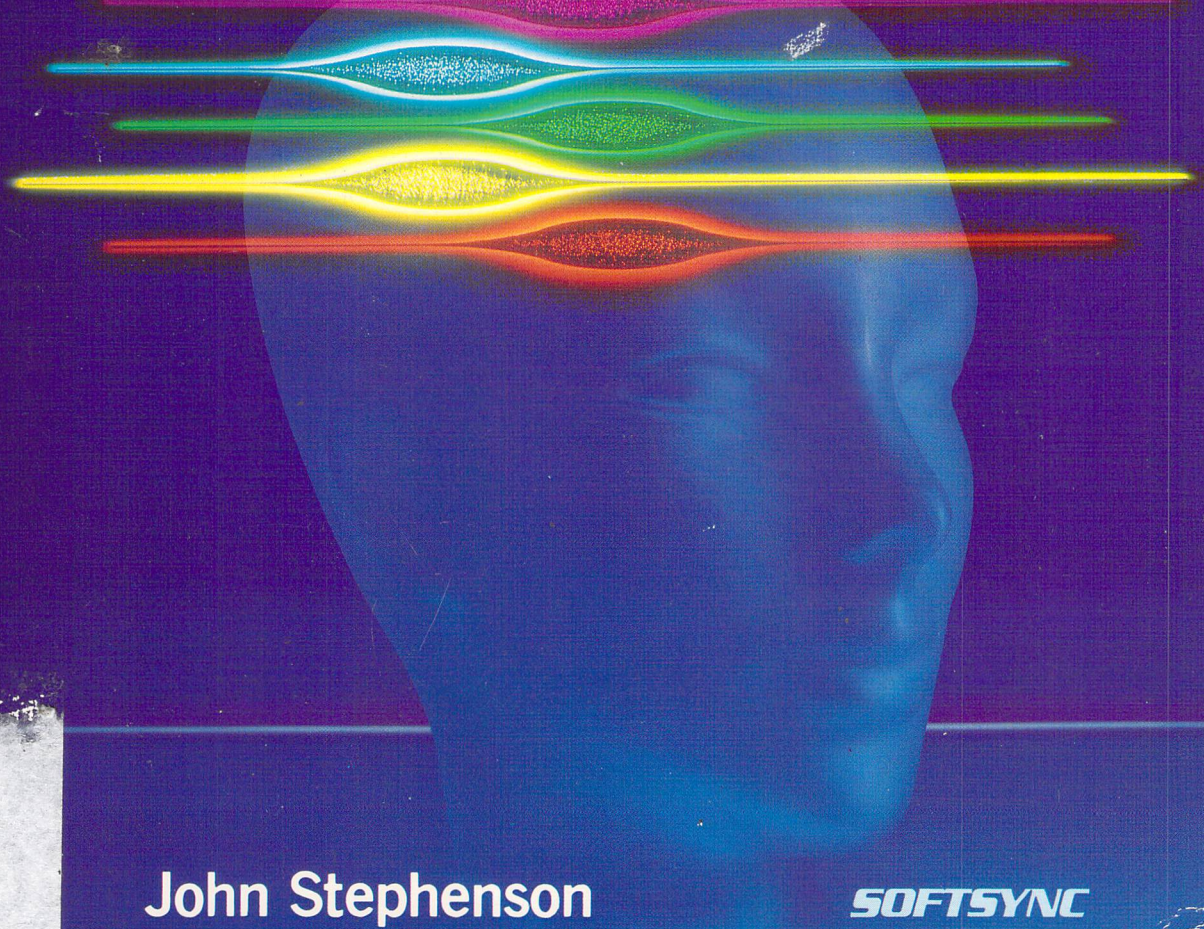
BRAIN GAMES for Kids & Adults Using the Commodore 64



John Stephenson

SOFTSYNC

BRAIN GAMES FOR KIDS AND ADULTS
Using the Commodore 64

Publishing Director: David Culverwell
Acquisitions Editor: Susan Love
Production Editor/Text Designer: Roberta Glencer
Art Director/Cover Design: Don Sellers
Assistant Art Director: Bernard Vervin
Cover Photography: George Dodson
Manufacturing Director: John A. Komsa

Typesetter: Sogitec, Inc., Lakewood, CA
Printer: R.R. Donnelley and Sons Company, Harrisonburg, VA
Typefaces: News Gothic (text), Antique (display), Eterna Monospace, (programs)

BRAIN GAMES FOR KIDS AND ADULTS Using the Commodore 64

John W. Stephenson, PhD
University of Saskatchewan
Saskatchewan, Canada

in association with

SOFTSYNC
14 East 34th St.
New York, NY 10016

Prentice/Hall PHI International

ENGLEWOOD CLIFFS, N.J. LONDON NEW DELHI SINGAPORE
SYDNEY TOKYO TORONTO RIO DE JANIERO WELLINGTON

Brain Games for Kids and Adults Using the Commodore 64

Library of Congress Cataloging in Publication Data

Stephenson, John W., 1940-
Brain games for kids and adults using the Commodore 64.

Includes index.
1. Computer games. 2. Commodore 64 (Computer)—
Programming. 3. Basic (Computer program language)
4. Mathematical recreations. I. Title.
GV1469.2.S74 1984 794.8'2 84-329
ISBN 0-13-080938-1

Prentice-Hall International, Inc., London
Prentice-Hall Canada, Inc., Scarborough, Ontario
Prentice-Hall of Australia, Pty., Ltd., Sydney
Prentice-Hall of India Private Limited, New Delhi
Prentice-Hall of Japan, Inc., Tokyo
Prentice-Hall of Southeast Asia Pte. Ltd., Singapore
Whitehall Books, Limited, Petone, New Zealand
Editora Prentice-Hall Do Brasil LTDA., Rio de Janeiro

Printed in the United States of America

84 85 86 87 88 89 90 91 92 93 94 1 2 3 4 5 6 7 8 9 10

This small book of fun and games is dedicated to
my children Margot, Blake, and James,
who have been a big help.

Limits of Liability and Disclaimer of Warranty

The author and publisher of this book have used their best efforts in preparing this book and the programs contained in it. These efforts include the development, research, and testing of the programs to determine their effectiveness. The author and the publisher make no warranty of any kind, expressed or implied, with regard to these programs, the text, or the documentation contained in this book. The author and the publisher shall not be liable in any event for claims of incidental or consequential damages in connection with, or arising out of, the furnishing, performance, or use of the text or the programs. The programs contained in this book and on any diskettes are intended for use of the original purchaser-user. The diskettes may be copied by the original purchaser-user for backup purposes without requiring express permission of the copyright holder.

Note to Authors

Do you have a manuscript or a software program related to personal computers? Do you have an idea for developing such a project? If so, we would like to hear from you. The Brady Co. produces a complete range of books and applications software for the personal computer market. We invite you to write to David Culverwell, Publishing Director, Brady Communications Company, Inc., Bowie, Maryland 20715.

Trademarks of Material Mentioned in This Text

Commodore 64 is a trademark of Commodore Business Machines, Inc.

CONTENTS

Preface ix

Introduction xi

Chapter I

Concentration Games 1
- Concentration 2
- Flash 7
- Knight 11
- Knightmare 15
- Simon 19
- Simon (Long Version) 21

Chapter II

Word Games 27
- Anagrams I 28
- Anagrams II (With Hints) 33
- Hangman 38
- Find a Word 43
- Enigma I 50
- Enigma II 53

Chapter III

Logic and Strategy Games 57
- Invader 58
- Probability 63
- Sea Hunt 66
- Verify 70
- Loose Change 74
- Nim I 79
- Nim II 81
- Nim III 85
- Towers of Hanoi 92

Chapter IV

Arithmetic Games 97
- Barter 98
- Base 101

Your Number Is Up 105
Cryptic 108
Human Arithmetic 112
Jugs 117

Chapter V

Geometry Games 121
Detonation 122
Quest 127
Trek 132
U-Boat 138
Warp 143

Chapter VI

Algebra Games 147
Monkey Puzzle 148
Obscure Ages 153
More Obscure Ages 156
Much More Obscure Ages 158
Product and Sums 162
Rally 166

Chapter VII

Root Finders 169
Fixed-Point 170
Bracket 174

Appendix 179
Disk Load 181
Loader 182
Loader (With Music) 186
Music Maker 192

Index 197

Instructions For Operating The Prerecorded Floppy Disk 203

Instructions For Making A Back-up Copy Of Your Brain Games Diskette 207

PREFACE

WARNING! These games may be hazardous to your peace of mind! You will not find the usual ZAP, ZING, and BOOM of the arcade games here. These are quiet, peaceful, challenging games of concentration and mental skill that can be played between just you and your computer, or a group of your friends can gather around and challenge the computer to battle. The games can be a lot of fun, but are you up to the challenge?

Can you keep your cool when lost in a four-dimensional time warp and your fuel supply is running low?

Will you be able to locate the pirates' treasure before the pirates find it and use you to scrub the barnacles off their ship?

How are you at finding your way around a three-dimensional chess board or cooly deducing the location of a bomb in a three-dimensional building of one million rooms before the bomb detonates?

Can you outwit your computer at games of strategy and logic?

These games are meant to entertain you. While you are having fun, you will learn something about programming and mathematics.

First of all, a lot of good stuff is crammed into very few lines of code. The reason for keeping the codes so tight and concise is that it minimizes the risk of typographical errors occurring while you enter the code on your machine. At the same time, it is not a great chore to enter 30 or 40 lines, and yet where else can you derive so much pleasure from so little an investment of your time?

Second, there are useful programming tips that illustrate short cuts in programming. And last but not least, the solutions to the games demonstrate some everyday, practical mathematics. If you are not familiar with programming, then you will find that the short programs in this book will serve as an introduction to programming techniques.

You don't have to use mathematics to solve the games. My children, aged 10, 14 and 17 have enjoyed playing the games for hours. The youngest uses the technique "by guess and by golly". The middle one delights in "breaking" into the execution of a program, and since the computer can not lie, he obtains the answers from the computer in response to print commands. However, if you are studying mathematics at the high school or university level, or you simply enjoy solving mathematical puzzles, you will delight in the challenge of trying to solve these games in the quickest and most elegant fashion.

In many cases, you are encouraged to go further and to seek answers to questions or problems that are pointed out in the description of a

program. It is my sincere hope that in these games, you will discover for yourself something new and interesting in mathematics.

Prof. John Stephenson
Department of Mathematics
University of Saskatchewan
Saskatoon, Saskatchewan

INTRODUCTION

This collection of short programs has been written in BASIC for the COMMODORE 64. The object was to keep the programs short and snappy. Sometimes as in "SIMON", "ANAGRAMS", "ENIGMA", and "NIM", an additional listing of a longer version of the same program is included when there is something to be gained through the use of the longer program. The point is that you will be able to enter a short program in less than 10 minutes. There is less chance of your making typographical errors with short programs but if you do make an error, it should not be too difficult to find the error and debug your code.

The programs in this book are generally made up of three parts.

Lines 1 to 99: Code for information about the game.
Lines 100 to 999: The game, or main part of the program.
Lines 1000 to 9999: Code for audio-visual effects.

In every case, lines 100 to 999 of the listings contain the essential part of the programs. The additional lines before 100 give information on how to play and contain a summary of the information given in the sections AIM OF THE GAME and HOW TO PLAY, which are placed in this book after each listing. The code in the lines following line 999 are for audio-visual effects. These additions make the programs more exciting to run, but are a pain to enter! To get started, you may want to enter the main part of the programs first. You will need to read the section on THE CODE to make sure that any references to the audio-visual coding is suppressed. At a later stage, you could add the extra lines. You may also leave out all the REM statements that are included in the listings to explain what the code is doing.

In order to keep the main part of the programs short, the use of screen displays and graphics had to be kept to the bare essentials of prompts and messages. The listings given in this book are identical to the screen displays of your entered programs. If your listing does not match the one given, there is something wrong in the entry of your code. It may or may not be crucial, but it will serve as a warning for you to check for possible errors in your code. If you do not want to type in these programs yourself, you can purchase a prerecorded floppy disk, and enjoy playing the longer versions of the games immediately.

These programs will be quite useful to beginners in programming. They will also give you a chance to become acquainted with analytic geometry, linear equations, quadratic equations, systems of linear and non-linear equations, conics, Lagrange Interpolation, probability and

statistics, binomial probability, binomial expansion, Pascal's triangle, the arithmetic of complex numbers, representation of numbers in different bases, Cartesean and polar coordinates, distances measured by different norms, the 1-norm and the Euclidean norm. There are also games of logic, concentration, and of course, arithmetic and algebra.

Unlike arcade games, these games do not use moving graphics. For the most part, the display is used for prompts, answers, and messages of praise or doom! The essential features of the games are that they develop concentration, imagination, logic and the ability to do mental arithmetic.

The games are collected into chapters which categorize them as to their general content. You might find it helpful to refer to the following where the games are categorized as to their specific nature or mathematical content.

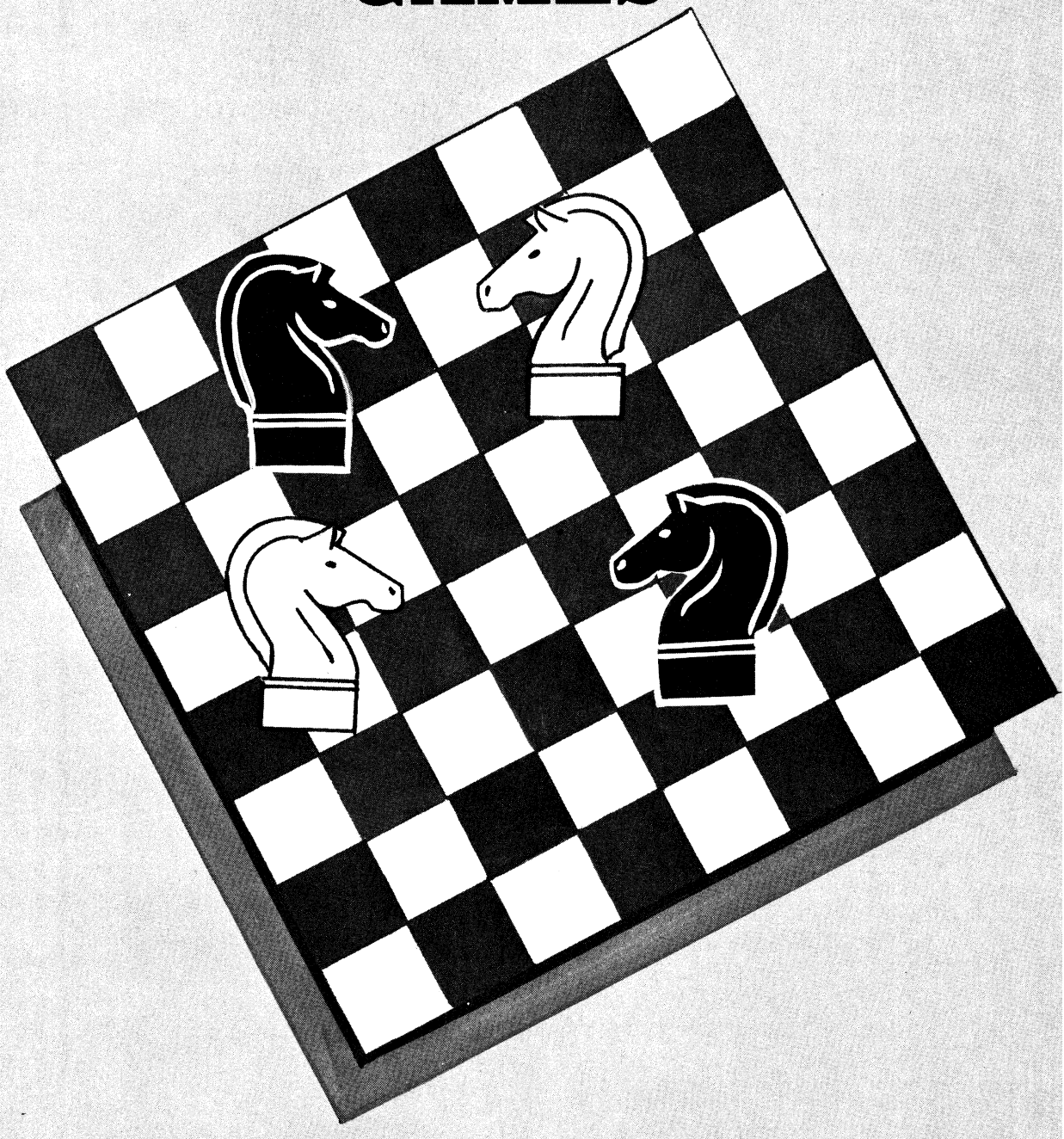
Content	Game
Algebra:	MONKEY PUZZLE, OBSCURE AGES, MORE OBSCURE AGES, MUCH MORE OBSCURE AGES, PRODUCT AND SUMS, RALLY
Analytic geometry:	DETONATION, QUEST, TREK, U-BOAT, WARP
Arithmetic:	BARTER, BASE, CRYPTIC, HUMAN ARITHMETIC, JUGS, PRODUCT AND SUMS, RALLY, YOUR NUMBER IS UP
Cartesian coordinates:	DETONATION, QUEST, TREK, U-BOAT, WARP
Complex numbers:	HUMAN ARITHMETIC
Concentration and memory:	CONCENTRATION, FLASH, KNIGHT, KNIGHTMARE, LOOSE CHANGE, SIMON, VERIFY
Cylindrical and polar coordinates:	U-BOAT
Distance measured by 1-norm:	QUEST, TREK, WARP
Euclidean distance:	DETONATION, U-BOAT
Gauss elimination and linear equations:	OBSCURE AGES, MORE OBSCURE AGES, MUCH MORE OBSCURE AGES, RALLY, TREK, WARP
Lagrange interpolation:	ANAGRAMS, HUMAN ARITHMETIC

Content	Game
Logic and strategy:	INVADER, KNIGHT, KNIGHTMARE, LOOSE CHANGE, MONKEY PUZZLE, NIM, INVADER, SEA HUNT, TOWERS OF HANOI
Number bases:	BASE, NIM, TOWERS OF HANOI, YOUR NUMBER IS UP
Probability and statistics:	INVADER, PROBABILITY, SEA HUNT
Quadratic equations and non-linear equations:	DETONATION, PRODUCT AND SUMS, U-BOAT
Word games:	ANAGRAMS, ENIGMA, HANGMAN, FIND A WORD
Zeros or roots:	FIXED-POINT, BRACKET

The experience gained from teaching mathematics for twenty years has gone into these games. It is my belief that it is possible to learn mathematics and the ability to think clearly and logically through the use of games. This can be a lot of fun for children of all ages, including those of us who are merely young in heart! The presentations and explanations of the games are given in a tutorial fashion, as if I were by your side explaining things to you as you go along. You should find this approach less formal and stuffy than your usual mathematics textbook. My children and their friends have enjoyed playing these games. It is my hope that you will enjoy playing them as much as we did.

CHAPTER I

CONCENTRATION GAMES



"Concentration"

```
10 PRINT "□":PRINT:PRINT "INSTRUCTIONS?
PRESS Y OR N":PRINT
20 GET A$:IF A$="" THEN 20
30 IF A$<>"Y"THEN 100
40 PRINT "NUMBERS ARE HIDDEN IN A SQUARE
 FULL OF *"
50 PRINT "YOU MUST FIND PAIRS OF MATCHIN
G NUMBERS."
60 PRINT "YOUR TRIES ARE COUNTED AND YOU
R GOAL IS "
70 PRINT "TO MATCH UP ALL OF THE PAIRS O
F NUMBERS "
80 PRINT "IN THE FEWEST ATTEMPTS"
90 PRINT:INPUT "PRESS RETURN TO START";
100 LET H=0
105 LET T=0
110 PRINT"□":PRINT:PRINT"THERE ARE FOUR
LEVELS OF DIFFICULTY":PRINT
115 PRINT "WHICH DO YOU WANT?": PRINT:PR
INT "PRESS 1,2,3 OR 4"
120 GET S$:IF S$="" THEN 120
130 LET S=2*VAL(S$)
140 IF S<2 OR S>8 THEN 110
145 PRINT:PRINT "SETTING UP THE GAME"
150 FOR I=1 TO S
160 FOR J=1 TO S
170 LET A(I,J)=INT(S*RND(0))+1
180 IF J=1 THEN 200
190 FOR K=1 TO (J-1):IF A(I,K)=A(I,J) TH
EN 170
195 NEXT K
200 NEXT J,I
210 FOR I=1 TO S:LET A$(I)=""
220 FOR J=1 TO S:LET A$(I)=A$(I)+" *"
230 NEXT J,I
235 GOSUB 600
238 IF S=2*INT(S/2) AND H=S*S/2 THEN 900
240 PRINT:INPUT"ENTER ROW AND COLUMN - F
OR EXAMPLE 1,2";I,J
245 IF I=-1 THEN 910
```

```
246 IF I<1 OR J<1 OR I>S OR J>S THEN 238
250 IF MID$(A$(I),2*J-1,2)<>" *" THEN 23
5
255 LET X=A(I,J)
260 GOSUB 400
265 PRINT:INPUT"ENTER ROW AND COLUMN -TR
Y FOR A MATCH";K,L
266 IF K=-1 THEN 910
270 IF K<1 OR L<1 OR K>S OR L>S OR (K=1
AND L=J) THEN 265
275 IF MID$(A$(K),2*L-1,2)<>" *" THEN 26
5
277 LET I=K:J=L
280 LET Y=A(I,J)
285 GOSUB 400
290 IF X=Y THEN PRINT "A MATCHING PAIR!"
:LET H=H+1:GOTO 238
295 PRINT "NOT MATCHING! GIVE IT ANOTHER
 TRY MATE"
300 FOR I=1 TO S:LET A$(I)=B$(I):NEXT I
305 FOR Z=1 TO 2000:NEXT Z
310 GOSUB 600
320 GOTO 238
390 REM:THIS SUBROUTINE REVEALS THE NUMB
ER BEHIND THE * AND COUNTS THE TURNS
400 LET T=T+.5
405 IF INT(T+.5)=(T+.5) THEN FOR K=1 TO
S:LET B$(K)=A$(K):NEXT K
410 LET A$(I)=LEFT$(A$(I),2*J-2)+STR$(A(
I,J))+RIGHT$(A$(I),2*S-2*J)
415 GOSUB 600
420 RETURN
590 REM:THIS SUBROUTINE DRAWS THE PICTUR
E OF THE GAME
600 PRINT "□ TO QUIT ENTER -1,-1"PRINT
TAB(3)
610 FOR M=1 TO S:PRINT STR$(M);:NEXT M
620 PRINT
630 FOR M=1 TO S:PRINT:PRINT M;A$(M):NEX
T M
640 RETURN
900 PRINT:PRINT "YOU SOLVED THE GAME IN"
;T;"MOVES"
910 PRINT:PRINT "AGAIN? PRESS Y OR N"
```

```
92Ø GET Z$:IF Z$="" THEN 92Ø
93Ø IF Z$="Y" THEN 1ØØ
94Ø STOP
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME.

"Concentration"

AIM OF THE GAME

This is a game that tests your concentration and memory. An array of S rows and S columns of asterisks (*) is set up. You decide on the size of the array by entering a number 1,2,3 or 4 in response to a prompt. The value of S is twice the number that you choose to enter. The numbers 1 to S are randomly distributed in each of the rows, but are hidden behind the asterisks. You are prompted to enter coordinates, (row I and column J) of two asterisks. The numbers behind your chosen asterisks are displayed. If you have selected a matching pair, they remain showing and you score a hit. If you fail to match a pair, your selection is covered up by asterisks and you try again. The object of the game is to match up all the pairs possible in the fewest number of tries.

Luckily the computer does not get to have a turn, this is strictly a game for you. The computer would be difficult to beat because like the proverbial elephant, it never forgets!

HOW TO PLAY

When you run this program, you are first told

```
THERE ARE FOUR LEVELS OF DIFFICULTY
WHICH DO YOU WANT?
PRESS 1,2,3 OR 4
```

Suppose you press the key 2. Immediately the message appears

```
SETTING UP THE GAME
```

and after a short time there appears the array and prompt

```
TO QUIT ENTER -1,-1
  1 2 3 4
1 * * * *
2 * * * *
3 * * * *
4 * * * *
ENTER ROW AND COLUMN—FOR EXAMPLE 1,2?
```

Suppose you enter 1,2. The number in the first row and second column is revealed

```
   1 2 3 4
1  * 3 * *
2  * * * *
3  * * * *
4  * * * *
ENTER ROW AND COLUMN - TRY FOR A MATCH?
```

Suppose you enter 2,1. The number in the second row first column is revealed. If it is a pair, you get a message

```
A MATCHING PAIR!
ENTER ROW AND COLUMN - FOR EXAMPLE 1,2?
```

and you continue on while the matching pair remains showing. On the other hand, if a pair does not match, you will be prompted to try again. After a brief moment, while you memorize the numbers you turned up, they are covered by asterisks and you are once again prompted to enter your choice of coordinates.

When you have matched all the numbers that can be paired, (in this case there are only eight pairs), you will be given the message

```
YOU SOLVED THE GAME IN T MOVES
```

where T will be the number of your tries. You are then invited to play again or to quit. To quit at any time, enter the pair -1,-1.

THE CODE

The program checks that you enter two different number pairs. It also checks that you enter the coordinates of a position that are not already matched. These checks are in lines 246, 25Ø, 27Ø and 275.

In setting up the game, the numbers 1 to S are jumbled up on each row. The game would be more difficult if the numbers were randomly located anywhere on the array. This would mean that a particular number might occur more than once in any given row. This is a fairly difficult programming exercise for a beginner, but I urge you to try it. Even if you do not succeed, you will have learned something about programming. My suggestion is to set up a routine to exchange pairs of numbers in the array A, and then thoroughly mix them up.

One other challenge I would like to leave with you is to design a subroutine in which the computer takes a turn when you miss. It would be easy to design a few lines that cause the computer to look at random locations. But what I had in mind was that the computer could use any information revealed in previous turns. You will then be able to play extremely challenging games with your computer, which would never miss using any relevant information.

"Flash"

```
10 PRINT "□":PRINT:PRINT "INSTRUCTIONS?
PRESS Y OR N":PRINT
20 GET A$:IF A$="" THEN 20
30 IF A$<>"Y" THEN 100
40 PRINT "THIS GAME IMPROVES THE SPEED O
F YOUR"
50 PRINT "READING. STARTING WITH A SINGL
E LETTER,"
50 PRINT "A STRING IS FLASHED ON THE SCR
EEN FOR A"
70 PRINT "MOMENT. YOU MUST ENTER THE SAM
E STRING."
80 PRINT "IF YOU ARE RIGHT, THEN A NEW S
TRING IS"
85 PRINT "FLASHED WITH ONE EXTRA CHARACT
ER."
90 PRINT "IF YOU PREFER, YOU MAY USE NUM
BERS.":PRINT
95 INPUT "PRESS RETURN TO GO ON";:REM TH
IS WAITS WHILE THE MESSAGE IS READ
100 PRINT "□"
110 PRINT "NUMBERS OR LETTERS?"
120 PRINT:PRINT "PRESS N OR L KEY"
130 GET C$: IF C$="" THEN 130
140 FOR I=1 TO 24:REM THIS LIMITS THE LE
NGTH OF THE STRING TO 24
150 PRINT "WORKING...":REM FOR LONGER ST
RINGS THE SET UP CAN TAKE A MOMENT
155 LET A$="":REM STARTS OFF WITH AN EMP
TY STRING EACH TIME
160 FOR J=1 TO I
170 IF C$="N" THEN 210
180 LET S=INT(26*RND(0))+ASC("A")
190 LET A$=A$+CHR$(S):REM CREATES A STRI
NG OF RANDOM LETTERS OF LENGTH I
200 GOTO 220
210 LET A$=A$+RIGHT$(STR$(INT(10*RND(0))
),1):REM STRING OF NUMBERS, NO BLANKS
220 NEXT J
230 PRINT "□"
```

```
235 PRINT "PRESS A KEY WHEN READY"
240 GET Z$:IF Z$="" THEN 240:REM GIVES Y
OU TIME TO GET READY
250 PRINT "□"
260 PRINT LEFT$(A$,I)
270 FOR T=1 TO 200+200*I:NEXT T
280 PRINT "□"
290 PRINT "TYPE WHAT YOU JUST SAW AND PR
ESS RETURN"
300 INPUT B$
310 IF B$<>A$ THEN 360
320 PRINT "CORRECT"
330 FOR T=1 TO 1000:NEXT T
340 NEXT I
350 STOP
360 PRINT "WRONG IT IS ";A$
370 PRINT:PRINT "AGAIN? PRESS Y OR N"
380 GET Z$:IF Z$="" THEN 380
390 IF Z$="Y" THEN 100
400 STOP
```

Note: The symbol □ represents SHIFT CLEAR/HOME.

"Flash"

AIM OF THE GAME

This program tests your ability at reading and remembering a sequence of letters or numbers that are flashed on the screen. You are asked for your choice of letters or numbers. Then starting with a single character and increasing by one character each time, a randomly generated sequence is flashed for a short time on the screen. Each time you enter a correct sequence another sequence with an additional character is flashed for you.

It is very difficult to get more than 1Ø characters correct. Psychologists have found that the average person can usually recall about 7 characters. This program stops at 24 characters, but it is not difficult to increase this limit by changing the limit in line 14Ø.

This program is designed to increase your concentration and your reading speed. You will find it very challenging.

HOW TO PLAY

When you run this program the screen clears and the following message appears.

```
NUMBERS OR LETTERS?
PRESS N OR L KEY
```

Suppose you press the L key. Immediately the screen clears, the word

```
WORKING . . .
```

appears momentarily (when you are up to sequences that are 24 characters in length it takes a few seconds to construct the sequence). Then you are prompted with the message

```
PRESS A KEY WHEN READY
```

The reason for this is that you can blink your eyes, clear your throat, scratch the itchy spot on your nose and generally prepare yourself for the flash. When you are ready you press any key. A single letter is flashed on the screen and disappears. Let us say it was

```
W
```

Then you are prompted with the message

```
TYPE WHAT YOU JUST SAW AND PRESS RETURN
```

You type the letter W and then enter it. The message

```
CORRECT
```

is displayed for a short time, then back to WORKING ... and the whole thing is repeated except that the sequence will contain two characters this time. Each time you enter a correct sequence the next sequence will contain one more character than the previous one.

If you should happen to enter an incorrect sequence, as we all do eventually, you will be given the message

WRONG IT IS ABCXYZ

(whatever the sequence was)

AGAIN? PRESS Y OR N

If you feel up to playing another game, press the Y key. Otherwise, press any other key to quit for a well deserved rest.

You may feel that a little more time is needed in order to view the sequence and memorize it. All you need to do is alter the statement in line 27Ø to 2ØØ + 3ØØ*I or something like that. Do not make the pause too long or you will defeat the purpose of the game.

Happy flashing!

"Knight"

```
10 PRINT "□":PRINT:PRINT:PRINT " INSTRU
CTIONS? PRESS Y OR N"
15 GET A$:IF A$="" THEN 15
20 IF A$<>"Y" THEN 110
25 PRINT "□":PRINT:PRINT
30 A$="■ ■ ■ ■  ":B$=" ■ ■ ■ ■":E$="-----
---":R$="--------":T$="|":Y$="|"
35 PRINT TAB(9);"ABCDEFGH":PRINT TAB(9);
R$
40 FOR I=1 TO 8:PRINT TAB(5);I;Y$;
45 LET C$=A$:IF I-2*INT((I/2))=0 THEN C$
=B$
50 PRINT C$;T$:NEXT I:PRINT TAB(9);E$
55 PRINT:PRINT "THIS GAME USES A CHESS B
OARD"
60 PRINT "A LETTER AND A NUMBER ARE USED
 TO MARK  EVERY POSITION ON THE BOARD"
65 PRINT "THE GAME IS TO MOVE A KNIGHT F
ROM ONE"
70 PRINT "POSITION TO ANOTHER IN AS FEW
MOVES AS"
75 PRINT "POSSIBLE (MOVES ARE IN 1-2 COM
BINATIONS)"
90 INPUT "PRESS RETURN TO START";
110 LET C=0
120 PRINT "□"
125 PRINT:PRINT"ENTER MOVE IN THE FORM L
ETTER NUMBER":PRINT "ENTER -1 TO STOP"
130 LET A=INT(8*RND(0))+ASC("A")
140 LET B=INT(8*RND(0))+ASC("A")
150 LET M=INT(8*RND(0))+ASC("1")
160 LET N=INT(8*RND(0))+ASC("1")
165 IF A=B AND M=N THEN 130
170 PRINT:PRINT "MOVE FROM ";CHR$(A);CHR
$(M);" TO ";CHR$(B);CHR$(N)
180 LET C=C+1
190 PRINT C;" ";CHR$(A);CHR$(M);" TO ";
200 INPUT A$
201 IF VAL(A$)=-1 THEN 275
205 LET D$=LEFT$(A$,1)
```

```
206 LET E$=RIGHT$(A$,1)
210 IF A=ASC(D$) OR M=ASC(E$) OR ABS(A-A
SC(D$))+ABS(M-ASC(E$))<>3 THEN 300
230 LET A=ASC(D$)
240 LET M=ASC(E$)
250 IF A=B AND M=N THEN 270
260 GOTO 180
270 PRINT "NUMBER OF MOVES =";C
275 PRINT "AGAIN? PRESS Y OR N"
276 GET Z$:IF Z$="" THEN 276
280 IF Z$="Y" THEN 110
290 STOP
300 PRINT " INVALID MOVE"
310 GOTO 180
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME. In line 30 use the graphics symbols on the +, E, R, T and Y keys.

"Knight"

AIM OF THE GAME

This is a game of concentration and teaches you to visualize a situation in your mind. This ability is essential in the study of mathematics. To start with, however, you might prefer to use a chess board with the letters A to H marked across the board from left to right, and the numbers 1 to 8 marked on the edge of the board starting with 1 at the bottom and ending with 8 at the top.

The game is to move a chess piece, called a knight, from one given location on a chess board to another given location. The knight moves in a peculiar way crossing three squares each move. It can move in any combination of 1 up or down and 2 across, or 2 up or down and 1 across. Your moves are checked and counted. The idea is to try to get to your final destination in the least number of moves. However, it is a bit like running a marathon because there is no penalty for being slow. The idea is to stick with it until you finish.

HOW TO PLAY

When you run this program, the screen clears and you are given a message like

```
ENTER MOVE IN THE FORM LETTER NUMBER
ENTER -1 TO STOP
MOVE FROM C2 TO C4
1     C2 TO ?
```

Your entry must be in a string, first a letter and then a number. Your move will be checked. If it is not a valid move, you will be asked to repeat the move. Suppose you were to enter D4. On the screen you would see

```
2     D4 TO ?
```

Suppose you now entered C4 (wishful thinking!). You would get the message

```
INVALID MOVE
3     D4 TO ?
```

Now let us go back to C2. We will not fool around any more. The message is

```
4     C2 TO ?
```

We enter E3. The message is

```
5     E3 TO ?
```

Now we can enter C4. The message we get is

```
NUMBER OF MOVES = 5
AGAIN? PRESS Y OR N
```

If you wish to play again press the Y key. To quit, you just press the key N. To quit during the game, enter the move -1 in response to the prompt to enter a move.

THE CODE

Lines 1Ø to 9Ø may be omitted. They only give some instructions on how to play the game. The INPUT in line 9Ø halts the execution so that the screen can be read.

Could you design a subroutine that determines the solution in the minimum number of moves? You could then have the computer print out that solution.

"Knightmare"

```
1Ø PRINT "◘":PRINT:PRINT "THIS IS A 3-D
VERSION OF KNIGHT"
2Ø PRINT "YOU SHOULD PLAY KNIGHT BEFORE
TACKLING"
3Ø PRINT "THIS HORROR!"
4Ø PRINT:PRINT "MOVES MUST BE ENTERED AS
 LETTER NUMBER"
5Ø PRINT "LETTER LIKE":PRINT TAB(4);"E5G
":PRINT
6Ø PRINT "EVERY COORDINATE MUST CHANGE"
7Ø PRINT "THE TOTAL CHANGE MUST BE 6":PR
INT
9Ø INPUT "PRESS RETURN TO START";
11Ø LET C=Ø
12Ø PRINT "◘"
125 PRINT:PRINT "ENTER MOVE IN THE FORM
'E5G' LETTER":PRINT "NUMBER LETTER"
126 PRINT "ENTER -1 TO STOP":PRINT
13Ø LET A=INT(8*RND(Ø))+ASC("A")
14Ø LET B=INT(8*RND(Ø))+ASC("A")
145 LET Q=INT(8*RND(Ø))+ASC("A")
15Ø LET M=INT(8*RND(Ø))+ASC("1")
16Ø LET N=INT(8*RND(Ø))+ASC("1")
165 LET P=INT(8*RND(Ø))+ASC("A")
166 LET Z=ABS(A-B)+ABS(M-N)+ABS(Q-P)
167 IF Z-2*INT(Z/2)<>Ø THEN 13Ø
17Ø PRINT "MOVE FROM ";CHR$(A);CHR$(M);C
HR$(Q);" TO ";CHR$(B);CHR$(N);CHR$(P)
18Ø LET C=C+1
19Ø PRINT C;" ";CHR$(A);CHR$(M);CHR$(Q);
" TO ";
2ØØ INPUT A$
2Ø1 IF VAL(A$)=-1 THEN 275
2Ø5 LET D$=LEFT$(A$,1)
2Ø6 LET E$=MID$(A$,2,1)
2Ø7 LET F$=RIGHT$(A$,1)
21Ø IF A=ASC(D$) OR M=ASC(E$) OR Q=ASC(F
$) THEN 3ØØ
215 IF ABS(A-ASC(D$))+ABS(M-ASC(E$))+ABS
(Q-ASC(F$))<>6 THEN 3ØØ
```

```
230 LET A=ASC(D$)
240 LET M=ASC(E$)
245 LET Q=ASC(F$)
250 IF A=B AND M=N AND Q=P THEN 270
260 GOTO 180
270 PRINT "NUMBER OF MOVES =";C
275 PRINT "AGAIN? PRESS Y OR N"
276 GET Z$:IF Z$="" THEN 276
280 IF Z$="Y" THEN 110
290 STOP
300 PRINT " INVALID MOVE"
310 GOTO 180
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME.

"Knightmare"

AIM OF THE GAME

This is a three-dimensional version of the chess game called "KNIGHT". You should try the standard two-dimensional game "KNIGHT" before tackling this game. As the name implies, you might have trouble going to sleep after playing this horror.

HOW TO PLAY

You are given a starting point and a finishing point on a three-dimensional chess board. The locations are given by a letter A to H, followed by a number 1 to 8, and then another letter A to H. Your knight can move a total of six squares in any combination of (2,2,2), (4,1,1) or (1,2,3). By any combination, I mean for instance, in the (1,2,3) move, any one of the coordinates can be changed by 1 unit, another by 2 units and the remaining coordinate by three units. A typical run would go like this. You are given the prompt

```
ENTER MOVE IN THE FORM 'E5G' LETTER
NUMBER LETTER
ENTER -1 TO STOP
MOVE FROM G5A TO E4B
1     G5A TO ?
```

Every coordinate must alter. So suppose you enter the move F7D. You would get the message

```
1     G5A TO ? F7D
2     F7D TO ?
```

Now you can enter E4B and you are home in 2 moves!

You must enter your moves in a string containing all the three new coordinates in the format, letter, number and then letter.

Your moves are checked each time for validity. Every coordinate must change and the total change must be precisely 6. When you get home you are prompted to play another game or to quit.

THE MATHEMATICS AND CODE

You write down the difference in the starting and ending coordinates. In the case given above it was -2, -1, 1. This means we have to back up two positions on the first coordinate, subtract one from the second and add one to the third coordinate. This is a total change of 4 so we cannot do it in one move. If we make the changes 1, -2, -3 we are left with -1,

–3, –2 by adding. This means we can get home in just two moves, and the moves made above in HOW TO PLAY accomplish this.

You will notice that I only used the (1, 2, 3) move, which I think should be the only one allowed. However, you might find the game difficult enough without adding this restriction! If you would like a more difficult challenge, you could limit yourself to the (1, 2, 3) combinations, or even add lines to the program to prohibit any other move. For instance, you might add the lines

```
216 LET X=ABS(A-ASC(D$))
217 LET Y=ABS(M-ASC(E$))
218 LET Z=ABS(O-ASC(F$))
219 IF X=Y OR X=Z OR Y=Z THEN 3ØØ
```

You will notice that in lines 166 and 167 I had to add (as an afterthought) the limitation that the difference in start and finish positions could not be odd. No such restriction was needed in the two-dimensional game. Can you work out why this restriction is necessary?

"Simon"

```
100 LET B$=""
110 PRINT "□":PRINT:PRINT"   THIS GAME O
F SIMON TESTS YOUR MEMORY"
120 PRINT "WOULD YOU LIKE NUMBERS OR LET
TERS? PRESS THE KEY N OR L"
130 GET A$:IF A$="" THEN 130
135 IF A$<>"N" AND A$<>"L" THEN 110
140 LET I=1
150 PRINT "□"
160 IF A$="N" THEN LET B$=LEFT$(B$,I-1)+
RIGHT$(STR$(10*RND(0)),1)
170 IF A$="L" THEN LET B$=LEFT$(B$,I-1)+
CHR$(26*RND(0)+ASC("A"))
180 FOR J=1 TO I
190 LET S$=MID$(B$,J,1):PRINT S$;
195 IF A$="N" THEN N=VAL(S$)
196 IF A$="L" THEN N=ASC(S$)-ASC("A")
200 FOR T=1 TO 500:NEXT T
210 NEXT J
220 PRINT "□":PRINT:FOR G=0 TO 10*RND(0)
+4:GET S$:NEXT:REM PREVENTS CHEATING
230 PRINT "ENTER THE SAME ";
235 IF A$="N" THEN PRINT "NUMBERS"
240 IF A$="L" THEN PRINT "LETTERS"
250 GOTO 1000
260 PRINT:PRINT "WRONG. IT IS"
265 FOR G=0 TO 10:GET S$:NEXT:REM: ERASE
S ANY ACCIDENTAL KEY PRESSES
270 PRINT LEFT$(B$,I)
275 IF I>1 THEN PRINT "YOUR SCORE WAS ";
I-1
280 PRINT "ANOTHER TURN? PRESS Y OR N"
290 GET A$:IF A$="" THEN 290
300 IF A$="Y" THEN 100
305 STOP
310 PRINT:PRINT "RIGHT ON"
320 FOR T=1 TO 2000:NEXT T
330 LET I=I+1
340 GOTO 150
1000 REM:CHECKS EACH ENTRY
```

```
1Ø1Ø LET J=Ø
1Ø2Ø GET S$:IF S$="" THEN 1Ø2Ø
1Ø3Ø PRINT S$;:J=J+1
1Ø4Ø IF A$="N" THEN N=VAL(S$)
1Ø5Ø IF A$="L" THEN N=ASC(S$)-ASC("A")
1Ø7Ø IF J<I AND S$=MID$(B$,J,1) THEN 1Ø2
Ø
1Ø8Ø IF S$=MID$(B$,J,1) THEN 31Ø
1Ø9Ø GOTO 26Ø
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME.

"Simon" (Long Version)

```
1Ø REM: THE HI/LO FREQUENCIES FOR THE NO
TES ARE READ FROM DATA
2Ø REM:SET UP ORGAN
25 FOR I=54272 TO 54296:POKE I,Ø;NEXT I:
REM:THIS SETS ALL VOICE QUANTITIES TO Ø
3Ø POKE 54296,15:REM SETS VOLUME TO LOUD
EST
35 FOR I=54277 TO 54291 STEP 7:POKE I,19
Ø:POKE I+1,255:NEXT:REM:SET SYNTHESIZER
4Ø PRINT "□":PRINT:PRINT "   NEED INSTRU
CTIONS? PRESS Y OR N"
45 GET A$:IF A$="" THEN 45
5Ø IF A$<>"Y" THEN 1ØØ
55 PRINT:PRINT "WE GIVE SOUNDS TO THE NU
MBERS Ø TO 9"
6Ø PRINT:PRINT "THEY ARE:":PRINT
7Ø FOR N=Ø TO 9:PRINT N:"GOSUB 8ØØØ
75 NEXT N
8Ø PRINT:PRINT:PRINT "LETTERS SOUND LIKE
 THE FOLLOWING":PRINT
85 FOR N=Ø TO 25:PRINT CHR$(ASC("A")+N);
:GOSUB 8ØØØ
9Ø NEXT N
1ØØ PRINT "□":PRINT:PRINT"   THIS GAME O
F SIMON TESTS YOUR MEMORY"
12Ø PRINT "WOULD YOU LIKE NUMBERS OR LET
TERS? PRESS THE KEY N OR L"
13Ø GET A$:IF A$="" THEN 13Ø
135 IF A$<>"N" AND A$<>"L" THEN 11Ø
14Ø LET I=1
15Ø PRINT "□"
16Ø IF A$="N" THEN LET B$=LEFT$(B$,I-1)+
RIGHT$(STR$(1Ø*RND(Ø)),1)
17Ø IF A$="L" THEN LET B$=LEFT$(B$,I-1)+
CHR$(26*RND(Ø)+ASC("A"))
18Ø FOR J=1 TO I
19Ø LET S$=MID$(B$,J,1);PRINT S$;
195 IF A$="N" THEN N=VAL(S$)
```

```
196 IF A$="L" THEN N=ASC(S$)-ASC("A")
200 GOSUB 8000
210 NEXT J
220 PRINT "□":PRINT:FOR G=0 TO 10*RND(0)
+4:GET S$:NEXT:REM PREVENTS CHEATING
230 PRINT "ENTER THE SAME ";
235 IF A$="N" THEN PRINT "NUMBERS"
240 IF A$="L" THEN PRINT "LETTERS"
250 GOTO 1000
260 PRINT:PRINT "WRONG. IT IS"
265 FOR G=0 TO 10:GET S$:NEXT:REM: ERASE
S ANY ACCIDENTAL KEY PRESSES
270 PRINT LEFT$(B$,I)
275 IF I>1 THEN PRINT "YOUR SCORE WAS ";
I-1
280 PRINT "ANOTHER TURN? PRESS Y OR N"
290 GET A$;IF A$="" THEN 290
300 IF A$="Y" THEN 100
305 STOP
310 PRINT:PRINT "RIGHT ON"
320 FOR T=1 TO 2000:NEXT T
330 LET I=I+1
340 GOTO 150
1000 REM:CHECKS EACH ENTRY
1010 LET J=0
1020 GET S$:IF S$="" THEN 1020
1030 PRINT S$;:J=J+1
1040 IF A$="N" THEN N=VAL(S$)
1050 IF A$="L" THEN N=ASC(S$)-ASC("A")
1060 GOSUB 8000
1070 IF J<I AND S$=MID$(B$,J,1) THEN 102
0
1080 IF S$=MID$(B$,J,L) THEN 310
1090 GOTO 260
8000 REM: THIS SUBROUTINE PLAYS NOTES
8005 REM: THE HI/LO FREQUENCIES FOR THE
NOTES ARE READ FROM DATA
8006 RESTORE:FOR HL=0 TO N:READ HI,LO:NE
XT HL
8010 FOR V=54272 TO 54286 STEP 7:POKE V+
1,HI:POKE V,LO:NEXT V
8020 FOR V=54276 TO 54290 STEP 7:POKE V,
33:NEXT V:REM THIS SETS THE WAVEFORM
8030 FOR D=1 TO 500:NEXT D:REM THIS IS T
```

```
HE DURATION OF THE NOTE
8Ø4Ø FOR V=54276 TO 5429Ø STEP 7:POKE V,
Ø:NEXT V:REM THIS TURNS OFF THE NOTE
8Ø5Ø RETURN
9ØØØ REM: DATA LISTED HERE
9Ø1Ø DATA 7,53,8,23,8,147,9,159,1Ø,2Ø5,1
1,114,12,216,14,1Ø7,16,47,17,37,19,63
9Ø2Ø DATA 21,154,22,227,25,177,28,214,32
,94,34,75,38,126,43,52,45,198,51,97
9Ø4Ø DATA 57,172,64,188,68,149,76,252,86
,1Ø5
```

"Simon"

AIM OF THE GAME

We have all played "Simon Says" or "Follow the Leader" at some time or other in our lives. This game provides two versions of Follow the Leader. You have your choice between numbers or letters. A sequence is displayed one character after another. At first, the sequence consists of a single character. After a short period of time, the sequence vanishes from sight, but hopefully, not from your memory, and you are asked to enter the same sequence. If you are correct, the same sequence is repeated with an extra character tacked on to the end.

HOW TO PLAY

When you run this program you are given the prompt

```
THE GAME OF SIMON TESTS YOUR MEMORY
WOULD YOU LIKE NUMBERS OR LETTERS?
PRESS THE KEY N OR L
```

You press the N key if you want numbers or the L key if you want a sequence of letters. You should find numbers easier since the choice for each member of the sequence is only ten possibilities. Suppose you press the L key. Immediately a single letter appears in the top left hand corner of the screen, say

```
B
```

This vanishes in about ½ a second. You are then prompted with the message

```
ENTER THE SAME LETTERS
```

Suppose you enter the letter B. The message

```
RIGHT ON
```

is displayed for a few seconds and then the sequence is repeated with an additional letter. In this case it turned out to be BA followed by BAG, followed by BAGM and so on.

This is a very demanding game and requires tremendous concentration. In some respects it is much easier than "FLASH", since the same sequence is repeated over and over. But this is the very aspect that makes this game so mentally exhausting.

When you finally make a mistake, unintentional or not, you get the message

```
WRONG. IT IS
BAGMP
```

(or whatever the actual sequence was)

```
YOUR SCORE WAS 4
ANOTHER TURN? PRESS Y OR N
```

If you want to quit press N or any key other than Y. If you wish to quit during the game, press the key RUN/STOP in response to a prompt.

THE CODE

The string variable B$ which stores the sequence is initialized to an empty string in line 1ØØ. In lines 16Ø and 17Ø, additional characters are added to the string B$ on the right hand end. Unfortunately, strings of numbers like STR$(2) would be stored as " 2" and not "2". The blank is stripped away in line 16Ø, by using the RIGHT$(..., 1) function.

Since the GET command only reads in one key press at a time and your keyboard has a memory buffer which stores key presses, lines 22Ø and 265 are included to prevent cheating and an accidental response to the prompt in line 28Ø about quitting.

In the long version, the lines before 1ØØ set up an organ-like sound and demonstrate the sounds given to each letter and number. The subroutine at line 8ØØØ plays the notes by setting the high and low frequencies and turning on the voices with the sawtooth waveform (33). After the duration in line 8Ø3Ø, the voices are turned off by changing the waveform to Ø.

The data in the lines after line 9ØØØ contain the high and low frequencies of the scale of notes used in the program.

If desired, you could store the frequencies of the twenty six notes in a matrix A(25,1). This removes the slight delay in playing the notes. You would need to make the following changes.

```
5 DIM A(25,1)
1Ø FOR I=Ø TO 25:FOR J=Ø TO 1:READ A(
I,J):NEXT J,I
```

You would delete line 8ØØ6 and change 8Ø1Ø to

```
8Ø1Ø for V=54272 TO 54286 STEP 7:POKE
V+1,A(N,Ø):POKE V,A(N,1):NEXT V
```

The reason I did not use this was because if you use the DISK LOAD and LOADER programs in the Appendix, an error is caused by the dimension statement in line 5 and automatic execution is aborted.

CHAPTER II

WORD GAMES

"Anagrams I"

```
10 PRINT"□":PRINT:PRINT "INSTRUCTIONS? P
RESS Y OR N":PRINT
20 GET A$:IF A$="" THEN 20
30 IF A$<>"Y" THEN 85
40 PRINT "THIS GAME GIVES YOU A JUMBLED
STRING OF "
50 PRINT "LETTERS AND IT IS YOUR JOB TO
FIND THE  "
60 PRINT "HIDDEN WORD.THE STRING ROMST,F
OR EXAMPLE"
70 PRINT "IS A SCRAMBLED VERSION OF STOR
M."
80 PRINT:INPUT "PRESS RETURN TO START";
85 FOR L=54272 TO 54296:POKE L,0:NEXT:PO
KE 54296,15:REM SETS UP VOICES
90 FOR I=1 TO 3:V=54269+7*I:POKE V+1,190
:POKE V+2,250:NEXT
95 PRINT "□"
100 LET G=INT(6*RND(0)):REM CHOOSES WORD
 LIST
105 ON G GOTO 120,130,140,150,160
110 LET A$="ALLYBAILCHITDIMEDIGITFLIRTGR
AINHOUSEFIDGETHOURLYLININGNOZZLE"
115 LET A$=A$+"MOVABLEOPINIONPARTNERRECO
VER"
119 GOTO 200
120 LET A$="RILETORNCLIPPOURFIFTHSURLYSL
OTSARGUESLOWLYFIRMLYGROWTHMENACE"
125 LET A$=A$+"FIGURESTRUMPETSMARTENBAST
ION"
129 GOTO 200
130 LET A$="FROGHALFMASTRENTWHEELTRUSTST
UFFSCRUBSCRIPTREBORNPUBLICMUTTER"
135 LET A$=A$+"INTRUDEHAUGHTYGIRAFFEENCH
ANT"
139 GOTO 200
140 LET A$="GUSHHAILLOVERENDWHARFTRULYST
UDYSCREWSCREENREALTYPUCKERMUTINY"
145 LET A$=A$+"INSTANTHATRACKGHOSTLYDEVE
LOP"
```

```
149 GOTO 200
150 LET A$="AGOGBACKCHATDIALCHASMFLINGGE
NUSTHIGHHECTICJIGGERLIGATEMOTHER"
155 LET A$=A$+"LIGHTENNOXIOUSPAPRIKAREBO
UND"
159 GOTO 200
160 LET A$="AIRYBABECHEFDIETCHESTFIGHTGE
RMSHEDGEHEIGHTJESTERLIMPETNOTICE"
165 LET A$=A$+"MOUTHEDOILSKINPARABLEREAL
IZE"
200 LET W=INT(4*RND(0)):REM CHOOSES ONE
OF FOUR WORDS
210 LET X=INT(4*RND(0))
220 LET L=X+4:REM CHOOSES WORD LENGTH
230 LET P=X*(2*X+14)+W*L+1
240 LET B$=MID$(A$,P,L):REM PLUCKS OUT A
WORD
250 LET E$=B$
260 LET F$=B$
270 FOR I=1 TO L:REM SCRAMBLES THE WORD
280 LET J=INT(L*RND(0))+1
290 LET C$=MID$(F$,J,1)
300 IF C$=" " THEN 280
310 LET B$=LEFT$(B$,I-1)+C$+RIGHT$(B$,L-
I)
320 LET F$=LEFT$(F$,J-1)+" "+RIGHT$(F$,L
-J)
330 NEXT I
340 IF B$=E$ THEN 260:REM CHECKS THAT TH
E WORD HAS BEEN SCRAMBLED
350 FOR T=1 TO 1000: NEXT T
360 PRINT "□"
370 PRINT TAB(5);B$
380 PRINT:PRINT "ENTER THE UNSCRAMBLED W
ORD"
390 PRINT:PRINT "ENTER . TO QUIT"
400 PRINT:INPUT D$
410 IF D$="." THEN PRINT "THE WORD IS ";
E$:GOTO 470
420 IF D$<>E$ THEN 450
430 PRINT:PRINT "RIGHT ON! NOW TRY THIS"
:Z=1:GOSUB 9000
440 GOTO 100
450 PRINT:PRINT "WRONG! TRY AGAIN":Z=0:G
```

```
OSUB 9000
460 GOTO 350
470 PRINT:PRINT "AGAIN? PRESS Y OR N"
480 GET A$:IF A$="" THEN 480
490 IF A$="Y" THEN 95
500 STOP
9000 REM PLAYS MUSIC
9010 RESTORE:IF Z=0 THEN 9020
9015 FOR I=1 TO 7*5:READ D:NEXT
9020 FOR I=1 TO 3:READ H(I),L(I):NEXT I:
READ D
9025 FOR I=1 TO 3:POKE 54269+7*I,33:NEXT
9030 IF D=-1 THEN 9080
9040 FOR I=1 TO 3:V=54266+7*I:POKE V,H(I
):POKE V-1,L(I):NEXT I
9050 FOR T=1 TO D:NEXT T
9060 GOTO 9020
9080 FOR I=1 TO 3:POKE 54269+7*I,0:NEXT:
REM TURNS OFF NOTE
9090 RETURN
9900 DATA 17,37,12,216,10,205,400
9910 DATA 14,107,10,205,8,147,400
9920 DATA 12,216,10,205,7,53,400
9930 DATA 10,205,8,147,7,53,400
9940 DATA -1,-1,-1,-1,-1,-1,-1
9950 DATA 38,126,28,214,9,159,400
9960 DATA 28,214,28,214,12,32,100
9970 DATA 32,94,25,177,12,216,100
9980 DATA 28,214,24,63,9,159,100
9990 DATA -1,-1,-1,-1,-1,-1,-1
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME.

"Anagrams I"

AIM OF THE GAME

Some people are good at anagrams, making words out of a jumble of letters. This program is designed to improve your skill with anagrams.

Mathematics can sometimes be like sorting out an anagram. You have all the facts, all the data, but they are jumbled up. With training, you learn how to make sense out of the chaos. By trial and error you patiently work through the different possibilities, until a flash of inspiration or insight guides you to the correct solution.

Of course, the list of words is short and fixed, so to make the game more interesting, you should have the word list altered by a friend, and then test your skill. To save memory space and to disguise the words, I have placed the "dictionary" of words in A$. There are 4 words of 4 letters followed by 4 of 5 letters, then 4 of 6 letters and finally 4 of 7 letters. You must replace words in precisely this fashion because the program is designed to pick out a sequence of letters from A$ that forms a word. I used a mathematical technique called interpolation to do this.

HOW TO PLAY

When you run this program a scrambled word containing from 4 to 7 letters is displayed in this fashion

```
FHAWR
ENTER THE UNSCRAMBLED WORD
ENTER . TO QUIT
```

After you enter your solution to the anagram, it is compared with the correct word and the message

```
RIGHT ON! NOW TRY THIS
```

or

```
WRONG! TRY AGAIN
```

appears. If you are right a new anagram is posed. If you were wrong, the same anagram is repeated.

If you get stuck on a word, enter a period (.) when prompted to enter the answer. The program will respond with the message

```
THE WORD IS WHARF (or whatever it is)
AGAIN? PRESS Y OR N
```

If you press the Y key another scrambled word is given to you. If you press the key N, the program will stop.

THE MATHEMATICS AND CODE

The start of the four letter words is at the first letter of A$. The start of the five letter words is at letter 17 of A$. Six letter words start at the 37th letter of A$ and seven letter words at the 61st letter of A$. The formula

```
P = X*(2*X+14) + 1
```

is used in line 23Ø to find the starting point of the words, where the length of the words is X+4.

Let us try out the formula. For X=Ø (words of length 4), P=1. For X=1 (words of length 5), P=17 and so on. A method of finding this formula is called Lagrange Interpolation. What I needed was P=1 for X=Ø, P=17 for X=1, P=37 for X=2, P=61 for X=3. I decided to subtract 1 from all the values of P to have easier numbers. So now I wanted P=Ø for X=Ø, P=16 for X=1, P=36 for X=2 and P=6Ø for X=3.

We construct the required function P of X by using polynomials. A polynomial P , for example, is a function like $P = 2x^2 - 3x + 4$. We build the polynomial P from basic polynomials that have the value 1 at only one point and are zero for all of the other points. In this case, we want a polynomial that is zero at X = Ø, 2 and 3. We start with R = X(X–2)(X–3) which has the required values of zero. We want R = 1 at X = 1. In this case, R = 2, so we divide by 2 to get

```
R = X(X–2)(X–3)/2
```

In a similar fashion, we define

```
S = –X(X–1)(X–3)/2
```

which has the value 1 at X = 2 and is zero at X = Ø, 1 and 3. Finally, we define

```
T = X(X–1)(X–2)/6
```

which has the value 1 at X = 3 and is zero at X = Ø, 1 and 2.

The interpolating polynomial P that we require is a combination of these basic polynomials.

```
P = R*16 + S*36 + T*6Ø
  = X*(2*X + 14)
```

"Anagrams II" (With Hints)

```
1Ø PRINT"□":PRINT:PRINT "INSTRUCTIONS?P
RESS Y OR N":PRINT
2Ø GET A$:IF A$="" THEN 2Ø
3Ø IF A$<>"Y" THEN 85
4Ø PRINT "THIS GAME GIVES YOU A JUMBLED
STRING OF "
5Ø PRINT "LETTERS AND IT IS YOUR JOB TO
FIND THE  "
6Ø PRINT "HIDDEN WORD.THE STRING ROMST,F
OR EXAMPLE"
7Ø PRINT "IS A SCRAMBLED VERSION OF STOR
M."
8Ø PRINT:INPUT "PRESS RETURN TO START";
85 FOR L=54272 TO 54296:POKE L,Ø:NEXT:PO
KE 54296,15:REM SETS UP VOICES
9Ø FOR I=1 TO 3:V=54269+7*I:POKE V+1,19Ø
:POKE V+2,25Ø:NEXT
95 PRINT "□"
1ØØ LET G=INT(6*RND(Ø)):REM CHOOSES WORD
LIST
1Ø5 ON G GOTO 12Ø,13Ø,14Ø,15Ø,16Ø
11Ø LET A$="ALLYBAILCHITDIMEDIGITFLIRTGR
AINHOUSEFIDGETHOURLYLININGNOZZLE"
115 LET A$=A$+"MOVABLEOPINIONPARTNERRECO
VER"
119 GOTO 2ØØ
12Ø LET A$="RILETORNCLIPPOURFIFTHSURLYSL
OTSARGUESLOWLYFIRMLYGROWTHMENACE"
125 LET A$=A$+"FIGURESTRUMPETSMARTENBAST
ION"
129 GOTO 2ØØ
13Ø LET A$="FROGHALFMASTRENTWHEELTRUSTST
UFFSCRUBSCRIPTREBORNPUBLICMUTTER"
135 LET A$=A$+"INTRUDEHAUGHTYGIRAFFEENCH
ANT"
139 GOTO 2ØØ
14Ø LET A$="GUSHHAILLOVERENDWHARFTRULYST
UDYSCREWSCREENREALTYPUCKERMUTINY"
```

```
145 LET A$=A$+"INSTANTHATRACKGHOSTLYDEVE
LOP"
149 GOTO 200
150 LET A$="AGOGBACKCHATDIALCHASMFLINGGE
NUSTHIGHHECTICJIGGERLIGATEMOTHER"
155 LET A$=A$+"LIGHTENNOXIOUSPAPRIKAREBO
UND"
159 GOTO 200
160 LET A$="AIRYBABECHEFDIETCHESTFIGHTGE
RMSHEDGEHEIGHTJESTERLIMPETNOTICE"
165 LET A$=A$+"MOUTHEDOILSKINPARABLEREAL
IZE"
200 LET W=INT(4*RND(0)):REM CHOOSES ONE
OF FOUR WORDS
205 LET C=0
210 LET X=INT(4*RND(0))
220 LET L=X+4:REM CHOOSES WORD LENGTH
230 LET P=X*(2*X+14)+W*L+1
240 LET B$=MID$(A$,P,L):REM PLUCKS OUT A
WORD
250 LET E$=B$
260 LET F$=B$
270 FOR I=1 TO L:REM SCRAMBLES THE WORD
280 LET J=INT(L*RND(0))+1
290 LET C$=MID$(F$,J,1)
300 IF C$=" " THEN 280
310 LET B$=LEFT$(B$,I-1)+C$+RIGHT$(B$,L-
I)
320 LET F$=LEFT$(F$,J-1)+" "+RIGHT$(F$,L
-J)
330 NEXT I
340 IF B$=E$ THEN 260:REM CHECKS THAT TH
E WORD HAS BEEN SCRAMBLED
350 FOR T=1 TO 1000: NEXT T
360 PRINT "□"
370 PRINT TAB(5);B$
380 PRINT:PRINT "ENTER THE UNSCRAMBLED W
ORD"
390 PRINT:PRINT "ENTER . TO QUIT"
400 PRINT:INPUT D$
410 IF D$="." THEN PRINT "THE WORD IS ";
E$:GOTO 470
420 IF D$<>E$ THEN 450
430 PRINT:PRINT "RIGHT ON! NOW TRY THIS"
```

```
:Z=1:GOSUB 9000
440 GOTO 100
450 PRINT:PRINT "WRONG! TRY AGAIN":Z=0:G
OSUB 9000
460 GOTO 510
470 PRINT:PRINT "AGAIN? PRESS Y OR N"
480 GET A$:IF A$="" THEN 480
490 IF A$="Y" THEN 95
500 STOP
510 PRINT:PRINT "WANT SOME HELP? PRESS Y
OR N"
520 GET Z$:IF Z$="" THEN 520
530 IF Z$<>"Y" THEN 360
540 LET C=C+1
550 PRINT:PRINT "THE WORD STARTS WITH":P
RINT
560 PRINT TAB(5);LEFT$(E$,C)
570 GOTO 350
9000 REM PLAYS MUSIC
9010 RESTORE:IF Z=0 THEN 9020
9015 FOR I=1 TO 7*5:READ D:NEXT
9020 FOR I=1 TO 3:READ H(I),L(I):NEXT I:
READ D
9025 FOR I=1 TO 3:POKE 54269+7*I,33:NEXT
9030 IF D=-1 THEN 9080
9040 FOR I=1 TO 3:V=54266+7*I:POKE V,H(I
):POKE V-1,L(I):NEXT I
9050 FOR T=1 TO D:NEXT T
9060 GOTO 9020
9080 FOR I=1 TO 3:POKE 54269+7*I,0:NEXT:
REM TURNS OFF NOTE
9090 RETURN
9900 DATA 17,37,12,216,10,205,400
9910 DATA 14,107,10,205,8,147,400
9920 DATA 12,216,10,205,7,53,400
9930 DATA 10,205,8,147,7,53,400
9940 DATA -1,-1,-1,-1,-1,-1,-1
9950 DATA 38,126,28,214,9,159,400
9960 DATA 28,214,28,214,12,32,100
9970 DATA 32,94,25,177,12,216,100
```

```
9980 DATA 28,214,24,63,9,159,100
9990 DATA -1,-1,-1,-1,-1,-1,-1
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME.

"Anagrams II" (With Hints)

If you were having trouble finding the correct word for the anagrams in the previous program, then you might prefer this version. It is the same game, except that if you are wrong, you are then prompted with

```
WANT SOME HELP? PRESS Y OR N
```

If you press the key Y, then you are told that

```
THE WORD STARTS WITH
W
```

for the case of WHARF scrambled up as FHAWR.

After a short pause, the hint disappears and the anagram is again presented. If you are still stuck and enter a wrong answer, and if you want more help, you are told that the word starts with

```
WH
```

and so on. Eventually the entire word will be presented to you, but I am sure that with a little help you will be able to solve the anagram.

"Hangman"

```
10 PRINT "□":PRINT:PRINT "INSTRUCTIONS?
PRESS Y OR N":PRINT
20 GET A$:IF A$="" THEN 20
30 IF A$<>"Y" THEN 100
40 PRINT "YOU MUST DISCOVER WHAT THE WOR
D IS BY"
50 PRINT "GUESSING LETTERS. YOU ONLY HAV
E 13 MOVES"
60 PRINT:INPUT "PRESS RETURN TO START";
100 PRINT "□":LET G=INT(6*RND(0)):REM RA
NDOMLY CHOOSES WORD LIST
101 LET Z$="*******":REM FORMS A SCREEN
102 LET L$=""
105 ON G GOTO 120,130,140,150,160
110 LET A$="ALLYBAILCHITDIMEDIGITFLIRTGR
AINHOUSEFIDGETHOURLYLININGNOZZLE"
115 LET A$=A$+"MOVABLEOPINIONPARTNERRECO
VER"
119 GOTO 200
120 LET A$="RILETORNCLIPPOURFIFTHSURLYSL
OTSARGUESLOWLYFIRMLYGROWTHMENACE"
125 LET A$=A$+"FIGURESTRUMPETSMARTENBAST
ION"
129 GOTO 200
130 LET A$="FROGHALFMASTRENTWHEELTRUSTST
UFFSCRUBSCRIPTREBORNPUBLICMUTTER"
135 LET A$=A$+"INTRUDEHAUGHTYGIRAFFEENCH
ANT"
139 GOTO 200
140 LET A$="GUSHHAILLOVERENDWHARFTRULYST
UDYSCREWSCREENREALTYPUCKERMUTINY"
145 LET A$=A$+"INSTANTHATRACKGHOSTLYDEVE
LOP"
149 GOTO 200
150 LET A$="AGOGBACKCHATDIALCHASMFLINGGE
NUSTHIGHHECTICJIGGERLIGATEMOTHER"
155 LET A$=A$+"LIGHTENNOXIOUSPAPRIKAREBO
UND"
159 GOTO 200
160 LET A$="AIRYBABECHEFDIETCHESTFIGHTGE
```

```
RMSHEDGEHEIGHTJESTERLIMPETNOTICE"
165 LET A$=A$+"MOUTHEDOILSKINPARABLEREAL
IZE"
200 LET W=INT(4*RND(0)):REM PICKS ONE OF
FOUR WORDS
210 LET X=INT(4*RND(0))
220 LET L=X+4:REM PICKS WORD LENGTH
230 LET P=X*(2*X+14)+W*L+1:REM INTERPOLA
TES
240 LET B$=MID$(A$,P,L):REM PLUCKS WORD
OUT OF A$
250 LET M=13:REM SETS LIMIT ON NUMBER OF
MOVES
255 LET L=LEN(B$)
260 LET F$=MID$(Z$,1,L)
270 GOTO 360
280 FOR I=1 TO L:REM CHECKS IF ENTERED L
ETTER IS IN THE WORD
290 IF MID$(B$,I,1)=D$ THEN LET F$=LEFT$
(F$,I-1)+D$+RIGHT$(F$,L-1)
300 NEXT I
360 PRINT "□":PRINT:PRINT
370 PRINT TAB(5);F$:PRINT
375 IF F$=B$ THEN 430:REM CHECK IF WORD
HAS BEEN FOUND
376 IF M<1 THEN PRINT "THE WORD IS ";B$:
GOTO 440:REM NO MOVES LEFT
380 PRINT M;"MOVES TO GO"
390 PRINT:PRINT "TYPE A LETTER USE A . T
O QUIT":PRINT
395 LET M=M-1
396 PRINT "LETTERS USED SO FAR":PRINT:PR
INT L$:PRINT
400 GET Z$:IF Z$="" THEN 400
405 LET D$=Z$
410 IF D$="." THEN PRINT "THE WORD IS ";
B$:GOTO 440
415 FOR I=1 TO LEN(L$):IF D$=MID$(L$,I,1
) THEN 420
416 NEXT I
417 LET L$=L$+D$
420 GOTO 280
430 PRINT "RIGHT ON"
440 PRINT: PRINT "AGAIN? PRESS Y OR N"
```

```
45Ø GET Z$:IF Z$="" THEN 45Ø
46Ø IF Z$="Y" THEN 1ØØ
47Ø STOP
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME.

"Hangman"

AIM OF THE GAME

This game is a version of the children's game, only there are no gallows and stick-figures. It is enough to know that you only have 13 attempts to find the correct word. This is more difficult, as a result, than "ANAGRAMS" where you not only know the length of the word, but also which letters appear in it.

HOW TO PLAY

When you run this program you are presented with a message like

```
       * * * * * * *
13 MOVES TO GO
TYPE A LETTER USE A . TO QUIT
LETTERS USED SO FAR
```

Suppose you enter the letter A, then the message is repeated with 12 in place of 13. Now suppose you enter the letter E, then the message becomes

```
       *E*E* * *
11 MOVES TO GO
TYPE A LETTER USE A . TO QUIT
LETTERS USED SO FAR
AE
```

You continue in this fashion until you run out of moves, or you find all the letters in the word. If you run out of moves, the word is given to you. If you get frustrated, and want to quit, just enter a period (.) instead of a letter and the word is given to you. In any case, you will be invited to play again, or to quit with the prompt

```
AGAIN? PRESS Y OR N
```

If you would like to alter the number of guesses allowed, change M=13 in line 25Ø to something that suits your fancy.

This program is a very simple program with no audio-visual effects. You might like to add your own. The usual graphics for this game is a gallows and a stick figure which is gradually drawn with each guess that is wrong. The program does not check repeated entries so that if you fail to notice a letter in the used letter list and use it a second time, it is a wasted move. It would be a simple exercise for the beginning programmer to add lines to run a check on the entry and print a message if the letter has already been used. Take a look at lines 28Ø to 3ØØ and use that for a

guide. Replace B$ with L$ in the checking part, and change the conclusion after the THEN.

"Find a Word"

```
1Ø PRINT "□":PRINT:PRINT "FIND A WORD IS
 A GAME WHERE YOU HAVE TO "
2Ø PRINT "FIND WORDS HIDDEN IN A SQUARE
OF LETTERS"
3Ø PRINT "THERE ARE FROM 7 TO 11 WORDS,U
P AND DOWN"
4Ø PRINT "ACROSS AND DIAGONAL"
5Ø PRINT:PRINT "ENTER . TO QUIT"
6Ø PRINT:INPUT "PRESS RETURN TO START";
1ØØ PRINT "□":PRINT:PRINT "GET A SNACK,
THIS WILL TAKE A MOMENT"
1Ø1 FOR I=Ø TO 9:FOR J=Ø TO 9:M$(I,J)="*
":NEXT J,I:REM FILLS M WITH "*"
1Ø2 LET KK=6+INT(5*RND(Ø)):N=KK:REM CHOO
SES NUMBER OF WORDS
1Ø3 WL=Ø:REM CHECKS NUMBER OF LETTERS IN
 ALL WORDS
1Ø4 FOR K=Ø TO KK:REM SELECTS WORDS
1Ø5 LET G=INT(6*RND(Ø))
1Ø8 ON G GOTO 12Ø,13Ø,14Ø,15Ø,16Ø
11Ø LET A$="ALLYBAILCHITDIMEDIGITFLIRTGR
AINHOUSEFIDGETHOURLYLININGNOZZLE"
115 LET A$=A$+"MOVEABLEOPINIONPARTNERRECO
VER"
119 GOTO 2ØØ
12Ø LET A$="RILETORNCLIPPOURFIFTHSURLYSL
OTSARGUESLOWLYFIRMLYGROWTHMENACE"
125 LET A$=A$+"FIGURESTRUMPETSMARTENBAST
ION"
129 GOTO 2ØØ
13Ø LET A$="FROGHALFMASTRENTWHEELTRUSTST
UFFSCRUBSCRIPTREBORNPUBLICMUTTER"
135 LET A$=A$+"INTRUDEHAUGHTYGIRAFFEENCH
ANT"
139 GOTO 2ØØ
14Ø LET A$="GUSHHAILLOVERENDWHARFTRULYST
UDYSCREWSCREENREALTYPUCKERMUTINY"
145 LET A$=A$+"INSTANTHATRACKGHOSTLYDEVE
LOP"
149 GOTO 2ØØ
```

```
150 LET A$="AGOGBACKCHATDIALCHASMFLINGGE
NUSTHIGHHECTICJIGGERLIGATEMOTHER"
155 LET A$=A$+"LIGHTENNOXIOUSPAPRIKAREBO
UND"
159 GOTO 200
160 LET A$="AIRYBABECHEFDIETCHESTFIGHTGE
RMSHEDGEHEIGHTJESTERLIMPETNOTICE"
165 LET A$=A$+"MOUTHEDOILSKINPARABLEREAL
IZE"
200 LET W=INT(4*RND(0)):REM CHOOSES ONE
OF FOUR WORDS
210 LET X=INT(4*RND(0))
220 LET L=X+4:REM CHOOSES WORD LENGTH
230 LET P=X(2*X+14)+W*L+1
240 LET B$(K)=MID$(A$,P,L):REM PLUCKS OU
T A WORD
241 IF K=0 THEN 246
242 FOR I=0 TO K-1:IF B$(K)=B$(I) THEN 1
05:REM CHECKS FOR DUPLICATES
245 NEXT I
246 WL=WL+LEN(B$(K))
250 NEXT K
255 IF WL>55 THEN 102:REM IF TOTAL LETTE
RS >55 CHOOSE A NEW SET OF WORDS
260 FOR K=0 TO KK
265 L=LEN(B$(K))
270 I=INT(10*RND(0)):J=INT(10*RND(0)):RE
M CHOOSES STARTING POINT OF WORD
272 FOR M=0 TO 7:C(M)=1:NEXT M
275 FOR S=0 TO L-1:REM CHECKS FOR WAY TH
E WORD CAN RUN
280 IF J+S>9 THEN C(0)=0
281 IF J+S>9 THEN 285
282 IF M$(I,J+S)<>"*" THEN C(0)=0
285 IF J-S<0 THEN C(1)=0
286 IF J-S<0 THEN 290
287 IF M$(I,J-S)<>"*" THEN C(1)=0
290 IF(I+S>9 OR J+S>9) THEN C(2)=0
291 IF(I+S>9 OR J+S>9) THEN 295
292 IF M$(I+S,J+S)<>"*" THEN C(2)=0
295 IF(I-S<0 OR J-S<0) THEN C(3)=0
296 IF(I-S<0 OR J-S<0) THEN 300
297 IF M$(I-S,J-S)<>"*" THEN C(3)=0
300 IF(I-S<0 OR J+S>9) THEN C(4)=0
```

```
301 IF(I-S<Ø OR J+S>9) THEN 3Ø5
3Ø2 IF M$(I-S,J+S)<>"*" THEN C(4)=Ø
3Ø5 IF(I+S>9 OR J-S<Ø) THEN C(5)=Ø
3Ø6 IF(I+S>9 OR J-S<Ø) THEN
3Ø7 IF M$(1+S,J-S)<>"*" THEN C(5)=Ø
31Ø IF I+S>9 THEN C(6)=Ø
311 IF I+S>9 THEN 315
312 IF M$(I+S,J)<>"*" THEN C(6)=Ø
315 IF I-S<Ø THEN C(7)=Ø
316 IF I-S<Ø THEN 32Ø
317 IF M$(I-S,J)<>"*" THEN C(7)=Ø
32Ø NEXT S
325 FOR M=Ø TO 7:IF C(M)=1 THEN 338
33Ø NEXT M
335 GOTO 27Ø
338 M=INT(8*RND(Ø)):IF C(M)=Ø THEN 338:R
EM CHOOSES DIRECTION OF WORD
34Ø ON M+1 GOTO 341,342,343,344,345,346,
347,348
341 U=Ø:B=1:GOTO 35Ø:REM SETS UP AND ACR
OSS PARAMETERS
342 U=Ø:B=-1:GOTO 35Ø
343 U=1:B=1:GOTO 35Ø
344 U=-1:B=-1:GOTO 35Ø
345 U=-1:B=1:GOTO 35Ø
346 U=1:B=-1:GOTO 35Ø
347 U=1:B=Ø:GOTO 35Ø
348 U=-1:B=Ø
35Ø FOR S=Ø TO L-1:M$(I+S*U,J+S*B)=MID$(
B$(K),S+1,1):NEXT S:REM PLACES WORDS
355 NEXT K
358 IF N=-1 THEN 4ØØ
359 D$(N)=""
36Ø PRINT "□":PRINT "FIND";N+1;"WORDS";
361 FOR I=Ø TO 9:FOR J=Ø TO 9
362 IF M$(I,J)="*" THEN M$(I,J)=CHR$(ASC
("A")+INT(26*RND(Ø))):REM FILLS M$
363 NEXT J,I
365 FOR I=Ø TO 9:PRINT:PRINT:FOR J=Ø TO
9:PRINT " ";M$(I,K);:NEXT J,I
366 PRINT "[S]":PRINT:PRINT:PRINT:PRINT TA
B(25);"WORDS FOUND"
367 FOR I=KK TO N STEP -1:PRINT TAB(25);
D$(I):NEXT I
```

```
368 PRINT "[S]":FOR I=1 TO 21:PRINT:NEXT I
:REM PRINTS AT SCREEN BOTTOM
37Ø PRINT:INPUT "ENTER A WORD YOU SEE";D
$(N)
371 IF N=KK THEN 375
372 FOR K=KK TO N+1 STEP -1:IF D$(N)=D$(
K) THEN 358:REM CHECKS REPEAT ENTRY
374 NEXT K
375 IF D$(N)="." THEN 5ØØ
38Ø FOR K=Ø TO KK:IF D$(N)<>B$(K) THEN 3
9Ø:REM CHECKS ENTRY AGAINST WORD LIST
385 PRINT "CORRECT":N=N-1:GOTO 399
39Ø NEXT K
395 PRINT "SORRY THAT IS NOT ONE THAT CO
UNTS"
399 FOR T=1 TO 1ØØØ:NEXT T:GOTO 358
4ØØ PRINT "[C]":PRINT:PRINT " WELL DONE! Y
OU FOUND ALL THE WORDS"
41Ø PRINT:PRINT "AGAIN? PRESS Y OR N"
42Ø GET A$:IF A$="" THEN 42Ø
43Ø IF A$="Y" THEN 1ØØ
44Ø STOP
5ØØ PRINT "[C]":PRINT "THE WORDS WERE":PRI
NT
51Ø FOR K=Ø TO KK:PRINT B$(K):NEXT K:GOT
O 41Ø
```

Note: The symbol [C] represents SHIFT CLEAR/HOME, and [S] represents HOME.

"Find a Word"

AIM OF THE GAME

Using the same dictionary of words as in the previous two games, we can make a very challenging game of find a word. Words are hidden in a 1Ø by 1Ø square of letters. The words can go up or down, across or back, or even diagonally. A random number of words are placed randomly in the square and the remaining positions in the square are filled with letters. You have to locate the hidden words.

HOW TO PLAY

There will be at least 7 words and at most 11 words hidden in the square. You will be told how many there are. When you RUN this game it takes about one minute to set up the puzzle. While this is going on, you get the message

```
GET A SNACK THIS TAKES A MOMENT
```

Then you will be presented with a puzzle like

```
FIND 8 WORDS

X K E S U O H J U H
T R U M P E T E L A
L J X B Y O F S N U          WORDS FOUND
P F L L F E P T C G
Y B U L H L D E A H
O R V S F B W R X T
T J U N T A X W B Y
S U N E G R T V N A
S G Z S A A B A I L
C M P F U P H K R E

ENTER A WORD YOU SEE?
```

If you enter the word BAIL, the message is repeated, only there will be 7 words to find, and the list of words found will contain the word BAIL.

You may find a word that accidently appears in the square. If your entry does not match one of the chosen words, you are told

```
SORRY THAT IS NOT ONE THAT COUNTS
```

and then the previous prompt is repeated.

It will not accept the same word twice. If you want to quit, just enter . (a period) when asked to enter a word. The list of words is displayed and you are prompted with

AGAIN? PRESS Y OR N

You press the key N to quit or Y to play another game.

THE CODE

This is a very long program for this book. The main difficulty was fitting the words into the square. The words are not allowed to intersect one another. Otherwise we would have to make sure that they intersected in a common letter and this would be too time consuming. I made the choice of not letting words intersect. But then it might become impossible to locate all the words in the 1Ø by 1Ø array. By trial and error, I found that if the chosen words had more than 65 letters in total, it might be impossible to locate all the words in the 1ØØ places available. So in line 255, if the total number of letters is greater than 55 (a considerable margin for error), then a new list of words is generated. You may experiment with the number of letters allowed by changing the limit in line 255.

Fitting the words into the square is done in lines 26Ø to 355. This is the part that takes all the time in setting up the puzzle. The square M$ is filled with "*" to start with in line 1Ø1. A random row I and column J is selected in line 17Ø. Lines 272 to 32Ø check to see which way the word could run, forward, backward, diagonally down to the right, diagonally up to the left, diagonally up to the right, diagonally down to the left, vertically down or up. The vector C contains a 1 or a Ø in each of eight positions corresponding to the above possibilities. A Ø means that that particular choice is blocked. Line 338 chooses one of the possible ways of locating the word. Lines 34Ø to 348 select the appropriate parameters and line 35Ø locates the word. For instance, if the word is being placed diagonally down to the left, starting from (I,J), U would be 1 and B would be 1. Then:

M$(I,J) = the first letter in the word B$(K)
M$(I+1,J+1) = the second letter in the word B$(K)
M$(I+2,J+2) = the third letter in the word B$(K)

If the word was BAIL and (I,J) was (3,5), then we would have

```
          Column 5
Row 3          B
                 A
                   I
                     L
```

Finally, lines 361 to 363 fill up the spaces in M$ that still contain *'s with randomly chosen letters. I use this loop to create a pause during the playing of the game so that the message about how many words to go is displayed on its own.

I did not add any audio-visual effects as I thought the listing was too long as it was. You might like to try the following. When a word is correctly located, it is displayed in reverse letters, or in a different color, or simply flashes for a moment while the congratulatory music plays!

"Enigma I"

```
11Ø PRINT "□"
12Ø LET C=INT(25*RND(Ø))+1
13Ø LET Z=ASC("Z")
135 PRINT "ENTER YOUR MESSAGE AND I WILL
 CODE IT."
136 PRINT
137 PRINT "USE A ";CHR$(34);".";CHR$(34)
;" TO END CODE."
138 PRINT
14Ø GET A$:IF A$="" THEN 14Ø
15Ø LET B=ASC(A$)
16Ø IF A$=" " THEN PRINT " ";:GOTO 14Ø
17Ø IF A$="." THEN 22Ø
18Ø LET D=B+C
185 IF A$<"A" OR A$>"Z" THEN PRINT A$;:G
OTO 14Ø
19Ø IF D>Z THEN LET D=D-26
2ØØ PRINT CHR$(D);
21Ø GOTO 14Ø
22Ø PRINT
23Ø PRINT "THE CODE IS PLUS?"
24Ø INPUT E
25Ø IF E=C THEN PRINT "RIGHT ON"
26Ø IF E<>C THEN PRINT "WRONG IT IS";C
27Ø PRINT:PRINT "AGAIN? PRESS Y OR N"
275 GET A$:IF A$="" THEN 275
28Ø IF A$="Y" THEN 11Ø
29Ø STOP
```

Note: In lines 14Ø and 275, the double quotes "" define an empty string. The symbol □ represents SHIFT CLEAR/HOME.

"Enigma I"

AIM OF THE GAME

This program converts your computer into a coding typewriter, something like the German coder of the second world war, which was called ENIGMA. As you type in a message, it is displayed in coded form. You terminate your message by typing a period. A space is accepted so you may use the space key to type a space in your message.

After your message has been typed, you are asked to solve the code. The letters in the alphabet have simply been shifted to the right by a constant number of letters. This is the number you are prompted to enter.

Of course it will not be challenging for you because you know what the original message was! But challenge a friend to solve the code. One person types in a message and then the other attempts to decode it.

After a little experience you will be a master at decoding secret messages.

This is a very simple form of coding and would be too easy to crack. Mathematicians and computer scientists are working on developing codes that are almost impossible to crack. So much important personal and financial information is stored in data banks, you can imagine how important it is to protect that information from unauthorized use.

HOW TO PLAY

The program commences with the prompt

```
ENTER YOUR MESSAGE AND I WILL CODE IT
USE A . TO END CODE.
```

Your keyboard will now act like a typewriter. Every key you press will give an instant response. If you press a letter key, a different letter will appear on the screen. Other characters will be typed as themselves except for a period, which will end the input. You will then be asked the question

```
THE CODE IS PLUS?
```

You enter your answer, it is checked and a message

```
RIGHT ON
```

or

```
WRONG IT IS 15 (or some such number).
```

Then you will be invited to play again or to quit.

You might like to try a much more challenging version of "ENIGMA". In this version, the letters of the alphabet are randomly mixed up! You

could not find a more difficult code to break. You have to look at patterns of repeating letters and three letter words and, by trial and error, try to break the code.

"Enigma II"

```
10Ø LET Y=ASC("A")-1
11Ø PRINT "□"
115 PRINT "GENERATING THE CODE"
116 C$="":D$="":REM INITIALIZE
119 FOR I=Ø TO 25
121 LET C$=C$+STR$(ASC("A")+1)
123 NEXT I
125 FOR I=1 TO 26
127 LET J=INT(26*RND(Ø))+1
129 IF MID$(C$,J,1)="*" THEN 127:REM CHE
CKS IF LETTER HAS BEEN ASSIGNED
131 LET C$=LEFT$(C$,J-1)+"*"+RIGHT$(C$,2
7-J):REM REPLACES LETTER BY *
133 LET B$=B$+CHR$(J+Y):REM Y DEFINED IN
 LINE 1ØØ, THIS CHOOSES A RANDOM LETTER
134 NEXT I
135 PRINT "□":PRINT:PRINT "ENTER YOUR ME
SSAGE AND I WILL CODE IT"
136 PRINT:PRINT "USE A ";CHR$(34);".";CH
R$(34);" TO END CODE."
14Ø GET A$: IF A$="" THEN 14Ø
15Ø LET B=ASC(A$)-Y:REM B IS A NUMBER FR
OM 1 TO 26
17Ø IF A$="." THEN 22Ø
175 IF A$<"A" OR A$>"Z" THEN PRINT A$;:G
OTO 9Ø1Ø:REM USE GOTO 14Ø TO OMIT SOUND
18Ø LET D$=MID$(B$,B,1):REM CHOOSES THE
 CODED FORM OF INPUT LETTER
19Ø PRINT D$;:GOTO 9Ø1Ø:REM MAKES SOUND
OF OLD TYPEWRITER (MAY OMIT GOTO 9Ø1Ø)
2ØØ GOTO 14Ø
22Ø PRINT:PRINT
23Ø PRINT "PRESS Ø FOR THE CODE, 1 FOR A
 NEW CODE"
235 GET A$:IF A$="" THEN 235
24Ø E=VAL(A$)
25Ø IF E<>Ø THEN 11Ø
26Ø PRINT "□"
27Ø FOR I=1 TO 26
28Ø PRINT CHR$(Y+1);"=";MID$(B$,I,1),
```

```
290 NEXT I
295 PRINT:PRINT:PRINT "AGAIN? PRESS Y OR
 N"
296 GET A$:IF A$="" THEN 296
297 IF A$="Y" THEN 110
298 STOP
300 PRINT " ";
310 GOTO 140
9000 REM:SOUND OF ANCIENT TYPEWRITER
9010 V=54296:W=54276:A=54277:H=54273:L=5
4272
9020 FOR X=13 TO 10 STEP-1:POKEV,X:POKEW
,129:POKEA,99:POKE H,40:POKE L,200:NEXT
9030 POKE W,0:POKE A,0:GOTO 140
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME.

"Enigma II"

HOW TO PLAY

When you run this program it commences with the message

```
GENERATING THE CODE
```

After a short period of time, a new message appears

```
ENTER YOUR MESSAGE AND I WILL CODE IT
```

You then type in your message. As you touch the keys the message appears in coded form. You cannot erase mistakes, so type carefully! Punctuation and other characters appear in their own form, except a period (.) which terminates your message. You are then prompted with

```
PRESS Ø FOR THE CODE, 1 FOR A NEW CODE
```

If you press the key Ø, a listing of the code appears, giving the code letter for each letter in the alphabet, for example

```
A=T    B=R    C=Y    D=F
E=G    F=O    G=L    H=V
I=J    J=C    K=N    L=B
M=D    N=H    O=P    P=A
Q=S    R=U    S=Q    T=M
U=Z    V=K    W=W    X=E
Y=I    Z=X
AGAIN? PRESS Y OR N
```

and you are prompted to play again or to quit.

THE CODE

You may find it inconvenient that the period is used to end the message. If you have a long message, with many sentences, you may like to use the period for punctuation. In that case, you should alter line 17Ø and replace the "." by "#" or something that you prefer.

There are also some small sound effects in lines 9ØØØ to 9Ø3Ø. You may omit these. Then in lines 175 and 19Ø you should replace

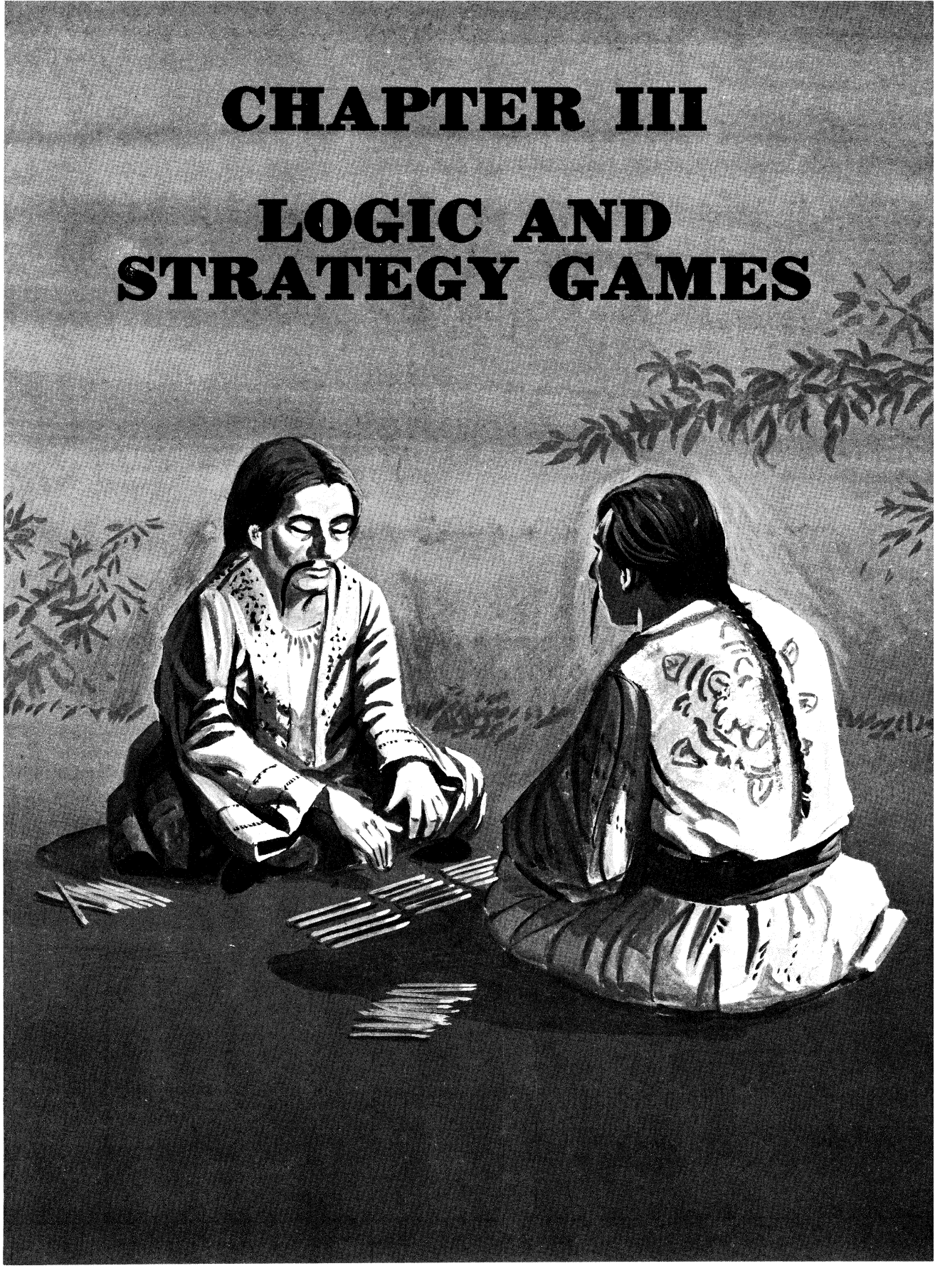
```
GOTO 9Ø1Ø
```

with

```
GOTO 14Ø
```

CHAPTER III

LOGIC AND STRATEGY GAMES



"Invader"

```
10 PRINT "□":PRINT:PRINT"INSTRUCTIONS? P
RESS Y OR N":PRINT
20 GET A$:IF A$="" THEN 20
30 IF A$<>"Y" THEN 100
40 PRINT "SUBMARINES ARE HIDDEN IN A 9 B
Y 9 SQUARE"
50 PRINT "THERE IS ONE SUBMARINE IN EVER
Y ROW":PRINT
60 PRINT "YOU MUST FIND THE SUBS IN 40 T
RIES"
70 PRINT:INPUT "PRESS RETURN TO START";
100 PRINT "□"
105 LET H=0
106 LET T=0
110 FOR I=1 TO 9: LET A$(I)="": LET A(I)
=INT(9*RND(0))+1
120 FOR J=1 TO 9: LET A$(I)=A$(I)+" ."
130 NEXT J,I
140 PRINT "□";TAB(3)
150 FOR I=1 TO 9: PRINT STR$(I);:NEXT I
160 PRINT
170 FOR I=1 TO 9:PRINT: PRINT I;A$(I):NE
XT I
180 IF H=9 THEN 270
185 IF T=40 THEN 300
186 PRINT "ENTER -1 TO QUIT MOVE=";T+1
190 INPUT "ENTER ROW NUMBER";I
200 IF I=-1 THEN 400
210 INPUT "ENTER COLUMN NUMBER";J
215 IF J=-1 THEN 400
220 IF I<1 OR J<1 OR I>9 OR J>9 THEN 140
225 LET T=T+1
230 IF MID$(A$(I),2*J,1)="●" THEN 140
235 IF A(I)=J THEN LET H=H+1
240 IF A(I)=J THEN LET S$=" ●"
245 IF A(I)<>J THEN LET S$=" *"
250 LET A$(I)=LEFT$(A$(I),2*J-2)+S$+RIGH
T$(A$(I),18-2*J)
260 GOTO 140
270 PRINT "YOU WIN IN";T;"TRIES"
```

```
280 GOTO 310
300 PRINT "HA, I WIN"
305 FOR T=1 TO 2000:NEXT:GOTO 400
310 PRINT "AGAIN? PRESS Y OR N"
320 GET Z$:IF Z$="" THEN 320
330 IF Z$="Y" THEN 100
340 STOP
400 REM DISPLAYS ALL THE SUBMARINES
405 PRINT "□":PRINT TAB(3);"THE SOLUTION
 IS"
410 PRINT TAB(3)
420 FOR I=1 TO 9:PRINT STR$(I);:NEXT I
430 PRINT
440 FOR I=1 TO 9:PRINT:PRINT I;
450 FOR S=1 TO 9:IF A(I)=S THEN PRINT "
●";:GOTO 470
460 PRINT " .";
470 NEXT S:PRINT:NEXT I
480 PRINT:INPUT "PRESS RETURN TO GO ON";
490 GOTO 310
```

Note: ● is the character obtained by pressing the SHIFT and Q keys together. The symbol □ represents SHIFT CLEAR/HOME.

"Invader"

AIM OF THE GAME

If you think that the Swedish Navy is inept at catching those small submarines that creep into its territorial waters and seem to escape at will, then here is your chance to see if you could do any better!

This game sets up a 9 by 9 array of points. The points are displayed on the screen in 9 rows and 9 columns. Each row is numbered, and each column is also numbered. There are 9 invading submarines located one to each row. It is your task to try to locate all the submarines with just 4Ø moves.

You can try to develop your own search patterns, or you can search randomly. In any case, you will learn something about the behavior of the random number function, RND.

HOW TO PLAY

When you run this program, you are confronted with an array of dots on the screen arranged in nine numbered rows and columns and a prompt.

```
     1   2   3   4   5   6   7   8   9
1    .   .   .   .   .   .   .   .   .
2    .   .   .   .   .   .   .   .   .
3    .   .   .   .   .   .   .   .   .
4    .   .   .   .   .   .   .   .   .
5    .   .   .   .   .   .   .   .   .
6    .   .   .   .   .   .   .   .   .
7    .   .   .   .   .   .   .   .   .
8    .   .   .   .   .   .   .   .   .
9    .   .   .   .   .   .   .   .   .
ENTER -1 TO QUIT MOVE = 1
ENTER ROW NUMBER?
ENTER COLUMN NUMBER?
```

You enter a row number, I, which should be an integer from 1 to 9. Then you enter a column number, J, also from 1 to 9. Your entries are checked in line 22Ø. If they are not in the range from 1 to 9 then the array and prompt are displayed again.

Suppose you enter 5 and then 8. Instantly, the point at the location row 5 and column 8 on the screen is replaced by * or ●. The * indicates that a submarine was not located at that point, while ● indicates that you have located a submarine.

A count of your attempts is kept by the variable T (for tries). You only have 4Ø tries in a game. A count of your successes is kept by the variable H (for hits). If H < 9 when T = 4Ø, then you have not been able to locate all the submarines in 4Ø tries and you have lost the game. The submarines do not move, so if you repeat the entry of a point, it is counted as a try, but it is a wasted move.

There are very sophisticated search methods programmed into the computers onboard the subchasers. These are based upon statistical theory. However, when all is said and done, if you are lucky, or have a sixth sense for this sort of thing, you will beat the computer every time.

To quit at any time, enter –1 when prompted to enter the row number or the column number. The position of the submarines will then be displayed.

THE MATHEMATICS

This game is meant to give you some experience with probability and the random number function. You start by setting up a simple search pattern, say test 1,5 then 2,5 and so on down the fifth column to 9,5. Then quarter the set of points by testing the 5th row, or another row if the point 5,5 has already located a submarine. When a submarine is found on a row, there is no need to test any more points on that row. Odd and even columns should have the same probability of containing a submarine. Similarly, columns less than the 5th should be chosen with the same frequency as columns greater than the 5th. Finally, as you get down to the last few submarines, you should check those columns that have not revealed a submarine.

However, after saying all this, in one game anything is possible! You might try a simple pattern of testing each row sequentially. That is, start at 1,1 then 1,2 and so on until you find the submarine in that row. Then you start on the second row. Since there are nine positions on each row, there is a better than 5Ø% chance of locating the submarine in the first 5 positions of each row. In fact, the probability is $\frac{5}{9}$, in other words, $55\frac{5}{9}$ %. Overall, this would amount to 45 tries for the 9 rows. What is the probability of successfully locating all the submarines in 45 tries by this method?

You could use this game as a research tool to investigate the answer to this question, but you would need a lot of patience! You would need to play many times, a hundred would do for a rough estimate. You would record the number of times this search pattern was successful. You would need to change line 185 to

```
185 IF T = 45 THEN 3ØØ
```

For those of you who are like me (a bit lazy), you could change the program so that the computer does the search for you. The following is a simple program that does this.

"Probability"

```
11Ø PRINT "□"
12Ø PRINT "ENTER NUMBER OF TESTS"
13Ø INPUT N
14Ø LET S=Ø
15Ø FOR K=1 TO N
16Ø LET H=Ø
17Ø FOR I=1 TO 9
18Ø LET H=H+INT(9*RND(Ø))+1
19Ø NEXT I
2ØØ IF H<46 THEN LET S=S+1
21Ø NEXT K
215 PRINT
22Ø PRINT "NUMBER OF WINS IN";N
225 PRINT "GAMES IS";S
23Ø PRINT "PROBABILITY IS";1ØØ*S/N;"%"
235 PRINT
24Ø PRINT "ANOTHER EXPERIMENT?"
245 PRINT "PRESS Y OR N"
25Ø GET A$:IF A$="" THEN 25Ø
26Ø IF A$="Y" THEN GOTO 11Ø
27Ø STOP
```

Note: The symbol □ represents SHIFT CLEAR/HOME.

"Probability"

When you run this program, you are prompted by the message

ENTER NUMBER OF TESTS

It takes about 30 seconds to run 100 tests, so be prepared for a long wait if you enter a large number. Usually 100 tests will give you a fairly accurate estimate of the probabilities.

I hope the answer you obtain agrees with the answer you worked out for yourself. If it does, pat yourself on the back and go raid the freezer for some of your favorite ice cream. If it does not agree with your answer, what the heck, go have some ice cream anyway!

Now I have a more difficult question for you. Suppose you only tried the first 5 positions on each row. That is, whether you were successful or not, you check out the 45 positions which are the first five positions in each row. What is the probability of finding all 9 submarines by this method?

Make the following alterations to "PROBABILITY" to conduct this experiment.

```
180 IF 6>INT(9*RND)+1 THEN LET H=H+1
200 IF H=9 THEN LET S=S+1
```

You will find it necessary to increase the number of tests to more than 100. By changing H=9 in line 50, I would suggest that you check the probability of finding 8 submarines by this method, and then 7, 6 and so on down to 0.

You should plot the results on a graph. If you change the grid dimension from 9 by 9 to 8 by 8, and search the first four positions in each row, these results would be symmetrical. That is, the probability of finding all 8 would be the same as finding none. These probabilities are called Binomial. You would study these and other probability distributions in an elementary course on probability and statistics. Your results should reproduce approximations to the following:

$$(4/9 + 5/9)^9$$

$$= (4/9)^9 + 9(4/9)^8(5/9) + 36(4/9)^7(5/9)^2 + 84(4/9)^6(5/9)^3$$

$$+ 126(4/9)^5(5/9)^4 + 126(4/9)^4(5/9)^5 + 84(4/9)^3(5/9)^6$$

$$+ 36(4/9)^2(5/9)^7 + 9(4/9)(5/9)^8 + (5/9)^9$$

This is a binomial expansion. If the probability of finding a submarine in the first five positions on a row is $5/9$ then the probability of not finding a

submarine must be $\frac{4}{9}$ (they always add up to 1). Since there are nine rows, we multiply ($\frac{4}{9}$ + $\frac{5}{9}$) by itself 9 times. The terms in this expansion all add up to 1 and they represent the probabilities of finding Ø, 1, 2, ..., 9 submarines respectively. Consequently, the probability of finding all nine submarines is $(\frac{5}{9})^9$ = .ØØ5 (approximately) or ½ %. So it is a very poor search pattern!

The numbers that multiply the powers in the expression are called the binomial coefficients. An easy way to find these numbers is to use Pascal's triangle.

```
                    1
                  1   1
                1   2   1
              1   3   3   1
            1   4   6   4   1
          1   5  1Ø  1Ø  5   1
        1   6  15  2Ø  15  6   1
      1   7  21  35  35  21  7   1
    1   8  28  56  7Ø  56  28  8   1
  1   9  36  84 126 126 84  36  9   1
```

Start and end each row with 1. Add the two numbers above for each entry in a row. Entries in the nth row are binomial coefficients for (a+b)n–1.

I hope my explanation has not caused you to do anything rash, like run away to join the Swedish Navy!

"Sea Hunt"

```
10 PRINT "□":PRINT:PRINT "INSTRUCTIONS?
PRESS Y OR N":PRINT
20 GET A$:IF A$="" THEN 20
30 IF A$<>"Y" THEN 100
40 PRINT "SHIPS OF SIZES 2,3,4 & 5 ARE H
IDDEN BY A"
50 PRINT "9 BY 9 SQUARE OF DOTS (.) YOU
HAVE UP TO"
60 PRINT "40 MOVES TO LOCATE THE SHIPS B
Y LOOKING "
70 PRINT "AT WHAT IS BEHIND EACH DOT"
80 PRINT:INPUT "PRESS RETURN TO START";
100 GOSUB 510
105 LET H=0
106 LET T=0
110 FOR I=1 TO 9: LET A$(I)=""
120 FOR J=1 TO 9: LET A$(I)=A$(I)+" ."
130 NEXT J,I
140 PRINT "□";TAB(3)
150 FOR I=1 TO 9: PRINT STR$(I);:NEXT I
160 PRINT
170 FOR I=1 TO 9:PRINT: PRINT I;A$(I):NE
XT I
180 IF H=14 THEN 270
185 IF T=40 THEN 300
186 PRINT "ENTER -1 TO QUIT MOVE=";T+1
190 INPUT "ENTER ROW NUMBER";I
200 IF I=-1 THEN 400
210 INPUT "ENTER COLUMN NUMBER ";J
215 IF J=-1 THEN 400
220 IF I<1 OR J<1 OR I>9 OR J>9 THEN 140
225 LET T=T+1
230 IF MID$(A$(I),2*J,1)="●" THEN 140
231 LET S$=""
232 FOR K=1 TO 5
240 IF A(I,K)=J THEN LET S$=" ●":LET H=H
+1
245 NEXT K
246 IF S$<>" ●" THEN LET S$=" *"
250 LET A$(I)=LEFT$(A$(I),2*J-2)+S$+RIGH
```

```
T$(A$(I),18-2*J)
260 GOTO 140
270 PRINT "YOU WIN IN";T;"TRIES"
280 GOTO 310
300 PRINT "HA, I WIN":FOR T=1 TO 2000:NE
XT:GOTO 400
310 PRINT:PRINT "AGAIN? PRESS Y OR N"
320 GET Z$:IF Z$="" THEN 320
330 IF Z$="Y" THEN 100
340 STOP
400 REM DISPLAYS ALL THE SHIPS
405 PRINT "□":PRINT TAB(3);"THE SOLUTION
 IS"
410 PRINT TAB(3)
420 FOR I=1 TO 9:PRINT STR$(I);"NEXT I
430 PRINT
440 FOR I=1 TO 9:PRINT:PRINT I;
450 FOR J=1 TO 9:P=0:FOR S=1 TO 5:IF A(I
,S)=J THEN PRINT " ●";
455 IF A(I,S)=J THEN P=1
460 NEXT S:IF P=0 THEN PRINT " .";
470 NEXT J:PRINT:NEXT I
480 PRINT:INPUT "PRESS RETURN TO GO ON";
490 GOTO 310
500 REM THIS SUBROUTINE SETS UP THE SHIP
S UP-DOWN OR ACROSS
510 PRINT "□":PRINT:PRINT
520 PRINT "SETTING UP THE GAME GO HAVE A
 SNACK THIS"
525 PRINT "WILL TAKE A MOMENT"
530 FOR I=1 TO 9
540 FOR J=1 TO 5
550 LET A(I,J)=0:NEXT J,I:REM INITIALIZE
560 FOR S=2 TO 5
565 LET I=INT((10-S)*RND(0))+1:REM START
ING POSITION OF SHIPS
566 LET J=INT(9*RND(0))+1
567 LET D=INT(2*RND(0)):REM SHIPS GO ACR
OSS IF D=0 AND DOWN IF D=1
570 IF D=0 THEN LET A(I,1)=J
575 IF D=1 THEN LET A(J,1)=I
576 FOR K=2 TO S
578 IF D=0 THEN LET A(I+K-1,1)=J
579 IF D=1 THEN LET A(J,K)=I+K-1
```

```
580 NEXT K,S
585 LET S=0
590 FOR I=1 TO 9
595 FOR J=1 TO 5
600 IF A(I,J)>0 THEN LET S=S+1:REM COUNT
S THE SQUARES OCCUPIED BY SHIPS
610 NEXT J,I
620 IF S<14 THEN 530:REM 2+3+4+5=14 IF S
 IS NOT 14 THEN SHIPS CROSS EACH OTHER
630 RETURN
```

Note: ● is the graphics character obtained by pressing the SHIFT and Q keys together. The symbol ◘ represents SHIFT CLEAR/HOME.

"Sea Hunt"

AIM OF THE GAME

This game is similar to "INVADER", but more difficult because there is less information to guide you. Ships of various lengths from 2 to 5 consecutive positions in the grid are hidden in the 9 by 9 grid of points. They can be vertical or horizontal. There is no longer any guarantee that every row will be occupied by a piece of a ship. There are a total of 14 hits to be made in order to win the game, but you are limited to 4Ø tries.

HOW TO PLAY

When you run this game a message is displayed

```
SETTING UP THE GAME GO HAVE A SNACK THIS
WILL TAKE A MOMENT
```

while the ships are being located randomly in the grid. A grid of points is then displayed and you are prompted to enter the row number and column number (I,J) of a grid point. The numbers I,J should be in the range from 1 to 9. An entry of –1 will quit the game and the location of the ships will be displayed. A prompt to play again is then given.

THE CODE

This program is much longer than "INVADER" because setting up the ships is more complicated. The subroutine from lines 5ØØ to 63Ø does this. The edge of the ships are located at a safe distance from the boundaries so they will lie inside the 9 by 9 grid. Then, after the ships have been randomly located, a count of the occupied points is made. If this turns out to be less than 14, then two or more ships must have a grid point in common. This set-up is discarded and another arrangement is made. For this reason the setting up can take some time.

As a programming exercise, see if you can add to the setting up routine so that ships can be located diagonally. You would choose four values for D in line 567. If D=3 then the ship would be located on a diagonal from left upper to right lower, and for D=4 the ship would be located in the opposite diagonal direction.

"Verify"

```
11Ø LET C=1
12Ø LET F=INT(1Ø*RND(Ø))
13Ø PRINT "□"
14Ø PRINT "YOU HAVE 1Ø GOLF BALLS ONE HA
S A":PRINT
15Ø PRINT "DIFFERENT WEIGHT FROM THE OTH
ERS. FIND":PRINT
16Ø PRINT "THE DEFECTIVE BALL BY COMPARI
NG ANY":PRINT
17Ø PRINT "NUMBER OF BALLS ON A BALANCE
."
18Ø PRINT
19Ø PRINT "ENTER BALL NUMBERS Ø12...9 IN
 A STRING"
195 PRINT:PRINT "FOR EXAMPLE,TO COMPARE
BALLS Ø,4,5,7 AND"
196 PRINT "9,ENTER THE STRING 'Ø4579'"
2ØØ PRINT:INPUT A$
21Ø FOR I=1 TO LEN(A$)
215 B$=MID$(A$,I,1):IF B$<"Ø" OR B$>"9"
THEN 13Ø:REM CHECKS FOR NUMBERS
22Ø IF VAL(MID$(A$,I,1))=F THEN 26Ø
23Ø NEXT I
24Ø PRINT "EQUAL"
25Ø GOTO 28Ø
26Ø PRINT "UNEQUAL"
27Ø PRINT
28Ø PRINT "GIVE UP? PRESS Y OR N"
29Ø GET B$:IF B$="" THEN 29Ø
3ØØ PRINT
31Ø IF B$="Y" THEN PRINT "YOU HAD";C;"MO
VES. THE BALL WAS?":GOTO 34Ø
32Ø LET C=C+1
33Ø IF B$="N" THEN 13Ø
34Ø INPUT Z
35Ø IF Z=F THEN PRINT "CORRECT"
36Ø IF Z<>F THEN PRINT "WRONG. IT WAS";F
37Ø PRINT "AGAIN? PRESS Y OR N"
38Ø GET Z$:IF Z$="" THEN 38Ø
```

```
390 IF Z$="Y" THEN 110
400 STOP
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME.

"Verify"

AIM OF THE GAME

This game is similar to "LOOSE CHANGE", but it is much easier. You should master this game before playing "LOOSE CHANGE".

You have 1Ø golf balls and a scale. One of the balls is defective as it does not weigh the same as the others. The balls are numbered Ø, 1, 2,..., 9. You may select any number of balls to compare their weight. You can even select an odd number because the computer has a spare good ball to make up the same number on each side of the scales.

Each time you compare a set of balls you are told whether they are equal or unequal. The number of moves you take are counted. If you can find the defective ball in 3 moves or less that is an excellent score. A good score is 4 or 5 moves. More than 5 moves and you need to practice a few times. Maybe you would prefer to play with real golf balls on a golf course!

HOW TO PLAY

When we run this program it commences with the message

```
YOU HAVE 1Ø GOLF BALLS ONE HAS A
DIFFERENT WEIGHT FROM THE OTHERS.FIND
THE DEFECTIVE BALL BY COMPARING ANY
NUMBER OF BALLS ON A BALANCE.
ENTER BALL NUMBERS Ø12...9 IN A STRING
FOR EXAMPLE,TO COMPARE BALLS Ø,4,5,7 AND
9,ENTER THE STRING 'Ø4579'
```

Suppose we enter the string 12345. In this case we receive the message

```
EQUAL
GIVE UP? PRESS Y OR N
```

We are not ready to quit so we press the key N. The first message is repeated. This time we might enter the string 678. Again we received the message that they are equal. We press N and enter Ø. This turned out to be OK so now we know that 9 is the defective ball. This time we press the key Y in response to

```
GIVE UP? PRESS Y OR N
```

We obtain the message

```
YOU HAD 3 MOVES. THE BALL WAS?
```

We enter 9 and obtain the response

```
CORRECT
AGAIN? PRESS Y OR N
```

If you wish to quit, press the key N. Otherwise, press the key Y and a new game will be generated.

"Loose Change"

```
11Ø LET C=1
115 LET N=INT(5*RND(Ø))+1
12Ø LET F=INT(12*RND(Ø))+ASC("A")
13Ø PRINT "□"
135 LET B$=""
14Ø PRINT:PRINT "ONE OF 12 DIMES HAS A D
IFFERENT WEIGHT"
145 PRINT:PRINT "THAN THE REST.FIND THE
ODD DIME BY "
15Ø PRINT:PRINT "WEIGHING";N;"AGAINST";N
;"."
155 PRINT:PRINT "WE REPRESENT THE COINS
BY LETTERS A TO L"
16Ø PRINT "TYPE THE COIN LETTERS A,B,C,.
..,L THAT"
165 PRINT:PRINT "YOU WISH TO COMPARE":PR
INT
17Ø LET I=Ø
18Ø LET W=Ø
19Ø GET A$:IF A$="" THEN 19Ø
195 IF A$<"A" OR A$>"L" THEN 19Ø
2ØØ IF A$=CHR$(F) THEN LET W=1
2Ø5 FOR J=1 TO I:IF A$=MID$(B$,J,1) THEN
 19Ø
2Ø6 NEXT J
2Ø7 LET B$=B$+A$
21Ø PRINT A$;
22Ø LET I=I+1
23Ø IF I<2*N THEN 19Ø
24Ø PRINT "□"
25Ø IF W=1 THEN PRINT "UNEQUAL"
26Ø IF W=Ø THEN PRINT "EQUAL"
27Ø PRINT "GIVE UP? PRESS Y OR N"
28Ø GET A$:IF A$="" THEN 28Ø
29Ø IF A$="N" THEN 35Ø
3ØØ PRINT:PRINT "YOU HAD";C;"MOVES. THE
COIN WAS?"
31Ø INPUT B$
32Ø IF ASC(B$)=F THEN PRINT "CORRECT"
33Ø IF ASC(B$)<>F THEN PRINT "WRONG! THE
```

```
 COIN IS ";CHR$(F)
34Ø GOTO 37Ø
35Ø LET C=C+1
36Ø GOTO 13Ø
37Ø PRINT:PRINT "AGAIN? PRESS Y OR N"
38Ø GET A$:IF A$="" THEN 38Ø
39Ø IF A$="Y" THEN 11Ø
4ØØ STOP
```

Note: In lines 135, 19Ø, 28Ø and 38Ø, "" are two quotes that define an empty string. The symbol ◘ represents SHIFT CLEAR/HOME.

"Loose Change"

AIM OF THE GAME

This is a game of strategy and skill. Someone has been making counterfeit dimes. "Why bother?", you might ask. That is not the question we are concerned with. You have 12 dimes, one of which is counterfeit. An illegal dime does not weigh the same as the genuine article. By comparing the weight of a chosen number of coins, you must determine which coin is the counterfeit in the fewest number of moves.

The coins are represented by the letters A,B,...,L. You are prompted to enter letters that represent coins by pressing the appropriate keys. The number of coins used in the comparison is chosen randomly. This means that you will have to develop a different strategy each time.

HOW TO PLAY

When you run this program, you are given a message like this

```
ONE OF 12 DIMES HAS A DIFFERENT WEIGHT
THAN THE REST. FIND THE ODD DIME BY
WEIGHING 3 AGAINST 3
WE REPRESENT THE COINS BY LETTERS A TO L
TYPE THE COIN LETTER A,B,C...,L THAT
YOU WISH TO COMPARE
```

The program makes your keyboard act like a typewriter. The first three letters you press would be the coins on one side of the scale. The last three letters represent the coins on the other side. Line 195 ignores any keys that are not one of the letters A to L and line 2Ø5 ignores any repetitions of the same letters. After you have entered a sufficient number of letters, 6 in this case, you will be given a message

```
UNEQUAL (or EQUAL as the case may be)
GIVE UP? PRESS Y OR N
```

If the fake coin was in your selection, then you would get the UNEQUAL message. You now press N to continue or Y to stop. Suppose we entered the letters A,B,C,D,E, and F and were told they were unequal. We would press N and when the prompt for letters appeared, we might enter the letters A,B,C,G,H and I. Suppose we now get the message

```
EQUAL
```

which means that one of D,E or F is the false coin. We would press N and enter the letters A,B,C,D,E and G. Suppose these turned out to be unequal, so then we would try A,B,C,D,G and H. If these turned out to be

equal, we would know that E is counterfeit. This time we would press Y and be given the message

```
YOU HAD 4 MOVES.THE COIN WAS?
```

We would enter E and the program would respond with

```
CORRECT
AGAIN? PRESS Y OR N
```

If you enter an incorrect guess, you would be told that you were wrong and the correct answer would be given.

THE MATHEMATICS AND CODE

If you are given the additional information of which way the scale tips, it is possible to design a scheme that guarantees locating the counterfeit coin in a minimum number of weighings, and in addition, you will know if it is heavy or light.

You might like to alter this program so that the information about which way the scale tips is given to you. The following are the steps you should take to alter the program, however, I will leave the actual alterations to you.

1) Choose H=Ø for light or 1 for heavy in line 125.
2) When checking each letter in line 2ØØ you will need to keep track of which side of the scale the entered letter is on. For instance, if I<N, then the letter being entered is on the left-hand side of the scale. You will need two lines in place of line 2ØØ. The first one, for example, could be

   ```
   2ØØ IF A$=CHR$(F) AND I<N THEN LET W=1
   ```

   Use the values of W to record the presence of the counterfeit coin (W>Ø), and whether the counterfeit coin is on the left (W=1) or the right (W=2).
3) Now the message should be altered according to which side of the scale the counterfeit coin is on, and whether it is heavy or light. You should replace lines 25Ø and 26Ø. The information you give will be one of the following:
   (a) EQUAL
   (b) UNEQUAL:LEFT SIDE DOWN
   (c) UNEQUAL:LEFT SIDE UP
   You do not say on which side the counterfeit is located, or whether it is heavy or light. Everything must be deduced from the meager information given in (a), (b) or (c).
4) In entering the answer you should first check that the coin has been correctly identified, and then ask if it is heavy or light. You

will need to include some extra lines between lines 31Ø and 35Ø. For instance

```
32Ø IF ASC(B$)=F THEN PRINT "CORRECT":GO
TO 335
325 PRINT "COIN IS ";CHR$(F);" IT IS ";
327 IF H=Ø THEN PRINT "LIGHT"
33Ø IF H=1 THEN PRINT "HEAVY"
333 GOTO 37Ø
335 PRINT "IS IT LIGHT OR HEAVY? ENTER Ø
 OR 1"
337 INPUT Z
34Ø IF Z=H THEN PRINT "CORRECT":GOTO 37Ø
345 PRINT "WRONG IT IS ";:GOTO 327
```

As a boy of 15 I can remember solving the problem in the case of comparing four coins against four. The teachers at my school were unable to solve it, so I spent the weekend working on it and was so excited when I found the solution. My family had to patiently sit through several sessions while I rehearsed the presentation I intended to give to the teachers in their coffee room before school started on Monday morning. You can imagine how I felt when I was told by the faculty that my mathematics teacher had found the solution and had already bored the entire staff to tears with his explanations before I had arrived at school! The solution, by the way, required only three weighings in this case. Can you design such a scheme?

"Nim I"

```
100 PRINT "□"
110 LET C=INT (10*RND(0))
120 LET B=C+INT (10*RND(0))
130 LET A=B+INT (10*RND(0))
150 PRINT "□"
160 PRINT A;TAB(3);B;TAB(6);C
165 PRINT
170 PRINT "YOUR MOVE"
180 PRINT " ENTER THE STACKS IN DECREAS
ING ORDER"
190 INPUT A,B,C
200 IF A+B+C=1 THEN 400
210 IF B=0 THEN 420
220 IF C=0 THEN 280
230 IF A=B THEN LET C=0
240 IF B=C AND C=1 THEN 340
250 IF B=C AND C<>1 THEN LET A=0
260 IF A>0 AND C>0 THEN 330
270 GOTO 150
280 IF A>B THEN 310
290 LET B=B-1
300 GOTO 150
310 IF B=1 OR A=1 THEN GOTO 420
315 LET A=B
320 GOTO 150
330 IF A+B+C-6>0 THEN 360
340 LET A=A-1
350 GOTO 150
360 LET S=B+C
370 IF A-6+S>0 AND S>2 AND S<6 THEN LET
A=6-S
380 IF A+B+C=6 THEN 150
390 GOTO 340
400 PRINT "YOU WIN"
410 GOTO 430
420 PRINT "I WIN"
430 PRINT: PRINT "AGAIN? PRESS Y OR N"
440 GET A$:IF A$="" THEN 440
```

```
45Ø IF A$="Y" THEN 11Ø
46Ø STOP
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME.

"Nim II"

```
1Ø PRINT "□":PRINT:PRINT "INSTRUCTIONS?
PRESS Y OR N":PRINT
2Ø GET A$:IF A$="" THEN 2Ø
3Ø IF A$<>"Y" THEN 1ØØ
4Ø PRINT "YOU HAVE THREE STACKS OF COUNT
ERS LISTED"
5Ø PRINT "IN DECREASING ORDER OF MAGNITU
DE.ON YOUR"
6Ø PRINT "MOVE YOU MAY TAKE COUNTERS FRO
M ONE PILE"
7Ø PRINT "YOU MAY TAKE ANYTHING FROM ONE
 TO ALL OF"
8Ø PRINT "THE COUNTERS FROM THAT ONE PIL
E"
9Ø PRINT:INPUT "PRESS RETURN TO START";
1ØØ PRINT "□"
11Ø LET C=INT(1Ø*RND(Ø))
12Ø LET B=C+INT(1Ø*RND(Ø))
13Ø LET A=B+INT(1Ø*RND(Ø))
15Ø PRINT "□"
16Ø PRINT A;TAB(3);B;TAB(6);C
165 PRINT
17Ø PRINT "YOUR MOVE":PRINT
18Ø PRINT " ENTER THE STACKS IN DECREAS
ING ORDER"
182 INPUT X,Y,Z
183 IF(X+Y+Z)>=(A+B+C) THEN 15Ø
184 IF (X<>A AND X<>B AND X<>C) AND (Y<>
A AND Y<>B AND Y<>C) THEN 15Ø
186 IF (X<>A AND X<>B AND X<>C) AND (Z<>
A AND Z<>B AND Z<>C) THEN 15Ø
188 IF (Y<>A AND Y<>B AND Y<>C) AND (Z<>
A AND Z<>B AND Z<>C) THEN 15Ø
189 IF X<Y OR X<Z OR Y<Z THEN 15Ø
19Ø LET A=X
192 LET B=Y
194 LET C=Z
2ØØ IF A+B+C=1 THEN 4ØØ
21Ø IF B=Ø THEN 42Ø
22Ø IF C=Ø THEN 28Ø
```

```
230 IF A=B THEN LET C=0
240 IF B=C AND C=1 THEN 340
250 IF B=C AND C<>1 THEN LET A=0
260 IF A>0 AND C>0 THEN 330
270 GOTO 150
280 IF A>B THEN 310
290 LET B=B-1
300 GOTO 150
310 IF B=1 OR A=1 THEN GOTO 420
315 LET A=B
320 GOTO 150
330 IF A+B+C-6>0 THEN 360
340 LET A=A-1
350 GOTO 150
360 LET S=B+C
370 IF A-6+S>0 AND S>2 AND S<6 THEN LET
A=6-S
380 IF A+B+C=6 THEN 150
390 GOTO 340
400 PRINT "YOU WIN"
410 GOTO 430
420 PRINT "I WIN"
430 PRINT: PRINT "AGAIN? PRESS Y OR N"
440 GET A$:IF A$="" THEN 440
450 IF A$="Y" THEN 110
460 STOP
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME.

"Nim I" and "Nim II"

AIM OF THE GAME

NIM is an ancient Chinese game for two players. Counters are arranged in stacks between the two players. There can be any number of stacks and any number of counters in each stack. When it is your turn you may take any number of counters from any one stack. The aim of the game is to force your opponent to take the last counter. That is, all the stacks are empty, except one that has only one counter in it.

There is a winning strategy for the player who goes first. In the simple versions of NIM presented here in "NIM I" and "NIM II", the program always lets you start.

HOW TO PLAY

There are only three stacks of counters. The number of counters in each stack is chosen randomly and will be from Ø to a maximum of 27. These numbers are displayed in decreasing order of magnitude.

In this simple version of NIM, you are prompted to enter your move. You will enter the number of counters left in the stacks after your move, one at a time, in decreasing order. The program will make dumb moves if you do not enter the numbers in decreasing order.

Let us suppose that you get the message

```
12  4  4
YOUR MOVE
ENTER THE STACKS IN DECREASING ORDER
```

Suppose you enter 4, 4 and then 4, then the response would be

```
4  4  Ø
```

and you would be prompted to enter your move again. Suppose you now enter 4, 2 and then Ø. The response would be

```
2  2  Ø
```

Finally, you would enter 2, 1 and then Ø and the computer would win because it would remove all the counters from the first stack leaving you the one counter in the second stack. The response would be:

```
I WIN
AGAIN? PRESS Y OR N
```

If you wish to quit you would press the key N or any key other than Y. By pressing the key Y you would commence a new game.

These short programs are meant to provide you with a challenging game, but not a perfect game, so you should be able to beat your computer! Version I is very short and easy to input. It is easy for you to cheat because your move is not checked and you can study the computer's response to any set up. In this way you will be able to learn some strategies. There are two winning strategies that the program does not check. It may not take you long to find them. On the other hand. . . .

Version II does not play a superior game, but it will not allow you to cheat. In this version your entries are checked to make sure that you adhere to the rules of the game.

THE CODE

Since I have not come across a program that plays a perfect game of "NIM", I decided to try to write one. The listing "NIM III" does, I believe, play a perfect game. However, a law of computing is that there is no such thing as a perfect program. All I can say is that this program seems to play a perfect game in the tests conducted at our home.

If you enter this program you will appreciate why I have tried to keep all the programs in this book as short as possible. When you type in the listing, I suggest you save your listing from time to time, in order to protect your work in case of a power failure.

"Nim III"

```
100 PRINT "□":PRINT:PRINT
110 REM: SET UP THE GAME
120 GOSUB 300
130 IF M$="N" THEN 160
140 REM: YOUR MOVE
150 GOSUB 400
160 REM: COMPUTER'S MOVE
170 PRINT "I AM THINKING"
180 GOSUB 500
190 REM: CHECK SPECIAL CASES
200 IF VAL(S$)=1 THEN PRINT:PRINT "YOU W
IN!":GOTO 270
210 GOSUB 600
220 REM: DISPLAY STACKS
230 GOSUB 950
240 IF Z=1 THEN PRINT:PRINT "I WIN, HURR
AH!"
250 IF Z<=1 THEN 270
260 GOTO 140
270 PRINT "AGAIN? PRESS Y OR N"
280 GET Z$:IF Z$="" THEN 280
290 IF Z$="Y" THEN 110
295 STOP
300 PRINT "DO YOU WANT THE FIRST MOVE?P
RESS Y OR N"
305 GET M$:IF M$="" THEN 305
310 PRINT:PRINT "ENTER NUMBER OF STACKS
FROM 3 TO 9"
315 GET S$:IF S$="" THEN 315
320 LET S=VAL(S$)
325 IF S<3 OR S>9 THEN 310
330 PRINT:PRINT "DO YOU WANT TO CHOOSE T
HE NUMBER OF"
335 PRINT "COUNTERS IN THE STACKS?"
340 PRINT:PRINT "PRESS Y OR N"
345 GET S$:IF S$="" THEN 345
350 PRINT "□"
355 FOR I=1 TO S
360 LET A(I)=INT(15*RND(0))+1
365 PRINT "STACK";I;"=";
```

```
370 IF S$="Y" THEN INPUT A(I)
375 IF A(I)<Ø THEN 365
380 IF S$<>"Y" THEN PRINT A(I)
385 NEXT I
390 RETURN
400 PRINT:PRINT "WHICH OF THE STACKS DO
YOU WISH TO"
405 PRINT "ALTER? ENTER 1 TO";S
410 INPUT MS
415 IF MS<1 OR MS>S THEN 400
420 IF A(MS)=Ø THEN 400
425 PRINT:PRINT "HOW MANY COUNTERS ARE Y
OU REMOVING?"
430 PRINT "ENTER 1 TO";A(MS)
435 INPUT MM
440 IF A(MS)<MM OR MM<1 THEN 425
445 LET A(MS)=A(MS)-MM
450 RETURN
500 REM: CONVERT STACKS TO BINARY
505 LET S$="Ø"
510 FOR J=1 TO S
515 LET B$(J)=""
520 LET T=A(J)
525 LET I=1
530 IF LEN(S$)<I THEN LET S$="Ø"+S$
535 LET LS=LEN(S$)
540 LET C=A(J)-2*INT(A(J)/2)
545 LET A(J)=INT(A(J)/2)
550 LET B$(J)=RIGHT$(STR$(C),1)+B$(J)
555 LET M$=STR$(VAL(MID$(S$,LS+1-I,1))+C
)
560 LET S$=LEFT$(S$,LS-I)+RIGHT$(M$,1)+R
IGHT$(S$,I-1)
565 LET I=I+1
570 IF A(J)>Ø THEN 530
575 LET A(J)=T
580 NEXT J
581 REM: FILL ALL B$(J) TO THE LENGTH LS
582 LET F$="ØØØØØØØØØØØØØØØØØØØØØØØØØØ
ØØØØØØ"
583 FOR J=1 TO S:LET B$(J)=RIGHT$(F$,LS-
LEN(B$(J)))+B$(J):NEXT J
585 RETURN
600 LET Q=Ø
```

```
605 LET R=0
610 LET P=0
615 LET R=0
617 LET W=0
620 FOR J=1 TO S
625 IF A(J)=1 THEN LET Q=J
630 IF A(J)=1 THEN LET W=W+1
635 IF A(J)>0 THEN LET P=P+1
640 IF A(J)>1 THEN LET R=J
645 NEXT J
650 IF P=1 THEN LET A(R)=1
655 IF P=2 AND W<>1 THEN GOSUB 700
660 IF P=2 AND W=1 THEN LET A(R)=0
665 IF P>2 AND P=(W+1) THEN GOSUB 680
670 IF P>2 AND P<>(W+1) THEN GOSUB 700
675 RETURN
680 IF P-2*INT(P/2)=0 THEN LET A(R)=0
685 IF P-2*INT(P/2)<>0 THEN LET A(R)=1
690 RETURN
700 REM: POSSIBLE SUBTRACTION
705 LET D=0
710 FOR I=1 TO LS
715 LET Y=VAL(MID$(S$,I,1)):IF Y-2*INT(Y
/2)>0 THEN LET D=D+2↑(LS-I)
720 NEXT I
725 REM: IF D>0 THEN COMPUTER CAN MAKE A
 PERFECT MOVE, OTHERWISE IT CAN NOT.
730 IF D>0 THEN 750
735 IF Q>0 THEN LET A(Q)=A(Q)-1
740 IF Q=0 AND R>0 THEN LET A(R)=A(R)-1
745 RETURN
750 REM: IF ONE STACK HAS THE MAXIMUM, R
EMOVE ALL THE COUNTERS.
755 FOR I=1 TO S
760 IF A(I)=D THEN LET A(I)=0:RETURN
765 NEXT I
770 REM: ADJUST A SUITABLE STACK.
775 FOR I=1 TO S
780 LET SUB=0
785 FOR J=1 TO LS
790 LET Y=VAL(MID$(S$,J,1)):IF Y-2*INT(Y
/2)=0 THEN 830
800 LET Y=VAL(MID$(B$(I),J,1)):IF Y-2*IN
T(Y/2)=0 THEN 825
```

```
810 LET SUB=SUB+2↑(LS-J)
820 GOTO 830
825 LET SUB=SUB-2↑(LS-J)
830 NEXT J
835 IF SUB>0 THEN 845
840 NEXT I
845 LET A(I)=A(I)-SUB
850 RETURN
950 PRINT "□"
955 LET Z=0
960 FOR I=1 TO S
965 PRINT "STACK";I;"=";A(I)
970 LET Z=Z+A(I)
975 NEXT I
980 RETURN
```

Note: The symbol □ represents SHIFT CLEAR/HOME.

"Nim III"

HOW TO PLAY

In this version there are up to 9 stacks of counters. There are many prompts given so that you can design the game to meet your requirements for difficulty. A typical run would proceed as follows.

```
DO YOU WANT THE FIRST MOVE? PRESS Y OR N
```

If you want the computer to have the first move you would press N. If you press any other key, you will have the first move. You have to make this choice before you see the stacks, so don't give the computer the first move if you want to win! The computer is absolutely ruthless in this game and plays to win. Only press N if you wish to study how the computer reacts to a certain given situation.

The next prompt is

```
ENTER NUMBER OF STACKS FROM 3 TO 9
```

All you need to do is press a key from 3 to 9. Any other key will not be accepted and the prompt will be repeated. Suppose you press the key 3. The next prompt appears.

```
DO YOU WANT TO CHOOSE THE NUMBER OF
COUNTERS IN THE STACKS?
PRESS Y OR N
```

If you press the key Y you get a series of prompts like

```
STACK 1 = ?
```

You enter a number that can be as large as you like. I would suggest that you keep it to less than 1ØØ for the sake of keeping the game short. If you press the key N, the number of counters are chosen for you and are kept to 15 or less. You could change this by altering line 36Ø.

Suppose you had pressed the key N. Then the situation might be something like this

```
STACK 1 = 8
STACK 2 = 15
STACK 3 = 9
WHICH OF THE STACKS DO YOU WISH TO
ALTER? ENTER 1 TO 3
```

Suppose you enter 2. Then the next prompt would be

```
HOW MANY COUNTERS ARE YOU REMOVING?
ENTER 1 TO 15
```

Suppose you enter 1Ø, then the computer would respond with

```
I AM THINKING
```

After a short period of time the stacks would be listed again and you would be prompted to make your move.

Finally, you get a message of victory or defeat and are asked if you wish to play another game.

THE MATHEMATICS AND CODE

The key to a winning strategy in this game is to convert the number of counters in each stack to binary. Then you add up all these numbers, using the usual decimal addition. The idea is to make a move that leaves an even number in every position in the sum. If that sounds confusing, let me show you with an example.

Suppose you had three stacks with 8, 15 and 9 counters. Converting to binary notation we would have

```
  8 = 1ØØØ
 15 = 1111
  9 = 1ØØ1
sum = 3112
```

In order to make each of these digits in the sum even, we would need to subtract 1 from the first, 1 from the second and 1 from the third. That is the number 111Ø = 14 should be subtacted. Only 15 is greater than this, so we would subtract 14 from 15 leaving 8, 1 and 9 in the respective stacks.

Sometimes it is not possible to do this because there is no number large enough. For instance

```
  5 = 1Ø1
  4 = 1ØØ
  6 = 11Ø
sum = 311
```

Here we need to subtract 111 = 7, but there is no number this big. The strategy then is to subtract when we can, and add if we cannot subtract. For example, let us adjust the number 5. We subtract 1ØØ = 4 to make the first column add up to an even number. We cannot subtract anything from the second column, so we add 1Ø = 2. Finally, we subtract 1 from the last column. So the adjustment to 5 is –4 + 2 – 1 = –3. This leaves us with

```
  2 = 1Ø
  4 = 1ØØ
  6 = 11Ø
sum = 22Ø
```

This does the trick since Ø is considered even. Similarly, we could have adjusted 4 by subtracting 4 and adding 3 to give

```
    5 = 1Ø1
    3 =  11
    6 = 11Ø
  sum = 222
```

As you can see, there is more than one way to make your move.

There are some special situations that occur, such as each stack having 1 counter, which would have to be handled differently.

The program is fairly complicated to a beginner, so I have used some block structure and remarks to make it easier for you to follow. All the action takes place in subroutines. The main program, which calls the subroutines, is in lines 1ØØ to 295. The subroutine in lines 3ØØ to 39Ø sets up the game. The subroutine in lines 4ØØ to 45Ø handles your move. Converting the stacks to binary and adding up the totals in each column is done in the subroutine from line 5ØØ to 585. Special cases are handled in lines 6ØØ to 69Ø, and the computer's move is done in lines 7ØØ to 85Ø. The display of the stacks is handled in the subroutine from 95Ø to 98Ø.

You might like to think about this question. How would you modify the program so that the object of the game is to take the last counter instead of forcing your opponent to take the last one?

"Towers of Hanoi"

```
11Ø PRINT "□":PRINT "THE NUMBER OF COUNT
ERS IS? ENTER 1 TO 9"
12Ø LET K=Ø
13Ø INPUT N:PRINT "□"
14Ø IF N<1 OR N>9 THEN 11Ø
15Ø LET M=Ø
16Ø LET A$(1)="ABCDEFGJHI"
17Ø LET A$(2)="|||||||||"
18Ø LET A$(3)=A$(2)
19Ø PRINT "□"
2ØØ LET M=M+1
21Ø FOR I=1 TO N
22Ø PRINT MID$(A$(1),I,1);TAB(9);MID$(A$
(2),I,1);TAB(18);MID$(A$(3),I,1)
23Ø NEXT I
24Ø IF K=1 THEN 45Ø
25Ø PRINT:PRINT "ENTER A STRING OF TWO D
IGITS FOR MOVE";M
255 PRINT "FOR EXAMPLE '13' MEANS MOVE T
HE TOPMOST "
26Ø INPUT "COUNTER FROM STACK ONE TO STA
CK THREE";B$
27Ø LET C=VAL(MID$(B$,1,1))
28Ø LET D=VAL(MID$(B$,2,1))
29Ø IF C<1 OR D<1 OR C>3 OR D>3 OR C=D T
HEN 43Ø
3ØØ FOR I=1 TO N
31Ø IF MID$(A$(C),I,1)="|" THEN 42Ø
32Ø FOR J=N TO 1 STEP -1
33Ø IF MID$(A$(D),J,1)<>"|" THEN 38Ø
34Ø LET A$(D)=LEFT$(A$(D),J-1)+MID$(A$(C
),I,1)+RIGHT$(A$(D),9-J)
35Ø IF J=N THEN 39Ø
36Ø IF MID$(A$(D),J,1)>MID$(A$(D),J+1,1)
 THEN 43Ø
37Ø GOTO 39Ø
38Ø NEXT J
39Ø LET A$(C)=LEFT$(A$(C),I-1)+"|"+RIGHT
$(A$(C),9-I)
4ØØ IF I=N AND J=1 AND D=3 THEN LET K=1
```

```
41Ø GOTO 19Ø
42Ø NEXT I
43Ø PRINT:PRINT "THE UNIVERSE IS DOOMED
- INVALID MOVE"
44Ø GOTO 48Ø
45Ø PRINT:PRINT "MOVES TO COMPLETE TASK
IS";M-1
46Ø IF M-1=INT(2↑N-1) THEN PRINT "THE UN
IVERSE IS SAFE, THANKS TO YOU"
47Ø IF M-1>INT(2↑N-1) THEN PRINT "UNFORT
UNATELY YOU HAD TOO MANY MOVES"
48Ø PRINT:PRINT "AGAIN? PRESS Y OR N"
49Ø GET Z$:IF Z$="" THEN 49Ø
5ØØ IF Z$="Y" THEN 11Ø
51Ø STOP
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME and "┼" is the SHIFT G symbol.

"Towers of Hanoi"

AIM OF THE GAME

In a monastery, high in the mountains of Tibet, a group of monks struggle valiantly to save the universe from destruction. Their task, imposed by a superior being from another universe, is to shift 6Ø counters of different magnitudes from one stack to another. They are allowed to form three stacks while doing this. At the beginning, all the counters are on the first stack, arranged in increasing order. Only one counter at a time can be moved. The problem is to move all the counters to the third stack in the least number of moves. There is one vital restriction. At no time should a greater counter be on top of a lesser one. One mistake and the universe, as we know it, ends. So the monks toil on, carefully contemplating each move.

HOW TO PLAY

In this game you get to choose the number of counters (from 1 to 9). The game commences with a prompt requesting you to enter the number of counters. You then get to move the counters, which are the letters A, B, C, ... , initially stacked in increasing alphabetical order from the top down in the first stack. For example, if you choose 3 counters, the computer would respond with

```
A     |     |
B     |     |
C     |     |

ENTER A STRING OF TWO DIGITS FOR MOVE 1
FOR EXAMPLE '13' MEANS MOVE THE TOPMOST
COUNTER FROM STACK ONE TO STACK THREE?
```

You are invited to enter your move as a two digit number. If you wish to move the top counter from stack 1 to stack 3, you would input 13. The computer would respond with the message

```
      |     |     |
B     |     |
C     |     A
```

You might now want to move the counter B from stack 1 to stack 2, so you would enter 12. The situation would then be displayed as

You would then enter 32 followed by 13, 21, 23 and finally 13. The completion of your task is then signalled by the message

```
MOVES TO COMPLETE TASK IS 7
THE UNIVERSE IS SAFE, THANKS TO YOU
AGAIN? PRESS Y OR N
```

If you wish to quit, press the key N. To play another game, press the key Y. Your moves are checked, if you make a mistake, the universe is doomed! If you take too many moves, you do not save the universe, but you do get another chance!

THE MATHEMATICS

I leave you to solve the following mathematical problems. In order to solve the problem posed by the game in the fewest number of moves, you will have to move the first counter into the correct stack. For an odd number of counters, you should move the first counter into which stack? Once you have this figured out, you will also know which stack to move the first counter into if you have an even number of counters. Now, what is the minimum number of moves for N counters? Solve the game for N = 1, 2, 3, ... and see if you can arrive at the general rule.

CHAPTER IV

ARITHMETIC GAMES

"Barter"

```
100 LET COW=INT(151*RND(0))+20
110 LET Z=INT(COW/19)
120 PRINT "□":PRINT
130 PRINT "     CATS ARE WORTH $1"
135 PRINT "     HENS ARE WORTH $3"
140 PRINT "     DOGS ARE WORTH $5"
145 PRINT "AND PIGS ARE WORTH $10":PRINT
150 PRINT "THERE ARE NO MORE THAN";Z+1;"
ANIMALS OF":PRINT
155 PRINT "EACH KIND. HOW MANY OF EACH K
IND WOULD":PRINT
160 PRINT "YOU BARTER FOR A $";COW;" COW
?":PRINT
165 PRINT "ENTER YOUR ANSWER IN A STRING
 LIKE 3233":PRINT:INPUT A$
170 PRINT "□"
180 FOR P=-1 TO 1
190 FOR Q=-1 TO 1
200 FOR R=-1 TO 1
210 FOR S=-1 TO 1
220 IF P+3*Q+5*R+10*S=COW-Z*19 THEN 240
230 NEXT S,R,Q,P
240 FOR I=1 TO 4
250 IF VAL(MID$(A$,I,1))>Z+1 THEN 290
260 NEXT I
270 LET A=VAL(MID$(A$,1,1))+3*VAL(MID$(A
$,2,1))+5*VAL(MID$(A$,3,1))
280 IF A+10*VAL(MID$(A$,4,1))=COW THEN P
RINT "YOU ARE CORRECT":GOTO 290
285 PRINT "YOU ARE WRONG!":PRINT
290 PRINT "MY ANSWER IS";P+Z;Q+Z;R+Z;S+Z
300 PRINT
310 PRINT "AGAIN? PRESS Y OR N"
330 GET A$:IF A$="" THEN 330
340 IF A$="Y" THEN 100
350 STOP
```

Note: The symbol □ represents SHIFT CLEAR/HOME.

"Barter"

AIM OF THE GAME

This is a game that encourages you to sharpen your mental arithmetic skills. You have a cow worth a given amount. The amount is chosen randomly. You are seeking a fair exchange for a number of cats, hens, dogs, and pigs, which are each worth a certain fixed amount. There is only a limited quantity of each of the cats, hens, dogs, and pigs. You are asked to determine how many of each you are willing to exchange for your cow.

HOW TO PLAY

When you run the program a problem is posed like:

```
     CATS ARE WORTH $1
     HENS ARE WORTH $3
     DOGS ARE WORTH $5
AND PIGS ARE WORTH $1Ø
THERE ARE NO MORE THAN 9 ANIMALS OF
EACH KIND. HOW MANY OF EACH KIND WOULD
YOU BARTER FOR A $54 COW?
ENTER YOUR ANSWER IN A STRING LIKE 3233
```

Note that your entry should be in a string, so if you decide that you want 3 cats, 2 hens, 3 dogs and 3 pigs, you would enter 3233. No individual number will ever exceed 9, so your entry will always contain only four digits.

The computer checks your answer. If you are right, the computer will print the message

```
YOU ARE CORRECT
```

Then right or wrong, you will get the answer computed by the computer,

```
MY ANSWER IS 3233
AGAIN? PRESS Y OR N
```

If you wish to quit, press the key N or any key other than Y. If you press Y, another problem is given to you.

THE MATHEMATICS

The mathematical solution to this problem is based on the fact that combinations of the numbers 1,3,5 and 1Ø give you all the numbers from 1 to 19. For instance, 7=1Ø–3, 17=1Ø+5+3–1. Each number is used at

most once, either by adding or subtracting. So if a cow is worth $6Ø, we divide 19 into 6Ø to get 3 with a remainder of 3. This means we take 3 of each of 1,3,5 and 1Ø and one more 3. So 3433 is the solution in the format required by this program. Another solution is 4243, so do not be surprised if your answer is different from the answer given by your computer. The solution is computed in lines 18Ø to 23Ø. If you wanted the game to have unique answers, you would have to specify that at least Z and no more that Z+1 animals of each kind should be bartered, and then you would change the limits in the FOR statements from -1 TO 1 to Ø TO 1. I leave you to experiment with that.

Here is a connected problem for you to think about.

The face value of the individual pieces in a currency are chosen so that transactions can be made with relatively few individual pieces of money. Design a system of coins so that the values from 1¢ to 99¢ can be made with the fewest number of coins.

"Base"

```
11Ø LET Z=ASC("A")-1Ø
115 PRINT "□"
12Ø PRINT "ENTER BASE 2 TO 36"
13Ø INPUT D:IF D< 2 OR D>36 THEN 115
135 PRINT "□"
136 LET S$=""
14Ø PRINT "BASE IS";D
15Ø PRINT:PRINT "ENTER ANY NUMBER"
16Ø PRINT "1 TO 4294967295"
17Ø INPUT A$
18Ø LET A=VAL(A$)
182 IF A<1 OR A>4294967295 THEN 135
185 PRINT:PRINT A$;" IN BASE";D;"IS"
19Ø GOSUB 3ØØ
2ØØ PRINT:PRINT
21Ø PRINT "PRESS Ø FOR NEW BASE"
215 PRINT "1 FOR A NEW NUMBER"
216 PRINT "ANYTHING ELSE TO STOP"
22Ø GET G$:IF G$="" THEN 22Ø
23Ø IF G$="Ø" THEN 115
24Ø IF G$="1" THEN 135
25Ø STOP
3ØØ LET I=1
32Ø LET B=A-D*INT(A/D)
33Ø LET A=INT(A/D)
335 IF B>9 THEN LET S$=S$+CHR$(Z+B)
336 IF B>9 THEN 35Ø
34Ø LET S$=S$+STR$(B)
35Ø LET I=I+1
36Ø IF A>Ø THEN 32Ø
37Ø FOR J=2*I TO 1 STEP -1
38Ø LET C$=MID$(S$,J,1)
385 IF C$<>" " THEN PRINT C$;
39Ø NEXT J
4ØØ RETURN
```

Note: The symbol □ represents SHIFT CLEAR/HOME.

"Base"

AIM OF THE GAME

This is a simple program that teaches you how to convert any decimal number of your choice to its representation in another base of your choice. Limitations are given in the prompts because:

(a) You will notice that numbers represented in base 2 contain only two digits, Ø and 1. Generally, a number represented in base N contains N digits, Ø, 1, 2,..., N–1. We do not have single characters for representing numbers greater than 9, unlike the Babalonians of antiquity, who used a base of 6Ø! We use the letters A, B, C,..., Z to represent the numbers 1Ø, 11, 12,..., 35 respectively. The program could easily be altered so that bases up to 6Ø could be used, but then we would have to use the graphics characters, and it is more difficult to recognize an increasing order. We would have to establish some order and commit it to memory. You will find that this limitation will not prevent you from obtaining the knowledge necessary to convert a number to its representation in any desired base.

(b) The limitation that the number must be less than 4294967295 is needed because this is the largest integer that your computer can store accurately. It is 2↑32–1, which in base 2 is represented by thirty two ones, as in 11111111111111111111111111111111. The numbers Ø and 1 are represented by these respective digits in any base. This program does not convert negative numbers, but it could be done easily. I leave you to make the necessary changes so that your number lies between the limits –4294967295 and 4294967295.

If you remove the restriction that numbers must be less than 4294967295 by removing line 182 in the code, then you can experiment to find out what happens when you enter numbers that exceed this limit.

The development of computer technology necessitated the study of representations of numbers in bases other than 1Ø. The number base usually used by a computer is binary, or base 2. This is the easiest because there are only two digits needed in the representation of numbers, Ø to 1. However, base 8 and base 16 are used in many computers.

The reason that strings were used in the code to store the representations was that some representations have up to 32 figures in them, and could not be displayed in number format.

HOW TO PLAY

When you run this program the first prompt you are given is

```
ENTER BASE 2 TO 36
```

Suppose you enter the number 2. The screen clears and you are given a new message and prompt

```
BASE IS 2
ENTER ANY NUMBER
1 TO 4294967295
```

Suppose you enter the number 15, then below the above message appears

```
15 IN BASE 2 IS
1111
```

This is the representation of 15 in base 2. Then you are given the prompt

```
PRESS Ø FOR NEW BASE
1 FOR A NEW NUMBER
ANYTHING ELSE TO STOP
```

If you enter the number Ø, then the screen will clear and the program returns to the prompt requesting you to enter the base. If you enter the number 1, then the same base is used and you are prompted to enter any number. To quit, you just enter any other character, Q for example.

THE MATHEMATICS

The mathematics in this game is repetitive division which is performed in the subroutine starting at line 3ØØ. For instance, in the example cited above, to express the number 15 in the base 2, use the following procedure.

2 | 15	
7	1
3	1
1	1
Ø	1

You divide 2 into 15 to get 7 with a remainder of 1. You keep dividing by 2 until you reach Ø, the remainders give you the required representation, in this case 1111. This binary representation means that:

15 = 1*(2↑3) + 1*(2↑2) + 1*(2) + 1

From the binary representation of 15, it is clear why this process works. Let us go through the process step by step.

Divide 15 by 2 to get 1*(2↑2)+1*(2↑1)+1 with a remainder of 1.
Divide again by 2 to get 1*(2↑1)+1 with a remainder of 1.
Divide again by 2 to get 1 with a remainder of 1.
Divide again by 2 to get Ø with a remainder of 1.

The representation of 15 in the base 3 is calculated in the following manner.

3 | 15	
5	Ø
1	2
Ø	1

The representation is given by 12Ø. Notice that the last remainder is the first digit in the required representation. This means that:

15 = 1*(3↑2) + 2*(3) + Ø

The representation of 15 in the base 4 is given by 33.

4 | 15	
3	3
Ø	3

You will have noticed that for a base N, the only digits occurring in the remainders are N–1, N–2,..., 1,Ø.

For bases greater than 1Ø, the arithmetic becomes more difficult for us, but not for the computer. We have to use the letters A, B,... to represent remainders which are greater than 9. For example, in finding the representation of 119 in the base 12, we proceed in the following manner.

12 | 119	
9	11
Ø	9

The representation is 9B, where B stands for the number 11.

Using A to represent a remainder of 1Ø, B for 11 and so on, up to Z for 35, we are able to represent any number in a base from 2 to 36.

When you have mastered this game, you might like to try your hand at doing arithmetic with numbers in various bases. For instance, try adding and subtracting numbers represented in base 2. Having mastered that, you might like to try multiplication and division.

The program "YOUR NUMBER IS UP" tests your ability at converting numbers in any base, back into numbers represented in the standard base 1Ø.

"Your Number Is Up"

```
11Ø LET S$=""
115 LET B=INT(35*RND(Ø))+2
12Ø PRINT "□"
13Ø PRINT "BASE IS";B
14Ø LET N=INT(999*RND(Ø))+36
15Ø LET N$=STR$(N)
16Ø GOSUB 3ØØ
17Ø PRINT:PRINT "NUMBER IS ";RIGHT$(S$,I
-1)
19Ø PRINT:PRINT "WHAT IS THE NUMBER IN B
ASE 1Ø?"
2ØØ INPUT G:PRINT
21Ø IF G=N THEN PRINT "YOU ARE CORRECT"
22Ø IF G<>N THEN PRINT "WRONG, IT IS";N
23Ø PRINT:PRINT "AGAIN? PRESS Y OR N"
24Ø GET A$:IF A$="" THEN 24Ø
25Ø IF A$="Y" THEN 11Ø
26Ø STOP
29Ø REM THIS CONVERTS N TO BASE B
3ØØ LET I=1
31Ø LET R=N-B*INT(N/B)
32Ø LET N=INT(N/B)
33Ø IF R>9 THEN LET S$=LEFT$(S$,32-I)+CH
R$(ASC("A")-1Ø+R)+RIGHT$(S$,I-1)
34Ø IF R>9 THEN 36Ø
35Ø LET S$=LEFT$(S$,32-I)+RIGHT$(STR$(R)
,1)+RIGHT$(S$,I-1)
36Ø LET I=I+1
37Ø IF N>Ø THEN 31Ø
38Ø LET N=VAL(N$)
39Ø RETURN
```

Note: The symbol □ represents SHIFT CLEAR/HOME.

"Your Number Is Up"

AIM OF THE GAME

This is the opposite of the game "BASE". You should first play "BASE" in order to learn how to convert any given decimal number to its representation in another base other than 1Ø. "YOUR NUMBER IS UP" tests your ability to convert numbers represented in any base, back into the base 1Ø.

You will find this a very challenging game since it tests your skill in converting numbers written in bases from 2 to 36. If you find this too difficult you could easily change the code so that bases from 2 to 16 are chosen. Simply change the 35 in line 115 to 15.

HOW TO PLAY

When you run this program, the screen clears and a message similar to this is displayed

```
BASE IS 11
NUMBER IS 5A2
WHAT IS THE NUMBER IN BASE 1Ø?
```

The representation 5A2 stands for
5*(11↑2) + 1Ø*(11) + 2, which is 5*(121) + 112 = 717. You enter 717 and you will get the message

```
YOU ARE CORRECT
AGAIN? PRESS Y OR N
```

If you press the key Y, the screen clears and a new problem is posed. If you press the key N, or any key other than Y, the program stops.

If you enter an incorrect answer, say you enter 617 in the above example, then you will receive the message

```
WRONG, IT IS 717
AGAIN? PRESS Y OR N
```

THE MATHEMATICS

The mathematics is the reverse of the mathematics in "BASE", so this time it is repetitive multiplication. Suppose the base is B, and the representation is 123. The number can be written as a polynomial

$$1*(B\uparrow 2) + 2*(B) + 3 = 1B^2 + 2B + 3$$

For example, 123 in base 1Ø is 1*1ØØ+2*1Ø+3. The simplest way to evaluate an expression with B not equal to 1Ø, is by nested multiplication, where we write the numbers as

((1*B)+2)*B+3

You can set this up in the following way:

	1	2	3
ADD		B	(B+2)*B
B*	1	B+2	(B+2)*B+3 = required decimal representation

This process is called SYNTHETIC DIVISION and is the most economical way of evaluating polynomials like

$P(X) = AX^2 + BX + C$

Of course, nothing much is gained with a polynomial of low degree, but when you have a polynomial like

$P(X) = AX^9 + BX^8 + CX^7 + \ldots + IX + J$

then the gains are substantial in both accuracy and time.

There is more to be gained by this process than meets the eye. This process of synthetic division is used to divide polynomials by linear factors. For instance, if you attempt to divide a linear factor like X–3 into the polynomial P(X), synthetic division gives you the remainder R and the quotient polynomial Q(X), where P(X) = Q(X)*(X–3) + R. R is the value of the polynomial at X = 3 since P(3) = Ø + R. What is more, in Calculus, the use of Newton's method for finding the roots of a polynomial requires the evaluation of the polynomial P(X) and its derivative P'(X). This is achieved by using synthetic division twice, once with the original polynomial, and a second time on the quotient polynomial arrived at in the process of synthetic division.

If you are familiar with Calculus, then you will observe that what we achieve from the factoring of P(X) into

P(X)=Q(X)*(X–3)+R

is that P'(X)=Q'(X)(X-3)+Q(X) when we differentiate. So that P'(3)=Q(3) which is the value of the quotient polynomial at X=3.

One final word. The numbers chosen in line 14Ø are mostly limited to 2 and 3 digits. If you would like to increase this, simply add a few more 9's to the number 999.

"Cryptic"

```
11Ø PRINT "□":PRINT:PRINT
12Ø FOR P=2 TO 9
127 LET S=Ø
13Ø PRINT "PRESS THE KEY + OR -"
14Ø GET OP$:IF OP$="" THEN 14Ø
145 IF OP$<>"+" AND OP$<>"-" THEN 11Ø
15Ø LET I=INT(15*RND(Ø))
155 LET M=1Ø↑(9-P)
157 LET L=1Ø↑(P-1)
16Ø LET C=INT(555555555/M*RND(Ø))
17Ø LET B=INT(444444444/M*RND(Ø))
18Ø IF OP$="+" THEN LET A=C-B
19Ø IF OP$="-" THEN LET A=C+B
2ØØ IF A<L OR B<L OR C<L THEN 16Ø
2Ø5 LET Z=4294967295
2Ø6 IF A>Z OR B>Z OR C>Z THEN 16Ø
21Ø LET B$(1)=STR$(A)
22Ø LET B$(2)=STR$(B)
23Ø LET B$(3)=STR$(C)
24Ø FOR J=1 TO 3
245 LET A$(J)=""
25Ø FOR K=2 TO P+1
255 LET B=VAL(MID$(B$(J),K,1))
26Ø LET A$(J)=A$(J)+CHR$(B+I+ASC("A"))
27Ø NEXT K
28Ø NEXT J
29Ø PRINT "□":PRINT:PRINT "THE FOLLOWING
 EQUATION IS IN CODE":PRINT
3ØØ PRINT A$(1);OP$;A$(2);"=";A$(3)
32Ø PRINT:PRINT "ENTER Q TO QUIT"
33Ø PRINT:PRINT "THE DECODED ANSWER IS?"
34Ø PRINT:INPUT Z$
345 LET S=S+1
35Ø IF VAL(Z$)<>C THEN 38Ø
355 PRINT:PRINT "CORRECT"
36Ø PRINT "NUMBER OF TRIES =";S
361 IF P<9 THEN PRINT "LET ME TRY YOU WI
TH A LONGER ONE"
363 FOR T=1 TO 4ØØØ:NEXT T
364 PRINT "□"
```

```
365 NEXT P
370 PRINT "AGAIN? PRESS Y OR N"
375 GET A$:IF A$="" THEN 375
376 IF A$="Y" THEN 110
377 STOP
380 IF Z$="Q" THEN PRINT "THE ANSWER IS"
;C:GOTO 370
385 PRINT "WRONG. ENTER A DIGIT."
390 INPUT W
395 PRINT "□"
400 FOR J=1 TO 3
410 FOR K=2 TO P+1
415 LET B=VAL(MID$(B$(J),K,1))
416 LET J$=CHR$(W+ASC("0"))
420 IF B=W THEN LET A$(J)=LEFT$(A$(J),K-
2)+J$+RIGHT$(A$(J),P+1-K)
430 NEXT K
440 NEXT J
460 GOTO 290
```

Note: The symbol □ represents SHIFT CLEAR/HOME.

"Cryptic"

AIM OF THE GAME

In this program you have the choice of problems in addition or subtraction. You are presented with an equation in code, where the digits in the numbers are represented by letters. For instance, the equation

123+234=357

might be displayed as

ABC+BCD=CEG

The game is to work out the answer 357 in the fewest number of moves. You are given clues as you proceed, and based upon these and your deductive reasoning powers, you work out the answer.

You are given two letter numbers to start with, and each time you solve a problem, the next problem involves numbers with one extra digit, up to a maximum of 9 digits.

HOW TO PLAY

When you run this program, you are prompted by the message

PRESS THE KEY + OR −

Say you press +, the screen clears and you are presented with an equation involving two digit numbers, for instance

```
THE FOLLOWING EQUATION IS IN CODE
LS+NP=PN
THE DECODED ANSWER IS?
```

Suppose you enter 53, since L and S appear in the problem, L can not be Ø, so it might be 1 and S would then be 8. Of course L could be 2 and then S would be 9, in which case, the correct answer would be 64. If you are wrong, the message

WRONG. ENTER A DIGIT

appears on the screen. You might enter the digit 6. The screen would clear and you would be presented with the equation

```
LS+N6=6N
THE DECODED ANSWER IS?
```

Now you know that the answer is 64, so you would enter 64. Then you would be given the message

```
CORRECT
NUMBER OF TRIES=2
```

LET ME TRY YOU WITH A LONGER ONE

After a pause, the screen clears and you are prompted to press + or – again.

THE CODE

You should be able to deduce the answer in 2 moves at most. If you would like to increase the difficulty, then you would need to replace the digits by letters in an arbitrary fashion, rather than in a sequence as this program does. I leave you to make the necessary changes.

Since the arithmetic with 9 digit numbers is not always exact, lines 2Ø5 and 2Ø6 guard against any of the numbers exceeding the largest integer 4294967295 which can be stored exactly inside the computer. You could attempt to change the code, so that every number is treated as a string. You would have to introduce subroutines to accomplish the arithmetic on the strings, but then the limitation of 9 digit numbers could be removed.

"Human Arithmetic"

```
110 PRINT "□"
120 LET J=INT(4*RND(0))
130 LET A=INT(19*RND(0))-9
140 LET B=INT(19*RND(0))-9
150 LET C=INT(19*RND(0))-9
160 LET D=INT(19*RND(0))-9
170 IF C=0 AND D=0 THEN 150
180 LET K=SQR(C*C+D*D)
190 IF J=3 AND K<>INT(K) THEN 150
200 LET P=42+(J-2)*(13*J*J-28*J-3)/6
210 PRINT "(";STR$(A);",";STR$(B);")";CH
R$(P);"(";STR$(C);",";STR$(D);")=(E,F)"
220 PRINT:PRINT "ENTER E AND F AS AN INT
EGER OR AS":PRINT
225 PRINT "A FRACTION LIKE 2/7"
230 PRINT:INPUT "ENTER E";E$
235 LET E=VAL(E$)
240 PRINT:INPUT "ENTER F";F$
245 LET F=VAL(F$)
250 ON J+1 GOTO 260,260,300,330
260 LET M=1-2*J
270 LET G=A+M*C
280 LET H=B+M*D
290 GOTO 370
300 LET G=A*C-B*D
310 LET H=A*D+B*C
320 GOTO 370
330 LET K=K*K
340 LET C=C/K
350 LET D=-D/K
360 GOTO 300
370 FOR I=1 TO LEN(E$)
375 IF MID$(E$,I,1)="/" THEN LET E=VAL(L
EFT$(E$,I-1))/VAL(RIGHT$(E$,LEN(E$)-I))
380 NEXT I
385 FOR I=1 TO LEN(F$)
390 IF MID$(F$,I,1)="/" THEN LET F=VAL(L
EFT$(F$,I-1))/VAL(RIGHT$(F$,LEN(F$)-I))
395 NEXT I
400 PRINT
```

```
405 IF G=E AND H=F THEN 430
410 PRINT "WRONG IT IS ";G;",";H
420 GOTO 440
430 PRINT "RIGHT ON"
440 PRINT:PRINT "AGAIN? PRESS Y OR N"
450 GET A$:IF A$="" THEN 450
460 IF A$="Y" THEN 110
470 STOP
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME.

"Human Arithmetic"

AIM OF THE GAME

Leopold Kronecker, a 19th century mathematician who developed the theory of algebraic numbers, once said "God gave us the natural numbers (integers greater than Ø); everything else was created by man". Hence the name for this program.

One of the most useful number systems invented is the one composed of COMPLEX NUMBERS. Numbers belonging to the real number system are used to represent the solutions of algebraic equations like $X^2 = 2$. We say X = + SQR(2) or – SQR(2), where the quantity SQR(2) is a real number whose square is precisely 2. We are unable to write SQR(2) as a fraction, so we can only approximately represent it as a decimal expansion 1.414213562... to any required accuracy.

If you want to solve an equation like $X^2 = -1$, what numbers do you use? The square of a non–zero real number is positive, so complex numbers were invented for this purpose. The key step is to introduce the quantity SQR(–1) called i. Then $i^2 = -1$ so we have $X^2 = i^2$ and the solution is X = +i and X = –i. Such numbers were referred to as "imaginary" numbers when they were first introduced, but now the term "complex" number is generally used.

In general, a complex number a + ib has two parts, a real part, a, and an imaginary part, b. We abbreviate the notation from a + b SQR(–1) to a + ib and finally to (a, b), which is an ordered pair. It is always understood that the second member of a pair is multiplied by i.

The program, "HUMAN ARITHMETIC", gives you practice in the arithmetic of complex numbers. Problems of addition, subtraction, multiplication and division are chosen at random. You are prompted to enter the solution as an ordered pair (E, F). First enter your guess for the number E and then enter your guess for the number F. If you are wrong, the correct answer is displayed. In either case, a new problem is posed after a pause.

The rules for addition and subtraction are simple.

$$(a, b) + (c, d) = (a + c, b + d)$$
$$(a, b) - (c, d) = (a - c, b - d)$$

However, multiplication and division are more complex! Keep in mind that $i^2 = -1$, then

$$\begin{aligned}(a, b) *(c, d) &= (a + ib) *(c + id)\\ &= a*c + i(a*d) + i(b*c) + i^2(b*d)\\ &= (a*c - b*d, a*d + b*c)\end{aligned}$$

Note that (a, Ø) acts like a non-complex number, as indeed it is. So (a,Ø) *(c, Ø) = a*c and (a, Ø) *(c, d) = (ac, ad). In fact the role of 1 in the arithmetic is played by the pair (1, Ø) .

Division (a, b)/(c, d) is a little tricky. We can divide by all complex numbers except (Ø, Ø). The trick is to define the multiplicative inverses for each complex number. That is, what is the number 1/(c, d)? A better way of saying this is, what is the complex number (x, y) such that (c, d) *(x, y) = (1, Ø) ?

You can verify by multiplication that the answer is

$$\left(\frac{c}{c^2 + d^2}, \frac{-d}{c^2 + d^2}\right)$$

So the rule is

$$(a, b)/(c, d) = (a, b) * \left(\frac{c}{c^2 + d^2}, \frac{-d}{c^2 + d^2}\right)$$

$$= \left(\frac{a * c + b * d}{c^2 + d^2}, \frac{b * c - a * d}{c^2 + d^2}\right)$$

HOW TO PLAY

When you run this program the screen clears and you are given a message like this (although the + symbol might be –, *or /)

```
(-9,7) + (-3,-3) = (E,F)
ENTER E AND F AS AN INTEGER OR AS
A FRACTION LIKE 2/7
ENTER E?
ENTER F?
```

You work out what you think E and F should be, in this case E=–12 and F=4, then you enter them one after the other. You will be given the message:

```
RIGHT ON
AGAIN? PRESS Y OR N
```

If you press the key Y you get another problem. Pressing any other key stops the game. If you enter an incorrect result, say –12,1Ø in the above example, you are given the message

```
WRONG IT IS -12,4
AGAIN? PRESS Y OR N
```

THE CODE

You should note that when an example is given for division, roundoff in the machine may cause a perfectly good answer to be rejected as wrong. For instance, a fraction like 1/7 and .142857143 would not be considered to be the same. For this reason, the answers E and F are entered as strings. The lines 37Ø and 395 scan the entries for a / . If it is there, the fraction is correctly evaluated. This allows you to enter your answer like 1/7 in precisely that fashion. If your entry is a number, it is treated as such.

"Jugs"

```
1Ø PRINT "◘":PRINT:PRINT "INSTRUCTIONS?
PRESS Y OR N":PRINT
2Ø GET A$:IF A$="" THEN 2Ø
3Ø IF A<>"Y" THEN 1ØØ
4Ø PRINT "YOU HAVE TO MEASURE A PRECISE
AMOUNT OF "
5Ø PRINT "MILK USING THREE JUGS OF DIFFE
RING SIZES"
6Ø PRINT "BY ADDING AND REMOVING AMOUNTS
 OF MILK  "
7Ø PRINT "WITH THE THREE JUGS"
8Ø PRINT:INPUT "PRESS RETURN TO START";
1ØØ PRINT "◘"
11Ø LET A=INT(1Ø*RND(Ø))
12Ø IF A-2*INT(A/2)=Ø THEN 11Ø
13Ø LET B=A+2*(INT(1Ø*RND(Ø))+1)
14Ø LET C=A+2*(INT(1Ø*RND(Ø))+1)
15Ø IF C=B THEN 14Ø
16Ø LET D=INT(2Ø*RND(Ø))
17Ø IF D=A OR D=B OR D=C OR D=Ø THEN 11Ø
18Ø PRINT:PRINT "YOUR THREE JUGS HOLD"
19Ø PRINT:PRINT STR$(A);",";STR$(B);" AN
D";STR$(C);" PINTS."
2ØØ PRINT:PRINT "HOW WOULD YOU GET";D;"P
INTS?"
21Ø PRINT:PRINT "ENTER YOUR SOLUTION IN
THE FORM -2,1,2 "
215 PRINT "WHICH MEANS YOU WOULD ADD THE
 SECOND JUG"
216 PRINT "PLUS TWICE THE THIRD, AND SUB
TRACT TWICE"
218 PRINT "THE FIRST JUG"
22Ø PRINT:INPUT "ENTER R,S,T";R,S,T
23Ø FOR L=1 TO 9
24Ø FOR I=-L TO L
25Ø FOR J=-L TO L
26Ø FOR K=-L TO L
27Ø IF I*A+J*B+K*C=D THEN 29Ø
28Ø NEXT K,J,I,L
29Ø IF A*R+B*S+C*T=D THEN PRINT:PRINT "R
```

```
IGHT ON":GOTO 3ØØ
295 PRINT:PRINT "YOU ARE WRONG!"
3ØØ PRINT:PRINT "MY ANSWER IS ";STR$(I);
",";STR$(J);" AND ";STR$(K)
31Ø PRINT:PRINT "AGAIN? PRESS Y OR N"
32Ø GET A$:IF A$="" THEN 32Ø
33Ø IF A$="Y" THEN 1ØØ
34Ø STOP
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME.

"Jugs"

AIM OF THE GAME

This is a painless way of practicing your arithmetic. You have three measuring jugs which hold different amounts of milk. By adding and subtracting the contents of the jugs, you have to get a precise quantity which is requested by a customer, who has a large empty container. There is usually more than one correct answer, so do not be surprised if your answer is different from the one given by the computer.

HOW TO PLAY

When you run the program, the screen clears and you are given a message like

```
YOUR THREE JUGS HOLD
5,11, AND 25 PINTS.
HOW WOULD YOU GET 2 PINTS?
ENTER YOUR SOLUTION IN THE FORM -2,1,2
WHICH MEANS YOU WOULD ADD THE SECOND JUG
PLUS TWICE THE THIRD, AND SUBTRACT TWICE
THE FIRST JUG
ENTER R,S,T?
```

One solution is –4 of the 5 pint jug, +2 of the 11 pint jug. So you would enter the numbers –4, then 2, and then Ø. The computer checks your answer, and also works out an answer for itself. Then you would get the message

```
RIGHT ON
MY ANSWER IS 1,2 AND -1
AGAIN? PRESS Y OR N
```

If your answer was not correct, the computer gives you the answer it computed. To obtain another problem, just press the key Y. If you wish to quit, just press the key N.

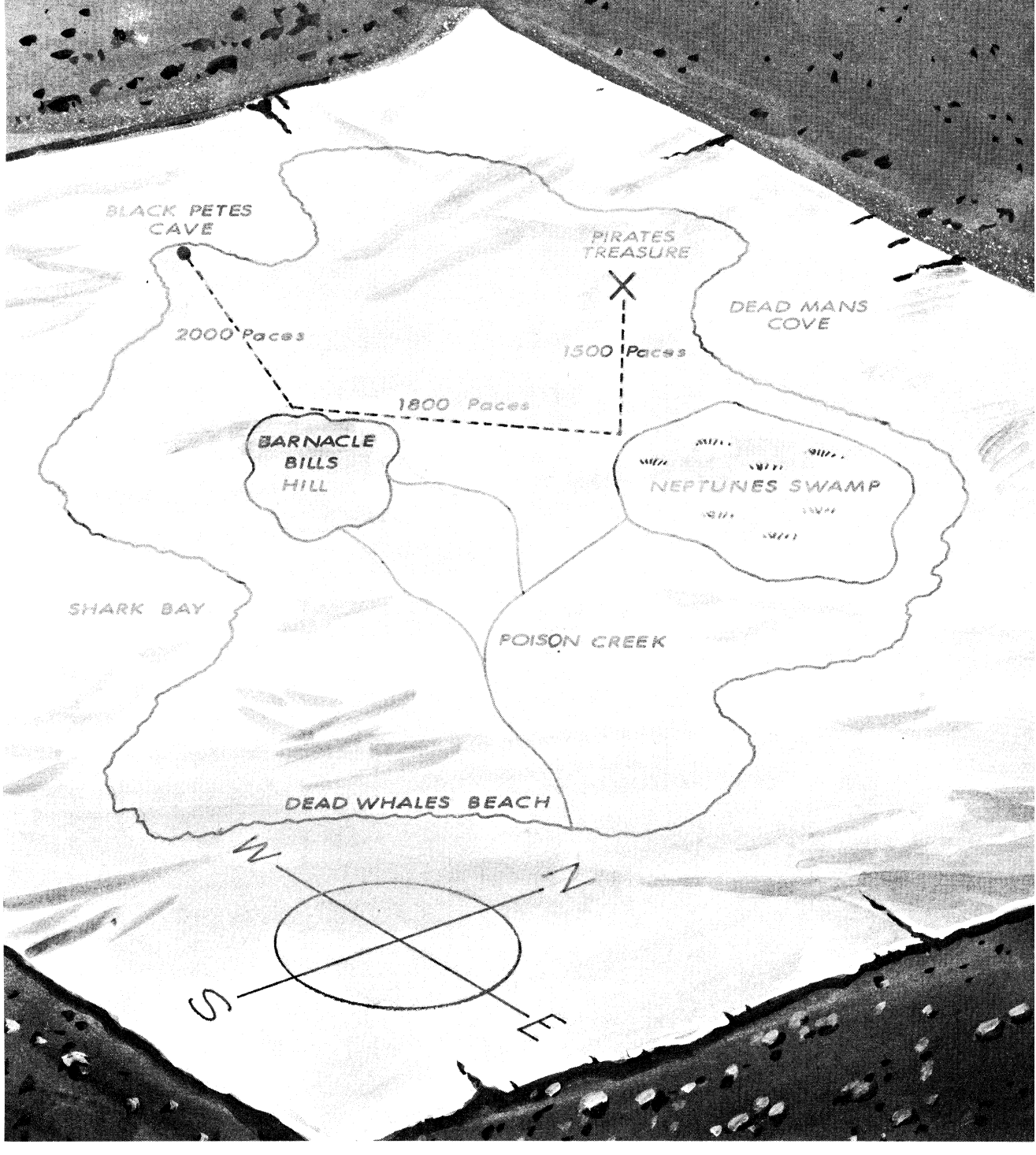
THE CODE

The lines 23Ø to 28Ø find a solution. The surprising thing in these problems is that at least one solution is usually found in the range –3 to 3 for the measuring jugs. You might like to increase the possible size of the measuring jugs by changing lines 11Ø to 14Ø, in which 1Ø*RND(Ø) could be altered to 2Ø*RND(Ø) or something else. You will notice that in these lines I have made the quantities A,B and C, which are the volumes of the

three jugs, to be always odd. The final quantity D can be either odd or even. Can you tell why I had to impose this restriction? It is an odd thought!

CHAPTER V

GEOMETRY GAMES



"Detonation"

```
100 REM:SET UP BOMB COORDINATES
110 FOR I=1 TO 3
120 LET X(I)=INT(100*RND(0))
130 IF X(I)=0 THEN 120
140 LET A(I)=0
150 NEXT I
160 LET C=0
170 PRINT "□"
180 LET D=0
190 FOR I=1 TO 3
200 LET D=D+ABS(X(I)-A(I))↑2
210 NEXT I
220 LET D=INT(SQR(D))
230 IF D=0 THEN 400
240 IF C=20 THEN 420
250 PRINT "YOU ARE AT";STR$(A(1));",";ST
R$(A(2));",";STR$(A(3)):PRINT
255 PRINT "IN A LARGE HOTEL WHICH HAS 10
00000 ROOMS"
260 PRINT "A TIME BOMB IS SET AT A DISTA
NCE";D:PRINT
261 PRINT "YOU HAVE";20-C;"MOVES TILL IT
 GOES OFF":PRINT
265 PRINT "PRESS S TO STOP RETURN TO CON
TINUE"
270 GET A$:IF A$="" THEN 270
280 PRINT:PRINT "ENTER 0 TO 99"
290 INPUT "ENTER COORDINATE A";A(1)
300 INPUT "ENTER COORDINATE B";A(2)
310 INPUT "ENTER COORDINATE C";A(3)
320 LET C=C+1
330 GOTO 170
400 PRINT "WELL DONE! YOU LOCATED THE BO
MB IN ONLY";C;"MOVES."
405 IF C<4 THEN PRINT "YOU MUST BE A GEN
IUS."
406 IF C>3 AND C<9 THEN PRINT"YOU ARE VE
RY CLEVER."
407 IF C>8 THEN PRINT "YOU NEED PRACTICE
."
```

```
41Ø GOTO 43Ø
42Ø PRINT"□":FORI=ØTO7:FORJ=ITO8:FORT=1T
O6:POKE 53281,J:NEXT T,J,I:POKE 53281,6
422 PRINT "□":PRINT:PRINT "BOOM! TOO LAT
E."
425 GOSUB 9ØØØ:REM PLAYS FUNERAL MUSIC
426 PRINT " BOMB WAS LOCATED AT ";X(1);
",";X(2);",";X(3)
43Ø PRINT:PRINT "ANOTHER GAME? PRESS Y O
R N"
44Ø GET A$:IF A$="" THEN 44Ø
45Ø IF A$="Y" THEN 11Ø
46Ø STOP
9ØØØ FOR L=54272 TO 54296:POKE L,Ø:NEXT:
POKE 54296,15
9Ø1Ø FOR I=1 TO 3:V=54269+7*I:POKE V+1,1
9Ø:POKE V+2,25Ø:NEXT I
9Ø2Ø RESTORE
9Ø3Ø FOR I=1 TO 3:READ H(I),L(I):NEXT I:
READ D
9Ø4Ø IF D=-1 THEN 9Ø8Ø
9Ø5Ø FOR I=1 TO 3:V=54266+7*I:POKE V,H(I
):POKE V-1,L(I):POKE 54269+7*I,33:NEXT
9Ø6Ø FOR T=1 TO 4*D:NEXT
9Ø65 FOR I=1 TO 3:POKE 54269+7*I,Ø:NEXT
9Ø7Ø GOTO 9Ø3Ø
9Ø8Ø FOR I=1 TO 3:POKE 54269+7*I,Ø:NEXT
9Ø9Ø RETURN
99ØØ DATA 17,37,12,216,8,147,1ØØ
991Ø DATA 17,37,12,216,8,147,2ØØ
992Ø DATA 17,37,12,216,8,147,5Ø
993Ø DATA 17,37,12,216,8,147,1ØØ
9935 DATA 2Ø,1ØØ,15,7Ø,1Ø,6Ø,2ØØ
994Ø DATA 19,63,14,1Ø7,9,159,5Ø
9945 DATA 19,63,14,1Ø7,9,159,2ØØ
995Ø DATA 17,37,12,216,8,147,5Ø
9955 DATA 17,37,12,216,8,147,2ØØ
996Ø DATA 16,47,13,156,11,114,5Ø
9965 DATA 17,37,12,216,8,147,2ØØ
9999 DATA -1,-1,-1,-1,-1,-1,-1
```

Note: The symbol □ represents SHIFT CLEAR/HOME.

"Detonation"

AIM OF THE GAME

A bomb has been planted somewhere in a large building that has one million rooms in it. Your job is to locate the bomb before it goes off! To give you some assistance, your computer tells you how far away you are from the bomb. You move to another room and the new distance from the bomb lets you know if you are getting any warmer or colder (which in our house around Easter egg-time means getting nearer or further away).

Each move is counted and you have a limit of only 2Ø moves until the bomb explodes. This is a challenging game and the suspense builds as you approach the moment of DETONATION. Fortunately no one is hurt if you fail to find the bomb in time, so you can have another go if you fail, and another, and ...

HOW TO PLAY

You start at the room with coordinates Ø,Ø,Ø. Each room is designated by a set of three coordinates, Ø to 99, Ø to 99, Ø to 99. With each turn you are given the distance, D, from your present position and the bomb and are prompted to enter new coordinates. The distance is measured by the Euclidean rule, or 2-norm, which is

$$D=SQR((A-X)^2 + (B-Y)^2 + (C-Z)^2)$$

Actually, you are given an integer, INT(D). The coordinates X, Y and Z are the bomb coordinates, while A, B and C are the coordinates for your present position. When you enter new coordinates, the values of A, B and C are altered to the new values that you enter.

A typical game goes as follows. The first message is something like this

```
YOU ARE AT Ø,Ø,Ø
IN A LARGE HOTEL WHICH HAS 1ØØØØØØ ROOMS
A TIME BOMB IS SET AT A DISTANCE 44
YOU HAVE 2Ø MOVES TILL IT GOES OFF
PRESS S TO STOP RETURN TO CONTINUE
```

When you press a key other than S, you are prompted to enter your coordinates A, B and C one after the other. These are your guesses for the bomb coordinates X, Y and Z respectively. Suppose you enter 44, then Ø and then Ø. The screen clears and a new message appears

```
YOU ARE AT 44,Ø,Ø
IN A LARGE HOTEL WHICH HAS 1ØØØØØØ ROOMS
```

```
A TIME BOMB IS SET AT A DISTANCE 22
YOU HAVE 19 MOVES TILL IT GOES OFF
```

and so on. In this case we overshot on the first coordinate. The first coordinate is likely to be around 33, so we could try 33,2Ø,Ø as our next guess. By trial and error you should be able to find the bomb. But if at first you don't succeed, maybe you should read the next section.

If you press the key S when prompted to STOP, you will be given the location of the bomb, for instance,

```
BOMB WAS LOCATED AT 39,11,19
ANOTHER GAME? PRESS Y OR N
```

If you want to quit, press the key N or any key other than Y.

THE MATHEMATICS

Mathematically, the solution is given by the point of intersection of three spheres inside the cubic building centered, as an example, at (Ø,Ø,Ø), (99,Ø,Ø) and (Ø,99,Ø).

That is, you must solve

$$X^2 + Y^2 + Z^2 = D_1^2$$

$$(X - 99)^2 + Y^2 + Z^2 = D_2^2$$

$$X^2 + (Y - 99)^2 + Z^2 = D_3^2$$

If you do this you are guaranteed to find the bomb in 3 moves! This is an example of the use of a branch of mathematics called analytic geometry which is a very useful mathematical technique. If you study calculus at a university, you will also study analytic geometry.

To solve this system of second degree equations, we subtract the first equation from the second to obtain

$$(X - 99)^2 - X^2 = D_2^2 - D_1^2$$

Expanding the first part we get

$$X^2 - 198X + (99)^2 - X^2 = D_2^2 - D_1^2$$

Cancelling the X^2 , we get by subtracting $(99)^2$ from both sides

$$-198X = -(99)^2 + D_2^2 - D_1^2$$

Since the distances D_1 and D_2 are given to you, all of the quantities on the right hand side are known, so we can find X by dividing by –198.

We do the same with the third equation. We subtract the first equation from the third equation to get

$$(Y - 99)^2 - Y^2 = D_3^2 - D_1^2$$

From which we obtain Y.

Finally, Z is found from the first equation, now that we know what X and Y are. Thus

$$Z^2 = D_1^2 - X^2 - Y^2$$

We take the positive square root for Z.

Note: All of this is perfectly correct in theory, provided the distances D_1, D_2 and D_3 are known exactly. However, in this game the distances are rounded off to integers. It usually makes no difference. However, I want to warn you that if, after all this analytic geometry and algebra, you are still distance 1 from the bomb, you will have to finish the problem "by guess and by golly".

If you fail to find the bomb in 2Ø moves, the bomb explodes with a BLAM and the coordinates where the bomb was located are displayed.

There is nothing sacred about the points (99,Ø,Ø) and so on that I used in my solution. You will find that the arithmetic is easier to do if you use the points (5Ø,Ø,Ø), (Ø,5Ø,Ø) and (Ø,5Ø,Ø). I leave you to work out the details.

THE CODE

Line 42Ø creates flashing lights by changing the screen colors. While lines 9ØØØ to 9999 make the funeral music. You could delete these lines and line 425 to shorten the listing. You would need to have a line 42Ø because it is referred to in lines 24Ø and 28Ø. You could use

```
42Ø GOTO 422
```

Note the difference in the output in lines 25Ø and 426. When you use STR$(A(1)) a blank appears only on the left of the number A(1), to allow the possibility of a negative sign.

"Quest"

```
10 PRINT "□":PRINT:PRINT:PRINT " NEED
INSTRUCTIONS? PRESS Y OR N"
20 GET A$:IF A$="" THEN 20
30 IF A$<>"Y" THEN 100
40 PRINT:PRINT "YOU ARE AT THE CENTER OF
 AN ISLAND WHICH"
50 PRINT "IS A SQUARE WITH SIDES FROM -9
9 TO 99    "
60 PRINT "TREASURE IS BURIED ON THE ISLA
ND AND YOU"
70 PRINT "WANT TO FIND IT IN 11 MOVES OR
 LESS      "
80 INPUT "PRESS RETURN TO GO ON";
100 PRINT "□"
110 LET C=0
120 LET A=0
130 LET B=0
140 LET T=INT(199*RND(0))-99
150 LET R=INT(199*RND(0))-99
160 IF T=A AND R=B THEN 120
170 LET D=ABS(A-T)+ABS(B-R)
180 IF D=0 THEN 250
190 PRINT "□":PRINT:PRINT
200 PRINT "YOUR PRESENT POSITION IS";STR
$(A);",";STR$(B):PRINT
205 PRINT "THE TREASURE IS BURIED";D;"PA
CES AWAY":PRINT
206 PRINT "ENTER -111 TO QUIT":PRINT
210 INPUT "ENTER COORDINATE A";A
215 IF A=-111 THEN 285
220 INPUT "ENTER COORDINATE B";B
225 IF B=-111 THEN 285
230 LET C=C+1
240 GOTO 170
250 PRINT:PRINT "YOU FOUND IT IN ONLY ";
C;"MOVES"
260 IF C<4 THEN PRINT "YOU'RE A GENIUS":
GOTO 290
270 IF C>=4 AND C<12 THEN PRINT "YOU ARE
 VERY SMART":GOTO 290
```

```
28Ø IF C>11 THEN PRINT "TOO SLOW,THE PIR
ATES GOT IT"
285 PRINT:PRINT "THE TREASURE WAS AT";T;
",";R
29Ø PRINT:PRINT "AGAIN? PRESS Y OR N"
3ØØ GET A$:IF A$="" THEN 3ØØ
31Ø IF A$="Y" THEN 11Ø
32Ø STOP
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME.

"Quest"

AIM OF THE GAME

You start at the center (Ø,Ø) of a square island with sides of length 199. Somewhere in the square is buried the pirates' treasure. The coordinates of the treasure are T and R which will be integers from –99 to 99. The difference between your coordinates (A,B) and the location of the treasure (T,R) is given to you at each turn and you are asked to supply new coordinates. You enter your new coordinates A and B one at a time. They can be any integers. The distance is different from the Euclidean distance used in "DETONATION". The distance from (A,B) to (T,R) is defined as

```
D = ABS(A-T) + ABS(B-R)
```

This measure of distance is referred to as the 1-norm.

A count of your moves is kept and if you are too slow, the pirates beat you to the treasure. This program is fun to run and it is not difficult to beat the pirates.

HOW TO PLAY

A typical run would proceed in this fashion. You are given the message

```
YOUR PRESENT POSITION IS Ø,Ø
THE TREASURE IS BURIED 131 PACES AWAY
ENTER -111 TO QUIT
ENTER COORDINATE A?
```

Suppose you enter 99 and then Ø to test the extreme of one coordinate. In this trial run we received the following message

```
YOUR PRESENT POSITION IS 99,Ø
THE TREASURE IS BURIED 185 PACES AWAY
```

So we were on the right track, but we overshot the mark. This means that we improved for a distance X, then we worsened by that same amount, and worsened another 27 units after that. So 99 = 2X + 27 which gives X = 36. So the first coordinate is 36. Let us try 36, followed by 95 (which adds up to 131). The message we got was

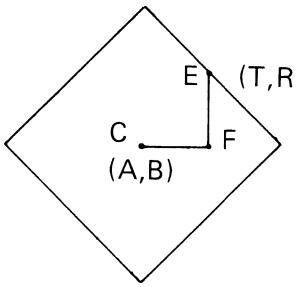
```
YOU FOUND IT IN ONLY 2 MOVES
YOU'RE A GENIUS
AGAIN? PRESS Y OR N
```

If you wish to play again, press the key Y. By pressing any other key, you will quit the game.

If the explanation for my move was not very helpful, you should read the next section where a different approach is suggested. Even if you have no trouble locating the treasure, you might find something of value in the mathematical description.

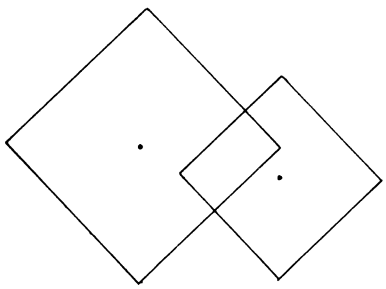
THE MATHEMATICS

The mathematical problem behind this game is to find the points of intersection of squares. This is because when distance is measured in the fashion described above, for a given distance D, the possible location of the point (T,R) lies on a square surrounding your location (A,B).



so CF = ABS (A−T)
FE = ABS (B−R)
D = CF + FE

When you move to a new location, then a new distance is computed and this defines a second square.



The two squares usually intersect at two points. (Why not no points?) However, in rare instances they may touch at only one point, or they may coincide along a portion of a common boundary so that there are an infinite number of points in common. In the latter case, you have to shift to a new location and waste a turn.

You are lucky if there is only one point in common because that is the solution. For the usual case of two points in common you only have to check one point. You have a 5Ø% chance of being right. In any case, you are guaranteed to find the solution in three attempts (unless you were unlucky and had to waste a turn).

So far I have only given you a geometrical interpretation of the solution to this problem. How do you go about producing numbers?

I suggest that instead of using equations as we have done in other cases, this time you should use graph paper. Locate an origin in the middle of your paper, draw a horizontal line and a vertical line through your origin. These are your coordinate axes. Now locate the point (A,B) on your map (it is the origin to start with) and draw the square which is a distance D from the point (A,B). You do this by locating the corners at (A+D,B), (A–D,B), (A,B+D) and (A,B–D). Now join the corners by straight lines using a ruler and you will have drawn the square. The treasure lies somewhere on this square. Now shift your coordinates to another point and draw the second square with the value of D given to you by the computer. From your graph, estimate the coordinates of the points of intersection of the squares and try entering them. After some trial and error, you will become more proficient at drawing graphs and you will easily be able to find the treasure before the pirates get to it because they do not have any graph paper!

THE CODE

Lines before 1ØØ may be left out since they only give instructions. Note the use of INPUT messages. Line 8Ø contains only a device for stopping the execution of the program while the screen message is being read.

"Trek"

```
1Ø PRINT "◘":PRINT:PRINT"INSTRUCTIONS? P
RESS Y OR N":PRINT
2Ø GET A$:IF A$="" THEN 2Ø
3Ø IF A$<>"Y" THEN 1ØØ
4Ø PRINT "YOUR SPACESHIP IS LOST IN A CU
BIC BLOCK"
5Ø PRINT "OF SPACE. HOME BASE IS SOMEWHE
RE IN THE"
6Ø PRINT "CUBE. YOU HAVE TO FIND THE COO
RDINATES"
7Ø PRINT "(H,O,M) OF HOME BASE, THEY ARE
 NUMBERS"
8Ø PRINT "BETWEEN Ø AND 99.":PRINT
9Ø INPUT "PRESS RETURN TO START";:REM WA
ITS WHILE SCREEN MESSAGE IS READ
1ØØ PRINT "◘"
11Ø LET E=Ø
12Ø LET A=Ø
13Ø LET B=Ø
14Ø LET C=Ø
15Ø LET H=INT(1ØØ*RND(Ø))
16Ø LET O=INT(1ØØ*RND(Ø))
17Ø LET M=INT(1ØØ*RND(Ø))
18Ø LET D=ABS(A-H)+ABS(B-O)+ABS(C-M):REM
 COMPUTES DISTANCE FROM HOME BASE
19Ø IF D=Ø THEN 3ØØ
2ØØ PRINT "◘"
21Ø PRINT "YOUR SPACESHIP IS AT"
22Ø PRINT A;",";B;",";C:PRINT
23Ø PRINT "THE DIFFERENCE BETWEEN YOUR C
OORDINATES":PRINT
24Ø PRINT "AND HOME BASE IS";D:PRINT
25Ø PRINT "ENTER COORDINATES FOR A JUMP
TO HOME":PRINT
255 PRINT "BASE.ENTER -1 TO QUIT.":PRINT
26Ø INPUT "ENTER YOUR ESTIMATE OF H";A
262 IF A=-1 THEN 38Ø
264 INPUT "ENTER YOUR ESTIMATE OF O";B
265 IF B=-1 THEN 38Ø
266 INPUT "ENTER YOUR ESTIMATE OF M";C
```

```
267 IF C=-1 THEN 380
270 LET E=E+1
275 PRINT "■":GOSUB 9000:REM FLASHES COL
ORS FOR SPACE JUMP AND PLAYS MUSIC
280 GOTO 180
300 PRINT "YOU ARE HOME IN";E;"MOVES"
310 IF E<4 THEN PRINT "YOU MUST BE A GEN
IUS"
320 IF E>3 AND E<12 THEN PRINT "YOU ARE
VERY CLEVER"
330 IF E>11 THEN PRINT "GOOD TRY"
340 PRINT:PRINT "AGAIN? PRESS Y OR N"
350 GET A$:IF A$="" THEN 350
360 IF A$="Y" THEN 100
370 STOP
380 PRINT:PRINT"HOME BASE WAS";STR$(H);"
,";STR$(O);",";STR$(M)
390 GOTO 340
9000 REM:SOUND AND COLORS
9010 FOR L=54272 TO 54296:POKE L,0:NEXT:
POKE 54296,15
9020 FOR I=1 TO 3:V=54269+7*I:POKE V+1,1
90:POKE V+2,250:NEXT I
9030 RESTORE
9035 FOR I=1 TO 3:POKE 54269+7*I,33:NEXT
 I
9040 FOR I=1 TO 3:READ H(I),L(I):NEXT I:
READ D
9050 IF D=-1 THEN 9090
9060 FOR I=1 TO 3:V=54266+7*I:POKE V,H(I
):POKE V-1,L(I):NEXT I
9070 FOR T=1 TO D/20:POKE 53281,9*RND(0)
:NEXT T:REM HOLDS NOTE, COLORS SCREEN
9080 GOTO 9040
9090 FOR I=1 TO 3:POKE 54269+7*I,0:NEXT:
POKE 53281,6:REM TURNS OFF MUSIC, COLORS
9095 RETURN
9900 DATA 11,114,0,0,5,185,1600
9910 DATA 17,37,11,114,8,147,1600
9920 DATA 22,227,17,37,11,114,1600
9930 DATA 28,214,17,37,14,107,100
9940 DATA 27,56,22,227,11,114,2800
9950 DATA 11,114,0,0,5,185,1600
9960 DATA 17,37,11,114,8,147,1600
```

```
997Ø DATA 22,227,17,37,11,114,16ØØ
998Ø DATA 28,214,17,37,14,1Ø7,1ØØ
999Ø DATA 3Ø,141,19,63,15,7Ø,28ØØ
9999 DATA -1,-1,-1,-1,-1,-1,-1
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME.

"Trek"

AIM OF THE GAME

This game is similar to the earlier game of "QUEST". You are the captain of a spaceship lost in space. You only know the absolute difference between your coordinates, (A,B,C) and HOME BASE, (H,O,M). The values of H,O and M can be anything from Ø to 99. You are asked to supply new coordinates each time you jump through space. The aim is to get to HOME BASE in the fewest number of turns. If you can do it in three moves or less you are a GENIUS. Good luck!

HOW TO PLAY

A typical run would proceed in this fashion. We are given the message

```
YOUR SPACESHIP IS AT
Ø,Ø,Ø
THE DIFFERENCE BETWEEN YOUR COORDINATES
AND HOME BASE IS 52
ENTER COORDINATES FOR A JUMP TO HOME
BASE, ENTER -1 TO QUIT.
ENTER YOUR ESTIMATE OF H?
```

Suppose we enter 52, then enter Ø and Ø in response to the prompts to enter O and M. The message is repeated, only this time the distance turned out to be 76! Evidently we overshot by 24. If we reduce the jump by 24 and enter 28,Ø,Ø then the difference would remain at 52. This means that the correct jump is to 14,Ø,Ø and the difference from home base would be 38. We could now try the warp coordinates 14, then 38, and then Ø. After this move, the distance turned out to be 2Ø. We continue to try different values in the second coordinate until we are satisfied that we can not get any closer to home by changes in the second coordinate. Then we change the last coordinate.

There is a very neat mathematical way of solving this problem in 3 moves. If you are interested, you should read the next section.

THE MATHEMATICS

The mathematical problem behind this game is to find the point of intersection of three planes. A plane in three-dimensional space is a linear equation in three variables, such as

$$2X + 3Y - 4Z = 5$$

You always start at the point (Ø,Ø,Ø), and the ship is located at an unknown point (H,O,M). All the coordinates are non-negative, so the first distance D1, supplied to you tells you that the sum of the coordinates H,O and M is D_1. We write this as

$$H + O + M = D_1$$

If you shift to a point say (99,Ø,Ø), then the new distance D_2 supplied to you by the computer tells you that the sum of the difference 99-H,O and M is D_2. We write this as

$$(99-H) + O + M = D_2$$

Similarly a shift to (Ø,99,Ø) gives you a third distance D3 and a third linear equation

$$H + (99-O) + M = D_3$$

The solution of these three equations guarantees that you will get home in three moves, but don't tell you friends how you did it! If you are having trouble solving these equations, read on.

You should subtract the first equation from the second one to get

$$(99-H) - H = D_2 - D_1$$

so that $99 - 2H = D_2 - D_1$. This means that

$$2H = 99 + D_1 - D_2 \text{ or } H = (99+D_1 - D_2)/2$$

and H should be an integer, otherwise you have made a mistake in your arithmetic.

In exactly the same way, when you subtact the first equation from the third, you will get

$$O = (99 + D_1 - D_3)/2$$

Then, from the first equation, you can find $M = D_1 - H - O$.

If you would like to practice Gauss Elimination on the matrix representing the three equations, then you start with the matrix

$$\begin{matrix} 1 & 1 & 1 & D_1 \\ -1 & 1 & 1 & D_2-99 \\ 1 & -1 & 1 & D_3-99 \end{matrix}$$

and add the first row to the second row to get

$$\begin{matrix} 1 & 1 & 1 & D_1 \\ Ø & 2 & 2 & D_1+D_2-99 \\ 1 & -1 & 1 & D_3-99 \end{matrix}$$

then subtract the first row from the third row to get

$$\begin{matrix} 1 & 1 & 1 & D_1 \\ Ø & 2 & 2 & D_1+D_2-99 \\ Ø & -2 & Ø & D_3-D_1-99 \end{matrix}$$

Now you add the second row to the third row to get

1	1	1	D_1
Ø	2	2	D_1+D_2-99
Ø	Ø	2	D_2+D_3-198

This is the equivalent system of equations that you can solve by back substitution. The last equation states that

$2M = D_2 + D_3 - 198$ or $M = (D_2 + D_3)/2 - 99$

The second to the last equation gives you

$O = (D_1 + D_2-99)/2 - M$

The first equation gives

$H = D_1 - O - M$

In this case, the formal process of Gauss Elimination takes longer than the elimination process described earlier. The point is, Gauss Elimination can be programmed easily. It does not pay attention to any particular short cuts that a system of equations may have, but plods a well defined path to the ultimate goal.

THE CODE

As usual in these listings, lines 1Ø to 9Ø give information, while lines 9ØØØ to 9999 create the sound and screen effects. You could ignore these lines and also delete line 275 for a much shorter program.

"U-Boat"

```
10 PRINT "□":PRINT:PRINT "INSTRUCTIONS?
PRESS Y OR N":PRINT
20 GET A$:IF A$="" THEN 20
30 IF A$<>"Y" THEN 100
40 PRINT "YOU ARE IN CONTROL OF A SUBCHA
SER. YOUR"
50 PRINT "SONAR GIVES THE RANGE AND BEAR
ING OF A"
60 PRINT "SUBMARINE. THE SUB IS LOCATED
SOMEWHERE"
70 PRINT "IN A SQUARE -99 TO 99, AND IS
AT A DEPTH"
80 PRINT "OF 0 TO 99. YOUR TASK IS TO FI
ND THE SUB"
90 INPUT "PRESS RETURN TO START";
100 PRINT "□"
110 LET S=INT(199*RND(0))-99:REM SET UP
SUBMARINE COORDINATES
120 LET U=INT(199*RND(0))-99
130 LET B=INT(99*RND(0))
140 LET C=0
150 LET X=0
160 LET Y=0
170 LET Z=0
180 GOSUB 400
190 IF S-X=0 THEN LET A=SGN(U-Y)*90:REM
SUB IS DIRECTLY NORTH OR SOUTH
200 IF S-X<>0 THEN LET A=INT(ATN((U-Y)/(
S-X))*180/π):REM SUB BEARING
210 IF S-X<0 THEN LET A=A+180
220 PRINT "SUBMARINE BEARING:";A;"DEGREE
S"
230 PRINT:PRINT "RANGE:";R;"METERS"
235 PRINT:PRINT "YOUR PRESENT POSITION I
S ";X;",";Y
240 PRINT:PRINT "ENTER COORDINATES X,Y A
ND Z FOR THE SUB. USE -111 TO QUIT."
250 PRINT:INPUT "ENTER COORDINATE X";X
252 IF X=-111 THEN 330
254 INPUT "ENTER COORDINATE Y";Y
```

```
256 IF Y=-111 THEN 330
258 INPUT "ENTER COORDINATE Z";Z
259 IF Z=-111 THEN 330
260 LET C=C+1
270 PRINT "□"
280 IF C=20 THEN 330
290 GOSUB 400
300 IF R>2 THEN 170
310 PRINT "GOT IT IN";C;"MOVES"
320 GOTO 340
330 FOR I=1 TO 9:FOR T=1 TO 9:POKE 53281
,9*RND(0):NEXT T,I:REM EXPLOSIVE COLOR
332 PRINT "□":POKE 53281,6:REM RETURNS S
CREEN COLOR TO BLUE
335 PRINT:PRINT "SUBMARINE GOT YOU!":PRI
NT "LOCATION WAS:";S;",";U;",";B
340 PRINT:PRINT "WANT TO TRY AGAIN?"
350 PRINT "PRESS Y OR N"
360 GET A$:IF A$="" THEN 360
370 IF A$="Y" THEN 100
380 STOP
400 LET R=SQR((S-X)↑2+(U-Y)↑2+(B-Z)↑2):R
EM COMPUTES RANGE
410 RETURN
```

Note: The symbol □ represents SHIFT CLEAR/HOME.

"U-Boat"

AIM OF THE GAME

You are in control of a subchaser. The detection devices, called SONAR, on your ship provide you with the range and bearing of the U-boat. Your job is to track down the U-boat in less than 2Ø moves. This is similar to the game "DETONATION", however, you have to use both the Cartesian coordinate system, (X,Y,Z) and the cylindrical polar coordinate system (R,A,Z).

The U-BOAT is located at the Cartesian coordinates (S,U,B). The S and U values are the coordinates in the X (or East) direction and the Y (or North) direction and are integers randomly chosen from -99 to 99. The value of B is the depth, and is an integer from Ø to 98. Evidently, this U-BOAT cannot fly!

Your coordinates initially are set to be X=Ø, Y=Ø and Z=Ø. Based upon the range and bearing of the submarine, you enter your guess at the coordinates S, U and B. Your subchaser moves to the new position X=S, Y=U and Z=Ø. The coordinate Z remains Ø since your subchaser remains on the surface, at least until you are hit by a torpedo, which does happen if you are too slow in finding the submarine!

A new range and bearing are given to you, and again you are prompted to enter your guess at the coordinates S,U and B. The chase continues until you locate the submarine, or run out of moves.

HOW TO PLAY

A typical run of this program commences with a message like this.

```
SUBMARINE BEARING: 82 DEGREES
RANGE: 1Ø5.569882 METERS
YOUR PRESENT POSITION IS Ø,Ø
ENTER COORDINATES X,Y AND Z FOR THE SUB.
USE -111 TO QUIT.
ENTER COORDINATE X?
```

You enter your guess of X, then Y and finally the depth Z. If you are not successful in locating the submarine, your subchaser will now be located at (X,Y,Ø). The coordinate Z is set to zero, since your subchaser remains on the surface, at least until you are hit by a torpedo (which does happen if you are too slow in finding the submarine!)

The distance is measured in the Euclidean or 2-norm, so the mathematical way of solving this game is similar to the method described

in "DETONATION". If you are interested in exploring this, you should read the next section.

THE MATHEMATICS

The mathematical way to solve this is to follow the method outlined in "DETONATION". You look for the intersection of three spheres, centered, say, at (Ø,Ø,Ø), (±5Ø,Ø,Ø), (Ø,±5Ø,Ø). The choice of the sign +5Ø or –5Ø is up to you, and depends on the initial bearing of the submarine which will tell you in which quadrant the submarine is located.

Suppose you try (5Ø,Ø,Ø) and (Ø,5Ø,Ø), then you will have to solve the equations

$$S^2 + U^2 + B^2 = R_1^2$$

$$(S-5Ø)^2 + U^2 + B^2 = R_2^2$$

$$S^2 + (U-5Ø)^2 + B^2 = R_3^2$$

where R_1, R_2 and R_3 are the initial range and the ranges at (5Ø,Ø,Ø) and (Ø,5Ø,Ø) respectively. Solve these equations and you are guaranteed to locate the submarine in 3 moves!

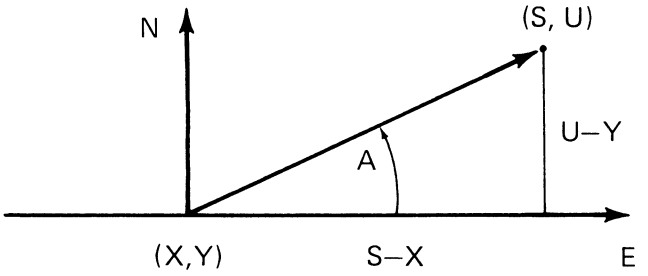
THE CODE

The range or distance in the program is the Euclidean distance worked out in the subroutine at line 4ØØ,

$$R = SQR((S-X)^2 + (U-Y)^2 + (B-Z)^2)$$

This is not the distance a crow flies, unless the submarine happens to be on the surface. Rather, it is the distance a fish swims, to coin a phrase! The value of R is not an integer, and this makes the arithmetic more difficult. If you would prefer an integer, put INT in front of the SQR in line 4ØØ, but that could be misleading as you found in "DETONATION".

The tricky piece of mathematics in this program, is in the calculation of the BEARING or direction of the submarine. Here, I use the usual polar coordinates, based on the East and North coordinates of your ship and the submarine.



The submarine is located at (S,U) and you are at (X,Y). The difference in the East direction is S–X while the difference in the North direction is U–Y. The BEARING, or angle measured in degrees in a counter clockwise direction from the East direction is A=ATN((U–Y)/(S–X)) .

Since the submarine could be in any of the four quadrants relative to your position, this angle can be positive or negative. In lines 190 to 210, I take care of this as well as the case when S=X, where the submarine is either directly north or south of you.

"Warp"

```
1Ø PRINT "□":PRINT:PRINT"INSTRUCTION? P
RESS Y OR N":PRINT
2Ø GET A$:IF A$="" THEN 2Ø
3Ø IF A$<>"Y" THEN 1ØØ
4Ø PRINT " YOUR SHUTTLE IS LOST IN A TIM
E WARP."
5Ø PRINT "YOUR INSTRUMENTS ONLY GIVE YOU
 THE NET"
6Ø PRINT "ABSOLUTE DIFFERENCE IN YOUR CO
ORDINATES"
7Ø PRINT "AND THE COORDINATES OF YOUR MO
THER SHIP"
8Ø PRINT "YOU HAVE FUEL FOR ONLY 2Ø MOVE
S TO FIND"
9Ø PRINT "THE COORDINATES X,Y,Z AND T O
F YOUR"
95 PRINT "MOTHER SHIP.":PRINT:INPUT "PRE
SS RETURN TO START";:REM MAKES A PAUSE
1ØØ PRINT "□"
11Ø LET E=2Ø
12Ø LET A=Ø
13Ø LET B=Ø
14Ø LET C=Ø
145 LET F=Ø
15Ø LET H=INT(1ØØ*RND(Ø)):REM SETS UP SH
IPS COORDINATES
16Ø LET O=INT(1ØØ*RND(Ø))
17Ø LET M=INT(1ØØ*RND(Ø))
175 LET G=INT(1ØØ*RND(Ø))
18Ø LET D=ABS(A-H)+ABS(B-O)+ABS(C-M)+ABS
 (F-G):REM COMPUTES DISTANCE
19Ø IF D=Ø THEN 3ØØ
2ØØ PRINT "□"
21Ø PRINT "YOUR SHUTTLE IS LOST IN A TI
ME WARP":PRINT
22Ø PRINT "DISTANCE FROM SHIP IS";D;"GAL
ACTIC UNITS":PRINT
23Ø PRINT "YOUR COORDINATES ARE";STR$(A)
;",";STR$(B);",";STR$(C);",";F:PRINT
24Ø PRINT "YOU HAVE FUEL FOR ONLY";E;"MO
```

```
VES"
250 PRINT:PRINT "ENTER COORDINATES Ø TO
99 OF YOUR SHIP"
255 PRINT:PRINT "ENTER -1 TO QUIT":PRINT
260 INPUT "ENTER COORDINATE X";A
261 IF A=-1 THEN 400
262 INPUT "ENTER COORDINATE Y";B
263 IF B=-1 THEN 400
264 INPUT "ENTER COORDINATE Z";C
265 IF C=-1 THEN 400
266 INPUT "ENTER COORDINATE T";F
267 IF F=-1 THEN 400
270 LET E=E-1
275 IF E=Ø THEN 400
276 PRINT "□":GOSUB 9000:REM FLASHES COL
ORS FOR SPACE JUMP AND PLAYS MUSIC
280 GOTO 180
300 PRINT "□":PRINT "SUCCESS! YOU ARE HO
ME IN";20-E;"MOVES":PRINT
310 IF E>16 THEN PRINT "YOU SHOULD BE A
STOCK BROKER!"
320 IF E<17 AND E>8 THEN PRINT "YOU ARE
VERY CLEVER"
330 IF E<9 THEN PRINT "GOOD TRY"
340 PRINT:PRINT "AGAIN? PRESS Y OR N"
350 GET A$:IF A$="" THEN 350
360 IF A$="Y" THEN 100
370 STOP
400 PRINT "□"
405 PRINT "TOO BAD YOU RAN OUT OF FUEL"
410 PRINT "THE SHIP WAS AT";H;O;M;G
420 GOTO 330
9000 REM:SOUND AND COLORS
9010 FOR L=54272 TO 54296:POKE L,Ø:NEXT:
POKE 54296,15
9020 FOR I=1 TO 3:V=54269+7*I:POKE V+1,1
90:POKE V+2,250:NEXT I
9030 RESTORE:REM RESETS DATA
9035 FOR I=1 TO 3:POKE 54269+7*I,33:NEXT
 I
9040 FOR I=1 TO 3:READ H(I),L(I):NEXT I:
READ D
9050 IF D=-1 THEN 9090
9060 FOR I=1 TO 3:V=54266+7*I:POKE V,H(I
```

```
):POKE V-1,L(I):NEXT I
9Ø7Ø FOR T=1 TO D/2Ø:POKE 53281,9*RND(Ø)
:NEXT T:REM HOLDS NOTE COLORS SCREEN
9Ø8Ø GOTO 9Ø4Ø
9Ø9Ø FOR I=1 TO 3:POKE 54269+7*I,Ø:NEXT:
POKE 53281,6:REM TURNS OFF MUSIC, COLORS
9Ø95 RETURN
99ØØ DATA 11,114,Ø,Ø,5,185,16ØØ
991Ø DATA 17,37,11,114,8,147,16ØØ
992Ø DATA 22,227,17,37,11,114,16ØØ
993Ø DATA 28,214,17,37,14,1Ø7,1ØØ
994Ø DATA 27,56,22,227,11,114,28ØØ
995Ø DATA 11,114,Ø,Ø,5,185,16ØØ
996Ø DATA 17,37,11,114,8,147,16ØØ
997Ø DATA 22,227,17,37,11,114,16ØØ
998Ø DATA 28,214,17,37,14,1Ø7,1ØØ
999Ø DATA 3Ø,141,19,63,15,7Ø,28ØØ
9999 DATA -1,-1,-1,-1,-1,-1,-1
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME.

"Warp"

AIM OF THE GAME

This is the most difficult of all the hide and seek type of games in this book. You are separated from your spaceship. You are in a shuttle lost in a time warp! Your instruments can only give you the total difference in your coordinates and those of the ship. There are four such coordinates, three spatial coordinates and one for time. You are prompted to enter new coordinates each move and you are given your new distance from the spaceship. You enter the coordinates one after the other. You only have sufficient fuel for 2Ø moves, after that you are doomed to float about in space forever!

HOW TO PLAY

When you run this program you are given a message like this

```
YOUR SHUTTLE IS LOST IN A TIME WARP
DISTANCE FROM SHIP IS 73 GALACTIC UNITS
YOUR COORDINATES ARE Ø,Ø,Ø,Ø
YOU HAVE FUEL FOR ONLY 2Ø MOVES
ENTER COORDINATES Ø TO 99 OF YOUR SHIP
ENTER -1 TO QUIT
ENTER COORDINATE X?
```

You enter your guess at the coordinates, one coordinate after the other. The new message will give you a new distance from your ship. If you try to solve this game by "guess and by golly", you will have a very difficult time solving it in the 2Ø moves allowed. But then again, you might be lucky.

THE MATHEMATICS

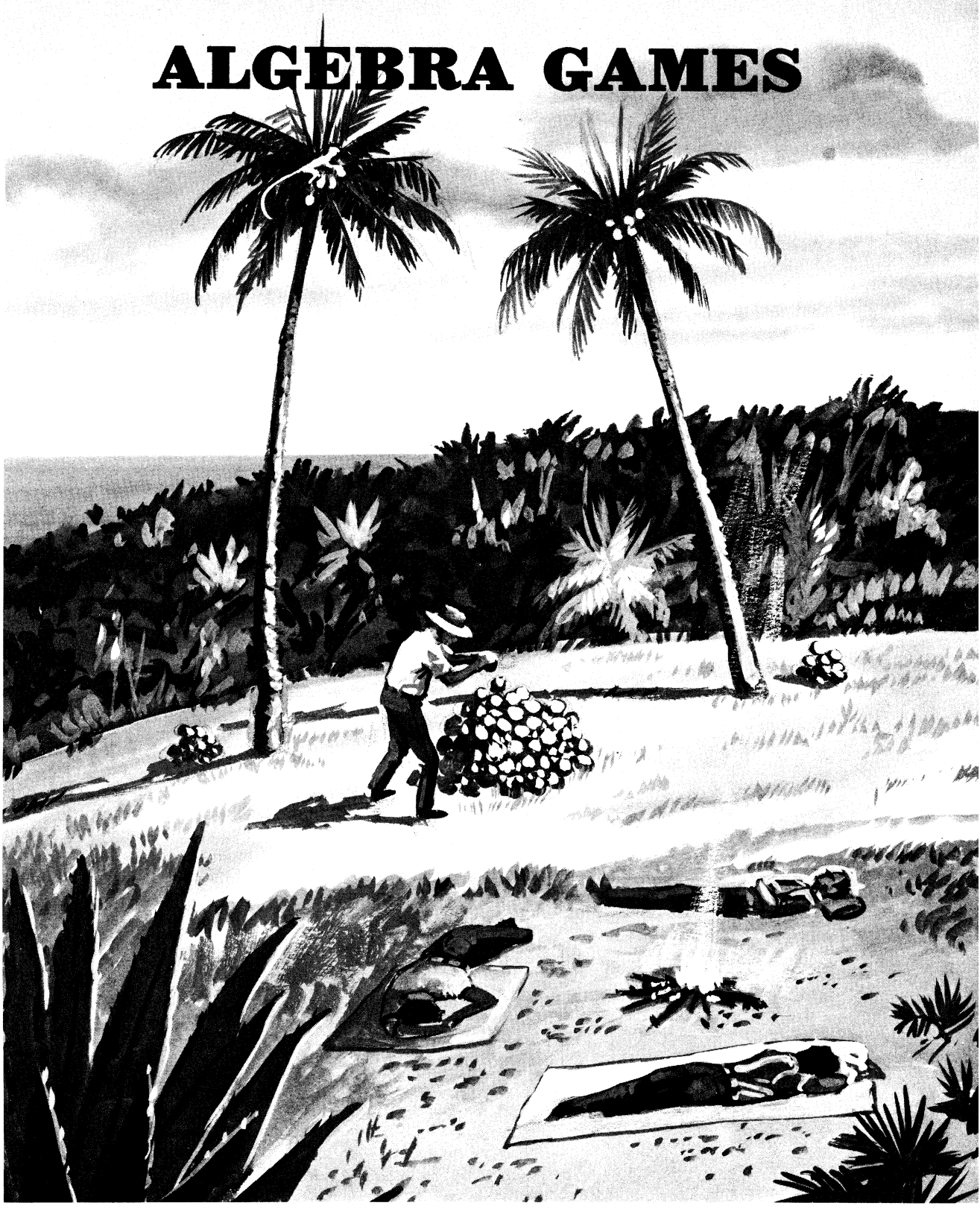
Using analytic geometry, the solution is found in four moves by finding the point of intersection of four hyperplanes. A hyperplane in the space with coordinates X, Y, Z, T is represented by a linear equation such as

$$2X + 3Y + 5Z + T = 24$$

I suggest that you go back to the game called "TREK" and master that game first. The mathematical explanation there will help you. This game is very similar and you should be able to find four linear equations in the four unknowns. Thus you are guaranteed a solution in four moves. If you do better than that, you should become a stock broker!

CHAPTER VI

ALGEBRA GAMES



"Monkey Puzzle"

```
100 FOR A=3 TO 8
105 PRINT"□"
110 LET B=INT((A-1)*RND(0))+1
120 PRINT A;"MEN COLLECT A PILE OF COCON
UTS THEY"
130 PRINT "INTEND TO SHARE.WHILE THE OTH
ERS SLEEP,"
140 PRINT "EACH MAN IN TURN WAKES UP AND
 DIVIDES"
150 PRINT "THE PILE EVENLY AND HE HAS";B
;"LEFT OVER"
155 PRINT "WHICH HE HANDS OVER TO A MONK
EY,THEN HE"
160 PRINT "HIDES HIS SHARE. NEXT MORNING
 THEY"
170 PRINT "SHARE THE REST OF THE COCONUT
S,GIVING";STR$(B)
175 PRINT "TO THE MONKEY."
176 PRINT
180 PRINT "HOW MANY DID THEY HAVE AT THE
 START?"
190 INPUT C:IF C=0 THEN 190
200 LET B=INT(A↑(A+1)-B*(A-1))
210 LET Z=INT(C/B)-C/B
220 IF Z<>0 THEN 270
230 PRINT "RIGHT.NOW TRY ONE MORE"
240 FOR T=1 TO 2000:NEXT T
250 NEXT A
270 FOR I=1 TO A+1
280 LET S=INT(C/A)
290 PRINT " DIVISION ";I;" IS ";S;"  MON
KEY GETS ";C-A*S
300 LET C=(A-1)*S
310 NEXT I
320 PRINT "YOUR ANSWER IS WRONG"
330 PRINT "IT SHOULD BE";B
340 PRINT:INPUT "PRESS RETURN TO GO ON";
350 PRINT:PRINT AGAIN? PRESS Y OR N"
360 GET A$:IF A$="" THEN 360
```

```
37Ø IF A$="Y" THEN A=A-1:GOTO 25Ø
38Ø STOP
```

Note: The symbol ◘ represents SHIFT CLEAR/HOME.

"Monkey Puzzle"

AIM OF THE GAME

This is a very tricky puzzle. Your ability to work out the answers will not necessarily depend upon the amount of mathematics you have studied. I ran into this problem at a party of university mathematics professors and their wives. None of the wives had any mathematics training beyond high school, and yet the most elegant solution to this puzzle was given by one of the wives, in a matter of a few minutes.

A pile of coconuts is divided up among a group of men in a strange fashion. First of all, one man divides the pile into equal piles and finds out that he has some left over, which he gives to a nearby monkey. (That is where the monkey comes into the story.) The man takes his share away, and heaps the remaining piles back into one pile. Each of the men, in turn, do the same thing. There is always the same number of extra coconuts, which are given to the monkey. Finally, they divide the pile once more, and again there is the same remainder given to the monkey.

The question you have to answer is: How many coconuts were there at the start?

The program provides you with all the information necessary to solve the problem. You start with three men. When you get a correct answer, you progress to more difficult problems.

HOW TO PLAY

When you run the program, a problem is posed. All the information needed to solve the problem is given to you. For instance, the first problem given to you might look like this

```
3 MEN COLLECT A PILE OF COCONUTS THEY
INTEND TO SHARE.WHILE THE OTHERS SLEEP,
EACH MAN IN TURN WAKES UP AND DIVIDES
THE PILE EVENLY AND HE HAS 1 LEFT OVER
WHICH HE HANDS OVER TO A MONKEY,THEN HE
HIDES HIS SHARE. NEXT MORNING THEY
SHARE THE REST OF THE COCONUTS,GIVING 1
TO THE MONKEY.
HOW MANY DID THEY HAVE AT THE START?
```

When you have worked out your answer, you enter it. Suppose you enter the number 121. The computer will work out the division at each stage, and display the remainder to be given to the monkey. This way you

can check the accuracy of your arithmetic. In this case, the computer will display the following:

```
DIVISION 1 IS 4Ø MONKEY GETS 1
DIVISION 2 IS 26 MONKEY GETS 2
DIVISION 3 IS 17 MONKEY GETS 1
DIVISION 4 IS 11 MONKEY GETS 1
YOUR ANSWER IS WRONG.
IT SHOULD BE 79
PRESS RETURN TO GO ON
AGAIN? PRESS Y OR N
```

You can see that the result of the second division is wrong. You also learn what the correct answer should be. You press Y and the program continues by asking you another problem. There are only two problems involving 3 men. In one, the monkey gets 1 coconut, in the other he gets 2. If you get the same problem, you will enter 79 and the computer will respond with:

```
RIGHT.NOW TRY ONE MORE
```

After a short pause, a new problem will be posed involving 4 men. Each time you get a correct answer, the new problem involves one more man. The maximum number of men is 8 because the numbers become too large for your computer with any more men!

THE MATHEMATICS

The solution is provided by the formula in line 2ØØ. Since any multiple of this solution is also a solution, line 21Ø checks whether your answer is a multiple of the solution. You may wonder how I arrived at the solution. Well I used an old mathematical trick of "borrowing" some hypothetical coconuts at the start to make the arithmetic work out exactly, without any remainder, and then the "borrowed" coconuts are returned after the final division is made.

To understand this process, we use a little algebra. We let X be the number of coconuts to start with. In the illustrative example given in HOW TO PLAY, after X has been divided by 3, there is one left over, then X+2 will be exactly divisible by 3 with none left over. So we borrow 2 coconuts to make the stack X+2. Now, after X+2 has been divided into 3 equal piles, each pile contains one more coconut than it should. So the first man, let's call him A, gives his extra one to the monkey. Then he hides his correct amount. The remaining two stacks will still contain 2 more coconuts than they are supposed to. This means that the next division by 3 will also work out with none remaining. This time the second man, call him B, gives his extra one to the monkey. And so on. Every time the division by 3 works out with no remainder, each man hides his correct

amount by giving one to the monkey, so the monkey gets the correct amount. In the last division, each man has an extra coconut. One man gives his to the monkey, the other two return one each of the borrowed coconuts. All the conditions have been satisfied. So X+2 is divisible by 3 a total of 4 times. So X+2 = constant *81. The constant can be 1,2,... . Say it is 1, then X+2=81 or X=79.

This solution is fairly neat, but becomes more difficult to generalize to the case of any number of men M and a remainder of R=1 or 2 or ... M–1. The following is the solution given by the wife of a mathematical colleague of mine.

Consider the number –2. If you divide –2 by 3, you get three equal shares of –1, and +1 left over! Give the +1 to the monkey, take one share of –1 away, and you are left with two shares of –1 for a total of –2. Thus, the number –2 is never changed by the process. Any number that is divisible by 3 a total of 4 times may be added to this –2. So we add $k*3^4=k*81$ where k=1,2,... . For k=1, we have

$$X = -2 + 81 = 79$$

This process is more easily generalized. The monkey gets the positive remainder, while the men each get a share which is the negative of the monkey's share.

Suppose we have M men, and the monkey gets a remainder of R. Then the men each get an equal share of -R. This means the "magic" number is

$$M*(-R) + R = -R*(M-1)$$

In the case worked out above, M=3, R=1 so the "magic" number is $-1*(3-1) = -2$.

Now we add a multiple of M^{M+1} to the "magic" number to get the required formula. The case when the multiple is 1 gives

$$X = M\uparrow(M+1) - R*(M-1)$$

"Obscure Ages"

```
100 PRINT "□"
110 LET P=INT(10*RND(0))+5
120 LET J=P+INT(20*RND(0))
130 LET R=J+INT(20*RND(0))
140 PRINT "RON IS";R-J;"YEARS OLDER THA
N JIM":PRINT
150 PRINT "RON IS";R-P;"YEARS OLDER THA
N PAT":PRINT
160 PRINT "THEIR COMBINED AGE IS";P+J+R:
PRINT
170 PRINT "WHAT ARE THEIR AGES?":PRINT
175 INPUT "ENTER PAT'S AGE";A
180 INPUT "ENTER JIM'S AGE";B
185 INPUT "ENTER RON'S AGE";C:PRINT
190 IF P=A AND J=B AND R=C THEN 220
200 PRINT "WRONG IT IS ";P;",";J;",";R
210 GOTO 230
220 PRINT "RIGHT ON"
230 FOR T=1 TO 2000:NEXT T
240 PRINT:PRINT "AGAIN? PRESS Y OR N"
250 GET A$:IF A$="" THEN 250
260 IF A$="Y" THEN 100
270 STOP
```

Note: The symbol □ represents SHIFT CLEAR/HOME.

"Obscure Ages"

AIM OF THE GAME

You are given information about the ages of RON, JIM and PAT. From this information you are asked to deduce their ages. This game is meant to give you practice at solving a linear system of algebraic equations. When you have worked out their ages, you enter their ages one after the other in the order requested by the prompting message on the screen.

HOW TO PLAY

A typical run starts with the following message

```
RON IS 6 YEARS OLDER THAN JIM
RON IS 24 YEARS OLDER THAN PAT
THEIR COMBINED AGE IS 72
WHAT ARE THEIR AGES?
ENTER PAT'S AGE?
ENTER JIM'S AGE?
ENTER RON'S AGE?
```

When you have worked out their ages, you enter them in the order requested, one after the other. Suppose you enter 1Ø, 28 and 34 respectively. You are given the message

```
RIGHT ON
AGAIN? PRESS Y OR N
```

If you enter an incorrect answer, then you receive the message

```
WRONG IT IS 1Ø,28,34
AGAIN? PRESS Y OR N
```

If you wish to have another go, press the key Y, otherwise press N.

THE MATHEMATICS

The mathematics behind this game is called linear algebra. In this case, you have to solve three linear equations simultaneously. Let us suppose that the ages of PAT, JIM and RON are represented by the letters P, J and R respectively. Then in the game presented in HOW TO PLAY, we conclude from the given data that

$$\begin{aligned} R - J &= 6 \\ R - P &= 24 \\ P + J + R &= 72 \end{aligned}$$

These equations are called linear equations, because of the variables P, J and R are only multiplied by constants and added together.

In order to solve these equations, you may manipulate them in three ways. You may interchange the order of the equations, add a multiple of one equation to another, or multiply any equation by a nonzero number.

In this case if we add the first two equations to the third equation we get.

3R = 1Ø2 so R = 34

We substitute R = 34 in R – J = 6 to get 34 – J = 6, so that J = 28 and similarly in R – P = 24 we get 34 – P = 24 so P = 10.

When you have mastered this process, you might like to try "MORE OBSCURE AGES". First, try to solve the game without reading the section on the mathematics.

"More Obscure Ages"

```
100 PRINT "□"
110 LET R=INT(25*RND(0))+10
120 LET Y=INT(20*RND(0))+2
125 IF Y>=R THEN 120
130 LET Q=INT(5*RND(0))+2
135 LET P=Q*(R-Y)+Y
140 LET Z=INT(5*RND(0))+2
145 IF Q=Z THEN 140
146 LET J=Z*(R-Y)+Y
147 IF P>100 OR J>100 THEN 110
150 PRINT Y;"YEARS AGO PAT WAS";Q;"TIMES
 RON'S AGE":PRINT
155 PRINT "AT THAT TIME.JIM WAS";Z;"TIME
S RON'S AGE.":PRINT
160 PRINT "THEIR COMBINED AGE NOW IS";R+
2*Y+Q*(R-Y)+Z*(R-Y);".":PRINT
170 PRINT "WHAT ARE THEIR AGES?":PRINT
175 INPUT "ENTER PAT'S AGE";A
180 INPUT "ENTER JIM'S AGE";B
185 INPUT "ENTER RON'S AGE";C:PRINT
190 IF A=P AND B=J AND C=R THEN 220
200 PRINT "WRONG IT IS ";P;",";J;",";R
210 GOTO 230
220 PRINT "RIGHT ON MATE!"
230 FOR T=1 TO 2000:NEXT T
240 PRINT:PRINT "AGAIN? PRESS Y OR N"
250 GET A$:IF A$="" THEN 250
260 IF A$="Y" THEN 100
270 STOP
```

Note: The symbol □ represents SHIFT CLEAR/HOME.

"More Obscure Ages"

HOW TO PLAY

This game is similar to "OBSCURE AGES" but is more difficult. A typical run will present you with a problem like this

```
 8 YEARS AGO PAT WAS 6 TIMES RON'S AGE
AT THAT TIME. JIM WAS 5 TIMES RON'S AGE.
THEIR COMBINED AGE NOW IS 84.
WHAT ARE THEIR AGES?
ENTER PAT'S AGE?
ENTER JIM'S AGE?
ENTER RON'S AGE?
```

You enter the ages in the order requested, if you are wrong, the correct answer is given to you. In either case, you are given a prompt to quit or go on to another problem.

THE MATHEMATICS

Once again you must solve linear equations. We let P, J and R be the respective ages of PAT, JIM and RON. Then from the information given we have

$$P - 8 = 6(R - 8) \quad \text{or} \quad P = 6R - 40$$
$$J - 8 = 5(R - 8) \quad \text{or} \quad J = 5R - 32$$

and finally,

$$P + J + R = 84$$

If we substitute for P and J in the final equation, which is like subtracting the first two equations from the third equation, we get

$$12R - 72 = 84 \text{ or } 12R = 156 \text{ or } R = 13$$

Substituting in the first two equations gives $P = 6(13) - 40$ or $P = 38$, and $J = 5(13) - 32 = 33$.

When you have mastered this game, you should try yourself on "MUCH MORE OBSCURE AGES". Try to solve the game without referring to the section on the mathematics.

"Much More Obscure Ages"

```
100 PRINT "□"
110 LET P=INT(25*RND(0))+10
120 LET Q=INT(5*RND(0))+2
130 LET Z=INT(5*RND(0))+2
135 IF Q=Z THEN 120
140 LET J=Q*P
143 LET R=(Z+Q)/2
145 IF INT(R)-R<0 THEN 120
147 LET R=R*P
148 IF P>99 OR J>99 OR R>99 THEN 110
150 PRINT "WHEN RON WAS AS OLD AS JIM IS
 NOW, HE":PRINT
155 PRINT "WAS";Q;"TIMES AS OLD AS PAT I
S NOW.":PRINT
156 PRINT "WHEN JIM IS AS OLD AS RON IS
NOW, RON":PRINT
157 PRINT "WILL BE";Z;"TIMES AS OLD AS P
AT IS NOW.":PRINT
160 PRINT "THEIR COMBINED AGE IS";P+J+R;
".":PRINT
170 PRINT "WHAT ARE THEIR AGES?":PRINT
175 INPUT "ENTER PAT'S AGE";A
180 INPUT "ENTER JIM'S AGE";B
185 INPUT "ENTER RON'S AGE";C:PRINT
190 IF A=P AND B=J AND C=R THEN 220
200 PRINT "WRONG IT IS ";P;",";J;",";R
210 GOTO 230
220 PRINT "VERY CLEVER OF YOU"
230 FOR T=1 TO 2000:NEXT T
240 PRINT:PRINT "AGAIN? PRESS Y OR N"
250 GET A$:IF A$="" THEN 250
260 IF A$="Y" THEN 100
270 STOP
```

Note: The symbol □ represents SHIFT CLEAR/HOME.

"Much More Obscure Ages"

HOW TO PLAY

This game is similar to the previous two games, but more difficult as the name implies. A typical problem looks like this

```
WHEN RON WAS AS OLD AS JIM IS NOW, HE
WAS 2 TIMES AS OLD AS PAT IS NOW.
WHEN JIM IS AS OLD AS RON IS NOW, RON
WILL BE 6 TIMES AS OLD AS PAT IS NOW.
THEIR COMBINED AGE IS 154.
WHAT ARE THEIR AGES?
ENTER PAT'S AGE?
ENTER JIM'S AGE?
ENTER RON'S AGE?
```

As before, you enter the ages in the requested order. If you are wrong, the correct answer is given to you. In any case, a prompt is given to continue with another problem or to quit.

THE MATHEMATICS

As before, we let the ages be represented by P,J and R respectively. From the information given in the problem, we have to introduce another unknown, say D, which is the difference in the ages of RON and JIM. Then we have

$$J = R - D = 2P$$
$$R + D = 6P$$
$$P + J + R = 154$$

If we add the first two equations, the D is eliminated to give 2R=8P or R=4P. We also have J=2P from the first equation, so substituting these into the third equation gives

$$7P=154 \text{ or } P=22$$

So J=44 and R=88.

THE CODE

The essential difficulty in preparing the code was to define the correct quantities. P and the multipliers are defined. J was automatically defined from these quantities. Now came the tricky part. I tried defining a

difference D, but the arithmetic would not work out. The solution was to define R, but only if the sum of the multipliers are even!

Now I leave you with the question, why was this restriction imposed? The multipliers are Q and Z, and R is defined to be

$$R=\frac{(Q + Z)}{2}*P \text{ , (see lines 143 to 147)}$$

GENERAL MATHEMATICS BACKGROUND

The three games involving OBSCURE AGES all involved systems of linear equations, which we solved by manipulating the equations to obtain a simpler equation involving just one unknown. We then substituted the value of this unknown back into other equations.

One of the major uses of computers is to solve very large systems of linear equations that may contain thousands of equations. The equations are stored in matrix form and many methods have been designed to solve these large systems economically and accurately. Numerous books have been written about matrix algebra and linear systems and a lot of research is being done right now on these problems. By the term matrix, we mean an array. For example the linear equations X + Y + Z = 4 , 2X – Y = 3, 2Y – Z = 5 can be represented by the array

1	1	1	4
2	–1	Ø	3
Ø	2	–1	5

Here we drop the X, Y, Z and = . The columns keep track of which symbol belongs to which number. Now in order to solve this system, we first reduce it to an equivalent system that has the same solution, but is simpler to solve. We subtract twice the numbers in the first row from the corresponding numbers in the second row so that the number 2 in the first column becomes Ø. Thus the equivalent system is

1	1	1	4
Ø	–3	–2	–5
Ø	2	–1	5

We now add the numbers in the third row to the corresponding numbers in the second row to get the equivalent system

1	1	1	4
Ø	–1	–3	Ø
Ø	2	–1	5

Now we add twice the numbers in the second row to the corresponding numbers in the third row which reduces the number 2 in the third row, second column to Ø. The equivalent system is

$$\begin{array}{cccc} 1 & 1 & 1 & 4 \\ \emptyset & -1 & -3 & \emptyset \\ \emptyset & \emptyset & -7 & 5 \end{array}$$

Now we solve this system which is called a row reduced system, where the matrix of coefficients is said to be triangular. The method of solution is called back-substitution. We start with the third row, that stands for

$$-7Z = 5$$

The solution is easy. It is $Z = -5/7$. The second to the last row stands for

$$-Y - 3Z = \emptyset \text{ or } Y = -3Z = 15/7$$

Finally, the first row stands for

$$X + Y + Z = 4$$

but now we know what Y and Z are, so

$$X = 4 - Y - Z = 4 - 15/7 + 5/7 = 18/7$$

This process is called Gauss Elimination.

"Product and Sums"

```
100 PRINT "□"
110 LET A=INT(19*RND(0))-9
120 LET B=INT(19*RND(0))-9
130 LET C=INT(19*RND(0))-9
140 IF A>=B OR A>=C OR B>=C THEN 110
150 IF A*B*C=0 THEN 110
160 PRINT "□"
170 PRINT "PRODUCT A*B*C =";A*B*C
180 PRINT:PRINT "SUM A*B+A*C+B*C =";A*B+
A*C+B*C
190 PRINT:PRINT "AND SUM A+B+C =";A+B+C
200 PRINT:PRINT "THE VALUES OF A,B AND C
 ARE FROM -9 TO 9"
205 PRINT "AND A<=B<=C. WHAT ARE A,B AND
C?":PRINT
210 INPUT "ENTER THE VALUE OF A";P
220 IF A<>P THEN 290
230 PRINT "A IS RIGHT"
240 INPUT "ENTER THE VALUE OF B";Q
250 IF B<>Q THEN 290
260 PRINT "B IS RIGHT"
270 INPUT "ENTER THE VALUE OF C";R
280 IF C=R THEN 320
290 PRINT:PRINT "WRONG! PRESS Q TO QUIT
ANY OTHER KEY TO TRY AGAIN.":PRINT
300 GET A$:IF A$="" THEN 300
305 IF A$="Q" THEN PRINT"THE VALUES ARE"
:PRINT A,B,C:PRINT:GOTO 330
310 GOTO 160
320 PRINT:PRINT "RIGHT ON"
330 PRINT "AGAIN? PRESS Y OR N"
340 GET A$:IF A$="" THEN 340
350 IF A$="Y" THEN 100
360 STOP
```

Note: The symbol □ represents SHIFT CLEAR/HOME.

"Product and Sums"

AIM OF THE GAME

This is another game in which you have to determine three unknowns from certain given facts. However, it is different from the "OBSCURE AGES" type of problems because in this case you have to solve a system of nonlinear equations. You may use algebra or arithmetic to solve these problems so this game could be included in the chapter on arithmetic games.

HOW TO PLAY

Prompts provide you with all the instructions necessary. A typical run would look like this

```
PRODUCT A*B*C=336
SUM A*B+A*C+B*C=-5Ø
AND SUM A+B+C=-7
THE VALUES OF A,B AND C ARE FROM -9 TO 9
AND A<=B<=C. WHAT ARE A,B AND C?
ENTER THE VALUE OF A?
```

Suppose you work out that A = –8, B = –6 and C = 7 then you would enter these numbers one by one. After each entry you are given a message. After you enter A you are told

```
A IS RIGHT
```

After you enter B, the message

```
B IS RIGHT
```

appears, and after you enter C you get the message

```
RIGHT ON
AGAIN? PRESS Y OR N
```

If you want another problem, press the Y key. If you wish to quit, just press any other key.

If you enter an incorrect entry then the message WRONG appears and you are prompted to quit or to try the same problem again.

THE MATHEMATICS

The arithmetic way of solving this problem is to find all the factors of the product. In the example given, this would be:

```
336 = 2*2*2*2*3*7
```

Since the product is positive, there has to be either two negative numbers or none. As the sum is negative, there cannot be no negatives, so there must be two. After a little thought, the numbers –8, –6 and 7 are found to fit all the facts given. Here we are guided by the factors, 8 = 2*2*2, 6 = 2*3 and 7 include all the factors in the product. Essentially the solution is obtained by mental trial and error.

The algebraic method is to eliminate some of the unknowns. We multiply the second equation

$$A*B + A*C + B*C = -5Ø$$

by A, to obtain

$$A*A*(B+C) + A*B*C = -5Ø*A$$

Now we substitute for the product A*B*C = 336, and from the sum, A + B + C = –7, we have that B + C = –7 – A. Thus the equation becomes

$$A*A*(-7-A) + 336 = -5Ø*A$$

or

$$A^3 + 7A^2 - 5ØA - 336 = Ø$$

which is a single cubic (degree three) equation in A. You will notice that the coefficients in this equation are the numbers (except for signs) that appeared in the statement of the problem. The reason for this is that the cubic equation for A contains all the information about the three unknowns. The three roots of this equation will represent the three values of A, B and C.

If we consider the three factors (X–A), (X–B) and (X–C) and multiply them together we get

$$(X-A)(X-B)(X-C)$$

Now if we expand the product we get

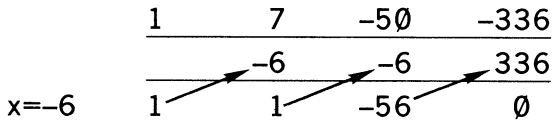
$$X^3 - (A+B+C)X^2 + (AB+AC+BC)X - ABC$$

If we substitute the values given for the sums and products we get

$$X^3 + 7X^2 - 5ØX - 336$$

The roots of this polynomial are the values A, B, and C that we seek.

By trial and error we first substitute X = 1, –1, 2, –2 and so on until we get X= –6. We find that the expression is zero. This means that one of the roots is –6, and one of the factors is X+6. We now divide the factor (X+6) into $X^3 + 7X^2 - 5ØX - 336$ using the process of synthetic division described in "YOUR NUMBER IS UP".



This means $X^3 + 7X^2 - 5ØX - 336 = (X+6)(X^2+X-56)$
The factors of $X^2 + X - 56 = (X+8)(X-7)$
Thus the required factors are found, and the roots are –8, 7, –6. The order required by the program makes A=–8, B=–6 and C=7.

THE CODE

For a more challenging game of product and sums, you might like to alter the definitions of A, B and C in lines 11Ø to 13Ø to allow larger numbers.

Note: To find out more about roots and root finding, check into the programs "FIXED-POINT" and "BRACKET" in Chapter VII of this book.

"Rally"

```
100 PRINT "□"
110 LET R=INT(9*RND(0))+5
120 LET C=R+INT(15*RND(0))+5
125 IF C<R THEN 110
130 LET I=INT(80*RND(0))+20
140 LET J=INT(80*RND(0))+20
150 LET T=I+J
160 LET D=R*I+C*J
165 IF D>1000 THEN 110:REM LIMIT TOTAL D
ISTANCE TO 1000 MILES
170 PRINT "YOU CAN RUN AT A SPEED OF";R;
"M.P.H.":PRINT
180 PRINT "AND YOU CAN CYCLE AT";C;"M.P.
H. HOW LONG":PRINT
190 PRINT "WOULD YOU RUN AND HOW LONG WO
ULD YOU":PRINT
195 PRINT "CYCLE SO AS TO COVER ";D;"MIL
ES IN";T:PRINT
196 PRINT "HOURS EXACTLY?":PRINT
200 INPUT "ENTER RUN TIME";A:INPUT "ENTE
R CYCLE TIME";B;PRINT
210 IF A=I AND B=J THEN PRINT "RIGHT"
220 IF A<>I OR B<>J THEN PRINT "WRONG. I
T IS";I;",";J
230 PRINT "AGAIN? PRESS Y OR N"
240 GET A$:IF A$="" THEN 240
250 IF A$="Y" THEN 100
260 STOP
```

Note: The symbol □ represents SHIFT CLEAR/HOME.

"Rally"

AIM OF THE GAME

This is a simple game where you are asked to cover a given distance in an exact time. You may run or ride a bicycle. The speeds for both modes of travel are given to you.

This is a quick and easy game to play and is designed to shape up your ability to do mental arithmetic. It involves the solution of linear equations, only this time you have two equations in two unknowns.

HOW TO PLAY

A hypothetical run could go like this. You are presented with a problem

```
YOU CAN RUN AT A SPEED OF 1 M.P.H.
AND YOU CAN CYCLE AT 3 M.P.H. HOW LONG
WOULD YOU RUN AND HOW LONG WOULD YOU
CYCLE SO AS TO COVER 264 MILES IN 118
HOURS EXACTLY?
ENTER RUN TIME?
ENTER CYCLE TIME?
```

Suppose you work out the run time to be 45 hours and the cycle time to be 73 hours. You would enter 45 and then 73. You are correct, and the computer responds with the message

```
RIGHT
AGAIN? PRESS Y OR N
```

If you want to try another problem, press the Y key. Press the N key to quit.

If you enter a wrong answer, you will be given the correct answer and prompted to continue or to quit.

THE MATHEMATICS

If we let the run time be R and the cycle time be C, then from the information given in the sample problem, we can write down two equations. They are

(1) $R + C = 118$
(2) $R + 3C = 264$

This system is particularly simple. We subtract equation (1) from equation (2) to obtain $2C = 146$ or $C = 73$. Substituting this into (1) gives us $R + 73 = 118$ and so $R = 45$.

You will always find that the run speed is at least 5 M.P.H. In that case you might end up with equations like

$$\begin{aligned} R + C &= 88 \\ 5R + 8C &= 644 \end{aligned}$$

This time we subtact 5 times the first equation from the second to get $3C = 2\emptyset 4$ or $C = 68$ and so $R = 2\emptyset$.

CHAPTER VII
ROOT FINDERS

$$A*B*C=-4$$

$$A*B+A*B+B*C=-4$$

$$A+B+C=1$$

$$A=?\ B=?\ C=?$$

"Fixed-Point"

```
100 DEF FNG(X)=-(P*X*X+Q*X+R)/(X*X)
110 PRINT "□": LET I=0
120 PRINT "DEFINE FUNCTION X=G(X) IN LIN
E 100"
130 PRINT:PRINT "NUMBER OF SIGNIFICANT D
IGITS? ENTER 1-8"
140 INPUT S
150 LET S=.5*10↑(1-S)
160 PRINT:PRINT "ENTER P,Q,R"
170 INPUT P,Q,R
180 PRINT:PRINT "ENTER INITIAL VALUE OF
THE ROOT"
190 INPUT X1
200 PRINT "□":PRINT "ITERATION ROOT  DIG
ITS"
210 LET I=I+1
220 LET X2=FNG(X1)
230 LET A=ABS((X2-X1)/X2)
240 PRINT I;TAB(5);X2;TAB(14);INT(1-LOG(
2*A)/LOG(10))
250 LET T=ABS((FNG(X2)-FNG(X1))/(X2-X1))
260 IF T>=1 THEN PRINT "NOT CONVERGING"
270 IF A<=S THEN PRINT "CONVERGED.ANOTHE
R ROOT? PRESS Y OR N":GOTO 300
280 LET X1=X2
290 GOTO 210
300 GET Z$:IF Z$="" THEN 300
310 IF Z$="Y" THEN LET I=0:PRINT "□":GOT
O 180
320 STOP
```

Note: The symbol □ represents SHIFT CLEAR/HOME.

"Fixed-Point"

This program and the next, "BRACKET", are not games. They are strictly mathematical problems, although experimenting with different numerical methods can be a sort of game. The idea is to find out which method works best on a given problem, and what pitfalls lurk in a method to trap the unwary.

Solving the equation F(X) = Ø is a very important problem. These values of X are referred to as zeros or roots of the function F. You came across this problem in "PRODUCTS AND SUMS".

There are many different numerical methods for finding zeros, but they do not always work equally well. In fact some very good, fast methods, like Newton's method, can fail miserably at times.

The method used in this program is not fast. It often fails to converge, but it is widely used. It is extremely simple to program, and has important generalizations to the problem of solving large systems of linear equations by iteration.

THE MATHEMATICS

The idea is to change the equation F(X) = Ø to an equation X = G(X). A fixed-point of the function G is a value of X such that X = G(X), then X – G(X) = F(X) = Ø and the fixed-point of G is a zero of F.

In "PRODUCTS AND SUMS", we needed to find a zero of the polynomial

$$F = X^3 + PX^2 + QX + R$$

In this case, P, Q and R are my names for the numbers that are the coefficients. There are many ways of constructing fixed-point expressions. Here are just two.

(1) We write F(X) = Ø as $X^3 = -(PX^2 + QX + R)$, then divide by X^2 to get

$$X = -(PX^2 + QX + R)/X^2$$

Thus, we define

```
G(X) = -(P*X*X + Q*X + R)/(X*X)
```

This is done in the program listing.

(2) We take a cube root in the expression above to get

$$X = -(PX^2 + QX + R) \uparrow (1/3)$$

In this case we define

```
G(X) = -(ABS(P*X*X + Q*X + R)) ↑(1/3)
```

Having constructed suitable fixed-point expressions, we now generate a sequence of numbers that we hope, will converge to a root.

We start with any approximate value of the root, call it X_0, in the case of the expression in (1), we should avoid the value of Ø. Any other number will do. Then, the program defines the sequence

$$X_1 = G(X_0)$$

$$X_2 = G(X_1)$$

$$X_3 = G(X_2)$$

.
.
.

and so on. Provided the rate of change in G is less than 1, in absolute value, this simple sequence will converge. The code estimates the rate of change using line 25Ø, which checks the value

$$ABS((G(X_2) - G(X_1))/(X_2 - X_1))$$

and gives a message

NOT CONVERGING

if this expression is not less than 1.

We must provide a stopping rule. The one in this program is based upon an estimate of the number of correct significant figures in the last iteration. The rule is given in the following.

If X is the exact value, and $\overline{X}$ an approximate value of X, then there are said to be S significant figures in $\overline{X}$ provided

$$ABS\left(\frac{X - \overline{X}}{X}\right) <= Ø.5*1Ø\uparrow(1-S)$$

Of course we do not know what X is, so in place of $\frac{X-\overline{X}}{X}$ we use the difference in the last two iterates, divided by the last iterate. This usually causes the program to stop too soon, so you should ask for more significant figures than you really need.

HOW TO PLAY

Messages and prompts give you all the instructions that you need. However, you will need to have some values of P, Q and R. I suggest you use the values given in the program "PRODUCTS AND SUMS". In which case P = 7, Q = –5Ø and R = –336. In every case the R in this program is – (PRODUCT), Q = (SUM OF PRODUCTS), and P = – (SUM). You can make up your own values, like 3,4 and 5, or 6,7 and 8 etc. Then experiment with different starting values.

I encourage you to experiment with different choices of the function G(X). Simply change the definition in line 1ØØ. You might like to try the simple quadratic, $PX^2 + QX + R = Ø$ and find a zero by constructing a fixed-point function G.

Note: If you are still reading this far, I want to encourage you to investigate the most important application of this method to iterative methods for large systems of linear equations.

We write a system of linear equations in matrix form as

$$AX = B$$

where A is a matrix with DIM A(N,M) say. X and B are vectors, DIM X(M) and DIM B(N). In this case we have N linear equations in M unknowns.

Instead of solving $AX - B = Ø$ for a zero, we split the matrix A, and form a fixed-point iteration scheme

$$AX = B \text{ becomes } X = D - GX$$

We then form the iterates X_0, X_1, ... starting from any initial X_0, and using

$$X_{I+1} = D - GX_I$$

This scheme will converge provided the rate of change, measured by the norm of G, is less than 1.

"Bracket"

```
100 DEF FNF(X)=X*X*X+P*X*X+Q*X+R
110 PRINT "□":LET I=0
120 PRINT "DEFINE FUNCTION F(X)=0 IN LIN
E 100"
130 PRINT:PRINT "NUMBER OF SIGNIFICANT D
IGITS? ENTER 1-8"
140 INPUT S
150 LET S=.5*10↑(1-S)
160 PRINT:PRINT "ENTER P,Q,R"
170 INPUT P,Q,R
180 PRINT:PRINT "ENTER X1,X2 THAT BRACKE
T THE ROOT"
190 INPUT X1,X2
200 LET X3=(X1-X2)/2
205 PRINT:PRINT "ENTER 1 FOR BISECTION M
ETHOD":PRINT
206 PRINT "OR 2 FOR REGULAR FALSI":INPUT
 M
210 PRINT "ITERATION ROOT  DIGITS"
220 LET F1=FNF(X1)
230 LET F2=FNF(X2)
235 IF SGN(F1)=SGN(F2) THEN PRINT "ROOT
 NOT BRACKETED":PRINT
236 IF SGN(F1)=SGN(F2) THEN PRINT "CONT
INUE? PRESS Y OR N":GOTO 340
240 LET I=I+1
250 LET X4=(X1+X2)/2:IF M=2 THEN LET X4=
(F1*X2-F2*X1)/(F1-F2)
260 LET F3=FNF(X4)
270 IF SGN(F3)=SGN(F1) THEN LET X1=X4
280 IF SGN(F3)=SGN(F2) THEN LET X2=X4
285 IF X4=0 THEN LET X4=.1E-10
290 LET A=ABS((X3-X4)/X4)
300 PRINT I;TAB(5);X4;TAB(14);INT(1-LOG(
2*A)/LOG(10))
310 LET X3=X4
320 IF A<=S THEN PRINT "CONVERGED.ANOTHE
R ROOT? PRESS Y OR N":GOTO 340
330 GOTO 220
340 GET Z$:IF Z$="" THEN 340
```

```
35Ø IF Z$="Y" THEN LET I=Ø:PRINT "□":GOT
O 18Ø
36Ø STOP
```

Note: The symbol □ represents SHIFT CLEAR/HOME.

"Bracket"

If you have not tried "FIXED-POINT", please do so before running this program.

HOW TO PLAY

Messages and prompts give you all the instructions that you need. As in the program "FIXED-POINT", you will need to enter values of P, Q and R that define a cubic $X^3 + PX^2 + QX + R$ for which you wish to find a zero. An essential difference in this program is that you must enter two initial values. These two values must bracket the zero. This means one must be less than the zero, while the other must be more. If you have no idea where the zero is, you might try -1ØØ,ØØØ and 1ØØ,ØØØ. This is crude, but it works.

This program provides you with a menu of two different methods. They are very similar in nature. The difference will be explained in the mathematics section. Simply experiment by choosing either of the two methods.

If you fail to choose initial values that bracket a root, then a message will be given to you prompting for new initial values.

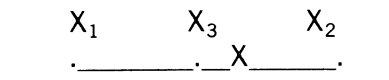
THE MATHEMATICS

There are two methods used in this program. The first method is called the method of BISECTION. It is a very simple idea, and it always works, provided the function is continuous. Polynomials are always continuous, so it will always find the real zeros of polynomials.

We wish to find a zero of the function F(X) that lies between two values X_1 and X_2. We simply evaluate the function at both X_1 and X_2, say $F_1 = F(X_1)$ and $F_2 = F(X_2)$. Now we choose the mid-point of the interval $[X_1, X_2]$. That is, let

$X_3 = (X_1 + X_2)/2$. We find $F_3 = F(X_3)$.

If the sign of F_3 is the same as the sign of F_1, we replace X_1 by X_3, otherwise we replace X_2 by X_3.



By doing this we have halved the width of the interval that contains the required zero. After 1Ø such halvings, we will have reduced the interval by a factor of 1Ø24!

This method is guaranteed to work. What is more, you will be able to locate all the zeros of the function F by this method. With "FIXED-POINT" you could only locate one zero, if you could find any at all!

The second method in this program is called REGULAR FALSI or the method of false position. It is similar to the method of BISECTION, but it tries to do better than just select the mid–point of the interval $[X_1, X_2]$. If the magnitude of F_1 is much greater than F_2, it stands to reason that the zero is more likely to be closer to F_2. So the new point X_3 is chosen as a weighted average of the points X_1 and X_2. In line 25Ø, we define the new point as

$$(F_1*X_2 - F_2*X_1)/(F_1 - F_2)$$

The reason for the subtractions is that F_1 and F_2 have opposite signs, so in effect, we will now be adding quantities with the same sign, (unless the values of X have different signs, in which case X_3 should be nearer Ø).

Note: This is a very powerful program for finding the zeros of any kind of function, not just polynomials. It has only one fault. That is, it can only find real roots. If roots are complex, then there are other methods designed for just that purpose.

PROBLEMS

Here are some problems for you to experiment with.

1. Find a root of $X^3-X-1=Ø$ in the interval [1,2] to an accuracy of 5 significant figures.
2. Find a root of $X-2^{-X}=Ø$ in the interval [Ø,1] to an accuracy of 5 significant figures.
3. Solve the equations
   (a) $3X = 2-3^X+X^2$
   (b) $X = \cos X$
   (c) $2\sin X = X$ near $X = 1.7$
   (d) $X = \tan X$ near $X = 1ØØ$ (difficult to bracket the root of X-tan X near X = 1ØØ, and gets considerably worse at 1ØØØ, 1Ø,ØØØ and so on).

Appendix

These utility programs have been written in order to save you time. The first two programs take all the work out of loading and running the programs in this book from disk, not that this task is particularly arduous! The music maker program helps you create your own tunes which you can add to the listings of the games in this book.

The first two programs, "DISK LOAD", and "LOADER", are meant to be used only if you have a disk drive. The idea is that once you have loaded the program "DISK LOAD", and typed in RUN, then without interrupting the execution, you can load and run programs continuously. These programs were written for the floppy disk that can be purchased with this book. If you have not purchased the prerecorded disk, and want to store some or all of the games in this book on your own floppy disk, you will find these programs very useful.

The first program, called "DISK LOAD", is not essential. This simply turns on different colors for the border and the screen and gives the credits. If you do not like my choice of colors, you should experiment with the settings in lines 23Ø to 3ØØ. "DISK LOAD" is used to load the program that does all the work. This program is called "LOADER".

When you run "DISK LOAD" it stops with the message

```
LOAD"LOADER",8
HIT RETURN AND THEN TYPE RUN
```

The curser is poised on the LOAD command so that when you press RETURN, the program, "LOADER" is loaded into the computer. You then type in RUN to execute "LOADER".

The program "LOADER" displays the chapters of the book, and the games in each chapter. When you make a selection, say the game "SIMON", the program stops with the message

```
LOAD"SIMON",8
PRESS RETURN
```

The curser is positioned on the LOAD command so that when you press the RETURN key, "SIMON" will be loaded.

In order to make an endless chain of loadings, you must alter every STOP statement in each game program. For example, in "SIMON" you must change line 3Ø5 from

```
3Ø5 STOP
```

to

```
3Ø5 LOAD"DISK LOAD",8
```

If you do not like a particular choice of color combinations that is generated in "DISK LOAD", you can always return to the standard colors by pressing the RESTORE key while you hold down the RUN/STOP key.

"Disk Load"

```
100 PRINT "□"
110 FOR I=1 TO 10:GOSUB 200
120 POKE 53280,B:REM: SETS BORDER COLOR
130 POKE 53281,S:REM: SETS SCREEN COLOR
140 FOR T=1 TO 99:NEXT T:REM PAUSES
150 NEXT I
160 PRINT:PRINT:PRINT CHR$(W);TAB(10);"R
OBERT J. BRADY CO."
165 PRINT:PRINT TAB(5);"A PRENTICE-HALL
PUBLISHING AND"
170 PRINT TAB(8);"COMMUNICATIONS COMPANY
"
175 PRINT:PRINT:PRINT TAB(15);"PRESENTS"
180 PRINT:PRINT:PRINT TAB(5);"BRAIN GAME
S FOR KIDS AND ADULTS"
185 PRINT:PRINT:PRINT TAB(18);"BY"
190 PRINT:PRINT TAB(11);"JOHN STEPHENSON
"
191 PRINT:PRINT TAB(15);"(C) 1984"
195 PRINT:PRINT:PRINT:PRINT "LOAD ";CHR$
(34);"LOADER";CHR$(34);",8"
196 PRINT:PRINT "  HIT RETURN AND THEN T
YPE RUN[S]":FOR I=1 TO 13:PRINT:NEXT:STOP
200 REM: THIS CHOOSES B=BORDER COLOR,S=S
CREEN COLOR AND W=WRITE COLOR
210 P=INT(8*RND(0))+1
220 ON P GOTO 230,240,250,260,270,280,29
0,300
230 B=3:S=10:W=28:RETURN
240 B=10:S=3:W=31:RETURN
250 B=7:S=3:W=31:RETURN
260 B=13:S=10:W=28:RETURN
270 B=6:S=3:W=31:RETURN
280 B=3:S=14:W=31:RETURN
290 B=10:S=0:W=150:RETURN
300 B=14:S=7:W=149:RETURN
```

Note: [S] in line 196 is the key CLEAR/HOME and sends the curser scurrying back to the top left hand corner of the screen. Do not use SHIFT and CLEAR/HOME as this would clear the screen as well.

"Loader"

```
100 PRINT "□"
110 Z=1:REM Z CONTROLS THE REPEAT DISPLA
Y IN CASE AN ENTRY IS NOT VALID
120 PRINT "THE CHAPTERS ARE:"
130 PRINT
140 PRINT " 1.CONCENTRATION GAMES"
150 PRINT " 2.WORD GAMES"
160 PRINT " 3.LOGIC AND STRATEGY GAMES"
170 PRINT " 4.ARITHMETIC GAMES"
180 PRINT " 5.GEOMETRY GAMES"
190 PRINT " 6.ALGEBRA GAMES"
195 PRINT " 7.ROOT FINDERS"
196 PRINT " 8.APPENDIX"
200 PRINT:PRINT "TO LIST GAMES PRESS A N
UMBER 1 TO 8"
210 PRINT:PRINT "PRESS S TO STOP"
220 GET A$:IF A$="" THEN 220
230 IF A$="S" THEN STOP
240 P=INT(VAL(A$)):REM P IS THE VALUE OF
 THE INPUT AND CONTROLS THE GOTO IN 200
250 IF P>0 AND P<9 THEN ON P GOTO 1000,2
000,3000,4000,5000,6000,7000,8000
255 REM:CONTROL COMES HERE WHEN ENTRY IS
 NOT VALID
260 PRINT "WRONG ENTRY TRY AGAIN"
270 FOR T=1 TO 1000:NEXT T:ON Z GOTO 100
,300:REM WAITS WHILE 260 IS VIEWED
300 PRINT "□"
310 Z=2
320 IF P<8 THEN PRINT "THE GAMES IN CHAP
TER ";A$;" ARE:":PRINT
325 IF P=8 THEN PRINT "THE APPENDIX PROG
RAMS ARE:":PRINT
330 FOR I=1 TO N
340 PRINT STR$(I);".";N$(I);REM THIS LIS
TS THE GAMES IN A CHAPTER
350 NEXT I
360 PRINT:PRINT "TO LOAD PRESS A NUMBER
1 TO ";N
370 PRINT:PRINT "PRESS S TO STOP, OR C T
```

```
O VIEW CHAPTERS"
380 GET C$:IF C$="" THEN 380
390 IF C$="S" THEN STOP
400 IF C$="C" THEN 100
410 Q=INT(VAL(C$))
420 IF Q>0 AND Q<N+1 THEN 440
430 GOTO 260
440 PRINT "□":REM NORMALIZE SCREEN
445 POKE 53280,14:POKE 53281,6:PRINT CHR
$(154)
450 REM:LOADING INSTRUCTIONS ARE HERE
460 PRINT "□":PRINT:PRINT:PRINT
470 PRINT "LOAD ";CHR$(34);N$(Q);CHR$(34
);",8"
480 PRINT:PRINT"PRESS RETURN";"Ⓢ"
490 STOP
999 REM:GAMES DATA IS LOADED INTO N$ BY
THESE SUBROUTINES
1000 N=5
1010 N$(1)="CONCENTRATION"
1020 N$(2)="FLASH"
1030 N$(3)="KNIGHT"
1040 N$(4)="KNIGHTMARE"
1050 N$(5)="SIMON"
1060 GOTO 300
2000 N=6
2010 N$(1)="ANAGRAMS I"
2020 N$(2)="ANAGRAMS II"
2030 N$(3)="HANGMAN"
2035 N$(4)="FIND A WORD"
2040 N$(5)="ENIGMA I"
2050 N$(6)="ENIGMA II"
2060 GOTO 300
3000 N=8
3010 N$(1)="INVADER"
3020 N$(2)="PROBABILITY"
3030 N$(3)="SEA HUNT"
3040 N$(4)="VERIFY"
3050 N$(5)="LOOSE CHANGE"
3060 N$(6)="NIM II"
3070 N$(7)="NIM III"
3080 N$(8)="TOWERS OF HANOI"
3090 GOTO 300
4000 N=6
```

```
4Ø1Ø N$(1)="BARTER"
4Ø2Ø N$(2)="BASE"
4Ø3Ø N$(3)="YOUR NUMBER IS UP"
4Ø4Ø N$(4)="CRYPTIC"
4Ø5Ø N$(5)="HUMAN ARITHMETIC"
4Ø6Ø N$(6)="JUGS"
4Ø7Ø GOTO 3ØØ
5ØØØ N=5
5Ø1Ø N$(1)="DETONATION"
5Ø2Ø N$(2)="QUEST"
5Ø3Ø N$(3)="TREK"
5Ø4Ø N$(4)="U-BOAT"
5Ø5Ø N$(5)="WARP"
5Ø6Ø GOTO 3ØØ
6ØØØ N=6
6Ø1Ø N$(1)="MONKEY PUZZLE"
6Ø2Ø N$(2)="OBSCURE AGES"
6Ø3Ø N$(3)="MORE OBSCURE AGES"
6Ø4Ø N$(4)="MUCH MORE OBSCURE AGES"
6Ø5Ø N$(5)="PRODUCT AND SUMS"
6Ø6Ø N$(6)="RALLY"
6Ø7Ø GOTO 3ØØ
7ØØØ N=2
7Ø1Ø N$(1)="FIXED-POINT"
7Ø2Ø N$(2)="BRACKET"
7Ø3Ø GOTO 3ØØ
8ØØØ N=3
8Ø1Ø N$(1)="DISK LOAD"
8Ø2Ø N$(2)="LOADER"
8Ø3Ø N$(3)="MUSIC MAKER"
8Ø4Ø GOTO 3ØØ
```

If you would like to add a little music to the program "LOADER", I would suggest that you add the lines before 1ØØ and after 9ØØØ in the following long version of "LOADER". You will also need to use lines 22Ø and 38Ø from this version.

The rest of this long version is included so that you never have to press RETURN and type RUN for each program that is loaded. When you RUN this long version, each program is loaded and run automatically without stopping.

You will need to change "DISK LOAD". Replace lines 195 and 196 with

```
195 LOAD "LOADER",8
```

In order to run continuously, there is an essential trick which you should know about. The program "DISK LOAD" is much shorter than the program "LOADER". This means that when you run "DISK LOAD", it will not be able to execute line 17Ø safely. The program "LOADER" will not be completely loaded.

The trick is to fool the computer into believing that "DISK LOAD" is a lot bigger than it really is! You do this by saving "DISK LOAD" in the following way. You temporarily insert the line

```
1 LOAD "DISK LOAD",8
```

in the program "LOADER" and then you run "LOADER". You should press RUN/STOP and RETURN to halt the execution of "DISK LOAD", otherwise you will lose it because it loads "LOADER" again. (You could prevent this by changing line 17Ø in "DISK LOAD" into a REM statement temporarily.)

When you have the program "DISK LOAD" in the computer memory by this means, it uses up the same storage as the much bigger program "LOADER". If you now save this version on a disk, it will take up as much disk memory as "LOADER" does. Delete the temporary changes and save the final correct versions on the disk. Now you are all set for continuous loading and running. However, all the game programs in this book must be saved in the same fashion. First you load the programs using "LOADER", then stop the execution by pressing the key RUN/STOP in answer to a program prompt (when a GET statement is being executed). Then you save this version of the game program. It will take as much memory as "LOADER".

"Loader" (With Music)

```
5 REM:SET UP ORGAN LIKE VOICES
10 FOR L=54272 TO 54296:POKE L,0:NEXT
15 POKE 54296,15
20 FOR I=1 TO 3:V=54269+7*I:POKE V+1,190
:POKE V+2,250:NEXT I
100 PRINT "□"
110 Z=1:REM Z CONTROLS THE REPEAT DISPLA
Y IN CASE AN ENTRY IS NOT VALID
115 Q=0:REM USED FOR STOP IN LINE 460
120 PRINT "   THE CHAPTERS ARE:"
130 PRINT
140 PRINT " 1.CONCENTRATION GAMES"
150 PRINT " 2.WORD GAMES"
160 PRINT " 3.LOGIC AND STRATEGY GAMES"
170 PRINT " 4.ARITHMETIC GAMES"
180 PRINT " 5.GEOMETRY GAMES"
190 PRINT " 6.ALGEBRA GAMES"
195 PRINT " 7.ROOT FINDERS"
196 PRINT " 8.APPENDIX"
200 PRINT:PRINT "TO LIST GAMES PRESS A N
UMBER 1 TO 8"
210 PRINT:PRINT "PRESS S TO STOP"
220 GET A$:IF A$="" THEN 9000
230 IF A$="S" THEN 440
240 P=INT(VAL(A$)):REM P IS THE VALUE OF
 THE INPUT AND CONTROLS THE GOTO IN 200
250 IF P>0 AND P<9 THEN ON P GOTO 1000,2
000,3000,4000,5000,6000,7000,8000
255 REM:CONTROL COMES HERE WHEN ENTRY IS
 NOT VALID
260 PRINT "WRONG ENTRY TRY AGAIN"
270 FOR T=1 TO 1000:NEXT T:ON Z GOTO 100
,300:REM WAITS WHILE 260 IS VIEWED
300 PRINT "□"
310 Z=2
320 IF P<8 THEN PRINT "THE GAMES IN CHAP
TER ";A$;" ARE:":PRINT
325 IF P=8 THEN PRINT "THE APPENDIX PROG
RAMS ARE:":PRINT
330 FOR I=1 TO N
```

```
340 PRINT STR$(I);".";N$(I);REM THIS LIS
TS THE GAMES IN A CHAPTER
350 NEXT I
360 PRINT:PRINT "TO LOAD PRESS A NUMBER
1 TO ";N
370 PRINT:PRINT "PRESS S TO STOP, OR C T
O VIEW CHAPTERS"
380 GET C$:IF C$="" THEN 9000
390 IF C$="S" THEN 440
400 IF C$="C" THEN 100
410 Q=INT(VAL(C$))
420 IF Q>0 AND Q<N+1 THEN 440
430 GOTO 260
440 PRINT "□":REM NORMALIZE SCREEN
445 POKE 53280,14:POKE 53281,6:PRINT CHR
$(154)
450 FOR I=1 TO 3:POKE 54269+7*I,0NEXT:R
EM TURNS OFF MUSIC
460 IF Q=0 THEN STOP
480 PRINT "LOADING ";N$(Q)
490 REM:LOADING INSTRUCTIONS ARE HERE
500 ON P GOTO 550,650,600,700,750,800,85
0,900
550 ON Q GOTO 555, 560,565,570,575
555 LOAD "CONCENTRATION",8
560 LOAD "FLASH",8
565 LOAD "KNIGHT",8
570 LOAD "KNIGHTMARE",8
575 LOAD "SIMON",8
600 ON Q GOTO 605,610,615,620,625,630,63
5,640
605 LOAD "INVADER",8
610 LOAD "PROBABILITY",8
615 LOAD "SEA HUNT",8
620 LOAD "VERIFY",8
625 LOAD "LOOSE CHANGE",8
630 LOAD "NIM II",8
635 LOAD "NIM III",8
640 LOAD "TOWERS OF HANOI",8
650 ON Q GOTO 655,660,665,670,675,680
655 LOAD "ANAGRAMS I",8
660 LOAD "ANAGRAMS II",8
665 LOAD "HANGMAN",8
670 LOAD "FIND A WORD",8
```

```
675 LOAD "ENIGMA I",8
680 LOAD "ENIGMA II",8
700 ON Q GOTO 705,710,715,720,725,730
705 LOAD "BARTER",8
710 LOAD "BASE",8
715 LOAD "YOUR NUMBER IS UP",8
720 LOAD "CRYPTIC",8
725 LOAD "HUMAN ARITHMETIC",8
730 LOAD "JUGS",8
750 ON Q GOTO 755,760,765,770,775
755 LOAD "DETONATION",8
760 LOAD "QUEST",8
765 LOAD "TREK",8
770 LOAD "U-BOAT",8
775 LOAD "WARP",8
800 ON Q GOTO 805,810,815,820,825,830
805 LOAD "MONKEY PUZZLE",8
810 LOAD "OBSCURE AGES",8
815 LOAD "MORE OBSCURE AGES",8
820 LOAD "MUCH MORE OBSCURE AGES",8
825 LOAD "PRODUCT AND SUMS",8
830 LOAD "RALLY",8
850 ON Q GOTO 855,860
855 LOAD "FIXED-POINT",8
860 LOAD "BRACKET",8
900 ON Q GOTO 905,910,915
905 LOAD "DISK LOAD",8
910 LOAD "LOADER",8
915 LOAD "MUSIC MAKER",8
999 REM:GAMES DATA IS LOADED INTO N$ BY
THESE SUBROUTINES
1000 N=6
1010 N$(1)="CONCENTRATION"
1020 N$(2)="FLASH"
1030 N$(3)="KNIGHT"
1040 N$(4)="KNIGHTMARE"
1050 N$(5)="SIMON"
1060 GOTO 300
2000 N=5
2010 N$(1)="ANAGRAMS I"
2020 N$(2)="ANAGRAMS II"
2030 N$(3)="HANGMAN"
2035 N$(4)="FIND A WORD"
2040 N$(5)="ENIGMA I"
```

```
2050 N$(6)="ENIGMA II"
2060 GOTO 300
3000 N=8
3010 N$(1)="INVADER"
3020 N$(2)="PROBABILITY"
3030 N$(3)="SEA HUNT"
3040 N$(4)="VERIFY"
3050 N$(5)="LOOSE CHANGE"
3060 N$(6)="NIM II"
3070 N$(7)="NIM III"
3080 N$(8)="TOWERS OF HANOI"
3090 GOTO 300
4000 N=6
4010 N$(1)="BARTER"
4020 N$(2)="BASE"
4030 N$(3)="YOUR NUMBER IS UP"
4040 N$(4)="CRYPTIC"
4050 N$(5)="HUMAN ARITHMETIC"
4060 N$(6)="JUGS"
4070 GOTO 300
5000 N=5
5010 N$(1)="DETONATION"
5020 N$(2)="QUEST"
5030 N$(3)="TREK"
5040 N$(4)="U-BOAT"
5050 N$(5)="WARP"
5060 GOTO 300
6000 N=6
6010 N$(1)="MONKEY PUZZLE"
6020 N$(2)="OBSCURE AGES"
6030 N$(3)="MORE OBSCURE AGES"
6040 N$(4)="MUCH MORE OBSCURE AGES"
6050 N$(5)="PRODUCT AND SUMS"
6060 N$(6)="RALLY"
6070 GOTO 300
7000 N=2
7010 N$(1)="FIXED-POINT"
7020 N$(2)="BRACKET"
7030 GOTO 300
8000 N=3
8010 N$(1)="DISK LOAD"
8020 N$(2)="LOADER"
8030 N$(3)="MUSIC MAKER"
8040 GOTO 300
```

```
9000 REM:THIS PLAYS THE MUSIC FROM DATA
9005 FOR I=1 TO 3:READ H(I),L(I):NEXT I:
READ D
9010 IF D=-1 THEN RESTORE:GOTO 9000
9020 FOR I=1 TO 3:V=54266+7*I:POKE V,H(I
):POKE V-1,L(I):NEXT I
9030 FOR I=1 TO 3:POKE 54269+7*I,33:NEXT
9040 FOR T=1 TO D:NEXT
9050 ON Z GOTO 220,380
9900 DATA 17,37,0,0,0,0,75
9905 DATA 16,47,0,0,0,0,75
9910 DATA 17,37,0,0,0,0,200
9912 DATA 12,216,0,0,0,0,200
9915 DATA 13,156,0,0,0,0,200
9917 DATA 17,37,0,0,0,0,75
9920 DATA 16,47,0,0,0,0,75
9922 DATA 17,37,0,0,0,0,200
9925 DATA 19,63,0,0,0,0,200
9927 DATA 12,216,0,0,0,0,200
9930 DATA 17,37,0,0,0,0,75
9932 DATA 16,47,0,0,0,0,75
9935 DATA 17,37,0,0,0,0,200
9937 DATA 19,63,0,0,0,0,200
9940 DATA 11,114,0,0,0,0,75
9942 DATA 12,216,0,0,0,0,75
9945 DATA 13,156,0,0,0,0,700
9947 DATA 12,216,0,0,0,0,75
9950 DATA 11,114,0,0,0,0,75
9952 DATA 10,60,0,0,0,0,75
9955 DATA 17,37,0,0,0,0,75
9957 DATA 16,47,25,177,0,0,75
9960 DATA 14,107,24,63,0,0,75
9962 DATA 12,216,25,177,0,0,75
9965 DATA 11,114,25,117,0,0,75
9967 DATA 10,60,17,37,0,0,75
9970 DATA 9,159,17,37,0,0,75
9972 DATA 8,147,20,100,0,0,200
9975 DATA 20,100,25,177,0,0,75
9977 DATA 20,100,24,63,0,0,75
9980 DATA 19,63,25,177,0,0,200
9981 DATA 17,37,28,214,0,0,200
9982 DATA 15,70,19,63,0,0,200
9983 DATA 14,107,25,177,0,0,75
9984 DATA 14,107,24,63,0,0,75
```

```
9985 DATA 15,7Ø,25,177,Ø,Ø,2ØØ
9986 DATA 17,37,28,214,Ø,Ø,2ØØ
9987 DATA 12,32,17,37,Ø,Ø,75
9988 DATA 12,32,19,63,Ø,Ø,75
9989 DATA 12,216,2Ø,1ØØ,Ø,Ø,2ØØ
999Ø DATA 14,1Ø7,2Ø,1ØØ,Ø,Ø,2ØØ
9991 DATA 12,32,19,63,Ø,Ø,75
9992 DATA 12,32,17,37,Ø,Ø,75
9993 DATA 12,216,15,7Ø,Ø,Ø,1ØØØ
9999 DATA -1,-1,-1,-1,-1,-1,-1
```

NOTE: There are two disadvantages with the automatic loading. The first is that the programs take longer to load from the disk. The second is the DIM statements have to be avoided. Otherwise, when you attempt to reload a program with a DIM statement, automatic execution will cease, and an error message will be given about attempting to redefine an array. If you type RUN, the program will execute without any problems, but it is annoying to have the automatic execution interrupted with an error message.

The final program in this appendix is "MUSIC MAKER". When you RUN this program, prompts are given for

WAVEFORM?	(you enter 17 for pipes, 33 for organ and 65 for piano)
ATTACK/DECAY?	(you enter anthing from Ø to 255, try 19Ø)
SUSTAIN/RELEASE?	(you enter anything from Ø to 255, try 25Ø)

The music will then play, with repeats until you press a key.

You may like to experiment with different sound characteristics. A swell and diminution can be created with the respective inputs, 33 and then 255 followed by 5Ø.

"Music Maker"

```
1Ø PRINT "□":RESTORE
9ØØØ REM:INITIALIZE ALL SETTINGS TO ZERO
9Ø1Ø FOR L=54272 TO 54296:POKE L,Ø:NEXT
9Ø2Ø REM:SET VOLUME TO MAXIMUM
9Ø3Ø POKE 54296,15
9Ø35 PRINT:PRINT
9Ø4Ø PRINT "THE WAVEFORM, ATTACK/DECAY A
ND SUSTAIN/RELEASE SETTINGS ARE USED "
9Ø5Ø PRINT "TO SYNTHESIZE INSTRUMENTS"
9Ø55 PRINT:PRINT:PRINT"INPUT -1 TO STOP"
9Ø56 PRINT:PRINT
9Ø6Ø INPUT "WAVEFORM";W:IF W=-1 THEN STO
P
9Ø61 IF W=65 THEN FOR I=54274TO54288 STE
P 7:POKEI,255:POKEI+1,Ø:NEXT
9Ø65 INPUT "ATTACK/DECAY";A:IF A=-1 THEN
STOP
9Ø66 INPUT "SUSTAIN/RELEASE";S:IF S=-1 T
HEN STOP
9Ø7Ø REM:SET THESE NUMBERS INTO EACH VOI
CE LOCATION
9Ø8Ø FOR I=1 TO 3:V=54269+7*I:POKE V+1,A
:POKE V+2,S:NEXT I
91ØØ REM:GET CREATES A LOOP TERMINATED B
Y PRESSING A KEY
91Ø1 PRINT "□"
91Ø2 PRINT "PRESS ANY KEY TO STOP NOTE A
ND RESTART, PRESS RUN/STOP TO STOP"
91Ø3 RESTORE
911Ø GET A$:IF A$="" THEN 913Ø
912Ø GOTO 1Ø
913Ø FOR I=1 TO 3:READ H(I),L(I):NEXT I
914Ø READ D
915Ø IF D=-1 THEN RESTORE:GOTO 913Ø
916Ø REM:SET VOICES
917Ø FOR I=1 TO 3:V=54266+7*I:POKE V,H(I
):POKE V-1,L(I):NEXT I
9175 FOR I=1 TO 3:POKE 54269+7*I,W:NEXT
918Ø REM:DURATION
919Ø FOR T=1 TO D:NEXT
```

```
9195 IF W=65 THEN FOR I=1 TO 3:POKE 5426
9+7*I,W-1:NEXT
92ØØ GOTO 911Ø
989Ø REM:
9895 REM:THE DATA FOR THE NOTES IS IN TH
E FORM H,L,H,L,H,L AND DURATION D
99ØØ DATA 17,37,Ø,Ø,Ø,Ø,75
99Ø5 DATA 16,47,Ø,Ø,Ø,Ø,75
991Ø DATA 17,37,Ø,Ø,Ø,Ø,2ØØ
9912 DATA 12,216,Ø,Ø,Ø,Ø,2ØØ
9915 DATA 13,156,Ø,Ø,Ø,Ø,2ØØ
9917 DATA 17,37,Ø,Ø,Ø,Ø,75
992Ø DATA 16,47,Ø,Ø,Ø,Ø,75
9922 DATA 17,37,Ø,Ø,Ø,Ø,2ØØ
9925 DATA 19,63,Ø,Ø,Ø,Ø,2ØØ
9927 DATA 12,216,Ø,Ø,Ø,Ø,2ØØ
993Ø DATA 17,37,Ø,Ø,Ø,Ø,75
9932 DATA 16,47,Ø,Ø,Ø,Ø,75
9935 DATA 17,37,Ø,Ø,Ø,Ø,2ØØ
9937 DATA 19,63,Ø,Ø,Ø,Ø,2ØØ
994Ø DATA 11,114,Ø,Ø,Ø,Ø,75
9942 DATA 12,216,Ø,Ø,Ø,Ø,75
9945 DATA 13,156,Ø,Ø,Ø,Ø,7ØØ
9947 DATA 12,216,Ø,Ø,Ø,Ø,75
995Ø DATA 11,114,Ø,Ø,Ø,Ø,75
9952 DATA 1Ø,6Ø,Ø,Ø,Ø,Ø,75
9955 DATA 17,37,Ø,Ø,Ø,Ø,75
9999 DATA -1,-1,-1,-1,-1,-1,-1
```

By changing the data statements in "MUSIC MAKER" you can create your own music. The following are a few samples.

MESSIAH - HALLELUJAH CHORUS

```
99Ø5 DATA 38,126,28,214,9,159,4ØØ
991Ø DATA 28,214,28,214,12,32,1ØØ
9915 DATA 32,94,25,177,12,216,1ØØ
992Ø DATA 28,214,24,63,9,159,1ØØ
9925 DATA Ø,Ø,Ø,Ø,Ø,Ø,2ØØ
993Ø DATA 38,126,28,214,9,159,4ØØ
9935 DATA 28,214,28,214,12,32,1ØØ
994Ø DATA 32,94,25,177,12,216,1ØØ
9945 DATA 28,214,24,63,9,159,1ØØ
995Ø DATA Ø,Ø,Ø,Ø,Ø,Ø,2ØØ
9955 DATA 38,126,28,214,12,32,25
```

```
9956 DATA Ø,Ø,Ø,Ø,Ø,Ø,1
996Ø DATA 38,126,28,214,12,32,25
9965 DATA 38,126,32,94,12,216,1ØØ
997Ø DATA 38,126,28,214,9,159,1ØØ
9975 DATA Ø,Ø,Ø,Ø,Ø,Ø,2ØØ
998Ø DATA 38,126,28,214,12,32,25
9981 DATA Ø,Ø,Ø,Ø,Ø,Ø,1
9982 DATA 38,126,28,214,12,32,25
9983 DATA 38,126,32,94,12,216,1ØØ
9984 DATA 38,126,28,214,9,159,1ØØ
9985 DATA Ø,Ø,Ø,Ø,Ø,Ø,2ØØ
9986 DATA 38,126,28,214,12,32,1ØØ
9987 DATA 36,85,25,177,1Ø,2Ø5,1ØØ
9988 DATA 38,126,24,63,9,159,1ØØ
9989 DATA 38,126,21,154,14,1Ø7,1ØØ
999Ø DATA 36,85,21,154,14,1Ø7,1ØØ
9991 DATA 38,126,24,63,159,4ØØ
9995 DATA Ø,Ø,Ø,Ø,Ø,Ø,4ØØ
9999 DATA -1,-1,-1,-1,-1,-1,-1
```

THE PLANETS (YEAR 2ØØ1)

```
99ØØ DATA 11,114,Ø,Ø,5,185,16ØØ
991Ø DATA 17,37,11,114,8,147,16ØØ
992Ø DATA 22,227,17,37,11,114,16ØØ
993Ø DATA 28,214,17,37,14,1Ø7,1ØØ
994Ø DATA 27,56,22,227,11,114,28ØØ
995Ø DATA 11,114,Ø,Ø,5,185,16ØØ
996Ø DATA 17,37,11,114,8,147,16ØØ
997Ø DATA 22,227,17,37,11,114,16ØØ
998Ø DATA 28,214,17,37,14,1Ø7,1ØØ
999Ø DATA 3Ø,141,19,63,15,7Ø,28ØØ
9999 DATA -1,-1,-1,-1,-1,-1,-1
```

BATTLE HYMN OF THE REPUBLIC

```
99ØØ DATA 19,63,19,63,19,63,1ØØ
99Ø1 DATA 19,63,12,216,6,1Ø8,3ØØ
99Ø2 DATA 19,63,19,63,19,63,1ØØ
99Ø3 DATA 19,63,12,216,6,1Ø8,3ØØ
99Ø4 DATA 17,37,17,37,17,37,1ØØ
99Ø5 DATA 16,47,12,216,6,1Ø8,3ØØ
99Ø6 DATA 19,63,19,63,19,63,1ØØ
99Ø7 DATA 25,177,12,216,6,1Ø8,3ØØ
99Ø8 DATA 28,214,28,214,28,214,1ØØ
```

```
9909 DATA 32,94,16,47,6,108,300
9910 DATA 32,94,32,94,32,94,100
9911 DATA 32,94,16,47,6,108,300
9912 DATA 28,214,28,214,28,214,100
9913 DATA 25,177,16,47,6,108,600
9914 DATA 25,177,16,47,6,108,300
9915 DATA 24,63,24,63,24,63,100
9916 DATA 21,154,12,216,4,73,300
9917 DATA 21,154,21,154,21,154,100
9918 DATA 21,154,12,216,4,73,300
9919 DATA 24,63,24,63,24,63,100
9920 DATA 25,177,12,216,4,73,300
9921 DATA 24,63,24,63,24,63,100
9922 DATA 25,177,12,216,4,73,300
9923 DATA 21,154,21,154,21,154,100
9924 DATA 19,63,12,216,6,108,300
9925 DATA 21,154,21,154,21,154,100
9926 DATA 19,63,12,216,6,108,300
9927 DATA 16,47,16,47,16,47,100
9928 DATA 19,63,12,216,6,108,400
9929 DATA 19,63,12,216,6,108,300
9930 DATA 19,63,19,63,19,63,100
9931 DATA 19,63,12,216,6,108,300
9932 DATA 19,63,19,63,19,63,100
9933 DATA 19,63,12,216,6,108,300
9934 DATA 17,37,17,37,17,37,100
9935 DATA 16,47,12,216,6,108,300
9936 DATA 19,63,19,63,19,63,100
9937 DATA 25,177,12,216,4,73,300
9938 DATA 28,214,28,214,28,214,100
9939 DATA 32,94,16,47,6,108,300
9940 DATA 32,94,32,94,32,94,100
9941 DATA 32,94,16,47,6,108,300
9942 DATA 28,214,28,214,28,214,100
9943 DATA 25,177,16,47,6,108,600
9944 DATA 25,177,19,63,16,47,800
9945 DATA 28,214,21,154,17,37,800
9946 DATA 28,214,21,154,17,37,800
9947 DATA 25,177,19,63,16,47,800
9948 DATA 24,63,19,63,17,37,800
9949 DATA 25,177,19,63,16,47,3200
9960 DATA 19,63,16,47,6,108,700
9961 DATA 17,37,14,107,6,108,100
9962 DATA 16,47,12,216,6,108,300
```

```
9963 DATA 19,63,16,47,6,108,100
9964 DATA 25,177,16,47,6,108,300
9965 DATA 28,214,17,37,6,108,100
9966 DATA 32,94,19,63,6,206,400
9967 DATA 32,94,19,63,9,159,400
9968 DATA 25,177,16,47,12,216,300
9969 DATA 25,177,16,47,12,32,100
9970 DATA 10,205,0,0,0,0,300
9971 DATA 9,159,0,0,0,0,100
9972 DATA 21,154,17,37,12,216,700
9973 DATA 24,63,19,63,12,216,100
9974 DATA 25,177,21,154,12,216,300
9975 DATA 24,64,19,63,12,216,100
9976 DATA 25,177,21,154,12,216,300
9977 DATA 21,154,17,37,12,216,100
9978 DATA 19,63,16,47,12,216,400
9979 DATA 19,63,16,47,9,159,400
9980 DATA 16,47,12,216,9,159,300
9981 DATA 16,47,12,216,8,147,100
9982 DATA 8,23,0,0,0,0,300
9983 DATA 7,53,0,0,0,0,100
9984 DATA 19,63,16,47,9,159,700
9985 DATA 17,37,14,107,9,159,100
9986 DATA 16,47,12,216,9,159,300
9987 DATA 19,63,16,47,9,159,100
9988 DATA 25,177,16,47,9,159,300
9989 DATA 28,214,17,37,9,159,100
9990 DATA 32,94,19,63,12,216,400
9991 DATA 32,94,20,100,12,32,400
9992 DATA 25,177,21,154,10,205,400
9993 DATA 25,177,21,154,16,47,400
9994 DATA 28,214,21,154,17,37,400
9995 DATA 28,214,21,154,17,37,400
9996 DATA 25,177,19,63,16,47,400
9997 DATA 24,63,19,63,17,37,400
9998 DATA 25,177,19,63,16,47,3200
9999 DATA -1,-1,-1,-1,-1,-1,-1
```

Index

1-norm, xii, 129, 135
2-norm, 124, 140

A

ABS, 12, 15, 18, 122, 127, 129, 132, 143, 170, 171, 172, 174
absolute value, 135, 172
addition, 90, 110
Algebra, viii, xii, 126, 151, 154, 163
algebraic, 114, 164
ANAGRAMS, vii, xi, xii, xiii, 28, 31, 33, 37, 41, 183, 187, 188
Analytic Geometry, xi, xii, 125, 126, 146
arithmetic, vii, xii, 97, 99, 104, 119, 126, 151, 160, 163, 167
ASC, 7, 11, 12, 15, 16, 19, 20, 21, 22, 45, 50, 53, 74, 75, 78, 101, 105, 108, 109
ATN, 138, 142

B

Babalonians, 102
back-substitution, 137, 161
BARTER, vii, xii, 98, 99, 184, 188, 189
BASE, vii, xii, xiii, 101, 102, 103, 104, 105, 106, 107, 184, 188, 189
basic polynomials, 32
binary, 90, 102, 103, 104
binomial coefficients, 65
binomial expansion, xii, 64
Binomial, 64
Bisection, 174, 176, 177
block structure, 91
BRACKET, viii, xiii, 165, 171, 174, 176, 184, 188, 189

C

Calculus, 106, 125
Cartesian coordinate, xii, 140
CHR$, 7, 11, 12, 15, 19, 21, 45, 50, 53, 54, 74, 75, 77, 78, 101, 108, 109, 181, 183
COMPLEX NUMBERS, xii, 114, 115
complex, 114, 177
CONCENTRATION, vii, ix, xii, 2, 5, 13, 24, 183, 187, 188
coordinate, xii, 5, 15, 17, 122, 124, 125, 127, 129, 131, 132, 135, 136, 138, 139, 140, 141, 143, 144, 146
CRYPTIC, viii, xii, 108, 110, 184, 188, 189
cubic equation, 164, 176
cylindrical polar coordinate, xii, 140

D

DATA, 23, 30, 35, 36, 123, 133, 134, 145, 190, 191, 193, 194, 195, 196
DEF FNF, 173
DEF FNG, 170
DETONATION, viii, xii, xiii, 122, 124, 129, 140, 141, 184, 188, 189
differentiate, 106
DIM, 25, 173, 191

DISK LOAD, 25, 179, 180, 181, 184, 185, 188, 190, 197

E

ENIGMA, vii, xi, xiii, 50, 51, 53, 55, 183, 188, 189
equation, 110, 125, 126, 131, 136, 137, 157, 159, 164, 168, 171
Euclidean distance, xii, 124, 129, 140, 141
even, 88, 91, 95, 160

F

factor, 107, 163, 164, 165
factoring, 107
FIND A WORD, vii, xiii, 43, 47, 183, 188, 189
FIXED-POINT, viii, xiii, 165, 170, 171, 172, 173, 176, 177, 184, 188, 189
FLASH, vii, xii, 7, 9, 24, 183, 187, 188
FNF, 174
FNG, 170

G

Gauss Elimination, xii, 136, 137, 161
GET, 2, 4, 7, 8, 11, 12, 16, 19, 20, 21, 22, 28, 30, 33, 35, 38, 39, 40, 46, 50, 53, 54, 58, 59, 63, 66, 67, 70, 74, 75, 79, 81, 82, 85, 93, 98, 101, 105, 108, 109, 113, 117, 118, 122, 123, 127, 128, 132, 133, 138, 139, 143, 144, 148, 153, 156, 158, 162, 166, 170, 175, 182, 183, 185, 186, 187, 192
GOSUB, 2, 3, 21, 22, 29, 30, 35, 66, 85, 87, 101, 105, 123, 133, 138, 139, 144, 181

H

HANGMAN, vii, xiii, 38, 41, 183, 184, 188, 189
HUMAN ARITHMETIC, viii, xii, 112, 114, 184, 188, 189

I

imaginary numbers, 114
INT, 2, 3, 7, 11, 15, 28, 29, 33, 34, 38, 39, 43, 44, 45, 50, 53, 58, 63, 64, 67, 70, 74, 79, 81, 85, 86, 87, 98, 101, 105, 108, 112, 117, 122, 124, 127, 132, 138, 141, 143, 148, 153, 156, 158, 162, 166, 170, 181, 182, 183, 186, 187
interpolating polynomial, 32
interpolation, 31
INVADER, vii, xiii, 58, 60, 69, 183, 187, 189

J

JUGS, viii, xii, 117, 119, 184, 188, 189

K

KNIGHT, vii, xiii, 11, 13, 15, 17, 183, 187, 188
KNIGHTMARE, vii, xiii, 15, 17, 183, 187, 188
Kronecker, 114

L

Lagrange Interpolation, xi, xii, 32

LEFT$, 3, 8, 12, 15, 19, 21, 22, 29, 34, 35, 39, 53, 58, 67, 86, 92, 105, 109, 112
LEN, 40, 45, 70, 86, 112
linear equation, xi, xii, 135, 136, 146, 155, 157, 160, 167, 171, 173
linear systems, 160
LOADER, 25, 179, 181, 182, 184, 185, 188, 190
LOG, 170, 174
LOOSE CHANGE, vii, xii, 72, 74, 76, 183, 187, 189

M

matrix, 25, 136, 160, 173
matrix algebra, 160
MID$, 3, 15, 19, 20, 21, 22, 29, 34, 39, 44, 45, 53, 54, 58, 66, 70, 74, 86, 87, 92, 98, 101, 108, 112
MONKEY PUZZLE, viii, xii, xiii, 148, 150, 184, 188, 189
MORE OBSCURE AGES, viii, xii, 155, 156, 157, 184, 188, 189
MUCH MORE OBSCURE AGES, viii, xii, 157, 158, 159, 184, 188, 189
multiplicative inverses, 115

N

nested multiplication, 107
NIM, vii, xi, xiii, 79, 81, 83, 84, 85, 89, 183, 187, 189
nonlinear equations, xi, xiii, 163
norm, xii, 124, 129, 135, 173

O

OBSCURE AGES, viii, xii, 153, 154, 157, 160, 163, 184, 188, 189
odd, 18, 95, 120
ON, 28, 33, 38, 43, 45, 112, 181, 182, 186, 187, 188, 190
ordered pair, 114

P

Pascal's triangle, xii, 65
polar coordinates, xii, 141
POKE, 21, 22, 23, 25, 28, 30, 33, 35, 54, 123, 133, 139, 144, 145, 181, 183, 186, 187, 190, 192, 193
polynomial, 31, 106, 164, 171, 176
probabilities, xi, xiii, 64, 65
PROBABILITY, vii, xiii, 61, 63, 64, 65, 183, 187, 189
PRODUCT AND SUMS, viii, xii, xiii, 162, 163, 165, 171, 172, 184, 188, 189

Q

quadratic, xi, xiii, 173
QUEST, viii, xii, 127, 129, 135, 184, 188, 189
quotient, 107

R

RALLY, viii, xii, 166, 167, 184, 188, 189
READ, 22, 25, 30, 35, 123, 133, 144, 190, 192
real roots, 176, 177
Regular FALSI, 174, 177
remainder, 103, 104, 106, 107, 152

repetitive multiplication, 106
repetitive division, 103
RETURN, 3, 7, 8, 11, 15, 23, 29, 33, 35, 38, 43, 58, 59, 66, 67, 68, 81, 86, 87, 88, 101, 105, 117, 122, 123, 127, 132, 133, 138, 139, 143, 145, 148, 181, 183, 184, 197
RESTORE, 22, 30, 35, 123, 133, 144, 180, 190, 192
RIGHT$, 3, 7, 12, 15, 19, 21, 25, 29, 34, 39, 53, 58, 67, 86, 92, 105, 109, 112
RND, 2, 7, 11, 15, 19, 21, 28, 29, 33, 34, 38, 39, 43, 44, 45, 50, 53, 58, 63, 64, 66, 70, 74, 79, 81, 85, 87, 98, 105, 108, 112, 117, 119, 122, 127, 132, 133, 138, 143, 148, 153, 156, 158, 162, 166, 181
root, xiii, 126, 164, 165, 170, 171, 172, 174

S

SEA HUNT, vii, xiii, 66, 69, 183, 187, 189
second degree, 125
sequence, 9, 10, 24, 25, 172
SGN, 138, 174
significant figures, 170, 172, 174, 177
SIMON, vii, xi, xii, 19, 21, 24, 179, 183, 187, 188
SQR, 112, 114, 122, 124, 139, 141
SQR(–1), 114
statistics, xii, xiii, 64
STEP, 21, 22, 23, 46, 54, 92, 192
STOP, 4, 8, 12, 16, 19, 22, 25, 30, 35, 40, 46, 50, 54, 59, 63, 67, 71, 75, 80, 82, 85, 93, 98, 101, 105, 109, 113, 118, 122, 123, 125, 128, 133, 139, 144, 148, 153, 156, 158, 162, 166, 170, 175, 179, 181, 182, 183, 186, 187, 192
stopping rule, 172
STR$, 3, 7, 19, 21, 25, 53, 58, 59, 66, 67, 86, 101, 105, 108, 112, 117, 118, 122, 126, 127, 133, 143, 148, 182, 187
string, 7, 25, 28, 33, 70, 72, 92, 94, 98, 102, 115
subroutine, 6, 91, 103
subtraction, 90, 119, 125
synthetic division, 106, 164

T

TAB, 11, 15, 29, 34, 35, 39, 45, 46, 58, 59, 66, 67, 79, 81, 92, 170, 174, 181
three-dimensional, 17, 135
TOWERS OF HANOI, vii, xiii, 92, 94, 183, 187, 189
TREK, viii, xii, 132, 135, 146, 184, 188, 189
two-dimensional, 17, 18

U

U-BOAT, viii, xii, 138, 140, 184, 188, 189

V

VAL, 2, 11, 15, 19, 20, 22, 53, 70, 85, 86, 87, 92, 98, 101, 105, 108, 109, 112, 182, 183, 186, 187

vectors, 173
VERIFY, vii, xii, 70, 72, 183, 187, 189

W

WARP, viii, xii, 143, 146, 184, 188, 189

Y

YOUR NUMBER IS UP, viii, xii, xiii, 104, 105, 106, 164, 184, 188, 189

Z

zero, xiii, 32, 140, 164, 171, 173, 176, 177

Instructions For Making A Back-Up Copy Of Your Brain Games Diskette

1. Insert a blank disk into the disk drive. Be sure to use a disk without anything on it that you wish to keep.
2. Type:

   OPEN15,8,15

   then press RETURN. Type:

   PRINT#15,"N:BRAIN-BACKUP,64"

   then press RETURN. This operation will blank and reformat your disk and will take a few minutes.
3. Type:

   CLOSE15

   Press RETURN. Type:

   LOAD"$",8

   Press RETURN. Type:

   LIST

   and press RETURN. At this point you should have on your screen:

   Ø "BRAIN-BACKUP "64 2A
   664 BLOCKS FREE

4. Now you are ready to copy the programs from the BRAIN GAMES disk to your back-up disk. This will involve LOADing each of 41 programs from the BRAIN GAMES disk into the computer and then SAVEing each on your disk.

   Follow these steps:
   a. Remove your back-up disk.
   b. Insert the BRAIN GAMES disk.
   c. Type: LOAD"CONCENTRATION",8 (Press RETURN)
   d. Remove the BRAIN GAMES disk.
   e. Insert your back-up disk.
   f. Type: SAVE"CONCENTRATION",8 (Press RETURN)

Follow the above steps for each of the programs listed on the next page. Be sure to type in the names exactly, including periods and spaces. For your convenience, a checklist is provided.

If you find you have misspelled a program name on your back-up disk, you can correct the program spelling by the following procedure:

1. Insert your back-up disk.
2. Type:

   OPEN15,8,15

   Press RETURN.
3. Type:

   PRINT#15,"R:NEWNAME=OLDNAME"

   Press RETURN. **NOTE:** In place of NEWNAME you should type the correctly spelled program name, and in place of OLDNAME you type the misspelled program name.
4. Type:

   CLOSE15

It is recommended that you use your back-up disk and put the original BRAIN GAMES disk in a safe place.

PROGRAM	STEPS a	b	c	d	e	f
"CONCENTRATION"						
"FLASH"						
"KNIGHT"						
"KNIGHTMARE"						
"SIMON"						
"ANAGRAMS I"						
"ANAGRAMS II"						
"HANGMAN"						
"FIND A WORD"						
"ENIGMA I"						
"ENIGMA II"						
"INVADER"						
"PROBABILITY"						
"SEA HUNT"						
"VERIFY"						
"LOOSE CHANGE"						
"NIM II"						
"NIM III"						
"TOWERS OF HANOI"						
"BARTER"						
"BASE"						
"YOUR NUMBER IS UP"						
"CRYPTIC"						
"HUMAN ARITHMETIC"						
"JUGS"						
"DETONATION"						
"QUEST"						
"TREK"						
"U-BOAT"						
"WARP"						
"MONKEY PUZZLE"						
"OBSCURE AGES"						
"MORE OBSCURE AGES"						
"MUCH MORE OBSCURE AGES"						
"PRODUCT AND SUMS"						
"RALLY"						
"FIXED-POINT"						
"BRACKET"						
"DISK LOAD"						
"LOADER"						
"MUSIC MAKER"						

Instructions for Operating the Prerecorded Floppy Disk

To load and execute the programs on the disk, type in

```
LOAD "DISK LOAD",8
```

then press RETURN. The program loads in less than 10 seconds. To execute the program, type RUN and press RETURN.

Sit back and relax, there is nothing more to do but to make your choices from the menu of chapters and programs that are displayed on your screen.

Brain Games for Kids and Adults Using the Commodore 64

John W. Stephenson, PhD

SOFTSYNC

Here are 40 exciting, challenging, educational games:

- ★ Can you find the treasure before the bloodthirsty pirates find you?
- ★ Can you find your way out of the four-dimensional time-warp before your fuel runs out?
- ★ Will your powers of deduction hold up as you desperately try to locate the bomb before it explodes?

Brain Games for Kids and Adults Using the Commodore 64 is the only way to find out. It's fun. It's entertaining. And it's a new way to learn about mathematics while you play! This remarkable book packs an enormous amount of information in just a few lines of code to give you more than games alone. It's a learning tool! You can guess at the solutions or work them out logically—either way you'll pick up some very practical knowledge. It couldn't be easier or more challenging. No matter your age or experience, a little concentration and imagination is all you need.

The challenge is yours. Get ready for Brain Games!

Contents

Introduction ★ Chapter I: Concentration Games ★ Chapter II: Word Games ★ Chapter III: Logic and Strategy Games ★ Chapter IV: Arithmetic Games ★ Chapter V: Geometry Games ★ Chapter VI: Algebra Games ★ Chapter VII: Root Finders ★ Appendix ★ Index

Also available. . .an accompanying diskette featuring all the games from the text.

Prentice/Hall PHI International

Cover design by Don Sellers

£5.95

ISBN 0-13-080938-1