DOCUMENT RESUME

ED 274 530 SE 047 225

AUTHOR	Boe, Thomas; And Others
TITLE	Microcomputers: A Course for Parents and Children. For Use with Personal Computers. No. 657.
INSTITUTION	Minnesota Educational Computing Consortium, St. Paul.; Minnesota Univ., Minneapolis.
SPONS AGENCY	National Science Foundation, Washington, D.C.
PUB DATE	1 Sep 83
GRANT	DISE-07872
NOTE	85p.; For related documents, see SE 047 223-224 and SE 047 226-228.
PUB TYPE	Guides - Non-Classroom Use (055) -- Guides - Classroom Use - Guides (For Teachers) (052)
EDRS PRICE	MF01/PC04 Plus Postage.
DESCRIPTORS	*Computer Literacy; Courseware; Elementary Education; *Elementary School Science; Instructional Materials; Intermediate Grades; *Microcomputers; *Parent Child Relationship; Parent Materials; Parent Participation; *Science Activities; Science Education; *Science Instruction; Science Materials; Teaching Guides
IDENTIFIERS	Informal Education; *Parent Child Program

ABSTRACT

This booklet is designed to supplement a short course on microcomputers for parents and their middle school children. The course activities are designed to increase the computer literacy of the participants and also to provide an opportunity for parents and children to spend time with each other. The course is intended to take place during a five-week period, with one session taught each week, preferably on Saturday mornings. Using a predict-and-discover approach, the sessions start with simple BASIC commands and move into such programming concepts as input, output, looping, and branching. Instruction is also provided on computer uses, peripherals, and software. The materials in the document include objectives, materials lists, and a description of the teaching activities for each of the five sessions. Also included are reproducible handout sheets which are to be used during the sessions. The appendices contain: (1) a final test (with the answer key); (2) a graduation certificate; (3) a glossary of microcomputer terms; (4) reference sheets for Apple and Atari computers; and (5) a list of services offered through the Minnesota Educational Computing Consortium. (TW)

This book was developed by Thomas Boe and Kent Kehrberg, Minnesota Educational Computing Consortium, and Camille Wainright, University of Minnesota. The course described here was developed as a part of the Out-of-School Science Experiences project directed by Dr. Eugene Gennaro and Dr. Patricia Heller (Associate Director), both from the University of Minnesota. The project received its funds from the National Science Foundation (DISE #07872).

The staff wishes to thank Dr. Roger Olstad, at the University of Washington, Dr. Michael Padilla, University of Georgia, and Drs. Kenneth Kelser and John Coulter, St. Cloud State University, for managing the testing of the course materials with parents and children in their parts of the country.

September 1, 1983

TABLE OF CONTENTS

INTRODUCTION

Background

This book contains the training materials for a course on microcomputers for parents and their sixth, seventh or eighth grade children. The course activities are designed to increase the computer literacy of the participants and also to provide an opportunity for parents and children to spend time with each other. Activities and experiences in the class are designed for people without previous computer background or programming skills.

Hardware Requirements

The course is designed to be usable with a variety of popular microcomputers, including Apple, Atari, Radio Shack, and Commodore models. In order to provide this flexibility, reference to special features unique to each of these machines has been omitted. Special key functions, graphics, and sound are not included in the activities. Reference sheets to guide the class participants in the basic use of Atari and Apple computers are included as appendices. Only a microcomputer and a display screen are needed. For use with very low-cost systems, program storage on disk drives or cassette recorders is not required.

Scheduling

This course is designed to be taught in five sessions of three hours each. Saturday mornings and weekday evenings are possible class session times. Providing a fifteen-minute break with refreshments midway through each session is a feature that most participants enjoy. Families may take turns bringing snacks, cups, coffee, fruit drinks, etc. The break gives participants a chance to become acquainted and to talk to the instructor about specific problems or concerns. Emphasize the importance of keeping the refreshments away from the area of the microcomputers, preferably in a separate room.

Participants

The maximum class size will be determined primarily by the number of microcomputers available. Each family group should have its own microcomputer for use throughout each session. While most families have one parent and one child participating, sets of three are also workable. A class size of 20-30 is reasonable, depending on the facilities.

Course Structure

The course is designed to provide maximum hands-on microcomputer experience. The worksheets have been created to minimize the need for introductory demonstrations of programming statements and structures.

In most cases a "discovery" approach is taken; that is, the statement is first introduced to the participants as a mystery. When appropriate, these introductory exercises require the participants to predict the result of a program or command before typing it in or RUNning it. This approach emphasizes the fact that commands in the BASIC language are composed of English language words, arranged logically (for the most part) into understandable "sentences". Often the participants' predictions will be correct, leading to a sense of genuine understanding and confidence. Even when incorrect, the act of predicting leads to an increased interest in the actual result.

Because the activities are designed to lead the students through a discovery of the characteristics of BASIC programming, preliminary comments by the teacher should be kept to a minimum. The emphasis, rather, should be on the follow-up of each activity. Prior to continuing with the next worksheet, share individuals' discoveries and clarify difficult points.

A more traditional presentation/discussion approach is utilized in non-programming portions of the course (topics such as: parts of the computer, computer functions, and uses of specific consumer software). Participation in these topics is promoted through a variety of methods, including collecting and sharing newspaper articles, visiting computer stores, and role-playing the parts of a computer. For the presentation periods, it is most productive to meet in an area separate from the computers, in order to focus everyone's attention on the topic.

Session 1

OBJECTIVES

The student should be able to:

- execute the BASIC commands PRINT, GOTO and END;
- use the system commands RUN, LIST, and NEW;
- identify the order of mathematical operations that apply in BASIC;
- combine numbered BASIC statements to create programs;
- create and operate programs that demonstrate looping and branching using GOTO.

MATERIALS

Classroom sets of Session 1 worksheets

TEACHING SUGGESTIONS

Getting Started (20 to 45 minutes)

- Purpose:

To introduce the class participants to each other and to introduce the course.

- Activity:

Begin the class with a description of the course goals, course structure, duration of classes, highlights of coming sessions, etc. Ask each family member to introduce his or her partner(s) in the course. Children introduce parents and vice versa. Besides giving his or her name, each should give one point of interest about the other.

In case testing is desired, a test form is included as an Appendix. This test may be used as a pre-test and then reformatted if a post-test is to be administered. Allow 25 to 30 minutes for testing.

Activity 1: Printing Out Stuff (30 minutes)

● Purpose:

To introduce the PRINT statement and some of its various uses. The PRINT statement is introduced in command mode, rather than from within a program. This allows the emphasis to be placed on the use of the statement and its effects without involving the separate concepts of a program.

● Preparation (10 minutes):

Discuss with the class how computers and people can communicate or interact with each other; animation and sounds in game programs are good examples of output. Input is the other important interaction that can occur between people and computers.

In this class, the students will learn how they can "talk" with the computer. The first thing they will learn to do is to make the computer "talk" to them.

Tell them that they will be typing BASIC statements into the computer that will use the computer's capability to print things on the screen. Their job is to determine the outcome and "rules" associated with these commands by using them.

Briefly describe how to turn on the computer and type in a command from the keyboard. Each different computer has special keys or functions that aren't standard. Some machines have a delete key for erasing errors; others have a backwards arrow for the same purpose. You should try to provide some type of reference sheet that can be used for your particular computer. Samples of such sheets developed by the Minnesota Educational Computing Consortium (MECC) for the APPLE II and ATARI computer systems are included in the Appendix section of this manual.

● Handout 1a (15 minutes):

During this and following activities, mingle with the class participants, acting mainly as a resource person for them. In this first activity they may have difficulty with:

a) finding special keys (* + / ") on the keyboard;

b) pressing RETURN or ENTER at the end of each command line;

c) typing in the word PRINT. (They may understand this as an instruction for them to type the information into the computer.)

The last activity on this first handout asks them to type their names into a PRINT statement without quotes. This will produce a 0 and perhaps a SYNTAX ERROR (depending on the computer model). You may explain this by saying that the computer interpreted the name as a number since it didn't have quotes around it. The numeric value of JANE SMITH is 0! Sorry, Jane.

• Follow-up (5 minutes):

Be certain that the participants understand the difference between the effects of a PRINT statement with and without quotation marks. Have them predict the resulting output of:

```
     PRINT "3 + 4 * 5"
and
     PRINT 3 + 4 * 5
```

This will also lead into the next activity effectively.

<u>Activity 2: Order Of Operations (30 minutes)</u>

• Purpose:

The computer performs mathematical operations in the same order as is generally employed in mathematics. This order, and the use of parentheses in complex mathematical operations, may be new to the students. Laying out these mathematical rules is important in order to make the computer's operations in future activities as understandable as possible.

• Preparation (5 minutes):

The follow-up from Activity 1 can provide the impetus for studying this topic. You may wish to predict the outcome of the first problem on the activity sheet and test it out as a group. Some will predict that the second statement will produce the value 35 (7 multiplied by 5). The actual output is 23 (3 plus 20).

• Handout 1b (20 minutes):

<u>Do not</u> tell the participants the rules of mathematical operations order during or preceding this activity. Have them converse among themselves and come to their own conclusions.

• Follow-up (5 minutes):

You may wish to translate a problem such as 20/(2 + 3) to a form like:

$$\frac{20}{2 + 3} = \frac{20}{5}$$

Record the class results for each of the statements on a blackboard or overhead transparency. Go over any of the statements which were not predicted accurately.

Activity 3: Writing and Changing Programs (40 minutes)

- Purpose:

The intent of this portion of the class is to introduce the concept of a program. In addition, the system commands RUN, LIST, and NEW are introduced. Program editing and the use of the semicolon in a PRINT statement are also developed in this activity.

- Preparation (5 minutes):

Play the "Devil's Advocate" in order to introduce the concept of a program. Ask why anyone would spend a lot of money to buy a computer when a $10 calculator could do the same calculations without even having to type PRINT. One thing a computer can do that a simple calculator can't do is to perform a group of commands in some specific order. Then each time you want that set of commands to be performed all you have to do is type RUN. Such a group of commands is called a program.

- Handout 1c (30 minutes):

Little teacher intervention is needed. One common pitfall is that some participants do not type in the line numbers along with the PRINT statements. Others may need help in figuring out what NEW and LIST have done.

- Follow-up (5 minutes):

Some of the things which probably need summing up or clarification include the use of semicolon (step 7), multiple PRINT functions on a line (steps 9 & 10), and spacing when putting PRINTs together (step 10). Many of the participants will ignore the spaces in the quotations in step 10. They will have a result like this:

```
MY COMPUTER IS6YEARS OLD.
```

You may need to emphasize that the computer doesn't "see" or PRINT out any spaces unless they are inside quotes.

Activity 4: Telling The Computer Where To GOTO (35 minutes)

• Purpose:

To introduce the programming statement GOTO and the concepts of looping and branching.

• Preparation:

So far the full capabilities of a program haven't been demonstrated. With the introduction of the GOTO statement, the power of a short program becomes obvious. This discovery has more effect if the results are unexpected. To that end the only preparation for this activity is to say that the students are about to investigate a new programming statement called GOTO. It is up to the students to discover what this command does in a program.

• Handout 1d (30 minutes):

You will probably have to give some direct hints as to how students might get their optional rocket to fly. One way is to type something on the screen and show what happens to it as you press the RETURN or ENTER key repeatedly below it. A "blank" PRINT line does the same thing as hitting the RETURN or ENTER key. How could you easily do a lot of "blank" PRINT statements? (GOTO).

• Follow-up (5 minutes):

Review the operation of the name-printing program, especially the process of adding a semicolon to the end in order to print the name over and over on the same line. Did individuals put spaces before or after their names so that the names wouldn't "run together"? Introduce the term loop for doing something over and over again.

Trace the steps of the third program (Handout 1d, part 3). Introduce the term branch. Point out that indescriminate use of GOTO can make programs hard to understand and hard to debug. This kind of program is sometimes called "spaghetti programming" because the direction of the program flow becomes quite tangled.

Have someone who was able to make a rocket "take off" demonstrate how it was done.

Home Activity #1 (5 minutes)

• Preparation:

Explain the purpose of Handout 1f and the other home activities. Students need something to do to review and sharpen their skills between classes, particularly if the classes are held only once a week. The activities will generally be puzzles that take a little time to figure out and for which a computer isn't necessary.

If home activities include writing or completing a short program, time will be given at the beginning of the next class to type in the programs and run them. Emphasize the importance of parent-child cooperation on the home activities.

PRINTING OUT STUFF

The following exercises will help you to discover how a computer can communicate to people by printing out information on its screen. If you had a printer, the computer could just as easily print things out on paper for you to take home.

Once the computer is turned on, you can start typing commands to the computer. Press the RETURN (or ENTER) key of your computer at the end of each line!!!

Type: PRINT "COMPUTERS CAN WRITE WORDS."

Now write a PRINT statement in the space below that will print out your name. Try it out on the computer.

On the left below are a series of PRINT statements. Type them into the computer, then record in the space on the right what the computer typed back to you:

You type:	The computer types:
PRINT "3 + 4"	
PRINT 3 + 4	
PRINT 3 - 2	
PRINT "3 * 4"	
PRINT 3 * 4	
PRINT 28 / 4	

What happens when you PRINT something inside of quotation marks (")?

What happens if you type PRINT, followed by a math problem, without quotation marks?

PRINTING OUT STUFF (continued)

Predict what the computer would print out if you typed PRINT, followed by a word (or sentence) without quotation marks?

Try to print out your name without quotation marks. What happens?

What math operation does * do?

What math operation does / do?

Computers use some strange symbols for math operations because the regular math symbols might be confusing (either to the computer or the person using it). The usual multiplication sign might be confused with the letter x. If the screen is fuzzy, a division sign (line with dots above and below) might look like a plus (+).

1b

ORDER OF OPERATIONS

In complicated mathematical problems the computer has to know which operation to do first. It follows the same "order of operations" that is used in algebra.

If you type: PRINT 3 + 4 * 2, what do you think the answer will be?

Your prediction:

Try it.

The computer prints:

The computer usually does problems from the left to the right, but it considers some operations to be more "important" than other operations. The more important operations are done first. Here are the rules:

- Anything inside parentheses ( ) is done before things outside of parentheses.
- Multiplication (*) and division (/) are done before addition and subtraction.

In the examples below, <u>predict</u> what the computer will print out, then try it on the computer.

You type:	Your prediction:	Computer prints:
PRINT 8 * 3 + 2		
PRINT 2 + 8 * 3		
PRINT 2 + (8 * 3)		
PRINT (2 + 8) * 3		
PRINT 20 / 2 + 3		
PRINT 20 / (2 + 3)		
PRINT 8 * 6 / 3 * 4		
PRINT (8 * 6) / (3 * 4)		
PRINT 8 * (6 / 3) * 4		

1c

WRITING AND CHANGING PROGRAMS

1. Type:
```
NEW
10 PRINT "I AM A COMPUTER"
20 PRINT "I CAN ADD 2 + 2"
30 PRINT "THE ANSWER IS"
40 PRINT 2 + 2

RUN
```
Did your program work the way you thought it would? ____________

2. Type:
```
LIST

15 PRINT "WATCH!"

LIST
```
What happened to the line: 15 PRINT "WATCH!"? ____________

3. Type:
```
RUN

15 PRINT "WATCH THIS NOW!"

LIST
```
What happened to line 15? ______________________________

4. Type:
```
RUN

15

LIST

RUN
```
What happened to line 15? ______________________________

Type:
```
15 PRINT

RUN
```
What does a "blank" PRINT statement do? ____________________

WRITING AND CHANGING PROGRAMS (continued)

5. Now try a few experiments of your own. Add some of your own lines. Replace some of the lines. Use LIST and RUN to check the results.

   What did LIST do to your program? ____________________

   RUN? ____________________

6. Type:
```
NEW
LIST
RUN
```

   What does NEW do to your program? ____________________

7. Now that your last program has "disappeared," you can start another one. Type in each of the program lines below and record in the space on the right what the computer prints out when you RUN the program.

   Type:
```
10 PRINT "THIS IS"
20 PRINT "ON THE SAME LINE"
RUN
```

   What did the computer print? ____________________

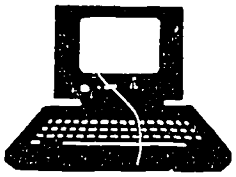
   Type:
```
10 PRINT "THIS IS";
RUN
```

   What did the semicolon do? ____________________

   The words ran together once the two commands were printed on the same line because there was no space after "IS". Retype line 10 with a space at the end of IS inside the quotes like this:

```
10 PRINT "THIS IS ";
RUN
```

   What happened? __________ ____________________



WRITING AND CHANGING PROGRAMS (continued)

8. Here's another program for you to change. This time you have to figure out what to do.

Type: NEW
```
10 PRINT "8 * 5 = "
20 PRINT 8 * 5

RUN
```

Change line 10 of this program so that the problem and its answer are written all on one line. Write your new line 10 below:

10 ____

Type: 10 (as you wrote it)

```
RUN
```

Did it work the way you wanted it to? ____________________

9. Below is another way to do the same thing with fewer lines. One print statement can print several different things on one line. Type in this one-line program and RUN it.

Type: NEW
```
10 PRINT "8 * 5 = "; 8 * 5

RUN
```

10. Here's an even more complicated example. Don't forget to put spaces inside the quotes so that the different parts of the PRINT statement don't run together.

Type: NEW

```
10 PRINT "MY COMPUTER IS "; 2*3 ;" YEARS OLD."

RUN
```

TELLING THE COMPUTER WHERE TO GOTO

1. Once upon a time some teachers made students write things over and over again as punishment for doing something wrong. In the past few years students have found such a punishment pretty easy to do with a computer. Type in the following program but don't RUN it yet!

```
NEW
10 PRINT "I WON'T SHOOT SPIT-WADS IN CLASS."
20 GOTO 10
```

What do you suppose the computer is going to do when it gets to line 20?

____

Test out your theory by RUNning the program.

GOTO 10 means: ______________________________

To stop the program...your computer has a key called RESET or BREAK or STOP. Press this key.

2. Write and RUN a NEW two line program that will write your name over and over again. Write the program steps below:

10 ______________________________

20 ______________________________

Writing your name in a column is fun, but why not write it all over the screen? PRINT your name, then keep PRINTing right after it on the same line. (A punctuation mark you know will do this, remember?) Rewrite line 10 below so that it will do this, then type it into the computer and RUN it.

10 ______________________________

Did it work? __________

If you didn't type it so that there would be a space at the end of your name, then it probably all ran together. If so, type line 10 so that your name would be separated by spaces, then RUN it again.

TELLING THE COMPUTER WHERE TO GOTO (continued)

3. Type in the program below and predict what it will say before you RUN it. Run it to check your prediction. Don't forget to type NEW.

Program:

```
10 GOTO 60
20 PRINT "PEOPLE ";
30 END
40 PRINT "LIKE ";
50 GOTO 20
60 PRINT "COMPUTERS ";
70 GOTO 40
```

Predicted result:

Actual result:

What did END do? ______________________________

Write a one line program that is simpler and more understandable than this one and does the same thing:

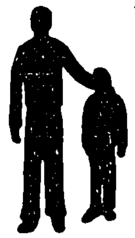
____

Do you think that this was a good use of GOTOs in a program? ________

Why? ______________________________

4. (Optional, for those who have the time and want a thrill)

Use PRINT statements and various combinations of letters on the keyboard such as X, I, V, etc. to design a simple rocket ship. Have your program "fly" your rocket up the screen. This can be done by repeatedly printing blank lines beneath your rocket picture.



SESSION 1 SUMMARY

Variations on PRINT

PRINT	causes a blank line to be printed
PRINT "HELLO"	causes the word HELLO to be printed
PRINT 7	causes the number 7 to be printed
PRINT "7"	causes the character 7 to be printed
PRINT 5 * 8	causes the number 40 to be printed
PRINT "5 * 8"	causes 5 * 8 to be printed
PRINT "5 + 7 = " 5+7	causes 5 + 7 = 12 to be printed
PRINT "TYPE YOUR NAME ";	using a semicolon (;) at the end of a PRINT statement prevents the computer from automatically going to the next line on the screen.

Other BASIC commands

GOTO 20	causes line 20 of the program to be executed next
END	stops the program

Computer system commands

NEW	erases any program from the computer memory.
RUN	makes the program in memory work or "go".
LIST	gives a list of all the commands of the program in memory.

Math symbols

+ Addition   - Subtraction   * Multiplication   / Division

Order of operations

1. Insides of parentheses ( ) are evaluated first
2. * or / next
3. Finally + or -.

HOME ACTIVITY #1

1. Read the "spaghetti" computer program printed out below. On the right-hand side write out the output as you predict it would appear on the computer's screen.

Program   Output

```
 10 GOTO 50
 20 PRINT "NEVER"
 30 PRINT "MAKE"
 40 GOTO 180
 42 PRINT "MISTAKES"
 45 PRINT "COOL"
 50 GOTO 80
 80 PRINT "COMPUTERS"
 90 GOTO 210
120 PRINT "FUN"
150 GOTO 40
160 PRINT "EVER"
180 GOTO 220
210 GOTO 150
220 PRINT "ARE"
230 GOTO 45
```

2. Bring to class next week a cartoon or an article from a newspaper or magazine about computers. If you bring in an article, be prepared to summarize it briefly for the class.

Session 2

OBJECTIVES

The student should be able to:

- use numeric and string variable names in a BASIC program;
- use both LET and INPUT for assigning variable values.

MATERIALS

Classroom sets of Session 2 worksheets.

TEACHING SUGGESTIONS

Home activity follow-up (10 minutes)

1. Before the class begins, load up (or type in) the program from the Home Activity #1. Have the class members tell their predictions for the program output. Run the program to test out the predictions.

2. Have participants share the computer cartoons and articles they brought in. Post them if possible.

Activity 1: Number Variables (50 minutes)

- Purpose:

To introduce the concept of a variable and the two common ways that values are assigned to variable names in a BASIC program.

• Preparation (5 minutes):

Introduce variables by saying that computers should be able to do more than just "parrot back" words and do a few calculations. You should be able to "communicate" with the computer during a program. You should be able to tell it some information, like your name and age, which it can "remember" and use later in the program.

There are two kinds of information it can remember:

1. numbers
2. groups or "strings" of characters (any of the numbers, letters or symbols from the keyboard)

Start by discussing how the computer remembers numbers.

Discuss how variable names are like the names on boxes in which a value may be kept. Draw a set of boxes similar to those used in the first activity and put different numbers in the boxes. Use variable names for the boxes like A, QT, and L3, or whatever the computer you are using will allow. Some computers (ATARI, for instance) will allow unique variable names like SUM, NUMBER, and WINNER$ which are more readable than the normal two character BASIC standard. Also, be sure to include numbers such as 2.6 and -34 which are not positive integers.

Point out that in BASIC there are two possible ways for these numbers to be placed in the variable boxes:

1. The programmer decides what will be stored in a variable name.

2. The program user decides what will be stored in a variable name.

The participants will write programs that use both methods (LET and INPUT).

• Handout 2a (35 minutes):

As you mix with the participants, you should point out the difference between the LET statements of lines 10 and 20 and those of lines 30 and 40 which use formulas. The idea that the value on the right is being put into the label on the left is less obvious when both sides contain variable names.

• Follow-up (10 minutes):

Discuss the difference between the way LET and INPUT worked. The participants should see that INPUT allows more flexibility in the program. A program that will multiply any three numbers that are typed into the keyboard is more useful than one that can only multiply the three numbers that the programmer had put into the program.

Compare PRINT and INPUT. PRINT allows the computer to "talk" to you. INPUT allows you to "talk" to the computer.

Activity 2: Variables Don't Have To Be Numbers (45 minutes)

- Preparation (5 minutes):

Discuss the idea of storing names, words, and sentences in variable "boxes". Mention that the point of this activit/ is to see how and why these "non-numbers" are stored in variables. Tell the class that there are a few differences between numeric and string variables in the way they are named and stored. They should be able to pick out what those differences are by doing the activity.

The most difficult part of this activity seems to be the concept of a "prompt line" for an input. Explain that in part 2 of this activity, the computer will be required to print a request statement (or prompt) indicating the information desired (name and address, in this case). Point to part 3 of the Number Variables (previous activity) for an example of such a prompt.

- Handout 2b (35 minutes):

The participants may need help in part 2 with prompts. Another common error with beginning programmers that may be encountered at this point is the use of a statement like: INPUT NA$ = "JIM".

- Follow-up (5 minutes):

What are the differences between the way numeric and string variables are used? Have the class point them out:

1. String variable names follow the same rules as numeric variables except they must end with a $.

2. A string which is being stored must be enclosed in quotation marks in a LET statement, unlike a number. One reason for this is that blank spaces are important characters. Without the quotes it might be hard to see that you wanted a string to end with ten blank spaces, for example.

Activity 3: A Project To Test Your Skills

● Handout 2c (25 minutes):

The participants may need some additional guidance in how to get started on this project. Tell them that they should start with prompts like:

NAME AN ANIMAL: WALRUS

where WALRUS is an animal name that the user of the program might have typed in response to the prompt. A few may try to "build" the sentence by using THE FEROCIOUS as a prompt and having the user input something after that. The sentence should only be printed out after the computer "knows" the name of an animal, a number, a thing, and a color.

● Follow-up (10 minutes):

Discuss how you might create a program which would write form letters of the type sent out by various companies which appear to be personalized. An example such as the one below may be created prior to class and loaded in for demonstration purposes.

```
100   PRINT "LAST NAME ";
110   INPUT LN$
120   PRINT "TITLE ";
130   INPUT TL$
140   LET PD$ = "ZAP TOOTHPASTE"
200   PRINT "DEAR "; TL$;" "; LN$; ","
210   PRINT "YOU MAY HAVE HEARD ABOUT OUR PRODUCT,"
220   PRINT PD$;". WE'RE SO SURE THE WHOLE"
230   PRINT LN$; " FAMILY WILL LOVE ";PD$
240   PRINT "THAT WE'RE SENDING YOU A FREE SAMPLE"
250   PRINT "WITH THIS LETTER."
260   PRINT
270   PRINT "YOURS TRULY,"
280   PRINT "MARY SEARS,"
290   PRINT PD$;" REPRESENTATIVE"
```

Each time the program is RUN, you can change the last name (Green, Horowitz, etc.), the title (Mr., Ms., etc.). If you add a GOTO 100 for line 300, you have a real assembly line program. The product name PD$ was used as a variable just in case Mary Sears wanted to use the same basic program to sell another of her company's products at a later date. Change line 140 to:

```
140   LET PD$ = "BIF DOGFOOD"
```

Ask the class why PD$ wasn't set up as an INPUT in the first place instead of a LET. (That would have been a lot slower if Mary wanted to send 1000 Zap Toothpaste letters).

Activity 4: Using Numbers in Formulas (40 minutes)

● Preparation (5 minutes):

In most of the program examples up to this point, only simple numbers have been placed into variable "boxes". In most real-life computer applications the results of some mathematical calculations are stored in variables as well. Examples might include a calculation of the amount someone is charged for electricity, the sales tax on a sale, or the speed a rocket might be expected to go. The activity which follows provides practice in doing such calculations in a program.

● Handout 2d (30 minutes):

The participants may need a hint that the underlined formula on each of the two examples can simply be entered into the program using a LET statement (for example: LET M = F/5280).

● Follow-up (5 minutes):

Look at a student-created sample of each of the two assigned programs. It is a small step from M = F/5280 to

```
50      LET M = F/5280
```

but it may not have been an obvious step for the students. Point out that LET, just like PRINT, can calculate the value of a mathematical operation. PRINT prints the result, LET stores the result in a variable. It is also worth explaining that you cannot INPUT a formula like this into a numeric variable. Only LET works this way.

Home Exercise #2

Handout 2e includes a sample program to analyze and a program to create. Both exercises are designed to reinforce participants' knowledge of variables.

2a

NUMBER VARIABLES

1. Read but don't RUN the program lines below. In the boxes to the right of the program, write in what numbers are being stored in each variable name. Lines 30 and 40 are different. In LET statements, the value on the right of the equal sign is stored in the variable name on the left.

```
10 LET N = 18
20 LET B = 2
30 LET A = N + B
40 LET M = N * B
50 PRINT "THE NUMBERS ARE "; N ;" AND "; B
60 PRINT "THE NUMBERS ADDED TOGETHER EQUAL "; A
70 PRINT "MULTIPLIED TOGETHER EQUAL "; M
```

N	B	A	M

In the space below predict what will be printed out by this program.

Type in and RUN the program. Record below what the computer actually prints out.

2. Change the program so that N=10 and B=4. Record the values of the variables in the boxes to the right below. Predict in the space below what the computer will print out when you RUN the program.

N	B	A	M

RUN the program.
Does it work the way you thought it would? ________

MINNESOTA EDUCATIONAL COMPUTING CONSORTIUM

NUMBER VARIABLES (continued)

3. You may not wish to change the program every time you want to change the numbers to add and multiply. You will now learn how to change the program above to make it easier to use. Add the lines below to your program. Don't type NEW!

```
 5 PRINT "TYPE IN YOUR FIRST NUMBER"
10 INPUT N
15 PRINT "TYPE IN YOUR SECOND NUMBER"
20 INPUT B
```

Run the program. This time you get to choose the numbers as you use the program. The values of N and B weren't written into the program itself. You PUT the them IN (INPUT) as you ran the program. Run it a few more times using different numbers. Record in the table below the values of N and B that you INPUT as well as the values of A (added) and M (multiplied).

N	B	A	M

Which way was it easier to use the program, when the numbers were defined with a LET or when you typed them in with an INPUT?

____

Notice that lines 30 and 40 still use LET. You can't INPUT a formula like N*B. You can only INPUT a number or a string of characters. You'll see how to INPUT a "string" in the next exercise.

VARIABLES DON'T HAVE TO BE NUMBERS

1. Predict in the spaces below what the following program will print out:

```
10 LET NA$ = "NORMAN"
20 LET AD$ = "568 NORTH MAIN STREET"
30 PRINT NA$ ;" LIVES AT "
40 PRINT AD$
```

Notice these spaces!

Prediction:

____

____

Type in and RUN the program. Was your prediction correct? ________

Since there was no semicolon (;) after " LIVES AT ", the street address is printed on the next line.

Notice that a variable name can be used to store a "string" of characters (letters, numbers or symbols). If you wish to store a "string" like this, the variable name must end with a dollar sign ($). Also notice that in the LET statements above, the string that is stored is enclosed with quotation marks to indicate it is a string of characters and not just a very long variable name.

2. Change lines 10 and 20 so that the program user can type in the strings for NA$ and AD$, just as you did in the addition and multiplication program. IMPORTANT: also add PRINT statements in lines 5 and 15 to tell the program users to type in their own names and addresses. Write these new lines below:

5 ________________________________

10 ________________________________

15 ________________________________

20 ________________________________

RUN this new version of the program using your own name and address. Run it a second time using another name and address.

2c

A PROJECT TO TEST YOUR SKILLS

The sentence below has some blanks that you can fill in with a variety of words (or even phrases):

THE FEROCIOUS (animal) WALKED PAST
(some number) SMALL (things) THAT
TURNED (color) WITH FRIGHT.

Write a program that asks for an animal, then a number, then a "thing", then a color. The program should then write out the sentence with the blanks filled in. Record your program lines in the space below:

USING NUMBERS IN FORMULAS: COMPUTING WITH THE COMPUTER

Mathematical formulas can be put into a program so that the program user can just type in some numbers and the computer prints out the solution to a problem. These formulas could be very complicated, but the two examples below use simple and useful formulas.

1. There are 5280 feet in a mile. To figure out how many miles there are in a certain number of feet, you would divide the number of feet by 5280:

   miles = feet / 5280 or M = F / 5280

   Write a program below which:

   a. asks the user for the number of feet (INPUT);

   b. uses the formula above to calculate miles (M);

   c. prints out the number of miles in a sentence like:

   ```
   THAT'S 14.63 MILES.
          or
   10560 FEET IS 2 MILES.
   ```

Test out this program by typing it into the computer and RUNning it.

2. Distance traveled is calculated by multiplying speed by the time spent traveling:

   distance = speed * time or <u>D = S * T</u>

   Write a program which calculates the distance traveled during a car (or airplane, or flying saucer) trip.

   The program should:

   a. ask for the speed (S) in miles per hour or kilometers per hour;

   b. ask for the travel time (T) in hours;

   c. calculate the distance (D);

   d. print the result in a sentence.

   Be sure to use the right label for the distance answer. Use <u>kilometers</u> if speed was in <u>kilometers per hour</u>. Use <u>miles</u> if the speed was in <u>miles per hour.</u>

2e

HOME ACTIVITY #2 - VARIABLES

1. "Play computer" with the program below:

N$	A$	B$

```
 10 LET A$ = "APPLE"
 20 PRINT "WHAT IS YOUR NAME ";
 30 INPUT N$
 40 LET B$ = "PET"
 50 LET B$ = A$
 60 LET A$ = B$
 70 PRINT "I'M HAPPY TO MEET YOU "; N$
 80 PRINT "MY NAME IS "; B$
 90 PRINT "I KNOW A COMPUTER NAMED "; A$
100 END
```

Print in the space below what lines 70 through 90 would print out if you typed in your own name while running this program.

2. Mrs. Grimshaw gives three tests to her computer programming class. She wants to be able to type a student's name into the computer along with the three test grades. She then wants a sentence printed out telling the student's average grade. Write the program for her. It should look something like this when it is run:

```
STUDENT'S NAME ?HAROLD FARQUAHR...
```

(How would you get the name on the same line as the question?)

```
TEST 1 ?76
TEST 2 ?83
TEST 3 ?58

HAROLD FARQUAHR HAS AN AVERAGE OF 72.33333
```

If you use T1 for test 1, T2 for test 2, and T3 for test 3, the formula for the total of the three tests is:

TL (total) = T1 + T2 + T3

The average is the total divided by the number of tests:

AV (average) = TL / 3

Write the program on a separate sheet of paper. You will enter this program into the computer in class in the next session. You will also have a chance to improve on this program in later sessions, so keep this program listing.

Session 3

OBJECTIVES

The student should be able to:

- use the FOR/NEXT structure in BASIC to do a program activity repeatedly a specified number of times;
- use the counter variable in a FOR/NEXT loop to produce variations in a repeated activity;
- use the STEP function in a FOR/NEXT loop;
- identify the major parts of a computer system and their functions;
- explain how to translate a 5-bit binary number into its decimal equivalent.

MATERIALS

Classroom set of Session 3 worksheets.

Six 8-1/2" x 11" sheets of heavy paper or oaktag with the numbers 1, 2, 4, 8, 16, and 32 drawn on them.

Optional: A microcomputer which may be opened up to display processor, translator (BASIC ROM), and memory chips (an APPLE computer, for example).

TEACHING SUGGESTIONS

Follow-up on Home Activity (20 minutes)

1. Have the class participants type in their test averaging program (problem 2) from the home activity sheet. It should take about 10 to 15 minutes to type in the program and debug it. The example below is one possible solution to the problem:

```
100   PRINT "STUDENT'S NAME ";
110   INPUT N$
120   PRINT
130   PRINT "TEST 1 ";
140   INPUT T1
150   PRINT "TEST 2 ";
160   INPUT T2
170   PRINT "TEST 3 ";
180   INPUT T3
190   PRINT
200   LET TL = T1 + T2 + T3
210   PRINT N$; " HAS AN AVERAGE OF "; TL/3
```

2. Have the class predict the output from problem 1 of their home activity. This program should be loaded into the class demonstration computer and ready to run. LIST the program to use as a visual aid when the class is predicting the output. RUN the program to test out the predictions.

This is the first time participants have seen a variable first assigned one value, then later set to a different value. It is important to emphasize that only the most recently assigned value of a variable is "remembered" by the computer.

Activity 1: Repeating (30 minutes)

• Purpose:

To introduce the use of FOR/NEXT to do a computer activity repeatedly a specific number of times.

• Preparation:

Distribute the handouts, saying only that the class will now learn about a "smarter" way to repeat things than repeating them forever with a GOTO statement.

• Handout 3a (25 minutes):

In part 2 of this activity some of the participants may need some help, since it is the first time they have designed a program which loops using a FOR/NEXT structure.

A few people may not be sure what program statements should be included between the FOR and the NEXT statements. This will only become a problem if they try the second optional activity, in which they type in a name and have it repeated 20 times. Some people will put the INPUT statement and prompt inside the loop, forcing the program user to type in a name 20 times.

Another common error is writing the FOR statement with a "-" instead of the "TO" as in this example:

```
                    FOR X = 1 - 20

which should be:    FOR X = 1 TO 20
```

• Follow-up (5 minutes):

Go over any prevalent difficulties you observed during the classroom activity. With a solution to the second problem loaded into the demonstration computer, modify the program to allow INPUT of the name which is to be repeated (last optional activity).

Show how to identify the lines which one wants to have repeated, putting a "box" around this set of lines. Put the FOR before the first of these lines, and the NEXT after the last of them. The prompt for the name and the INPUT line should not be repeated in this example. so thev should not be between the FOR and the NEXT lines.

Activity 2: Doing Something Different Each Time (30 minutes)

• Purpose:

To use the counter variable from the FOR loop to vary the activity each time a loop is executed.

• Handout 3b (25 minutes):

In part 1 the counter starts with 0. You may wish to point out as you move about the room that the computer can count from any starting point. Some people may wish to try starting from +3 or -10 as a follow-up.

Part 3 is an excellent time to emphasize "boxing in" the part of the program which is to be repeated. As before, the prompt line and the INPUT statement should not be included in the loop.

• Follow-up (5 minutes):

Field general questions on the activity and demonstrate a successful version of the optional problem of part 3.

Activity 3: Step by Step (20 minutes)

• Purpose:

To introduce the use of the STEP option in a BASIC FOR/NEXT loop.

• Handout 3c (15 minutes):

This activity is very straight forward. Encourage participants to try different STEP values in the programs if they have time.

• Follow-up (5 minutes):

Respond to questions. Demonstrate that the STEP value need not be an integer. (For example, change the Blast Off program so that it has a STEP of -.5.)

Activity 4: How the Computer Works (65 minutes)

• Purpose:

In this exercise, students will learn to identify the functional parts of a microcomputer system including input, output, processor, memory, storage, and translator unit (language ROM).

The exercise will also introduce students to the concepts of the binary number system, which will be applied in a functioning digital (human) computer.

1. Parts of the Computer System (10 minutes)

Distribute Handout 3d, Computer Parts, to the class. Explain the need for the various components listed on the sheet. On the demonstration computer system, point out the input, output, and storage devices.

If you have a computer in which the processor, memory chips, and BASIC ROM chips are visible, show those parts as well. On an APPLE II+ or APPLE //e computer, all of these components are easily seen under the cover. An ATARI 800 computer has removable RAM memory cartridges and a removable BASIC language ROM cartridge, but the processor is not easily accessible. Many other computer systems are similar to the ATARI, in that the processor chip is not visible.

An APPLE computer, or another computer in which all the components may be displayed, is a worthwhile teaching aid. When all the computer works are visible and can be categorized into six simple functions, the computer becomes much less of a "mystery box."

2. Binary Numbers (10 minutes)

Ask the class the meaning of the term 'digit'. Two important definitions should arise: 1) any of the numerals in our Arabic numbering system, and 2) fingers or toes. Undoubtedly our decimal number system developed on the base ten because of the number of fingers on human hands.

Briefly introduce the concepts of an alternate number system. Explain that the binary system is used in computers because of the ease of representing two states with electricity—on or off, positive or negative, or high and low—corresponding to a 1 or a 0. Write the equivalents for the decimal numbers from 1 to 10 on the board for the binary number system.

Decimal	Binary (base 2)
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010

Notice that our decimal system "counts" by means of two basic steps: 1) successively larger digits (one through nine) appear in the first column; 2) once the maximum digit is reached (9), the counting proceeds by clearing the column to zero, and "carrying" a one, producing the number ten (10).

Counting in the binary system is similar, except that only 0 and 1 are possible. When a zero exists in a column, the next number is one (1), just as in the decimal system. But, continuing to count, adding another one is much like adding one to a nine in the decimal system; the column is cleared to zero, and a one is carried.

Practice counting in the binary system by having them continue the series through the binary equivalent to twenty (20). (They should arrive at 10100.)

Next have the class study this series and mark each value in which a one first appears in a new column. (See below.) In the decimal system we are accustomed to recognizing the significance of a digit's position--the tens' position, or hundreds', or thousands'. The position of a digit in the binary system is also significant, and indicates the values below.

Decimal	Binary (base 2)
1	1
2	10
4	100
8	1000
16	10000

3. Human Computer (45 minutes)

A dynamic demonstration of the use of binary numbers in a digital computer can be performed using class members in a human computer. This activity is an expansion of the ideas presented by William A. Leonard in the article, "A Students Computer that Really Works," published in The Mathematics Teacher, December, 1970.

****
NOTE: This is a complex activity that could easily involve over an hour in the classroom. To limit the activity to 45 minutes, only the demonstration of addition, subtraction, and multiplication by successive addition should be attempted (parts A through D). The rest of the activity is reproduced here for classes that may want to further explore this human analogy, perhaps as one of the optional activities during the last class session.
****

A. Introduction: Have five students move their chairs forward, turn around and sit down facing the rest of the class. Give the student to the right a large card with the number 1 printed on it. This card should be held in the student's left hand (or taped on). Give the succeeding students across the front, the numbers 2, 4, 8, and 16.

Explain to the class that this front row of students is going to be the processor of a computer. They are to operate under the following rules:

1) If you are tapped, put your right hand up.
2) If you are tapped, and your right hand is up, put it DOWN and tap the person on your right.

The rules should be posted on a large sign. Enter sample numbers into the Processor by tapping the appropriate students on the shoulder. For example, to enter the number 3, you would tap students 1 and 2. They should put their hands up and the number 3 can then be understood by converting it back to decimal system, by adding the cards they are holding (1 and 2). Introduce the command CLEAR. When you say the word CLEAR, all students in the Processor with their hands up should put them down. CLEAR the unit, enter other numbers, verify results, and CLEAR the unit again.

Practice counting with the human computer by slowly and successively tapping the shoulder of student 1, counting audibly to the class, "1, 2, 3, . . .", so that they can visibly see the relationship between decimal values and our binary representations. Write a number on the board, such as 11010, in the binary form. Ask them to determine its decimal value (26). LOAD the number into the Processor to display it.

B. Addition: The power of the Processor lies in its ability to add numbers. Demonstrate this by first entering the number 4 (tap student). The answer can be read by adding the cards of all students with their hands up (8 and 4 or 12).

Use CLEAR and try the following successively more difficult additions:

```
6 + 4        =    10
5 + 7        =    12
16 + 3 + 4   =    23
20 + 5 + 4   =    29
```

Ask the class to determine the largest value with which this computer can deal (31). Ask them how a computer could be built to handle larger numbers. (Add more chairs.)

Add one more student—number 32—to the front row. Explain that each student in the Processor represents one BIT in a computer (from Binary digIT) and that the row is one BYTE. In effect you have modelled a 6-bit byte computer. Most microcomputers have 8-bit bytes. How large a number can such a computer handle? (128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255). Larger numbers are handled by grouping bytes together.

C. Subtraction: Can this processor subtract? Demonstrate the subtraction problem 25 - 7 by first loading the number 7 (tap students 1, 2 and 4). Then define a new command, the word COMPLEMENT; at this command, students are to instantly do exactly the opposite of their current state. That is, if the hand is up, put it down; or if down, put it up (but do not tap your neighbor). Issue the command COMPLEMENT, then load the number 25.

NOTE: When bit number 32 on the end puts his/her hand down, there is no one to tap. Let the students guess the proper action. He/she must get up, walk around and tap bit number one, then be seated again. If all goes well, adding the cards of the bits with raised hands should yield the number 18 (25-7) and, indeed, the Processor can subtract.

Demonstrate another simple subtraction problem: 20 minus 3

```
LOAD 3
COMPLEMENT
LOAD 20 (remind 32 to tap 1)
```

Answer: 17

D. Multiplication: Ask the class if they could multiply with this computer. One method for multiplying is by successive additions. Demonstrate multiplying 5 times 3 by entering the value 5, three times.

Optional: Greater speed in multiplication is accomplished by the use of the binary SHIFT, resulting in a multiplication by 2. Define the SHIFT command as "Put your arm in the same position as the person on your left." Prior to a SHIFT command, prepare the students with a "Get ready to SHIFT," meaning, look at your left neighbor's hand.

Demonstrate this procedure by multiplying 7 times 2. Enter the number 7 by tapping numbers 1, 2, and 4. Say "Get ready to SHIFT;" issue the command SHIFT. Student 1 will have to drop his or her arm, since there is no one on the left. The value remaining should now equal 14, much faster than entering 1, 2, and 4 a second time. Practice multiplying a few simple values by two. Then multiply 6 times 4 by loading the value 6, then performing two SHIFTs. Ask the class how the problem 5 times 7 could be done using this SHIFT technique; the solution below uses two SHIFTs and one addition:

Problem:	5 x 7		(Current Value)
		Enter 7	7
		SHIFT (2 x 7)	14
		SHIFT (2 x 14)	28
		Enter 7	35
Answer: 35			

E. Division (Optional): Division may be accomplished by reversing the multiplication procedures. Thus you may use a reverse SHIFT (a shift to the condition of the person on the right). Take care to choose only very simple problems, with no remainder or fractional answer (6 or 14 divided by 2, for example).

F. Microcomputer Components (optional): Label the front row of students the Processor by placing a card near them, or on the first student. Next introduce the MEMORY unit of a computer by asking the remainder of the classroom to participate. Announce that the first row will be row A, the next row B, and so on.

O	O	O	O	O	O	Processor
O	O	O	O	O	O	Row A
O	O	O	O	O	O	Row B
O	O	O	O	O	O	Row C
O	O	O	O	O	O	Row D

Enter the number 22 into the Processor by tapping numbers 16, 4, and 2.

Introduce the command STORE; explain that a memory row will be identified and the people in that row must mimic the bin in front of them in the Processor. Try this by saying STORE into B. The appropriate hands in row B should go up, and Arithmetic hands should stay up. On the command CLEAR, the processor hands should go down, but memory (row B) hands should stay up.

Enter the number 3, store it into row A, and CLEAR the Processor. Now use the new command LOAD. On the instruction, "LOAD B", the Processor bits should mimic the bits directly in front of them from row B. When LOAD A is announced next, students should think of this as being tapped by their corresponding bit from row A. If their hand is already up, they may have to drop it and tap their neighbor. At this point, the current status of hands should register the value of 25. If all is functioning, CLEAR and temporarily turn off the computer (all hands down).

INPUT: Ask a student to volunteer to become the INPUT part of your computer. Provide a card on which the following program is written:

```
10    LET A = 3
20    LET B = 5
30    LET C = A + B
40    PRINT C
```

Remind the class that INPUT is information that comes from a device like a keyboard or a game control.

OUTPUT: Ask another student to volunteer as the OUTPUT part of the computer. This student should stand by the board and write anything on the board that you say.

PROCESSOR: The teacher is acting as the control part of the processor, directing and controlling all the other computer functions. The students are performing the arithmetic function of the processor.

Begin Computing! Ask INPUT to begin reading the card, one line at a time. As this occurs, you should write the characters on the board. (Normally each character would be stored in the various rows of memory but too few people are present to carry this out.) After the first line of the program is read (LET A = 3), enter it into the Processor, then STORE it in row B, and CLEAR the Processor. Next LOAD A into the Processor. LOAD B into the Processor (which should automatically add 3 plus 5, displaying 8). STORE the Processor contents into row C. PRINT the contents of row C by examining the bits, converting to 8 and telling OUTPUT to PRINT 8 on the board.

The human computer is now complete. You should review the main parts of a computer again to be sure the students understand the analogy presented here.

Home Activity (10 minutes)

• Purpose:

To reinforce the use of the FOR/NEXT loop using STEP.

To familiarize the participants with computer systems designed for home consumers and to reinforce computer terms and concepts learned in this session.

• Preparation (10 minutes):

1. Briefly describe the thermometer problem on Handout 3e. Indicate that the class members will type in their thermometer program at the beginning of the next class session.

2. Identify the purpose of the computer store visit activity. Briefly go over the type of information which will be gathered. List some of the computer systems which might be investigated, and mention some of the local stores where this information might be gathered. Suggest using the yellow pages as a resource for those who do not know what stores carry specific machines.

You should also try to get a feeling for which computers are of interest to the group. Encourage class members to choose machines to review in such a way that a variety of computers may be compared during Session 4.

REPEATING

1. You've seen this program before:

```
10 PRINT "I WON'T SHOOT SPIT WADS IN CLASS."
20 GOTO 10
```

What did this program do? ______________________________

____

Here's a similar program:

```
 5 FOR T = 1 TO 10
10 PRINT "I WON'T SHOOT SPIT WADS IN CLASS."
20 NEXT T
30 PRINT "DONE."
```

What do you think this program will do when it is run?

____

Type it in and RUN it.

What did it do? ______________________________

____

A piece of a program that repeats itself is called a loop because it goes "around and around" in circles. The GOTO example at the top of the page is called an "infinite loop" because it goes on forever (until you stop it with a BREAK or RESET). The second version you did is called a controlled loop because it is "smart" enough to stop after a certain number of times. Here's how it works.

```
5 FOR T = 1 TO 10
```

Variable T is a "counter". It starts out being equal to 1.

The computer follows the program steps until it gets to:

```
20 NEXT T
```

This lines means go back to the beginning of the loop (line 10) and use the next counting value of T (2). This keeps happening until the NEXT T is 11. Since the computer has been told only to count to 10 the loop is finished and it goes on to the next line in the program (line 30).

Loops with counters are very useful since they allow the computer to do similar things over and over again a certain number of times.

REPEATING (continued)

2. Write a program using a FOR-NEXT loop which will print your name out 20 times and then stop. Write the program steps below. Use the spit wad program as an example.

OPTIONAL:

Modify this program so that it prints a blank line before and after each time it prints your name.

Modify the program so that you can INPUT any name, and have it printed 20 times.

DOING SOMETHING DIFFERENT EACH TIME

1. Sometimes you may want to use the counter number itself in a program. Since it works like any other variable, that is easily done. Predict what will be printed out by this program, and write in the box below what you expect to see printed out.

```
10 FOR N = 0 TO 5

20 PRINT N, N*N

30 NEXT N
```

Prediction:	Actual:

Type in the program.

RUN it.

Was your prediction correct? ______________

If your prediction was not correct, write what the screen actually showed.

2. Write a program which will show the multiplication tables for the number 3. The output you want is shown below.

```
3 * 0 = 0
3 * 1 = 3
3 * 2 = 6
3 * 3 = 9
3 * 4 = 12
3 * 5 = 15
3 * 6 = 18
3 * 7 = 21
3 * 8 = 24
3 * 9 = 27
3 * 10 = 30
```

Program:

3. Change the previous program so that the user can request the multiplication tables 0 through 10 for a specific number (not just 3). See the example. Write your program in the space below. RUN it to make sure that it works.

```
TABLES FOR WHAT NUMBER ?6

 6 * 0 = 0
 6 * 1 = 6
 6 * 2 = 12
 .
 .
 .
 6 * 10 = 60
```

3c

STEP BY STEP

1. You've seen that a counter doesn't have to start with 1. You also don't have to count by ones. If you want to count by some other amount you can add a STEP to your program. Type in the program below and RUN it.

```
10 FOR Q = 0 TO 100 STEP 10
20 PRINT Q
30 NEXT Q
40 END
```

Describe what STEP 10 did: ____________________

____

Change a line of the program above so that it counts to 100 by twos. Run it to test it out. Write the step in the space below:

____

2. In the space at the right, predict what the following program will print out. Type in the program and test your prediction.

```
10 FOR T = 10 TO 0 STEP -1
20 PRINT T
30 PRINT
40 NEXT T
50 PRINT "BLAST OFF!"
60 END
```

Prediction:

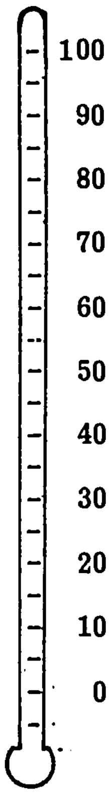
COMPUTER PARTS AND THEIR FUNCTIONS

Computer term	Function	Example
Input	Take in information	keyboard light pen joystick game controls "mouse"
Output	Send out information	TV or monitor display printer speaker
Processor	Do arithmetic Direct computer operations.	microprocessor (Sometimes called CPU for Central Processing Unit.)
Memory	"Remember" program instructions (short term)	memory chips or memory cartridges (RAM)
Interpreter	Translate English to "computer" (permanent memory)	ROM chips ROM cartridge
Storage	Hold information for later use (long term)	disk drive cassette tape

Thermometer Problem

For a certain program you want to produce a picture of a thermometer. The picture you want to make looks something like this:

Idea Program output Program:



```
A
H 100
H
H  90
H
H  80
H
H  70
H
H  60
H
H  50
H
H  40
H
H  30
H
H  20
H
H  10
H
H   0
H
O
```

As you can tell, the output is a repeating pattern in which capital H's are used for the thermometer tube and its marks. The program prints an H, followed by a number, then one without a number. It then repeats the process, printing up a number that is lower by 10 each time. The last number it prints is 0. The top of the thermometer is the letter A. The bottom is the letter O.

In the space next to the thermometer, write the program that would make this pattern. <u>You will type this program in and test it out during the next class session.</u>

3f

HOME ACTIVITY #3

This activity is designed to give you a chance to practice your computer vocabulary, to learn more about the different kinds of microcomputers, and to look at some of the kinds of computer software which are available.

In order to complete this activity, you have to visit a store that sells microcomputers. It may be a store that specializes in selling computers or a department store or electronics store.

You will be given an opportunity to choose a microcomputer brand which interests you. The activities at the store are in two parts: Part I should be completed by the parent and Part II by the son or daughter. During the class session next week you will be asked to share the information you found; pamphlets and brochures will be helpful in describing the microcomputer of your choice to the rest of the class.

Part I:

1. What brand of microcomputer did you investigate? ____________

2. What store did you visit? ____________

3. If you purchased the lowest cost system, . . .

   a. is a display screen or TV monitor included? ________

   b. is a disk drive included in the price? ________

   c. is a casette recorder included? ________

   d. how much RAM memory does it have? ________

   e. does it accept solid state cartridges? ________

4. What is the price of this microcomputer system? ________

5. Describe the computer's keyboard. ____________

   Would the keyboard be easy to type on? ____________

6. How many characters does the computer print across the screen? ____

7. How many lines does it print down the screen? ________

3f

HOME ACTIVITY #3 (continued)

8. Does the microcomputer print both upper and lower case letters on the screen? ____________

9. Can a disk drive be attached to this micromputer? ________

   What is the cost of the first disk drive? ____________

   What is the cost of a second disk drive? ____________
   (Include the cost of any required hardware or extra memory.)

10. What are the matrix dimensions for graphics on the screen? (For example, some computers print 49 dots by 127 dots on the screen.) ______________

11. Does the micromputer have the capacity to display in color? ___________

12. What special features of this micromputer make it better than its competitors.?

____

____

MINNESOTA EDUCATIONAL COMPUTING CONSORTIUM

HOME ACTIVITY #3 (continued)

PART II:

1. How many different games do you estimate are available to run on this brand of microcomputer? ________________

2. How much does the typical game cost? ____________________________

3. Do they sell any special peripherals such as game paddles or joysticks which can be used with the games? __________________

   If so, what are they and how much do they cost? ______________

   ____

4. What is the name of the game that looks the most fun to you? _______

   ____

   Briefly describe the game: ______________________________

5. Does the store sell any educational programs for this brand of microcomputer? If so, describe one that looks interesting.

   ____

   ____

6. If you were purchasing a microcomputer today, would you buy this brand?

   ____

   Why or why not? ________________________________

   ____

MINNESOTA EDUCATIONAL COMPUTING CONSORTIUM

Session 4

OBJECTIVES

The student should be able to:

- use the IF/THEN, GOTO, and END statements in programs to create conditional branches;
- personally investigate and discuss consumer features of one brand of home computer.

MATERIALS

Classroom set of Session 4 worksheets

TEACHING SUGGESTIONS

Follow-up on Home Activity #3 (20 minutes)

Allow class membeıs to type in their thermometer programs at this time. The program below is one solution to this problem:

```
100 PRINT "A"
110 FOR T=100 TO 0 STEP -10
120 PRINT "H ";T
130 PRINT "H"
140 NEXT T
150 PRINT "0"
    END
```

Activity 1: Branching (30 minutes)

- Purpose:

To introduce the IF/THEN command to accomplish a simple program branch.

- Preparation (10 minutes):

Discuss briefly the concept of a flowchart. For various reasons, flowcharts don't easily lend themselves to the use of BASIC. Participants will, however, see flowcharts used and they should understand that these tools are graphical representations of the "flow" of a program.

Up to this point the class participants have seen three different program forms:

```
Linear          Infinite loop               Counting loop

10              10                          40 FOR T=1 TO 10

20              20                          50

30              30            forever       60            10 times

40              40                          70

50              50 GOTO 10                  80 NEXT T
```

Every program that they have written to this point has been built from one or more of these basic forms. For instance, the thermometer program starts with a linear section, (100 to 110), goes through a counting loop section, (110 to 140), then ends with another linear section, (140 TO 150). You might wish to write a listing of this program on a blackboard or overhead transparency and just draw in arrows to show the program flow.

Introduce the desire to go elsewhere in the program based on the result of a "test" performed by the program. Indicate that such a two-way choice is called a <u>branch</u> because it divides the program flow in two parts just as a tree branch divides the wood in a tree two ways. A branch can go backwards in a program, similar to a loop, or it can go forward in the program, skipping some lines of the program. The first program that the class participants will create will be an example of such a forward branch.

• Handout 4a (20 minutes):

The activity sheet is very straight-forward. The only difficulty that may occur has been calculated to provide difficulty. The program has a bug. Everything works fine if they answer "GEORGE" to the question; line 40 is skipped and the program jumps to line 50, printing: RIGHT, IT WAS GEORGE!

If they answer the question wrong, however, they get both the right and the wrong response:

```
SORRY IT WAS GEORGE, NOT MARTHA
RIGHT, IT WAS GEORGE!
```

This problem leads in to an introduction to the most common programming problem beginners have with BASIC IF/THEN branches. The next activity provides some direction in handling this problem.

Activity 2: Untangling the Branches (45 minutes)

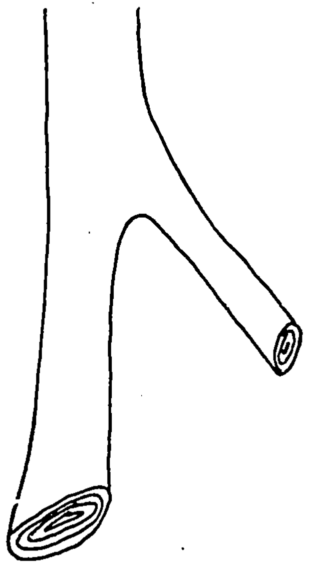
● Purpose:

To learn how to avoid executing both the TRUE and the FALSE functions of a forward IF/THEN branch when the test is FALSE.

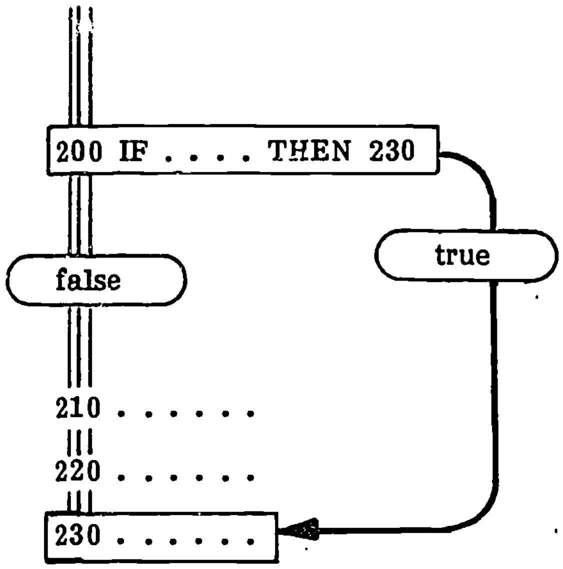
● Preparation (10 minutes):

Discuss the bug that appeared in the George Washington program. The problem is that a program branch, unlike a tree branch, eventually merges back into the "trunk" like this:

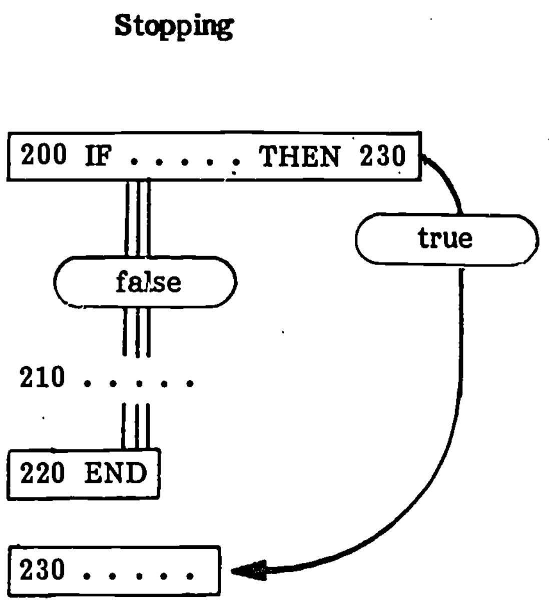
Tree Branch

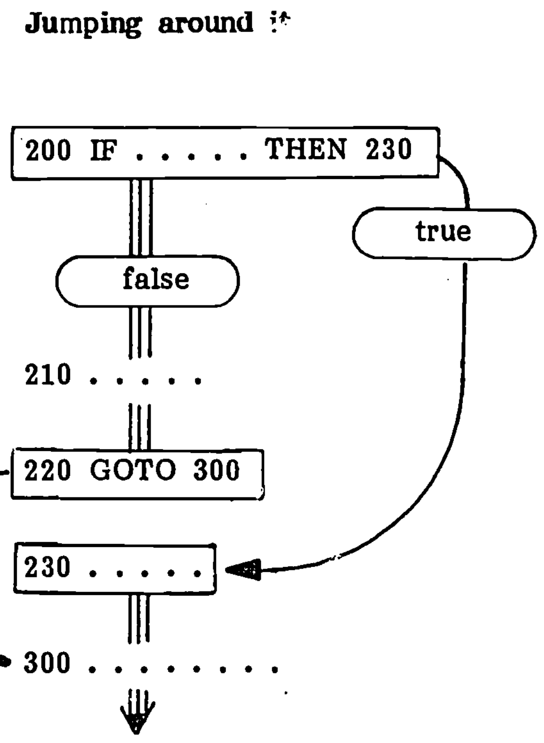


Program Branch



There are two ways to avoid this "collision" of the program flow paths at line 230. First, if the program is over after the branch is executed, an END statement immediately before line 230 would stop the FALSE case from running into the TRUE case. If the program is to continue after this branch, a GOTO can be used immediately before line 230 to jump around part of the program:





Tell the class participants that they will have a chance to practice both of these strategies in the next worksheet as they do steps 2 and 3.

• Activity (25 minutes):

In part 2 the class participants should put the END statement right before line 50 (at line 45, for instance). In part 3 this line has to be changed to a GOTO.

In part 3 you may have to encourage the participants to add only <u>one</u> additional question at first. Some will get involved in the process and add a whole string of questions on this quiz. After the <u>first</u> question they add, have them test out the program by answering questions right and wrong. Once they have the program functioning properly with two questions (by adding a GOTO statement) they may add extra questions, testing the program after each one.

• Follow-up (10 minutes):

Demonstrate one of the class members' programs that successfully asks two questions and responds appropriately, depending upon whether the question is answered correctly or not. Choose one which only asks two questions to simplify the analysis of the program listing. Ask the class how additional questions might be added to this program.

Introduce some additional tests which may be used in IF/THEN statements such as:

> greater than	< less than	>= greater than or equal to
<= less than or equal to	<> not equal	

When comparing numbers, the meanings of such comparisons are obvious. It may surprise the students that tests like "greater than" can be used with strings as well. If one string (containing a word or group of words) is greater than another, it means it comes later in alphabetical order. You may demonstrate this with the following program:

```
100 PRINT "TYPE IN A WORD: ";
110 INPUT A$
120 PRINT "TYPE IN ANOTHER WORD: ";
130 INPUT B$
140 IF A$ < B$ THEN 200
150 PRINT A$; " COMES AFTER "; B$
160 END
200 PRINT A$; " COMES BEFORE "; B$
```

The class used " = " to see if GEORGE was typed in during the quiz program. In that application they could have used < > (not equal) to see if GEORGE was not typed in instead.

Activity 3: Branching Off On Your Own (25 minutes)

- Purpose:

To create an original program which branches based on used [illegible]

- Handout 4b (20 minutes):

In this program there is some latitude for variation. The program as described could check to see if the response is equal to "YES" or it could check to see if it is not equal to "YES". Some students will write the program to check for "NO" instead. Either way, there is a problem if the user types in "Y", "N", "NOPE", or something else. The intent of problem 5 is to handle such responses.

- Follow-up (5 minutes):

Respond to questions. You may wish to demonstrate a program created by one of the class members.

Activity 4: Store Visit Reports (50 minutes)

Ask for participants to report on the various computers that they investigated. As an individual completes a report, ask those who researched the same computer what they would add to the report. To keep track of the information, create a comparison table for the various brands on the blackboard or on an overhead transparency as the reports are given.

Point out the difficulty of comparing "Apples to oranges" like this. How can one fairly compare prices of computers with very different capabilities, memory, etc.? How important is availability of software? For educational use, what computers might be best? For programming only? For business? For entertainment? Is there a big difference in the cost of various peripheral devices for the machines (such as disk drives)?

Home Activity Preparation (5 minutes)

Problem 1 of Handout 4c puts together all of the programming concepts learned so far: input, output, looping, and branching. The class has to figure out what the IF/THEN and GOTO lines should say in order to make the guessing game work correctly. This program will be typed in and extended at the beginning of the next class session.

Problem 2 is a simple branching program for the class to create. Tell the class that they will have time at the beginning of session 5 to type this program into the computer and try it out.

BRANCHING

One thing that a computer can do rather easily is to make "decisions". The way that a computer makes a decision using a BASIC program is by using an IF - THEN statement. IF - THEN statements in BASIC work like IF - THEN statements in English:

$\boxed{\text{IF}}$ you mow my lawn $\boxed{\text{THEN}}$ I will pay you $10.

Answer:	Result:
YES	GET $10
NO	DON'T KNOW WHAT MIGHT HAPPEN

IF you can answer YES or TRUE to the statement between IF and THEN (you mow the lawn) THEN you will get $10.

IF you can't say YES to the "test" statement THEN the consequences are unknown. (In this case, you might get $5 anyway, as a present!) In a BASIC program you have to go on to the next line in order to find out what the NO answer will cause the program to do. A YES answer most often will direct the program to a line other than the following line. This type of IF - THEN statement will say something like:

```
IF X=3 THEN 230
```

THEN 230 means the same as THEN GOTO 230. If X is equal to 3 then the program will branch and go to line 230. This type of decision is called a branch because the program operation can follow two different paths, just like a branch in a road or a branch in a tree.

1. To see how branching works in a BASIC program, "play computer" with the following program listing. Don't type it in yet!

```
10 PRINT "WHAT WAS PRESIDENT WASHINGTON'S FIRST NAME ";
20 INPUT N$
30 IF N$="GEORGE" THEN 50
40 PRINT "SORRY, IT WAS GEORGE, NOT ";N$
50 PRINT "RIGHT, IT WAS GEORGE!"
60 END
```

   Predict what would be printed if you typed GEORGE for the answer:

   Predict what would be printed if you typed MARTHA instead of GEORGE:

BRANCHING (continued)

Type in the program and try both [illegible] ses out. Were your predictions correct?

Describe what did happen if your predictions were incorrect:

Don't turn off your computer or type NEW! After your teacher talks to you for a few minutes, you should be able to fix up this program so that it works well.

2. If you just wanted to ask one question in the "Washington" program, then an END is the easiest way to keep the "RIGHT" response from being printed when the answer is wrong.

   Figure out where you need to put an END statement and write the program line (including number) that you would add to the program in the space below:

   Try this new version of the program out with both correct and incorrect responses. Does it work well now?

3. Without changing the line you just added, add some more lines to the program to ask another question (Lincoln's first name perhaps). Write the lines below, starting with line 60:

   Run this new version of the program. What happens when you answer the first question correctly?

   What happens if you answer the first question wrong?

BRANCHING (continued)

Rewrite one of the lines so that the program goes on to the second question, no matter whether you get the first question wrong or right. Write the new line below:

Try it out. Does it work well now?

BRANCHING OFF ON YOUR OWN

1. Write a computer program which will ask if you would like to go for a picnic. If you answer YES, then print out GOOD, SO WOULD I. If you don't answer YES, print out: OKAY, LET'S LEAVE IT FOR THE ANTS. Record your program steps below:

   Test out this program. When you answer SURE, what would this program print out?

2. Add to this program so that if the answer is not YES, then the next line checks to see if it is NO. If the answer isn't NO either, then ask the question again. Write any lines you needed to add in order to do this in the space below:

   Test your program with answers like NOPE, OF COURSE, JUMP IN THE LAKE, and so on.

4c

HOME ACTIVITY #4 - BRANCHING

1. Written below is part of a number guessing game. The programmer changes the "mystery number" (line 10) each time a different number is desired. The player gets eight guesses (counted with a FOR-NEXT loop). If the number guessed is too low, an appropriate response is printed. The same is true for a number which is too high. Fill in the blank program lines so that the program will work this way.

```
10 LET N = 27
20 PRINT "I'M THINKING OF A NUMBER FROM 1 TO 100"
30 FOR G = 1 TO 8
35 PRINT
40 PRINT "GUESS #"; G
50 INPUT AN

60 ______________________________          Testing AN to see if
                                           it is high or low.
70 ______________________________

80 PRINT "THAT'S IT!"

90 ______________________________          If it's right, what to
                                           do next?
100 PRINT "THAT'S TOO LOW."

110 ______________________________         Where do you want to
                                           go from here?
120 PRINT "THAT'S TOO HIGH."
130 NEXT G
135 PRINT
140 PRINT "I'M SORRY, THAT'S ALL YOUR GUESSES."
150 PRINT "THE NUMBER WAS "; N
```

As a review . . . can you still do this?

2. Write a computer program which will ask your age. If your age is greater than 15, have the computer print out a special message. If your age is 15 or less, have the computer print out a different message.

Session 5

OBJECTIVES

The student should be able to:

- give examples of common computer software applications including entertainment, educational, and business-utility programs;
- (cite and follow through other objectives to meet their needs and desires.)

MATERIALS

Post-test (optional). Reorder the test items from the way they appeared in the pre-test

Class evaluation sheets (optional; not included)

Certificates

Software including games, educational applications, and business applications, such as a word processing package, a database package, and a spreadsheet package

Additional materials as needed, depending upon the activities chosen for presentation or TEACHING SUGGESTIONS below

TEACHING SUGGESTIONS

Follow-up on Home Activity (60 minutes)

Allow class participants to type in their home activity programs and try them out. Have them start with the second program, then work on the first program (guessing game). Once they have the guessing game program running, it can be modified to be a little more exciting.

1. Have them replace line 10 with several lines which would allow one player to type in a number, then let a second player guess that number. One solution is:

```
 5    PRINT " FIRST PLAYER TYPE IN A NUMBER (1 TO 100)"
 7    INPUT N
10    HOME (or CLEAR, or whatever clears the screen on your particular
           computer system.)
15    PRINT "SECOND PLAYER MAY NOW START GUESSING."
```

2. Demonstrate the ability of the computer to produce random numbers for this type of game. On a TRS-80, for instance, the line:

```
10 LET N = RND (100)
```

will randomly generate an integer from 1 to 100. For several other brands of computers (including APPLE and ATARI) the following command may be used to do the same thing:

```
10 LET N = INT (RND (1) * 100 + 1)
```

In this formula 100 is the number of different possible integers which the formula may generate. The "+1 " at the end is the value of the lowest integer which it can produce. Do not attempt to analyze the formula any further than this for the class. Have the participants modify their programs so that the computer "thinks up" the number to be guessed.

Consumer software (30 - 60 minutes)

- Purpose:

To acquaint the participants with commercially available software that is appropriate for home use.

To relate the operation of commercial software to the programming experiences of the group.

- Activity:

Various types of commercially available programs may be desirable for home use. In this session the teacher should briefly demonstrate a variety of the most common applications. These may include:

Entertainment/game programs

These are most interesting to the children in the group. Where appropriate, discuss programming techniques which might have been used. Specifically, point out cases of input, output, looping, and branching. Using Pac-Man as a well-known example, output includes the PRINTing of text at the beginning of the game, drawing the graphics (maze) and a great deal of animation. Although the participants in this course have not used graphics, they have done animation with the "rocket" program. Input comes from a joystick or a keyboard. Branching of the IF/THEN variety occurs in various places in the game. If Pac-Man runs into a ghost monster, AND if the ghost monster is blue, Pac-Man gets some points. If the ghost monster is another color, THEN Pac-Man gets some points. IF Pac-Man runs into a wall, THEN he stops until the joystick is moved appropriately. Looping happens in the game as well. For instance, Pac-Man gets three "lives" before the game is over. . . (FOR PAC = 1 TO 3).

Business utility software

These applications will mainly be of interest to the parents in the group, but they can be made relevant to the children as well. Do not go into depth on any one package. Spend only 5 to 10 minutes on any single package you demonstrate.

Word processing can impress even children if you show them some of the powerful things that such a system can do to save time and energy. On a sample document, do a global word change, changing "APPLE" to "ATARI" or the like. This is a very powerful and impressive capability. Also show how easily large portions of text can be deleted or moved about in the document. Insert a phrase inside some sentence that completely changes the meaning of the sentence.

A database program may be used to demonstrate the computer's data retrieval and analysis capabilities. Pass around a report that has been generated by a database system that you may use. Inspect on the computer screen what a single record from that database looks like. Retrieve some information to the screen that would be difficult to do without a computer (all the computer stores in the northwest part of town which stock APPLE and ATARI computers).

Educational programs

A variety of educational programs are available for use on home computers, and this is one of the most commonly cited reasons for buying a home computer. Demonstrate one or two of these. Point out the advantages of a computer for the particular applications and also ask class participants to identify examples of output, input, looping and branching in each example. If possible, list one of these programs to show the common BASIC statements which they have learned.

Optional Activity (30 - 60 minutes)

- Purpose:

To broaden the participants' knowledge and interest in computing. To respond to interests expressed by the participants which have not been addressed by the curriculum materials.

- Activity:

In the course of an intensive class of this type, many questions arise which one may not be able adequately to address within the introductory curriculum included here. Some of these questions may interest only one individual, while others may be of broad interest and applicability.

Depending on the length of the software demonstration and on whether a post test will be given, there may be 30 to 90 minutes available for additional topics of particular interest to the class. Such topics might include:

Computer peripherals

Various peripherals are important to the home user. Certainly a printer is the most common and useful. A discussion of differences in print speed, cost, capability, etc., or between dot matrix and letter quality printers, may be useful for those interested in buying computer hardware. Having a dot matrix printer at the session to demonstrate would be desirable but not necessary. Personal Computing magazine annually runs a printer comparison article, which normally discusses rather well the differences between common types of printers. You may wish to duplicate the latest of these articles to share with the participants.

Other peripherals which one might show include joysticks, game controls, graphic tablets, light pens, and so on. Have the participants identify these items as input or output devices.

Human computer revisited

If the class enjoyed the human computer activity but were not able to complete it due to the amount of time involved, they may choose to complete the activity at this time.

Timeshare computer system

If the classroom has easy access to a telephone, you may wish to demonstrate how a microcomputer can be used as a terminal to a timeshare system. Show how a "log-in" procedure is required on such a system and how to find out what programs are available through the system that you have accessed.

Show some of the unique capabilities of a timeshare system. An electronic mail system or an electronic bulletin board is a good application to demonstrate the ability of various users to communicate with each other through the computer. Database systems with large data bases also show the unique abilities of a large computer.

Group discussions

Various topics lend themselves to a group discussion concerning the use of computers. The future of computer is a good topic of this type. Social impact questions about computers will certainly arise. How about privacy in the future? What will computers be capable of in ten years? How much will they cost? Who will have them? What daily activities will be easier, or at least different, because of the computer in the home? What is computer crime?

Post-test and Graduation (40 minutes)

If you have chosen to evaluate the class participants, administer the post test at this time. This test should take about 25 minutes. This is also a good time to pass out class evaluation forms to get feedback on your instruction, the class content, and the class structure. A "graduation" ceremony is also suggested. A master of a certificate which you may present is included as an Appendix.

APPENDICES

TEST

Name ______________________

Write the letter of the correct answer to the right. | **Answer**

1. Which of the following statements or combinations of statements will cause the computer to print the word CONGRATULATIONS on the screen?

   a. PRINT CONGRATULATIONS
   b. LET A=CONGRATULATIONS PRINT
   c. PRINT "CONGRATULATIONS"
   d. PRINT: CONGRATULATIONS

2. You have typed the following program into the computer:

```
10  PRINT "MY NAME"
20  PRINT "IS"
30  END
```

You then typed the following two lines:

```
NEW
25  PRINT "MICKEY MOUSE"
```

When you list your program, it will look like:

a.
```
10  PRINT "MY NAME"
20  PRINT "IS"
30  END
25  PRINT "MICKEY"
```

b.
```
NEW
10  PRINT "MY NAME"
20  PRINT "IS"
25  PRINT "MICKEY"
30  END
```

c.
```
10  PRINT "MY NAME"
20  PRINT "IS"
25  PRINT "MICKEY"
30  END
```

d.
```
15  PRINT "MICKEY"
```

3. Choose the correct output for the computer program shown below:

Answer

```
10  LET A = 3
20  LET B = 4
30  LET C = A
40  LET B = C
50  LET A = B
60  PRINT A;B
```

a. 3 4
b. 4 3
c. 3 3
d. 4 4

4. Which of the following would be the correct line 20 to add to the computer program below?

```
10  PRINT "WHAT IS YOUR NAME ";
20
30  PRINT "HI, "; N$; "GLAD TO MEET YOU"
40  END
```

a. 20 INPUT N$
b. 20 PRINT N$
c. 20 LET N$ = "NAME"
d. 20 INPUT N$ = "NAME"

5. What will be the output from the following computer program?

```
10  LET F = 0
20  LET F = F + 1
30  PRINT F
40  IF F  25 THEN 20
50  END
```

a. A vertical row of numbers, from 0 to 25, down the screen.
b. A horizontal row of numbers, 1 to 25 across the screen.
c. A vertical row of members, 1 to 24, down the screen.
d. Just the numbers 1 and 25.

6. Which of the following is <u>not</u> a computer input device? Answer

a. game controls
b. card reader
c. printer
d. light pen

7. Which of the following activities is most easily done with a word processing package?

a. Keep track of the names, addresses, and phone numbers of your friends.
b. Write letters to your friends.
c. Enter all of your recipes and search for specific ones.
d. Balance your bank book.

8. Suppose you write a program, type it into the microcomputer and RUN it; until it is erased, it remains current in the:

a. RAM
b. ROM
c. Central Processing Unit
d. Disk drive

9. In the equation below, what will the microcomputer determine to be the correct solution for the value of Y?

```
LET Y = ((5 * 10) + 30/3 - 1)
```

a. 65
b. 100
c. 59
d. None of these

10. Which decimal value below is equal to the binary number 10101?

a. 42
b. 111
c. 21
d. None of these

11. Consider the following program:

```
NEW
10  FOR X = 0 TO 100 STEP 5
20  PRINT X
30  NEXT X
40  END
```

Which of the programs below will cause exactly the same output to be printed on the screen?

a.
```
NEW
10  LET X = 0
20  LET X = X + 5
30  PRINT X
40  GOTO 10
```

b.
```
NEW
10  LET X = 0
20  PRINT X
30  LET X = X + 5
40  PRINT X
50  IF X   100 THEN
    GOTO 30
60 END
```

c.
```
NEW
10  LET X = 100
20  LET N = 100/5
30  PRINT N
40  GOTO 10
```

d.
```
NEW
10  LET X = 5
20  PRINT X
30  LET X = X + 5
40  PRINT X
50  GOTO 30
```

Answer

12. Which of the following statements will cause the number eighteen to be printed on the monitor screen?

a. PRINT 3   3*2
b. "PRINT 18"
c. PRINT "15 + 3"
d. PRINT 36/2

13. Consider the following program: | Answer

```
NEW
 10  PRINT "THIS PROGRAM WILL COMPUTE AN
     AVERAGE SCORE"
 20  PRINT
 30  PRINT "HOW MANY TESTS HAVE BEEN TAKEN?"
 40  INPUT N
 50  LET T = 0
 60  FOR X = 1 TO N
 70  PRINT "SCORE FOR TEST "; x
 80  INPUT S
 90  LET T = S/N
100  NEXT X
110  LET A = T/N
120  PRINT "THE AVERAGE IS ";A
130  END
```

In order for this program to run <u>properly</u>:

a. Change line 90 to LET T = T + S
b. Omit line 50
c. Change line 100 to NEXT N
d. No changes are necessary; it will run properly as written.

---

14. Of the following, which is the best reason for using a computer to help solve a problem?

a. The solution to the problem must be absolutely accurate.
b. The answer must be saved for future reference.
c. The problem must be solved logically.
d. Many rapid calculations are required for the solution.

---

15. In an 8-bit computer, the maximum value which can be represented by one byte is:

a. 256
b. 128
c. 8
d. 10 million

ANSWER KEY

Answers	Objectives
1. c	1-a
2. d	1-b 1-d
3. c	2-a 2-b
4. a	2-a 2-b
5. c	4-a
6. c	4-c 5-a
7. b	5-a
8. a	3-d
9. b	1-c
10. c	3-c
11. b	3-a 3-b 4-a
12. d	1-a
13. a	4-a
14. d	4-b 5-a
15. a	3-d

Shared Learning Experience in Computing

This is to certify that

____

has Completed

____

at the ______________________________

Signed: ______________

Dated: ______________

GLOSSARY OF MICROCOMPUTER TERMS

BASIC
A computer language which can be used when programming most microcomputers.

Bit
One "binary digit", a 0 or a 1, which the computer uses as a code to do arithmetic and send information through its electronic circuits.

Byte
The number of bits required to "code" a character (letter or digit) for storage in the computer (commonly 8 bits).

Chips
Small rectangular electrical circuit boards inside a microcomputer.

Cursor
The prompting symbol displayed as a white square on the video monitor that shows where the next character will appear.

Data Base System
A program that allows one to store information on a computer diskette and can recall this information at a later time.

Disk Drive
A mechanical device capable of reading and writing information on a diskette.

Diskettes
The thin plastic disks used by a computer to store information (also known as floppy disks).

Game Control
Computer accessory which has a button and a knob, used to control action in game programs.

Hardware
The physical pieces of a computer.

Input
Information put into a computer system.

Interface Card
A circuit board used to connect the APPLE II to peripheral devices (e.g., printer or disk drive).

GLOSSARY OF MICROCOMPUTER TERMS

Joy Stick
Computer accessor which has a button, and a movable "stick" used to control action in game programs.

K
Symbol meaning approximately 1,000 bytes or characters able to be stored in a computer's program memory.

Memory
Chips inside a computer which have the capacity to store information.

Microprocessor or Controller (CPU)
Central unit of a microcomputer. Controls the flow of all the information going into and coming out of the system and performs mathematical operations.

Monitor
Video display unit similar to a television used as a means of displaying output.

Output
Information sent out of a computer, to the monitor, printer, or to storage via disk drive.

Peripheral
Hardware attachments to a computer, allowing for capabilities beyond processing. Peripherals include printers, disk drives, monitors, modems, etc.

Program
A list of instructions which enables the computer to perform a process.

RAM
Acronym for random-access-memory. Memory which "remembers" the current program. The contents of this memory change each time a different program is run. (It is erasable, like a blackboard.)

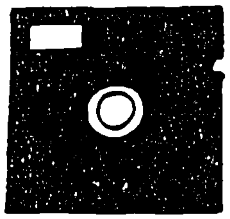
ROM
Acronym for read-only-memory. This type of memory is built into the machine and cannot be changed. It is used to remember the BASIC language, for example. (It is not erasable, like a book.)

Software
The programs used by a computer.

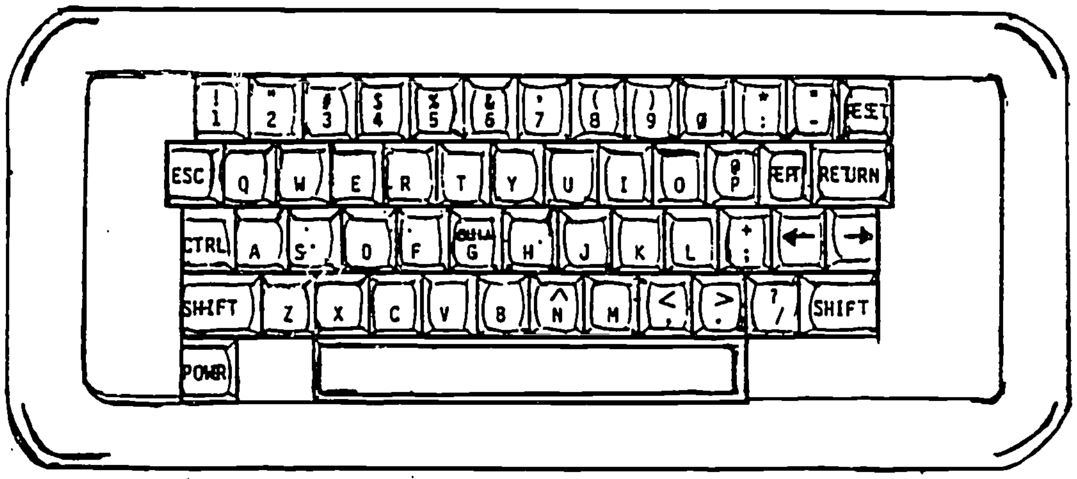
Word Processor
A program which allows one to write, edit, and print letters or other documents with the computer.

GETTING TO KNOW YOUR APPLE

Equipment



FLOPPY DISKETTE:	A 5¼ inch "record" that contains a series of computer programs.
DISK DRIVE:	A box unit that selects the program or data of your choice, displays the available list of programs on the diskette, and saves or deletes programs.
MONITOR OR TELEVISION:	Either a monitor or a television set that displays the information.



APPLE Computer Keyboard

The APPLE Computer keyboard looks much like the keyboard of a typewriter. Some special keys are noted below:

POWR **POWER** Light—Although this key displays **POWER**, it is only a light to show whether the system is on. It cannot turn on the system. The ON-OFF switch is on the back left side of the computer.

RETURN **RETURN** Key—Press this key when you are finished typing a line or when this manual notes RETURN.

SPACE BAR—This is the long rectangular key at the bottom of the keyboard. Most programs will print PRESS SPACE BAR TO CONTINUE at the bottom of the screen. Use the space bar at that point. In general, use the **RETURN** key after you type information into the computer and the space bar to proceed from one screen to another.

BACKSPACE Key—The ← key is used to erase mistakes. Each time you press it, the flashing white square called the "cursor" moves back one space. To correct mistakes, press ← until you locate the mistake, make the correction, and continue typing the line.

RESET Key—Press this key if something goes wrong or if you want to work with a different diskette.

SHIFT Key—Use the computer **SHIFT** key like that of a typewriter. If a key displays two characters, you may hold down the **SHIFT** key while typing to print the upper character. For example, holding down the **SHIFT** key and typing the ; key will print +.

CTRL (Control) Key—In some of the computer instructions, you must hold down the **CONTROL** button while pressing another key. For example, the notation CONTROL C means press the **CONTROL** key and the C key simultaneously. Control characters do not appear on the screen.

Keys That Can Cause Confusion

ZERO—The zero is on the top row of keys. Do not use the letter O interchangeably with this number key. In this manual zeros do not have slashes through them.

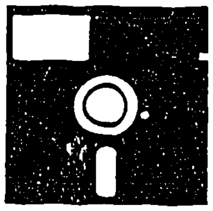
1 (One)—The number one is on the top row of keys. Do not use it interchangeably with a lower-case L (l).

GETTING TO KNOW YOUR ATARI COMPUTER

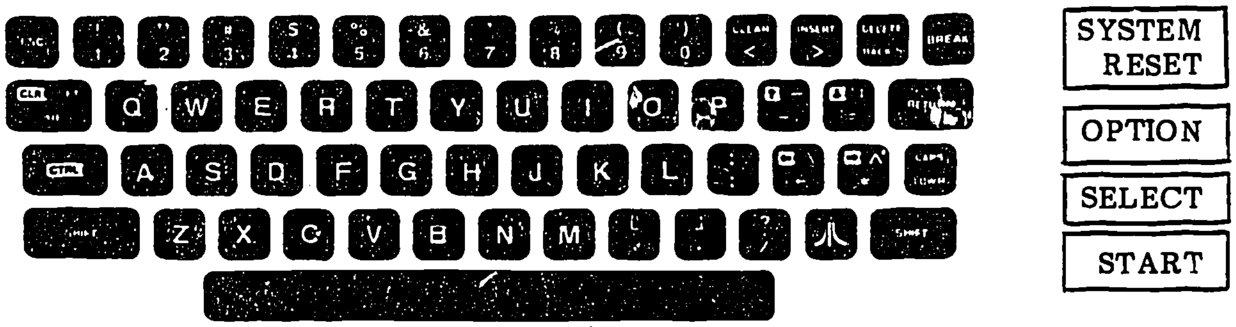
Equipment

ATARI COMPUTER CONSOLE:	The computer and keyboard.
BASIC LANGUAGE CARTRIDGE:	A cartridge (containing the BASIC computer language) that is inserted into the console above the keyboard.

TELEVISION:	A television set used to display information.
DISK DRIVE:	A unit that holds and reads the diskette.
DISKETTE:	A 5¼ inch "record" that contains a series of computer programs.



ATARI Computer Keyboard



The ATARI Computer keyboard looks much like the keyboard of a typewriter. Some special keys are noted below:

RETURN Key—When you are finished typing either a response to a question or a line in a program, you send the information to the computer by pressing the **RETURN** key.

BACK S (Backspace) Key—Each time you press the **BACK S** key, the cursor backs up one space and erases each letter it passes over. This feature allows you to correct typographical errors easily.

BREAK Key—Press this key to stop the execution of a program. The program will remain in the computer memory and may be run again. If BREAK doesn't work to stop the program, try the RESET key.

SYSTEM RESET

RESET Key—Like the **BREAK** key, the **RESET** key stops program execution. It also clears the screen. To restart, type RUN"D:HELLO".

ESC (Escape) Key—While you are using MECC diskettes, press the **ESCAPE** key <u>twice</u> in response to a question to stop program execution. The computer will ask whether you wish to run the program again. If you do not, the computer will display the diskette menu, and you may choose another program.

SHIFT Key—Use the computer SHIFT key like that of a typewriter. If a key displays two characters, you may hold down the **SHIFT** key while typing to print the upper character. For example, holding down the **SHIFT** key and typing will print !.

CAPS/LOWR (Capitals/Lower case) Key—When you press this key, the computer begins typing in lower-case letters. To capitalize individual letters, you must hold down the **SHIFT** key as with a typewriter. To switch back to all capitals, hold down the **SHIFT** key, and press the **CAPS/LOWR** key again.

CTRL (Control) Key—Hold down the **CONTROL** key while pressing another key if indicated by the computer instructions.

<u>Keys That Can Cause Confusion</u>

0 (Zero)—The zero is on the top row of keys. <u>Do not</u> use the letter O interchangeably with this number key.

1 (One)—The number one is on the top row of keys. <u>Do not</u> use it interchangeably with a lower-case L (l).

GETTING STARTED

Radio Shack Model III/4

EQUIPMENT NEEDED

To use this courseware you will need a Radio Shack Model III/4 computer 48K with one disk drive.

OPERATING THE SYSTEM

WARNING: To avoid damaging the programs that are stored on the diskette, be sure to insert the diskette into the disk drive after turning on the power switch for the system. Also, be sure to remove the diskette from the disk drive before turning off the system's power switch.

1. Turn on the power and insert the diskette

   - Plug your computer into a wall outlet or a power source.
   - Turn on the power switch for your computer. The switch is under the right front side of your computer.
   - A red light will go on next to the slot on the disk drive that you should use. While the light is on, follow these steps:
     - Open the drive door.
     - Position the diskette with the label on top.
     - Gently insert the diskette until it stops.
     - Close the drive door.
   - If you are not able to insert the diskette before the red light goes out, shut off the computer, and try again.

2. Run a program

   - The Radio Shack logo will appear.
   - When you are asked to enter a date, use this format: month/day/year (e.g., 08/19/83).
   - Press the ENTER key for the time prompt. It is not necessary to enter the time.
   - The MECC logo will appear, followed by the menu of programs available.
   - Select the program you wish to use.

SPECIAL KEYS

- The left arrow key ← is used to make changes or corrections prior to pressing the ENTER key.
- The space bar is used to advance frames when a response is not required.
- The ENTER key is used to advance a frame after a student response is entered.
- From the main menu frame, the teacher can gain access to special teacher options for a program. Hold down the SHIFT key, down arrow key ↓ , and the letter T key simultaneously. Release the keys and then press the ENTER key.
- The CLEAR key is pressed twice to exit a program. "Do you want to try again?" will then appear on the screen.
- Hold down the SHIFT key to enter upper-case letters.

MECC SERVICES

The Minnesota Educational Computing Consortium is an organization established in 1973 to assist Minnesota schools and colleges in implementing educational co ..puting. MECC provides a variety of services to education, including 1) development and distribution of computer courseware; 2) in-service training for educators and development of materials for conducting training; 3) instructional computing assistance through newsletters and computer purchase contracts; 4) technical support, including timeshare computing, hardware maintenance, and local area networking; and 5) management information services, including the development of statewide payroll and accounting software and administrative computer packages. MECC's knowledge and expertise in the educational computing field comes from a decade of working with and providing leadership for hundreds of local educators on a daily basis.

- MECC Educational Computing Catalog
  Catalogs containing instructional computing courseware, all-purpose training materials, and administrative software are published in March and September each year and are distributed at no charge. To request a catalog, write or call MECC Distribution (Telephone: 612/638-0627).

- **MECC Memberships**
  Non-Minnesota non-profit educational institutions may obtain annual service agreements with MECC which qualify them to obtain MECC courseware and training at special reduced prices. For up-to-date pricing and procedural information on these memberships, write or call MECC Institutional Memberships (Telephone: 612/638-0611).

- **Training Programs**
  MECC staff conducts educational computing workshops for educators throughout the United States. For information on workshop schedules, or to arrange a special training activity, write or call MECC User Services (Telephone: 612/638-0626).

- **MECC Newsletters**
  MECC distributes general information and technical newsletters on a regular basis during the school year. To obtain, write or call indicating your interest to MECC Newsletters (Telephone: 612/638-0606).

- **Help Line**
  If you have any problems using MECC courseware with your computer, write or call the Help Line (Telephone: 612/638-0638).

- **Visits**
  All requests for visits to MECC must be scheduled in advance by writing to MECC or calling 612/638-0606.

MECC
3490 Lexington Avenue North
St. Paul, MN 55112
(General Information: 612/638-0600)