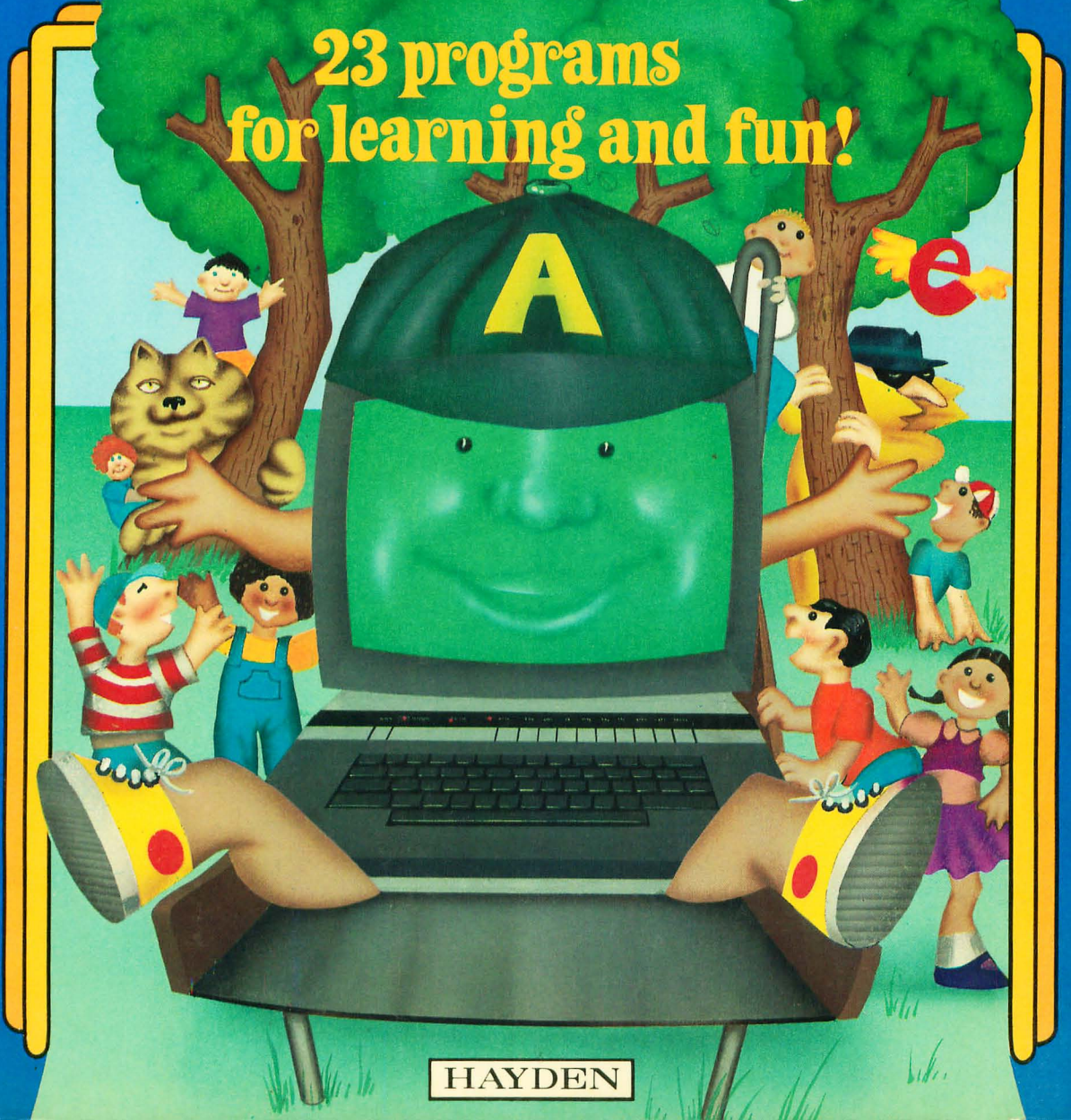


# The Atari<sup>®</sup> Playground

Fred D'Ignazio

23 programs  
for learning and fun!



HAYDEN





# **The Atari<sup>®</sup> Playground**

**Fred D'Ignazio**



HAYDEN BOOK COMPANY, INC.  
Rochelle Park, New Jersey

*For my team of young programmers, Howard Boggess, Angela Bradshaw, Joni Burdette, Brian François, Beth Ann Hostutler, Howard Levine, Mack McGhee, Melissa Perdue, and Scott Rainey, and their hardworking coach David James.*

*Atari® is a registered trademark of Atari, Inc., a division of Warner Communications, and is not affiliated with Hayden Book Company, Inc.*

Acquisitions Editor: GARY MARKMAN

Production Editor: MARSHALL E. OSTROW

Art Director: JIM BERNARD

Cover art by: SUSAN STURGILL/GEORGE BAQUERO

Illustrated by: NETWORK GRAPHICS/SUSAN STURGILL

Compositor: ART, COPY, & PRINT, INC.

Printed and bound by: ARCATA BOOK GROUP/FAIRFIELD GRAPHICS DIV.

Copyright © 1983 by HAYDEN BOOK COMPANY, INC. All rights reserved. No part of this book may be reprinted, or reproduced, or utilized in any form or by any electronic, mechanical, or other means, now known or hereafter invented, including photocopying and recording, or in any information storage and retrieval system, without permission in writing from the Publisher.

*Printed in the United States of America*

---

1	2	3	4	5	6	7	8	9	PRINTING
83	84	85	86	87	88	89	90	91	YEAR

---



# PREFACE

In this book you will find twenty-three learning games grouped by subject area. All the games are simple, fun, and educational. Some games teach word skills, others number skills. There are alphabet games, a Spelling Bee, music games, a Computer Crayon, Disappearing Ghosts, a Number Race, and even a News Bulletin.

Each game has its own chapter. The chapter begins with a **For Parents and Teachers** section that briefly describes the game and what things children might learn by playing it.

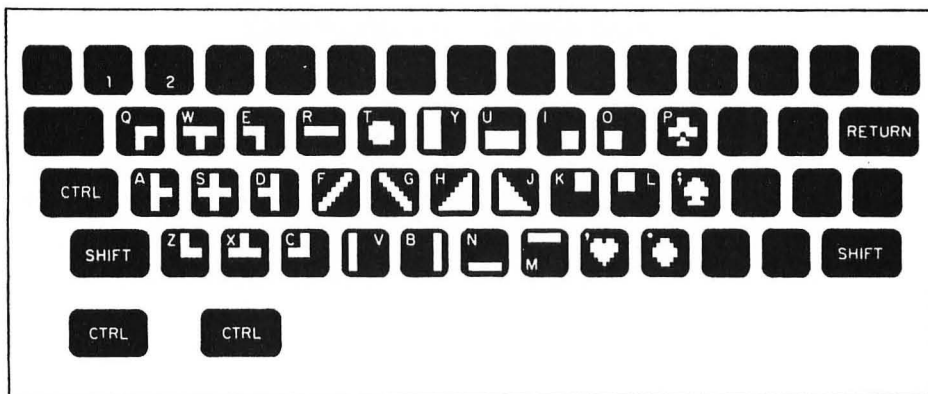
Next comes the **For Kids** section. This is often a story made up about the game. You can use the story to interest your children in the game and appeal to their imaginations. Most of the games are extremely simple, but with a little bit of fantasy they can be as exciting as any arcade game.


Next comes **The Program**. Following the program listing in many chapters is a **Typing Hints** section that shows you how to do tricks on the keyboard. The **Highlights** section then takes you on a guided tour of the important lines in the game. The **Variables** section explains the important variables—the memory cubbyholes in which the computer stores important letters and numbers. Finally, there is the **Do-It-Yourself** section that suggests things you might do to polish or improve the original game.

BASIC commands and statements in this book are written in capital letters: for example, GET, SAVE, END, FOR, and NEXT. Buttons on the keyboard are written in bold capital letters: for example, **RETURN**, **CTRL**, and **BREAK**.

Graphics characters will be used in many of the games. The use of these special characters is explained in the **Typing Hints** sections that follow the program listings. For your handy reference, the Control Graphics Keyboard is shown

here. When you use these special characters, be sure to hold down the **CTRL** key while pressing the selected graphics key.



If you are using the Atari 600XL, 800XL, 1200XL, 1400XL, or 1450 XLD rather than the Atari 400 or Atari 800, remember that the Atari-symbol key (⌘), which is used to get “reverse video” on the 400 and 800, is replaced by the  key. On the Atari 600XL and 800XL this key is located at the lower righthand corner of the keyboard. On the Atari 1200XL, 1400XL, and 1450XLD this key is located above the keyboard near the righthand side.

## Programs by Kids for Kids ...

Several programs in this book were written by young people. My editor at Hayden, Gary Markman, suggested that I enlist local high school students. I contacted David James, the computer science instructor at Patrick Henry High School. Patrick Henry High School is in Roanoke, Virginia, only two blocks from my home. David said he and his students would love to write some programs for my book.

I lent David my extra Atari computer, and he and his students began programming. Everybody worked hard. They



had only two weeks to write the programs. Students came to school early and left late. They got permission to skip some of their other classes. They took the computer home with them at night.

David spent the two weeks ferrying the computer around to the students' homes. One student lost an entire program when a Roanoke snowstorm caused a power failure.

But, somehow, the programs all got done. The students showed them off to a reporter from the *Roanoke Times and World News*, and an article about the students appeared in the paper two days before Christmas. To celebrate this occasion I took David and his students out for a pizza dinner, compliments of Hayden.

David writes: "Within the two short weeks that these programs were developed, a wide range of emotions were displayed—from excitement to frustration. The excitement at the chance to publish a program far outweighed (in the end) the frustration experienced during program development.

These students were willing to stay after school, work late at night, work over the weekend, and come early, before school, in order to meet the deadline. This type of motivation and dedication is the reward that I as a teacher receive."

## **Build your Own Guideposts ...**

As you are typing in the games in this book, make them your own: add comment lines (REM commands) that explain what each program does, **in your own words**. Also, at the beginning of each game, add PRINT commands to explain the game rules to your children.

I hope you have as much fun using this book as I did in writing it!

*Fred D'Ignazio*

### **EQUIPMENT NEEDED**

To use the programs in this book, you will need the following equipment:

- An Atari 400, 800, 600XL, 800XL, 1200XL, 1400XL, or 1450XLD, with 16K RAM (minimum).
- A black and white or, preferably, a color monitor (TV).
- An Atari 410 Program Recorder or an Atari 810 Disk Drive.



# CONTENTS

<b>FACES...</b>	<b>1</b>
1. Happy Face	2
2. Sad Face	7
 <b>ALPHABET...</b>	 <b>11</b>
3. Alphabet Al	12
4. The ABC Song	16
5. Find the Capital Letter!	20
6. Catch the Wild Letter!	25
 <b>WORDS...</b>	 <b>29</b>
7. The Skywriter	30
8. Spelling Bee	33
9. Musical Words	38
 <b>NUMBERS...</b>	 <b>43</b>
10. Goose Eggs	44
11. Scrambled Eggs	47
12. The Number Race	50
13. Guess My Number	54
14. The Rich King	58
15. The Disappearing Ghosts	67
 <b>COLORS...</b>	 <b>75</b>
16. Name that Color!	76
 <b>MUSIC...</b>	 <b>81</b>
17. Do-Re-Mi	82
18. Scrambled Notes	86

**DRAWING...****91**

19. The Computer Crayon 92

**KNOWLEDGE...****101**

20. What's Your Number? 102

**HAND-EYE...****107**

21. Up-Down-Left-Right 108

22. Up, Up and Away! 112

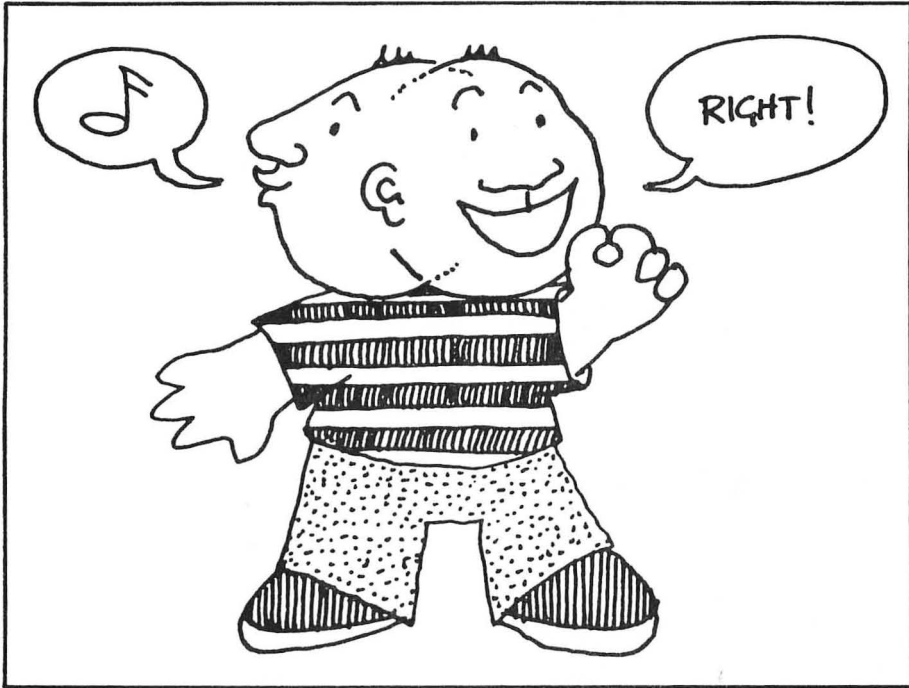
**IMAGINATION...****117**

23. News Bulletin 118



---

# FACES



# 1 **HAPPY FACE**



## **For Parents and Teachers ...**

This is a game subroutine or “helper program.” You can attach the subroutine onto most of the game programs in this book. When children get a correct answer, the subroutine prints a happy face, plays a happy tune, and congratulates the child.

## **For Kids ...**

When you do something right, you expect a smile and a word of praise, right?

Here’s a little program that draws a happy face on the TV screen, prints out the message “RIGHT!” and plays a happy “whistle.”

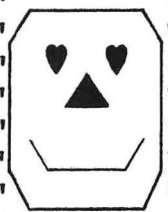
## The Program ...

Program Name: **HAPPY**

```

1000 REM ***HAPPY FACE & SOUND
1010 POSITION 14,12:PRINT "
1020 POSITION 14,13:PRINT "
1030 POSITION 14,14:PRINT "
1040 POSITION 14,15:PRINT "
1050 POSITION 14,16:PRINT "
1060 POSITION 14,17:PRINT "
1070 POSITION 14,18:PRINT "
1080 POSITION 14,19:PRINT "
1090 POSITION 15,21:PRINT "RIGHT!"
1110 FOR W=1 TO 10:FOR A=50 TO 25 STEP -
4:SOUND 0,A,10,10:FOR T=1 TO 2:NEXT T
1120 NEXT A:NEXT W:SOUND 0,0,0,0
1125 FOR I=12 TO 19:POSITION 14,I:PRINT
"      ":NEXT I:POSITION 15,21:PRINT "
"
1130 RETURN

```



## Typing Hints ...

To make the happy face, you need to use graphics characters. The **Control Graphics Keyboard** is pictured in the Preface of this book.

You can type a graphics character two ways:

- 1. Hold the control key (**CTRL**) down and push the button with the character you want to type. For example, hold down **CTRL** and press the **T** key and you have a **T**.
- 2. Hold the **CTRL** key down and push the **CAPS LOWR** key. Now you can type the special graphics characters without holding down the **CTRL** key. (To get out of this mode, hold the **SHIFT** key down and push the **CAPS LOWR** key again.)

Here are the graphics characters you will need to draw the happy face.

<b>In line Number ...</b>	<b>To get ...</b>	<b>Which is (are) ...</b>	<b>Press ...</b>
1010, 1060	—	top of face and mouth	CTRL N
1020	/ and \	left and right upper corners of face	CTRL F and CTRL G
1030, 1040 1050, 1060	and	left and right sides of face	CTRL V and CTRL B
1030	♥	eyes	CTRL ,
1040	▲ and ▼	left and right sides of nose	CTRL H and CTRL J
1060, 1070	\ and /	left and right corners of mouth and lower left and right corners of face	CTRL G and CTRL F
1080	—	bottom of face	CTRL M

## Highlights ...

This is one of the simplest programs in the book. On lines 1010 to 1080, it prints out the happy face. On line 1090, it prints the message "RIGHT!" On lines 1110 to 1125 it makes the whistle sound.

The program uses the sound command—SOUND—to make the whistle.

Look at all the FOR and NEXT commands. These commands make the computer do something over and over. This is called a computer **loop**. The loop begins with the FOR command. It ends with the NEXT command. Every command between the FOR and NEXT commands is part of the loop.



The FOR command determines how many times the computer circles around the loop. For example, look at FOR W = 1 TO 10 on line 1110. This tells the computer to set the loop counter W to 1. Each time the computer races around the loop it adds one to the loop counter (W). The computer does ten loops (from W = 1 to W = 10). Then it goes on and does something else.

Also, note that the program starts on line 1000 and ends on line 1130 with the command RETURN. That is because this program is really a subroutine—a helper program. It is not supposed to run on its own. It is supposed to help another program. (You can still test this subroutine by itself. When you finish typing it in, just type GOSUB 1010.)

You will want to attach this subroutine onto the tail end of almost all the other programs in the book. That way, whenever you get the right answer in any of the programs, the computer will draw a happy face, whistle, and shout “RIGHT!”

Be sure to save this subroutine on a disk by using the LIST command, or by using the LIST “C:” command if you are using a tape recorder.

When you have typed in one of the other programs, copy this subroutine back into memory using the ENTER command if the subroutine is on disk or ENTER “C:” if it is on tape.

Once it is entered, this subroutine will become part of the other program. Then you can save the combined, new program with a SAVE (on the disk) or CSAVE (on the tape) command.

## Variables ...

W	Whistle counter—10 whistles.
A	Pitch counter and pitch (notes played in the whistle).
T	Delay counter—slows down whistle.
I	Row counter—used in erasing face.

## **Do-It-Yourself ...**

Try **this** happy face for awhile. When you get tired of it, change it to a new face. All you have to do is type in new graphics characters. Change the eyes. Change the nose. Change the mouth. How about hair? Or a hat? And some ears?

## 2 **SAD FACE**



### **For Parents and Teachers ...**

This game subroutine is to be used along with the Happy Face subroutine. It can be attached to most of the games in this book. When children answer the computer's question incorrectly, the subroutine draws a sad face on the TV screen, makes a sad sound, and prints the message "SORRY ... TRY AGAIN."

### **For Kids ...**

When you miss a question or get the wrong answer, what do you expect? Probably a sad face. Maybe a sad sound. And, hopefully, encouragement to try again.

Here is a little program that does all that. It prints a sad face on the TV screen. Under the face it prints the message "SORRY ... TRY AGAIN." And it makes a sad sound.

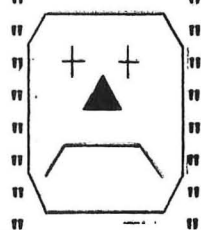
## The Program ...

Program Name: **SAD**

```

2000 REM ***SAD FACE & SOUND
2010 POSITION 14,12:PRINT "
2020 POSITION 14,13:PRINT "
2030 POSITION 14,14:PRINT "
2040 POSITION 14,15:PRINT "
2050 POSITION 14,16:PRINT "
2060 POSITION 14,17:PRINT "
2070 POSITION 14,18:PRINT "
2080 POSITION 14,19:PRINT "
2090 POSITION 9,21:PRINT "SORRY ... TRY
AGAIN"
2110 SOUND 0,100,10,8
2120 FOR PAUSE=1 TO 100:NEXT PAUSE
2130 SOUND 0,240,10,14
2140 FOR PAUSE=1 TO 200:NEXT PAUSE
2150 SOUND 0,0,0,0
2160 FOR PAUSE=1 TO 300:NEXT PAUSE
2165 FOR I=12 TO 19:POSITION 14,I:PRINT
"          ":NEXT I:POSITION 9,21:PRINT "
2170 RETURN

```



## Typing Hints ...

» To make the sad face, you need to use graphics characters. The **Control Graphics Keyboard** is pictured in the preface of this book.

<b>In line Number ...</b>	<b>To get ...</b>	<b>Which is (are) ...</b>	<b>Press ...</b>
2010	—	top of face	<b>CTRL N</b>
2020, 2060	/ and \	left and right upper corners of face and left and right corners of mouth	<b>CTRL F and CTRL G</b>
2030, 2040 2050, 2060	and	left and right sides of face	<b>CTRL V and CTRL B</b>
2030	+	eyes	<b>CTRL S</b>
2040	▲ and ▼	left and right sides of nose	<b>CTRL H and CTRL J</b>
2060, 2080	—	the mouth and bottom of face	<b>CTRL M</b>
2070	\ and /	left and right lower corners of face	<b>CTRL G and CTRL F</b>

## Highlights ...

The sad face is similar to the happy face. Lines 2010 to 2080 print out the sad face. Line 2090 prints the sad message. Lines 2110 to 2160 play the sad sound. Line 2165 erases the sad face and the message.

To test the sad face, just type GOSUB 2010.

Note that the sad face starts on line 200. The Sad Face is a subroutine, just like the Happy Face. You can attach it to the tail end of all your other programs.

First, save it on a disk with a LIST command. Or save it on a tape with a LIST "C:" command.

After you have typed another program, copy the Sad Face back into memory with an ENTER command (disk) or ENTER "C:" (tape).

## 10 THE ATARI PLAYGROUND

Using the ENTER or ENTER "C:" command you can add the happy and sad faces to all your other programs.

First you type in a new program.

Then you ENTER the Happy Face.

Then you ENTER the Sad Face.

Last, you save the combined, new program with a SAVE command (disk) or CSAVE (tape).

### **Variables ...**

**PAUSE** Delay Counter—Computer pauses between sounds.

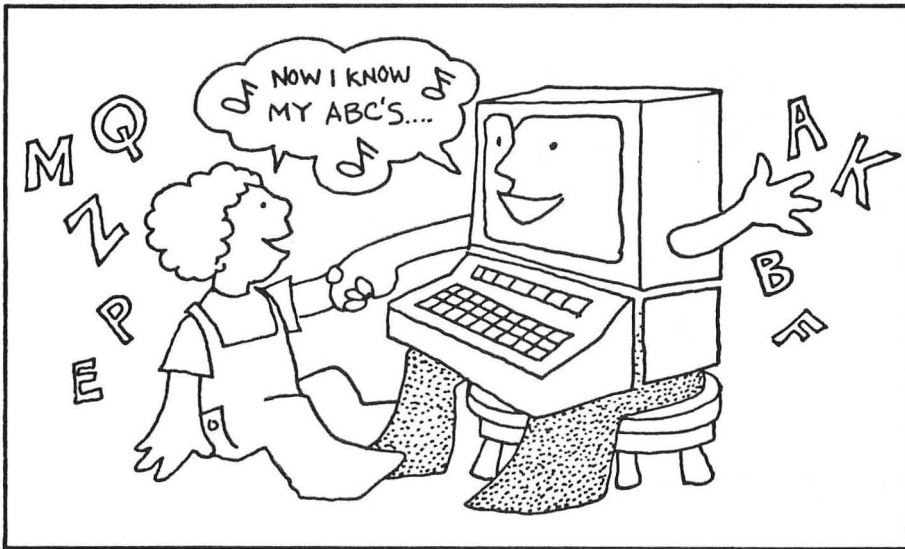
**I** Row counter—used in erasing face.

### **Do-It-Yourself ...**

Try this sad face for awhile. When you get tired of it, change it to a new face. All you have to do is type in new graphics characters. Change the eyes. Change the nose. Change the mouth. How about hair? Or a hat? And some ears?

---

# THE ALPHABET





3

# ALPHABET AL



## For Parents and Teachers ...

This game helps children learn the letters of the alphabet. The letters appear on the screen, and the children have to find them on the computer keyboard.

## For Kids ...

Remember Johnny Appleseed? Johnny went all over America planting seeds. The seeds sprouted and grew into apple trees with bright-red, juicy apples.

But what would have happened if Johnny had planted “letter” seeds instead of apple seeds? Then letters would have sprouted all over America in place of apples.

In the program below we have created a new American folk hero, “Alphabet Al.” Al is crazy about letters (the way Johnny was crazy about apples). He journeys across America

with a bag of letter seeds strung over his shoulder. Each time he reaches a new state, he races across it, planting seeds to make a new letter of the alphabet.

Type in and run the Alphabet Al program. You won't see Al himself, but you can follow his path by watching "alphabet" seeds sprout into letters. Listen to the noises the letters make when they blossom.

When Al is finished planting each letter, his trail of letters disappears. But before Al goes to a new state, he stops and asks you what letter he has just planted. He keeps asking the question until you get it right. Then, off he goes again, planting new letters.

## The Game ...

Program Name: **AL**

```
40 REM *** ALPHABET AL
45 DIM A$(1)
50 GRAPHICS 0
60 POKE 752,1
65 POSITION 9,8:PRINT "*** ALPHABET AL
  ***":FOR PAUSE=1 TO 1000:NEXT PAUSE:PRI
NT " \ "
70 FOR X=65 TO 90
80 COLOR X
100 PRINT " \ "
120 FOR I=1 TO 25
130 DRAWTO INT(RND(1)*39),INT(RND(1)*24)

140 SOUND 0,INT(RND(1)*220)+10,10,10
150 NEXT I
152 SOUND 0,0,0,0:FOR PAUSE=1 TO 200:NEX
T PAUSE:PRINT " \ "
153 POSITION 5,5:PRINT "WHAT LETTER DID
AL PLANT";:INPUT A$
154 IF A$=CHR$(X) THEN GOSUB 1010:GOTO 1
60
155 GOSUB 2010:GOTO 100
160 NEXT X
170 POKE 752,0
180 END
```

## Highlights ...

Al politely introduces himself (on lines 65-66) before madly dashing off to plant "letter" seeds.

On line 100, a PRINT "\ " command erases the screen each time Al enters a new state. On line 60, a POKE 752,1 erases the cursor (the little square of light that shows your current position). On line 170, a POKE 752,0 brings it back.

On line 130, a DRAWTO command traces Al's path of letters across the screen. On line 140, a SOUND command creates each letter's musical note as it appears.

The numbers 65 to 90 are the Atari ASCII codes for the upper-case letters. The program uses a FOR-NEXT command (FOR X = 65 TO 90 ... NEXT X) to create a loop that prints the 26 letters in the alphabet.

COLOR X selects each letter in the alphabet. For example, when X = 65 then COLOR 65 prints A's. COLOR 66 prints B's. COLOR 90 prints Z's.

On line 154 you see the command GOSUB 1010. This calls the happy face subroutine (Program 1: HAPPY) on lines 1000-1130. Before you run the Alphabet Al program, remember to copy in the happy face subroutine with the ENTER command (disk) or ENTER "C:" command (tape).

On line 155, you find the command GOSUB 2010. This calls the sad face subroutine (Program 2: SAD) on lines 2010-2170. Before you run the Alphabet Al program, remember to copy in the sad face subroutine with the ENTER command (disk) or ENTER "C:" command (tape).

When each subroutine has finished its job, it reaches a RETURN command (see lines 1130 and 2170) and bounces back to the command immediately following the GOSUB command.

## Variables ...

A\$	Your Guess—Which letter Al planted.
X	Letter of the alphabet (Atari ASCII code).
I	Counter of FOR-NEXT loop (Al's path of 25 letters for each state).

## Do-It-Yourself ...

Al runs through the alphabet from A to Z. Figure out a way to make him plant letters from Z to A (i.e., backwards).

Hint #1: Look at the FOR-NEXT loop on line 70. Hint #2: A FOR-NEXT loop can go backwards. In the FOR command, remember to include STEP -1. This makes the counter go backwards from the higher number to the lower number, one number at a time.

How would you make Al print letters at random?

Hint #1: You need to change lines 70 and 160.

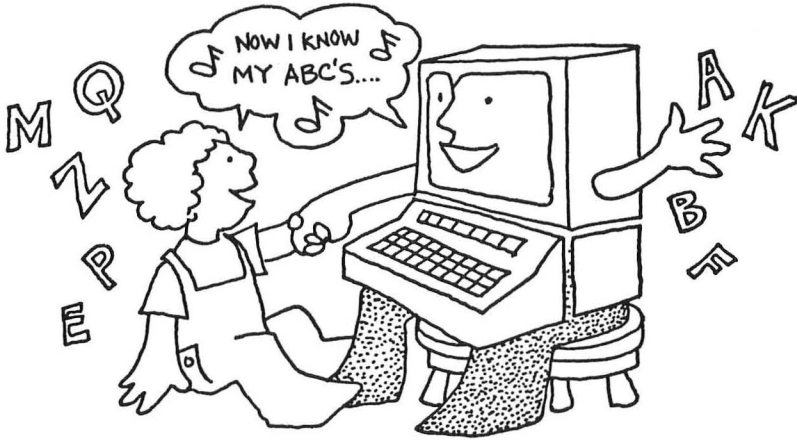
Hint #2: To compute a number representing a random letter of the alphabet, use the command `INT(RND(1)*26)+65`. This gives you a random number between 65 and 90 (the Atari ASCII codes representing the upper-case letters).

If Al's pace is too frantic and you'd like to slow him down a bit, you can add a delay loop at line 145 by typing between lines 140 and 150 this line:

```
145 FOR D=1 TO 200:NEXT D
```

You can also increase the amount of time the letters are on the screen before Al erases them and pops his question. Just increase the PAUSE loop on line 152 from 200 to 400 (or even higher).

# THE ABC SONG



## For Parents and Teachers ...

This game helps younger children learn the alphabet. It displays the letters of the alphabet while it plays the “Alphabet Song.”

If your child isn’t quick-tongued now, he or she will be after singing along with this program.

The program plays the ABC Song very slowly the first time. As it plays, it flashes the appropriate letter of the alphabet on the TV screen.

Your child will sing along and be proud that he or she can sing faster than the computer.

But, wait. The computer plays the ABC Song again. This time it plays faster.

Then it plays again—faster—and faster—and faster!

My young son has a ball trying to keep up with the computer. And he’s becoming an alphabet expert, too!

## The Game ...

Program Name: **ABCSONG**

```
50 REM *** THE ABC SONG
55 GRAPHICS 0
60 POKE 752,1
65 POSITION 8,8:PRINT "*** THE ABC SONG
   ***"
66 FOR PAUSE=1 TO 1000:NEXT PAUSE
70 EXTRA=80
80 DIM A$(5)
90 GRAPHICS 2+16:RESTORE
100 FOR LETTER=65 TO 90
110 POSITION 9,5:PRINT #6;CHR$(LETTER)
120 READ MUSIC:READ TIME
130 SOUND 0,MUSIC,10,10
140 FOR PLAY=1 TO TIME*EXTRA:NEXT PLAY
145 SOUND 0,0,0,0
150 NEXT LETTER
160 FOR LYRIC=1 TO 14
170 READ MUSIC:READ TIME:READ A$
175 POSITION 8,5:PRINT #6;A$
180 SOUND 0,MUSIC,10,10:FOR PLAY=1 TO TI
ME*EXTRA:NEXT PLAY
190 SOUND 0,0,0,0
195 POSITION 8,5:PRINT #6;"      "
200 NEXT LYRIC
205 IF EXTRA<=10 THEN 250
210 GRAPHICS 0:POKE 752,1:POSITION 16,8:
PRINT "FASTER";:INPUT A$
220 IF A$(1,1)<>"Y" THEN 240
225 IF EXTRA>20 THEN EXTRA=EXTRA-10:GOTO
   90
230 EXTRA=EXTRA-3
235 GOTO 90
240 IF A$(1,1)<>"N" THEN 210
250 PRINT " \ ":POKE 752,0:END
3000 REM *** ALPHABET NOTES
3010 DATA 121,5,121,5,81,5,81,5,72,5,72,
   5,81,10
3020 DATA 91,5,91,5,96,5,96,5,108,2.5,10
   8,2.5,108,2.5,108,2.5,121,10
```

```

3030 DATA 81,5,81,5,91,10,96,5,96,5,108,
10
3040 DATA 81,10,91,10,96,10,108,10
3060 DATA 121,5,NOW,121,5,I,81,5,KNOW,81
,5,MY,72,5,A,72,5,B,81,10,C'S
3070 DATA 91,5,NEXT,91,5,TIME,96,5,WON'T
,96,5,YOU,108,5,SING,108,5,WITH,121,10,M
E

```

## Highlights ...

On line 90, the program sets the TV screen to Graphics Mode 2 + 16. This lets the program display the largest characters. The "+16" eliminates the text window at the bottom of the screen.

On lines 100 to 150 is a FOR-NEXT loop which reads the musical notes in from the DATA statements on lines 3010 to 3040. It prints the appropriate letter of the alphabet, and plays the note.

The FOR-NEXT loop on line 140 makes the computer play each note according to its value—half note, quarter note, etc.

On lines 160 to 200 is a FOR-NEXT loop that reads the musical notes in from the DATA statements on lines 3060 to 3070. It prints the words "NOW I KNOW MY ABC'S, NEXT TIME WON'T YOU SING WITH ME"—and plays the notes.

When the computer finishes its song, on line 210 it asks you if you want to hear the song again, faster.

If you answer "YES" (or "Y"), the computer shortens the delay between the notes of the song by reducing the size of the variable EXTRA.

If you answer "NO" or if EXTRA is less than or equal to 10, then the computer ends the program. By the time EXTRA is reduced to 10, the ABC Song is sailing along, and only the most quick-tongued toddler (or adult) can keep up with it.

Take a look at lines 170, 220, and 240. You see that the variable A\$ can be used to store the words in the song—"Now I know ..."—or one-letter answers. A\$(1,1) tells the computer to



look at the characters starting at the first character and ending at the first character. This way you get the computer to look at only the first character in A\$.

## Variables ...

- EXTRA** Sets the speed of the ABC Song.
- A\$** Stores the words in the ABC Song ("Now I know ...").
- LETTER** Counter—Atari ASCII code for alphabet letters.
- MUSIC** Stores musical notes (READ from DATA statements).
- TIME** Stores duration of musical notes (READ from DATA statements).
- LYRIC** Counter—Lets computer play the 14 words at the end of the song ("Now I know ...").
- PLAY** Counter—Delay Loop to keep note playing a certain amount of time.

## Do-It-Yourself ...

The ABC Song runs through the alphabet from A to Z. Why not turn the song around and make it run through the letters backwards—from Z to A?

Also, if the song is too fast or too slow, just change the value of EXTRA. I've set it to 80 on line 70. This seemed to be the right starting speed for my child. But your children might like to sing faster.

5

# FIND THE CAPITAL LETTER!



## **For Parents and Teachers ...**

This game helps children learn the capital letters in the alphabet. It also teaches them the location of the letter keys on the computer keyboard.

## **For Kids ...**

It's a race!

The race is between you and the computer. The computer flashes a letter on the TV screen. You have only a few seconds to find that letter on the keyboard and punch it in.

If you find the letter in time, the computer smiles, whistles, and says, "RIGHT!"

If you don't, the computer moans, looks sad, and tells you to try again. Then you get another chance.

If you are fast and you beat the computer, you can ask the computer to print the letters faster.

## The Game ...

Program Name: **FINDBIG**

```

50 REM *** FIND THAT CAPITAL LETTER!
60 DIM L(26):FOR I=0 TO 25:READ A:L(I)=A
:NEXT I
62 GRAPHICS 0:POKE 752,1
65 POSITION 6,8:PRINT "*** FIND THAT LE
TTER! ***"
66 FOR PAUSE=1 TO 1000:NEXT PAUSE
70 TOP=750
80 DIM A$(1)
100 GRAPHICS 2+16
110 OPEN #1,0,4,"K:"
120 FOR LETTER=65 TO 90
125 COUNT=0
126 POSITION 9,4:PRINT #6;CHR$(LETTER)
128 COUNT=COUNT+1:IF COUNT=TOP THEN 160
130 CHILD=PEEK(764):IF CHILD=255 THEN 12
8
135 CLOSE #1
140 POKE 764,255
150 IF CHILD=L(LETTER-65) THEN GRAPHICS
0:POKE 752,1:GOSUB 1010:GRAPHICS 2+16:GO
TO 170
160 GRAPHICS 0:POKE 752,1:GOSUB 2010:GRA
PHICS 2+16:GOTO 125
170 NEXT LETTER
171 IF TOP<=50 THEN 180
172 GRAPHICS 0:POKE 752,1:POSITION 16,8:
PRINT "FASTER";:INPUT A$
173 IF A$(1,1)<>"Y" THEN 177
175 TOP=TOP-100:GOTO 100
177 IF A$(1,1)<>"N" THEN 172
180 GRAPHICS 0:POKE 752,0:END
3000 REM *** CAPITAL LETTERS
3010 DATA 63,21,18,58,42,56,61,57,13,1,5
,0,37,35,8,10,47,40,62,45,11,16,46,22,43
,23

```

## Highlights ...

The key to beating the computer is speed. Your child needs to find a letter quickly and push the right button.

The Alphabet AI program also made the child enter a letter. But speed wasn't important, so the child needed to enter the letter *and* push the **RETURN** button.

This time we give the child a break. The child doesn't need to push the **RETURN** button, just the letter button.

We accomplish this by opening the keyboard with an **OPEN** command on line 120. When the keyboard is **OPEN**, the computer immediately grabs input from the keyboard without waiting for a **RETURN**. Each time you press a button, you send a character to the computer.

When you press the letters in the alphabet, you send a code for each letter to the computer. But, when the keyboard is **OPEN**, you are not sending the standard Atari ASCII code. The number for each letter is different.

So that the computer can match your child's input with each letter in the alphabet, it needs the 26 codes for the 26 letters. We load these numbers into a numeric array called **L** on line 60. Now **L** has the 26 numbers stored in it. **L(0)** has the code for **A**. **L(1)** has the code for **B**. **L(25)** has the code for **Z**.

The computer prints the letters in the alphabet using a **FOR-NEXT** loop beginning on line 120. **LETTER**, the loop counter, varies from 65 to 90. These are the Atari ASCII codes for the 26 letters.

The computer prints the letter (line 126). Then, on line 128, it counts. If its count reaches 750 (the initial value of **TOP**), the computer jumps to line 160, and the sad face appears.

Lines 128 and 130 are a small loop. On line 130 the computer checks to see if memory position 764 has a new number in it. (The old number was 255.) The new number is the child's answer—stored in **CHILD**. Hopefully it is the code for a letter of the alphabet.

If 255 is still tucked away in memory position 764, the computer hops back to line 128 and continues counting.

If a new number is in 764, the computer (on line 150) checks to see if the child's answer (CHILD) matches the current letter of the alphabet. To do this, it has to cross-reference its two codes for the alphabet.

Here's how it works. If we take the ASCII letter counter, LETTER, and subtract 65 from it, we get the location in the array L where the number for the same letter is stored. For example, the OPEN code number for the letter A is 63. It is stored in L(0). The ASCII code for A is 65. The counter LETTER starts at 65. On line 126, the computer prints the letter A first, because LETTER starts at 65.

On line 150, the computer comes to the command:

**IF CHILD=L(LETTER-65)**

The remainder after subtracting 65 from LETTER (which is also 65) is 0. So the command translates to:

**IF CHILD=L(0)**

If the child has pushed the A button, then a 63 will be in CHILD. And a 63 is also in L(0), so a match is made. The child has entered the right letter!

After the child has raced the computer through all the 26 letters in the alphabet, the computer (on line 172) asks if the child would like to race again. Only this time the race will be faster.

If the child says yes, the computer jumps back to line 100 and starts the race all over. But first it subtracts 100 from the highest number to which it counts after it prints each letter. The first time through the alphabet the computer counts to 750 after printing each letter. The second time through, it counts only to 650.

Remember to ENTER in the happy face and the sad face subroutines. They will fit nicely between the main program and the DATA statements beginning on line 3000.

## Variables ...

<b>L</b>	Array (list) where the OPEN code numbers for the letters in the alphabet are stored.
<b>PAUSE</b>	Delay loop counter.
<b>TOP</b>	How high the computer counts after printing each letter.
<b>A\$</b>	Stores the child's yes (Y) or no (N) answer.
<b>LETTER</b>	Loop counter—Varies from 65 to 90, the Atari ASCII codes for the letters in the alphabet.
<b>COUNT</b>	Counter—for counting after printing each letter.
<b>CHILD</b>	The child's answer—stores the OPEN code number for the button the child pushed.

## Do-It-Yourself ...

This game teaches a lot of things: letter recognition, letter sequence, the location of the letters on the keyboard, and hand-eye coordination. But after awhile the child might get tired going through the alphabet from A to Z. Why not jazz the program up by having it go through the alphabet backwards or at random? (Look at the Do-It-Yourself section in earlier chapters for programming hints.)

6

# CATCH THE WILD LETTER!



## **For Parents and Teachers ...**

This game helps children match lower-case letters to upper case letters. It also helps teach them the location of the letter keys on the computer keyboard.

## **For Kids ...**

The computer has trapped a wild letter! The computer flashes a picture of the letter on the TV screen. Imagine that the letter is trapped in a large cage, deep in a forest of heart-shaped trees. The letter will stay inside the cage for a few seconds. Then it will escape. You need to shut the cage door. To shut it, you need to find the letter button on the keyboard and punch it quickly.



If you can't find the right button in time, don't worry. You will get another chance.

If you catch all the letters, the computer will play the game with you again. But next time the wild letters will be even wilder—they'll try to escape from their cage even faster!

## The Game ...

Program Name: **FINDLITL**

```

50 REM *** CATCH THE LOWER-CASE
55 REM *** LETTER!
60 DIM L(26):FOR I=0 TO 25:READ A:L(I)=A
  :NEXT I
62 GRAPHICS 0:POKE 752,1
65 POSITION 4,8:PRINT "*** CATCH THE WI
LD LETTER ***"
66 FOR PAUSE=1 TO 1000:NEXT PAUSE
70 TOP=750
80 DIM A$(1)
100 GRAPHICS 2+16:POKE 756,226
110 OPEN #1,0,4,"K:"
120 FOR LETTER=97 TO 122
122 COUNT=0
  » 124 POSITION 8,3:PRINT #6;" — ":POSITIO
N 8,4:PRINT #6;"| |":POSITION 8,5:PRINT
#6;" — "
126 POSITION 9,4:PRINT #6;CHR$(LETTER)
128 COUNT=COUNT+1:IF COUNT=TOP THEN 160
130 CHILD=PEEK(764):IF CHILD=255 THEN 12
8
135 CLOSE #1
140 POKE 764,255
150 IF CHILD=L(LETTER-97) THEN GRAPHICS
0:POKE 752,1:GOSUB 1010:GRAPHICS 2+16:PO
KE 756,226:GOTO 170
160 GRAPHICS 0:POKE 752,1:GOSUB 2010:GRA
PHICS 2+16:POKE 756,226:GOTO 122
170 NEXT LETTER
171 IF TOP<=50 THEN 180
172 GRAPHICS 0:POKE 752,1:POSITION 16,8:

```

```

PRINT "FASTER";:INPUT A$
173 IF A$(1,1)<>"Y" THEN 177
175 TOP=TOP-100:GOTO 100
177 IF A$(1,1)<>"N" THEN 172
180 GRAPHICS 0:POKE 752,0:END
3000 REM *** CAPITAL LETTERS
3010 DATA 63,21,18,58,42,56,61,57,13,1,5
,0,37,35,8,10,47,40,62,45,11,16,46,22,43
,23

```

## Typing Hints ...

» On line 124 you have three POSITION and PRINT #6; commands. Following the PRINT commands are double quotes (" "). Inside the double quotes are special graphics characters that make up the box that holds the lower-case letters.

To get the box's top ( \_\_\_\_\_ ), hold the **CTRL** key down and press the **N** key three times.

To get the box's sides ( | | ) hold the **CTRL** key down and press the **V** key, the space bar, and the **B** key.

To get the box's bottom ( \_\_\_\_\_ ), hold the **CTRL** key down and press the **M** key three times.

## Highlights ...

This program teaches a child the lower-case letters in the alphabet and how to type those letters on the (upper-case) keyboard.

The program is only slightly different from the FINDBIG program in the last chapter. LOAD FINDBIG into the computer, make a couple changes, and you have this new program.

First, look at line 100. You add the POKE 756,226 so the computer can print lower-case letters in Graphics Mode 2.

Next, look at line 120. You want to vary the loop counter, LETTER, from 97 to 122. The Atari ASCII codes for the lower-case letters are 97(a) to 122(z).

On line 150, you need to subtract 97 from LETTER (instead of 65).

On line 124, you need to draw the box around each letter. (See the Typing Hints section that follows the program listing.)

That's it!

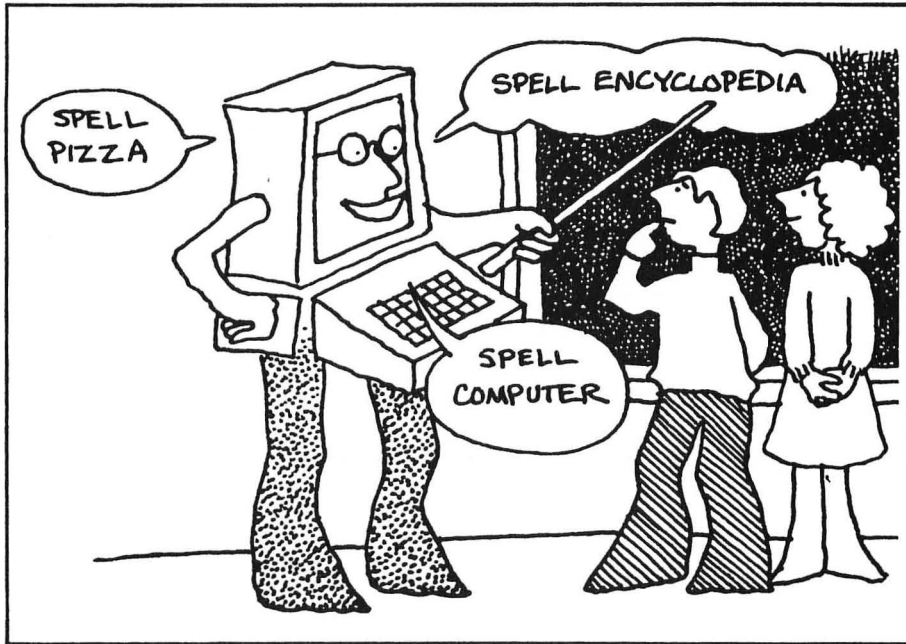
## **Do-It-Yourself ...**

The hearts on the screen appear automatically when you enter Graphics Mode 2 and POKE 756,226. The POKE statement lets you print the lower-case letters, but it also lets you print the Atari graphics characters. The heart is a graphics character. It is equivalent to a space in Graphics Mode 2.

The letters are now surrounded by a simple box. You can change the box to a jar, a chest, or whatever you wish. Also, you can make a pretty display—a web, Chinese boxes within boxes—surrounding the letters.

---

# WORDS



7

# THE SKYWRITER



## **For Parents and Teachers ...**

This is an imagination game. The computer asks children for any sequence of letters or numbers. Then it displays the letters or numbers, over and over, all across the TV screen. This is a good game for teaching children how to spell their own names or the names of their pets and other members of their family.

## **For Kids ...**

Have you ever seen a skywriter? A skywriter is an airplane that flies high up in the sky and leaves a white smokey trail. Sometimes the skywriter draws letters, words, and even pictures.

Wouldn't it be nice if you had your own skywriter and could draw your name in big, puffy cloud letters that everyone could see?

Buying a real skywriter would be expensive. But why not use your computer? The TV screen is blue so you can pretend it is the sky. The letters are white just like the skywriter's cloud letters. The computer can draw your name on the screen really fast, just like a skywriter streaking across the sky.

But why stick just to your name? Let's have the computer "skywriter" draw any message you can think up.

## The Game ...

Program Name: **SKYWRITER**

```
50 REM *** THE SKYWRITER
60 PRINT " \ ":POKE 752,1
65 POSITION 7,9:PRINT "*** THE SKYWRIT
ER ***"
70 FOR PAUSE=1 TO 1000:NEXT PAUSE
80 DIM N$(30)
100 PRINT " \ "
110 POSITION 3,8:PRINT "WHAT IS YOUR MES
SAGE";:INPUT N$
115 IF N$="" THEN 180
120 PRINT " \ "
130 FOR I=1 TO 150
140 PRINT N$;" ";
145 GOSUB 3010
150 NEXT I
160 FOR PAUSE=1 TO 1000:NEXT PAUSE
170 GOTO 100
180 POKE 752,0:PRINT " \ ":END
3000 REM *** SKYWRITER SOUND
3010 FOR K=1 TO 10
3020 SOUND 0,250,10,10
3030 NEXT K
3035 SOUND 0,0,0,0
3040 RETURN
```

## Highlights ...

This is a simple program. It asks you for your message, and then it prints it on the TV screen 150 times. Each time it prints the message, it calls a subroutine (on line 3010) to play a musical tone.

When the program has finished printing your message, it pauses for a moment. Then it erases the screen and asks you for a new message. If you don't have a new message, just press the **RETURN** button and the program will end.

## Variables ...

<b>PAUSE</b>	FOR-NEXT delay loop counter.
<b>N\$</b>	Stores your message.
<b>I</b>	FOR-NEXT loop counter—PRINT loop.
<b>K</b>	FOR-NEXT loop counter—SOUND loop in subroutine.

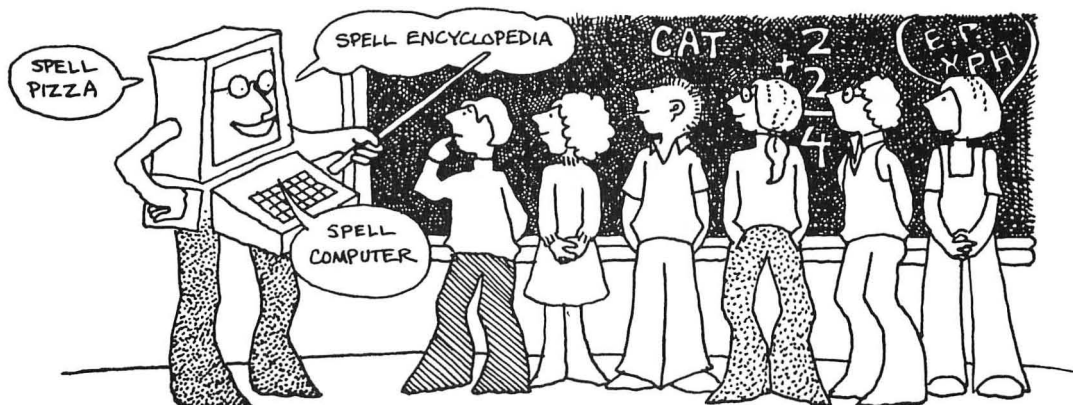
## Do-It-Yourself ...

The airplane is invisible, but you can create an airplane out of graphics characters (see the **Control Graphics Keyboard** in the preface). You can access the graphics characters by holding down the **CTRL** button and pressing the appropriate letter. You can “fly” the airplane across the sky and print the child's message.

You can add other things to the sky: birds, clouds, and other planes.

The plane noise is pretty simple. You can experiment with the **SOUND** command to make the plane noise sound more like a real plane.

# SPELLING BEE



## For Parents and Teachers ...

This game helps children practice spelling. You or your children enter the words into the computer, and then the computer quizzes the children on each word. The children can take the computer quiz over and over until they have mastered all the words.

## For Kids ...

The computer first asks you for 10 words you would like to know how to spell. Maybe they are the 10 words that will be appearing on your spelling test tomorrow morning.

You type in the 10 words, one at a time.

The computer lines you up against the wall and starts quizzing you. It shows you word number one—"PIZZA." Then



the word disappears. The computer asks, "WORD?"

"PEETSAH," you type.

The computer frowns, moans, and says, "SORRY ... TRY AGAIN."

You try again: "PIZZA."

"RIGHT!" says the computer. It smiles and whistles. Then it flashes word number two.

## The Game ...

Program Name: **SPELLBEE**

```

50 REM *** SPELLING BEE
60 DIM A$(25),B$(25)
62 GRAPHICS 0:POKE 752,1
65 POSITION 8,8:PRINT "***  SPELLING BEE
   ***"
66 FOR PAUSE=1 TO 1000:NEXT PAUSE
70 DIM W1$(25),W2$(25),W3$(25),W4$(25),W
5$(25)
75 DIM W6$(25),W7$(25),W8$(25),W9$(25),W
10$(25)
100 FOR SPELL=1 TO 10
110 PRINT " \ ":POSITION 13,8:PRINT "WORD
   #";SPELL;:INPUT A$
120 ON SPELL GOSUB 3010,3020,3030,3040,3
050,3060,3070,3080,3090,3100
130 NEXT SPELL
140 FOR SPELL=1 TO 10
150 ON SPELL GOSUB 4010,4020,4030,4040,4
050,4060,4070,4080,4090,4100
160 PRINT " \ ":POSITION 13,8:PRINT "WORD
   : ";A$
170 FOR PAUSE=1 TO 800:NEXT PAUSE
180 PRINT " \ ":POSITION 13,8:PRINT "WORD
   ";:INPUT B$
190 IF A$=B$ THEN GOSUB 1010:GOTO 210
200 GOSUB 2010:GOTO 160
210 NEXT SPELL
220 PRINT " \ ":POSITION 8,8:PRINT "SAME
WORDS AGAIN";:INPUT A$

```

```
230 IF A$(1,1)="Y" THEN 140
240 IF A$(1,1)<>"N" THEN 220
250 POKE 752,0:PRINT " \ ":END
3000 REM *** STORE SPELLING WORDS
3010 W1$=A$:RETURN
3020 W2$=A$:RETURN
3030 W3$=A$:RETURN
3040 W4$=A$:RETURN
3050 W5$=A$:RETURN
3060 W6$=A$:RETURN
3070 W7$=A$:RETURN
3080 W8$=A$:RETURN
3090 W9$=A$:RETURN
3100 W10$=A$:RETURN
4010 A$=W1$:RETURN
4020 A$=W2$:RETURN
4030 A$=W3$:RETURN
4040 A$=W4$:RETURN
4050 A$=W5$:RETURN
4060 A$=W6$:RETURN
4070 A$=W7$:RETURN
4080 A$=W8$:RETURN
4090 A$=W9$:RETURN
4100 A$=W10$:RETURN
```

## Highlights ...

This program does everything in two small loops.

First, on lines 100 to 130, the computer lets you enter each word you want included in the Spelling Bee. It stores your answer (A\$) in the "W" variables, from W1\$ to W10\$. It expects 10 spelling words.

The computer knows which W variable to store the word in using the FOR-NEXT loop counter, SPELL. Look at line 120. There are 10 line numbers to jump (GOSUB) to. SPELL acts like an index to show the computer where to jump. For example, if SPELL equals 1, then the computer jumps to the first line number (3010). If SPELL equals 2, the computer jumps to the second line number (3020). Finally, when SPELL equals 10, the computer jumps to the tenth line number (3100).

Each of the lines (3010-3100) is a one-line subroutine. Each subroutine copies the spelling word from A\$ to the appropriate W variable. Then the computer reaches RETURN and bounces back to get more spelling words.

The computer uses the second loop (lines 140 to 210) to quiz you on the spelling words you just entered.

This time it uses the loop counter, SPELL, to transfer the right spelling word from a W variable back to A\$.

On line 160, the computer flashes the spelling word on the screen. Then, on line 180, it erases the word and asks you to spell it.

If you are wrong, the computer jumps to the sad face subroutine. Then it gives you a chance to spell the word again.

After you have spelled all 10 words correctly, the computer asks you (on line 220) if you want to practice the words again. If you do, it jumps back to line 140 and begins the spelling bee a second time.

The computer will keep quizzing you until you drop. Or until you've mastered all the words.

Remember to ENTER the happy face and sad face subroutines. They fit between the main program (lines 50 to 250) and the W subroutines beginning on line 3000.

## Variables ...

A\$	Stores correct spelling of spelling-bee words.
B\$	Stores your answer.
W1\$- W10\$	Store the 10 spelling-bee words.
SPELL	Loop counter—index to subroutine for storing and retrieving spelling-bee words.
PAUSE	Delay loop counter.

## Do-It-Yourself ...

There are lots of things you can do to this program. You can expand it so that it quizzes you on more than 10 words at a time. To expand it, you need to change the number 10 on lines 100 and 140 to a larger number. You need to add new W variables from line 3110 on, and from line 4110 on. The new variables would be W11\$, W12\$, and so on. There's lots of room for adding new variables.

Another thing you can do would be to grade your child's score. You need to have a counter, COUNT, add one each time your child gets a word wrong. After the test, you form a percentage by computing  $((10-\text{COUNT})/10)*100$ . Then you print out the percentage on the TV screen for your child to see.

# MUSICAL WORDS



## For Parents and Teachers ...

This game helps children learn important words and numbers, such as their name, phone number, address, etc. The child types in a word (or number) and the computer sets it to music and plays it like a song. The computer accepts only the correct letters or numbers. All other keys on the computer keyboard are “asleep.”

## For Kids ...

Pretend that you have a magic horn. When you blow it, out pops a pretty sound. But something else comes out, too: a letter.

You try playing the horn again. A new sound comes out. And out flies a different letter.

You play lots of different sounds on your horn. Letters come zooming out of the horn. They bump together and form a word. The word looks familiar. It's your name. And it's musical!

You can make your name musical. But that's not all. Try setting your mom's name to music. Or your dad's name. Or your little sister's name. Or the name of your favorite pet gerbil. Next, try out your phone number, your street name, your city, and your state. Like your name, they can all be musical.

## The Game ...

Program Name: **WRDMUSIC**

```

50 REM *** WORD MUSIC
60 PRINT "\ ":POKE 752,1
65 POSITION 8,8:PRINT "***      WORD MUSIC
   ***"
66 FOR PAUSE=1 TO 1000:NEXT PAUSE
70 DIM WD$(15)
80 DIM MUSIC(15)
82 FOR I=1 TO 15:READ N:MUSIC(I)=N:NEXT
  I
100 GRAPHICS 0:POKE 752,1
110 POSITION 10,8:PRINT "YOUR WORD";:INP
   UT WD$
115 IF WD$="" THEN POKE 752,0:PRINT "\ ":
   END
130 GRAPHICS 2
135 POKE 752,1
140 PRINT "          ***  TYPE YOUR WORD  ***
   "
145 OPEN #1,4,0,"K:"
147 POSITION 4,3
150 FOR I=1 TO LEN(WD$)
160 GET #1,L
165 IF L=155 THEN CLOSE #1:GOTO 100
170 IF CHR$(L)<>WD$(I,I) THEN 160
180 PRINT #6;CHR$(L);
181 SOUND 0,MUSIC(I),10,15
182 FOR PAUSE=1 TO 100:NEXT PAUSE
183 SOUND 0,0,0,0

```

```
190 NEXT I
200 FOR PAUSE=1 TO 300:NEXT PAUSE
210 GOSUB 3510
215 CLOSE #1
220 GOTO 130
3500 REM *** PLAY WORD/NOTES
3510 GRAPHICS 2:POKE 752,1
3515 PRINT "    ***  A LITTLE WORD MUSIC
***"
3517 FOR J=1 TO 5
3518 POSITION 4,3:PRINT #6;"
"
3519 POSITION 4,3
3520 FOR I=1 TO LEN(WD$)
3530 PRINT #6;WD$(I,I);
3540 SOUND 0,MUSIC(I),10,15
3550 FOR PAUSE=1 TO 100:NEXT PAUSE
3560 NEXT I
3570 NEXT J
3575 SOUND 0,0,0,0
3580 RETURN
4000 DATA 243,193,162,121,96,81,60,47,40
,182,144,121,91,72,60
```

## Highlights ...

The program begins by reading musical notes into the array MUSIC on line 82.

Next the program asks you for a word. If you are finished with the game, you just press the **RETURN** button.

The computer goes into Graphics Mode 2. It asks you to type in your musical word. You type in the same word as before, only this time each letter is set to music. The computer accepts only the letters in the word.

When you have entered the entire word, the computer prints the word five times, playing it like a song. Then it erases the word and asks you to enter it again. If you are tired of this

word, the computer asks you if you want a new word. If you press the **RETURN** button, the game is over.

## Variables ...

- PAUSE** Delay loop counter.
- WD\$** Stores word or number that you want set to music.
- MUSIC** Saves 15 musical tones to match with letters or numbers.
- I** First (line 82)—it is a counter for a **READ** loop; later (line 150)—it is a counter for the loop in which the child types in the letters or numbers; later (line 3520)—it is a counter for the loop that prints the letters or numbers and plays the musical notes.
- L** Atari ASCII byte—key typed by child.
- J** Counter for loop that plays word five times.

## Do-It-Yourself ...

The program accepts a maximum of 15 characters (letters or numbers) at a time. You can increase the size of **WD\$** and **MUSIC** so that the computer can accept longer words and numbers. Then you need to change the **POSITION** command on line 147 and the **PRINT** command on line 180 so that, if the word or number can't fit on the current line, the computer continues it neatly on the next line (or skips a line).

Also, the program always uses the same 15 musical notes, no matter what letters or numbers the child enters. You can replace the **DATA** command on line 4000 with a longer list of musical notes. To find the note values, see the accompanying list of **Pitch Values for Musical Notes**.



**PITCH VALUES FOR MUSICAL NOTES****HIGH NOTES**

C	29
B	31
A <sup>#</sup> , B <sup>b</sup>	33
A	35
G <sup>#</sup> , A <sup>b</sup>	37
G	40
F <sup>#</sup> , G <sup>b</sup>	42
F	45
E	47
D <sup>#</sup> , E	50
D	53
C <sup>#</sup> , D <sup>b</sup>	57
C	60
B	64
A <sup>#</sup> , B	68
A	72
G <sup>#</sup> , A <sup>b</sup>	76
G	81
F <sup>#</sup> , G <sup>b</sup>	85
F	91
E	96
D <sup>#</sup> , E <sup>b</sup>	102
D	108
C <sup>#</sup> , D <sup>b</sup>	114

**MIDDLE C**

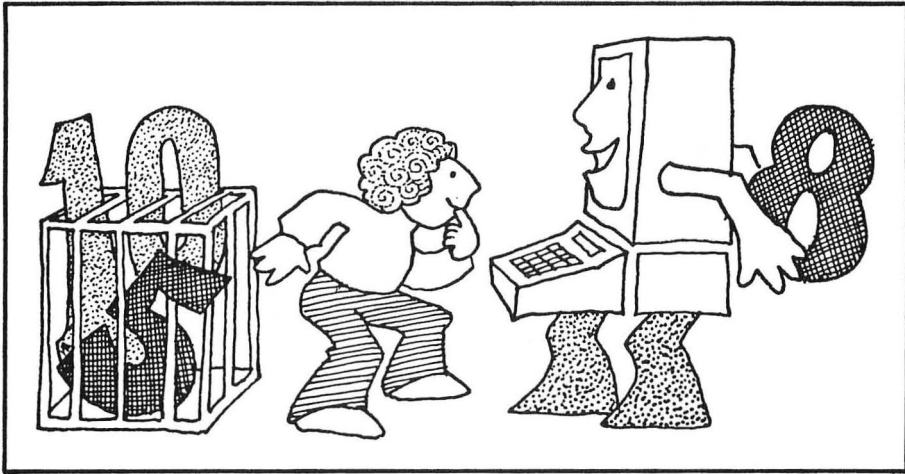
C	121
B	128
A <sup>#</sup> , B <sup>b</sup>	136
A	144
G <sup>#</sup> , A <sup>b</sup>	153
G	162
F <sup>#</sup> , G <sup>b</sup>	173
F	182

**LOW NOTES**

D	193
D <sup>#</sup> , E <sup>b</sup>	204
D	217
C <sup>#</sup> , D <sup>b</sup>	230
C	243

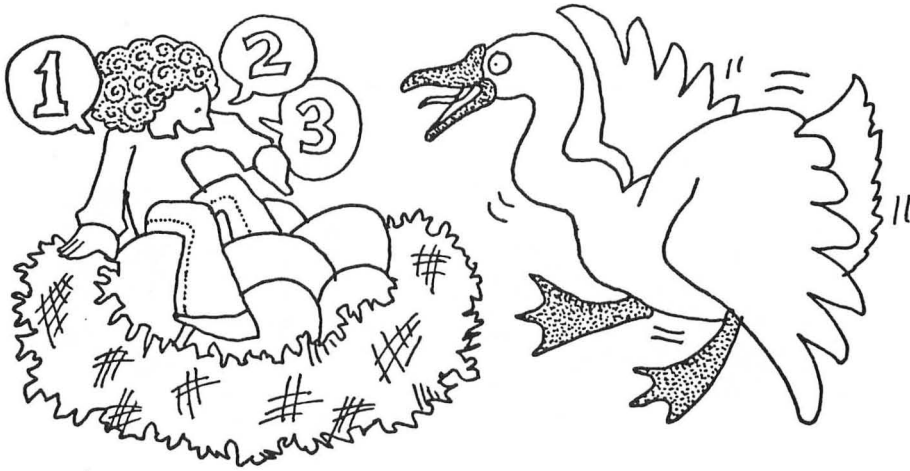
---

# NUMBERS



# 10

## GOOSE EGGS



### **For Parents and Teachers ...**

This game helps younger children learn how to count. It also teaches them where to find the number keys on the computer keyboard.

### **For Kids ...**

Pretend you live on a farm. You go out to the goose coop. You find 1 goose egg in the goose nest.

The mama goose gets on the nest and lays a new egg.

That's 2 goose eggs.

Then the mama goose gets on the nest and lays another egg.

That's 3 goose eggs.

The mama goose keeps laying new eggs. You keep counting.

How many eggs can she lay?

How many eggs can you count?

## The Game ...

Program Name: **GOOSE**

```
50 REM *** COUNTING GOOSE EGGS
62 GRAPHICS 0:POKE 752,1
65 POSITION 5,8:PRINT "*** COUNT THE GO
OSE EGGS ***"
67 FOR PAUSE=1 TO 1000:NEXT PAUSE
90 PRINT " \ "
100 FOR Y=0 TO 12 STEP 2:FOR X=0 TO 38 S
TEP 2
105 COUNT=COUNT+1
110 POSITION X,Y:PRINT "O"
120 POSITION 5,15:PRINT "HOW MANY GOOSE
EGGS";:INPUT EGG
125 POSITION 5,15:PRINT "
"
130 IF EGG<>COUNT THEN GOTO 140
135 GOSUB 1010:REM *** RIGHT ANSWER!
138 GOTO 150
140 GOSUB 2010:REM *** WRONG ANSWER
146 GOTO 120
150 NEXT X:NEXT Y
160 POKE 752,0
170 END
```

## Highlights:

This is a short and simple program.

On line 100 you set up a FOR-NEXT loop to position the eggs (letter O's) on the TV screen. X stands for the column. Y stands for the row. You place the eggs two spaces apart to make them easy to count. You put the X loop **inside** the Y loop to make the new eggs appear column by column (from left to right) across the screen.

On line 120 the computer asks how many eggs the goose has laid. After you have answered, on line 125 the computer erases the question and answer.

If you guess correctly, the computer jumps to the happy

face subroutine on line 1010 (remember to ENTER it from tape or disk).

If you guess wrong, the computer jumps to the sad face subroutine on line 2010 (remember to ENTER it from tape or disk).

## Variables ...

X	Loop Counter—column on TV screen for drawing the goose egg.
Y	Loop Counter—row on TV screen for drawing the goose egg.
COUNT	Counter—how many goose eggs are on the screen.
EGG	Your guess.

## Do-It-Yourself ...

This is a very simple program. You can add lots of bells and whistles. For example, you can make the goose eggs out of the special graphics characters. (See the **Control Graphics Keyboard** shown in the preface.)

Or you can have the computer make a noise each time the goose lays a new egg.

Or you can even draw a picture of the goose out of graphics characters.

11

# SCRAMBLED EGGS



## For Parents and Teachers ...

In the last chapter your children counted goose eggs along with the computer. But did you notice that the computer always counted the eggs **in order**?

In this chapter the computer scrambles the goose eggs. It has the option of drawing anywhere from **zero** goose eggs to **24** goose eggs on the TV screen. The children have to count the eggs and type in the correct answer. Then the computer erases the old eggs and draws a bunch of new ones.

## The Game ...

Program Name: **GOOSE2**

```
50 REM *** COUNTING GOOSE EGGS  
62 GRAPHICS 0:POKE 752,1
```

```
65 POSITION 5,8:PRINT "*** COUNT THE GO  
OSE EGGS ***"  
67 FOR PAUSE=1 TO 1000:NEXT PAUSE  
90 PRINT " \ "  
100 COUNT=0  
101 TOTAL=INT(RND(1)*25)  
103 IF TOTAL=0 THEN 220  
105 FOR Y=6 TO 10 STEP 2:FOR X=12 TO 26  
STEP 2  
110 POSITION X,Y:PRINT "O"  
115 GOSUB 3010  
120 COUNT=COUNT+1  
130 IF COUNT>=TOTAL THEN POP :POP :GOTO  
220  
150 NEXT X:NEXT Y  
220 POSITION 8,15:PRINT "HOW MANY GOOSE  
EGGS";:INPUT EGG  
225 POSITION 8,15:PRINT "  
"  
230 IF EGG<>COUNT THEN GOTO 240  
232 POSITION 15,4:PRINT EGG;" EGGS"  
235 GOSUB 1010:REM *** RIGHT ANSWER!  
237 FOR PAUSE=1 TO 500:NEXT PAUSE  
238 GOTO 90  
240 GOSUB 2010:REM *** WRONG ANSWER  
246 GOTO 220  
3000 REM *** EGG SOUND  
3010 FOR K=1 TO 10  
3020 SOUND 0,140,10,15  
3030 NEXT K  
3040 SOUND 0,0,0,0  
3050 RETURN
```

## Highlights ...

This program is very similar to the GOOSE program. For details, take a look at the last chapter.

In the old program, GOOSE, we got the computer to count the eggs in order (1, 2, 3, and so on) by using a FOR-NEXT loop. In the new program, GOOSE2, we get the computer to choose

the number of eggs at random by using the RND function on line 101.

If the computer has chosen to print zero eggs (TOTAL equals 0), then, on line 103 we have the computer jump around the "print-eggs" loop and immediately ask how many eggs it has printed. (Since it has printed no eggs, the answer is 0).

The FOR-NEXT loop on lines 105 to 150 prints the eggs in the center of the TV screen. Each time the computer "lays" an egg, it calls a SOUND subroutine on line 3010. The computer keeps count of how many eggs it has printed by using the variable COUNT. The computer stops printing eggs when COUNT reaches TOTAL (line 130).

In this new program, if you count the number of eggs correctly, the computer prints that number above the rows of eggs on the TV screen. Then (still displaying the eggs and the number) the computer prints the happy face. The happy face disappears, but the rows of eggs and the number linger for a final moment. Then the computer erases the screen and draws a new bunch of eggs for you to count.



12

# THE NUMBER RACE



## **For Parents and Teachers ...**

This program helps children learn numerals and number sequence. It also helps them learn the location of the numerals on the computer keyboard.

## **For Kids ...**

How fast can you find a number? Can you beat the computer?

In this game, the computer flashes a number on the picture screen. The number is between 0 and 9. As soon as the number appears, the computer begins counting. If you can punch the number button on the keyboard before the computer is done, you win. If not, the computer wins.

You win the first round if you can beat the computer for all ten numbers (0 to 9). But the computer's not beaten yet. It will

play you four more rounds, each faster than the last. If you beat the computer all five rounds, it quits and you are the winner!

## The Game ...

Program Name: **FINDNUM**

```
50 REM *** FIND THAT NUMBER!
60 PRINT " \ ":POKE 752,1
62 POSITION 5,8:PRINT "***   FIND THAT N
   UMBER!   ***"
64 FOR PAUSE=1 TO 1000:NEXT PAUSE
65 DIM N(10):N(0)=50:N(1)=31:N(2)=30:N(3
   )=26:N(4)=24:N(5)=29:N(6)=27:N(7)=51:N(8
   )=53:N(9)=48
70 TOP=300
80 DIM A$(1)
100 GRAPHICS 2+16
110 OPEN #1,0,4,"K:"
120 FOR NUMBER=48 TO 57
125 COUNT=0
126 POSITION 9,4:PRINT #6;CHR$(NUMBER)
128 COUNT=COUNT+1:IF COUNT=TOP THEN 160
130 A=PEEK(764):IF A=255 THEN 128
135 CLOSE #1
140 POKE 764,255
150 IF A=N(NUMBER-48) THEN GRAPHICS 0:PO
   KE 752,1:GOSUB 1010:GRAPHICS 2+16:GOTO 1
   70
160 GRAPHICS 0:POKE 752,1:GOSUB 2010:GRA
   PHICS 2+16:GOTO 125
170 NEXT NUMBER
171 IF TOP<=50 THEN 180
172 GRAPHICS 0:POKE 752,1:POSITION 16,8:
   PRINT "FASTER";:INPUT A$
173 IF A$(1,1)<>"Y" THEN 177
175 TOP=TOP-50:GOTO 100
177 IF A$(1,1)<>"N" THEN 172
180 GRAPHICS 0:POKE 752,0:END
```

## Highlights ...

This program is a cousin of the Find the Capital Letter and Catch the Wild Letter programs. For a detailed description of the program, look at chapters five and six.

The major difference between this program and the others is that the FOR-NEXT loop counter, NUMBER, varies from 48 to 57. These are the ATASCII codes (in decimal) for the numbers 0 to 9.

The keyboard is opened with an OPEN command on line 110. Then, on line 130, when a person presses one of the number buttons, the number is stored in an internal code in variable A. The codes representing the numbers 0 through 9 are stored in an array named N: in N(0), N(1), N(2), N(3), N(4), N(5), N(6), N(7), N(8), and N(9).

Line 50 uses N to test for a match between the number the computer flashed on the screen and the person's answer. If the person typed a 0, a 50 (the internal code for 0) would be stored in variable A. The ATASCII code for 0 is 48. On line 150, then, the match question would translate as:

IF 50=N(48-48)

or

IF 50=N(0)

Back on line 60, we loaded the number 50 into N(0), so the question translates to:

IF 50=50

So there is a match. That means the computer flashed a 0 on the screen, and the person pushed the 0 button before the computer finished counting. The person wins!

## Variables ...

<b>N</b>	Numerical array variable—used for storing the internal keyboard code for the numbers 0 to 9.
<b>TOP</b>	How high the computer counts each round. On the first round, the computer counts to 300. On the fifth round, the computer only counts to 100.
<b>A\$</b>	For Yes (Y) or No (N) answers to the computer's question—do you want me to count faster?
<b>NUMBER</b>	FOR-NEXT loop counter—represents the ATASCII code for the numbers 0 to 9.
<b>COUNT</b>	Counter—how high the computer has counted after flashing the number on the screen. Varies between 1 and TOP.
<b>A</b>	Stores the internal keyboard code of the button the person pushes.

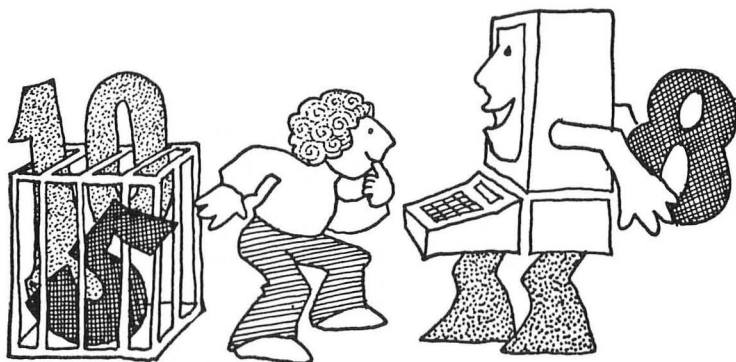
## Do-It-Yourself ...

The program uses the normal happy-face and sad-face subroutines. You can add these subroutines to the program (using the ENTER command). Then you might modify the subroutines to make the computer print different messages if it won or lost. For example, if you won, the computer might say, "WOW! THAT WAS FAST!" or something similar. If you lost, it might say, "YOU SLOWPOKE! THAT WASN'T EVEN MY FASTEST!"

You can add PRINT commands to the front of the program to make the computer challenge the person to a button-pushing duel. You can add some PRINT commands at the end of the program to congratulate a person if he or she wins all four rounds and beats the computer.

13

# GUESS MY NUMBER



## For Parents and Teachers ...

This game helps children learn numbers and number sequence. It helps them become familiar with and adept at manipulating the “greater-than” and “less-than” relationship among numbers. It helps them learn efficient techniques for searching through a group of numbers.

## For Kids ...

In this game, the computer “thinks up” a number between 1 and 20. It asks you your name. (Let’s say your name is Hatty.) “HATTY,” the computer says, “I AM THINKING OF A NUMBER BETWEEN 1 AND 20. WHAT IS IT?”

You try to guess the computer’s number. Let’s say you guess “10.”

The computer might make a sad face and say: “SORRY!”

TOO HIGH! TRY AGAIN!"

You guess the number that is smaller than your last number. You guess "5."

The computer makes another sad face. "TOO HIGH!" it says.

You try again. You guess a number that's even smaller. You guess "3."

The computer makes a happy face. "RIGHT!" it says. It plays a happy tune.

The computer asks if you want to play again. You type "Y" or "YES" and the computer thinks up a new number for you to guess.

Here's a hint to help you guess numbers really fast: always pick a number about halfway between the highest possible number and the lowest possible number. This seems like a strange way to play the game, but it really works.

For example, in the beginning, the computer thinks up a number between 1 and 20. So you should think up a number halfway between 1 and 20. Your number should be 10.

Let's say the computer says 10 is too low. That means the computer's number must be between 11 and 20. This time you need to pick a number halfway between 11 and 20. Your number should be 16.

Let's say the computer says 16 is too high. That means the computer's number must be between 11 and 15. You need to pick a number halfway between 11 and 15. Your number should be 13.

The computer says: "RIGHT!" You guessed the number!

Now you try this trick. It's good for finding a single number buried in a lot of numbers—like finding a needle hidden in a haystack.

## The Game ...

Program Name: **GUESS**

```
50 REM *** NUMBER GUESSING GAME
60 PRINT " \ ":POKE 752,1
```

```

65 POSITION 6,8:PRINT "*** GUESS MY NUM
BER! ***"
66 FOR PAUSE=1 TO 1000:NEXT PAUSE
70 DIM A$(1)
80 DIM N$(15)
100 PRINT " \ "
102 PRINT "WHAT IS YOUR NAME";:INPUT N$
105 N=INT(RND(1)*20)+1
110 PRINT " \ "
120 PRINT N$;" , I AM THINKING OF A NUMBE
R"
130 PRINT "BETWEEN 1 AND 20."
140 PRINT
145 FOR PAUSE=1 TO 500:NEXT PAUSE
150 PRINT "WHAT IS IT";:INPUT G
160 IF G=N THEN GRAPHICS 0:POKE 752,1:GO
SUB 1010:GOTO 200
170 GRAPHICS 0:POKE 752,1:GOSUB 2010:GRA
PHICS 0:POKE 752,1:GOTO 150
200 PRINT " \ "
210 PRINT "WANT TO PLAY AGAIN, ";N$;:INP
UT A$
220 IF A$="Y" THEN 105
230 IF A$<>"N" THEN 200
240 GRAPHICS 0:POKE 752,0:END

```

## Typing Hints ...

» To get the reverse video words on lines 2011 and 2013 in the **Highlights** section of this chapter, you should press the Atari-symbol button (🎮). To turn off the reverse video, press the button again.

## Highlights

The key command in this game is the command on line 105 with the RND function. The RND function randomly selects a number between 1 and 20 and stores it in N.

The children's guess is INPUT into G on line 150. On line 160, the computer checks for a match between G and N. If the numbers match, the computer calls the Happy Face subroutine. If not, the computer calls the Sad Face subroutine.

The Sad Face subroutine has been slightly modified. Lines 2010 to 2013 now look like this:

```
2010 IF G<N THEN 2013
➤ 2011 POSITION 13,10:PRINT "TOO HIGH!!"
2012 GOTO 2015
➤ 2013 POSITION 13,10:PRINT "TOO LOW!!"
```

Lines 2085 to 2100 look like this:

```
2085 POSITION 13,21:PRINT "TRY AGAIN!"
2100 FOR PAUSE=1 TO 300:NEXT PAUSE
```

All SOUND commands following line 2085 have been erased. Children are sure to guess the wrong number several times before guessing the right number. Hearing the sad sound so many times would be irritating and discouraging.

## Variables ...

PAUSE	Delay-loop counter.
A\$	Answer to the question "WANT TO PLAY AGAIN?"
N\$	Your name.
N	Computer's mystery number.
G	Your guess.

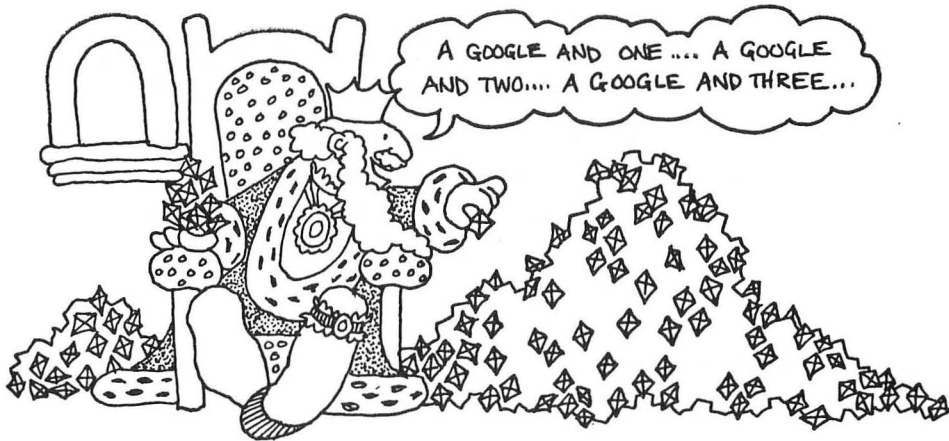
## Do-It-Yourself ...

For younger children you can shrink the range of numbers chosen by the computer. You need to change the PRINT command on line 130 and the RND function on line 105.

For older children you might want to expand the range of numbers the computer selects. You might also want to think up new categories of numbers for the children to guess; for example, even numbers, odd numbers, prime numbers, numbers only divisible by 11, squared numbers, square roots, etc. Numbers have personalities, just as people do. The "Guess My Number" game can teach children those personalities.



# THE RICH KING



## For Parents and Teachers ...

This game helps children practice addition. It helps them sharpen their concentration and memory.

A rich king goes from room to room in his castle, counting his diamonds. The children follow the king through the different-colored rooms. They count the diamonds in each of the rooms. At the end they add up the diamonds in all the rooms.

## For Kids ...

Once upon a time, there was a country named Cybernia. The king of Cybernia was a fat, little man. But he was very, very rich. All of his riches were in the form of diamonds. The king lived in a huge castle. And all the rooms of the castle were

filled with diamonds. Each room was a different color, so the king could remember where he had been and not go back to the same room twice.

Every morning when the king woke up, the first thing he did was walk through all the rooms of his castle to count his diamonds. Every night just before the king went to bed, he put on his pajamas and slippers and brushed his teeth. Then he walked through all his castle rooms and counted his diamonds again.

At night the little king saw diamonds floating in his dreams.

The king had only one problem. He wasn't very good at counting. This made the king very nervous. Sometimes he was so nervous that he couldn't sleep at night, and so, he could not dream about his beautiful diamonds.

To make matters worse, every day the king got a new wagonload of diamonds from the diamond miners in his kingdom. And every day the king had to spend lots of his diamonds to pay his bills.

The king was getting very jittery and very sleepy. Then he came up with a solution. He would hire a diamond counter to help him count his diamonds. Whom did he hire?

**You!**

You are the king's diamond counter. It is your job to follow the little king through the rooms of his castle, counting the diamonds.

You can do this two ways. Either you can count the diamonds one at a time: 1, 2, 3, 4, and so on, through all the rooms. Or, you can count the diamonds in each room, write down the number on a piece of paper, and then add up the numbers for all the rooms to get the total number of diamonds.

Good luck! If you get the number wrong, the king looks sad and moans. He grabs your hand and leads you back through the rooms in the castle to count the diamonds again.

Each time you get the number of diamonds right, the king smiles at you, shouts "RIGHT!" and plays a happy tune on his royal whistle. Maybe someday he will even give you a bag of diamonds as a reward!

## The Game ...

Program Name: **KING**

```
5 REM *** COUNTING DIAMONDS
10 REM PROGRAM AUTHORS.
20 REM BETH ANN HOSTUTLER AND
30 REM JONI BURDETTE.
60 GRAPHICS 2+16
65 POSITION 8,2:PRINT #6;"THE"
66 POSITION 8,4:PRINT #6;"RICH"
67 POSITION 8,6:PRINT #6;"KING"
69 FOR PAUSE=1 TO 1000:NEXT PAUSE
70 DIM A$(3)
80 DIM B$(38)
81 B$="
      "
85 DIM C$(20)
92 DIM C(20)
93 RESTORE 6000
94 FOR I=1 TO 20:READ N:C(I)=N:NEXT I
95 DIM M$(1)
100 REM GENERATE NUMBER OF
105 REM DIAMONDS(1-50).
110 DIAMONDS=INT(RND(0)*50)+1
115 TOTAL=0
120 RESTORE 6000
200 FOR J=1 TO 20
202 GRAPHICS 8
205 POKE 752,1
210 SETCOLOR 2,C(J),10
215 SETCOLOR 1,0,0
220 COLOR 1
225 REM X AND Y ARE COORDINATES TO
227 REM BEGIN DRAWING KING.
230 X=250+INT(RND(0)*45):Y=INT(RND(0)*12
0)
235 REM STMTS 240-410 DRAW KING.
240 PLOT X-7,Y:PLOT X,Y:PLOT X+7,Y
250 PLOT X-7,Y+1:PLOT X-6,Y+1:PLOT X-1,Y
+1:DRAWTO X+1,Y+1:PLOT X+6,Y+1:PLOT X+7,
Y+1:PLOT X-7,Y+2
260 PLOT X-6,Y+2:PLOT X-2,Y+2:DRAWTO X+2
,Y+2:PLOT X+6,Y+2:PLOT X+7,Y+2:PLOT X-7,
```

```
Y+3:DRAWTO X-5,Y+3
270 PLOT X-3,Y+3:DRAWTO X+3,Y+3
280 PLOT X+6,Y+3:PLOT X+7,Y+3:PLOT X-6,Y
+4:DRAWTO X+6,Y+4:PLOT X-6,Y+5:DRAWTO X+
6,Y+5:PLOT X-5,Y+6
290 DRAWTO X+5,Y+6:PLOT X-4,Y+7:DRAWTO X
+4,Y+7:PLOT X-5,Y+8:PLOT X+5,Y+8:PLOT X-
6,Y+9:PLOT X-3,Y+9
300 PLOT X+2,Y+9:PLOT X+6,Y+9:PLOT X-7,Y
+10:PLOT X+3,Y+10:PLOT X+6,Y+10:PLOT X-7
,Y+11:PLOT X+3,Y+11
310 PLOT X+6,Y+11:PLOT X-6,Y+12:PLOT X-2
,Y+12:PLOT X+2,Y+12:PLOT X+6,Y+12:PLOT X
-5,Y+13:DRAWTO X-3,Y+13
320 PLOT X+5,Y+13:PLOT X-4,Y+14:PLOT X+4
,Y+14:PLOT X-3,Y+15:DRAWTO X+3,Y+15
330 PLOT X-4,Y+16:DRAWTO X+4,Y+16:PLOT X
-4,Y+17:PLOT X-2,Y+17:DRAWTO X+5,Y+17:PL
OT X-4,Y+18:PLOT X-1,Y+18
340 DRAWTO X+6,Y+18:PLOT X-4,Y+19:PLOT X
+1,Y+19:DRAWTO X+6,Y+19:PLOT X-5,Y+20:PL
OT X+2,Y+20
350 DRAWTO X+7,Y+20:PLOT X-5,Y+21:PLOT X
-1,Y+21:PLOT X+3,Y+21:DRAWTO X+7,Y+21:PL
OT X-6,Y+22:PLOT X-1,Y+22
360 PLOT X+1,Y+22:PLOT X+3,Y+22:DRAWTO X
+8,Y+22:PLOT X-6,Y+23:PLOT X-2,Y+23:PLOT
X+1,Y+23:PLOT X+3,Y+23
370 DRAWTO X+8,Y+23:PLOT X-6,Y+24:PLOT X
-2,Y+24:PLOT X+1,Y+24:PLOT X+4,Y+24:DRAW
TO X+9,Y+24
380 DRAWTO X+8,Y+23:PLOT X-6,Y+24:PLOT X
-2,Y+24:PLOT X+1,Y+24:PLOT X+4,Y+24:DRAW
TO X+9,Y+24
390 PLOT X-7,Y+25:DRAWTO X-7,Y+36:PLOT X
-2,Y+25:PLOT X+1,Y+25:PLOT X+4,Y+25:DRAW
TO X+8,Y+25:PLOT X-3,Y+26
400 PLOT X+1,Y+26:PLOT X+5,Y+26:DRAWTO X
+9,Y+26:PLOT X-2,Y+27:PLOT X+1,Y+27:PLOT
X+5,Y+27:DRAWTO X+8,Y+27
410 PLOT X,Y+28:PLOT X+1,Y+28:PLOT X+6,Y
+28:DRAWTO X+6,Y+36:PLOT X-7,Y+36:DRAWTO
X+6,Y+36
420 GOSUB 3000:REM PRINT DIAMONDS.
430 REM ARE ALL DIAMONDS PRINTED?
```

```
435 IF TOTAL<DIAMONDS THEN 437
436 POP :GOTO 438
437 NEXT J
438 TRAP 438
440 PRINT " \ "
450 POKE 656,1:PRINT "HOW MANY DIAMONDS
IN ALL ROOMS";:INPUT ANS
455 REM IS PLAYER'S SUM CORRECT?
460 IF ANS=DIAMONDS THEN 480
470 GRAPHICS 0:POKE 752,1:GOSUB 2010:REM
SAD FACE
475 TOTAL=0:GOTO 200
480 GRAPHICS 0:POKE 752,1:GOSUB 1010:REM
HAPPY FACE
490 GRAPHICS 0:POKE 752,1
500 POSITION 14,8:PRINT "PLAY AGAIN";:IN
PUT M$
510 IF M$="Y" THEN 100
520 IF M$<>"N" THEN 490
530 GRAPHICS 0:POKE 752,0:END
3000 REM *** DIAMONDS SUBROUTINE
3010 ROOM=INT(RND(1)*21)+1
3012 IF TOTAL+ROOM>DIAMONDS THEN ROOM=DI
AMONDS-TOTAL
3015 COUNT=0
3018 FOR B=7 TO 119 STEP 56
3020 FOR A=20 TO 200 STEP 30
3030 REM STMTS 3040-3160 PRINT THE
3035 REM DIAMONDS
3040 PLOT A,B:DRAWTO A-10,B+7:PLOT A-10,
B+7:DRAWTO A-10,B+13:PLOT A-10,B+13:DRA
WTO A,B+20:PLOT A,B+20
3050 DRAWTO A+10,B+13:PLOT A+10,B+13:DRA
WTO A+10,B+7:PLOT A+10,B+7:DRAWTO A,B:PL
OT A,B:DRAWTO A,B+6
3060 PLOT A,B+6:DRAWTO A-4,B+9:PLOT A-4,
B+9:DRAWTO A-10,B+7:PLOT A-4,B+9:DRAWTO
A-4,B+11:PLOT A-4,B+11
3070 DRAWTO A-10,B+13:PLOT A-4,B+11:DRA
WTO A,B+14:PLOT A,B+14:DRAWTO A,B+20:PLOT
A,B+14:DRAWTO A+4,B+11
3080 PLOT A+4,B+11:DRAWTO A+10,B+13:PLOT
A+4,B+11:DRAWTO A+4,B+9:PLOT A+4,B+9:DR
AWTO A+10,B+7
```

```
3100 PLOT A+4,B+9:DRAWTO A,B+6
3110 COUNT=COUNT+1:TOTAL=TOTAL+1
3120 IF COUNT=ROOM THEN POP :POP :GOTO 3
157
3150 NEXT A
3155 NEXT B
3157 RESTORE 5000+J-1:READ C$
3160 PRINT "COUNT DIAMONDS IN ";C$;" ROO
M."
3162 PRINT
3163 PRINT "PRESS 'RETURN' FOR NEXT ROOM
";
3164 INPUT A$
3165 RETURN
5000 DATA GREEN
5001 DATA GOLD
5002 DATA LAVENDER
5003 DATA DARK BLUE
5004 DATA RED-ORANGE
5005 DATA PURPLE
5006 DATA ORANGE
5007 DATA GRAY
5008 DATA ORANGE-GREEN
5009 DATA TURQUOISE
5010 DATA SKY-BLUE
5011 DATA PINK
5012 DATA LIGHT-ORANGE
5013 DATA LIGHT-BLUE
5014 DATA SEA-GREEN
5015 DATA YELLOW-GREEN
5016 DATA GOLD
5017 DATA LAVENDER
5018 DATA DARK BLUE
5019 DATA RED-ORANGE
6000 DATA 12,1,5,7,3,6,2,0,14,10,8,4,15,
9,11,13,1,5,7,3
```

## Background ...

The Rich King game was designed and programmed by Joni Burdette and Beth Ann Hostutler, two students at Patrick Henry High School in Roanoke, Virginia.

The girls took the high-school computer course to enhance their interest in computers. They became so excited with computers that they began planning for a computer career. Both girls became enthusiastic computer gamers at a local game arcade.

## Highlights ...

This program reads twenty colors into the array, C. These are the colors of the king's rooms. The SETCOLOR values are in the DATA command on line 6000. The color names are in the DATA commands on lines 5000 to 5019.

The major program loop is on lines 200 to 437. The room the king is in depends on the loop counter, J.

When we enter each room the king is in a different position. This is accomplished with the RND functions on line 230. They select random values for the X and Y variables that determine the king's column and row position on the TV screen.

The king is drawn on lines 227 to 410.

The diamonds are drawn by the subroutine beginning on line 3000.

## Variables ...

PAUSE	Delay-loop counter.
A\$	Accepts RETURN button—to advance to king's next diamond room.
B\$	Filled with empty spaces—to erase text messages.
C\$	Name of the color of the current diamond room.
C	Array—stores the 20 SETCOLOR values (for up to 20 diamond rooms).

<b>I</b>	Loop counter.
<b>N</b>	Temporary storage for room-color value (on its way to the color array C).
<b>M\$</b>	Your answer to the question PLAY AGAIN?
<b>DIAMONDS</b>	The total number of diamonds in all of the king's rooms.
<b>TOTAL</b>	Total number of diamonds counted at any point in the game.
<b>J</b>	Loop counter—main program loop (allows the king to visit up to 20 diamond rooms).
<b>X</b>	Column position of the king.
<b>Y</b>	Row position of the king.
<b>ANS</b>	Your count—the number of diamonds in all of the king's rooms.
<b>ROOM</b>	Number of diamonds in the current room.
<b>COUNT</b>	Number of diamonds drawn in the current room at any point in the diamond-drawing loop.
<b>B</b>	Row position of a diamond.
<b>A</b>	Column position of a diamond.

## Do-It-Yourself ...

The program puts a lot of diamonds in each room (anywhere from 1 to 21 diamonds). This results in the king visiting fewer rooms. You can change the RND function on line 3010 to make the computer put fewer diamonds in each room. This way the king can visit more rooms, and the children won't have to keep track of so many diamonds in each room.

For younger children you can reduce the total number of diamonds by modifying the RND function on line 110.



Also, for younger children, you can have the king introduce himself at the beginning of the program (using big PRINT #6 letters in Graphics Mode 2—"HI! I'M THE RICH KING."). You can have the king explain the game and advise the child to have a piece of paper to record the number of diamonds in each of the different-colored rooms. "When I am done walking through my castle," the king might say, "you can add up all the diamonds and tell me how rich I really am."

Finally, you might consider having the king draw the diamonds more slowly, so that very young children can add them up as they appear on the screen.

# 15 **THE DISAPPEARING GHOSTS**



## **For Parents and Teachers ...**

This game helps children practice subtraction.

A number of ghosts appear above their tombstones. They disappear. Then some of the ghosts reappear. The game asks children how many of the ghosts disappeared and didn't return.

## **For Kids ...**

Imagine that it is late at night. You and your best friend are late for dinner. You take a shortcut across an old cemetery.

You hear a noise. You both turn around.

You are surrounded by ghosts! They are everywhere, peeking over the tops of tombstones.

You start to scream!

The ghosts disappear.

Then some of them return!

This time you don't take the time to scream. You plan to do some quick disappearing yourself. You grab your friend's hand and run like the wind!

You return home, puffing and panting. You tore your pants scrambling over the cemetery fence. You look over your shoulder. No ghosts.

You sit down at your computer. You load in the GHOSTS game and type **RUN**.

There they are again! Ghosts' heads pop over the top of 12 tombstones. You count them.

The ghosts disappear.

Then seven of the ghosts return.

The computer asks you: "HOW MANY GHOSTS THE FIRST TIME?"

You answer: "12."

The computer asks you: "HOW MANY GHOSTS THE SECOND TIME?"

You answer: "7."

Then the computer asks you: "HOW MANY OF THE GHOSTS DISAPPEARED?"

This time you kept a careful count. You type the number 5.

A happy blue and white ghost appears on the TV screen. "RIGHT!" it says.

## The Game ...

Program Name: **GHOSTS**

```
5 REM *** DISAPPEARING GHOSTS
10 REM ANGELA BRADSHAW
50 GRAPHICS 2+16
51 POSITION 4,3:PRINT #6;"DISAPPEARING"
52 POSITION 7,6:PRINT #6;"GHOSTS"
53 FOR PAUSE=1 TO 1000:NEXT PAUSE
60 DIM A(10):REM STORES VALUES OF LIMIT
ON PRINT LOOP
70 A(1)=15:A(2)=45:A(3)=75:A(4)=105
```

```
75 A(5)=135:A(6)=165:A(7)=195
80 A(8)=225:A(9)=255:A(10)=285
90 DIM A$(1)
100 N=INT(RND(0)*25)+1
105 N2=INT(RND(0)*25)+1
106 IF N<N2 THEN Z=N:N=N2:N2=Z
107 OLDN=N:OLDN2=N2
108 GRAPHICS 8:POKE 752,1
110 SETCOLOR 2,0,0:COLOR 1
➤ 122 POKE 657,11:PRINT "count the ghosts!"
125 COUNT=0:Y=0
130 REM SET PRINT LOOPS INITIAL LIMIT
140 L=A(10)
165 N1=N
170 REM SET LIMIT ON PRINT LOOP
190 REM RESET # OF GHOSTS LEFT
202 IF N<=0 THEN 260
205 IF N<=10 THEN L=A(N)
210 GOSUB 3010:REM PRINT GHOSTS
220 Y=Y+30
230 N=N-10
250 GOTO 170
260 REM BOTH SETS PRINTED?
270 COUNT=COUNT+1
280 IF COUNT=2 THEN 330
305 FOR DELAY=1 TO 2000:NEXT DELAY
310 N=N2
312 Y=0
315 L=A(10)
316 GRAPHICS 8:POKE 752,1
318 SETCOLOR 2,0,0:COLOR 1
➤ 319 POKE 657,9:PRINT "here they are again"
320 GOTO 170
330 TRAP 330
335 PRINT " \ "
337 POKE 656,0:POKE 657,4
➤ 340 PRINT "HOW MANY GHOSTS THE 1ST TIME"
;:INPUT E
342 IF E=N1 THEN 344
343 GOTO 360
➤ 344 PRINT "HOW MANY GHOSTS THE 2ND TIME"
;:INPUT F
345 IF F=N2 THEN 347
346 GOTO 360
347 PRINT :PRINT "HOW MANY GHOSTS DISAPP
```

```

EARED";:INPUT G
350 IF G=N1-N2 THEN 380
360 GRAPHICS 0:POKE 752,1:GOSUB 2010
370 N=OLDN:N2=OLDN2:IF F=0 OR F=N2 THEN
108
375 COUNT=1:GOTO 310
380 GRAPHICS 0:POKE 752,1:GOSUB 1010
390 GRAPHICS 0:POKE 752,1
400 POSITION 13,8:PRINT "PLAY AGAIN";:IN
PUT A$
410 IF A$="Y" THEN 100
420 IF A$<>"N" THEN 390
430 GRAPHICS 0:POKE 752,0:END
3000 REM PRINT GHOSTS
3010 FOR X=15 TO L STEP 30
3060 PLOT X-4,Y:DRAWTO X+4,Y
3070 PLOT X-5,Y+1:DRAWTO X+5,Y+1
3080 PLOT X-7,Y+2:DRAWTO X+7,Y+2
3090 PLOT X-8,Y+3:DRAWTO X+8,Y+3
3100 PLOT X-9,Y+4:DRAWTO X+9,Y+4
3110 PLOT X-9,Y+5:DRAWTO X-6,Y+5
3120 PLOT X-4,Y+5:DRAWTO X+4,Y+5
3130 PLOT X+7,Y+5:DRAWTO X+9,Y+5
3140 PLOT X-10,Y+6:DRAWTO X+10,Y+6
3150 PLOT X-10,Y+7:DRAWTO X+10,Y+7
3160 PLOT X-10,Y+8:DRAWTO X+10,Y+8
3170 PLOT X-11,Y+9:DRAWTO X+11,Y+9
3180 PLOT X-11,Y+10:DRAWTO X+11,Y+10
3190 PLOT X-11,Y+11:DRAWTO X+11,Y+11
3200 PLOT X-11,Y+12:DRAWTO X+11,Y+12
3210 PLOT X-12,Y+13:DRAWTO X+12,Y+13
3220 PLOT X-12,Y+14:DRAWTO X+12,Y+14
3230 PLOT X-12,Y+15:DRAWTO X+12,Y+15
3240 PLOT X-13,Y+16:DRAWTO X+13,Y+16
3250 PLOT X-13,Y+17:DRAWTO X+13,Y+17
3260 FOR PAUSE=1 TO 100:NEXT PAUSE
3265 NEXT X
3270 RETURN

```

## Typing Hints ...

➤ To type the words on lines 122, 319, 340, and 344 in reverse video, press the Atari-symbol button (⌘). To turn off the reverse video, press the button again.

## Background ...

The Disappearing Ghosts game was designed and programmed by Angela Bradshaw, a student at Patrick Henry High School in Roanoke, Virginia.

In order to finish her program on schedule, Angela took the Atari computer home from school with her. She spent an entire weekend teaching the computer how to draw the ghosts' shape just right.

When she returned to school on Monday morning she showed off her ghosts. Her teacher decided that the ghosts' shape still wasn't realistic enough. Angela had to rewrite the entire program.

Angela was disappointed. She almost gave up. But instead, she spent all day Monday creating a new ghosts program.

On Tuesday Angela showed her teacher the new ghosts. He liked them! And Angela was proud of her new program. This is the program she created.

## Highlights ...

The number of ghosts in the first group is chosen by the RND function on line 100. The number of the ghosts in the second group is chosen by the RND function on line 105.

The ghosts are drawn in Graphics Mode 8 by the subroutine beginning on line 3000.

## Variables ...

**PAUSE** Delay-loop variable.

**DELAY** Delay-loop variable.

**A** Array—10 values that correspond to the final column position (X-value) of the 10 ghosts.

**A\$** Your answer to the question PLAY AGAIN?

N	Number of the ghosts in the first group.
N2	Number of the ghosts in the second group.
Z	Temporarily stores the number of ghosts in the first group (only used when the number of ghosts in the first group is less than the number of ghosts in the second group—subtracting the second number from the first would result in a negative number, so the numbers are reversed).
OLDN	Value of N is stored in OLDN, in case the child gets the answer wrong and the ghosts have to be redrawn.
OLDN2	Stores value of N2 in case ghosts need to be redrawn.
COUNT	Counts number of groups of ghosts (total of two groups).
Y	Row position of the ghosts.
L	Stores the column position of the last ghost in the current row (there may be up to 10 ghosts in a row).
G	Your answer—How many ghosts disappeared?
X	Column position of current ghost being drawn on the TV screen.

## Do-It-Yourself ...

The program doesn't keep a count of the number of incorrect answers. You can create a variable, `WRONG`, and increment it (`WRONG=WRONG+1`) each time the child types in the wrong answer. Then if `WRONG > 2`, you can print the correct answer and have the computer make the same number of ghosts disappear again.

You might also consider adding sound effects to this program. You can start the program with eerie noises—wind whistling, howling wolves, and so forth. Then, each time a ghost appears, you can have the computer make a whooshing or popping noise.

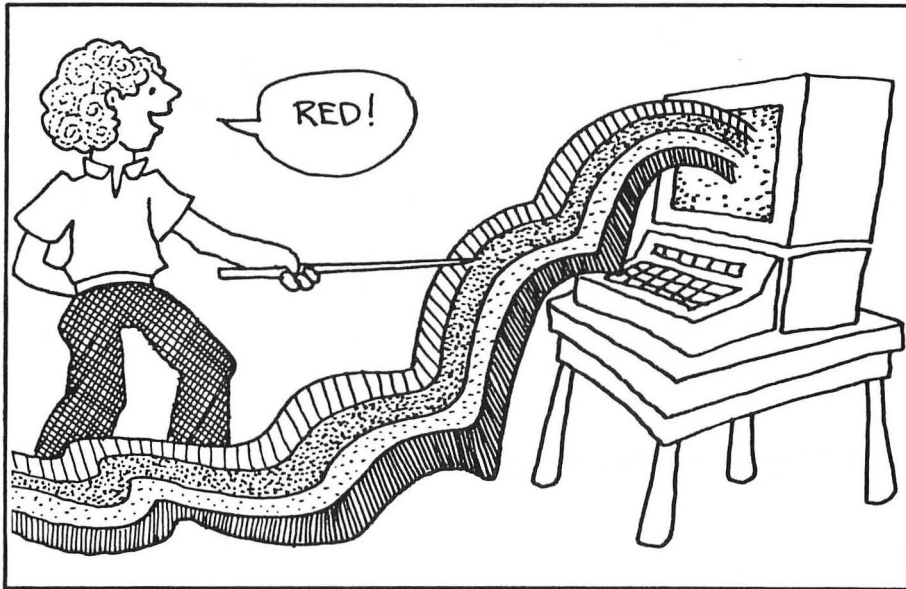
You can also make the ghosts appear in a different way. The ghosts disappear now by having the computer reenter Graphics Mode 8 (thus erasing the entire screen). Instead, you can redraw the ghosts using COLOR 0. This will erase the ghosts one at a time. It will make it easier for the children to subtract the ghosts and get the right answer. It might also make the scene in the cemetery a little more realistic and a little scarier!





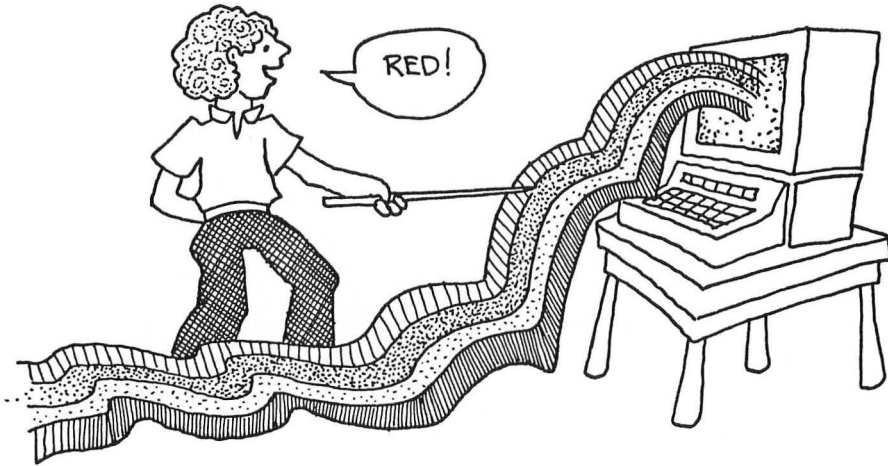
---

# COLORS



16

# NAME THAT COLOR!



## For Teachers and Parents ...

This game helps children learn colors and the color names. The computer flashes a color on the picture screen. It also displays a color name on the middle of the screen. If the color name does not match the color, the child presses the **SPACE** bar and a new color name appears. The child asks the computer for new color names until he finds the color name that matches the color. Then he presses the **RETURN** button.

## For Kids ...

Pretend that you are in a carnival funhouse. Each of the rooms in the funhouse is a different color. You are stuck in a room until you guess its color. Then you can go on to the next room. Can you escape from the funhouse?

## The Game ...

Program Name: **COLOR**

```
50 REM *** NAME THAT COLOR!
60 PRINT "\ ":POKE 752,1
62 POSITION 7,8:PRINT "*** NAME THAT CO
LOR! ***"
64 FOR PAUSE=1 TO 1000:NEXT PAUSE
65 DIM NAME$(6),GUESS$(6)
69 DIM CHART$(60)
70 CHART$="GREEN BLUE PURPLEPINK ORANG
ERED GRAY WHITE BLACK GOLD "
80 GRAPHICS 0
90 POKE 752,1
100 FOR I=1 TO 10
110 READ HUE,LUM,NAME$
120 SETCOLOR 2,HUE,LUM
130 FOR J=1 TO 55 STEP 6
135 POSITION 16,8:PRINT " "
140 POSITION 16,8:PRINT CHART$(J,J+5)
150 POKE 764,255
155 OPEN #1,4,0,"K:"
160 KEY=PEEK(764):IF KEY=255 THEN 160
165 CLOSE #1
170 IF KEY=33 THEN 210
180 IF KEY<>12 THEN KEY=255:GOTO 160
185 GOSUB 2510:REM *** REMOVE BLANKS
190 IF GUESS$=NAME$ THEN GOSUB 1010:POP
:GOTO 220
200 GOSUB 2010
210 NEXT J
215 IF GUESS$<>NAME$ THEN 130
220 NEXT I
250 POKE 752,0:GRAPHICS 0:END
2500 REM *** REMOVE EMPTY SPACES
2510 GUESS$=CHART$(J,J+5)
2520 FOR TAIL=6 TO 4 STEP -1
2530 IF GUESS$(TAIL,TAIL)<>" " THEN POP
:GOTO 2550
2540 NEXT TAIL
2550 GUESS$=GUESS$(1,TAIL)
2560 RETURN
```

```
3000 REM *** COLOR DATA
3010 DATA 2,15,GOLD,0,0,BLACK,0,8,GRAY,0
,15,WHITE,3,5,RED,2,12,ORANGE,4,15,PINK
3020 DATA 5,6,PURPLE,6,2,BLUE,12,15,GREE
N
```

## Highlights ...

The success of this program depends on how well the colors on the screen resemble the true colors. For example, if the computer's pink looks more like purple, it is going to be hard for your child to get the right answer.

Try the program out on your TV set. To get the colors just right, you might have to play with the color and tint knobs. If the colors still aren't right, you might have to experiment with the SETCOLOR command. Change the luminances and hues until you are satisfied.

This program is made of two FOR-NEXT loops and a new subroutine. The outer loop (lines 100 to 220) takes you through the 10 colored rooms in the funhouse: the Green Room, the Blue Room, the Purple Room, the Pink Room, the Orange Room, the Red Room, the Gray Room, the White Room, the Black Room, and the Gold Room.

The inner loop (lines 130 to 210) prints the name of the 10 colors, one at a time, on the TV screen (the wall of the room). If you think the name matches the color of the room, you press the **RETURN** button. If not, you press the **SPACE** bar, and a new name appears.

If you guess right, line 190 calls the Happy Face subroutine (GOSUB 1010). Then you POP out of the inner loop and go on to the next colored room in the funhouse.

If you guess wrong, line 200 calls the Sad Face subroutine (GOSUB 2010). Then the inner loop prints a new color name on the wall of the room. If the loop is complete and all the colors have been printed, then (on line 215) the program goes back to line 130, and the inner loop begins naming the colors again. You then have another chance.

When you have made it through all the rooms, the outer loop ends, and (on line 250) the program ends. You have escaped from the funhouse!

The new subroutine (lines 2500 to 2560) removes the blank spaces from a color name stored in the color-name string called **CHART\$** and stores the name in **GUESS\$**. The color names flashed on the TV screen come from **CHART\$**. The **real** color (to match the color on the screen) is **READ** into **NAME\$** on line 110. The color in **NAME\$** has no blanks. So before a comparison between **NAME\$** and **GUESS\$** can be made, the trailing blanks have to be removed from the color name in **GUESS\$**.

After the color name in **GUESS\$** loses its blanks, the subroutine ends, and (on line 190) the program looks for a match between **GUESS\$** (your guess) and **NAME\$** (the true color of the room).

## Variables ...

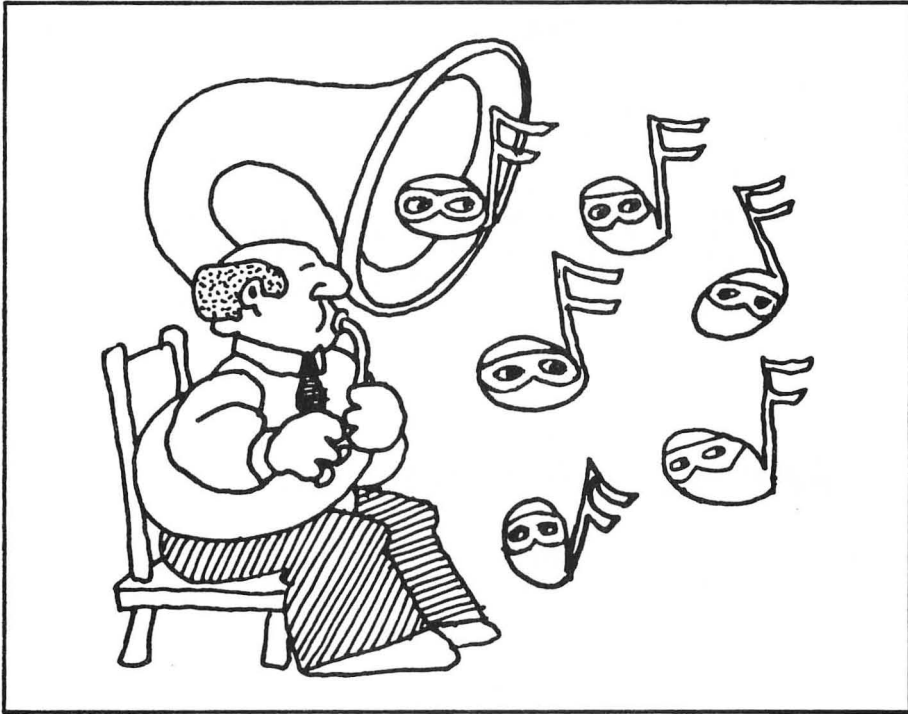
<b>NAME\$</b>	The name of the color of a funhouse room.
<b>GUESS\$</b>	The color name that you guess.
<b>CHART\$</b>	A string containing all the possible colors.
<b>I</b>	FOR-NEXT loop counter (outer loop).
<b>HUE</b>	Color setting for SETCOLOR.
<b>LUM</b>	Luminance (brightness) setting for SETCOLOR.
<b>J</b>	FOR-NEXT loop counter (inner loop).
<b>KEY</b>	The button you press—either the <b>RETURN</b> button (keyboard code = 12) to select a color, or the <b>SPACE</b> bar (keyboard code = 33) to have the computer print out a new color name.
<b>TAIL</b>	FOR-NEXT counter in subroutine—character position in <b>GUESS\$</b> (checking for blank space following color name).

## **Do-It-Yourself ...**

Like the other programs in this book, **COLOR** can form the nucleus or core of a much more elaborate program. As it stands, the program makes no reference to a funhouse. You can add **PRINT** commands to the beginning of the program and announce "WELCOME TO THE FUNHOUSE!" The **PRINT** commands can give children directions for escaping from the funhouse. You can even add sound and graphics commands to make the funhouse seem more realistic.

---

# MUSIC





# 17 **DO-RE-MI**



## **For Parents and Teachers ...**

This game helps children learn the names of the notes of the C-major scale in the octave from middle C to high C. It also helps them learn the pitch of each note.

## **For Kids ...**

Pretend you are sitting outside, on the grass, on a warm summer evening. You look up at the stars and the moon, and you hear music. Actually, what you hear is a single note floating somewhere, invisible, high in the air.

You have your flashlight with you. You turn on the light and take out your **Kid's Guide to Nighttime Music**. You try to identify the musical note playing in the dark sky.

Is it C? D? E? F? G? A? B? Or High C?

You guess that it is middle C. You look up at the moon and hum the note. "Middle C," you say. "Is it Middle C?"

The moon begins to smile. "Right!" it calls. The moon and stars play a happy tune.

Now a new mystery note glides through the air. You switch your flashlight back on and turn your book's pages. Is the new note a middle C? Is it a D? An E?

## The Game ...

Program Name: **NOTEGAME**

```

50 REM *** DO-RE-MI GAME
60 PRINT " \ ":POKE 752,1
65 POSITION 6,8:PRINT "***      DO-RE-MI  G
AME      ***"
67 FOR PAUSE=1 TO 1000:NEXT PAUSE
70 DIM A$(1)
80 DIM KEY(8)
90 DIM MUSIC(8)
92 GOSUB 3010
100 GRAPHICS 2:POKE 752,1
110 GOSUB 3510
115 OPEN #1,4,0,"K:"
120 FOR N=1 TO 8
130 SOUND 0,MUSIC(N),10,15
140 FOR J=1 TO 8
142 POSITION 7,5:PRINT #6;"      "
145 IF J<>8 THEN 150
147 POSITION 7,5:PRINT #6;"HI-";CHR$(KEY
(J))
148 GOTO 170
150 POSITION 8,5:PRINT #6;CHR$(KEY(J))
170 GET #1,K
180 IF K=32 THEN 230
190 IF K<>155 THEN 170
200 IF J=N THEN GRAPHICS 0:POKE 752,1:GO
SUB 1010:GRAPHICS 2:POKE 752,1:POP :GOTO
240
210 GRAPHICS 0:POKE 752,1:GOSUB 2010:GRA
PHICS 2:POKE 752,1:SOUND 0,MUSIC(N),10,1

```

```
5:GOSUB 3510
230 NEXT J
235 GOTO 140
240 IF N<8 THEN GOSUB 3510
245 NEXT N
250 GRAPHICS 0:POKE 752,1
260 POSITION 13,8:PRINT "PLAY AGAIN";:IN
PUT A$
270 IF A$="Y" THEN CLOSE #1:GOTO 100
280 IF A$<>"N" THEN 250
290 GRAPHICS 0:POKE 752,0:END
3000 REM *** LOAD KEYS
3010 FOR I=1 TO 8
3020 READ K,M
3030 MUSIC(I)=K
3035 KEY(I)=M
3040 NEXT I
3050 RETURN
3500 REM *** SCREEN MESSAGE
3510 PRINT "      ***   WHICH NOTE?      ***"

3520 RETURN
5000 DATA 121,67,108,68,96,69,91,70,81,7
1,72,65,64,66,60,67
```

## Highlights ...

When you RUN this program, the computer shifts into Graphics Mode 2. The computer plays a single musical note. On the screen, in gold, appears a note. In the text area at the bottom of the screen, the computer prints the question: "WHICH NOTE?"

If you believe that the note on the screen matches the note being played, press the **RETURN** button. If not, press the **SPACE** bar. Each time you press the **SPACE** bar, the note on the screen changes to the next higher note, until you reach high C. Then the computer goes back to the beginning of the octave and displays the first note, (middle) C.

If you get a note wrong, the computer makes a sad face. Then it keeps playing the same note until you guess it right.

If you get a note right, the computer makes a happy face. Then it tests you on the next note in the octave. The note

displayed on the screen goes back to middle C. You have to press the **SPACE** bar in order to change the note to match the new note being played.

The computer lets you correctly guess eight notes. Then it asks you if you want to "PLAY AGAIN?" If you type "Y" or "YES," you get to play a new game with eight more notes.

## Variables ...

<b>PAUSE</b>	Delay-loop counter.
<b>A\$</b>	Accepts answer to "PLAY AGAIN?" question.
<b>KEY</b>	Array—button associated with each note—C, D, E, F, G, A, B, and C (for high C).
<b>MUSIC</b>	Array—musical tones (middle C through high C).
<b>N</b>	Loop counter—major loop (lets you guess eight notes).
<b>J</b>	Loop counter—this loop selects the note displayed on the TV screen.
<b>K</b>	Your guess (which note?).
<b>I</b>	Loop counter—in subroutine (beginning on line 3000) for reading in eight musical notes and codes for the eight keys representing the musical notes C, D, E, etc.
<b>M</b>	Holds the code value for one key.

## Do-It-Yourself ...

You can do well at this game by keeping track of the last note the computer played. The next note the computer plays will always be one note higher.

For a look at a game that scrambles the notes, turn to the next chapter.

# 18

## SCRAMBLED NOTES



### **For Parents and Teachers ...**

This game will help children learn musical notes of the C-major scale in the octave from middle C to high C. Since the notes are randomly selected by the computer, the game helps children learn to identify the different notes just by hearing them.

### **For Kids ...**

Take a look at the “For Kids” section in the last chapter.

A hint: This game is tougher than the last game. In this game the notes don't get higher one by one. In this game, the notes are all scrambled. You never know which note will be played next!

## The Game ...

Program Name: **NOTEGAM2**

```
50 REM *** SCRAMBLED NOTES
60 PRINT " \ ":POKE 752,1
65 POSITION 6,8:PRINT "***      SCRAMBLED N
OTES      ***"
67 FOR PAUSE=1 TO 1000:NEXT PAUSE
70 DIM A$(1)
80 DIM KEY(8)
90 DIM MUSIC(8)
92 GOSUB 3010
100 FIRST=1:LAST=8:HOP=1:GOSUB 4010
103 FOR PAUSE=1 TO 300:NEXT PAUSE
105 FIRST=8:LAST=1:HOP=-1:GOSUB 4010
106 FOR PAUSE=1 TO 300:NEXT PAUSE
108 GRAPHICS 2:POKE 752,1
110 GOSUB 3510
115 OPEN #1,4,0,"K:"
120 FOR O=1 TO 10
125 N=INT(RND(1)*8)+1
130 SOUND 0,MUSIC(N),10,15
140 FOR J=1 TO 8
142 POSITION 8,5:PRINT #6;"      "
145 IF J<>8 THEN 150
147 POSITION 8,5:PRINT #6;"HI-";CHR$(KEY
(J))
148 GOTO 170
150 POSITION 9,5:PRINT #6;CHR$(KEY(J))
170 GET #1,K
180 IF K=32 THEN 230
190 IF K<>155 THEN 170
200 IF J=N THEN GRAPHICS 0:POKE 752,1:GO
SUB 1010:GRAPHICS 2:POKE 752,1:POP :GOTO
240
210 GRAPHICS 0:POKE 752,1:GOSUB 2010:GRA
PHICS 2:POKE 752,1:SOUND 0,MUSIC(N),10,1
5:GOSUB 3510
230 NEXT J
235 GOTO 140
240 IF O<10 THEN GOSUB 3510
245 NEXT O
```

```

250 GRAPHICS 0:POKE 752,1
260 POSITION 10,8:PRINT "AGAIN";:INPUT A
$
270 IF A$="Y" THEN CLOSE #1:GOTO 100
280 IF A$<>"N" THEN 250
290 GRAPHICS 0:POKE 752,0:END
3000 REM *** LOAD KEYS
3010 FOR I=1 TO 8
3020 READ K,M
3030 MUSIC(I)=K
3035 KEY(I)=M
3040 NEXT I
3050 RETURN
3500 REM *** SCREEN MESSAGE
3510 PRINT "      ***      WHICH NOTE?      ***"

3520 RETURN
4000 REM *** PLAY NOTES
4010 GRAPHICS 2:POKE 752,1
4012 PRINT "      *** LISTEN TO THE NOTES *
**"
4015 FOR J=FIRST TO LAST STEP HOP
4020 SOUND 0,MUSIC(J),10,15
4040 POSITION 8,5:PRINT #6;"      "
4050 IF J<>8 THEN 4080
4060 POSITION 8,5:PRINT #6;"HI-";CHR$(KEY(J))
4070 GOTO 4090
4080 POSITION 9,5:PRINT #6;CHR$(KEY(J))
4090 FOR PAUSE=1 TO 200:NEXT PAUSE
4100 NEXT J
4105 SOUND 0,0,0,0
4110 RETURN
5000 DATA 121,67,108,68,96,69,91,70,81,7
1,72,65,64,66,60,67

```

## Highlights ...

This game is very similar to the game in the last chapter, but there are some changes.

First, when the game begins, the computer plays all eight

notes in the octave and displays the note names on the TV screen. Then it plays the notes backwards, beginning at high C and ending at middle C. While the computer is playing, listen carefully and try to associate each note name with the note being played.

Next the computer erases the screen and randomly picks a note from among the eight notes in the octave. This makes the game much more challenging than the previous game in which the computer always picked the next higher note.

Also, if you guess a note wrong, the computer doesn't go back and display the first note, middle C. This time it displays the note following the note you guessed (incorrectly).

In the last game, you had to guess eight notes correctly to complete the game. In this game, you have to guess 10 notes correctly.

## Variables ...

<b>N</b>	Stores the random note (any note from middle C to high C).
<b>O</b>	Loop counter—major program loop (controls the number of notes you will be challenged with).
<b>J</b>	Loop counter—in subroutine beginning at 4000 (controls playing all the notes in the octave).
<b>FIRST</b>	First note to be played in the loop.
<b>LAST</b>	Last note played in the loop.
<b>HOP</b>	Increment by which the loop is increased or decreased. The first time around, HOP is +1, so the computer plays the octave forward from middle C to high C. The second time, HOP is -1, so the computer plays the octave <b>backwards</b> from high C to middle C.

The other variables are the same as in NOTEGAME. (See the "Variables" section in the last chapter.)



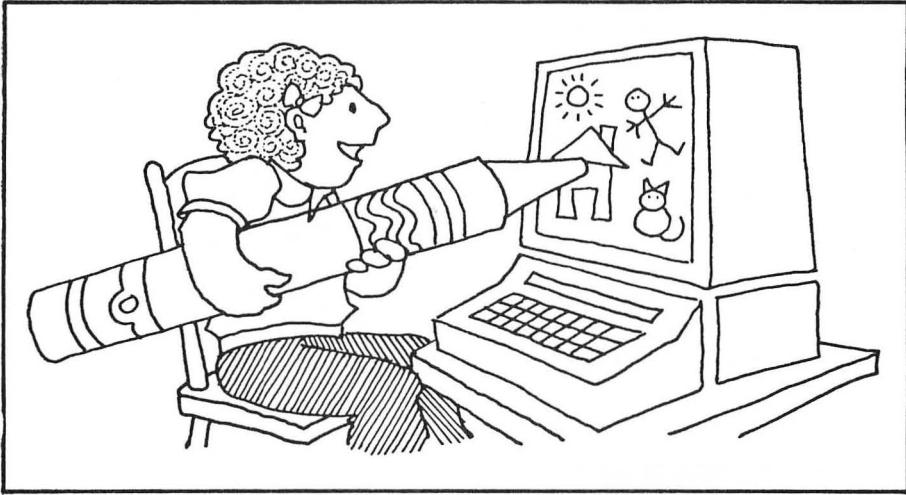
## **Do-It-Yourself ...**

This game plays only the notes of the C-major scale in the octave from middle C to high C. It would be neat if you added the other major and minor scales and expanded the number of notes to include lower (bass) octaves and higher (treble) octaves. This would make the game more challenging and would help children learn more about music.

Also, the game, as it now stands, only uses notes. You can include other facets of a musical score, such as rests and tempo. The game can play a little tune using different tempos (4/4 time, 3/4 time, etc.), and the children can try to guess which. Also, you can have the computer play a scale, and the children can try to guess its name.

---

# DRAWING



# THE COMPUTER CRAYON



## For Parents and Teachers ...

This game turns the computer into an electronic crayon and the TV screen into a sketchpad. This game appeals to all ages. Yet even a younger child can use the game to develop his or her artistic abilities. And it's a great imagination game. The TV screen is empty, and the child can fill it with whatever he or she dreams up.

## For Kids ...

Load the CRAYON program and type **RUN**.

First you get to choose the color of your crayon. It can be an orange crayon, a green crayon, or a blue crayon. Or it can be a special crayon that makes all three colors!

In order to choose the color of your crayon, you type the crayon number (1, 2, 3, or 4). Then the computer screen

darkens. The screen is your electronic sketchpad. You can draw anything you want.

The way you draw is by using the **arrow** buttons on the right side of the keyboard. To draw upward you press the **up-arrow** (↑) button. To draw downward you press the **down-arrow** (↓) button. To draw to the right you press the **right-arrow** (→) button. To draw to the left you press the **left-arrow** (←) button.

Try it out.

What happens when you come to the edge of the screen? Right. The computer stops you. You stay right there until you push a new arrow button and draw in a different direction.

Now that you know how to draw, you need to know something else: how to erase. Erasing is as easy as drawing. You decide which direction you want to erase, you press the correct arrow button, **and at the same time**, you hold down the **CTRL** button (on the lefthand side of the keyboard).

Try erasing. Backtrack over some of the lines you just drew on your electronic paper.

Now keep holding down the **CTRL** button and hold the arrow keys down hard. This makes the little square block (the "crayon") zip around the screen fast. As long as you hold down the **CTRL** button, the crayon doesn't draw anything. But you can still see it. It's like a ghost. It blinks off and on, so you know exactly where it is.

What if you have drawn a picture, and you want to draw something new? Do you have to erase each little color block on the TV screen?

No. Just press the **RETURN** button. Poof! Your drawing vanishes. Now you can try something new.

What should you draw? You can draw anything! I love to draw monster faces, giant letters, and railroad tracks. You might try some mazes, animals, bridges, and planets. Or some beautiful shapes and patterns—floating cubes, squares, spirals, and boxes inside boxes.

The screen is empty. Pick out a crayon and start drawing.

## The Game ...

Program Name: **CRAYON**

```

50 REM *** COMPUTER CRAYON
55 PRINT " \ ":POKE 752,1
60 POSITION 5,8:PRINT "*** THE COMPUTER
   CRAYON ***"
65 FOR PAUSE=1 TO 1000:NEXT PAUSE
70 DIM DRAW(3,4)
72 FOR J=1 TO 4
73 FOR I=1 TO 3
74 READ K
76 DRAW(I,J)=K
77 NEXT I
78 NEXT J
100 GOSUB 4010:REM * CHOOSE COLOR
102 GRAPHICS 3+16
105 X=19:Y=11
107 OPEN #1,4,0,"K:"
110 FOR T=1 TO 3
120 N=DRAW(T,C)
125 TRAP 125
130 GET #1,K
132 IF K=155 THEN POP :CLOSE #1:GOTO 102

135 IF K<42 THEN S=N:N=0
140 IF K=45 OR K=28 THEN Y=Y-1:GOTO 174
150 IF K=61 OR K=29 THEN Y=Y+1:GOTO 174
160 IF K=43 OR K=30 THEN X=X-1:GOTO 174
170 IF K=42 OR K=31 THEN X=X+1:GOTO 174
172 GOTO 130
174 GOSUB 3010:REM * ERROR CHECK
175 IF N=0 THEN POSITION X,Y:PRINT #6;CH
R$(S):FOR U=1 TO 5:NEXT U:POSITION X,Y:P
RINT #6;CHR$(N):GOTO 185
176 POSITION X,Y:PRINT #6;CHR$(N)
185 NEXT T
190 GOTO 110
3000 REM *** ERROR CHECK SUBROUTINE
3010 IF X<0 THEN X=0
3020 IF X>39 THEN X=39
3030 IF Y<0 THEN Y=0
3040 IF Y>23 THEN Y=23

```

```
3050 IF X=39 AND Y=23 THEN X=38:Y=22
3060 RETURN
4000 REM *** PICK COLOR
4010 GRAPHICS 0:POKE 752,1
4020 TRAP 4010
4030 POSITION 10,5:PRINT "1.  ORANGE"
4040 POSITION 10,7:PRINT "2.  GREEN"
4050 POSITION 10,9:PRINT "3.  BLUE"
4060 POSITION 10,11:PRINT "4.  ALL "
4070 POSITION 10,16:PRINT "COLOR (1,2,3,
4) " ;:INPUT C
4075 IF C>=1 AND C<=4 THEN 4080
4076 GOTO 4010
4080 RETURN
5000 DATA 49,49,49,50,50,50,51,51,51,49,
50,51
```

## Highlights ...

On lines 70 to 78, the program loads 12 numbers into the DRAW array. These numbers represent the color of the crayon. The 49 in Atari ASCII code represents a 1. When the computer obeys a PRINT #6;"1" in Graphics Mode 3, an orange block appears.

Likewise, the 50 in Atari ASCII code represents a 2, and the 51 represents a 3. A PRINT #6;"2" will make a green block. A PRINT #6;"3" will make a blue block.

You select the color using the subroutine beginning on line 4000.

All the drawing takes place inside the loop on lines 110 to 185. Elaborate checks are made for each key you press.

First, on line 132, the computer checks to see if you pressed the RETURN button. If so, it leaves the drawing loop, erases the screen, and restarts the drawing loop.

Second, the computer checks to see if you pressed the arrow keys alone or if you pressed the arrow keys along with the CTRL button. If you pressed the CTRL button, that means you wanted to erase instead of draw.

Third, the computer calls a subroutine (at line 3000) to make sure that you are not about to "fall off" the TV screen.

Line 175 takes care of erasing. It displays the drawing cursor for a count of five. Then it erases the cursor and the current color block. It does the erasing with a `PRINT #6;CHR$(N)`, where `N=0` (a space in Atari ASCII).

Note the `TRAP` commands on line 125 and 4020. The `TRAP` command “traps” errors. It is valuable when the computer is expecting a number (for example, after an `INPUT C` command or a `GET #1,K` command). If you don’t have a `TRAP` command, and you don’t type a number, the program crashes. If you have a `TRAP` command, the computer jumps automatically to the line indicated in the `TRAP` command.

A `TRAP` is like a switch. It is on until it is used. Then it is off. The way to turn it on again is to have the computer obey it a second time. You can do this by having the `TRAP` line number be the same line number as the `TRAP` command. For example:

```
125 TRAP 125
```

## Variables ...

<b>PAUSE</b>	Delay-loop counter.
<b>DRAW</b>	Array—Holds color values.
<b>J</b>	Loop counter.
<b>I</b>	Loop counter.
<b>K</b>	Reads in color values; later, reads in arrow value.
<b>X</b>	Column where color block is drawn.
<b>Y</b>	Row where color block is drawn.
<b>T</b>	Loop counter—main drawing loop.
<b>N</b>	Color value for current color block.
<b>C</b>	Points to current color in crayon array <code>DRAW</code> .
<b>S</b>	Saves color value (of <code>N</code> ) when you want to erase.

## Do-It-Yourself ...

There are dozens of things you can do to improve the CRAYON program. For example, you can use the SETCOLOR and the COLOR commands to increase the number of crayon colors the children can choose from.

Or you can draw in a different graphics mode. The CRAYON program uses Graphics Mode 3. But you can change the Error-Check subroutine (lines 3000-3060) to reflect the new screen size, and you can have the children draw in the higher-resolution graphics modes (4 to 11).

You might also be interested in saving the childrens' drawings. Some of them can take a long time and turn out quite beautiful and elaborate. It's a shame to turn off the computer and have them disappear. The computer can read each of the **pixels** (color blocks) on the screen and copy them onto tape or disk. Later the picture can be recalled from tape or disk and flashed instantly on the screen.

The following lines can be added to the program CRAYON and SAVED onto disk or tape. This will give you a combined new version of the program.

```
52 DIM FNAME$(20)
133 IF K=19 THEN GRAPHICS 35:GOTO 500
134 IF K=12 THEN GRAPHICS 3:GOTO 700
500 REM * SAVE DISPLAY *
502 TRAP 40000
503 TRAP 505:PRINT " \ "
504 LPRINT
505 PRINT "ENTER SAVE DEVICE:NAME (C: OR
    D:NAME) "
510 INPUT FNAME$
515 OPEN #2,8,0,FNAME$
520 LOC=PEEK(88)+PEEK(89)*256
525 FOR Q=LOC TO LOC+239
530 PRINT #2;PEEK(Q)
535 NEXT Q
540 CLOSE #2
545 GRAPHICS 51
550 GOTO 130
```



```
700 REM *LOAD DISPLAY *
702 PRINT " \ "
705 PRINT "ENTER LOAD DEVICE:NAME (C: OR
    D:NAME) "
710 INPUT FNAME$
715 OPEN #2,4,0,FNAME$
720 LOC=PEEK(88)+PEEK(89)*256
725 FOR Q=LOC TO LOC+239
730 INPUT #2,A
735 POKE Q,A
740 NEXT Q
745 CLOSE #2
750 GRAPHICS 51
755 GOTO 130
```

To SAVE the TV screen drawing, follow this procedure:

1. LOAD and RUN the CRAYON program (combined new version).
2. Draw a picture on the TV screen.
3. Press **CTRL** and **S** when you are ready to SAVE the picture.
4. Decide on a name for the picture. Type **C:NAME** when prompted. You will hear a beep—be sure to press **REC** and **PLAY** on the tape player, then press **RETURN** on the computer. For disk, type **D:NAME** when prompted.

To LOAD a previously SAVED TV screen drawing, follow this procedure:

1. LOAD and RUN the CRAYON program (combined new version).
2. Press **CTRL** and **L**.
3. Remember the name for the previously SAVED picture. Rewind the tape to the location (counter number) where the picture was SAVED. Type **C:NAME** when prompted. You will hear a beep—press **PLAY** on the

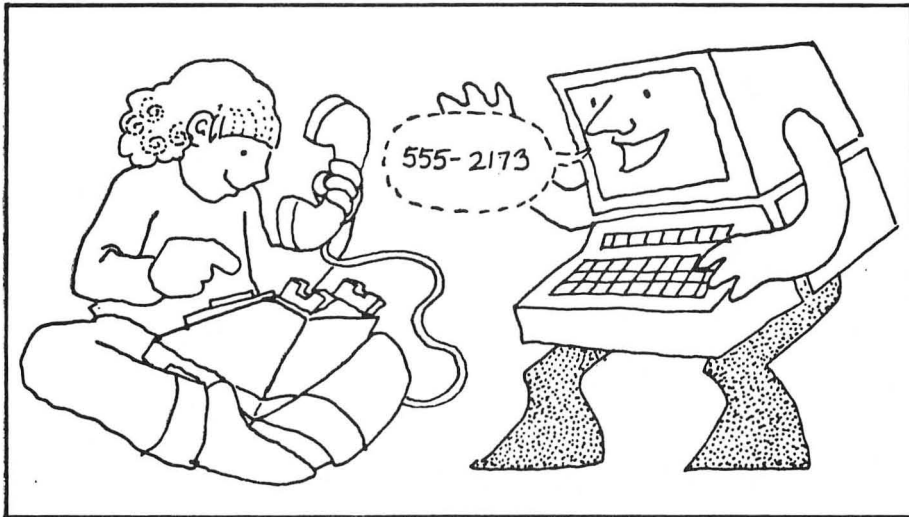
tape recorder, then press **RETURN** on the computer.  
For disk, type **D:NAME** when prompted.

Your TV screen will then display the previously **SAVED** picture. You can change the drawing, if you wish, and **SAVE** it again under a different name, or the same name.



---

# KNOWLEDGE



# WHAT'S YOUR NUMBER?



## For Parents and Teachers ...

This game will help younger children learn their telephone number.

## For Kids ...

The computer is going to help you learn your phone number.

When you type RUN, the phone game begins. The computer asks you for your name and your phone number. Your mom or dad can write it down on a piece of paper. Then you can type the number on the computer's typewriter. Then hide the paper—and don't peek.

The computer will ask you for the number again. Can you remember it?

If you can't remember, try guessing. After three wrong guesses, the computer will let you peek at the number. Then it will hide it and ask if you want to play the game again. Type "YES" and see if you can get the number right.

If it takes you a long time, don't worry. The computer never loses its patience and never gets angry. It won't growl at you or stick out its tongue. It keeps playing the game until you know your phone number perfectly.

## The Game ...

Program Name: **PHONE**

```

5 REM *** LEARN YOUR PHONE NUMBER
10 REM THIS PROGRAM HELPS YOUNGER      CH
   ILDREN TO LEARN THEIR PHONE NUMBER
20 REM MELISSA PERDUE
50 GRAPHICS 0:POKE 752,1
55 POSITION 6,8:PRINT "***  PHONE NUMBER
   GAME  ***"
56 FOR PAUSE=1 TO 1000:NEXT PAUSE
60 REM N$ STORES PLAYER'S NAME
65 DIM N$(30)
70 REM SPH$ STORES PHONE NUMBER
75 DIM SPH$(8)
80 REM FPH$ STORES PHONE NUMBER AS
82 REM REMEMBERED WITHOUT AID
85 DIM FPH$(8)
87 DIM A$(1)
120 GRAPHICS 0:POKE 752,1
130 POSITION 15,8
140 PRINT "HELLO!"
145 POSITION 9,11
150 PRINT "WHAT IS YOUR NAME";:INPUT N$
160 PRINT " \ "
165 POSITION 10,8
170 PRINT "HI,THERE, ";N$;"!"
175 POSITION 10,11
180 PRINT "WHAT IS YOUR"
185 POSITION 10,13
186 PRINT "PHONE NUMBER";:INPUT SPH$

```

```
250 PRINT " \ "  
260 POSITION 12,9  
270 PRINT "CAN YOU REMEMBER"  
271 POSITION 15,11:PRINT "ALL THAT?"  
272 POSITION 14,14  
275 PRINT "LET'S TRY!!!"  
280 FOR DELAY=1 TO 1000:NEXT DELAY  
281 TRAP 281  
282 COUNT=0  
285 PRINT " \ "  
287 POSITION 12,9  
288 PRINT "WHAT IS YOUR"  
289 POSITION 12,11  
290 PRINT "PHONE NUMBER";:INPUT FPH$  
300 IF SPH$=FPH$ THEN PRINT " \ ":GOSUB 1  
010:GOTO 350  
310 COUNT=COUNT+1  
320 IF COUNT>=3 THEN 340  
330 PRINT " \ ":GOSUB 2010:GOTO 285  
340 PRINT " \ ":POSITION 10,8  
345 PRINT "YOUR PHONE NUMBER IS"  
346 POSITION 16,11:PRINT SPH$  
347 FOR DELAY=1 TO 1500:NEXT DELAY  
350 PRINT " \ "  
355 POSITION 13,8:PRINT "PLAY AGAIN";:IN  
PUT A$  
360 IF A$="Y" THEN 282  
370 IF A$<>"N" THEN 350  
380 PRINT " \ ":POKE 752,0:END
```

## Highlights ...

The "What's Your Number?" phone game was developed by Melissa Perdue, a student at Patrick Henry High School in Roanoke, Virginia.

The phone game program stores the phone number in the variable SPH\$. When you enter your guess, it stores that in FPH\$. If FPH\$ matches SPH\$ then the Happy Face appears. If not, the wrong-answer counter, COUNT, is incremented by 1, and the Sad Face appears. If COUNT is greater or equal to 3,

then the computer prints the phone number for you to study. Then it asks if you would like to play the phone game again.

## Variables ...

PAUSE	Delay-loop counter.
N\$	Your name.
SPH\$	Phone number.
FPH\$	Phone number guess.
A\$	Answer to the question "PLAY AGAIN?"

## Do-It-Yourself ...

Your children can use this game to learn their seven-digit phone number or their 10-digit phone number including the area code.

With just a few changes, they can also use this game to learn other phone numbers—their friends' numbers, police, fire department, grandparents, movie theaters, etc.

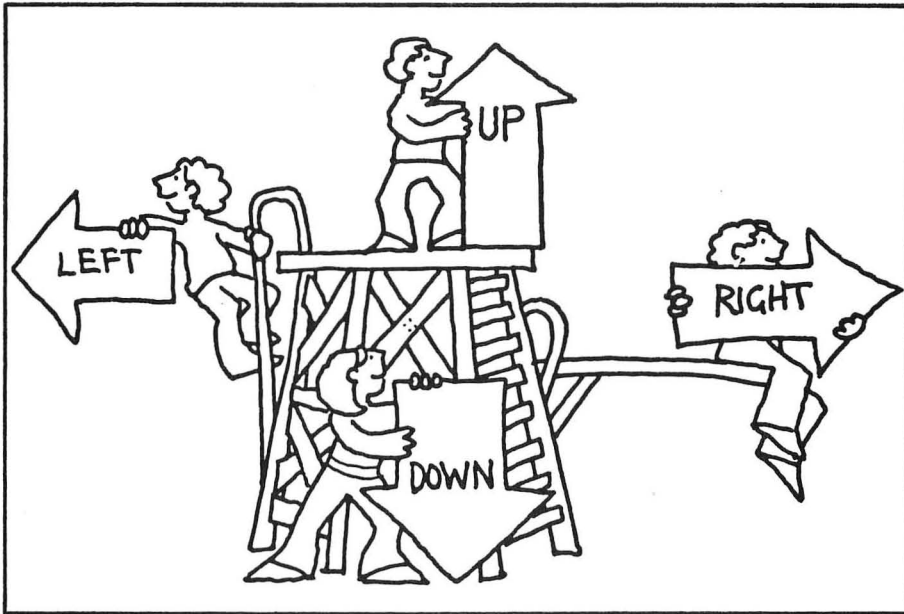
Right now, if the children get their number wrong three times, the program displays the correct number in Graphics Mode 0. You might want to change the program to have the correct number displayed in Graphics Mode 2. Then the digits would be very large and easier for the children to study.





---

# HAND-EYE



21

# UP-DOWN- LEFT-RIGHT



## For Parents and Teachers ...

This game helps children learn directions: up, down, left, and right. It also helps them recognize the direction words.

## For Kids ...

Look for the **arrow** keys on the keyboard. Do you see them? Now, you're ready.

The computer asks you: "WHICH DIRECTION?"

On the screen you see an arrow and a word. What arrow do you see? It's the up arrow. The arrow is pointing to the word "UP." Say the word out loud. Shout it out!

Can you find the same arrow on the computer keyboard? Press it. Hurray!

Now a different arrow and a different word appear. This is the down arrow and the word "DOWN." Shout out the word

"DOWN!" Find the down arrow on the keyboard and press it.

Next, a different arrow and a different word appear. This is the left arrow and the word "LEFT." Shout out the word "LEFT!" Find the left arrow on the keyboard and press it.

One last arrow and one last word appear. This is the right arrow and the word "RIGHT." Shout out the word "RIGHT!" Find the right arrow on the keyboard and press it.

The game is over. The computer will ask you if you want to play again. You should type in "YES" or "Y." This time you are on your own. Good luck!

## The Game ...

Program Name: **UPDOWN**

```

0 GRAPHICS 0
50 REM *** UP/DOWN GAME
60 PRINT " \ ":POKE 752,1
65 POSITION 3,8:PRINT "***  UP - DOWN -
LEFT - RIGHT  ***"
66 FOR PAUSE=1 TO 1000:NEXT PAUSE
80 DIM A$(1)
90 DIM B(4):B(1)=45:B(2)=61:B(3)=43:B(4)
=42
100 OPEN #1,4,0,"K:"
110 FOR D=1 TO 4
111 GRAPHICS 2:POKE 752,1
112 POKE 756,226
115 SETCOLOR 0,0,0
120 PRINT "      *** WHICH DIRECTION? **
*"
130 ON D GOSUB 3010,3510,4010,4510
150 GET #1,K
160 IF K=B(D) THEN GRAPHICS 0:POKE 752,1
:GOSUB 1010:GOTO 200
170 GRAPHICS 0:POKE 752,1:GOSUB 2010:GOT
O 111
200 NEXT D
205 CLOSE #1
210 GRAPHICS 0:POKE 752,1
220 POSITION 15,8:PRINT "AGAIN";:INPUT A
$

```

```

230 IF A$="Y" THEN 100
240 IF A$<>"N" THEN 210
250 GRAPHICS 0:POKE 752,0:END
3000 REM *** UP SUBROUTINE
» 3010 POSITION 8,3:PRINT #6;"u p"
3020 POSITION 8,5:PRINT #6;CHR$(27);CHR$
(28)
3030 POSITION 8,5:PRINT #6;CHR$(32)
3040 RETURN
3500 REM *** DOWN SUBROUTINE
» 3510 POSITION 8,5:PRINT #6;"down"
3520 POSITION 8,3:PRINT #6;CHR$(27);CHR$
(29)
3530 POSITION 8,3:PRINT #6;CHR$(32)
3540 RETURN
4000 REM *** LEFT SUBROUTINE
» 4010 POSITION 5,5:PRINT #6;"left ";CHR$
(27);CHR$(30)
4020 POSITION 11,5:PRINT #6;CHR$(32)
4030 RETURN
4500 REM *** RIGHT SUBROUTINE
» 4510 POSITION 5,5:PRINT #6;CHR$(27);CHR$
(31);" right"
4520 POSITION 5,5:PRINT #6;CHR$(32)
4530 RETURN

```

## Typing Hints ...

» You get the two different colors on the screen by entering the direction names in reverse video. To turn on the reverse video, just press the Atari-symbol key (⌘). To turn it off, press the key again.

## Highlights ...

This program uses the alternate character set in Graphics Mode 2. The alternate character set replaces the upper-case letters and numbers with the lower-case letters and the Atari graphics symbols. You get this character set with the POKE 756,226 command on line 112. You also need to use the SETCOLOR 0,0,0 command on line 115. Otherwise, you will

have little hearts all over the screen.

All four directions run inside of the loop on lines 110 to 200. The “up” screen is printed by the subroutine beginning on line 3000. The “down” screen is printed by the subroutine beginning on line 3500. The “left” screen is printed by the subroutine beginning on line 4000. The “right” screen is printed by the subroutine beginning on line 4500.

Now look at lines 3020, 3520, 4020, and 4520. These lines print the different arrows. You get the arrows to appear on the screen by first printing an escape character—CHR\$(27). Since you don’t want the escape character to stay on the screen, you immediately erase it on lines 3030, 3530, 4030, and 4530.

The arrow keys are a combination of the **CTRL** button and the - key (Atari ASCII code 28) for up, the = key (Atari ASCII code 29) for down, the + key (Atari ASCII code 30) for left, and the \* key (Atari ASCII code 31) for right. But you don’t have to hold down the **CTRL** key. That’s because the program automatically translates a - to an up arrow, a = to a down arrow, a + to a left arrow, and a \* to a right arrow. To do this, it uses the B array. (See lines 90 and 160.)

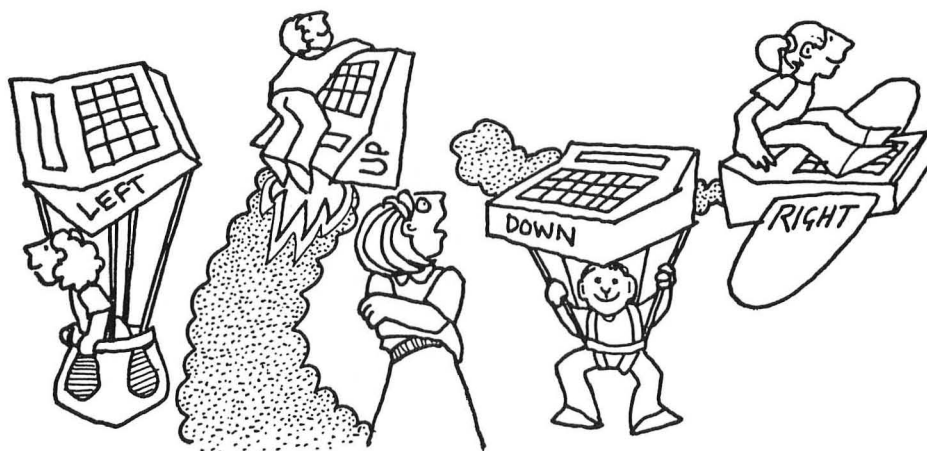
## Variables ...

<b>PAUSE</b>	Delay-loop counter.
<b>A\$</b>	Accepts your answer to question “PLAY AGAIN?”
<b>B</b>	Array—Contains values for arrow keys (values are for arrow buttons alone, not arrow buttons plus the <b>CTRL</b> button).
<b>D</b>	Direction-loop counter (for 4 directions).
<b>K</b>	Accepts arrow value.

## Do-It-Yourself ...

There are lots of ways to spice up this program. But before you start, turn to the next chapter.

# UP, UP AND AWAY!



## For Parents and Teachers ...

Like the game in the last chapter, this game helps children learn directions—up, down, left, and right—and direction names.

This game is an expanded version of the up-down game from the last chapter. It shows what happens when you add animation and sound effects to the game.

## For Kids ...

This game is a lot like the game in the last chapter. Only this time the arrows move and play funny music.

To play this game, just search for the arrows on the computer keyboard. Press the arrow that matches the arrow on the TV screen. Remember to shout out the directions: “UP!” “DOWN!” “LEFT!” and “RIGHT!”

## The Game ...

Program Name: **UPDOWN2**


```

0 GRAPHICS 0
50 REM *** ANIMATED UP/DOWN GAME
60 PRINT " \ ":POKE 752,1
65 POSITION 3,8:PRINT "***  UP - DOWN -
LEFT - RIGHT  ***"
66 FOR PAUSE=1 TO 1000:NEXT PAUSE
80 DIM A$(1)
90 DIM B(4):B(1)=45:B(2)=61:B(3)=43:B(4)
=42
100 OPEN #1,4,0,"K:"
110 FOR D=1 TO 4
111 GRAPHICS 2:POKE 752,1
112 POKE 756,226
115 SETCOLOR 0,0,0
120 PRINT "      *** WHICH DIRECTION? **
*"
130 ON D GOSUB 3010,3510,4010,4510
150 GET #1,K
160 IF K=B(D) THEN GRAPHICS 0:POKE 752,1
:GOSUB 1010:GOTO 200
170 GRAPHICS 0:POKE 752,1:GOSUB 2010:GOT
O 111
200 NEXT D
205 CLOSE #1
210 GRAPHICS 0:POKE 752,1
220 POSITION 15,8:PRINT "AGAIN";:INPUT A
$
230 IF A$="Y" THEN 100
240 IF A$<>"N" THEN 210
250 GRAPHICS 0:POKE 752,0:END
➤ 3000 REM *** UP SUBROUTINE
3010 POSITION 8,1:PRINT #6;"u p"
3012 FOR L=1 TO 3
3013 FOR J=10 TO 2 STEP -1
3015 SOUND 0,J*25,10,15
3020 POSITION 9,J:PRINT #6;CHR$(27);:POK
E 85,9:PRINT #6;CHR$(28)
3035 FOR PAUSE=1 TO 20:NEXT PAUSE
3036 IF J=2 AND L=3 THEN 3038
3037 POSITION 9,J:PRINT #6;" "
```



```
3038 NEXT J
3039 NEXT L
3040 SOUND 0,0,0,0
3050 RETURN
3500 REM *** DOWN SUBROUTINE
➤ 3510 POSITION 8,9:PRINT #6;"down"
3512 FOR L=1 TO 3
3513 FOR J=0 TO 8
3515 SOUND 0,J*25,10,15
3520 POSITION 9,J:PRINT #6;CHR$(27);:POK
E 85,9:PRINT #6;CHR$(29)
3535 FOR PAUSE=1 TO 20:NEXT PAUSE
3536 IF J=8 AND L=3 THEN 3538
3537 POSITION 9,J:PRINT #6;" "
3538 NEXT J
3539 NEXT L
3540 SOUND 0,0,0,0
3550 RETURN
4000 REM *** LEFT SUBROUTINE
➤ 4010 POSITION 0,5:PRINT #6;"left"
4012 FOR L=1 TO 3
4013 FOR J=19 TO 4 STEP -1
4015 SOUND 0,INT(250/J),10,15
4020 POSITION J,5:PRINT #6;CHR$(27);:POS
ITION J,5:PRINT #6;CHR$(30)
4035 FOR PAUSE=1 TO 20:NEXT PAUSE
4036 IF J=4 AND L=3 THEN 4038
4037 POSITION J,5:PRINT #6;" "
4038 NEXT J
4039 NEXT L
4040 SOUND 0,0,0,0
4050 RETURN
4500 REM *** RIGHT SUBROUTINE
➤ 4510 POSITION 15,5:PRINT #6;"right"
4512 FOR L=1 TO 3
4513 FOR J=0 TO 14
4515 SOUND 0,INT(250/(J+1)),10,15
4520 POSITION J,5:PRINT #6;CHR$(27);:POK
E 85,J:PRINT #6;CHR$(31)
4535 FOR PAUSE=1 TO 20:NEXT PAUSE
4536 IF J=14 AND L=3 THEN 4538
4537 POSITION J,5:PRINT #6;" "
4538 NEXT J
4539 NEXT L
4540 SOUND 0,0,0,0
4550 RETURN
```

## Typing Hints ...

» You get the two different colors on the screen by entering the direction names in reverse video. To turn on the reverse video, just press the Atari-symbol key () . To turn it off, press the key again.

## Highlights ...

This program is longer than the other up-down program, but it is mostly the same. Only the subroutines are larger.

Each subroutine now has two loops. The outer (L) loop sets the number of times the arrows will move across the screen. The inner (J) loop controls the pitch of the SOUND command and the location of the arrow on the screen.

Each time an arrow is drawn, an escape symbol must appear first and then be immediately erased. In three of the subroutines, this is done with a POKE 85,9 or POKE 85,J command (see lines 3020, 3520, and 4520). Stored in memory location 85 is the current-column position for the cursor. The program prints the escape character—PRINT #6;CHR\$(27)—then backspaces one space with the POKE command. Then it prints the arrow right on top of the escape character.

## Variables ...

J	Loop counter for moving-arrow/SOUND loop.
L	Loop counter to set number of times arrow moves across the screen.

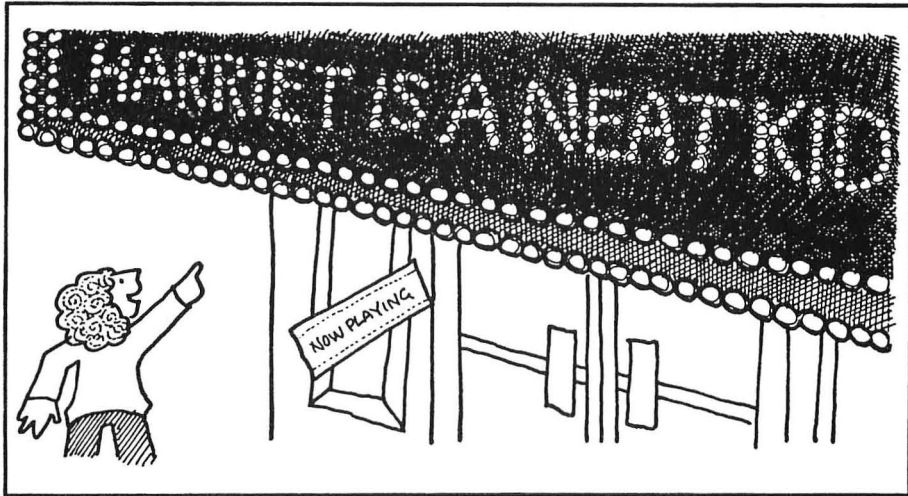
Other variables are the same as in the last program.

## **Do-It-Yourself ...**

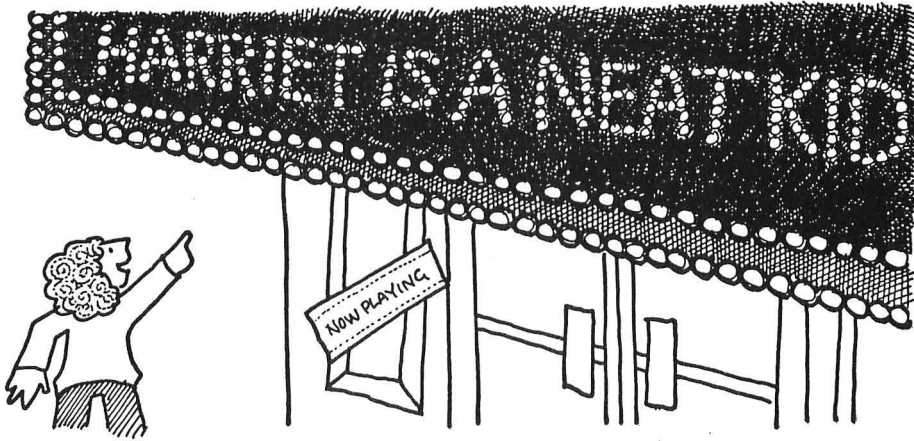
The program lets children enter directions only by pressing an arrow button. It would be nice if the program would also accept direction names: LEFT, RIGHT, UP, and DOWN.

---

# IMAGINATION



# NEWS BULLETIN!



## For Parents and Teachers ...

This is an imagination game. Children pretend the computer is a giant marquee in a big city. They get to think up any message they want to put on that marquee.

This game will encourage children to compose unusual sentences and messages. It will help them develop their writing and typing skills.

## For Kids ...

How would you like to see your name on TV? Or in the movies? How would it feel to go to a theater and see your name up there in lights along with famous movie stars? Wouldn't it be great to be mentioned on a "World News" program some night?

Or, have you ever thought about making up your own newspaper headlines? They could be serious, or they could be crazy, weird, or funny.

You can make up your own headlines. And you can be in them, too! How? On your computer—using the News Bulletin game.

When you load and run the News Bulletin game, it asks you to enter your news bulletin. Type in whatever you think up. The game turns your message into large headline letters that march across the TV screen. Each letter makes its own special musical note.

Now turn out the lights in your room. Sit down on the floor beneath the glowing computer TV screen. Pretend it is nighttime and you are in a huge city. Look up. Your bulletin appears, and millions of people can see it!

## The Game ...

Program Name: **BULLETIN**

```
50 REM *** NEWS BULLETIN
60 PRINT " \ ":POKE 752,1
65 POSITION 8,8:PRINT "*** NEWS BULLETIN
! ***"
67 FOR PAUSE=1 TO 1000:NEXT PAUSE
70 DIM M$(180)
80 DIM MUSIC(90)
90 LOW=48:HIGH=57:GOSUB 3010
92 LOW=65:HIGH=90:GOSUB 3010
93 READ M:MUSIC(32)=M
100 GRAPHICS 0:POKE 752,1
110 PRINT "YOUR NEWS BULLETIN":INPUT M$
115 IF M$="" THEN GRAPHICS 0:POKE 752,0:
END
120 GRAPHICS 2:POKE 752,1:C=0
130 PRINT " *** YOUR NEWS BULLETIN!! **
*"
135 TOP=18
137 POSITION 1,0:PRINT #6;"*** BULLETIN
***"
```

```
140 FOR Y=2 TO 11
142 TOP=18
144 IF LEN(M$)<=C+18 THEN 150
146 IF M$(C+18,C+18)<>" " THEN GOSUB 351
0
150 FOR X=1 TO TOP
160 C=C+1
165 IF C<=LEN(M$) THEN 170
167 FOR PAUSE=1 TO 500:NEXT PAUSE
168 POP :GOTO 120
170 POSITION X,Y:PRINT #6;M$(C,C)
171 T=ASC(M$(C,C))
172 IF T=32 THEN 180
173 IF T>=48 AND T<=57 THEN 180
174 IF T>=65 AND T<=90 THEN 180
175 SOUND 0,MUSIC(32),10,15
176 GOTO 190
180 SOUND 0,MUSIC(ASC(M$(C,C))),10,15
190 FOR PAUSE=1 TO 30:NEXT PAUSE
200 SOUND 0,0,0,0
210 NEXT X
250 NEXT Y
260 GOTO 100
3000 REM *** LOAD MUSICAL NOTES
3010 FOR I=LOW TO HIGH
3020 READ M
3030 MUSIC(I)=M
3040 NEXT I
3050 RETURN
3500 REM *** SHRINK LINE SUBROUTINE
3510 FOR TOP=17 TO 5 STEP -1
3520 IF M$(C+TOP,C+TOP)=" " THEN POP :GO
TO 3550
3530 NEXT TOP
3550 RETURN
5000 DATA 29,31,33,35,37,40,42,45,47,50,
53,57,60,64,68
5010 DATA 72,76,81,85,91,96,102,108,114,
121,128,136,144,153,162,173,182,193
5020 DATA 204,217,230,243
```

## Highlights ...

The Bulletin game has a musical tone associated with each letter in the alphabet and with the numbers 0 through 9. These tones are loaded into the MUSIC array by the subroutine beginning on line 3000. This subroutine is called right at the beginning of the program.

The bulletin screen appears in Graphics Mode 2. Each line can be 18 letters or numbers long. There can be up to 10 lines in your bulletin.

The computer looks ahead to the end of each new line. If a word is too big to fit on the line, the computer bumps that word to the next line. On the current line it prints only the words preceding that word in the bulletin. (See the subroutine beginning on line 3500, and see lines 144 and 146.)

The computer has a musical tone associated with each letter or number. If the computer encounters a space or punctuation symbol, it prints it on the screen then plays a special tone.

## Variables ...

<b>PAUSE</b>	Delay-loop counter.
<b>M\$</b>	Your Bulletin.
<b>MUSIC</b>	Array—stores the musical tones associated with each letter and number.
<b>LOW</b>	Low value in MUSIC where tones are stored.
<b>HIGH</b>	High value in MUSIC where tones are stored.
<b>M</b>	A single musical tone read from DATA statement.
<b>C</b>	Points to current letter or number in bulletin string (M\$) that is about to be printed.
<b>X</b>	Column position for current letter or number.



<b>Y</b>	Row position for current line.
<b>TOP</b>	Final column position for letter or number on the current line on the TV screen.
<b>T</b>	The Atari ASCII value of the current letter or number in the bulletin.
<b>I</b>	Loop counter for reading tones into MUSIC.

## **Do-It-Yourself ...**

Once you start the News Bulletin program, it keeps going. This is so you won't have to tend to it. You can leave a message for a member of your family and leave town. The bulletin program will display that message even if your family member arrives hours later.

You can change the program so that it stops after it repeats the bulletin a certain number of times. You can have it jump back to line 100. Then, if you want to exit the program, you just press the **RETURN** button when the computer asks you for a new bulletin. (See line 115.)

You might also change the way the letters march across the TV screen. Real electronic banners set whole words in motion. The words sail across the giant screen from right to left, like a fleet of ships in a regatta. You can do that on your computer. You need to pull letters and words from M\$ and start them out on the right side of the screen. Then march them all together over to the left side of the screen. When all the words have marched across the screen, go back to the beginning of M\$ and start again.



# The Atari® Playground

Fred D'Ignazio

A simple, entertaining introduction to the world of the Atari for youngsters. Participate in a Spelling Bee, draw with a Computer Crayon, chase wild letters, watch ghosts appear and disappear, and play games against the Atari. Twenty-three exciting short stories and original programs teach children word and number skills in an easygoing, friendly style that children love.

Contains instructions for using the special Atari graphics keys. Suggestions in each chapter explain how to modify the programs to make the adventures even more exciting. The programs are listed in the book and can easily be typed into the Atari. A cassette tape is also available containing all the programs ready to run.

*More adventures by Fred D'Ignazio ...*

## **ATARI® IN WONDERLAND**

Written in the same exciting style and format as *The Atari Playground*, *Atari in Wonderland* guides youngsters on a series of new adventures that will enhance their word and number skills. The twenty-two stories and programs include arithmetic games, a roller coaster ride that teaches angle measure, a Quiz Show, a reflex test, a program to create songs, and much more. #5771-7, paper, 150 pages.



HBJ BOOKS COMPANY, INC.  
Hightstown, New Jersey

ISBN 0-8104-5770-9