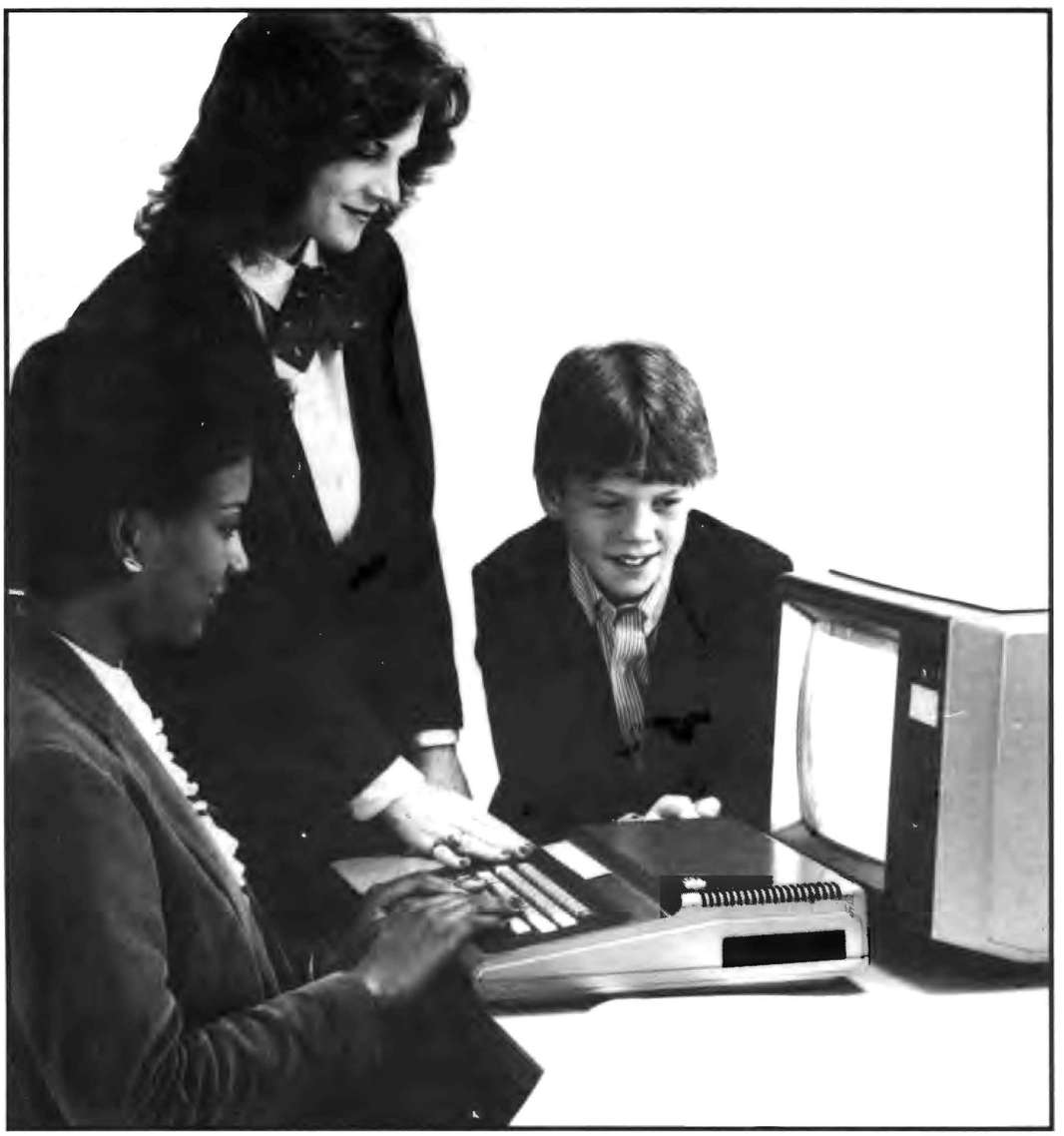
4-H COMPUTER PROJECT II:
Learning About Programming



Virginia Cooperative Extension
Virginia Tech • Virginia State

Publication 388-142
Reprinted 1992

Virginia Tech
VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY

Prepared by George A. Duncan, Lee Hays, and George Turner, Department of Agricultural Engineering, with assistance from Anna Lucas, 4-H Program Specialist; and Richard Maurer, 4-H and Community Development Specialist; University of Kentucky, Lexington, Kentucky.

Computer Project II: *Learning About Programming*

Introduction

Welcome back to the world of computers. In *Computer Project I: Learning About Computers* you learned how the parts of the computer function, how to operate a computer keyboard, how to do calculations on a computer, how to take care of cassette tapes and diskettes, how diskettes and cassette tapes store information, and how to *load* and *run* programs from the cassette tapes and diskettes. In this project you will learn about programming a computer.

In Project I we discussed the word "program." Do you remember the meaning of a program? Let's review. A program is the statements and commands stored in the computer by the operator to instruct the computer how to perform. Programs must be written to the specific format for each type of computer. If you do not follow the format to the letter, the computer will not understand. This is one characteristic that makes each type of computer unique.

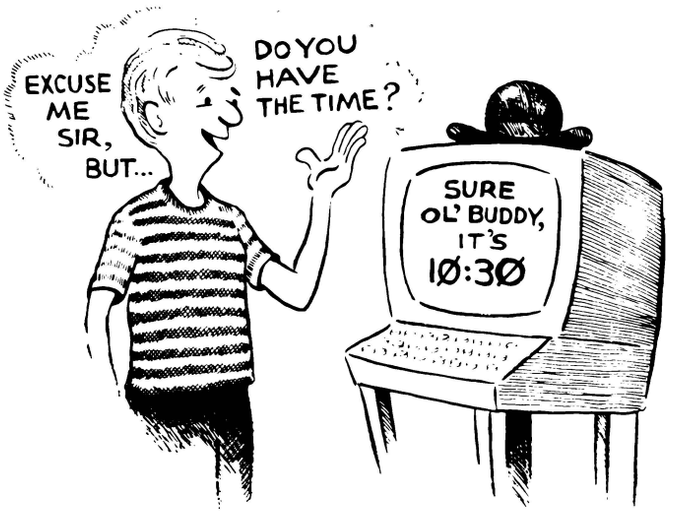
In this project you will be asked to type and run programs. The programs are already written for you in ordinary language. What you must do is look on pp. 37-38 and select the computer language—the matching computer commands or keys for your unique machine. You must type in the computer commands letter by letter. To type the keys, you simply locate and press the correct ones. Following the exercises on commands and their functions, you will complete activities in which you write your own programs.

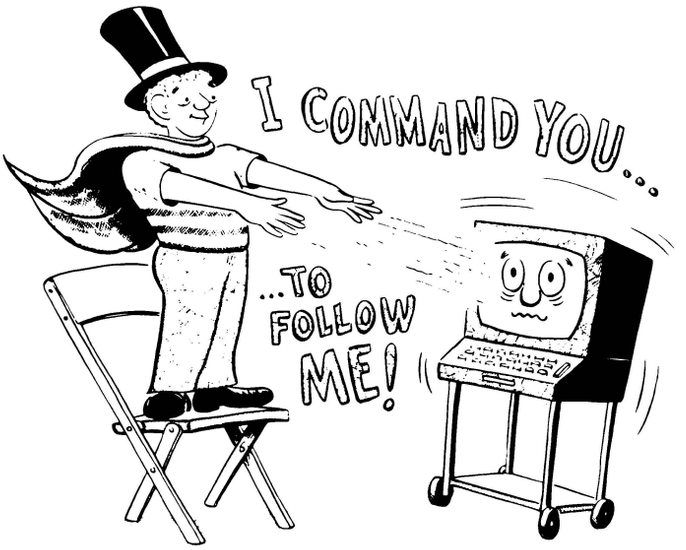
What You Will Learn in This Project

- How to communicate with the micro-computer
- How to make information appear on the video screen
- How to write, run and list a program
- How to use input and produce output in various ways
- How to recognize specific computer instructions

What You Will Do in This Project

- Type commands
- Write programs
- Run programs
- Change short programs
- Give a demonstration on a computer
- Keep a record of your 4-H computer project





Computers Are All Around Us

Computers are used widely. You can find them in banks and businesses, in many government offices, in schools, and in some homes, maybe yours. You can even find them in typewriters, microwave ovens, cars, and wristwatches.

Where have you seen computers used in your community?

_____

_____

_____

_____

_____

You, a Programmer?

The following activities will let you type short programs into the computer and see what happens. By observing the results of certain commands, you will learn how to program the computer. Each activity shows you a new command word and asks you to apply the command. Be sure to learn how these command words work since you will be using them to write programs in this computer project. The activities in this project book use the BASIC computer language—the language used by most microcomputers.

Before You Begin

First you must know the type of computer you will be using. Then you must find the computer keys or commands for your computer that match the English instructions below. Use the list on pp. 37-38 and your computer's instruction manual. When you find the proper key or command for each of these instructions, write the names in the spaces provided for an easy reference. If a computer is available, find the keys on the keyboard. The four instructions below will occur quite frequently as you work through this project. Write the name of the key or command that matches each instruction.

- stop execution key: _____
- clear key: _____
- enter/return key: _____
- new program command: _____

The following is a description of important keys on the keyboard and symbols that appear on the video display (screen). You will use these often during this project. If a computer is available, locate the keys and symbols. Remember, you will have to check the list on pp. 37-38 to find some of these keys.

Key or Symbol	Use
stop execution key	A key (or keys) that stops the computer and tells it to wait for a new command.
clear key	A key (or keys) that clears the video display but does not erase information in memory.
enter or return	A key that causes the computer to respond to any statement or information just typed (used as **ENTER/ RETURN** in the rest of this project book).
> or **OK** or]	These symbols are called the prompt and mean that the computer is ready to receive a new statement or command. Computers have different prompts. These are some examples. The prompt appears on the screen after you press the **ENTER/RETURN** key.
_ or ■	The underline is called the cursor and shows you, on the screen, where you will type the next character. Some microcomputers use a blinking cursor. Also, the cursor could be a white rectangle rather than an underline.
	The left arrow is the backspace key on many microcomputers and is handy to correct errors. On some equipment it erases each character you backspace over. Just type or space again, and you are on your way.
	Quotation marks
	Comma
	Semicolon
	"at" symbol
	Equal key (S)*
	Subtraction key
	Multiplication key (S)*
	Addition key (S)*
	Period or decimal key
	Divide key
	Exponential key
	Greater than
	Less than
	These are the numbers one and zero, which are two of the 1Ø numbers on the keyboard.
	These are the letters I and O, which are two of 26 letters on the keyboard. Notice the difference between the two letters and the two numbers.
shift	Shift key is used with (S)* keys.
space bar	This is called the space bar key. If a blank is needed in a certain spot while typing, press the bar.

*(S) means you must press the shift key while pressing that key to produce the symbol desired.

Activity 1. Your First Program

You will learn how to clear the video; print words onto the video; use the *NEW, END, RUN,* and *LIST* commands; and the importance of line numbers.

1. First, you may need to power-up the computer and obtain the prompt indicating the BASIC language is ready to use. (Check the operations manual for your computer.) Then prepare the computer for a new program by pressing the stop execution key, clear key, **ENTER/RETURN** key; typing the new program command and pressing the **ENTER/RETURN** key. (For example, for one type of computer you may press **BREAK, CLEAR, ENTER:** type **NEW** and **ENTER.)**

2. Locate the following instructions in the list on pp. 37-38 and see how specific commands or statements for the computer are written. (See *Sample Program* below. Remember, this is an example for one type of computer. Yours may be different.)

Instructions	**Sample Program**
10 Clear the screen	**10 CLS**
20 Print to video "your name" (Put your name between the quotes.)	**20 PRINT "your name"**
30 End of program	**30 END**
Run the program	**RUN**

Now do the same program for your machine. Look up the various computer commands and write the proper statements and commands for your machine. You may need to refer to the instruction manual for your machine.

Instructions	**Write Your Program**
10 Clear the screen	**10**
20 Print to video "your name" (Hint: Put your name between the quotes.)	**20**
30 End of program	**30**
Run the program	**RUN**

3. If a computer is available, type your program into it. If you make a mistake in typing, use the backspace key to back up and correct the mistake. When you have finished line 30, type **RUN** and press **ENTER/RETURN.** What happened? If you had an error in the program, a "syntax" or other error message occurred. Don't worry; skip down to paragraph 5, **LIST** your program, check for the error, correct it, **RUN** the program again and continue with paragraph 4.

4. Congratulations! You have written and **RUN** your first computer program! Let's review the steps that occurred. Typing the command word **NEW** tells the computer to forget the old program, if there was one, and to receive a new program. The clear screen command clears the video display. Your name appears on the top line as a result of the **PRINT** command. **END** signals the end of the program. The numbers identify each command or statement line and the order of execution. **RUN** causes the computer to execute the program. Easy, isn't it?

5. To see what your program looks like, press the clear key, type **LIST,** and press **ENTER/RETURN.** The program that is stored in the computer's memory will appear on the video. You may correct or change any line by merely typing the entire line again. The computer will insert the line in the proper place in the program according to the line numbers.

The Importance of Line Numbers

The line numbers you have been using, 10, 20, 30, 40, etc., identify each command or statement line and determine the order of execution. You could have used the numbers 1, 2, 3, 4, 5, 345, 678, etc. Usually, numbers are skipped so new lines can be added between the existing lines as you will later learn and do. Each line may contain up to 255 characters in most versions of the **BASIC** language. Line numbers are useful when you want to change the commands in specific lines or change the order of execution of a program. Line numbers are like house numbers; you can tell exactly where to go in a program. Now let's try a few changes and additions.

6. Put a new name on line 20. Try it by typing:

20 PRINT "new name"

and press **ENTER/RETURN.** Type **RUN** and press **ENTER/RETURN.** The new name appears. Notice that by typing an existing line number with new information following, you replace the old information with new information, whereas a new line number will add another line in numerical order.

7. Add the following lines and run the program:

20 PRINT "your name"
22 PRINT "your address"
24 PRINT "your city, state, zip"

What happened? _____

_____

_____

8. Now you may better understand why line numbers have spaces left between them. You may want to add more lines later.

9. Try some other names and addresses by retyping lines 20, 22 and 24.

What have you told the computer to do? _____

_____

10. The **PRINT** command used alone will cause a blank line to be added. It is useful when you want to put blank lines in your output on the video.

Are You Ready to Write Your Own Program?

1. Type the new program command and press the **ENTER/RETURN** key.

2. Write a short program to clear the screen, print the day, month and year on one line and run the program.

Write your program below. Remember to include line numbers beside each line.

_____

_____

_____

_____

3. Now type your program into the computer and run the program. If the day, month and year do not appear on the screen, then type **LIST** and press **ENTER/RETURN.** Check for your mistake and correct it by typing the line over. Did you get information from the previous program? If so, remember to type **NEW** when starting a new program.

Activity 2. Going By the Road Map!

You will learn how to use the GOTO command and a flowchart. The GOTO command allows program transfers to specific line numbers; it lets you skip or repeat lines. A flowchart is a diagram showing how a program works. It lets you follow the commands and statements just like you would follow a road map through the country.

1. Very few programs follow all the line numbers in an exact order from beginning to end. You may want to repeat or skip portions of the program. The **GOTO** command and the line numbers allow you to do this.

2. Retype a short program you used in Activity 1 into the computer and use the following line instead of the **END** command:

30 GOTO 20
(Note: Line 20 should be a **PRINT"words"** command in your program.)

3. Type **RUN** and press **ENTER/RETURN.**
What happened? _____

_____

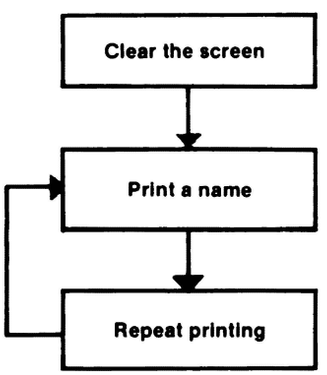
How do you properly stop the execution of the program? _____

4. The **GOTO** command tells the computer to go back to line 20 and repeat execution of line 20 and all lines that follow up to line 29. This forms a "loop" that continues indefinitely until you stop the computer. The proper way to stop the computer is to press the stop execution key (**BREAK** or equivalent) you reviewed on page 4.

5. Try some other **GOTO** commands in your program.

6. Following is a block diagram to illustrate the program you have just used. These diagrams can be called "flowcharts" and are very helpful in planning a long, complicated program or determining how one works. Sketch a flowchart of one of the programs you used in Activity 1 or 2. You will learn more about flowcharts later in this project.

Sample Flowchart



Your Flowchart

Activity 3. Punctuation , ; : It Changes Things!

You will learn the importance of the comma, semicolon and colon (, ; :) with the print to video instruction.

1. Prepare the computer for a new program by pressing the stop execution key, clear key and **ENTER/RETURN.** Type the new program command and press **ENTER/RETURN.**

2. Locate the following instructions in the table, and write the commands or statements for your computer in the spaces to the right of the instructions.

Instructions	**Write Your Program**
10 Clear the screen	_____
20 Print to video "your name"	_____
30 Go to line 20	_____
Run the program	_____

No punctuation after **PRINT** " " causes your name to be printed how?

_____

3. Stop the program and clear the screen. (Remember how?)
Type in a new line 20 by adding a *comma* at the end:

20 PRINT "your name",

Run this program. Your name will be printed in how many columns? _____

4. Stop this program. Type a new line 20 by adding a *semicolon* in place of the comma:

20 PRINT "your name";

Run this program. Your name will fill the screen and may "march" left or right, depending on the spaces in your name. Why? _____

_____

5. Stop the program and add a few *spaces* to your name between the " ", such as:

20 PRINT "your name ";

Then run the program. What is the difference? _____

_____

6. Stop the program and type a new line 20 with a *colon* and another **PRINT** command on the same line:

20 PRINT "your name":PRINT "your address"

Run this program.
What does the colon permit you to do? _____

_____

7. Stop the program and type a new line 20 with multiple commas on the same line.

20 PRINT "your name", "your favorite food", "your weight"

Run this program. Then press the stop execution key.
What do the multiple commas produce on the video display?

_____

_____

8. Stop the program and type a new line 20 with multiple semicolons on the same line. Insert your favorite number between the quotes.

20 PRINT "your favorite number"; "your favorite number minus one";

Run this program. Press the stop execution key. What do the multiple semicolons produce on the video display?

_____

_____

How would a single semicolon after **PRINT** " " cause your name to be printed?

_____

_____

Are You Ready to Write Your Own Program?

1. Prepare the computer for a new program.

2. Write a program to clear the screen and print your name and city on one line in columns. On the next line print your favorite hobby and then your birthday. The video display should look as follows:

your name	your city
your hobby	your birthday

Write your program here. Remember to write the line numbers beside each line.

_____

_____

_____

_____

3. Now type your program into the computer. Type **RUN** and press **ENTER/RETURN.** If the program does not work, press the clear key, type **LIST** and press the **ENTER/RETURN** key to see your program and check for a mistake. To correct a mistake, retype the line number and a new or corrected statement.

Frequently numbers will be omitted between numbered lines so that other lines of information may be added later. Usually line numbers will increase by 10, to allow you to go back and add extra information if needed. The computer automatically arranges the lines in sequence beginning with the smallest number and ending with the largest number. It "executes" the commands in the same sequence, unless a special command, such as **GOTO,** tells the computer to "jump" to some other line number.

Let's Review

You have used lots of new words and symbols to communicate with the computer. Stop for a moment and write the meaning for these computer commands or statements.

CLS or **HOME** _____

GOTO _____

PRINT " ", _____

PRINT " "; _____

PRINT " ": _____

LIST _____

NEW _____

END _____

RUN _____

Line number _____

Flowchart _____

If you had trouble remembering the meaning for these computer commands, refer to the first two activities. *Do not* go on until you know and understand these commands and keys.

Activity 4. Where Am I?

You will learn how to print information beginning in different columns. This procedure lets you put information in table form.

1. Clear the computer for a new program. (Remember how from Activities 1 and 2?)

2. Locate the following statements in the table, and write them in the appropriate language for your computer in the spaces to the right of the statements. Then type your program into the computer!

Instructions	**Write Your Program**
10 Clear the screen	_____
20 Print in column 25 "your name"	_____
30 Go to line 20	_____
Run the program	_____

Remember to press **ENTER/RETURN** key after each command.
With this program you can position your name anywhere on the line by putting a column number in the ( ).

3. Stop the program and clear the screen.

4. Type in a new line 20 as above except changing the column 25 to some other number that is not larger than the screen width. Run the program. Do you see how you can move your name anywhere on the horizontal line?

Understanding the Print Command

The **PRINT** command is used as an "output" command to show results on the video display. There are several forms or versions of the **PRINT** command. Here are some of the easy ones.

PRINT TAB(n):
This command prints information at a specified tab or column position anywhere from 0 (the left of the screen) to the end of the screen line. You put the desired position number in the ( ). A blank space between the T's of **PRINT** and **TAB** is optional on some microcomputers and required by other microcomputers. Using the blank space is good for clear reading of the statement, therefore, its use is recommended. *Do not* put a space between **TAB** and **( ).** In the following examples, notice the use of proper punctuation between the commands.

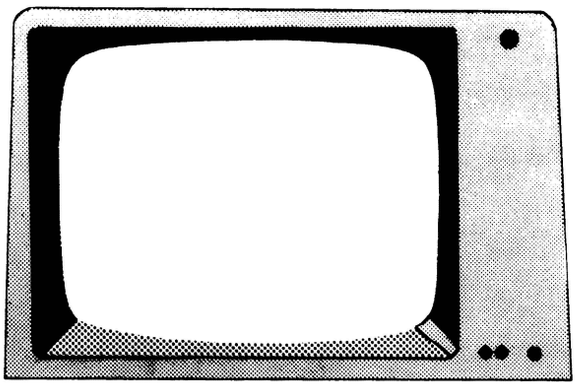
```
PRINT TAB(3) "name";TAB(20) "address";
PRINT TAB(3) "city, state  zip code"
```

To put several items on the same line, use a semicolon after each entry and repeat **TAB(n).**

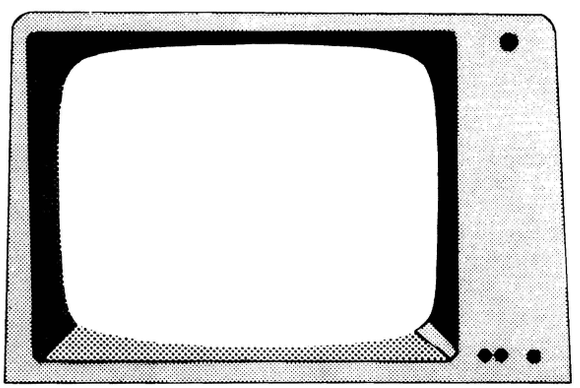
Exercises Using the Print Command

1. On the following page, show on video screens where the **PRINT** commands will print the words.

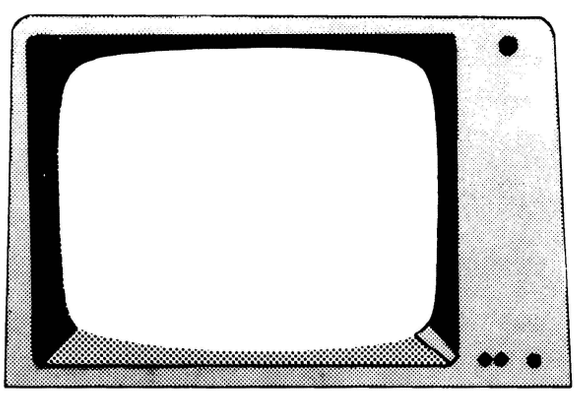
a. **10 PRINT "FRED"**



b. **10 PRINT TAB(20) "SALLY"**



c. **10 PRINT "SALLY LEAPER"**
20 PRINT "2468 SHORT STREET"
30 PRINT "PARIS, US 42471"



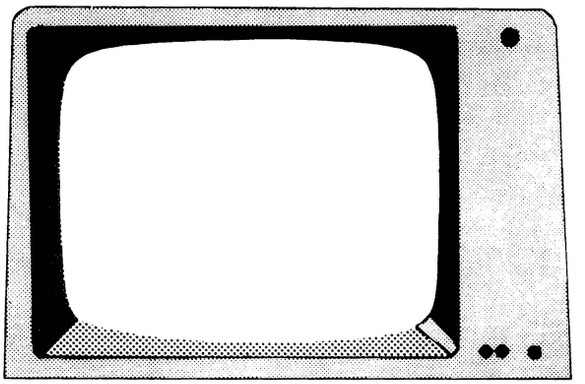
_____

_____

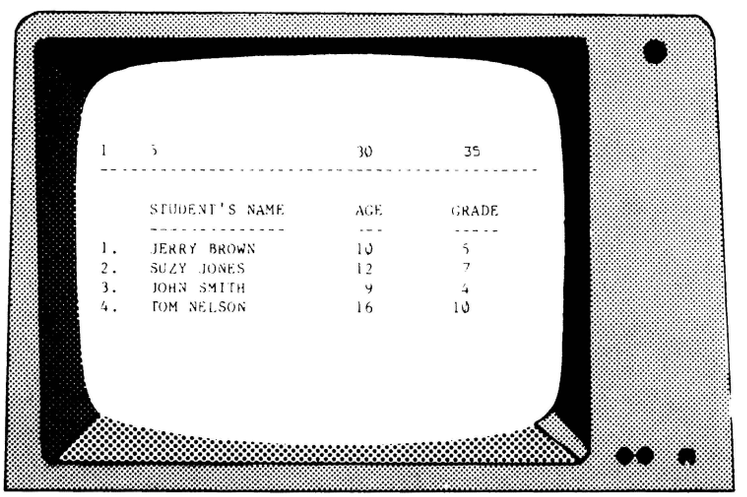
_____

_____

d. **10 PRINT "TIM ANGEL";TAB(15) "103 STATE ST"**
20 PRINT TAB(10) "MIDWAY, US 40671"



2. Write the **PRINT** commands that tell the computer to print the information below.



Write your program here:

_____

_____

_____

_____

_____

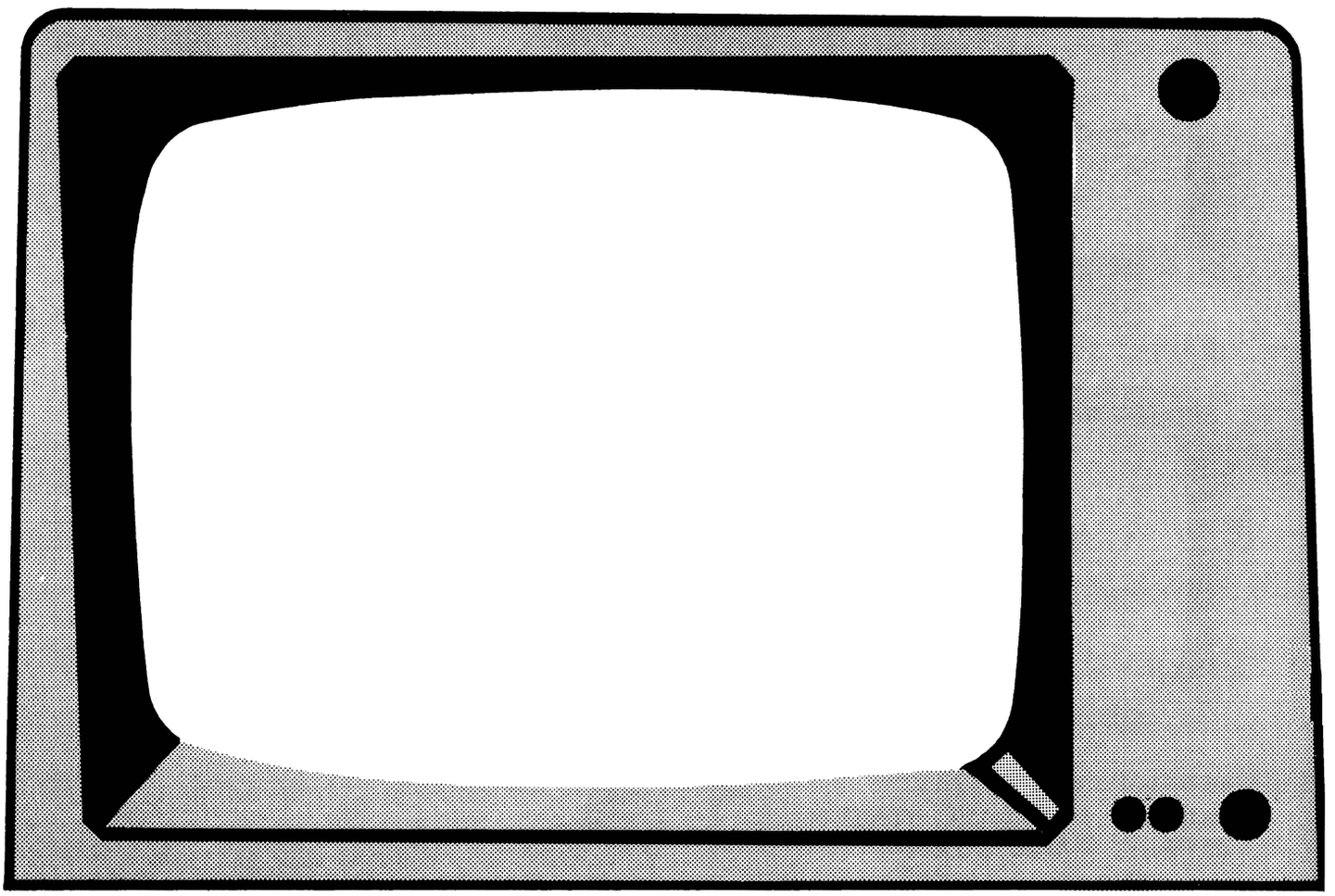
_____

_____

_____

Run your program to see how you did. Did you write it correctly? _____

3. Think of some information (poem, verse, message, letter, etc.) you would like to program into the computer. Write it on the video display outline below. Then write **PRINT** commands to program the information into the computer.



Write your program here:

Type the program and run it. Does it look the way you thought it would? _____ If not, rework it until it appears on the video display as it should.

Activity 5. A Pinch Hitter!

You will learn how to use a variable for characters and a variable for numbers. A variable is used to represent or "pinch hit" for something else. By using a variable, you can use one or two letters for a long name, numbers or other information.

1. Clear the computer for a new program.

2. Locate the following instructions in the table. Write them in the appropriate language for your computer in the spaces to the right of the instructions.

Instructions	Write Your Program
10 Clear the screen	_____
20 A variable for characters equals "your name"	_____
30 Print to video the variable for characters	_____
40 Go to line 30	_____

Does this program look familiar? Where did you get the same results on the screen?

_____

What is different? _____

_____

3. Remember, a variable is used to represent or "pinch hit" for something else. By using a variable, you can use one or two letters for a long name or other information. The symbol $ after a variable tells the computer to expect letters or words. For humans, a dollar sign $ indicates money. The computer understands this symbol to mean that alphabetic letters or other characters are coming. Stop the program and retype line 30 as follows. Then run the program again.

30 PRINT A$, (Use the comma.)

4. Then try:

30 PRINT A$; (Use the semicolon.)

and run the program.

5. A letter without the $ symbol is a pinch hitter (variable) for a number, not a letter. Try the following lines and run the program.

```
20 X = 1
30 PRINT X
```

(Remember, **PRINT 3 - 2** was used in *4-H Computer Project I, Learning About Computers*.)

6. Can you think of some other examples using **A$**= and **X=?**

NOTE: You can use any one or two letters, or a letter and a number, for a variable in most versions of the BASIC programming language. Only one-letter variables are shown here for simplicity. Can you think of some different variable names using two letters or a letter and a number?

Are You Ready to Write Your Own Program?

1. Prepare the machine for a new program.

2. Write a program to set the variables C$ and T$ equal to:
   C$ = "the name of the computer you are using"
   T$ = "the time you began using the computer"
   Print the two variables so the output appears as below. Program the words **MACHINE:** and **TIME:** to start in column 6 on your video screen. The name of your computer and the time you began using the computer should start in column 18.

MACHINE:	the computer name
TIME:	the time

Write your program here:

3. Type and run your program. If the program doesn't work (an error message occurs), press the clear key, type **LIST** and press the **ENTER/RETURN** key to see your program and check for a mistake. To correct the problem, retype the line number and a corrected statement. Run the program again.

Activity 6. Entering More Than Once

You will learn the input command, how to input characters and numbers using variables, and how to use the word prompts with the input command. The INPUT command causes the computer to pause and ask for some additional information. The information is stored in memory with the variable until you command it to be printed.

1. Clear the computer for a new program.

2. Locate the following instructions and write them in the appropriate commands for your computer in the spaces to the right of the instructions.

Instructions	**Write Your Program**
10 Clear the screen	_____
15 Print to video the words "ENTER YOUR MESSAGE"	_____
20 Input characters using a variable for characters	_____
30 Print to video a blank line	_____
31 Print to video the words "THIS IS YOUR MESSAGE"	_____
32 Print to video the variable for characters	_____
40 Print to video "- - - - - - - - - - -"	_____
45 Go to line 15	_____
50 End of program	_____
Run the program	_____

Do you see "**ENTER YOUR MESSAGE**" on the video? There may be a prompt symbol following, depending on the type of computer. The computer wants you to type in some information. (Now type in your name or other message and press **ENTER/RETURN**.)

Remember, the **INPUT** command causes the computer to pause and ask for some additional information. The information is stored in memory with the variable until you command it to be printed.

3. Did your program work? Here's how it should work. Line 15 prints the words "**ENTER YOUR MESSAGE**" to the video. Line 20 is an **INPUT** command which takes the characters you type and stores them in memory as the variable for characters. The **INPUT** command gave you a prompt symbol to let you know the computer was ready to receive your information. After you typed your message or information and pressed **ENTER/RETURN,** the program continued execution of lines 30 through 45 by printing a blank line, the words "**THIS IS YOUR MESSAGE**", the message you entered, a dashed line, and looped back to lines 15 and 20 prompting for another message. This loop repeats until you stop execution of the program.

4. If your program didn't work properly or gave a "syntax" or other error message, **LIST** the program, check your commands and punctuation, and try rerunning the program until it works properly. Then you can learn some variations of the **INPUT** command in the following paragraphs.

5. Stop the program. Retype lines 15, 20, 31 and 32 as follows:

15 PRINT "ENTER YOUR NUMBER"
20 INPUT X
31 PRINT "THIS IS YOUR NUMBER"
32 PRINT X

Type **RUN** and press **ENTER/RETURN.** There should be a prompt on the screen alerting you that the computer is waiting for data. After the input prompt, type your favorite number and press **ENTER/RETURN.** Remember, variables without $ will accept only numbers from the user. What happened?

_____

_____

6. Stop the program. **LIST** the program. Retype lines 15, 20 and 32 as follows:

15 (Note: type 15 and press **ENTER/RETURN** to eliminate this command.)
20 INPUT "ENTER YOUR MESSAGE"; A$
32 PRINT A$

Type **RUN** and press **ENTER/RETURN.** After the input prompt, type your name or message. Press **ENTER/RETURN.** What is different about this variation of the **INPUT** command?

_____

_____

7. Stop the program. **LIST** the program, retype lines 32 and 45 as follows:

32 PRINT "OUTPUT",A$
45 GOTO 31

Run the program. After the input prompt, type your name or message and press **ENTER/RETURN.** What happened?

_____

_____

8. Can you think of some uses for the **INPUT** command? _____

_____

9. Stop the program. **LIST** the program and retype lines 20 and 32 as follows:

20 INPUT A$,B$
32 PRINT A$,B$,A$

Run the program. After the input prompt appears type: **HI, THERE.** Make sure you place the comma to separate the two inputs. Press **ENTER/RETURN.** What happened?

_____

_____

10. Stop the program. **LIST** the program and retype lines 20 and 32. Type a new line 33.

```
20 INPUT "ENTER YOUR NAME AND GRADE";X$,Y
32 PRINT X$
33 PRINT Y
45 PRINT
```

Type **RUN** and press **ENTER/RETURN.** After the input prompt, type your name and your grade in school separated by a comma. Remember that the $ symbol after a letter indicates alphanumeric information—information containing letters and numbers. Without the $ symbol the numbers are the only data that the computer will accept. What happened?

_____

_____

Before You Write Your Own Program

1. Prepare your computer for a new program.
2. Locate the following instructions in the table and write the commands or statements for your computer in the spaces to the right of the instructions.

Instructions	Write Your Program
10 Clear the screen	_____
20 Input data using two variables for numbers	_____
30 Print to video one variable for numbers;"+"; additional variable for numbers	_____
40 End of program	_____
Run the program	_____

3. After the input prompt or once the machine is in a waiting state, type: **3,1.** Make sure you place the comma to separate the two inputs. Press **ENTER/RETURN.** What happened?

_____

_____

4. Now retype line 30 as below:

```
30 PRINT X;"+";Y;"=";X+Y
```

Run the program. Input the numbers **3,1.** Make sure you place the comma properly. The output should look as below:

3+1 = 4

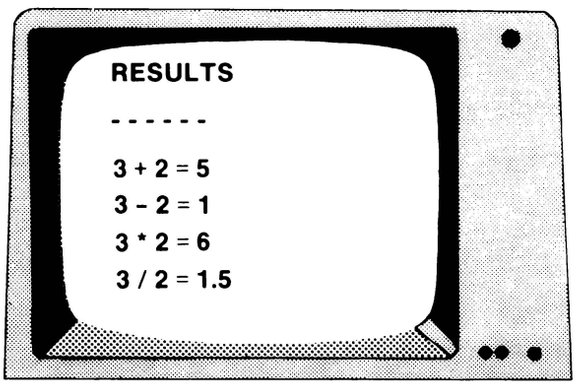
Did you get the right output? _____

5. Stop the program and clear the video. Type **LIST,** press **ENTER/RETURN.** Notice line 30 with the **X+Y** at the end of the statement. Remember how we used the calculator in 4-H Computer Project I to find the answer to **X+Y?** For example, **PRINT 3+1** will compute and print **4** on the computer video. The statement above using X and Y as variables works in the same manner. The semicolons, as you should already know, display every item following one another on the *same* line. Now try writing your own program using the same ideas in this exercise.

Are You Ready to Write Your Own Program?

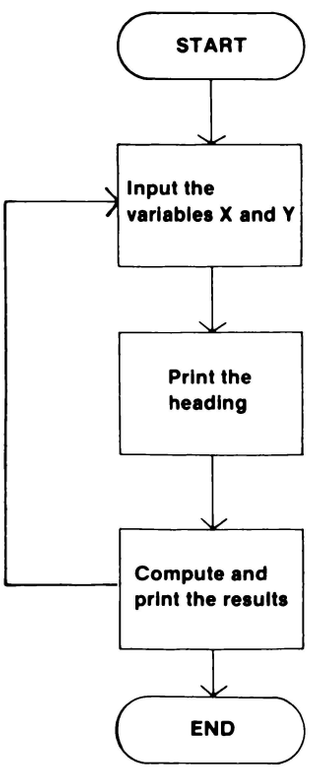
1. Prepare the machine for a new program.

2. Write a program with the input variables of X and Y. The values of X and Y are not given. You must input them with an input statement and enter them from the keyboard. Then compute and print the sum, difference, product and quotient of the two numbers X and Y with a title heading called **RESULTS.** Review the section "Before You Write Your Own Program" to help you solve the program. For example, the output would look like this if the inputs were 3 for X and 2 for Y. Write the program to repeat the input and print loop so you can use different numbers.



3. The program should accept the two inputs separated by a comma.

4. To help you write your program look at the following flowchart. It shows the order in which the computer will read and perform your program directions. (For an explanation of a few flowchart symbols refer to p. 36.)



Write Your Program Here

Write your program here:

_____

_____

_____

_____

_____

_____

_____

_____

Let's Review

Stop for a moment and answer these questions.

1. What command word will show your program on the screen? _____

2. What does a $ symbol mean to a computer when you are using it after a variable in the commands? _____

3. What does a **TAB(25)** mean to the computer when used in a **PRINT** statement?

_____

4. What program command allows you to enter data from the keyboard to the computer within a program? _____

If you had trouble with these questions, refer to the first five activities. Do not go on until you know and understand the answers to the questions.

Activity 7. Is It True?

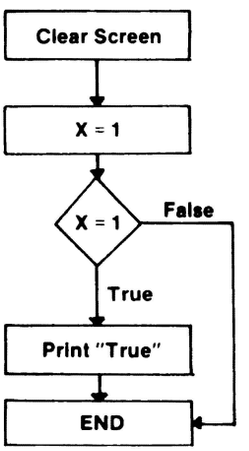
You will learn the instruction: IF an expression is true, THEN perform certain commands. The IF . . . THEN command allows logical comparisons of numbers or characters and then the performance of certain commands depending on the truth or falsity of the comparison.

1. Clear the video and memory for a new program.

2. Locate the following instructions in the table. The sample program shows the proper commands for one computer.

Instructions	Sample Program
10 Clear the screen	10 CLS
20 A variable for numbers equal to 1	20 X = 1
30 IF X equals 1 is true THEN print to video "TRUE"	30 IF X=1 THEN PRINT "TRUE"
40 End of program	40 END
Run the program	RUN

Flowchart



Now do the same program for your machine by looking up the instructions and writing the statements and commands.

Instructions	Write Your Program
10 Clear the screen	
20 A variable for numbers equal to 1	
30 IF X equals 1 is true THEN print to video "TRUE"	
40 End of program	
Run the program	

Type your program into the computer. Type **RUN.** Press **ENTER/RETURN.** The program should print **TRUE** because the condition X = 1 is true. Type **NEW** when finished to get ready for the program below.

3. Try this program on your own:

Instructions	Write Your Program
10 Clear the screen	
20 A variable for numbers equal to 4	

30 IF X is greater than 3 THEN print in column 10 "TRUE" (Hint: Look up the greater than symbol in the table.) _____

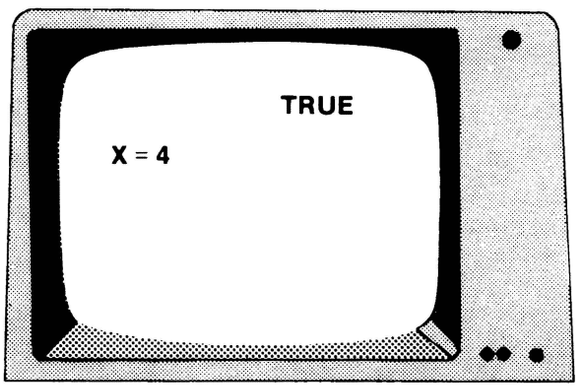
40 PRINT to video "X=" and the variable for numbers _____

50 End of program _____

Run the program _____

What did the program print on the screen? _____

The program should have printed:



The expression X is greater than 3 is true, and as a result, the word **TRUE** and X=4 are printed on the screen. If the variable X is not greater than 3 (a FALSE condition), the computer will skip anything listed after the **THEN** in the IF...THEN statement, and the program execution continues on the next line. Therefore, the command **PRINT "TRUE"** would never be executed by the computer when the program is run.

4. Stop the program. Type in a new line 20 by changing the 4 to a 2 as follows:

20 X = 2.

Run this program. What is printed now? _____

It should have been printed only **X = 2** because 2 is less than 3. (Look up the less than symbol in the list on p. 5.) Thus, the **IF** statement is false. The computer skips what follows **THEN** and **prints X=2 as stated by the next line.**

Are You Ready to Write Your Own Program?

1. Prepare the machine for a new program.

2. Write a program to input values for the variables X and Y. The program should print the message **RESULT < 10** if the quotient of X divided by Y is less than 10, the message **RESULT > 10** if the quotient is greater than 10, and the message **RESULT = 10** if the

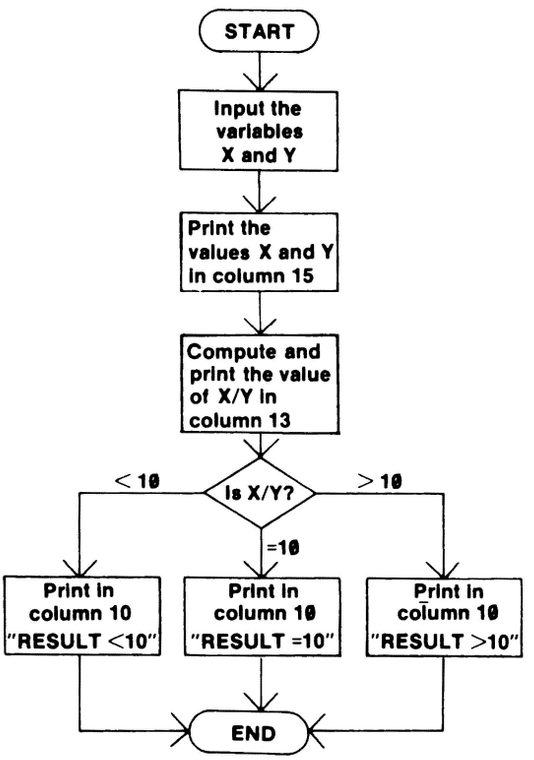
quotient equals 10. For example, with inputs X = 3 and Y = 1 the output should look the same as below including printing in the desired columns.

Columns: 0----------10-----15----------

```
X=3
Y=1
X/Y=3
RESULT <10
```

3. To help you write your program look at the following flowchart. It shows the order in which the computer will read and perform your program directions.

Write your program here:



4. Type **RUN** and press **ENTER/RETURN.** If the program doesn't work, then clear the video and list the program to check for a mistake. To correct the problem, retype the line number and the new statement. Keep trying until you get a correct program—this process of correcting your logic is valuable in computer programs.

5. What part of this program caused you the most trouble? _____

_____

Activity 8. Counting Fast!

Using a variable for numbers, you will learn how to index from a low value to a high value in a loop with a NEXT command and STEP increment. The FOR . . . NEXT command forms a loop that causes the computer to count from the smaller value given to the larger value and to perform statements between the FOR line and the NEXT line.

1. Prepare the computer for a new program.

2. Locate the following instructions in the table and write them in the appropriate commands for your computer in the spaces to the right of the instructions.

Instructions	Write Your Program
10 Clear the screen	_____
20 Use a variable for numbers and index from 1 to 10	_____
30 Print to video the variable value for the number	_____
40 Index to the next variable number	_____
50 End of program	_____
Run the program	_____

Did your program run successfully the first try? _____ If not, check for mistakes and try again.
If the program did run, can you count this fast? _____

Remember, the **FOR...NEXT** command forms a loop that causes the computer to count from the smaller value given to the larger value and perform the statements between the **FOR** line and the **NEXT** line. In this example, the value of X, the "counter," is printed out during each time through the "loop." (NOTICE: The loop is increased by 1 until it reaches 10.)

3. Retype line 20 as below and run the program.

20 FOR X = 1 to 20 STEP 2

What does the **STEP** command do? _____

_____

4. List your program.

5. Now retype line 30 as below and run the program.

30 PRINT X, X + 1 (Notice the comma.)

How did the numbers appear on the video? _____

_____

Are You Ready to Write Your Own Program?

1. Prepare the machine for a new program.

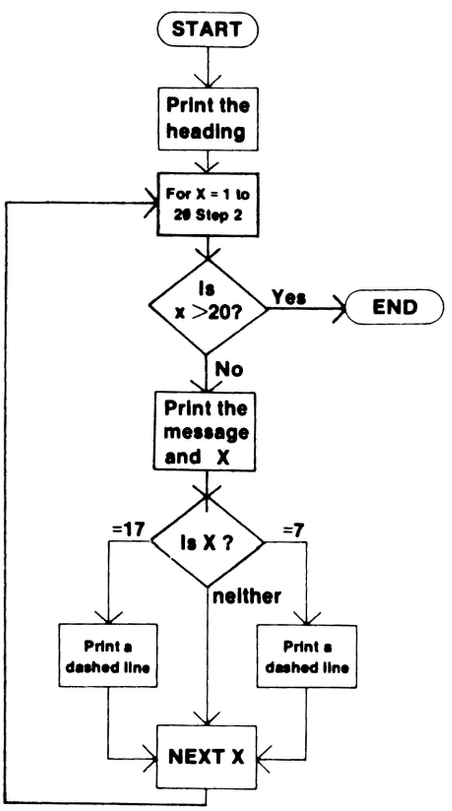
2. Use the **FOR...NEXT** command and write a program to loop through the numbers of 1 to 20 with a **STEP** of 2 and print the odd numbers. Print the output in the same format as on the

following page. After the odd numbers 7 and 17 notice the dashed line. Write the commands to also print these lines. (HINT: Use **IF...THEN** statements and print 10 dashes together.)

Example Output:

```
ODD NUMBERS 1-20
----------------
NUMBER 1
NUMBER 3
NUMBER 5
NUMBER 7
----------------
NUMBER 9
NUMBER 11
NUMBER 13
NUMBER 15
NUMBER 17
----------------
NUMBER 19
```

3. To help you write your program, look at the following flowchart. It shows the order in which the computer should read and perform your program directions.



Write your program here:

4. Now, run the program. If the program doesn't execute, then list it on the video display. Check for any mistakes and correct them by typing the line number and the new statement. Keep trying until you are successful, or ask for help if you cannot find the mistake or error.

Activity 9. Aligning Numbers in Columns*

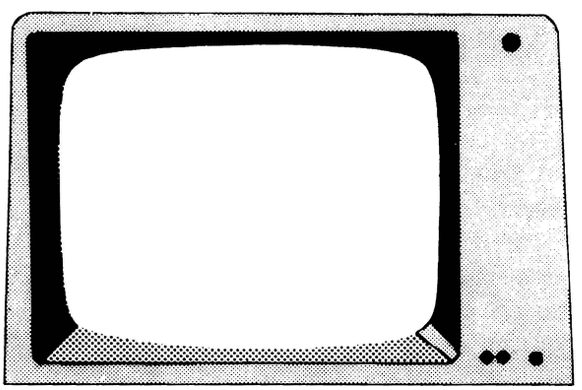
You will learn how to align numbers and data on the video using the format symbols ##### and to print in columns using the same format symbols. Numbers often need to be aligned by the left-most digit, the right-most digit, or by a decimal. This exercise, which works only on some computers, shows you how to align data and decimal numbers easily and properly.

1. Clear the computer for a new program.

2. Locate the following instructions on the table. Then write them in the appropriate commands for your computer in the spaces to the right of the instructions.

Instructions	Write Your Program
10 Clear the screen	_____
20 Print to video the number 6 on one line Print to video the number 116 on a second line Print to video the number 43 on a third line Print to video four dash marks on a fourth line Print to video the sum of 6+116+43 on a fifth line	_____
29 Print to video two blank lines	_____
30 End of program	_____
Run the program	_____

3. Write in the box below how the numbers appeared on the video. (Be careful to line them up the way that they appear on the screen.)

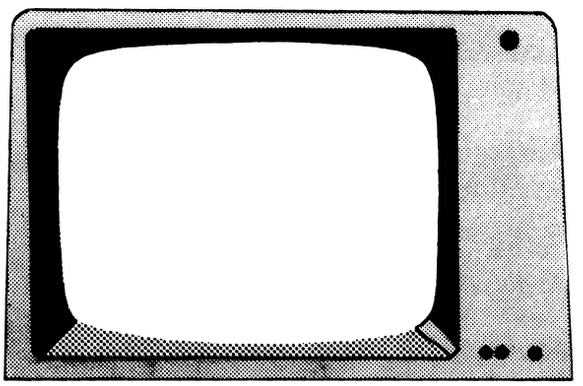


*The commands in Activity 9 are not usable on all microcomputers.

4. List the program. Type line 20 as follows:

```
20 PRINT USING "###";6:PRINT USING "###";116:PRINT USING "###";43:
PRINT"----":PRINT USING "###";6 + 116 + 43
```

(Double-check your typing before pressing **ENTER/RETURN**.) Run the program and write in the box how the numbers appeared.



What did the command **USING "###";** do? _____

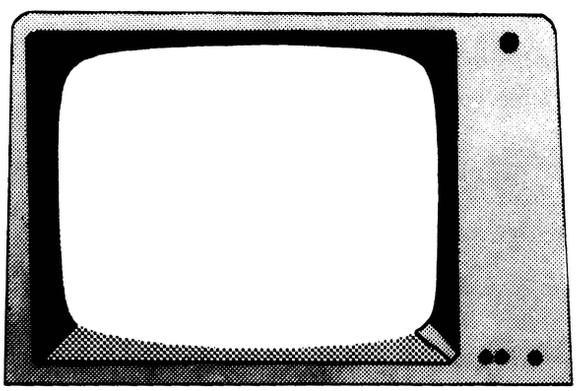
What is the difference between the video displays in paragraphs 3 and 4? _____

_____

5. List the program. Retype line 20 and add lines 22 and 24 as follows:

```
20 PRINT USING "###.##";5.14
22 PRINT USING "$###.##";5.14
24 PRINT USING "$$##.##";5.14
```

Run the program and write in the box how these numbers appeared.



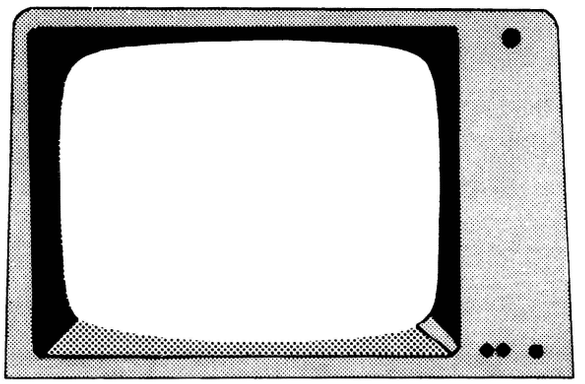
What was different about these commands? _____

_____

6. List the program. Retype lines 20, 22 and 24 as follows:

```
20 PRINT USING "###,###.##";1234.56
22 PRINT TAB(7) USING "###,###.##";1234.56
24 PRINT USING "I OWE JOE $$##.##";5
```

Run the program and write in the box below how these numbers and words appeared.



Let's Review

1. To test your knowledge answer the following questions by checking true or false.

The command **PRINT 38** aligns the numbers by a left-hand column: T___ F___

The command **PRINT USING "###";38** aligns the number by a right-hand column: T___ F___

The command **PRINT USING "$###.##";1.51** puts a $ sign beside the number (no blank spaces): T___ F___

The command **PRINT USING "$$###.##";1.51** puts a $ sign beside the number (no blank spaces): T___ F___

2. Write a statement with the **PRINT, TAB and USING** commands to properly show the following information on the video display, beginning in column 5: **MY BANK ACCOUNT IS $1,234.56**

Important to Remember

- Use a # symbol for each digit in your number, plus one extra for a minus sign if one should occur.
- Use a . to show where you want the decimal to be.
- Use two $ symbols to put the $ sign beside the number (no blank spaces).
- Place a comma properly in the # symbols to produce a comma in the printed number.
- Use TAB(n) command to place numbers or words in a desired column.
- Words as well as numbers can be included in the USING command by putting the words within the quotations.

Answers: 1. T
T
F
T

Activity 10. How Big Is Big?

You will learn the meaning of these words: size of numbers, integer, single precision and double precision variables. You will also learn how to specify the type variable for small and large numbers to be accurate.

1. Clear the computer for a new program.

2. Locate the following instructions on the table. Then write them in the appropriate commands for your computer in the spaces to the right of the instructions.

Instructions	Write Your Program
10 Clear the screen	
20 Print to video the message "AN INTEGER IS A WHOLE NUMBER WITHOUT A DECIMAL."	
25 Print to video the message "AN INTEGER NUMBER CANNOT BE LARGER THAN 32,767 ON SOME MACHINES."	
30 Print to video the message "A SINGLE PRECISION NUMBER HAS A DECIMAL AND UP TO 6 ACCURATE DIGITS."	
35 Print to video the message "A DOUBLE PRECISION NUMBER HAS A DECIMAL AND UP TO 16 ACCURATE DIGITS."	
40 Print to the video the message "THE FOLLOWING ARE EXAMPLES FOR C=4/3"	
45 Set a variable for numbers equal to 4	
46 Set an additional variable for numbers equal to 3	
47 Set an integer variable equal to the first variable for the number divided by the additional variable for numbers	
48 Set a single precision variable equal to the first variable for numbers divided by the additional variable for numbers	
49 Set a double precision variable equal to the first variable for numbers divided by the additional variable for numbers	

*The material in Activity 10 is not usable on all machines. If the material is not applicable to your machine then read over the material. You will learn some important concepts about the size of numbers.

50 Print to video "INTEGER ="; and the integer variable _____

55 Print to video "SINGLE PRECISION ="; and the single precision variable _____

60 Print to video "DOUBLE PRECISION ="; and the double precision variable _____

65 End of program _____

Run the program _____

If you do homework and divide 4 by 3, what result do you get? _____

What result did the computer give?

INTEGER = _____

SINGLE PRECISION = _____

DOUBLE PRECISION = _____

Which result is most accurate? _____

Which result is least accurate? _____

3. The symbol % after a variable tells the computer to make the number an INTEGER. In the same way, the symbol ! means SINGLE PRECISION and # means DOUBLE PRECISION. Engineers designing the machine make these decisions.

Use the symbols %, ! or # to tell what type number each of the following should be for proper accuracy.

NUMBER	**SYMBOL**
1. 1Ø46	_____
2. 1.347	_____
3. -5	_____
4. 1,632.4478	_____
5. Ø.44789	_____
6. -1.345	_____

Let's Review

Stop for a moment and answer these questions.

1. If you need a dollar sign next to numbers (for example, $12.98), how would you write the computer command? _____

_____

2. How would you ask the computer to do the following—print your name 10 times and after five names print a line using the IF...THEN command?

_____

_____

If you have trouble with these questions, refer to the first 10 activities. Do not go on to the third project until you know and understand the answers to these questions.

Answers to number 3:

1. %
2. !
3. %
4. #
5. !
6. !

Activity 11. DATA Statements and READ Commands

You will learn how to use DATA statements and READ commands for storing and retrieving data or information from within a program. The DATA statements allow you to define and store numbers, words, names, etc. in a program. The READ commands enable you to retrieve and use the stored data in the program.

1. The following program is an example of how the **READ** statement acquires information from the **DATA** statement. If a microcomputer is available, type the program into the microcomputer and run it. Observe what is displayed.

```
 5 CLS (or HOME)
10 READ A,B,C
20 DATA 1,2,3
30 PRINT A
40 PRINT B
50 PRINT C
60 END
```

The value of **A** is 1, **B** is 2, and **C** is 3. The **READ** statement obtains and assigns the values one by one from first to last in the **DATA** statement. The comma separates the variables in the **READ** statement and separates the data elements in the **DATA** statement.

2. The following example uses words (or characters) instead of numbers. Notice the difference in the variables used. (Remember the difference from Activity 5.) Type and run this program and observe the results.

```
 5 CLS
10 READ A$,B$,C$
20 DATA "HI", "HELLO", "GREETINGS"
30 PRINT A$
40 PRINT B$
50 PRINT C$
60 END
```

The variables A$, B$ and C$ are assigned values in the same manner as the numbers in the above example.

3. The next example of **DATA** and **READ** statements uses only one variable to read all three numbers. Type this program and run it.

```
 5 CLS
10 FOR I = 1 TO 3
20 READ A
30 PRINT A
40 NEXT I
50 DATA 10,20,30
60 END
```

Notice that each time a **READ A** statement is executed, it reads the next value in the **DATA** statement. The variable **A** is assigned a new value each time through the **FOR...NEXT** loop. Using words or characters will work the same way if the proper variable type is used.

4. Remember that the number of times the **READ** statement is executed cannot exceed the number of items in the **DATA** statement or an error will occur. There must be one **DATA** value for each **READ** variable, or each time the **READ** command seeks a **DATA** value.

DATA statements can be anywhere in a program. If you plan to add data to a program over time, then put the **DATA** statements at the end of the program so line numbers can be continued consecutively. The above examples can assist you in developing future programs in Project III. Be sure you understand them before proceeding.

Activity 12. Arrays

You will learn about simple arrays and how to use arrays. Arrays provide a fast and organized way of handling large amounts of data.

1. Arrays are valuable techniques in computer programming. They give the programmer the ability to assign many values to the same variable name. Arrays can be defined in different dimensions, but in this project you will work with the one-dimensional array. There are two types of arrays: numeric and character.

Example of a numeric array of eight elements:

1 4 7 8 12 11 6 50

Example of a character array of eight elements:

"CAT" "DOG" "SHEEP" "HOUSE" "DOCTOR" "DENTIST" "POLICEMEN" "HILL"

2. The numeric and character arrays are assigned in the following format in the **BASIC** language.

Numeric Array	Character Array
A(1) = 1	A$(1) = "CAT"
A(2) = 4	A$(2) = "DOG"
A(3) = 7	A$(3) = "SHEEP"
A(4) = 8	A$(4) = "HOUSE"
A(5) = 12	A$(5) = "DOCTOR"
A(6) = 11	A$(6) = "DENTIST"
A(7) = 6	A$(7) = "POLICEMAN"
A(8) = 50	A$(8) = "HILL"

Let's examine the meaning of the array **A(2) = 4.** The "**A**" is a numeric variable for the numeric array. The "**(2)**" is the position in the array in which you are going to assign a value.

Think of this process as the second "mailbox" in a row and you want to place a value of 4 in the mailbox.

3. Notice the rest of the eight elements. The assigning of a character array is basically the same except for the $ following the variable. There is a special dimension **(DIM)** statement in **BASIC** required to define an array. This **DIM** statement allows the creation of arrays. For example, if you want a numeric array of eight elements with the variable name "**A,**" place this **DIM** statement at the beginning of the program.

```
10 DIM A(8)
```

4. The following program will assign all the values of the numeric array listed previously. The character array would be represented in a similar manner. Type and **RUN** this program and observe the results.

```
 5 CLS
10 DIM A(8)
20 A(1) = 1
30 A(2) = 4
40 A(3) = 7
50 A(4) = 8
60 A(5) = 12
70 A(6) = 11
80 A(7) = 6
90 A(8) = 50
100 FOR I = 1 TO 8
110 PRINT A(I)
120 NEXT I
130 END
```

Notice the loop beginning at line 100. This loop is used to print the values of the array. The "**A(I)**" allows you to print all eight values in a loop by simply changing the value of "**I**" each time. You can input the values of all the positions in the array in a similar procedure.

5. The **DATA** and **READ** statement can be used to load information into an array. After studying the previous **DATA** and **READ** statement section, look at the example below. Type and run the program and observe the results.

```
 5 CLS
10 DIM A(8)
20 FOR I = 1 TO 8
30 READ A(I)
35 PRINT A(I)
40 NEXT I
50 DATA 2,4,7,8,12,11,6,50
60 END
```

This program will allow you to enter the contents of the **DATA** statement into array "**A.**" Using this concept allows you to enter large amounts of data into arrays—whether it is character or numeric data.

Completing and understanding this exercise on arrays is valuable in helping you complete the activities in this project. There is a more advanced way to enter large amounts of data; that is, through the use of files. While this method is not used in this project, you can learn more about files using the materials on the **BASIC** language for your computer.

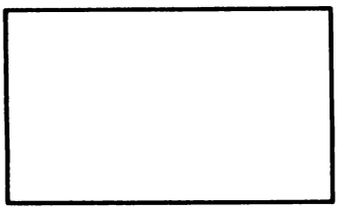
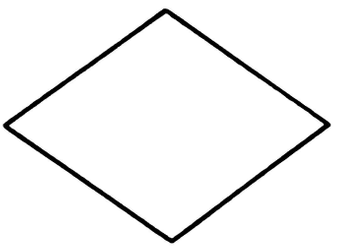
For More Information

- Check your local library for books on microcomputers.
- Go to the nearest retail store that sells microcomputers and ask to see a demonstration of the equipment.
- Ask the banks, travel agents and car dealers in your community if their records are computerized and how they use computers to do work.

Demonstrations and Illustrated Talks

Giving a 4-H demonstration can be a good learning experience for you and for others. Work with your leader in deciding on a good topic. You may want to:

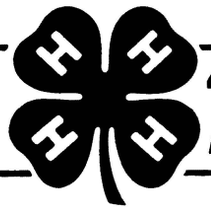
- Explain some of the command words you have learned and show how they make the computer do what you want it to do.
- Demonstrate "how to program" a simple program on the computer.
- Make the 4-H pledge appear in the center of the screen.
- Use your imagination and come up with some good ideas!

Flowchart Symbols		Meaning
Ovals		A START or STOP point in a flowchart
Square/rectangles		Indicates statements that tell what to do
Diamonds		Indicates decisions (Yes/No questions) or where choices are needed

INSTRUCTIONS	COMMAND KEYS AND STATEMENTS FOR VARIOUS MICROCOMPUTERS	YOUR COMPUTER
1. Clear the memory	**CLEAR** or **ESC & @** or **CTRL & HOME**	
2. Enter/Return key	**ENTER** or **RETURN**	
3. Stop execution key	**BREAK** or **CTRL & C** or **CTRL & SCROLL/LOCK**	
4. New program command	**NEW**	
5. Clear the screen	**CLS** or **HOME**	
6. Print to video any words	**PRINT"any words"**	
7. List the commands of a program	**LIST**	
8. Run the program	**RUN**	
9. End the program	**END**	
10. A variable for numbers	**X, Y, AX, A3** or 1 or 2 letters or letter and number pairs	
11. Print to video a variable value for numbers	**PRINT X**	
12. Print to video some words and a variable value for numbers	**PRINT"any words";X**	
13. Print to video two or more variable values for numbers	**PRINT X;Y;AX**	
14. A variable for characters	**A$, B$, AX$, A3$** or any 1 or 2 letters or letter and number pair with $ symbol	
15. Print to video a variable value for characters	**PRINT A$**	
16. Print to video some words and a variable value for characters	**PRINT"any words";A$**	
17. Print to video two or more variables for characters	**PRINT A$;B$**	

18.	Print to video a blank line	**PRINT**
19.	Go to a line number	**GOTO line number**
20.	Print beginning in a specific column	**PRINT TAB(any column)** within screen width
21.	Input data (numbers) using a variable for numbers	**INPUT X**
22.	Input characters using a variable for characters	**INPUT A$**
23.	Input data using variables for two or more number or characters sets	**INPUT X;A$**
24.	Print to video a word prompt and input numbers and characters into a variable	**INPUT"anywords";X,A$**
25.	IF an expression is true THEN perform certain commands	**IF expression THEN command**
26.	Use a variable for numbers and index from a low number to a higher number	**FOR X = low no. TO high no.**
27.	Index to the next variable number	**NEXT X**
28.	Index in STEP increments	**FOR X = low no. TO high no. STEP no.**
29.	Print using a columnar alignment for numbers	**PRINT USING"###";X**
30.	Print using a dollar sign before numbers	**PRINT USING "$###.##";X**
31.	An integer number variable	**C%, CX%, C3%** or any 1 or 2 letters or letter and number pair with the % symbol
32.	A single precision number variable	**C!, CX!, C3!,** " "
33.	A double precision number variable	**C#, CX#, C3#,** " "
34.	Read data or character values from within the program	**READ variable, variable,** etc.
35.	Data statements containing numeric or character values to be read into variables (Note: Use quotes around any character value containing imbedded commas.)	**DATA value, value,** etc.
36.	Screen width of your computer (32, 40, 64, or 80 characters)	

Note: There are many other commands available in the BASIC microcomputer language. Refer to your computer manual or other books to learn these valuable commands.



4-H Computer Project II:
Learning About Programming

PROJECT RECORD FORM

Name _____ **School** _____

County _____ **Birth Date** _____

Name of 4-H Club/Group _____ **Today's Date** _____

A. Tell what you learned in this project (for example, learned how to operate the keyboard of a microcomputer).

_____

_____

_____

_____

_____

_____

B. List any activity related to this project in which you participated, such as group meetings, tours, exhibits, demonstrations.

_____

_____

_____

_____

_____

C. List any awards or recognition you have received in this project.

_____

_____

_____

_____

_____

D. If you helped others with their computer project, give the number of people you helped and what you did to help them.

_____

_____

_____

_____

_____

E. Write a project story telling what you did and learned including how the project helped you, who helped you with the project, and why computers are important to us.

APPROVED: _____ _____

Project Leader *Parent*

_____

County Extension Agent