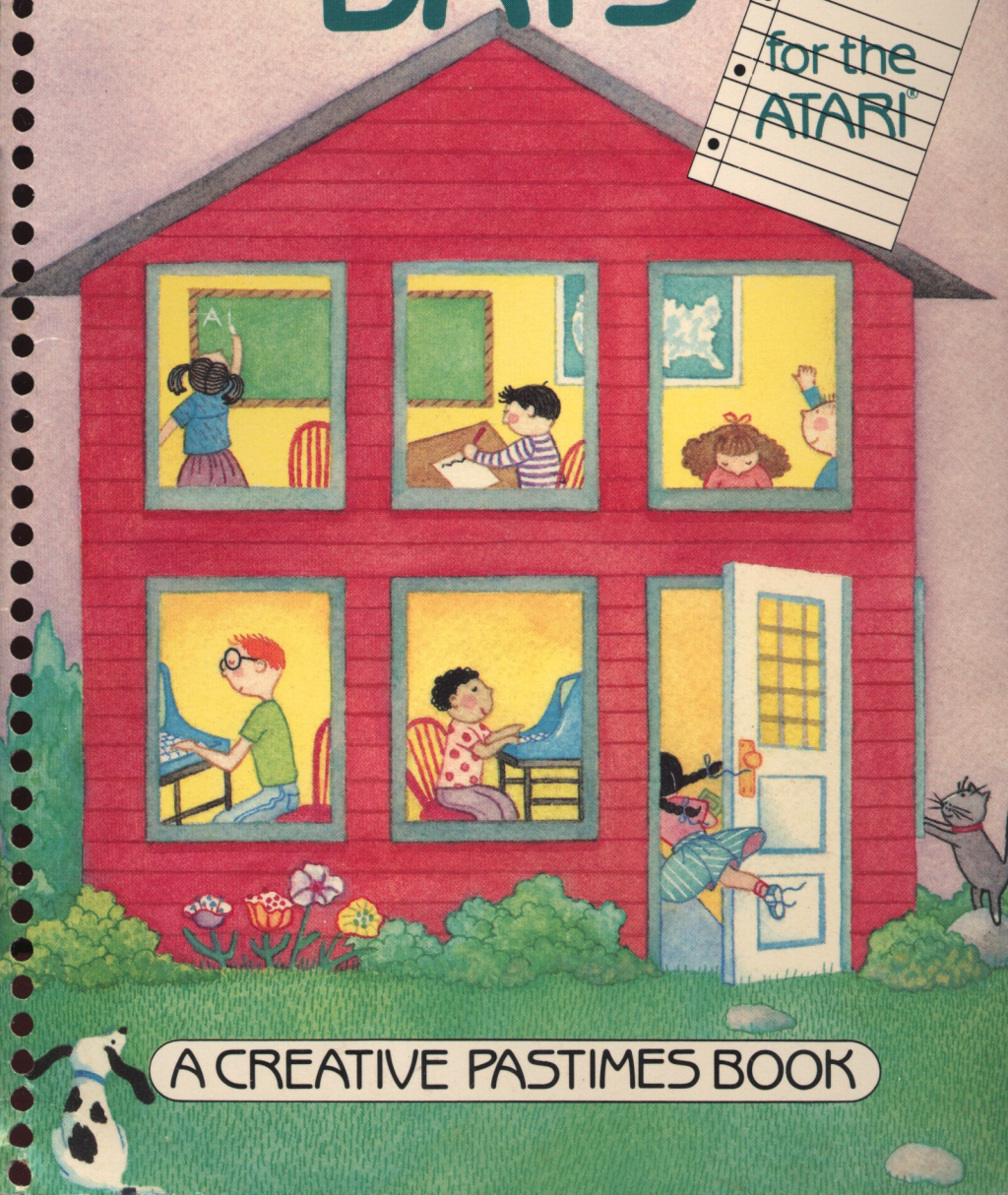
CLAIRE BAILEY PASSANTINO

SCHOOL DAYS

for the ATARI®



A CREATIVE PASTIMES BOOK

6/1984

This book belongs to

Shane & Chuck Evans

School Days
for the ATARI® Computer

Claire Bailey Passantino

Text Illustrations by Nancy Gurganus

A Reston Computer Group Book
Reston Publishing Company, Inc.
A Prentice-Hall Company
Reston, Virginia

A CREATIVE PASTIMES BOOK

Library of Congress Cataloging in Publication Data

Passantino, Claire Bailey.
School days for the Atari computer.

(A Creative pastimes book)
Summary: Presents twenty programs that include games, contests, riddles, songs, and other activities to use with BASIC on the Atari computer.
1. Computer games—Juvenile literature. 2. Atari computer—Programming—Juvenile literature. 3. Basic (Computer program language)—Juvenile literature.
[1. Computer games. 2. Atari computer—Programming. 3. Basic (Computer program language) 4. Programming (Computers)] I. Gurganus, Nancy, ill. II. Title.
III. Series.
GV1469.2.P37 1984 794.8'2 84-3471
ISBN 0-8359-6886-3

Cover illustration by Bethann Thornburgh
Cover design by Carol Conway

10 9 8 7 6 5 4 3 2 1

Printed in the United States of America

Contents

Foreword

No matter how much kids hate school, there are some things about it that are undeniably fun. Hopefully the programs in this book will make you think fondly of your own days at school.

School Days is the third of the Itty Bitty Bytes books for the ATARI® Computer. The programs in this book were developed on the ATARI 400/800™ Computer, but they will work as well on the 600XL™, 800XL™, and 1400XL™.

Each book in the series is designed to bring you twenty fun programs and, at the same time, teach you something about BASIC. Typing in the programs is hard work. You must be careful to number the lines exactly, spell the words correctly, and put in all the right punctuation. But you will feel extra good when your program runs smoothly!

Always read the explanations next to each program. Sometimes there are additions or changes that you can make to improve the program. Once you see what the program is all about, feel free to experiment with it—and by all means combine programs that work well together. The more you can do to make a program reflect your own special personality, the more you will enjoy it. Let me know if you find some nifty changes!

The Itty Bitty Bytes books have grown out of the teaching experiences I've had with my computer students. Special thank-yous go to all my COMPU-KIDS—and to my own kids, too—who keep bringing me one good idea after another. They are my severest critics, but my most outstanding asset.

Happy computing!

Claire Bailey Passantino

ATARI® is a registered trademark of Atari, Inc. 400, 800, 600XL, 800XL, and 1400XL are trademarks of Atari, Inc.

A Note to Parents and Teachers

You bought the computer. You read the manuals. You did the demo programs. You and/or your children may even have taken some computer classes. But now the computer is just sitting there. Everyone was so enthusiastic. *What happened?*

The novelty of a new computer will take you just so far. Beyond that point, a continuing interest in this incredible tool is directly related to its usefulness. "Not useful" equates to "not used." So the problem becomes, what can computers do that children would find useful? What kinds of things would encourage children to expend the energy needed to create their own computer programs?

Each Itty Bitty Bytes book in the Creative Pastimes series is packed with computer activities appealing to young programmers. Simple games, contests, races, pictures, designs, songs, riddles, charts, tests, and more—all are designed to be fun while reinforcing beginning computer concepts and skills.

Besides providing fun, there are fringe benefits to having children write their own programs. In working with children, I have found that computer programming encourages them to:

- Think creatively.
- Use logical thinking skills.
- Attend to details.
- Take small steps to achieve a goal.
- Personalize programs.
- Develop pride and self-esteem.
- Appreciate packaged software.

To help children enjoy doing their own programs, here are some suggested DOs and DON'Ts:

- DO encourage children to type in their own programs. With younger children, bargain: "You type this line and I'll type the next one." (Save the long lines for yourself!)
- DON'T criticize typing expertise. Speed and correct fingering are typing skills that are minimally related to computer programming.
- DO allow children to make mistakes.
- DO help them find the errors they've made. (This is called "debugging" the program.)
- DON'T worry when there is an error message. This means that a mistake has been made. Check the program for "bugs." (The ATARI Computer will give you an "ERROR" when you go over a "READY" signal. Ignore this message. The computer is trying to "READ Y" and there is no "Y" to read.)

- DO encourage children to read and understand the program explanations.
- DON'T, however, force the issue. Some people learn by reading. Others learn by doing. As skills are repeated over and over in different contexts, children may just "catch on."
- DO be aware of some common pitfalls. Remember to:
  1. Use line numbers.
  2. Press RETURN after you type in a line.
  3. Give great attention to spelling and punctuation. Quotation marks, commas, semicolons, colons, and even spaces are very important.
  4. Save your program before you turn off the computer.
- DO help children save their programs on tape or disk so they can use them again and again. If you have a printer, use it to make "hard" copies of each program. People like to see themselves in print.
- DON'T enter commands in reversed or lower-case characters. (Reversed characters are produced by pressing "ノ|ヽ." Press ノ|ヽ again to return to regular characters. Lower-case letters are produced by pressing the "CAPS/LOWR" key. To return to capital letters, press the "SHIFT" and the "CAPS/LOWR" keys simultaneously.
- DO praise children for a job well done. And enjoy the programs that they've created.
- DO modify and use programs that you yourself find useful.

It is my sincere hope that the Itty Bitty Bytes books will help you and your children establish a healthy working relationship with your computer. Take that computer off the shelf! And let me know how things are working out!

School Days
for the ATARI® Computer

Main ideas: Using OPEN...GET for instant input from the keyboard
Using FOR...NEXT in graphics to fill in an area
Making stripes and checks

10 Clears the screen
14 A reminder that the T-shirt drawing is coming up.
15 The shirt will be drawn in color 1, which is usually gold.
20–60 Draws the top of the T-shirt, a rectangle that extends from boxes 28 to 50 across and from boxes 2 to 6 down.
70–110 Draws another rectangle for the bottom of the shirt.
120 A remark that the pants drawing is coming up.
130 The pants will be drawn in color 2, which is usually light green.
140–180 Draws a big rectangle for the pants.
190 Draws a line from (39,24) to (39,26) in color 0, which is the background color, to form the legs of the pants.

At this point you may want to stop and have some fun. You may try different combinations to produce stripes and checks. Simply change one or all of the following lines by adding the words STEP 2. For example:

```
2Ø FOR X=28 TO 5Ø STEP 2
```

The lines you can play with are lines 20, 30, 70, 80, 140, and 150. Change some of them—or all of them—and see the results.

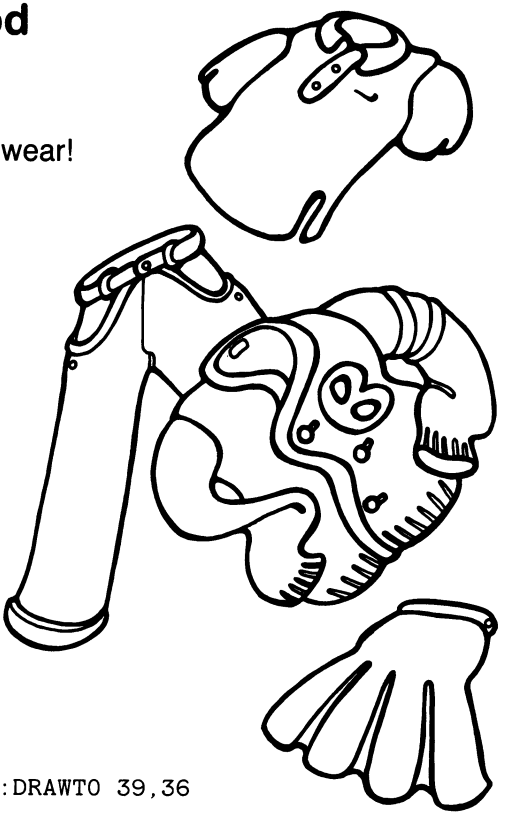
The last section allows you to select the color combination of your choice.

200 A reminder that the color selection section is coming up.
201 Prints this message in the text window.
205 Opens the keyboard for input.
210–220 Selects a random color from the 16 colors available (refer to the Setcolor Chart in the Appendix). Also selects a random shade for the color (0 is the darkest; 15 is the lightest). Changes color 1 by using setcolor 0.
230–235 Selects a color and a shade for the pants, which are drawn in color 2. To change color 2, you use setcolor 1.
240 Gets the code number of any key you press. (In effect, this just makes the program pause until you press a key.)
250 Goes to line 210 to change the colors again.

When the drawing color matches the background color, you go topless—or bottomless. Oh my!

Clothing the Bod

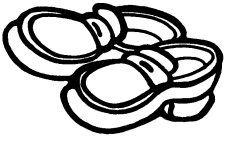
It's time to go to school.
You can't go naked.
Better find something to wear!



```
10 GR.5
14 REM T-SHIRT
15 COLOR 1
20 FOR X=28 TO 50
30 FOR Y=2 TO 6
40 PLOT X,Y
50 NEXT Y
60 NEXT X
70 FOR X=34 TO 44
80 FOR Y=7 TO 16
90 PLOT X,Y
100 NEXT Y
110 NEXT X
120 REM PANTS
130 COLOR 2
140 FOR X=34 TO 44
150 FOR Y=17 TO 36
160 PLOT X,Y
170 NEXT Y
180 NEXT X
190 COLOR 0:PLOT 39,24:DRAWTO 39,36
```

When you add the following section, you may press any key to change outfits.

```
200 REM SELECT COLORS
201 PRINT "PRESS ANY KEY TO CHANGE OUTFITS!"
205 OPEN #1,4,0,"K"
210 T=INT(16*RND(1)):S=INT(16*RND(1))
220 SE.0,T,S
230 P=INT(16*RND(1)):SH=INT(16*RND(1))
235 SE.1,P,SH
240 GET #1,N
250 GOTO 210
```



Main ideas: Writing a song using READ...DATA
Setting up titles in Graphics 2

10 Graphics 2 makes big letters. (Refer to the Graphics 2 diagram in the Appendix.)

20–30 Positions and prints the title in the middle of the screen. If you don't use the #6, the title appears at the bottom in the text window!

200 Reads the note (N) and the time (T) or number of beats to play the note. To find the numeric equivalents of the notes, refer to the Musical Notes in the Appendix.

201 Checks for the flag data (see line 320). When it reads a "−1" as data, the song is over because you go to line 330.

210 Makes the sound.

215–216 Counts to 100 for each beat (T). To speed up the song, make the pause less than 100. To slow it down, make the pause longer than 100.

217 Turns off the sound. This is important when you repeat identical notes. Without the break, the notes melt together to form one long sound.

220 Goes back to line 200 to read the numbers for the next note and time.

250–310 The data lines that hold the information for the song. Each line holds a series of numbers arranged in pairs—note, time, note, time, note, time, and so forth.

320 The flag data that signals the program to end.

For those of you who read these explanations, here's a little treat. You can change the color of the letters in the word AMERICA. Simply make some changes in line 30. Try using lower-case letters for the word "america"; try pressing the ⎇ key and using capital letters; try pressing the ⎇ key and using lower-case letters; or try some of each for a multicolored effect.

You can also change the background color by adding a line 15:

```
15 SETCOLOR 4,10,15
```

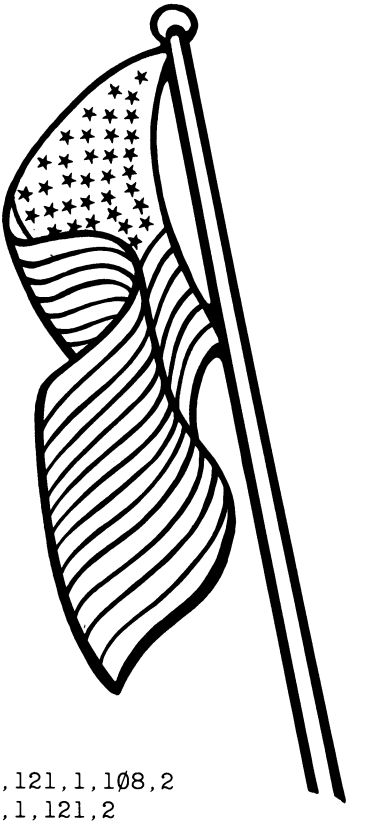
The background in Graphics 2

A color from 0 to 15 from the Setcolor Chart in the Appendix

A shade of color from 0 to 15 (the lightest shade is 15)

The American Way

School days in America all begin the same way. Everyone says the pledge to the flag. Then comes a patriotic song. Good morning, boys and girls!



```
10 GR.2
20 POS.6,4
30 PRINT #6;"AMERICA"
200 READ N,T
201 IF N=-1 THEN GOTO 330
210 SOUND 0,N,10,8
215 FOR P=1 TO 100*T
216 NEXT P
217 SOUND 0,0,0,0
220 GOTO 200
250 DATA 121,2,121,2,108,2,128,3,121,1,108,2
260 DATA 96,2,96,2,91,2,96,3,108,1,121,2
270 DATA 108,2,121,2,128,2,121,6
320 DATA -1,-1
330 END
```

Add these lines to finish the song:

```
280 DATA 81,2,81,2,81,2,81,3,91,1,96,2
290 DATA 91,2,91,2,91,2,91,3,96,1,108,2
300 DATA 96,2,91,1,96,1,108,1,121,1,96,3,91,1,81,2
310 DATA 72,1,91,1,96,2,108,2,121,6
```

Main ideas: Inputting variables into a text
Using OPEN...GET
Using FOR...NEXT to repeat an action

In this program, TN$ stands for the teacher's name, N$ stands for the student's name, S$ stands for a school subject, RN$ stands for a relative's name, A$ stands for an animal, PL$ stands for a place, and PN$ stands for the parent's name. All of these are alpha variables (variables containing letters) and each has a string sign ($). They are called *string variables.*

N stands for a number. It is a *numeric variable* and has no string sign ($).

5 Dimensions the string variables (tells how many letters may be used for each variable).
10 Clears the screen.
20–81 Allows you to input the information. Notice the use of a semicolon (;). What happens if you leave it off?

100 Clears the screen.
105 Positions the first line of the letter 7 lines down.
110–210 Prints the letter. Be careful about quotation marks, semicolons, and spaces if you want the letter to turn out right.

By the way, did I tell you about the punishment for lying?

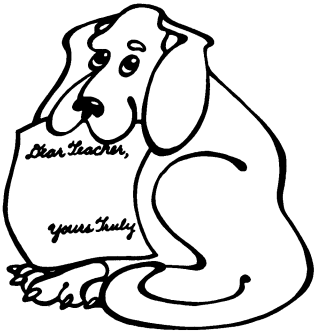
```
25Ø PRINT:PRINT "PRESS H TO SEE WHAT ";N$
26Ø PRINT "HAD TO DO FOR HOMEWORK."
27Ø OPEN #1,4,Ø,"K"
28Ø GET #1,N
29Ø IF N<>72 THEN 28Ø
295 GR.Ø
3ØØ FOR P=1 TO 5ØØ
31Ø PRINT P;" I WILL NOT TELL LIES."
32Ø NEXT P
```

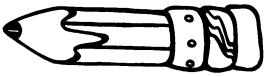
250–260 Prints the direction, inserting the name of the student.
270 Opens the keyboard for input.
280 Gets the ASCII code number for the key pressed.
290 If you press anything except the H (ASCII code 72), you go back to line 280.
300–320 The sentence in line 310 is printed 500 times. See what happens if you leave out line 320. (You need it to make the program loop back to line 310.) Can you make it print 1000 times?

The Dog Ate It!

Oh no! You forgot to do your homework. It's a good thing you're so clever. Here's your homework excuse. You get an A for creative writing.

```
5 DIM TN$(2Ø),N$(2Ø),RN$(2Ø),A$(15),PL$(2Ø),PN$(2Ø),
  S$(2Ø)
1Ø GR.Ø
2Ø PRINT "TEACHER'S NAME";
21 INPUT TN$
3Ø PRINT "YOUR NAME";
31 INPUT N$
35 PRINT "A SCHOOL SUBJECT";
36 INPUT S$
4Ø PRINT "NAME OF ONE OF YOUR RELATIVES"
41 INPUT RN$
5Ø PRINT "KIND OF ANIMAL";
51 INPUT A$
6Ø PRINT "A NUMBER";
61 INPUT N
7Ø PRINT "A PLACE";
71 INPUT PL$
8Ø PRINT "YOUR PARENT'S NAME";
81 INPUT PN$

1ØØ GR.Ø
1Ø5 POS.2,7
11Ø PRINT "DEAR ";TN$;","
12Ø PRINT "    ";N$;" DID NOT DO THE ";S$
13Ø PRINT "HOMEWORK LAST NIGHT BECAUSE"
14Ø PRINT RN$;" CAME TO VISIT."
15Ø PRINT
16Ø PRINT RN$;"'S ";A$;" GRABBED IT"
17Ø PRINT "AND TOOK ";N;" BITES OUT OF IT."
18Ø PRINT "THEN HE RAN AWAY WITH IT TO "
19Ø PRINT PL$
2ØØ PRINT "            YOURS TRULY,"
21Ø PRINT "            ";PN$
```

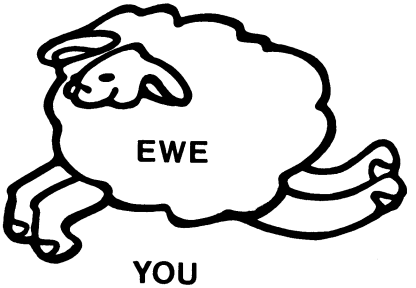


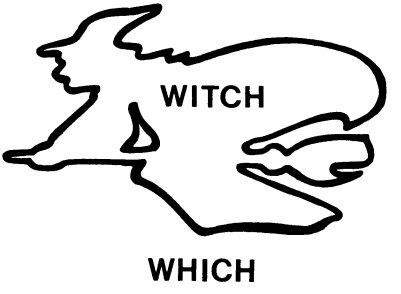


Main ideas: A simple test using READ...DATA
A test of homonyms

5 Dimensions the string variables (tells how many letters may be used for each variable).
10 Clears the screen.
20 Reads the homonym hint and the answer (from the data lines 500 to 540). The hint goes right up to the comma and includes the choices in parentheses.
21 If the word "END" is read, the program skips to line 100. (The word "END" is used as the flag data in line 900.)
30 Prints the homonym hint.
40 Allows you to input your guess (G$).
50 Prints "RIGHT!" if your guess was correct.
60 Prints a "WRONG" message and tells you the answer if your guess was incorrect.
70 Prints 2 blank lines.
80 Goes back to line 20 to read the next homonym hint and answer.
100–120 Prints a silly message at the end of the test.
500–540 The data lines that hold the information for the test. Remember that the H$ goes all the way up to the comma. The A$ comes after the comma.
900 The flag data that signals the program to end (used in line 21).

To make the game longer, add more homonyms between lines 540 and 900. To make the game harder, leave out the choices in the parentheses and let the person think of the homonym needed.





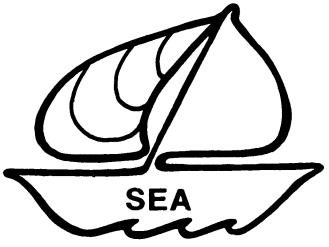
Spelling Counts!

Today's spelling lesson is about homonyms. Eye no watt a homonym is, dew ewe?

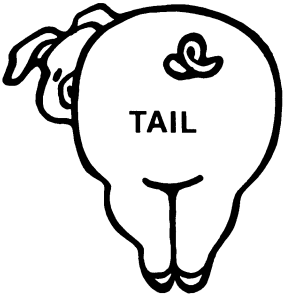
HAIR

```
5 DIM H$(35),A$(1Ø),G$(1Ø)
1Ø GR.Ø
2Ø READ H$,A$
21 IF H$="END" THEN 1ØØ
3Ø PRINT H$
4Ø INPUT G$
5Ø IF G$=A$ THEN PRINT "RIGHT!!"
6Ø IF G$<>A$ THEN PRINT "WRONG, IT WAS ";A$
7Ø PRINT:PRINT
8Ø GOTO 2Ø
1ØØ PRINT:PRINT "THIS IS THE END."
11Ø PRINT "HOW DID EWE DEW?"
12Ø END
5ØØ DATA COMES OUT AT HALLOWEEN (WHICH/WITCH),WITCH
51Ø DATA EYES DO THIS (SEE/SEA),SEE
52Ø DATA A RABBIT (HARE/HAIR),HARE
53Ø DATA A GOOD STORY (TAIL/TALE),TALE
54Ø DATA PENCILS DO THIS (RIGHT/WRITE),WRITE
9ØØ DATA END,END
```

TO, TOO



SEE



TALE

Main ideas: Using RND for arithmetic tests
Poking hearts into Graphics 2

10 Graphics 2 screen (for large letters).
20–30 Positions and prints the title on the screen.
40–50 Selects 2 numbers from 0 to 10.
60 Prints the problem in the text window.
70 Allows you to answer.
80 If you answered correctly, you skip to line 200.
90 If you answered incorrectly, prints this "WRONG" message.
100 Goes back to line 60 and prints the problem again. (You keep guessing till you get it right!)
200 A remark (to save the line number for later).
205 Prints "RIGHT!"
250 Goes back to line 40 to make up a new problem.

As for the jazzy addition:

11 Selects red for the letters on the screen: Setcolor 0 (for the letters), 4 (for red), 4 (for a dark shade).
12 Selects pink for the background: Set color 4 (for the background), 4 (for red), 10 (for a light shade).
200 A remark to remind you that this section makes the hearts.
210–240 A loop which repeats 20 times:
220 This poke adds 32 to the ATASCII code. The space (character 32) becomes character 64, which is a heart! The upper-case letters become lower-case letters.
225 Pauses briefly while the computer counts to 10.
230 Pokes back the original screen with capital letters and no hearts.

For multiplication, notice the changes in lines 60 and 80. You can print an "X" in the problem, but you need a star (*) to compute the answer.

For subtraction, you change lines 60 and 80, but you also need line 55, which checks to make sure that the first number is larger than the second number. (If the first number is smaller than the second number, you will get a negative answer. Ex: 3−5=−2.)

For division, you have several things to consider:

40 To increase the difficulty, gradually increase 11 to 101, or more!
50 Instead of numbers from 0 to 9, the +1 gives you numbers from 1 to 10. This is important, because division by 0 is impossible.
55 Checks to make sure that the dividend is larger than the divisor.
56 Checks to make sure that the answer is a whole number. Problems with remainders are rejected.
60 Prints the problem. The computer uses a slash (/) for division.
80 Checks to see if you guessed correctly.

We Love Math!

Mrs. Tiddlebaum's class *loves* math!

```
1Ø GR.2
2Ø POS.3,4
3Ø PRINT #6; "WE LOVE MATH"
4Ø X=INT(11*RND(1))
5Ø Y=INT(11*RND(1))
6Ø PRINT "WHAT IS ";X;"+";Y;
7Ø INPUT G
8Ø IF G=X+Y THEN GOTO 2ØØ
9Ø PRINT "WRONG! TRY AGAIN!"
1ØØ GOTO 6Ø
2ØØ REM
2Ø5 PRINT "RIGHT!"
25Ø GOTO 4Ø
```

See what I mean:

```
11 SE.Ø,4,4
12 SE.4,4,1Ø
2ØØ REM HEARTS
21Ø FOR H=1 TO 2Ø
22Ø POKE 756,226
225 FOR P=1 TO 1Ø:NEXT P
23Ø POKE 756,224
24Ø NEXT H
```

Try multiplication:

```
6Ø PRINT "WHAT IS ";X;"X";Y;
8Ø IF G=X*Y THEN GOTO 2ØØ
```

Or subtraction:

```
55 IF X<Y THEN 4Ø
6Ø PRINT "WHAT IS ";X;"-";Y;
8Ø IF G=X-Y THEN GOTO 2ØØ
```

Or division:

```
4Ø X=INT(1Ø1*RND(1))
5Ø Y=INT(1Ø*RND(1))+1
55 IF X<Y THEN 4Ø
56 IF INT(X/Y)<>X/Y THEN 4Ø
6Ø PRINT "WHAT IS ";X;"/";Y;
8Ø IF G=X/Y THEN GOTO 2ØØ
```

Main ideas: Creating a data list
Reading the data to print a selective list

Entering data lines takes a long time, and it's easy to make mistakes. Find a good friend, and work on this project together. You can check each other as you go along. Here are some tips:

1. Enter the data one line at a time, beginning at line 400. (The first one is done for you.)
2. Count by 10s when you number the data lines (400, 410, 420, etc.).
3. Remember to start each data line with the word DATA.
4. Put a comma after each piece of data, but not at the end of the line.
5. Always enter the data in the same order: state, abbreviation, capital, flower, bird. For example:

```
4ØØ DATA ALABAMA,AL,MONTGOMERY,CAMELLIA,YELLOWHAMMER
```

6. Run the program after you enter each line to check for errors.

When you finish, you should have a listing of all 50 states. (If you get tired, you can stop after 10 or 20 states are entered. Add the rest next week when your typing fingers recover.) Later we will change the program so it does different things.

Here is an explanation of the basic program:

5	Dimensions the string variables (tells how many letters may be used for each variable).
10	Clears the screen.
40	Reads the data in this order: ST$ (name of the state), AB$ (abbreviation), C$ (capital), F$ (flower), B$ (bird).
50	Checks for the flag data to end the program.
60	Prints the name of the state.
70	Goes back to line 40 to read more data.
400–999	The data lines that hold the information about each state. (Follow the pattern in line 400.)
1000	The flag data that signals the program to end. Line 40 is waiting to read 5 words, so write the word END 5 times.

To see a little of your list at a time, press BREAK to stop the list as it scrolls up the screen. Then type "CONT" to see a little more.

Save this program so you can use the data over and over again in new ways.

If you are using a cassette recorder, follow the directions in the Appendix on page 54.

State the States

In social studies you are studying the United States. For extra credit, you and your best friend have volunteered to enter this data into your classroom computer. Good luck!

```
5 DIM ST$(2Ø),AB$(2),C$(2Ø),F$(2Ø),B$(2Ø)
1Ø GR.Ø
4Ø READ ST$,AB$,C$,F$,B$
5Ø IF ST$="END" THEN END
6Ø PRINT ST$
7Ø GOTO 4Ø
4ØØ DATA ALABAMA,AL,MONTGOMERY,CAMELLIA,YELLOWHAMMER
  ı
  ı
  ı   (You add the rest of the data here!)
  ı
  ↓
1ØØØ DATA END,END,END,END,END
```

INFORMATION ABOUT THE STATES

ABBREVIATION	STATE	CAPITAL	FLOWER	BIRD
AL	ALABAMA	MONTGOMERY	CAMELLIA	YELLOWHAMMER
AK	ALASKA	JUNEAU	FORGET-ME-NOT	GROUSE
AZ	ARIZONA	PHOENIX	CACTUS BLOSSOM	CACTUS WREN
AR	ARKANSAS	LITTLE ROCK	APPLE BLOSSOM	MOCKINGBIRD
CA	CALIFORNIA	SACRAMENTO	POPPY	QUAIL
CO	COLORADO	DENVER	COLUMBINE	BUNTING
CT	CONNECTICUT	HARTFORD	MOUNTAIN LAUREL	ROBIN
DE	DELAWARE	DOVER	PEACH BLOSSOM	BLUE HEN
FL	FLORIDA	TALLAHASSEE	ORANGE BLOSSOM	MOCKINGBIRD
GA	GEORGIA	ATLANTA	ROSE	THRASHER
HI	HAWAII	HONOLULU	HIBI$CUS	GOOSE
ID	IDAHO	BOISE	SYRINGA	BLUEBIRD
IL	ILLINOIS	SPRINGFIELD	VIOLET	CARDINAL
IN	INDIANA	INDIANAPOLIS	PEONY	CARDINAL
IA	IOWA	DES MOINES	ROSE	GOLDFINCH
KS	KANSAS	TOPEKA	SUNFLOWER	MEADOWLARK
KY	KENTUCKY	FRANKFORT	GOLDENROD	CARDINAL
LA	LOUISIANA	BATON ROUGE	MAGNOLIA	PELICAN
ME	MAINE	AUGUSTA	PINE CONE	CHICKADEE
MD	MARYLAND	ANNAPOLIS	BLACKEYED SUSAN	ORIOLE
MA	MASSACHUSETTS	BOSTON	MAYFLOWER	CHICKADEE
MI	MICHIGAN	LANSING	APPLE BLOSSOM	ROBIN
MN	MINNESOTA	ST.PAUL	LADY SLIPPER	LOON
MS	MISSISSIPPI	JACKSON	MAGNOLIA	MOCKINGBIRD
MO	MISSOURI	JEFFERSON CITY	HAWTHORN	BLUEBIRD
MT	MONTANA	HELENA	BITTERROOT	MEADOWLARK
NB	NEBRASKA	LINCOLN	GOLDENROD	MEADOWLARK
NV	NEVADA	CARSON CITY	SAGEBRUSH	BLUEBIRD
NH	NEW HAMPSHIRE	CONCORD	LILAC	PURPLE FINCH
NJ	NEW JERSEY	TRENTON	VIOLET	GOLDFINCH
NM	NEW MEXICO	SANTA FE	YUCCA	ROADRUNNER
NY	NEW YORK	ALBANY	ROSE	BLUEBIRD
NC	NORTH CAROLINA	RALEIGH	DOGWOOD	CARDINAL
ND	NORTH DAKOTA	BISMARCK	ROSE	MEADOWLARK
OH	OHIO	COLUMBUS	CARNATION	CARDINAL
OK	OKLAHOMA	OKLAHOMA CITY	MISTLETOE	FLYCATCHER
OR	OREGON	SALEM	OREGON GRAPE	MEADOWLARK
PA	PENNSYLVANIA	HARRISBURG	MOUNTAIN LAUREL	GROUSE
RI	RHODE ISLAND	PROVIDENCE	VIOLET	RED HEN
SC	SOUTH CAROLINA	COLUMBIA	JESSAMINE	WREN
SD	SOUTH DAKOTA	PIERRE	PASQUEFLOWER	PHEASANT
TN	TENNESSEE	NASHVILLE	IRIS	MOCKINGBIRD
TX	TEXAS	AUSTIN	BLUEBONNET	MOCKINGBIRD
UT	UTAH	SALT LAKE CITY	LILY	GULL
VT	VERMONT	MONTPELIER	CLOVER	THRUSH
VA	VIRGINIA	RICHMOND	DOGWOOD	CARDINAL
WA	WASHINGTON	OLYMPIA	RHODODENDRON	GOLDFINCH
WV	WEST VIRGINIA	CHARLESTON	RHODODENDRON	CARDINAL
WI	WISCONSIN	MADISON	VIOLET	ROBIN
WY	WYOMING	CHEYENNE	PAINTBRUSH	MEADOWLARK

Main ideas: Using IF...THEN to find information on the data list
Using the data list to make a chart
Using the data list to play a game

To find information:
Simply add an IF...THEN statement at line 60. Try the examples, and make up some of your own.

To make a chart:

20 Prints the headings.
30 Underlines the headings.
60 Tells the computer what to print.

To play a game:

6 Allows 20 letters for your answer.
60–62 Prints a question for you to answer.
63 Prints "RIGHT!" if your answer was correct.
64 Prints a "WRONG" message if your answer was incorrect.

To score the game:

50 Prints the score at the end of the game. Since there are 50 states and you get 2 points for each correct answer, your maximum score is 100.
63 Keeps track of the number correct.

Trivia Expert

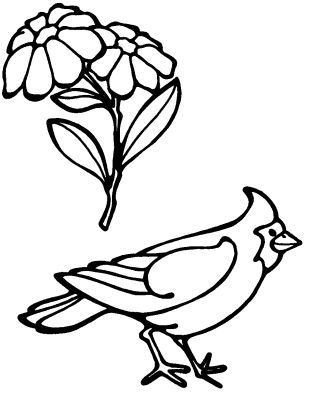
Let the computer turn you into a trivia expert. Who else could possibly know the state bird of Wyoming? Now that you put in all that data, let the computer help you.

To find information:

```
6Ø IF ST$="WYOMING" THEN PRINT B$

6Ø IF ST$="CONNECTICUT" THEN PRINT C$

6Ø IF AB$="CT" THEN PRINT C$

6Ø IF AB$="AL" THEN PRINT B$

6Ø IF B$="ROBIN" THEN PRINT ST$

6Ø IF F$="ROSE" THEN PRINT ST$
```

To make a chart:

```
2Ø PRINT "STATE","CAPITAL"
3Ø PRINT "-----","-------"
6Ø PRINT ST$,C$

2Ø PRINT "STATE","FLOWER"
3Ø PRINT "-----","------"
6Ø PRINT ST$,F$

2Ø PRINT "STATE","BIRD"
3Ø PRINT "-----","----"
6Ø PRINT AB$,B$
```



To play a game:

```
6 DIM A$(2Ø)
6Ø PRINT "THE STATE IS ";ST$
61 PRINT "WHAT IS THE ABBREVIATION?";
62 INPUT A$
63 IF A$=AB$ THEN PRINT "RIGHT!"
64 IF A$<>AB$ THEN PRINT "WRONG, IT WAS ";AB$
```

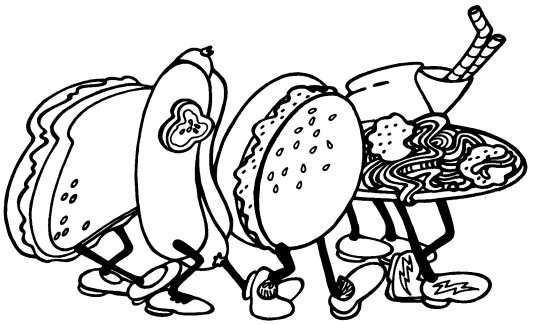
To score the game:

```
5Ø IF ST$="END" THEN PRINT "YOUR SCORE IS ";C*2:END
63 IF A$=AB$ THEN PRINT "RIGHT!":C=C+1
```

Main ideas: A contest or race decided at random
String variables and numeric variables

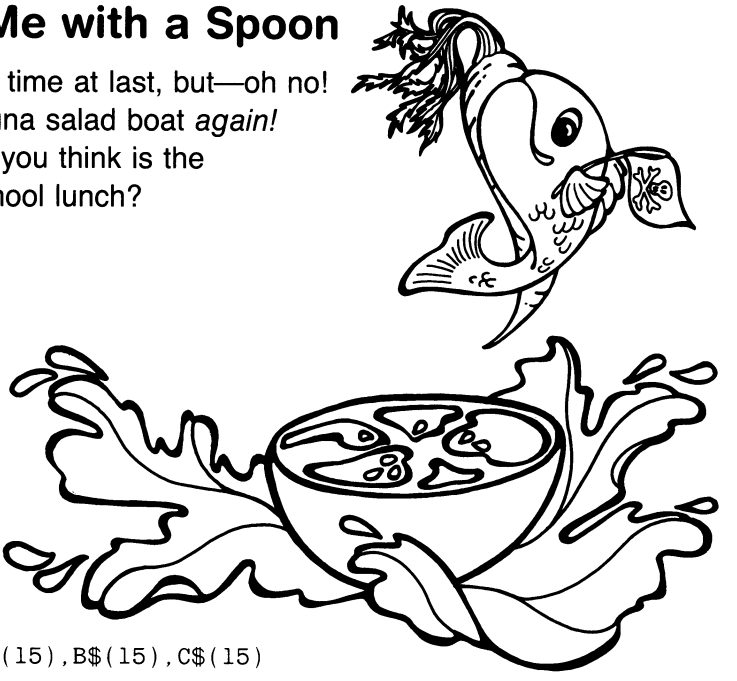
Remember that A$, B$, and C$ are words (the lunches) and that A, B, and C are numbers (the positions of the lunches on the screen).

5 A$, B$, and C$ may have up to 15 letters each.
10 Clears the screen.
20–50 Allows you to input your choices for the 3 worst lunches.
110 Clears the screen.
115 Gets rid of the cursor.
120–130 Positions and prints the name of the race.
140–160 Prints the 3 worst lunches.
170 At the beginning of the race, all 3 lunches start at position 1. Did you notice that the lunches are string variables (A$, B$, C$), but that their positions are numeric variables (A, B, C)? For example, A is the number of the box where A$ is located.
180 Picks a number from 1 to 3 as the vote for the worst lunch.
190 If it picked a 1, A$ gets the vote, and a star is printed at location (A,6) right below the A$ label. A goes up 1, so that next time the second box will be filled in, then the third box, and so forth.
200 This is what happens if the computer picked a 2 in line 180.
210 This is what happens if the computer picked a 3 in line 180.
220 If any of the "competing" lunches gets to box 40 (off the screen), then the race is over, and the program goes to line 250.
230 Goes back to line 180 to see which lunch gets the next vote.
250 Prints this message at the bottom of the screen when there is a winner—or should I say "loser"?
260 Freezes the screen so there is no READY signal.



Gag Me with a Spoon

It's lunch time at last, but—oh no! It's the tuna salad boat *again!* What do you think is the worst school lunch?



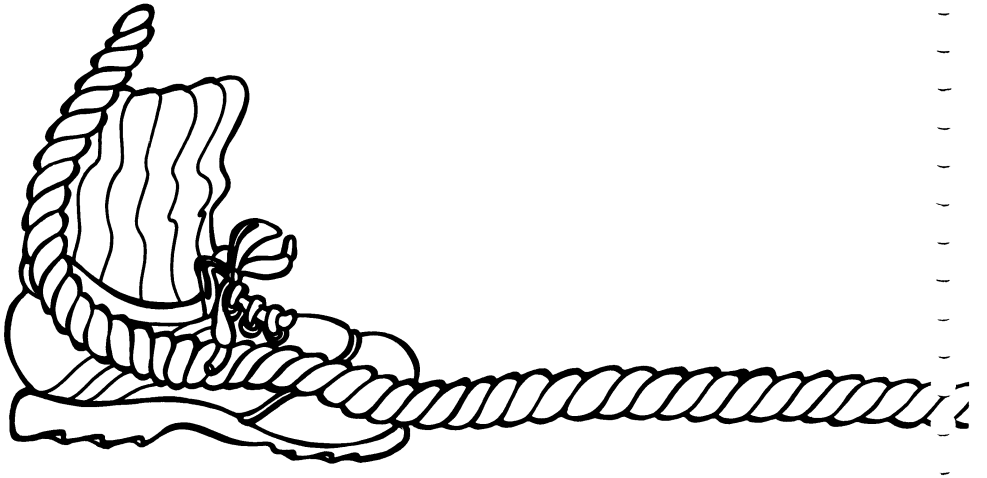
```
5 DIM A$(15),B$(15),C$(15)
10 GR.0
20 PRINT "WHAT ARE THE 3 WORST SCHOOL LUNCHES?"
30 INPUT A$
40 INPUT B$
50 INPUT C$
110 GR.0
115 POKE 752,1
120 POS.12,2
130 PRINT "THE WORST LUNCH"
140 POS.1,5:PRINT A$
150 POS.1,10:PRINT B$
160 POS.1,15:PRINT C$
170 A=1:B=1:C=1
180 V=INT(3*RND(1))+1
190 IF V=1 THEN POS.A,6:PRINT "*":A=A+1
200 IF V=2 THEN POS.B,11:PRINT "#":B=B+1
210 IF V=3 THEN POS.C,16:PRINT "%":C=C+1
220 IF A=40 OR B=40 OR C=40 THEN 250
230 GOTO 180
250 POS.2,21:PRINT "LET'S DROP IT FROM THE SCHOOL
    MENU!"
260 GOTO 260
```

Main ideas: Dimensioning string variables
Inserting variables into a text
Pressing any key to continue

In this program, the following variables are used:

L$ = Letter of the alphabet
G$ = Girl's name
B$ = Boy's name
PL$ = Place
TH$ = Thing

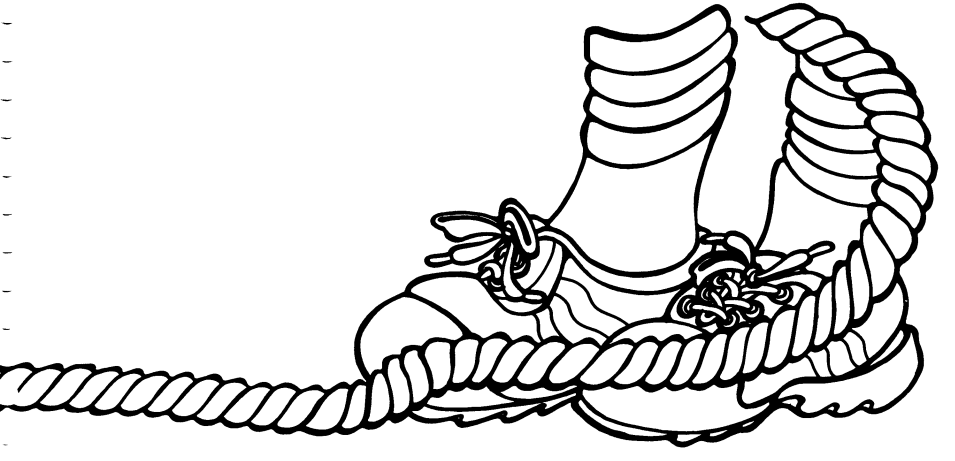
5 Allows a certain number of letters for each string variable used in the program. For example, L$ may have only 1 letter, B$ may have up to 15 letters, and so forth.
6 Clears the screen.
10–100 Allows you to input your ideas.
110 Clears the screen.
120–150 Prints the poem. Notice the semicolons. They make the computer print the next word right next to the last word (or space) printed.
155–160 Positions and prints the message.
170 Opens the keyboard for input.
180 Gets the code number of any key you press. (The number doesn't really matter because the program automatically continues to line 190 as soon as you press a key, whatever it is.)
190 Sends you back to the beginning of the program at line 6. Notice that the program does not go back to line 5. Doing so would produce an error because the string variables can be dimensioned only once during the run of a program.



Jump Rope

After lunch the girls like to jump rope. (I guess the boys don't know how!) This is one of their favorite poems. Why not make up some verses of your own?

```
5 DIM L$(1),G$(15),B$(15),PL$(2Ø),TH$(2Ø)
6 GR.Ø
1Ø PRINT "LETTER OF ALPHABET";
2Ø INPUT L$
3Ø PRINT "GIRL'S NAME THAT BEGINS WITH ";L$
4Ø INPUT G$
5Ø PRINT "BOY'S NAME THAT BEGINS WITH ";L$
6Ø INPUT B$
7Ø PRINT "NAME OF PLACE BEGINNING WITH ";L$
8Ø INPUT PL$
9Ø PRINT "SOMETHING THAT BEGINS WITH ";L$;" (PLURAL)"
1ØØ INPUT TH$
11Ø GR.Ø
12Ø PRINT L$;" MY NAME IS ";G$
13Ø PRINT "AND MY HUSBAND'S NAME IS ";B$
14Ø PRINT "WE COME FROM ";PL$
15Ø PRINT "AND WE BRING BACK ";TH$
155 POS.2,2Ø
16Ø PRINT "PRESS ANY KEY TO CONTINUE"
17Ø OPEN #1,4,Ø,"K"
18Ø GET #1,N
19Ø GOTO 6
```



Main ideas: Using OPEN...GET for instant input
Drawing without joysticks

10 Graphics 3 screen.
15 Prints this message in the text window.
20 Makes the background green.
30 The drawing will be in gold (color 1). To change this color, you may use:

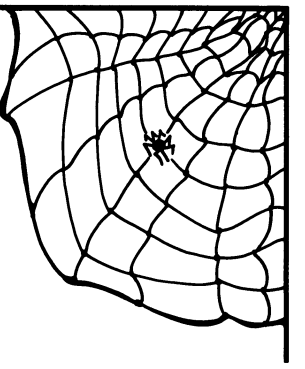
```
SETCOLOR Ø,__,__
```

A number from 0 to 15 for the setcolor

A number from 0 to 15 for the shade

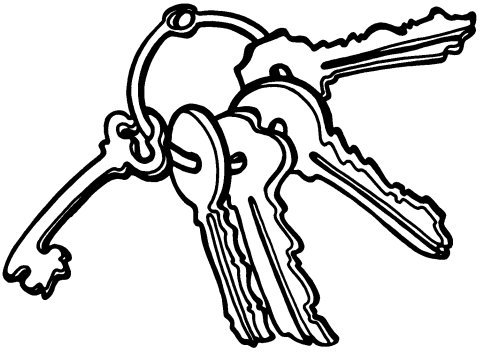
40 Opens the keyboard for instant input.
50 Your starting position on the screen.
55–56 Picks the location (H,V) where Joe's keys are hiding.
60 Plots your current location.
70 Checks to see if you have pressed a key. If so, the computer gets the ASCII code number of the key.
80 If you pressed ↑ on the − key (code 45), the computer subtracts 1 from Y and moves you up the screen.
90 If you pressed ↓ on the = key (code 61), the computer adds 1 to Y and moves you down the screen.
100 If you pressed → on the * key (code 42), the computer adds 1 to X and moves you to the right.
110 If you pressed ← on the + key (code 43), the computer subtracts 1 from X and moves you to the left.
120–150 These lines keep you on the screen.
155 Checks to see if you're close enough to find Joe's keys. To make the game harder, make the number less than 3. To make it easier, give yourself a bigger radius, and make the number more than 3.
160 Goes back to line 60 to plot your current location.

Joe's Keys



Joe, the custodian, is crawling
around on his hands and knees.
He has lost his keys.
Poor Joe needs
all the help he can get.
Why don't you help
Joe find his keys?

```
1Ø GR.3
15 PRINT "HELP JOE FIND HIS KEYS!"
2Ø SE.4,12,2
3Ø C.1
4Ø OPEN #1,4,Ø,"K"
5Ø X=2Ø:Y=19
55 H=INT(4Ø*RND(1))
56 V=INT(2Ø*RND(1))
6Ø PL.X,Y
7Ø GET #1,N
8Ø IF N=45 THEN Y=Y-1
9Ø IF N=61 THEN Y=Y+1
1ØØ IF N=42 THEN X=X+1
11Ø IF N=43 THEN X=X-1
12Ø IF X<Ø THEN X=Ø
13Ø IF X>39 THEN X=39
14Ø IF Y<Ø THEN Y=Ø
15Ø IF Y>19 THEN Y=19
155 IF X<H+3 AND X>H-3 AND Y<V+3 AND Y>V-3 THEN
    PRINT "YOU FOUND THEM!!":GOTO 155
16Ø GOTO 6Ø
```



Main ideas: A multiple-choice test
Scoring the test

5 Dimensions the string variables (the animal names). They may have up to 15 letters.
10 Graphics 2 screen.
19–80 Prints the answer choices on the screen.
100 A remark.
120 Reads the name of an animal (AN$) and its classification number (CL).
121 Checks for the word END, which is the signal to end the program.
130–140 Prints the question.
150 Allows you to input your guess (G).
160 Prints "RIGHT!" if your guess was correct.
170 Prints a "WRONG" message and tells you the right answer if your guess was incorrect.
175–180 Pauses and skips a line.
190 Goes back to line 120 to read the next animal.
300–310 Data lines.
400 Flag data.

To score the game:

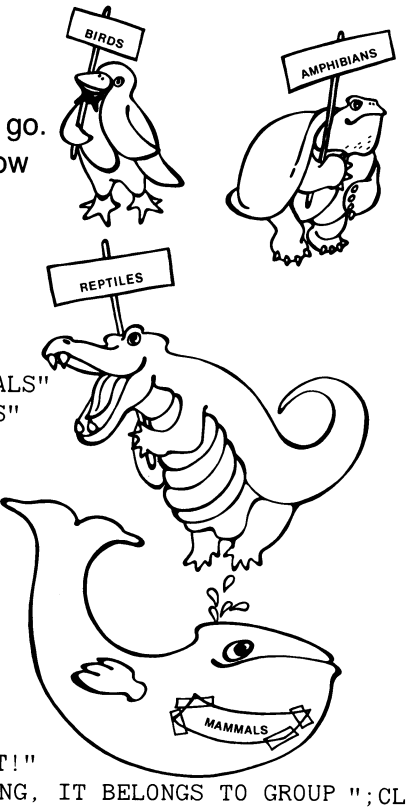
110 At the start of the game, both the total number of questions (T) and the number of correct responses (C) equal 0.
121 Sends you to line 200 for the score.
155 Adds 1 to the total (T) each time you input a guess.
160 Adds 1 to the number correct (C) each time you give a right answer.
200–220 Computes and prints the score:
210 The formula to compute your score.
215 Prints the number of questions you answered correctly.
220 Prints the percentage you got right.
230 Keeps the scoring information in the text window.

The data lines 300 and 310 hold the information for the test. You may use lines between 310 and 400 to add more data if you want to make the test longer. If you choose to add more data, be sure to follow the data pattern. You must type the animal, the number, the animal, the number, etc.

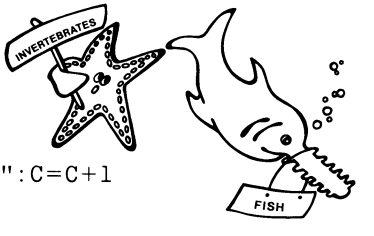
Line 400 is the flag data that signals the program to end (see line 121). Be sure to include the flag data at the end of your data list. Since line 120 has the computer read a word and a number, the flag data must contain a word and a number.

Science

Lunch time is over. The bell rings, and it's back to class you go. It's time for science. Did you know that all animals belong to one of these six groups?



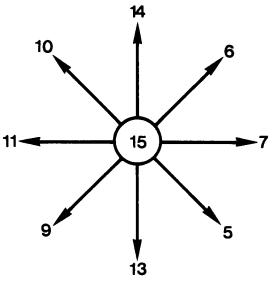

```
5 DIM AN$(15)
1Ø GR.2
19 POS.1,1
2Ø PRINT #6;"CLASSES OF ANIMALS"
3Ø PRINT #6;" 1.INVERTEBRATES"
4Ø PRINT #6;" 2.FISH"
5Ø PRINT #6;" 3.AMPHIBIANS"
6Ø PRINT #6;" 4.REPTILES"
7Ø PRINT #6;" 5.BIRDS"
8Ø PRINT #6;" 6.MAMMALS"
1ØØ REM GAME
12Ø READ AN$,CL
121 IF AN$="END" THEN END
13Ø PRINT "PICK A NUMBER"
14Ø PRINT "WHAT IS A ";AN$
15Ø INPUT G
16Ø IF G=CL THEN PRINT "RIGHT!"
17Ø IF G<>CL THEN PRINT "WRONG, IT BELONGS TO GROUP ";CL
175 FOR P=1 TO 5ØØ:NEXT P
18Ø PRINT
19Ø GOTO 12Ø
3ØØ DATA DOG,6,TURTLE,4,STARFISH,1,WHALE,6,TURKEY,5,
    FROG,3,SALMON,2
31Ø DATA SALAMANDER,3,LIZARD,4,PEACOCK,5,BAT,6,PERSON,
    6,SPIDER,1
4ØØ DATA END,Ø

11Ø T=Ø:C=Ø
121 IF AN$="END" THEN 2ØØ
155 T=T+1
16Ø IF G=CL THEN PRINT "RIGHT!":C=C+1
2ØØ REM SCORE
21Ø SC=INT(C/T*1ØØ+.5)
215 PRINT "YOU GOT ";C;" CORRECT OUT OF ";T
22Ø PRINT "YOUR SCORE IS ";SC;"%"
23Ø GOTO 23Ø
```



Main ideas: Using the joystick to draw on the screen
Selecting colors from the Setcolor Chart

For understanding best how this program works, enter it in sections. Before you start, try this short program, and see if you can find all the numbers on the diagram:

```
1 PRINT STICK(Ø)
2 GOTO 1
```



Now type "NEW" and start the program.

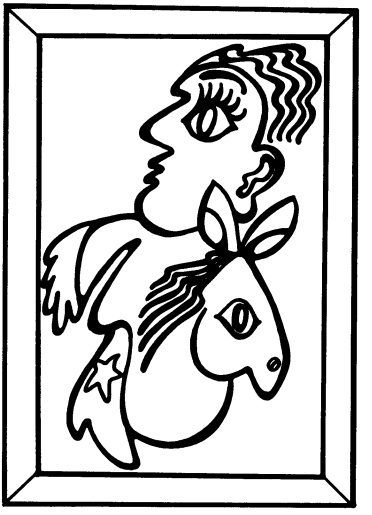
10 Graphics 7 is 160 × 96 boxes. The +16 removes the text window.
20–21 Picks the starting point.
25 Color 1 is gold.
30 Plots the starting point.
40–70 Checks the position of the joystick, and changes X and Y accordingly.
170 Goes back to line 30 to plot the next point.

80–110 Allows you to draw on the diagonal.
120–150 Keeps you from going off the screen. (If you do not add these lines, you will get an error message if you try to go off the screen!)

Selecting your own colors will make the drawing even better. To choose your colors, refer to the Setcolor Chart in the Appendix.

1–9 Allows you to input a color and a shade for the background, and a color and a shade for the drawing.
26 Sets the color of the background.
27 Sets the color of the drawing.

Modern Art



The art teacher is absent, and the sub lets you draw whatever you like. Maybe you can exhibit your work at the Modern Art Museum!

```
10 GR.7+16
20 X=INT(160*RND(1))
21 Y=INT(96*RND(1))
25 COLOR 1
30 PL.X,Y
40 IF STICK(0)=7 THEN X=X+1
50 IF STICK(0)=11 THEN X=X-1
60 IF STICK(0)=13 THEN Y=Y+1
70 IF STICK(0)=14 THEN Y=Y-1
170 GOTO 30
```

Draw on the diagonal, too!

```
80 IF STICK(0)=6 THEN X=X+1:Y=Y-1
90 IF STICK(0)=5 THEN X=X+1:Y=Y+1
100 IF STICK(0)=10 THEN X=X-1:Y=Y-1
110 IF STICK(0)=9 THEN X=X-1:Y=Y+1
```

Keep the drawing on the paper!

```
120 IF X>159 THEN X=159
130 IF X<0 THEN X=0
140 IF Y>95 THEN Y=95
150 IF Y<0 THEN Y=0
```

Choose your own colors!

```
1 GR.0
2 PRINT "BACKGROUND COLOR (0-15)";
3 INPUT B
4 PRINT "SHADE OF BACKGROUND (0-15)";
5 INPUT SB
6 PRINT "COLOR OF DRAWING (0-15)";
7 INPUT D
8 PRINT "SHADE OF DRAWING (0-15)";
9 INPUT SD
26 SE.4,B,SB
27 SE.0,D,SD
```

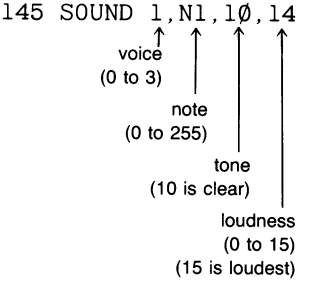
Main ideas: Using SOUND with one or two "voices"
Using OPEN...GET for immediate input
Using colons to combine statements

10 Clears the screen.
20–35 Positions and prints the directions.
40 Opens the keyboard for input.
50 Gets the ASCII code number of the key you press.
51 Add this line just in case you press a wrong key. It sends you back to line 50 and waits for you to press a key from 0 to 8.
55 If you press 0, the note becomes 0 (no sound).
60–130 If you press 1, the note becomes 121 (middle C). If you press 2, the note becomes 108 (D), and so forth. (Refer to the Musical Notes in the Appendix to see the numbers for the whole keyboard!)
140 Plays the sound.
150 Goes back to line 50 to get the number for the next note.

Additions to lines 55 to 130 can be made easily by pressing CTRL and the arrow keys (↑ →) to get to the end of each line. Then make the addition and press RETURN. Don't forget the colon (:), which is used to combine statements.

The additions are for a second "voice." When you play a song, you will now hear two notes (or voices). Line 140 plays the first voice (SOUND 0). Line 145 plays the second voice (SOUND 1). If you pressed 0, both voices play note 0, which means that the sound is turned off.

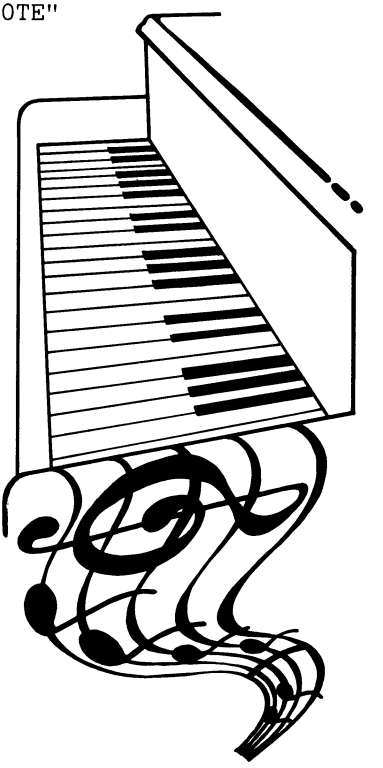
What do the numbers in a sound statement stand for?



Music

The music teacher has asked everyone to compose a tune. It's time to play your song. Do a good job!

```
10 GR.0
20 POS.5,10
30 PRINT "TO MAKE MUSIC,PRESS KEYS 1-8"
33 POS.5,11
35 PRINT "    PRESS 0 TO STOP NOTE"
40 OPEN #1,4,0,"K"
50 GET #1,K
51 IF K<48 OR K>56 THEN 50
55 IF K=48 THEN N=0
60 IF K=49 THEN N=121
70 IF K=50 THEN N=108
80 IF K=51 THEN N=96
90 IF K=52 THEN N=91
100 IF K=53 THEN N=81
110 IF K=54 THEN N=72
120 IF K=55 THEN N=64
130 IF K=56 THEN N=60
140 SOUND 0,N,10,8
150 GOTO 50
```



You'll get extra credit if you can play in octaves. To do this, make these *additions* to the lines above:

```
55 :N1=0
60 :N1=60
70 :N1=53
80 :N1=47
90 :N1=45
100 :N1=40
110 :N1=35
120 :N1=31
130 :N1=29
```

Then add this new line:

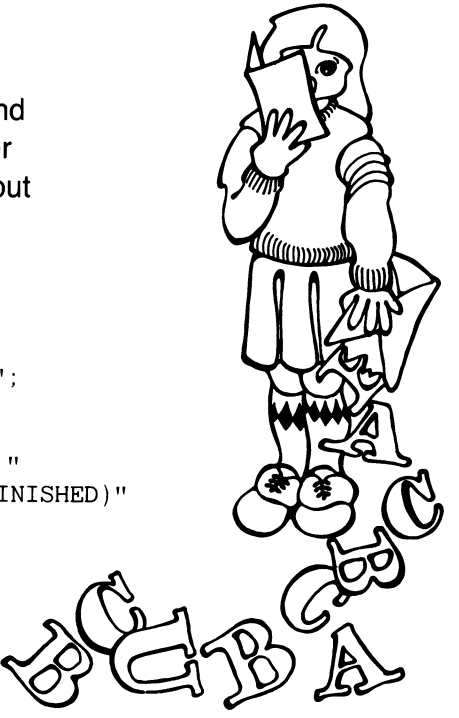
```
145 SOUND 1,N1,10,14
```

Main ideas: Using INPUT to introduce flag data
Averaging a list of numbers

2 Dimensions the student's name to 30 letters.
3 Opens the keyboard for input with the word "GET" (line 210).
5 Clears the screen.
10–20 Allows you to input the student's name.
25 Skips a line.
30–40 Prints the directions.
50 At the beginning of this exercise, both the number of tests (T) and the total points scored (TOT) equal 0.
60 Allows you to input a test score (SC).
61 Checks for the flag data. If you type "−1", this is a signal to the computer and you are sent to line 100.
70 The total number of tests (T) goes up 1.
80 The test score (SC) is added to the total points scored (TOT). For example, if the total points is 0 and you score 90 on the first test, the new sum is 90. If you score 90 on the second test, the sum becomes 180, and so forth.
90 Goes back to line 60 for the next test score.
100–150 Finds and evaluates your average:
105 The average (AV) is the total points scored (TOT), divided by the number of tests (T). For example, if the sum of all the test scores is 450 and you took 5 tests, your average is 450 ÷ 5, or 90.
110 Prints the average.
120–150 Gives you a grade. You get an A if you scored 90 or more, a B if you scored between 80 and 90, a C if you scored between 70 and 80, and a U if you scored less than 70. U is unsatisfactory!
200–210 Waits for you to press a key before starting over.
220 Goes back to line 5 to clear the screen and start a new average.

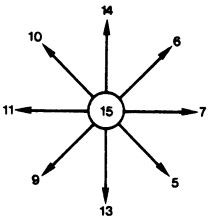
Report Card

Today is report card day. Average your test grades, and find out what you got on your report card. You can figure out your friends' grades, too!




```
2 DIM N$(3Ø)
3 OPEN #1,4,Ø,"K"
5 GR.Ø
1Ø PRINT "NAME OF STUDENT";
2Ø INPUT N$
25 PRINT
3Ø PRINT "SCORES ON TESTS:"
4Ø PRINT "(TYPE -1 WHEN FINISHED)"
5Ø T=Ø:TOT=Ø
6Ø INPUT SC
61 IF SC=-1 THEN 1ØØ
7Ø T=T+1
8Ø TOT=TOT+SC
9Ø GOTO 6Ø
1ØØ REM FIND AVERAGE
1Ø5 AV=TOT/T
11Ø PRINT "THE AVERAGE IS ";AV
12Ø IF AV>=9Ø THEN PRINT N$;" GETS AN A!":GOTO 2ØØ
13Ø IF AV>=8Ø THEN PRINT N$;" GETS A B!":GOTO 2ØØ
14Ø IF AV>=7Ø THEN PRINT N$;" GETS A C!":GOTO 2ØØ
15Ø PRINT N$;" GETS A U!"
2ØØ PRINT:PRINT "PRESS ANY KEY TO CONTINUE."
21Ø GET #1,N
22Ø GOTO 5
```

Main ideas: Using the joystick to move through "traffic"
Locating a position and using its contents
Using a flash of color as a GOSUB



10 Clears the screen.

20 At the beginning of the game, the number of times through the loop (T) is 0. Your position (P) is 20 (halfway across the screen). The number of collisions (C) is 0.

30 Gets rid of the cursor. (Leave this line out, and see the difference!)

40 STICK(0) is the first joystick (insert it in Port 1). At rest, STICK(0) equals 15. When you push up, STICK(0) equals 14; when you push down, STICK(0) equals 13; etc.

50 If you push the joystick towards the right, the computer adds 1 to your position.

60 If you push the joystick towards the left, it subtracts 1 from your position.

70–80 Keeps you on the screen.

90 Locates the position you are about to occupy on the screen, and finds the ASCII code number (N) of what is there.

100 The ASCII code for a space is 32, so if anything other than 32 is there, you are sent to line 200 to have a collision!

110 The background color becomes black (color 0, shade 0). Setcolor 2 sets the background color in Graphics 0. See the Setcolor Chart in the Appendix to select a different color.

120 Finds your position on the screen, and prints a design. (I pressed the /|\ key and the space bar to get a reversed space.)

130 Selects a random number from 0 to 38 for the box in which the car will be printed.

140 Prints a car (to make the car, I pressed the CTRL key and the semicolon simultaneously) on the bottom line of the screen. Because it's the bottom line, the screen scrolls upwards, giving the illusion of movement.

150 Adds 1 to the number of times through the loop.

160 When you've gone through the loop 300 times, the game is over. (To make the game longer, use a number larger than 300.)

170 Goes back to line 40 to check if you've moved the joystick.

180–190 Prints the score and ends the program.

200–270 Every time the computer locates something at your position besides a space, you have a collision:

205–210 Adds 1 to the number of collisions and beeps.

220–250 Makes 10 flashes of color.

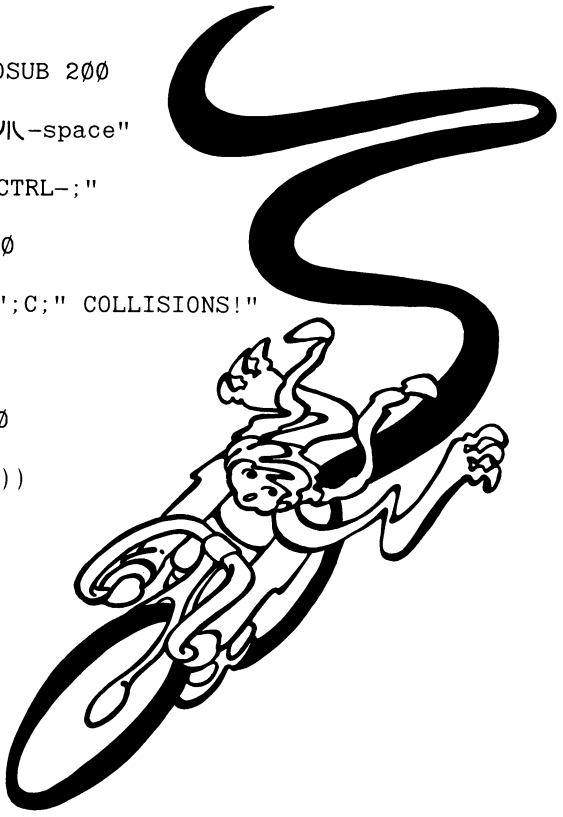
260 Turns off the beep.

270 Returns to the program.

The Ride Home

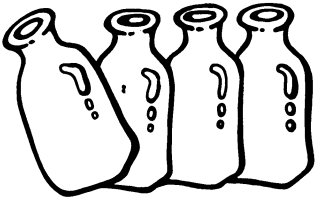
Your Mom is really going to like your report card. She'll probably make your favorite after-school treat. You decide to race home on your bicycle. Sometimes you even race against traffic. Watch out!

```
10 GR.0
20 T=0:P=20:C=0
30 POKE 752,1
40 ST=STICK(0)
50 IF ST=5 OR ST=6 OR ST=7 THEN P=P+1
60 IF ST=9 OR ST=10 OR ST=11 THEN P=P-1
70 IF P>38 THEN P=38
80 IF P<0 THEN P=0
90 LOCATE P,12,N
100 IF N<>32 THEN GOSUB 200
110 SE.2,0,0
120 POS.P,12:PRINT "ﾉﾑ-space"
130 X=INT(39*RND(1))
140 POS.X,23:PRINT "CTRL-;"
150 T=T+1
160 IF T=300 THEN 180
170 GOTO 40
180 PRINT "THAT WAS ";C;" COLLISIONS!"
190 END
200 REM COLLISION
205 C=C+1
210 SOUND 0,100,10,10
220 FOR F=1 TO 10
230 HUE=INT(16*RND(1))
240 SE.2,HUE,6
250 NEXT F
260 SOUND 0,0,0,0
270 RETURN
```



Main ideas: Counting backwards
Playing a song using READ...DATA

10 Begins a FOR...NEXT loop that will count backwards from 99 to 1. (The end of the loop is at line 60.)
20 Clears the screen.
21 Stops the cursor.
22 Picks a number from 0 to 15 for the color. (Refer to the Setcolor Chart in the Appendix on page 55.)
23 Picks a number from 0 to 15 for the shade of the color. (Since the letters are invisible if the shade is 10 or 11, the program will choose again if it picks either of those numbers.)
24 Changes the color of the text window. (In Graphics 0, the text window is the whole screen.)
25 Inserts the number for B, and positions and prints the first line of the song.
30 Reads the note (N) and the time (T) that the note gets. The time is given in number of beats.
31–34 Checks for flag data. Look at lines 500 to 530, noticing the data at the end of each line. Each number signals the computer to go to a certain line of the program, which directs it to print the next line of the song on the screen.
40 Plays the note read if it's not time to print a new line.
50 Holds the note while counting to 50 for each beat. To speed up the song, make the number less than 50. To slow up the song, make the number more than 50.
51 Cuts off the sound.
55 Goes to line 30 to read the numbers for the next note and the number of beats it gets.
60 When a verse is complete, you go back to the beginning of the loop for the next number of bottles.
100–410 The lines that relate to the flag data (see lines 31 to 34):
400 After a whole verse is played, the data for the notes is restored so it can be read again for the next verse.
410 Keeps going back for more as long as there are enough bottles.
415 Skips 2 lines.
420 Prints the question.
430 Ends the program.



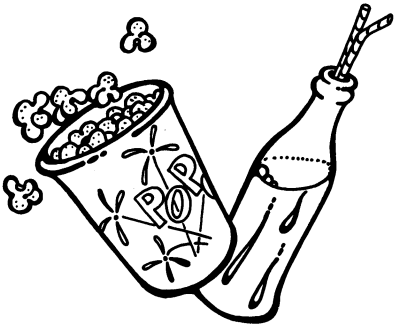
Field Trip

Today's the day of the big game. Your school is playing an arch rival. The school has provided busses to get to the game. On the bus, the kids sing this old classic.

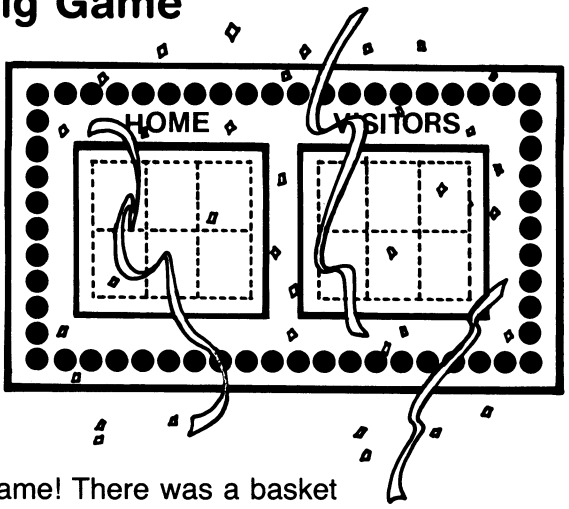
```
10 FOR B=99 TO 1 STEP-1
20 GR.0
21 POKE 752,1
22 C=INT(16*RND(1))
23 SH=INT(16*RND(1)):IF SH=10 OR SH=11 THEN 23
24 SE.2,C,SH
25 POS.3,6:PRINT B;" BOTTLES OF MILK ON THE WALL"
30 READ N,T
31 IF N=-1 THEN 100
32 IF N=-2 THEN 200
33 IF N=-3 THEN 300
34 IF N=-4 THEN 400
40 SOUND 0,N,10,8
50 FOR P=1 TO 50*T:NEXT P
51 SO.0,0,0,0
55 GOTO 30
60 NEXT B
100 POS.8,8:PRINT B;" BOTTLES OF MILK -"
110 GOTO 30
200 POS.9,10:PRINT "TAKE ONE DOWN AND"
210 POS.10,12:PRINT "PASS IT AROUND"
220 GOTO 30
300 POS.2,14:PRINT B-1;" BOTTLES OF MILK ON THE WALL!!"
310 GOTO 30
400 RESTORE
410 IF B>1 THEN GOTO 60
415 PRINT:PRINT
420 PRINT "HOW CAN THE BUSDRIVER DRIVE WITH
    HIS   FINGERS IN HIS EARS?"
430 END
500 DATA 91,1,91,1,91,1,91,1,91,1,91,1,91,1,91,
    1,91,3,-1,-1
510 DATA 81,1,81,1,81,1,108,1,108,1,108,1,81,6,-2,-2
520 DATA 96,2,96,1,96,2,96,1,96,1,96,1,96,1,96,3,-3,-3
530 DATA 121,1,121,1,121,1,108,1,108,1,96,1,91,1,91,1,
    91,1,91,3,-4,-4
```

Main ideas: Random outcome of a game
Initializing a variable and incrementing by twos

10 Dimensions the string variables (allows 20 letters for each team name).
20 Clears the screen.
30–60 Allows you to input the names of the teams.
70 Clears the screen.
80 Stops the cursor.
90–120 Positions and prints the title.
130 Initializes the scores. At the beginning of the game, your team (Y) and their team (TH) have no points.
140–200 Since there's a basket a minute during the game, this loop repeats 60 times:
150 Picks a 0 or a 1.
160 If a 0 was picked, you get 2 points.
170 If a 1 was picked, they get 2 points.
180–190 Prints the score (note the use of the semicolon and the comma).
200 Completes the loop, and goes back for another loop (line 140).
210–240 Prints a final message:
220 Prints this message if the score is tied.
230 Prints this message if the score is not tied.
240 A dummy line which keeps going to itself. This prevents a READY signal (because the program never ends).



The Big Game



What a game! There was a basket scored every minute. Here's how the game turned out:

```
10 DIM Y$(20),TH$(20)
20 GR.0
30 PRINT "THE NAME OF YOUR TEAM";
40 INPUT Y$
50 PRINT "THE NAME OF THEIR TEAM";
60 INPUT TH$
70 GR.0
80 POKE 752,1
90 POS.14,10
100 PRINT "THE BIG GAME"
110 POS.10,11
120 PRINT "**********"
130 Y=0:TH=0
140 FOR M=1 TO 60
150 B=INT(2*RND(1))
160 IF B=0 THEN Y=Y+2
170 IF B=1 THEN TH=TH+2
180 POS.10,12
190 PRINT Y$;" ";Y,TH$;" ";TH
200 NEXT M
210 POS.15,20
220 IF Y=TH THEN PRINT "REMATCH!"
230 IF Y<>TH THEN PRINT "FINAL!!"
240 GOTO 240
```

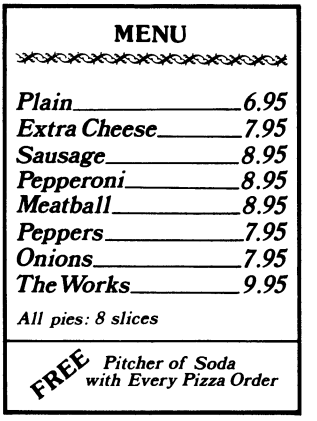
Main ideas: Setting up a chart
Rounding off figures

5 Dimensions the kind of pizza (K$) to 15 letters.
10 Clears the screen.
20–50 Allows you to input the kind of pizza and the price.
100 Clears the screen.
110–120 Positions and prints the kind of pizza and the price.
121 Figures the tip (TP) as 15% of the price.
122 Rounds the tip to the nearest 100th (as in dollars and cents).
123 Positions and prints the tip.
124 The total price equals the price of the pizza plus the tip.
125 Positions and prints the total price.
126 Skips a line.
130–150 Prints and underlines the column headings for the chart.
160–180 Figures the cost per person and the number of slices per person if there are between 1 and 8 persons sharing the pizza.
160 Starts the loop.
165 Figures the cost to the nearest 100th.
166 Figures the number of slices to the nearest 10th (all the pizzas have 8 slices).
170 Prints the number of persons, the cost per person (in dollars and cents), and the number of slices per person (in 10ths).
180 Completes the loop.

Notice the use of commas in setting up the chart. A comma sends the print action to the next print zone.

Pizza

After the game, you and some friends go out for pizza. It's hard to decide what to get. How much will it cost? How much is the tip? How much should each person chip in? How much may each person eat? Let the computer do your arithmetic.




```
5 DIM K$(15)
1Ø GR.Ø
2Ø PRINT "WHAT KIND OF PIZZA DO YOU WANT?"
3Ø INPUT K$
4Ø PRINT "HOW MUCH DOES IT COST?"
5Ø INPUT P
1ØØ GR.Ø
11Ø POS.8,1:PRINT K$;" PIZZA"
115 PRINT "------------------------------"
12Ø POS.1Ø,3:PRINT "COST IS $";P
121 TP=.15*P
122 TP=INT(1ØØ*TP+.5)/1ØØ
123 POS.1Ø,4:PRINT "TIP IS $";TP
124 P=P+TP
125 POS.1Ø,5:PRINT "TOTAL COST IS $";P
126 PRINT
13Ø PRINT "NO.OF","COST/","SLICES/"
14Ø PRINT "PERSONS","PERSON","PERSON"
15Ø PRINT "-------","------","-------"
16Ø FOR N=1 TO 8
165 C=INT(1ØØ*P/N+.5)/1ØØ
166 SL=INT(1Ø*8/N+.5)/1Ø
17Ø PRINT N,C,SL
18Ø NEXT N
```

Main ideas: Using READ...DATA to draw boxes
Using RND to make a game

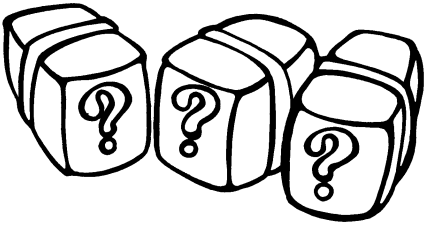
5 A remark.
13 Graphics 3 screen. (Refer to that diagram on page 50.)
14 Color 1 is gold.
15 Reads 15 X positions. (A vertical line will be drawn at each X position.)
20 Reads the first number for X.
30–50 Sets all the X points from Y=6 to Y=10.
60 Goes back to read the next X number and set the next column of points.
70 Restores the data so it can be used again.
80 The data that holds the information for the X coordinates.
100 A remark that this is the game section.
110 Picks the box for the money (box 1, 2, or 3).
120–130 Allows you to guess where the computer put the money.
140 If you guessed correctly, you skip to line 200.
150 Goes back to line 110 to hide the money again.
200 A remark that this is the flashing section.
210–250 If you find the money, 20 flashes are repeated:
220 Picks a setcolor from 0 to 15.
230 Colors the boxes in the setcolor chosen.
240 Colors the background in a different color (5 colors away).
250 Completes the loop and goes back to line 210 to flash again.
260 After the flashing, goes back to line 13 so you can play again.

If you want to bet on the game, make the suggested additions.

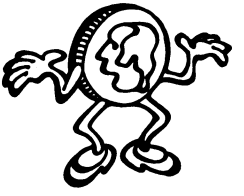
10 Clears the screen.
12 Allows you to input the amount of money you are starting out with.
115 Prints your current total.
145 Subtracts 25¢ when you lose.
146 Lets you know when you run out of money.
205 Adds 50¢ to your total when you win.

Double or Nothing

Tomorrow is the school fair. "Double or Nothing" is your favorite game. Here's how it goes: The person at the booth has three little boxes. A half dollar is hiding in one of them. Guess which one!



```
5 REM DRAW BOXES
13 GR.3
14 C.1
15 FOR P=1 TO 15
2Ø READ X
3Ø FOR Y=6 TO 1Ø
4Ø PLOT X,Y
5Ø NEXT Y
6Ø NEXT P
7Ø RESTORE
8Ø DATA 1Ø,11,12,13,14,17,18,19,2Ø,21,24,25,26,27,28
1ØØ REM GAME
11Ø M=INT(3*RND(1))+1
12Ø PRINT "WHICH DO YOU CHOOSE?";
13Ø INPUT G
14Ø IF G=M THEN 2ØØ
15Ø GOTO 11Ø
2ØØ REM FLASH
21Ø FOR F=1 TO 2Ø
22Ø C=INT(16*RND(1))
23Ø SE.Ø,C,6
24Ø SE.4,C+5,12
25Ø NEXT F
26Ø GOTO 13
```



Winning money makes the game a lot more interesting. You pay 25¢ to play. If you guess the right box, you keep the half dollar. Good luck! Let's keep track of your winnings:

```
1Ø GR.Ø
11 PRINT "HOW MUCH MONEY DO YOU HAVE?";
12 INPUT T
115 PRINT "YOU NOW HAVE $";T
145 T=T-.25
146 IF T<=Ø THEN PRINT "SORRY YOU ARE BROKE!":END
2Ø5 T=T+.5Ø
```

Main ideas: Using READ, DATA, and RESTORE
Incrementing a total

This program finds the number of days between any two dates that you select. It does so by converting each date to its numerical position in the year. For example, January 1 is day 1 in the year, January 31 is day 31, February 1 is day 32, and December 31 is day 365. The program converts each date by adding up the number of days in each month preceding a given date and then adding on the number of days in the given month. To find out what day April 10th is, we reason like this:

April 10 or 4/10 = 31 + 28 + 31 + 10 = 100

(31 + 28 + 31: days in the 3 months preceding April; 10: days in April)

If the month is January (which is the first month), you don't have to add up any months. Just use the number of the day: January 5 is day 5, and January 20 is day 20. Now let's look at the program lines individually:

10 Clears the screen.

20–30 At the beginning of the program, day 1 (D1) and day 2 (D2) are set to 0.

40–70 Allows you to input the beginning date—month and day. (Enter numbers for the months—1 for January, 2 for February, 3 for March, etc.).

90 If the month is January, you skip to line 150.

100 Counts to the number just before the given month. (If the month is 5, for example, it counts to 4; that is, if the month is May, it will add up the number of days in the first 4 months.)

110 Reads the number of days in each month (data line 500).

120 Adds the number of days to day 1.

130 Finishes the loop begun in line 100.

140 Restores the data, which will be read again to figure day 2.

150 Adds the number of the day to the total days in the previous months.

160 Prints the number of day 1.

165 Skips a line.

170–290 Figures the number of day 2.

295 Prints a blank line.

300 Subtracts to find the number of days between day 1 and day 2.

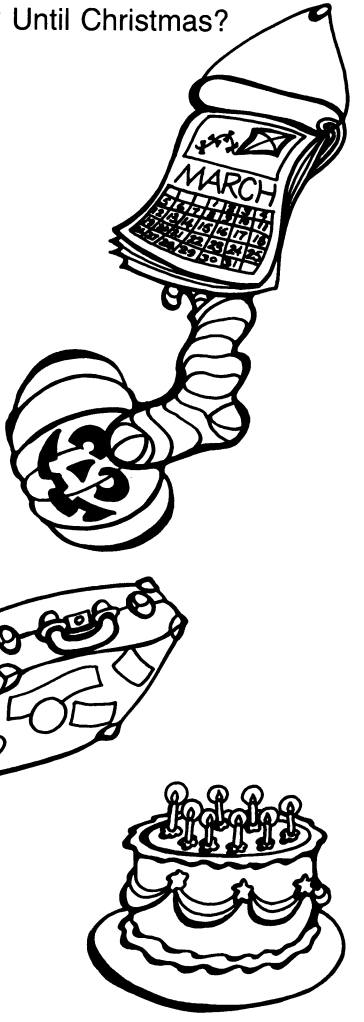
305 Fixes things up when you go over the first day of the year. For instance, without this line, the computer would say that there are −364 days from December 31 to January 1!

310 Prints the results.

500 The data line that holds the number of days in each month.

Days to Go

No matter how much fun school is, everyone likes to figure out how many days until summer vacation. While you're at it, how many days until your birthday? Until Christmas? Until Halloween? Until whenever?




```
1Ø GR.Ø
2Ø D1=Ø:D2=Ø
3Ø PRINT "HOW MANY DAYS"
4Ø PRINT "FROM MONTH";
5Ø INPUT M
6Ø PRINT "        DAY";
7Ø INPUT D
9Ø IF M=1 THEN GOTO 15Ø
1ØØ FOR X=1 TO M-1
11Ø READ N
12Ø LET D1=D1+N
13Ø NEXT X
14Ø RESTORE
15Ø LET D1=D1+D
16Ø PRINT "DAY 1 IS #";D1
165 PRINT
17Ø PRINT "  TO MONTH";
18Ø INPUT M
19Ø PRINT "        DAY";
2ØØ INPUT D
22Ø IF M=1 THEN GOTO 28Ø
23Ø FOR X=1 TO M-1
24Ø READ N
25Ø LET D2=D2+N
26Ø NEXT X
27Ø RESTORE
28Ø LET D2=D2+D
29Ø PRINT "DAY 2 IS #";D2

295 PRINT
3ØØ T=D2-D1
3Ø5 IF T<Ø THEN T=T+365
31Ø PRINT "THERE ARE ";T;" DAYS TO GO."
5ØØ DATA 31,28,31,3Ø,31,3Ø,31,31,3Ø,31,3Ø,31
```

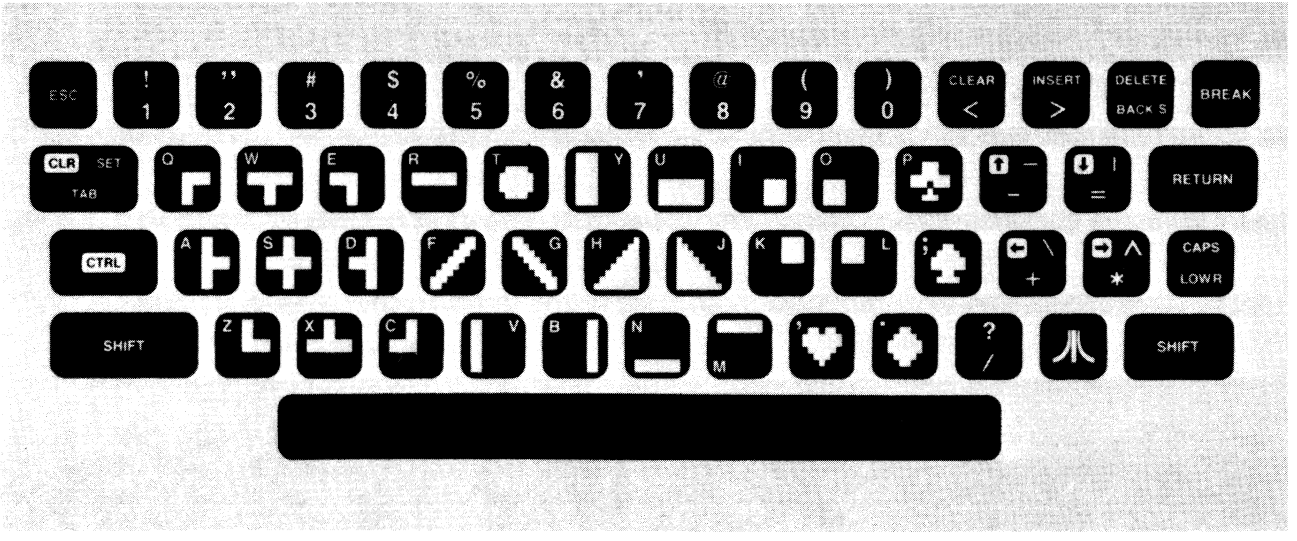
APPENDIX

Abbreviations

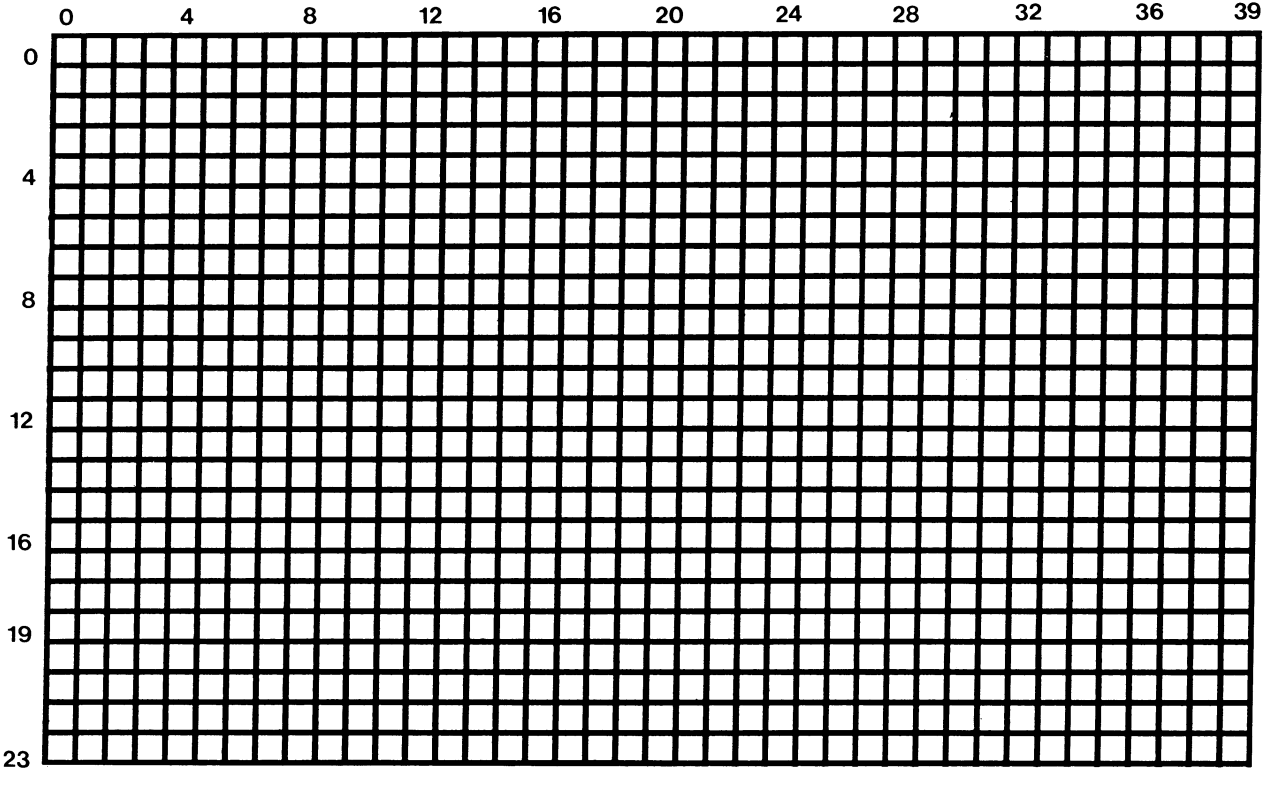
This is a partial list of the abbreviations available. Use them, but remember what they stand for!

C.	COLOR
D.	DATA
DR.	DRAWTO
F.	FOR
G.	GOTO
GOS.	GOSUB
GR.	GRAPHICS
I.	INPUT
L.	LIST
LOC.	LOCATE
N.	NEXT
PL.	PLOT
POS.	POSITION
PR.	PRINT
?	PRINT
RES.	RESTORE
RET.	RETURN
SE.	SETCOLOR
SO.	SOUND

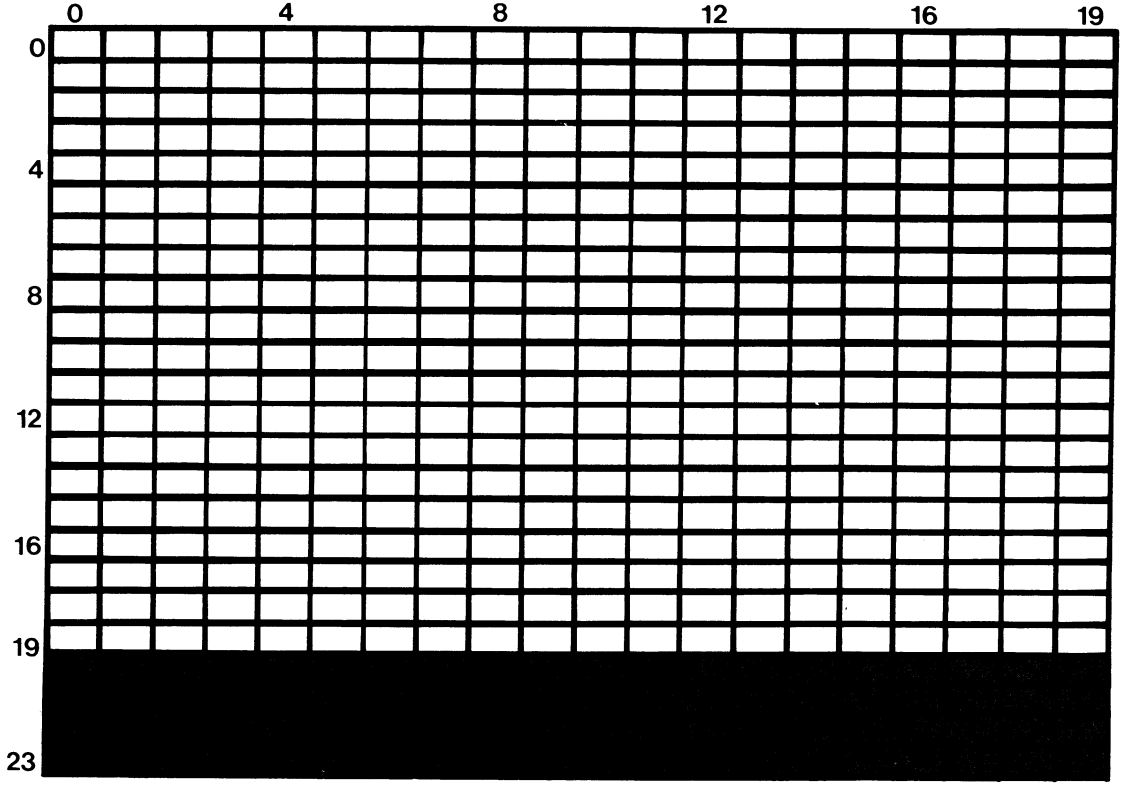
Control Graphics Keyboard



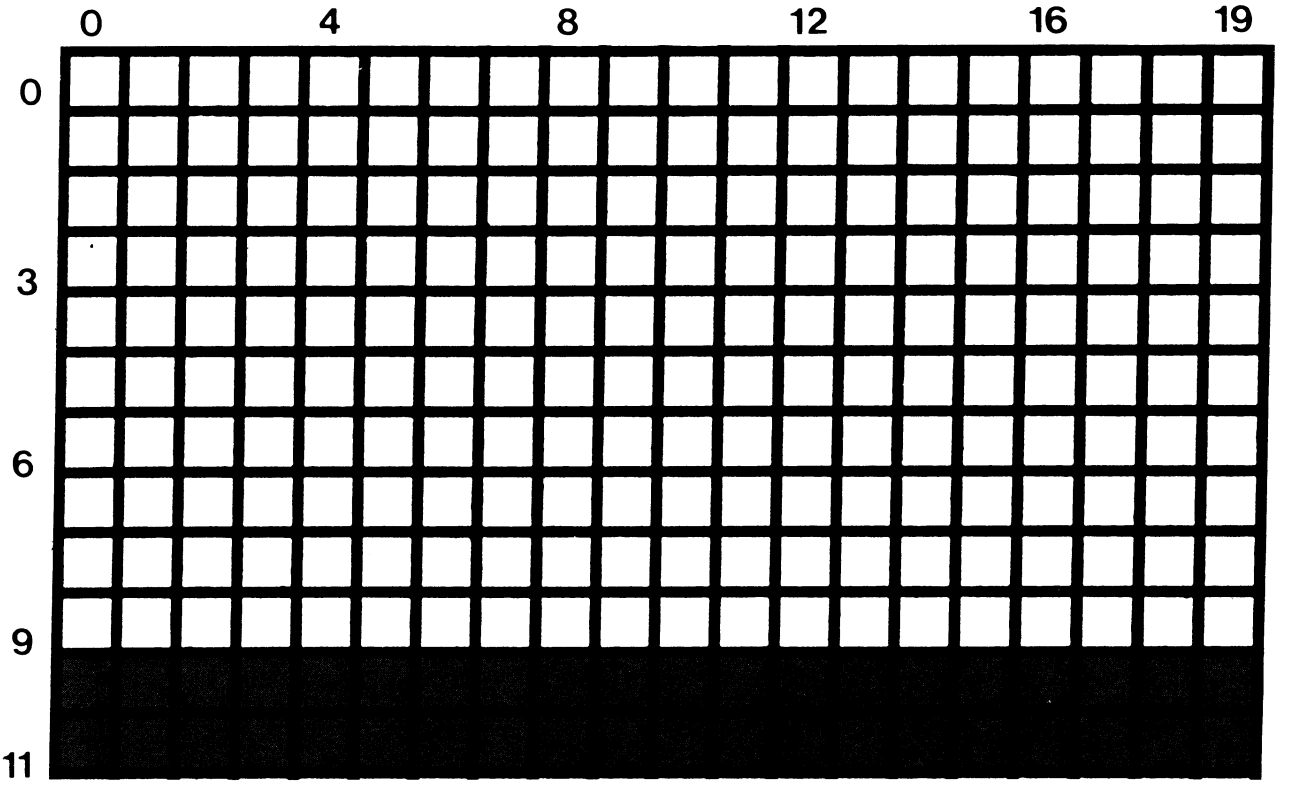
Graphics 0 Sheet



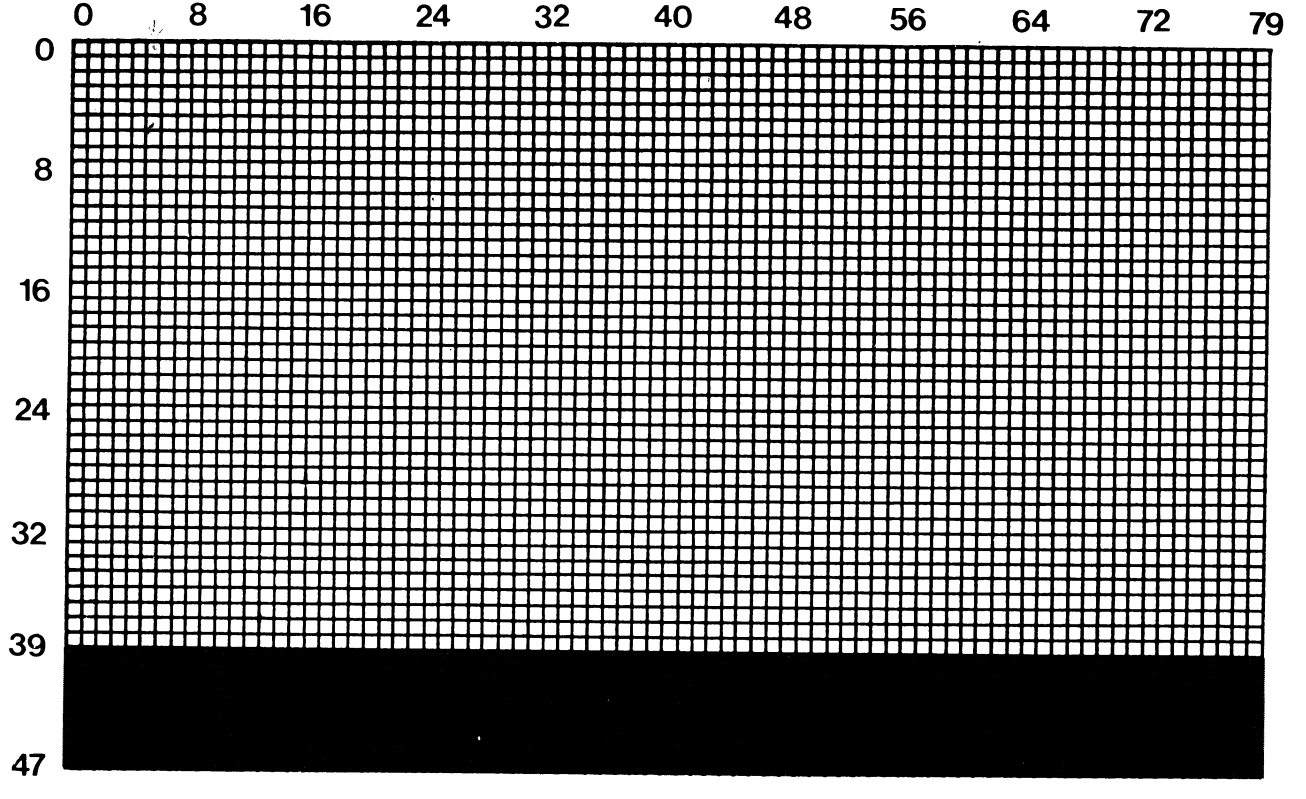
Graphics 1 Sheet



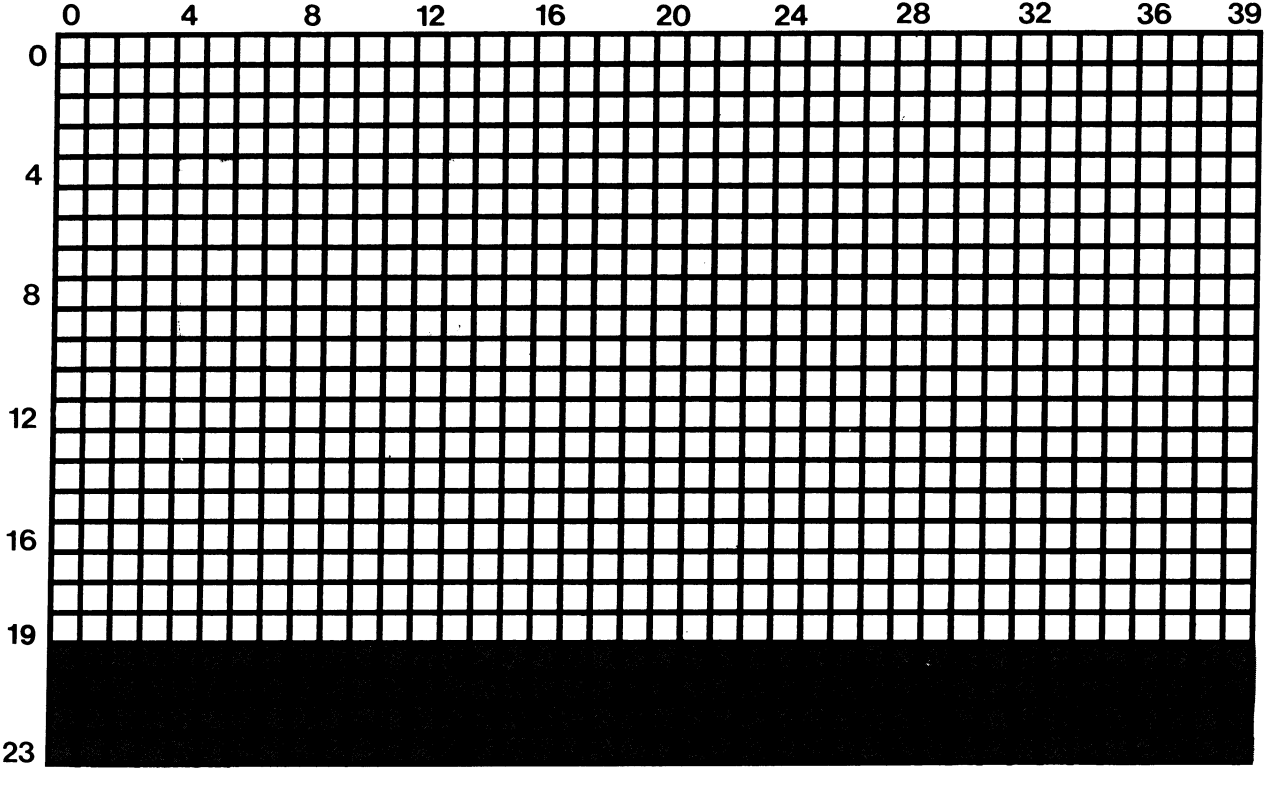
Graphics 2 Sheet



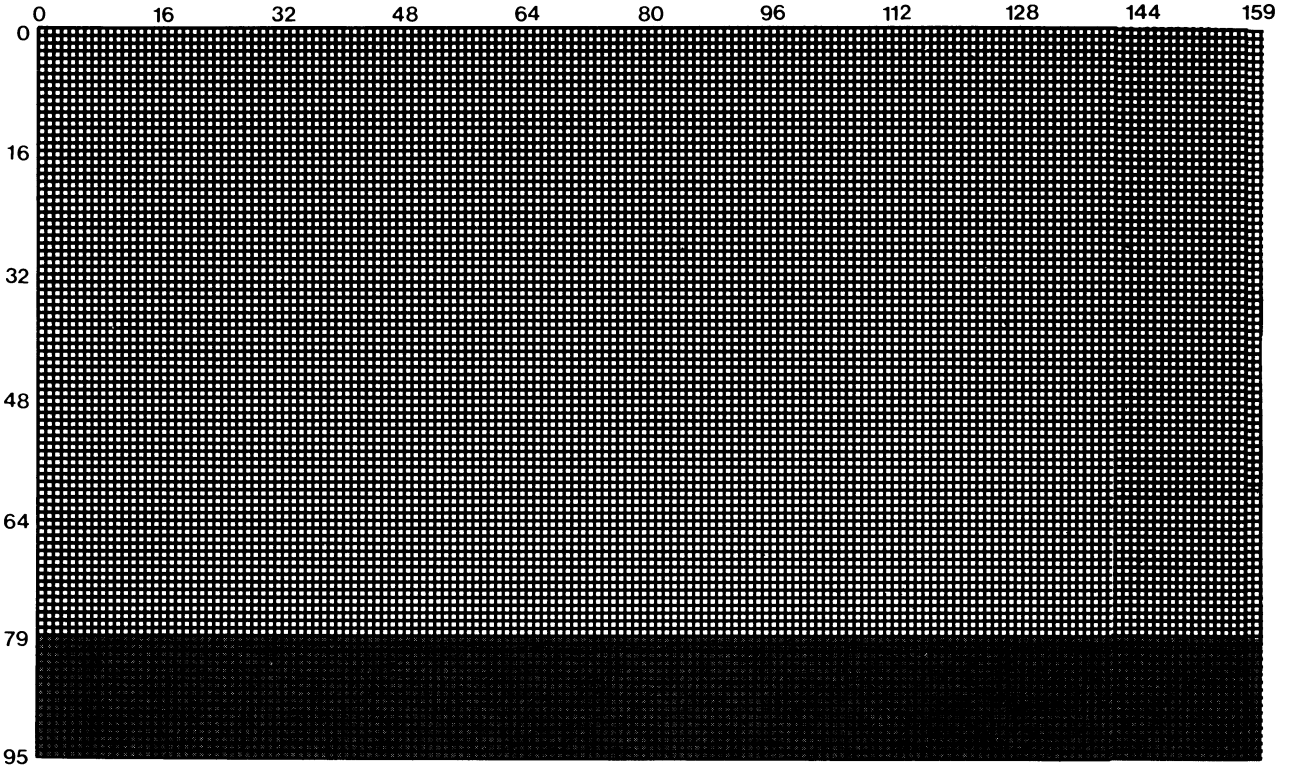
Graphics 3 Sheet



Graphics 5 Sheet

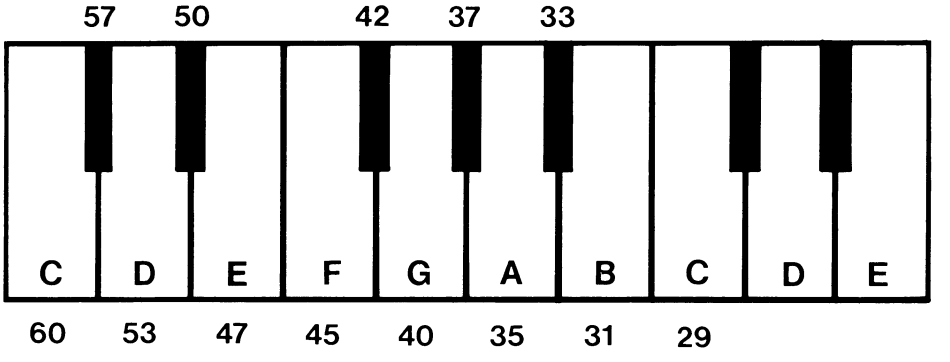


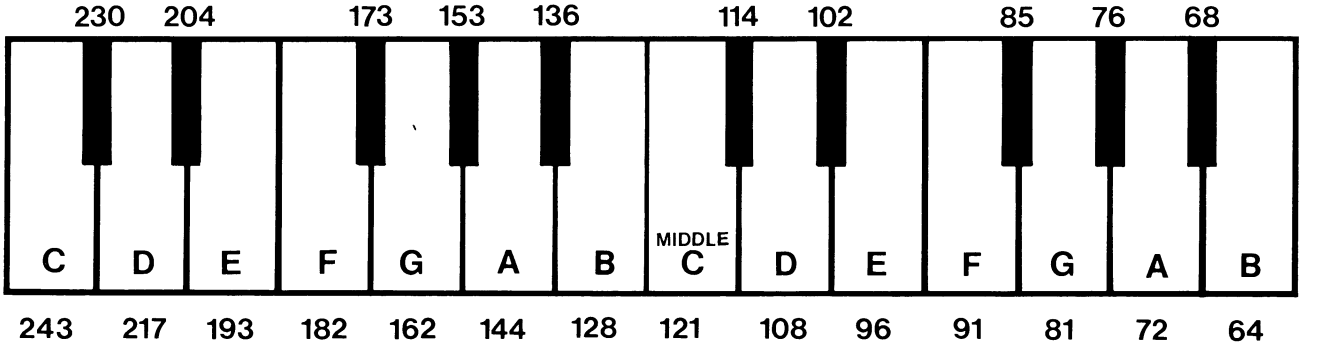
Graphics 7 Sheet



Musical Notes

Use the numbers assigned to the notes to create your own computer songs!





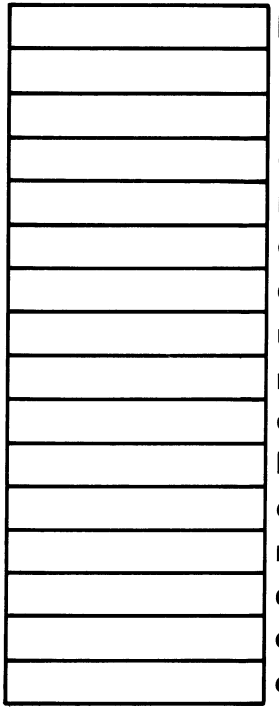
Saving Programs on Cassette

Are you having problems? I find this procedure to be more reliable than the usual CSAVE–CLOAD. It takes a little longer, but it saves a lot of time and aggravation in the long run.

1. Position the tape at a new location. (Don't tape over old programs!) Remember the number.
2. Type LIST "C:" and press RETURN.
3. The computer will buzz twice. Make sure two buttons on the cassette recorder are down, PLAY and RECORD. Then press RETURN again.
4. When the program is saved, the screen will say READY.
5. To verify that your program is saved, reposition the tape and type ENTER "C:".
6. The computer will buzz once. Make sure that the PLAY button is down. Press RETURN.
7. If you get no errors, the program is properly saved. If you do get an error message, panic not! The original program is still in memory. Find a new place on the tape and try to save it again, beginning with step 2.
8. Save your money for a disk drive.

Setcolor Chart

Color it in to help you out!



0	black
1	rust
2	red-orange
3	dark orange
4	red
5	dark lavender
6	cobalt blue
7	ultramarine blue
8	medium blue
9	dark blue
10	blue-gray
11	olive green
12	medium green
13	dark green
14	orange-green
15	orange

$6.95

CLAIRE BAILEY PASSANTINO

SCHOOL DAYS

School days, school days,
dear old Golden Rule days!

In spite of what some children say, school *is* fun! Claire Bailey Passantino has created bit-sized computer programs that allow your child to relive the exciting anticipation-filled first day of school, the often funny excuses for lost homework, challenging math and spelling quizzes . . . even the after-school basketball games.

School Days, another Itty Bitty Bytes book in the Creative Pastimes series, encourages your child to explore the magic of your home computer. The short, easy-to-do programming activities are designed to delight your youngsters' eyes, ears and mind. Each program is accompanied by programming hints and suggestions for parents and teachers.

Help your child learn, grow and play—the computer way!

Some of the activities in *School Days* include:
Clothing the Bod ★ The Dog Ate It ★ Spelling Counts! ★ State the States ★ Jump Rope ★ Science ★ Report Card ★ The Big Game ★ Field Trip ★ We Love Math

Other Itty Bitty Bytes books:
Itty Bitty Bytes of Space
Matilda the Cat

For more information on Creative Pastimes Books write to:

Reston Computer Group
Reston Publishing Company, Inc.
11480 Sunset Hills Road
Reston, Virginia 22090

0-8359-6886-3