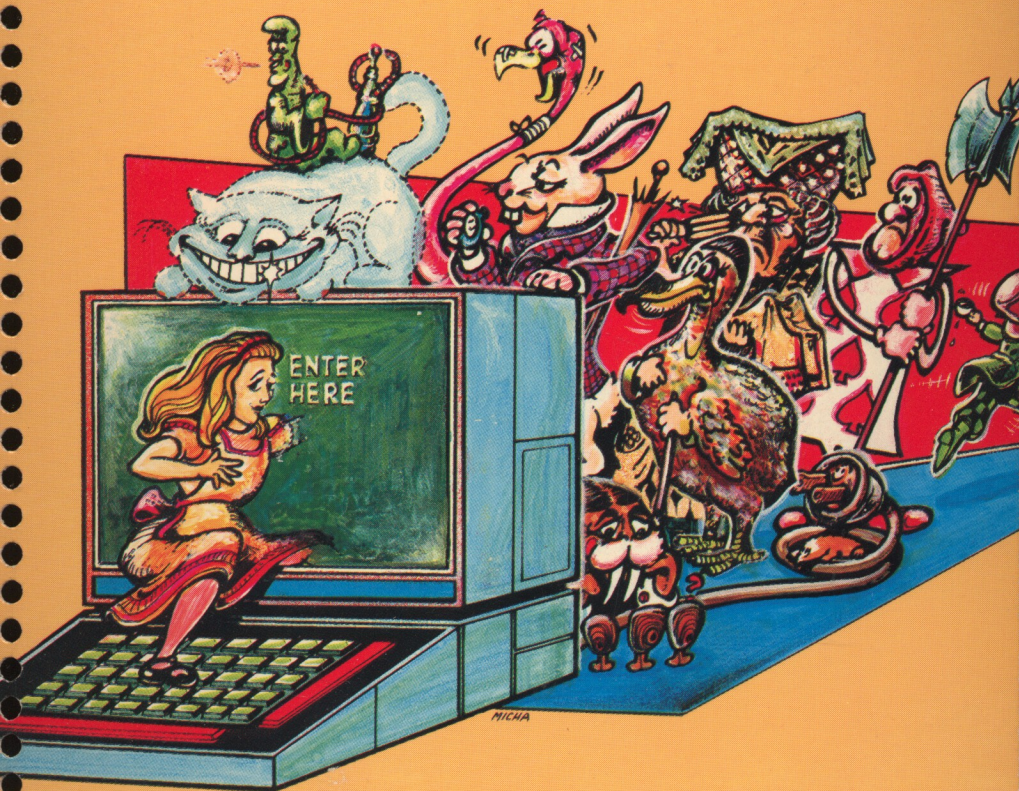


ATARI® PUZZLEMENTS? for the Atari® home computer HERBERT KOHL



A CREATIVE PASTIMES BOOK

This book belongs to

ATARI® PUZZLEMENTS? for the Atari® home computer



by HERBERT KOHL

illustrations by C. MICHA



A Creative Pastimes Book
Reston Publishing Company
A Prentice-Hall Company
Reston, Virginia

Library of Congress Cataloging in Publication Data

Kohl, Herbert R.

Atari puzzlements for the ATARI home computer.

"A Reston Computer Group book."

1. Computer games. 2. Puzzles. 3. Atari computer--
Programming. 4. Basic (Computer program language)--Study
and teaching. I. Title.

GV1469.2.K63 1984

794.8'2

84-3259

ISBN 0-8359-0113-2

Interior design and production by Laura Cleveland.

This book is published by Reston Publishing Company which is not affiliated with Atari, Inc. and Atari is not responsible for any inaccuracies. ATARI is a registered trademark of Atari, Inc.

Copyright © 1984 by Reston Publishing Company, Inc., A Prentice-Hall Company, Reston, Virginia 22090.

All rights reserved. No part of this book may be reproduced in any way or by any means without written permission from the publisher.

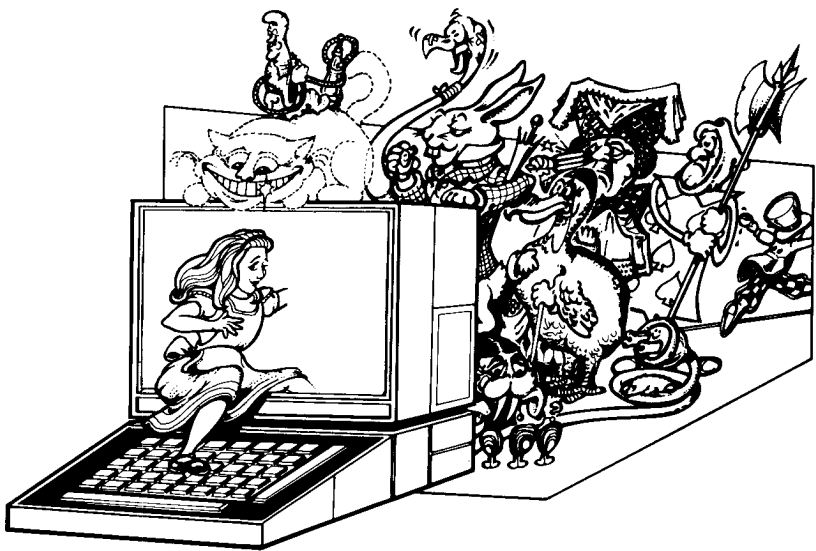
10 9 8 7 6 5 4 3 2 1

Printed in the United States of America.

Contents

Introduction	1
Error Puzzles	7
Single Line Scrambles	27
Line Number Scrambles	47
Missing Line Puzzles	69
Control Graphics Puzzles	89
Appendix: Planning Sheets	115





Introduction

Most computer puzzles are versions of ordinary word games, such as crossword puzzles or anagrams, that draw their vocabulary from the world of computing. The puzzles here are different. They provide games, amusements, and challenges from within the world of computing itself and require some familiarity with Atari BASIC. The puzzles in this first volume are quite simple; however, we plan additional volumes with more complex and sophisticated puzzles. You do not have to be a computer buff or a skillful programmer to solve them. In fact, they would be an ideal complement to any text or reference manual for a person of any age who is just learning BASIC and is beginning to feel the power of creating programs.

We have not included any PEEKs or POKEs in the puzzles and have not used programs longer than 15 lines. These puzzles are designed to help you think in BASIC, learn to read a program, and understand something about program structure. All the commands in the book are included in the **ATARI BASIC Reference Manual** or in any introductory text on BASIC. Since the programs are not that long or complex (though some are tricky and not easy to solve automatically), many can be done with a paper and pencil. In fact, it's a good idea to try to think through many of these problems without using your computer. Program design is a mental activity that is computer-assisted and working on computer problems without using a computer can help develop that skill. In order to help you sketch out solutions, we've provided some formats designed to help with problem solving. They are in the Appendix to the book and you should feel free to copy them.

There are five different types of problems in the book: Error Statement Problems, Line Scrambles, Line Number Scrambles, Missing Code Lines, and Control Character Graphics Design Problems. Some of the programs deal with numbers; others deal with words. Color and graphics are often used. The programs have been selected to illustrate certain aspects of Atari BASIC such as nested loops, subroutines, branching, and the mix of graphics, words, and numbers. You might even find it useful to incorporate some of the smaller programs into larger programs you build yourself.

The answers to the programs will provide explanations of what the program does and give hints that might help you solve other problems. They will also give you a correct program. **Remember, however, that often there is more than one correct answer and, if you feel correct,**

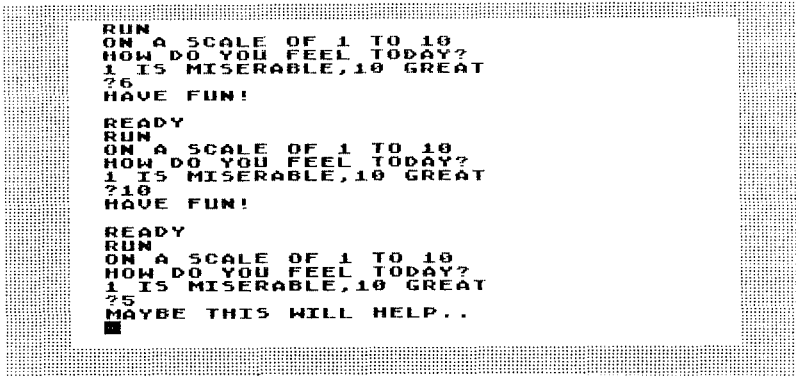
run your program. If it works you have come up with another solution!!

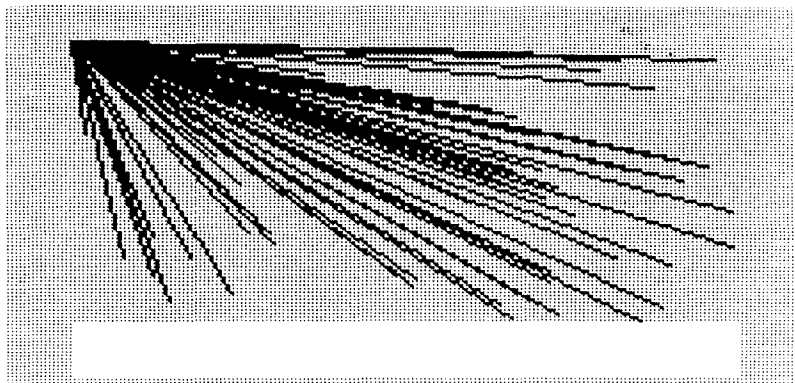
Here is a simple example that should give you a sense of the different types of puzzles in the book:

```
10 PRINT "ON A SCALE OF 1 TO 10"  
20 PRINT "HOW DO YOU FEEL TODAY?"  
30 PRINT "1 IS MISERABLE,10 GREAT"  
40 INPUT X  
50 IF X>5 THEN PRINT "HAVE FUN!":END  
60 PRINT "MAYBE THIS WILL HELP..":FOR  
X=1 TO 400:NEXT X  
70 GRAPHICS 7:COLOR 1  
80 PLOT 0,0  
90 DRAWTO RND(1)*159,RND(1)*80  
100 GOTO 80
```

This program asks how you feel on a scale of 1 to 10, 1 being miserable and 10 great. If you answer with a number above 5 the computer prints "HAVE FUN!" and the program ends. If you answer 5 or less and indicate you are not feeling too great, the computer goes into a Graphics mode (line 70) and draws an unending series of rays of different length emanating from the upper left of the screen (point 0,0). It gives the appearance of a sun and tries to cheer you up.

To let you know what a program is supposed to do we have provided screen pictures or, as they are usually called, **screen dumps** to illustrate the program as it is supposed to run. Here are two screen dumps of the sample program:





This program can also illustrate the nature of different puzzle types in this book.

Error Puzzles

In these puzzles, there will be some error for you to figure out. It could be on line 90, for example, which would then read:

```
90 error DRAWTO RND(1)*159;RND(1)*80
```

The error would be the ; between 159 and the second RND. It should be a , or the program wouldn't work.

Here's another possible error. Can you figure it out?

```
30 error PRINT "1 IS MISERABLE ,10 GREAT
```

You probably guessed that the error was the missing " at the end of the line.

Single Line Scrambles

In these puzzles, one line is all scrambled up. You have to unscramble it to make the program work. Here's a scramble of line 50:

```
50 error PRINT 5 THEN >"x" END FUN IF : HAVE
```

The screen dumps will give you the clues needed to unscramble the line and make the program work.

Line Number Scrambles

Instead of just scrambling a line these puzzles mix up all the line numbers. You have to renumber each line to put the program in order and have it run as planned. This requires some thought and experimentation and it is here that you are likely to find more than one correct unscrambling of a program.

A very simple scramble would just reverse all of the line numbers so that

- line 10 becomes line 100
- line 20 becomes line 90
- line 30 becomes line 80
- etc.

However, you are not likely to come upon such patterned scrambles. The line number patterns do not provide hints to the unscrambling of the lines. You have to think through to the program structure to solve the puzzle.

Missing Line Puzzles

In this variant of program puzzles, one line of code is missing. It might be in our sample:

```
40 PRINT "????????????????"
```

or

```
80 PRINT "????????????????"
```

PRINT "????????????????" is the indicator of the missing code line. That does not mean that PRINT, ", or ? necessarily appear on the missing line.


```
LIST
```

```
5 REM REPLACL THE ?'S BY CONTROL
```

```
6 REM CHARACTERS TO MAKE THE DESIGN
```

```
10 PRINT "~~~~~ ???? 4 11";
```

```
20 GOTO 10
```

```
LIST
```

```
0 REM SAMPLE.2
```

```
5 REM REPLACE THE ?'S BY CONTROL
```

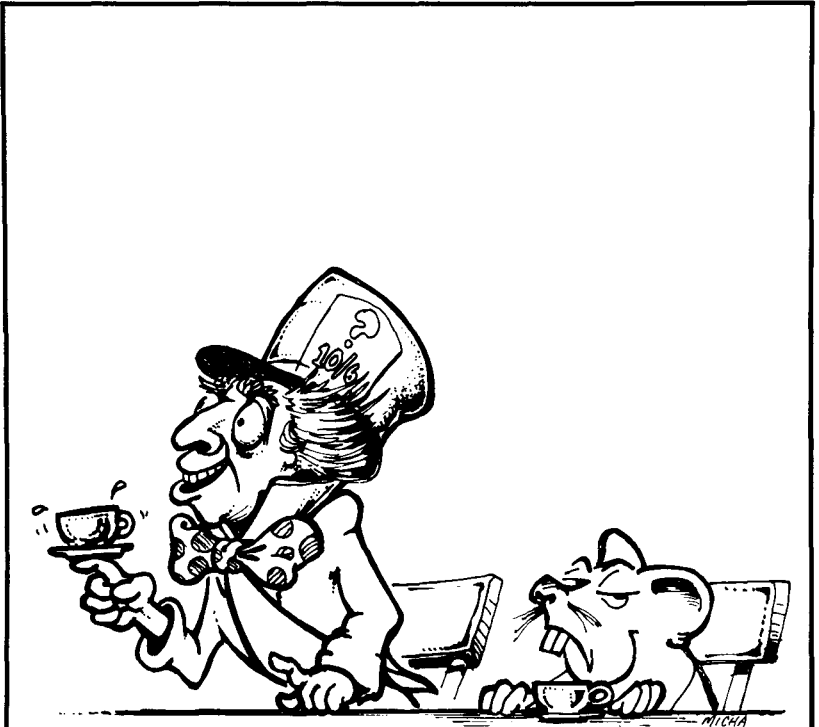
```
6 REM CHARACTERS TO MAKE THE DESIGN
```

```
10 PRINT "???????? ???? ??????";
```

```
20 GOTO 10
```



Now that you have a sense of the kinds of challenges in this book, try your hand at solving them. Each section begins simply and then moves on to more complex demands. The answers and their explanations are at the end of each chapter. Hope you have as much fun solving the puzzles as I had inventing them.



Error Puzzles

or getting your program
to do what you plan

One of the most frustrating things about learning to program a computer is that you can work hard at a program and make a tiny mistake that throws the whole thing off. It is particularly annoying if you have not internalized the programming language you are using. This first section contains some puzzles that embody the simplest mistakes everybody makes when learning to program. These mistakes should be looked at as puzzles to solve rather than as signs of your inability to master computing.



1

A SIMPLE CRYSTAL BALL

This program asks whether you like the number 3 or 4 better. When run, this is how it is supposed to respond:

```
RUN
WHICH NUMBER IS YOUR FAVORITE
OF THESE TWO: 3 OR 4 ?
?3
YOU HAVE MYSTICAL POWERS.

READY
RUN
WHICH NUMBER IS YOUR FAVORITE
OF THESE TWO: 3 OR 4 ?
?4
YOU ARE DEEP AND INTUITIVE

READY
■
```

However, your program simply bombs when you input the number 3. It seems OK if you input 4:

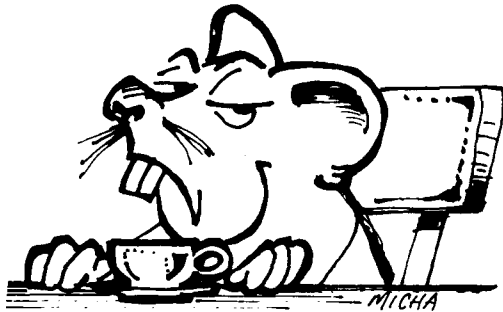
```
RUN
WHICH NUMBER IS YOUR FAVORITE
OF THESE TWO: 3 OR 4 ?
?3

READY
RUN
WHICH NUMBER IS YOUR FAVORITE
OF THESE TWO: 3 OR 4 ?
?4
YOU ARE DEEP AND INTUITIVE

READY
■
```

Here is the program. What is the mistake?

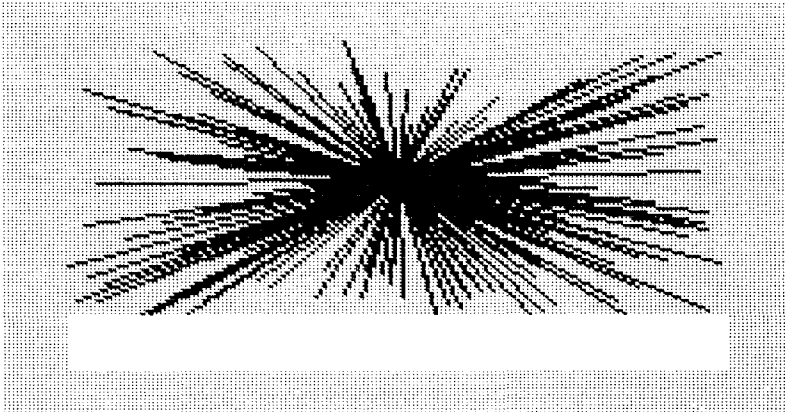
```
10 PRINT "WHICH NUMBER IS YOUR FAVORIT  
E"  
20 PRINT "OF THESE TWO: 3 OR 4 ?"  
30 INPUT X  
40 IF X=3 THEN GOTO 1010  
50 IF X=4 THEN GOTO 1500  
1000 PRINT "YOU HAVE MYSTICAL POWERS."  
1010 END  
1500 PRINT "YOU ARE DEEP AND INTUITIVE  
"  
1510 END
```



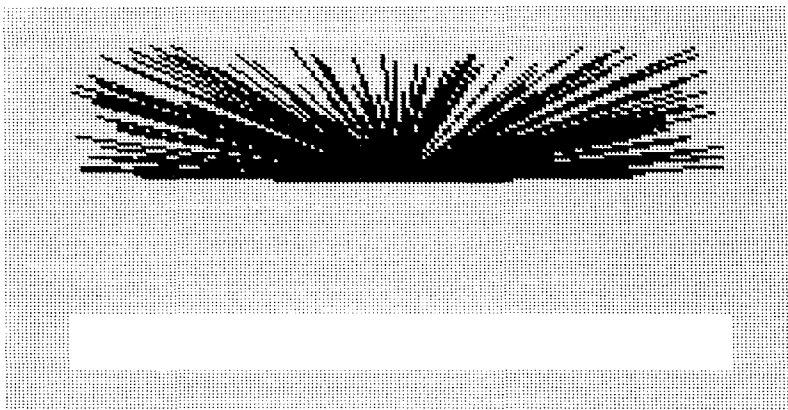
2

NOT ENOUGH SUN

In this program, the intent was to have a full sun as in this screen dump:



Instead you got this on the screen:



How could you change this program to give you the full sun?

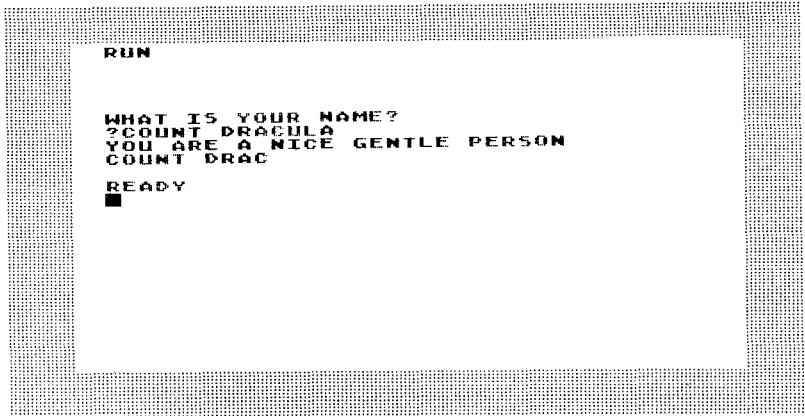
```
10 GRAPHICS 7:COLOR 1
20 PLOT 80,40
30 DRAWTO RND(1)*159,RND(1)*40
40 GOTO 20
```

3

COUNT DRACULA

You write a simple program asking your friend's name. You want the computer to tell your friend that he or she is a nice person. One of your friends decides to play around and type in Count Dracula and here is what happens with your program:

```
10 PRINT "WHAT IS YOUR NAME?"
20 DIM A$(10)
30 INPUT A$
40 PRINT "YOU ARE A NICE GENTLE PERSON
"
50 PRINT A$
```



```
RUN
WHAT IS YOUR NAME?
?COUNT DRACULA
YOU ARE A NICE GENTLE PERSON
COUNT DRAC
READY
■
```

How can you fix your program so that it gives Dracula his full compliments like this?

RUN

WHAT IS YOUR NAME?
?COUNT DRACULA
YOU ARE A NICE GENTLE PERSON
COUNT DRACULA

READY

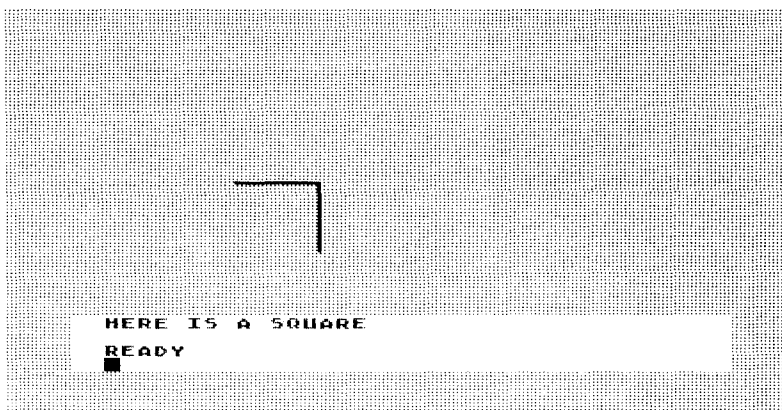


4

ALL I WANT IS A SQUARE

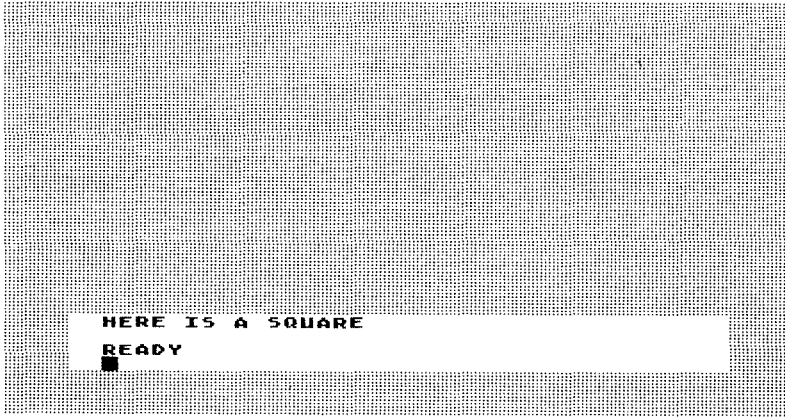
You try to draw a square in Graphics 7 and instead you get the following:

```
10 GRAPHICS 7:COLOR 1
20 PRINT "HERE IS A SQUARE"
30 PLOT 40,40:DRAWTO 60,40
40 DRAWTO 60,60:DRAWTO 60,40
50 DRAWTO 40,40
```

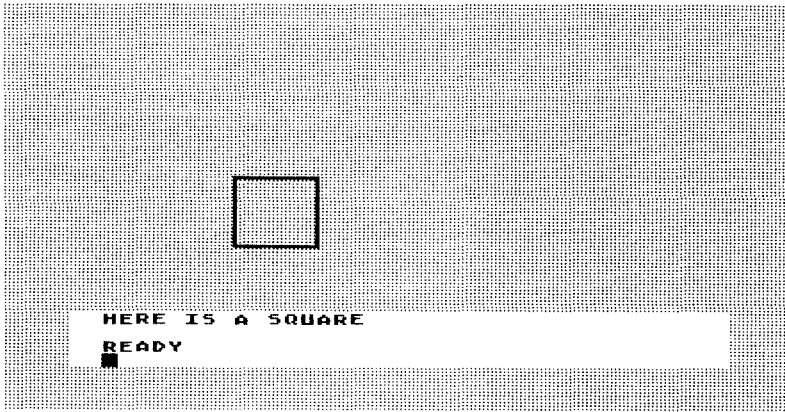


So you try to modify the program and you get:

```
10 GRAPHICS 7:COLOR 4
20 PRINT "HERE IS A SQUARE"
30 PLOT 40,40:DRAWTO 60,40
40 DRAWTO 60,60:DRAWTO 40,60
50 DRAWTO 40,40
```



All you really want is this:



How do you do it and what went wrong with the other two programs?

5

IN FIVE YEARS

This simple program is supposed to ask a person's age and then tell them how old they will be in five years. It should run like this:

```
RUN

HOW OLD ARE YOU?
?12
IN FIVE YEARS YOU WILL BE
17
READY
■
```

Instead it runs like this:

```
RUN

HOW OLD ARE YOU?
?12
IN FIVE YEARS YOU WILL BE
5
READY
■
```

Here's the program. Where's the error?

```
10 PRINT "HOW OLD ARE YOU?"  
20 INPUT A  
30 PRINT "IN FIVE YEARS YOU WILL BE"  
40 PRINT X+5
```

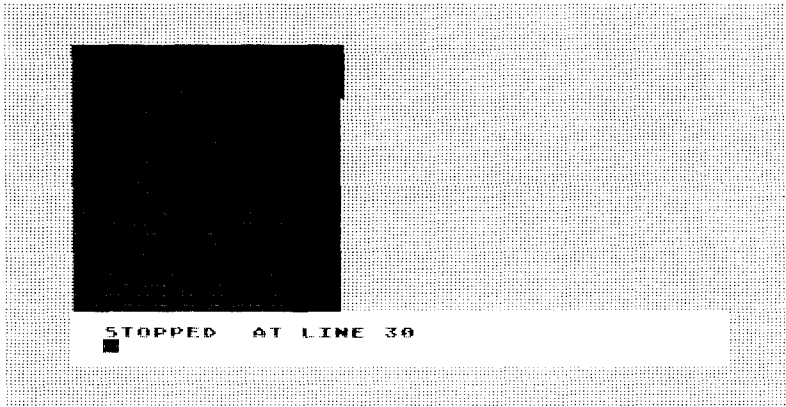

6

FILLING THE SCREEN

This is a reverse puzzle. Here is a program that will progressively fill the screen up from top to bottom and left to right.

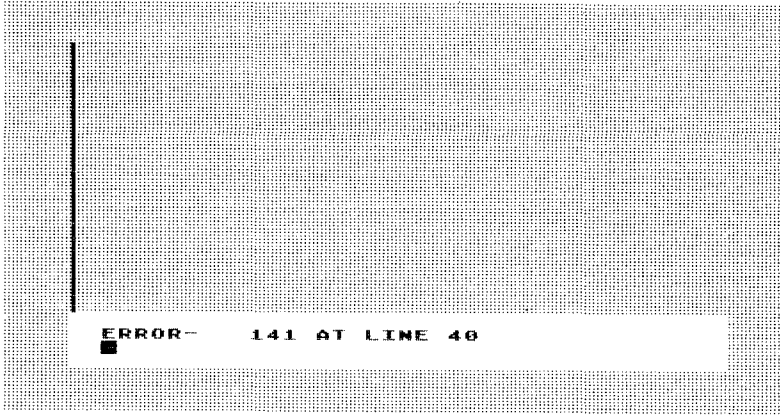
```
10 GRAPHICS 7:COLOR 1
20 FOR X=1 TO 159
30 FOR Y=1 TO 80
40 PLOT X,Y
50 NEXT X
60 NEXT Y
```

Now here is a screen dump of the program stopped in midcourse. If it were left running the whole screen would be full.

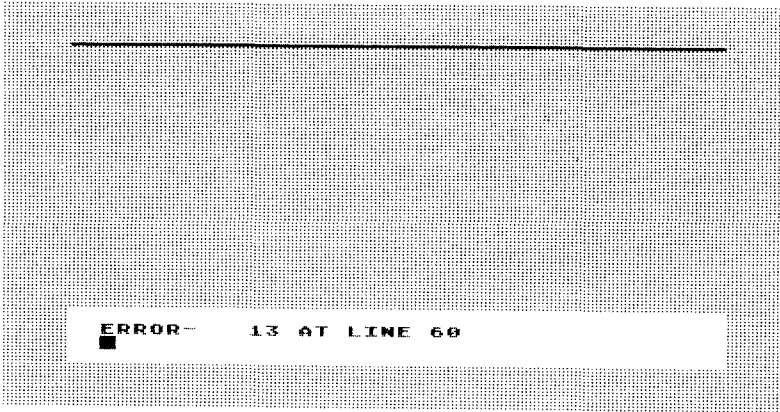


Now here are two screen dumps of programs that almost do what the first program does. A simple error, however, leads to a totally different structure on the screen. Notice that these programs lead to ERROR statements. Can you redo the correct program above to create these two ERROR statements and their associated screens?

Error 6A



Error 6B



Answers

1

At line 40 you instructed the program to GOTO line 1010 which ended the program. You always have to be careful that you refer your program to exactly the lines you want, and it makes sense to check all GOTO statements if you have a program that doesn't run properly. Here's the program with the intended reference:

```
10 PRINT "WHICH NUMBER IS YOUR FAVORIT  
E"  
20 PRINT "OF THESE TWO: 3 OR 4 ?"  
30 INPUT X  
40 IF X=3 THEN GOTO 1000  
50 IF X=4 THEN GOTO 1500  
1000 PRINT "YOU HAVE MYSTICAL POWERS."  
1010 END  
1500 PRINT "YOU ARE DEEP AND INTUITIVE  
"  
1510 END
```

2

The problem was at line 30. You only used half the screen. You have to be careful about the screen dimensions of the Graphics modes you are programming in. At line 30, the number 40 should be 80 as in this program:

```
10 GRAPHICS 7:COLOR 1  
20 PLOT 80,40  
30 DRAWTO RND(1)*159,RND(1)*80  
40 GOTO 20
```

3

Here is a program that will give Dracula his due:

```
10 PRINT "WHAT IS YOUR NAME?"
20 DIM A$(25)
30 INPUT A$
40 PRINT "YOU ARE A NICE GENTLE PERSON
"
50 PRINT A$
```

Notice that at line 20 the DIM A\$(10) has been changed to DIM A\$(25). When you are storing words in Atari BASIC it is very important to make sure that you make your dimension statement large enough to contain the largest response you expect. The program, as changed for Dracula, wouldn't work if your friend's name was John Wellington Smith Frankenstein. Sometimes it makes sense if you are not writing too long a program to overdimension your variables in order to anticipate your friends' tricks. DIM A\$(100) couldn't hurt in this case.

4

Here is the program:

```
10 GRAPHICS 7:COLOR 1
20 PRINT "HERE IS A SQUARE"
30 PLOT 40,40:DRAWTO 60,40
40 DRAWTO 60,60:DRAWTO 40,60
50 DRAWTO 40,40
```

In one case, you made mistakes with the coordinates you drew. It is often useful to sketch out a drawing on graph paper before trying to put coordinates in a program.

In the other case, you used COLOR 4 in your program. If you look in your Atari reference manual you'll see that the number 4 doesn't work with the COLOR command and so nothing happened. It's important to play with COLOR and to PLOT around to get familiar with your Graphics 7 screen.

5

```
10 PRINT "HOW OLD ARE YOU?"
20 INPUT X
30 PRINT "IN FIVE YEARS YOU WILL BE"
40 PRINT X+5
```

Notice that in the error program you INPUT A and then added +5. However, there was no X to add anything with and your variables did not agree so the computer printed 5 no matter what age was entered. When line 20 was changed to 20 INPUT X the program worked. It is essential to check all of your variables and see that they are properly referenced to each other. This is a basic principle of programming.

6

6A:

```
10 GRAPHICS 7:COLOR 1
20 FOR X=1 TO 80
30 FOR Y=1 TO 159
40 PLOT X,Y
50 NEXT Y
60 NEXT X
```

Error 141 says that the cursor in the program is out of range for the program—that is you have sent it off the screen and bombed the program. If you look at program 6A you'll see that you plotted Y (the row) to go up to 159 and it can only go up to 80. Here is a good example of a nested loop error. The program works the following way:

- 1—Plot the first X
 - 2—Plot the first 159 Y's
 - 3—Plot the second X
 - 4—Plot the first 159 Y's again
 - 5—Plot the third X
 - 6—Plot the first 159 Y's again
- etc. for 159 times through the loop.

This program never gets beyond the second step. It plots 1, Y for Y=1 to 80 giving you the line on the left of the screen. As soon as it gets to the point 0,81 the cursor runs off the screen and you get the ERROR statement. It is essential to check the references of the points you plot in a program to be sure that they do not exceed the number of points in the Graphics mode you are using, especially in nested FOR/NEXT loops.

6B:

```
10 GRAPHICS 7:COLOR 1
20 FOR X=1 TO 159
30 FOR Y=1 TO 80
40 PLOT X,Y
50 NEXT Y
60 NEXT X
```

This is another FOR/NEXT loop trap. ERROR 13 says that there is no matching FOR statement for the last NEXT. This error is common enough to have a built-in error detector for it. What happens in this program that does not work is that:

- 1—you begin with X=1
- 2—and Y=1
- 3—and plot 1,1 and
- 4—go to the next X and plot 2,1
- 5—and then go to the next X and plot 3,1
- 6—and plot 4,1 all the way up to 159,1

This gives you the horizontal line at the top of the screen. Once this line has been drawn, the program bombs. It has no way to get to the NEXT Y. The nested loops have not been ordered correctly. Whenever you nest loops in order to have two things working in conjunction, you have to structure the loops so that the first loop begins and ends the nesting, the second loop is second from the beginning and second from the end, etc. Nested loops are like sets of parentheses where each (has to have a matching). Here's how three nested loops would have to be structured to work:

```
10 FOR X=1 TO 10
20 FOR Y=1 TO 52
```

```
30 FOR Z=1 TO 5
40 do something in the program
50 NEXT Z
60 NEXT Y
70 NEXT X
```

The order in is XYZ and the order out is ZYX. It is a good programming habit to check for correct order in and out of nested loops. As you saw in our error program, if you don't do it the computer will automatically do it for you. It won't fix the error, however, so it's best to check yourself.





Single Line Scrambles

The puzzles in this section consist of a short program with one line all scrambled up. There is a screen dump accompanying each program so that you can tell what the program is supposed to do. An ERROR statement marks the scrambled line so you don't have to figure out which is the problem line. Before you begin, here is an example of scrambled and unscrambled lines:

Scrambled:

```
50 Error 1 GOSUB TO:10FOR=X500XNEXT:
```

Unscrambled:

```
50 FOR X=1 TO 10:GOSUB 500:NEXT X
```



1

MULTIPLICATION

We'll start with a simple multiplication program. As you can see from the screen dump, the program gives you multiplication problems, as well as an opportunity to try again if you get the wrong answer.

```

RUN
HERE'S A SIMPLE MULTIPLICATION PROBLEM

5 TIMES 8 =
?39
TRY AGAIN
?48
GOOD GOING! HERE'S ANOTHER EXAMPLE:
HERE'S A SIMPLE MULTIPLICATION PROBLEM

4 TIMES 4 =
?16
GOOD GOING! HERE'S ANOTHER EXAMPLE:
HERE'S A SIMPLE MULTIPLICATION PROBLEM

3 TIMES 3 =
?■

```

Here's the scrambled program:

```

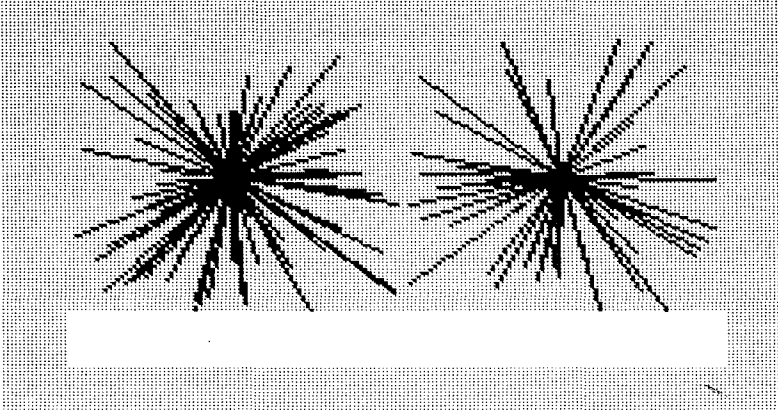
10 PRINT "HERE'S A SIMPLE MULTIPLICATI
ON PROBLEM"
20 PRINT
30 LET X=INT(RND(1)*10)
40 LET Y=INT(RND(1)*10)
50 PRINT X;" TIMES ";Y;" ="
60 INPUT Z
70 ERROR- XYZ*THEN100IFGOTO=
80 PRINT "TRY AGAIN":GOTO 60
100 PRINT "GOOD GOING! HERE'S ANOTHER
EXAMPLE:"
110 GOTO 10

```

2

DOUBLE SUN

This is a graphics puzzle which plots a double sun on the screen. Each sun takes a turn drawing its rays, resulting in this image on the screen:



Here is the scrambled program:

```
10 GRAPHICS 7:COLOR 1
20 FOR X=1 TO 50
30 PLOT 40,40
40 ERROR- (RND*80) RND*80 (DRAWTO (*, *1
50 NEXT X
60 FOR Y=1 TO 50
70 PLOT 120,40
80 DRAWTO RND(1)*79+80,RND(1)*80
90 NEXT Y
100 GOTO 20
```

3

HOURS TO SECONDS

This simple program converts hours to seconds. It asks you how many seconds there are in any number of hours. It won't ask for seconds in more than 10 hours since the calculating gets boring at that point. Nevertheless, it could:

```

RUN
THERE ARE 60 MINUTES IN AN HOUR.
THERE ARE 60 SECONDS IN AN MINUTE.
HOW MANY SECONDS ARE THERE IN 1 HOURS?
?3600
ABSOLUTLEY!
READY
RUN
THERE ARE 60 MINUTES IN AN HOUR.
THERE ARE 60 SECONDS IN AN MINUTE.
HOW MANY SECONDS ARE THERE IN 7 HOURS?
?12400
NOT QUITE.
?25200
ABSOLUTLEY!
READY
■
```

Here's the scrambled program:

```

10 PRINT "THERE ARE 60 MINUTES IN AN H
OUR."
20 PRINT "THERE ARE 60 SECONDS IN AN M
INUTE."
30 LET X=INT(RND(1)*10)+1
40 PRINT "HOW MANY SECONDS ARE THERE I
N ";X;" HOURS?"
50 INPUT Y
60 ERROR- PRINT-THEN!END"60ABSOLUTEL
Y&0*IF*X":Y
70 PRINT "NOT QUITE.":GOTO 50
```

4

TIMED CALCULATIONS

This program sees if you can do some simple number manipulation under some time pressure. It mixes multiplication with simple addition by asking you to add 1 to your times answer. Of course, any number could be added and the challenge could be made more complex:

```
RUN
CAN YOU CALCULATE QUICKLY?
LET'S SEE.

WHAT IS..
(2 TIMES 9) +1=

WHAT IS YOUR ANSWER?
?17
TRY AGAIN.
WHAT IS YOUR ANSWER?
?19
YOU GOT IT!!!!

READY
■
```

Here is the scrambled program:

```
10 PRINT "CAN YOU CALCULATE QUICKLY?"
20 PRINT "LET'S SEE."
30 PRINT
40 PRINT "WHAT IS.."
50 LET X=INT(RND(1)*50)
60 PRINT
70 ERROR- "!!"="+"+PRINT2XTIMES) (1
80 FOR Z=1 TO 150:NEXT Z
90 PRINT
100 PRINT " WHAT IS YOUR ANSWER?"
110 INPUT Y
120 IF Y=(2*X)+1 THEN PRINT "YOU GOT I
T!!!!":END
130 PRINT "TRY AGAIN.":GOTO 100
```

5

A SIMPLE NUMBER GAME

This is a simple number comparison game for young children. It asks whether one number is greater than another. If you look closely at the program you'll see that the computer will never select two equal numbers. It teaches the use of $<$ and $>$.

```
RUN
HERE'S A SIMPLE NUMBER GAME:
IS 10 GREATER OR LESS THAN ?
TYPE > IF IT IS GREATER AND < IF IT IS
LESS
?>
YOU GOT IT
READY
RUN
HERE'S A SIMPLE NUMBER GAME:
IS 10 GREATER OR LESS THAN 11
TYPE > IF IT IS GREATER AND < IF IT IS
LESS
?■
```

Here is the scrambled program:

```
10 PRINT "HERE'S A SIMPLE NUMBER GAME:"
"
20 ERROR- =XRND(INT)20(+LET)*1
30 IF X=10 THEN GOTO 20
40 PRINT "IS 10 GREATER OR LESS THAN "
IX
50 DIM A$(1)
60 PRINT
70 PRINT "TYPE > IF IT IS GREATER AND
< IF IT IS LESS"
80 INPUT A$
90 IF X<10 THEN GOTO 1000
100 IF X>10 THEN GOTO 2000
1000 IF A$=">" THEN PRINT "YOU GOT IT"
:END
1010 PRINT "SORRY. TYPE RUN AND TRY A
GAIN IF YOU LIKE.":END
2000 IF A$="<" THEN PRINT "YOU GOT IT"
:END
2010 PRINT "SORRY. TYPE RUN AND TRY A
GAIN IF YOU LIKE.":END
```

6

MINUTE/DAY CONVERSION

Here's a minute/day conversion program that tells you how many minutes there are in any number of days. It does not quiz you, but answers your question instead:

```

RUN
HERE'S A PROGRAM THAT WILL TELL
YOU HOW MANY MINUTES THERE ARE IN
ANY NUMBER OF DAYS.
HOW MANY DAYS?
?3

THERE ARE 4320 MINUTES IN 3 DAYS
DO YOU WANT TO CHANGE THE NUMBER OF
DAYS?
?YES
HOW MANY DAYS?
?7

THERE ARE 10080 MINUTES IN 7 DAYS
DO YOU WANT TO CHANGE THE NUMBER OF
DAYS?
?NO

READY
■
```

Here is the scrambled program:

```

10 PRINT "HERE'S A PROGRAM THAT WILL T
ELL"
20 PRINT "YOU HOW MANY MINUTES THERE A
RE IN"
30 PRINT "ANY NUMBER OF DAYS."
40 PRINT
50 PRINT "HOW MANY DAYS?"
60 CLR
70 INPUT X
80 GOSUB 500
90 PRINT "DO YOU WANT TO CHANGE THE NU
MBER OF DAYS?"
100 DIM A$(3)
110 INPUT A$
120 IF A$="YES" THEN GOTO 50
130 END
500 PRINT
510 ERROR= X";60;PRINT"HERE*MINUT
ES*DAYS" "IN"ARE";24
520 RETURN
```


7

QUICK ADDITION

This puzzle challenges you to add a string of six numbers together quickly:

```

RUN
HERE IS A LIST OF NUMBERS
TRY TO ADD THEM TOGETHER QUICKLY
4 2 9 8 8 2
WHAT IS YOUR ANSWER
?25
TRY ONCE MORE.
?31
TRY ONCE MORE.
?33
YOU'VE GOT IT.

READY
■
```

Here is the scrambled program:

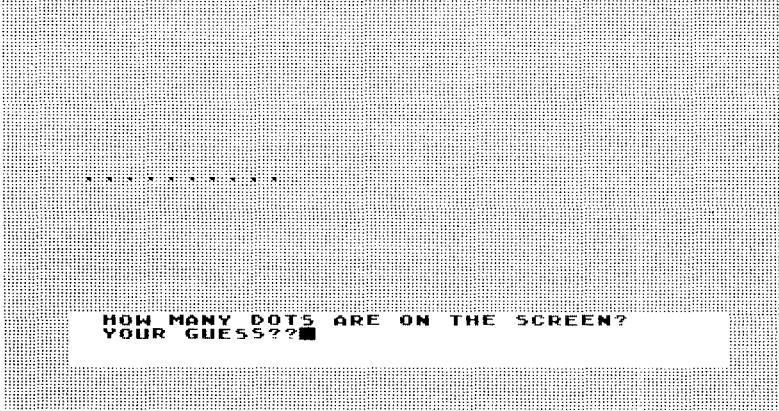
```

10 PRINT "HERE IS A LIST OF NUMBERS"
20 PRINT "TRY TO ADD THEM TOGETHER QUI
CKLY"
30 FOR Z=1 TO 1000:NEXT Z
40 DIM A(6)
50 FOR X=1 TO 6
60 ERROR- AX(((RNDINT)/))10*=
70 PRINT A(X);" ";
80 NEXT X
90 FOR Z=1 TO 2000:NEXT Z
100 PRINT
110 PRINT "WHAT IS YOUR ANSWER"
120 INPUT Y
130 IF Y=A(1)+A(2)+A(3)+A(4)+A(5)+A(6)
THEN PRINT "YOU'VE GOT IT.":END
140 IF Y<>A(1)+A(2)+A(3)+A(4)+A(5)+A(6)
) THEN PRINT "TRY ONCE MORE.":GOTO 120
```

8

DOT COUNTING

This is a dot counting game.



Here is the scrambled program:

```
10 GRAPHICS 7:COLOR 1
20 PRINT "HOW MANY DOTS ARE ON THE SCR
EEN?"
30 LET X=INT(RND(1)*15)+1
40 LET W=5
50 FOR Y=1 TO X
60 PLOT W,40
70 ERROR- SWW=LET+
80 NEXT Y
90 PRINT "YOUR GUESS?";
100 INPUT Z
110 IF Z=X THEN PRINT "EXACTLY!!!":END
120 PRINT "COUNT AGAIN....":GOTO 100
```

9

RIDICULOUS

This program can print the word "RIDICULOUS"—any number of times—perhaps a ridiculous venture but one whose structure is used in some programming contexts:

```

RUN
HOW MANY TIMES WOULD YOU LIKE ME TO
PRINT YOUR NAME OR ANY OTHER WORD?
235
WHAT WORD WOULD YOU LIKE ME TO PRINT
35 TIMES?
?RIDICULOUS
RIDICULOUS RIDICULOUS RIDICULOUS RIDIC
ULOUS RIDICULOUS RIDICULOUS RIDICULOUS
RIDICULOUS RIDICULOUS RIDICULOUS RIDI
CULOUS RIDICULOUS RIDICULOUS RIDICULO
S RIDICULOUS RIDICULOUS RIDICULOUS RID
ICULOUS RIDICULOUS RIDICULOUS RIDICULO
US RIDICULOUS RIDICULOUS RIDICULOUS RI
DICULOUS RIDICULOUS RIDICULOUS RIDICU
OUS RIDICULOUS RIDICULOUS RIDICULOUS R
IDICULOUS RIDICULOUS RIDICULOUS RIDICU
LOUS
READY
■
```

Here is the scrambled program:

```

10 PRINT "HOW MANY TIMES WOULD YOU LIK
E ME TO"
20 PRINT "PRINT YOUR NAME OR ANY OTHER
WORD?"
30 INPUT X
40 PRINT "WHAT WORD WOULD YOU LIKE ME
TO PRINT ";X;" TIMES?"
50 DIM A$(50)
60 INPUT A$
70 ERROR- -XTD1YFOR
80 PRINT A$;" "
90 NEXT Y
100 END
```

10

BARBER'S PARADOX

This little scramble is a version of the classical Barber's Paradox: There is a barber in a town who shaves all and only people in the town who don't shave themselves. Does he shave himself?

I've taken the paradox to school you might say:

```

RUN
JULIA IS A KIND AND LOVELY PERSON.
SHE DOES HOMEWORK FOR ALL AND ONLY
PEOPLE WHO DON'T DO HOMEWORK FOR
THEMSELVES.
DOES JULIA DO HER OWN HOMEWORK?
?YES
THEN SHE DOESN'T DO IT.
JULIA IS A KIND AND LOVELY PERSON.
SHE DOES HOMEWORK FOR ALL AND ONLY
PEOPLE WHO DON'T DO HOMEWORK FOR
THEMSELVES.
DOES JULIA DO HER OWN HOMEWORK?
?NO
THEN SHE DOES DO IT.
JULIA IS A KIND AND LOVELY PERSON.
SHE DOES HOMEWORK FOR ALL AND ONLY
PEOPLE WHO DON'T DO HOMEWORK FOR
THEMSELVES.
DOES JULIA DO HER OWN HOMEWORK?
?■

```

```

10 PRINT "JULIA IS A KIND AND LOVELY P
ERSON. "
20 PRINT "SHE DOES HOMEWORK FOR ALL AN
D ONLY"
30 PRINT "PEOPLE WHO DON'T DO HOMEWORK
FOR"
40 PRINT "THEMSELVES. "
50 PRINT "DOES JULIA DO HER OWN HOMEWO
RK?"
60 CLR
70 DIM A$(3)
80 INPUT A$
90 ERROR= 10".THENGOTO"A$=DOESN'TIF"
SHEYES:PRINT"ITDOTHEN
100 IF A$="NO" THEN PRINT "THEN SHE DO
ESN'T DO IT.":GOTO 10

```

Answers

1

```
10 PRINT "HERE'S A SIMPLE MULTIPLICAT:  
ON PROBLEM"  
20 PRINT  
30 LET X=INT(RND(1)*10)  
40 LET Y=INT(RND(1)*10)  
50 PRINT X;" TIMES ";Y;" ="  
60 INPUT Z  
70 IF Z=X*Y THEN GOTO 100  
80 PRINT "TRY AGAIN":GOTO 60  
100 PRINT "GOOD GOING! HERE'S ANOTHER  
EXAMPLE:"  
110 GOTO 10
```

As you can see, line 70 (the scrambled line) checks for correct answers using the variable Z which is the answer you input. If $Z=X*Y$ then you got the program and are sent to line 100 for congratulations and a chance at another example. If $Z < > X*Y$ the program moves to line 80 which sends you back to the problem you missed. This little technique can be incorporated in games where you want to give people many chances to answer a question as well as generate new questions.

2

```
10 GRAPHICS 7:COLOR 1  
20 FOR X=1 TO 50  
30 PLOT 40,40  
40 DRAWTO RND(1)*80,RND(1)*80  
50 NEXT X  
60 FOR Y=1 TO 50  
70 PLOT 120,40  
80 DRAWTO RND(1)*79+80,RND(1)*80  
90 NEXT Y  
100 GOTO 20
```

The scrambled line 40 is a DRAWTO statement, one of the most powerful Graphics commands in Atari BASIC. It selects random points on the left side of the screen (half the screen is approximately 80 columns long and 80 columns wide) and draws from the center point of that half-screen to the chosen random point. This double sun effect is a good way to dress up the ending of a game.

3

```
10 PRINT "THERE ARE 60 MINUTES IN AN H  
OUR."  
20 PRINT "THERE ARE 60 SECONDS IN AN M  
INUTE."  
30 LET X=INT(RND(1)*10)+1  
40 PRINT "HOW MANY SECONDS ARE THERE I  
N ";X;" HOURS?"  
50 INPUT Y  
60 IF Y=X*60*60 THEN PRINT "ABSOLUTLEY  
!":END  
70 PRINT "NOT QUITE.":GOTO 50
```

The key to unscrambling this puzzle is to know that you calculate the number of seconds in an hour using the formula X (the number of hours) $\times 60 \times 60$ which is the conversion formula. This form of program can be changed to ask about converting feet or miles to inches, meters to centimeters, etc. It is a simple and generalizable form of a conversion quiz program.

4

```
10 PRINT "CAN YOU CALCULATE QUICKLY?"  
20 PRINT "LET'S SEE."  
30 PRINT  
40 PRINT "WHAT IS.."  
50 LET X=INT(RND(1)*50)  
60 PRINT  
70 PRINT "(2 TIMES ";X;" ) +1="  
80 FOR Z=1 TO 1500:NEXT Z  
90 PRINT  
100 PRINT " WHAT IS YOUR ANSWER?"  
110 INPUT Y
```

```

120 IF Y=(2*X)+1 THEN PRINT "YOU GOT I
T!!!!":END
130 PRINT "TRY AGAIN.":GOTO 100

```

This is simply an unscrambling of the integer random number function in Atari BASIC. Many beginning programmers confuse $RND(1)*X$, which gives a decimal answer, with $INT(RND(1)*X)$ which gives an integer. There are times when having integers is essential to a game or program. Also, it is important to pay attention to the placement of parentheses in the INT statement. A simple mistake like $INT(RND(1))*X$ can cause quite a mess. Try to figure out the different effects of $INT(RND(1))*6$ and $INT(RND(1)*X)$. What you will see is the result of the fact that $INT(RND(1))$ is always 0.

5

```

10 PRINT "HERE'S A SIMPLE NUMBER GAME:
"
20 LET X=INT(RND(1)*20)
30 IF X=10 THEN GOTO 20
40 PRINT "IS 10 GREATER OR LESS THAN "
;X
50 DIM A$(1)
60 PRINT
70 PRINT "TYPE > IF IT IS GREATER AND
< IF IT IS LESS"
80 INPUT A$
90 IF X<10 THEN GOTO 1000
100 IF X>10 THEN GOTO 2000
1000 IF A$=">" THEN PRINT "YOU GOT IT"
:END
1010 PRINT "SORRY. TYPE RUN AND TRY A
GAIN IF YOU LIKE.":END
2000 IF A$="<" THEN PRINT "YOU GOT IT"
:END
2010 PRINT "SORRY. TYPE RUN AND TRY A
GAIN IF YOU LIKE.":END

```

The key to unscrambling this is to figure out that there is only one variable in this challenge. $(2*X)+1$ is the major part of the reconstruction of the scrambled line. There are also some syntactical tricks that have to be mastered to use the PRINT statement with variables. You can print a variable along with words or other numbers but variables do not go in quotes as in this line:

```
10 PRINT X*Y
```

and in this more complex line:

```
10 PRINT X^Y;"=";Z
```

Note that the = is not a variable and has to be enclosed in quotes.

This program can be modified for practice in the kind of calculations that are common in beginning algebra. For example, it can take the form:

What is $((2*X)-(5*Y))^2$?

6

```
10 PRINT "HERE'S A PROGRAM THAT WILL T  
ELL"  
20 PRINT "YOU HOW MANY MINUTES THERE A  
RE IN"  
30 PRINT "ANY NUMBER OF DAYS."  
40 PRINT  
50 PRINT "HOW MANY DAYS?"  
60 CLR  
70 INPUT X  
80 GOSUB 500  
90 PRINT "DO YOU WANT TO CHANGE THE NU  
MBER OF DAYS?"  
100 DIM A$(3)  
110 INPUT A$  
120 IF A$="YES" THEN GOTO 50  
130 END  
500 PRINT  
510 PRINT "THERE ARE ";X*60*24;" MINUT  
ES IN ";X;" DAYS"  
520 RETURN
```

Line 510 combines calculation with printing and, like the previous conversion puzzle (number 3), can be used to perform many different types of conversions. In unscrambling this, the thing to watch for is the placement of quotes and semicolons. A misplaced quote or semicolon can destroy the entire program.


```

10 PRINT "HERE IS A LIST OF NUMBERS"
20 PRINT "TRY TO ADD THEM TOGETHER QUICKLY"
30 FOR Z=1 TO 1000:NEXT Z
40 DIM A(6)
50 FOR X=1 TO 6
60 A(X)=INT(RND(1)*10)
70 PRINT A(X); " ";
80 NEXT X
90 FOR Z=1 TO 2000:NEXT Z
100 PRINT
110 PRINT "WHAT IS YOUR ANSWER"
120 INPUT Y
130 IF Y=A(1)+A(2)+A(3)+A(4)+A(5)+A(6)
    THEN PRINT "YOU'VE GOT IT.":END
140 IF Y<>A(1)+A(2)+A(3)+A(4)+A(5)+A(6)
    ) THEN PRINT "TRY ONCE MORE.":GOTO 120

```

This program makes use of a powerful aspect of Atari BASIC—the array. A(X) is dimensioned on line 40 to contain six numbers. The scrambled line 60 places a random integer between 0 and 10 in each position. That way, lists of numbers can be generated and used as in this program. Beginning and intermediate programmers should take advantage of the ability to store and use lists that this function provides. For examples of the use of arrays see your **Atari BASIC Reference Manual** or look up arrays in a text if you are not already familiar with them.

```

10 GRAPHICS 7:COLOR 1
20 PRINT "HOW MANY DOTS ARE ON THE SCREEN?"
30 LET X=INT(RND(1)*15)+1
40 LET W=5
50 FOR Y=1 TO X
60 PLOT W,40
70 LET W=W+5
80 NEXT Y
90 PRINT "YOUR GUESS?";
100 INPUT Z
110 IF Z=X THEN PRINT "EXACTLY!!!":END
120 PRINT "COUNT AGAIN....":GOTO 100

```

The LET W=W+5 statement represents a very powerful self-incrementing process that is often used in computer programs. For example, a simple program

```
10 LET X=0
20 FOR Y=1 TO 100
30 LET X=X+1:PRINT X
40 NEXT X
```

will give you all the numbers from 1 to 100. If line 39 were changed to

```
30 LET X=X+2:PRINT X
```

you would get the numbers from 2 to 100 counting by twos. This can give you great control over how to change the variables in your programs. In the case of the program in this puzzle, W=W+5 was used to draw points that were five points apart so that they would be seen as distinct on the screen.

9

```
10 PRINT "HOW MANY TIMES WOULD YOU LIKE ME TO"
20 PRINT "PRINT YOUR NAME OR ANY OTHER WORD?"
30 INPUT X
40 PRINT "WHAT WORD WOULD YOU LIKE ME TO PRINT ";X;" TIMES?"
50 DIM A$(50)
60 INPUT A$
70 FOR Y=1 TO X
80 PRINT A$;" "
90 NEXT Y
100 END
```

This is a scramble of the FOR statement of the FOR/NEXT loop used in the program to produce the ridiculous repetition. Notice that the number of times through the loop (the X variable) changes each time you run the program. Most beginning programmers use fixed loops like FOR X=1 TO 59, but sometimes it is more useful to change the number of times through the loop as the program is running.

```
10 PRINT "JULIA IS A KIND AND LOVELY P  
ERSON."  
20 PRINT "SHE DOES HOMEWORK FOR ALL AN  
D ONLY"  
30 PRINT "PEOPLE WHO DON'T DO HOMEWORK  
FOR"  
40 PRINT "THEMSELVES."  
50 PRINT "DOES JULIA DO HER OWN HOMEWO  
RK?"  
60 CLR  
70 DIM A$(3)  
80 INPUT A$  
90 IF A$="YES" THEN PRINT "THEN SHE DO  
ESN'T DO IT.":GOTO 10  
100 IF A$="NO" THEN PRINT "THEN SHE DO  
ES DO IT.":GOTO 10
```

The scrambled line 90 as well as line 100 make the paradox continue until you are bored and shut down the computer. However, the lines illustrate a technique you can add to many of the games and quizzes in this section. It will allow them to run many times instead of ending after just one play. Here's a very simple example:

```
10 PRINT "WHAT NUMBER IS THIS"  
20 X = INT(RND(1)*20)  
25 PRINT X  
30 INPUT Y  
40 IF X = Y THEN PRINT "GREAT":GOTO 10  
50 IF X < > Y THEN PRINT "TRY AGAIN":GOTO 30
```




Line Number Scrambles

The puzzles in this section leave all the code in a program intact. All they do is scramble the line numbers. These numbers are not scrambled according to any pattern so there is no use trying to solve the puzzles by trying to figure out a pattern for transforming the line numbers. You have to pay attention to the screen dumps and try to put a program structure together that will do what the illustrations show.

Scrambling line numbers of even the simplest programs can destroy them. For example, take this two-line program:

```
10 PRINT "HO HO HO!";  
20 GOTO 10
```

This will fill the screen up with a jolly greeting. Reverse the line numbers and you get:

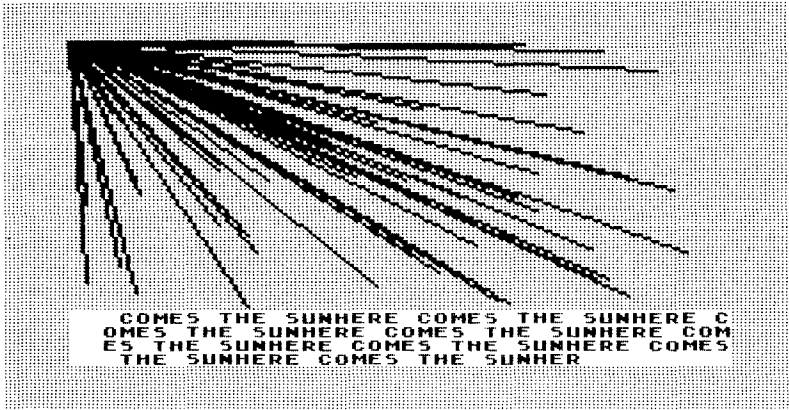
```
10 GOTO 10  
20 PRINT "HO HO HO!";
```

This program will lock up at line 10 and stay there until you press the **BREAK** key. Nothing will happen. In the simplest way, this is an illustration of the importance of program structure in BASIC. Now for the puzzles.

1

HERE COMES THE SUN

This program shines and lets us know endlessly that the sun is coming and filling up the screen. Here is a picture of what the program does and a scrambled version of the program. Unscramble the program and, if you feel like it, make it fancier.

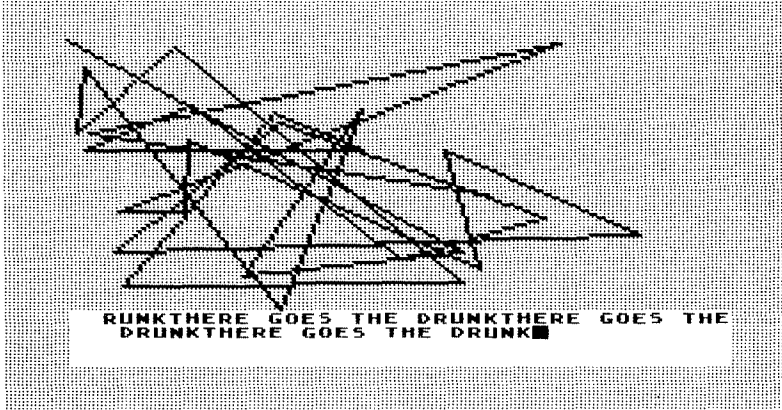


```
10 FOR X=1 TO 75:NEXT X
20 GOTO 20
30 GRAPHICS 7:COLOR 1
40 PRINT "HERE COMES THE SUN";
50 DRAWTO RND(1)*159,RND(1)*80
60 PLOT 0,0
```

2

THERE GOES THE DRUNK

This is a simple variant on Puzzle 1 that produces a completely different result. You should be able to think through the difference and reconstruct the program that ambles with the drunk.



```
10 DRAWTO RND(1)*159,RND(1)*80
20 GOTO 30
30 PRINT "THERE GOES THE DRUNK";
40 GRAPHICS 7:COLOR 1
50 FOR X=1 TO 100:NEXT X
60 PLOT 0,0
```


3

A POETIC SCRAMBLE

Here is a little poem with some input about the weather. The input, combined with the poem, should help you reassemble the scrambled program.

```
RUN
HOW IS THE WEATHER TODAY?
?LOUSY

WELL, WHETHER YOU LIKE IT
OR WHETHER NOT
LOUSY WEATHER
IS THE ONLY KIND OF WEATHER
WE GOT.

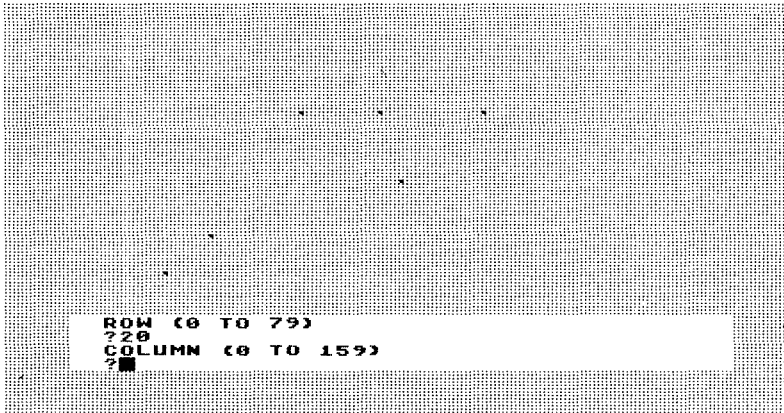
READY
■
```

```
10 PRINT A$;" WEATHER"
20 PRINT "WELL, WHETHER YOU LIKE IT"
30 INPUT A$
40 PRINT "IS THE ONLY KIND OF WEATHER"
50 DIM A$(50)
60 PRINT "OR WHETHER NOT"
70 PRINT "WE GOT."
80 PRINT
90 PRINT "HOW IS THE WEATHER TODAY?"
```

4

WHERE AM I?

This program allows you to plot a point on the screen in Graphics 7 and see the point on the screen. When unscrambled, it is a good way to become familiar with the positions on the screen.



```
10 GOTO 20
20 INPUT B
30 PRINT "ROW (0 TO 79)"
40 PLOT A,B
50 PRINT "COLUMN (0 TO 159)"
60 GRAPHICS 7:COLOR 1
70 INPUT A
```

5

A LITTLE DIFFERENT ORDER

Here is a scrambled program and what it does when you run it:

```
RUN
?56
DON'T ASK ME WHY. PLEASE
GIVE ME TWO NUMBERS TO ADD.
THE ANSWER IS A GLORIOUS 56
I AM CRAZY ABOUT ADDING BUT
THANK YOU. THAT WAS DELICIOUS!
?37
FIRST NUMBER-SECOND NUMBER-
READY
■
```

Here is the scrambled program and a dump of what the unscrambled program is supposed to do. What is the unscrambled program?

```
RUN
I AM CRAZY ABOUT ADDING BUT
DON'T ASK ME WHY. PLEASE
GIVE ME TWO NUMBERS TO ADD.
FIRST NUMBER-?34
SECOND NUMBER-?56
THANK YOU. THAT WAS DELICIOUS!
THE ANSWER IS A GLORIOUS 90
READY
■
```

```
10 INPUT X
20 PRINT "DON'T ASK ME WHY. PLEASE"
30 PRINT "GIVE ME TWO NUMBERS TO ADD."
40 PRINT "THE ANSWER IS A GLORIOUS ";X
+Y
50 PRINT "I AM CRAZY ABOUT ADDING BUT"
60 PRINT "THANK YOU. THAT WAS DELICIOUS!"
70 INPUT Y
80 PRINT "FIRST NUMBER-";
90 PRINT "SECOND NUMBER-";
```



6

CAN YOU COUNT?

This program picks a number and asks you what comes next.

```
RUN
WHAT NUMBER COMES AFTER 2
?4
TRY AGAIN. YOU CAN DO IT!!!
?6
TRY AGAIN. YOU CAN DO IT!!!
?3
RIGHT ON!!!
READY
■
```

```
10 PRINT "WHAT NUMBER COMES AFTER ";X
20 IF Y<>X+1 THEN PRINT "TRY AGAIN. YOU
CAN DO IT!!!"
30 LET X=INT(RND(1)*11)
40 IF Y=X+1 THEN PRINT "RIGHT ON!!!":E
ND
50 GOTO 30
60 INPUT Y
```

7

NEXT LETTER

This program is supposed to ask what letter comes next in the alphabet after one chosen by the computer. When scrambled, here is what it does:

```
RUN
9TH IN THE ALPHABET?
?4
ERROR-- 9 AT LINE 20
■
```

Here is the scrambled program and a screen dump of it running unscrambled:

```
RUN
WHAT LETTER COMES 5TH IN THE ALPHABET?
?D
GUESS AGAIN
?E
RIGHT
READY
■
```

```
10 PRINT X;"TH IN THE ALPHABET?"
20 INPUT B$
30 PRINT "GUESS AGAIN"
40 PRINT "WHAT LETTER COMES ";
50 IF A$(X,X)=B$ THEN PRINT "RIGHT":EN
D
60 A$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
70 DIM B$(1)
80 GOTO 70
90 DIM A$(26)
100 LET X=INT(RND(1)*26)+1
```



8

NAME NAME

This program in Graphics 2 asks for your name and then prints it out 12 times in large letters. Graphics 2 is nice for bold statements. Here is what the program looks like on the screen and the scrambled version of the program. Put it together and print the names of your family and friends. (Joshua is my son's name.)

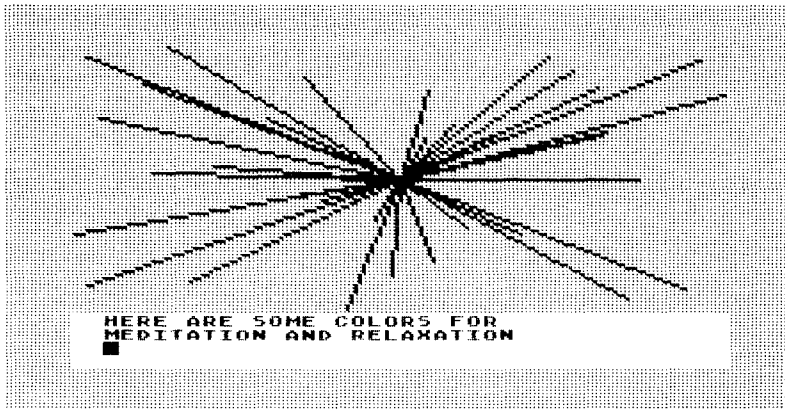


```
10 PRINT #6;A$
20 FOR Z=1 TO 12
30 DIM A$(50)
40 PRINT #6;"WHAT IS YOUR NAME?"
50 PRINT #6;A$;" "
60 PRINT #6;"WHAT A NICE NAME!"
70 NEXT Z
80 GRAPHICS 2:COLOR 1
90 INPUT A$
100 PRINT #6
```

9

MEDITATION

A screen dump can hardly capture this program which cycles through all of the colors and luminances of the Atari computer while slowly drawing a starburst. The program keeps on changing but slowly enough to allow you to relax and look. Here's an inadequate black and white dump of the program as well as the scrambled program:



```
10 NEXT X
20 DRAWTO RND(1)*159,RND(1)*80
30 PLOT 80,40
40 GRAPHICS 2
50 GRAPHICS CLR
60 PRINT "HERE ARE SOME COLORS FOR"
70 GOTO 10
80 FOR Y=1 TO 75:NEXT Y
90 SETCOLOR 4,RND(1)*16,RND(1)*15
100 FOR X=1 TO 50
110 PRINT "MEDITATION AND RELAXATION"
```

10

AN OPTIMISTIC PSYCHIATRIST'S PROJECTIVE TEST

This test asks you for a color preference. All the responses are in full color and are positive because the psychiatrist that taught me the test never found a person she didn't believe in. The black and white can't show you the passionate red, the sky blue, or the bright yellow but your Atari can. Try to unscramble the program and then add color upon color until you can make positive statements corresponding to all the 128 colors and luminances that you can control on the Atari. Finding 128 positive statements will be a good exercise in itself, whether you can unscramble the program or not.

```
RUN
WHICH OF THESE THREE COLORS IS YOUR
FAVORITE.....
?RED■ RED      BLUE      YELLOW
```

```
YOU ARE
PASSIONATE
```

```
READY
■
```

THE SKY
IS THE LIMIT

READY
■

YOU SHINE
WITH JOY

READY
■

```
10 PRINT #6;"          WITH JOY"  
20 IF D#=B# THEN GOTO 2000  
30 POSITION 0,3  
40 SETCOLOR 0,13,14  
50 PRINT #6;"          THE SKY"  
60 GRAPHICS 2  
70 DIM A$(10),B$(10),C$(10),D$(10)  
80 PRINT #6;"    IS THE LIMIT"  
90 END  
100 PRINT #6;"          YOU ARE"  
1000 PRINT "          RED          BLUE          YELL  
OW"  
1010 PRINT #6;"          YOU SHINE"
```

```
1020 A$="RED":B$="BLUE":C$="YELLOW"  
1030 SETCOLOR 0,4,7  
1040 POSITION 0,3  
1050 END  
2000 GRAPHICS 2  
2010 PRINT "FAVORITE....."  
2020 IF D$=A$ THEN GOTO 1000  
2030 PRINT "WHICH OF THESE THREE COLOR  
8 IS YOUR"  
2040 PRINT #6;"    PASSIONATE"  
2050 END  
3000 GRAPHICS 2  
3010 IF D$=C$ THEN GOTO 3000  
3020 INPUT D$  
3030 PRINT  
3040 SETCOLOR 0,8,7  
3050 POSITION 0,3
```

Answers

1

```
10 GRAPHICS 7:COLOR 1
20 PLOT 0,0
30 DRAWTO RND(1)*159,RND(1)*80
40 PRINT "HERE COMES THE SUN";
50 FOR X=1 TO 75:NEXT X
60 GOTO 20
```

In Graphics programs, especially those with GOTO statements and loops, which line the GOTO refers to and the position the Graphics statement plays within the loop are essential. In fact, try this program with the following simple line change and see what happens:

```
60 GOTO 10
```

2

```
10 GRAPHICS 7:COLOR 1
20 PLOT 0,0
30 DRAWTO RND(1)*159,RND(1)*80
40 PRINT "THERE GOES THE DRUNK";
50 FOR X=1 TO 100:NEXT X
60 GOTO 30
```

Compare the unscrambled version of this program with that of Puzzle 1. Notice that, once again, a simple change in the reference of a GOTO statement totally changes the nature of the program results.

3

```
10 PRINT "HOW IS THE WEATHER TODAY?"
20 DIM A$(50)
30 INPUT A$
40 PRINT
50 PRINT "WELL, WHETHER YOU LIKE IT"
60 PRINT "OR WHETHER NOT"
70 PRINT A$;" WEATHER"
80 PRINT "IS THE ONLY KIND OF WEATHER"
90 PRINT "WE GOT."
```

Almost any saying can be turned into a fun program using this form.
Try an input to the following:

Too many X's spoil the Y.

An X in time saves Y.

Better X than Y.

4

```
10 GRAPHICS 7:COLOR 1
20 PRINT "COLUMN (0 TO 159)"
30 INPUT A
40 PRINT "ROW (0 TO 79)"
50 INPUT B
60 PLOT A,B
70 GOTO 20
```

After you understand how this program works, try to do it in the different Atari Graphics modes. It is a good way to become familiar with the variety of screens the Atari allows you to use in your programs.

5

```
10 PRINT "I AM CRAZY ABOUT ADDING BUT"  
20 PRINT "DON'T ASK ME WHY. PLEASE"  
30 PRINT "GIVE ME TWO NUMBERS TO ADD."  
40 PRINT "FIRST NUMBER-";  
50 INPUT X  
60 PRINT "SECOND NUMBER-";  
70 INPUT Y  
80 PRINT "THANK YOU. THAT WAS DELICIOUS!"  
90 PRINT "THE ANSWER IS A GLORIOUS ";X  
+Y
```

Of course, this program can easily be extended to multiplication. See if you can get a bit more complex and extend it to division so the answer comes out even and to subtraction so the answer comes out positive. This will require a few more lines of code guaranteeing these conditions.

6

```
10 LET X=INT(RND(1)*11)  
20 PRINT "WHAT NUMBER COMES AFTER ";X  
30 INPUT Y  
40 IF Y=X+1 THEN PRINT "RIGHT ON!!!":E  
ND  
50 IF Y<>X+1 THEN PRINT "TRY AGAIN. Y  
OU CAN DO IT!!!"  
60 GOTO 30
```

Here are some interesting modifications:

Can you tell me what 5 times X is?

Can you tell me what 5 times X divided by 3 is? (In this case set the program up so that the answer comes out even.)

7

```
10 DIM A$(26)
20 A$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
30 LET X=INT(RND(1)*26)+1
40 PRINT "WHAT LETTER COMES ";
50 PRINT X;"TH IN THE ALPHABET?"
60 DIM B$(1)
70 INPUT B$
80 IF A$(X,X)=B$ THEN PRINT "RIGHT":EN
D
90 PRINT "GUESS AGAIN"
100 GOTO 70
```

Notice that the alphabet is stored in this program in A\$ which was dimensioned to hold 26 letters: DIM A\$(26). Also notice that the key to the program is the powerful BASIC command A\$(X,X) which allows you to choose the Ath member of A\$. This is used in many code and word programs. In its more complex form it has two variables A\$(X,Y) which will allow you to use all the members of A\$ from X to Y. Thus, if A\$ contained the alphabet then A\$(X,Y) could have X=2 and Y=5 as values. It would become A\$(2,5) and the command 10 PRINT A\$(2,5) would result in the computer printing out:

BCDE

8

```
10 GRAPHICS 2:COLOR 1
20 PRINT #6;"WHAT IS YOUR NAME?"
30 DIM A$(50)
40 INPUT A$
50 PRINT #6;A$
60 PRINT #6;"WHAT A NICE NAME!"
70 PRINT #6
80 FOR Z=1 TO 12
90 PRINT #6;A$;" ";
100 NEXT Z
```



```

10 GRAPHICS 7
20 PRINT "HERE ARE SOME COLORS FOR"
30 PRINT "MEDITATION AND RELAXATION"
40 SETCOLOR 4,RND(1)*16,RND(1)*15
50 FOR X=1 TO 50
60 PLOT 80,40
70 DRAWTO RND(1)*159,RND(1)*80
80 FOR Y=1 TO 75:NEXT Y
90 NEXT X
100 GRAPHICS CLR
110 GOTO 10

```

Play with the Atari SETCOLOR command and Graphics can provide beautiful effects even for the beginning programmer. Try to change the colors, the luminances, and the pauses that keep the color on the screen. (This is done with empty FOR/NEXT loops like FOR X=1 TO 100:NEXT X which is empty since it tells you to do nothing during those cycles.) You can get pretty good at experimenting with these powerful ideas built into the BASIC of the Atari.

```

10 DIM A$(10),B$(10),C$(10),D$(10)
20 A$="RED":B$="BLUE":C$="YELLOW"
30 PRINT "WHICH OF THESE THREE COLORS
IS YOUR"
40 PRINT "FAVORITE....."
50 PRINT
60 PRINT "      RED      BLUE      YELLOW"
"
70 INPUT D$
80 IF D$=A$ THEN GOTO 1000
90 IF D$=B$ THEN GOTO 2000
100 IF D$=C$ THEN GOTO 3000
1000 GRAPHICS 2
1010 SETCOLOR 0,4,7
1020 POSITION 0,3
1030 PRINT #6;"      YOU ARE"
1040 PRINT #6;"      PASSIONATE"
1050 END
2000 GRAPHICS 2
2010 SETCOLOR 0,8,7

```

```
2020 POSITION 0,3
2030 PRINT #6;"      THE SKY"
2040 PRINT #6;"      IS THE LIMIT"
2050 END
3000 GRAPHICS 2
3010 SETCOLOR 0,13,14
3020 POSITION 0,3
3030 PRINT #6;"      YOU SHINE"
3040 PRINT #6;"      WITH JOY"
3050 END
```

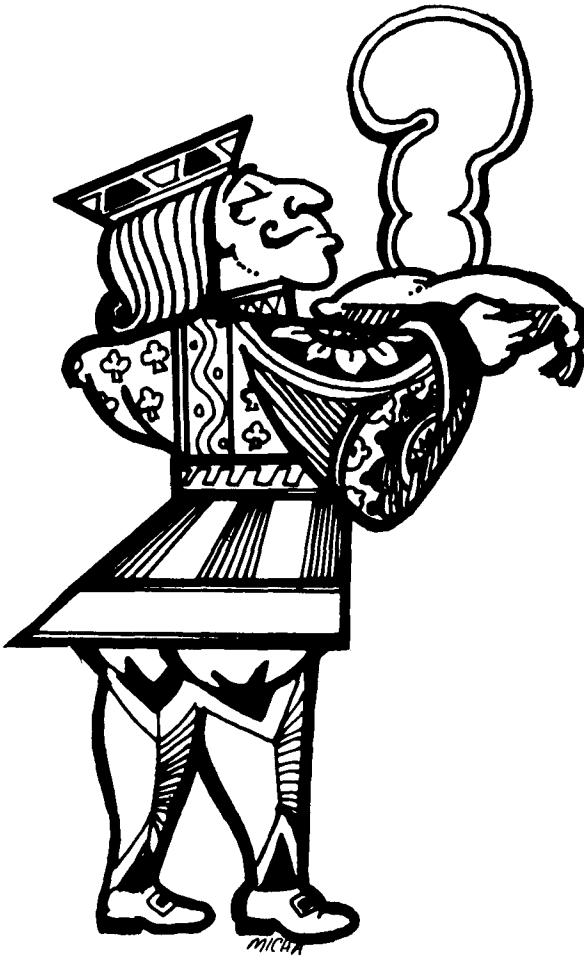
Here are a few color character equivalents I've managed to come up with:

- orange—abundant and nourishing
- pink—shy but perceptive
- dark blue—deep and curious



Missing Line Puzzles

In these puzzles, one line is missing from the programs. The missing line is indicated by the line number and the statement PRINT"?????????????". In choosing what lines to drop out, the main consideration was to leave out some essential part of the program structure. The screen dumps should give you enough of an idea of how the program runs to allow you to think your way through to the full program. If you come up with a line that works and is not the one given in the answer, please write and let me know. There are many ingenious ways to solve programming problems.



1

COUNTING BACKWARDS

This program asks you for an integer and then counts backwards from that number to 1 and prints out the list.

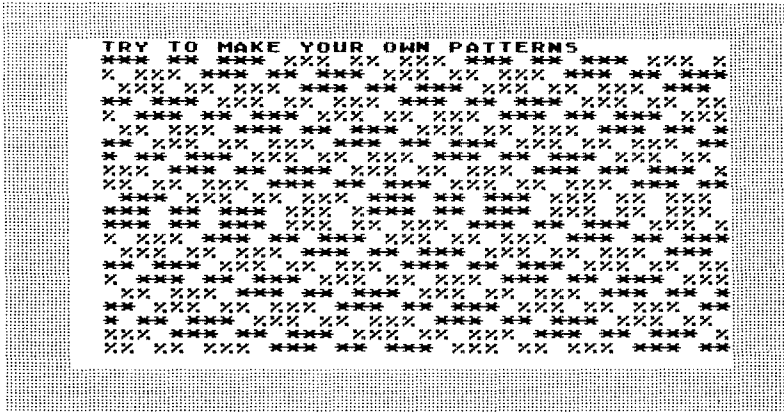
```
RUN
WATCH ME COUNT BACKWARDS.
WHAT NUMBER SHOULD I START FROM?
?15
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
READY
■
```

```
10 PRINT "WATCH ME COUNT BACKWARDS."
20 PRINT "WHAT NUMBER SHOULD I START FROM?"
30 INPUT X
40 FOR Z=1 TO X
50 PRINT X
60 PRINT "?????????????????"
70 NEXT Z
```

2

PATTERN MAKING

This program prints out a pattern. One aspect of the pattern is left out for you to figure out.



```
10 PRINT "HERE'S A EASY PATTERN TO PRI  
NT WITH A COMPUTER."  
20 PRINT "TRY TO MAKE YOUR OWN PATTERN  
8"  
30 FOR X=1 TO 750:NEXT X  
40 PRINT "?????????????????"  
50 PRINT "%X% %X %X% ";  
60 GOTO 40
```

3

A GOOD DAY

This program asks you if it's a good day for you. Here are the responses:

```
READY  
RUN  
IS THIS A GOOD DAY FOR YOU?  
? YES  
SHARE IT WITH YOUR FRIENDS  
READY  
■
```



```
MAYBE THIS WILL MAKE YOU FEEL BETTER..  
■
```

And here is the program with the missing line of code:

```
10 PRINT "IS THIS A GOOD DAY FOR YOU?"
20 PRINT "?????????????????"
30 INPUT A$
40 IF A$="YES" THEN PRINT "SHARE IT WITH YOUR FRIENDS":END
50 IF A$="NO" THEN GOTO 100
100 GRAPHICS 7:COLOR 1
110 PRINT "MAYBE THIS WILL MAKE YOU FEEL BETTER..."
120 PLOT 159,80
130 DRAWTO RND(1)*159,RND(1)*80
140 GOTO 120
```

4

GUESS

This is a letter counting game. It flashes a word on the screen and asks you to guess how many letters it has. You can count the letters but estimating the length of the word is more fun.



Here is the program with the missing line of code:

```
10 DIM A$(20),B$(20),C$(20)
20 A$="HOPEFUL"
30 B$="INDEPENDENCE"
40 C$="TROUBLE"
50 PRINT "HERE ARE SOME WORDS TO STUDY
FOR A FEW "
60 PRINT "SECONDS. GUESS HOW MANY LET
TERS ARE"
70 PRINT "IN EACH WORD."
80 FOR X=1 TO 750:NEXT X
100 PRINT "?????????????????????"
110 PRINT #6;A$
120 PRINT "YOUR GUESS?"
130 INPUT W
140 IF W=LEN(A$) THEN PRINT "GREAT! TR
Y ANOTHER":FOR X=1 TO 750:NEXT X:GOTO
200
```

```
150 IF W<>LEN(A*) THEN PRINT "TAKE AND  
THER LOOK":FOR X=1 TO 750:NEXT X:GOTO  
100  
200 GRAPHICS 2:COLOR 1  
210 PRINT #6;B*  
220 PRINT "YOUR GUESS?"  
230 INPUT Z  
240 IF Z=LEN(B*) THEN PRINT "GREAT! TR  
Y ANOTHER":FOR X=1 TO 750:GOTO 300  
250 IF Z<>LEN(B*) THEN PRINT "TAKE AND  
THER LOOK":FOR X=1 TO 750:NEXT X:GOTO  
200  
300 GRAPHICS 2:COLOR 1  
310 PRINT #6;C*  
320 PRINT "YOUR GUESS?"  
330 INPUT H  
340 IF H=LEN(C*) THEN PRINT "GREAT! YO  
U GOT ALL THREE":END  
350 IF H<>LEN(C*) THEN PRINT "TAKE AND  
THER LOOK":FOR X=1 TO 750:NEXT X:GOTO  
300
```

5

NEXT LETTER

This program asks for a letter and then gives you the next letter in the alphabet. It won't be tricked if you ask for the letter after Z.

```

RUN
GIVE ME A LETTER IN THE ALPHABET
AND I'LL TELL YOU THE NEXT LETTER.
DON'T TRY TO TRICK ME WITH Z BECAUSE
I'VE ALREADY LEARNED THAT IT IS THE
LAST LETTER.
?M
THE NEXT LETTER IS X

READY
RUN
GIVE ME A LETTER IN THE ALPHABET
AND I'LL TELL YOU THE NEXT LETTER.
DON'T TRY TO TRICK ME WITH Z BECAUSE
I'VE ALREADY LEARNED THAT IT IS THE
LAST LETTER.
?R
THE NEXT LETTER IS S

READY
■
```

```

10 PRINT "GIVE ME A LETTER IN THE ALPH
ABET"
20 PRINT "AND I'LL TELL YOU THE NEXT L
ETTER."
30 PRINT "DON'T TRY TO TRICK ME WITH Z
BECAUSE"
40 PRINT "I'VE ALREADY LEARNED THAT IT
IS THE"
50 PRINT "LAST LETTER."
60 DIM A$(26),B$(1)
70 LET A$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
80 INPUT B$
90 FOR X=1 TO 26
100 PRINT "?????????????????????????"
110 NEXT X
```

6

THIRD LETTER

Here is a puzzle that uses the same concept as Puzzle 5. You should be able to get this one more easily:

```
RUN
WHAT IS YOUR NAME?
?SOCRATES
DID YOU KNOW THAT THE THIRD
LETTER IN YOUR NAME IS..C
READY
■
```

```
10 PRINT "WHAT IS YOUR NAME?"
20 DIM A$(25)
30 INPUT A$
40 PRINT "DID YOU KNOW THAT THE THIRD"
50 PRINT "LETTER IN YOUR NAME IS..!"
60 PRINT "????????????????????"
```

7

NEXT NUMBER

Recently you tried a "next letter" puzzle. Here is a next number puzzle:

```
RUN
WHAT NUMBER COMES AFTER...38
?37
TRY AGAIN...
?32
TRY AGAIN...
?1
TRY AGAIN...
?56
TRY AGAIN...
?39
ABSOLUTELY!
READY
■
```

```
10 PRINT "WHAT NUMBER COMES AFTER..";
20 LET X=INT(RND(1)*100)+1
30 PRINT X
40 INPUT Y
50 IF Y=X+1 THEN PRINT "ABSOLUTELY!":E
ND
60 IF Y<>X+1 THEN PRINT "TRY AGAIN..."
70 PRINT "????????????????????"
```

8

CALCULATE

Here is a simple arithmetic challenge:

```

RUN
PICK A NUMBER FROM 1 TO 20
?5
WHAT IS 4 TIMES THAT NUMBER MINUS 3?
?14
TRY AGAIN...
?12
TRY AGAIN...
?17
YOU GOT IT!!
READY
■
```

Here is the program for it with a missing line:

```

10 PRINT "PICK A NUMBER FROM 1 TO 20"
20 INPUT X
30 PRINT "WHAT IS 4 TIMES THAT NUMBER
MINUS 3?"
40 INPUT Y
50 PRINT "?????????????????????"
60 IF Y=Z THEN PRINT "YOU GOT IT!!":EN
D
70 IF Y<>Z THEN PRINT "TRY AGAIN..."
80 BOTO 40
```

9

COMPARISON

This program generates two random numbers and asks you which is the larger. It does not give you a second chance but gives you another problem instead.

```
HERE'S A REAL SIMPLE MATH GAME
WHICH NUMBER IS LARGER
21          OR          29
?29
RIGHT, TRY ANOTHER ONE
WHICH NUMBER IS LARGER
15          OR          48
?15
SORRY TRY ANOTHER EXAMPLE
WHICH NUMBER IS LARGER
43          OR          44
?44
RIGHT, TRY ANOTHER ONE
WHICH NUMBER IS LARGER
27          OR          18
?27■
```

Figure out the missing line:

```
10 PRINT "HERE'S A REAL SIMPLE MATH GA
ME"
20 LET X=INT(RND(1)*50)
30 LET Y=INT(RND(1)*50)
40 IF X=Y THEN GOTO 20
50 PRINT "WHICH NUMBER IS LARGER"
60 PRINT
70 PRINT X, "OR", Y
80 PRINT
90 INPUT Z
100 IF Z=X THEN GOTO 1000
110 PRINT "?????????????????????????"
1000 IF X>Y THEN PRINT "RIGHT, TRY AND
THER ONE":GOTO 20
1010 PRINT " SORRY TRY ANOTHER EXAMPLE
":GOTO 20
2000 IF Y>X THEN PRINT "RIGHT, TRY AND
THER ONE":GOTO 20
2010 PRINT " SORRY TRY ANOTHER EXAMPLE
":GOTO 20
```

10

DICE

This program sets up the computer to roll a pair of dice and then gives you the total of the two rolls. It can be incorporated into many game programs.

```

RUN
I KNOW HOW TO ROLE DICE
HERE'S MY ROLE:
DIE 1:      1
DIE 2:      3
TOTAL:      4
DO YOU WANT ME TO ROLL AGAIN?
?YES
HERE'S MY ROLE:
DIE 1:      5
DIE 2:      3
TOTAL:      8
DO YOU WANT ME TO ROLL AGAIN?
?NO
```

Fill in the missing line:

```

10 PRINT "I KNOW HOW TO ROLE DICE"
20 PRINT "HERE'S MY ROLE:"
30 LET X=INT(RND(1)*6)+1
40 LET Y=INT(RND(1)*6)+1
50 PRINT
60 PRINT "DIE 1:",X
70 PRINT
80 PRINT "DIE 2:",Y
90 PRINT
100 PRINT "?????????????????????"
110 PRINT
120 PRINT "DO YOU WANT ME TO ROLL AGAI
N?"
130 CLR
140 DIM A$(3)
150 INPUT A$
160 IF A$="YES" THEN GOTO 20
170 PRINT "THAT'S ALL FOLKS.":END
```


Answers

1

```
10 PRINT "WATCH ME COUNT BACKWARDS."  
20 PRINT "WHAT NUMBER SHOULD I START F  
ROM?"  
30 INPUT X  
40 FOR Z=1 TO X  
50 PRINT X  
60 LET X=X-1  
70 NEXT Z
```

The missing line decreases the size of X each time the program runs through the loop. You could count backwards by 2's, 3's, or any other number.

2

```
10 PRINT "HERE'S A EASY PATTERN TO PRI  
NT WITH A COMPUTER."  
20 PRINT "TRY TO MAKE YOUR OWN PATTERN  
S"  
30 FOR X=1 TO 750:NEXT X  
40 PRINT "*** ** *** ";  
50 PRINT "%% % %%% ";  
60 GOTO 40
```

It is very easy to make patterns with your Atari. In fact, one chapter of this book contains a whole series of moderately complex pattern puzzles. What you have to look for in making patterns is the arrangement of blank spaces as much as the placement of symbols. The ; is what creates a continuous pattern across the screen and, since the line scrolls around, you will often find patterns that don't look at all like you expect them to. It is a delightful idle activity to explore patterns at random and develop an intuitive sense of how lines like these will look when repeated on the screen in an infinite loop:

```

$$$$$  ?  $$$@$$@ $$$
%[%[%%  %[%[%%  %[%[%%
@@@ ~~~ [[ [ ]]] ~~~ @@@

```

3

```

10 PRINT "IS THIS A GOOD DAY FOR YOU?"
20 DIM A$(3)
30 INPUT A$
40 IF A$="YES" THEN PRINT "SHARE IT WITH YOUR FRIENDS":END
50 IF A$="NO" THEN GOTO 100
100 GRAPHICS 7:COLOR 1
110 PRINT "MAYBE THIS WILL MAKE YOU FEEL BETTER..."
120 PLOT 159,80
130 DRAWTO RND(1)*159,RND(1)*80
140 GOTO 120

```

The challenge here is to define the lower left-hand point of the Graphics 7 screen. Try to do the same program in other Graphics modes and see how the modes differ. You can mix Graphics modes within a program and it is useful to explore them all in order to be able to choose the right Graphics format for your programming purposes.

4

```

10 DIM A$(20),B$(20),C$(20)
20 A$="HOPEFUL"
30 B$="INDEPENDENCE"
40 C$="TROUBLE"
50 PRINT "HERE ARE SOME WORDS TO STUDY FOR A FEW "
60 PRINT "SECONDS. GUESS HOW MANY LETTERS ARE"
70 PRINT "IN EACH WORD."
80 FOR X=1 TO 750:NEXT X
100 GRAPHICS 2:COLOR 1

```

```

110 PRINT #6;A$
120 PRINT "YOUR GUESS?"
130 INPUT W
140 IF W=LEN(A$) THEN PRINT "GREAT! TR
Y ANOTHER":FOR X=1 TO 1500:NEXT X:GOTO
200
150 IF W<>LEN(A$) THEN PRINT "TAKE AND
THER LOOK":FOR X=1 TO 750:NEXT X:GOTO
100
200 GRAPHICS 2:COLOR 1
210 PRINT #6;B$
220 PRINT "YOUR GUESS?"
230 INPUT Z
240 IF Z=LEN(B$) THEN PRINT "GREAT! TR
Y ANOTHER":FOR X=1 TO 1500:GOTO 300
250 IF Z<>LEN(B$) THEN PRINT "TAKE AND
THER LOOK":FOR X=1 TO 750:NEXT X:GOTO
200
300 GRAPHICS 2:COLOR 1
310 PRINT #6;C$
320 PRINT "YOUR GUESS?"
330 INPUT H
340 IF H=LEN(C$) THEN PRINT "GREAT! YO
U GOT ALL THREE":END
350 IF H<>LEN(C$) THEN PRINT "TAKE AND
THER LOOK":FOR X=1 TO 1500:NEXT X:GOTO
300

```

This program uses Graphics 2, a large letter print mode. The program also uses a command you may not be familiar with but which is very useful: LEN(A\$). This command gives you the length of a word or sequence of letters stored in a string. Thus, if you DIM A\$(50) and INPUT A\$ as "THE", LEN(A\$)=3. It can let you know how much room is left in a string, can create size-word relationships, etc. You will probably encounter it in many programs you copy from computer magazines.

5

Here is the program to fill out:

```

10 PRINT "GIVE ME A LETTER IN THE ALPH
ABET"
20 PRINT "AND I'LL TELL YOU THE NEXT L
ETTER."

```

```

30 PRINT "DON'T TRY TO TRICK ME WITH Z
  BECAUSE"
40 PRINT "I'VE ALREADY LEARNED THAT IT
  IS THE"
50 PRINT "LAST LETTER."
60 DIM A$(26),B$(1)
70 LET A$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
80 INPUT B$
90 FOR X=1 TO 26
100 IF B$=A$(X,X) THEN PRINT "THE NEXT
  LETTER IS ";A$(X+1,X+1)
110 NEXT X

```

The missing line involves using a function, A\$(X,X) which gives you the Xth member of A\$. It is more fully explained in the answers to other puzzles that use the same function. What you should note here is that A\$(X+1,X+1) gives you the letter or symbol stored after X in A\$. What do you think you'd get from A\$(X+3,X+3)?

6

```

10 PRINT "WHAT IS YOUR NAME?"
20 DIM A$(25)
30 INPUT A$
40 PRINT "DID YOU KNOW THAT THE THIRD"
50 PRINT "LETTER IN YOU NAME IS..";
60 PRINT A$(3,3)

```

7

```

10 PRINT "WHAT NUMBER COMES AFTER..";
20 LET X=INT(RND(1)*100)+1
30 PRINT X
40 INPUT Y
50 IF Y=X+1 THEN PRINT "ABSOLUTELY!":E
  ND
60 IF Y<>X+1 THEN PRINT "TRY ABAIN..."
70 GOTO 40

```

This puzzle deals with setting up a loop to let a person try to correct a wrong answer. The reference of GOTO statements is essential to how a program runs. If you tried

```
70 GOTO 20
```

the target number and, therefore, the correct answer would be changed. It is useful to deliberately mess up your own programs and see what happens. Sometimes you'll make interesting and useful discoveries. Other times you'll internalize ERROR statements and help yourself create bug free programs.

8

```
10 PRINT "PICK A NUMBER FROM 1 TO 20"  
20 INPUT X  
30 PRINT "WHAT IS 4 TIMES THAT NUMBER  
MINUS 3?"  
40 INPUT Y  
50 LET Z=4*X-3  
60 IF Y=Z THEN PRINT "YOU GOT IT!!":EN  
D  
70 IF Y<>Z THEN PRINT "TRY AGAIN..."  
80 GOTO 40
```

In this case, you have to figure out how to perform the calculation asked for correctly and then give its result a new variable name, Z. If you forget to use this new variable you won't be able to use the result of the calculation in other parts of the program without doing a recalculation. In many programs, the introduction of new variables to indicate the results of processes or calculations is a convenient programming tool.

9

```
10 PRINT "HERE'S A REAL SIMPLE MATH GA  
ME"  
20 LET X=INT(RND(1)*50)  
30 LET Y=INT(RND(1)*50)  
40 IF X=Y THEN GOTO 20  
50 PRINT "WHICH NUMBER IS LARGER"  
60 PRINT
```

```

70 PRINT X,"OR",Y
80 PRINT
90 INPUT Z
100 IF Z=X THEN GOTO 1000
110 IF Z=Y THEN GOTO 2000
1000 IF X>Y THEN PRINT "RIGHT, TRY AND
THER ONE":GOTO 20
1010 PRINT " SORRY TRY ANOTHER EXAMPLE
":GOTO 20
2000 IF Y>X THEN PRINT "RIGHT, TRY AND
THER ONE":GOTO 20
2010 PRINT " SORRY TRY ANOTHER EXAMPLE
":GOTO 20

```

This is another puzzle in which you have to figure out the correct GOTO reference. Instead of sending you back to an earlier part of the program as most of the GOTO puzzles did, this one sends you to the end of the program. It is important to realize that the GOTO command can allow you to jump all over your program.

10

```

10 PRINT "I KNOW HOW TO ROLE DICE"
20 PRINT "HERE'S MY ROLE:"
30 LET X=INT(RND(1)*6)+1
40 LET Y=INT(RND(1)*6)+1
50 PRINT
60 PRINT "DIE 1:",X
70 PRINT
80 PRINT "DIE 2:",Y
90 PRINT
100 PRINT "TOTAL:",X+Y
110 PRINT
120 PRINT "DO YOU WANT ME TO ROLL ABAI
N?"
130 CLR
140 DIM A$(3)
150 INPUT A$
160 IF A$="YES" THEN GOTO 20
170 PRINT "THAT'S ALL FOLKS.":END

```

Line 100 lets you both add the total and print it out at the same time. There are times when it is convenient to use the PRINT statement in conjunction with a computation, especially if you do not want to store the result of the computation in any other part of the program.

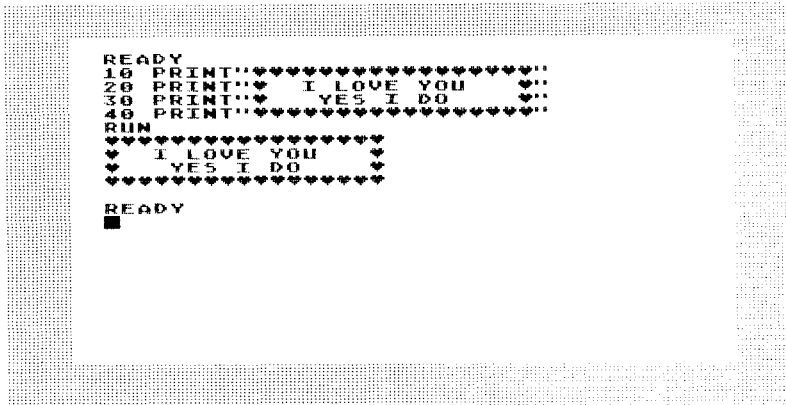


Control Graphics Puzzles

The Atari computer has a control graphics keyboard that is invisible unless you press the control key **CTRL** and then a letter. (Consult your **ATARI BASIC Reference Manual** for a diagram of the Control Graphics Keyboard.)

The shapes on the CTRL keyboard can be used to make very interesting designs and pictures. They can also be mixed with letters to make borders and designs within your ordinary text. They do not need a special Graphics command and print straight from the keyboard. They can also be embodied in PRINT statements along with numbers and letters and can be stored in dimensioned strings the way letters can. Here are a few simple examples of the way control character graphics programs work. They should help familiarize you with their function and help you understand how to solve the puzzles in this chapter.

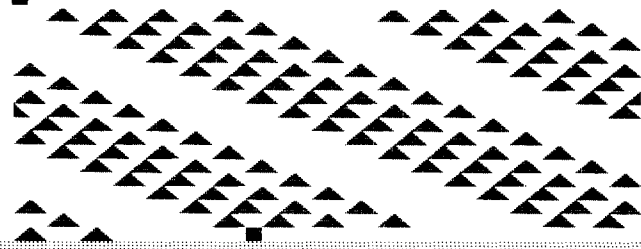
The first sample simply uses the combination **[CTRL][.]** to produce a heart which is used as a border around a declaration of love:



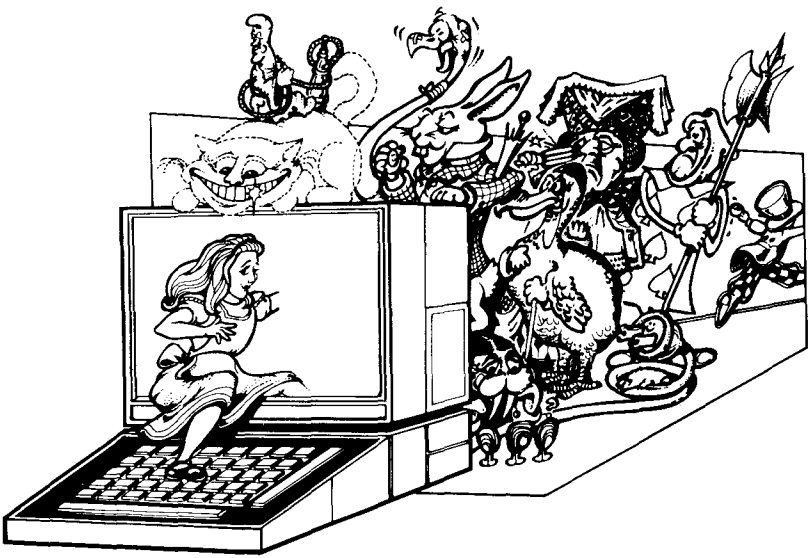
```
READY
10 PRINT"❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤"
20 PRINT"❤ I LOVE YOU ❤"
30 PRINT"❤ YES I DO ❤"
40 PRINT"❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤"
RUN
❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤
❤ I LOVE YOU ❤
❤ YES I DO ❤
❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤❤
READY
■
```

The second example uses the numbers 1, 2, 3, and 4 along with the **[CTRL][.]** character to produce a field of numbers followed by diamonds that have the same value as the numbers. Notice that the whole program is just two lines long.


```
LIST
10 PRINT "▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲";
20 GOTO 10
READY
■
```



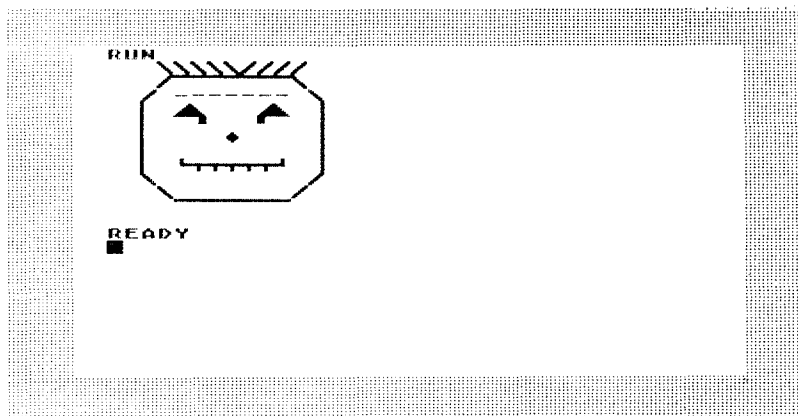
Now for some control graphics problems. Use the [CTRL] character chart and the sketch pad in the appendix to help you solve them if you haven't memorized the control characters and don't happen to have a computer available.



1

PLASTIC SURGERY I

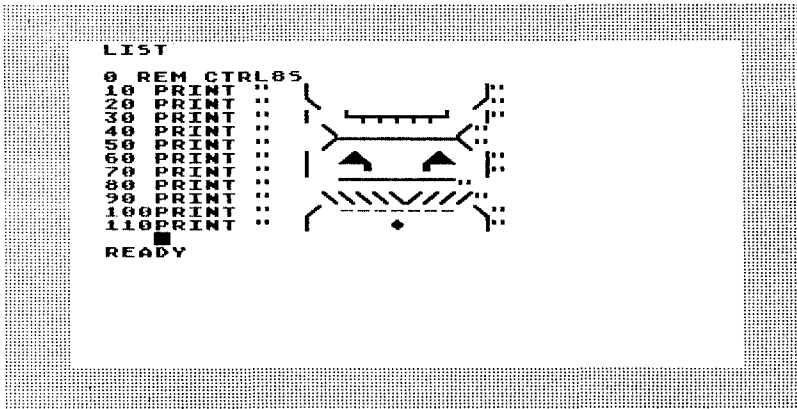
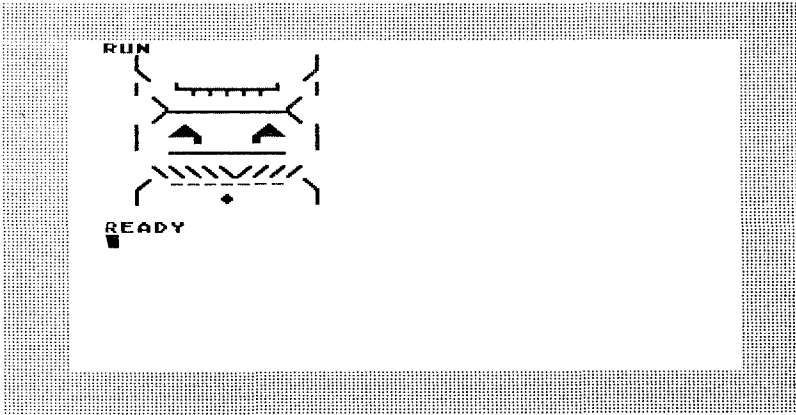
Here is a face, perhaps not the most beautiful one in the world but nevertheless one with character. Using [CTRL] graphics try to reconstruct it.



2

PLASTIC SURGERY II

Here's the same face scrambled up as well as the program that generates the scramble. Try to reconstruct the original face by reordering the line numbers without looking at the answers to [CTRL] Puzzle 1. The answer to this puzzle is exactly the same as the answer to Puzzle 1.

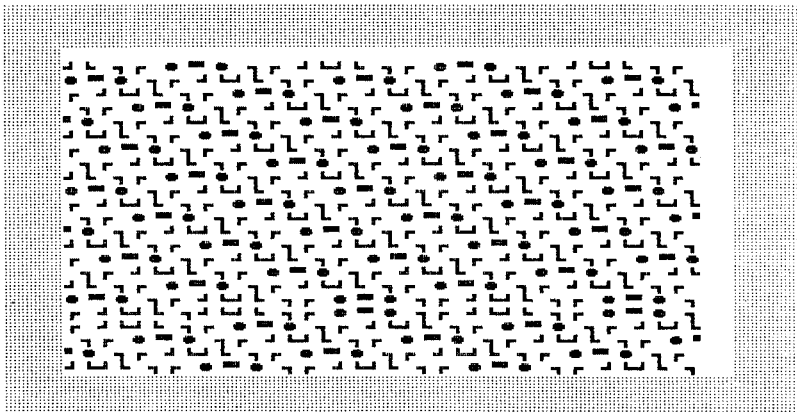


4 to 9

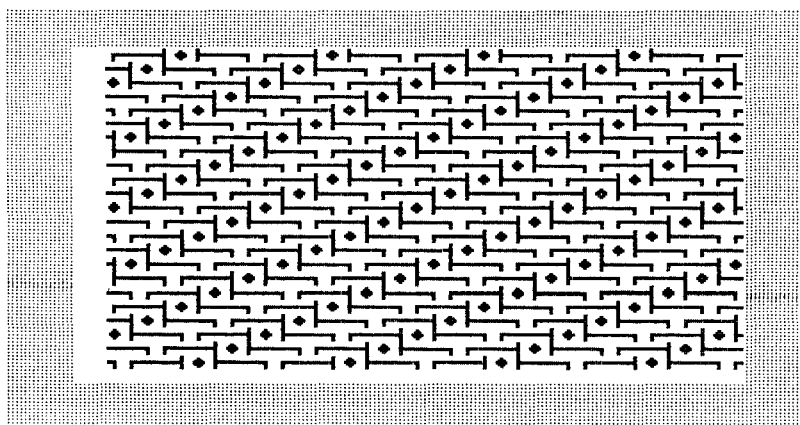
DESIGN DECIPHERING

This is a series of [CTRL] character designs for you to decipher and try to reproduce. Remember that some of the characters are made up of combinations of other characters. Also, the use of blank spacing is essential to get the design. I suggest you do some sketching and experimenting with combinations of [CTRL] characters in the course of trying to reproduce these designs. There is a sketch pad in the Appendix which is a reproduction of the grid on the Graphics O screen which uses regular print as well as [CTRL] graphics.

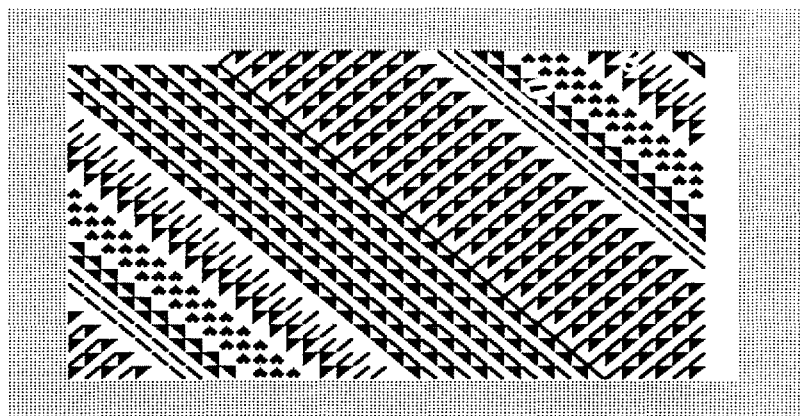
[CTRL] Design 4: Almost Music



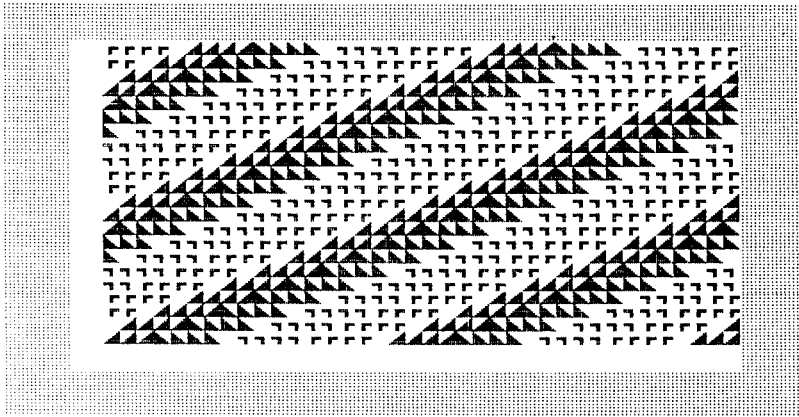
[CTRL] Design 5: Dizzy Roads, Shaky Bridges



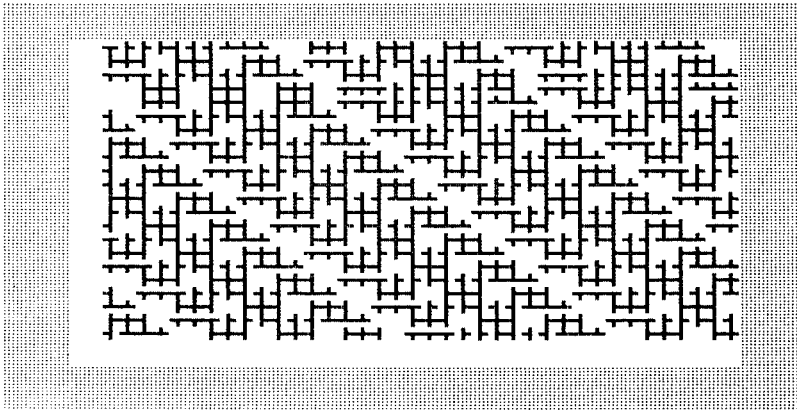
[CTRL] Design 6: Stripes at Cross Purposes



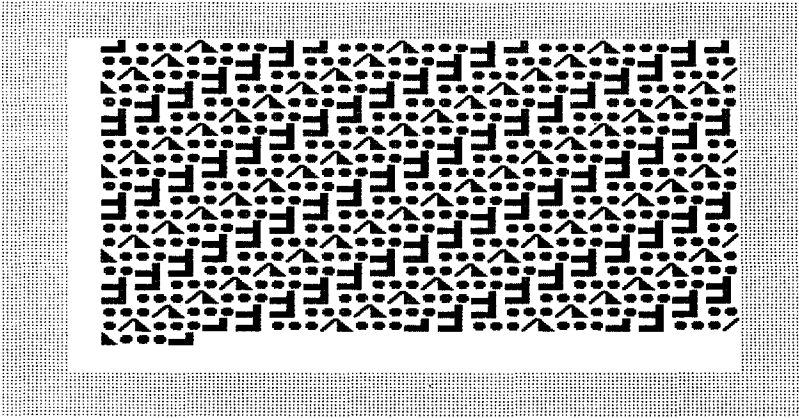
[CTRL] Design 7: Clear Cutting in the Mountains



[CTRL] Design 8: Fences and Roads



[CTRL] Design 9: Dense Environment



10

ALPHABET RECONSTRUCTION

Now that you've spent some time dealing with some fairly complicated designs, here's a simple alphabet to reconstruct. It doesn't use [CTRL] characters but does use the keyboard and the program structure of BASIC in the same way that the previous program did.

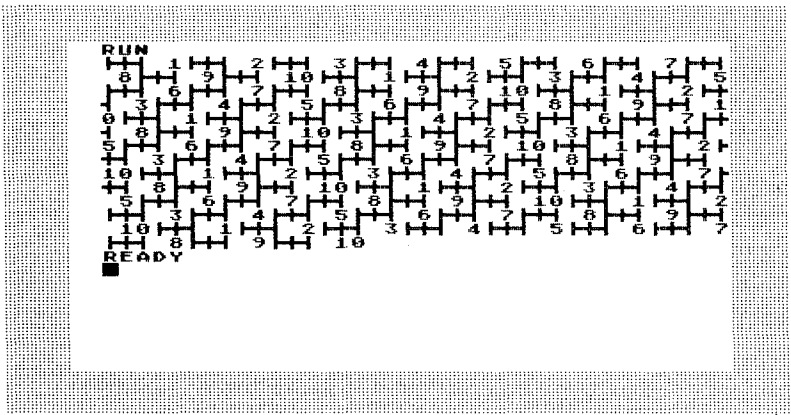
```

MNOP QRS & TUV W X & YZ ABC DEFG HIJK
LMNOP QRS & TUV W X & YZ ABC DEFG HIJK
K LMNOP QRS & TUV W X & YZ ABC DEFG HIJ
JK LMNOP QRS & TUV W X & YZ ABC DEFG HI
IJK LMNOP QRS & TUV W X & YZ ABC DEFG
HIJK LMNOP QRS & TUV W X & YZ ABC DEFG
HIJK LMNOP QRS & TUV W X & YZ ABC DEF
G HIJK LMNOP QRS & TUV W X & YZ BC DEF
G HIJK LMNOP QRS & TUV W X & YZ ABC DE
FG HIJK LMNOP QRS & TUV W X & YZ ABC D
EFG HIJK LMNOP QRS & TUV W X & YZ ABC
DEFG HIJK LMNOP QRS & TUV W X & YZ ABC
C DEFG HIJK LMNOP QRS & TUV W X & YZ A
BC DEFG HIJK LMNOP QRS & TUV W X & YZ
ABC DEFG HIJK LMNOP QRS & TUV W X & YZ
Z ABC DEFG HIJK LMNOP QRS & TUV W X &
YZ ABC DEFG HIJK LMNOP QRS & TUV W X &
& YZ ABC DEFG HIJK LMNOP QRS & TUV W X
```

11

NUMBER GRAPHICS

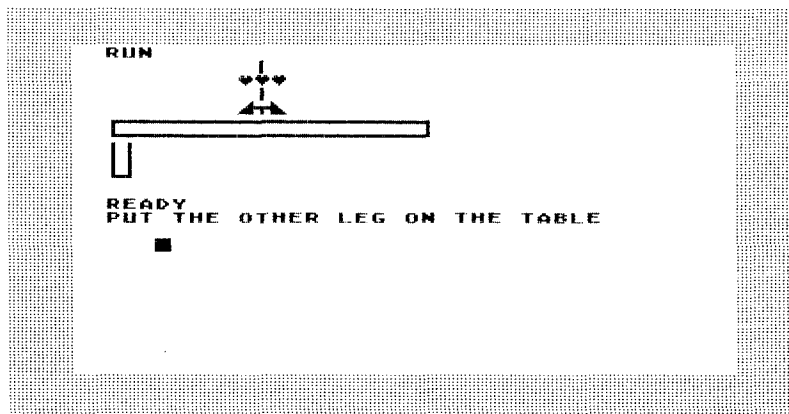
Now that you've had a chance to relax, here's a slightly more complex problem using a mix of [CTRL] graphics and numbers. Notice that the numbers change in the design in a regular way. The program structure is somewhat different than the ones you've seen before but the program is not much longer.



12

FINISH THE TABLE

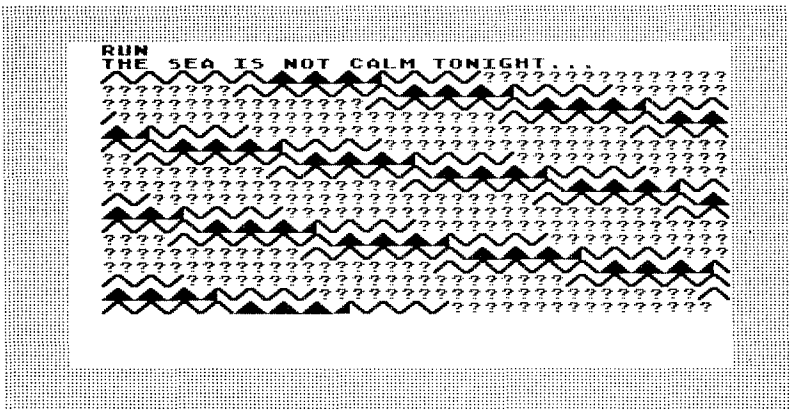
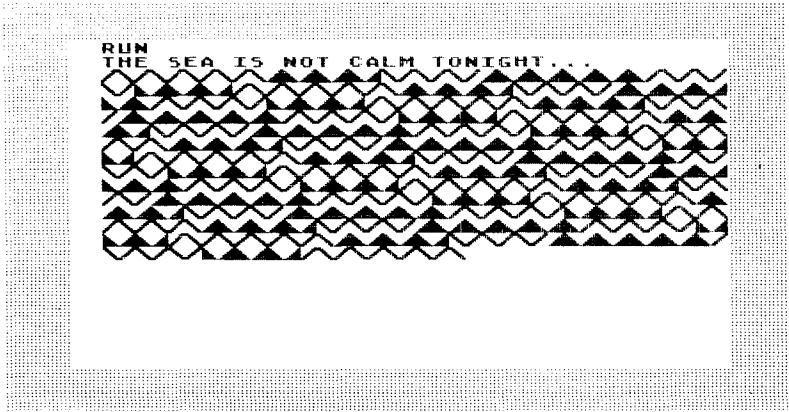
Put the missing leg on the table using [CTRL] graphics. Of course, you'll have to write the whole program for the table to do it.



13

STORMY WEATHER

The sea is not calm tonight as you can see from the screen dump below. Under that is a puzzling sea, one that questions the very nature of calmness. Can you write programs for each of these screen dumps?



14

SYMBOL ADDITION

Here is a game using numbers and [CTRL] characters. Three symbols represent numbers. You are asked to memorize the symbol/number equivalents and then do addition using the symbols. It is a good memory game that can be made quite complex. Write a program for this game.

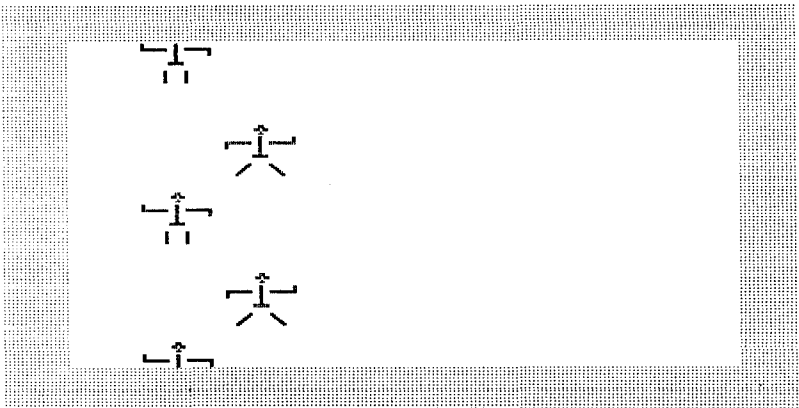
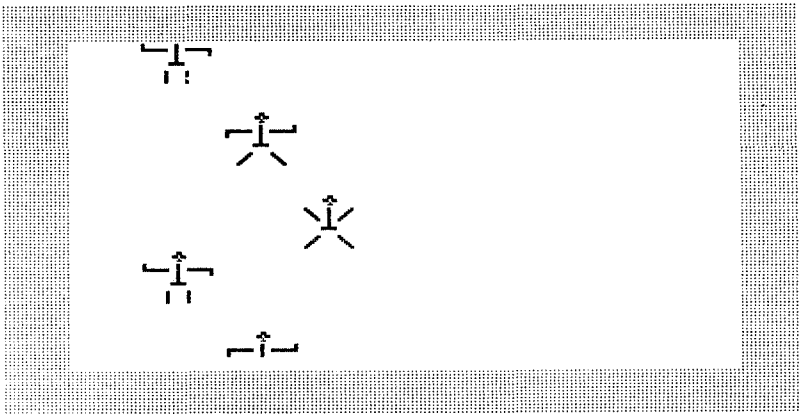
```
RUN
HERE'S A LITTLE SYMBOL-NUMBER GAME:
STUDY THIS TABLE OF VALUES:
  ♡ = 1
  + = 2
  ♣ = 3
WHEN YOU HAVE THE TABLE MEMORIZED
PRESS ?
?■
```

```
HOW MUCH DO THESE ADD UP TO?
♡ ♡ ♡ ♡ ♡ ?6
TRY ANOTHER TIME.
?3
TRY ANOTHER TIME.
?5
YOU GOT IT DOWN!
READY
■
```

15

SIMPLE ANIMATION

Here are two simple animations you can make with control character graphics. You can actually get quite complex with them, although, of course, you won't be able to duplicate the work of Walt Disney or George Lucas. With Graphics 8, you might be able to get close but here is a modest stick figure beginning. These two dumps show the repetitions of a figure that moves up the screen while changing position and posture. See if you can recreate the figures and have them move as in the screen dumps.



Answers

1 and 2

LIST

```
0 REM CTRL8U
10 PRINT " "
20 PRINT " "
30 PRINT " "
40 PRINT " "
50 PRINT " "
60 PRINT " "
70 PRINT " "
80 PRINT " "
90 PRINT " "
100PRINT " "
110PRINT " "
```



READY



LIST

```
0 REM CTRL8U
10 PRINT " "
20 PRINT " "
30 PRINT " "
40 PRINT " "
50 PRINT " "
60 PRINT " "
70 PRINT " "
80 PRINT " "
90 PRINT " "
100 PRINT " "
110 PRINT " "
```



READY

Notice that there are two answers given, one that duplicates the face exactly and a second that pushes the chin off at lines 100 and 110. When run they will give you the exact same thing. The reason that answer 2 seems pushed in is that your Atari automatically puts a space between the line number and the command following it. Since 10 to 99 have only two digits, they line up. 100 and 110 have three digits so they push the command over one space. You have to back it up in your mind to recon-

Clear Cutting in the Mountains

```
LIST
0 REM CTRL2U
10 PRINT "FFFF ▲▲▲▲▲";
20 GOTO 10
READY
■
```

Fences and Roads

```
LIST
0 REM CTRL10U
10 PRINT "FH FH FH";
20 PRINT "FH FH FH";
30 GOTO 10
READY
■
```

Dense Environment

```
LIST
0 REM CTRL11U
10 PRINT ".....^.....";
20 GOTO 10
READY
■
```

10

```
LIST
0 REM CTRL12U
10 PRINT "ABC DEFG HIJK LMNOP ";
20 PRINT "QRS & TUV W X & YZ ";
30 GOTO 10
READY
■
```

Notice that I used three lines in the program instead of two. Using the ; at the end of lines 20 and 30 creates the continuity of the program.

```
LIST
0 REM CTRLSU
10 FOR X=1 TO 10
20 FOR Y=1 TO 10
30 PRINT "HI ";Y;
40 NEXT Y
50 NEXT X
```

```
READY
```

Notice that the changes in the numbers throughout the program are determined by the nested FOR/NEXT loops. If you are not familiar with nested loops, it would be useful to trace the program step by step. For example, the X loop instructs you to print what is inside the Y loop 10 times and then go back to the start and keep on repeating the Y loop 10 full runs.

Once you understand this and trace it through you should be able to use nested loops in many different contexts.

This drawing problem can lead to dozens of others. Try to make up some for your family and friends, and have them be as creative in drawing and challenging you as well.

LIST

```
0 REM CTRL4U
10 PRINT " "
20 PRINT " "
30 PRINT " "
40 PRINT " "
50 PRINT " "
60 PRINT " "
70 PRINT " "
80 PRINT " "
90 PRINT " "
```

READY

LIST

```
0 REM CTRL6U
10 PRINT "THE SEA IS NOT CALM TONIGHT."
20 PRINT " "
30 PRINT " "
40 GOTO 20
```

READY

LIST

```
0 REM CTRL65
10 PRINT "THE SEA IS NOT CALM TONIGHT.
. ."
20 PRINT "~~~~~";
30 PRINT "????????????????????";
40 GOTO 20
■
READY
```

14

LIST 0,150

```
0 REM CTRL75
10 PRINT "HERE'S A LITTLE SYMBOL-NUMBE
R GAME:"
20 DIM A$(1),B$(1),C$(1)
30 LET A$="♥":LET B$="+":LET C$="▲"
40 PRINT "STUDY THIS TABLE OF VALUES:"

50 PRINT "    ♥ = 1"
60 PRINT "    + = 2"
70 PRINT "    ▲ = 3"
80 PRINT "
90 PRINT "
100 PRINT "WHEN YOU HAVE THE TABLE MEM
ORIZED"
110 PRINT "PRESS 7"
120 INPUT X
130 IF X=7 THEN PRINT "R"
140 IF X=7 THEN LET Y=2????????????????
150 ERROR- LET Z=0????????????????????
160 ERROR- LET Z=0????????????????????
170 PRINT "HOW MUCH DO THESE ADD UP TO
?"
180 FOR M=1 TO Z
190 IF Y=1 THEN PRINT "♥ ";
200 IF Y=2 THEN PRINT "+ ";
210 IF Y=3 THEN PRINT "▲ ";
220 NEXT M
230 INPUT N
240 IF N=Y*Z THEN PRINT "YOU GOT IT DO
WN!";END
250 PRINT "TRY ANOTHER TIME.":GOTO 230

■
READY
```

```

READY
LIST
0 REM CTRL111
10 PRINT "
20 PRINT "
30 PRINT "
40 PRINT "
50 PRINT "
60 FOR X=1 TO 100:NEXT X
70 PRINT "
80 PRINT "
90 PRINT "
100 PRINT "
110 PRINT "
120 FOR X=1 TO 100:NEXT X
130 GOTO 10
READY

```

```

0 REM CTRL11
10 PRINT "
20 PRINT "
30 PRINT "
40 PRINT "
50 PRINT "
59 PRINT "
60 FOR X=1 TO 100:NEXT X
70 PRINT "
80 PRINT "
90 PRINT "
100 PRINT "
110 PRINT "
120 FOR X=1 TO 100:NEXT X
130 PRINT "
140 PRINT "
150 PRINT "
160 PRINT "
170 PRINT "
180 FOR X=1 TO 100:NEXT X
190 GOTO 10
READY

```

The key structural elements of these programs could be described as:

PRINT Figure 1—PAUSE TO SEE IT

PRINT Figure 2—PAUSE TO SEE IT

—GO BACK TO FIGURE 1

Teasing out the structure of programs in this way can help you create your own programs or modify those other people design.

Appendix

Planning Sheets

These pages are designed to help you solve some of the problems in this book. They consist of a series of TV or monitor screens with room under them to write lines of code. They can be used in many different ways and some readers will certainly develop their own aids to solve the puzzles. Here are two examples of how they can be used for two puzzle versions of this simple program:

```
10 PRINT "HOW OLD ARE YOU?"
20 INPUT X
30 LET Y=1983-X
40 PRINT "YOU WERE BORN IN ";Y
```

Puzzle version 1:missing line of code

```
10 PRINT "HOW OLD ARE YOU?"
20 INPUT X
30 PRINT "???????????????"
40 PRINT "YOU WERE BORN IN ";Y
```

Puzzle version 2:scrambled line number version

```
10 INPUT X
20 PRINT "HOW OLD ARE YOU?"
30 PRINT "YOU WERE BORN IN ";Y
40 LET Y=1983-X
```

How old are you?

10 PR. "How old are you"

How old are you?

10 PR. "How old are you"

How old are you?

?

20 INPUT X

How old are you?

?

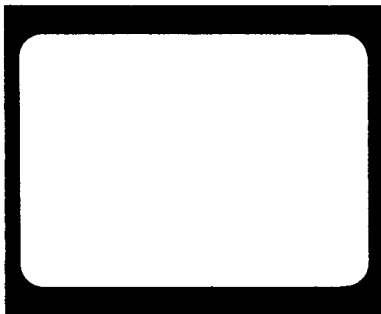
20 INPUT X

Same as above

30 LET X = 1983 - X

?

30 ? need to get to Y ↓

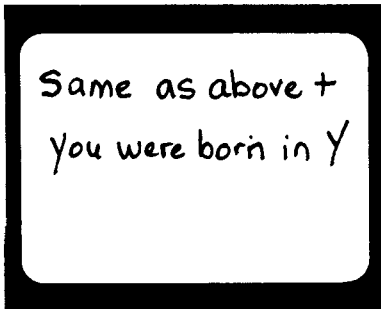


30 $Y = \text{year} - \text{how old}$



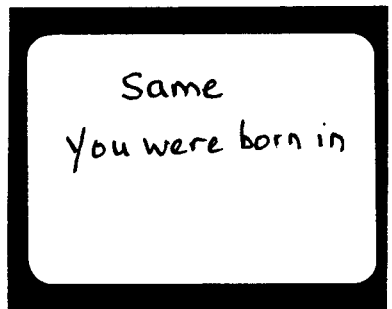
SOLUTION

30 LET $Y = 1983 - X$



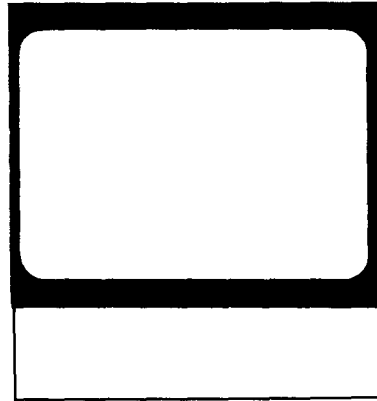
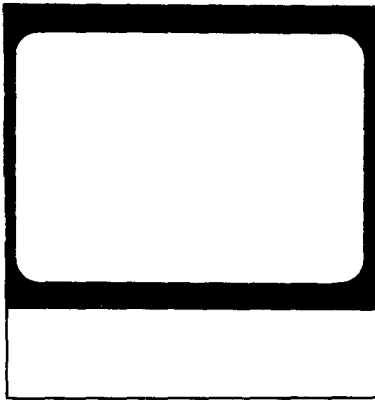
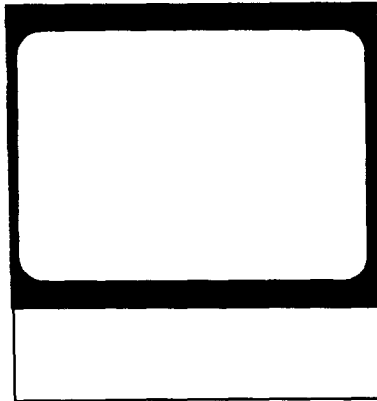
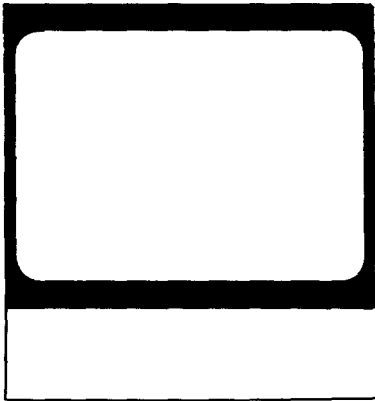
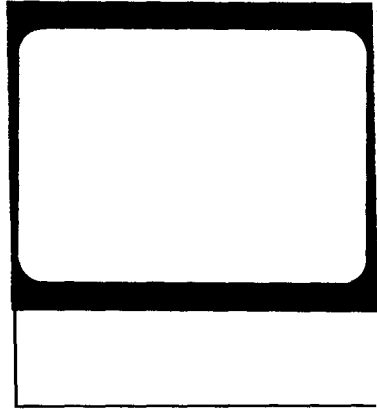
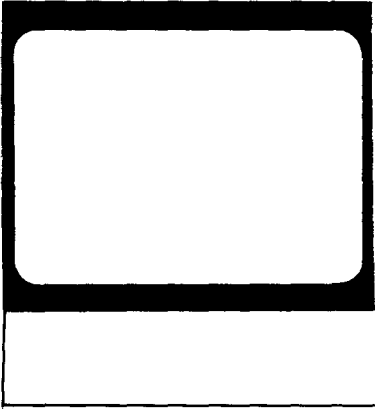
Same as above +
you were born in Y

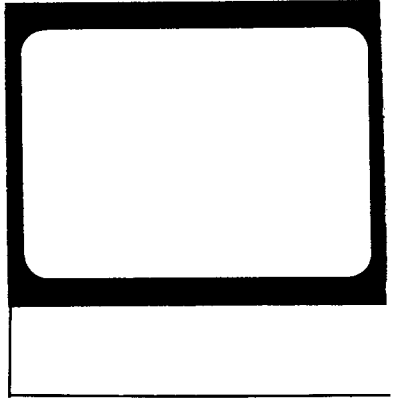
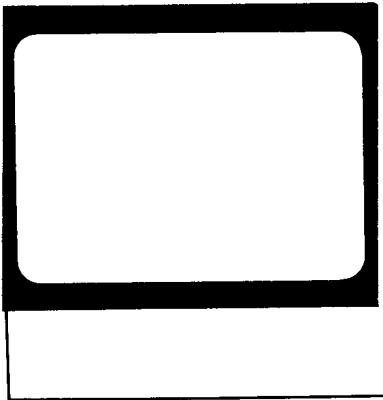
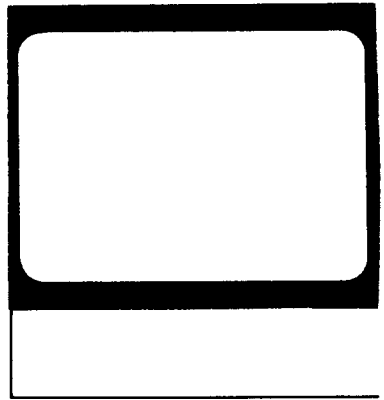
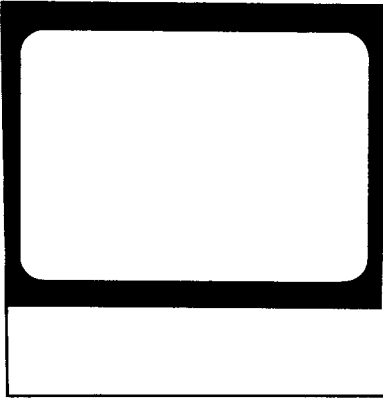
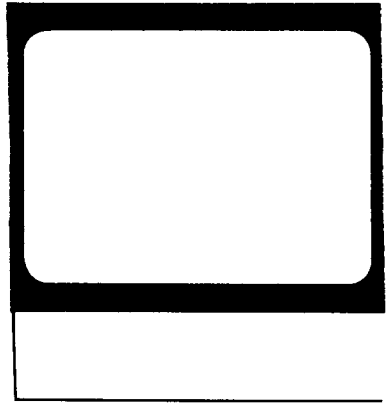
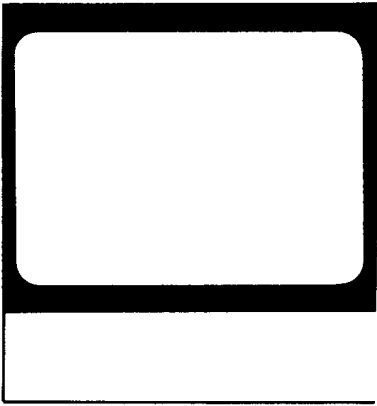
40 PR. "you were
born in"; Y



Same
you were born in

40 you were
born in; Y

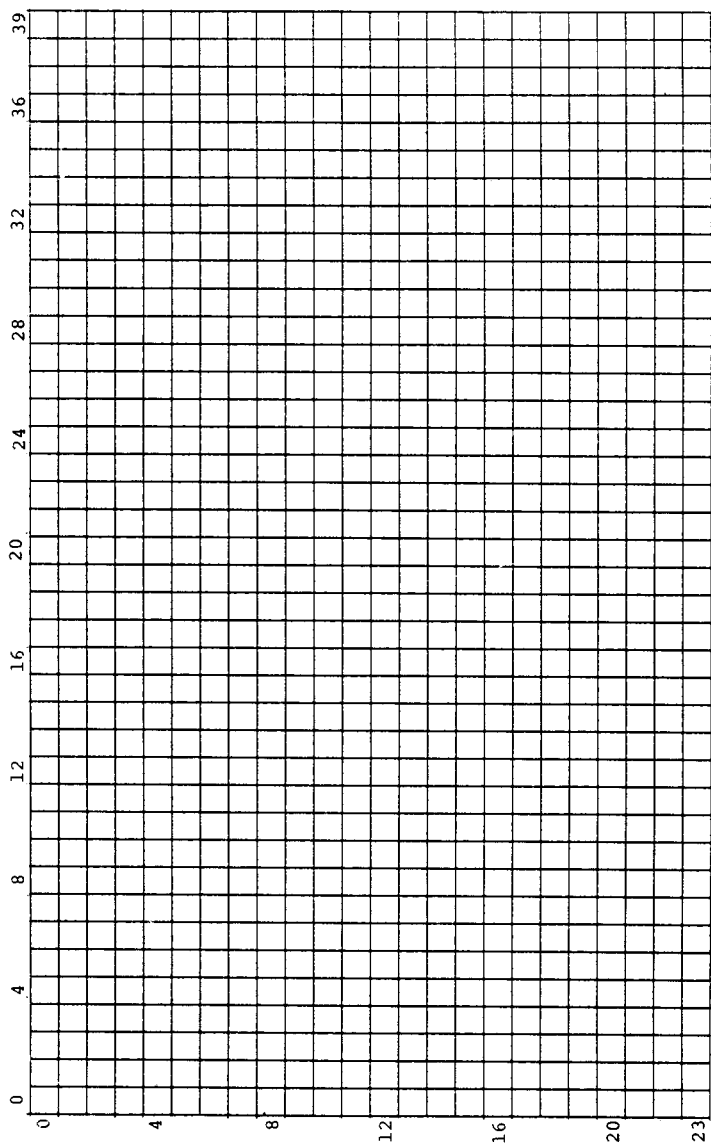


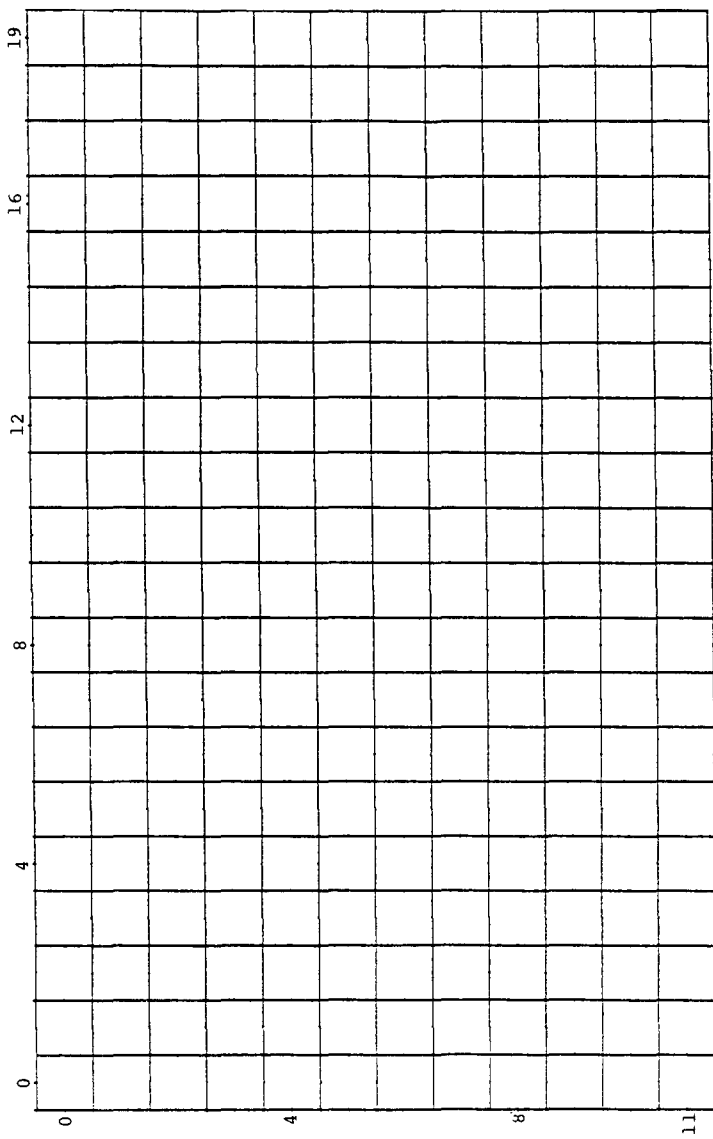


SKETCH PAD
FOR
CONTROL GRAPHICS
AND TEXT

READY

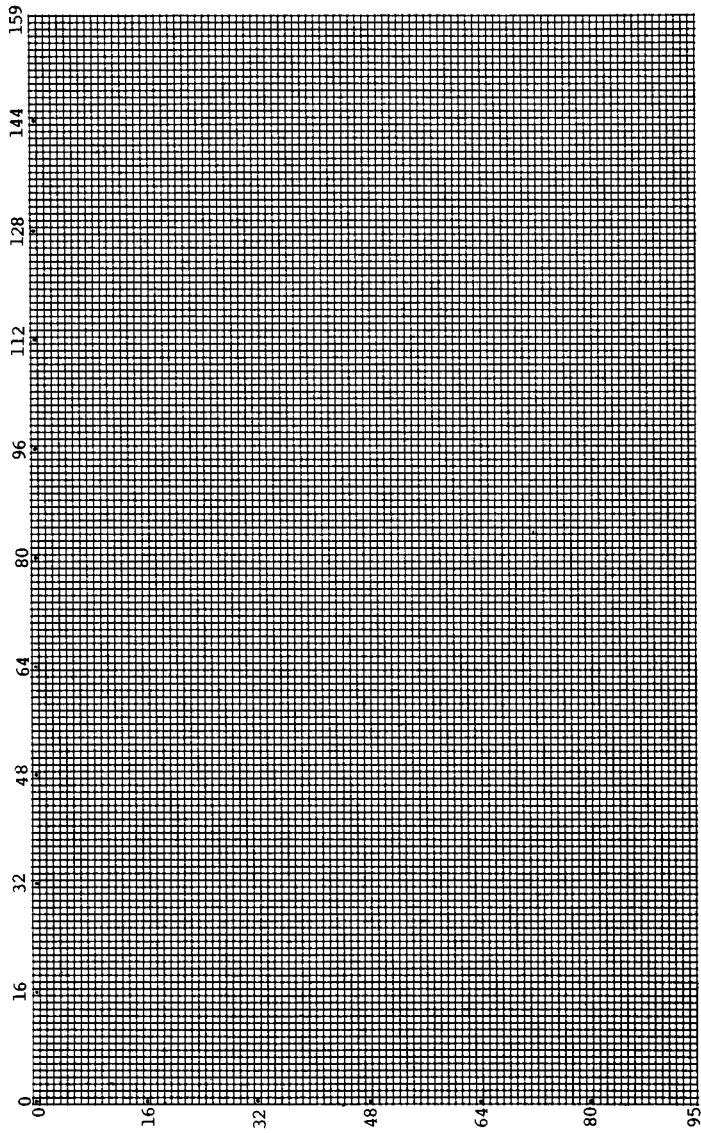






Graphics Mode 2





\$6.95



**ATARI®
PUZZLEMENTS**

Herbert Kohl

This book of puzzles will make you long for rainy Sunday afternoons rather than moan about them! Written for beginners (but not without plenty of challenges), **ATARI PUZZLEMENTS** will make you think—and, as an added bonus, it teaches you to think in BASIC. Most of the puzzles can also be worked out on paper, so you can while away the tedium of airplane trips or waiting for the dentist and still sharpen your computer skills.

Mastering the simple programs so elegantly presented for your ATARI Home Computer will provide hours of entertainment for the whole family. Even young readers will enjoy experimenting with combinations of the programs to come up with unique challenges of their own. And, how refreshing to find that your solutions don't have to match an answer key to be exactly, entirely correct.

A Creative Pastimes Book
RESTON PUBLISHING COMPANY, INC.

A Prentice-Hall Company
Reston, Virginia



illustrations by **C. MICHA**

ATARI is a registered trademark of Atari, Inc.

0-8359-0113-0