TI™ IN Wonderland

HAYDEN

TI™ in Wonderland

Fred D'Ignazio

HAYDEN BOOK COMPANY
a division of Hayden Publishing Company, Inc.
Hasbrouck Heights, 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.

Acquisitions Editor: LES PURIFICACIÓN
Production Editor: LORI WILLIAMS
Developmental Editor: KAREN PASTUZYN
Conversion by: BRUCE MITCHELL
Book Design: 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 GRAPHICS CO.:
FAIRFIELD GRAPHICS DIVISION

Library of Congress Cataloging in Publication Data

D'Ignazio, Fred.
TI in wonderland.

1. Computer games—Juvenile literature. 2. TI-99/4A (Computer)—Programming—Juvenile literature. 3. BASIC (Computer program language)—Juvenile literature.
I. Title.
GV1469.2.D545 1984 794.8'2 84-4493
ISBN 0-8104-6415-2

1 2 3 4 5 6 7 8 9 PRINTING

84 85 86 87 88 89 90 91 92 YEAR

PREFACE

This is a book of learning games you can type in on your TI-99/4A computer. You and your children can use this book to learn with the computer and about the computer. And while you are learning you are going to have fun.

If your children need to practice their spelling, have them play Scrambled Bees. If they are learning about fractions, give them some practice dividing up a pizza pie in Pepperoni, Please!

There are also programs on phonetics (Fat Cat), colors and sounds (The Pied Piper), multiplication (The Hamburger Contest), learning state names (Al's Tour of the States), Spanish (Uno! Dos! Tres!), and French (Un! Deux! Trois!).

There is even a child-size word processor program called Book Report. Children can use it to write poems, stories, and, of course, book reports.

There are twenty-one learning games in this book. The games are grouped by subject area. They are educational, simple, and designed to engage your children's imaginations.

Most of the games are short. That means they are easy to enter into the computer. You can learn a lot about programming in BASIC by reading the description of each game and typing it in.

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

Next is a **For Kids** section that weaves a story around each game and encourages children to use their imaginations when they play the game.

The Program comes next. It is the listing of the commands in the game. Following the program listing (in several chapters) is a **Typing Hints** section that explains how to type special parts of the program into the computer. Then comes a **Highlights** section that points out the major sections of the program. This is followed by a **Variables** section that lists all the variables (the pigeonholes in the computer's memory that store important numbers and letters). The chapter ends with a **Do-It-Yourself** section that gives you suggestions for expanding or enhancing the original game.

BASIC statements and commands in this book are written in capital letters: for example, FOR, NEXT, SAVE, CALL, and END. References to buttons on the keyboard are written in bold capital letters: for example, **CTRL**, **FCTN**, and **ENTER**.

Many of the games use graphics (picture-making) characters. These characters are defined by the CALL CHAR statement on the TI. You may reference the two manuals supplied with your TI-99/4A, namely, *Beginner's BASIC* and *User's Reference Guide* for a detailed description of the CALL CHAR statement.

Thanks to My Young Assistants ...

Young people designed several of the learning games in this book. My editor at Hayden, Gary Markman, suggested that I find some high school students to help me with the programming. I spoke to David James, the computer science instructor at Patrick Henry High School, which is only two blocks from my home, in Roanoke, Virginia. Within a few days David and his students were designing programs for this book.

The students came to school early and left late. They even got permission to leave other classes in order to work on the original programs for this book. David spent several weeks carrying their one computer back and forth between the school and the students' homes, so the students could work on the programs in the evenings and on weekends. At one point, all the programs mysteriously disappeared, probably due to a defective program recorder.

But, somehow, the programs all got done. To celebrate the completion of the programs and to thank the students, I took David and his team of young programmers out for a pizza dinner, compliments of Hayden.

According to David, this book is just the start for his students. Now they are anxious to begin working on new books and maybe even start their own "Learning Games" software company.

Make These Programs Your Own ...

These programs are intentionally short and simple. That makes them easy to type in, easy to understand, and easy to use. It also makes them easy to modify. As you are typing them in, make these programs your own. For example, add comment lines (REM commands) **in your own words** that explain what each program does. Also, at the beginning of each game, add PRINT commands to explain the game rules to your children.

I hope you and your children have as much fun using this book as I had writing it!

Fred D'Ignazio

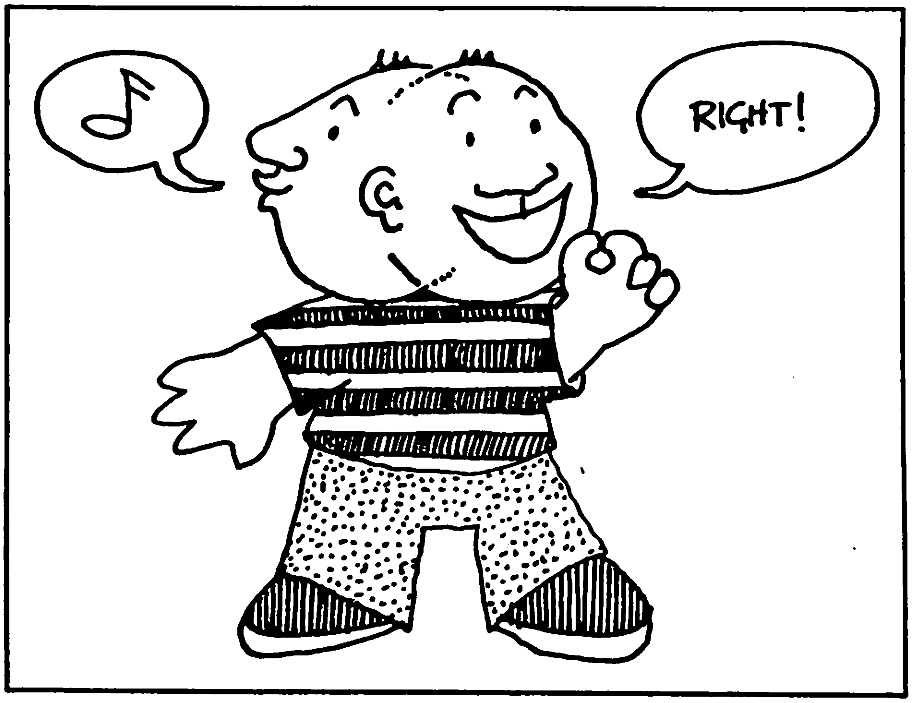
EQUIPMENT NEEDED

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

- A Texas Instruments TI-99/4A with 16K RAM (minimum).
- A black and white or, preferably, a color monitor (TV).
- A data cassette tape recorder or a disk drive system.

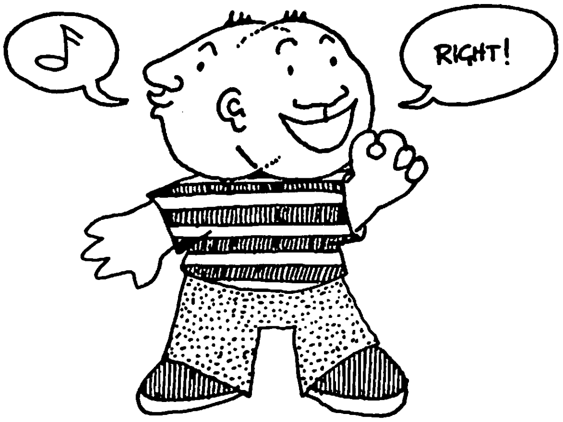
CONTENTS

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 ***
1010 CALL CHAR(129,"071F3F7F73F3FFFF")
1020 CALL CHAR(130,"E0F8FCFECECFFFFF")
1030 CALL CHAR(131,"FFFFEF777B3C1F07")
1040 CALL CHAR(132,"FFFFF7EEDE3CF8E0")
1050 CALL HCHAR(20,15,129)
1060 CALL HCHAR(20,16,130)
1070 CALL HCHAR(21,15,131)
1080 CALL HCHAR(21,16,132)
1090 A$="right!"
1100 FOR Q=1 TO 6
1110 CALL HCHAR(22,12+Q,ASC(SEG$(A$,Q,1)))
1120 NEXT Q
1130 FOR W=1 TO 3
1140 FOR Q=600 TO 980 STEP 50
1150 CALL SOUND(100,Q,10)
1160 NEXT Q
1170 NEXT W
1180 FOR Q=0 TO 2
1190 CALL HCHAR(20+Q,13,32,6)
1200 NEXT Q
1210 RETURN
```

Highlights ...

This is one of the simplest programs in the book. Lines 1010 to 1040 define the characters needed for happy face using the CALL CHAR command. Lines 1050 to 1080 print out the happy face using the CALL CHAR command. On lines 1090 to 1120, it prints the message "RIGHT!" On lines 1130 to 1170 it makes the whistle sound. Lines 1180 to 1200 erase the happy face from the display.

The program uses the sound command—CALL 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 Q = 1 TO 6 on line 1100. This tells the computer to set the loop counter Q to 1. Each time the computer races around the loop it adds one to the loop counter (Q). The computer does six loops (from Q = 1 to Q = 6). Then it goes on and does something else.

Also, note that the program starts on line 1000 and ends on line 1210 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 RUN 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 command SAVE DSK.1 HAPPY or the command SAVE CS1 if you are using a tape recorder.

Variables . . .

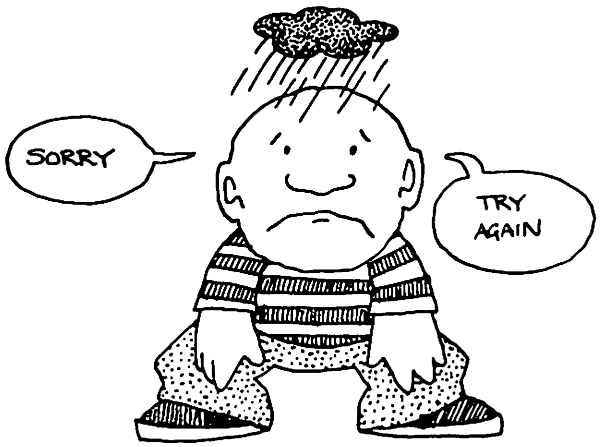
W	Whistle counter—6 whistles.
Q	Pitch counter and pitch (notes played in the whistle) and general variable.
T	Delay counter—slows down whistle.
I	Row counter—used in erasing face.

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 define new graphics characters in lines 1010 to 1040. 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 ***
2010 CALL CHAR(129,"071F3F7F73F3FFFF")
2020 CALL CHAR(130,"E0F8FCFECECFFFFF")
2030 CALL CHAR(131,"FFFFFC7B773F1F07")
2040 CALL CHAR(132,"FFFF3FDEEEFCF8E0")
2050 CALL HCHAR(20,15,129)
2060 CALL HCHAR(20,16,130)
2070 CALL HCHAR(21,15,131)
2080 CALL HCHAR(21,16,132)
2090 A$="sorry ... try again"
2100 FOR Q=1 TO 19
2110 CALL HCHAR(22,Q+6,ASC(SEG$(A$,Q,1)))
2120 NEXT Q
2130 CALL SOUND(500,311,10)
2140 CALL SOUND(1000,131,10)
2150 FOR PAUSE=1 TO 300
2160 NEXT PAUSE
2170 FOR Q=0 TO 2
2180 CALL HCHAR(20+Q,7,32,19)
2190 NEXT Q
2200 RETURN
```

Typing Hints ...

Load the Happy Face subroutine before typing in any of the Sad Face subroutine using the OLD CS1 command. After you add the Sad Face subroutine, save the combined program onto a cassette using the SAVE CS1 command. This combined routine will need to be loaded before any future program using this routine is typed.

Before you type in a program that will use these routines, follow this procedure:

First you enter (OLD) the combined Happy Face and Sad Face subroutine.

Then you type in a new program.

Last, you save the combined, new program with a SAVE CS1 command (tape) or SAVE DSK1.NAME (disk).

Highlights . . .

The Sad Face is similar to the Happy Face. Lines 2010 to 2040 define the characters. Lines 2050 to 2080 print out the sad face. Lines 2090 to 2120 print the sad message. Lines 2130 and 2140 play the sad sound. Lines 2170 to 2190 erase the sad face and the message.

To test the Sad Face, just type RUN 2010.

Note that the Sad Face starts on line 2000. The Sad Face is a subroutine, just like the Happy Face. You can attach it to the tail end of all your other programs. Be sure to follow the instructions in the preceding section, Typing Hints.

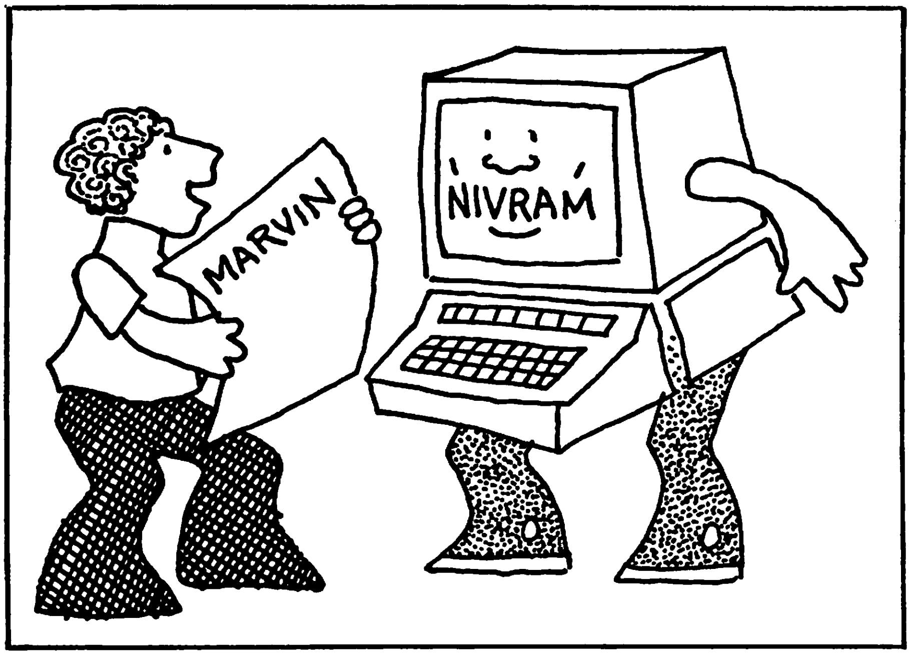
Variables . . .

PAUSE	Delay counter—computer pauses between sounds.
Q	Row counter—used in erasing face.

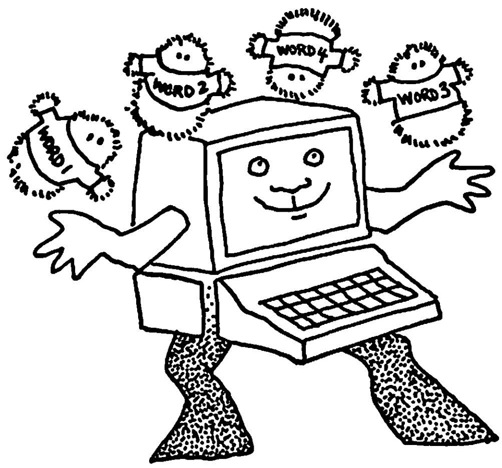
Do-It-Yourself . . .

Try this sad face for awhile. When you get tired of it, change it to a new face. All you have to do is change the redefined characters in lines 2010 to 2040. Change the eyes. Add a nose. Change the mouth. How about hair? Or a hat? And some ears?

WORDS



3 SCRAMBLED BEES



For Parents and Teachers ...

This game helps children practice spelling. You or your children enter ten words into the computer. The computer arranges the words in a random order and then 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 ...

In this chapter we create a "scrambled" spelling bee. The computer juggles the words around. Then it gives you the spelling bee with the words in the new order. And if you ask the computer to quiz you again on the same words, it will scramble the words again. Then it will give you the spelling bee.

The Game ...

Program Name: **SCRAMBEE**

```
50 REM  * SCRAMBLED SPELLING BEE *
55 RANDOMIZE
60 CALL CLEAR
70 PRINT TAB(4);"*** SCRAMBLED BEE ***"
80 FOR Q=1 TO 11
90 PRINT
100 NEXT Q
110 DIM W$(10)
115 FOR PAUSE=1 TO 1000
120 NEXT PAUSE
130 FOR SPELL=1 TO 10
140 CALL CLEAR
150 PRINT "WORD #";SPELL;
160 INPUT W$(SPELL)
170 NEXT SPELL
175 S$=" 0123456789 "
180 FOR SPELL=10 TO 1 STEP -1
190 S=INT(RND*(SPELL-1)+2)
200 P=ASC(SEG$(S$,S,1))-47
210 S$=SEG$(S$,1,S-1)&SEG$(S$,S+1,12)
220 CALL CLEAR
225 T=12-INT(LEN(W$(P))/2)
230 PRINT TAB(T);"WORD: ";W$(P)
240 FOR Q=1 TO 11
250 PRINT
260 NEXT Q
270 FOR PAUSE=1 TO 500
280 NEXT PAUSE
290 CALL CLEAR
300 INPUT "  WORD: ":A$
310 IF A$=W$(P)THEN 320
312 GOSUB 2000
314 GOTO 220
320 GOSUB 1000
330 NEXT SPELL
340 CALL CLEAR
350 INPUT "SAME WORDS AGAIN? ":A$
360 IF SEG$(A$,1,1)="Y" THEN 175
370 IF SEG$(A$,1,1)<>"N" THEN 340
380 GOTO 130
390 END
```

Typing Hints ...

Remember to load the Happy-Sad routine before typing in this program.

Highlights ...

In this program, the computer scrambles the words to be spelled. It chooses the sequence on line 190.

On line 200, the computer uses the following function to pick each of the 10 spelling-bee words:

```
ASC(SEG$(S$,S,1))-47
```

We interpret this function (as the computer does) from the inside out. Let's say that this is the first time through the FOR-NEXT loop (lines 180 to 330). SPELL is the loop counter, so SPELL is equal to 10.

Next, we know the variable S was chosen on line 190. It is a random number between 2 and 11. Let's say that S equals 3.

If we plug in the value of S, we get the function:

```
ASC(SEG$(S$,3,1))-47
```

We know that SEG(S$,3,1) selects the 3rd character in the string S$. The 3rd character in S$ is a "1".

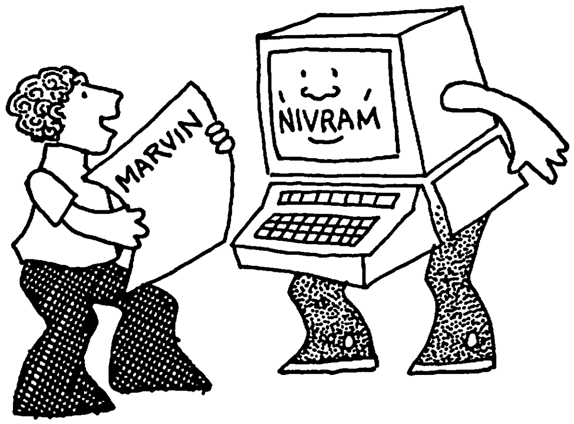
If we plug in the 1 we get the function:ASC("1")-47). The ASC function converts a number in string (nonarithmetic) form into ASCII value. The true number is then obtained by subtracting 47 from the ASCII value. The result of ASC("1")-47 is the number 1.

Now we are ready to pick our spelling-bee word. Look at line 230. The value stored in P tells the computer that W$(P) is the selected word.

Voila!

Note that on line 330, we now send the computer back to line 180. Line 180 is where the routine for juggling the words and choosing a new scrambled word starts.

4
BACKWORDS



For Parents and Teachers . . .

This game is mostly for fun. But it teaches children to look at words in a playful, original way. It shows them that words aren't fixed and frozen. They can be flipped over and turned around. The result is amusing and often surprising.

For Kids . . .

Did you ever wonder what your name looks like spelled backwards? I did. So I tried this computer game.

When the computer typed "FRONTWORDS?" I typed my name—FRED.

A message flashed on the TV screen:

```
BACKWORDS MACHINE NOW WORKING
```

Moments later, the computer printed my name forwards and backwards. Forwards it was FRED. Backwards it was DERF!

What does **your** name look like backwards? Or your mom's name? Or your dad's? Or your favorite hero's name?

Use this game to find out.

The Game ...

Program Name: **BACKWORD**

```
50 REM  ** BACKWORDS GAME **
60 CALL CLEAR
70 PRINT " *** THE BACKWORDS GAME ***"
80 FOR Q=1 TO 11
90 PRINT
100 NEXT Q
105 FOR PAUSE=1 TO 1000
110 NEXT PAUSE
115 CALL CLEAR
120 PRINT "FRONTWORDS:"
125 INPUT M$
130 IF M$="" THEN 115
135 CALL CLEAR
140 PRINT " BACKWORDS MACHINE WORKING"
142 FOR Q=1 TO 11
144 PRINT
146 NEXT Q
150 FOR PAUSE=1 TO 1000
160 NEXT PAUSE
170 M2$=""
180 FOR I=LEN(M$)TO 1 STEP -1
190 M2$=M2$&SEG$(M$,I,1)
200 NEXT I
210 CALL CLEAR
220 PRINT "FRONTWORDS:"
230 PRINT
240 PRINT M$
250 PRINT
260 PRINT
270 PRINT
```

```
280 PRINT "BACKWORDS:"
290 PRINT
300 PRINT M2$
310 PRINT
320 PRINT
330 PRINT
340 INPUT "PLAY AGAIN?":A$
350 IF SEG$(A$,1,1)="Y" THEN 115
360 IF SEG$(A$,1,1)<>"N" THEN 340
370 CALL CLEAR
380 END
```

Highlights ...

The key to this program is the loop on lines 180 to 200. The name you typed in is stored in M$. The computer takes the last letter of the name in M$ and places a copy of it in the **first** position in M2$. Then it takes the next-to-the-last letter in M$, copies it, and then places it in the **second** position in M2$. The computer continues this process until it has reached the very first letter in M$. It puts this at the **end** of the letters it has stored in M2$. In this way, the computer turns the entire word around. What went frontwards into M$ comes out backwards into M2$.

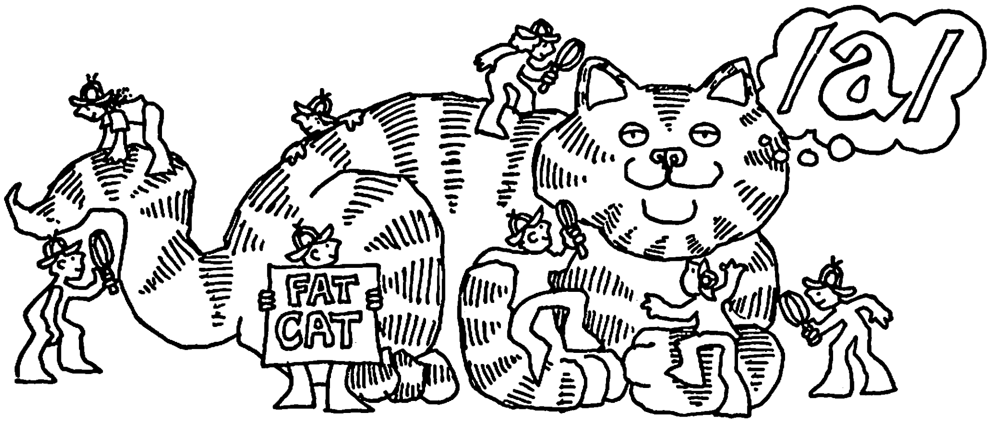
Do-It-Yourself ...

You might modify this program so that you can store the "backwords" on tape or disk.

Also, "backwords" are just an example of the limitless ways you can play around with words on your computer. Electronic words aren't fixed or frozen. They are plastic or clay—infinitely malleable.

One way to get your computer to play with words is to translate all the words from upper-case letters into lower-case letters. For example, the upper-case letters in the alphabet (A to Z) are represented by ASCII values of 65 to 90. The lower-case letters are represented by 97 to 122. You can change the Backwords program so that every time you type in a letter A, for example, the program adds 32 to the ASCII value and changes an "A" to an "a." The same goes for all the other letters. This way, children can get their names typed out in lower-case letters.

5
FAT CAT



For Parents and Teachers ...

This game helps children learn phonetic symbols and phonetic sounds. The computer displays the phonetic symbols on the TV screen. Then it flashes a word on the screen. The child says the word aloud and tries to guess which phonetic sound is hidden inside the word. An arrow points to the first phonetic symbol. To move the arrow to a new symbol, the child presses the **SPACE** bar. To select a symbol the child presses the **ENTER** button.

For Kids ...

Did you ever notice how certain words rhyme or sound alike? Words like **red** and **bed**, or **big** and **pig**, or **fat** and **cat**?

The reason the words rhyme is that they have the same phonetic sound (or **phoneme**) hidden inside them. For example, hiding inside the words red and bed is the phoneme /e/. Hidden inside the words big and pig is the phoneme /i/. And inside the words fat and cat is the phoneme /a/.

In this game you get to play detective. The computer will flash a word on your TV screen. Say the word out loud a couple of times. Now look at the list of phonemes on the screen. Your job as a sound detective is to discover which phoneme is hidden inside the word.

The Game ...

Program Name: **FATCAT**

```
50 REM  *** FATCAT GAME ***
55 RANDOMIZE
60 CALL CHAR(136,"7C10107C10101000")
70 CALL CHAR(137,"0008047E04080000")
80 CALL CLEAR
90 PRINT "  *** THE FAT CAT GAME ***"
100 FOR Q=1 TO 11
110 PRINT
120 NEXT Q
130 FOR PAUSE=1 TO 900
140 NEXT PAUSE
150 CALL CLEAR
160 REM  ** SET UP SCREEN **
170 PRINT "   /a/      /sh/"
180 PRINT
190 PRINT "   /ch/     /s/"
200 PRINT
210 PRINT "   /e/      /th/"
220 PRINT
230 PRINT "   /f/      /Ŧ/"
240 PRINT
250 PRINT "   /l/"
260 FOR Q=1 TO 10
270 PRINT
280 NEXT Q
285 PRINT "  ENTER=YES     SPACE=NO";
290 FOR M=1 TO 5
```

```
300 TRY=0
310 REM  ** SELECT WORD **
320 IF TRY=0 THEN 4000
330 REM  ** PRINT WORD **
340 FOR L=1 TO LEN(W$)
350 CALL HCHAR(9,20+L,ASC(SEG$(W$,L,1)))
360 NEXT L
370 C=1
380 FOR X=4 TO 13 STEP 9
390 FOR Y=5 TO 13 STEP 2
395 IF C=10 THEN 470
400 CALL HCHAR(Y,X,137)
410 CALL KEY(0,ANS,S)
420 IF S=0 THEN 410
430 IF ANS=32 THEN 450
440 IF ANS=13 THEN 500
450 CALL HCHAR(Y,X,32)
460 C=C+1
470 NEXT Y
480 NEXT X
490 GOTO 370
500 IF PH=C THEN 590
510 GOSUB 2000
520 TRY=1
530 GOTO 400
590 GOSUB 1000
600 CALL HCHAR(Y,X,32)
610 X=14
620 Y=14
625 CALL HCHAR(9,21,32,10)
630 NEXT M
640 CALL CLEAR
650 INPUT "DO YOU WANT TO PLAY AGAIN":A$
660 IF SEG$(A$,1,1)="Y" THEN 150
670 IF SEG$(A$,1,1)<>"N" THEN 650
680 CALL CLEAR
690 END
4000 REM  ** SELECT WORD **
4010 RESTORE
4020 PH=INT(RND*9+1)
4030 WD=INT(RND*5+1)
4040 FOR L=1 TO (PH-1)*5+WD
4050 READ W$
4060 NEXT L
4070 GOTO 330
```

```
4100 DATA FAT,CAT,MAN,HAT,FAST
4110 DATA RICH,WITCH,LUNCH,CATCH,MUCH
4120 DATA RED,HEN,BED,LEG,PET
4130 DATA STUFF,OFF,CUFF,MUFF,HUFF
4140 DATA BELL,STILL,WILL,ROLL,SELL
4150 DATA SHIP,FISH,SPLASH,WISH,DISH
4160 DATA GRASS,GLASS,CROSS,DRESS,MESS
4170 DATA THIN,BATH,CLOTH,WITH,THICK
4180 DATA THAT,THIS,THEN,THAN,THEM
```

Typing Hints ...

To obtain the correct character in line 230, press the **CTRL** and the **H** keys at the same time. When you do this, you will see either a blank space or a strange-looking character. Don't worry. Complete typing the line. When the program is RUN, the correct character will appear.

To get the lower-case characters, release the **ALPHA/ LOCK** key. To get back to upper-case characters, depress the ALPHA/LOCK key.

Remember to load the Happy-Sad routine before typing in this program.

Highlights ...

This game uses four major subroutines. The first routine (starting on line 170) prints the list of phonetic symbols. The second routine (starting on line 320) selects the word with the mystery phoneme hidden inside. The third routine (starting on line 340) prints the word on the TV screen. The fourth routine (starting on line 370) moves the "choice" arrow each time you press the **SPACE** bar.

Variables . . .

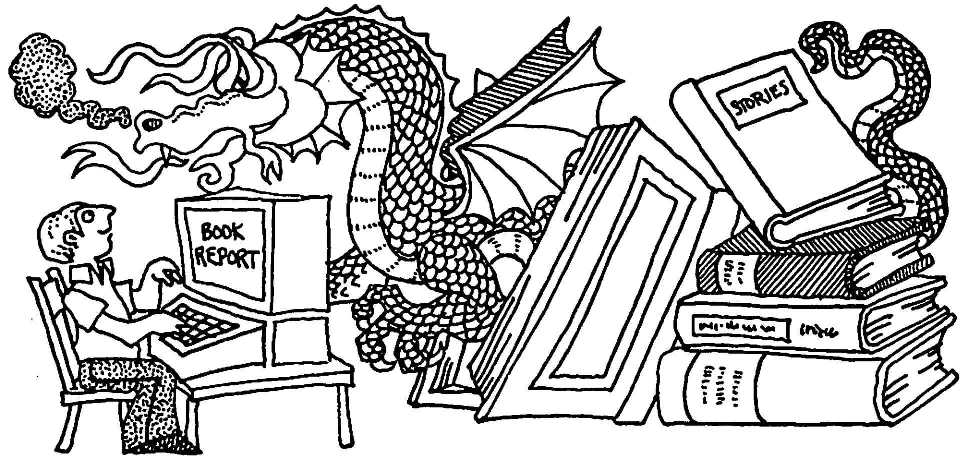
PAUSE	Delay loop counter.
A$	For child's answer to "PLAY AGAIN?"
W$	Word with mystery phoneme.
M	Loop counter—controls number of words/game.
TRY	If TRY=1, child has given an incorrect answer.
PH	Pointer to list of words containing a certain phoneme.
WD	Pointer to one word in the list.
C	Pointer to phoneme symbol (child's answer).
X	Current column position of arrow.
Y	Current row position of arrow.
L	Loop counter for randomly picking word from list.

Do-It-Yourself . . .

This is only a small sample of the different phonetic symbols that represent word sounds. You can add new sounds by changing the list of sounds in the routine beginning on line 170. You can add new words with these sounds by changing the DATA commands beginning on line 4100.

Also, please note that some of the words in the current list have more than one phonetic sound hidden inside. Yet the current program only recognizes one of those sounds as a correct answer. How would you modify the program to make it recognize the other sound or sounds?

6
BOOK REPORT



For Parents and Teachers . . .

This game will help children with their writing assignments, especially book reports. The program asks the child for pertinent information—the date, the child's name, the author, and the title of the book. Then the computer asks the child "WHAT HAPPENED FIRST?" When the child enters this information, the computer asks "WHAT HAPPENED NEXT?"

The child responds to the computer's questions and enters the report, sentence by sentence. When finished, the child types END or THE END. Then the computer prints the entire book report, all nicely formatted, on the TV screen.

For Kids . . .

Did you ever have to write a book report or tell a story, and you didn't know where to start?

Here's a sure-fire way to get you going. Pretend that the computer is a person. And imagine that the person can hardly wait to hear about a neat book you just read or a story you made up inside your head.

What does the person ask you to get you started? He says, "What happened first?" So you tell him.

It sounds exciting, so he asks, "What happened next?" So you tell him that, too.

Each time you tell him a little bit of the story, he wants to hear more. So he keeps asking "What happened next?"

Before you know it, you've finished describing the book you read or the story you made up. When he asks you "What happened next?" you simply say "END" or "THE END."

Then, sit back. The computer hasn't forgotten a word you said. It is going to replay the entire story or book report right before your eyes.

As it types the story on the TV screen, you can be busy copying it in your notebook. Then it's ready to turn in to your teacher at school tomorrow morning.

The Game ...

Program Name: **BKREPORT**

```
50 REM * THE BOOK REPORT *
60 CALL CLEAR
70 PRINT "    *** THE BOOK REPORT ***"
80 FOR Q=1 TO 11
90 PRINT
100 NEXT Q
110 DIM PO(20)
115 DIM R$(30)
120 FOR PAUSE=1 TO 900
130 NEXT PAUSE
140 CALL CLEAR
150 PRINT "WHO'S OUT THERE"
160 INPUT NM$
170 CALL CLEAR
180 PRINT "TODAY'S DATE"
190 INPUT DT$
```

```
200 CALL CLEAR
210 PRINT "BOOK TITLE"
220 INPUT T$
230 CALL CLEAR
240 PRINT "AUTHOR"
250 INPUT A$
260 CALL CLEAR
270 R=1
275 PO(0)=0
280 PRINT "WHAT HAPPENED FIRST"
290 INPUT L$
300 R$(0)=L$&" "
310 P=1
315 CALL CLEAR
320 PRINT "WHAT HAPPENED NEXT"
330 INPUT L$
340 IF L$="" THEN 400
350 IF (L$="END")+(L$="THE END")THEN 430
360 L$=L$&" "
370 R$(P)=L$
380 P=P+1
390 GOTO 315
400 PO(R)=P
410 R=R+1
420 GOTO 315
430 CALL CLEAR
440 L$="REPORT BY "&NM$
450 Y=1
460 GOSUB 910
470 L$=DT$
480 Y=3
490 GOSUB 910
500 L$=T$
510 Y=5
520 GOSUB 910
525 L$=""
530 FOR Q=1 TO LEN(T$)
540 L$=L$&"-"
550 NEXT Q
560 Y=6
570 GOSUB 910
580 L$="BY "&A$
590 Y=8
595 GOSUB 910
600 R=0
```

```
610 LT=10
650 FOR SS=0 TO P
660 IF SS<>PO(R)THEN 690
670 R$(SS)="   "&R$(SS)
680 R=R+1
690 P1=1
700 IF P1+28>LEN(R$(SS))THEN 780
710 FOR W=P1+28 TO P1 STEP -1
720 IF SEG$(R$(SS),W,1)<>" " THEN 760
730 L$=SEG$(R$(SS),P1,W-P1)
740 P1=W+1
750 W=0
760 NEXT W
770 GOTO 790
780 L$=SEG$(R$(SS),P1,P1+27)
781 GOTO 1500
785 P1=P1+28
790 Y=LT
800 X=2
810 GOSUB 920
820 LT=LT+1
830 IF LT<>23 THEN 865
835 LT=10
840 Y=24
845 L$="PRESS SPACE TO CONTINUE"
850 GOSUB 910
860 GOSUB 1010
862 CALL HCHAR(10,1,32,480)
865 IF (P1=LEN(R$(SS)))+(P1>LEN(R$(SS)))THEN 875
870 GOTO 700
875 NEXT SS
880 Y=LT
885 L$=R$(SS)
887 GOSUB 920
888 L$="THE END"
889 Y=LT+2
890 GOSUB 910
895 GOTO 895
900 REM * PRINT A LINE *
910 X=15-INT(LEN(L$)/2)
920 FOR Q=1 TO LEN(L$)
930 CALL HCHAR(Y,X+Q,ASC(SEG$(L$,Q,1)))
940 NEXT Q
950 RETURN
1000 REM * CHECK KEY *
```

```
1010 CALL KEY(0,K,S)
1020 IF S=0 THEN 1010
1030 RETURN
1500 IF PO(R)=SS+1 THEN 785
1510 R$(SS+1)=L$&R$(SS+1)
1520 GOTO 875
```

Highlights ...

The computer copies the entire story or book report into a set of character string arrays called R$.

When you are done entering a book report or story into the computer and you type THE END (or, simply, END), the computer prints out a title, the date, your name, and the entire text.

The subroutine beginning on line 910 centers all the headings, including the book title, the author, the date, and your name.

The routine on lines 530 to 550 underlines the book title. The routine beginning on line 600 prints the text of the report or story.

If a word is too long to fit at the end of a line, the computer doesn't break the word in two. Instead, it lifts the entire word and places it on the next line.

When your child is writing the report or story and comes to the end of a paragraph, he or she should answer the computer's question ("WHAT HAPPENED NEXT?") by pressing the **ENTER** button (see line 340).

When you press the **ENTER** button without entering any other information, the computer receives a **null** character and places a pointer in variable PO. When the computer prints out the report it looks at array PO where it stored the locations of the beginning of the paragraphs. When the value stored in PO equals the string number (SS), it ends an old paragraph and creates a new, indented paragraph automatically.

If the book report or story is too big to fit on the screen, the computer stops at the bottom of the screen and asks you to "PRESS SPACE TO CONTINUE." Then it erases only the text on the screen (leaving the title information and your name intact), and fills the screen with new text. It keeps doing this until it comes to the end of the story or report. At the end, it prints the words "THE END," centered on the screen.

Notice the use of variable LT that keeps track of the position of the print line. We use the lines between 10 and 22 to print the text.

Variables ...

PAUSE	Delay loop counter.
NM$	Child's name.
T$	Book title.
A$	Author's name.
R$	The text of the story or report.
L$	A single text line.
DT$	The current date.
PO	Array—location of the beginning of paragraphs.
LT	Current text print line.
SS	Current string pointer being printed.
Y	Variable for vertical position for print routine.

Do-It-Yourself ...

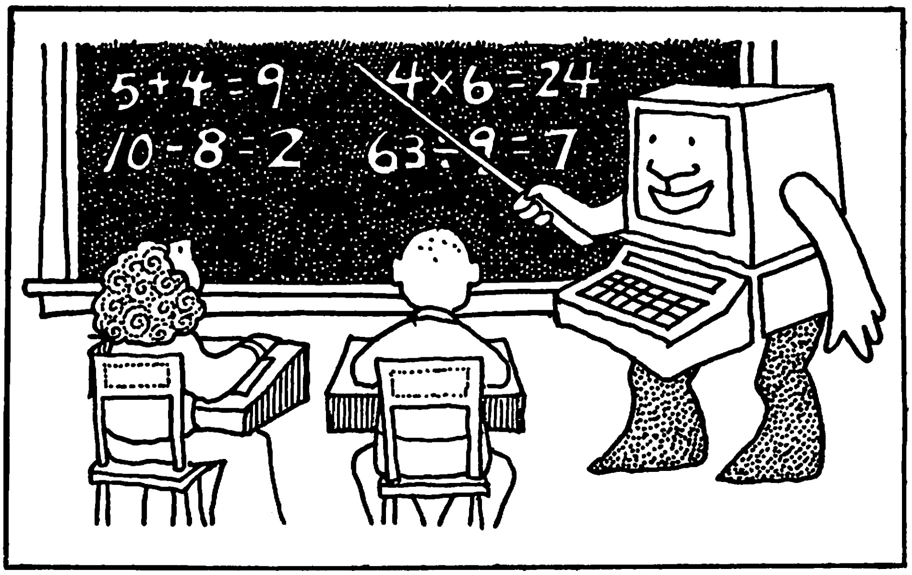
You have the rough beginnings of a word processor here. It lets you enter reports, letters, stories, or whatever you choose in an orderly, neat-looking format.

How about a subroutine that takes the report and stores it on a tape or disk? Your child might spend a long time typing in a report and might want to save sections of it at one time on tape or disk to work on later.

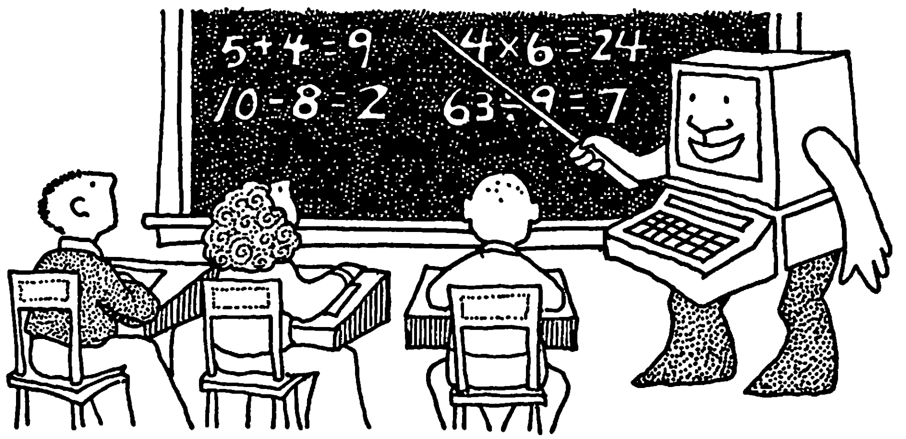
And how about printing the report? Do you have a printer? If you do, why make your child copy the whole report? Instead, add a subroutine that lets the child review the report on the TV screen and then type it out automatically on a printer.

There are a lot of things you can do with this program. With a few additions, it can turn your computer into the family's electronic typewriter.

NUMBERS



7
THE ARITHMETIC GAME



For Parents and Teachers ...

This game helps children learn addition, subtraction, multiplication, and division.

For Kids ...

When I was a child, I used to work on an arithmetic problem and then get frustrated because I didn't know if my answer was right or wrong. I wanted to know the answer right away. But I had to wait until the teacher graded my homework paper or my test. Sometimes that took days or even weeks. By the time I got my paper back, the problem was no longer fresh in my mind. If I got it right, I didn't know why. If I got it wrong, I didn't know why. And I no longer really cared.

Now you can solve arithmetic problems and learn instantly if you got them right or wrong. How? You can play the

Arithmetic Game. It's like hiring the computer as your own private arithmetic teacher.

When you RUN The Arithmetic Game, the computer displays a "menu" on the TV screen. If you press a **1**, the computer challenges you with five addition problems. If you press a **2**, it throws five subtraction problems at you. If you press a **3**, you get five multiplication problems. If you press a **4**, you get five division problems.

The computer displays the problem on the screen. Then you try to think up the answer. If you want to, get out a piece of scrap paper and a pencil. That's what I do when I'm working on arithmetic problems.

Think up your answer to the problem, and then type it in. If you want to change your answer, press the **FCTN** key and the left arrow key at the same time. When your answer is on the screen, press the **ENTER** button.

If you get the answer right, the computer draws a happy face, says "RIGHT!," and plays a happy tune.

If you get the answer wrong, the computer draws a sad face, groans, and encourages you to "TRY AGAIN!"

The Game ...

Program Name: **MATH**

```
50 REM  ** ARITHMETIC GAME **
60 CALL CLEAR
70 PRINT " *** THE ARITHMETIC GAME ***"
80 GOSUB 500
110 FOR PAUSE=1 TO 900
120 NEXT PAUSE
130 RANDOMIZE
190 CALL CLEAR
200 PRINT TAB(8);"1. ADDING"
210 PRINT
220 PRINT TAB(8);"2. SUBTRACTING"
230 PRINT
240 PRINT TAB(8);"3. MULTIPLYING"
250 PRINT
260 PRINT TAB(8);"4. DIVIDING"
```

```
270 PRINT
275 PRINT
280 INPUT "WHICH NUMBER (1,2,3 OR 4)?":B
290 IF B<1 THEN 190
300 IF B>4 THEN 190
305 FOR L=1 TO 5
310 N1=INT(RND*10+1)
320 N2=INT(RND*10+1)
330 CALL CLEAR
340 ON B GOSUB 3010,4010,5010,6010
342 A$=""
344 C=0
345 CALL KEY(0,A,S)
346 IF S=0 THEN 345
347 IF (A=8)*(C=0)THEN 345
348 IF A=8 THEN 710
354 IF A=13 THEN 360
356 A$=A$&CHR$(A)
357 CALL HCHAR(24,21+C,A)
358 C=C+1
359 GOTO 345
360 K=VAL(A$)
362 IF K=W THEN 370
365 GOSUB 2000
368 GOTO 330
370 GOSUB 1000
375 NEXT L
380 CALL CLEAR
390 INPUT "PLAY AGAIN?":A$
400 IF SEG$(A$,1,1)="Y" THEN 190
410 IF SEG$(A$,1,1)<>"N" THEN 380
420 CALL CLEAR
430 END
500 FOR Q=1 TO 11
510 PRINT
520 NEXT Q
530 RETURN
700 REM * BACKSPACE ROUTINE *
710 IF LEN(A$)>1 THEN 740
720 A$=""
730 GOTO 750
740 A$=SEG$(A$,1,LEN(A$)-1)
750 C=C-1
760 CALL HCHAR(24,21+C,32)
770 GOTO 345
```

```
3000 REM  ** ADDING **
3010 PRINT TAB(12);"ADDING"
3020 GOSUB 500
3030 PRINT TAB(8);N1;"+";N2;"= ?";
3040 W=N1+N2
3050 RETURN
4000 REM  ** SUBTRACTING **
4010 PRINT TAB(9);"SUBTRACTING"
4020 GOSUB 500
4030 IF N1>N2 THEN 4070
4040 W=N1
4050 N1=N2
4060 N2=W
4070 W=N1-N2
4080 PRINT TAB(8);N1;"-";N2;"= ?";
4090 RETURN
5000 REM  ** MULTIPLYING **
5010 PRINT TAB(9);"MULTIPLYING"
5020 GOSUB 500
5030 PRINT TAB(8);N1;"X";N2;"= ?";
5040 W=N1*N2
5060 RETURN
6000 REM  ** DIVIDING **
6010 PRINT TAB(10);"DIVIDING"
6020 GOSUB 500
6030 N3=N1*N2
6040 W=N3/N2
6050 PRINT TAB(7);N3;"/";N2;"= ?";
6060 RETURN
```

Typing Hints ...

Remember to load the Happy-Sad routine before typing in this program.

Highlights ...

The game begins (on line 200) by displaying the "menu." The menu lets you choose addition, subtraction, multiplication, or division.

The program uses a FOR-NEXT loop on lines 305 to 375 to control the number of the problems the computer "thinks up." Right now, the computer comes up with five problems per game. Then it asks if you want to "PLAY AGAIN?" If you type "Y" or "YES," the computer calls the menu. You can continue solving the same kind of problems (addition, subtraction, multiplication, or division), or you can try something new.

The RND functions on lines 310 and 320 are set to randomly choose two numbers between 1 and 10. These are the numbers added together, subtracted, multiplied, or divided.

The actual arithmetic games are a group of subroutines beginning on line 3010. The selected games will be called five times by a GOSUB command on line 340. The subroutines for addition, subtraction, multiplication, and division start on lines 3010, 4010, 5010, and 6010, respectively.

Variables . . .

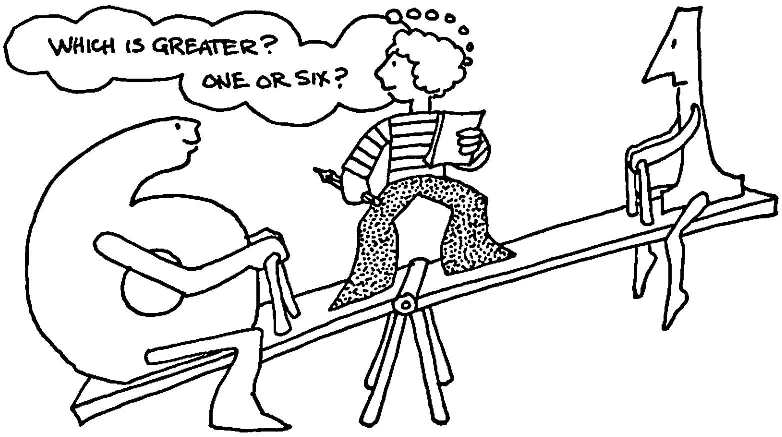
C	Horizontal position of current input.
PAUSE	Delay loop counter.
A$	Your answer to the question "PLAY AGAIN?"
K	Stores your answer to the arithmetic problem.
L	Loop counter—main loop.
N1	First random number for arithmetic problem.
N2	Second random number for arithmetic problem.
B	Which type of arithmetic problems you choose (addition, subtraction, multiplication, or division).
A	One digit of your answer.
W	The computer's answer to the arithmetic problem.
N3	Stores the dividend of the division problem.

Do-It-Yourself . . .

The program does not keep track of how many incorrect answers a child enters. It would be good to create an incorrect answer COUNT variable. Increment the variable (COUNT = COUNT + 1) each time the child makes a wrong answer. When COUNT = 3, have the computer display the correct answer on the TV screen. Then have the program ask the child the same problem again.

8

GREATER THAN WHAT?



For Parents and Teachers ...

This game helps children learn the size relationships between numbers. It shows them two numbers, and then asks them if the first number is greater than the second, less than the second, greater than or equal to the second, or less than or equal to the second.

The game helps children learn the number comparison symbols: > (greater than), < (less than), >=(greater than or equal to), and <= (less than or equal to).

For Kids ...

Numbers describe things—apples, dragons, mudpies, and worms. The bigger the number, the more things. For example, if a blue room has 1000 worms in it and a red room has 2 worms in it, which room has more worms in it? The blue room, right?

That's because a 1000 is **greater than** 2. You can compare the numbers by drawing a thousand worms and then two more worms. Or you can take a shortcut and write:

$$1000 > 2$$

The > symbol means **greater than.** A thousand is greater than two.

You can also write the numbers like this:

$$2 < 1000$$

The < symbol means **less than.** Two is less than a thousand.

But what happens if you are out exploring an enchanted forest some afternoon after school, and you discover two caves. You turn on your pocket flashlight, enter one cave, and find 14 dragons. You back out carefully, enter the second cave and find 14 more dragons. You sneak out of the second cave and run all the way home.

You call the police and the fire department. Moments later, they are at your front door. They ask you what you saw.

To describe what you just saw, you take out a piece of paper and draw 14 dragons and then 14 more dragons.

You look at the cave pictures with all the dragons. Which cave had more dragons? Neither. They both had the same number of dragons. You can draw a big = (equal sign) between the two pictures of dragons. Or you can take a shortcut and write:

$$14 = 14$$

But what if you change your mind? In the first cave (the dirty cave), you think you saw shadows of other dragons. That means the dirty cave might have more dragons than the clean cave. You know that the clean cave had only 14 dragons. But some extra dragons might have been hiding in the dirty cave, behind the dragon garbage and the piles of old bones and dirty treasure you saw.

Now you can't say how many dragons were in the dirty cave. You use the symbol X to represent this mystery number. Is there anything you can say about X?

Yes. You know X is at least 14 and maybe more. You can write this as:

X >= 14

This means that X is **greater than or equal to** 14. You can also write this as:

14 <= X

This means 14 is **less than or equal to** X (the number of dragons in the dirty cave).

The police and fire fighters are satisfied with your description. They fly in a dragon S.W.A.T. team from China. The team comes equipped with lots of silk dragon nets. You lead the team back to the caves. You hope that there aren't too many more dragons in the dirty cave.

After you return from dragon hunting, sit down at the computer and play the Greater Than What? game. The computer flashes two numbers on the screen. You have to decide if one number is greater than the other; or less than the other; or greater than or equal to the other; or less than or equal to the other.

Think of the numbers as dragons.

In this game the computer expects a yes or no answer. You give a "YES" answer by pressing the **ENTER** button. You give a "NO" answer by pressing the **SPACE bar.**

The Game ...

Program Name: **GREATER**

```
50 REM   * GREATER-THAN GAME *
60 CALL CLEAR
70 PRINT "** THE GREATER-THAN GAME **"
80 GOSUB 500
```

```
110 FOR PAUSE=1 TO 900
120 NEXT PAUSE
130 RANDOMIZE
190 CALL CLEAR
200 PRINT TAB(2);"1. GREATER THAN .......>"
210 PRINT
220 PRINT TAB(2);"2. LESS THAN...........<"
230 PRINT
240 PRINT TAB(2);"3. GREATER THAN/EQUAL..>="
250 PRINT
260 PRINT TAB(2);"4. LESS THAN/EQUAL.....<="
265 PRINT
270 PRINT
275 PRINT
280 INPUT "WHICH NUMBER (1,2,3 OR 4)?":B
290 IF B<1 THEN 190
300 IF B>4 THEN 190
305 FOR L=1 TO 5
310 N1=INT(RND*10+1)
320 N2=INT(RND*10+1)
330 CALL CLEAR
340 ON B GOSUB 3010,4010,5010,6010
342 PRINT "  ENTER=YES      SPACE=NO";
345 CALL KEY(0,K,S)
346 IF S=0 THEN 345
348 IF K=13 THEN 360
350 IF K=32 THEN 360
356 GOTO 345
360 IF K=W THEN 365
362 GOSUB 2000
364 GOTO 330
365 GOSUB 1000
370 NEXT L
380 CALL CLEAR
390 INPUT "PLAY AGAIN?":A$
400 IF SEG$(A$,1,1)="Y" THEN 190
410 IF SEG$(A$,1,1)<>"N" THEN 380
420 CALL CLEAR
430 END
500 FOR Q=1 TO 11
510 PRINT
520 NEXT Q
530 RETURN
3000 REM * GREATER-THAN (>) *
3010 PRINT TAB(10);N1;">";N2
3020 GOSUB 500
```

```
3030 IF N1>N2 THEN 3060
3040 W=32
3050 RETURN
3060 W=13
3070 RETURN
4000 REM  * LESS-THAN (<) *
4010 PRINT TAB(10);N1;"<";N2
4020 GOSUB 500
4030 IF N1<N2 THEN 3060 ELSE 3040
5000 REM   * GREATER THAN/EQUAL *
5010 PRINT TAB(9);N1;">=";N2
5020 GOSUB 500
5030 IF N1=N2 THEN 3060
5040 IF N1>N2 THEN 3060 ELSE 3040
6000 REM  * LESS THAN/EQUAL *
6010 PRINT TAB(9);N1;"<=";N2
6020 GOSUB 500
6030 IF N1=N2 THEN 3060
6040 IF N1<N2 THEN 3060 ELSE 3040
```

Typing Hints ...

Remember to load the Happy-Sad routine before typing in this program.

Highlights ...

This game is similar to The Arithmetic Game. When the program begins, it calls the menu routine beginning on line 200. You can choose four different kinds of number comparison problems: greater than (>), less than (<), greater than or equal (>=), or less than or equal (<=).

The FOR-NEXT loop on lines 305 to 370 is the main program loop. It sets the number of problems per game. Right now the loop is set to offer you five problems per game.

The two RND functions on lines 310 and 320 randomly select the numbers to be compared. The ON B GOSUB command on line 340 calls the appropriate subroutine, based on your choice of problem type (greater than, less than, greater than or equal, less than or equal).

The four subroutines beginning at lines 3000, 4000, 5000, and 6000 handle the four types of problems. They display the problem on the TV screen. They accept your answer, and check to see if it is correct. If it is correct, they call the Happy Face subroutine. If it is incorrect, they call the Sad Face subroutine.

You give a "YES" answer by pressing the **ENTER** button (an ASCII value of 13). You give a "NO" answer by pressing the **SPACE** bar (an ASCII value of 32).

Variables ...

PAUSE	Delay loop counter.
A$	Answer to the question "TRY AGAIN?"
L	Loop counter—main problem loop.
N1	First number the computer randomly selects.
N2	Second number the computer randomly selects.
B	Your choice of problem type (>, <, >=, or <=).
K	Your answer (**ENTER** = YES, **SPACE** bar = NO).
W	Stores either a 13 or a 32 for the correct answer.

Do-It-Yourself ...

You can add a new problem type—not equal (<>).

You can combine the Greater Than What? game and The Arithmetic Game. First the children would have to do an arithmetic problem. Then they would have to use one of the five number comparison symbols (>, <, >=, <=, or <>).

Or you can combine the two games by asking questions like:

```
5 + 9 > 14 + 0 ?
```

Or you could help older children practice algebra with problems like:

```
IF X = 100 THEN IS X/5 >= .1X + 9 ?
```

9
THE HAMBURGER CONTEST

For Parents and Teachers ...

This game helps children practice multiplication.

The computer draws from one to nine hamburgers and from one to nine people. It asks, for example, if four people eat three hamburgers each, how many hamburgers are eaten?

For Kids ...

Do you like hamburgers?

I hope so, because you've been elected by your school to go to the first national KIDS' HAMBURGER EATING CONTEST.

You go to Washington, D.C. You and eight other children sit down at a huge picnic table. On top of the table are nine stacks of juicy hamburgers. The stacks of burgers are so high that they reach into the sky.

The President of the United States comes to the table, wishes you all good luck, and fires a pistol into the sky. You and the other children start munching burgers.

Half an hour later, the President fires the pistol again. “Boy!” he says. “A lot of burgers were eaten here today. You each ate seven burgers, and there are nine of you. If nine people each eat seven burgers, how many burgers are eaten?”

You still have a half-eaten burger in your mouth. But you have the answer. You raise your hand. “MFFF!” you say.

“What was that?” the President asks.

You spit the hamburger out of your mouth. Now you can talk. “63!” you yell. “63 hamburgers.”

“RIGHT!” says the President.

The Game ...

Program Name: **BURGER**

```
10 REM * HAMBURGER CONTEST *
20 REM *     MACK MCGHEE     *
25 CALL CLEAR
30 PRINT " *** HAMBURGER CONTEST ***"
40 FOR Q=1 TO 11
50 PRINT
60 NEXT Q
70 CALL CHAR(136,"00001F2028212110")
75 CALL CHAR(137,"0000F00828080810")
80 CALL CHAR(138,"0B080403")
85 CALL CHAR(139,"A0204080")
90 CALL CHAR(140,"00000F102020207F")
95 CALL CHAR(141,"0000F008040404FE")
100 CALL CHAR(142,"7F7F2020201F")
105 CALL CHAR(143,"FEFE040404F8")
110 CALL CHAR(144,"C0E070381C0E0703")
115 CALL CHAR(145,"03070E1C3870E0C0")
120 RANDOMIZE
150 FOR PAUSE=1 TO 900
160 NEXT PAUSE
170 MAN=INT(RND*9+1)
180 HAM=INT(RND*9+1)
182 B=MAN
```

```
184 T=HAM
190 CALL CLEAR
200 C=0
210 FOR Y=3 TO 11 STEP 4
220 FOR X=3 TO 11 STEP 4
230 IF C>=MAN THEN 290
240 CALL HCHAR(Y,X,136)
250 CALL HCHAR(Y,X+1,137)
260 CALL HCHAR(Y+1,X,138)
270 CALL HCHAR(Y+1,X+1,139)
280 C=C+1
290 NEXT X
300 NEXT Y
310 C=0
320 FOR Y=3 TO 11 STEP 4
330 FOR X=21 TO 29 STEP 4
340 IF C>=HAM THEN 390
350 CALL HCHAR(Y,X,140)
360 CALL HCHAR(Y,X+1,141)
370 CALL HCHAR(Y+1,X,142)
380 CALL HCHAR(Y+1,X+1,143)
390 C=C+1
400 NEXT X
410 NEXT Y
420 CALL HCHAR(7,16,144)
430 CALL HCHAR(7,17,145)
440 CALL HCHAR(8,16,145)
450 CALL HCHAR(8,17,144)
500 A$="IF "&STR$(B)
510 IF B>1 THEN 540
520 A$=A$&" PERSON EATS "&STR$(T)
530 GOTO 550
540 A$=A$&" PEOPLE EAT "&STR$(T)
550 IF T=1 THEN 580
560 A$=A$&" BURGERS"
570 GOTO 590
580 A$=A$&" BURGER"
590 IF (B>1)*(T>1)+(B>1)*(T=1)THEN 610
600 GOTO 620
610 A$=A$&" EACH"
620 A$=A$&","
630 Y=14
640 GOSUB 900
650 A$="HOW MANY BURGERS ARE EATEN?"
660 Y=16
```

```
670 GOSUB 900
680 C=0
685 A$=""
690 CALL KEY(0,ANS,S)
700 IF S=0 THEN 690
710 IF ANS=13 THEN 755
720 CALL HCHAR(18,16+C,ANS)
730 A$=A$&CHR$(ANS)
740 C=C+1
750 GOTO 690
755 IF VAL(A$)=T*B THEN 760
756 GOSUB 2000
757 GOTO 190
760 GOSUB 1000
765 CALL CLEAR
770 INPUT "  PLAY AGAIN?":A$
780 IF SEG$(A$,1,1)="Y" THEN 170
790 IF SEG$(A$,1,1)<>"N" THEN 765
800 CALL CLEAR
810 END
900 REM  * PRINT TEXT *
910 P=INT(16-LEN(A$)/2)
920 FOR Q=1 TO LEN(A$)
930 CALL HCHAR(Y,P+Q,ASC(SEG$(A$,Q,1)))
940 NEXT Q
950 RETURN
```

Typing Hints . . .

Remember to load the Happy-Sad routine before typing in this program.

Background . . .

The Hamburger Contest game was designed by Mack McGhee, a student at Patrick Henry High School in Roanoke, Virginia.

Mack writes: "Working on this program was a small adventure for me. When I began taking computer science at the beginning of this year, I never dreamed that after only a couple months I would be writing a computer program for a book.

"After changing the program many times and completely starting over once, I noticed the deadline approaching. My teacher, David James, began taking us home from school at 5 or 6 every night. He carried the class's one computer to all the different kids' houses. Sometimes he would arrive with the computer at my house as late as 10:30 PM. Once, on a school night, I worked on my program until 2:30 in the morning.

"Doing this program was a good experience. I hope you have as much fun playing it as I did making it."

Highlights ...

Lines 70 to 85 redefine the ASCII character codes 136 to 139 as the people.

Lines 90 to 105 redefine the ASCII character codes 140 to 143 as the hamburger.

Lines 110 and 115 redefine the ASCII character codes 144 and 145 as the parts of the times sign (×).

The computer randomly selects the number of hamburgers and people with RND functions on lines 170 and 180. The number of people and hamburgers varies between one and nine.

On lines 200 to 300, the computer uses two loops (representing columns and rows) to draw the pictures of the people in the hamburger-eating contest.

On lines 310 to 410, the computer uses two more loops (representing columns and rows) to draw the pictures of the hamburgers on the TV screen.

Only a single hamburger and only a single person is actually plotted. But the loops vary the columns and rows so that between one and nine copies of the hamburger and person are made on the screen.

Variables . . .

PAUSE	Delay loop counter.
A$	Your answer to the question PLAY AGAIN?
HAM	Random number of hamburgers selected by the computer.
MAN	Random number of people selected by the computer.
C	Counter for number of people and hamburgers being printed and is dependent on values of HAM and MAN.
T	Stores same value as HAM.
B	Stores same value as MAN.
ANS	Carries the ASCII code of the answer received from the keyboard.
X	Horizontal position of people and hamburgers.
Y	Vertical position of people and hamburgers.

Do-It-Yourself . . .

You can add a wrong-answer counter to the game. You can increment the counter by one each time a child gives the wrong answer. If a child gives the wrong answer three times in a row, you can have the computer print out the correct answer and then give the child the same problem again.

You can experiment with ways to make the color of the people different from the color of the hamburgers.

You can add sound-effects to the game, such as the munching noises you might hear in a real hamburger contest.

How about making the people different shapes? You can have the computer draw fat faces and skinny faces, faces with long hair and short hair, and big noses and little noses.

10
PEPPERONI, PLEASE!

For Parents and Teachers ...

This game will help children practice their division and fractions.

The computer draws a Sicilian (square) pizza pie on the TV screen. It divides the pie into anywhere from two to eight slices. Then it draws a random number of people on the screen. For example, the computer might ask: If there are three people trying to eat six slices of pizza, how many slices does each person get?

When the child has correctly answered the question (or answers the question incorrectly three times in a row), the computer prints the correct answer (2 slices) and displays, as a fraction, the amount of the whole pizza each person gets (2/6 or 1/3).

For Kids . . .

Imagine that one night you and a bunch of your friends go to a pizza restaurant. You order the biggest Sicilian (square) pizza, but there are a lot of people, and everyone is starved.

The pizza arrives, and people attack it. But no one has figured out how much pizza each person should get. Everyone tries to grab as many slices of pizza as they can reach. Fights break out. People have pieces sticking out of their ears, their pockets, and their socks. The pizza is gone, and nobody got more than a couple of bites.

You go home and take a long shower. You use lots of soap and hot water to try to scrub the mozzarella cheese and mushrooms out of your hair.

Now imagine that you plan to go with these same friends to a pizza restaurant again tonight. This time you plan to be prepared. You sit down at your computer and RUN your new game, Pepperoni, Please!

The game draws a picture of a pizza on the TV screen. The pizza has between two and eight slices. Then the game draws anywhere from one to eight people on the screen. The game asks: "HOW MANY SLICES PER PERSON?" and "HOW MANY SLICES IN THE WHOLE PIZZA?"

You spend an hour playing the game. Now you're ready to go to the restaurant with your friends. No matter how many people go, and no matter how many slices of pizza there are, you will know how much pizza to give each person.

The Game . . .

Program Name: **PIZZA**

```
5 REM    * PIZZA,PLEASE! *
10 REM  * HOWARD BOGGESS *
20 CALL CLEAR
30 PRINT TAB(8);"PIZZA, PLEASE! "
40 PRINT
45 PRINT
50 PRINT TAB(7);"AN EXERCISE IN"
```

```
60 PRINT
70 PRINT
80 PRINT TAB(10);"FRACTIONS"
90 FOR Q=1 TO 7
95 PRINT
100 NEXT Q
110 GOSUB 900
120 FOR DELAY=1 TO 900
130 NEXT DELAY
140 RANDOMIZE
150 COUNT=0
160 M=INT(RND*7+2)
170 CALL CLEAR
180 ON M GOSUB 1,3010,3510,4010,4510,5010,5510,6
010
182 CALL HCHAR(24,1,32,30)
185 A$="COUNT THE SLICES"
186 GOSUB 710
190 FOR DELAY=1 TO 1500
200 NEXT DELAY
210 CALL HCHAR(24,1,32,30)
220 A$="COUNT THE PEOPLE"
221 GOSUB 710
240 ON M GOTO 1,250,260,270,280,290,300,310
250 REM  * 2 PEOPLE *
252 N=M
255 GOTO 320
260 REM  * 3 PEOPLE *
262 N=M
265 GOTO 320
270 REM  * 2 OR 4 PEOPLE *
272 N=4/INT(RND*2+1)
275 GOTO 320
280 REM  * 5 PEOPLE *
282 N=M
285 GOTO 320
290 REM  * 2,3 OR 6 PEOPLE *
292 N=6/INT(RND*3+1)
295 GOTO 320
300 REM  * 7 PEOPLE *
302 N=M
305 GOTO 320
310 REM  * 2,4 OR 8 PEOPLE *
312 N=INT(RND*4+1)
314 IF N=3 THEN 312
```

```
315 N=8/N
320 GOSUB 7000
330 FOR DELAY=1 TO 900
340 NEXT DELAY
350 CALL HCHAR(24,1,32,30)
360 A$="HOW MANY SLICES PER PERSON?"
365 GOSUB 710
370 GOSUB 810
375 CALL HCHAR(24,30,K+48)
376 FOR Q=1 TO 100
378 NEXT Q
380 YY=K
390 CALL HCHAR(24,1,32,30)
400 A$="HOW MANY SLICES IN ALL?"
405 GOSUB 710
410 GOSUB 810
415 CALL HCHAR(24,28,K+48)
416 FOR Q=1 TO 100
418 NEXT Q
420 YZ=K
425 CALL HCHAR(24,1,32,30)
430 IF (YY=M/N)*(YZ=M)THEN 465
435 GOSUB 2000
440 COUNT=COUNT+1
450 IF COUNT>2 THEN 470
460 GOTO 360
465 GOSUB 1000
470 GOSUB 8000
480 CALL CLEAR
490 INPUT "  PLAY AGAIN?":A$
500 IF SEG$(A$,1,1)="Y" THEN 150
510 IF SEG$(A$,1,1)<>"N" THEN 480
520 CALL CLEAR
530 END
700 REM  * PRINT MESSAGE *
710 P=15-INT(LEN(A$)/2)
720 FOR Q=1 TO LEN(A$)
730 CALL HCHAR(24,P+Q,ASC(SEG$(A$,Q,1)))
740 NEXT Q
750 RETURN
800 REM  * ANSWER ROUTINE *
810 CALL KEY(0,K,S)
820 IF S=0 THEN 810
830 IF (K<49)+(K>57)THEN 810
850 K=K-48
```

```
860 RETURN
900 REM  * LOAD CHARACTERS *
910 CALL CHAR(136,"3C7EDBFFDBE77E3C")
915 CALL CHAR(137,"1818FF999918183C")
920 CALL CHAR(138,"3C666666666E7E7")
925 CALL CHAR(144,"FF80808080808080")
930 CALL CHAR(145,"FF01010101010101")
935 CALL CHAR(146,"80808080808080FF")
940 CALL CHAR(147,"01010101010101FF")
950 RETURN
3000 REM    * 2 SLICES *
3010 FOR Y=7 TO 9 STEP 2
3020 X=6
3030 GOSUB 3210
3040 NEXT Y
3050 RETURN
3200 REM    * DRAW SLICES *
3210 CALL HCHAR(Y,X,144)
3220 CALL HCHAR(Y,X+1,145)
3230 CALL HCHAR(Y+1,X,146)
3240 CALL HCHAR(Y+1,X+1,147)
3250 RETURN
3500 REM    * 3 SLICES *
3510 FOR Y=7 TO 11 STEP 2
3520 X=6
3530 GOSUB 3210
3540 NEXT Y
3550 RETURN
4000 REM    * 4 SLICES *
4010 FOR Y=7 TO 9 STEP 2
4020 FOR X=5 TO 7 STEP 2
4030 GOSUB 3210
4040 NEXT X
4050 NEXT Y
4060 RETURN
4500 REM    * 5 SLICES *
4510 FOR Y=5 TO 13 STEP 2
4520 X=6
4530 GOSUB 3210
4540 NEXT Y
4550 RETURN
5000 REM    * 6 SLICES *
5010 FOR Y=7 TO 11 STEP 2
5020 FOR X=5 TO 7 STEP 2
```

```
5030 GOSUB 3210
5040 NEXT X
5050 NEXT Y
5060 RETURN
5500 REM    * 7 SLICES *
5510 FOR Y=3 TO 15 STEP 2
5520 X=6
5530 GOSUB 3210
5540 NEXT Y
5550 RETURN
6000 REM    * 8 SLICES *
6010 FOR Y=5 TO 11 STEP 2
6020 FOR X=5 TO 7 STEP 2
6030 GOSUB 3210
6040 NEXT X
6050 NEXT Y
6060 RETURN
7000 REM  * DRAW PEOPLE *
7010 FOR X=14 TO 14+(N-1)*2 STEP 2
7020 CALL HCHAR(8,X,136)
7030 CALL HCHAR(9,X,137)
7040 CALL HCHAR(10,X,138)
7050 NEXT X
7060 RETURN
8000 REM * DISPLAY FRACTIONS*
8010 CALL CLEAR
8020 IF M/N<>1 THEN 8050
8030 PRINT " 1  SLICE PER PERSON"
8040 GOTO 8060
8050 PRINT M/N;" SLICES PER PERSON"
8060 PRINT
8070 PRINT
8080 PRINT M;" TOTAL PIZZA PIECES"
8090 PRINT
8100 PRINT
8110 PRINT "    EACH PERSON EATS"
8120 PRINT
8130 PRINT
8140 PRINT M/N;"/";M;" OF THE PIZZA"
8150 PRINT
8160 PRINT
8170 FOR DELAY=1 TO 1500
8180 NEXT DELAY
8190 RETURN
```

Typing Hints ...

Load the Happy-Sad routine before typing in this program.

Background ...

The Pepperoni, Please! game was designed by Howard Boggess, a student at Patrick Henry High School in Roanoke, Virginia.

Howard had been using computers for five years. In 1978 his dad bought him an INTERACT computer. Although the computer cost $500, it was primitive compared to today's computers. It didn't understand decimal numbers or **strings** (letters and words). Still, for Howard, it was an excellent learning tool.

Howard began working on this game, Pepperoni, Please!, two weeks before it was due. But he was plagued with bad luck. He writes: "I took a computer home over a weekend, with only four days left to finish. I had finished six out of seven possible pizzas by Sunday morning at 3 AM. I was watching the snow fall outside my bedroom window and putting the finishing touch on the seventh pizza. That's when all the power went out and I lost everything!"

Howard worked all day Sunday on his program. He went to school Monday, but he got out of some of his classes to continue working on the program. He continues: "The program was due on Tuesday. Monday night I took the computer home and pieced together the whole program in three hours, eating pizza the whole time. After the program was finished I thought I'd be too sick to eat pizza ever again, but that's what I had for dinner that night."

Highlights ...

The computer draws the pizza and the people using character graphics.

The subroutine beginning on line 3200 draws the pizza pie. The subroutine beginning on line 7000 draws the people.

The seven subroutines to draw the different number of pizza slices (randomly selected, from 2 to 8) begin on lines 3000, 3500, 4000, 4500, 5000, 5500, and 6000.

The subroutine to display the answer and the fraction of pizza allotted to each person begins on line 8000.

The computer chooses the number of pizza slices using an RND function on line 160.

Lines 250 to 315 take care of matching the number of randomly chosen pizza slices with the right number of people. To simplify the exercise, the computer chooses a number that results in each person getting whole slices of pizza (no fractional slices or pieces left over).

Variables ...

A$	Your answer to the question PLAY AGAIN?
DELAY	Delay loop counter.
M	Number of slices in the pizza.
N	Number of people drawn on the screen.
YY	Your answer—number of slices per person.
YZ	Your answer—number of slices in the whole pizza.

Do-It-Yourself ...

You might experiment with making the pizza pie and the people appear in different colors. You can do this by changing the color using the CALL COLOR command before drawing either the pizza or the people.

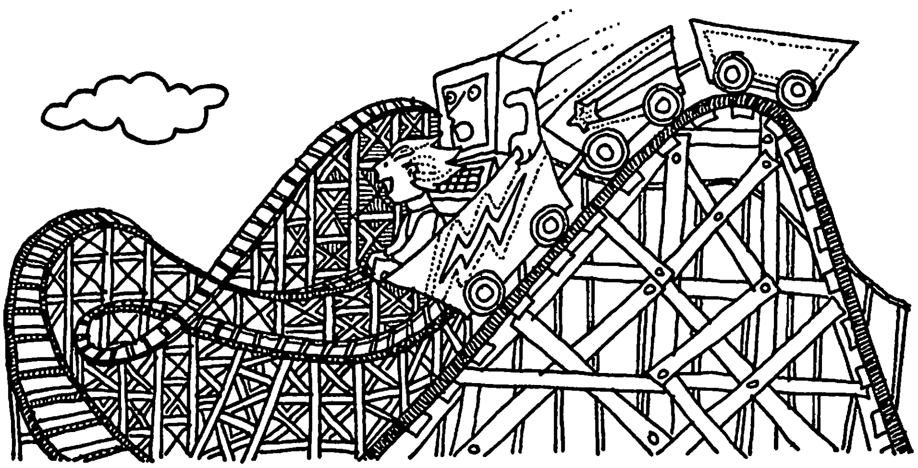
Also, by using the CALL CHAR command you can make the pizza slices triangular in shape.

You can also add sound-effects, such as noises that sound like people munching and slurping juicy pizza pies.

Also, you can show how some of the fractions can be reduced. Sometimes the program (see the subroutine beginning on line 8000) prints fractions like 2/8, 2/4, and 3/6. It would be nice if the program showed that these fractions are equivalent to 1/4, 1/2, and 1/2.

Finally, when the child answers the question correctly, you can have the number of pizza slices that go to one person turn a different color or be filled in with a redefined character.

11
THE 3-D ROLLER COASTER



For Parents and Teachers ...

This game helps children learn about the **cosine** function in trigonometry. It draws a colorful 3-D cosine curve on the TV screen. With modifications, the program can draw curves of the other trigonometric functions.

For Kids ...

How would you like to become a video artist? You can make interesting, beautiful pictures by using a few simple recipes. The recipes you use are mathematical formulas.

If you didn't use the formulas, you would have to tell the computer to draw each square or point in the picture. The formulas take care of calculating all of the points automatically.

Also, video art is a good way to spice up your math homework. By themselves, mathematical formulas can be complicated and boring. But they all describe some sort of shape. That shape might be beautiful. You can get your computer to draw the shapes on the TV screen. It makes math more interesting, and it is a lot easier than drawing the shapes yourself on a piece of graph paper.

This game is an example of video art using a cosine function. The program draws a picture of a 3-D roller coaster. The roller coaster looks like it is standing up on the TV screen.

The Game ...

Program Name: **ROLLER**

```
50 REM  ** ROLLER COASTER **
60 CALL CLEAR
70 PRINT "   *** ROLLER COASTER ***"
80 FOR Q=1 TO 11
90 PRINT
100 NEXT Q
110 CALL CHAR(135,"D6D6D6D6D6D6D6D6")
120 CALL CHAR(136,"FFFFFFFFFFFFFFFF")
130 CALL COLOR(13,16,2)
140 CALL COLOR(14,16,4)
145 CALL CHAR(144,"FFFFFFFFFFFFFFFF")
146 CALL COLOR(15,13,4)
150 FOR PAUSE=1 TO 900
160 NEXT PAUSE
170 CALL CLEAR
180 CALL SCREEN(2)
190 FOR Q=1 TO 32
195 CHR=135
200 IF ABS(COS(Q/2.3))<.9 THEN 215
210 CHR=136
215 S=5+4*COS(Q/2.3)
220 CALL VCHAR(S,Q,CHR,14)
225 CALL VCHAR(S+14,Q,144,24-(S+14))
230 NEXT Q
240 GOTO 240
```

Highlights ...

This is a very short program—only 25 lines.

On lines 110, 120, and 145 we redefine characters to use in the graph.

The program uses a single loop—from lines 190 to 230. First the program draws one line of the roller coaster. Then it draws the "space" under the roller coaster. Then it goes back to the top of the loop (line 190) and draws a new line and a new space. It continues drawing lines and spaces until it reaches the righthand edge of the screen.

Variables ...

PAUSE	Delay loop counter.
Q	Loop counter—the column on the TV screen where the roller coaster will be drawn.
CHR	Character selected for drawing.

Do-It-Yourself ...

This program is more than video art—more even than 3-D video art. It enables you to visualize simple wave functions in three dimensions. It changes numbers and formulas into a beautiful picture.

The program uses the COS (cosine) trigonometric function. You can experiment with it by using other functions—such as the SIN (sine), SQR (square root), and LOG (logarithm) functions.

Using the trigonometric equivalences shown in the accompanying table of derived functions, you can also create tangent, cotangent, secant, and cosecant functions.

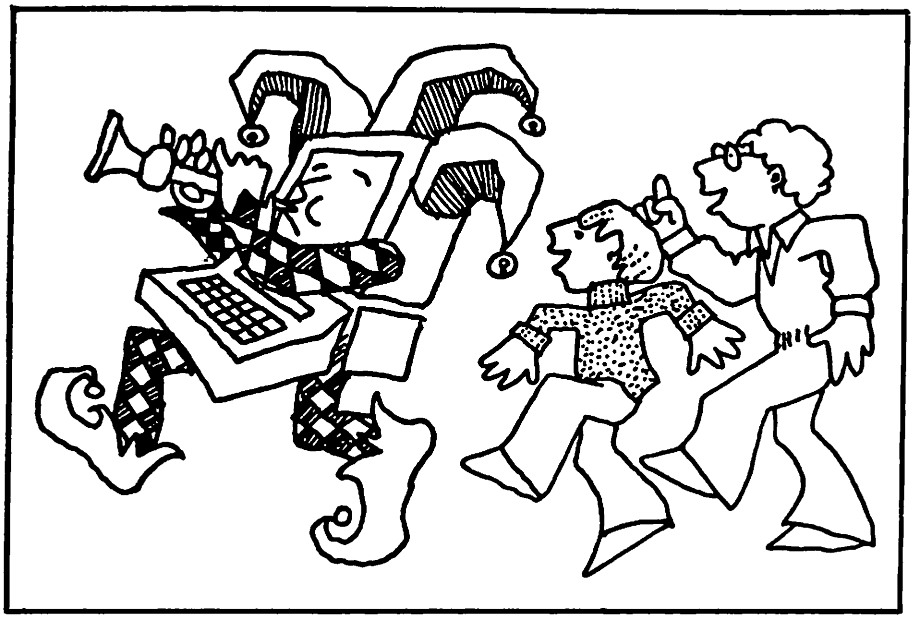
TABLE OF DERIVED TRIGONOMETRIC FUNCTIONS

TAN(X) = SIN(X)/COS(X)
COT(X) = COS(X)/SIN(X)
SEC(X) = 1/COS(X)
CSC(X) = 1/SIN(X)

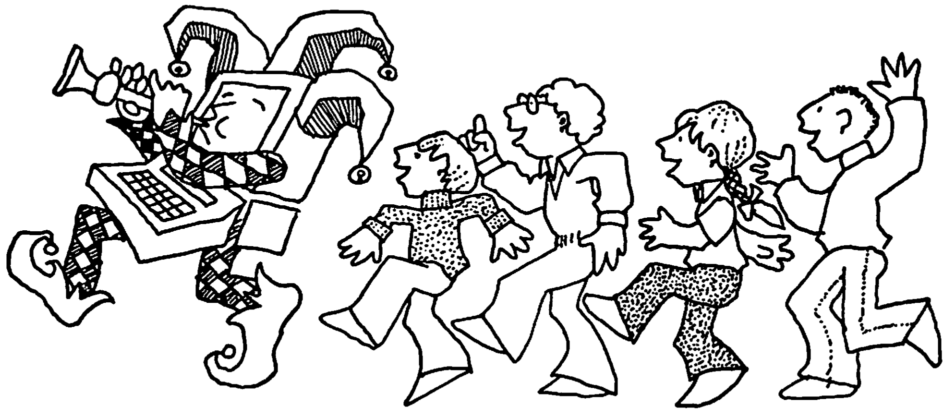
To get your child to interact with these functions, you can add variables that change the shape of the roller coaster. The child can enter different numbers to see how they affect the roller coaster's shape.

You can add a SOUND command to have the computer make a new sound as it draws each new line in the roller coaster—higher sounds for roller coaster peaks and lower sounds for dips. This will heighten the illusion that the computer (or child) is riding the roller coaster.

COLORS



12
THE PIED PIPER



For Parents ...

This is a memory and concentration game. The computer creates a chain of colors and musical tones. The colors and tones are represented by the keys 1 to 4 on the computer keyboard. First the colors are displayed and the tones play, and then the child presses the keys. The keys have to be in the same sequence as the colors and tones. If the child can remember the exact sequence of colors and tones, the chain grows one link at a time. You can adjust how many colors and tones the child has to remember to win the game.

For Kids ...

In this game, the computer becomes a Pied Piper. It plays a musical tone and flashes a color on the TV screen, and you have to follow it.

The computer starts by playing a single tone and flashing a single color. If you press the right button, the same tone plays and the same color flashes on the TV screen. You won the game.

The computer asks you: "PLAY AGAIN?" If you answer YES or Y, the computer plays a new tone and flashes a new color. If you press the right button and match the color and tone, the computer then plays a second tone and a second color. Then it repeats the first tone and the first color. Now you have to press two buttons. The first button is for the first color and tone. The second button is for the second color and tone. If you get them both right, you win again.

But the computer still isn't done. Now it is ready to challenge you with three colors and three tones. If you press the right three buttons, then it comes back with four colors and tones—and on, and on.

How high does the computer go? The computer will play 50 games in a row. It has only 4 colors and 4 tones to work with. But it can piece these together into a string of up to 50 colors and tones. If you can remember 50 colors and tones in a row, the computer gives up. You are a genius!

The object of the game is to see how long you can follow the computer Pied Piper. It keeps stringing colors and tones together. And you try to match those colors and tones by pressing the color/tone buttons in the right order.

When the computer flashes the color red on the screen and plays a low C note, you press the **1** button.

When the computer flashes the color yellow and plays a middle C note, you press the **2** button.

When the computer flashes the color green and plays a high C note, you press the **3** button.

And when the computer flashes blue and plays a high-high C, you press the **4** button.

What happens if the computer strings four colors and tones in a row? Here's an example. If the computer:

FLASHES	RED	RED	BLUE	YELLOW
PLAYS	Low C	Low C	Hi-Hi C	Middle C

You wait until the computer plays the four colors and notes. Then you press four buttons: **1 - 1 - 4 - 2**. The computer will print a happy face. You won!

If you missed one of the colors (say you typed **1 - 1 - 3 - 2**), the computer will print a sad face and then ask you if you want to start a new game.

The Game ...

Program Name: **PIPER**

```
50 REM  * PIED PIPER *
60 CALL CLEAR
70 PRINT "   *** THE PIED PIPER ***"
80 FOR Q=1 TO 11
90 PRINT
100 NEXT Q
110 DIM COMBO(50)
120 FOR Q=1 TO 4
130 READ COLOR(Q),PICH(Q)
140 NEXT Q
142 FOR PAUSE=1 TO 900
145 NEXT PAUSE
150 TOP=1
155 FLAG=0
160 FOR TOTAL=1 TO TOP
170 COMBO(TOTAL)=INT(RND*4+1)
175 NEXT TOTAL
180 FOR PLAY=1 TO TOP
190 CALL CLEAR
200 CALL SOUND(600,PICH(COMBO(PLAY)),10)
210 CALL SCREEN(COLOR(COMBO(PLAY)))
220 FOR T=1 TO 500
230 NEXT T
240 NEXT PLAY
250 CALL CLEAR
255 CALL SCREEN(4)
260 PRINT TAB(10);"YOUR TURN"
270 FOR Q=1 TO 11
280 PRINT
290 NEXT Q
300 FOR TRY=1 TO TOP
```

```
310 CALL KEY(0,A,S)
320 IF (S=0)+(S=-1)THEN 310
330 IF (A<49)+(A>52)THEN 310
340 A=A-48
350 IF A=COMBO(TRY)THEN 360
352 GOSUB 2000
354 FLAG=1
356 TRY=TOP+1
358 GOTO 430
360 CALL SCREEN(COLOR(COMBO(TRY)))
370 CALL SOUND(600,PICH(COMBO(TRY)),10)
380 NEXT TRY
390 FOR PAUSE=1 TO 300
400 NEXT PAUSE
420 GOSUB 1000
430 CALL CLEAR
440 INPUT "   PLAY AGAIN?":A$
450 IF SEG$(A$,1,1)="Y" THEN 500
460 IF SEG$(A$,1,1)<>"N" THEN 430
470 CALL CLEAR
480 END
500 TOP=TOP+1
505 IF FLAG=1 THEN 150
510 IF TOP<=50 THEN 160
520 GOTO 470
900 DATA 7,131,11,262,13,523,5,1047
```

Typing Hints . . .

Remember to load the Happy-Sad routine before typing in this program.

Highlights . . .

This program has two main loops. The first (lines 180 to 240) flashes the colors and plays the computer-selected tones. The second loop (lines 300 to 380) lets the child enter his or her answers and then checks them against the computer's list.

If you push the right button (**1**, **2**, **3**, **4**) to select the right color and tone, the computer will display the color and play the tone. Also look at lines 500 to 520. These lines keep track of the number of colors you must match. Line 505 is a flag that is set if the answer entered is incorrect. We use it to return to line 150 to start over with one tone and one color.

Variables ...

A	Variable used to translate the internal keyboard code to the numbers 1, 2, 3, and 4.
COMBO	Array variable—stores the number of the correct key (1, 2, 3, or 4) for up to 50 rounds of colors and tones. The key-number is selected at random (see line 170).
PICH	Array variable—contains the tone associated with each key.
TOTAL	FOR-NEXT loop counter—the number of colors and tones in the current round of the current game.
TOP	Counter—the total number of colors and tones in the current game.
COLOR	Array variable used to store the color values.
PLAY	FOR-NEXT loop counter—the number of colors displayed and tones played by the computer in the current round of the current game.
T	FOR-NEXT loop counter—delay loop to play the computer's tone and display the computer's picture for a certain length of time.
TRY	FOR-NEXT loop counter—the number of colors and tones you have to select in the current round of the current game.
PAUSE	FOR-NEXT loop counter—delay loop.
FLAG	Used to reset tone and color counters (1 = reset).

Do-It-Yourself . . .

You might want to change the TOP variable to something less miraculous than 50. For example, if you have a five-year-old playing the Pied Piper game, you might set TOP (see line 510) to go only up to 5 or 10. Then, if the child is able to follow the Pied Piper for 5 or 10 colors and tones, you can reward her. You can put some PRINT commands after line 420 congratulating her on doing so well.

Also, you can put some PRINT commands up front to introduce the Pied Piper and explain the rules of the game.

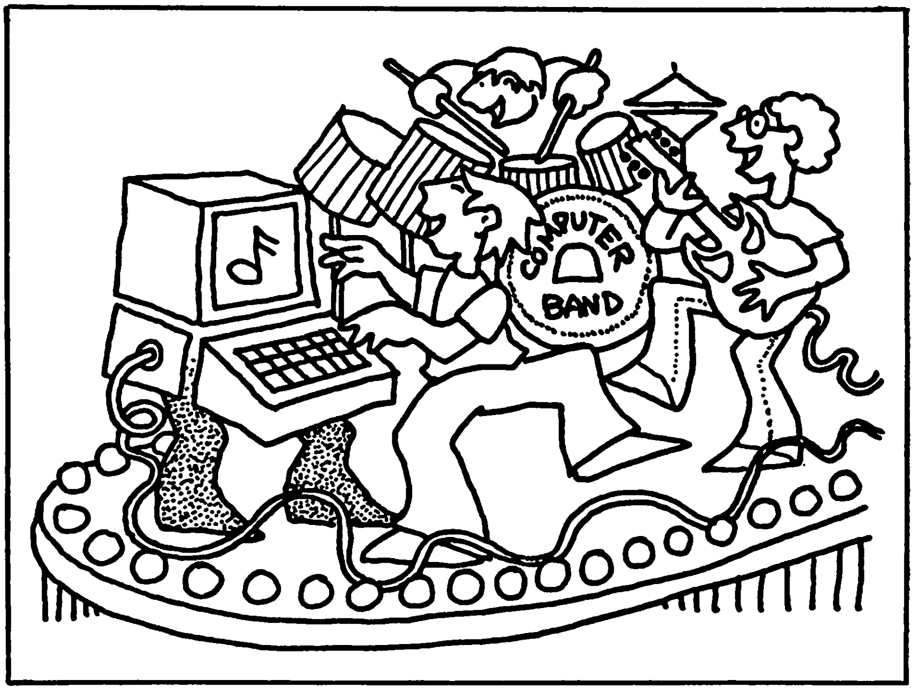
You can also make the game harder. You can create a Super Piper game by adding new tones and colors. Presently the game has only four colors and tones to work with. If you double this number, a child will have to keep track of eight sounds and eight tones strung together in combinations of up to 50 sounds and tones at a time.

You can also speed the game up. You can make the computer play the tones and display the colors in a flash by modifying the delay loop on line 220. For example, you can change the loop from FOR T=1 TO 500 to FOR T=1 TO 100.

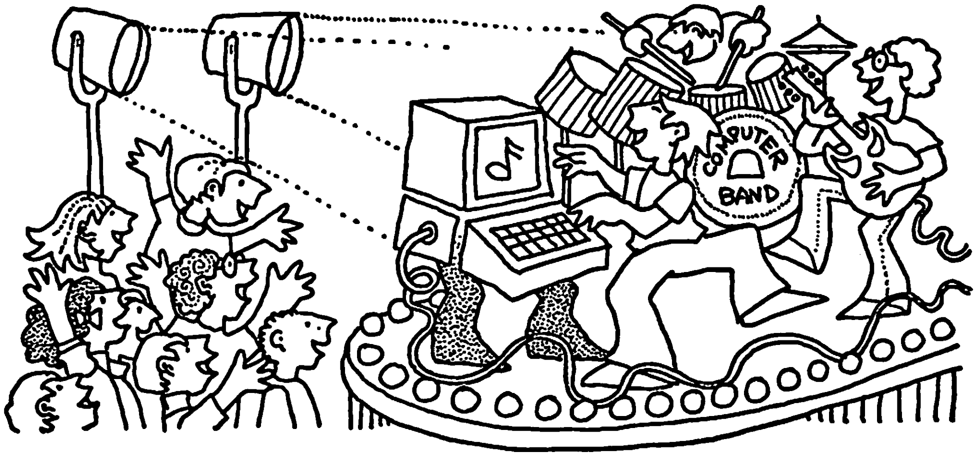
If anyone ever beat the computer at the Super Piper game, it would be a true feat of mental wizardry.

Perhaps you and your children are up to the challenge!

MUSIC



13
MAKE UP A SONG



For Parents and Teachers . . .

This game will help children learn how to create musical chords and tunes. It will teach them how different notes can be combined in beautiful, harmonious, or unusual ways.

For Kids . . .

Pretend that your computer is an electronic guitar. You strum chords on a guitar. You combine the chords into songs. Now you can do that on your computer, too.

When you RUN the CHORDS program, it will first play all the notes between middle C and high C (including the sharps). Listen carefully to these notes as the computer plays them.

Then the computer will ask you to "PICK A NOTE." Actually you get to pick out a chord consisting of three notes. A chord is a combination of two or more notes played together. To choose the notes in your chord, you need to press only one of two buttons—the SPACE bar or the ENTER button. You press the ENTER button to select a note to go in your chord. You press the SPACE bar to change the notes on the TV screen until the note displayed is the one you want to select.

You can create a song out of chords—one chord played right after the other. This game will let you create a song with up to 10 chords.

After you have created a single chord of three notes, the computer will ask you if you want to create a new chord. Answer "Y" or "YES" until you have created all the chords in your song.

When you answer "N" or "NO," the computer will play the chords in your song. First it will show how the chords are built out of the individual notes you selected. Then it will play the notes together as chords. It will play the chords one right after the other as a song.

After the computer has played all the chords, it will ask you if you want to hear the old chords again. If you type "N" or "NO," it will ask you if you want to make new chords. If you don't, the game will end. But if you do want to make new chords and answer "Y" or "YES," the computer will erase your old song and let you create a new song.

The Game ...

Program Name: **CHORDS**

```
50 REM * PLAY CHORDS GAME *
60 CALL CLEAR
70 PRINT "  *** PLAY-A-CHORD GAME ***"
80 FOR Q=1 TO 11
90 PRINT
100 NEXT Q
110 DIM N$(13),KEY(13),PLAY(2,10)
120 GOSUB 3010
130 FOR PAUSE=1 TO 900
140 NEXT PAUSE
150 GOSUB 4010
```

```
160 FOR C=1 TO 10
170 FOR O=0 TO 2
180 FOR J=1 TO 13
190 CALL CLEAR
200 CALL SOUND(500,KEY(J),5)
210 P=15-INT(LEN(N$(J))/2)
220 PRINT TAB(P);N$(J);
230 CALL KEY(0,K,S)
240 IF S=0 THEN 230
250 IF K=32 THEN 300
260 IF K<>13 THEN 230
270 PLAY(O,C)=J
280 J=14
290 GOTO 320
300 NEXT J
310 GOTO 180
320 NEXT O
330 CALL CLEAR
340 INPUT "ANOTHER CHORD? ":A$
350 IF SEG$(A$,1,1)="Y" THEN 380
360 IF SEG$(A$,1,1)<>"N" THEN 330
370 GOTO 390
380 NEXT C
390 GOSUB 4510
400 GOSUB 5010
402 FOR PAUSE=1 TO 300
403 NEXT PAUSE
410 CALL CLEAR
420 INPUT "PLAY AGAIN? ":A$
430 IF SEG$(A$,1,1)="Y" THEN 400
440 IF SEG$(A$,1,1)<>"N" THEN 410
450 CALL CLEAR
460 END
3000 REM  * LOAD KEYS *
3010 FOR I=1 TO 13
3020 READ N$(I),KEY(I)
3030 NEXT I
3040 RETURN
3050 DATA C,262,C#,277,D,294,D#,311,E,330,F,349,
F#,370
3060 DATA G,392,G#,415,A,440,A#,466,B,494,HI-C,5
23
4000 REM * PLAY UP/DOWN *
4010 CALL CLEAR
4012 PRINT " *** LISTEN TO THE NOTES ***"::
```

```
4018 FOR I=1 TO 13
4020 GOSUB 4210
4030 NEXT I
4040 FOR PAUSE=1 TO 200
4050 NEXT PAUSE
4052 CALL CLEAR
4055 PRINT " *** LISTEN TO THE NOTES ***"::
4060 FOR I=13 TO 1 STEP -1
4070 GOSUB 4210
4080 NEXT I
4090 RETURN
4200 REM * PLAY NOTE *
4210 CALL SOUND(500,KEY(I),5)
4240 PRINT TAB(14);N$(I)
4270 FOR PAUSE=1 TO 100
4280 NEXT PAUSE
4290 RETURN
4500 REM * PLAY NOTES *
4510 FOR F=1 TO C
4520 CALL CLEAR
4525 PRINT "  *** NOTES IN THE CHORD ***"::
4530 FOR E=0 TO 2
4540 CALL SOUND(500,KEY(PLAY(E,F)),5)
4550 PRINT TAB(14);N$(PLAY(E,F))
4560 NEXT E
4570 FOR PAUSE=1 TO 200
4580 NEXT PAUSE
4590 NEXT F
4600 RETURN
5000 REM * PLAY CHORDS *
5010 FOR F=1 TO C
5020 CALL CLEAR
5030 PRINT "*** LISTEN TO THE CHORDS ***"::
5040 PRINT TAB(5);N$(PLAY(0,F));TAB(14);N$(PLAY
(1,F));TAB(23);N$(PLAY(2,F))
5050 CALL SOUND(500,KEY(PLAY(0,F)),5,KEY(PLAY(1,
F)),5,KEY(PLAY(2,F)),5)
5060 FOR PAUSE=1 TO 100
5070 NEXT PAUSE
5080 NEXT F
5090 RETURN
```

Highlights . . .

Look at the DATA commands on lines 3050 and 3060. Each DATA command has two values: a musical tone and the string that holds the name of the note. The "#" symbol signifies that the note is a **sharp.**

Each time you select a note, the value of the note (a pointer to the actual note values stored in the KEY array and the note-name values stored in the N$ array) is stored in a two-dimensional (rows-and-columns) array called PLAY. There are three rows in the PLAY array to store the three notes in each chord. There are 10 columns in the PLAY array to store the (up to) 10 chords you have selected for your song.

The two new chief subroutines are the Play Notes in Chords subroutine beginning on line 4500 and the Play Chords subroutine beginning on line 5000. These subroutines look at the PLAY array and select the chords, one at a time. The Play Notes subroutine plays the individual notes in each chord. The Play Chords subroutine plays all three notes together to make a single chord sound. It also plays all the chords in the song.

Variables . . .

A$	Your answer to "ANOTHER CHORD?" and "PLAY AGAIN?"
PLAY	Two-dimensional array—the values in PLAY act as pointers to the musical note values stored in the KEY array and the note-name values stored in the N$ array.
KEY	Array—stores frequency (tone) of the notes.
C	Loop counter—major loop (controls number of chords chosen—you may choose up to 10 chords).
O	Loop counter—controls the three notes you choose for each chord.

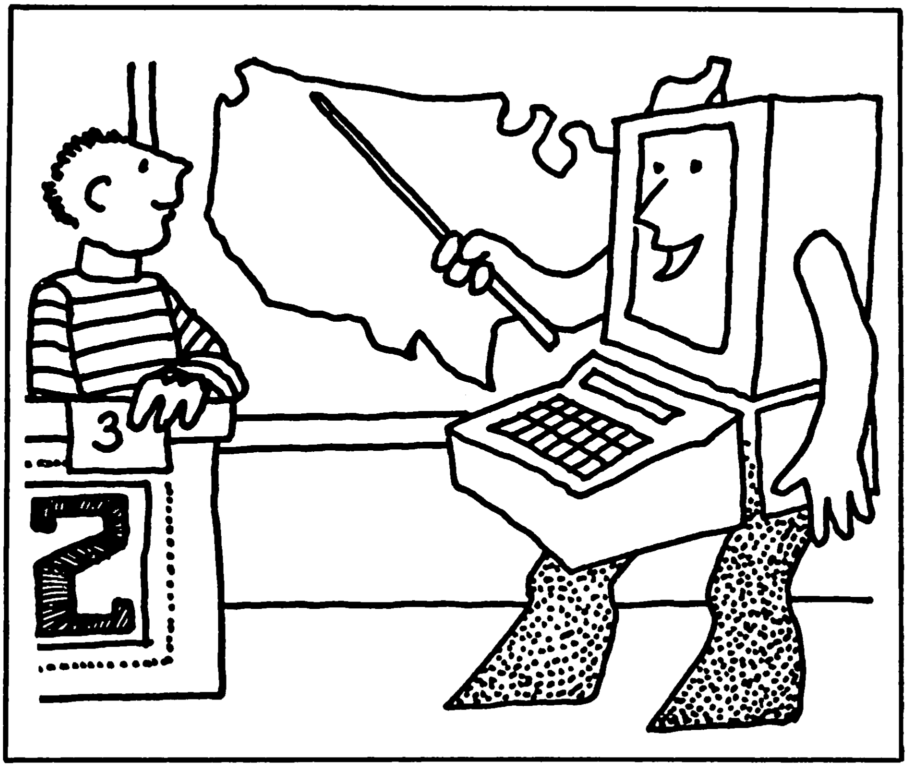
J	Loop counter—controls the 13 notes that you have to choose from (displays the notes on the TV screen).
F	Loop counter in Play Notes in Chords subroutine—controls playing of each of up to 10 chords you have selected.
E	Loop counter in Play Notes in Chords subroutine—controls playing of the three notes in each chord.
N$	String array—stores letters for the notes.
P	Sets position of note on screen.

Do-It-Yourself . . .

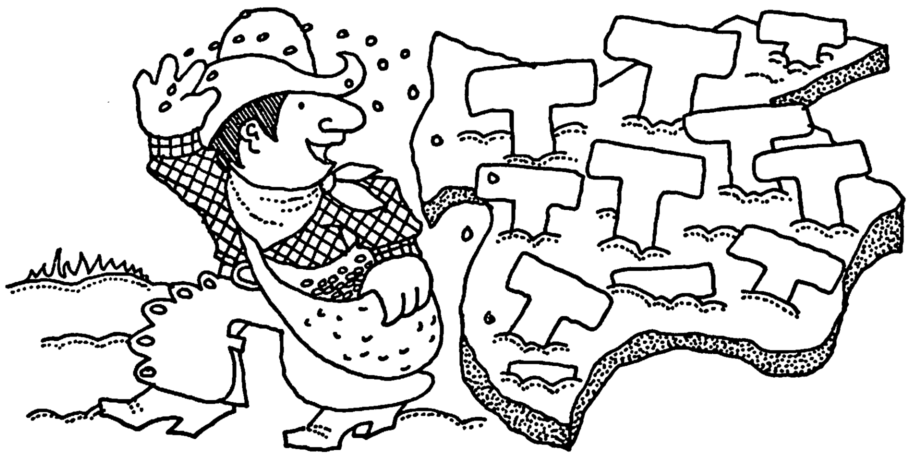
This program can be simplified or made even more elaborate. For example, you might want to create a simpler game in which children compose a song out of notes instead of chords. Or you can load in a larger number of notes so that children can create chords from up to three octaves on the musical keyboard.

Finally, you might also consider giving children the chance to select the tempo of the song and other components of the musical score such as note duration. For example, children could select eighth-notes, quarter-notes, half-notes, or whole-notes.

KNOWLEDGE



14
AL'S TOUR OF THE STATES



For Parents and Teachers ...

This game teaches children the names of all the states. A letter appears on the screen, and the children have to guess which state begins with that letter. If they can't think of any states or they can't spell the state name, the computer will let them peek at the state names beginning with each letter.

For Kids ...

Here's Alphabet Al sowing "alphabet" seeds all over the country. When Al visits a state he finds out the first letter in the state's name and then plants only that letter. For example, Al plants T's all over Texas, A's all over Alaska, and N's all over New Jersey.

Al doesn't plant any B's, E's, J's, Q's, X's, Y's, or Z's, since there are no states beginning with those letters.

After Al enters a new state and plants a letter, he stops and asks you to guess what state he is in. If you guess a state that begins with the letter Al has just planted, you win. If not, Al lets you try again.

If you can't think of a state beginning with the right letter, just press the **ENTER** button. Al will flash all the states beginning with that letter on the TV screen. When you're through studying the state names, just press **ENTER** again (or any other key). Then Al gives you another chance.

The Game . . .

Program Name: STATES

```
10 REM * AL'S TOUR OF THE STATES *
20 CALL CLEAR
30 PRINT "* AL'S TOUR OF THE STATES *"
40 FOR Q=1 TO 11
50 PRINT
60 NEXT Q
70 FOR PAUSE=1 TO 1000
80 NEXT PAUSE
90 FOR X=65 TO 90
95 IF (X=66)+(X=69)+(X=74)+(X=81)+(X>87)THEN 270
100 CALL CLEAR
110 FOR I=1 TO 30
120 CALL SOUND(100,INT(RND*1000+130),10)
130 CALL VCHAR(INT(RND*20+3),INT(RND*28+3),X)
140 NEXT I
150 FOR PAUSE=1 TO 300
160 NEXT PAUSE
170 CALL CLEAR
175 PRINT "    WHAT STATE IS AL IN?"
177 FOR Q=1 TO 11
178 PRINT
179 NEXT Q
180 A$=""
181 C=0
182 CALL KEY(0,K,S)
183 IF S=0 THEN 182
184 IF K=13 THEN 190
185 A$=A$&CHR$(K)
```

```
186 CALL HCHAR(14,10+C,K)
187 C=C+1
188 GOTO 182
190 IF A$="" THEN 3010
200 RESTORE
210 FOR STATE=1 TO 50
220 READ B$
230 IF ASC(SEG$(A$,1,1))<ASC(SEG$(B$,1,1))THEN 2
50
240 IF (B$=A$)*(ASC(SEG$(A$,1,1))=X)THEN 265
250 NEXT STATE
260 GOSUB 2000
262 GOTO 100
265 GOSUB 1000
270 NEXT X
280 END
3000 REM * DISPLAY STATES *
3010 CALL CLEAR
3020 PRINT "STATES BEGINNING WITH ";CHR$(X)
3030 PRINT "-----------------------"
3040 RESTORE
3050 FOR N=1 TO 50
3060 READ B$
3070 IF ASC(SEG$(B$,1,1))=X THEN 3100
3090 GOTO 3110
3100 PRINT B$
3110 NEXT N
3120 CALL KEY(0,K,S)
3130 IF S=0 THEN 3120
3140 GOTO 100
4000 DATA ALABAMA,ALASKA,ARIZONA,ARKANSAS,CALIFO
RNIA,COLORADO,CONNECTICUT
4010 DATA DELAWARE,FLORIDA,GEORGIA,HAWAII,IDAHO,
ILLINOIS,INDIANA,IOWA
4020 DATA KANSAS,KENTUCKY,LOUISIANA,MAINE,MARYLA
ND,MASSACHUSETTS,MICHIGAN
4030 DATA MINNESOTA,MISSISSIPPI,MISSOURI,MONTANA
,NEBRASKA,NEVADA
4040 DATA NEW HAMPSHIRE,NEW JERSEY,NEW MEXICO,NE
W YORK,NORTH CAROLINA
4050 DATA NORTH DAKOTA,OHIO,OKLAHOMA,OREGON,PENN
SYLVANIA,RHODE ISLAND
4060 DATA SOUTH CAROLINA,SOUTH DAKOTA,TENNESSEE,
TEXAS,UTAH,VERMONT,VIRGINIA
4070 DATA WASHINGTON,WEST VIRGINIA,WISCONSIN,WYO
MING
```

Typing Hints . . .

Remember to load the Happy-Sad routine before typing in this program.

Highlights . . .

On line 95, the program checks for letters that do not begin state names: B's, E's, J's, Q's, X's, Y's, or Z's. If X equals the ASCII code for that number, the program jumps to line 270 and a new code (the next letter in the alphabet) is chosen.

On line 175 the program asks what state Al is in. If you just press **ENTER**, then the program jumps to the Display States subroutine on line 3010.

On lines 3020 and 3030, the program prints out a title. Next (on lines 3050 to 3110) the program searches through the state names listed in the DATA statements on lines 4000 to 4070. The names are in alphabetical order. When the program arrives at the first state name beginning with Alphabet Al's current letter, it stops searching, and (on line 3100) prints that name.

On line 3110, the computer checks to see if it has read all 50 names and if it hasn't, it returns to line 3050 to continue checking. If it finds more names that begin with Al's letter, it prints those out, too.

The list of names stays on the screen until you press **ENTER** (or any other button). The program takes care of this on lines 3120 and 3130. On line 3120, it looks at the keyboard. If you haven't pressed a button yet, S will equal 0. After you press any button, S will not equal 0, telling the computer to continue.

Line 3120 directs the computer's attention to the keyboard for one-button messages from you. Line 3140 turns the computer's attention away from the keyboard and back to the program.

When you enter the name of a state, it gets stored in A$. The computer checks its list of state names (DATA statements 4000 to 4070) on lines 200 to 250.

If there is a match between your answer (A$) and the state name the computer pulls from the list (B$), then (on line 240) the computer jumps to the Happy Face subroutine on line 1010.

If the computer can't find a match it jumps (on line 260) to the Sad Face subroutine.

Variables ...

A$	Your answer—what state Al is in.
B$	A state from the computer's list.
X	Counter—the ASCII code for the letters in the alphabet.
I	Counter—Al plants 30 letters in each state.
STATE	Counter—computer searches through list of 50 states to find a match with your answer.
N	Counter—computer searches through list of 50 states to display states beginning with certain letter.

Do-It-Yourself ...

Al runs through the alphabet from A to Z. You can make him run through the alphabet backwards, or you can have him hop around the alphabet.

Hint #1: Look at the FOR-NEXT loop on line 90. 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 90 and 270. Hint #2: To compute a number representing a random letter of the alphabet, use the command INT(RND*26)+65. This gives you a random number between 65 and 90 (the 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 130 and 140 this line:

```
135 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 150 from 300 to 400 (or even higher).

Here is another idea. Why not have Al plant things other than letters? Letters would still fly around the screen, but they would stand for foreign countries, fruits, sports cars, rock stars, or whatever.

Hint: You will need to change the title on lines 10 and 30. You will need to change the question on line 175. You will need to change the titles on lines 3000 and 3020. You will need to replace the DATA statements on lines 4000 to 4070. You will also need to change the FOR statements on lines 210 and 3050 to the number of pizza toppings, animals, or whatever you are having Al plant (the old number is 50 for the 50 states listed in the DATA statements).

15 WHERE DO YOU LIVE?

For Parents and Teachers ...

This game helps children learn their address, including their street, apartment, city, state, and zip code. It teaches them how to format their address on an envelope.

For Kids ...

This game helps you learn your address and how to write your address on an envelope when you send a letter.

The computer asks you for your first name. Next it asks for your last name. Then it prints out your name and address, just the way you would write them on an envelope to mail to your grandparents or to a friend.

Then the computer asks you to type in your address, one part at a time. It asks you for your street, your apartment (if you live in an apartment), your city, your state, and your zip code.

If you can't remember part of your address, don't worry. Try guessing. The computer will give you three chances, then it will let you peek at the part of the address you can't remember. Then it will hide it and ask you to type it in yourself.

Maybe some parts will take dozens of tries to get just right. But the computer won't lose its temper or grumble. It never gets upset. It just keeps playing the game until you know your whole address.

The Game ...

Program Name: **ADDRESS**

```
5 REM * LEARN YOUR ADDRESS *
10 REM * THIS PROGRAM HELPS YOUNG CHILDREN TO LE
ARN THEIR ADDRESS *
20 REM  * MELISSA PERDUE *
30 CALL CLEAR
40 PRINT "     ** ADDRESS GAME **"
50 FOR Q=1 TO 11
55 PRINT
57 NEXT Q
60 FOR Q=1 TO 5
65 READ ADN$(Q)
67 NEXT Q
70 FOR Q=1 TO 5
74 READ RAD$(Q)
76 NEXT Q
80 FOR PAUSE=1 TO 900
90 NEXT PAUSE
100 CALL CLEAR
102 PRINT "HELLO! WHAT IS YOUR"
105 PRINT
110 INPUT "FIRST NAME? ":N1$
115 INPUT "LAST NAME? ":N2$
117 N$=N1$&" "&N2$
120 CALL CLEAR
130 PRINT "HI, THERE, ";N$;"!"
140 PRINT
```

```
150 PRINT "YOUR ADDRESS"
160 PRINT "------------"
165 PRINT
170 PRINT N$
172 PRINT RAD$(1)
173 IF RAD$(2)="*" THEN 175
174 PRINT RAD$(2)
175 PRINT RAD$(3)&", "&RAD$(4)&" "&RAD$(5);
176 FOR Q=1 TO 7
177 PRINT
178 NEXT Q
180 FOR PAUSE=1 TO 2000
185 NEXT PAUSE
190 PRINT "CAN YOU REMEMBER"
200 PRINT
210 PRINT "ALL THAT?"
220 PRINT
230 PRINT "LET'S TRY!!!"
240 FOR PAUSE=1 TO 1000
250 NEXT PAUSE
260 FOR J=1 TO 5
265 C=0
270 CALL CLEAR
275 IF RAD$(J)="*" THEN 555
280 PRINT "WHAT IS YOUR ";ADN$(J)&":"
330 FOR Q=1 TO 11
340 PRINT
350 NEXT Q
355 GAD$=""
356 P=0
360 CALL KEY(0,A,S)
370 IF S=0 THEN 360
380 IF A=13 THEN 420
385 GAD$=GAD$&CHR$(A)
390 P=P+1
400 CALL HCHAR(14,5+P,ASC(SEG$(GAD$,P,1)))
415 GOTO 360
420 IF GAD$<>RAD$(J)THEN 430
422 GOSUB 1000
424 GOTO 555
430 C=C+1
440 IF C>=3 THEN 460
450 GOSUB 2000
455 GOTO 270
460 CALL CLEAR
470 PRINT "YOUR ";ADN$(J);" IS:"
480 PRINT
```

```
490 PRINT RAD$(J)
500 FOR Q=1 TO 9
510 PRINT
520 NEXT Q
530 FOR PAUSE=1 TO 1500
540 NEXT PAUSE
555 NEXT J
557 CALL CLEAR
560 INPUT "   PLAY AGAIN? ":A$
570 IF SEG$(A$,1,1)="Y" THEN 260
580 IF SEG$(A$,1,1)<>"N" THEN 557
590 CALL CLEAR
600 END
5000 DATA STREET OR ROAD
5010 DATA APARTMENT
5020 DATA CITY
5030 DATA STATE
5040 DATA ZIP
6000 DATA 2117 CARTER ROAD SW
6010 DATA *
6020 DATA ROANOKE
6030 DATA VIRGINIA
6040 DATA 24015
```

Typing Hints ...

Remember to load the Happy-Sad routine before typing in this program.

Highlights ...

The Where Do You Live address game was developed by Melissa Perdue, a student at Patrick Henry High School in Roanoke, Virginia.

The address game program has a sample (Roanoke) address entered on lines 6000 to 6040. You will need to replace this with your address. Your street goes on line 6000. Your apartment goes on 6010. If you don't live in an apartment, just enter an * with the DATA command at 6010. Your city goes on line 6020, your state at 6030, your zip code at 6040.

The address categories are stored in DATA statements on lines 5000 to 5040 (i.e., STREET OR ROAD, APARTMENT, CITY, STATE and ZIP).

The main action in the program takes place inside the loops from lines 60 to 76. The program READs in the address categories (lines 60 to 67) and the actual address (lines 70 to 76).

Each time the child gets a wrong answer, the wrong-answer counter, C, is incremented by 1. If C is less than 3, the computer keeps asking for the same part of his or her address. When C reaches 3, the computer displays the correct answer, then erases it and lets the child try again.

At the beginning of the game, the program displays the child's name and address in the format in which it would appear on an envelope. If a category has an * (for example, your apartment address), the program skips to the next category (line 173).

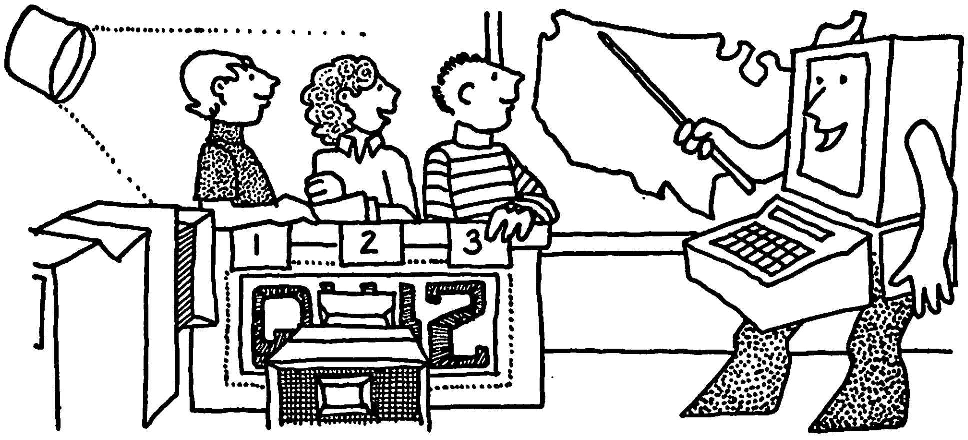
Variables ...

PAUSE	Delay loop counter.
N$	Your name.
N1$	First name.
N2$	Last name.
A$	Answer to question "PLAY AGAIN?"
GAD$	Address guess.
ADN$	Address category (street, city, etc.).
RAD$	Address (your street name, city name, etc.).
J	Loop counter—main loop of game.
C	Wrong-answer counter.

Do-It-Yourself ...

With just a couple of changes, this game can help children learn other people's addresses—friends, grandparents, etc.

16 QUIZ SHOW



For Parents and Teachers ...

This game helps children learn some interesting facts. The questions in this game can be easily replaced by questions focusing on a particular subject: history, geography, current events, language arts, etc. Also, many new questions can be added.

The game is self-grading. It keeps track of the children's correct and incorrect answers. The children's score is printed at the end of the quiz.

For Kids ...

Pretend that you get a letter in the mail. You have been invited to be on a TV quiz show!

You go to the library, and take out dozens of books. You read the newspaper every day. You study your encyclopedia.

You feel so full of facts and figures that they might soon pop out of your ears!

The big day arrives. You go to the TV studio. The show begins. They put makeup on your face, so you don't look pale under the cameras. The stage is hot. The lights are bright. You feel nervous. Here comes the first question: Who is the father of our country? George Washington? John Adams? Thomas Jefferson?

For a moment, you throw a blank. None of the names sounds right. Then you recover. "George Washington?" you say.

The quiz show announcer's happy face appears. "Right!" he says. Happy music floods the studio. The other kids look at you enviously. How could you be so sharp, so cool under pressure?

You relax. This is going to be easy.

The Game ...

Program Name: **QUIZ**

```
50 REM *** QUIZ  SHOW ***
60 REM *  SCOTT RAINEY  *
70 REM * BRIAN FRANCOIS *
75 CALL CLEAR
80 PRINT "     *** QUIZ SHOW ***"
90 FOR Q=1 TO 11
100 PRINT
110 NEXT Q
120 FOR PAUSE=1 TO 900
130 NEXT PAUSE
140 RESTORE
150 FOR QUES=1 TO 15
155 FLAG=0
160 READ Q$,AA$,AB$,AC$,ANS$
170 CALL CLEAR
180 PRINT Q$
190 PRINT
200 PRINT "A) ";AA$
210 PRINT
220 PRINT "B) ";AB$
230 PRINT
```

```
240 PRINT "C) ";AC$
250 FOR S=1 TO 7
260 PRINT
270 NEXT S
280 CALL KEY(0,A,S)
290 IF S=0 THEN 280
295 CALL HCHAR(10,LEN(Q$)-INT(LEN(Q$)/28)*28+4,A)
300 IF ANS$=CHR$(A)THEN 330
305 IF FLAG=1 THEN 320
310 AW=AW+1
315 FLAG=1
320 GOSUB 2000
325 GOTO 170
330 AR=AR+1
335 GOSUB 1000
340 NEXT QUES
350 CALL CLEAR
360 PRINT TAB(5);"YOUR SCORE IS:"
370 PRINT
380 PRINT
390 PRINT TAB(5);AR;"RIGHT"
400 PRINT
410 PRINT TAB(5);AW;"WRONG"
420 PRINT
425 PRINT
430 C=INT(AR/3)+1
450 ON C GOTO 460,480,500,520,540,560
460 A$="OH! OH! OH!"
470 GOTO 570
480 A$="YOU NEED TO STUDY MORE!"
490 GOTO 570
500 A$="YOU NEED A LITTLE WORK."
510 GOTO 570
520 A$="YOU DID OKAY."
530 GOTO 570
540 A$="YOU ARE PRETTY SMART!"
550 GOTO 570
560 A$="YOU'RE A GENIUS!"
570 PRINT A$;
580 FOR PAUSE=1 TO 2000
600 NEXT PAUSE
610 CALL CLEAR
620 END
3000 DATA WHO IS THE FATHER OF OUR    COUNTRY?,G
EORGE WASHINGTON,JOHN ADAMS,THOMAS JEFFERSON,A
```

```
3010 DATA WHAT IS THE CAPITAL OF THE  UNITED STA
TES?,NEW YORK,WASHINGTON D.C.,PHILADELPHIA,B
3020 DATA WHO IS THE CURRENT PRESIDENTOF THE UNI
TED STATES?,JIMMY CARTER,RICHARD NIXON,RONALD REA
GAN,C
3030 DATA WHICH COUNTRY GIVES ITS     PEOPLE MOR
E FREEDOM?,RUSSIA,POLAND,UNITED STATES,C
3040 DATA WHICH COUNTRY WAS THE FIRST TO PUT A M
AN IN SPACE?,RUSSIA,UNITED STATES,CHINA,A
3050 DATA WHO DISCOVERED ELECTRICITY?,BENJAMIN F
RANKLIN,ISAAC NEWTON,THOMAS JEFFERSON,A
3060 DATA WHO DISCOVERED AMERICA IN   1492?,AMER
IGO VESPUCCI,VIKINGS,CHRISTOPHER COLUMBUS,C
3070 DATA WHAT COUNTRY WAS THE FIRST  TO PUT MAN
 ON THE MOON?,RUSSIA,UNITED STATES,BRITAIN,B
3080 DATA WHAT WAS THE FIRST COUNTRY  TO ENTER T
HE NUCLEAR AGE?,UNITED STATES,CHINA,RUSSIA,A
3090 DATA WHAT WAR WON THE UNITED     STATES HER
 FREEDOM?,WAR OF 1812,REVOLUTION ,CIVIL WAR,B
3100 DATA WHO INVENTED THE TELEPHONE?,THOMAS EDI
SON,ALEXANDER GRAHAM BELL,BENJAMIN FRANKLIN,B
3110 DATA WHEN IS THE AMERICAN        INDEPENDEN
CE DAY?,NOVEMBER 25,OCTOBER 31,JULY 4,C
3120 DATA WHO WAS THE PRESIDENT OF THEUNITED STA
TES AT THE START  OF THE CIVIL WAR?
3125 DATA ABRAHAM LINCOLN,ANDREW JACKSON,U.S. GR
ANT,A
3130 DATA WHICH OF THE FOLLOWING IS A NUMERIC VA
RIABLE?,AX$,Z,45,B
3140 DATA WHAT TYPE OF COMPUTER IS    THIS?,ATAR
I,TEXAS INSTRUMENTS,TIMEX,B
```

Typing Hints ...

Remember to load the Happy-Sad routine before typing in this program.

Background ...

The Quiz Show game was designed by Scott Rainey and

Brian Francois, students at Patrick Henry High School in Roanoke, Virginia.

The boys had only two weeks to put this program together. Also, they had to use a faulty program recorder. At the end of each day they would save the current version of their program. The next morning they would try to load the program back into the computer. But the program had disappeared. The recorder had "eaten" it!

In order to complete the program on time, the boys had to get permission to skip some of their classes at school. They finally completed the program on the day it was due.

They write: "In order to complete this program, we had to overcome many obstacles. We hope that your child will learn and benefit from our work."

Highlights ...

The CALL CLEAR at the beginning of the question PRINT routine causes the computer to clear the screen.

There are two counters. The AR counter keeps track of correct answers. The AW counter keeps track of the wrong answers. At the end of the quiz, your score is printed along with a rating. The rating goes from "YOU'RE A GENIUS!" (for all 15 questions correct) to "OH! OH! OH!" (for all 15 questions wrong).

Variables ...

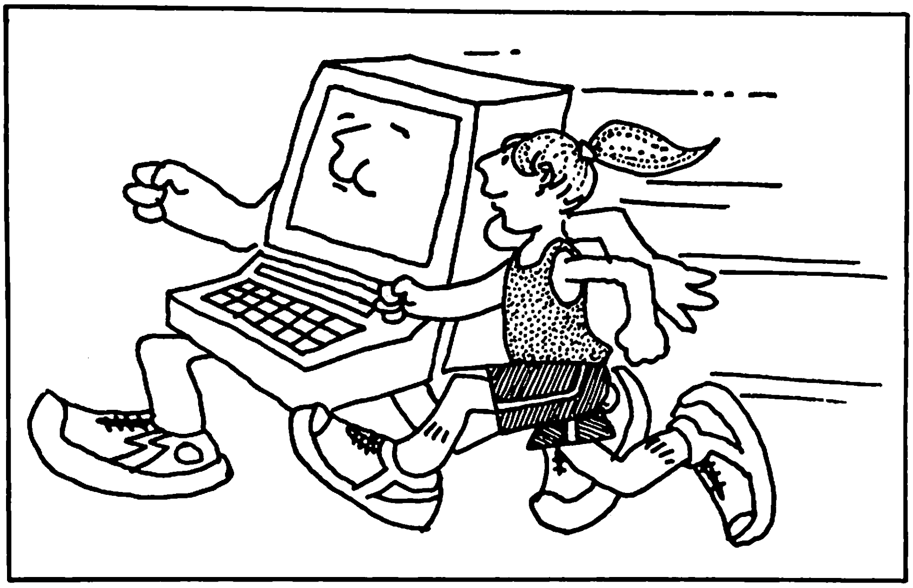
AA$, AB$, AC$	The multiple-choice answers.
ANS$	The correct one-letter answers.
Q$	The question read from data.
PAUSE	Delay loop counter.
A	Accepts your one-letter answer to quiz show questions.

AR	Counter for right answers.
AW	Counter for wrong answers.
FLAG	Keeps track if answer was wrong and gives question again if FLAG = 1.

Do-It-Yourself . . .

These questions are just a sample of the types of questions you can think up for the quiz show game. You can set up a quiz show based on your favorite facts or trivia. Do you like movies? Sports? Astronomy? Adventure games? Mysteries? Rock music? You can set up a quiz show on any of these subjects, or on anything you want.

HAND-EYE



17
HOW FAST ARE YOU?

For Parents and Teachers ...

This game helps children develop hand-eye coordination and concentration.

The computer draws a line from left to right across the screen. As soon as the child sees the line appear on the lefthand side of the screen, he or she presses the SPACE bar. When the SPACE bar is pressed, the line stops. The faster the child reacts, the better the score.

For Kids ...

How fast are you?

To find out, step right up and have a duel with the computer.

The computer will print a red light on the TV screen. This tells you to "GET READY!" As soon as you see the light on the screen turn green, press the **SPACE** bar. The light will turn red, and the computer will tell you how fast you are.

The computer will also tell you how much time passed while the green light was on.

Keep playing the game to see if you can improve your time.

If the game gets too simple, change the rules. If you are right-handed, start pressing the **SPACE** bar with your left hand. If you are left-handed, start using your right hand. Or use your nose. Or your elbow. Or turn around and use a mirror.

Let your whole family try the game. Have a family championship. Maybe you can even teach your dog or cat how to play!

The Game ...

Program Name: **FAST**

```
50 REM * HOW FAST ARE YOU? *
60 CALL CLEAR
70 PRINT "  *** HOW FAST ARE YOU? ***"
80 FOR Q=1 TO 11
90 PRINT
100 NEXT Q
110 CALL CHAR(136,"071F3F7F7FFFFFFF")
115 DIM M$(17)
120 CALL CHAR(137,"E0F8FCFEFEFFFFFF")
130 CALL CHAR(138,"FFFFFF7F7F3F1F07")
140 CALL CHAR(139,"FFFFFFFEFEFCF8E0")
145 GOSUB 510
150 FOR PAUSE=1 TO 900
160 NEXT PAUSE
170 CALL CLEAR
175 CALL SCREEN(15)
180 CALL COLOR(14,7,15)
190 CALL HCHAR(12,15,136)
200 CALL HCHAR(12,16,137)
210 CALL HCHAR(13,15,138)
220 CALL HCHAR(13,16,139)
230 FOR PAUSE=1 TO 1500
240 NEXT PAUSE
250 C=0
```

```
260 CALL COLOR(14,4,15)
270 CALL KEY(0,K,S)
280 C=C+1
290 IF S<>0 THEN 310
295 IF C=500 THEN 310
300 GOTO 270
310 CALL COLOR(14,7,15)
320 PRINT "YOUR TIME WAS";C*.0545;"SEC.";
325 GOSUB 610
330 FOR PAUSE=1 TO 1500
340 NEXT PAUSE
350 GOTO 170
500 REM * LOAD MESSAGES *
510 FOR Q=1 TO 17
520 READ M$(Q)
530 NEXT Q
540 RETURN
550 DATA ACE!!,PRO!,VERY FAST!,FAST,QUICK,AVERAGE
,OK,SLOW,VERY SLOW,SLOW AS A TURTLE
560 DATA SLOW AS A SNAIL,SLOW AS A SLOTH,WAKE UP,
WAKE UP,YOU NEED HELP
570 DATA YOU NEED HELP,IS ANYBODY THERE?
600 REM * SELECT MESSAGE *
610 S=INT(C/2)
620 IF S>=1 THEN 650
630 S=1
640 GOTO 670
650 IF S<18 THEN 670
660 S=17
670 P=INT(15-LEN(M$(S))/2)
680 FOR Q=1 TO LEN(M$(S))
690 CALL HCHAR(18,P+Q,ASC(SEG$(M$(S),Q,1)))
700 NEXT Q
710 RETURN
```

Highlights ...

The graphic characters that make up the red light are defined in lines 110, 120, 130, and 140. We set the screen background to gray in line 175. We set the color of the light to red using the CALL COLOR command in line 180. Lines 190 to 220 draw the light.

If you press the SPACE bar before the computer starts, it will just ignore the input. Subroutine 600 prints the right message on the screen. On line 320, the child's time is printed on the screen.

C*.0545 (line 320) represents the time in seconds that it took for you to press the SPACE bar.

Variables . . .

PAUSE	Delay loop counter.
M$	Message—computer tells you how well you did.
P	Positioning for message printing.
S	Selects appropriate message based on value of C (lines 610 to 710).

Do-It-Yourself . . .

The program always starts the race after it counts to 1500 on line 230. This makes it possible to anticipate the computer and get a better score than would otherwise be possible. To change this, you can put in an RND function and have the computer start the race randomly and unpredictably.

You might change the color used for the background. You could use different colored lights.

You might also define a little figure (a horse, man, hare, or tortoise) and have it race across the screen.

FOREIGN LANGUAGE

18

UNO! DOS! TRES!

For Parents and Teachers ...

This game helps children learn how to count from zero to 10 in Spanish.

For Kids ...

When you RUN this program, the screen turns blank. At the bottom, the computer asks "NUMBER (0-10)?" You can type any number from **0** to **10**. Let's say you type a **5**. The computer puts a 5 on the screen and underneath it the word for 5 in Spanish:

```
      5

    CINCO
```

When the computer prints the number, it plays a special musical tone. The computer asks you to say the name of the number, first in English and then in Spanish.

If you keep practicing the numbers in Spanish you will soon be able to say them all without any help from the computer:
"CERO! UNO! DOS! TRES! CUATRO! CINCO! SEIS! SIETE! OCHO! NUEVE! DIEZ!"

The Game ...

Program Name: **SPANISH**

```
50 REM * COUNT IN SPANISH *
60 CALL CLEAR
70 PRINT "  *** COUNT IN SPANISH ***"
80 FOR Q=1 TO 11
90 PRINT
100 NEXT Q
110 DIM NUM$(11)
120 DIM SOU(11)
130 FOR Q=0 TO 10
140 READ NUM$(Q),SOU(Q)
150 NEXT Q
160 FOR PAUSE=1 TO 900
170 NEXT PAUSE
175 FOR I=1 TO 11
180 CALL CLEAR
190 INPUT "ENTER A NUMBER (0-10)? ":K
195 IF (K>10)+(K<0)THEN 180
200 CALL CLEAR
210 CALL SOUND(1000,SOU(K),10)
220 GOSUB 510
230 GOSUB 530
240 PRINT " SAY IN ENGLISH AND SPANISH";
250 FOR PAUSE=1 TO 1000
260 NEXT PAUSE
270 NEXT I
280 CALL CLEAR
290 INPUT "  PLAY AGAIN? ":A$
300 IF SEG$(A$,1,1)="Y" THEN 175
```

```
310 IF SEG$(A$,1,1)<>"N" THEN 280
320 CALL CLEAR
340 END
500 REM  * PRINT NUMBER *
510 A$=STR$(K)
515 Y=11
520 GOTO 540
530 A$=NUM$(K)
535 Y=13
540 L=INT(15-LEN(A$)/2)
550 FOR Q=1 TO LEN(A$)
560 CALL HCHAR(Y,L+Q,ASC(SEG$(A$,Q,1)))
570 NEXT Q
580 RETURN
5000 DATA CERO,262,UNO,294,DOS,330,TRES,349,CUAT
RO,392,CINCO,440,SEIS,494,SIETE,523,OCHO,587
5010 DATA NUEVE,659,DIEZ,698
```

Typing Hints ...

Remember to load the Happy and Sad routine before typing in this program.

Highlights ...

The musical tones that accompany each number and its name are in the DATA command on lines 5000 and 5010. The musical tones are read into the SOU array on lines 130 to 150, and the number names are read into the NUM$ array also on lines 130 to 150.

The program lets you choose any 11 numbers from 0 to 10. You can count from 0 to 10. Or you can repeat the number 5 (CINCO) 11 times. The major program loop begins on line 175 and ends on line 270.

The program INPUTs your chosen number into the variable K. If you type something other than a number in response to the question on line 190, the computer will ask you the same question over again.

The program will then assign A$ the character that is represented by the ASCII value of K, using the STR$ command in line 510. Next, it will print this character using the line formatter subroutine located on lines 540 to 570. Finally, the program will use this same line formatter to center and print the name NUM$(K) and the number selected by K. Both the number and its name will be centered automatically by the formatter subroutine (lines 500 to 570).

Variables ...

PAUSE	Delay loop counter.
SOU	Array—stores musical tones for each Spanish number name.
I	Loop counter.
NUM$	String array—stores the Spanish name for the number.
A$	Stores your answer to "PLAY AGAIN?"
Q	Loop counter.
K	The number you choose.
Y	Horizontal position for screen print.

Do-It-Yourself ...

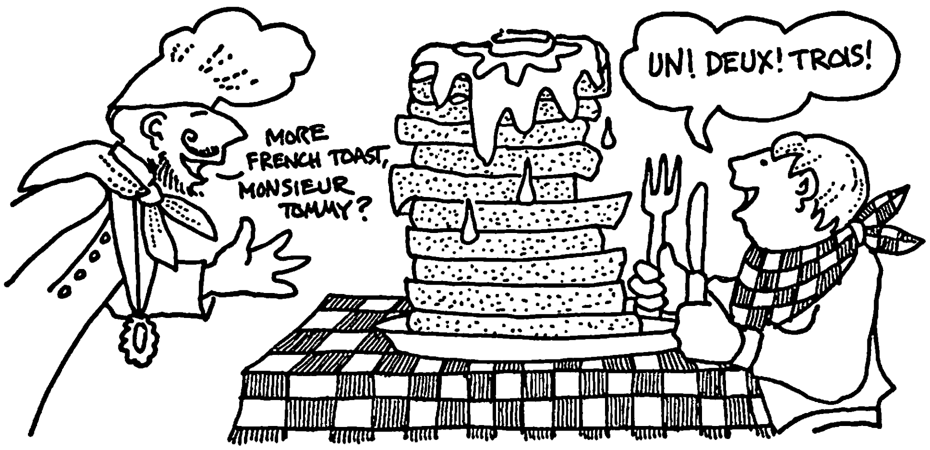
The program only accepts numbers (0, 1, 2, etc.) to indicate which Spanish number name you want to practice. You might modify the program to accept English number names—"zero," "one," "two," and so on. Or, you could have the program accept Spanish number names—"uno," "dos," "tres," etc. This would help the children learn how to spell the names.

A neat feature for the program to have would be automatic counting. At the beginning and end of the game, the program could count from 0 to 10 in Spanish. It would display all the numbers and play all the musical tones.

Also, the program could be expanded to Spanish words and Spanish phrases. It could also display the phonetic pronunciation of each of the numbers, words, and phrases to help the children say the words correctly.

19

UN! DEUX! TROIS!



For Parents and Teachers ...

This game helps children learn how to count from zero to 10 in French.

For Kids ...

When you RUN this program, the screen turns blank. At the bottom, the computer asks "NUMBER (0-10)?" You can type any number from zero to 10. Let's say you type a 5. The computer puts a 5 on the screen and underneath it the word for 5 in French:

```
        5

      CINQ
```

When the computer prints the number, it plays a special musical tone. The computer asks you to say the name of the number, first in English and then in French.

If you keep practicing the numbers in French you will soon be able to say them all without any help from the computer: "NUL! UN! DEUX! TROIS! QUATRE! CINQ! SIX! SEPT! HUIT! NEUF! DIX!"

The Game ...

Program Name: **FRENCH**

```
50 REM ** COUNT IN FRENCH **
60 CALL CLEAR
70 PRINT "  *** COUNT IN FRENCH ***"
80 FOR Q=1 TO 11
90 PRINT
100 NEXT Q
110 DIM NUM$(11)
120 DIM SOU(11)
130 FOR Q=0 TO 10
140 READ NUM$(Q),SOU(Q)
150 NEXT Q
160 FOR PAUSE=1 TO 900
170 NEXT PAUSE
175 FOR I=1 TO 11
180 CALL CLEAR
190 INPUT "ENTER A NUMBER (0-10)? ":K
200 CALL CLEAR
210 CALL SOUND(1000,SOU(K),10)
220 GOSUB 510
230 GOSUB 530
240 PRINT " SAY IN ENGLISH AND FRENCH";
250 FOR PAUSE=1 TO 1000
260 NEXT PAUSE
270 NEXT I
280 CALL CLEAR
290 INPUT "  PLAY AGAIN? ":A$
300 IF SEG$(A$,1,1)="Y" THEN 175
310 IF SEG$(A$,1,1)<>"N" THEN 280
320 CALL CLEAR
340 END
```

```
500 REM  * PRINT NUMBER *
510 A$=STR$(K)
515 Y=11
520 GOTO 540
530 A$=NUM$(K)
535 Y=13
540 L=INT(15-LEN(A$)/2)
550 FOR Q=1 TO LEN(A$)
560 CALL HCHAR(Y,L+Q,ASC(SEG$(A$,Q,1)))
570 NEXT Q
580 RETURN
5000 DATA NUL,262,UN,294,DEUX,330,TROIS,349,QUATR
E,392,CINQ,440,SIX,494,SEPT,523,HUIT,587,NEUF,659
,DIX,698
```

Typing Hints ...

Remember to load the Happy and Sad routine before typing in this program.

Highlights ...

The musical tones that accompany each number and its name are in the DATA command on line 5000. The musical tones are read into the SOU array on lines 130 to 150, and the number names are read into the NUM$ array also on lines 130 to 150.

The program lets you choose any 11 numbers from 0 to 10. You can count from 0 to 10. Or you can repeat the number 5 (CINQ) 11 times. The major program loop begins on line 175 and ends on line 270.

The program INPUTs your chosen number into the variable K. If you type something other than a number in response to the question on line 190, the computer will ask you the same question over again.

The program will then assign A$ the character that is represented by the ASCII value of K, using the STR$ command in line 510. Next, it will print this character using the line formatter subroutine located on lines 540 to 570. Finally, the program will use this same line formatter to center and print the name NUM$(K) and the number selected by K. Both the number and its name will be centered automatically by the formatter subroutine (lines 500 to 570).

Variables ...

PAUSE	Delay loop counter.
SOU	Array—stores musical tones for each French number name.
I	Loop counter.
NUM$	String array—stores the French name for the number.
A$	Stores your answer to "PLAY AGAIN?"
Q	Loop counter.
K	The number you choose.
Y	Horizontal position for screen print.

Do-It-Yourself ...

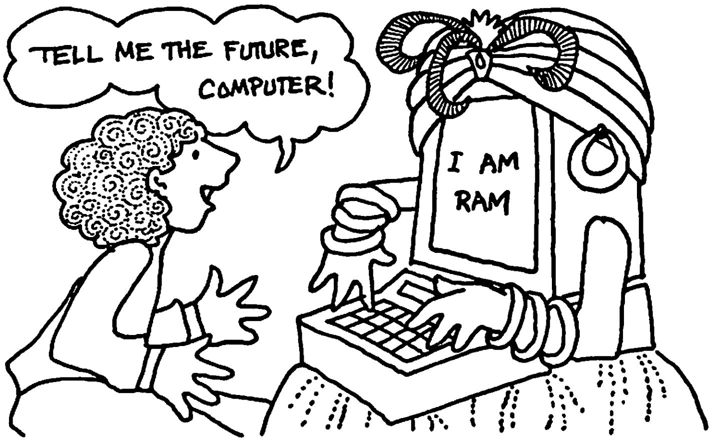
The program only accepts numbers (0, 1, 2, etc.) to indicate which French number name you want to practice. You might modify the program to accept English number names—"zero," "one," "two," and so on. Or, you can have the program display only the English name and the digit. Then the child has to type in the correct French name—"un," "deux," "trois," etc.

It would also be neat if the program could automatically count and display the numbers. At the beginning and end of the game, the program would count from zero to 10 in French. It would display all the numbers and play all the musical tones.

Also, the program could be expanded to French words and French phrases. It could also display the phonetic pronunciation of each of the numbers, words, and phrases to help the chidren say the words correctly.

IMAGINATION

20
THE FORTUNE TELLER



For Parents and Teachers . . .

This is an imagination game. The computer plays the part of a fortune teller and pretends it can see into the child's future. This program will motivate children to develop their writing and typing skills.

For Kids . . .

Imagine that it's late at night and you are asleep.

A crash of thunder wakes you. You hear eerie music coming up the stairs. You sneak downstairs to see what is making the music.

It's the computer. Strange messages appear on the computer's TV screen:

I AM RAM. I AM A WIZARD.

I AM 10,000 YEARS OLD.

I LIVE INSIDE YOUR COMPUTER.

This isn't real. You pinch yourself to see if you are dreaming. Ouch! Nope.

New messages appear:

WHEN YOU TURN ON YOUR COMPUTER,
I WAKE UP.

THEN I CAN SEE THE FUTURE.

I CAN SEE **YOUR** FUTURE!

All of a sudden, you start to shiver.

The messages continue:

ASK ME ANY QUESTION WITH A
YES OR NO ANSWER.

WHAT IS YOUR QUESTION?

Wow! You could use this computer wizard to predict the outcome of elections, or bet on horse races and make a million dollars. Or you could become a superstar reporter and find out the news even before it happens!

What should be your first question? You have it! You type madly.

The wizard flashes a new message:

I AM GAZING AT MY CRYSTAL BALL ...

GAZING ...

GAZING ...

GAZING ...

I SEE YOUR FUTURE!

YOUR QUESTION:

WILL I EVER SCORE A MILLION POINTS IN THE NEW ARCADE GAME **MEATBALL WARS**?

THE ANSWER IS ...

The wizard pauses for dramatic suspense.

You press your nose against the glass of the TV screen. You are so close the words fuzz up. Your eyes start to cross. "What's the answer?" you cry.

The wizard finally answers:

YES!!

"Hurray!" you cheer.

You run off to tell the other members of your family about the fortune teller who lives inside your computer.

The Game ...

Program Name: **FORTUNE**

```
50 REM ** FORTUNE TELLER **
55 RANDOMIZE
60 CALL CLEAR
70 PRINT " *** THE FORTUNE TELLER ***"
80 FOR Q=1 TO 11
90 PRINT
100 NEXT Q
110 FOR PAUSE=1 TO 1000
120 NEXT PAUSE
130 CALL CLEAR
140 PRINT "I AM RAM. I AM A WIZARD."
150 PRINT
160 PRINT "I AM 10,000 YEARS OLD."
170 PRINT
180 PRINT "I LIVE INSIDE YOUR COMPUTER."
190 FOR PAUSE=1 TO 2500
200 NEXT PAUSE
```

```
210 CALL CLEAR
220 PRINT "WHEN YOU TURN ON YOUR       COMPUTER,"
230 PRINT
240 PRINT "I WAKE UP."
250 PRINT
260 PRINT "THEN I CAN SEE THE FUTURE."
270 PRINT
280 PRINT "I CAN SEE YOUR FUTURE!"
290 FOR PAUSE=1 TO 2500
300 NEXT PAUSE
310 CALL CLEAR
320 PRINT "ASK ME A QUESTION WITH A"
330 PRINT
340 PRINT "YES OR NO ANSWER."
350 PRINT
360 PRINT "WHAT IS YOUR QUESTION"
370 INPUT M$
380 CALL CLEAR
390 PRINT "I AM GAZING AT MY CRYSTAL   BALL..."
400 PRINT
410 PRINT
420 FOR PAUSE=1 TO 500
430 NEXT PAUSE
440 FOR I=1 TO 3
450 PRINT "GAZING ..."
460 FOR PAUSE=1 TO 200
470 NEXT PAUSE
480 PRINT
490 NEXT I
500 PRINT "I SEE YOUR FUTURE!"
510 FOR PAUSE=1 TO 600
520 NEXT PAUSE
530 PRINT
540 PRINT
550 PRINT "YOUR QUSETION:"
560 PRINT
570 PRINT M$
580 PRINT
590 PRINT
600 PRINT "THE ANSWER IS ...";
610 FOR PAUSE=1 TO 1500
620 NEXT PAUSE
630 IF INT(RND*2+1)=1 THEN 660
640 PRINT " NO!!";
650 GOTO 670
```

```
660 PRINT " YES!!";
670 FOR PAUSE=1 TO 500
680 NEXT PAUSE
690 CALL CLEAR
700 PRINT "WANT ME TO TELL YOUR"
710 INPUT "FUTURE AGAIN? ":A$
720 IF SEG$(A$,1,1)="Y" THEN 310
730 IF SEG$(A$,1,1)<>"N" THEN 690
740 CALL CLEAR
750 END
```

Highlights . . .

This program is long, but it consists almost entirely of PRINT statements.

The key to the program is on line 630. There the computer swami "flips a coin" to decide your future. The RND function makes the computer choose either a 1 or a 2. If the computer chooses a 1, it answers your question "YES!!" If the computer chooses a 2, it answers your question "NO!!"

What makes this program a success, of course, is obviously not programming knowhow. Instead it is atmosphere and imagination. Sit down and try asking the computer to predict **your** future. Before you know it, you will be asking the computer some pretty serious questions. It's easier than you think to come under the fortune teller's spell.

Variables . . .

PAUSE	Delay loop counter.
A$	Accepts your answer to "TELL YOUR FUTURE AGAIN?"
M$	Accepts child's question.
I	Loop counter—prints "GAZING . . ." three times.

Do-It-Yourself . . .

You can add all sorts of bells and whistles to this program to heighten the illusion that a real 10,000-year-old wizard lives inside the computer. For example, the wizard can ask the child his or her name. Information in DATA statements can correlate with a particular name and the wizard can impress the child with how much he knows about him or her.

You can also add SOUND commands that make sound effects and eerie noises. You might even consider adding a crystal ball on the screen or a glimpse of the wizard's ancient face.

This program is an example of how a good game can be 90% imagination and only 10% perspiration.

21
SECRET AGENT

For Parents and Teachers ...

This is an imagination game. It takes words and sentences and turns them into secret codes for children to pass around and try to figure out. This game might act as an incentive to encourage your children to practice writing and typing on the computer.

For Kids ...

Pretend you are agent Triple-Nine. You have a secret message that you have to deliver to the president of a small country nestled in the Andes Mountains in South America. The message contains the plans to a powerful, new, top-secret Quark bomb. The president needs this bomb to defend his country against an imminent attack of robot guerillas from a neighboring country.

You are almost ready to board your private jet and fly to the president's country. But first, in case you meet with foul play, you need to translate the bomb plans into a secret code. Then, even if the plans fall into the enemy's hands, they will be useless.

You can invent a secret code. Then you can take each letter of each word in the bomb plans and translate it.

But this would take forever. Anyway, you already have a coding machine. It's your computer.

You turn on the computer, load the Secret-Agent program, and type **RUN**. The computer asks you for your code name. You type **999**. "GOOD NAME!" says the computer. "I LIKE THAT!"

You type in the secret plans, one line at a time. A message flashes on the TV screen: "CODING MACHINE NOW WORKING." Moments later the coded bomb plans appear on the screen. You copy them down and destroy the original plans. You board your airplane and head for South America.

The Game ...

Program Name: **SECRET**

```
50 REM ** SECRET AGENT **
60 CALL CLEAR
70 PRINT " *** SECRET AGENT GAME ***"
80 FOR Q=1 TO 11
90 PRINT
100 NEXT Q
110 FOR PAUSE=1 TO 900
120 NEXT PAUSE
130 CALL CLEAR
140 PRINT "WHAT IS YOUR CODE NAME="
150 INPUT N$
160 CALL CLEAR
170 PRINT "GOOD NAME! I LIKE IT!"
175 PRINT
180 PRINT N$;", WHAT IS YOUR SECRET"
190 INPUT "MESSAGE?":M$
210 CALL CLEAR
220 PRINT " CODING MACHINE NOW WORKING"
```

```
230 FOR Q=1 TO 11
240 PRINT
250 NEXT Q
255 M2$=""
260 FOR Q=1 TO LEN(M$)
270 IF SEG$(M$,Q,1)=" " THEN 300
280 M2$=M2$&CHR$(ASC(SEG$(M$,Q,1))+1)
290 GOTO 310
300 M2$=M2$&" "
310 NEXT Q
320 FOR PAUSE=1 TO 1000
330 NEXT PAUSE
340 CALL CLEAR
350 PRINT "ORIGINAL MESSAGE:"
360 PRINT
370 PRINT M$
380 PRINT
390 PRINT
400 PRINT "CODED MESSAGE:"
410 PRINT
420 PRINT M2$
430 PRINT
440 PRINT
450 INPUT "PLAY AGAIN?":A$
460 IF SEG$(A$,1,1)="Y" THEN 510
470 IF SEG$(A$,1,1)<>"N" THEN 450
480 CALL CLEAR
490 END
510 CALL CLEAR
520 GOTO 180
```

Highlights ...

This is an extremely simple program. It begins and ends with some PRINT commands. In between is a small loop on lines 260 to 310 that translates your message from English to secret code.

Line 280 is the key line. On line 280 the computer takes a letter or number in your message, converts it to its ASCII code value, then adds one. Next, it converts this number back into some new letter, character, or punctuation symbol. The message in its original form is stored in M$. The new, coded message is stored in M2$.

Variables ...

PAUSE	Delay loop counter.
A$	Your answer to question "NEW MESSAGE?"
N$	Your secret-agent name.
M$	Original message.
M2$	Coded message.
Q	Counter for coding loop.

Do-It-Yourself ...

This game is good at coding secret messages but not at decoding them. You can modify the program to make it into a new, secret-message decoder. To do that you need to change line 280. Instead of adding 1 to the ASCII value of M$(I,I), you have the program subtract 1.

You might also modify the program so it stores your child's secret messages on tape or disk. That way the messages don't have to be copied by hand back into the computer each time the child wants to code or decode them.

6415-2

TI™ in Wonderland

Fred D'Ignazio

A simple, entertaining introduction to the world of the TI-99/4A for youngsters. Learn how to write a book report, learn angle measure while "riding" a 3-D roller coaster, create songs, test reflexes, appear on a quiz show, and learn to count in French and Spanish. Twenty-one exciting short stories and original programs teach children word and number skills in an easygoing, friendly style that children love.

Contains instructions for generating TI graphics. 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 be easily typed into the TI.

HAYDEN BOOK COMPANY
a division of Hayden Publishing Company, Inc.
Hasbrouck Heights, New Jersey
ISBN 0-8104-6415-2