

Sharon Boren

# A PET<sup>®</sup> In The Classroom



Also for the COMMODORE 64™ and VIC 20™

Activity Workbook

 dilithium Press



**A PET<sup>®</sup>**



**IN THE  
CLASSROOM**



**A PET<sup>®</sup>  
IN THE  
CLASSROOM  
Activity  
Workbook**

**Sharon Boren**



dilithium Press  
Beaverton, Oregon

©1983 by dilithium Press. All rights reserved.

No part of this book may be reproduced in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage and retrieval system without permission in writing from the publisher, with the following exceptions: any material may be copied or transcribed for the nonprofit use of the purchaser, and material (not to exceed 300 words and one figure) may be quoted in published reviews of this book.

Where necessary, permission is granted by the copyright owner for libraries and others registered with the Copyright Clearance Center (CCC) to photocopy any material herein for a base fee of \$1.00 and an additional fee of \$0.20 per page. Payments should be sent directly to the Copyright Clearance Center, 21 Congress Street, Salem, Massachusetts 01970.

10      9      8      7      6      5      4      3      2

**A PET in the Classroom: Activity Workbook** is part of a three-book set. As a workbook, it is not eligible to be catalogued by the Library of Congress. The following data applies to **A PET for Kids**, the principle text in the set.

#### **Library of Congress Cataloging in Publication Data**

Boren, Sharon, 1956—

A PET for kids.

Includes index.

Summary: A guide for programming the PET computer in BASIC. Designed for elementary and junior high students.

1. PET (Computer)—Programming—Juvenile literature. 2. Basic (Computer program language)—Juvenile literature. (1. Programming (Computers) 2. PET (Computer)—Programming. 3. Basic (Computer program language))

I. Title.

QA76.8.P47B673 1983 001.64'2 83-7399

ISBN 0-88056-122-X

PET is a registered trademark of Commodore Business Machines

Printed in the United States of America

dilithium Press  
8285 S.W. Nimbus  
Suite 151  
Beaverton, Oregon 97005

# TABLE OF CONTENTS

Exploring PET'S Keyboard #1-#5	1
Screen Game #1-Computer Safari	7
Screen Game#2-Computer Roundup	8
Screen Game#3-Mine the Diamonds	9
Screen Game#4-Crude-Oil Caper	10
Screen Game#5-Word Treasures	11
Keyboard Creation #1-Maze Maker	12
Keyboard Creation #2-PET	13
Keyboard Creation #3-Design Your Own	14
Component 1 Fun Page	15
Programming Your PET #1-Speak	17
Programming Your PET #2-Speaking Nonstop	18
Programming Your PET #3-Topsecret	19
Programming Your PET #4-Computer Art	20
Programming Your PET #5-Slanted Art	21
Programming Your PET #6-Shuttle Launch	22
Programmer's Pastime #1-#4	24
Component 2 Fun Page	30
Programmer's Pastime #5-#18	32
Component 3 Fun Page	54
Programmer's Pastime #19-#28	56
Component 4 Fun Page	69
Programmer's Pastime #29-#36	71
Just for Fun	94
Programmer's Pastime #37	96
Component 5 Fun Page	98
Programmer's Pastime #38-#53	100
Component 6 Fun Page	141
Programmer's Pastime #54-#72	143
Component 7 Fun Page	185

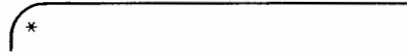




# EXPLORING PET'S KEYBOARD #1

Finish drawing the key (or keys) you must press to get PET to type what is shown on the screen.

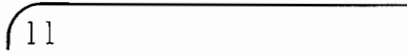
1. 



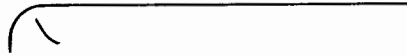
2.  



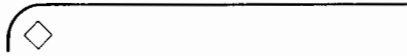
3.  



4.  



5.  



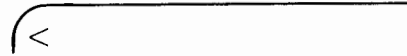
6. 



7.  



8. 



9.  













10. 



# EXPLORING PET'S KEYBOARD

## #2

Finish drawing the key (or keys) you must press to get PET to perform each special function.

1.  Move the cursor up.
2.  Send the cursor home.
3.  Slow down the screen output.
4.  Clear the screen and send the cursor home.
5.  Move the cursor to the right.
6.  End reverse-field printing.
7.  Move the cursor to the left.
8.  Delete.
9.  Lock the SHIFT.
10.  Insert a space.

# EXPLORING PET'S KEYBOARD

## #3

1. Turn PET on.
2. Press .  
What did the cursor do? \_\_\_\_\_  
\_\_\_\_\_
3. Press .  
What did the cursor do now? \_\_\_\_\_  
\_\_\_\_\_
4. Press .  
What did the cursor do? \_\_\_\_\_  
Why? \_\_\_\_\_  
\_\_\_\_\_
5. Press the  bar five times.  
What happened to the writing on the screen?  
\_\_\_\_\_  
\_\_\_\_\_
6. Press  five times.  
Which way did the cursor move? \_\_\_\_\_  
\_\_\_\_\_
7. Press  and  five times.  
Which way did the cursor move? \_\_\_\_\_  
\_\_\_\_\_
8. Press  five times.  
Which way did the cursor move? \_\_\_\_\_  
\_\_\_\_\_  
Did anything happen to the writing on the screen? \_\_\_\_\_  
\_\_\_\_\_  
This is how we move the cursor around the screen without erasing any of the writing.
9. Press  and  five times.  
Which way did the cursor move? \_\_\_\_\_  
\_\_\_\_\_



# EXPLORING PET'S KEYBOARD

## #4

1. Turn PET on.
2. Send the cursor home.  
What key did you press to make this happen?

3. Press the  bar five times.  
What happened to the writing on the screen?

---

4. Press  and watch what happens.  
What happened to the writing? \_\_\_\_\_

---

5. Make the top row of writing move all the way over to the cursor's home.  
Which key did you press?   
How many times did you press it? \_\_\_\_\_

---

6. Press  , then  .  
Press both of these keys in that order five more times.  
What happened to the writing? \_\_\_\_\_

---

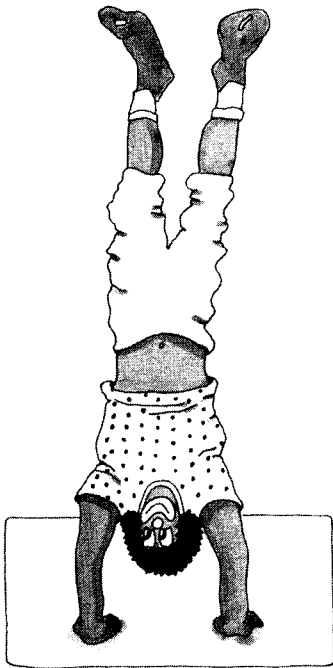
7. What is left on the screen? Copy the writing in the space below:

---

8. Press  and  .  
What happened? \_\_\_\_\_

---

Remember: this doesn't work on PETs that speak 4-Ø BASIC.



# EXPLORING PET'S KEYBOARD

## #5

1. Turn PET on.
2. Clear the screen.  
Which keys did you press to make this happen?
3. Type a ♥.  
Which keys did you press?
4. Move the cursor to the beginning of the fourth line down.  
Which key did you press?   
How many times? \_\_\_\_\_
5. Press .  
Type HELLO. How is the writing on the screen different? \_\_\_\_\_
6. Now type PET.  
Our writing doesn't look right because we didn't put a **SPACE** between HELLO and PET.
7. Press  and  three times in that order.  
Where did the cursor move? \_\_\_\_\_
8. Now press  and .  
What happened? \_\_\_\_\_

\_\_\_\_\_

This is another way we can correct typing mistakes.

9. Move the cursor to the middle of the screen.  
Which keys did you press?
10. Press  and .  
Type I ♥ PET!  
How is the writing different? \_\_\_\_\_  
\_\_\_\_\_  
Why? \_\_\_\_\_
11. Change the writing to make it say I LOVE PET!  
Use these keys only:  and , and  and .
12. Directly underneath, type I LOVE PET! in reverse field.  
Which key did you have to press before you could type?

---

## Quick Review

Tell what each key does.

1.  \_\_\_\_\_

\_\_\_\_\_

2.  \_\_\_\_\_

\_\_\_\_\_

3.  \_\_\_\_\_

\_\_\_\_\_

4.  bar \_\_\_\_\_

\_\_\_\_\_

5.  \_\_\_\_\_

\_\_\_\_\_



Tell what happens when these keys are pressed.

6.  and  \_\_\_\_\_

\_\_\_\_\_

7.  and  \_\_\_\_\_

\_\_\_\_\_

8.  and  \_\_\_\_\_

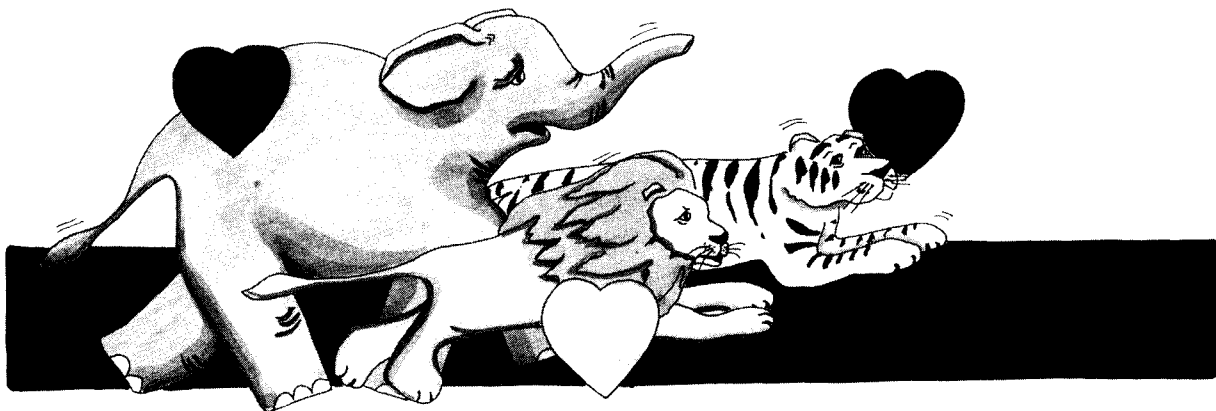
\_\_\_\_\_



# SCREEN GAME #1

## Computer Safari

1. Turn on PET.
2. Clear the screen so it is blank. What did you press to make it blank? \_\_\_\_\_
3. Press  SHIFT and hold it down. Press the A key five times. Draw what appeared on the screen. \_\_\_\_\_  
These are trees in the wild forest.
4. Type LION. Type a  . (Can you figure out how?) What did you press to type the  ? \_\_\_\_\_
5. Press  SHIFT and hold it down. Press the X key 12 times. Draw the graphic that was printed on the screen. \_\_\_\_\_
6. Press  CURSR. Now press SHIFT +  CURSR 22 times.
7. Make 10 trees appear on the screen.
8. Type ELEPHANT. Type a  .
9. Make six more trees appear on the screen.
10. Go to the next screen line below. (Follow step 6 to help you remember how.)
11. Make three trees. Type TIGER. Type a  . Make 12 more trees.
12. Go to the next screen line below.
13. THE HUNT BEGINS!!! Without erasing anything, move the cursor to the  of the animal you wish to capture. Then erase the  . If you erase anything besides the  , you lose.
14. Which animals did you capture? \_\_\_\_\_



# SCREEN GAME #2

## Computer Roundup

1. Turn on PET.
2. Clear the screen so it is blank.  
What did you press to make it blank?\_\_\_\_\_
3. Press  SHIFT and hold it down. Press the X key seven times. Press  SHIFT and hold it down. Press the  π key. On your screen should be seven trees and one sheep.
4. On the same line of your screen make four trees and one sheep, four trees and one sheep, until the whole line is filled up and the cursor moves to the line below.
5. Now make two more lines of sheep and trees. You may decide how many trees and sheep to put into your lines.
6. THE ROUNDUP BEGINS!!! Your flock of sheep is lost in the forest. As the shepherd, you must get your lost flock back together again.
7. The only way you may round up your sheep is by erasing the trees. Then your sheep may find their way again. BE CAREFUL! Do not erase any sheep—only trees! If you accidentally erase any sheep, they are lost from the flock and will forever graze happily in that big pasture in the sky. GOOD LUCK!! May you find all the sheep in your lost flock.
8. Which keys did you press to erase the trees?  
\_\_\_\_\_  
\_\_\_\_\_

Good Luck,  
Partner!

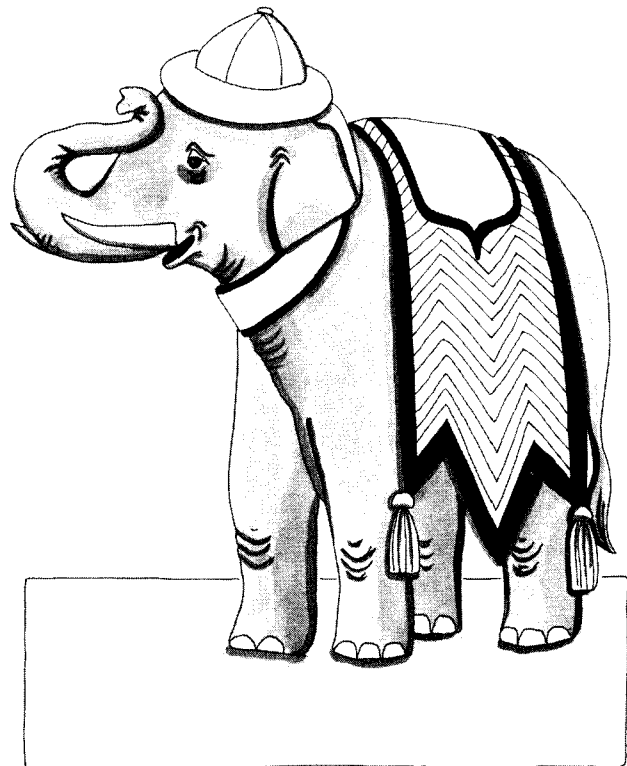


# SCREEN GAME #3

## Mine the Diamonds

Created by Wendy Cheldelin





1. Turn PET on.
2. Clear the screen.
3. Press  SHIFT and hold it down. Press the W key five times.
4. Press  SHIFT and hold it down. Now press the Z key once. On your screen should be five rocks and one diamond.
5. On the same line of the screen, please make four or five more rocks followed by one diamond until the line is filled up, and the cursor has moved to the line below.
6. You may make two or three more screen lines of rocks and diamonds.
7. Now the challenge begins! Mine the diamonds by erasing all of the rocks. If you erase a diamond—you lose!



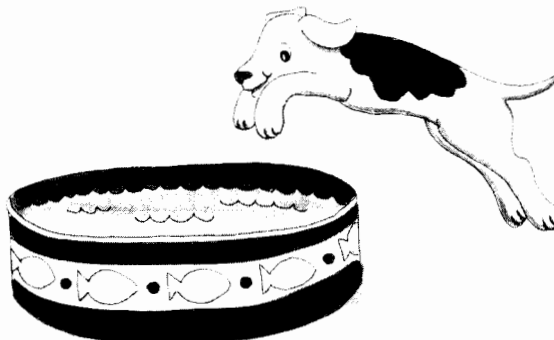
# SCREEN GAME #4

## Crude-Oil Caper

Created by Jake Coleman

1. Turn on PET.
2. Clear the screen.
3. Press . Now press the N and M keys seven times in that order.
4. Press the Q key once.
5. Take turns pressing the N and M keys and the Q key until you finish the screen line.  
You should have ocean waves and crude-oil barrels on the screen.
6. Now you may make two or three more lines of oil barrels floating in the ocean.  
You may decide how many oil barrels to put on one screen line.
7. The caper begins! Using the , , and  keys, pull all of the oil barrels to the left side of the screen. You may erase the waves, but if you erase an oil barrel, you lose!

**Good luck—the world's energy supply depends on you!**



# SCREEN GAME #5

## Word Treasures

### Idea Created by Eric Youngquist

1. Turn PET on.
2. Clear the screen.
3. Press the G key once.  
Press  SHIFT and hold it down. Press the ? key six times.
4. Press the O key once.  
Press  SHIFT and hold it down. Press the ? key six or seven times.
5. Press the L key once.  
Again, press the  SHIFT and ? keys six or seven times.
6. Press the D key once.  
Holding down the  SHIFT, press the ? key four times.
7. On your screen should be the letters G-O-L-D separated by the ■ graphic.
8. For the second line of your screen, separate the letters of the word SILVER with the same graphic.
9. Make two more screen lines the same way using the words RUBIES and SAPPHIRES.
10. To win the treasures, you must move all the letters of the words together and erase all of the graphic symbols. If you erase a letter, you lose the treasure!!!

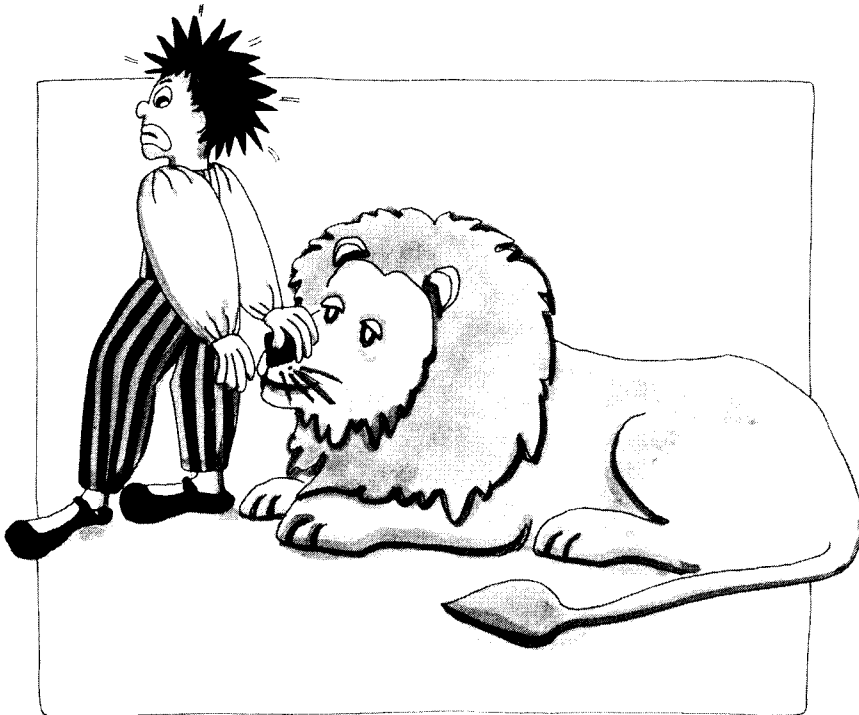
# KEYBOARD CREATION #1

## Maze Maker

1. Turn PET on.
2. Clear the screen.
3. Lock the  SHIFT into place.  
When you press the & key, PET should print a solid square: ■  
You will use this graphic for your maze wall.
4. Begin the maze craze by making the maze path on the screen.  
Be sure to plan for dead ends and tricky places.
5. When your maze path is completed, move the cursor home.
6. Your challenge is to move the cursor through the maze without touching any of the maze walls. Can you reach the end without bumping into a wall?

### Time Yourself

1st try \_\_\_\_\_  
2nd try \_\_\_\_\_  
3rd try \_\_\_\_\_



# KEYBOARD CREATION #2

## PET Pac Man

1. Turn PET on.
2. Clear the screen.
3. Make a maze on the screen.
4. Make monsters in the maze path by pressing the \* key.
5. PET Pac Man's challenge: (the cursor is the Pac Man) You must move the PET Pac Man through the maze to the end. PET Pac Man may not touch a maze wall and may not touch a monster.
6. If PET Pac Man touches a wall, you must start all over. If PET Pac Man touches a monster, you lose. Good luck!!!

### Time Yourself

1st try \_\_\_\_\_

2nd try \_\_\_\_\_

3rd try \_\_\_\_\_

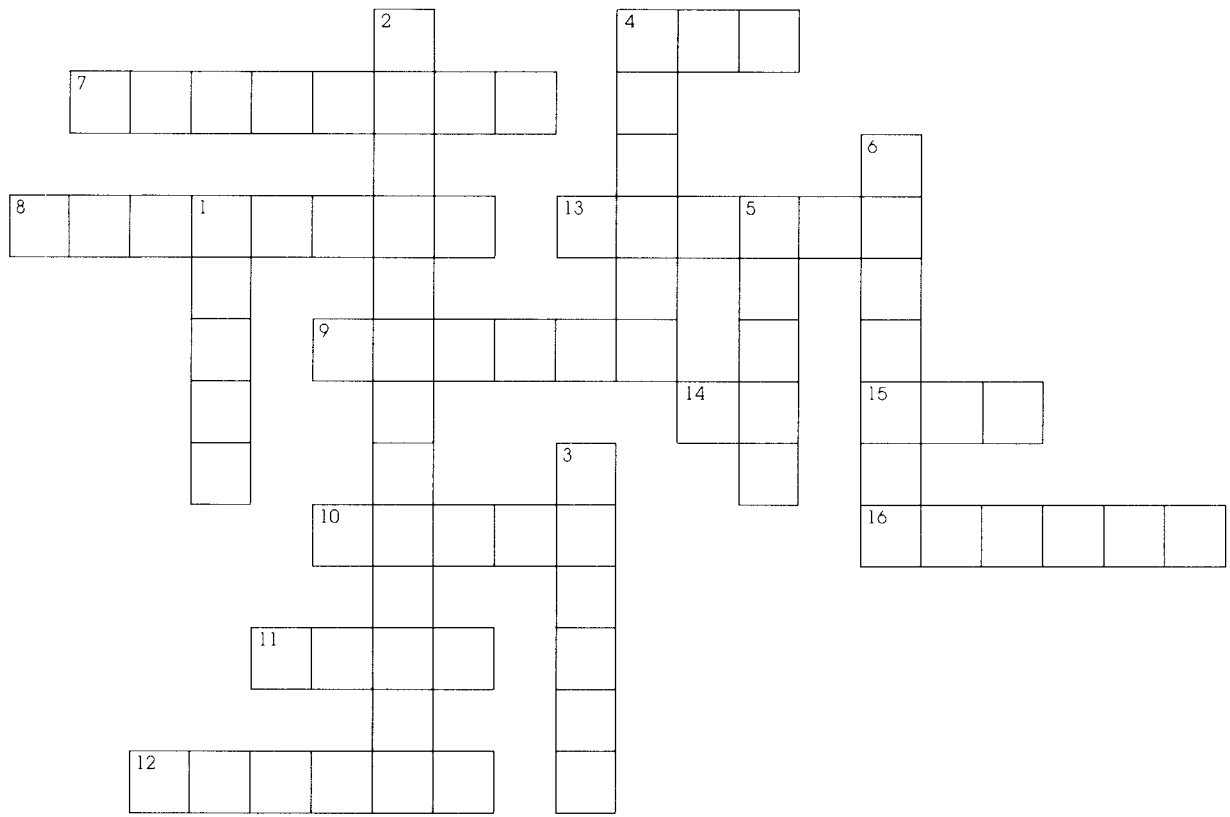


# **KEYBOARD CREATION #3**

## **Design Your Own**

1. Using PET's keyboard and screen, design your own game. Write clear directions on how to play your game in the space below.
2. Ask a friend to play your game!

# COMPONENT 1 FUN PAGE



### Down

1. The language that PET speaks.
2. What type of computer is PET?
3. What is another word for "erase"?
4. We must always press this key when we are done typing a line.
5. Hold this key down as you press another key and PET will print the symbol at the top or front of the key.
6. The set of directions a computer uses.

### Across

4. Means "rewind" on the cassette player.
7. Symbols used to make pictures and borders.
8. Something that both a computer and a typewriter have.
9. A part of PET made with a Cathode Ray Tube.
10. Always press this key between words and numbers that you type.
11. The space it takes to store one letter in PET's memory.
12. What is another word for "add"?
13. The blinking white square that shows you where PET will type next on the screen.
14. Means "fast forward."
15. Means "record."
16. The place where PET remembers programs.

---

## Evaluate Yourself

1. In component 1, I did \_\_\_\_\_  
because \_\_\_\_\_  
\_\_\_\_\_
2. Component 1 was \_\_\_\_\_  
\_\_\_\_\_
3. Tell about the good parts of the component:  
\_\_\_\_\_  
\_\_\_\_\_
4. Tell about the bad parts of the component:  
\_\_\_\_\_  
\_\_\_\_\_





# PROGRAMMING YOUR PET #1

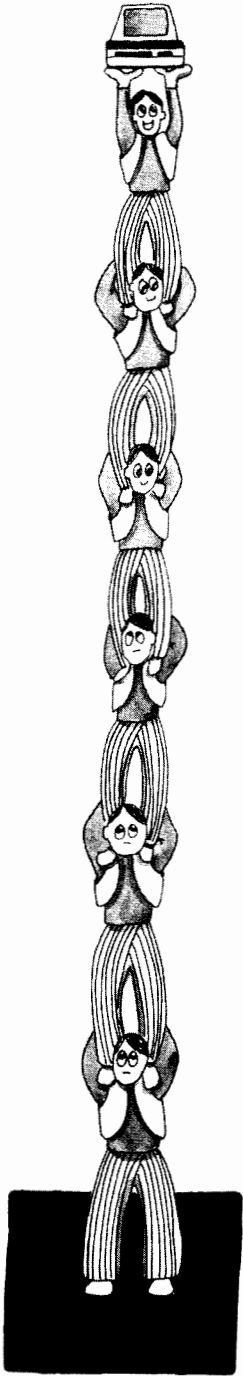
## Speak

1. Write a program that tells PET to print:  
COMPUTER PROGRAMMING IS FUN!
2. Make sure each line of your program begins with a line number.
3. Check your program to make sure there are no mistakes.
4. RUN your program on the computer.

### Use this format

```
10 PRINT "      "  
20 END
```

### Write your program here



# PROGRAMMING YOUR PET #2

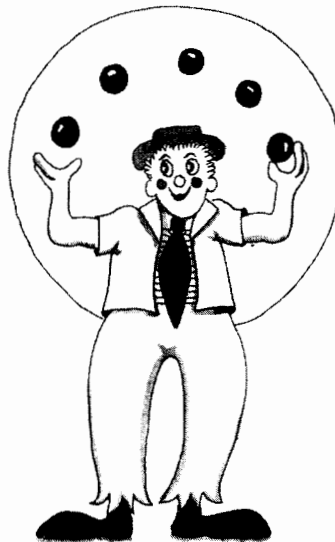
## Speaking Nonstop

1. Write a program that tells PET to print your name over and over again!
2. RUN your program on PET.

### Use this format

```
10 PRINT "          "  
    (Type your name inside the quotes.)  
20 GOTO 10
```

### Write your program here



# PROGRAMMING YOUR PET #3

## Top-Secret

1. Your mission is to write a program that tells PET to print a top-secret message in a secret code. Use the graphic symbols on the keys as your code.
2. For example, if we wanted a word in our message to say SAW, the code would be ♥ ♠ ◻ because these graphic symbols appear at the top or front of the keys for S, A, and W.
3. Give your program to a friend. Have your friend RUN it on PET and try to decode the secret message.

**Your success as a secret agent depends on this program—good luck!** (If you're not finished, this page self-destructs in two days.)

**Write your program here**



# PROGRAMMING YOUR PET #4

## Computer Art

1. Write a program that tells PET to print a design using graphic symbols.
2. Make PET print your design over and over on the screen.
3. RUN the program on PET.

**Write your program here**



# PROGRAMMING YOUR PET #5

## Slanted Art

1. Write a program that tells PET to print slanted designs using the graphic symbols.
2. To look slanted, the design must be printed over and over on the screen.
3. RUN your program on PET.

### Use this format

```
10 PRINT ""  
20 GOTO 10
```



Don't forget the ; !

### Write your program here



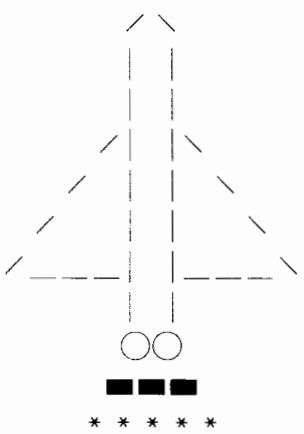
# PROGRAMMING YOUR PET #6

## Shuttle Launch

1. You can write a program that will tell PET to make moving pictures by using these five things:
  - line numbers
  - PRINT statements
  - quotation marks
  - GOTO statement
  - graphic symbols
2. Use the format below to create a program that launches a rocket.
3. RUN the program on PET.

### Use this format

```
10 PRINT  ""
20 PRINT  ""
30 PRINT  ""
40 PRINT  ""
50 PRINT  ""
60 PRINT  ""
70 PRINT  ""
80 PRINT  ""
90 PRINT  ""
100 PRINT ""
110 PRINT ""
120 PRINT ""
130 PRINT ""
140 GOTO 10
```



**NOTE:** It is important to leave lines 120 and 130 blank inside the quotation marks so the rockets are spaced out when they move on the screen.

---

**Write your program here**

A large, empty rectangular box with a thin black border, intended for writing a program. The box is centered on the page and occupies most of the vertical space below the instruction.

You may have more than 140 lines in your program.

# PROGRAMMER'S PASTIME #1

Write the symbol that PET uses for each arithmetic operation.

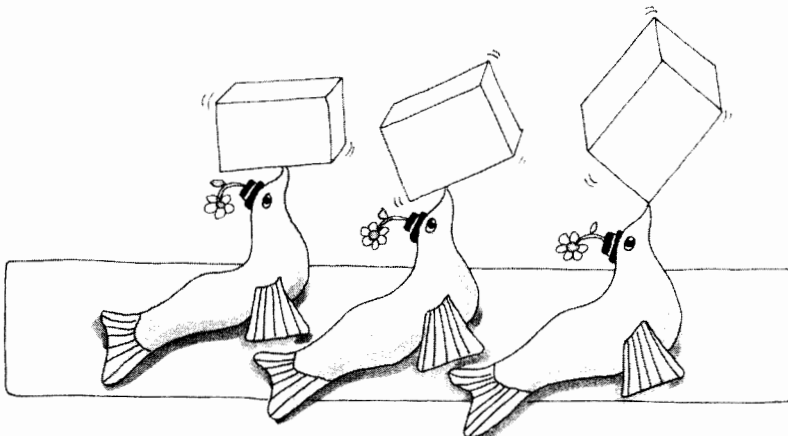
addition \_\_\_\_\_ multiplication \_\_\_\_\_  
powers \_\_\_\_\_ subtraction \_\_\_\_\_  
division \_\_\_\_\_ square root \_\_\_\_\_

How would you type each equation to get answers from PET?

1.  $457 + 99 \times 6$  \_\_\_\_\_
2.  $\sqrt{64}$  \_\_\_\_\_
3.  $26 \div 2^2$  \_\_\_\_\_
4.  $777 \times 555 \div 222$  \_\_\_\_\_
5.  $8^3 - 16$  \_\_\_\_\_
6.  $\sqrt{22} \div 88$  \_\_\_\_\_
7.  $\sqrt{49} + 765$  \_\_\_\_\_
8.  $98 + 88 \times 66 \div 2^4$  \_\_\_\_\_

Show how you would type the equations above using only one ? (PRINT) statement.

\_\_\_\_\_  
\_\_\_\_\_





# PROGRAMMER'S PASTIME #2

In what order does PET perform arithmetic in equations of many numbers?

\_\_\_\_\_ are done first.  
\_\_\_\_\_ are done second.

and \_\_\_\_\_ are done third.  
(left to right)

and \_\_\_\_\_ are done last.  
(left to right)

Use your mental powers and write the answers that PET would give for these equations. Remember to do the arithmetic in the same order that PET would!

- 1.  $2 * 3 + 1$  \_\_\_\_\_
- 2.  $2 + 8 * 2 * 1$  \_\_\_\_\_
- 3.  $3 * 3 + 9 + 20$  \_\_\_\_\_
- 4.  $11 + 4 * 3$  \_\_\_\_\_
- 5.  $22 + 8 + 12 * 1$  \_\_\_\_\_

Try some more.

- 1.  $8 / 2 - 3$  \_\_\_\_\_
- 2.  $20 / 4 + 6 - 5$  \_\_\_\_\_
- 3.  $30 - 10 / 2$  \_\_\_\_\_
- 4.  $50 - 20 / 10 + 3$  \_\_\_\_\_
- 5.  $16 / 2 + 8 / 2$  \_\_\_\_\_
- 6.  $14 / 2 + 4 / 2 - 6$  \_\_\_\_\_



# PROGRAMMER'S PASTIME #3

Use your mental powers and write the answers the computer would give for these equations.

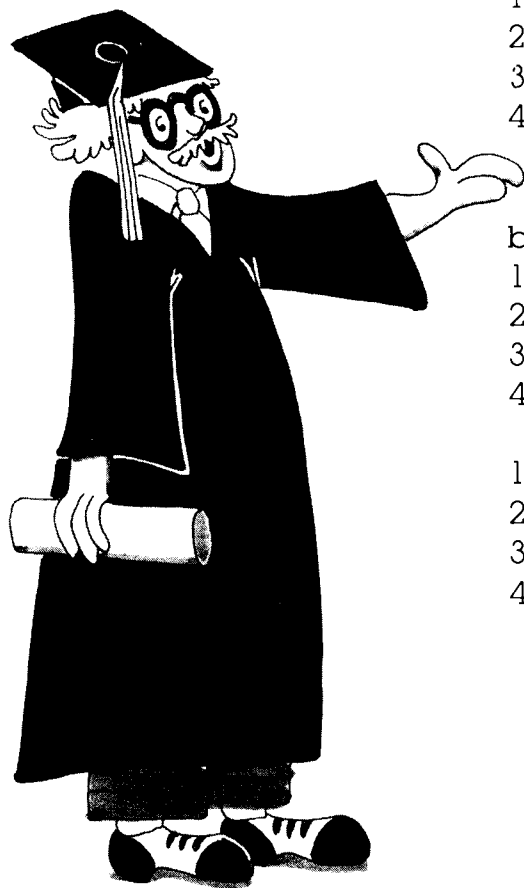
1.  $4 * 4 + 6 / 2$  \_\_\_\_\_
2.  $8 / 4 + 3 * 3$  \_\_\_\_\_
3.  $20 - (4 + 5 * 2) + 15$  \_\_\_\_\_
4.  $6 + 14 / 7 * 5$  \_\_\_\_\_
5.  $(9 / 3 - 2) * (7 * 2 + 4)$  \_\_\_\_\_
6.  $(7 + 2 - 4 + 6 * 1) / 1$  \_\_\_\_\_

## POWER!

Powers are also called **EXPONENTS**. Read the following examples and figure out on your own how powers work.

1.  $10 \uparrow 1 = 10 * 1 = 10$
2.  $10 \uparrow 2 = 10 * 10 = 100$
3.  $10 \uparrow 3 = 10 * 10 * 10 = 1000$

1.  $2 \uparrow 1 = 2 * 1 = 2$
2.  $2 \uparrow 2 = 2 * 2 = 4$
3.  $2 \uparrow 3 = 2 * 2 * 2 = 8$
4.  $2 \uparrow 4 = 2 * 2 * 2 * 2 = 16$



Use your mental powers and solve the powers by filling in the blanks.

1.  $3 \uparrow 1 =$  \_\_\_\_\_  $=$  \_\_\_\_\_
2.  $3 \uparrow 2 =$  \_\_\_\_\_  $=$  \_\_\_\_\_
3.  $3 \uparrow 3 =$  \_\_\_\_\_  $=$  \_\_\_\_\_
4.  $3 \uparrow 4 =$  \_\_\_\_\_  $=$  \_\_\_\_\_

1.  $4 \uparrow 1 =$  \_\_\_\_\_  $=$  \_\_\_\_\_
2.  $4 \uparrow 2 =$  \_\_\_\_\_  $=$  \_\_\_\_\_
3.  $4 \uparrow 3 =$  \_\_\_\_\_  $=$  \_\_\_\_\_
4.  $4 \uparrow 4 =$  \_\_\_\_\_  $=$  \_\_\_\_\_

## CHALLENGE

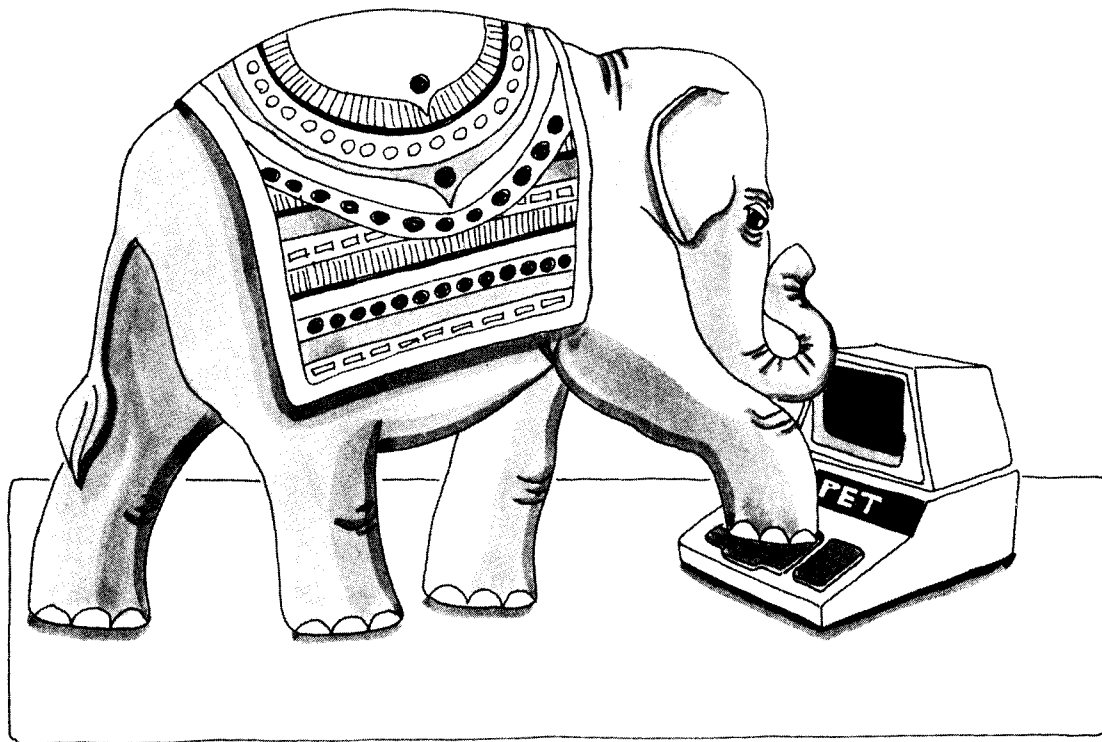
1.  $2 \uparrow 4 * 5$  \_\_\_\_\_
2.  $5 * 2 \uparrow 4$  \_\_\_\_\_
3.  $7 + 3 \uparrow 2$  \_\_\_\_\_
4.  $5 \uparrow 2 - 13$  \_\_\_\_\_
5.  $100 / 10 \uparrow 2$  \_\_\_\_\_
6.  $12 \uparrow 2 - 5 * 10$  \_\_\_\_\_
7.  $8 \uparrow 2 * (4 + 5)$  \_\_\_\_\_
8.  $200 - (6 + 3 \uparrow 3) + 7$  \_\_\_\_\_

In each of the following equations, put a number 1 under the part the computer would do first, a 2 under the second, and so on.

**Example:**  $82 + 72 / 2 - (4 + 22) + 9 \uparrow 3$   
                   4  3  5  1  6  2

1.  $3000 - (13 - 6 * 4) + 8 \uparrow 3 + 9$

2.  $900 / 7 \uparrow 2 + (16 - 9 \uparrow 4) \uparrow 3$   
 - - - - -



# PROGRAMMER'S PASTIME #4

Cowboy Clyde typed in the following equations, but PET wouldn't give him answers. Do you know why?

Find out what's wrong with the way his equations are typed. Write the correct way to type the equations in the blanks.

1. ?  $62+4 \times 20$  \_\_\_\_\_
2.  $23 / 4 * 6$  \_\_\_\_\_
3. ?  $SQR 16$  \_\_\_\_\_
4.  $80 / 4+2 \times 3 * SQR 25$  \_\_\_\_\_

## CHALLENGE

If you want certain arithmetic in a long equation to be done FIRST, you should put parentheses around them. (PET always does what's in parentheses first.) Let's say you want  $4+3 * 2$  to equal 14.

If you type:  $4+3 * 2$  PET will give you 10  
first

because  $3*2=6$  is done first. (Multiplication is done before addition.)

$$4+6=10$$

If you want  $4+3 * 2$  to equal 14, you must use parentheses like this:  $(4+3) * 2$ .

first

$$7 * 2 = 14$$



Rewrite each equation below and put parentheses around what PET should do first to make the equation TRUE.

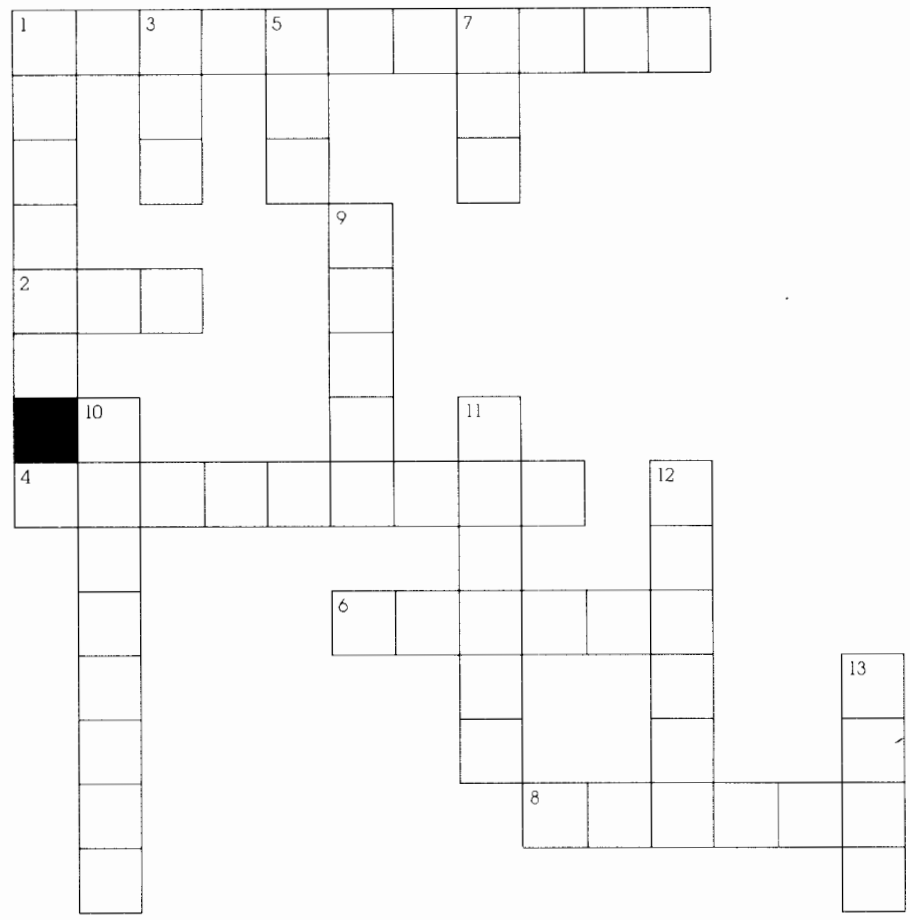
**HINT:** You might have to use **two** sets of parentheses for some equations.

- Example:**  $7 * 3 + 2 = 35$        $7 * (3 + 2) = 35$
1.  $9 / 2 + 1 = 3$       \_\_\_\_\_
  2.  $6 * 2 + 2 = 24$       \_\_\_\_\_
  3.  $3 \uparrow 3 - 1 = 9$       \_\_\_\_\_
  4.  $4 + 2 - 1 * 5 = 9$       \_\_\_\_\_
  5.  $12 - 3 + 6 / 3 = 9$       \_\_\_\_\_
  6.  $20 - 10 \uparrow 4 + 2 = 10002$       \_\_\_\_\_
  7.  $12 / 4 + 2 = 2$       \_\_\_\_\_
  8.  $9 - 5 \uparrow 2 = 16$       \_\_\_\_\_
  9.  $50 + 10 / 18 + 10 + 2 = 2$       \_\_\_\_\_
  - \*10.  $10 + 4 - 7 \uparrow 2 * 1 + 3 - 3 = 193$       \_\_\_\_\_

**\*means it's an extra tough problem!**



# COMPONENT 2 FUN PAGE



### Down

1. In a long equation, PET does \_\_\_\_\_ second.
3. Type the \_\_\_\_\_ command when you want to see PET do a trick or run a program.
5. Which command do you type when you want to clear PET's memory or do something new?
7. Which statement in a program tells PET that your program has ended?
9. Which statement in a program tells PET to write something?
10. A \_\_\_\_\_ mark is a shortcut used in place of a PRINT statement.
11. Many equations on one line must be separated by \_\_\_\_\_.
12. Using PET as a calculator is called \_\_\_\_\_ mode programming.

13. In a program, \_\_\_\_\_  
10  
20  
30 are called \_\_\_\_\_ numbers.

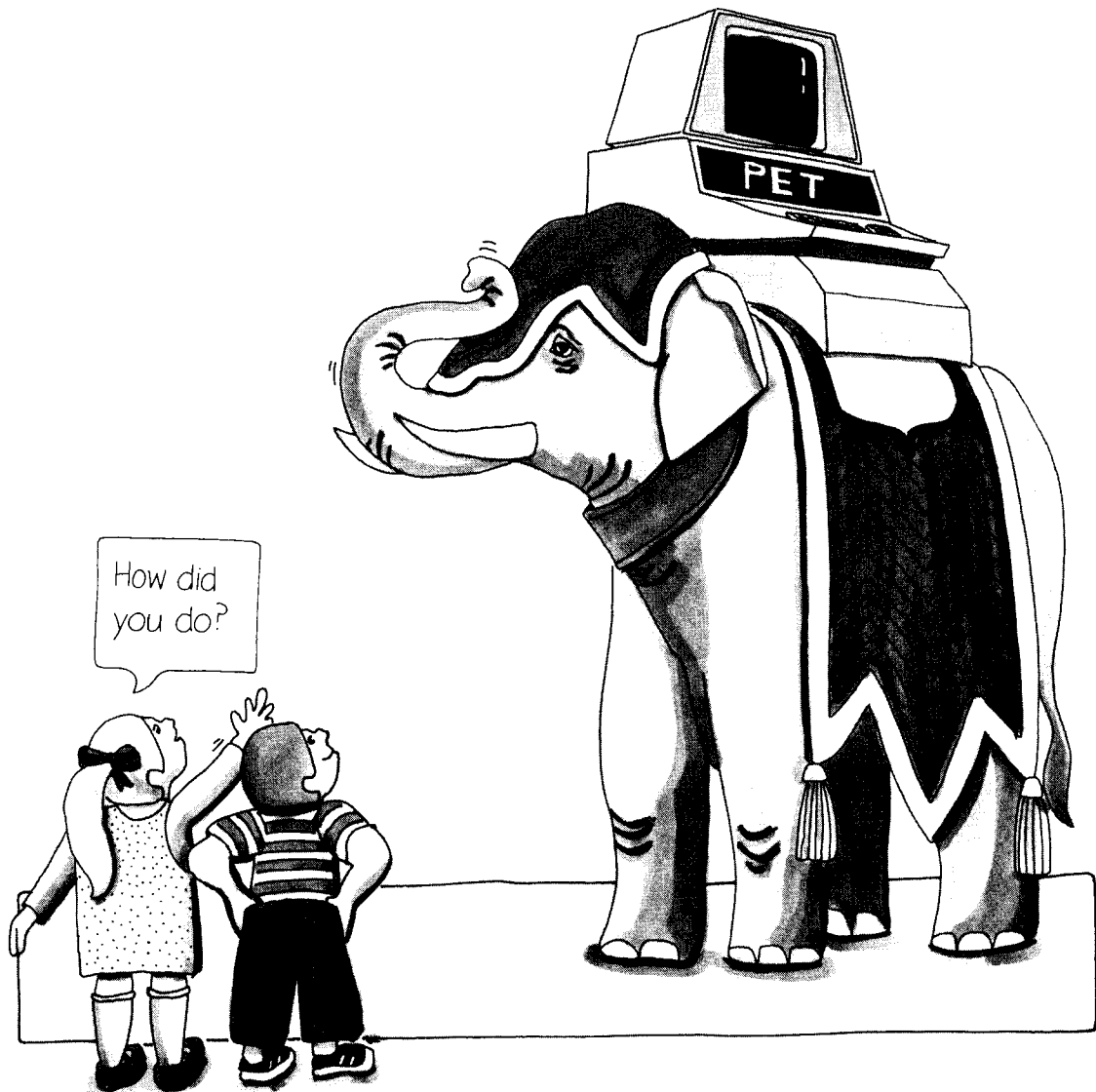
### Across

1. In a long equation, PET does the operation in \_\_\_\_\_ first.
2. Which command do you type when PET is finished loading a game and is ready to play?
4. Put \_\_\_\_\_ marks around what you want PET to say in a program.
6. Every step in a program must begin with a line \_\_\_\_\_.
8. Which key should you press when you are done typing a line and you want to go to the next line?

---

## Evaluate Yourself

1. In component 2, I did \_\_\_\_\_  
because \_\_\_\_\_  
\_\_\_\_\_.
2. Component 2 was \_\_\_\_\_  
\_\_\_\_\_.
3. Tell about the best parts of the component:  
\_\_\_\_\_  
\_\_\_\_\_.
4. Tell about the parts of the component that were hard for you: \_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_.
5. Tell about the parts of the component that you like the least: \_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_.

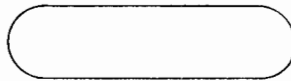
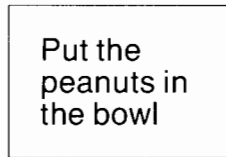
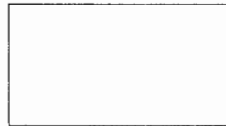
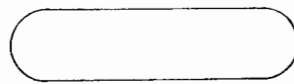


# PROGRAMMER'S PASTIME #5

For each flow chart, fill in each blank box with the step you think would fit. Make sure your steps are in the right order.

## ALGORITHM / FLOW CHART #1 HOW TO FEED YOUR PET ELEPHANT

### Missing Steps

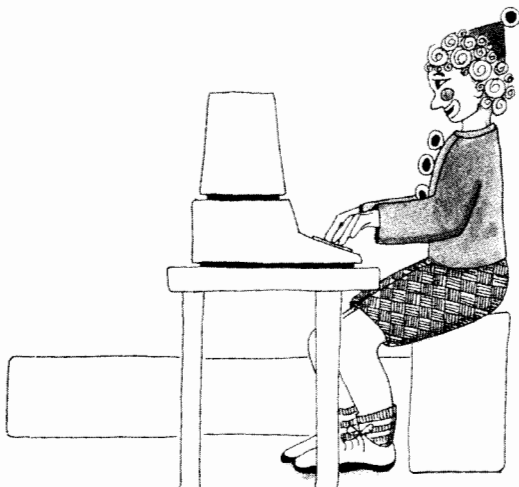


STOP

Call your pet elephant.

START

Get out the peanuts.

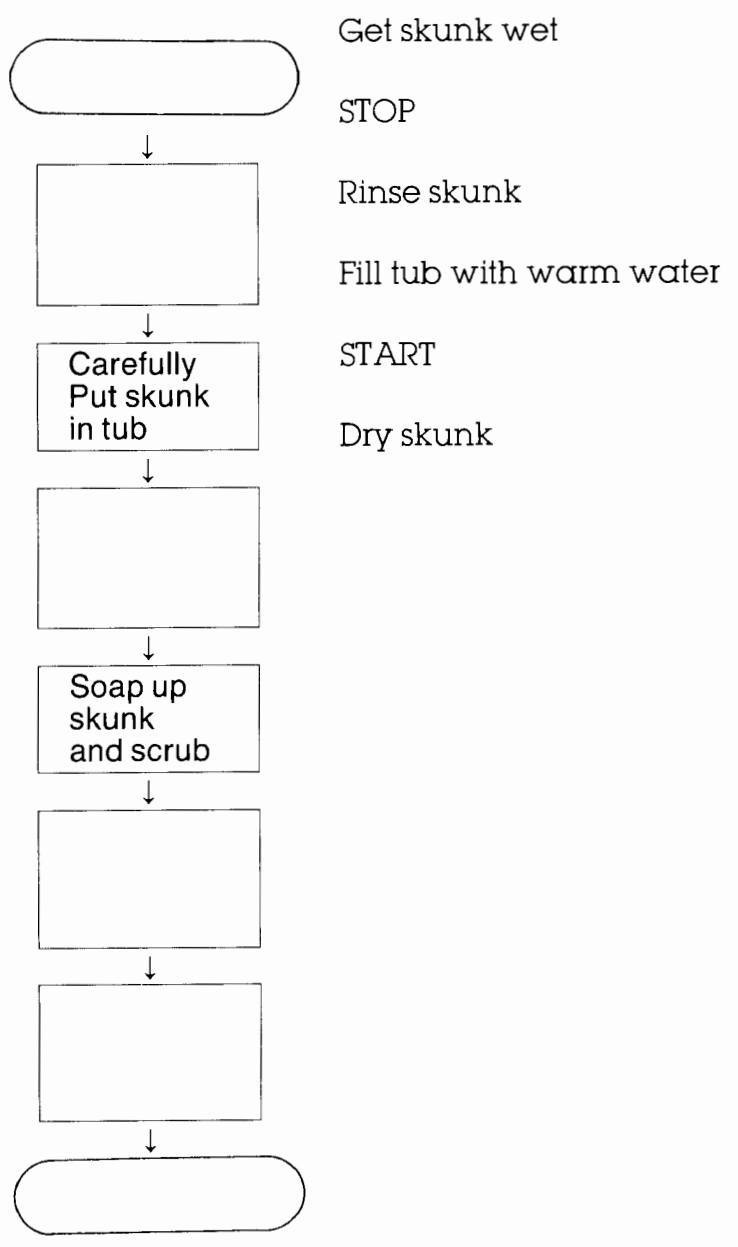




ALGORITHM / FLOW CHART #2

HOW TO WASH YOUR PET SKUNK

**Missing Steps**



Get skunk wet

STOP

Rinse skunk

Fill tub with warm water

START

Dry skunk

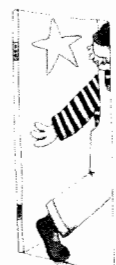
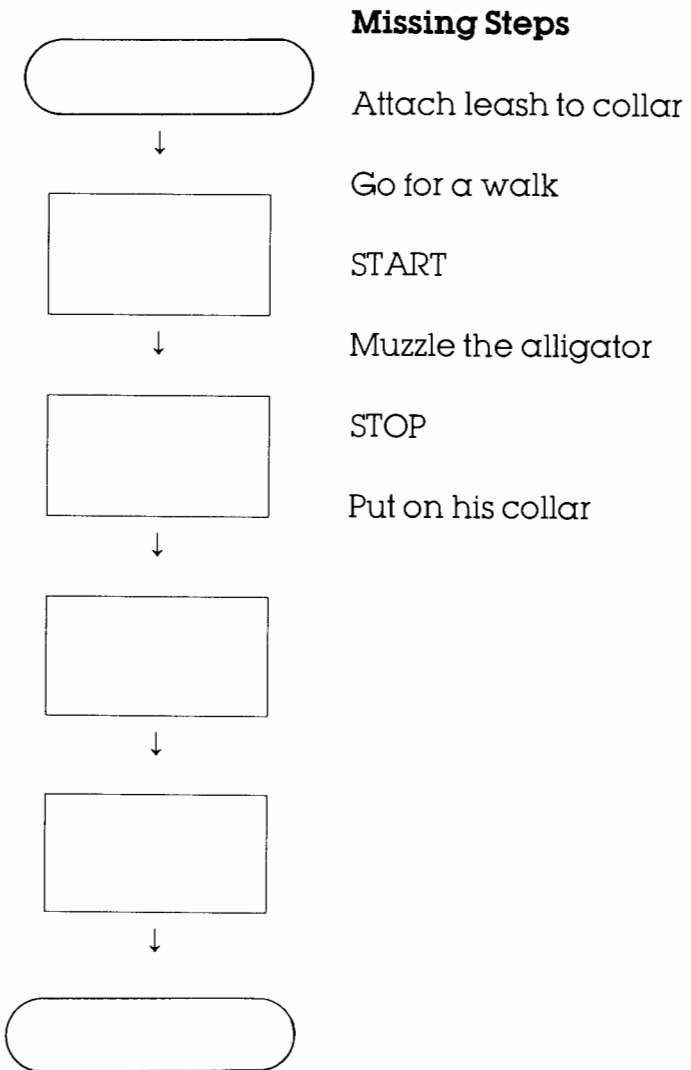
Be sure to point the skunk's tail AWAY from you! Or your pet skunk may have to give you a bath!



# PROGRAMMER'S PASTIME #6

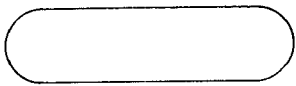
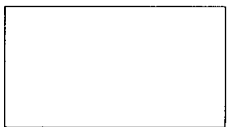
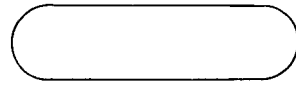
For each flow chart, fill in each blank box with the step you think would fit. Make sure your steps are in the right order.

## ALGORITHM / FLOW CHART #1 HOW TO WALK YOUR PET ALLIGATOR



ALGORITHM / FLOW CHART #2  
HOW TO MAKE AN ICE CREAM CONE

**Missing Steps**



Put away ice cream

Scoop out ice cream

START

Get out ice cream

STOP

Get out a cone

Put scoop of ice cream  
into cone

Pack the ice cream  
firmly into cone

Eat ice cream cone



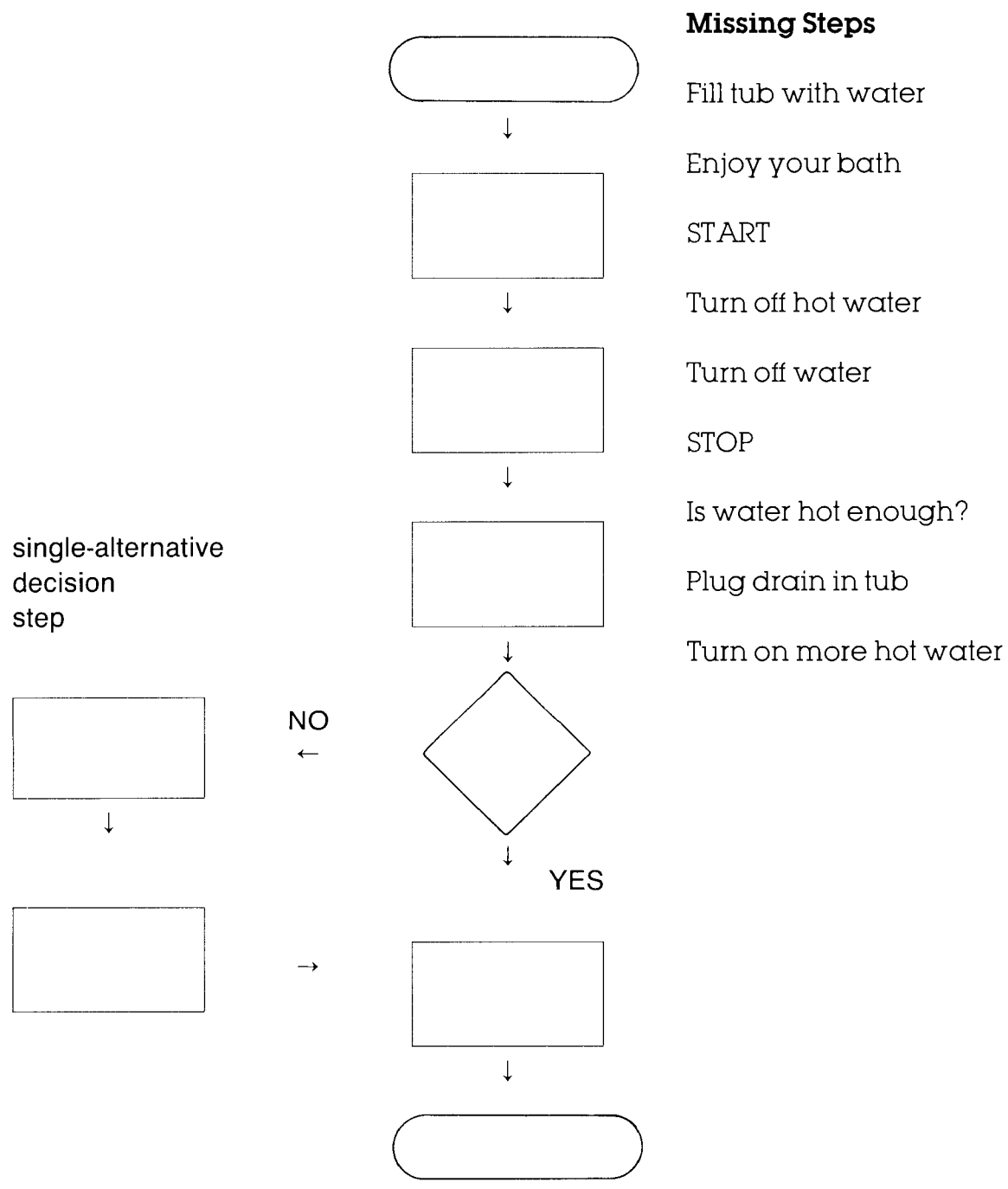
# **PROGRAMMER'S PASTIME #7**

Design an algorithm for how to make a peanut butter and banana sandwich. Write your algorithm in flow chart form.

# PROGRAMMER'S PASTIME #8

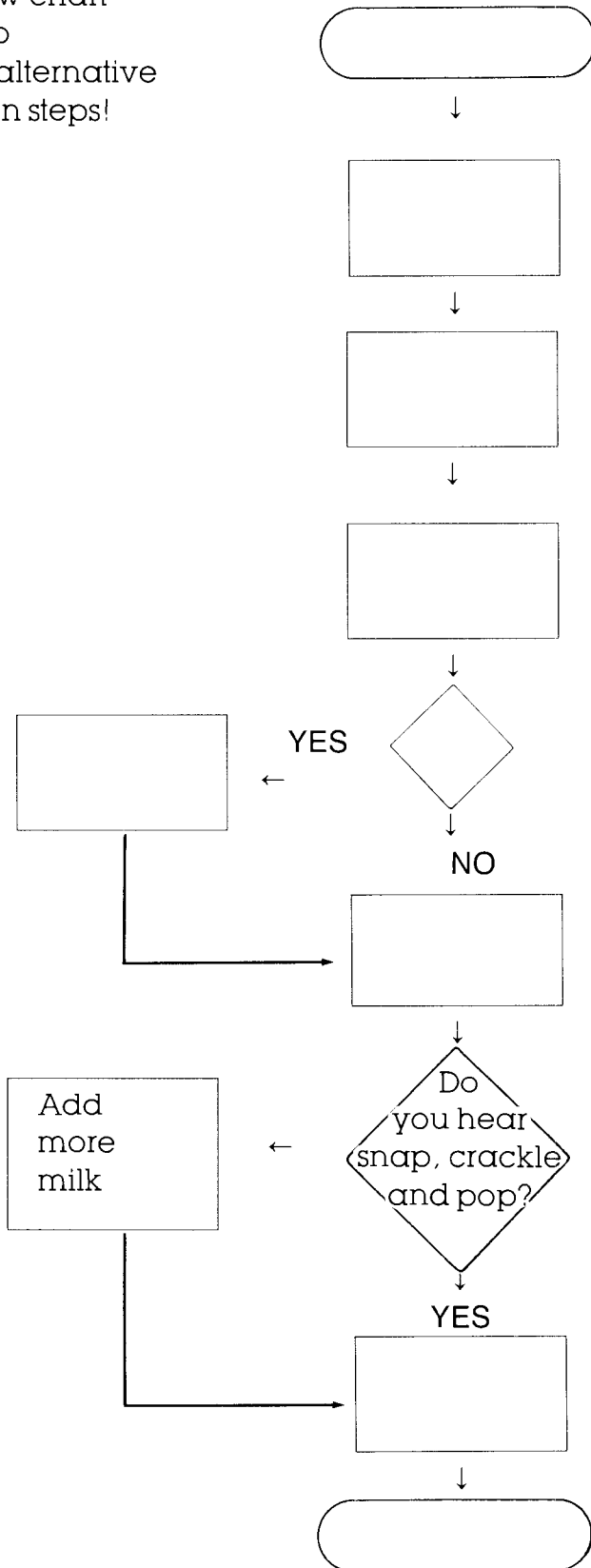
For each flow chart, fill in the blank boxes with the steps you think would fit. Make sure your steps are in the right order.

ALGORITHM / FLOW CHART #1  
HOW TO TAKE A HOT BATH



ALGORITHM / FLOW CHART #2  
HOW TO EAT A BOWL OF RICE KRISPIES

This flow chart has two single-alternative decision steps!



**Missing Steps**

- Eat your cereal
- Get out bowl & spoon
- STOP
- Add milk
- Pour Rice Krispies into bowl
- Is there too much cereal in your bowl?
- Get out Rice Krispies and milk
- Remove some cereal from bowl and put back into box
- START

# PROGRAMMER'S PASTIME #9

Here are the steps of an algorithm to make a chocolate milkshake. Put the steps in order and make a flow chart. Be sure to show the SINGLE-ALTERNATIVE DECISION STEP.

## Steps

STOP

Add  $\frac{1}{8}$  cup chocolate sauce

Get out blender

Put 3 scoops of ice cream into blender

Is milkshake too thick?

Get out ingredients

Add 1 cup milk

Blend all ingredients

Add more milk and blend ingredients again

Pour milkshake into glass

START

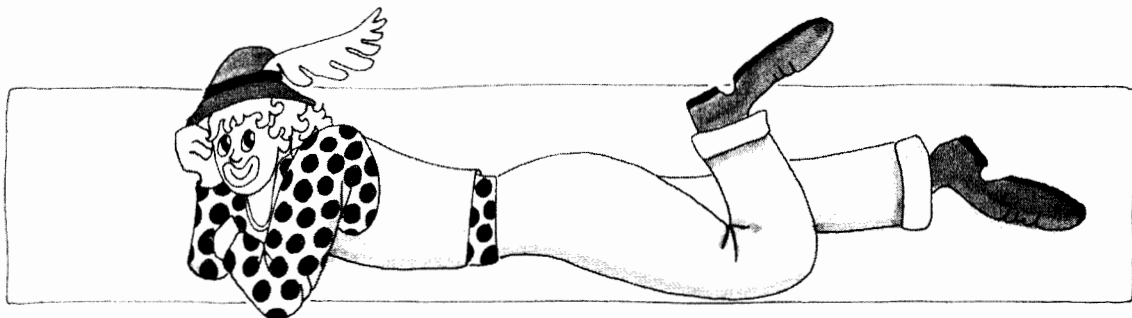
Drink your milkshake

# PROGRAMMER'S PASTIME #10

Write an algorithm on how to fix yourself a cold glass of water. Make the algorithm into a flow chart. Your flow chart should have a SINGLE-ALTERNATIVE DECISION STEP.

**Example:** Ask the question, "IS THE WATER COLD ENOUGH?"

For the NO answer detour, your step might say  
ADD ICE CUBES

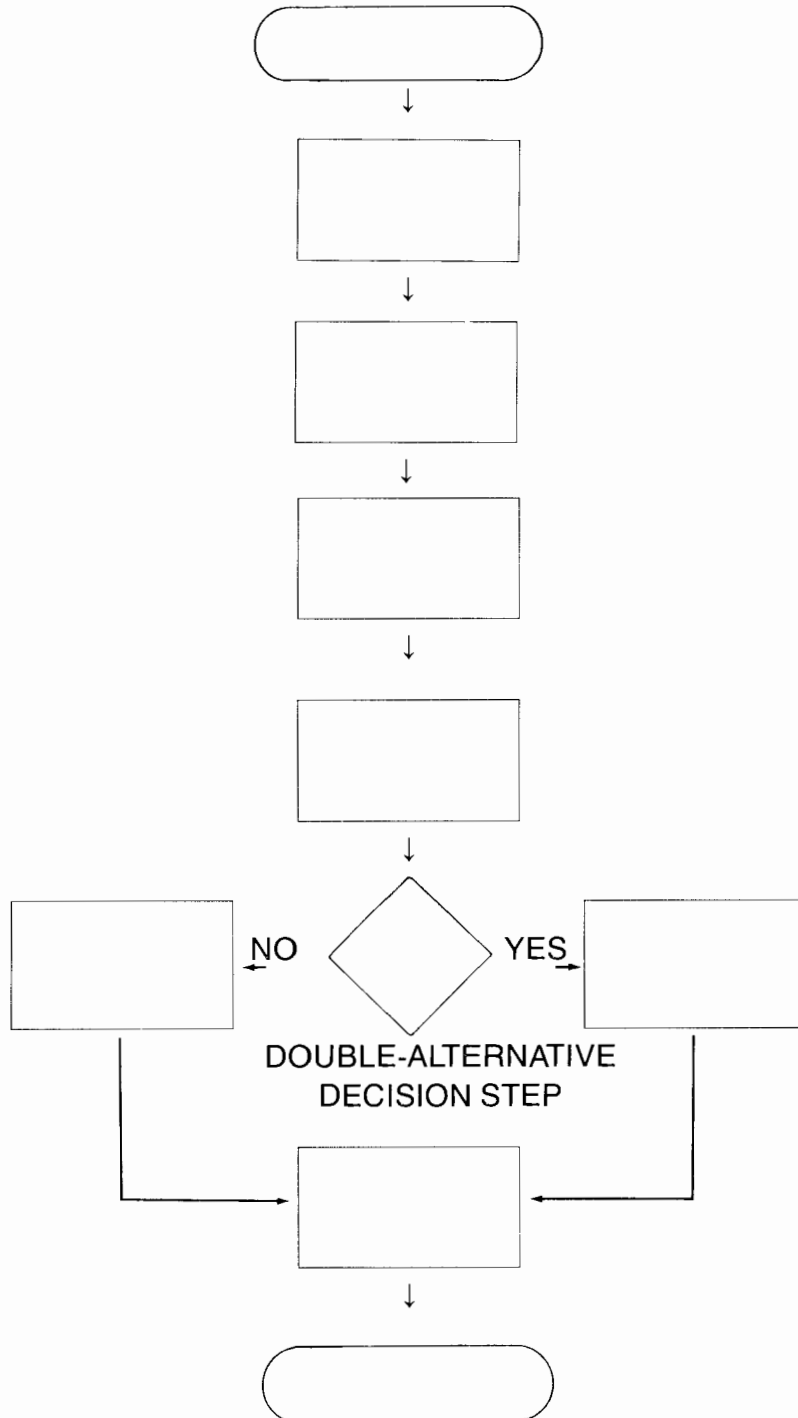




# PROGRAMMER'S PASTIME #11

For each flow chart, fill in the blank boxes with the steps you think would fit. Make sure your steps are in the right order.

ALGORITHM/FLOW CHART ON HOW TO TEACH YOUR PET BULL  
TO COME WHEN YOU CALL



**Missing Steps**

Hold out handful of hay

Climb out of corral

START

Gently get bull's attention by calling his name

Pet bull. Quickly give him the hay

Say "COME TORO" over and over

Turn and run as fast as you can

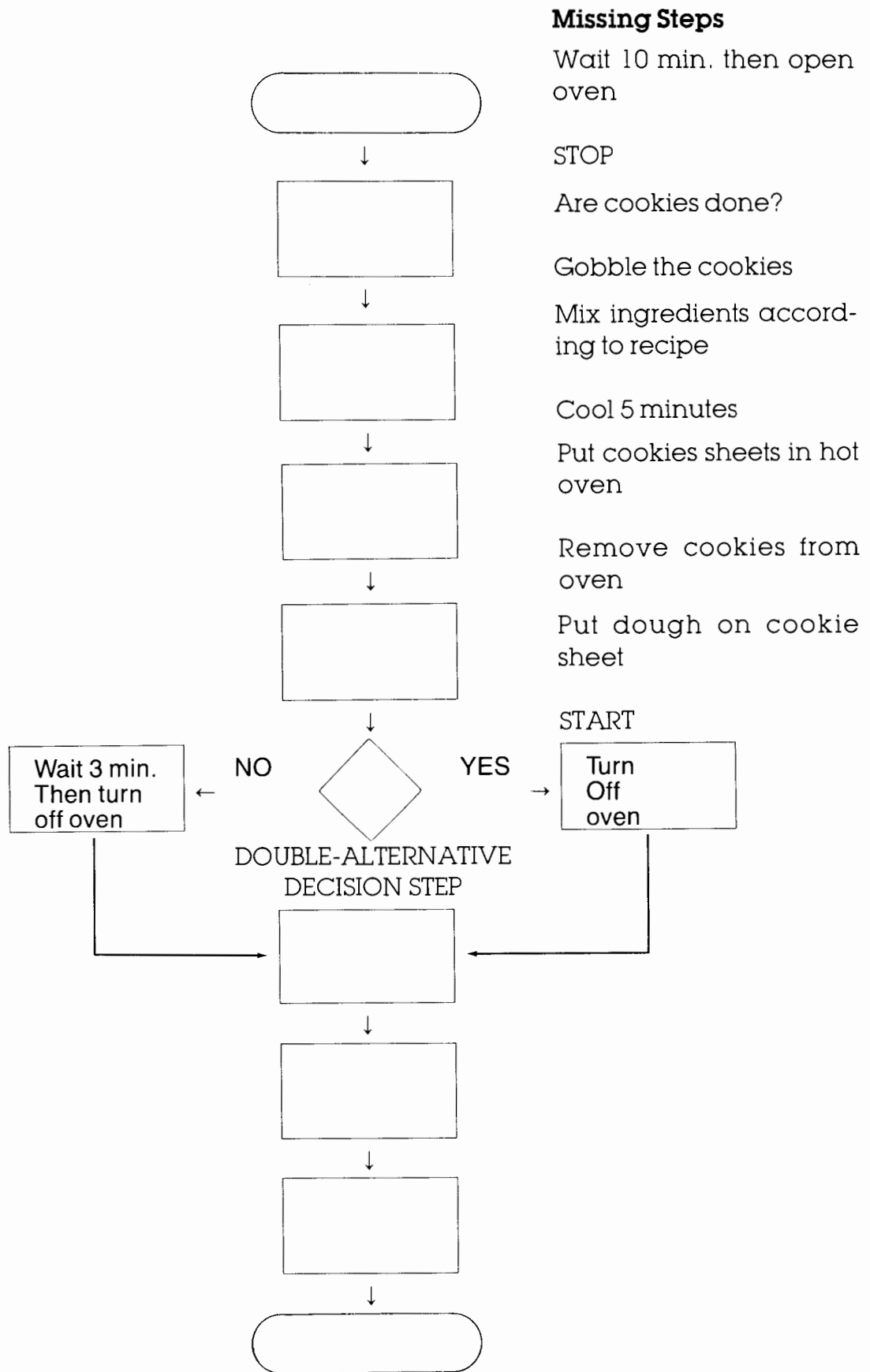
STOP

Climb into corral.

Approach bull carefully

Is the bull charging at you?

ALGORITHM / FLOW CHART ON HOW TO BAKE COOKIES



# PROGRAMMER'S PASTIME #12

Write an algorithm and flow chart on how to hit a baseball with a bat. Your flow chart must have a DOUBLE-ALTERNATIVE DECISION STEP.

**Example:** Question: IS THE BALL IN THE STRIKE ZONE?

**YES**  
KEEP YOUR EYE  
ON THE BALL

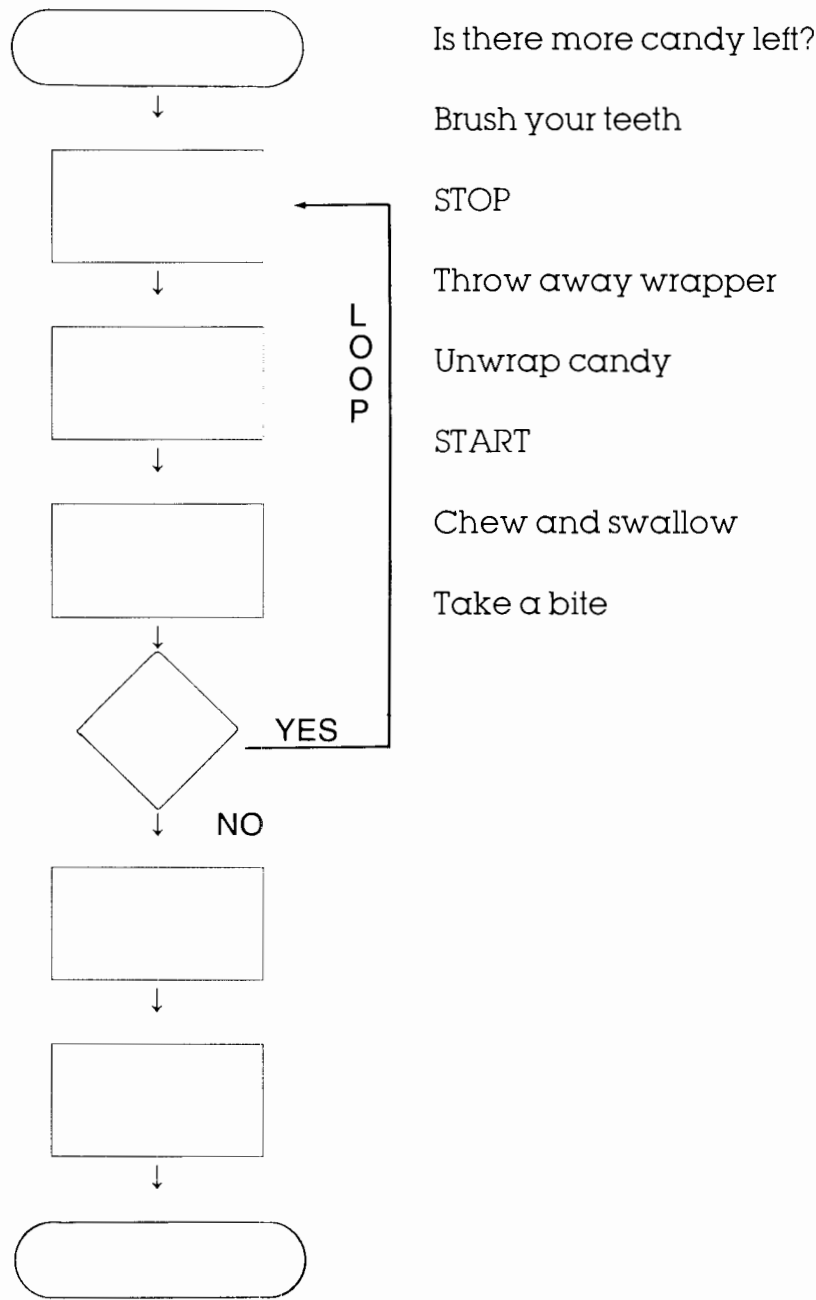
**NO**  
STEP AWAY  
FROM THE PLATE

# PROGRAMMER'S PASTIME #13

Fill in the blank boxes of the flow chart with the steps you think should fit. Make sure the steps fit the loop.

ALGORITHM / FLOW CHART  
ON HOW TO EAT CANDY

### Missing Steps



# PROGRAMMER'S PASTIME #14

Write an algorithm for how to wash your hair.  
Write the algorithm in flow chart form. Your flow  
chart should have a LOOP.

**Example:**

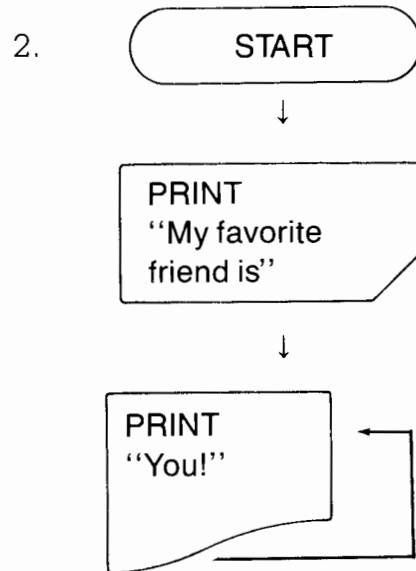
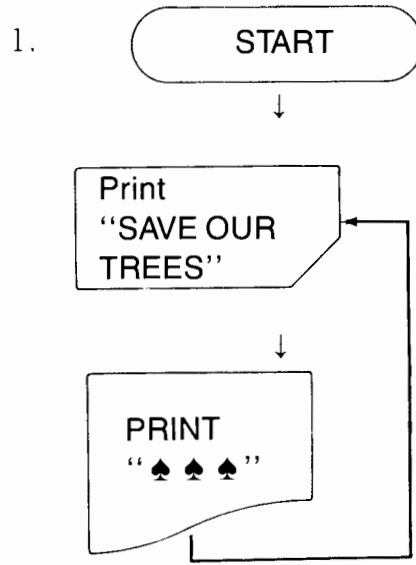
Question: IS THERE SOAP IN YOUR HAIR?

LOOP: YES → RINSE YOUR HAIR

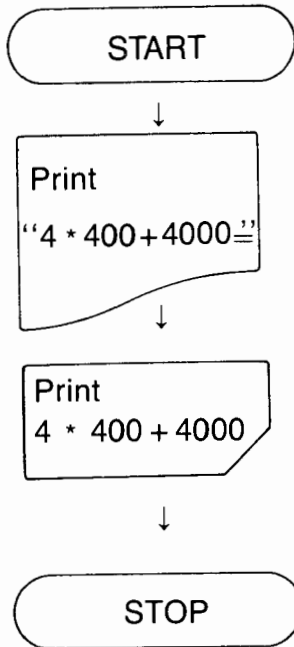


# PROGRAMMER'S PASTIME #15

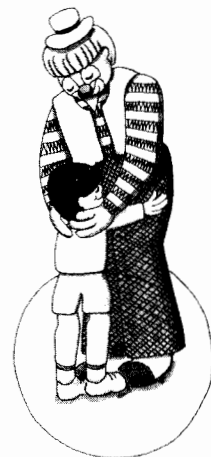
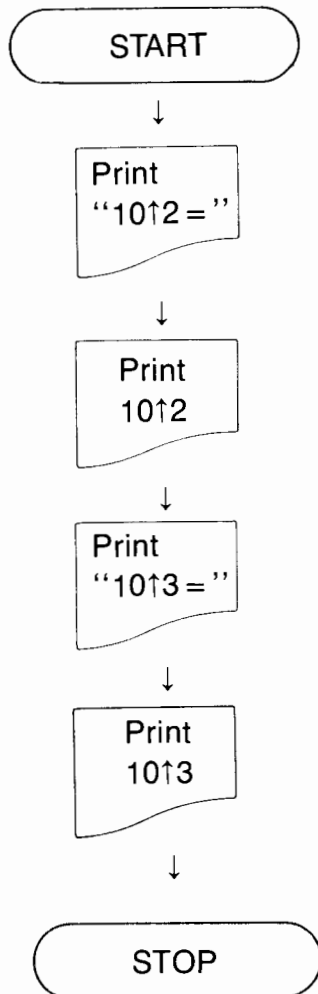
For each flow chart, write a BASIC program in the space beside it.



3.



4.



# PROGRAMMER'S PASTIME #16

Write an algorithm in a flow chart for each problem. Then write a BASIC program for PET to follow.

1. Tell PET to print over  
and over:

```
PETER PIPER  
PICKED A PECK  
OF PICKLED PEPPERS
```

**Flow chart**

**Program**

2. Tell PET to print

```
100*10 - 1000 =  
0
```

**Flow chart**

**Program**



---

3. Tell PET to print

```
I CAN DO  
56789*1234  
WHICH EQUALS  
70077626
```

**Flow chart**

**Program**

4. TELL PET TO PRINT

```
SOMEWHERE  
OVER THE RAINBOW  
MANY COMPUTERS TRY  
EQUATIONS LIKE  
4*10=  
(answer to 4*10)
```

**Flow chart**

**Program**

# PROGRAMMER'S PASTIME #17

For each program, write what you think PET would print.

**Example:**

```
10 ? ``10 + 500 + 200 = ``  
20 ? 10 + 500 + 200  
30 GOTO 20
```

**PET would print**

```
10 + 500 + 200 =  
710  
710
```

. (The dots mean that  
. 710 would be  
. printed forever.)

```
1. 10 ? ``600 - 400 + 64 = ``  
20 ? 600 - 400 + 64  
30 END
```

```
2. 10 ? ``I LOVE      ``  
20 ? `` YOU !!!     ``  
30 GOTO 20
```

```
3. 10 ? ``SHE SELLS``  
20 ? ``SEA SHELLS``  
30 ? ``BY THE SEASHORE``  
40 ? ``I CAN SAY IT ! ``  
50 GOTO 40
```

**PET would print**

# PROGRAMMER'S PASTIME #18

For each program, show how PET would print the equation on the screen.

**Example:**

```
10 ? ``10+37 = ``
    10+37
20 END
```

1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
1	0	+	3	7	=					4	7								

(1st Print Zone)

(2nd Print Zone)

```
1. 10 ? ``66+33 = ``;
    66+33
20 END
```

1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10

(1st Print Zone)

(2nd Print Zone)

```
2. 10 ? ``88-44+2 = ``;
    88-44+2
20 END
```

1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10

(1st Print Zone)

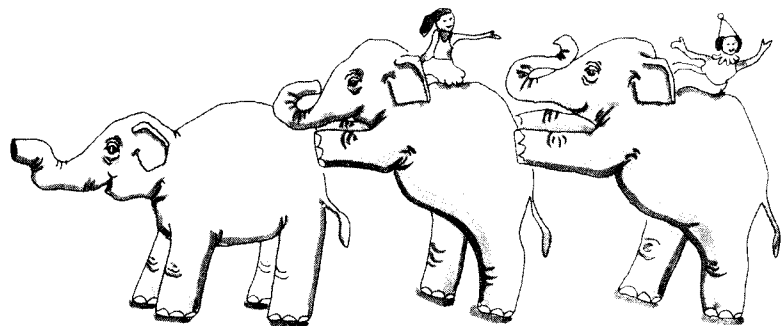
(2nd Print Zone)

```
3. 10 ? ``100+200+
    300 = `` , 100+
    200+300
20 END
```

1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10

(1st Print Zone)

(2nd Print Zone)



```

4. 10 ? ``10*50=``;
    10*50
    20 ?
    30 ? ``10*50=``;
    10*50
    40 END

```

1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
---	---	---	---	---	---	---	---	---	----	---	---	---	---	---	---	---	---	---	----

(1st Print Zone) (2nd Print Zone)

```

5. 10 ? ``60 / 20 =``,
    60 / 20
    20 ?
    30 ? ``60 / 20 =``,
    60 / 20
    40 END

```

1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
---	---	---	---	---	---	---	---	---	----	---	---	---	---	---	---	---	---	---	----

(1st Print Zone) (2nd Print Zone)

```

6. 10 ? ``5-6=``; 5-6
    20 END

```

1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
---	---	---	---	---	---	---	---	---	----	---	---	---	---	---	---	---	---	---	----

(1st Print Zone) (2nd Print Zone)

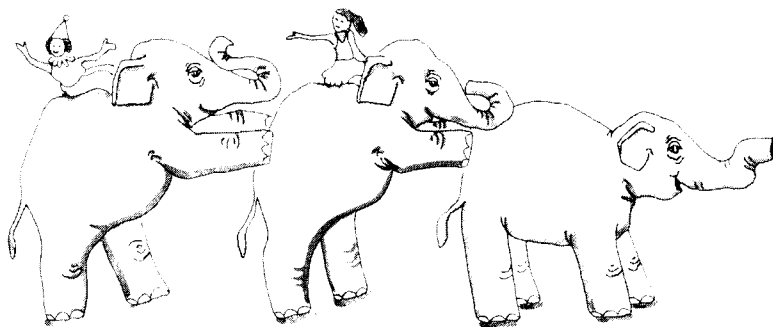
```

7. 10 ? ``8-7=``; 8-7
    20 ?
    30 ? ``7-8=``; 7-8
    40 END

```

1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
---	---	---	---	---	---	---	---	---	----	---	---	---	---	---	---	---	---	---	----

(1st Print Zone) (2nd Print Zone)



---

Write a program that tells PET to print what is seen on the screens below.

**Screens**

**Programs**

8.  $\left\{ \begin{array}{l} \text{PET IS A . . .} \\ \text{NUMBER CRUNCHER} \end{array} \right.$

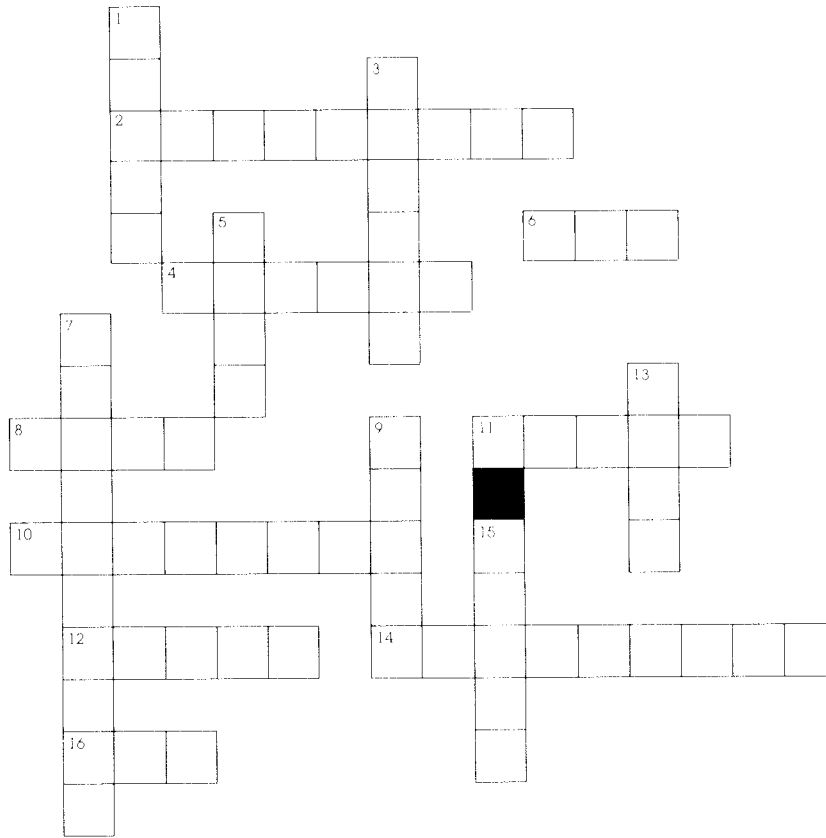
9.  $\left\{ \begin{array}{l} 10 - 9 = \quad 1 \\ 9 - 10 = -1 \end{array} \right.$

10.  $\left\{ \begin{array}{l} 5 - 6 = -1 \\ 6 - 5 = \quad 1 \end{array} \right.$

11.  $\left\{ \begin{array}{l} \text{IF YOU CAN DO} \\ \text{YOU'RE A . . .} \\ \text{WHIZ KID} \\ \text{WHIZ KID} \\ \text{WHIZ KID} \\ \quad \cdot \\ \quad \cdot \\ \quad \cdot \end{array} \right. \quad 818$



# COMPONENT 3 FUN PAGE



## Down

1. In the flow-diagramming process, we write our algorithm in a flow \_\_\_\_\_.
3. One detour from a decision box in a flow chart is a \_\_\_\_\_ alternative decision step.
5. In a flow chart, a step that is repeated is called a \_\_\_\_\_.
7. The rectangle-shaped boxes in a flow chart that tell you to do something are called \_\_\_\_\_ boxes.
9. PET's screen has four print \_\_\_\_\_.
13. The last step in a flow chart is \_\_\_\_\_.
15. A \_\_\_\_\_ tells PET to go to the next print zone and then begin printing.

## Across

2. An \_\_\_\_\_ is a step-by-step method that can be used to solve a problem.
4. Two detours from a decision box in a flow chart is a \_\_\_\_\_ alternative decision step.
6. Each print zone on PET's screen can hold \_\_\_\_\_ characters.
8. The BASIC command to loop is \_\_\_\_\_.
10. The diamond-shaped boxes in a flow chart that ask you questions are called \_\_\_\_\_ boxes.
11. One line across PET's screen holds \_\_\_\_\_ characters.
12. The first step in a flow chart is \_\_\_\_\_.

- 
14. A \_\_\_\_\_ tells PET to hold the cursor in place until the next PRINT command.
  16. Each PRINT statement tells PET to print on a \_\_\_\_\_ line.

### Evaluate Yourself

1. Component 3 was \_\_\_\_\_  
because \_\_\_\_\_  
\_\_\_\_\_
2. The best parts of the components were \_\_\_\_\_  
\_\_\_\_\_
3. The parts I liked the least were \_\_\_\_\_  
\_\_\_\_\_
4. The most valuable thing I learned in this component was \_\_\_\_\_  
because \_\_\_\_\_  
\_\_\_\_\_



# PROGRAMMER'S PASTIME #19

For each LET statement, fill in the CONTENTS of the memory cell mailbox.

1. 10 LET OP = 33



2. 20 LET VT = 17



3. 30 LET M1 = 3000



4. 40 LET D = 640



For each LET statement, fill in the ADDRESS and the CONTENTS of the memory cell mailbox.

1. 10 LET FF = 99



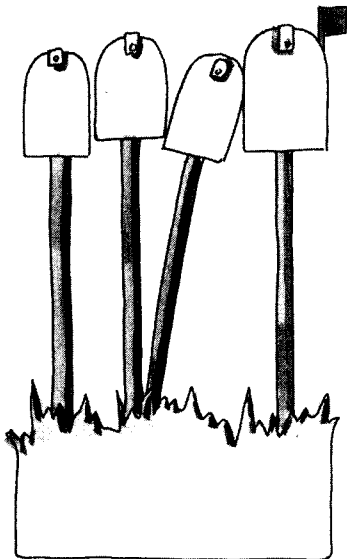
2. 20 LET PQ = 5



3. 30 LET N = 0



4. 40 LET W7 = 62



Read each variable. If the variable is one of the three types we learned about, it is safe to use in programs. Write NO if the variable is not one of the three types. Write YES if it is.

- |         |       |          |       |
|---------|-------|----------|-------|
| 1. PR   | _____ | 7. X3    | _____ |
| 2. ITCH | _____ | 8. BB    | _____ |
| 3. A    | _____ | 9. 2CC   | _____ |
| 4. E1   | _____ | 10. 23D  | _____ |
| 5. 3X   | _____ | 11. FD7  | _____ |
| 6. 7Z   | _____ | 12. KK11 | _____ |



# PROGRAMMER'S PASTIME #20

Read each program. Then write what PET would print as the output. If you can, check your answers by running the programs on PET.

## Program

## Output

1. 10 LET ZB = 14  
20 ? ZB  
30 END



2. 10 LET T2 = 77  
20 ? ``T2``  
30 END



3. 10 LET U = 182  
20 ? ``U``  
30 ? U  
40 END



4. 10 LET RC = 7  
20 ? ``RC IS``  
30 ? RC  
40 END



5. 10 LET G4 = 66  
20 ? ``G4 IS``;  
30 ? G4  
40 END



6. 10 LET L = 007  
20 ? ``JAMES BOND IS``;  
30 ? L  
40 END



# PROGRAMMER'S PASTIME #21

Read each program. Then write what PET would print as the output. If you can, check your answers by running the programs on PET. Pay close attention to safe and unsafe variables.

## Program

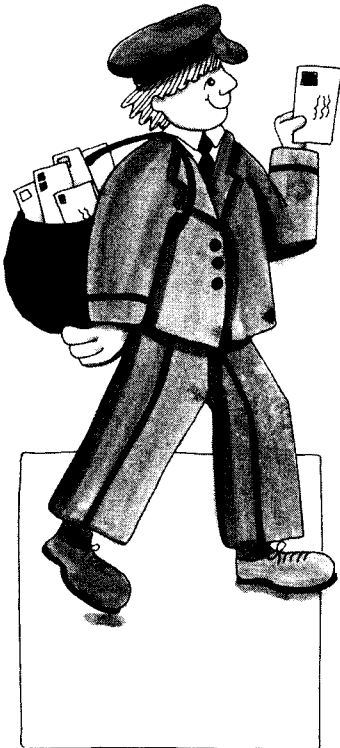
## Output

```
1. 10 LET RB4 = 40
    20 LET RB5 = 50
    30 LET RB1 = 10
    40 ? RB5; "IS";
    50 ? RB1; "MORE";
    60 ? "THAN"; RB4
    70 END
```

```
2. 10 LET T = 5
    20 LET V = 25
    30 ? "THE SQUARE
        ROOT OF";
    40 ? V; "IS"; T
    50 END
```

```
3. 10 ? "MY FAVORITE
    NUMBER IS";
    20 LET D = 333
    30 ? D
    40 GOTO 30
```

```
4. 10 ? "MY FAVORITE
    NUMBER IS";
    20 LET D = 333
    30 ? D;
    40 GOTO 30
```



# PROGRAMMER'S PASTIME #22

Read each program. Then write what PET would print as the output.

## Program

## Output

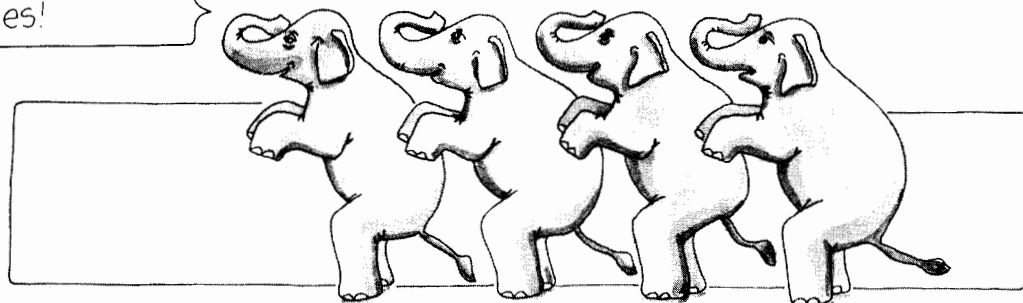
```
1. 10 LET N = 12
    20 LET R = 6
    30 ? N/R
    40 END
```

```
2. 10 LET N = 12
    20 LET R = 6
    30 ? "N+R=" ; N+R
    40 END
```

```
3. 10 LET N = 12
    20 LET R = 6
    30 ? N " + " R " = " ;
        N+R
    40 END
```

```
4. 10 LET Q = 10
    20 LET R = 20
    30 LET S = 30
    40 LET T = 40
    50 ? T/R
    60 ? Q " + " S " = " ; T
    70 ? S - Q ; " = " S
        " - " Q
    80 END
```

Pay close attention  
to safe and unsafe  
variables!



# PROGRAMMER'S PASTIME #23

For each LET statement, fill in the contents of the memory cell mailboxes.

1. 10 LET P = 41  
20 LET Q = P

P	Q
---	---

2. 10 LET A = 36  
20 LET E = 16  
30 LET I = A

A	E	I
---	---	---

3. 10 LET L = 4  
20 LET M = 2  
30 LET N = 4 + 2

L	M	N
---	---	---

4. 10 LET B1 = 3  
20 LET B2 = 6  
30 LET B3 = B1 + B2

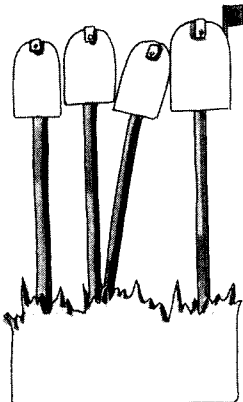
B1	B2	B3
----	----	----

5. 10 LET AB = 10  
20 LET AC = 15  
30 LET AD = AC + 5  
40 LET AE = AB + 10

AB	AC	AD	AE
----	----	----	----

6. 10 LET GP = 19  
20 LET GQ = GP - 4  
30 LET GR = GQ + 4  
40 LET GS = GR \* 1

GP	GQ	GR	GS
----	----	----	----



# PROGRAMMER'S PASTIME #24

Read each program. Then write what PET would print as the output. Check your answers by running the programs on PET.

## Program

## Output

1. 10 LET PJ = 17  
20 LET J2 = 34  
30 LET J4 = PJ + J2  
40 ? J4  
50 END

2. 10 LET B = 2  
20 ? B  
30 LET B = 100  
40 ? B  
50 END

3. 10 LET X1 = 2  
20 LET X2 = X1 \* 5  
30 LET X3 = X2 / X1  
40 ? X3  
50 END

4. 10 LET E6 = 3  
20 LET E7 = 12  
30 ? "PRODUCT",  
"QUOTIENT"  
40 ? E6 \* E7, E7 / E6  
50 END





---

### Program

### Output

```
5.10 LET HI = 16
    20 LET HJ = HI + 4
    30 ?HJ + 10
    40 END
```

```
6.10 LET M = 16
    20 LET N = 14
    30 ?M + N
    40 LET N = 12
    50 ?M + N
    60 END
```

```
7.10 LET Z1 = 8
    20 LET Z2 = Z1 - 2
    30 ?Z2 + Z1 / 2
    40 END
```

```
8.10 LET T1 = 6
    20 LET T1 = 7
    30 ?T1
    40 GOTO 30
    50 END
```

```
9.10 LET J = 11
    20 LET K = 22
    30 LET J = 17
    40 ?K + J
    50 END
```



# PROGRAMMER'S PASTIME #26

In each program there is at least one mistake. Find the mistakes and circle the line numbers where you found them. Then write each statement the correct way in the space to the right.

## Program

## Correction

### Example:

```
10 LET 4 = D
20 ? D
30 END
```

```
10 LET D = 4
```

```
1. 10 LET L = 44
   20 LET 6 = M
   30 ? L + D
   40 END
```

```
2. 10 ? Y
   20 LET Y = 66
   30 END
```

```
3. 10 LET B52 = 6
   20 LET H = 3
   20 ? H
   40 END
```

```
4. 10 ? ``A + B``; A + B
   20 LET A = 3
   30 LET B = 4
   40 END
```





---

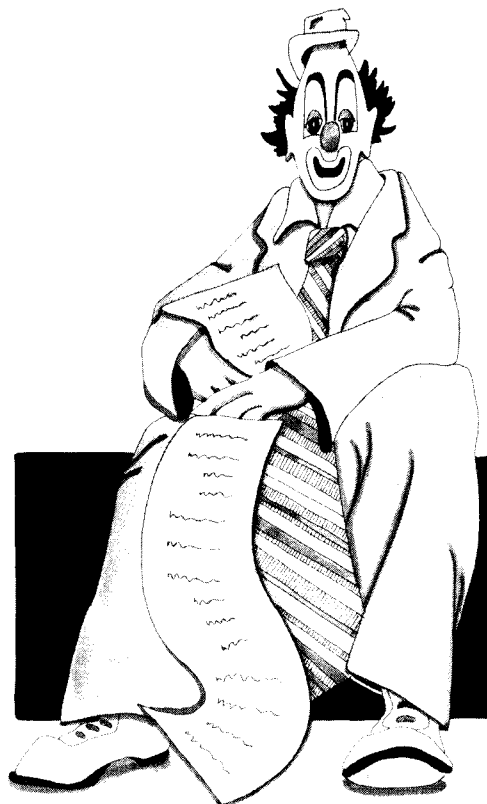
**Program**

```
5.10 LET XY = 5  
20 LET 3 = VW  
30 ? XY + VW  
40 END
```

```
6.10 LET X = 99  
20 LET C = 4  
? X - C  
40 END
```

```
7.10 LET D + 2 = 10  
20 LET E = 4  
30 ? D - E  
40 END
```

```
8.10 LET XYZ = 2  
20 LET B1 = 9  
30 ? B1  
40 END
```

**Correction**

# PROGRAMMER'S PASTIME #27

Using any shortcuts you have learned so far, rewrite the long programs below to make them as short as possible.

## Program

### Example

```
10 LET B = 49
20 LET E = 409
30 PRINT B + E
40 END
```

## Shortcut

```
10 LET B = 49: LET E = 409
20 ? B + E
30 END
```

```
1. 10 LET R = 50
   20 LET N = 22
   30 ? R - N
   40 ? R + N
   50 END
```

```
2. 10 LET S1 = 98
   20 LET S2 = 89
   30 LET S3 = 889
   40 PRINT ``S3 - S2 - S1 = ``; S3 - S2 - S1
   50 PRINT ``S2 * S1 = ``; S2 * S1
   60 END
```

```
3. 10 LET U = 40
   20 LET T = 20
   30 ? U + T
   40 ? U - T
   50 ? U * T
   60 ? U / T
   70 END
```

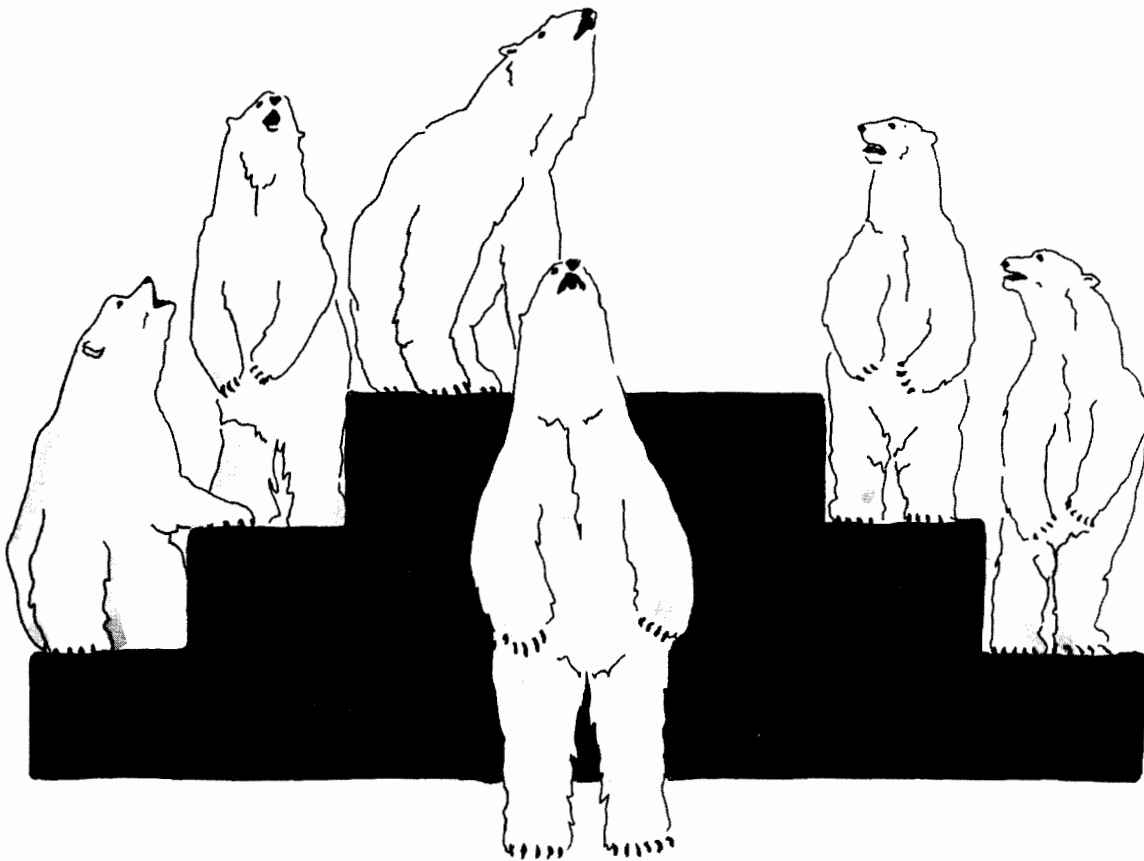
---

**Program**

```
4. 10 LET Y2=2
    20 LET Y3=22
    30 LET Y4=Y2+2
    40 LET Y5=Y3+22
    50 PRINT "Y4="; Y4
    60 PRINT "Y5="; Y5
    70 END
```

**Shortcut**

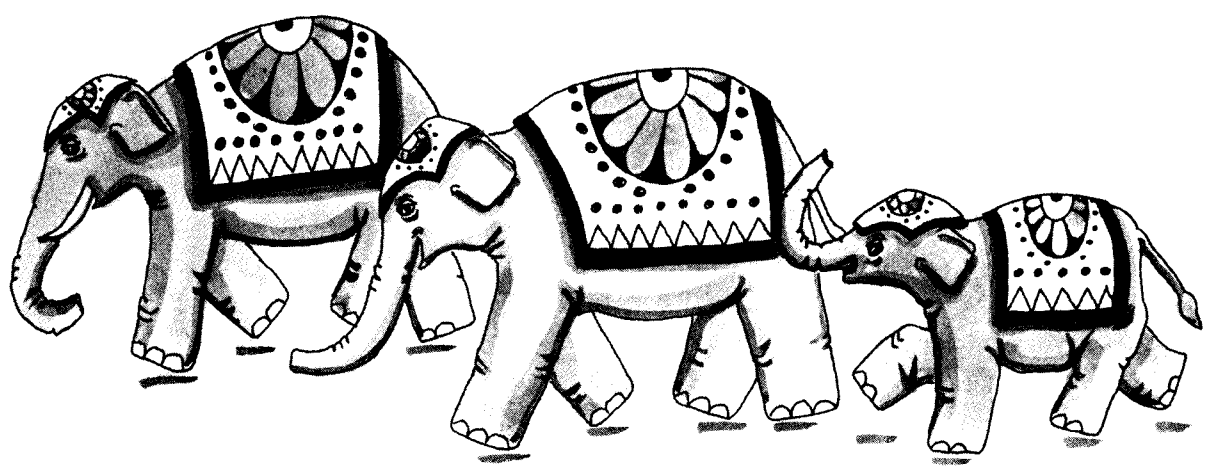
```
5. 10 LET D2=9
    20 LET D4=18
    30 ? "D2+D4=";
    40 ? D2+D4
    50 ? "D4-D2=";
    60 ? D4-D2
    70 END
```



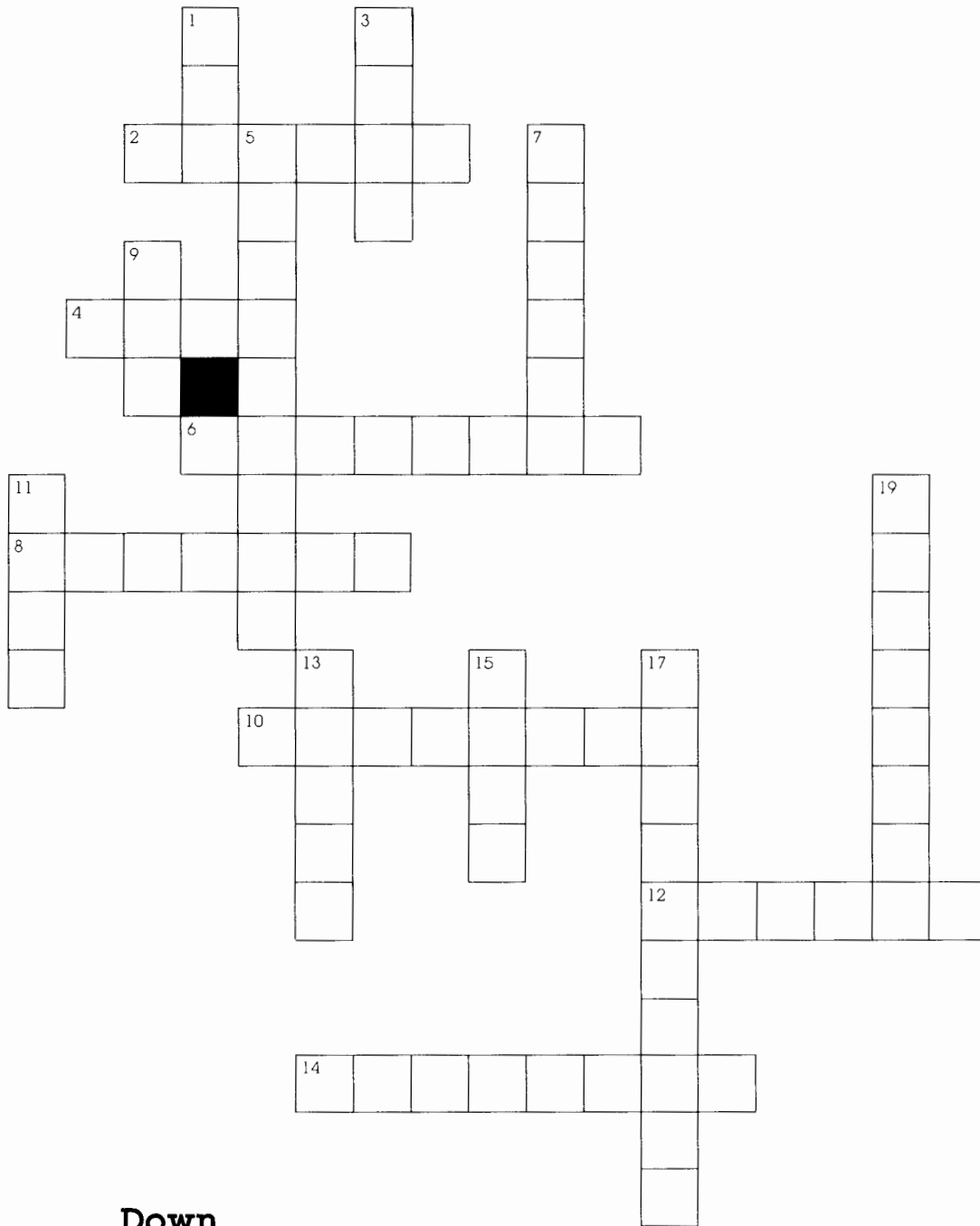
# PROGRAMMER'S PASTIME #28

For each E notation number, write the whole number or decimal that it stands for.

E Notation	Whole number or decimal
<b>Example:</b> $7.982 E + 07$	79820000
1. $3.26 E + 11$	_____
2. $1.23 E - 04$	_____
3. $7.2 E + 08$	_____
4. $4.679 E - 06$	_____
5. $2.22 E + 07$	_____
6. $6.46982 E - 03$	_____
7. $5.3211 E + 12$	_____
8. $9.011 E - 05$	_____
9. $8.0045 E + 13$	_____
10. $1.467 E - 07$	_____
11. $8.006 E + 09$	_____
12. $6.002 E - 03$	_____
13. $3.9826 E - 05$	_____
14. $1.976345 E + 08$	_____



# COMPONENT 4 FUN PAGE



### Down

- |   |  |
|---|--|
| <p>1. How many pieces of information can a memory cell store?</p> <p>3. If PET sees a variable that has not been introduced by a LET statement, PET will automatically give that variable a value of _____.</p> <p>5. PET's memory can be thought of as electronic _____.</p> | <p>7. Whatever PET prints is called _____.</p> <p>9. The _____ statement assigns a value to a variable.</p> <p>11. Information is also called _____.</p> <p>13. In a LET statement, the variable must always come <b>before</b> the _____.</p> |
|---|--|

15. When you introduce the same variable more than once in a program, PET will remember the \_\_\_\_\_ value you gave it.
17. We can use commas and \_\_\_\_\_ to change the arrangement of the output in a program.
19. PET will change numbers with more than nine digits into E \_\_\_\_\_.
4. A mailbox in PET's memory is also called a memory \_\_\_\_\_.
6. A memory cell consists of the address and the \_\_\_\_\_.
8. Each memory cell mailbox has its own \_\_\_\_\_.
10. The address of a memory cell is also called a \_\_\_\_\_.
12. We can use \_\_\_\_\_ to shorten our program in between LET statements.

**Across**

2. Which part of PET's brain allows it to do many of the tricks we teach it?
14. The type of number PET can't understand is a \_\_\_\_\_.

**Evaluate Yourself**

1. Component 4 was \_\_\_\_\_ because \_\_\_\_\_
2. The best parts of the component were \_\_\_\_\_
3. The parts I liked the least were \_\_\_\_\_
4. The most valuable thing I learned in this component was \_\_\_\_\_ because \_\_\_\_\_

I'm getting good at talking to PET!



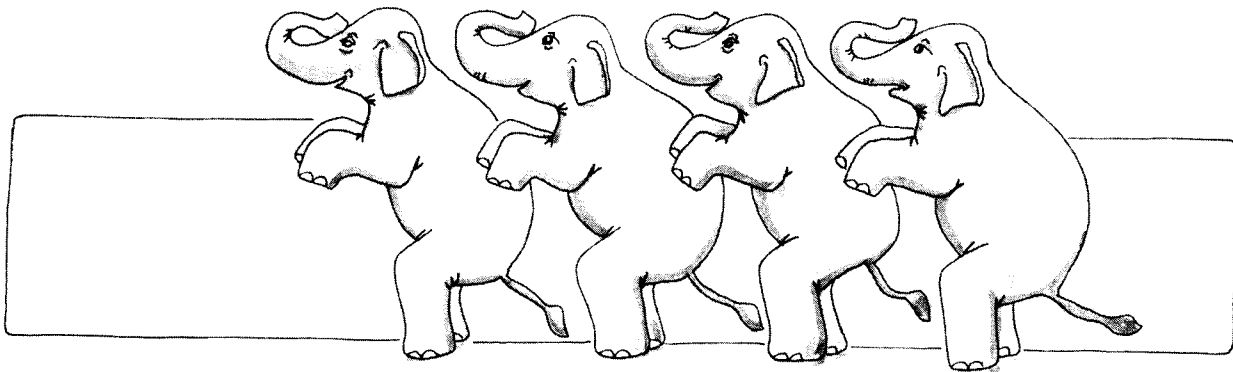
# PROGRAMMER'S PASTIME #29

Read each program. Then write what you think PET would print as the OUTPUT.  
Run the programs on PET to check your answers.

## Program

## Output

- 10 FOR Q = 2 TO 6  
20 ?Q  
30 NEXT Q  
40 END
- 10 FOR Q = 2 TO 4  
20 ? ``Q = ``; Q  
30 NEXT Q  
40 END
- 10 FOR A = 1 TO 5  
20 ? ``HELLO FRIEND!``  
30 ? ``HOW ARE YOU?``  
40 NEXT A  
50 END
- 10 FOR D = 1 TO 3  
20 ?D  
30 ?D + 10  
40 NEXT D  
50 END
- 10 LET P = 3  
20 FOR Q = 1 TO 3  
30 ? P `` + `` Q `` = ``; P + Q  
40 NEXT Q  
50 END



---

**Program**

**Output**

6. 10 FOR B = 1 TO 5  
20 ? ``B``, ``B+B``, ``B\*B``  
30 ? B, B+B, B\*B  
40 NEXT B  
50 END

7. 10 ? ``MULTIPLICATION TABLE FOR 7``  
20 FOR K = 1 TO 12  
30 ? K, ``TIMES 7 = ``; K\*7  
40 NEXT K  
50 END

8. 10 FOR G = 1 TO 10  
20 ? ``♥``  
30 NEXT G  
40 END

9. 10 FOR G = 1 TO 10  
20 ? ``♥``;  
30 NEXT G  
40 END

10. 10 FOR S = 1 TO 10  
20 LET S = S\*S  
30 ? S, S / S  
40 NEXT S  
50 END

Can you explain how this program works? \_\_\_\_\_

---

---

---

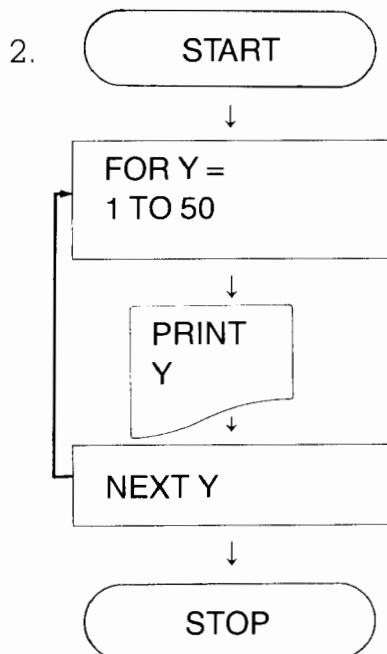
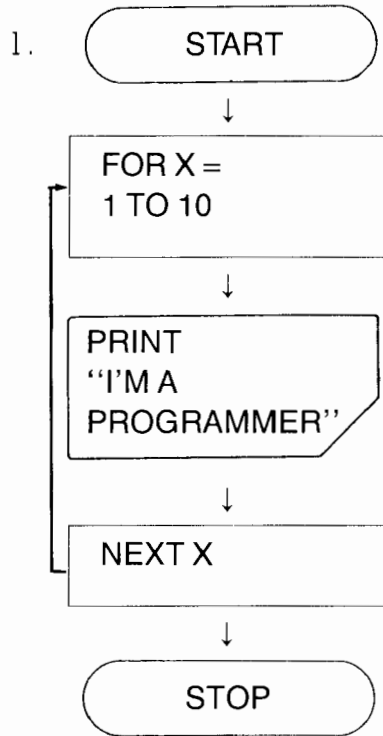


# PROGRAMMER'S PASTIME #30

Write a program for each flow chart. Be sure to use a FOR-NEXT loop. Run your programs on PET to make sure they work.

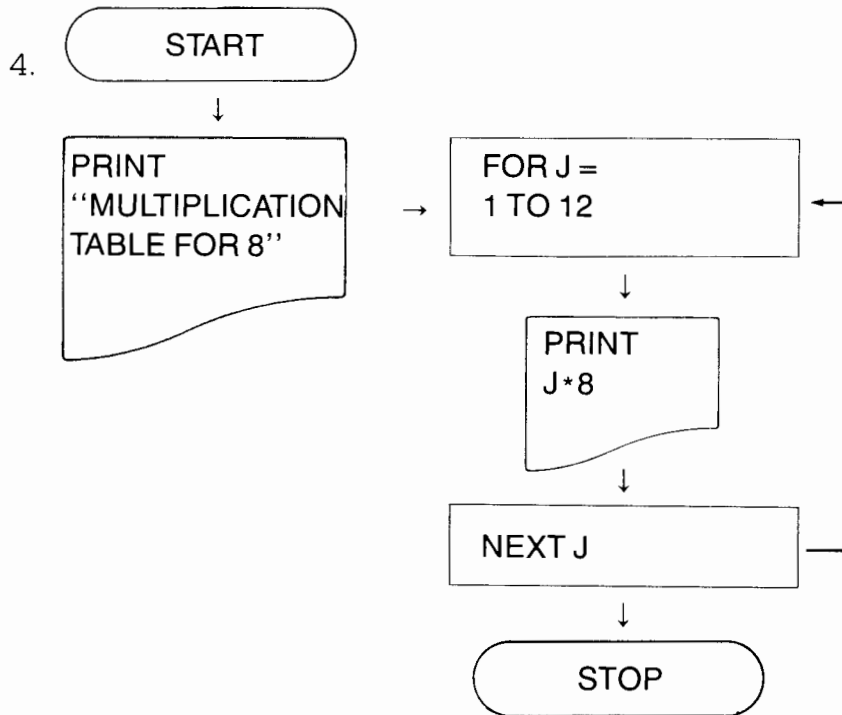
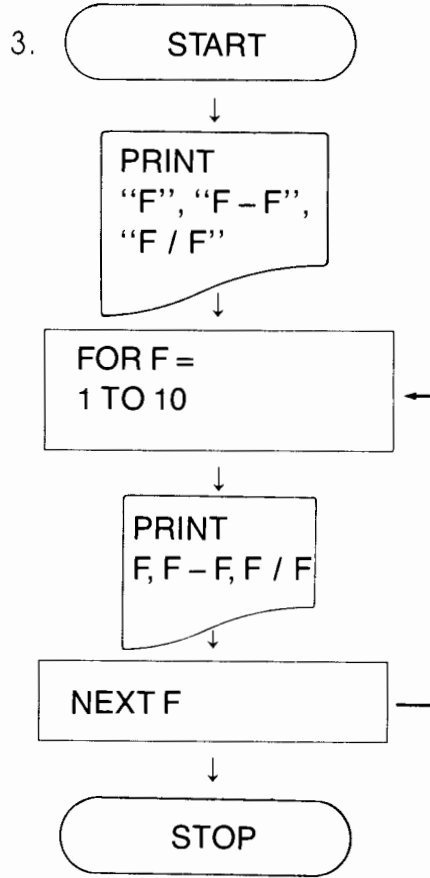
## Flow chart

## Program



**Flow chart**

**Program**



---

For each program description, write an algorithm in flow chart form. Then write the program. Run each program on PET to make sure it works.

**Description**

**Flow chart**

**Program**

5. Add something new to the program in #4 so it prints: J; ``TIMES 8 = ``; J\*8 each time the loop is done.

6. Write a program that prints the numbers from 1 to 20, their squares ( $X^2$ ), and their cubes ( $X^3$ ).

---

**Description****Flow Chart****Program**

7. Write a program that prints \* twenty times on one line.

8. Write a program that introduces  $D = 5$  and  $P = 1$  to 5. Make the program add  $D$  plus each value of  $P$ , and print the sums of  $D + P$ .

# PROGRAMMER'S PASTIME #31

In each program there is one mistake. Find each mistake and circle the line number where you found the mistake. Then write the statement the correct way in the space to the right.

## Program

## Correction

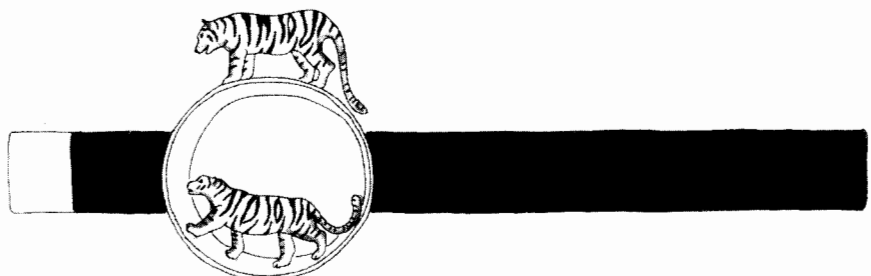
```
1. 10 FOR P = 1 - 40
    20  ?P
    30 NEXT P
    40 END
```

```
2. 10 FOR W IS 6 TO 30
    20  ?W
    30 NEXT W
    40 END
```

```
3. 10 FOR E = 3 TO 10
    20  ?E
    30 E NEXT
    40 END
```

```
4. 10 FOR L = 1 TO 4
    20  ?L
    30  ?L*2
    40
    50 END
```

```
5. 10 LET G = 4
    20 FOR H = 5 TO 9
    30  ?G+H
    40 NEXT G
    50 END
```



# PROGRAMMER'S PASTIME #32

Read each program. Write what you think PET would print as the OUTPUT. Run the programs on PET to check your answers.

## Program

## Output

- 10 FOR F = 0 TO 8 STEP 2  
20 ? F  
30 NEXT F  
40 END
- 10 FOR J = 18 TO 0 STEP - 3  
20 ? J  
30 NEXT J  
40 END
- 10 FOR B2 = 3 TO 21 STEP 3  
20 ? ``HOWDY``  
30 NEXT B2  
40 END
- 10 LET N = 5  
20 FOR T = 1 TO N  
30 ? T  
40 NEXT T  
50 END
- 10 LET M2 = 10  
20 FOR S = 0 TO 12 STEP M2/5  
30 ? S  
40 NEXT S  
50 END



---

### Program

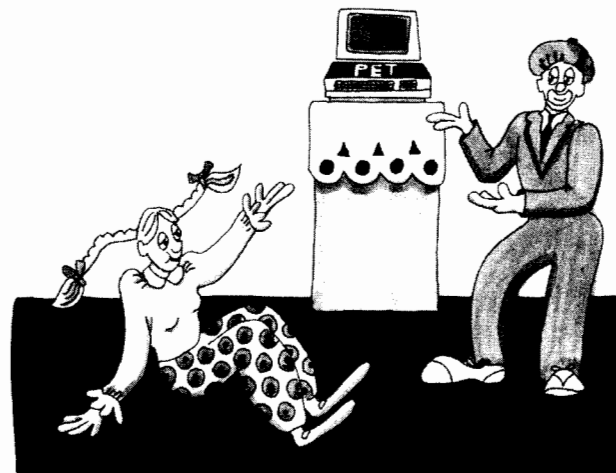
```
6. 10 ? "IF SEPT 1 IS A SUNDAY THEN"  
    20 FOR JJ = 1 TO 31 STEP 7  
    30 ? "SEPT"; JJ, "IS A SUNDAY"  
    40 NEXT JJ  
    50 END
```

```
7. 10 LET PX = 8  
    20 FOR A7 = 0 TO 10 STEP PX / 4  
    30 ? A7  
    40 NEXT A7  
    50 END
```

```
8. 10 FOR ZZ = 1 TO 14 STEP 4  
    20 ? ZZ  
    30 NEXT ZZ  
    40 END
```

```
9. 10 FOR BD = 20 TO 2 STEP - 5  
    20 ? BD  
    30 NEXT BD  
    40 END
```

### Output



# PROGRAMMER'S PASTIME #33

For each program description, write an algorithm in flow chart form. Then write the program. Run each program on PET to make sure it works.

## Description

## Flow chart

## Program

1. Write a program that tells PET to count from 0 to 40 by fours.

2. Write a program that tells PET to count backwards from 8 to 0.

3. Write a program that tells PET to print "HELLO" five times. Use a STEP statement.



---

**Description****Flow chart****Program**

4. Write a program that tells PET to print the numbers 0 through 21 STEP N / 4. Make N = 12.

5. Write a program that tells PET to print the numbers 1, 4, 7, 10 and 13. Use a STEP statement.

# PROGRAMMER'S PASTIME #34

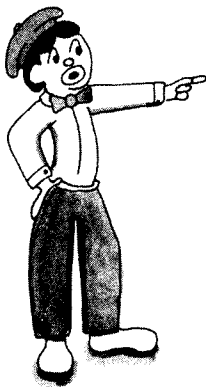
Study each program. Write what you think PET would print as the OUTPUT. Run each program to check your answers.

Program	Output
---------	--------

1. 10 LET C = 0 20 FOR FL = 1 TO 100 STEP 10 30 ? ``THINK`` 40 LET C = C + 1 50 ? C 60 NEXT FL 70 END	
--	--

2. 10 LET C = 0 20 ? ``BRAIN POWER`` 30 LET C = C + 1 40 ? C 50 GOTO 20	
---	--

3. 10 LET C = 0 20 FOR FL = 1 TO 8 30 LET C = C + 1 40 ? C 50 ? ``AWESOME`` 60 NEXT FL 70 END	
---	--



---

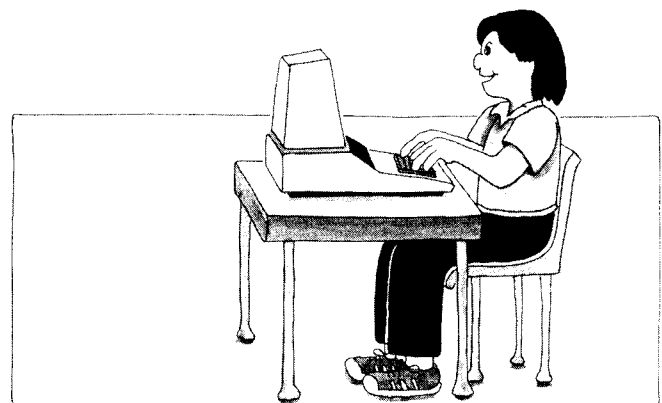
## Program

```
4.10 LET C = 0
    20 FOR Z = 1 TO 50 STEP 10
    30   ? ``RAINBOW``
    40   LET C = C + 1
    50 NEXT Z
    60 ? ``I PRINTED``
    70 ? ``RAINBOW``
    80 ? C ``TIMES``
    90 END

5.10 LET C = 0
    20 FOR K = 5 TO 1 STEP - 1
    30   ? K
    40   ? ``COOKIES``
    50   LET C = C + 1
    60 NEXT K
    70 ? ``ONCE THERE WERE`` C
        ``COOKIES``
    80 ? ``NOW THERE ARE`` K
    90 END

6.10 LET C = 0
    20 FOR FL = 1 TO 4
    30   LET C = C + 1
    40   ? C
    50   ? ``JELLY BEANS``
    60 NEXT FL
    70 ? ``A TOTAL OF`` C ``JELLY
        BEANS``
    80 END
```

## Output



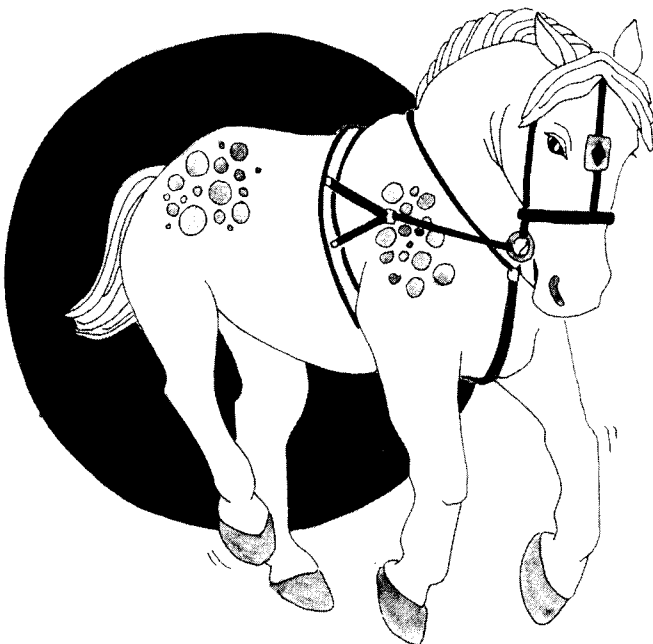
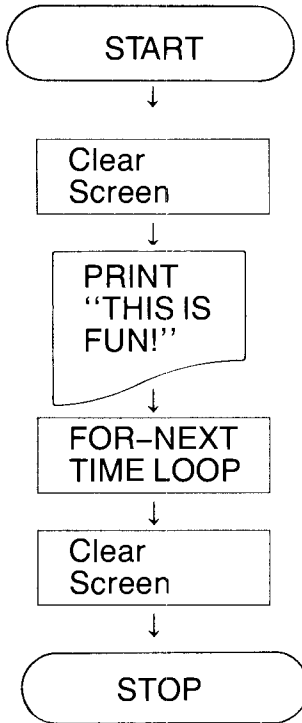
# PROGRAMMER'S PASTIME #35

Write a program for each flow chart, then run the programs.

## Flow Chart

## Program

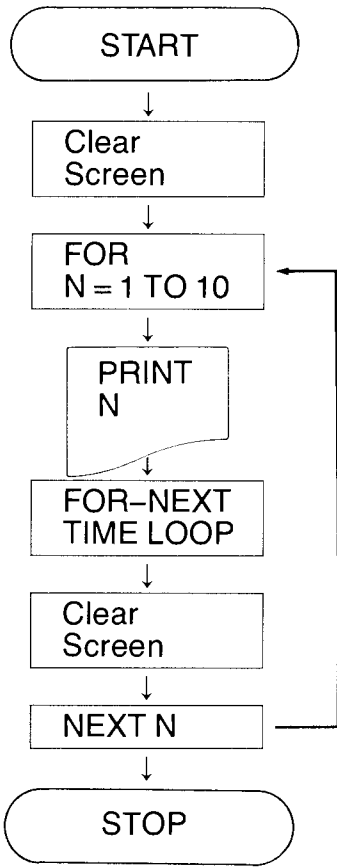
1.



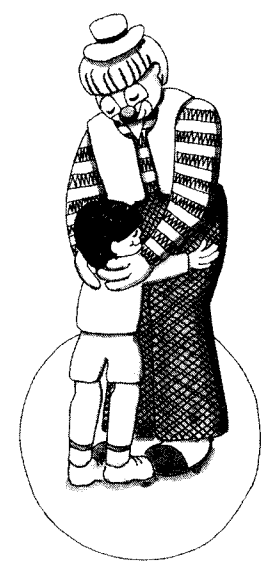
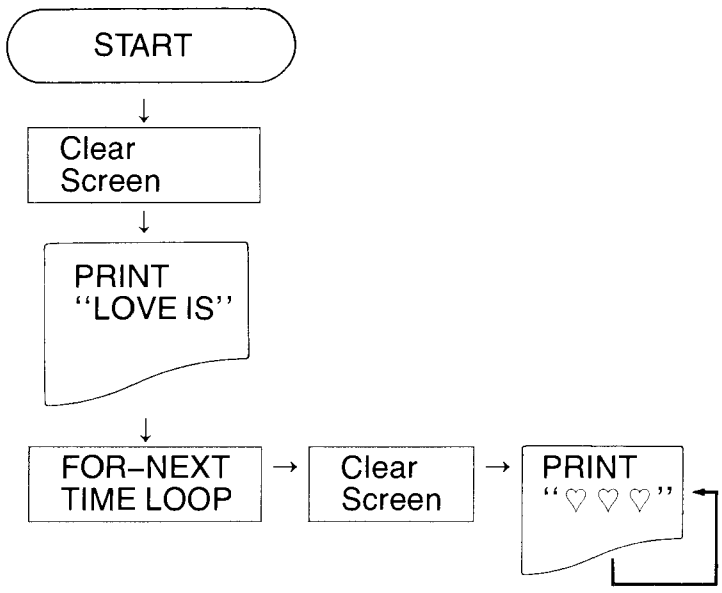
**Flow Chart**

**Program**

2.

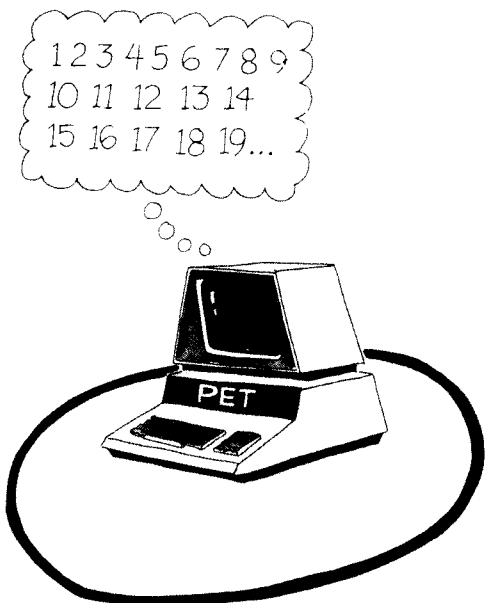
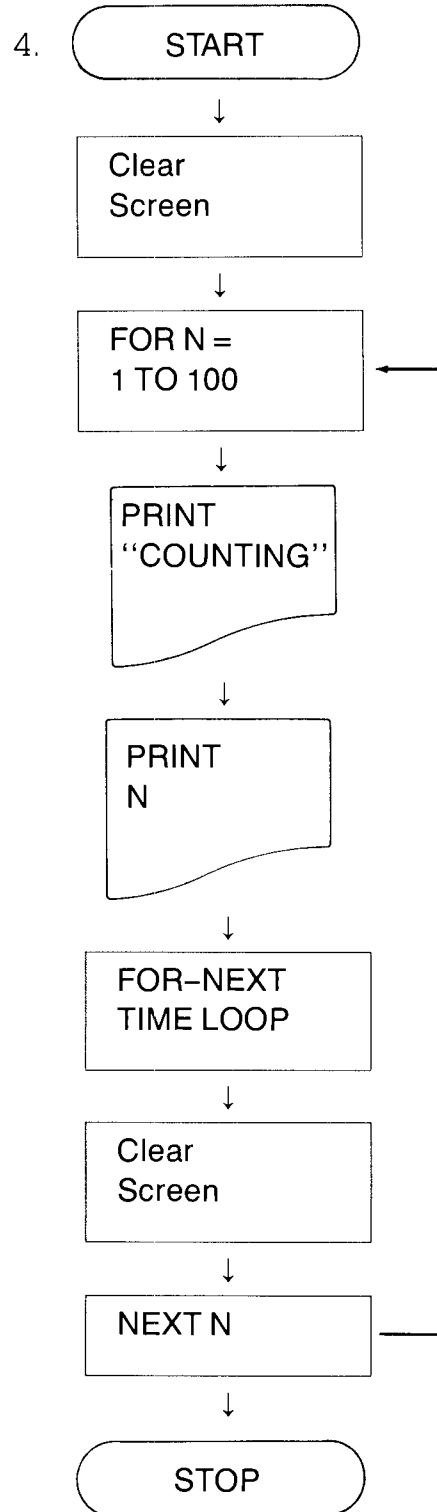


3.



**Flow chart**

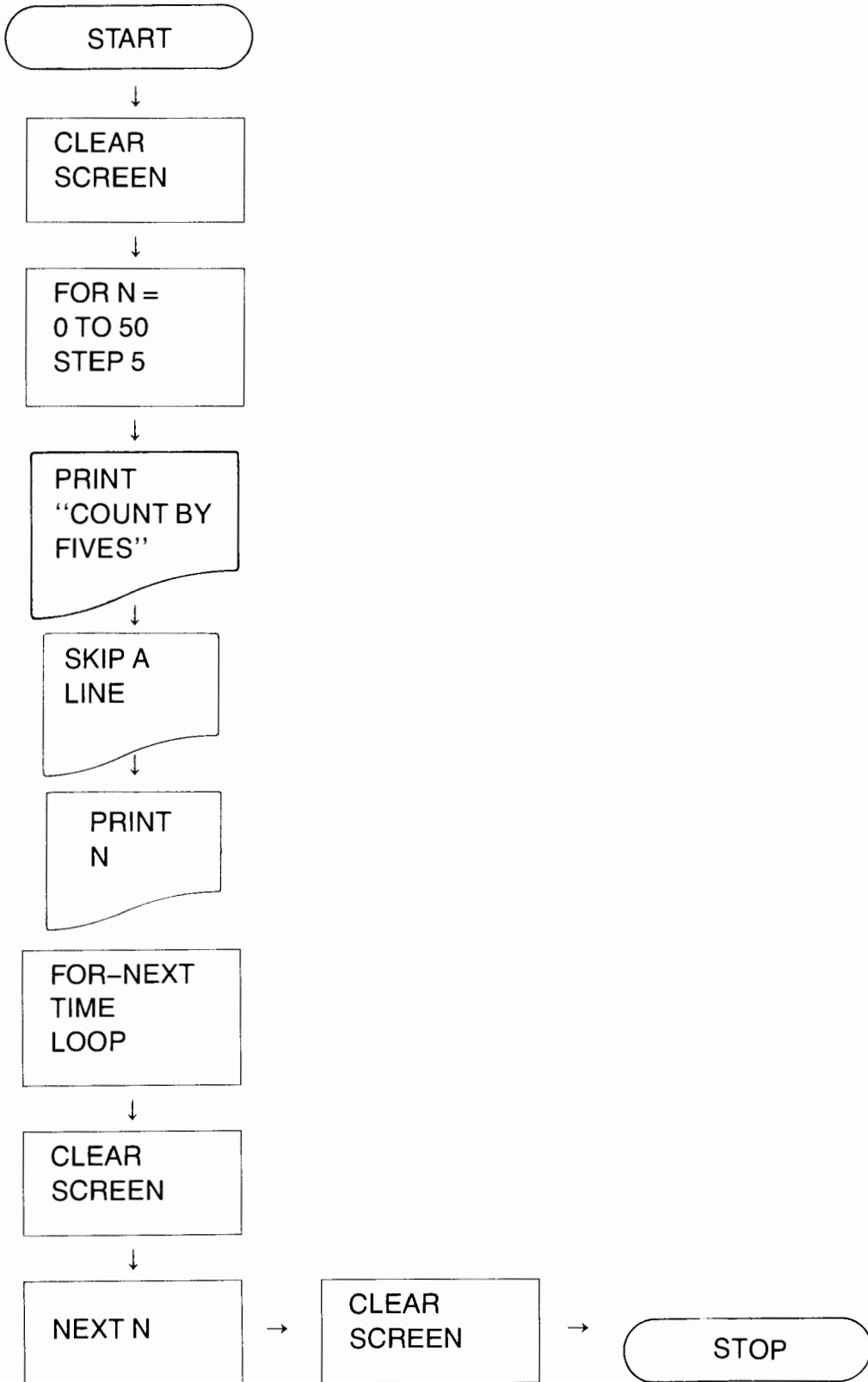
**Program**



**Flow chart**

**Program**

5.



# PROGRAMMER'S PASTIME #36

You have learned how to program PET to move down a number of lines on the screen and then begin printing. To do this, you used a statement like this:

```
? " CLR HOME CURSR CURSR CURSR "
```

This tells PET to move down four lines, leaving three blank lines.

By using the cursor keys in other ways, you can also tell PET to move across the screen or **up** a few lines.

To get PET to move up three lines, leaving a blank two lines long, type:

```
? " SHIFT CURSR CURSR CURSR "
```

If you want PET to move up **six** lines, leaving a blank of five lines, type  $(6 + 2 = )8$  `SHIFT` and `CURSR` keys.

The FORMULA for moving up is: to make PET move N lines up, leaving a blank of N - 1 lines, type N + 2 `SHIFT` and `CURSR` keys in quotation marks.

When you type " `SHIFT CURSR` " inside quotation marks, PET prints a circle in reverse field like this:



HINT: Before you can tell PET to move up on the screen the cursor must already be on a lower line of the screen!

To make PET move to the **right** type:

```
? " CURSR the number of spaces you want PET to move "
```

When you type `CURSR` inside quotation marks, PET prints a right bracket in reverse field like this: `]`.



The program below tells PET to print an \*, move five spaces to the right and print another \* in the **next** space.

<b>Program</b>	<b>Output</b>
10 ? " SHIFT CLR HOME "	*            *
20 ? " * CRSR CRSR CRSR CRSR CRSR * "	

To make PET move to the **left** type: SHIFT CRSR in quotation marks.

Let's add another line to our program.

<b>Program</b>	<b>Output</b>
30 ? " * CRSR CRSR CRSR CRSR CRSR CRSR CRSR CRSR	*            *
CRSR CRSR * SHIFT CRSR CRSR CRSR CRSR * "	*            *            *

LINE 30 tells PET to print an \*, count over 10 spaces, print an \* in the **next** space, and then count **left** two spaces and print another \* in the **next** space to the left.

If you want PET to move six spaces to the left, leaving a blank of five spaces, type (6 + 1 = )7 " SHIFT CRSR " keys.

The FORMULA for moving **left** is: to make PET move N spaces to the left, leave a blank N - 1 spaces wide, type N + 1 SHIFT CRSR keys in quotation marks.

When you type " SHIFT CRSR " inside quotation marks, PET prints a vertical line in reverse field like this: 

--

HINT: Before you can tell PET to move left on the screen, the cursor must already be somewhere near the right half of the screen!

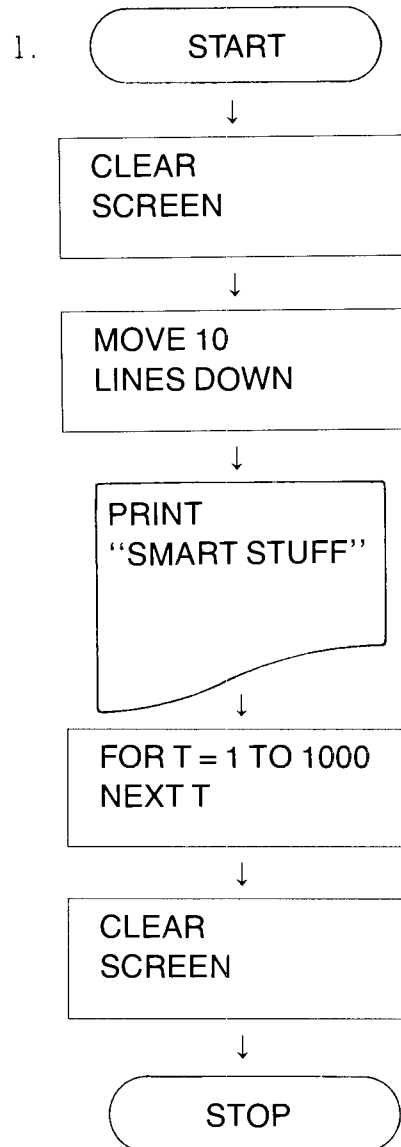


---

Write a program for each flow chart. Use the tricks you have learned to make PET clear the screen and place the writing on a certain line. RUN each program on PET.

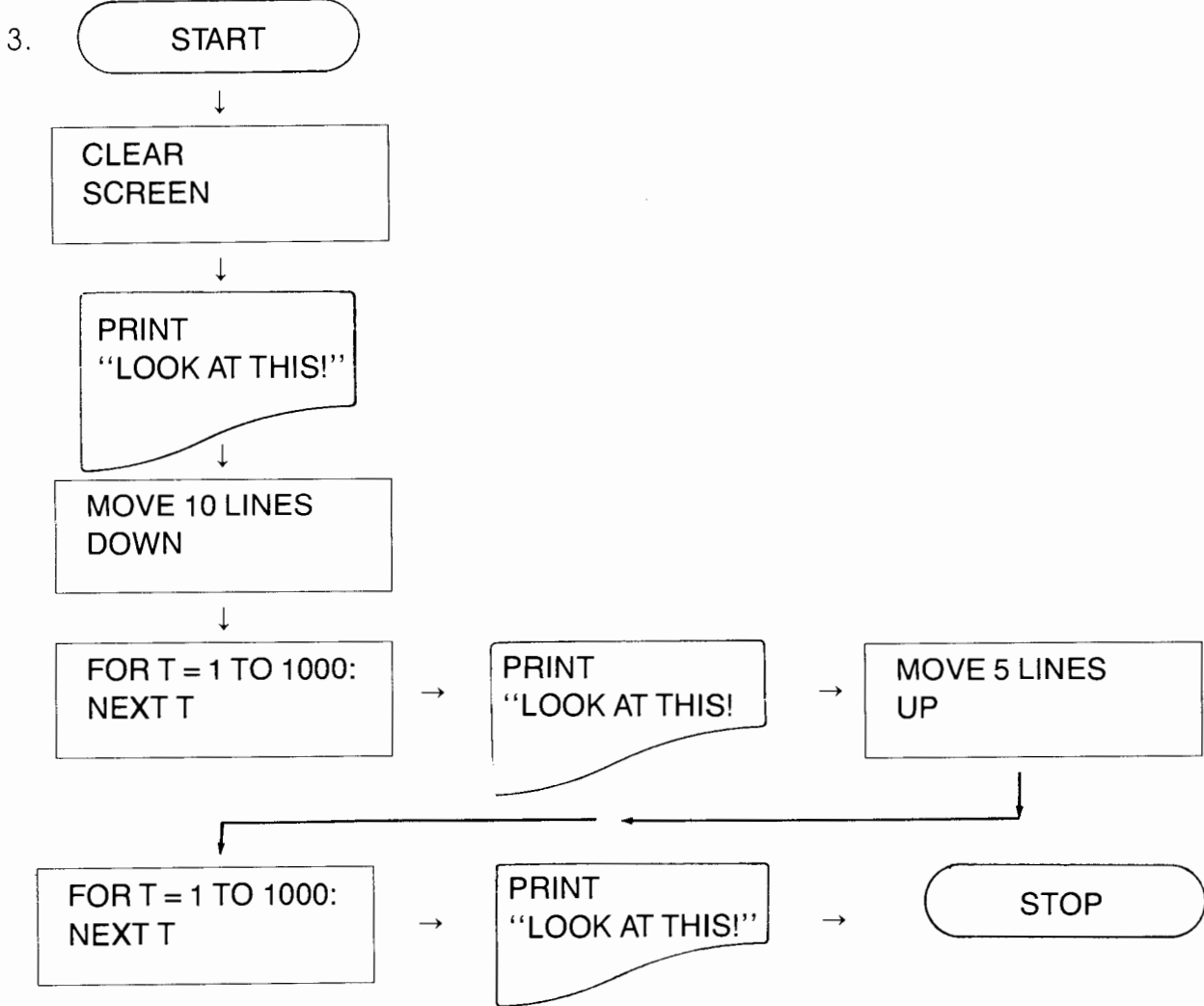
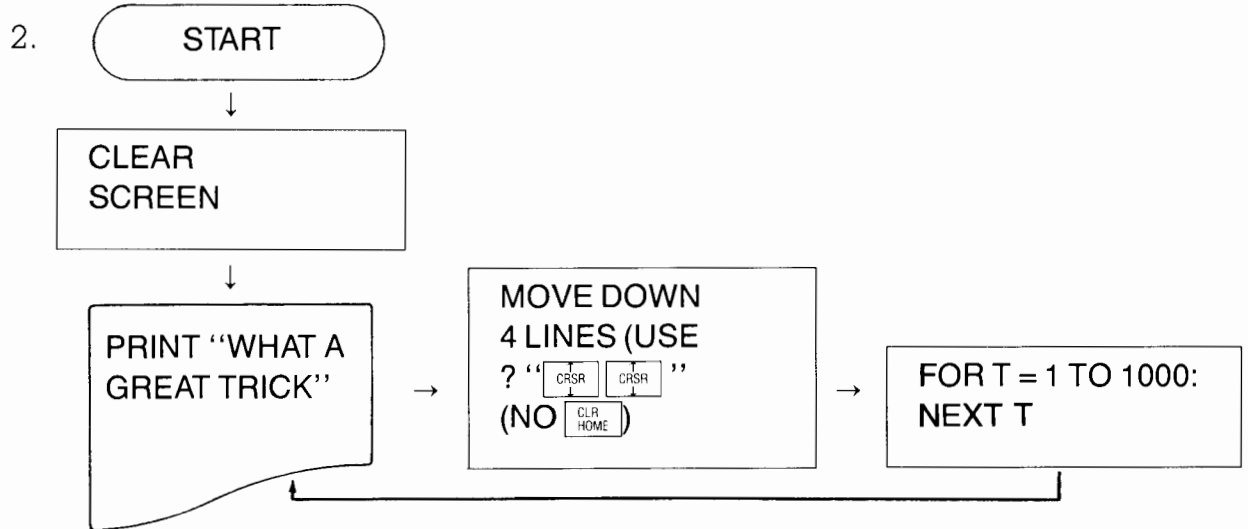
**Flow chart**

**Program**



**Flow chart**

**Program**



**Flow chart**

**Program**

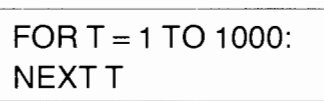
4.



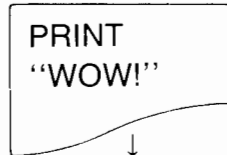
START



CLEAR  
SCREEN



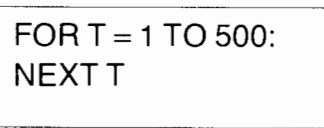
FOR T = 1 TO 1000:  
NEXT T



PRINT  
"WOW!"



MOVE DOWN  
15 LINES



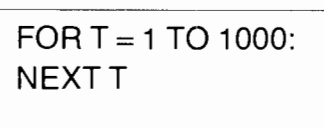
FOR T = 1 TO 500:  
NEXT T



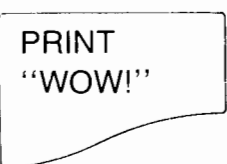
PRINT  
"WOW!"



MOVE UP  
7 LINES



FOR T = 1 TO 1000:  
NEXT T



PRINT  
"WOW!"

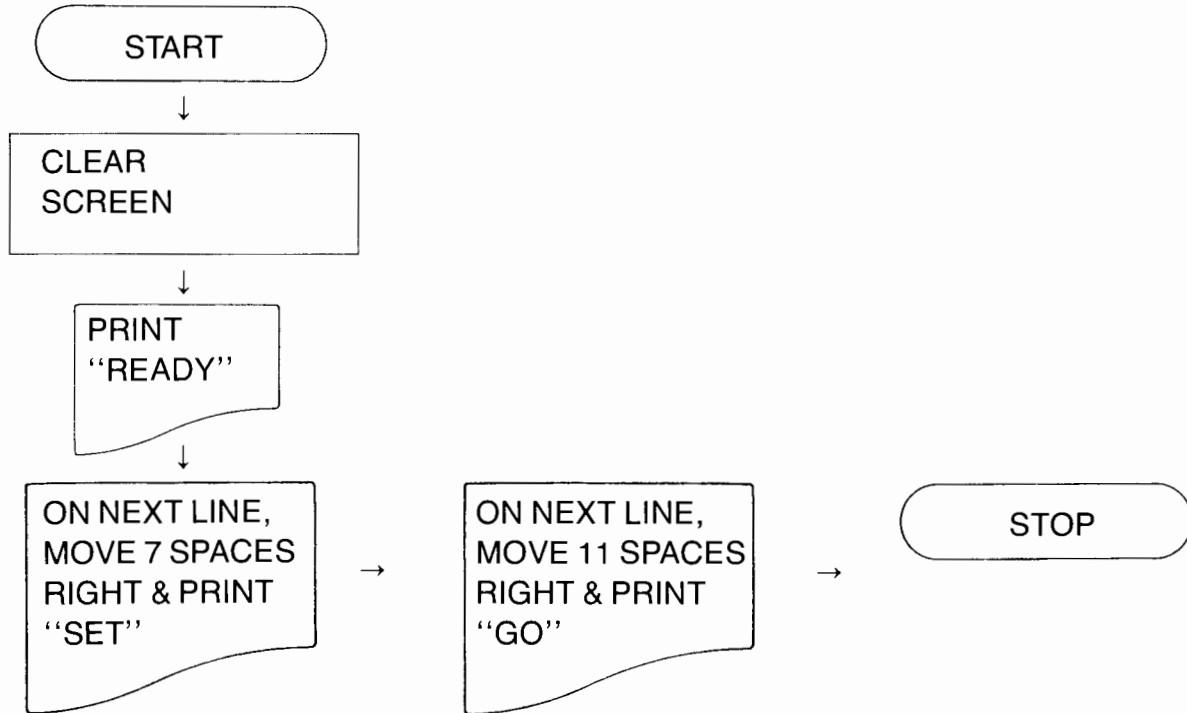


STOP

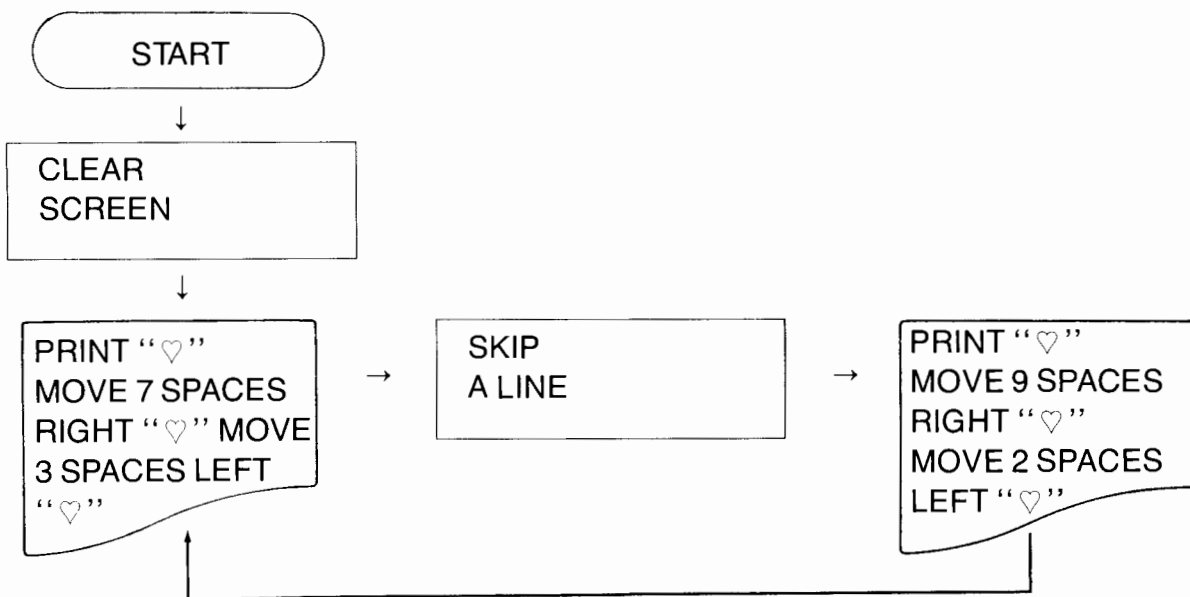
**Flow chart**

**Program**

5.



6.



# JUST FOR FUN

RUN each program below and describe PET's output.

## Program

## Describe output

```

1. 10 ? "SHIFT CLR HOME "
    20 FOR L = 1 TO 39
    30 ? " " SHIFT Q CRSR ";
    40 NEXT L
    50 FOR M = 1 TO 39
    60 ? " " SHIFT CRSR CRSR Q CRSR ";
    70 NEXT M
    80 GOTO 20
  
```

```

2. 10 ? "SHIFT CLR HOME "
    20 FOR L = 1 TO 24
    30 ? " " SHIFT CRSR CRSR Q CRSR ";
           ↑      ↑
           SHIFT SHIFT
           OFF  ON
    40 NEXT L
    50 FOR M = 1 TO 24
    60 ? " " SHIFT CRSR CRSR Q CRSR ";
    70 NEXT M
    80 GOTO 20
  
```

**Note:**  means type the SPACE BAR once.

SHIFT OFF means let go of the  to type the key.

SHIFT ON means press the  again to type the key.

means leave the  down through the whole line unless you see a SHIFT OFF message.

---

**Program**

```
3. 10 ? ``SHIFT CLR HOME ``  
20 ? ``FALLING METEORS``  
30 ? ``SHIFT CLR HOME ``;  
40 FOR L = 1 TO 24  
50 ? ``CRSR SHIFT Q CRSR ``;  
60 NEXT L  
70 FOR M = 1 TO 24  
80 ? ``SHIFT CRSR CRSR CRSR CRSR ``;  
90 NEXT M  
100 GOTO 20
```

**Describe output**

4. As you have probably noticed, PET can count very fast. RUN this program and count to find out how many seconds it takes PET to count from 1 to 2000.

```
10 ? ``SHIFT CLR HOME ``  
20 ? ``GET READY``  
30 FOR T = 1 TO 1000: NEXT T  
40 ? ``START``  
50 FOR T = 1 TO 2000: NEXT T  
60 ? ``STOP``  
70 END
```

How many seconds did it take PET to count from 1 to 2000?

How many seconds do you think it would take YOU to count from 1 to 2000?

5. Write a program to time PET to see how long it takes PET to count from 1 to 5000. The only line that will be different from program #4 is LINE 50.

**Program**

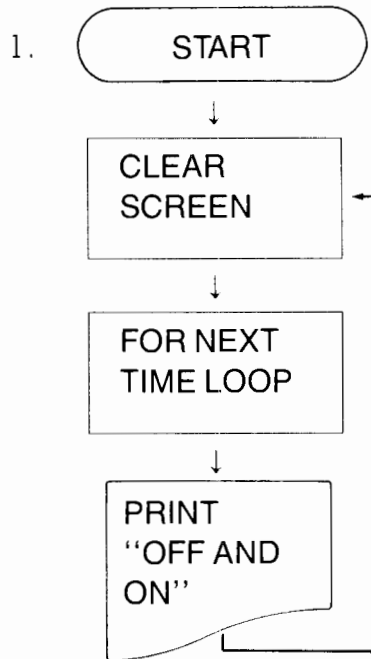
How many seconds did it take PET to count from 1 to 5000?

# PROGRAMMER'S PASTIME #37

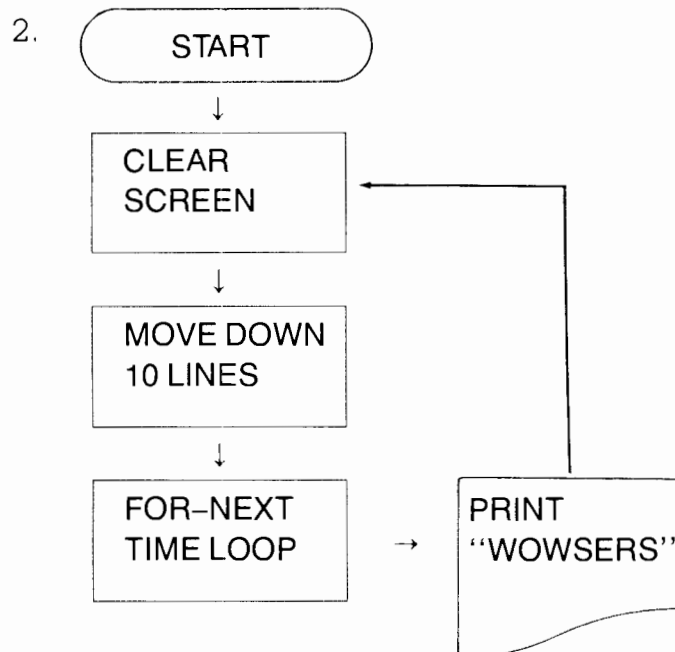
Write a program for each flow chart, then run your programs on PET to make sure they work.

## Flow chart

## Program



This is the basic algorithm for making something blink.





---

3. Use your expertise and your imagination to write two of your own programs that make something blink. You can even make graphics or pictures blink! Don't be afraid to experiment.

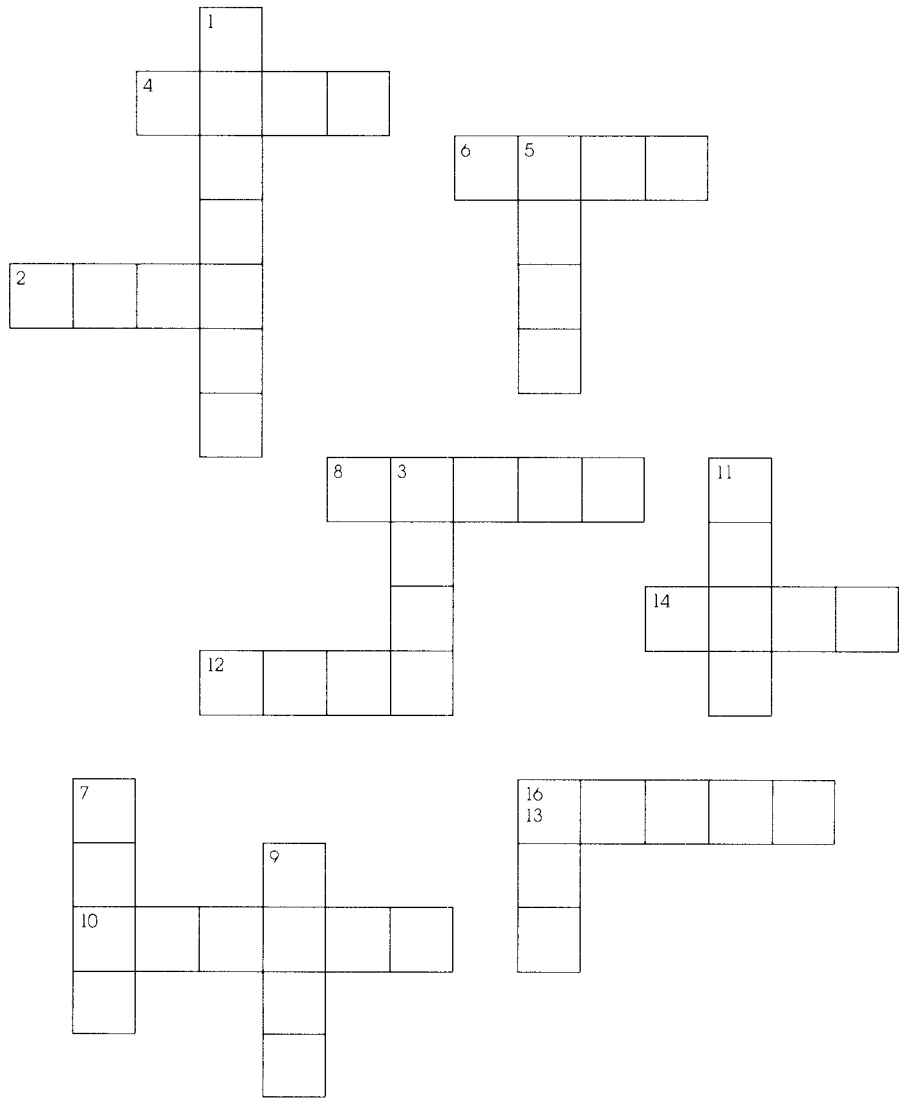
**Flow chart**

**Program**

1.

2.

# COMPONENT 5 FUN PAGE



**Down**


1. A FOR-NEXT loop creates \_\_\_\_\_ controlled loops in a program.
3. A counter also lets you keep track of how many times you have done a \_\_\_\_\_.
5. To slow down PET's printing, use a FOR-NEXT \_\_\_\_\_ loop.
7. When you copy a program onto a cassette tape, you use the \_\_\_\_\_ command.
9. To see how a program is written, type \_\_\_\_\_.
11. Press the OFF  
RVS key and your pro-

gram will \_\_\_\_\_ down.

13. A mistake in a program is called a \_\_\_\_\_.

**Across**

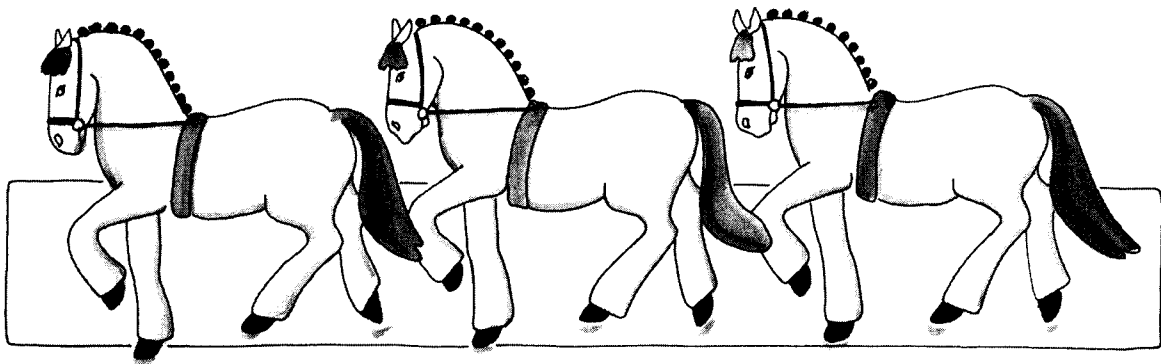
2. Every FOR statement must have a \_\_\_\_\_ statement after it somewhere in the program.
4. We indent the \_\_\_\_\_ of a FOR-NEXT loop.
6. We tell PET to count in a certain pattern by using the \_\_\_\_\_ command.
8. The statement ? " SHIFT CLR  
HOME " tells PET to \_\_\_\_\_ the screen.

- 
10. To make sure your program was copied correctly, use the \_\_\_\_\_ command.
  12. We push  to make a program \_\_\_\_\_ running.
  14. \_\_\_\_\_ tells a program to continue running.
  16. We \_\_\_\_\_ out of a program when we stop it before it has finished running.

### Evaluate Yourself

1. Component 5 was \_\_\_\_\_  
because \_\_\_\_\_  
\_\_\_\_\_
2. The best parts of the component were \_\_\_\_\_  
\_\_\_\_\_
3. The parts I liked the least were \_\_\_\_\_  
\_\_\_\_\_
4. The most valuable thing I learned in this component was \_\_\_\_\_  
because \_\_\_\_\_  
\_\_\_\_\_

Other comments:



# PROGRAMMER'S PASTIME #38

Study each program and write what you think PET would print as the output. Run the programs to check your answers.

## Program

## Output

1. 10 LET F\$ = "44"  
20 LET F = 44  
30 ? F\$ : ? F  
40 END
2. 10 LET G\$ = "6 + 32 = "  
20 LET G = 6 + 32  
30 ? G\$ ; G  
40 END
3. 10 LET A\$ = "ADDITION"  
20 LET B\$ = "SUBTRACTION"  
30 ? A\$ , S\$  
40 LET A = 8 + 8  
50 LET S = 8 - 8  
60 ? "8 + 8 = " ; A , "8 - 8 = " ; S  
70 END
4. 10 LET Q\$ = "HI HO"  
20 LET R\$ = "SILVER!"  
30 ? Q\$ , R\$  
40 GOTO 30
5. 10 LET Z\$ = "YOU'RE OUTA SIGHT!"  
20 FOR C = 1 TO 20  
30 ? Z\$  
40 NEXT C

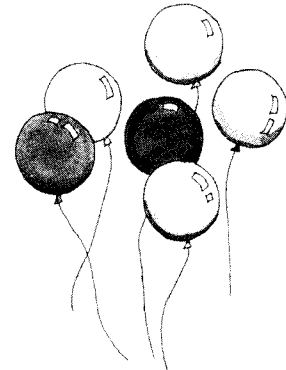
# PROGRAMMER'S PASTIME #39

Each program contains at least one mistake (there may be more!). Find the mistake, circle the line number where you found it, then write the statement(s) the correct way in the space to the right.

## Program

```
1. 10 LET AZ$ = "YES"  
   20 LET BY$ = "NO"  
   30 ? AZ$, BY$  
   40 END
```

## Correction



```
2. 10 LET T$ = "THE TIME"  
   20 LET U$ = "IS NOW"  
   30 ? T, U  
   40 END
```

```
3. 10 LET J$ = UP, UP  
   20 LET K$ = AND AWAY  
   30 ? J$ : ? K$  
   40 END
```

```
4. 10 LET "PARTRIDGE IN" = P$  
   20 LET "A PEAR TREE" = T$  
   30 ? P$, T$  
   40 END
```

# PROGRAMMER'S PASTIME #40

Study each program. Write what you think PET would print as the output. You may write what you would answer for each INPUT statement. Run the programs to check your work.

## Program

## Output

1. 10 ? "HOW OLD ARE YOU";  
20 INPUT A  
30 ? "WHAT'S IT LIKE TO BE"; A;  
40 INPUT A\$  
50 ? "I'M GLAD TO HEAR IT'S  $\phi$ "; A\$  
60 END
2. 10 ? "HOW OLD ARE YOU?"  
20 INPUT A  
30 ? "WHAT'S IT LIKE TO BE"; A; "?"  
40 INPUT A\$  
50 ? "SO YOU ARE  $\phi$ " A\$ " $\phi$  TODAY"  
60 END
3. 10 ? "HOW MANY BROTHERS AND"  
20 ? "SISTERS DO YOU HAVE?"  
30 ? "TYPE NUMBER OF BROTHERS,"  
40 ? "A COMMA,"  
50 ? "AND NUMBER OF SISTERS"  
60 INPUT B, S  
70 LET T = B + S  
80 ? "SO YOU HAVE"; T; " $\phi$  SIBLINGS"  
90 END

---

**Program**

```
4. 10 ? "CHOOSE TWO NUMBERS
    AND"
    20 ? "I WILL ADD THEM FOR YOU"
    30 ? "TYPE 1ST NUMBER, COMMA,"
    40 ? "THEN TYPE THE 2ND NUMBER"
    50 INPUT F, S
    60 ?
    70 ? F + S = ; F + S
    80 ?
    90 ? "I'M A WHIZ!"
    100 END
```

**Output**

```
5. 10 ? "TYPE IN 2 NUMBERS"
    20 ? "SEPARATED BY A COMMA"
    30 INPUT O, T
    40 ?
    50 ? O + T = ; O + T
    60 ? O - T = ; O - T
    70 ? O * T = ; O * T
    80 ? O / T = ; O / T
    90 ? "SEE... I TOLD YOU"
    100 ? "I WAS A WHIZ!"
    110 END
```



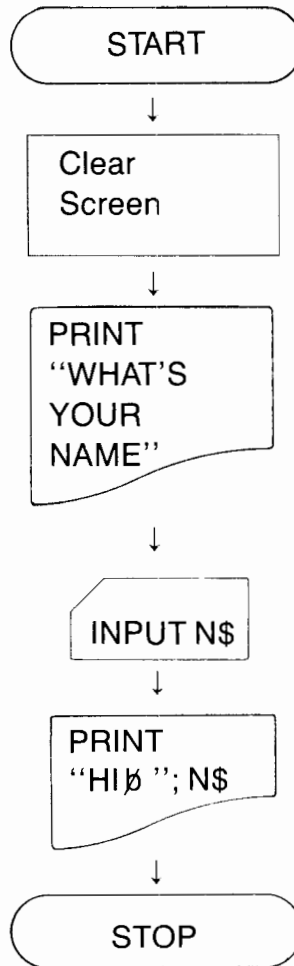
# PROGRAMMER'S PASTIME #41

Write a program for each flow chart. Run your programs on PET to check for bugs.

Flow chart

Program

1.

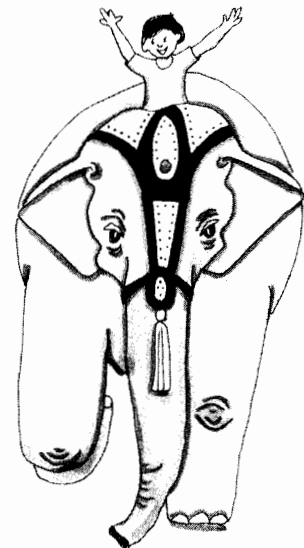
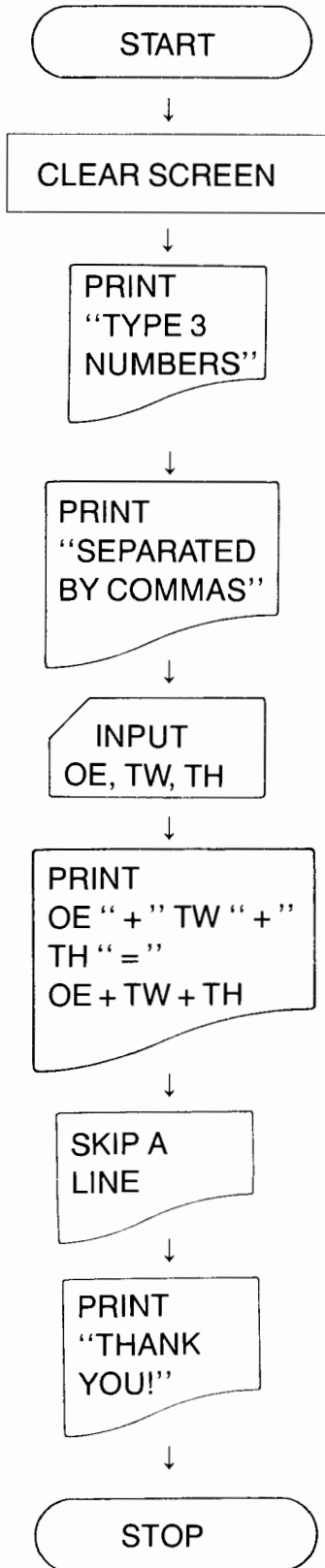




**Flow chart**

**Program**

2.



Flow chart

Program

3.

START



CLEAR  
SCREEN



PRINT  
"WHAT IS  
YOUR  
ADDRESS"



INPUT A\$



SKIP A  
LINE



PRINT  
"YOU LIVE  
AT Ø"; A\$



STOP



---

Write three programs using the INPUT statement. Write the flow chart for the algorithm first, then write the program. Debug your programs by running them on PET.

**Flow Chart**

**Program**

# PROGRAMMER'S PASTIME #42

Write each equation as an IF-THEN statement.

### Question

- Example:** Is A equal to C?
1. Is L\$ equal to "MAYBE"?
  2. Is F1 not equal to FZ?
  3. Is GH greater than HI?
  4. Is S\$ less than or equal to F\$?
  5. Is X times B less than P times Q?
  6. Is T divided by W greater than or equal to W times B?
  7. Is P\$ greater than M\$?
  8. Is the square root of Y equal to D?
  9. Is G\$ not equal to "NO"?
  10. Is 10 divided by 5 less than 14 divided by 2?
  11. Is Y\$ equal to the square root of 64?
  12. Is A plus B greater than D\$?

### IF-THEN statement

IF A = C THEN \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_



# PROGRAMMER'S PASTIME #43

For each question write the COMPLEMENT IF-THEN statement.

**Question**

**Complement**

**Example**

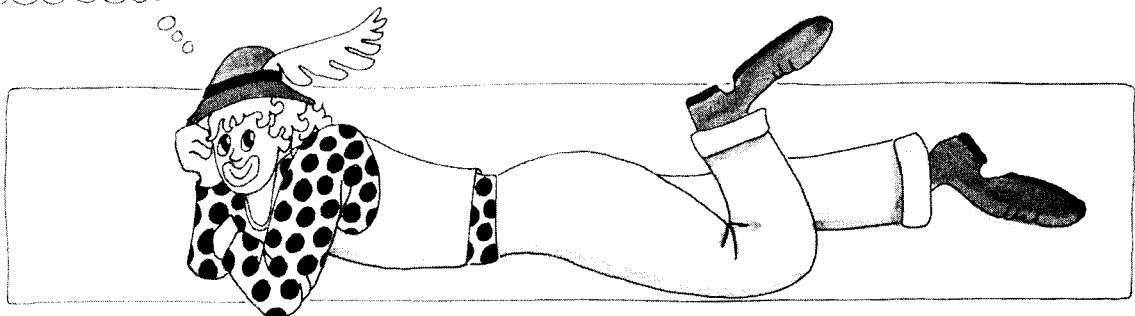
**if-then statement**

Is P\$ equal to "YES"?

If P\$ < > "YES" THEN

- |  |       |
|--|-------|
| 1. Is QR greater than 2?               | _____ |
| 2. Is Z\$ not equal to "END"?          | _____ |
| 3. Is F less than or equal to P?       | _____ |
| 4. Is G\$ equal to "JEEPERS"?          | _____ |
| 5. Is S1 greater than or equal to S2?  | _____ |
| 6. Is DD less than 444?                | _____ |
| 7. Is X greater than Y?                | _____ |
| 8. Is A\$ greater than or equal to 79? | _____ |
| 9. Is P\$ not equal to "YES"?          | _____ |
| 10. Is VP less than or equal to JK?    | _____ |

= > < > = < =



# PROGRAMMER'S PASTIME #44

The location of the IF-THEN statement in a program is very important. If it is put in the wrong place, the program won't work properly. The IF-THEN statement must come **after** the LET or INPUT statements that introduce the variables in the IF-THEN statement. For example:

## Program

```
10 IF P < Q THEN 50
20 LET P = 5
30 LET Q = 7
40 GOTO 60
50 ? "P IS SMALLER"
60 END
```

## Output

PET does not print anything because the IF-THEN statement is before the LET statements that introduce the variables.

In each of the following programs, the IF-THEN statement is in the wrong place. Rewrite the programs so they are correct.

## Incorrect program

```
1. 10 IF Z = 2 THEN 60
    20 LET A = 6
    30 LET B = 8
    40 LET Z = 2
    50 ? "Z = 2"
    60 END
```

## Corrected program

```
2. 10 FOR Y = 1 TO 6
    20 IF Q$ = "STOP"
      THEN 60
    30 LET Q$ = "GO"
    40 ? Q$
    50 NEXT Y
    60 END
```



---

**Incorrect program**

```
3. 10 ? "WHAT COLOR
    ARE YOUR
    EYES?"
    20 IF E$ = "BLUE"
        THEN 100
    30 INPUT E$
    40 ? "ARE THEY
        DIFFERENT
        COLORS?"
    50 IF D$ = "YES"
        THEN 120
    60 INPUT D$
    70 ? "THEY ARE 1
        COLOR"
    80 ? "THEY ARE NOT
        BLUE"
    90 GOTO 130
    100 ? "WHAT NICE
        BLUE EYES!"
    110 GOTO 130
    120 ? "WHAT
        COLORFUL
        EYES!"
    130 END
```

**Corrected program**

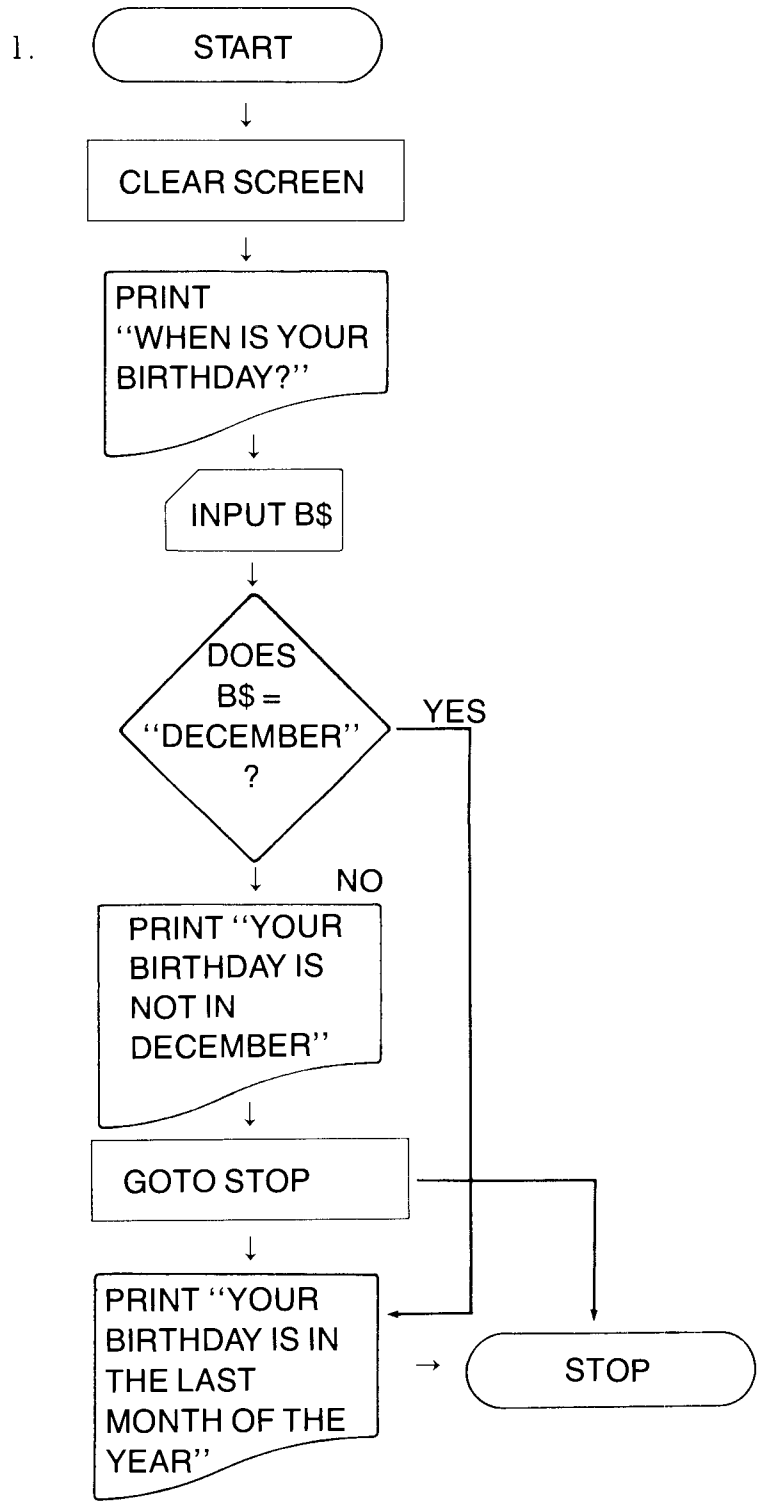
```
4. 10 IF F$ = "NO" THEN
    60
    20 ? "ARE
        COMPUTERS
        FUN?"
    30 INPUT F$
    40 ? "YOU'RE
        RIGHT!"
    50 GOTO 70
    60 ? "YOU'RE NO
        FUN!"
    70 END
```



# PROGRAMMER'S PASTIME #45

Study each flow chart, then write a program.  
Debug your programs by running them on PET.

## Flow chart



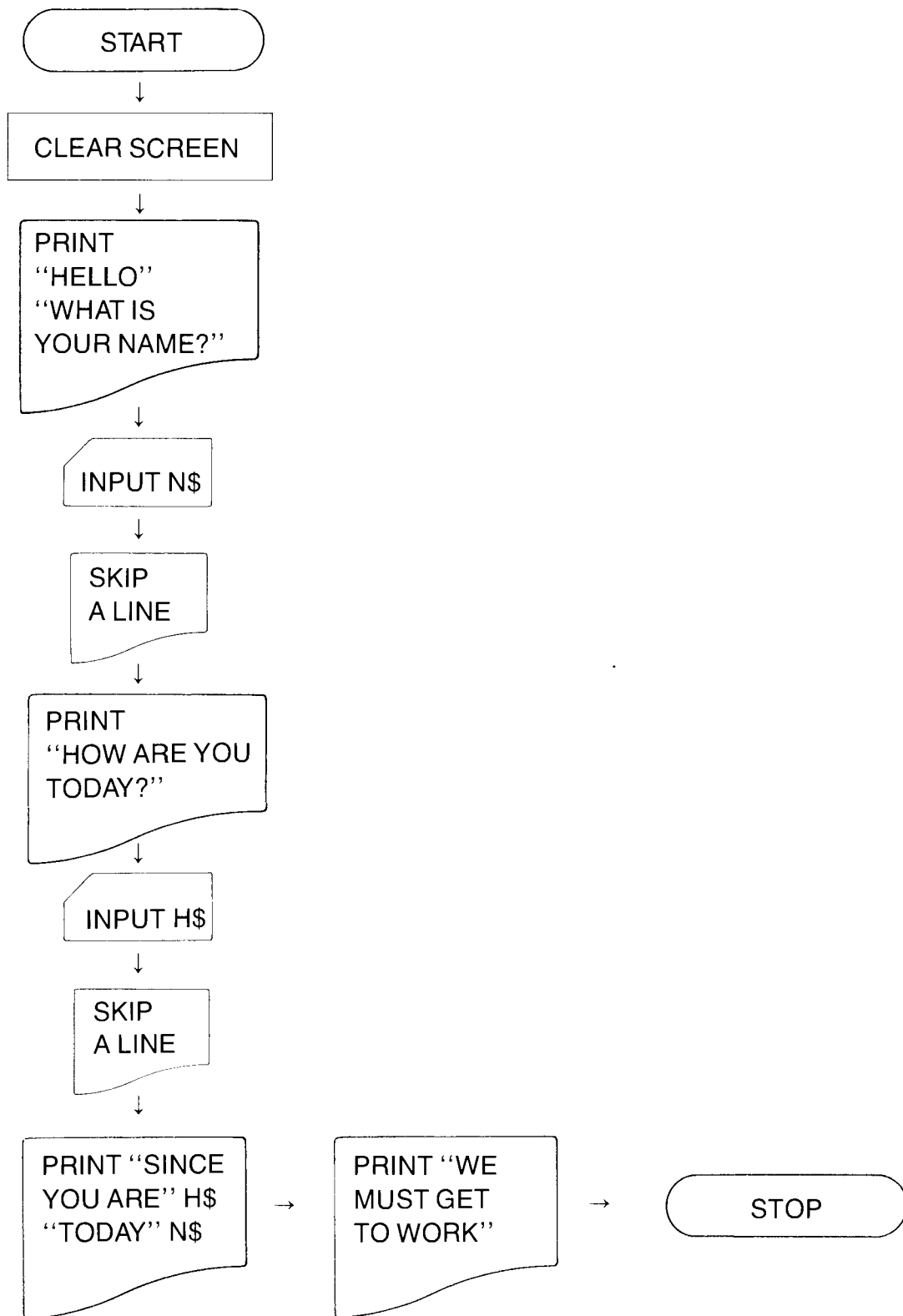
## Program



**Flow chart**

**Program**

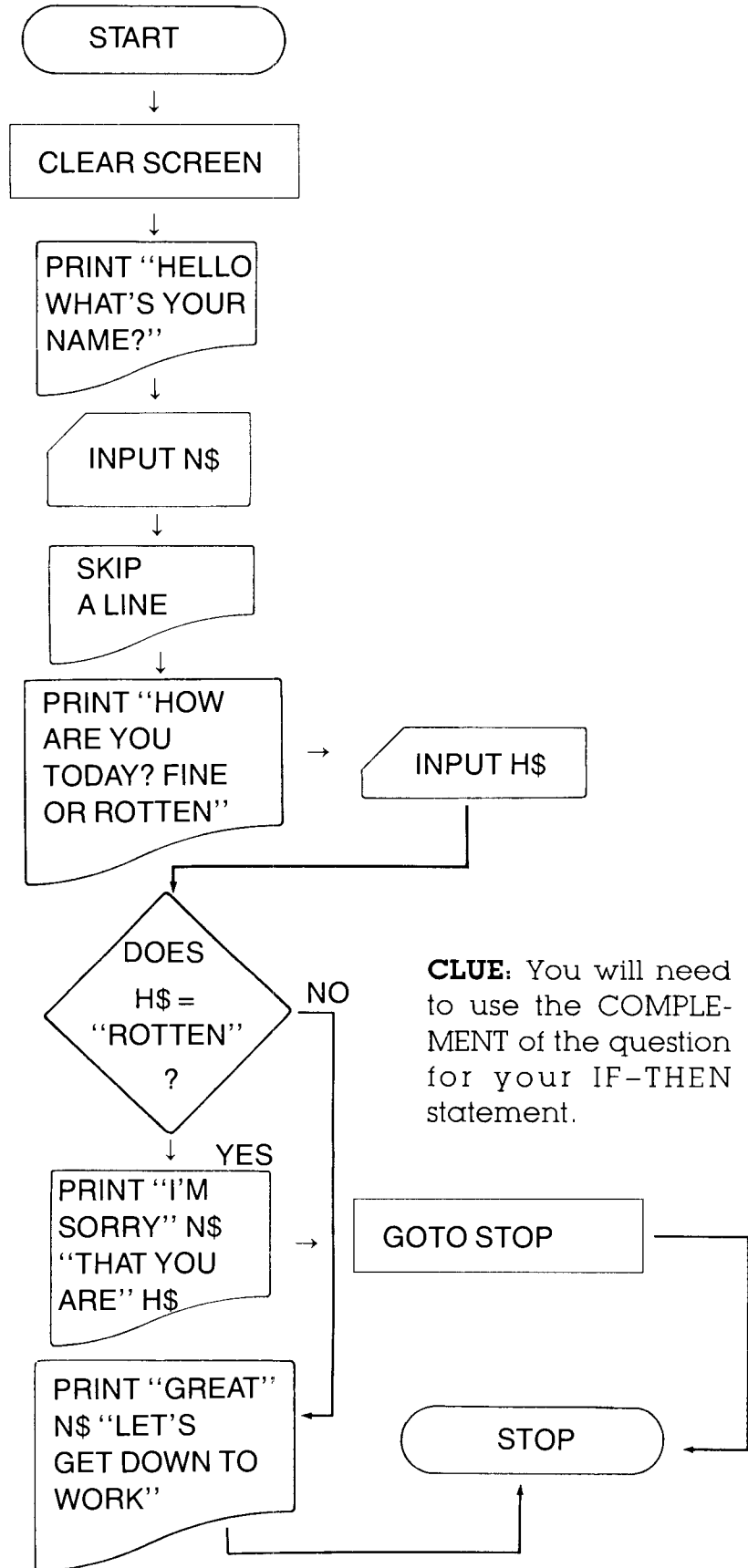
2.



**Flow chart**

**Program**

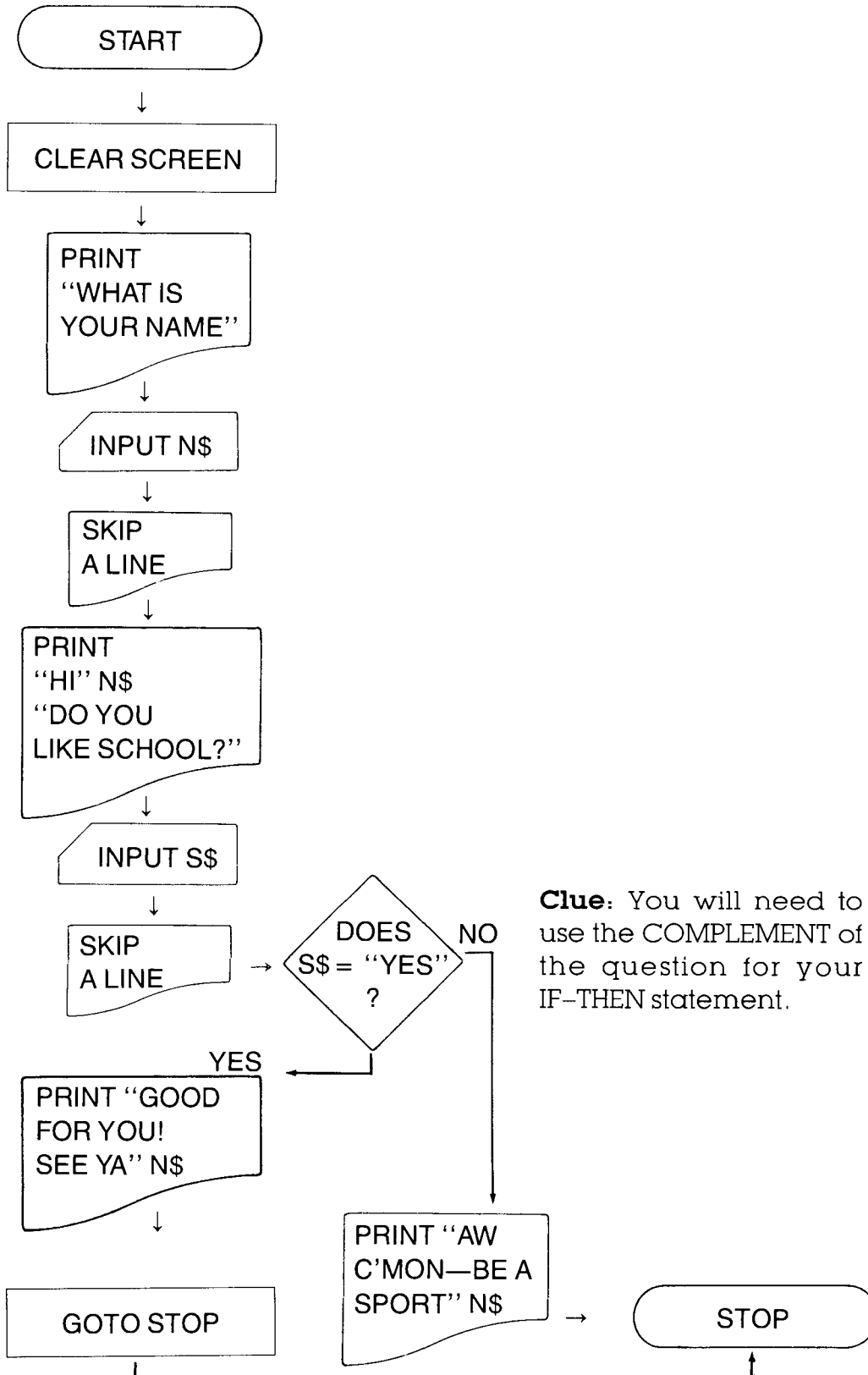
3.



Flow chart

Program

4.



**Clue:** You will need to use the COMPLEMENT of the question for your IF-THEN statement.

# PROGRAMMER'S PASTIME #46

For each description, write an algorithm in flow chart form and write a program for the flow chart. Debug each program by running it on PET.

## Description

## Flow chart

## Program

1. Alphabetize  
"HIP" and "HIPPO"

2. Alphabetize  
"GUSTO" and  
"GROOVEY"

3. Alphabetize  
"AARDVARK" and  
"ZEBRA"



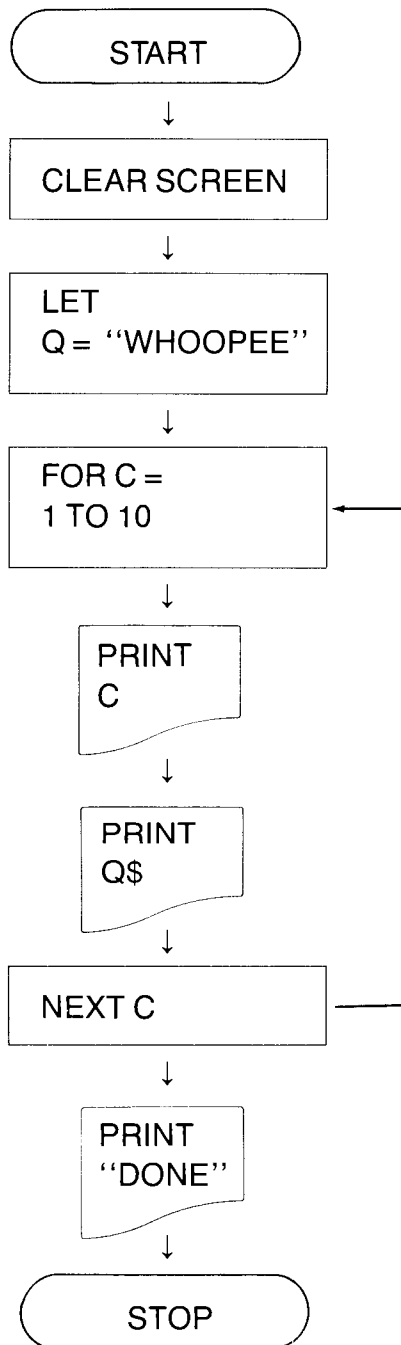
# PROGRAMMER'S PASTIME #47

Study each flow chart and then write a program. Use REM statements where appropriate and indent to show good programming style. Debug your programs by running them on PET.

## Flow chart

## Program

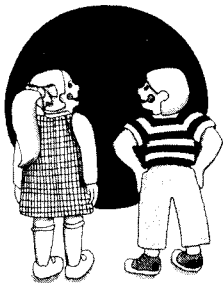
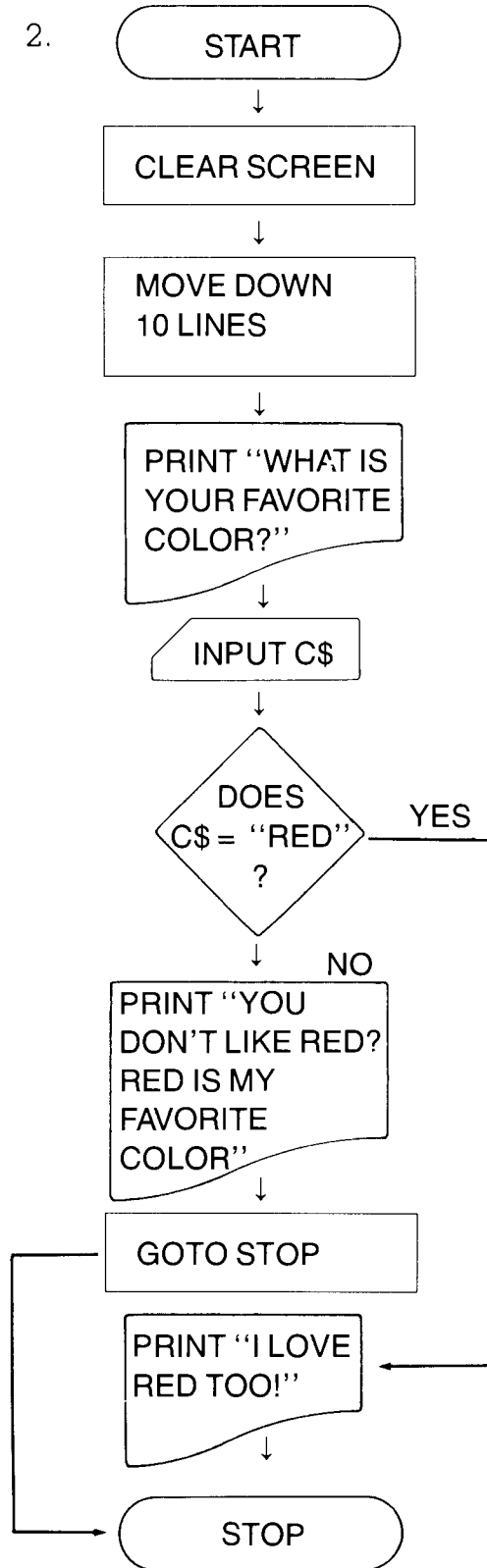
1.



Flow chart

Program

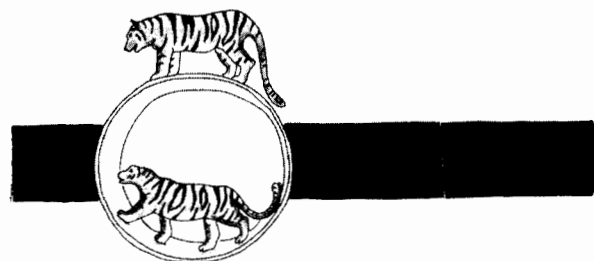
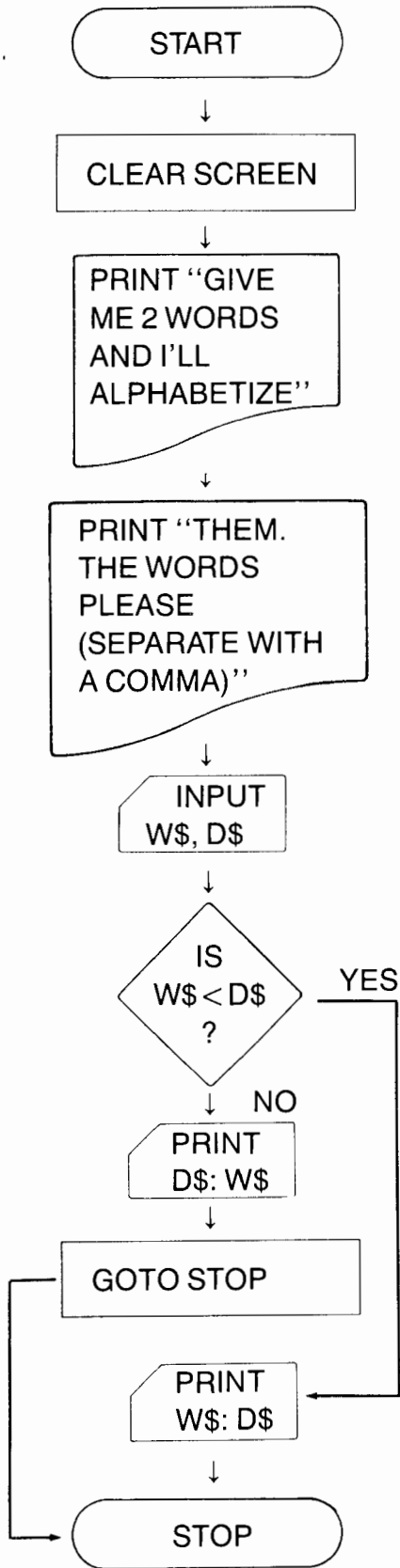
2.



Flow chart

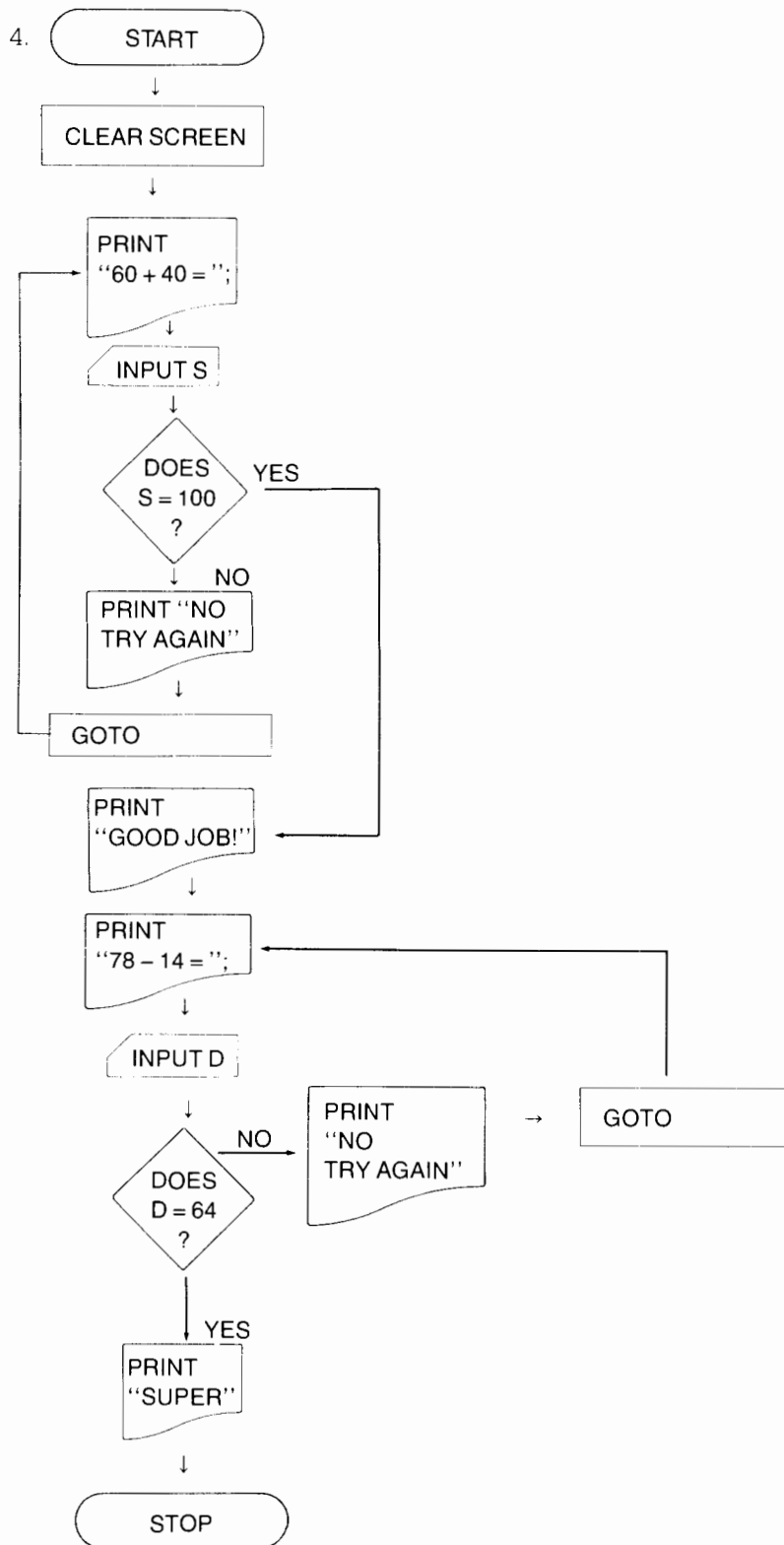
Program

3.



Flow chart

Program





# PROGRAMMER'S PASTIME #48

Study each program and write what you think PET would print as the OUTPUT—including error messages. Check your answers by running the programs on PET.

## Program

## Output

1. 10 READ Z\$, L\$  
20 ?Z\$, L\$  
30 GOTO 10  
40 DATA "YOU",  
"ARE", "A",  
"HOT-SHOT"  
50 END
  
2. 10 DATA 4,14,41,6,  
16,61,3,13,31  
20 READ Q, R, S  
30 ?Q+R+S  
40 GOTO 20  
50 END
  
3. 10 READ G, H  
20 DATA 44,66,88,  
22,110  
30 ?G, H  
40 GOTO 10  
50 END
  
4. 10 DATA 14,7,2,16,8,  
2, -99, -99, -99  
20 READ A, L, B  
30 IF A = -99 THEN  
60  
40 ? A - L - B  
50 GOTO 20  
60 END





### Program

```
5. 10 READ R1, R2
    20 IF R1 = - 1 THEN
        60
    30 ?R$*R2
    40 GOTO 10
    50 DATA 2,2,3,3,4,
        4,5,5, - 1, - 1
    60 END

6. 10 FOR L = 1 TO 4
    20 READ D$, E$
    30 ?D$, E$
    40 NEXT L
    50 DATA "A", "E",
        "I", "O"
    60 DATA "U", "Y",
        "ARE",
        "VOWELS"
    70 END

7. 10 FOR L = 1 TO 2
    20 READ S1, S2, S3
    30 DATA 8,2,4,6,
        2,3
    40 ?S1*S2*S3
    50 NEXT L
    60 END
```

### Output

# PROGRAMMER'S PASTIME #49

In each of the following programs there are mistakes. Circle the line number(s) with the mistake and make your correction in the space to the right. If something has been left out, add it to the program.

## Program

## Correction

1. 10 READ P,A,N  
20 ?P,A,N  
30 DATA 400, 8%, 6  
40 END
2. 10 READ N\$, A  
20 ?N\$, A  
30 DATA "KIM IS",  
3\*4  
40 END
3. 10 ? "NAME",  
"AGE"  
20 READ A, N\$  
30 ? A, N\$  
40 DATA "HARVEY",  
14  
50 END
4. 10 ? "NAME", "AGE"  
20 READ N\$, A  
30 ?N\$, A  
40 DATA "HARVEY",  
14 YEARS OLD  
50 END



---

**Program****Correction**

5. 10 READ F\$  
20 ? "DAILY MENU"  
30 ? F\$  
40 END

6. 10 READ X,Y,Z  
20 ? "THE PRODUCT  
OF 3 NUMBERS"  
30 ? X\*Y\*Z  
40 DATA 4,5  
50 END

7. 10 ? "COUNTING"  
20 READ DATA  
30 DATA 1,2,3,4  
40 END

8. 10 ? "NAME", "AGE"  
20 READ N\$, A  
30 ? N\$, A  
40 GOTO 20  
50 DATA "BOB",  
"BILL",10,11



# PROGRAMMER'S PASTIME #50

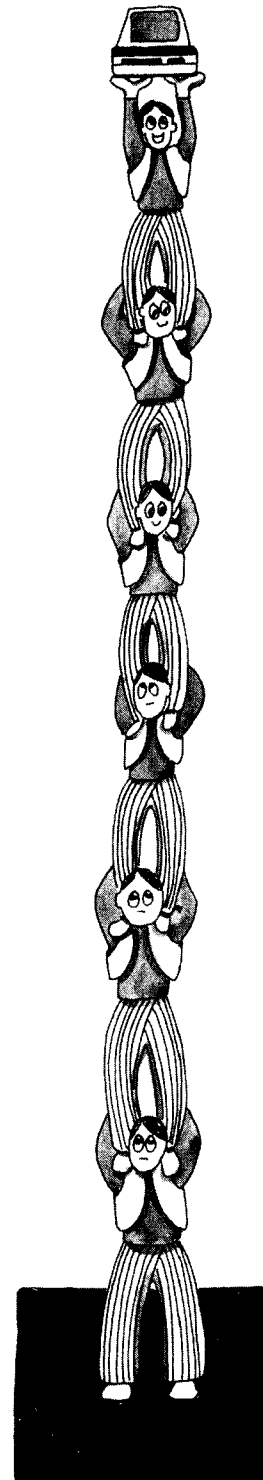
READ-DATA statements can help you write shorter programs. Rewrite each program using READ-DATA statements to shorten it. Try to write each program so you don't get an error message.

## Long program

```
1. 10 ? "MULTIPLYING
    2 NUMBERS"
20 LET P = 60
30 LET Q = 129
40 LET R = 410
50 LET S = .6
60 ? P,Q, P*Q
70 ? R,S, R*S
80 END

2. 10 ? "TEST SCORES"
20 ? "NAME",
    "SCORE"
30 LET A$ = "JOE"
40 LET A = 98
50 LET B$ = "TOM"
60 LET B = 52
70 LET C$ = "KRIS"
80 LET C = 95
90 LET D$ = "GAIL"
100 LET D = 75
110 LET E$ = "BOB"
120 LET E = 72
130 ? A$, A
140 ? B$, B
150 ? C$, C
160 ? D$, D
170 ? E$, E
180 END
```

## Short program

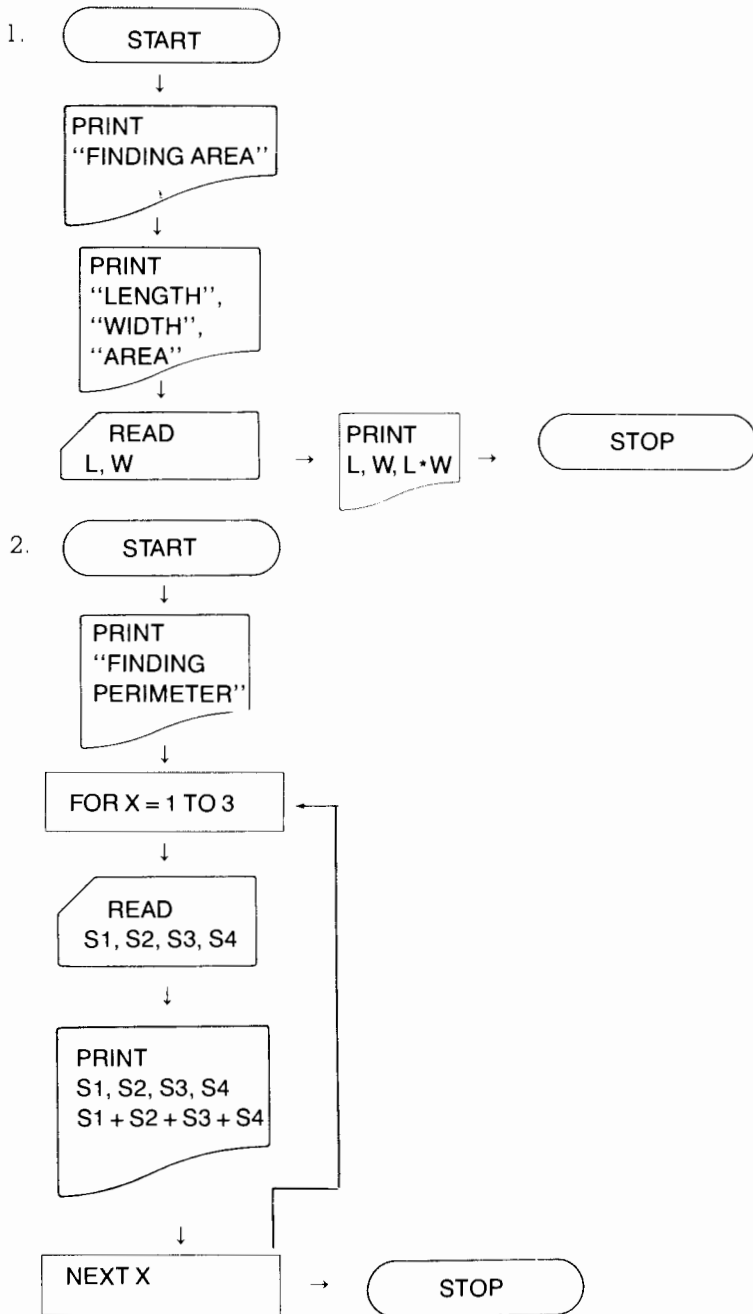


# PROGRAMMER'S PASTIME #51

Study each flow chart. Then write a program using a READ-DATA statement. Use any DATA that you feel will work in the DATA statement. Debug your programs by running them on PET.

## Flow chart

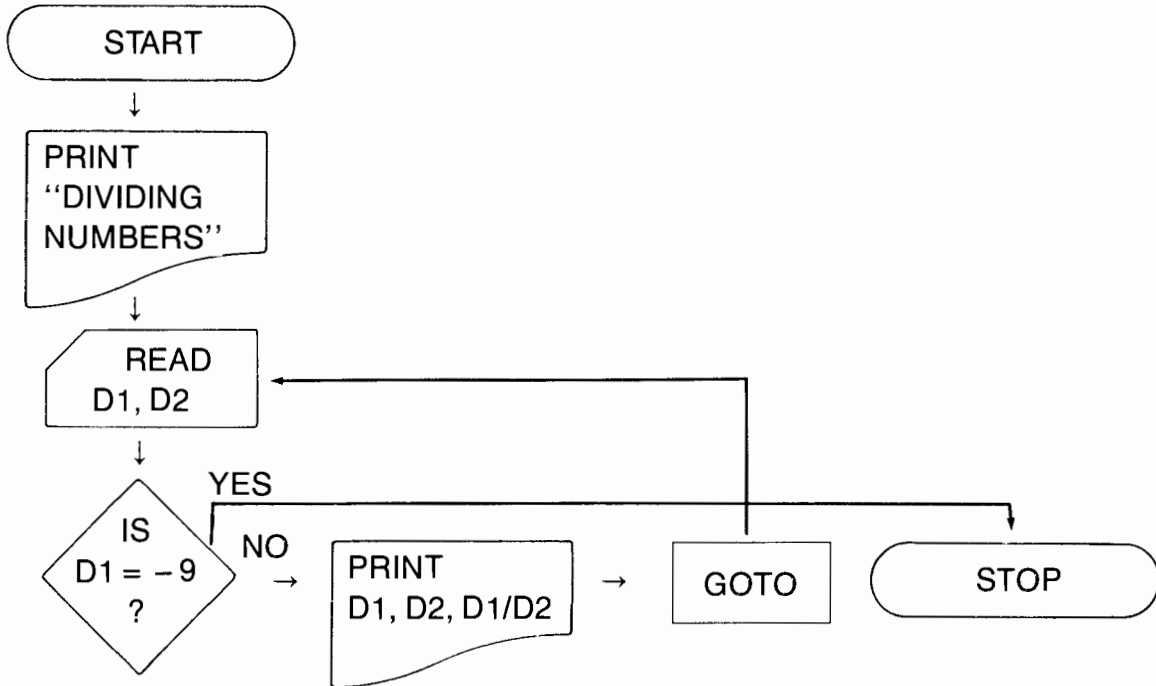
## Program



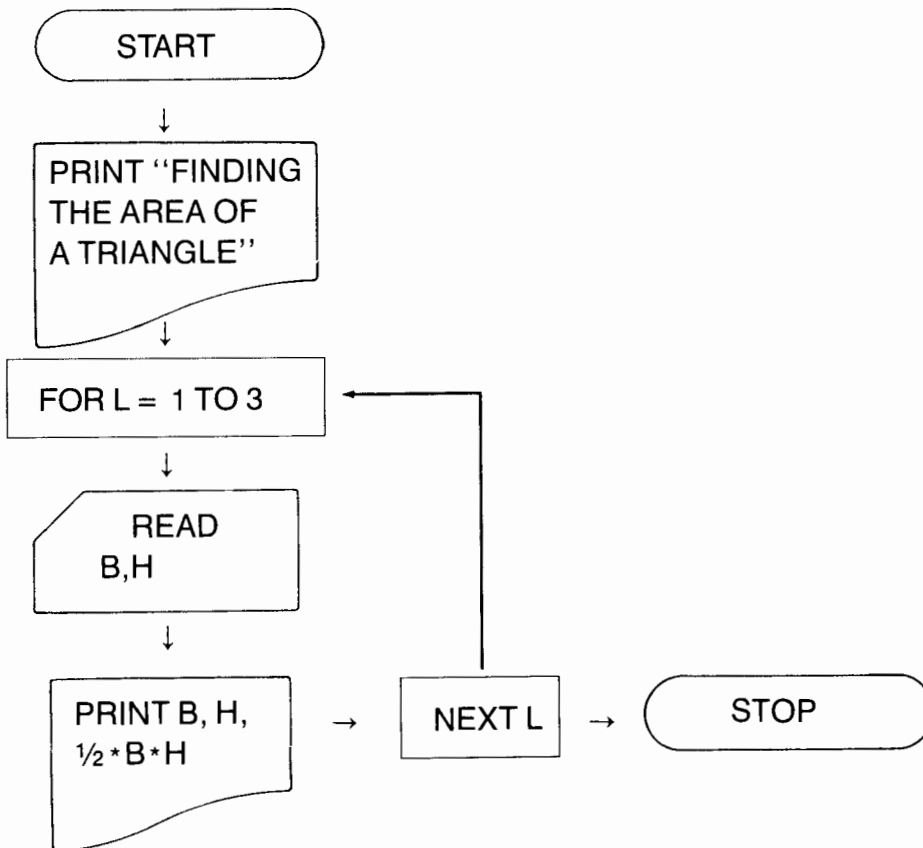
**Flow chart**

**Program**

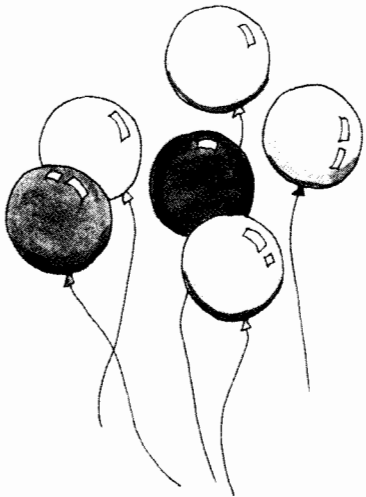
3.



4.



# PROGRAMMER'S PASTIME #52



Using what you know about READ-DATA statements:

1. Write a program that multiplies three numbers.

**Flow chart**

**Program**

2. Write a program that lists the names of your friends.

**Flow chart**

**Program**



# PROGRAMMER'S PASTIME#53

Use the problem-solving approach to get PET to solve the following problems.

## Problem 1

The teacher gave your class a test on programming the computer. The test scores were:

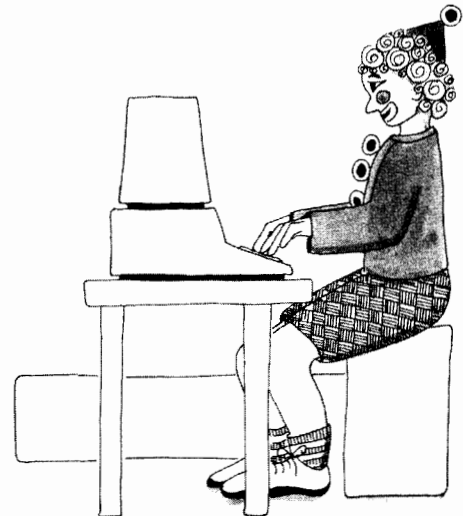
Jill Jarvis	73%	Your teacher needs to know the <b>average</b> test score.
Katie O'Keefe	98%	
Tommy Templeton	67%	
Susie Sunbeam	82%	
You	90%	

Write a program that tells PET to calculate and print the average score.

HINT: To find the average of 5 numbers, add them together and divide by 5.

1. THINK about the problem.
2. Make your DATA TABLE here.

3. Write the ALGORITHM (steps and equations).



---

4. FLOW CHART

5. CODE the program

6. DEBUG

7. REVISE

---

### Problem 2

You are the new manager of the PEPPY PIZZA Restaurant and you need the help of a computer. Write a program that will allow you to INPUT the number of small, medium, and large pizzas sold during a day.

Have PET print out the total number of pizzas sold and how much money you made.

PRICES:	small	\$4.30
	medium	\$5.50
	large	\$7.25

OUTPUT HINT:

HOW MANY PIZZAS: (SMALL, MEDIUM, LARGE)

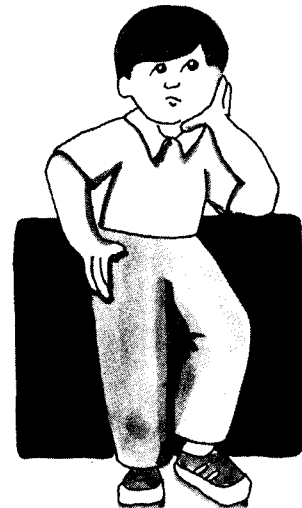
? \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_

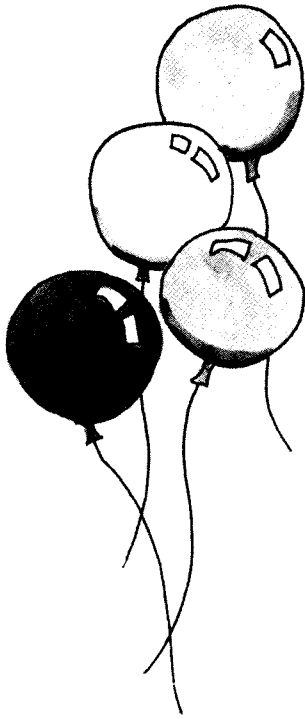
THERE WERE \_\_\_\_\_ PIZZAS SOLD TODAY.

PEPPY PIZZA MADE \$\_\_\_\_\_.

1. THINK about the problem.
2. DATA TABLE

3. ALGORITHM





4. FLOW CHART

5. CODE the program

6. DEBUG  
7. REVISE

---

### **Problem 3**

Write a program that will allow you to INPUT your age in years, months, and days. Example: 9 years, 3 months, 17 days.

Have PET calculate and print how many days, hours, and minutes old you are.

HINT: There are normally 365 days in a year and 30 days in a month. There are exactly 24 hours in a day and 60 minutes in an hour.

1. THINK about the problem.
2. DATA TABLE

3. ALGORITHM

---

4. FLOW CHART

5. CODE the program

6. DEBUG

7. REVISE

---

#### Problem 4

You just got hired as a SUPER SCOOPER at the DIPPER DELIGHT Ice Cream Store. Write a program that will allow you to INPUT how many hours you worked for the week.

Have PET calculate and print hours worked and your salary for the week if you make \$3.25 an hour.

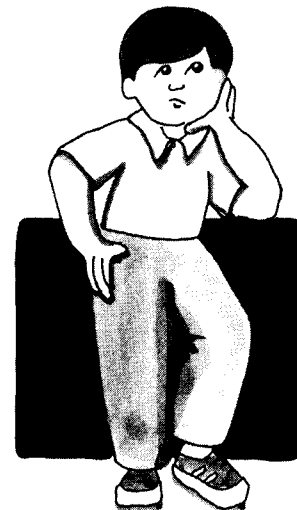
OUTPUT HINT:

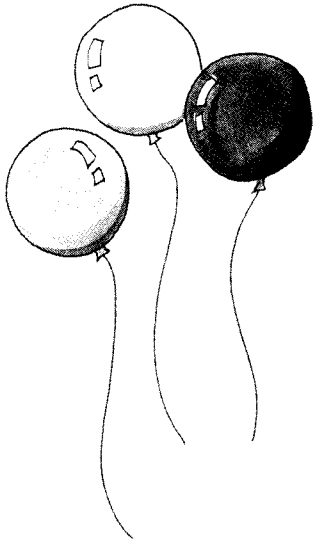
HOW MANY HOURS DID YOU WORK? \_\_\_\_\_

YOU WORKED \_\_\_\_\_ HOURS AND MADE  
\$\_\_\_\_\_.

1. THINK about the problem.
2. DATA TABLE

3. ALGORITHM





4. FLOW CHART

5. CODE the program

6. DEBUG  
7. REVISE



---

### Problem 5

Add to the problem you wrote for PROBLEM 4 so that PET can calculate overtime pay. (Overtime is any hours worked **over** 40 hours a week.) You get paid \$4.75 for every hour of overtime you work.

Add this to your OUTPUT:

YOU WORKED \_\_\_\_\_ OVERTIME HOURS AND  
MADE \$\_\_\_\_\_ IN OVERTIME.  
YOUR TOTAL PAY FOR THE WEEK IS \$\_\_\_\_\_.  
(Total pay is regular pay + overtime pay.)

HINT: You will need a decision box in your flow chart to ask:

IS  $H > 40$ ?

1. THINK about the problem.
2. DATA TABLE

3. ALGORITHM

---

4. FLOW CHART

5. CODE the program

6. DEBUG  
7. REVISE

---

**Problem 6**

You are the famous sportscaster H.E. Nosell. You have been asked to calculate the batting averages of Big League Baseball players. Write a program that allows you to INPUT a player's name, hits, and times at bat.

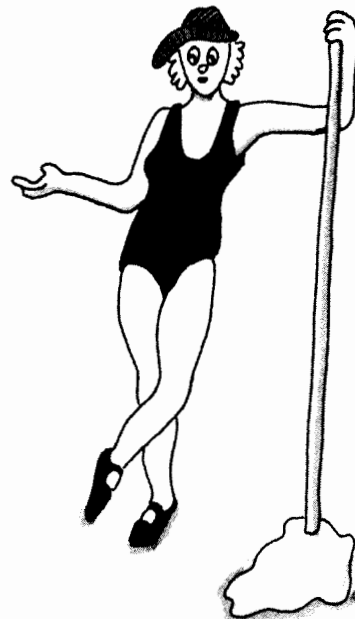
Have PET calculate and print the player's name and batting average.

HINT: To calculate batting average, use this equation:

$$1000 * \text{hits} / \text{times at bat}$$

1. THINK about the problem.
2. DATA TABLE

3. ALGORITHM



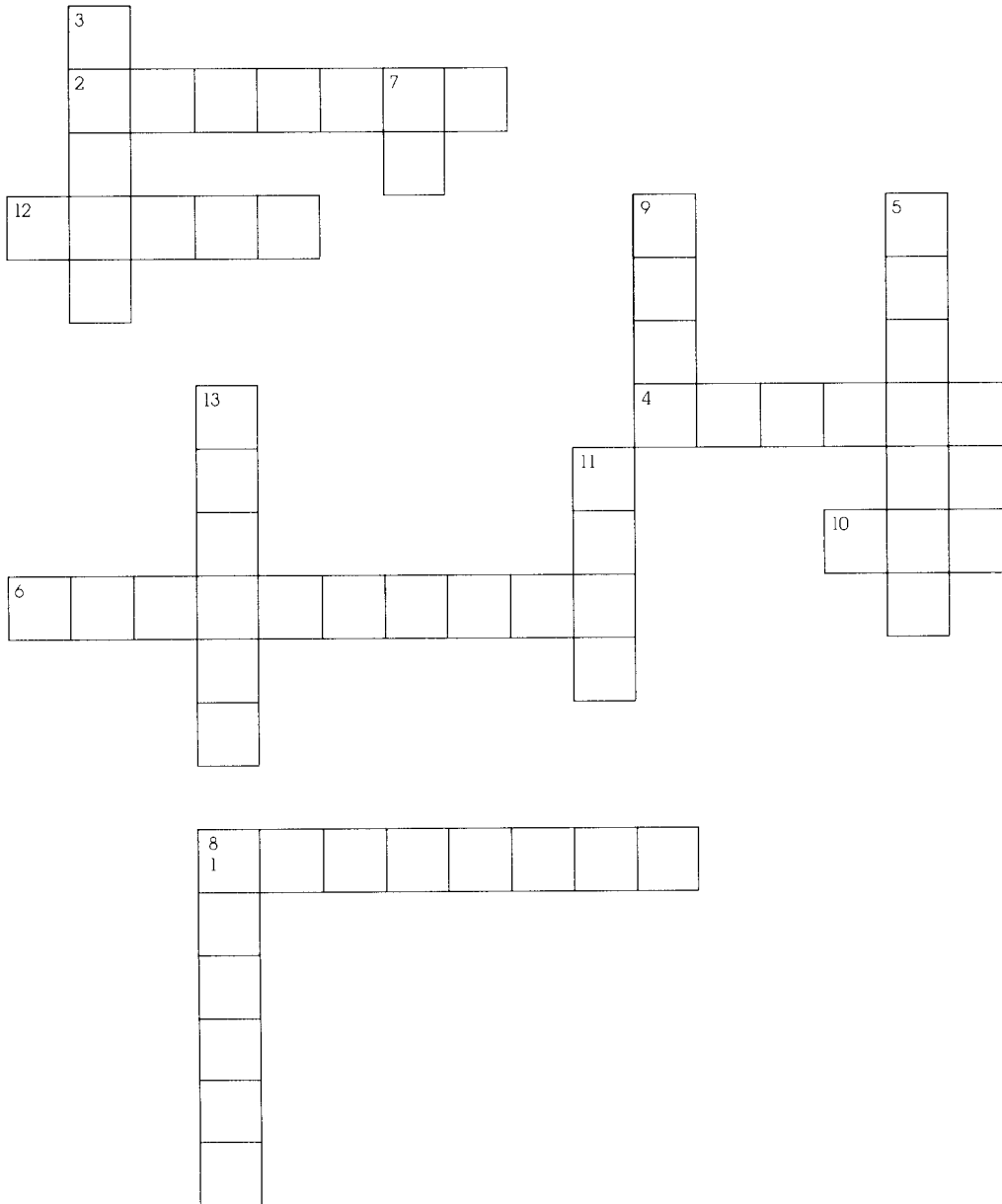
---

4. FLOW CHART

5. CODE the program

6. DEBUG  
7. REVISE

# COMPONENT 6 FUN PAGE



## Down

1. An alphanumeric variable is also called a \_\_\_\_\_ variable.
3. We can interact with the computer by using an \_\_\_\_\_ statement.
5. This sign,  $>$ , means \_\_\_\_\_ than.
7. The statement used for making comparisons is \_\_\_\_\_-THEN.
9. The statements that let you change your data are the \_\_\_\_\_-DATA statements.
11. A \_\_\_\_\_ table lists the variables you are using in a program.
13. \_\_\_\_\_ variables are the answers that PET will give you.

---

### ACROSS

2. A variable that stores a number. \_\_\_\_\_ in the alphabet.
4. A string variable is written as a letter followed by a \_\_\_\_\_ sign.
6. The opposite of a question is called its \_\_\_\_\_.
8. When PET is alphabetizing words, it knows that "A" is the \_\_\_\_\_ letter.
10. You should document your programs by using the \_\_\_\_\_ statement.
12. We use \_\_\_\_\_ data to let PET know that we are at the end of our data list.

### Evaluate Yourself

1. Component 6 was \_\_\_\_\_  
because \_\_\_\_\_
2. The best parts of the component were \_\_\_\_\_
3. The parts I liked the least were \_\_\_\_\_
4. The most valuable thing I learned in this component was \_\_\_\_\_  
because \_\_\_\_\_

Other comments:

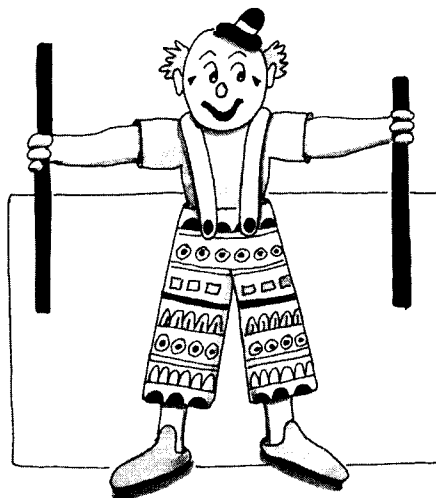


# PROGRAMMER'S PASTIME #54

For each conversion problem, identify the important parts by writing: HEADING, FOR-NEXT LOOP, CONVERSION EQUATION next to the lines in the program. Then write what you think PET would print as the output.

Program	Important parts	Output
<b>Example:</b>		
10 REM CONVERT FEET TO METERS		FEET            METERS
20 ? "FEET", "METERS"	HEADING	1                .3
30 ?		2                .6
40 FOR F = 1 TO 10	FOR-NEXT LOOP	3                .9
50 ? F, F * .3	CONVERSION	4                1.2
60 NEXT F	EQUATION	5                1.5
70 END		6                1.8
		7                2.1
		8                2.4
		9                2.7
		10               3

1. 10 REM CONVERT FEET  
TO YARDS  
20 ? "FEET",  
"YARDS"  
30 ?  
40 FOR F = 1 TO 12  
STEP 2  
50 ? F, F/3  
60 NEXT F  
70 END



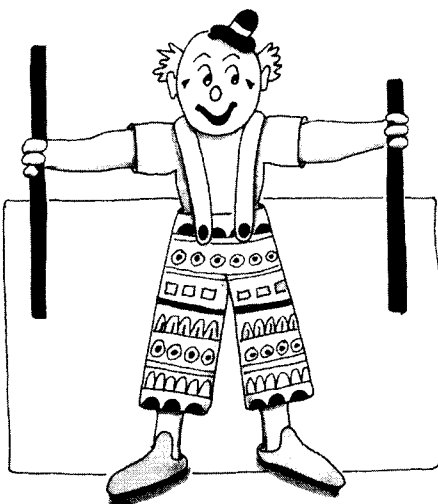
---

**Program****Important parts****Output**

```
2. 10 REM CONVERT
    TEASPOONS TO
    TABLESPOONS
    20 ? ``TEASPOONS``,
        ``TABLESPOONS``
    30 FOR T = 3 TO 18
        STEP 3
    40 ? T, T/3
    50 NEXT T
    60 END

3. 10 REM CONVERT
    POUNDS TO OUNCES
    20 ? ``POUNDS``,
        ``OUNCES``
    30 FOR P = 1 TO 6
    40 ? P, P*16
    50 NEXT P
    60 END

4. 10 REM CONVERT
    YARDS TO INCHES
    20 ? ``YARDS``,
        ``INCHES``
    30 FOR Y = 1 TO 5
    40 ? Y, Y*36
    50 NEXT Y
    60 END
```





# PROGRAMMER'S PASTIME #55

Write a conversion program for each problem. Make sure your program has a heading, FOR-NEXT loop, and conversion equation. Run your programs on PET to check for bugs.

## Problem

## Program

1. Convert 1-20 inches  
to centimeters.  
CONVERSION  
EQUATION:  
Centimeters = I\*2.5

2. Convert 1-20  
kilometers to miles.  
CONVERSION  
EQUATION:  
Miles = K/1.6

3. Convert 1-20  
pounds to grams.  
CONVERSION  
EQUATION:  
Grams = P\*454

---

**Problem**

4. Convert 1-10 liters to  
quarts.

CONVERSION

EQUATION:

$$\text{QUARTS} = L / 3.8$$

**Program**

5. Convert 0°-100°  
Fahrenheit to  
Celsius.

CONVERSION

EQUATION:

$$^{\circ}\text{C} = 5 * (\text{F} - 32) / 9$$

6. Convert 1-100  
pounds to  
kilograms.

CONVERSION

EQUATION:

$$\text{Kilograms} = P * .45$$



# PROGRAMMER'S PASTIME #56

Use the problem-solving approach to get PET to solve the following conversion problems.

1. Jed needs to find out what decimal  $\frac{6}{7}$  stands for. Write a program that lists the fractions  $\frac{1}{7}$  through  $\frac{7}{7}$  and the decimals they stand for. CONVERSION EQUATION:  $\text{Decimal} = X/7$

1. THINK about the problem.
2. DATA TABLE

3. ALGORITHM

---

4. FLOW CHART

5. CODE the program

6. DEBUG

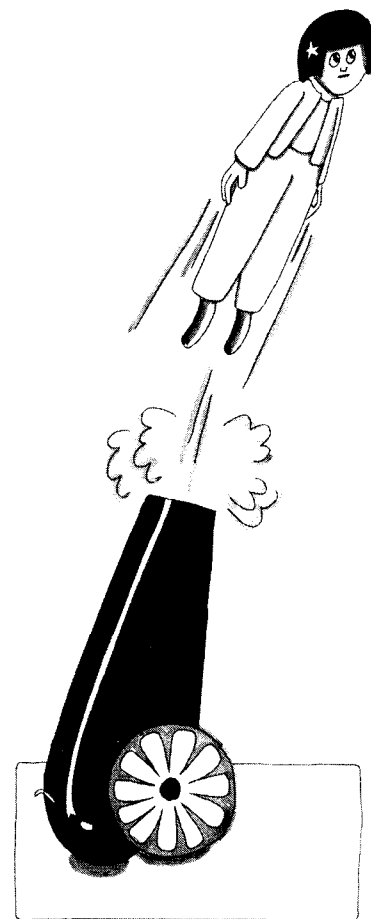
7. REVISE

---

2. Amy Astronaut is going to the moon. She learned that because the gravity on the moon is only  $\frac{1}{6}$  of the earth's gravity, she will weigh less on the moon. Write a program that asks you to INPUT how much you weigh. Then have PET print how much you would weigh on the moon. CONVERSION EQUATION: moon weight = earth weight / 6

1. THINK about the problem
2. DATA TABLE

3. ALGORITHM



---

4. FLOW CHART

5. CODE the program

6. DEBUG  
7. REVISE

---

3. Add to program #2 so PET will print a conversion table of weight on earth from 10 pounds to 100 pounds and the equal moon weights **after** printing the output in program #2.

1. FLOW CHART

2. CODE

3. DEBUG  
4. REVISE

---

## CHALLENGE

4. Fred's class took a test in which there were 20 questions asked. Fred's score was 16 correct out of 20, or  $\frac{16}{20}$ . Fred wants to know what percentage this would be. Write a program that lists the percentages for the test scores  $\frac{1}{20}$  through  $\frac{20}{20}$ .

$\underline{X}$  = number answered correctly  
20 = total number of questions

$\underline{P}$  = percentage  
100 = total

CONVERSION EQUATION:  $P = X * 100 / 20$

1. THINK about the problem
2. DATA TABLE

3. ALGORITHM



---

4. FLOW CHART

5. CODE the program

6. DEBUG  
7. REVISE



---

## CHALLENGE

5. Change program #4 so PET asks you to INPUT how many test questions (T), and how many questions you answered correctly (C). Have PET print your score and the percentage you got correct.

HINT: score = C out of T  
percentage =  $C * 100 / T$

1. FLOW CHART
2. CODE

3. DEBUG
4. REVISE

# PROGRAMMER'S PASTIME #57

RUN the program five times on PET. Each time the program is run, write down the random numbers that PET printed. Then write the lowest and highest numbers in the list.

RUN #1

numbers					
lowest					
highest					

**Program**  
 10 FOR L = 1 TO 5  
 20     LET X = 10\*RND(1)  
 30     ? X  
 40 NEXT L  
 50 END

RUN #2

numbers					
lowest					
highest					

RUN #3

numbers					
lowest					
highest					

RUN #4

numbers					
lowest					
highest					

RUN #5

numbers					
lowest					
highest					

# PROGRAMMER'S PASTIME #58

Read each RND function. Figure out what the lowest and highest random numbers will be that PET could print.

**Function**

**Pet will print  
random numbers  
between:**

**Example:**

LET X = 18 \* RND(1)

0 and 18

1. LET X = 300 \* RND(1)

\_\_\_\_\_ and \_\_\_\_\_

2. LET X = RND(1)

\_\_\_\_\_ and \_\_\_\_\_

3. LET X = 3 \* RND(1)

\_\_\_\_\_ and \_\_\_\_\_

4. LET X = 67 \* RND(1)

\_\_\_\_\_ and \_\_\_\_\_

5. LET X = 100 \* RND(1)

+ 1

\_\_\_\_\_ and \_\_\_\_\_

6. LET X = 25 \* RND(1) + 1

\_\_\_\_\_ and \_\_\_\_\_

7. LET X = 116 \* RND(1)

+ 1

\_\_\_\_\_ and \_\_\_\_\_

8. LET X = 39 \* RND(1) + 1

\_\_\_\_\_ and \_\_\_\_\_

9. LET X = 436 \* RND(1)

\_\_\_\_\_ and \_\_\_\_\_

10. LET X = 77 \* RND(1) + 1

\_\_\_\_\_ and \_\_\_\_\_

11. LET X = 43 \* RND(1) + 1

\_\_\_\_\_ and \_\_\_\_\_

12. LET X = 13 \* RND(1)

\_\_\_\_\_ and \_\_\_\_\_

13. LET X = 59 \* RND(1) + 1

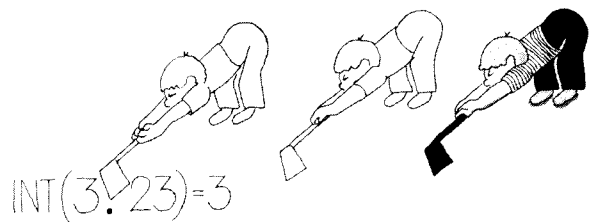
\_\_\_\_\_ and \_\_\_\_\_



# PROGRAMMER'S PASTIME #59

INTEGERS are whole numbers. The INT function TRUNCATES a decimal to make it an integer. Read each INT function. Then write what PET could print for the output.

Function	Output
<b>Example:</b> 10 LET C = 4.96 20 ?INT(C)	4
1. 10 LET X = 66.823 20 ?INT(X)	_____
2. ?INT(4.89)	_____
3. 10 LET R = 992.01 20 ?INT(R)	_____
4. ?INT(63.49321)	_____
5. 10 LET BD = -16.003 20 ?INT(BD)	_____
6. 10 LET P1 = 43.001 20 ?INT(P1)	_____
7. ?INT(660.666)	_____
8. ?INT(-33.23)	_____
9. 10 LET S = 4120.7 20 ?INT(S)	_____
10. ?INT(-999.999)	_____



# PROGRAMMER'S PASTIME #60

We use both the INT and RND functions to tell PET to print a random integer. Read each function. Then write the two numbers that PET must create random integers **between**. Remember the equation:

$$\text{INT}((B - A + 1) * \text{RND}(1) + A)$$

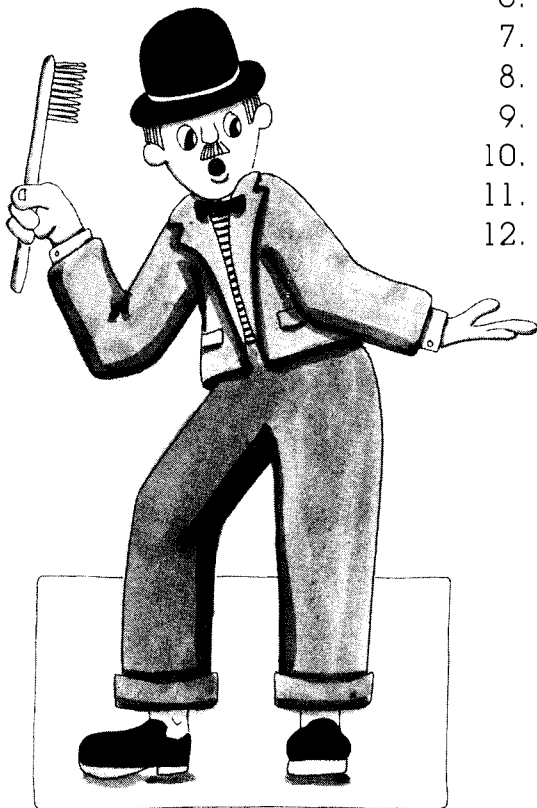
**Function** **PET will print random integers between:**

**Example:**

$\text{INT}(40 * \text{RND}(1) + 26)$  26 and 65

DO:  $40 + 26 - 1 = 65$

- |   |       |     |       |
|---|-------|-----|-------|
| 1. $\text{INT}(14 * \text{RND}(1) + 3)$   | _____ | and | _____ |
| 2. $\text{INT}(221 * \text{RND}(1) + 99)$ | _____ | and | _____ |
| 3. $\text{INT}(3 * \text{RND}(1) + 2)$    | _____ | and | _____ |
| 4. $\text{INT}(22 * \text{RND}(1) + 16)$  | _____ | and | _____ |
| 5. $\text{INT}(55 * \text{RND}(1) + 28)$  | _____ | and | _____ |
| 6. $\text{INT}(77 * \text{RND}(1) + 75)$  | _____ | and | _____ |
| 7. $\text{INT}(94 * \text{RND}(1) + 33)$  | _____ | and | _____ |
| 8. $\text{INT}(101 * \text{RND}(1) + 66)$ | _____ | and | _____ |
| 9. $\text{INT}(63 * \text{RND}(1) + 7)$   | _____ | and | _____ |
| 10. $\text{INT}(80 * \text{RND}(1) + 45)$ | _____ | and | _____ |
| 11. $\text{INT}(46 * \text{RND}(1) + 23)$ | _____ | and | _____ |
| 12. $\text{INT}(39 * \text{RND}(1) + 19)$ | _____ | and | _____ |



# PROGRAMMER'S PASTIME #61

Write an RND function for each description.

**Create random numbers between and including:**

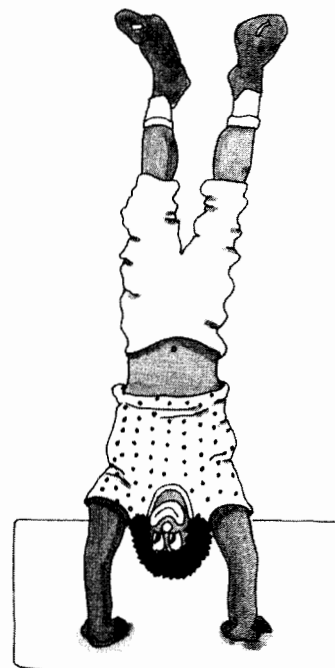
**Example:**

.0001 and 9.9999

**Function**

LET X = 10\*RND(1)

- 1. .0001 and 14.9999 \_\_\_\_\_
- 2. .0001 and 92.9999 \_\_\_\_\_
- 3. 1.0001 and 45.9999 \_\_\_\_\_
- 4. .0001 and 70.9999 \_\_\_\_\_
- 5. 1.0001 and 26.9999 \_\_\_\_\_
- 6. .0001 and 106.9999 \_\_\_\_\_
- 7. .0001 and 66.9999 \_\_\_\_\_
- 8. 1.0001 and 211.9999 \_\_\_\_\_
- 9. 1.0001 and 31.9999 \_\_\_\_\_
- 10. 1.0001 and 441.9999 \_\_\_\_\_
- 11. .0001 and 89.9999 \_\_\_\_\_
- 12. 1.0001 and 53.9999 \_\_\_\_\_
- 13. 1.0001 and 382.9999 \_\_\_\_\_
- 14. .0001 and 554.9999 \_\_\_\_\_



# PROGRAMMER'S PASTIME #62

Write an INT and RND function for each description. Remember the equation:

$$\text{INT}((B - A + 1) * \text{RND}(1) + A)$$

B = largest number    A = smallest number

**To print random integers between**

**Example:** 5 and 18

DO:  $\text{INT}((18 - 5 + 1) * \text{RND}(1) + 5)$

**Function**

$\text{INT}(14 * \text{RND}(1) + 5)$

1. 16 and 48
2. 2 and 10
3. 10 and 100
4. 1 and 50
5. 33 and 99
6. 50 and 100
7. 75 and 100
8. 27 and 41
9. 62 and 300
10. 49 and 52

---

---

---

---

---

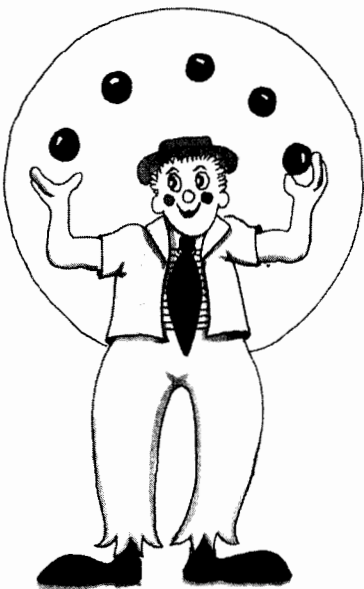
---

---

---

---

---





# **PROGRAMMER'S PASTIME #63**

Make a flow chart and write a program for each problem. Debug your programs by running them on PET.

1. Write a program that will print 10 random decimals between 1 and 100 and then print the integer for each.

**Flow Chart**

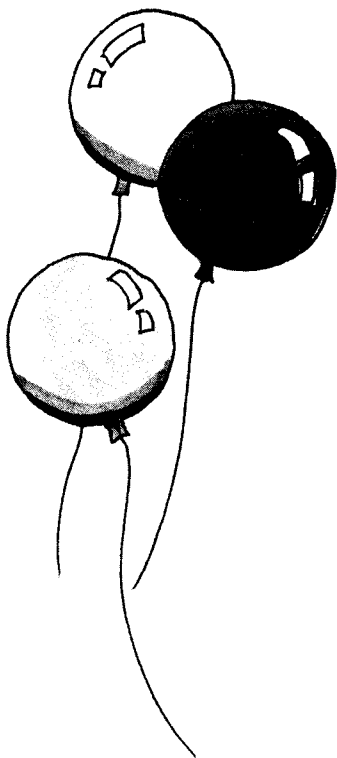
**Program**

---

2. Write a CAI program that asks a student to multiply two random numbers between 1 and 10.

**Flow chart**

**Program**



---

3. You can use the INPUT statement in a program with the RND and INT functions. Write a program so that PET will ask you to type in two integers. Then have PET print 10 random integers between those two numbers.

OUTPUT HINT: TYPE IN TWO NUMBERS  
AND I WILL CREATE TEN  
RANDOM INTEGERS BETWEEN  
THOSE TWO NUMBERS  
?  
10 RANDOM INTEGERS  
BETWEEN AND ARE:

**Flow chart**

**Program**

# PROGRAMMER'S PASTIME #64

Run the following programs on PET to see how the TAB function works. Write down PET's output and answer the questions.

Program	Output
---------	--------

1. 10 FOR X = 1 TO 10 20 ?TAB(5); ``π`` 30 NEXT X 40 END	
---	--

2. 10 FOR X = 1 TO 10 20 ?TAB(X); ``π`` 30 NEXT X 40 END	
---	--

What caused each new  $\pi$  to be indented one more space?

---

3. 10 LET K\$ = ``*`` 20 ?K\$ 30 ?TAB(3);K\$ 40 ?TAB(6);K\$ 50 ?TAB(9);K\$ 60 END	
--	--

---

**Program****Output**

```
4. 10 LET K$ = "*"
    20 ? K$;
    30 ? TAB(3);K$;
    40 ? TAB(6);K$;
    50 ? TAB(9);K$;
    60 END
```

What caused the \* to be printed on the same line?

---

**Just For Fun**

Run each program and answer the questions.

```
5. 10 FOR X = 1 TO 25
    20 ?TAB(Q*Q / 10); "♥"
    30 NEXT X
    40 END
```

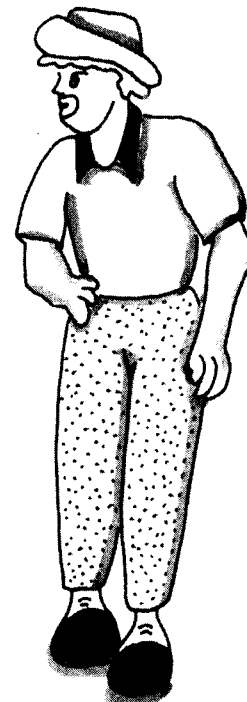
Why does the ♥ get printed where it does?

---

```
6. 10 FOR D = 0 TO 6.3 STEP .2
    20 LET G = SIN(D)
    30 ?TAB(20*G + 20); "CURVEY SCURVEY"
    40 NEXT D
    50 END
```

What part of the program do you think causes the strange shape of the output?

---



---

```
7. 10 FOR D = 0 TO 6.3 STEP .2
    20 LET G = SIN(D)
    30 ? TAB(20*G+20); ``CURVEY``
    40 NEXT D
    50 END
```

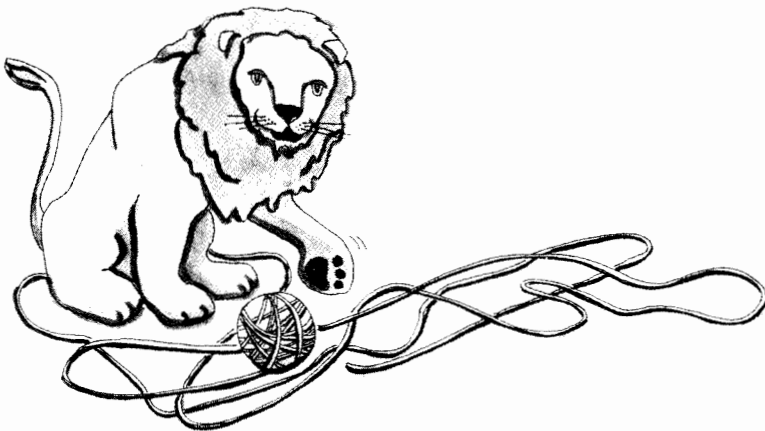
How is the curve in the output of this program different from the output curve in program #6?

---

```
8. 10 FOR D = 0 TO 9.5 STEP .2
    20 LET G = COS(2*D) + SIN(D)
    30 ? TAB(15*G + 30); ``FLAKEY SNAKEY``
    40 NEXT D
    50 END
```

Why do you think two snakes appear on the screen?

---



# PROGRAMMER'S PASTIME #65

Use easy graphics to make PET draw the pictures described below.

## Description

## Program

1. Write a program using PRINT statements to make PET draw a large set of your initials.
2. Write a program using PRINT statements to make PET draw a triangle.
3. Write a program that draws the same triangle but uses the TAB and SPC functions.



---

**Description****Program**

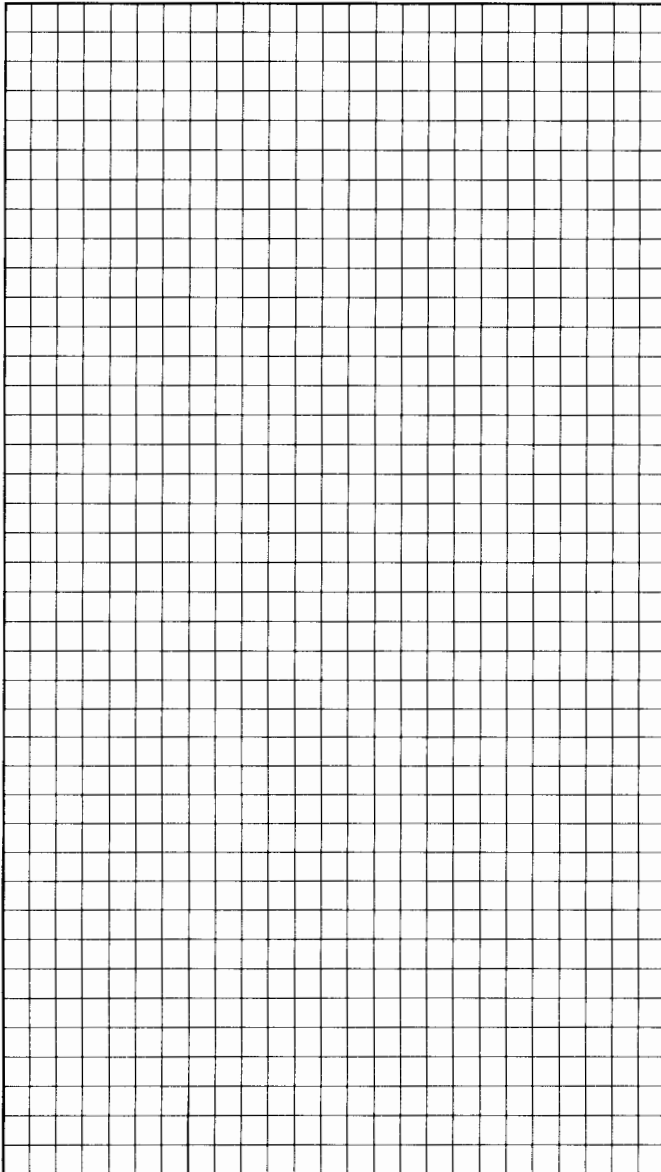
4. Write a program using PRINT statements to make PET draw a Christmas tree with a star at the top.

5. Write a program that draws the same Christmas tree, but uses the TAB and SPC functions.



# PROGRAMMER'S PASTIME #66

Use the grid below to create a picture that you would like PET to draw. (The grid is the same size as PET's screen.) On another piece of paper, write a program that will cause PET to draw your picture. Use many different graphics characters and be creative! Run your program.



Top of Screen



# PROGRAMMER'S PASTIME #68

Use the formula:

$$\text{LOCATION} = 32767 + (\text{ROW} - 1) * 40 + \text{COLUMN}$$

to calculate the POKE command for the following screen locations.

## Screen location

## Poke command

### Example:

ROW 8 AND COLUMN 39

POKE 32767 + 319

DO:  $32767 + (8 - 1) * 40 + 39$

$32767 + 7 * 40 + 39$

$32767 + 280 + 39$

$32767 + 319$

1. ROW 5 AND COLUMN 3
2. ROW 19 and COLUMN 7
3. ROW 7 and COLUMN 14
4. ROW 2 and COLUMN 16
5. ROW 24 and COLUMN 28
6. ROW 13 and COLUMN 9
7. ROW 4 and COLUMN 30
8. ROW 22 and COLUMN 40
9. ROW 17 and COLUMN 21
10. ROW 25 and COLUMN 36

---

---

---

---

---

---

---

---

---

---

---

---

# PROGRAMMER'S PASTIME #69

Use the techniques you have learned about programming for graphics and write programs for the following descriptions.

## Description

1. Write a program that will print a border around PET's screen.

## Program

2. Add to program #1 so PET prints your name in the center of the screen after the border has been drawn.

3. Create your own program using POKE and the INT and RND functions. Try using some animation. Explain what your program does.



# PROGRAMMER'S PASTIME #70

You can teach PET to write in lower-case letters (instead of capitals) by using the POKE command.

ALTERNATE MODE*	POKE 59468, 14	Changes writing from capitals to lower case.
-----------------	----------------	--

\*When PET is in alternate mode, you cannot use the graphics characters.

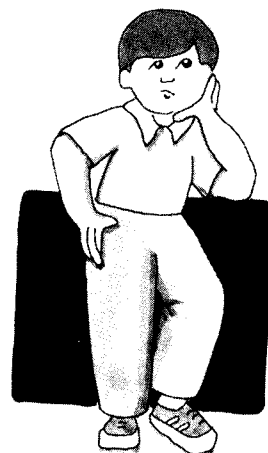
STANDARD MODE	POKE 59468, 12	Changes the writing from lower-case letters back to capitals.
---------------	----------------	---

## Description

## Program

1. Write a program that prints your full name in standard mode, again in alternate mode, and then finally in standard mode again.

2. Create a program of your own that prints output in both standard and alternate modes.



# PROGRAMMER'S PASTIME #71

Use what you know about a good game program to write the game programs described below.

1. It is more meaningful to the user when the computer calls him or her by name—it makes the interaction more personal.

Write a GUESS A NUMBER game program that asks the user's name and calls the user by name throughout the program.

1. THINK about the program
2. DATA TABLE

3. ALGORITHM

---

4. FLOW CHART

5. CODE the program

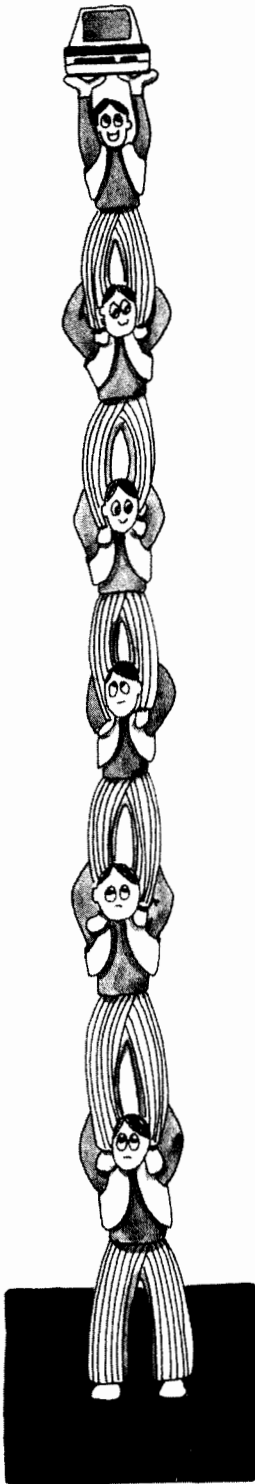
6. DEBUG  
7. REVISE



---

2. Revise the game program in Chapter 45 so a player must answer "YES" or "NO" in line 390.

If anything else is typed for INPUT, make PET print the question in LINE 380 again. This helps make the program GOOF PROOF.





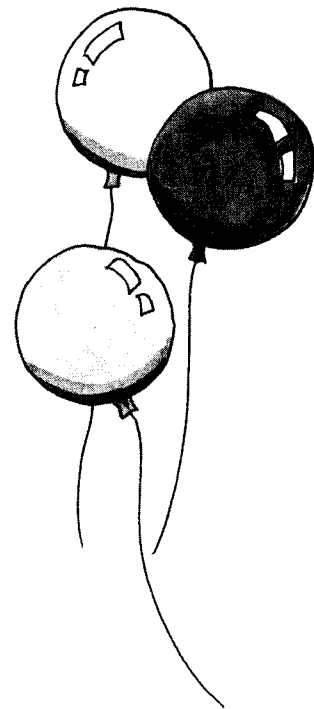
---

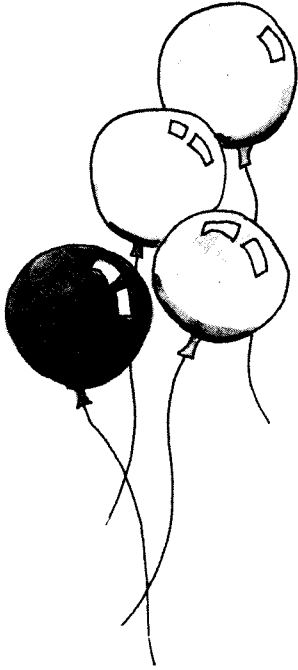
3. Add something to the game program in #2 so PET tells the user how many tries it took before he or she guessed the correct number.

HINT: Use a COUNTER, C.

Set C at 0 before the first guess. After the first guess add 1 to the counter:  $C = C + 1$ .

Then after each of the next guesses, make sure one more is added to the counter. When the correct guess is guessed, make PET print IT TOOK YOU \_\_\_\_\_ GUESSES.





4. Add something to the game program in #3 so PET asks the user the top number in the range they wish to guess (For example, 1 to \_\_\_\_\_?)

After the user types in the top number, use it in the RND function to create a random integer between 1 and the top number.

HINT: IF N = the top number THEN you would use this RND function:

LET X = INT(N \*RND(1)+ 1)

---

5. Make up a computer game that uses a die. Write a program using all of the good style techniques you've learned.

The RND function for the throw of the die must choose a random integer between 1 and 6.

1. THINK about the program.
2. DATA TABLE

### 3. ALGORITHM



---

4. FLOW CHART

5. CODE the program

6. DEBUG  
7. REVISE

---

6. Create a computer program for any game you like. Include the five things every good game program should have. Try using animation. Be creative!

1. THINK about the program
2. DATA TABLE

3. ALGORITHM



---

4. FLOW CHART

5. CODE the program

6. DEBUG  
7. REVISE

# PROGRAMMER'S PASTIME #72

## Fun Features

Did you know that PET can tell time? PET has a built-in clock that keeps real time in a 24-hour cycle by hours, minutes, and seconds. The string variable `TIMES` keeps track of the time.

To SET the clock when you turn on the computer, type:

```
TIMES = `(hhmmss)`
```

For hh type in two numbers for the HOUR of the day. The day begins at 00 for 12 AM and goes to 23 for 11 PM. PET is on a 24-hour cycle so you can tell the difference between AM and PM by the hour number. The hours from 00 to 11 stand for AM, and the hours from 12 to 23 stand for PM. The clock returns to 00 at midnight and starts all over for the next day.

For mm type in the MINUTES—00 to 59. SS means SECONDS—also 00 to 59.

When setting `TIMES` to the actual time, type in a time a few seconds in the future. When that actual time rolls around, press  to set the clock.

Example: `TIMES = `091401``

---

To find out what time it is type:

```
?TIMES
```

and PET will give you the time in hours, minutes, and seconds.

Another way to use the clock is to type:

```
TIMES = "000000"
```

when you begin using the computer. When you are done, you can refer to the clock to see how long you have been using the computer.

The clock keeps time until PET is turned off. The clock needs to be reset when PET is turned back on again.

### **Ding-a-Ling**

The PETs that speak 4.0 BASIC are equipped with a bell (sounds more like a beep) which rings when the computer is turned on, and when the cursor moves through the fifth column from the right edge of the screen.

To get the bell to ring during a program use this statement:

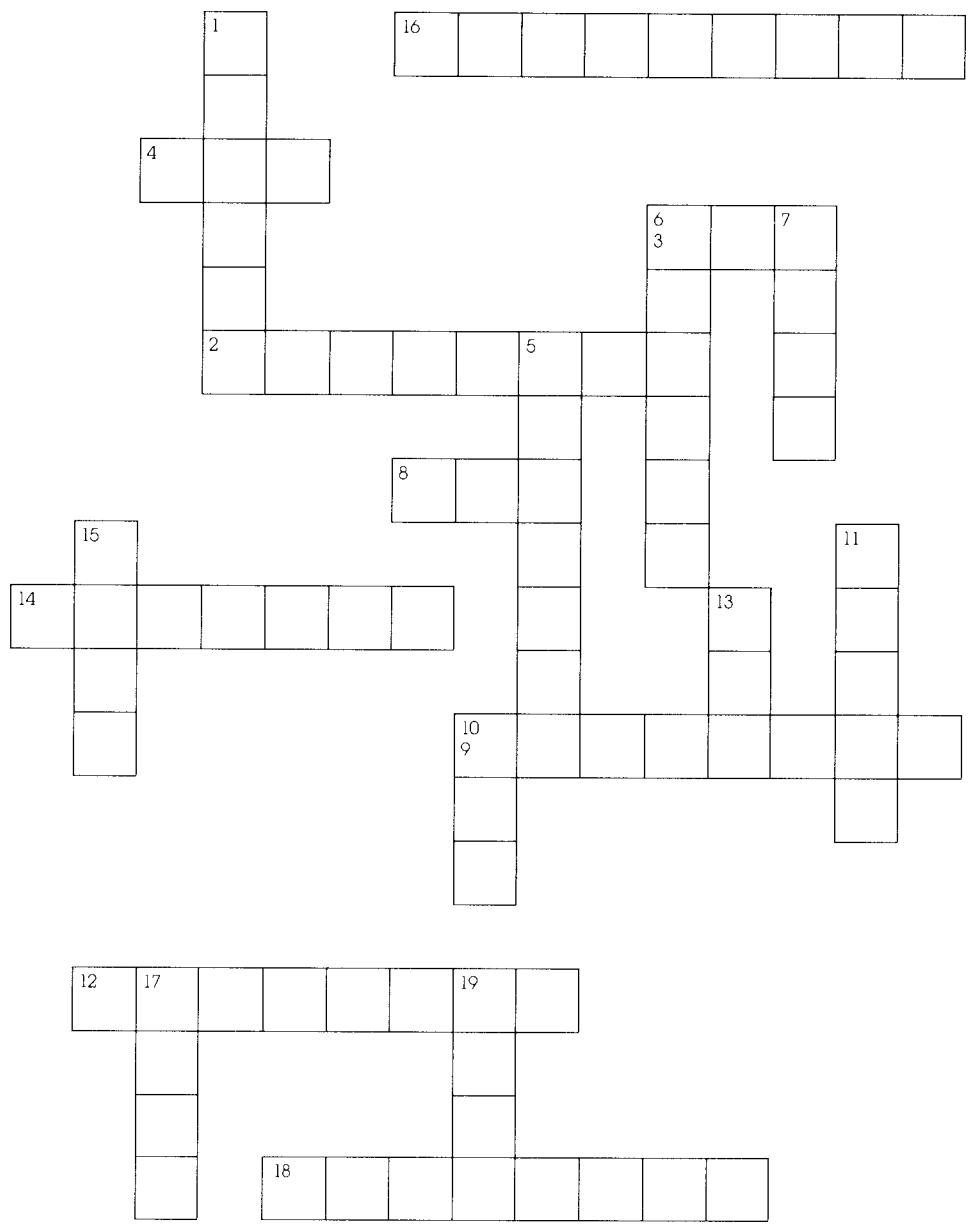
```
10 ?CHR$(7)
```

### **Ideas**

1. Practice using PET's clock.
2. Create a game program that uses the bell. For example, when the correct answer is given, make the bell ring.
3. Create a program that draws on the screen and uses the bell for sound effects.



# COMPONENT 7 FUN PAGE



**Down**

- |   |   |
|---|---|
| <p>1. Convert means to _____.</p> <p>3. A word meaning "having no pattern or special purpose."</p> <p>5. Another name for a whole number.</p> <p>7. The INT function rounds a decimal _____ to the nearest integer.</p> <p>9. A function used to control screen output.</p> | <p>11. PET's screen is _____ spaces wide.</p> <p>13. Stands for the space-over function.</p> <p>15. This command writes data into a location in PET's screen memory.</p> <p>17. PET's screen has 25 _____.</p> <p>19. Every character has a special _____ number.</p> |
|---|---|

---

**Across**

- 2. The most important part of a conversion program is the conversion \_\_\_\_\_.
- 4. Stands for "Computer-Aided Instruction."
- 6. Stands for random function.
- 8. The function that creates whole numbers in a program.
- 10. Means to "chop off."
- 12. Pictures are drawn on PET's screen by using \_\_\_\_\_.
- 14. PET's screen has 40 \_\_\_\_\_.
- 16. Giving life and motion to your drawing is \_\_\_\_\_.
- 18. Write your programs to be user \_\_\_\_\_.

**Evaluate Yourself**

- 1. Component 7 was \_\_\_\_\_  
because \_\_\_\_\_
- 2. The best parts of the component were \_\_\_\_\_
- 3. The parts I liked the least were \_\_\_\_\_
- 4. The most valuable thing I learned in this component was \_\_\_\_\_  
because \_\_\_\_\_

Other comments:



This workbook is an excellent tool for teaching children how to program a microcomputer in BASIC. It's an individualized, self-paced approach that accompanies the book **A PET FOR KIDS**.

Most activities in this workbook can be done without a computer, so can be used as seatwork (solving the problem of 1 computer and 25 students).

Written approximately at the 4th grade reading level, this workbook:

- CAN ALSO BE USED WITH THE COMMODORE 64 AND THE VIC 20
- Approaches programming in a fresh, fun way
- Is easy for kids to understand
- Contains illustrations interesting to kids
- Includes worksheets to go along with each chapter of **A PET FOR KIDS**
- Provides practice and reinforcement for programming skills your students learn
- Has activities that focus on problem solving, improved thinking skills and creativity
- Concludes each unit with fun pages that test your students' learning progress

**A PET FOR KIDS** and **A PET IN THE CLASSROOM: Teachers Manual** also available.



dilithium Press  
PROGRAMMING