FLOWCHART



LOST TREASURE

Scenario

You have landed somewhere on an island that has treasure, woods, mountains, a cave, a bluff, an oak tree, and, of course, sea water all around. Your objective is to find the treasure as quickly as possible without falling into the shark-infested water.

You can move north (N), east (E), south (S), or west (W) one square at a time. Your compass, however, is not very accurate. There is only an 80% chance that you will move in the intended direction. There is a 20% chance you will move diagonally to the left or to the right. Each time that you move you will receive feedback regarding the type of terrain on which you are traveling.

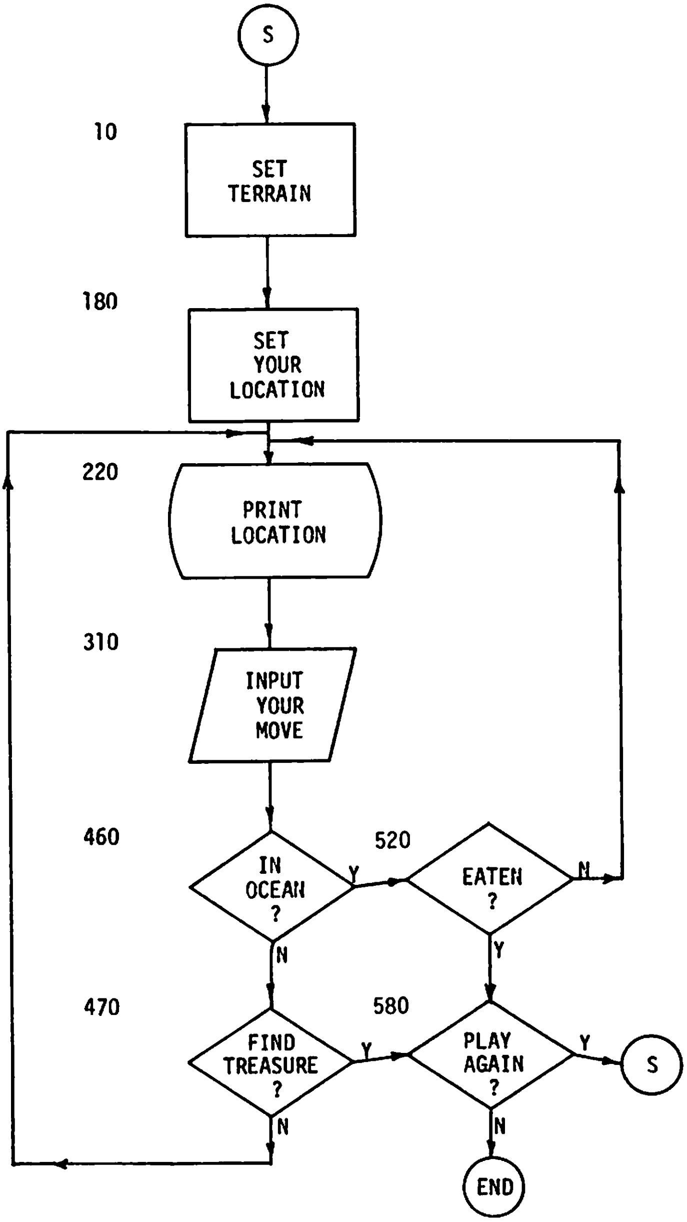
If you fall into the sea, you will be placed back on the square occupied prior to your unfortunate move, unless you disturb the sharks. The chance that the sharks will eat you the first time you fall in is 20%. The second time you fall in the chance of being eaten is 70%. The third time you fall in will be your last!

Since you have a map of the island, you will be able to determine your approximate position. For example, if you are in the woods and you move east two squares and find that you are in mountains, then you are most likely located in the north-east corner of the island. The reason you can't be sure of the exact location is that you may have veered off to the right or left. With practice, you should be able to find the treasure in less than fifteen moves.

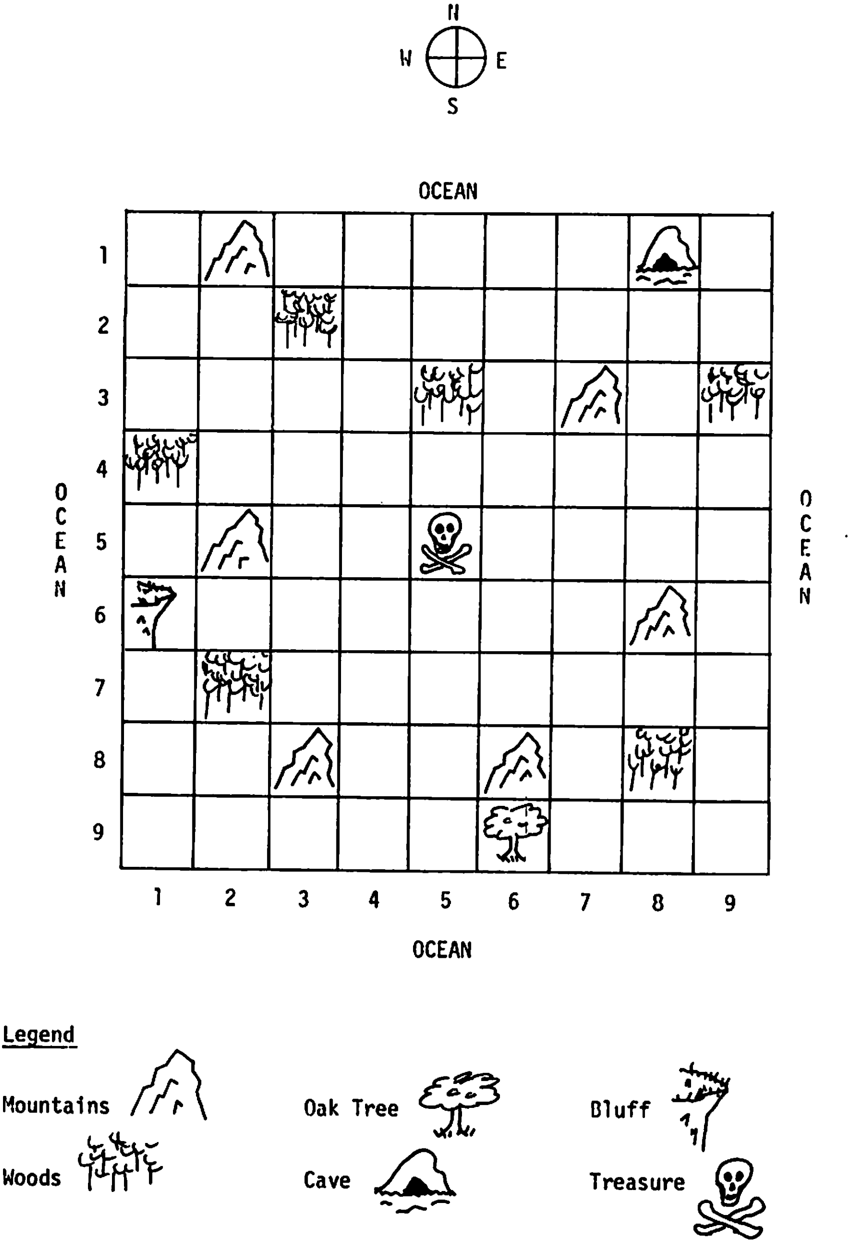
Sample Run

```
RUN

YOU ARE IN THE CLEAR.
MOVE(NESW)?  S
YOU FELL IN THE OCEAN.
EATEN BY SHARKS!
PLAY AGAIN? Y

YOU ARE IN THE CLEAR.
MOVE(NESW)?  S

YOU ARE IN THE WOODS.
MOVE(NESW)?  N

.
.
.

.
.

YOU ARE IN THE MOUNTAINS.
MOVE(NESW)?  E

.
.
.

YOU ARE IN THE WOODS.
MOVE(NESW)?  S

.
.
.

YOU ARE IN THE CLEAR.
MOVE(NESW)?  E
YOU FOUND THE TREASURE
IN 9 MOVES.
PLAY AGAIN?
```

LOST TREASURE FLOWCHART



LOST TREASURE MAP



LOST TREASURE PROGRAM

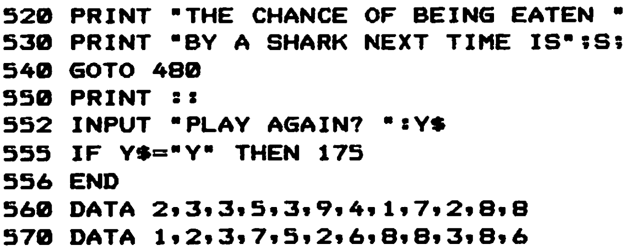
Variables

L(R,C)	Locations
S	Probability of being eaten by shark
R	Your row
C	Your column
RT, CT	Temporary storage
T	Number of turns

Listing

```
5 REM  SET TERRAIN
6 RANDOMIZE
7 CALL CLEAR
10 DIM L(9,9)
20 S=.2
30 FOR I=1 TO 9
35 FOR J=1 TO 9
40 L(I,J)=0
50 NEXT J
55 NEXT I
60 FOR I=1 TO 6
70 READ R,C
80 L(R,C)=1
90 NEXT I
100 FOR I=1 TO 6
110 READ R,C
120 L(R,C)=2
130 NEXT I
140 L(1,8)=3
150 L(6,1)=4
160 L(9,6)=5
170 L(5,5)=6
175 REM  YOUR LOCATION
180 R=INT(9*RND+1)
190 C=INT(9*RND+1)
200 IF SQR((R-5↑2+(C-5)↑2)<2 THEN 180
205 REM  START MAIN LOOP
210 FOR T=1 TO 100
215 PRINT ::
220 PRINT "YOU ARE ";
230 J=L(R,C)+1
240 ON J GOSUB 250,260,270,280,290,300
245 GOTO 310
250 PRINT "IN THE CLEAR."
255 RETURN
260 PRINT "IN THE WOODS."
265 RETURN
270 PRINT "IN THE MOUNTAINS ."
275 RETURN
280 PRINT "IN A CAVE."
```

```
285 RETURN
290 PRINT "ON A BLUFF."
295 RETURN
300 PRINT "NEAR AN OAK TREE."
305 RETURN
310 INPUT "MOVE(NESW)? ":M$
320 RT=R
325 CT=C
330 IF M$="N" THEN 334 ELSE 340
334 R=R-1
335 GOSUB 380
340 IF M$="E" THEN 344 ELSE 350
344 C=C+1
345 GOSUB 420
350 IF M$="W" THEN 354 ELSE 360
354 C=C-1
355 GOSUB 420
360 IF M$="S" THEN 364 ELSE 370
364 R=R+1
365 GOSUB 380
370 GOTO 460
375 REM  MOVE SUBROUTINE
380 J=INT(10*RND+1)
390 IF J>2 THEN 395 ELSE 400
395 RETURN
400 IF J<>1 THEN 410
405 C=C+1
406 RETURN
410 C=C-1
415 RETURN
420 J=INT(10*RND+1)
430 IF J>2 THEN 435 ELSE 440
435 RETURN
440 IF J<>1 THEN 450
445 R=R+1
446 RETURN
450 R+R-1
454 RETURN
455 REM  IN OCEAN,FOUND TREASURE?
460 IF (R<1)+(R>9)+(C<1)+(C>9)THEN 490 ELSE 470
470 IF L(R,C)=6 THEN 475 ELSE 480
475 PRINT "YOU FOUND TREASURE IN";T
476 GOTO 550
480 NEXT T
490 PRINT "YOU FELL INTO THE OCEAN"
500 IF RND<S THEN 505 ELSE 510
505 PRINT "EATEN BY SHARKS!"
506 GOTO 550
510 S=S+.5
511 R=RT
512 C=CT
513 IF S>1 THEN 514 ELSE 520
514 S=1
```



```
520 PRINT "THE CHANCE OF BEING EATEN "
530 PRINT "BY A SHARK NEXT TIME IS";S;
540 GOTO 480
550 PRINT ::
552 INPUT "PLAY AGAIN? ":Y$
555 IF Y$="Y" THEN 175
556 END
560 DATA 2,3,3,5,3,9,4,1,7,2,8,8
570 DATA 1,2,3,7,5,2,6,8,8,3,8,6
```

LOST TREASURE MODIFICATIONS

Minor

1. Probability of first shark attack -- line 20
2. Grid size -- lines 30, 135, 180, 190, 460
3. Number of woods -- lines 60, 560
4. Number of mountains -- lines 100, 570
5. Landmarks' locations -- lines 140, 150, 160
6. Location of the treasure -- line 170
7. Movement error -- lines 380, 420
8. Amount you disturb shark -- line 510

Major

1. Vary number and amount of treasure.
2. Add parameters of water and/or food to maintain your energy level.
3. Hunt a moving treasure.
4. Modify direction of movement.
5. Add quicksand.
6. Include landmarks placed at random that are not on the map.
7. Randomly place treasure before each hunt.

GONE FISHING

You are going on a fishing trip. The sea is an 8X8 grid, forming 64 fishing locations. You will start at the dock, square (1,1), and try to catch as many pounds of fish as you can. You may move one square at a time horizontally or vertically by entering a north(N), south(S), east(E), or west(W). Entering an F allows you to fish in the same place again, and a B allows you to start another fishing trip immediately. If you select a direction that takes you off the grid, your ship will sink. You must return to the dock in sixty moves, which is equivalent to six hours. If you don't return in time, half of your catch will spoil.

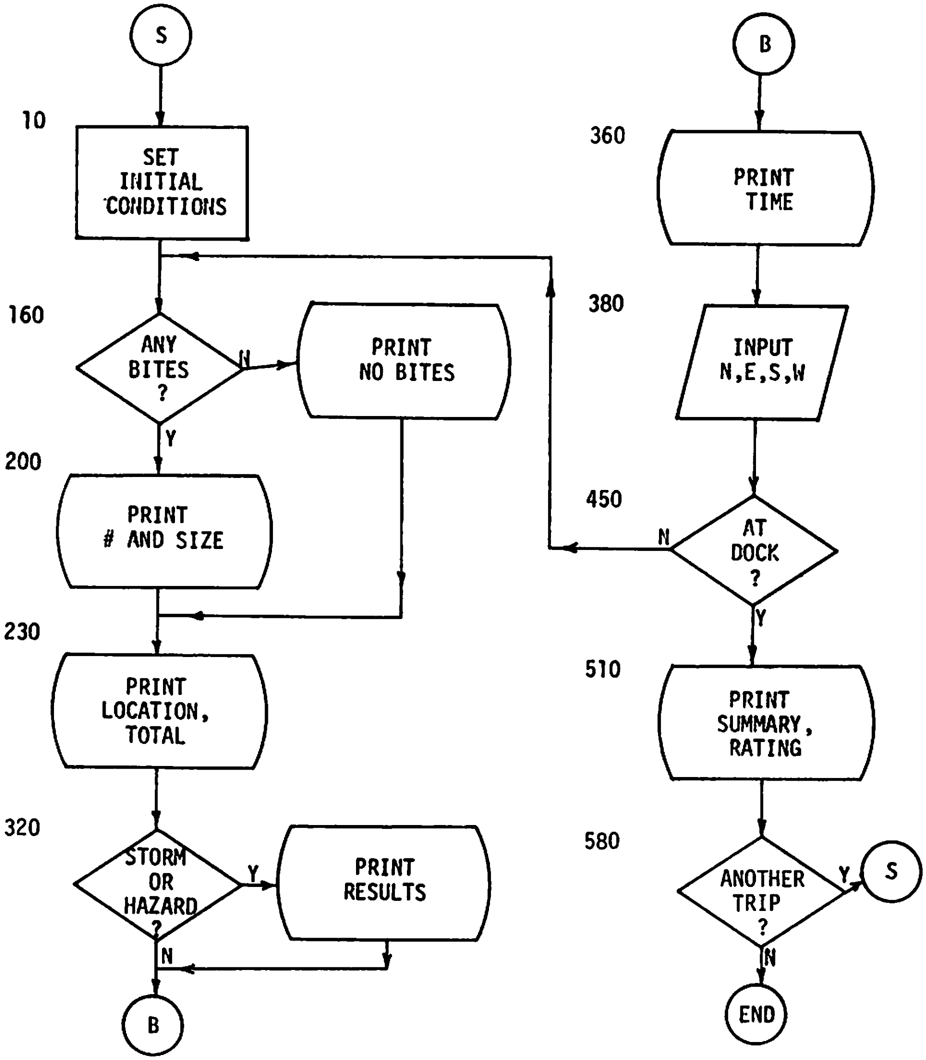
The chance of catching fish is different for each square and is determined at the beginning of the trip. The chance of catching fish in a given square will remain the same throughout the trip or will decrease if the fish are scared by a shark. The maximum number of fish that can be caught in each square (density) is also determined at the beginning of the simulation. This number varies from 1 to 5. The maximum number of fish you can catch in a square will decrease only if sea gulls eat some of the bait. The maximum weight of a fish in a particular square is the product of the row and column; therefore, the further out you go, the bigger the fish.

The longer you fish, the greater the chance of an afternoon storm occurring. If you hit a storm, you will lose .5 hour. One of the more difficult manuvers of the trip is to fish as long as necessary to accumulate a large catch without getting lost in a storm. Also, there is a 4% chance that you will experience some unexpected event during each move of the trip. Be sure you return to the dock before six hours have elapsed. Your rating as a fisherman will be the number of pounds of fish you catch divided by five.

You may wish to use the fishing grid on page 39 to record the best fishing spots. A small marker can be used to keep track of your location on the grid.

```
RUN

NO BITES
AT LOCATION  1  1
TOTAL LBS. THIS
TRIP IS 0.
YOU HAVE FISHED
FOR 0 HOURS.
MOVE(N,S,E,W,F,B)?  E

NO BITES
AT LOCATION  1  2
TOTAL LBS. THIS
TRIP IS 0.
YOU HAVE FISHED
FOR .1 HOURS.
MOVE(N,S,E,W,F,B)?  S

YOU CAUGHT 1 FISH,
EACH WEIGHING 2 LBS.
AT LOCATION  2  2
TOTAL LBS. THIS
TRIP IS 2.
YOU HAVE FISHED
FOR .2 HOURS.
MOVE(N,S,E,W,F,B)?  S

NO BITES
AT LOCATION  3  2
TOTAL LBS. THIS
TRIP IS 2.
YOU HAVE FISHED
FOR .3 HOURS
MOVE(N,S,E,W,F,B)?  E

YOU CAUGHT 4 FISH,
EACH WEIGHING 2 LBS.
AT LOCATION  3  3
TOTAL LBS. THIS
TRIP IS 10.
YOU HAVE FISHED
FOR .4 HOURS.
MOVE(N,S,E,W,F,B)?  E

.
.
.
NO BITES
AT LOCATION  4  6
TOTAL LBS. THIS
TRIP IS 10.
SEA GULLS ATE SOME
OF YOUR BAIT.
CATCH WILL BE SMALLER
THIS TRIP.
YOU HAVE FISHED
FOR .8 HOURS.
MOVE(N,S,E,W,F,B)?  S

            .
.           .
.
.

YOU CAUGHT 4 FISH,
EACH WEIGHING 15 LBS.
AT LOCATION  4  8
TOTAL LBS. THIS
TRIP IS 155.
YOU CAUGHT A
50 LB. SHARK.
TOTAL LBS. THIS
TRIP IS 205.
YOU HAVE FISHED
FOR 1.8 HOURS.
MOVE(N,S,E,W,F,B)?  W

.
.
.
YOU CAUGHT 1 FISH,
EACH WEIGHING 3 LBS.
AT LOCATION  3  3
TOTAL LBS. THIS
TRIP IS 208.
WATER SPOUT
DISPLACES YOU.
NOW AT LOCATION  4  5
YOU HAVE FISHED
FOR 2.6 HOURS.
MOVE(N,S,E,W,F,B)?  W

.
.
.

NO BITES
AT LOCATION  1  2
TOTAL LBS. THIS
TRIP IS 211.
YOU HAVE FISHED
FOR 3.2 HOURS
MOVE(N,S,E,W,F,B)?  W

YOU ARE BACK AT THE
DOCK AFTER 3.2 HOURS
OF FISHING.
CLEAN 211 LBS.
OF FISH.
YOU RATE 42
AS A FISHERMAN.
ANOTHER TRIP?
```

GONE FISHING FLOWCHART



GONE FISHING PROGRAM

Variables

P(I,J)	The probability of catching a fish
D(I,J)	The maximum number of fish in square (I,J), from 1 to 5
W	Weight of each fish caught, from 1 to RXC
P	The total number of pounds of fish caught at a given time
R	Row in which you are fishing
C	Column in which you are fishing
N	Number of fish caught in a given turn
T	Time in tenths of an hour, maximum 6 hours
M$	Move(N,E,S,W,F,B), where N,E,S, and W are directions, F allows you to fish again in the same square, and B allows you to start the fishing trip over again

Listing

```
1 REM *** GONE FISHING ***
5 REM  SET PROBABILITIES AND DENSITY
6 DIM P(8,8), D(8,8)
7 RANDOMIZE
10 CALL CLEAR
20 FOR I=1 TO 8
22 FOR J=1 TO 8
30 P(I,J)=.7*RND
40 D(I,J)=INT(RND*5+1)
50 NEXT J
55 NEXT I
60 P(1,1)=0
62 LB=0
63 R=1
64 C=1
145 REM  MAIN LOOP
150 FOR T=0 TO 6 STEP .1
155 PRINT
160 IF (RND>P(R,C))+(D(R,C)<1)THEN 162 ELSE 170
162 PRINT "NO BITES"
163 GOTO 220
170 N=INT(RND*D(R,C)+1)
180 W=INT(RND*R*C)+1
190 LB=LB+N*W
195 PRINT
200 PRINT "YOU CAUGHT";N;"FISH,"
210 PRINT "EACH WEIGHING";W;"LBS.,"
220 PRINT "AT LOCATION";R;C
230 PRINT "TOTAL LBS. THIS TRIP IS";LB;"."
325 REM  UNEXPECTED EXPERIENCES
330 IF RND<T/60 THEN 335 ELSE 340
335 PRINT "STORM--LOST 1/2 HOUR"
337 T=T+.5
340 J=INT(100*RND)+1
350 IF J>4 THEN 370
360 ON J GOSUB 600,700,800,900
```

```
370 PRINT "YOU HAVE FISHED FOR";T;"HOURS."
380 INPUT "MOVE(N,S,E,W,F,B)? ":M$
390 IF M$="E" THEN 395 ELSE 400
395 C=C+1
400 IF M$="N" THEN 405 ELSE 410
405 R=R-1
410 IF M$="W" THEN 415 ELSE 420
415 C=C-1
420 IF M$="S" THEN 425 ELSE 430
425 R=R+1
430 IF M$="B" THEN 435 ELSE 440
435 GOTO 5
440 IF (R<1)+(R>8)+(C<1)+(C>8)THEN 445 ELSE 450
445 PRINT "GROUNDED--SUNK!"
446 GOTO 550
450 IF (R=1)*(C=1)THEN 500 ELSE 460
460 NEXT T
470 PRINT "TIME UP. THE SUN HAS SET."
480 PRINT "HALF OF YOUR CATCH HAS SPOILED."
490 LB=LB/2
495 REM  SUMMARY OF TRIP
500 IF T=0 THEN 505 ELSE 510
505 PRINT "STILL AT DOCK"
506 GOTO 10
510 PRINT "YOU ARE BACK AT THE DOCK"
520 PRINT "AFTER";T;"HOURS OF FISHING."
530 PRINT "CLEAN";LB;"LBS. OF FISH."
540 PRINT "YOU RATE";INT(LB/5);"AS A FISHERMAN."
550 PRINT ::
552 INPUT "ANOTHER TRIP? " : X$
555 IF X$="Y" THEN 7
560 END
595 REM  SUBROUTINE
600 IF R+C<9 THEN 660
610 PRINT "FISH SCARED BY SHARK."
620 PRINT "NOT BITING AS OFTEN."
630 FOR I=1 TO 8
635 FOR J=1 TO 8
640 P(I,J)=P(I,J)-.1
650 NEXT J
655 NEXT I
660 RETURN
700 PRINT "SEA GULLS ATE SOME OF YOUR BAIT."
710 PRINT "CATCH WILL BE SMALLER THIS TRIP."
720 FOR I=1 TO 8
725 FOR J=1 TO 8
730 D(I,J)=D(I,J)-1
740 NEXT J
745 NEXT I
750 RETURN
800 PRINT "WATER SPOUT DISPLACES YOU."
810 R=INT(8*RND+1)
820 C=INT(8*RND+1)
```

```
830 PRINT "YOU ARE NOW AT LOCATION";R;C
840 T=T+.2
850 RETURN
900 PRINT "YOU CAUGHT A 50 LB.SHARK."
910 LB=LB+50
920 PRINT "TOTAL LBS. THIS TRIP IS";LB;"."
930 RETURN
```

GONE FISHING MODIFICATIONS

Minor

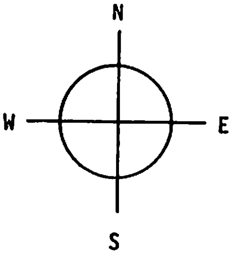
1. Grid size -- lines 20, 22, 440, 630, 635, 720, 725, 810, 820
2. Maximum probability of catching fish in a square -- line 30
3. Maximum density of fish in a square -- line 40
4. Maximum time of fishing -- line 150
5. Storm probability -- line 330
6. Rating scale -- line 540

Major

1. Catch different kinds of fish, such as, sharks, whales, or mermaids.
2. Change the goal to catching the biggest fish.
3. Use fuel to run the boat.
4. Add a choice of hook sizes and fishing depth.
5. Add different kinds of hazards, such as whales, reefs, UFO's.
6. Let fishing success depend on time of day.
7. Fix weather conditions and fishing conditions at the beginning of the trip.
8. Utilize sonar devices to help locate fish.
9. Allow ship to move in a diagonal direction.

FISHING MAP

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



SPACE FLIGHT

Scenario

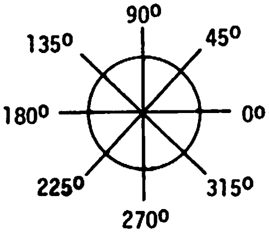
In this simulation, you are living in the year 2062 as the captain of a space ship. Your orders are to deliver medical supplies from Alpha at coordinates (10,10) to Beta at coordinates (80,80). Your rating as a space pilot will depend upon how fast you can make the trip.

During each time interval, you will be able to determine the following information:

1. Total time elapsed
2. Location in terms of X and Y coordinates
3. Amount of fuel left
4. Speed
5. The angle at which you are moving
6. Your distance from the planet.

To change direction or to increase or decrease speed, you can fire one of two kinds of rockets: main (M) and half (H). These rockets take one unit and 1/2 unit of fuel, respectively. A "C" will allow you to coast for five time intervals.

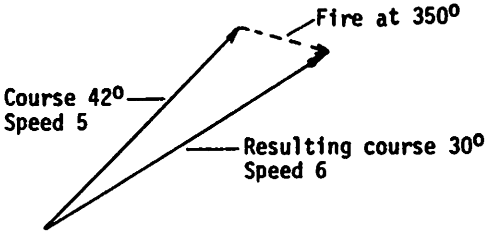
Once you decide how much fuel you are going to burn, you must decide on the direction in which you will be firing the rockets. You are able to rotate your space ship with small thrusters as it drifts in space. The directions are shown below:



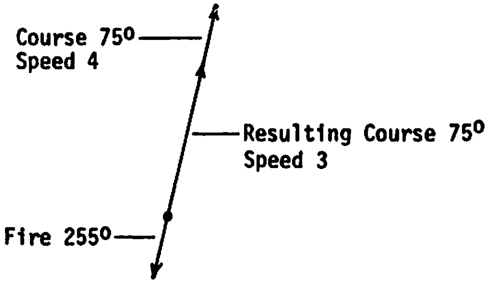
Once you fire your main rocket for three or four turns to increase your speed, you can conserve fuel by drifting through space. You must start to fire in the opposite direction to slow down before arriving at Beta. In order to meet arrival conditions, you must be within a distance of one and at a speed of less than one.

You may wish to make copies of the grid at the end of this section to aid in plotting your course. If you find that you are off course, you may have to fire a "correction" rocket. In order to estimate the angle of firing, you can use a force diagram as shown below.

Example 1: Correction



Example 2: Retrofire

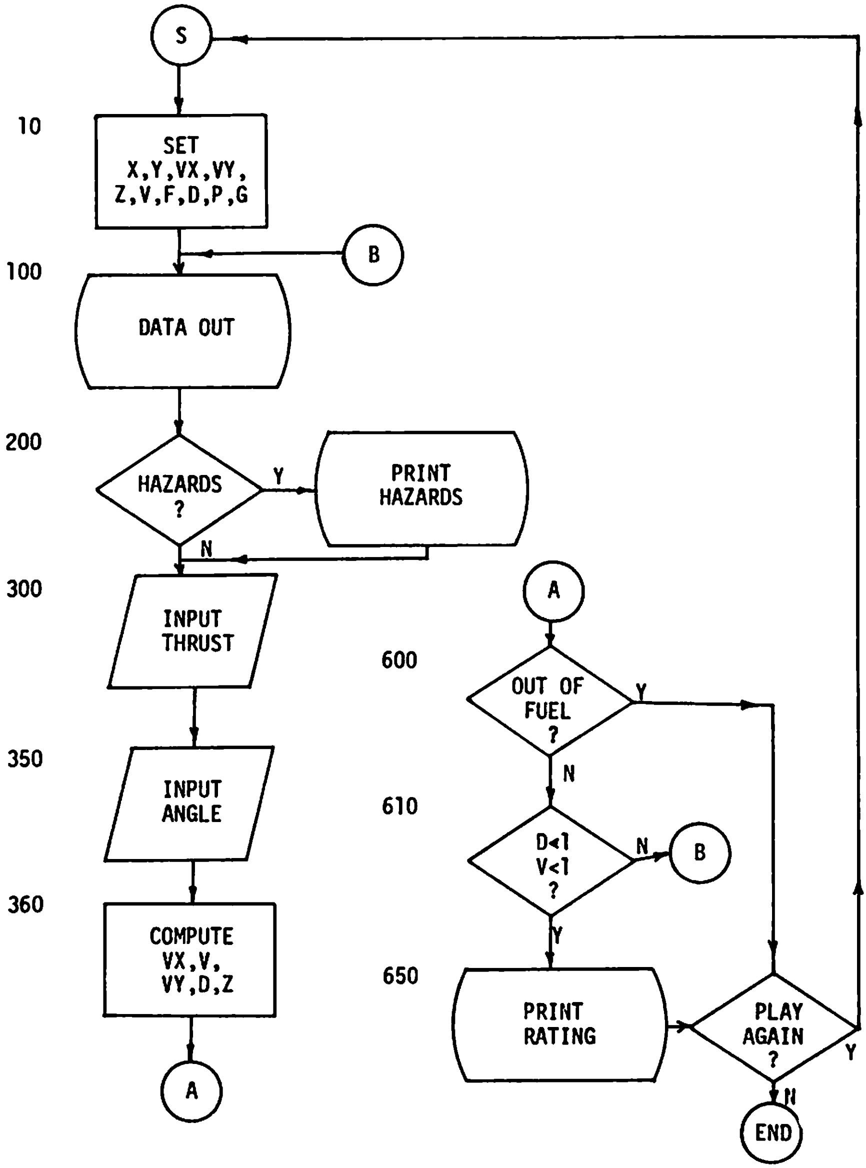


Sample Run

```
    DATA READOUT
0 HOURS   10 LITERS
LOC:  10  10
VELOCITY:  0
0 DEGREES
DISTANCE:  98.995
COMMAND(O,M,H,C)?  M
ANGLE?  45

    DATA READOUT
.01 HOURS   9 LITERS
LOC:  10.6776  10.67
VELOCITY:  .9529
45 DEGREES
DISTANCE:  98.942

        .
        .
        .
```

```
    DATA READOUT
.05 HOURS   5 LITERS
LOC:  20.1487  20.8211
VELOCITY:  5.0035
50 DEGREES
DISTANCE:  84.1685
PROBLEMS: SUPPORT SYSTEM
COMMAND(O,M,H,C)?  O

        .
        .
        .

    DATA READOUT
.33 HOURS   1 LITERS
LOC:  79.1844  81.0019
VELOCITY:  .0231
58 DEGREES
DISTANCE:  1.2918
COMMAND(O,M,H,C)?  H
ANGLE?  315
ARRIVED
TRIP TOOK .33 HOURS.
YOUR RATING IS 66.
PLAY AGAIN?  N
READY
```

SPACE FLIGHT FLOWCHART



SPACE FLIGHT PROGRAM

Variables

X,Y	Location
VX,VY	Speed
Z	Angle of coast
V	Velocity
T	Time
D	Distance to planet
J	Index for hazards
F	Fuel
A	Angle input
L,M	Temporary Variables
R	Rating
F1	Coast count
G	Accuracy of gyros

Listing

```
1 REM *** SPACE FLIGHT ***
5 CALL CLEAR
7 RANDOMIZE
10 X=10
12 Y=10
14 F=10
16 D=98.995
18 P=3.1416
20 G=1
30 FOR T=0 TO 10 STEP .01
100 PRINT "DATA READOUT:"
105 PRINT
110 PRINT T;"HOURS";TAB(15);F;"LITERS"
120 PRINT "LOC X    :";X
122 PRINT "LOC Y    :";Y
125 PRINT "VELOCITY:";V
130 PRINT "DEGREES :";Z
140 PRINT "DISTANCE:";D
142 PRINT :::
200 J=INT(50*RND+1)
210 IF J<6 THEN 215 ELSE 290
215 PRINT "PROBLEMS"
220 ON J GOSUB 230,240,250,260,270
222 PRINT
225 GOTO 290
230 PRINT "GYROS ANGLE ERROR"
232 G=G+1
234 RETURN
240 PRINT "FUEL LINE"
242 F=F-.5
244 RETURN
250 PRINT "LIFE SUPPORT"
252 T=T+.05
254 RETURN
260 PRINT "ALIENS"
```

```
262 VX=0
264 VY=0
266 RETURN
270 PRINT "METEORS"
272 VX=VX+RND-.5
274 VY=VY+RND-.5
280 RETURN
290 IF F1>0 THEN 295 ELSE 300
295 F1=F1-1
296 GOTO 450
300 INPUT "COMMAND (O,M,H,C)? ":C$
310 IF C$="M" THEN 312 ELSE 320
312 B=1
314 GOTO 350
320 IF C$="H" THEN 322 ELSE 330
322 B=2
324 GOTO 350
330 IF C$="C" THEN 332 ELSE 340
332 F1=5
340 GOTO 450
350 INPUT "ANGLE? ":A
352 A=A+(20*G*RND-10*G)
360 A=A*P/180
370 L=COS(A)
372 M=SIN(A)
374 F=F-1/B
380 VX=VX+(1+.4*RND-.2)*L/B
390 VY=VY+(1+.4*RND-.2)*M/B
400 IF (VX=0)*(VY>=0)THEN 402 ELSE 410
402 Z=90
404 GOTO 450
410 IF (VY=0)*(VY<0)THEN 412 ELSE 420
412 Z=270
414 GOTO 450
420 Z=ATN(VY/VX)
422 Z=Z*180/P
430 Z=Z+INT(10+RND)
435 Z=INT(Z)
440 IF VX<0 THEN 442 ELSE 450
442 Z=Z+180
450 X=X+VX
452 Y=Y+VY
530 V=SQR(VX↑2+VY↑2)
540 D=SQR((X-80)↑2+(Y-80)↑2)
600 IF F<0 THEN 602 ELSE 610
602 PRINT "OUT OF FUEL"
604 GOTO 660
610 IF (D<1)*(V<1)THEN 612 ELSE 620
612 PRINT "ARRIVED"
614 GOTO 630
620 NEXT T
630 PRINT ;"THE TRIP TOOK ";T;"HOURS."
640 R=200*T
```

```
650 PRINT "YOUR RATING IS ";R
660 PRINT ::
662 INPUT "PLAY AGAIN ? ":C$
665 IF C$="Y" THEN 5
670 END
```

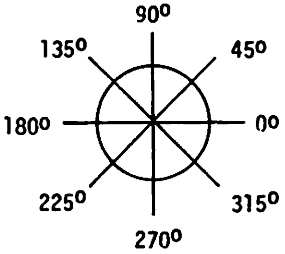
SPACE FLIGHT MODIFICATIONS

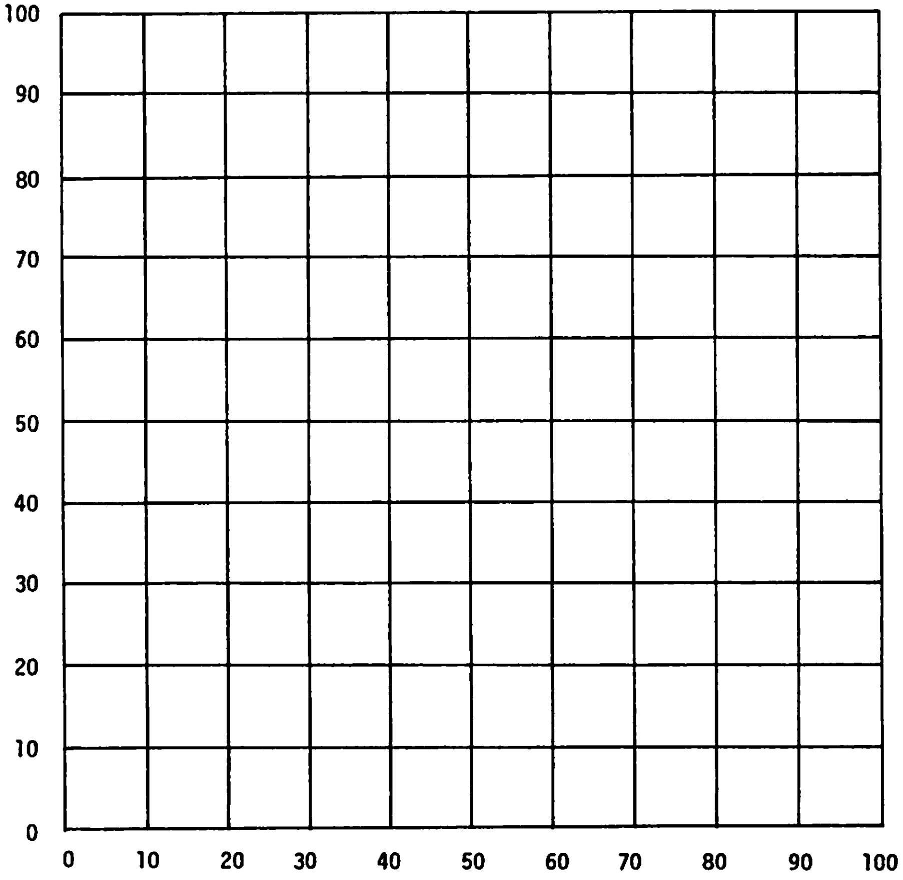
Minor

1. Starting position -- lines 10, 12
2. Amount of fuel -- line 14
3. Time limit -- line 30
4. Planets location -- line 540
5. Arrival conditions -- line 610
6. Probability of problems -- line 200

Major

1. One must fire small thruster rockets to rotate ship.
2. Have meteors hit ship.
3. Use meteor shields.
4. Fight aliens.
5. Visit more than one planet.
6. Provide planets with gravitational force.
7. Have refueling stations.


90°
135°
45°
180°
0°
225°
315°
270°


100
90
80
70
60
50
40
30
20
10
0
0
10
20
30
40
50
60
70
80
90
100

FOREST FIRE

Scenario

A lightning storm has ignited fires in a forest. Your task is to put out the fires and save as many trees as possible. The forest is divided into 81 sectors formed by a 9X9 grid. Each sector is identified by the number of its row and column. The symbol, ".", represents woods, an "*" represents fire, and a blank space represents burnt out woods.

The chance of an existing fire spreading to adjacent wooded areas is 70%. Fires last for nine turns before burning out.

You have two weapons with which to fight the fire. You can drop chemicals that are designed to extinguish the fires in a specified sector. The chance that the drop will affect the fires in this sector and its eight adjacent sectors is 50%. For example, if there are six fires burning in a nine-square area, approximately three will be affected by the chemicals. The effect of chemicals is to reduce the number of turns before the fire burns out by three. Since a fire lasts only nine turns, three successful chemical hits will be needed to extinguish a fire. If the fire has been burning for six turns, then one hit will suffice.

The second weapon available to you is a backfire. To start a backfire, you must respond to the row input with a zero. You will then be asked for a backfire row and column. The sector in which a backfire is started must be wooded. This backfire will not spread and will burn out in the next turn, forming a barrier against the spread of fire.

Your rating will be the number of trees remaining after all the fires are out, plus 30.

Sample Run

```
        #1
  1 2 3 4 5 6 7 8 9
1 . . . . . . . . .
2 . . . . . . . . .
3 . . . . . . . . .
4 . . . . . . . * .
5 . . . . . . . * .
6 . . . * . . . . .
7 . . . . . . . . .
8 . . . . . . . . .
9 . . . . . . . . .

ROW?  0
BACKFIRE ROW?  4
BACKFIRE COLUMN?  7
```

```
        #2
  1 2 3 4 5 6 7 8 9
1 . . . . . . . . .
2 . . . . . . . . .
3 . . . . . . . . .
4 . . . . . . * * .
5 . . . . . . . * .
6 . . . * . . . . .
7 . . * . . . . . .
8 . . . . . . . . .
9 . . . . . . . . .

ROW?  0
BACKFIRE ROW?  5
BACKFIRE COLUMN?  7
```

```
        #3
  1 2 3 4 5 6 7 8 9
1 . . . . . . . . .
2 . . . . . . . . .
3 . . . . . . . . *
4 . . . . . . . * .
5 . . * . . . * * .
6 . . . * . . . * .
7 . . * . . . . . .
8 . . . . . . . . .
9 . . . . . . . . .

ROW?  0
BACKFIRE ROW?  6
BACKFIRE COLUMN?  7
```

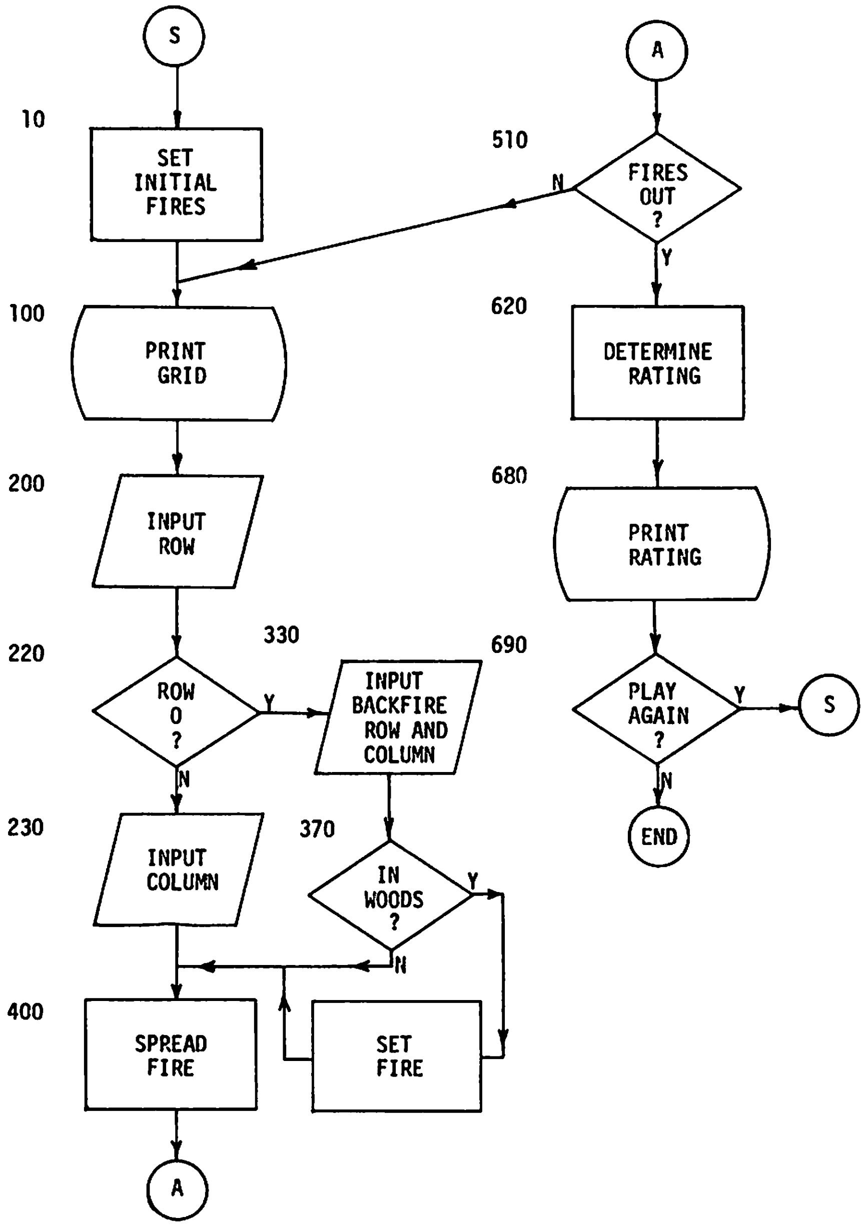
```
        #4
  1 2 3 4 5 6 7 8 9
1 . . . . . . . . .
2 . . . . . . . . .
3 . . . . . . . . *
4 . . . . . .   * .
5 . . * . . .   * *
6 . . . * . . * * .
7 . . * . . . . . .
8 . . . . . . . . .
9 . . . . . . . . .

ROW?  6
COLUMN?  3
```

```
        #11
  1 2 3 4 5 6 7 8 9
1 . . .   * * . * *
2 . . . . * * *   .
3 . . . . .
4 . . . . . .     .
5 . *   . . .
6 *       . .
7 . *   . . . *
8 . .   . . . . *
9 . .   * . . * . .

ROW?  6
COLUMN?  2
```

```
        #12
  1 2 3 4 5 6 7 8 9
1 . . .   * * .   *
2 . . . .   * *   .
3 . . . . .
4 . . . . . .     .
5 . *   . . .
6         . .
7 .     . . . *
8 . .   . . * .
9 . .   * . . * . .

ROW?  8
COLUMN?  7
```

```
        #16
  1 2 3 4 5 6 7 8 9
1 . . .     * .
2 . . . .         .
3 . . . . .
4 . . . . . .     .
5 .     . . .
6         . *
7 .     .
8 . .   . .   .
9 . .     . .   . .

ROW?  6
COLUMN?  6


YOUR RATING IS 69.
PLAY AGAIN?
```

FOREST FIRE FLOWCHART



FOREST FIRE PROGRAM

Variables

L(R,C)	Burnt woods: 0, fire: 1-9, woods: 10, temporary variable: 11
R	Row
C	Column
I	Row number increment
J	Column number increment
A	Adjacent row
B	Adjacent column
F	Count
T	Temporary variable
R	Rating

Listing

```
1 REM  *** FOREST FIRE ***
5 RANDOMIZE
6 DIM L(9,9)
7 CALL CLEAR
8 CALL CHAR(128,"18183C7EFFFF1818")
9 CALL COLOR(13,13,4)
12 CALL CHAR(136,"27172C6EFEFE1818")
13 CALL COLOR(14,9,4)
15 CALL COLOR(3,2,14)
16 CALL COLOR(4,2,14)
20 FOR R=1 TO 9
22 FOR C=1 TO 9
30 L(R,C)=10
40 NEXT C
42 NEXT R
50 FOR I=1 TO 3
60 R=INT(9*RND+1)
70 C=INT(9*RND+1)
80 L(R,C)=9
90 NEXT I
95 REM  PRINT GRID
100 FOR I=1 TO 9
102 CALL HCHAR(1,2*I+7,48+I)
103 CALL HCHAR(2*I+2,6,48+I)
104 NEXT I
110 FOR R=1 TO 9
130 FOR C=1 TO 9
135 TR=46
140 IF L(R,C)=10 THEN 142 ELSE 150
142 TR=128
150 IF (L(R,C)>0)*(L(R,C)<10)THEN 152 ELSE 160
152 TR=136
160 CALL HCHAR(2*R+2,2*C+7,TR)
170 NEXT C
180 NEXT R
195 REM  INPUT ROUTINE
200 PRINT "ROW ?";
202 CALL KEY(3,KEY,ST)
```

```
203 IF ST=0 THEN 202
204 R=KEY-48
205 IF (R<0)+(R>9)THEN 202
206 CALL HCHAR(24,7,KEY)
220 IF R=0 THEN 330
230 PRINT "  COLUMN ?";
232 CALL KEY(3,KEY,ST)
233 IF ST=0 THEN 232
234 C=KEY-48
235 IF (C<1)+(C>9)THEN 232
236 CALL HCHAR(24,17,KEY)
250 FOR I=-1 TO 1
255 FOR J=-1 TO 1
260 A=R+I
265 B=C+J
270 IF (A<1)+(A>9)+(B<1)+(B>9)THEN 310
280 IF (L(A,B)<1)+(L(A,B)=10)THEN 310
290 IF RND>.5 THEN 310
300 L(A,B)=L(A,B)-3
310 NEXT J
315 NEXT I
320 GOTO 400
330 PRINT " BK ROW ?";
331 CALL KEY(3,KEY,ST)
332 IF ST=0 THEN 331
333 R=KEY-48
334 IF (R<1)+(R>9)THEN 331
335 CALL HCHAR(24,16,KEY)
340 PRINT " BK COL ?";
341 CALL KEY(3,KEY,ST)
342 IF ST=0 THEN 341
343 C=KEY-48
344 IF (C<1)+(C>9)THEN 341
345 CALL HCHAR(24,25,KEY)
370 IF L(R,C)=10 THEN 372 ELSE 400
372 L(R,C)=2
395 REM  SPREAD FIRE
400 FOR R=1 TO 9
405 FOR C=1 TO 9
410 IF (L(R,C)<1)+(L(R,C)>9)THEN 500
420 IF L(R,C)<3 THEN 500
430 I=INT(3*RND-1)
440 J=INT(3*RND-1)
450 A=R+I
460 IF (A<1)+(A>9)+(B<1)+(B>9)THEN 500
470 IF L(A,B)<>10 THEN 500
480 IF RND<.3 THEN 500
490 L(A,B)=11
500 NEXT C
502 NEXT R
503 CALL CLEAR
505 REM  BURN FIRE AND COUNT
510 F=0
520 FOR R=1 TO 9
```

```
530 FOR C=1 TO 9
535 CALL SOUND(100,-1*R*.7,C)
540 T=L(R,C)
550 IF T=11 THEN 552 ELSE 560
552 T=9
560 IF (T>0)*(T<10)THEN 562 ELSE 570
562 T=T-1
564 F=F+1
570 L(R,C)=T
580 NEXT C
585 NEXT R
590 IF F<1 THEN 620
600 GOTO 100
615 REM  COUNT WOODS
620 C=0
630 FOR R=1 TO 9
635 FOR C=1 TO 9
640 IF L(R,C)=10 THEN 642 ELSE 650
642 W=W+1
650 NEXT C
652 NEXT R
660 R=W+30
670 CALL CLEAR
680 PRINT "YOUR RATING IS";R;
690 PRINT ::
700 INPUT "PLAY AGAIN? ":X$
710 IF X$="Y" THEN 7
720 END
```

FOREST FIRE MODIFICATIONS

Minor

1. Number of beginning fires -- line 50
2. Location of beginning fires -- lines 60, 70
3. Probability of putting out fire -- line 290
4. Amount fire burns out each turn -- line 300
5. Size of backfire -- line 370
6. Probability of spread -- line 480
7. Size of spread fires -- line 550
8. Rating scale - lines 660, 680

Major

1. Change grid size.
2. Randomly choose location of beginning fires.
3. Add time to move from one place to another.
4. Have wind speed and direction affect the spread of the fire.
5. Include barriors such as lakes and roads.
6. Have some of the sectors burn faster than others.

NAUTICAL NAVIGATION

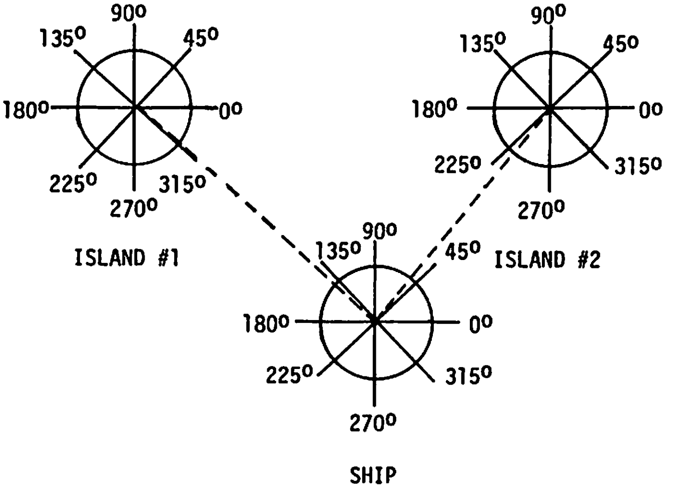
Scenario

Your task is to navigate a sailboat that has an electronic direction finder to three different islands in the South Pacific. You do not have to dock at the islands, but only come close enough to make a visual sighting. The minimum sighting distance will vary from five to ten miles, depending upon weather conditions.

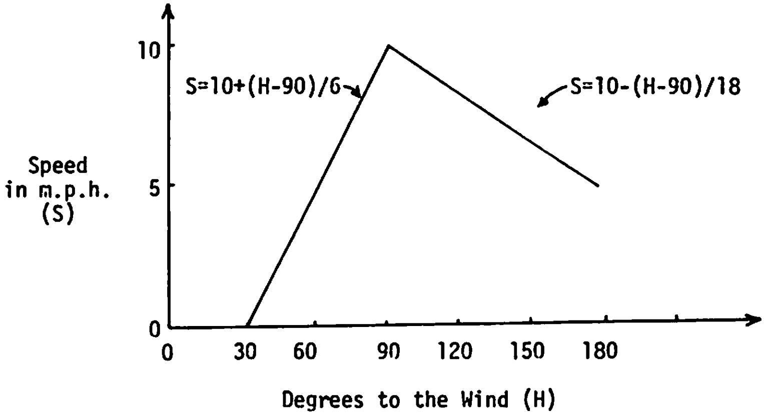
The islands are located at coordinates (200,300), (600,300), and (300,100). Your starting location will be approximately (200,200). You will need graph paper and an inexpensive protractor and ruler in order to plot your course.

Each turn you will receive information about your bearings in degrees from each of the three islands. For convenience, you will also receive the bearings from the ship to each of the islands. The example below shows how the bearings are determined. If you know the bearing from two of the three islands, you can locate the ship; however, there are some random errors in the readings, so it might be wise to use the readings from all three islands.

Bearing from island #1: 317°; bearing to island #1: 138°.
Bearing from island #2: 230°; bearing to island #2: 50°.



After you locate your position, you must determine your heading and the length of time you wish to remain on this course. You can use the heading from the ship to the island of your destination to determine the ship's heading. Since you are in a sailboat, your speed will depend on your direction with respect to an easterly wind. In order to make any progress toward the East, you must tack at either 45° or 315°. The speed of the sailboat as a function of its direction is shown in the graph below.



The fastest speed of ten miles per hour is acheived when the boat is perpendicular to the wind -- heading either directly north (90^o) or south (270^o). When the boat is running with the wind directly behind it, its speed is about half the maximum speed or five m.p.h.

Once you determine the heading, you must determine the length of time you wish to remain on the heading or the length of time you wish to travel before the next navigational check. The speed at 70^o is about 6.7 m.p.h. In ten hours, you would travel about 67 miles. Of course, the wind speed varies; so you may wish to make one or two navigational checks on a long run.

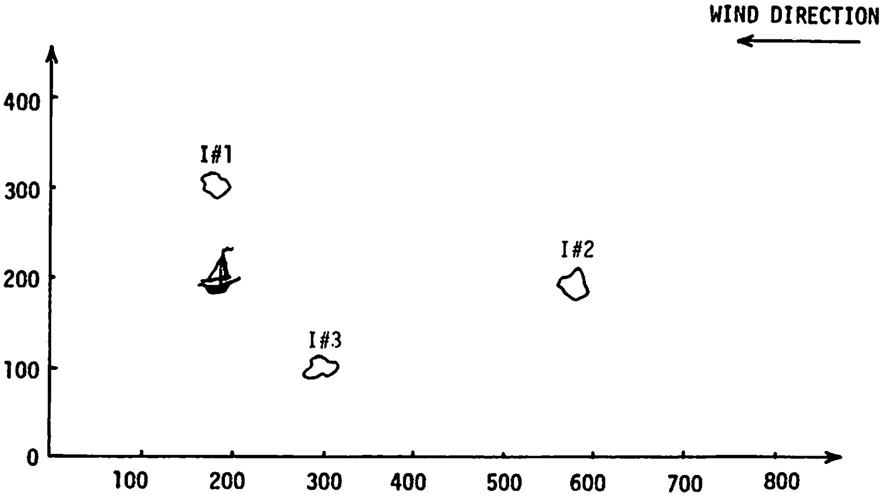
You can visit the three islands in any order. You must compute the angle and time so the end of a run is within five to ten miles of an island. Since visibility conditions vary, you may have to wait for a turn to allow sighting conditions to improve.

Your rating as a navigator will depend on the number of navigational checks required and the amount of time for the trip. A good sailor should be able to complete the trip with a rating close to 100.

Sample Run

```
NAVIGATION CHECK  1
BEARING FROM 1  279   TO 1   99
BEARING FROM 2  197   TO 2   17
BEARING FROM 3  136   TO 3   316
ELAPSED TIME  0
HEADING?  99
TIME?  33

NAVIGATION CHECK  2
BEARING FROM 1  97    TO 1   277
BEARING FROM 2  158   TO 2   338
BEARING FROM 3  108   TO 3   288
ELAPSED TIME  32.969
HEADING?  277
TIME?  20

NAVIGATION CHECK  3
VISITED  1
BEARING FROM 1  84    TO 1   264
BEARING FROM 2  179   TO 2   359
BEARING FROM 3  115   TO 3   295
ELAPSED TIME  52.957
HEADING?  295
TIME?  30

NAVIGATION CHECK  4
VISITED  1
BEARING FROM 1  296   TO 1   116
BEARING FROM 2  201   TO 2   21
BEARING FROM 3  117   TO 3   297
ELAPSED TIME  82.924
HEADING?  297
TIME?  10
```

```
NAVIGATION CHECK  5
VISITED  1
BEARING FROM 1  296   TO 1   116
BEARING FROM 2  209   TO 2   29
BEARING FROM 3  114   TO 3   294
ELAPSED TIME  92.883
HEADING?  294
TIME?  3

NAVIGATION CHECK  6
VISITED  1
VISITED  3
BEARING FROM 1  296   TO 1   116
BEARING FROM 2  212   TO 2   32
BEARING FROM 3  119   TO 3   299
ELAPSED TIME  95.856
HEADING?  60
TIME?  120

NAVIGATION CHECK  7
VISITED  1
VISITED  3
BEARING FROM 1  35    TO 1   215
BEARING FROM 2  92    TO 2   272
BEARING FROM 3  58    TO 3   238
ELAPSED TIME  215.833
HEADING?  272
TIME?  28

TRIP COMPLETED
IN 243.859 HOURS.
7 NAVIGATIONAL CHECKS
YOUR RATING IS  66
PLAY AGAIN?
```



NAUTICAL NAVIGATION PROGRAM

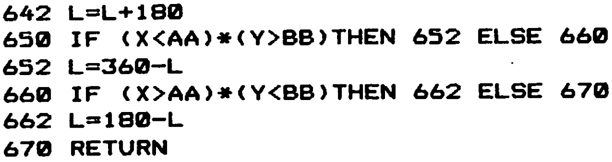
Variables

D(3)	Set to 1 if arrived at destination
A(3),B(3)	Coordinates of islands
X,Y	Coordinates of ship
E	Total elapsed time
C	Number of navigational checks
L	Angle bearing from island
H	Heading of ship
T	Time for one leg of trip
A,B	Temporary variables
Y$	Play again

Listing

```
1 REM *** NAUTICAL NAVIGATION ***
5 REM  PLACE ISLANDS AND SHIP
7 RANDOMIZE
8 CALL CLEAR
10 DIM A(3),B(3),D(3)
15 CALL CLEAR
20 E=0
22 P=3.14159
30 FOR I=1 TO 3
40 READ AA,BB
50 A(I)=10*AA
52 B(I)=10*BB
60 D(I)=0
70 NEXT I
80 DATA 20,30,60,20,30,10
90 X=175+50*RND
92 Y=175+50*RND
95 REM  START MAIN LOOP
100 FOR C=1 TO 100
105 PRINT ::
110 PRINT "NAVIGATION CHECK";C;":"
115 PRINT
120 FOR I=1 TO 3
130 IF D(I)=1 THEN 132 ELSE 140
132 PRINT "VISITED ";I
140 NEXT I
150 FOR I=1 TO 3
160 AA=A(I)
162 BB=B(I)
170 GOSUB 600
172 L=L+2.5-5*RND
180 L=L+180
182 IF L>360 THEN 184 ELSE 190
184 L=L-360
190 PRINT "BEARING FROM";I;"IS";INT(L)
200 IF L>=180 THEN 202 ELSE 210
202 L=L-180
204 PRINT TAB(22);" TO";INT(L)
```

```
206 GOTO 220
210 IF L<180 THEN 212 ELSE 220
212 L=L+180
214 PRINT TAB(22);" TO";INT(L)
220 NEXT I
225 REM  INPUT
230 PRINT "ELAPSED TIME";INT(E)
235 PRINT
240 INPUT "HEADING? ":H
250 H=H+5-10*RND
260 INPUT "TIME? ":T
263 T=ABS(T)
270 CO=COS(H*P/180)
272 SI=SIN(H*P/180)
280 IF H>180 THEN 282 ELSE 290
282 H=360-H
290 IF H<30 THEN 292 ELSE 300
292 S=0
300 IF (H>=30)*(H<90)THEN 302 ELSE 310
302 S=10+(H-90)/18
310 IF H>90 THEN 312 ELSE 320
312 S=10-(H-90)/18
320 S=S+2*RND-1
330 T=T+(.1*RND-.05)
340 X=X+T*S*CO
350 Y=Y+T*S*SI
360 E=E+T
400 FOR I=1 TO 3
410 DD=SQR((X-A(I))↑2+(Y-B(I))↑2)
420 IF DD<5+10*RND THEN 422 ELSE 430
422 D(I)=1
430 NEXT I
440 IF D(1)+D(2)+D(3)=3 THEN 442 ELSE 450
442 GOTO 500
450 NEXT C
460 PRINT "EXCEED NAVIGATION CHECK"
462 GOTO 530
500 PRINT "TRIP COMPLETED IN ";E;"HOURS"
510 PRINT "NO. OF CHECKS IS";C
520 PRINT "RATING IS";170-(INT(E+10*C/3))
530 PRINT ::
540 INPUT "PLAY AGAIN? ": X$
550 IF X$="Y" THEN 15
560 END
600 IF (X=AA)*(Y>BB)THEN 602 ELSE 610
602 L=270
604 RETURN
610 IF (X=AA)*(Y<BB)THEN 612 ELSE 620
612 L=90
614 RETURN
620 N=ABS(Y-BB)/ABS(X-AA)
630 L=ATN(N)
635 L=180*L/P
640 IF (X>AA)*(Y>=BB)THEN 642 ELSE 650
```



```
642 L=L+180
650 IF (X<AA)*(Y>BB)THEN 652 ELSE 660
652 L=360-L
660 IF (X>AA)*(Y<BB)THEN 662 ELSE 670
662 L=180-L
670 RETURN
```

NAUTICAL NAVIGATION MODIFICATIONS

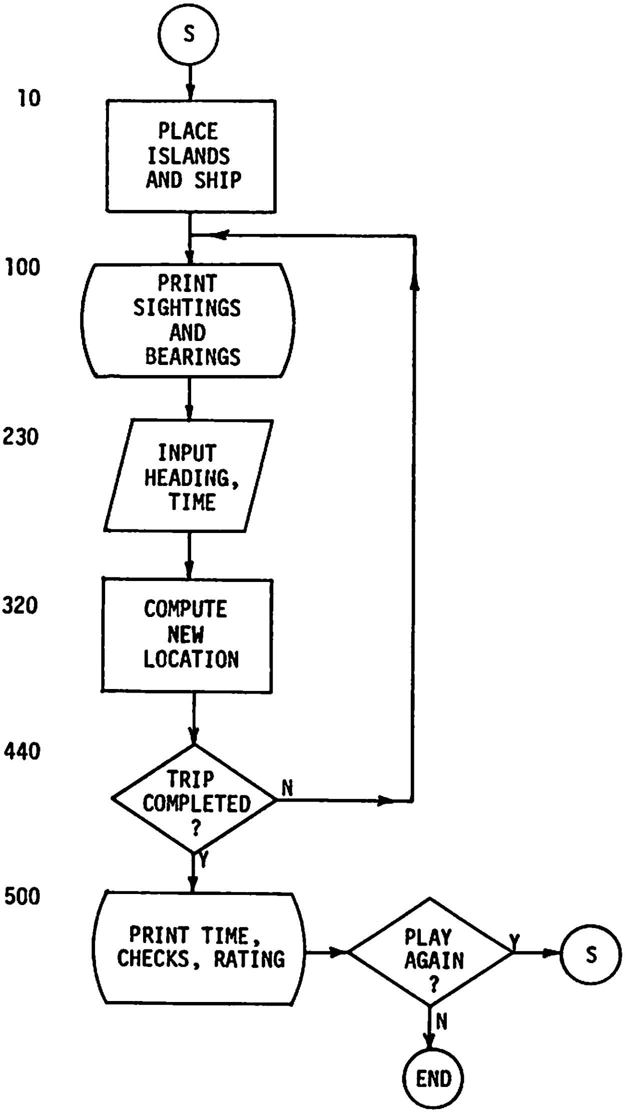
Minor

1. Location of islands -- line 80
2. Starting place of ship -- lines 90, 92
3. Error in angle -- line 172
4. Input error -- line 250
5. Speed error -- line 320
6. Time error -- line 330
7. Sighting criteria -- line 420
8. Rating -- line 520

Major

1. Change number of islands.
2. Have storms.
3. Have wind direction change.

NAUTICAL NAVIGATION FLOWCHART



BUSINESS MANAGEMENT

Scenario

In this simulation you manage a small factory that produces three different kinds of products (P1 - P3). Three different kinds of raw materials (R1 - R3) are required to produce the products. Each product requires exactly two raw materials with a different subscript. For example, to manufacture one unit of P2, you would need a unit of R1 and a unit of R3. To manufacture one unit of P3, you would need a unit of R1 and R2.

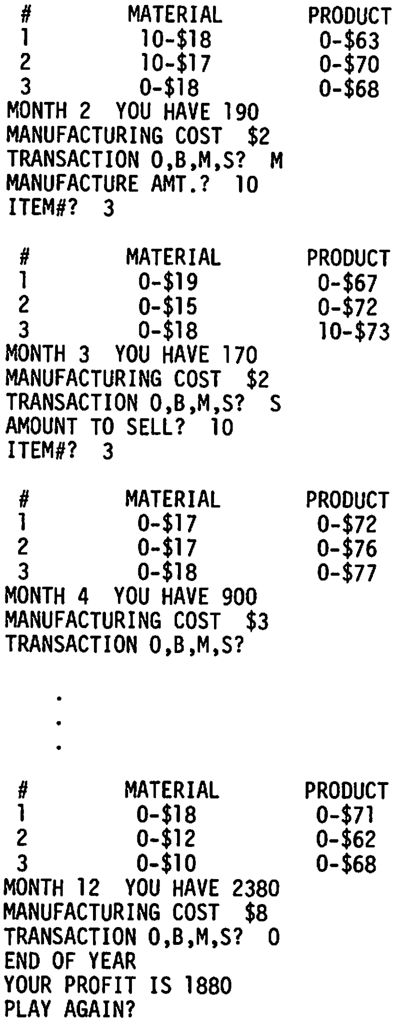
The cost of raw materials varies from $10 to $20 per unit. It costs from $1 to $9 per unit to manufacture a product from raw materials. The selling price of each finished product varies from $50 to $90 per unit. Prices of raw materials and manufacturing costs will vary by not more than $2 per turn. Prices of finished products will vary by not more than $5 per turn.

You will receive a data report at the beginning of each turn. This report will give you the number of units you have on hand, available cash, and the manufacturing costs. You can buy, manufacture, or sell each turn. In order to manufacture a given product, you must have enough of the correct kind of materials on hand.

After twelve turns (months), the materials and/or products that you have on hand will be automatically sold at the current prices and your profit will be computed.

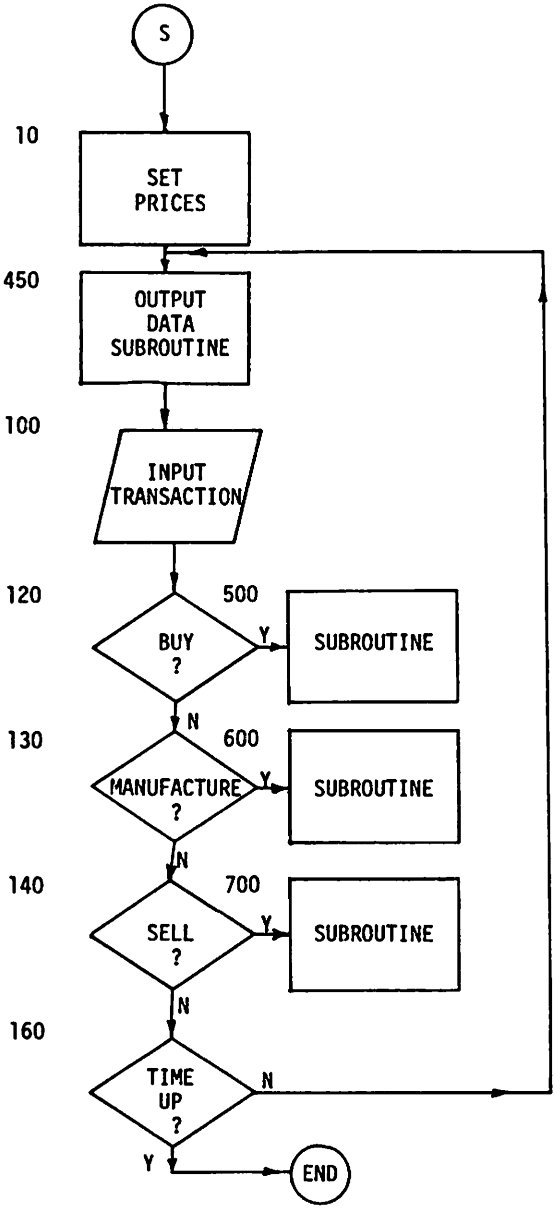
Sample Run

```
 #        MATERIAL      PRODUCT
 1         0-$16         0-$72
 2         0-$15         0-$72
 3         0-$17         0-$73
MONTH 0  YOU HAVE 500
MANUFACTURING COST  $2
TRANSACTION O,B,M,S?  B
AMT. OF MATERIALS?  10
ITEM#?  2

 #        MATERIAL      PRODUCT
 1         0-$16         0-$67
 2        10-$16         0-$71
 3         0-$16         0-$73
MONTH 1  YOU HAVE 350
MANUFACTURING COST  $1
TRANSACTION O,B,M,S?  B
AMT. OF MATERIALS?  10
ITEM#?  1
```



```
 #         MATERIAL       PRODUCT
 1          10-$18         0-$63
 2          10-$17         0-$70
 3          0-$18          0-$68
MONTH 2  YOU HAVE 190
MANUFACTURING COST  $2
TRANSACTION O,B,M,S?  M
MANUFACTURE AMT.?  10
ITEM#?  3

 #         MATERIAL       PRODUCT
 1          0-$19          0-$67
 2          0-$15          0-$72
 3          0-$18          10-$73
MONTH 3  YOU HAVE 170
MANUFACTURING COST  $2
TRANSACTION O,B,M,S?  S
AMOUNT TO SELL?  10
ITEM#?  3

 #         MATERIAL       PRODUCT
 1          0-$17          0-$72
 2          0-$17          0-$76
 3          0-$18          0-$77
MONTH 4  YOU HAVE 900
MANUFACTURING COST  $3
TRANSACTION O,B,M,S?

     .
     .
     .

 #         MATERIAL       PRODUCT
 1          0-$18          0-$71
 2          0-$12          0-$62
 3          0-$10          0-$68
MONTH 12  YOU HAVE 2380
MANUFACTURING COST  $8
TRANSACTION O,B,M,S?  O
END OF YEAR
YOUR PROFIT IS 1880
PLAY AGAIN?
```

BUSINESS MANAGEMENT FLOWCHART



BUSINESS MANAGEMENT PROGRAM

Variables

R(I)	Number of raw materials
C(I)	Cost of one unit of raw material
F(I)	Number of finished products
P(I)	Price of one unit of finished product ($50-$90)
C	Cash on hand
M	Manufacturing costs ($1-$9) per unit
T	Time
N	Item number
A	Amount
T$	Input O,B,M,S

Listing

```
1 REM *** BUSINESS MANAGEMENT ***
2 RANDOMIZE
3 CALL CLEAR
5 REM  SET PRICES
10 DIM R(3),C(3),F(3)
20 CH=500
23 M=2
30 FOR I=1 TO 3
40 R(I)=0
42 F(I)=0
50 C(I)=INT(3*RND+15)
60 P(I)=INT(10*RND+70)
70 NEXT I
75 CALL CLEAR
80 FOR T=0 TO 12
85 PRINT
90 GOSUB 450
100 PRINT :;"MONTH";T;"YOU HAVE $";CH;
105 PRINT "MANUFACTURING COSTS ARE $";M;:
110 INPUT "TRANSACTION O,B,M,S? ":T$
120 IF T$="B" THEN 122 ELSE 130
122 GOSUB 500
130 IF T$="M" THEN 132 ELSE 140
132 GOSUB 600
140 IF T$="S" THEN 142 ELSE 150
142 GOSUB 700
150 GOSUB 300
160 NEXT T
165 REM SUMMARY
167 CALL CLEAR
170 PRINT "END OF YEAR"
180 FOR I=1 TO 3
190 CH=CH+R(I)*C(I)
200 CH=CH+F(I)*P(I)
210 NEXT I
220 CH=CH-500
230 PRINT "YOUR PROFIT IS";CH
```

```
240 INPUT "PLAY AGAIN? ":Y$
250 IF Y$="Y" THEN 20
260 END
295 REM  CHANGE PRICE SUBROUTINE
300 FOR I=1 TO 3
310 J=INT(5*RND-2)
320 J=C(I)+J
330 IF (J<10)+(J>20)THEN 310
340 C(I)=J
350 J=INT(11*RND-5)
360 J=P(I)+J
370 IF (J<50)+(J>90)THEN 350
380 P(I)=J
390 NEXT I
400 J=INT(5*RND-2)
410 J=M+J
420 IF (J<1)+(J>9)THEN 400
430 M=J
440 RETURN
445 REM  OUTPUT DATA
450 PRINT "ITEM:  MATERIALS: PRODUCT:";:;
460 FOR I=1 TO 3
470 PRINT I;TAB(7);R(I);"$";C(I);TAB(19);F(I);"$";P(I)
480 NEXT I
490 RETURN
495 REM BUY RAW MATERIALS
500 INPUT "AMOUNT OF MATERIALS? ":A
510 INPUT "ITEM #? ":N
520 IF (N<1)+(N>3)THEN 522 ELSE 530
522 PRINT "ERROR"
523 RETURN
530 CH=CH-A*C(N)
540 IF CH<0 THEN 570
550 R(N)=R(N)+A
560 RETURN
570 CH=CH+A*C(N)
580 PRINT "INSUFFICIENT FUNDS"
590 RETURN
595 REM  MANUFACTURE
600 INPUT "MANUFACTURE AMOUNT? ":A
605 INPUT "ITEM #? ":N
610 IF (N<0)+(N>3)THEN 612 ELSE 620
612 PRINT "ERROR"
613 RETURN
620 CH=CH-A*M
630 IF CH<0 THEN 632 ELSE 640
632 CH=CH+A*M
633 RETURN
640 FOR I=1 TO 3
650 IF I=N THEN 680
660 R(I)=R(I)-A
670 IF R(I)<0 THEN 672 ELSE 680
672 PRINT "MATERIALS GONE"
673 R(I)=R(I)+A
```

```
674 CH=CH+A*M
675 RETURN
680 NEXT I
682 F(N)=F(N)+A
684 RETURN
695 REM  SELL
700 INPUT "AMOUNT TO SELL? ":A
702 INPUT "ITEM #? ":N
710 IF (N<0)+(N>3)THEN 712 ELSE 720
712 PRINT "ERROR"
714 RETURN
720 F(N)=F(N)-A
730 IF F(N)<0 THEN 760
740 CH=CH+A*P(N)
750 RETURN
760 F(N)=F(N)+A
770 PRINT "PROODUCTS GONE"
780 RETURN
```

BUSINESS MANAGEMENT MODIFICATIONS

Minor

1. Starting amounts -- lines 20, 50, 60
2. Number of turns -- line 80
3. Amount raw materials vary -- line 310
4. Range of raw materials -- line 330
5. Amount products vary -- line 350
6. Range of products -- line 370
7. Amount manufacturing costs vary -- line 400
8. Range of manufacturing costs -- line 420

Major

1. Increase number of raw materials and finished products.
2. Have a storage fee.
3. When you buy, prices increase.
4. When you sell, prices decrease.
5. Borrow money with interest.
6. Add random events, such as strikes, shortage of materials, fires, no demand.
7. Provide names for raw materials and products.

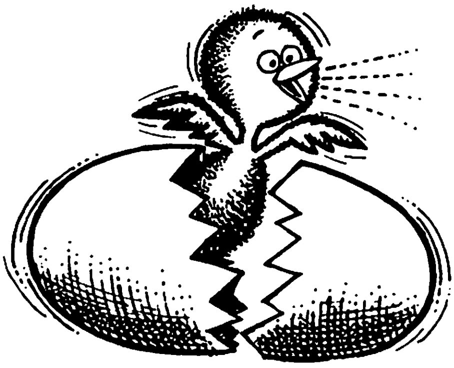
RARE BIRDS

Scenario

In this simulation you attempt to identify as many birds as possible in a ten hour period. First, you must choose a place to watch birds. It must be in the swamp (S), the water (W), the desert (D), or the forest (F). Then you must choose a time of day -- morning (M), or evening (E). Finally, you must choose to look up in the sky -- high (H) or on the ground -- low (L). There are sixteen different birds that can be identified The birds are classified as small or big, yellow or blue, shortbeaked or long beaked, and female or male.

After you have selected a place to watch birds, you will receive one clue about the bird and the length of time it took you to spot it. If no bird is spotted in a two-hour period, you may try a new place. After receiving your clue, you then have an opportunity to identify the bird. You should refer to the bird watching chart to determine where the birds are seen and their specific characteristics. The birds with the larger numbers are observed more frequently.

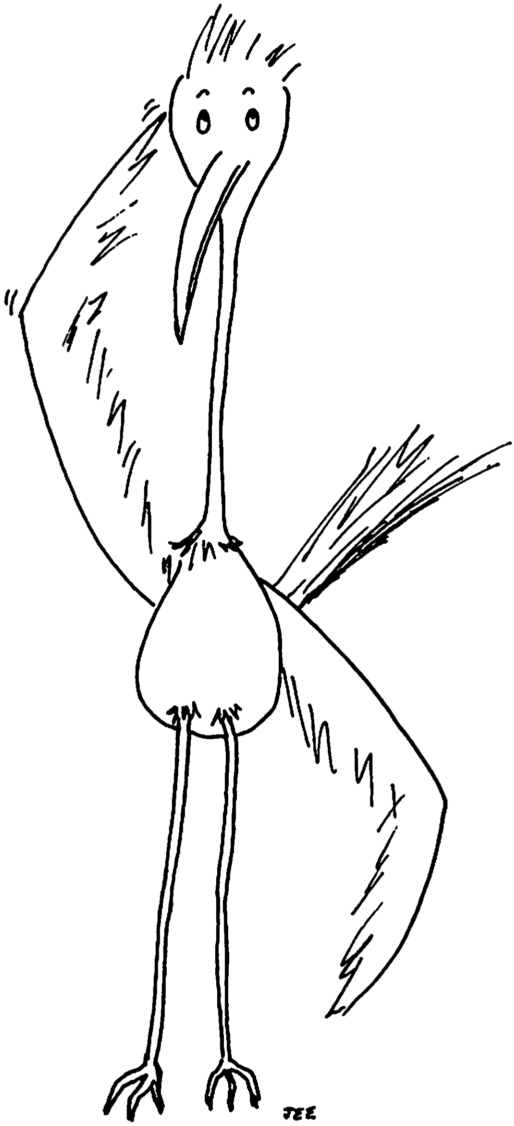
If your first identification is not correct, you will have an opportunity to try again. Each time you try, however, one point will be subtracted from your final rating. If you identify a bird that you have identified correctly before, you will be notified of the fact and may try a new place. Your final rating is determined by multiplying ten times the number of birds identified and subtracting one for each incorrect identification.



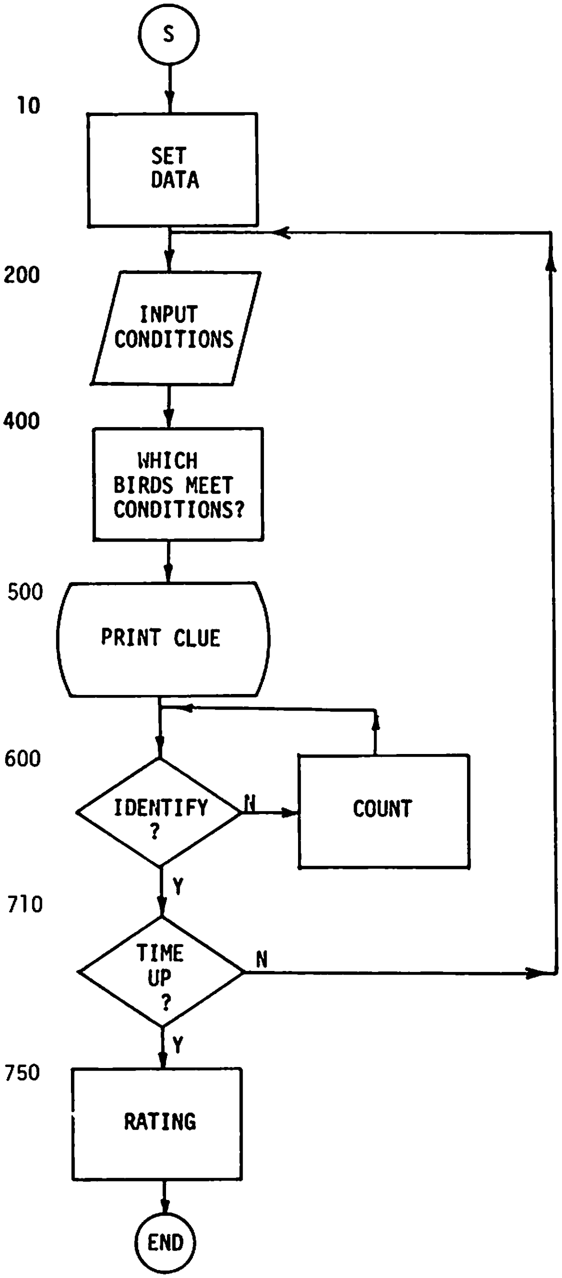
Sample Run

```
PLACE S,W,D,F?  S
WHEN M,E?  E
WHERE H,L?  L
THE BIRD IS YELLOW
TIME LAPSE:  1.28
TOTAL TIME:  1.28
IDENTIFY 1-16?  12

INCORRECT
IDENTIFICATION
IDENTIFY 1-16?  11
A NEW ONE!

PLACE S,W,D,F?  W
WHEN M,E?  E
WHERE H,L?  H
THE BIRD IS BIG
TIME LAPSE:  .18
TOTAL TIME:  1.46
IDENTIFY 1-16?  11

INCORRECT
IDENTIFICATION
IDENTIFY 1-16?  9
A NEW ONE!
.
.
.

PLACE S,W,D,F?  S
WHEN M,E?  E
WHERE H,L?  L
NO SIGHTINGS
.
.
.

TIME UP
YOU SAW BIRD  #1
YOU SAW BIRD  #6
YOU SAW BIRD  #9
YOU SAW BIRD  #12
YOU SAW BIRD  #15
YOU SAW BIRD  #16
YOUR RATING IS  57
PLAY AGAIN?
```

Note: the first line "PLEASE WAIT" precedes the above.



RARE BIRDS FLOWCHART



Variables

B(I,J)	I is bird (1-16); J is characteristic (1-14)
N$(I)	Name characteristic
P(I)	Probability of sighting
K,I,J,Q,N	Temporary variables
L$	Place
T$	When
A$	Where
I	Lapsed time for one sighting
H	Total time
B_1	Number of identifications
C_1	Number of birds identified

Listing

```
5 REM  SET DATA
6 RANDOMIZE
7 CALL CLEAR
10 H=0
15 DIM B(16,14),ID(16),N$(8),P(16)
20 PRINT "PLEASE WAIT"
25 FOR I=1 TO 16
30 B(I,14)=0
40 P(I)=1/(17-I)
50 READ N
60 FOR J=12 TO 1 STEP -1
70 Q=INT(N/2)
80 B(I,J)=2*(N/2-Q)
90 N=Q
100 NEXT J
110 NEXT I
111 REM
120 DATA 2128,1121,594,355,3220
130 DATA 2725,2454,1703,1528,1017
140 DATA 2042,3067,3516,3773,4030,4031
141 REM
150 FOR I=1 TO 8
160 READ N$(I)
165 NEXT I
170 DATA BIG,SMALL
180 DATA BLUE,YELLOW
190 DATA LONG BEAKED,SHORT BEAKED,FEMALE,MALE
191 REM
195 REM  INPUT PLACE
200 FOR I=1 TO 16
205 ID(I)=0
206 NEXT I
208 PRINT
210 INPUT "PLACE S,W,D,F? ":L$
220 INPUT "WHEN M,E? ":T$
230 INPUT "WHERE H,L? ":A$
232 PRINT
235 REM
260 IF L$="S" THEN 265 ELSE 270
```

```
265 ID(1)=1
270 IF L$="W" THEN 275 ELSE 280
275 ID(2)=1
280 IF L$="D" THEN 285 ELSE 290
285 ID(3)=1
290 IF L$="F" THEN 295 ELSE 300
295 ID(4)=1
300 IF T$="M" THEN 305 ELSE 310
305 ID(5)=1
310 IF T$="E" THEN 315 ELSE 320
315 ID(6)=1
320 IF A$="H" THEN 325
325 ID(7)=1
330 IF A$="L" THEN 335 ELSE 340
335 ID(8)=1
340 FOR I=1 TO 16
345 B(I,13)=0
346 NEXT I
350 FOR I=1 TO 16
355 FOR J=1 TO 8
360 IF (B(I,J)<>ID(J))*(B(I,J)=0)THEN 390
370 NEXT J
380 B(I,13)=1
390 NEXT I
391 REM
395 REM  FIND BIRDS
400 FOR I=1 TO 2 STEP .02
410 F=INT(16*RND+1)
420 IF B(J,13)<1 THEN 440
430 IF RND<P(J)THEN 460
440 NEXT I
450 PRINT "NO SIGHTINGS"
455 H=H+I
456 GOTO 200
460 H=H+I
470 K=INT(4*RND+1)
480 N=B(J,K+8)
490 PRINT "THE BIRD IS ";N$(2*K-N)
492 PRINT "TIME LAPSE:";I
493 PRINT "TOTAL TIME:";H
494 REM
495 REM  INPUT ID
499 PRINT
500 INPUT "IDENTIFY 1-16? ":I
505 PRINT
510 IF I=J THEN 530
520 PRINT "NOT CORRECT IDENTIFICATION"
525 C1=C1+1
526 GOTO 500
530 IF B(J,14)=1 THEN 535 ELSE 540
535 PRINT "ALREADY SPOTTED"
536 GOTO 550
540 PRINT "A NEW ONE!"
545 B(J,14)=1
```

```
550 IF H>10 THEN 570
560 GOTO 200
565 REM
570 PRINT "TIME UP"
580 FOR I=1 TO 16
590 IF B(I,14)=1 THEN 595 ELSE 600
595 B1=B1+1
600 NEXT I
610 PRINT "YOUR RATING IS";10*B1-C1
640 END
```

RARE BIRDS MODIFICATIONS

Minor

1. Probability of sighting -- line 40
2. Time interval per turn -- line 400
3. Total time -- line 550
4. Rating formula -- line 610

Major

1. Increase number of birds.
2. Increase characteristics of birds.

Note: The birds' characteristics are stored in decimal format in statements 120, 130, and 140. Statements 50-110 convert the decimal numbers into binary and store the binary digits in B(I,J).

BIRD WATCHING CHART

BIRD	PLACE	WHEN	WHERE	SMALL	BIG	YELLOW	BLUE	SHORT-BEAKED	LONG-BEAKED	MALE	FEMALE
1	S	E	L	S		Y		S		M	
2	W	E	H	S		Y		S			F
3	D	E	L	S		Y			L	M	
4	F	E	H	S		Y			L		F
5	SW	M	L	S			B	S		M	
6	S D	M	H	S			B	S			F
7	S F	M	L	S			B		L	M	
8	WD	M	H	S			B		L		F
9	W F	ME	HL			Y		S		M	
10	DF	ME	HL		B	Y		S			F
11	WDF	ME	HL		B	Y			L	M	
12	S DF	ME	HL		B	Y			L		F
13	SW F	M	HL		B		B	S		M	
14	SWD	M	HL		B		B	S			F
15	SWDF	M	HL		B		B		L	M	
16	SWDF	M	HL		B		B		L		F

DIAMOND THIEF

Scenario

An expensive diamond is stolen from a museum. Your job, as the detective assigned to the case, is to determine who stole the diamond and at what time. You deduce the solution by studying the responses made by five different suspects, one of whom is guilty. Your rating is determined by how quickly you can identify the thief.

The five suspects were wandering through a nine room museum from one p.m. to twelve midnight. They never stayed in the same room for two consecutive hours, although they may have returned to the same room more than once.

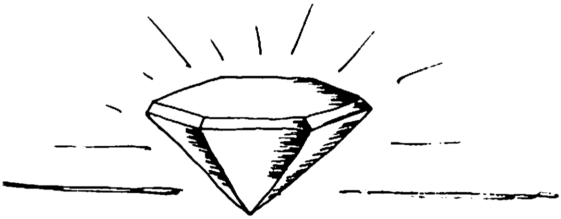
You determine who you want to question and a specific time from one to twelve. The suspect responds by giving the following information:

1. Suspect's location at specified time
2. Whether or not the diamond was seen in room #5 at the specified time
3. Who was with the suspect
4. Who the suspect saw in adjacent rooms

There is a catch, however. The innocent suspects can forget the exact room they were in and may name adjacent rooms 5% of the time instead. There is also a 5% chance that innocent people will make errors in naming people in the room with them or people whom they saw. The thief makes errors 50% of the time. Any statement made about room #5 or any statement made about the diamond is always true.

The diamond was stolen at the end of the time interval; therefore, the thief or people in room #5 with the thief will claim to have seen the diamond during the time it was stolen. Of course, after the diamond was stolen, suspects will not have seen it.

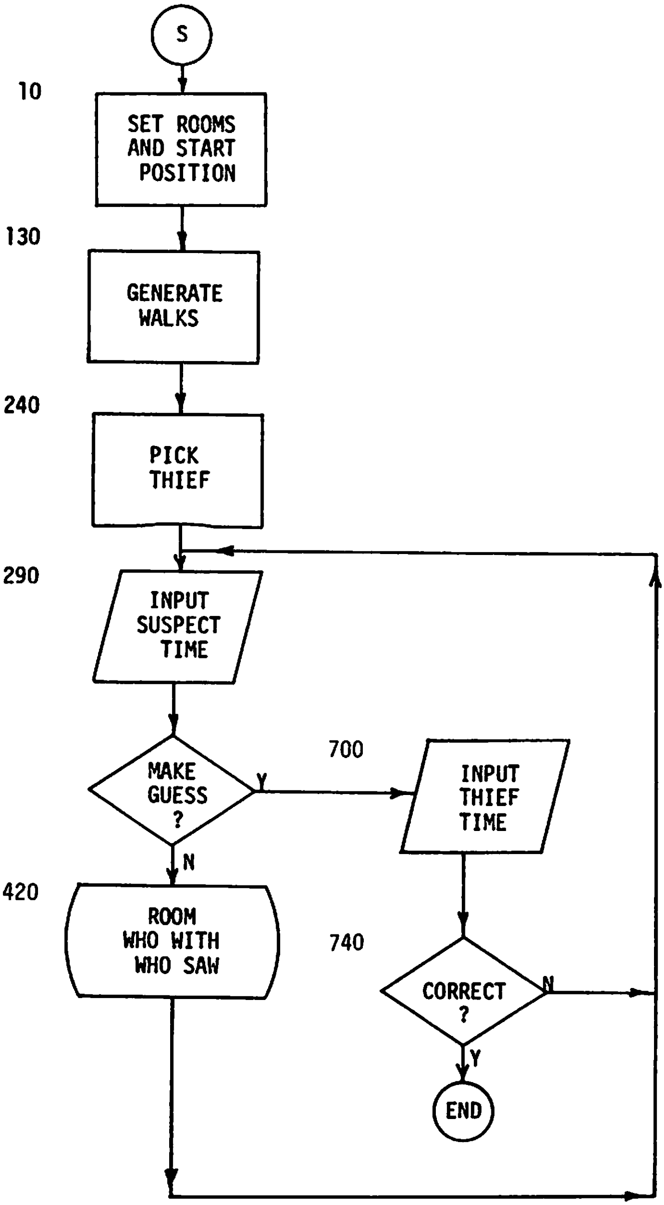
When you think you know who the thief is and the time it was stolen, you should enter a zero in response to "suspect?". If you get either the thief or the time correct, you will get another chance, but will lose a ten question penalty on the final rating.



Sample Run

```
RUN
PLEASE WAIT
SOMEONE STOLE
THE DIAMOND!
QUESTION  1
SUSPECT (1-5)?  1
TIME?  6
SUSPECT 1 AT TIME 6
I WAS IN ROOM 8
I WAS WITH 3
I SAW 4

QUESTION 2
SUSPECT (1-5)?  4
TIME?  6
SUSPECT 4 AT TIME 6
I WAS IN ROOM 9
I SAW 1

QUESTION 3
SUSPECT (1-5)?  2
TIME?  6
I WAS IN ROOM 6
I SAW 4

QUESTION 4
SUSPECT (1-5)?  5
I WAS IN ROOM 1

QUESTION 5
SUSPECT (1-5)?  3
TIME?  7
I WAS IN ROOM 9
I WAS WITH 2
I SAW 4
.
.
.

QUESTION 15
SUSPECT (1-5)?  4
TIME?  4
I WAS IN ROOM 5
I SAW THE DIAMOND
I WAS WITH 3

QUESTION 16
SUSPECT (1-5)?  0
GUILTY SUSPECT?  4
TIME OF CRIME?  4

YOU GOT 'EM
THE THIEF IS 4
AT TIME 4
YOUR RATING IS 84
PLAY AGAIN?
```

DIAMOND THIEF FLOWCHART



DIAMOND THIEF

Variables

A(I,J)	Adjacent rooms
L(I,J)	Room where person I is located at J time
T	Time of theft
D	Thief
P	Probability
S	Suspect
G	Time of guess
A	Temporary variable
I,J,K	Indices

Listing

```
1 REM *** DIAMOND THIEF ***
2 DIM A(9,3),L(5,12)
5 CALL CLEAR
7 RANDOMIZE
10 RESTORE
12 Q=1
14 PRINT "WAIT"
20 FOR I=1 TO 9
30 FOR J=1 TO 3
40 READ AD
50 A(I,J)=AD
60 NEXT J
62 NEXT I
70 DATA 2,4,0,1,3,0,2,6,0
80 DATA 1,5,7,4,6,8,3,5,9
90 DATA 4,8,0,5,7,9,6,8,0
100 FOR I=1 TO 5
110 L(I,1)=INT(RND*9+1)
120 NEXT I
130 FOR I=2 TO 12
140 FOR J=1 TO 5
150 K=INT(3*RND+1)
160 L(J,I)=A(L(J,I-1),K)
170 IF L(J,I)=0 THEN 150
180 NEXT J
182 NEXT I
190 T=INT(12*RND+1)
200 FOR I=1 TO 5
210 IF L(I,T)=5 THEN 240 ELSE 220
220 NEXT I
230 GOTO 190
240 D=INT(5*RND+1)
250 IF L(D,T)<>5 THEN 240
260 PRINT "SOMEONE STOLE THE DIAMOND!"
270 REM    MAIN LOOP
278 PRINT
280 PRINT "QUESTION";Q
290 INPUT "SUSPECT? ":S
```

```
300 IF S<1 THEN 800
310 IF S>5 THEN 290
320 INPUT "TIME? ":G
330 IF (G<1)+(G>12)THEN 320
338 PRINT
340 PRINT "SUSPECT";S;"AT TIME";G;":"
350 IF S=D THEN 355 ELSE 360
355 P=.5
360 IF S<>D THEN 365 ELSE 370
365 P=.05
370 IF (RND>P)+(L(5,6)=5)THEN 375 ELSE 380
375 AD=L(S,G)
376 GOTO 410
380 I=INT(3*RND+1)
390 AD=A(L(S,G),I)
400 IF (AD=0)+(AD=5)THEN 380
409 PRINT
410 PRINT "I WAS IN ROOM";AD
420 IF AD<>5 THEN 450
430 IF T<G THEN 432 ELSE 450
432 PRINT "I DID NOT SEE THE DIAMOND!"
433 GOTO 450
440 PRINT "I SAW THE DIAMOND"
450 IF RND<P THEN 510
460 FOR I=1 TO 5
470 IF I=S THEN 500
480 IF L(S,G)<>L(I,G)THEN 500
490 PRINT "I WAS WITH";I
500 NEXT I
502 GOTO 540
510 I=INT(7*RND+1)
512 IF I=S THEN 510
520 IF I<6 THEN 522
522 PRINT "I WAS WITH";I
540 IF RND<P THEN 640
550 FOR I=1 TO 3
560 AD=A(L(S,G),I)
570 IF AD=0 THEN 610
580 FOR J=1 TO 5
590 IF L(J,G)=AD THEN 592 ELSE 600
592 PRINT "I SAW";J
600 NEXT J
610 NEXT I
620 GOTO 700
640 J=INT(10*RND+1)
650 IF J<5 THEN 652 ELSE 700
652 PRINT "I SAW";J
700 IF RND>P THEN 770
710 K=INT(10*RND+1)
720 IF (K<6)*(K<>J)THEN 722 ELSE 770
722 PRINT "I SAW";K
770 Q=Q+1
772 GOTO 270
800 INPUT "GUILTY SUSPECT? ":S
```

```
810 IF (S<1)+(S>5)THEN 800
820 INPUT "TIME OF CRIME? ":G
830 IF (G<1)+(G>12)THEN 820
840 IF (S=D)*(G=T)THEN 842 ELSE 850
842 PRINT "YOU GOT 'EM!"
844 GOTO 870
850 IF (S=D)+(G=T)THEN 852 ELSE 860
852 PRINT "PARTLY RIGHT"
854 Q=Q+10
856 GOTO 270
860 PRINT "BETTER GIVE UP"
862 Q=Q+100
870 PRINT "THE THIEF IS";D;"AT TIME";T
930 PRINT "YOUR RATING IS "; 100 - Q
940 PRINT ::
950 INPUT "PLAY AGAIN? " :Y$
960 IF Y$="Y" THEN 5
970 END
```

DIAMOND THIEF MODIFICATIONS

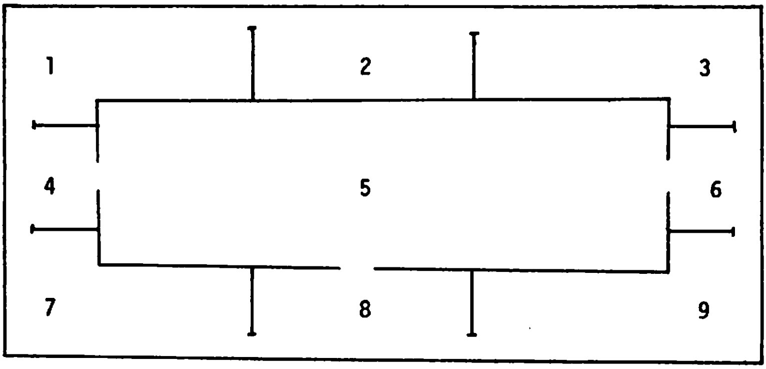
Minor

1. Probability of thief lying -- line 355
2. Probability of innocent suspect lying -- line 365

Major

1. Change room design.
2. Have an accomplice.
3. Jewel is hidden after it is stolen.
4. A guard is roaming around the museum as well.
5. Give suspects and rooms actual names, for example, Mr. Smith is in the Red Room.

MUSEUM FLOOR PLAN



		TIME											
		1	2	3	4	5	6	7	8	9	10	11	12
SUSPECT	1												
	2												
	3												
	4												
	5												

THE DEVIL'S DUNGEON

The Legend

For many years now you have heard rumors of large quantities of gold hidden in a maze of caves whose connecting passageways lead deep beneath the earth of an occasionally active volcano. The stories tell of monsters and demons who roam through the caves, poisonous gas, tremors from the volcano, and one man who returned from these perils alive and named the caves The Devil's Dungeon.

After much searching, you have located the wealthy, solitary man who survived a journey through the dungeon; and he has agreed to see you. Although now very old and in poor health, he tells you everything he can remember about the dungeon.

The Dungeon

There is much gold still remaining in this maze of caves called The Devil's Dungeon; and the stories of demons, monsters and poisonous gas are true. There are sixteen rooms on each level of the dungeon, although many may be blocked by rockfalls caused by volcanic tremors. The number of levels is unknown. Perhaps it is bottomless, for the creatures encountered inside the dungeon were certainly not from the earth as we know it.

Rooms and Passageways

You will begin your adventure in Room #1 at Depth #1. The contents of the room you occupy and the numbers of the adjacent rooms will be listed. You may move to an adjacent room by entering one of the adjacent room numbers. If the output reads: MOVE FROM 2 TO ?, all adjacent rooms on your present level are blocked. If a "slide" to a room is indicated, you may use it by entering that room number; however, it is a one-way passage and cannot be used to return to the first room. A simple map of connecting rooms at each depth will prove invaluable, even though you can receive a list of the rooms you have visited and their respective adjacent rooms any time you enter an 88.

Descending into the Dungeon

Movement to a lower depth can be achieved by using a dropoff. Fifty percent of the rooms at a given depth have dropoffs. To drop to a lower depth, enter any negative number when you are in one of these rooms. You will then find yourself in the same room on the next lower level. The configuration of rooms on this level will not be the same, and a new map must be drawn. Once you have left a given depth, you can never return. You cannot move up.

A dropoff can be created by using the Magic Wand, which you carry with you at all times. The use of the Magic Wand, however, is very risky, because 40% of the time it backfires. When a backfire occurs, your strength and speed are reduced by 50%. When the use of the wand is your only alternative, you must enter 99. If the wand works, it will clear out everything in the room and create a dropoff. If the wand backfires, you will remain in the same place with 50% of the strength and speed you had before using the wand. The Magic Wand can be used repeatedly in every room except Room #1. If you enter a 99 while in Room #1, the simulation will terminate.

Tremors

The contents and arrangements of rooms on each level remain the same throughout the journey. When you return to a room, everything will be the same, except, perhaps, the gold or monster. (See Gold and Monster.) The same passageways will be there leading to the same adjacent rooms, unless a tremor occurs. When a tremor occurs, some of the passageways may be blocked and others may be opened. To determine the effect of a tremor on passageways, you can enter an 88 to get a listing of open adjacent rooms to the rooms you have visited.

Room #1

Room #1 is very important on every level. It is the only room from which you may leave the dungeon by entering a 99. Room #1 is the only place at which you can increase your strength and speed. There are no hazards in this room. When you drop to a lower level, you will want to locate Room #1 as soon as possible.

Speed and Strength

Speed and strength are two qualities that must be maintained throughout your journey in order to survive. Both speed and strength are needed to kill a monster, but speed alone is needed to run from the monster. The curse of a demon affects your speed, and the poisonous gas affects your strength. You begin your journey with 100 units of both speed and strength. Each time you move to another room, your strength and speed will decrease by your depth. If you are at depth #4, the value of both your speed and strength will be decreased by 4 whenever you move. If at any time your strength or speed becomes zero or less, you are declared dead.

Experience

You begin with zero experience points. Everytime you move, your experience points are increased by your depth level number. You can also acquire up to the value of twice a monster's strength in experience points by killing the monster. One experience point is gained for every piece of gold found. Experience points can be traded for strength and speed, one for one, by entering a zero while in Room #1 at any depth. You will then be asked how many points you want added to your speed and to your strength.

Monsters

If a monster is present in a room, its speed and strength will be listed immediately after your speed and strength. If you elect to fight the monster, you must enter a zero. The monsters are faster and stronger in rooms with larger numbers and at lower depths. If your speed is faster than a monster's speed, you have a greater chance of attacking first. If your strength is greater, you have a better chance of killing it. If your speed and strength are two or three times that of the monsters', you will kill them most of the time. When you run from a monster instead of fighting it, speed is important. If a monster hits you on your way out of the room, you will lose 20% of the monster's strength. The monster cannot hit you if you use a dropoff or the Magic Wand in its room.

Demons and Poisonous Gas

About 25% of the rooms on each level have demons and about 25% of the rooms have poisonous gas Neither of these hazards can be eliminated, but you can escape from them. The demons and gas are always in these rooms and they should be avoided when possible. If you enter a room with demons or gas, there is a 40% chance that you will be cursed or gassed. If you are cursed, you will lose one-half of your strength. You can always escape being cursed or gassed by moving to a lower level.

Gold

The maximum amount of gold that could be in a room is stated when you enter the room. This quantity is directly related to the room number and depth. The amount of gold you actually find is given when you leave the room. This amount is a percentage of the maximum, randomly determined. You cannot take gold from a room unless you move to another room on the same level. Once you leave a room carrying gold, the gold is yours for the rest of the journey. Sometimes demons in the room with the gold will steal it as you leave. But whether you leave the room with the gold or demons steal it, when you return to that room, there will no longer be any gold there. You can take gold from a room only one time. If a monster is present in a room containing gold, you must kill the monster before you can take the gold. If you leave the room without killing the monster, the gold and the monster will remain in the room and be there when you return.

Summary

		Enter
In Room #1	to trade experience for strength and speed	0
	to end adventure	99
In any room except #1	to move to adjacent room on the same level	adjacent room #
	to fight monster	0
	to use a dropoff	any negative number
	to use Magic Wand	99
In any room	to list rooms visited	88

Sample Run

```
PLEASE WAIT
GOLD  0    EXP  0    DEPTH  1
SPEED:  100
STRENGTH:  100

SLIDE TO 2
MOVE FROM 1
TO 7?  7
```

```
GOLD  0    EXP  1    DEPTH  1
SPEED:  99
STRENGTH:  99

SLIDE TO 2
MOVE FROM 7
TO 1  2  6?  6
```

```
GOLD  0    EXP  2    DEPTH  1
SPEED:  98
STRENGTH:  98

MONSTER'S SPEED:  6
STRENGTH:  7
DROPOFF
MOVE FROM 6
TO 7  14?  14
```

```
ESCAPED
GOLD  0    EXP  3    DEPTH  1
SPEED:  97
STRENGTH:  97

MAXIMUM GOLD  57
MOVE FROM 14
TO 6?  6
```

```
                    .
                    .
                    .
```

```
GOLD  25    EXP  31    DEPTH  1
SPEED:  94
STRENGTH:  5

MONSTER'S SPEED:  8
STRENGTH:  5
DEMONS
MAXIMUM GOLD  9
MOVE FROM 2
TO 5  7?  0
```

```
YOU ATTACK
MONSTER DEAD!
GOLD  25    EXP  41    DEPTH  1
SPEED:  93
STRENGTH:  91

DEMONS
MAXIMUM GOLD  9
MOVE FROM 2
TO 5  7?  5
```

```
YOU FOUND 6 PIECES OF GOLD
GOLD  31    EXP  48    DEPTH  1
SPEED:  92
STRENGTH:  90

MAXIMUM GOLD  21
MOVE FROM 5
TO 2  3  11?  11
```

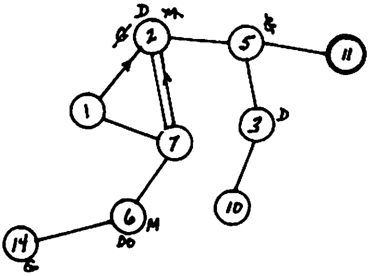
```
                    .
                    .
                    .
```

```
GOLD  46    EXP  70    DEPTH  1
SPEED:  84
STRENGTH:  82

SLIDE TO 2
MOVE FROM 1
TO 7?  0
```

```
EXP  70
SPEED:  84
STRENGTH:  82
ADD SPEED?  34
EXP LEFT  36
ADD STRENGTH?  36
GOLD  46    EXP  0    DEPTH  1
SPEED:  118
STRENGTH:  118

SLIDE TO 2
MOVE FROM 1
TO 7?  7
```

```
                    .
                    .
                    .
```

MAP OF DEPTH 1
DRAWN BY PLAYER



```
GOLD  46    EXP  2    DEPTH  1
SPEED:  116
STRENGTH:  116
MONSTER'S SPEED:  6
STRENGTH:  7

DROPOFF
MOVE FROM 6
TO 7  14?  -1
```

```
GOLD  46    EXP  2    DEPTH  2
SPEED:  114
STRENGTH:  114
MONSTER'S SPEED:  14
STRENGTH:  24

SLIDE TO 9
MOVE FROM 6
TO 2  4  12?  4
```

.
.
.

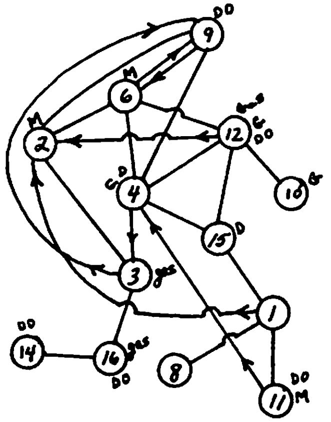
```
GOLD  179    EXP  2    DEPTH  2
SPEED:  138
STRENGTH:  137
MONSTER'S SPEED:  30
STRENGTH:  30

SLIDE TO 4
DROPOFF
MOVE FROM 11
TO 1?  -1
```

```
GOLD  179    EXP  2    DEPTH  3
SPEED:  135
STRENGTH:  134

POISONOUS GAS
SLIDE TO 6
MOVE FROM 11
TO 4  7  13?  7
```

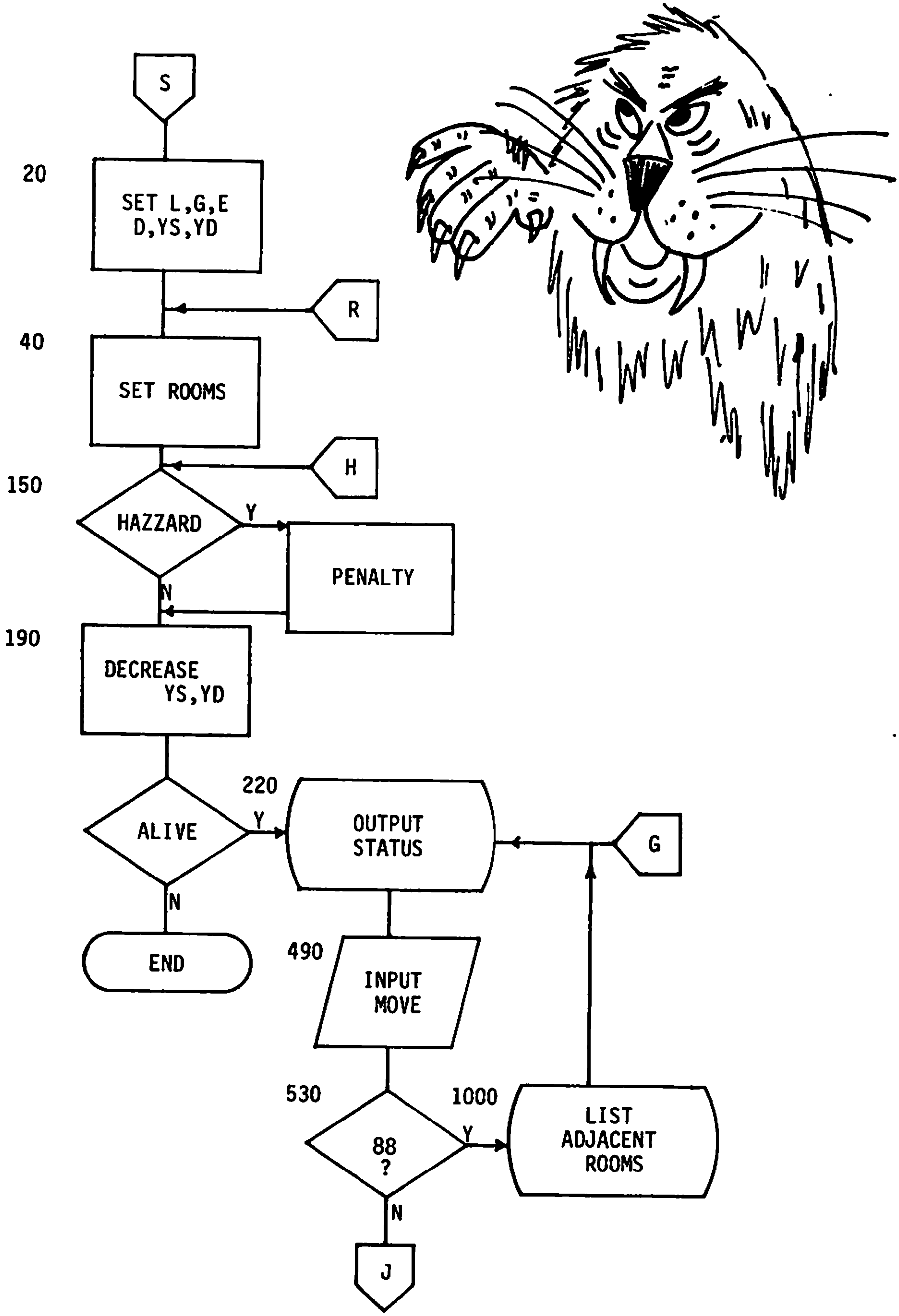
.
.
.

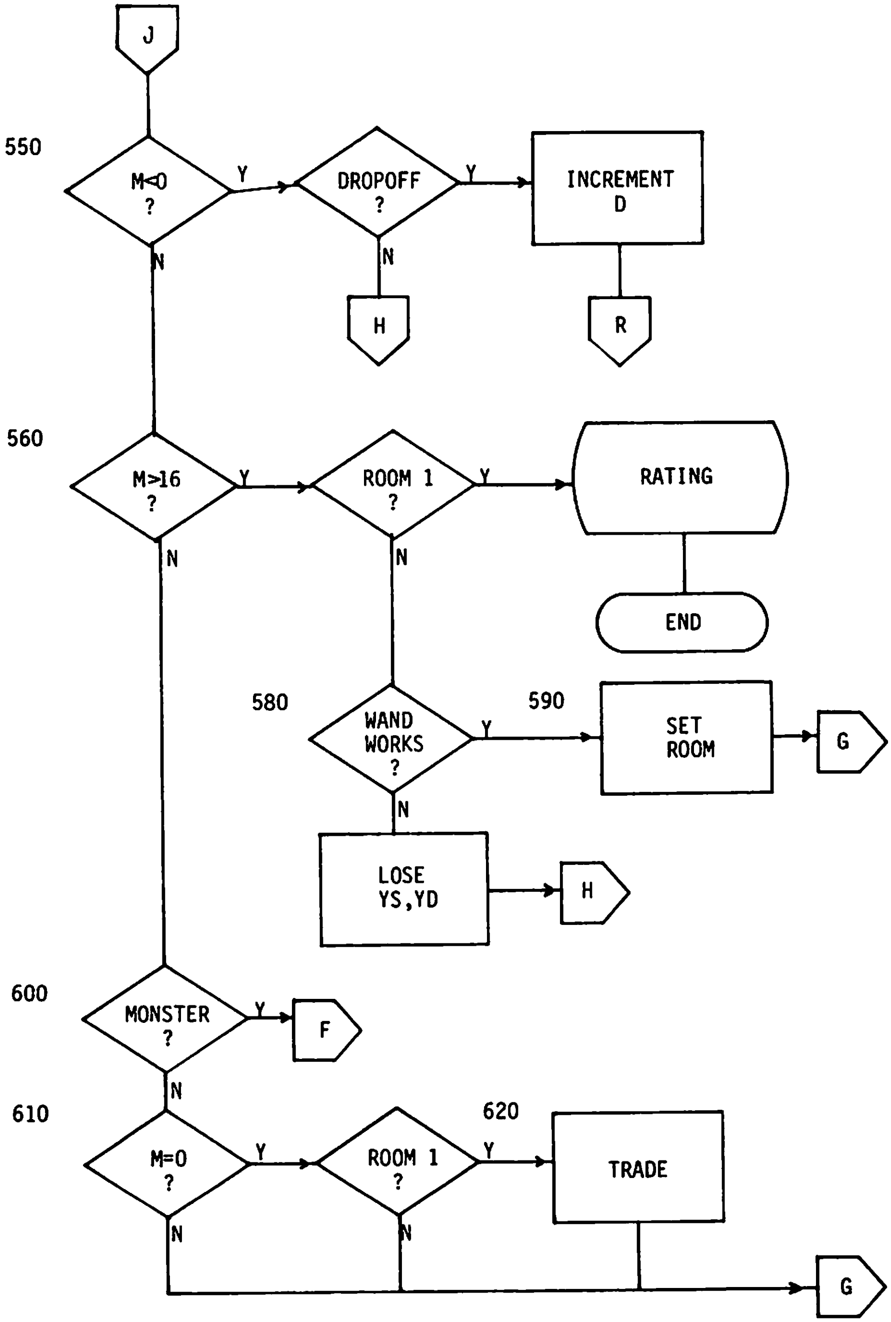
```
GASSED
GOLD  179    EXP  5    DEPTH  3
SPEED:  132
STRENGTH:  64
MONSTER'S SPEED:  42
STRENGTH:  27

MOVE FROM 7
TO 2  6  11  13?  0
```

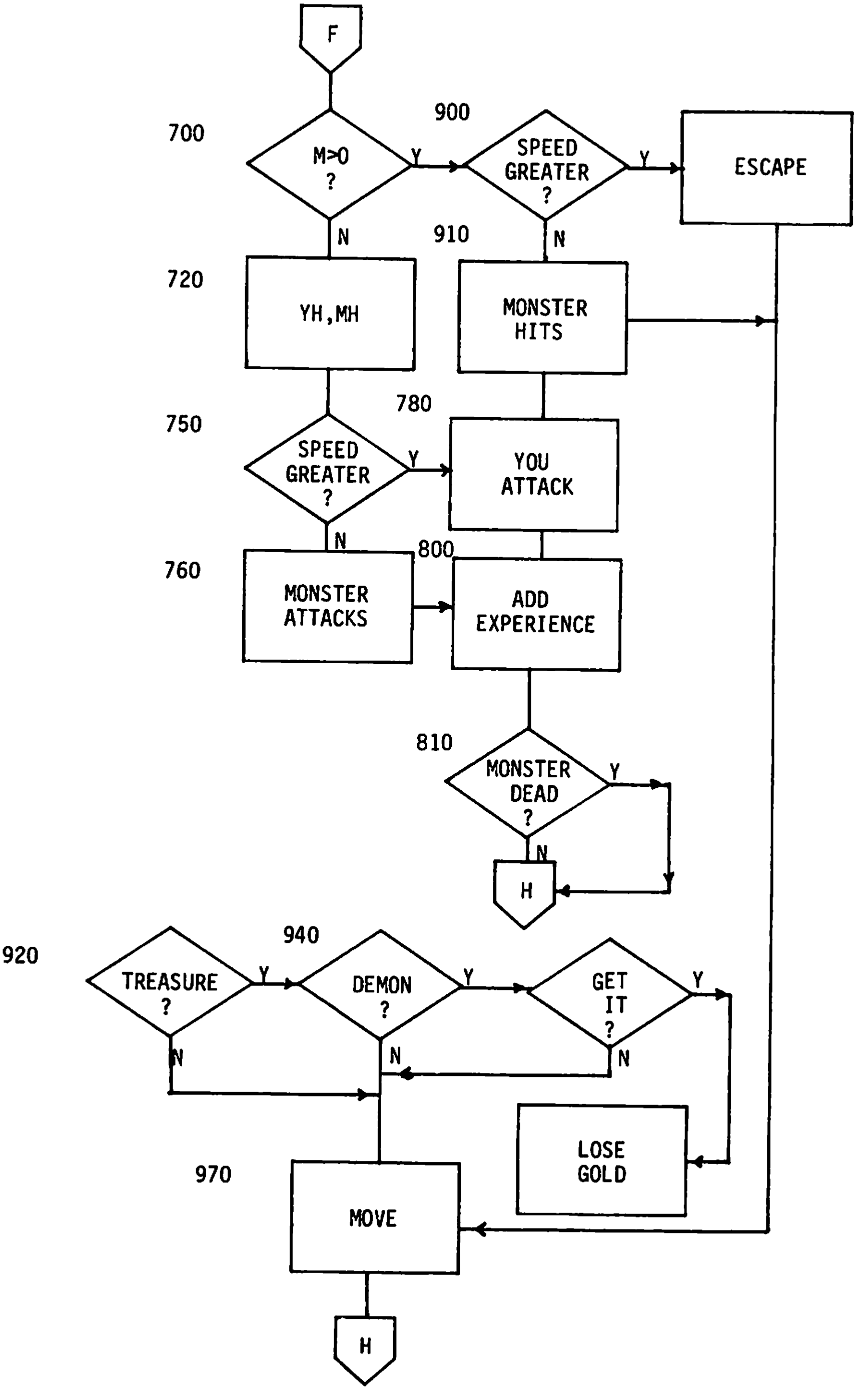
MAP OF DEPTH 2
DRAWN BY PLAYER



THE DEVIL'S DUNGEON FLOWCHART




J
550
M<0
?
Y
N
DROPOFF
?
Y
N
INCREMENT
D
H
R
560
M>16
?
Y
N
ROOM 1
?
Y
N
RATING
END
580
WAND
WORKS
?
Y
N
590
SET
ROOM
G
LOSE
YS,YD
H
600
MONSTER
?
Y
N
F
610
M=0
?
Y
N
ROOM 1
?
Y
N
620
TRADE
G


F
700
M>0
?
Y
N
900
SPEED
GREATER
?
Y
N
ESCAPE
910
MONSTER
HITS
720
YH,MH
750
SPEED
GREATER
?
Y
N
780
YOU
ATTACK
760
MONSTER
ATTACKS
800
ADD
EXPERIENCE
810
MONSTER
DEAD
?
Y
N
H
920
TREASURE
?
Y
N
940
DEMON
?
Y
N
GET
IT
?
Y
N
LOSE
GOLD
970
MOVE
H

THE DEVIL'S DUNGEON PROGRAM

Variables

R(16)	0 - 524287	Specifies contents of room
L(65)	1 - 16	Lists adjacent rooms
F(16)	0 or 1	Set flags for adjacent rooms
X(19)	0 or 1	Flags for room contents (see below)
B(16)	0 or 1	Flags rooms already visited
L	1 - 16	Your location
G_1		Amount of gold in room -- depends on depth, size of room, and random factor
G		Total amount of gold that you have accumulated
E		Total experience points -- gained by moving, fighting, running, collecting gold -- can be traded for strength and speed
D	1 - ∞	Depth
YS		Your strength -- you die if it drops to 0
YD		Your speed -- you die if it drops to 0
YH		Your hit when fighting
MS		Monster's strength -- depends upon depth, size of room, and random factor
MD		Monster's speed
MH		Monster's hit when fighting
I,J		Indices
F	0 or 1	Flag for monster present
N,Q,R		Temporary variables
S		Slide
M		Move to
T		Treasure
S(1),X(12)		Demon
X(2)		Monster
S(3),X(4),X(5)		Monster's strength
X(6),X(7),X(8)		Monster's speed
X(9),X(11)		Poisonous gas
X(10)		Treasure
S(14)		Slide
X(15) - X(18)		Slide to room
X(19)		Dropoff
X		Number of rooms

Listing

```
1 REM *** DEVIL'S DUNGEON ***
5 REM  SET
7 CALL CLEAR
8 RANDOMIZE
10 DIM R(16),L(65),F(16),X(19),B(16)
20 LL=1
22 XX=16
24 G=0
26 E=0
30 D=1
32 YS=101
34 YD=101
40 FOR I=0 TO 65
42 L(I)=0
```

```
44 NEXT I
50 FOR I=1 TO XX
52 N=INT(3*RND+1)
54 IF I=N THEN 62 ELSE 70
62 N=3
70 FOR J=1 TO N
80 RR=INT(64*RND+1)
90 IF L(RR)<>0 THEN 80
100 L(RR)=I
110 NEXT J
120 R(I)=INT(524287*RND)
122 B(I)=0
130 NEXT I
132 B(LL)=1
140 R(1)=24576
142 FOR I=1 TO 19
144 X(I)=0
146 NEXT I
148 REM  HAZARDS
150 IF RND<.01 THEN 152 ELSE 160
152 PRINT "TREMOR"
153 FOR I=1 TO 20
154 L(I)=INT(XX*RND+1)
156 NEXT I
160 IF RND<.01 THEN 162 ELSE 170
162 PRINT "TREMOR"
163 FOR I=1 TO 20
164 L(I)=0
165 NEXT I
170 IF (X(1)*X(12)=1)*(RND<.4)THEN 172 ELSE 180
172 PRINT "CURSED!"
173 YD=INT(.5*YD)
180 IF (X(9)*X(11)=1)*(RND<.4)THEN 182 ELSE 190
182 PRINT "GASSED!"
183 YS=INT(.5*YS)
185 REM  TEST
190 YD=YD-D
200 YS=YS-D
210 IF (YS<=0)+(YD<=0)THEN 212 ELSE 215
212 PRINT "YOU'RE DEAD"
213 END
215 REM  OUTPUT
220 PRINT ::
222 PRINT "GOLD";G
230 PRINT "EXPERIENCE";E
235 PRINT "DEPTH";D
240 PRINT "YOUR SPEED";YD
241 PRINT "YOUR STRENGTH";YS
245 GOSUB 250
247 GOTO 310
249 REM  ADJACENT ROOMS
250 FOR I=1 TO XX
252 F(I)=0
254 NEXT I
```

```
260 FOR I=1 TO 64
262 IF LL<>L(I)THEN 300
280 IF (L(I+1)<>0)*(L(I+1)<>LL)THEN 282 ELSE 290
282 F(L(I+1))=1
290 IF (L(I-1)<>0)*(L(I-1)<>LL)THEN 292 ELSE 300
292 F(L(I-1))=1
300 NEXT I
302 RETURN
305 REM  CONVERT
310 N=R(LL)
320 FOR I=1 TO 19
322 Q=INT(N/2)
324 X(I)=2*(N/2-Q)
326 N=Q
327 NEXT I
329 REM  MONSTERS, DEMONS, GAS
330 IF X(2)=0 THEN 332 ELSE 340
332 MS=0
333 GOTO 380
340 IF FF=1 THEN 370
350 MS=D*(X(3)+2*X(4)+4*X(5)+LL)
360 MD=D*(X(6)+2*X(7)+4*X(8)+LL)
370 PRINT "MONSTER'S SPEED";MD
372 PRINT "STRENGTH";MS
380 IF X(1)*X(12)=1 THEN 382 ELSE 390
382 PRINT "DEMONS"
390 IF X(9)*X(11)=1 THEN 392 ELSE 400
392 PRINT "GAS"
395 REM  TREASURE
400 IF X(10)<>1 THEN 402 ELSE 410
402 T=0
403 GOTO 430
410 T=X(11)+2*X(12)+4*X(13)+1
420 PRINT "MAXIMUM GOLD";T*LL*D+1
425 REM  SLIDES AND DROPOFFS
430 S=X(15)+2*X(16)+4*X(17)+8*X(18)+1
440 IF S>XX THEN 442 ELSE 450
442 S=1
450 IF S=0 THEN 452 ELSE 460
452 S=1
460 IF (X(14)=0)+(S=LL)THEN 480
465 PRINT
470 PRINT "SLIDE TO: ";S
480 IF X(19)*X(13)=1 THEN 482 ELSE 490
482 PRINT "DROPOFF"
485 REM  INPUT MOVE
490 PRINT "MOVE FROM";LL;"TO";
500 FOR I=1 TO XX
510 IF (F(I)=1)*(I<>LL)THEN 512 ELSE 520
512 PRINT I;
520 NEXT I
530 INPUT M
532 IF M=88 THEN 1000
540 IF (M<0)*(X(19)*X(13)=1)THEN 542 ELSE 550
```

```
542 D=D+1
543 FF=0
544 GOTO 40
550 IF M<0 THEN 552 ELSE 560
552 PRINT "NO DROPOFF"
553 GOTO 150
560 IF (M>XX)*(LL=1)THEN 562 ELSE 570
562 PRINT "YOU FOUND";G;"GOLD"
563 END
570 IF M<XX THEN 600
575 REM  MAGIC WAND
580 IF RND<.4 THEN 582 ELSE 590
582 PRINT "BACKFIRE"
583 YS=INT(.5*YS)
584 YD=INT(.5*YD)
585 GOTO 150
590 PRINT "WAND WORKS"
592 R(LL)=266240
593 GOTO 220
595 REM  TRADE
600 IF MS>0 THEN 700
610 IF (M<>0)+(LL<>1)THEN 920
619 PRINT
620 PRINT "EXPERIENCE";E
622 PRINT "SPEED";YD
624 PRINT "STRENGTH";YS
625 INPUT "ADD SPEED? ":N
630 IF E-N<0 THEN 632 ELSE 640
632 PRINT "NEED MORE EXPERIENCE"
634 GOTO 620
640 E=E-N
642 YD=YD+N
644 PRINT "EXPERIENCE LEFT";E
650 INPUT "ADD STRENGTH? ":N
660 IF E-N<0 THEN 662 ELSE 670
662 PRINT "NEED MORE EXPERIENCE"
664 GOTO 650
670 E=E-N
672 YS=YS+N
680 GOTO 220
695 REM  FIGHT
700 FF=1
710 IF M>0 THEN 900
720 YH=INT(RND*YS)
722 MH=INT(RND*MS)
730 IF YH>MS THEN 732 ELSE 740
732 YH=MS
740 IF MH>YS THEN 742 ELSE 750
742 MH=YS
750 IF RND*YD>RND*MD THEN 780
760 PRINT "MONSTER ATTACKS"
762 YS=YS-MH
764 MS=MS-INT(.5*MH)
770 GOTO 800
```

```
780 PRINT "YOU ATTACK"
782 MS=MS-YH
784 YS=YS-INT(.5*MH)
800 E=E+2*YH
810 IF MS<=0 THEN 812 ELSE 820
812 PRINT "MONSTER DEAD"
814 R(LL)=R(LL)-2
816 GOTO 150
820 PRINT "MONSTER STILL ALIVE"
825 GOTO 150
895 REM  RUN
900 IF RND*YD>RND*MD THEN 902 ELSE 910
902 PRINT "ESCAPED"
904 GOTO 970
910 PRINT "MONSTER HIT YOU"
912 YS=YS-INT(.2*MS)
913 GOTO 970
915 REM  TREASURE
920 IF T=0 THEN 970
930 G1=INT(RND*T*LL*D)+1
940 IF (X(1)*X(12)=1)*(RND<.4)THEN 942 ELSE 950
942 PRINT "DEMON GOT GOLD"
943 G1=0
950 PRINT "YOU FOUND";G1;"GOLD"
952 G=G+G1
954 R(LL)=R(LL)-512
960 E=E+G1
965 REM  MOVE
970 IF (F(M)=1)+(M=S)THEN 972 ELSE 980
972 LL=M
973 FF=0
974 E=E+D
975 B(LL)=1
976 GOTO 150
980 PRINT "NOT ADJACENT"
1000 L1=LL
1002 FOR K=1 TO XX
1010 IF B(K)<>1 THEN 1070
1020 PRINT K;"--";
1030 LL=K
1035 GOSUB 250
1040 FOR J=1 TO XX
1050 IF (F(J)=1)*(J<>K)THEN 1052 ELSE 1060
1052 PRINT J;
1060 NEXT J
1062 PRINT
1070 NEXT K
1080 LL=L1
1085 GOTO 220
```

THE DEVIL'S DUNGEON MODIFICATIONS

Minor

1. To change initial amount of gold or initial amount of experience, change the appropriate variable in line 20.
2. To begin at a lower level, increase D in line 30.
3. To begin with a different amount of strength or speed, change YS and/or YD in lines 32/34.
4. To increase the probability of a tremor, increase .01 in line 150 and/or line 160.
5. To increase the probability of being cursed by a demon/gassed, increase the .4 in line 170.
6. To increase the effect of being cursed/gassed, decrease the .5 in lines 173/183.
7. To double the monster's strength/speed, insert a statement, MS=2*MS/MD=2*MD at line 355/365.
8. To increase the probability of demons/gas in a room from 25% to 50%, remove the X(12)/X(11) from lines 170/180 and 380/390.
9. To double the treasure, insert the statement, T=2*T in line 415.
10. To increase the probability of a dropoff in a room from 25% to 50%, remove the X(13) from lines 480 and 540.
11. To increase the probability of the wand backfiring, increase the .4 in line 580.
12. To increase the effect of the wand backfiring, decrease the .5 in lines 583/584.
13. To increase the amount the monster loses/you lose when attacking, increase the .5 in lines 764/784.
14. To increase the amount of experience you gain while fighting, increase the 2 in line 800.
15. To increase the amount you lose when getting hit while running from the monster, increase the .2 in line 912.

Major

1. Weapons and equipment must be bought with gold before starting on the journey.
2. There could be different sized monsters, determined by the expression, X(3)+2*X(4)+4*X(5) in line 350. Each monster could be named, ie, Glub, Knaw, Slurp, Hairy, ...
3. The treasures could be in various sized containers, determined by the expression, X(11)+2*X(12)+4*X(13) in line 410.
4. The number of rooms at each level could be determined randomly.
5. Some rooms could be light and others dark.
6. Some monsters or demons could appear at random rather than be assigned to specific rooms.
7. A mean magician could relocate you in another room.
8. You could accidentally fall into a pit that drops you to a lower level.

LIFE

Scenario

In this simulation, you begin as a bum with one hundred dollars in your pocket. Your objective is to become an executive of a large company and earn lots of money.

As a bum, you won't have many choices in the beginning. The M's next to each category indicate that you have reached your maximum potential in that category. After a few turns, one or more of the M's will disappear, allowing you to move the "*" with the up or down arrow to the appropriate category. The right or left arrow can be used to select a level within the category. Each category has five levels.

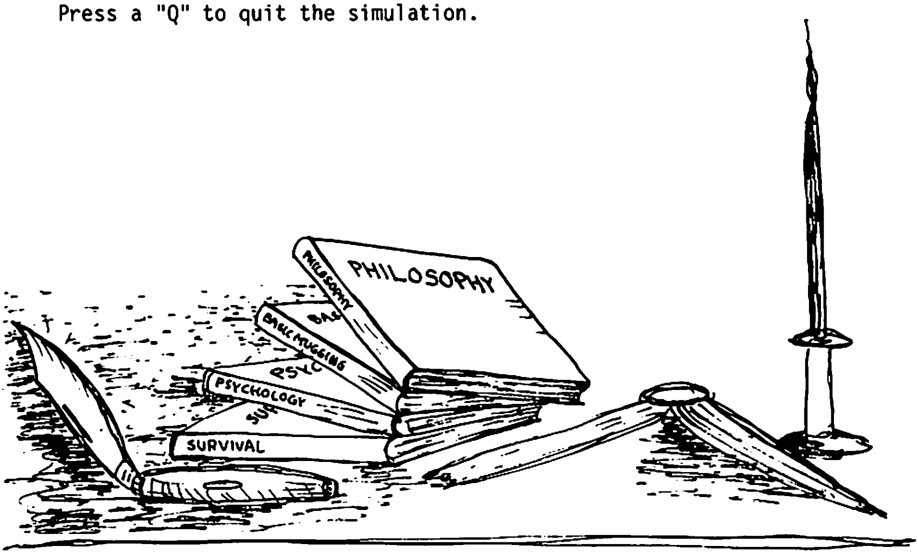
One of the first categories to become available to you for a choice will be education. You should take advantage of the educational opportunity and attend high school. You must attend at each educational level for four turns before you can progress to the next level. The number to the far right of the educational level indicates the number of turns you must attend at that level before you may move on. Education is important because it opens the door to better jobs -- at which you will earn more money.

Your recreation as a bum is mugging. Because this is also your only form of income, it might be wise to stay with it until a job opportunity comes along. But if you get sent to jail for mugging, your future job opportunities will be limited.

In making "life choices," you can live beyond your means, since it is OK to have negative cash.

The higher the level you attain in each category, the more money you can make on the job. In other words, a high life style tends to influence your income in a positive way.

Press a "Q" to quit the simulation.



Sample Run

```
                              LIFE CHOICES

JOB                         M * BUM
EDUCATION                   M   SELFTAUT (SIC)                  1
FOOD                        M   DUMPSTER
FUN                         M   MUGGING
HOUSING                     M   STREET
TRANSPORT                   M   FEET
CLOTHING                    M   JEANS



                         PRESS ARROWS OR ENTER
```

```
                              LIFE EVENTS

JOB
EDUCATION                   FOUND BOOK
FOOD
FUN
HOUSING
TRANSPORT                   FLAT TIRE
CLOTHING                    FOUND OLD SHOES



                              PRESS ENTER
```

```
                               FINANCIAL

EXPENSES:                               INCOME:

EDUCATION                  0            CASH               119
FOOD                      16            MOM                 12
FUN                        0            MUGGING             14
HOUSING                    0                              ____
TRANSPORT                  0
CLOTHING                   3            TOTAL              145
                        ____

TOTAL                     19            >>>>>   -           19

                                                          ____
                                   BALANCE                 126



                              PRESS ENTER
```

LIFE

Variables

TU	Turn number
R	Row
CH	Cash on hand
I,J,K	Temporary variables
KK	Constant
ST(5)	Status
MX(7)	Maximum for category
TI$(7)	Titles
LE$(5)	Labels for levels
X$	Temporary
S	Status
K	Key input
RI, RE	Random event
JA	Jail level
G$	Good event
B$	Bad event
C(7,5)	Cost
EX	Expense
TOT	Total
ED(5)	Educational level

Program Listing

```
1 REM *** LIFE ***
100 RESTORE :: RANDOMIZE
110 TU=0 :: R=1 :: CH=100
120 CALL CLEAR
130 FOR I=1 TO 7
140 ST(I)=1 :: MX(I)=1
150 READ TI$(I)
160 NEXT I
170 FOR I=1 TO 7
180 FOR J=1 TO 5
190 READ LE$(I,J),C(I,J),G$(I,J),B$(I,J)
200 NEXT J
210 NEXT I
215 LE$(2,6)="ED TV"
230 REM START LOOP
240 CALL CLEAR :: TU=TU+1
250 DISPLAY AT(1,9)BEEP:"LIFE CHOICES"
260 DISPLAY AT(24,5):"PRESS ARROWS OR ENTER"
270 FOR I=1 TO 7
280 IF ST(I)=MX(I)THEN X$="M" ELSE X$=" "
290 DISPLAY AT(I+5,1):TI$(I)
300 DISPLAY AT(I+5,13):X$
310 DISPLAY AT(I+5,15):LE$(I,ST(I))
320 NEXT I
330 R=6 :: DISPLAY AT(7,26):ED(ST(2))
340 DISPLAY AT(1+5,14)SIZE(1):"*"
350 CALL KEY(3,K,S) :: IF S=0 THEN 350
360 DISPLAY AT(R,14)SIZE(1):" "
```

```
370 IF K=88 THEN R=R+1 :: IF R=13 THEN R=6
380 IF K=69 THEN R=R-1 :: IF R=5 THEN R=12
390 DISPLAY AT(R,14)SIZE(1):"*"
400 IF K=83 THEN ST(R-5)=ST(R-5)-1
        :: IF ST(R-5)=0 THEN ST(R-5)=1
410 IF K=68 AND ST(R-5)<MX(R-5)THEN
        ST(R-5)=ST(R-5)+1
420 DISPLAY AT(R,15):LE$(R-5,ST(R-5))
430 IF R=7 THEN DISPLAY AT(7,26):ED(ST(2))
440 IF ST(R-5)=MX(R-5)THEN X$="M" ELSE X$=" "
450 DISPLAY AT(R,13)SIZE(1):X$
460 IF K=13 THEN 490
465 IF K=81 THEN END
470 K=0 :: GOTO 350
490 REM RANDOM EVENTS
500 RI=0
510 IF ST(4)=1 THEN RI=RI+INT(RND*20)
520 IF ST(4)=1 AND RND>.9 THEN CALL CLEAR ::
        DISPLAY AT(22,12):"IN JAIL" ::
        CH=.5*CH :: JA=JA+.1 :: GOTO 680
530 FOR I=1 TO 7
540 KK=5↑(ST(I)):: KK=KK+3
550 DISPLAY AT(1,9)BEEP:"LIFE EVENTS"
560 DISPLAY AT(I+5,13):""
570 IF RND>.1 THEN 660
580 RE=INT(RND*3+1)
590 ON RE GOSUB 600,600,630 :: GOTO 660
600 DISPLAY AT(I+5,13):G$(I,ST(I));::
        RI=RI+INT(RND*KK)
610 IF I>2 THEN C(I,ST(I))=C(I,ST(I))*(1-RND*.1)
620 RETURN
630 DISPLAY AT(I+5,13):B$(I,ST(I));::
        RI=RI-INT(RND*KK)
640 IF I>2 THEN C(I,ST(I))=C(I,ST(I))*(1+RND*.1)
650 RETURN
660 NEXT I
670 DISPLAY AT(24,1):" "
680 DISPLAY AT(24,10):"PRESS ENTER"
690 CALL KEY(3,K,S):: IF S=0 THEN 690
700 REM STYLE INCREASE
710 S=0
720 FOR I=3 TO 7 :: S=S+ST(I):: NEXT I
730 C(1,ST(1))=C(1,ST(1))*(1+S/400)
750 REM FINANCIAL
760 CALL CLEAR
770 DISPLAY AT(1,10)BEEP:"FINANCIAL"
780 DISPLAY AT(3,1):"EXPENSES:" ::
        DISPLAY AT(3,17):"INCOME:"
790 FOR J=2 TO 7
800 DISPLAY AT(J+3,1):TI$(J)
810 NEXT J
820 EX=0
830 FOR J=2 TO 7
840 REM INFLATION
```

```
850 FOR K=1 TO 5
860 C(J,K)=C(J,K)*(1+RND*.1)
870 NEXT K
880 EX=EX+C(J,ST(J))
890 DISPLAY AT(J+3,12):USING "####":C(J,ST(J))
900 NEXT J
910 DISPLAY AT(12,1):"TOTAL" ::
       DISPLAY AT(11,12):"----"
920 DISPLAY AT(12,12):USING "####":EX
930 DISPLAY AT(6,17):"JOB"
940 IF ST(4)=1 AND ST(1)=1 THEN DISPLAY
       AT(6,17):"MOM"
950 DISPLAY AT(6,25):USING "####":C(1,ST(1))
960 DISPLAY AT(5,17):"CASH"
970 DISPLAY AT(5,24):USING "#####":CH
980 DISPLAY AT(7,17):"MISC"
990   IF   ST(4)=1   AND   ST(1)=1   THEN   DISPLAY
AT(7,17):"MUGGING" :: RI=RI+10
1000 DISPLAY AT(7,25):USING "####":RI
1010 DISPLAY AT(8,24):"-----" ::
       DISPLAY AT(10,17):"TOTAL"
1020 TOT=CH+RI+C(1,ST(1))
1030 DISPLAY AT(10,24):USING "#####":TOT
1040 DISPLAY AT(13,24):"-----"
1050 DISPLAY AT(12,17)BEEP:">>>>> -" ::
       DISPLAY AT(12,25):USING "####":EX
1060 DISPLAY AT(15,13):"BALANCE" ::
       DISPLAY AT(15,22):USING "#######":TOT-EX
1070 CH=TOT-EX
1075 DISPLAY AT(17,13):"TURN # ";TU
1080 DISPLAY AT(24,10):"PRESS ENTER"
1090 CALL KEY(3,K,S):: IF S=0 THEN 1090
1110 REM ADJUST MAXIMUMS
1120 REM JOBS
1125 IF RND>.9-JA THEN 1160
1130 FOR J=1 TO 4
1140 IF ED(J)>=4 THEN MX(1)=J+1
1150 NEXT J
1160 IF ED(5)>=4 THEN C(1,5)=C(1,5)+500
1170 REM EDUCATION
1180 FOR J=1 TO 5
1190 IF ST(2)=J THEN ED(J)=ED(J)+1
1200 IF ED(J)>=4 THEN MX(2)=J+1
1210 NEXT J
1220 REM ADJUST THE REST
1225 IF TU>12 THEN KK=.7 ELSE KK=.9
1230 FOR J=3 TO 7
1240 IF RND>KK AND MX(J)<5 THEN MX(J)=MX(J)+1
1250 NEXT J
1260 GOTO 240
1270 DATA JOB,EDUCATION,FOOD,FUN,HOUSING,
       TRANSPORT,CLOTHING
1280 DATA BUM,12,FOUND MONEY,GOT ROBBED
1290 DATA TRAINEE,500,BIG BONUS,LOST TOOLS
```

```
1300 DATA SALES,700,NICE COMMISSION,LOST SALE
1310 DATA MANAGER,1500,PROMOTION,DEMOTION
1320 DATA EXECUTIVE,4000,GOOD TRANSFER,BAD TRANSFER
1330 DATA SELFTAUT,0,FOUND BOOK,HIT ON HEAD
1340 DATA HIGH SCHOOL,10,A IN ALGEBRA,FLUNKED
      BIOLOGY
1350 DATA SPECIAL,40,B IN DRAFTING,D IN MECHANICS
1360 DATA B.S.,200,A IN SPANISH,D IN CALCULUS
1370 DATA M.BA.,300,A IN MANAGEMENT,F IN ACCOUNTING
1380 DATA DUMPSTER,15,FOUND APPLE,FOOD POISONING
1390 DATA FAST-FOOD,150,INVITED TO MOM'S,ANEMIA
1400 DATA BASIC,100,VISIT FRIENDS,OVERATE
1410 DATA BALANCED,150,TAKEN TO DINNER,GOT STUFFED
1420 DATA GOURMET,300,GREAT MEAL,OVER INDULGED
1430 DATA MUGGING,0,BIG HEIST,GOT CAUGHT
1440 DATA TV,5,GREAT MOVIE,TUBE BLEW
1450 DATA MOVIES,20,GREAT MOVIE,DULL MOVIE
1460 DATA PARTIES,50,MET A FRIEND,A DULL PARTY
1470 DATA ORGIES,200,MET SEVERAL FRIENDS,GOT SICK
1480 DATA STREET,0,FOUND SHELTER,GOT ROBBED
1490 DATA HOME,50,BIG BED,NO PRIVACY
1500 DATA ROOMMATE,150,SHARE EXPENSES,LOUD ROOMMATE
1510 DATA APARTMENT,300,NICE PRIVACY,LOUD NEIGHBOR
1520 DATA TOWN HOUSE,600,JAZZY PLACE,BIG
      MAINTENANCE
1530 DATA FEET,0,LEARNED TO JOG,SPRAINED FOOT
1540 DATA BIKE,5,SAVE ON GAS,FLAT TIRE
1550 DATA KLUNKER,150,CHEAP PARTS,BAD TRANSMISSION
1560 DATA WAGON,250,LOTS OF ROOM,BAD FUEL PUMP
1570 DATA MERCEDES,400,GOOD MECHANIC,TUNE UP
1580 DATA JEANS,2,FOUND SHOES,GOT FLEES
1590 DATA CASUAL,40,CLOTHING SALE,TORE PANTS
1600 DATA POLYESTER,40,CLOTHING GIFT,PANTS RIPPED
1610 DATA SUITS,100,BIG SALE,RIPPED PANTS
1620 DATA CUSTOM,250,BIG SALE,CHANGE IN STYLE
```

LIFE MODIFICATIONS

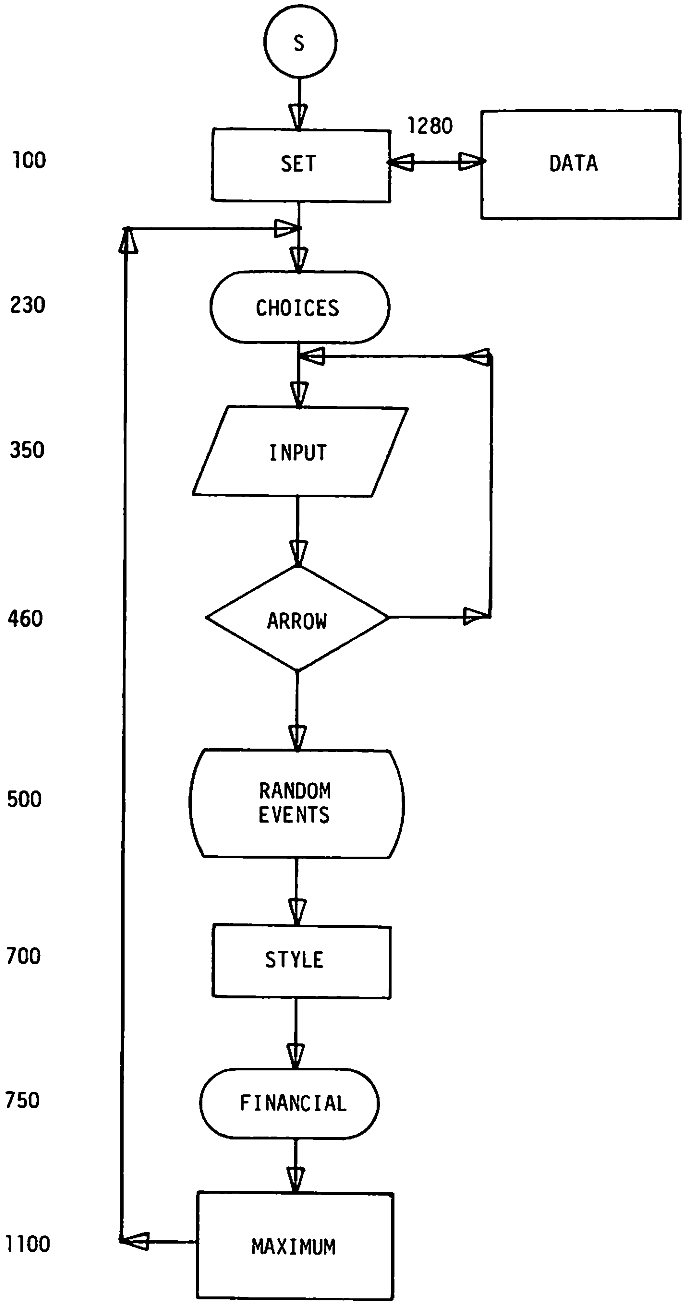
Minor

1. Starting cash -- line 110
2. Mugging income -- lines 510, 990
3. Probability of jail -- line 520
4. Penalty for jail -- line 520
5. Random event costs -- line 540
6. Rate of inflation -- line 860
7. Probability of jobs -- line 1225
8. Advanced education -- line 1160
9. Change words in data -- lines 1270-

Major

1. Number of categories and levels
2. Loans and interest
3. Income tax
4. Define relationships between variables

LIFE FLOWCHART



6404-7

STIMULATING SIMULATIONS FOR THE TI-99/4A™

C. W. Engel

Here is an exciting handbook containing thirteen BASIC "simulation programs," which are actually game programs. Each of the programs is presented with a listing, sample run, instructions, and program documentation, including a flowchart and ideas for variations. "This book is a good starting point for the computer hobbyist who wishes to explore the use of the small computer in simulating real events." *Computer Notes*

HAYDEN BOOK COMPANY, INC.
Hasbrouck Heights, New Jersey

ISBN 0-8104-6404-7