APF™
IMAGINATION MACHINE

BASIC
Tutor.
MA-500

apf electronics inc.

BASIC TUTOR

Chapter 1
What is BASIC TUTOR
and How to Use It

This is not like any other ordinary book or computer program. BASIC TUTOR is designed to let you proceed exactly at your own pace and level. This program is then applicable for three distinct levels of understanding.

If you're NEW to PROGRAMMING, BASIC TUTOR will ease any fears you have of working with computers and computer language. WHY? Because the unique INPUT MONITORING SYSTEM will not only tell you what you did wrong but will tell you how to do it correctly. There is no reason to be intimidated, because BASIC TUTOR enables you to actually learn from your mistakes. Start with Chapter 2 and continue through Chapter 4 to learn the basics of using the Imagination Machine. Then continue with lessons 1-22 (Chapters 5-6) to see a sample of what software is and what your computer can do.

If you have TRIED SOME PROGRAMMING but have gotten lost with "those" other books, then BASIC TUTOR is for you. Each lesson gives you the details on a step-by-step basis of what's happening inside the computer. Get acquainted with the Imagination Machine in Chapters 2-4, then proceed with the lessons that interest you. If you already understand BASIC syntax and are familiar with the terms, BASIC TUTOR will give you help in pulling the pieces together for more insight on how to write a program. For example, we'll show you how to use a FOR-NEXT loop, we use it in Lesson 3 to show averaging. If you want to use color graphics to make motion, then go to Lesson 18 and watch Snoopy fly an airplane. Are you looking to make music and edit it? Try Lesson 21. As you will see, the learning possibilities are endless and we will carefully show you how to bring together those BASIC words you know into your own unique programs.

If you're already an EXPERIENCED PROGRAMMER, then BASIC TUTOR is for you. It will be your library resource of

programming ideas to build upon. Even experienced programmers, need to learn by looking at and comparing other people's programs. Just scan through the examples and see how the Imagination Machine handles graphics, color and sound. Whenever you need a particular function, like moving an object on the screen with the hand controllers, just turn to one of the examples and see the actual sequence of instructions required. Get acquainted with the APF BASIC through the dictionary section. Besides detailed definitions and examples, the dictionary will direct you to a lesson with a sample program.

Chapters 2, 3 and 4 give the fundamental basics of the Imagination Machine and its language.

Chapter 5 contains 12 lessons which will introduce you to the necessary keywords and functions of APF BASIC. In Chapter 5, all lessons incorporate the unique INPUT MONITORING SYSTEM, which corrects an error if you make a mistake.

Chapter 6 contains 10 lessons which are more complex than those programs in Chapter 5. The intent of this chapter is to give you examples of more elaborate programming and to introduce some more new keywords.

In both Chapters 5 & 6, lessons are written to teach by example. Each lesson contains a working and usable program. When loaded and run, the program gives a detailed explanation of what is happening and why.

Once you've mastered the BASIC language, Chapter 7 will give you an idea how to create and write your own programs.

Finally, Chapter 8 gives a complete dictionary of terms.

Chapter 2

THE IMAGINATION MACHINE SYSTEM

SYSTEM HOOKUP

Your Imagination Machine is comprised of an MP-1000 console and a computer console. These two join together and then connect to your television. Your Owner's Manual contains the instructions for hooking everything up on pages 4-11. Whenever you want to use your computer, bring it to its Ready State and have the cursor on the screen.

SYSTEM DESCRIPTION

Before proceeding with learning about BASIC, let's take a look at what is called the "HARDWARE" of the system.

The television is how the Imagination Machine "speaks" to you. It is on your television screen that all messages, questions and pictures from the Imagination Machine will be seen.

The keyboard is how you will speak to the computer. You will answer questions by typing on the keyboard, and all your input to the computer will come through the keyboard. The keyboard consists of 53 keys. Although most are similar to a typewriter, there are some differences which we will review.

LETTER KEYS A-Z: Your Imagination Machine deals only with upper case letters. Pressing any letter key will cause an entry of the upper case letter only.

NUMERAL KEYS (0-9): Pressing any of these inputs a number. Note there is a 0 and 1 key. Never use the letter key "O" for zero or letter key "L" for a 1.

SHIFT KEY: Operating similarly to a typewriter, the shift key is used to enter the symbols above several numeral and letter keys. Examples: To enter the %, press the shift key first, HOLD IT DOWN and then press the key with numeral 5. Remember, there is no shift lock.

3

GOSUB		LET	DATA	INPUT	DIM	FOR	RETURN	THEN	ON	GOTO	TO	RESTORE	
	IF	STEP	STOP	READ	NEXT	PRINT	END	LIST	REM	USING	CSAVE	CLOAD	
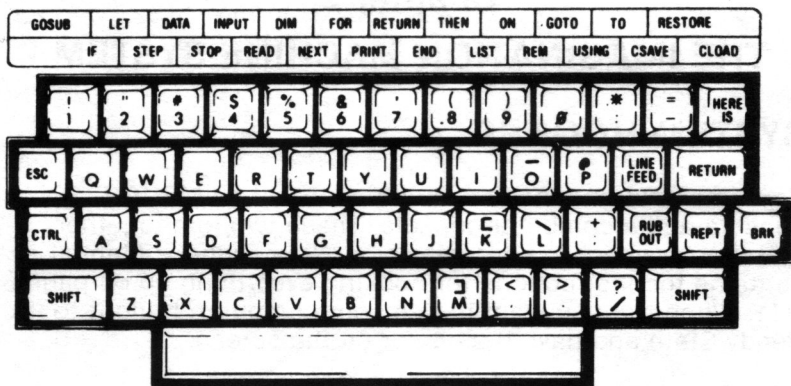
If you would like to try out the keyboard, take a break and go ahead. After power-up, remember to press RESET. Hit the EN button on either hand controller and you will see the blue cursor appear — you're all set to type. You will see the entries on your TV and the cursor will move along after each entry. Try the shift key, too.

RUBOUT: This is like a backspace. Unlike a typewriter, we can backup and erase the last entered key very easily. Press some keys, They should appear on the screen. Now press rubout. The last one disappears. Press rubout again. You can backup the cursor to the beginning of the line.

RETURN: This is a special key. It is like a typewriter Return Key but does more than move the cursor to the beginning of the next line. When you press RETURN, it tells the Imagination Machine to look at what you have just keyed in. It's like telling the computer "I'm finished. It's your turn."

BRK: Break Key. When the computer is "running," and a program or the cursor is not on screen, pressing the Break Key will tell it to stop and return control to the keyboard. If this does not work, then something else has gone wrong and you will have to press the Reset Button.

CTRL: This is like another Shift Key. You will notice an inlay plate above the top row of keys. Those words are "BASIC

4

Keywords." When you want to enter one of those keywords, you can do so by pressing the CTRL Key and one of the keys on the top 2 rows. The upper line of words corresponds to the top row of keys and the lower line of words corresponds to the row of keys below. Press CTRL Key, hold it down and press Y. The computer will automatically enter the word PRINT. Continue to hold down CTRL Key and press 6 key, the computer will automatically enter the word FOR.

ESC: Escape Key. Primary use in BASIC TUTOR is in Chapter 5 to load programs into computer without typing them in. The Escape Key is also used for editing programs, as described in Chapter 7.

LINE FEED: This is not used in the present version of the system.

HERE IS: This is not used in the present version of the system.

THE TAPE DECK

Below is a picture of the built-in tape recorder on the Imagination Machine. This is used to load and save programs.

Digital Tape Counter

Built-In Program Recorder

Audio Microphone Jack

Cassette Volume Control

Built-in Speaker for Audio from Tape
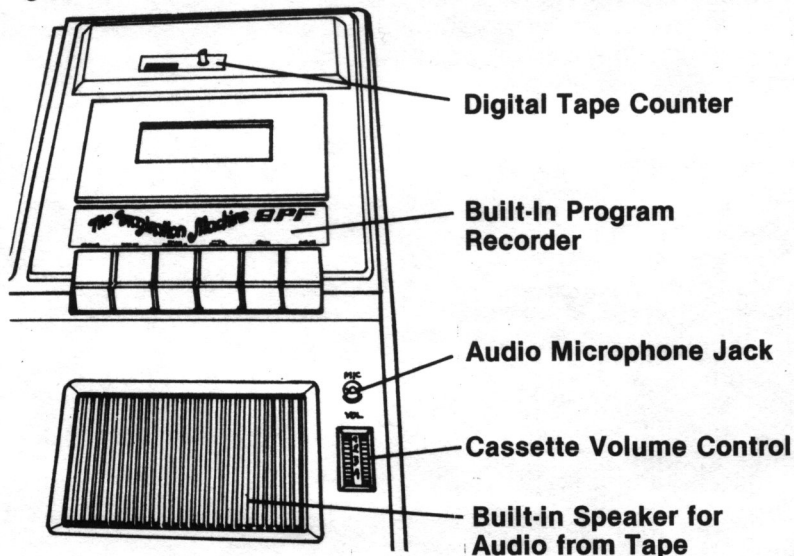
Chapter 3
GETTING STARTED

When people communicate, they do it through language.
Unfortunately, computers do not understand English.
Therefore, you are about to embark upon learning a new
language that computers recognize. Relax! This will be far
easier, fun and more useful than learning French, Spanish or
even Japanese.

The most common universal language of computers is
called BASIC. The original BASIC was designed at
Dartmouth College in 1963. Since then, there have been
many variations or dialects of BASIC but this manual will
only discuss the Imagination Machine dialect.

BASIC uses symbolic words to tell the computer what to do.
Some of these are regular English words (such as PRINT, IF,
FOR) and you probably have a general idea of what they
mean. Others are modifications of English words (like
GOTO) and you probably have an idea of what they
represent.

Once you learn the words, you then have to learn to make
grammatically correct statements in BASIC. That is,
statements which the computer can understand and
execute.

WHAT COMPUTERS ARE GOOD AT

There are basically 3 things computers are good at and you
should always keep them in mind.

1. Following the instructions you programmed.
2. Doing calculations far faster than is humanly possible.
3. Filing items and being able to find them quickly.

Once you learn the BASIC language, you can have the
Imagination Machine perform these tasks.

"HANDS-ON"

The first thing you need to know about computers is that you won't hurt anything by typing on the keyboard, even if you make lots of errors. If you turn off the power or press RESET, you will lose the information entered, but the computer will be unharmed.

You will learn how to program your computer by experimenting with it. If your computer is not in its ready state, then insert the BASIC Cartridge, turn on the computer, press **RESET**, and press the **EN** Key on the left keypad.

The blue square on your TV screen, which is called a cursor, shows you that the computer is waiting for you to type an instruction on the keyboard.

Type **HELLO** and press the **RETURN** Key. If you make a mistake, use the **RUBOUT** Key to back up and correct it.

The computer printed **WHAT** on the screen because **HELLO** is not one of the words used in BASIC.

> **WHAT** is an "error message" — when the computer tells you that you've done something wrong. A complete list of error messages will be found in Appendix A at the back of the book for your reference, after you become more familiar with programming.

PRINTING CALCULATIONS

One of the BASIC words the computer understands is **PRINT**, and you can tell the computer to print on the screen.

Let's try a calculation first:

PRINT 1 + 2 then **RETURN** Key
3

The computer printed 3 on the screen, which is the result of 1 + 2. Notice that we did not type **PRINT 1 + 2 =** because in the 1 + 2 calculation, the **RETURN** Key functions like an

8

equals key. The computer is a very high powered calculator that figures results of problems instantly using algebraic logic, and uses the **RETURN** Key as an equals key for calculations.

> If you type **PRINT** and then an arithmetic problem, followed by the **RETURN** Key, the computer prints out ONLY the result on the line after the problem, not the entire problem.

Note: When keywords such as **PRINT** are used in this manual, you can either type them in one letter at a time, or you can use the **CTRL** Shift Function. Simply press **CTRL** Key and the **Y** Key and **PRINT** appears on screen — try it. Hold down **CTRL** Key, press **Y** then type **1 + 2 RETURN**.

TRY SOME MORE

To show multiplication problems, use the (*) key and to do division use (/) key. Remember to press **RETURN** after each entry.

You Type	The Computer Prints
PRINT 5 − 3 press **RETURN**	2
PRINT 2 + 4 press **RETURN**	6
PRINT 3/2　 press **RETURN**	1.5000
PRINT 5*12 press **RETURN**	60

PRINTING WORDS

You might want to print something that isn't a mathematical computation.

To do this, we enclose what we want printed in a pair of quotation marks (the " key is above the 2 key).

Type the following:

PRINT "HELLO" then press **RETURN** Key.

The computer printed **HELLO** on the screen because it was

in quotation marks.

> **PRINT "1 + 2"** then **RETURN** Key
>
> This time the computer printed 1 + 2. We can enclose anything in quotes and the computer will print it, but when you enclose a calculation in " " it does NOT print the result.

Now try some other examples — remember to hit the **RETURN** Key after each message.

Type **PRINT "GOODBYE"**

Type **PRINT "MY COMPUTER LOVES ME"**

> Remember — if you want something to be printed on the screen you must use quotation marks (") to enclose the words you want to appear.

MULTIPLE PRINT STATEMENTS

(Printing more than one item)

To ask for more than one item to be printed without repeating the word **PRINT**, we separate the list of items we want printed with what is known as a delimiter.

> A delimiter is something that sets the limits of or marks the boundaries. The two delimiters used in BASIC for a print statement are a comma (,) and a semicolon (;).
>
> The difference between (,) and (;) is:
>
> (;) **PRINTS** the items one right after another with no spaces between them.
>
> (,) **PRINTS** the items into preset columns (like a preset tab on a typewriter). On the Imagination Machine, there are 4 columns and they are 8 spaces wide each.

Let's try some examples to illustrate this.

Type **PRINT "HELLO", "IMAGINATION MACHINE"** and press **RETURN** Key

The Imagination Machine will put the results on the screen in preset column form. Now try typing the same thing with a semicolon.

Type **PRINT "HELLO"; "IMAGINATION MACHINE"** and press **RETURN** Key

The computer put the same information on the screen but without spaces between the items you typed in.

Let's try another example. Type

PRINT "HELLO" "IMAGINATION MACHINE"

The Imagination Machine has put an error statement on the screen — **PRINT DELIMETER** — this means you did not put a proper delimeter (; or ,) between the elements you wished to be typed in and the computer automatically informs you that a mistake has been made.

Now let's try some examples with numbers.

Type **PRINT 5 − 3, 2 + 4, 3/2, 5*12** then press **RETURN** Key

The results will appear once again in preset columns

2   6   1.5000   60

Try it with semicolons instead of commas, type:

PRINT 5 − 3; 2 + 4; 3/2; 5*12 press **RETURN**
261.500060

As you can see, there are no spaces printed between the results using semicolons.

STORED AND IMMEDIATE MODE

Up until now, we have been giving the Print Command in what is known as the **"IMMEDIATE MODE."** You type the command, press RETURN, and the computer looks at your request and then immediately does it. It's quick and efficient, but if you want any of these commands to be done again, you will have to retype them.

Let's go on to the **STORED MODE.** To tell the computer to store a statement for usage later instead of immediately doing it, we simply precede the statement with a number. There are certain criteria for using this number: it must be a positive number (no negative sign), have no decimal point and be between **0** and **9999.** Let's try something, type:

10 PRINT 5 + 4 then **RETURN** Key

What happened? It looks like nothing. Well, the computer recognized that we wanted the statement stored in its memory and not executed now.

How do we get it back, and how do we get the computer to do it? Let's introduce two **SYSTEM COMMANDS.**

The first is **LIST.**

> **LIST** — instructs the computer to print all stored statements it has in its memory.

Type **LIST** and hit **RETURN** Key
10 PRINT 5 + 4

The computer printed on the screen the Statement 10 that you entered before. It might not be in the exact format as you entered it, since the computer re-formats statements when it lists them for easier readability and uniformity.

The second System Command is **RUN.**

> **RUN** — tells the computer to go find the stored statements and execute them. It will execute the lowest numbered step first, and each statement in sequence.

12

Type **RUN** — the display should show a **9**. This is the result of Statement **10** that is stored.

Type **LIST** — the program statement is still there.

Type **RUN** — it prints **9** again since Statement **10** is still stored. Next, let's try adding more statements.

> To add a statement, type the new statement number followed by the statement.

Type:

30 PRINT "I AM THE IMAGINATION MACHINE"

Type:

20 PRINT "I LOVE YOU"

Type:

LIST and press **RETURN**

10 PRINT 5 + 4
20 PRINT "I LOVE YOU"
30 PRINT "I AM THE IMAGINATION MACHINE"

All 3 steps are there, but not in the order you entered them. Steps **20** and **30** have been reversed. That's because the computer places steps in memory, not in the order you entered them but in ascending order of their statement numbers. Statement **20** comes after **10** and before **30**.

Now let's run the program. Type **RUN** and the computer will printout on the screen:

9
I LOVE YOU
I AM THE IMAGINATION MACHINE

These are the results of each of the 3 statements in our program. Next let's change line **10**.

> To CHANGE A LINE type the line number and the new statement and **RETURN**. To DELETE A LINE type the line number and **RETURN**. The computer will automatically replace the existing line.

Type **10 PRINT "HI _____"** instead of _____, type your name. Now type **RUN**

The computer will print to the screen:

HI (YOUR NAME)
I LOVE YOU
I AM THE IMAGINATION MACHINE

If it did the above, you have now entered and changed your first computer program. Congratulations!

Now let's learn a little more about the **LIST** Command.

Type **LIST** — you will see all 3 lines of the program.

10 PRINT "HI Your Name**"**
20 PRINT "I LOVE YOU"
30 PRINT "I AM THE IMAGINATION MACHINE"

> Type **LIST 20** — only Line **20** will be listed
> Type **LIST 10,2** — this means list starting at Line **10**, two lines (so lines **10** and **20** will be listed).

THE GOTO COMMAND

We are ready to learn another important BASIC Command. It is **GOTO**, which is a grouping of 2 English words to make one. The grouping of words is less for us to type in and less for the computer to try and understand. This keeps communications simple with the computer.

Normally the program starts from the lowest numbered step, then second lowest number and continues until it reaches the end of the program.

The **GOTO** command allows us to change the normal sequence of statement execution.

Add Line 25 (to add a line, just type in the line and **RETURN** Key). Type **25 GOTO 10**.

List the program and study it. Type **LIST**. The TV screen will show

10 PRINT "HI (YOUR NAME)"
20 PRINT "I LOVE YOU"
25 GOTO 10
30 PRINT "I AM THE IMAGINATION MACHINE"

The program will go from step **10**, to **20**, to **25** and since **25** says **GOTO 10,** it will go back to line **10** instead of continuing to line **30** and keep repeating the procedure. Try a **RUN** command.

The program runs and keeps repeating the two Print Statements. It runs very quickly and you probably won't be able to read the message.

To stop the program we press the **BREAK** Key (its right below the **RETURN** Key).

Now look at what is on the screen. It says:

HI (YOUR NAME)
I LOVE YOU
HI (YOUR NAME)
I LOVE YOU .....

and ignores Line **30** completely. This is because Line **25** says go back to Line **10** and the sequence starts again. What happens on the screen when this program runs is called SCROLLING.

SCROLLING

When the screen fills up with 16 lines, it has to make room for more. It moves the bottom 15 lines up, and puts the next print output on the bottom line. This is called SCROLLING.

To stop the scrolling, we press the BREAK Key.

> **BREAK KEY** — interrupts the computer and instructs it to return to the immediate mode. This is another System Command like **RUN** or **LIST**.

Up to this point we have learned:

IMMEDIATE AND STORED MODE
BASIC COMMANDS: PRINT, GOTO
SYSTEM COMMANDS: RUN, LIST, BREAK

Go on to the next chapter. We will continue with some more elementary rules and procedures. If you are not clear on any of the above, then re-read this chapter before continuing.

DICTIONARY

This dictionary contains more detailed and comprehensive definitions of the keywords covered in this Chapter. If you wish to learn more about these keywords and their variations, continue reading this dictionary section. If not, please go on to Chapter 4

BREAK — Interrupts the computer and tells it to return to the immediate mode. This is a System Command which allows keyboard interruption of the computer's program.

DELIMETER — Something that sets the limits of or marks the boundaries. The two delimeters used in BASIC for a Print Statement are a , and a ; Semicolon prints the items one right after the other with no spaces between them while a comma prints the items into preset columns. There are 4 columns that are 8 spaces each on the Imagination Machine.

END — Terminates program execution and returns cursor to screen. This informs the computer that complete program is over, as opposed to STOP command (see Lesson 1) which can be used at lower line number within program and is often used to stop execution if certain things are true.

ERROR MESSAGE — The computer tells you you've done something wrong. A complete list of error messages will be found at Appendix A at the back of this book. Examples include WHAT (where the computer doesn't under a command), ARITHMETIC OVERFLOW, etc.

GOTO — A BASIC Command which allows you to change the normal sequence of statement execution. Instead of following sequential statements, the program jumps back or goes to a specified line number.

IMMEDIATE MODE — Tells the computer to execute a command at once rather than store it for later use. Once you type the command, you have to retype it for the computer to do it over again.

LIST — A System Command that is used to display your program. Each program line will appear in numerical order starting at lowest step.

Example:

LIST displays all statements in sequential order.
LIST 60 will list Line 60 only, or if there is no Line 60 in program will list first line number after Line 60.

Another example:

LIST 60, 3 will list 3 lines starting at line 60.
LIST, 3 lists first 3 lines of program
LIST 9000, lists starting at Line 9000 thru remaining lines of program

PRINT — A BASIC Command which tells the computer to display on the screen something you have enclosed in quotation marks or to display a variable value or the results of an arithmetic problem. **PRINT "HELLO"** puts HELLO on the screen. **PRINT A** when dealing with the variable A puts the value of A to the screen when inputted correctly.
PRINT 2*6 puts 12 to the screen (the results of a calculation).
PRINT 1 + 2, 4 − 3, (comma) tells the computer to do multiple print statements and results will be in pre-set columns.
PRINT "HI"; "IMAGINATION MACHINE"; (semicolon) tells computer to do multiple print statements and results appear on screen right after each other without any spacing between them.
**PRINT A$** prints the value of the textual variable called A$.

Example:

```
10  PRINT "IS THIS PRINTED";
20  PRINT "ON THE LINE"
```

Computer prints **IS THIS PRINTEDON THE LINE** because ; tells the computer not to move cursor after each print statement.

Type:

```
10  PRINT "IS THIS PRINTED",
20  PRINT "ON THE LINE"
```

Computer prints:

IS THIS PRINTED   ON THE LINE

because the , tells the computer to put the message on the screen in preset columns in one line.

Another example:

```
10  PRINT 123
20  PRINT
30  PRINT 456
```

Computer prints:

123
(blank line)
456

RUN — A System Command which tells the computer to find the stored statement it has in its memory and execute them starting with the lowest line number. It also tells the computer to clear the values of all variables to 0.

STEP NUMBER — Precedes all programming statements that we wish to place in the computer's stored memory. Must be a positive number, with no negative sign. It must also have no decimal point and be between 0 and 9999.

STORED COMMANDS — Statements which the computer stores for usage at a later time instead of taking immediate action and executing them at once. It's very elemental to programming that a stored command be preceded with a step number.

Chapter 4

ELEMENTARY DEFINITIONS AND RULES

PROGRAMMING

All machines require that we follow a set procedure of rules to make them work for us. This procedure or sequence of steps is a program. A program for using the telephone, for example, could look like this:

1 Lift telephone receiver
2 Wait for dial tone
3 Dial seven digits for the number
4 If a person or machine answers, then begin conversation
5 If no answer after ten rings, then hang up
6 When conversation is finished, hang up

What we want to do with a computer is to give it a procedure or sequence of steps (a program) that it should follow.

In the above example we used English structured sentences. If we do that with the Imagination Machine, it will not understand what we want. We have to use BASIC structured sentences or statements.

In the telephone example above we numbered each step. We must also do this in BASIC. However, instead of using consecutive numbers, we use every tenth number (10, 20, 30, etc.).

FLOW CHARTS

In designing or writing a program we have to take the idea of what we want to happen and break it down into the procedure of steps required. In the telephone call example, we broke the idea of making a call down into a procedure of 6 steps to be followed. We wrote these steps down (1 per line) in English-type sentences.

Another form of breaking down the ideas is to use what is called a FLOW CHART.

> A FLOW CHART is a pictorial representation of the steps and the sequence they follow. A FLOW CHART is easier to follow than a written explanation.

We make extensive use of Flow Charts in lessons throughout this book.

In a Flow Chart we use 2 types of boxes. One is called an ACTION BOX and the other a DECISION BOX.

An Action Box describes a statement that should do something. It has one entry and one exit.

ENTER

```
┌─────────────────────┐
│     ACTION BOX      │
│   (DO SOMETHING)    │
└─────────────────────┘
```
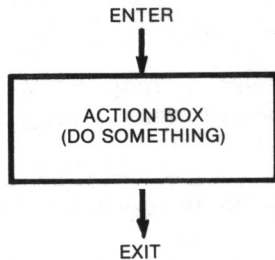
EXIT

A DECISION BOX describes a statement that can exit through one of two paths depending upon a certain condition being true or false. It has one entry and two exits.

ENTER

FALSE ◄───── DECISION ─────► TRUE
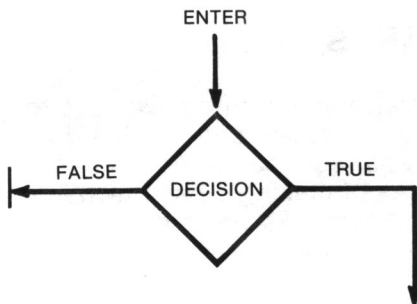
In the Flow Chart, we connect the boxes with lines showing the possible flows from one box to another.

Let's look at the Flow Chart for the telephone call.
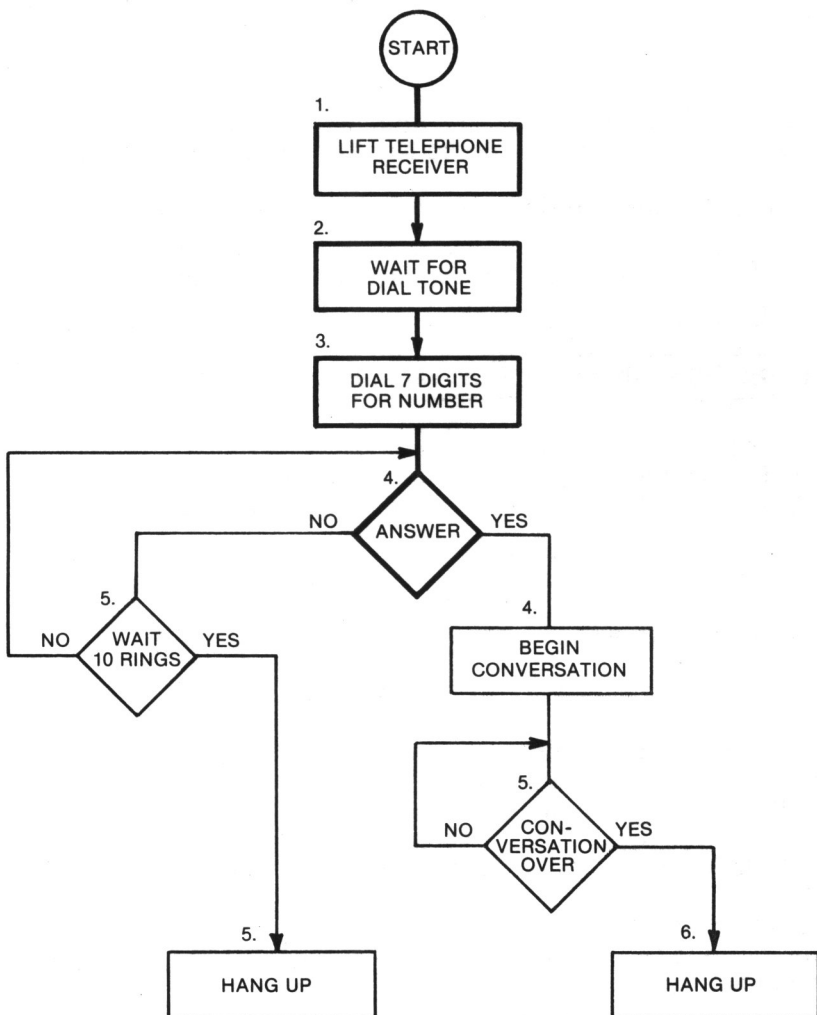
EQUATIONS AND VARIABLES

Although it does sound like a complicated mathematical course, there's no reason to be afraid or overwhelmed. We are going to keep things simple.

> An EQUATION simply states the procedure or steps followed to achieve a result.

An example:

The Cost to fill up a car's gas tank is equal to the Price per gallon of gas multiplied by the Number of gallons to fill up.

Cost to fill tank = Price per gallon x Number of gallons.

> A VARIABLE is something whose value can change or vary.

> In the above there are three variables:

1) The price per gallon can vary.
2) The number of gallons to fill up can vary.
3) and consequently the total cost to fill up would vary.

> A symbol is a shortcut way of designating a variable.

Now let's rewrite the equation above using symbols.

$$COST = PRICE*NUMBER$$

= is the symbol for equals
* is the symbol for times or multiplied by
COST is the symbol for total cost to fill up.
PRICE is the symbol for Price per gallon of gas.
NUMBER is the symbol for the Number of gallons to fill up.

For example, that means you don't have to type in "cost to fill up the tank" — COST becomes the symbol for the entire meaning.

ASSIGNMENT

What we want to do with variables is assign a value to them. In BASIC, we do this with an ASSIGNMENT STATEMENT.

> An ASSIGNMENT STATEMENT, as its name implies, assigns or gives a value to a variable. The value could be a number, a result of a calculation, a name or a word.

An Assignment Statement always has an EQUAL SYMBOL (=) in it. On the left side of the equal sign is a variable name. On the right side is a "value" we want assigned to that variable name. For further explanation, let's program the gasoline cost example which we used on the preceding page.

Type:

10 PRICE = 1.3
20 PRINT PRICE

TYPE **RUN** and hit **RETURN** Key and you will see the number 1.3000 printed on the screen. This is the result of statement **20** which printed the result of the ASSIGNMENT STATEMENT that you made in Line **10**. Let's continue with the program.

Type:

30 NUMBER = 18.4
40 PRINT NUMBER

Now let's try running the program so far. Type **RUN**, hit **RETURN** Key. You will see 1.3000 and 18.4000. We have, in effect, told the computer that whenever it sees the variable PRICE, to use 1.3000 in its place and the variable NUMBER to use 18.4000 in its place.

Let's continue once more with the program. Type:

50 COST = PRICE*NUMBER
60 PRINT COST

Now let's try running the program. Type **RUN** and hit **RETURN** key and you will see that the cost is 23.92.

STATEMENT 50 was another ASSIGNMENT STATEMENT. First, the computer went to the right side of the equal sign and instantly computed a value by doing the calculation of Price*Number. When it gets that value, it assigns it to COST. STATEMENT 60 is a PRINT STATEMENT. It won't print the letters "C-O-S-T" because they are not in quotes. Instead it recognized COST as a variable name, goes and finds its value (which it computed in Statement 50) and prints its value.

In some BASIC Assignment Statements, the word LET can appear before the variable name. In the Imagination Machine APF BASIC, this is not necessary. The Imagination Machine recognizes the Assignment Statement with or without the word LET.

VARIABLE RE-ASSIGNMENT

We can also alter or calculate a new value assigned to the variable.

Since the proper way to look at an Assignment Statement is the value on the right side is assigned to the variable name on the left side; changing the value on the right side automatically changes the value which the computer gives to the variable. For example, type:

```
10 COST = 5
20 COST = COST + 1
30 PRINT COST
```

Now type **RUN** and press **RETURN**. You will see the COST is 6, for the Imagination Machine automatically reassigned a new value to COST by your statement in Step **20** which takes the old value of Cost and adds 1 to it:

```
20 COST New = COST Old + 1
```

VARIABLES NAMES

In the example we made up symbols for the three variables in our equation (Cost, Price and Number). There are only 3 restrictions on variable names in the Imagination Machine.

1. Variable names must start with a letter (A-Z)
2. A variable name can be up to 15 characters long, but cannot have any of the keywords used in BASIC contained in them. A list of keywords is in Appendix B. If we had used the word TOTAL instead of COST, it would be a problem since TO is a keyword in BASIC.
3. Although you can use up to 15 letters for a variable name, the Imagination Machine likes things easy so it only remembers the first two letters. This means that two variables, such as COST and COLD, would look the same to the Imagination Machine.

Changing the value of one would change the other.

For example, type:

10 COST = 100
20 COLD = 110
30 PRINT COST, COLD

TYPE **RUN**

The computer will print on screen

110   110

> The Imagination Machine always looks at the last assignment done for each variable name — and it reads both Cost and Cold as the same variable.

DICTIONARY

ASSIGNMENT — A statement, as its name implies, that ASSIGNS or gives a value to a variable. The value could be a number, a result of a calculation, a name or a word. An ASSIGNMENT Statement always has an equal symbol (=) in it. The variable name is on the left side of the equal sign and the value we want assigned to the variable is on the right side of the equal sign. In some BASIC ASSIGNMENT Statements, the word LET can appear before the variable name. In the Imagination Machine BASIC, this is not necessary, since the Imagination Machine recognizes the ASSIGNMENT Statement with or without the word LET. A new value for a variable can be altered or calculated by changing the value on the right side, which automatically changes the value which the computer gives to the variable. This is known as a Variable ASSIGNMENT. See VARIABLES in this dictionary section.

EQUATION — Stating the procedure or steps followed to achieve a result. See examples in Chapter 4, page 24 on gasoline costs.

FLOW CHART — Visual programming aid that is a pictorial representation of program statements showing them in sequence. A Flow Chart contains ACTION BOXES that describe a statement that should do something (with one entry and one exit); and DECISION BOXES that describe a statement that can do one of two things depending upon a condition being true or false (one entry and two exits). Boxes are connected with lines showing the possible flow from one box to another. See examples in Chapter 4, page 22.

LET — The word LET can appear before a variable name in an ASSIGNMENT Statement. However, in APF BASIC, this is not necessary since the Imagination Machine recognizes the ASSIGNMENT Statement with or without the word LET. Example:

10 LET PRICE = 1.3   The word LET is optional.

28

SYMBOLS — A shortcut way of designating a variable. In the equation Cost = Price*Number, Cost becomes the SYMBOL for "cost to fill up the tank" in an equation to compute gasoline expenditures. SYMBOLS save time since they shorten variable designations.

VARIABLES — Something whose value can change or vary. Usually we use a symbol as a variable and assign a value to it. There are two types of variables; number variables or letter variables. The maximum number of variable names allowed in APF BASIC in any one program is 26.

Variable Names can be up to 15 characters long but only the first two characters are used. Names cannot contain a BASIC keyword.

Number Variables — can have a value up to 13 digits long ranging from +999,999,999.9999 to −999,999,999.9999.

Letter Variables — the value of a letter variable can contain up to 100 characters. The maximum length or number of characters a letter will have must be defined with a dimension statement (see Lesson 8). The first character must be a letter. Letter Variable names must end with dollar sign ($). Example of correct variable names: A, A$, A1, A1$, A2SAM, A2SAM$. Examples of incorrect variable names: 3AL, A$B, ~~A3$~~.
3A$

Chapter 5
TAPE ONE PROGRAMS (lessons 1-12)

Tape One of Basic Tutor contains 12 lesson-programs that show you how to use your computer to create electronic games, do arithmetic, compute loans and interest rates, and even make your own video art. Each lesson also takes one or more of the keywords in BASIC and shows how it is used. Each lesson contains a program you can type into the computer or load automatically. Instructions are given on how to run each program and what the results are. Each lesson also contains flow chart diagrams and more detailed explanations of each step in the program. Finally, there is a dictionary section explaining in detail each keyword introduced in every lesson.

SPECIAL INPUT MONITORING MODE

The 12 lesson programs on Tape One contain a special and unique monitoring mode. When a lesson is loaded from tape and started (by a RUN command), you are no longer free to do what you want with the Imagination Machine. Each lesson is designed for you to study a specific program. Because these lessons might be the first time you have tried to enter and run a program, we are going to carefully monitor your inputs to make sure you are doing everything correctly. If you try to type a statement not in the selected lesson, we won't accept it. The message will be: **THIS LINE IS NOT IN THE PROGRAM**. If you try to run the program lesson before completely entering it, we won't let you. If you type in a statement line and type it wrong, we'll tell you what you've done wrong and how the line should read.

We let you try again (and again) until you get it right. We want to help you. You must learn to be able to read a program from a book and type it into the machine in order to learn how to write your own programs in BASIC.

DICTIONARY MODE

If at any time while you are typing in program steps you

have a question on the meaning of a BASIC word, just press the **?** key. You'll see a selection of words that are in the computer memory. By typing in one of those words followed by another **?** you'll see a definition and brief explanation to remind you of the meaning. See dictionary pages at the back of each lesson for a more detailed explanation.

AUTO ENTRY MODE

If you don't want to take the time to type in the individual program steps in each lesson, or just want to see how each lesson runs, we've made it very easy for you. After starting lesson with RUN Command, just press the **ESC** (Escape Key on the left hand side of the keyboard) and then the **RETURN** Key. The program will be automatically typed in for you and let you list the statements or RUN the program. After a lesson is typed in, you simply type **LIST** and hit the **RETURN** Key and the computer will show you the program. Type **RUN** and press the **RETURN** Key and the program will run for you.

GENERAL LOADING INSTRUCTIONS

1. With the APF BASIC cartridge inserted into the computer console cartridges slot, turn on the MP-1000 and your Imagination Machine Console. Press the **EN** Key on the hand controller and press the eject button on built-in cassette player to open the cassette player cover.
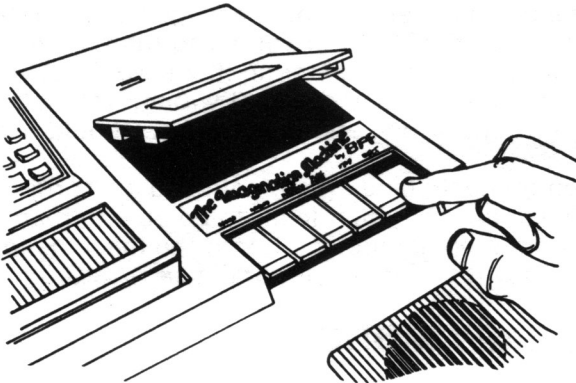
2. Insert the cassette tape with Lessons 1-12 in the cassette player with the cassette label facing up. Close the cassette deck cover.
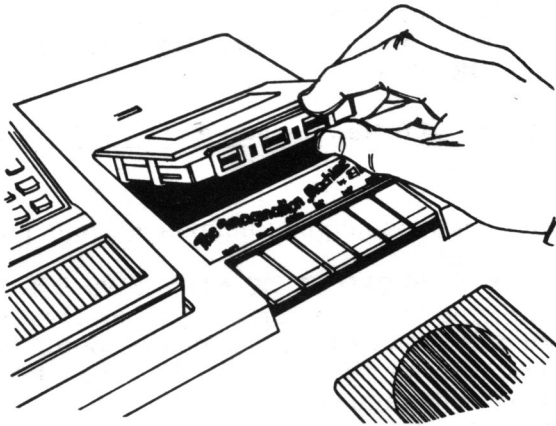
3. Type **CLOAD** and press the **RETURN** Key.

If You Are Starting With Lesson 1

This message will appear on your screen: **REWIND TAPE, PRESS PLAY THEN RETURN KEY.**

A) Rewind the tape to the beginning for Lesson 1 only. Push digital tape button above the cassette player so that it reads 000.
B) Press the **PLAY/SAVE** button on the tape player.
C) Adjust the volume control next to the speaker on the keyboard and listen to the introduction on Lesson 1.
D) Press the **RETURN** Key when you hear the BEEP announcing the start of Lesson 1.
E) When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key.

If You Are Continuing With The Next Lesson And Your Previous Program is still Running

A) After you type **CLOAD** and press the **RETURN** Key, the message on the screen will say **PRESS PLAY THEN RETURN KEY.**

B) Press the **PLAY/SAVE** button on the tape player.

C) Adjust the volume control next to the speaker on the keyboard and listen for the introduction to that lesson.

D) Press the **RETURN** Key when you hear the BEEP announcing the start of the Lesson.

E) When the word **OK** appears on the screen, type **RUN** and press the **RETURN** Key.

If You Want To Go To A New Lesson And Your Tape Is Not In The Machine

A) Follow the general loading instructions above in points 1, 2 and 3.

B) After you have typed **CLOAD,** DO NOT REWIND TAPE as you are instructed to on the screen. Instead, press the **PLAY/SAVE** button and adjust the volume control next to the speaker on the keyboard and listen to the introduction for the lesson. If it is the lesson you want, press the **RETURN** Key. If it is not the lesson you want, DO NOT PRESS THE RETURN KEY. Use the Fast Forward mode to advance the tape to the correct audio track that will instruct you on when to press **RETURN** for the lesson you want.

C) When you have reached the introduction to the lesson you want, press the **RETURN** Key when you hear the BEEP announcing the start of the lesson.

D) When the word **OK** appears on the screen, type **RUN** and press the **RETURN** Key.

Lesson 1

MATHEMATICIAN
INPUT, STOP, PRECEDENCE

With this program you can perform the 4 basic math functions (Addition, Subtraction, Multiplication, Division) with any 2 numbers. It will demonstrate how fast a computer works and how it performs math functions automatically after given sufficient input.

LOADING INSTRUCTIONS

Follow the steps for loading BASIC TUTOR Lesson 1 in Chapter 5, page 32. After you type **CLOAD**, remember to rewind tape all the way to the beginning for Lesson 1 and press the digital tape counter above the cassette portion of the Imagination Machine. This will bring the tape counter to 000. After Lesson 1 has loaded, the word **OK** will appear on the screen. The BASIC TUTOR tape will stop automatically. LEAVE THE TAPE IN THE MACHINE. Type **RUN**, press **RETURN** Key, and you can begin Lesson 1.

REMEMBER

1. This Lesson has a special input monitoring mode which is in control of the machine. It will help you if you make a mistake, but it will also prevent you from typing in extraneous material not in your program. If you attempt to type in such material, the machine will give you an error message **THIS LINE NOT IN PROGRAM**.
2. If you have a question on any of the BASIC words, remember to use the dictionary mode. Simply hold down the **SHIFT** Key and press **?** A list of BASIC keywords will appear. Type in the keyword, hold down the **SHIFT** Key once again and hit the **?** key. BASIC TUTOR will give you a brief definition of that word.
3. If you do not wish to type in the entire program at this time, or you merely want to see the program run, hit **ESC** Key (the Escape Key) and the **RETURN** Key and the computer will automatically type in the program for you.

ENTER MATHEMATICIAN

Now type each line of the program as listed below. Press **RETURN** Key at the end of each line.

```
10      INPUT  "NUMBER A=",A
20      INPUT  "NUMBER B=",B
30      PRINT  "A+B=";A+B
40      PRINT  "A-B=";A-B
50      PRINT  "A*B=";A*B
60      PRINT  "A/B=";A/B            NO SPACE
70      INPUT  "GO AGAIN ( 0=NO, 1=YES )",K
80      IF  K=1  GOTO 10
90      STOP
```

HOW TO USE MATHEMATICIAN

After you finish entering your program into the computer,
type **RUN** and press **RETURN** Key. The computer asks for an
input number for A. Type **8** *and press* **RETURN** Key. It asks
for an input number for B. Type **4** and press the **RETURN**
Key. It will show

A + B = 12
A − B = 4
A*B = 32
A/B = 2

It asks the question if you want to go again. Type a **1** and
press the **RETURN** Key and let's try a second example. For
the input number for A, type **9**. For the input number for B,
type **0**. The computer will show

A + B = 9
A − B = 9
A*B = 0
A/B =

The computer will give you an error message for A/B:

Line 5102 Division By Zero. This means the computer is
telling you that A/B (9/0) is an impossible calculation.

TO GET BACK TO THE BASIC TUTOR PROGRAM AFTER
THIS ERROR MESSAGE TYPE **RUN** AND PRESS THE
RETURN KEY. THIS ELIMINATES THE ERROR MESSAGE
AND BRINGS YOU BACK TO THE BASIC TUTOR MODE. TO
START TO RUN YOUR PROGRAM AGAIN, TYPE **RUN** AND
PRESS THE **RETURN** KEY A SECOND TIME. THIS
RESTARTS LESSON 1. YOU DO NOT HAVE TO RE-ENTER
THE PROGRAM.

Let's try a third example. For the input number A, type **1**. For the input number for B, type **3**. The computer will show

A + B = 4
A − B = − 2
A*B = 3
A/B = 0.3333

Note in A/B, the Imagination Machine has 4 places to the right of the decimal. This is standard in APF BASIC, and always shows 4 digits to the right.

Let's try one more example. Type **1** to go again. For the input number A, type **100000** (no commas please). For the input number for B, type **100000** (no commas please). The computer will show

A + B = 200000
A − B = 0
A*B =

It will give you an error message of **LINE 5102 ARITHMETIC OVERFLOW.** This means that the multiplication result is larger than the greatest number that the computer can handle. The computer is really telling you that you have an error, that the error exists in Line 5102 of its program, and what the error was.

In APF BASIC, the maximum number you can have is 999999999.9999. This is 13 digits — 9 to the left of the decimal, 4 to the right.

TO GET BACK TO THE BASIC TUTOR PROGRAM AFTER THIS ERROR MESSAGE TYPE **RUN** AND PRESS THE **RETURN** KEY. THIS ELIMINATES THE ERROR MESSAGE AND BRINGS YOU BACK TO THE BASIC TUTOR MODE. TO START TO RUN YOUR PROGRAM AGAIN, TYPE **RUN** AND PRESS THE **RETURN** KEY A SECOND TIME. THIS RESTARTS LESSON 1. YOU DO NOT HAVE TO RE-ENTER THE PROGRAM.

MATHEMATICIAN
LESSON 1

RUN

TV

10
KEYBOARD → INPUT A - - → NUMBER A ?

20
KEYBOARD → INPUT B - - → NUMBER B ?

30
PRINT A + B - - → A + B =

40
PRINT A−B - - → A−B =

50
PRINT A*B - - → A*B =

60
PRINT A/B - - → A/B =

70
KEYBOARD → INPUT K - - → GO AGAIN (0 = NC, 1 = YES)

80
K = 1   YES
         NO

90
STOP
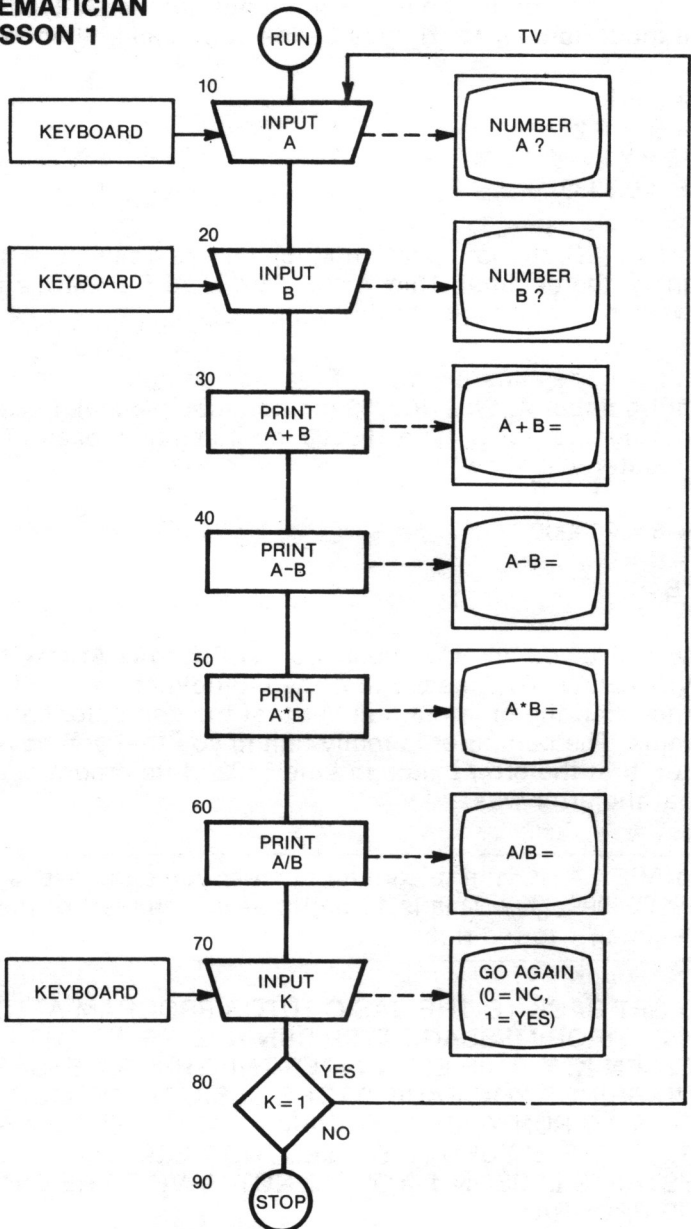
Sometimes action boxes are circles and are used instead of squares to show where the program starts and stops.

CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 1?

You can now continue with Lesson 1 to see how Mathematician works and learn how to write certain programming statements, or you can go directly to Lesson 2. If you want to go directly to Lesson 2 (Area Calculator) and your BASIC TUTOR tape is still in the machine, type **CLOAD**, follow the instructions on the screen and the computer will automatically begin loading. If you want to go directly to Lesson 2 (Area Calculator) and your BASIC TUTOR tape is not in the machine, see page 32, Chapter 5 instructions. If you want to continue with Lesson 1, type **LIST** and press the **RETURN** Key which will bring back the listing of your program (or you can even type **RUN** and press the **RETURN** Key and continue).

HOW MATHEMATICIAN WORKS
(See flow chart on preceding page)

After a calculation is completed and the computer asks for **MORE ?**, press the **0** key and press **RETURN** Key to stop your program. Now type **LIST** and press **RETURN** to list it on the screen. You will see 9 program statements.

This program makes use of INPUT STATEMENTS in order to get the computer to ask questions and get inputs from the computer keyboard into the computer memory.

Line 10 is an INPUT STATEMENT. Quotes are used because they are defining the portion of the statement that will appear on the TV screen. In this case, the message **NUMBER A = ?** will appear on the TV screen and the computer will wait for you to input (type in) a response and press the **RETURN** Key. The value that you type in, is assigned to the variable A. If you did not have the phrase in quotation marks, it would simply printout a **?**. The computer is simply telling you "I want input from you that will be assigned to the variable A."

Line 20 is another INPUT STATEMENT that takes a message that is in quotation marks "**NUMBER B =** " and puts it to the screen. The value typed in for B is assigned to the variable called B.

Line 30 is a PRINT STATEMENT. It will cause 2 things to be printed on the screen. It tells the computer to print $A + B =$ on the screen and then the results of the calculation $A + B$. There's a semicolon between the 2 items, which means that there will be no space between the 2 items that are printed. To print $A + B$, it takes the value of A and B and adds them together and the result is printed.

Line 40 is another PRINT STATEMENT and again causes 2 things to be printed on the screen. First it tells the computer to print on the screen $A - B =$ and then it prints the results of the calculation $A - B$.

Line 50 is another PRINT STATEMENT for multiplication.

Line 60 is another PRINT STATEMENT for division.

Line 70 is our third INPUT STATEMENT of the program.

First the computer puts a message on the screen to prompt. It tells you what the computer would like as possible inputs from the keyboard. Typing in your response (either **0** or **1**) and pressing the **RETURN** Key assigns that value (either **0** or **1**) to the variable named K.

Line 80 looks at the value that you have entered for K and if it is a **1**, tells the program to go back to Line **10** (**GOTO 10**) and continue. If **0** or any other number is typed, then the program continues to Line **90**. This is also known as an IF-THEN statement which we will cover in Lesson 2.

Line 90 is a stop statement. It simply directs the computer to stop executing the program statements and go back to the BASIC TUTOR Mode.

DICTIONARY

This dictionary contains detailed and comprehensive definitions on all of the BASIC keywords introduced in Lesson 1. If you wish to learn more about these keywords and their variations, continue reading this dictionary section. If you wish to go on to Lesson 2, simply type **CLOAD**, press the **RETURN** Key, follow the instructions that appear on the screen and the Imagination Machine will begin loading Lesson 2.

INPUT — allows an assignment to a variable by entering the value from the keyboard.

SIMPLE FORM

INPUT variable name

Will put a **?** to the screen and wait for a value to be typed in from the keyboard. Press **RETURN** Key to indicate value is keyed in.

Example:

INPUT P   Will put a **?** to screen and then whatever value is entered will be assigned to the variable P.

**INPUT A$** prints a question mark and waits for a value to be entered from the keyboard. A$ is called a textual or string-type of variable.*

After a letter or series of letters have been typed and **RETURN** Key pressed, A$ is set equal to the letters typed. The value of A$ will be letters as opposed to a numerical value.

*   *   *   *

VARIATIONS

1. You can have a message appear on the screen before the **?** by placing the message in quotes, following the word INPUT.

   INPUT **"PRICE",P**
   Note the use of **,** to separate the message and variable name.
   INPUT **"YOUR NAME", A$**   Prints **YOUR NAME?**
   After the letters of your name have been typed and **RETURN** Key pressed, A$ is set equal to the name.

*   *   *   *

2. MULTIPLE INPUT
   More then 1 input can be requested by a single input command. Each message or variable name must be separated by a comma.

*Note: String variables must be dimensioned before used in an input statement — see Lesson 8.

Example:

INPUT "PRICE", P, "COST", C
This places the word PRICE on the screen and waits for an input. The input is assigned to variable P. Then it places the word COST on the screen and waits for an input. This next input is assigned to variable C.

＊ ＊ ＊ ＊

3. THE COMMA
   In an Input Statement, the COMMA takes on special meaning if typed in from the keyboard. It acts similar to pressing the **RETURN** Key.
   A. If during a single Input Statement a comma is pressed, it immediately ends the input for that value and anything else typed is ignored except the Return Key. The difference from the Return Key is that the computer will not go on to the next BASIC Statement until the Return Key is pressed.

Example:

10 INPUT P
20 PRINT P

When the above is run and the ? appears, enter for P the following:

12,34 RETURN

What is printed will be the **12**. The **34** is ignored since the comma was pressed during the input.

Next example:

10 DIM A$ (5)
20 INPUT "NAME", A$
30 PRINT A$
Run the above and enter for NAME
FR, ED
Only the **FR** is inputted and printed

B. If during a multiple input statement, the comma is pressed after the first entry, the computer goes on to ask for the second entry.

10 INPUT "PRICE", P, "COST", C
20 PRINT P

Enter the above and type **RUN**. When the word **PRICE?** appears type **12**. Then Type **,** and press the **RETURN** Key. The word **COST** will then appear. Type **34** and then press the **RETURN** Key again.

If after the value for **P** is entered and a comma is pressed, the word COST will appear on the same line of the screen and await an entry for **C**.

* * * *

4. Special Note: In APF BASIC the BREAK Key is not operative during an Input Statement.

STOP — causes a program to immediately stop running and return control to immediate mode of operation.

RULES OF PRECEDENCE

When the Imagination Machine has to evaluate a mathematical expression, it has set rules of which operations are done first. Highest precedence is exponentiation. Next is multiplication or division. Last is addition or subtraction. Where two operations of the same precedence occur, the expression is evaluated from left to right. These rules of precedence can be offset by using ( ) or [ ] to group operations that you want done first.

PRINT 2 + 6 / 3
4 (division done before addition)
PRINT (2 + 6) / 3
2.6666 (brackets change precedence)

Lesson 2

AREA CALCULATOR

IF—THEN, MULTI STATEMENTS

With this program you can compute the area of any rectangle by just entering its length and width. You can also add or subtract additional areas automatically.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 1 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 2. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 2. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 2.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 2, follow the loading instructions on Chapter 5, page 32. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 12. This should position your BASIC TUTOR tape for Lesson 2.

*(Note: On some machines the exact number on the digital tape counter may vary slightly from the number we have indicated. Don't worry. As long as your audio track is correct and you push the **RETURN** Key when told to do so on the audio track, your lesson will load perfectly.)*

REMEMBER

1. This Lesson has a special input monitoring mode which is in control of the machine. It will help you if you make a mistake, but it will also prevent you from typing in extraneous material not in your program. If you attempt to type in such material, the machine will give you an error message **THIS LINE NOT IN PROGRAM.**

2. If you have a question on any of the BASIC words, remember to use the dictionary mode. Simply hold down the **SHIFT** Key and press **?**. A list of BASIC keywords will appear. Type in the keyword, hold down the **SHIFT** Key once again and hit the **?** key. BASIC TUTOR will give you a brief definition of that word.
3. If you do not wish to type in the entire program at this time, or you merely want to see the program run, hit **ESC** Key (the Escape Key) and the **RETURN** Key and the computer will automatically type in the program for you.

ENTER AREA CALCULATOR

Now type each line of the program as listed below. Press **RETURN** Key at the end of each line. If you prefer, you can enter the entire program automatically by pressing the **ESC** Key.

```
10   A=0:T=0
20     INPUT  "LENGTH",L
30     INPUT  "WIDTH",W
40   A=L*W
50     PRINT  "THIS AREA",A
60   T=T+A
70     PRINT  "TOTAL AREA",T
80     INPUT  "ANOTHER CALCULATION( 1=YES )",K
90     IF K=1 THEN   GOTO 20
```

AREA CALCULATOR

After you've finished entering your program into the computer you're ready to use it.

What It Does

The program begins by requesting the length and width. After they have been entered, the program prints the area.

HOW TO USE

Here's an example to calculate simple area:

1. Enter **9** for length and **12** for width. (Remember to press **RETURN** Key after every number you enter). The total area is 108 square feet (or yards or inches).
2. When it asks for a 1 or 0 — type **0** and press **RETURN** Key. The program has stopped. You can LIST it or type **RUN** and return for another calculation.

Now let's see how area calculator can be used for multiple calculations. Let's figure how many square feet of surface area there are in a room we wish to paint. Type **RUN**.

First add the area of a wall. Type **9** for length. If the ceiling is 10 feet high, use this figure for the width of the area of the wall. Enter **10** for the width. The area of the wall is 90 square feet.

Second add the opposite wall by typing **1** (for another calculation) and press **RETURN** Key then **9 RETURN, 10 RETURN**. This adds another 90 square feet.

Third, assume that only one of the other two walls is to be painted and add it to the total. Enter **1** for another calculation and **12** and **10** for the length and width. This adds 120 square feet to the total.

Fourth, subtract any areas that won't be covered. For example, enter **1** for another calculation, **8**, and **−3** for a doorway that's 8 feet high, and 3 feet wide. You will use a minus sign because you want to subtract this area from the total. This area is −24 square feet. Notice that the total area went down from 300 to 276 square feet.

Fifth, to subtract a double window that's 6 by 4 feet, just type **1** for another calculation, **6**, and **−4**. This subtracts another 24 feet and computes total area to be painted as 252 square feet.

In a similar way, you can use this program to compute the area of a lawn, the number of square yards of carpeting to buy, and similar area problems. Just remember that all dimensions used in a single calculation must be measured in the same unit (feet, yards, square centimeters, etc.).

When you've finished a calculation (or made an error), just type **(0)** to signal that you don't want to do another calculation. This stops your program. To start again with the total reset to zero, just type **RUN** and press **RETURN** Key. The RUN command always sets all variables to have a value of zero.

AREA CALCULATOR
LESSON 2

RUN

| 10 | AREA AND TOTAL = 0 |

| KEYBOARD | → | 20 INPUT LENGTH | ┄→ | LENGTH?■ |

| KEYBOARD | → | 30 INPUT WIDTH | ┄→ | WIDTH?■ |

| 40 | COMPUTE AREA |

| 50 | PRINT AREA | ┄→ | THIS AREA |

| 60 | COMPUTE TOTAL |

| 70 | PRINT TOTAL | ┄→ | TOTAL AREA |

| KEYBOARD | → | 80 INPUT K | ┄→ | ANOTHER CALCULATION 1 = YES■ |

90 K = 1   YES

NO

STOP
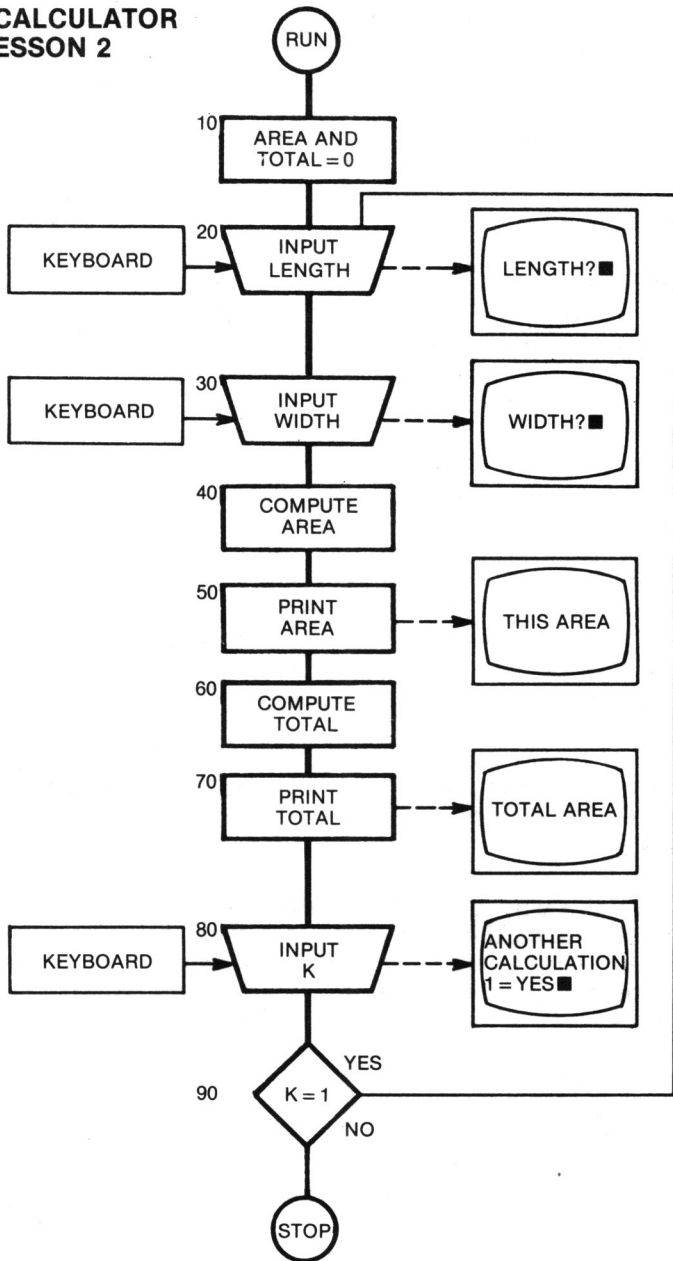
CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 2?

You can now continue with Lesson 2 to see how Area Calculator works and learn how to write certain programming statements, or you can go directly to Lesson 3. If you want to go directly to Lesson 3 (Find the Average) and your BASIC TUTOR tape is still in the machine, type **CLOAD**, follow the instructions on the screen and the computer will automatically begin loading. If you want to go directly to Lesson 3 (Find the Average) and your BASIC TUTOR tape is not in the machine, see page 32, Chapter 5 for instructions. If you want to continue with Lesson 2, type **LIST** and press the **RETURN** Key which will bring back the listing of your program (or you can even type **RUN** and hit the **RETURN** Key and continue running the program).

HOW AREA CALCULATOR WORKS
(See flow chart on preceding page)

Press the **0** key and press **RETURN** Key to stop your program when it asks for another calculation. Now type **LIST** and press **RETURN** Key to list it on the screen.

This program contains two formulas for computing area and total. The rest of the program is used to input the information, print the results, and MAKE A DECISION WHETHER TO CONTINUE OR NOT.

Line 10 is actually two instructions separated by a colon (:). This is known as a MULTI-STATEMENT line. The first statement sets the VARIABLE A to have a value of zero. This letter A is used to keep track of the area to be added to or subtracted from the total. The VARIABLE T is also set to zero because it is used to represent the total area. We want both A and T to start with zero values.

Line 20 is an input statement. It's how we can get inputs to the computer. It prints **LENGTH?** and then sets L equal to the number you input from the keyboard.

49

Line 30 is another input statement. It prints **WIDTH?** and sets W equal to the number you input from the keyboard. An input statement prints the words in quotes and waits for an input.

Line 40 is the formula for area (Area = Length times Width). This sets A equal to the computed area.

Line 50 prints the words **THIS AREA** and the value of A computed in **Line 40**.

Line 60 is the formula for the total (New total = the Old Total plus the current Area just calculated).

Line 70 prints the words **TOTAL AREA** and the value of T.

Line 80 prints **ANOTHER CALCULATION (1 = YES)** and waits for an input for the variable K. The program at this point needs you to make decision. Do you want to add more area to the present total or are we finished?

Line 90 is the decision statement. Based upon the input given in **Line 80**, the program can do 1 of 2 things. It can go back to **Line 20** (if the input for K was 1) or it can go to the next statement. **Line 90** is an IF-THEN statement. The IF statement tests a condition that follows the word IF. If the condition is true, the program will execute statements following the word THEN. If false, BASIC will continue to next numbered step. In this case, there is no next statement and therefore BASIC automatically assumes that the program has ended.

DICTIONARY

This dictionary contains detailed and comprehensive definitions on all of the BASIC keywords used in Lesson 2. If you wish to learn more about these keywords and their variations, continue reading this dictionary section. If you wish to go on to Lesson 3, simply type **CLOAD**, press the **RETURN** Key and follow instructions. The Imagination Machine will begin loading with Lesson 3.

: MULTISTATEMENTS PER LINE — APF BASIC allows you to put more than 1 BASIC statement per numbered line. Multiple statements are simply separated by a colon (:). You can have as many statements per line as long as the total number of characters does not exceed 128 (which is 4 Screen Lines).

Example:

10 PRINT A: PRINT: PRINT B: INPUT C: PRINT A*B, C*B
20 PRINT "IM FINISHED": STOP

* * * *

IF-THEN — The IF Statement tells the computer to make a test of something, and based on the result, to do one of 2 things. The IF Statement allows the computer to change its course of action or change the flow of program execution.

For example, in Line 90 of lesson 2:

90 IF K = 1 THEN GOTO 20

This statement tests the condition of K. If K has a value of 1 then it will direct the program back to Line 20 (K = 1 is true). Or if the test is false (K is a number other than 1) it will continue the program on to the next statement.

The expression must be evaluated to be either true or false. If it is true, the statements following the word THEN will be done.

If it is false, then all statements after the word THEN are ignored and the program goes to the next numbered line. The word THEN is often an assumed word and does not have to be used in an IF-THEN statement.

51

* * * *
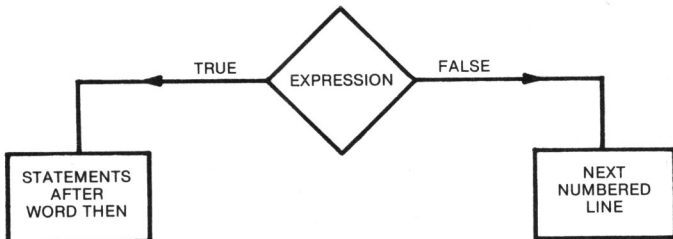
IF A = B GOTO 300 goes to Line 300 if A and B are equal. If
they are not equal, the program goes to the next line. The
equal sign is not used as meaning assignment, but is used
as a test indicator or relational operation.

IF A = B THEN C = 5   sets C equal to five if A and B are
equal.

IF A = B THEN PRINT "EQUAL"   prints **EQUAL** if true.

* * * *

10 INPUT X
20 IF X = 0 THEN PRINT "ZERO": STOP
30 PRINT X,X*X: GOTO 10.

In Line 20 above, if X has a value of zero then the computer
prints **ZERO** and stops. Notice that if the test is true, *all*
statements after the word **THEN** are executed. If X has a
value of anything except zero, then it goes onto Step 30.

Relational Operators

IF Y < 50 THEN GOTO 100

< means "less than" and is called a relational operator.
Other relational operators allowed in an IF Statement are:

A = B means A EQUAL B
A < B means A LESS THAN B

A > B means A GREATER THAN B
A < > B means A NOT EQUAL TO B
A < = B means A LESS THAN OR EQUAL B
A > = B means A GREATER THAN OR EQUAL B

* * * *

The statements following the word THEN can be any type of
statements including another IF Statement.

**10 IF X = 1 THEN IF Y = Z THEN IF PRICE < COST THEN
PRINT "OK"**
IF A$ = B$ THEN GOTO 2000

Transfers program to Line 2000 if A$ is less than B$. The
value of A$ is compared to the value of B$ alphabetically.

Lesson 3

FIND THE AVERAGE

FOR-NEXT, STEP TO

With your computer, you can easily solve any problem that can be stated as a formula. In this lesson we will use a formula for finding the average of several numbers to show you how. Also, we will see how to create "loops" in programs.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 2 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 3. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 3. The audio on the tape will inform you when to press the RETURN Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 3.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 3, follow the loading instructions on Chapter 5, page 32. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 24. This should position your BASIC TUTOR tape for Lesson 3.

(Note: On some machines the exact number on the digital tape counter may vary slightly from the number we have indicated. Don't worry. As long as your audio track is correct and you push the RETURN Key when told to do so on the audio track, your lesson will load perfectly.)

REMEMBER

1. This Lesson has a special input monitoring mode which is in control of the machine. It will help you if you make a mistake, but it will also prevent you from typing in extraneous material not in your program. If you attempt to type in such material, the machine will give you an error message **THIS LINE NOT IN PROGRAM**.

2. If you have a question on any of the BASIC words, remember to use the dictionary mode. Simply hold down the **SHIFT** Key and press **?**. A list of BASIC keywords will appear. Type in the keyword, hold down the **SHIFT** Key once again and hit the **?** key. BASIC TUTOR will give you a brief definition of that word.

3. If you do not wish to type in the entire program at this time, or you merely want to see the program run, hit **ESC** Key (the Escape Key) and the **RETURN** Key and the computer will automatically type in the program for you.

ENTER FIND THE AVERAGE

Enter this program into your computer by typing the nine statements below. Copy these instructions and pressing the **RETURN** Key at the end of each line. If you wish, you can automatically enter this program by pressing the **ESC** Key.

```
10  T=0
20  INPUT "HOW MANY NUMBERS",X
30  FOR L=1 TO X
40  PRINT L,
50  INPUT "VALUE",V
60  T=T+V
70  NEXT L
80  A=T/X
90  PRINT "THE AVERAGE IS ",A
```

RUN FIND THE AVERAGE

After your program is entered just type **RUN** and press the **RETURN** Key.

The program will ask how many numbers you wish to average together by putting a message on the screen, **HOW MANY NUMBERS?**. Type **5** on the keyboard to show that you wish the average of five numbers. Press **RETURN** Key after you've typed **5**.

54

Now the computer will ask for the value of each of the five numbers. Enter these five numbers and press the **RETURN** Key after each number.

3 RETURN
12 RETURN
6 RETURN
2 RETURN
7 RETURN

The average of five numbers is the sum of the numbers divided by five. You have already typed this formula into the computer (Line 80) and the program has printed the correct answer: **6**.

Now use this program to find the average of some other groups of numbers. Make up any numbers you like or use this program to average your grades, your golf scores, etc.

Type **RUN** and press the **RETURN** Key to re-start the program. Enter how many numbers you wish to average and press **RETURN** Key. For each number, the program will request a value. Type the value and press **RETURN** Key for each number. After you've entered values for all the numbers you requested, the program will calculate and print the average and then stop.

CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 3?

You can now continue with Lesson 3 to see how FIND THE AVERAGE works and learn how to write certain programming statements, or you can go directly to Lesson 4. If you want to go directly to Lesson 4 (Counting Machine) and your BASIC TUTOR tape is still in the machine, type **CLOAD**, follow the instructions on the screen and the computer will automatically begin loading. If you want to go directly to Lesson 4, (Counting Machine) and your BASIC TUTOR tape is not in the machine, see page 00, Chapter 5 for instructions. If you want to continue with Lesson 3, type **LIST** and press the **RETURN** Key which will bring back the listing of your program (or you can even type **RUN** and press the **RETURN** Key and continue).

FIND THE AVERAGE
LESSON 3

RUN

| 10 | TOTAL = 0 |

| KEYBOARD | → | 20 | INPUT X | ⇢ | HOW MANY NUMBERS?☐ |

| 30 | LOOP L = 1 TO X |

| 40 | PRINT NUMBER (L) | ⇢ | 1 |

| KEYBOARD | → | 50 | INPUT V | ⇢ | 1 VALUE?☐ |

| 60 | ADD VALUE TO TOTAL |

70 MORE NUMBERS ? YES / NO

| 80 | COMPUTE AVERAGE |

| 90 | PRINT AVERAGE | ⇢ | THE AVERAGE is 00 |

STOP
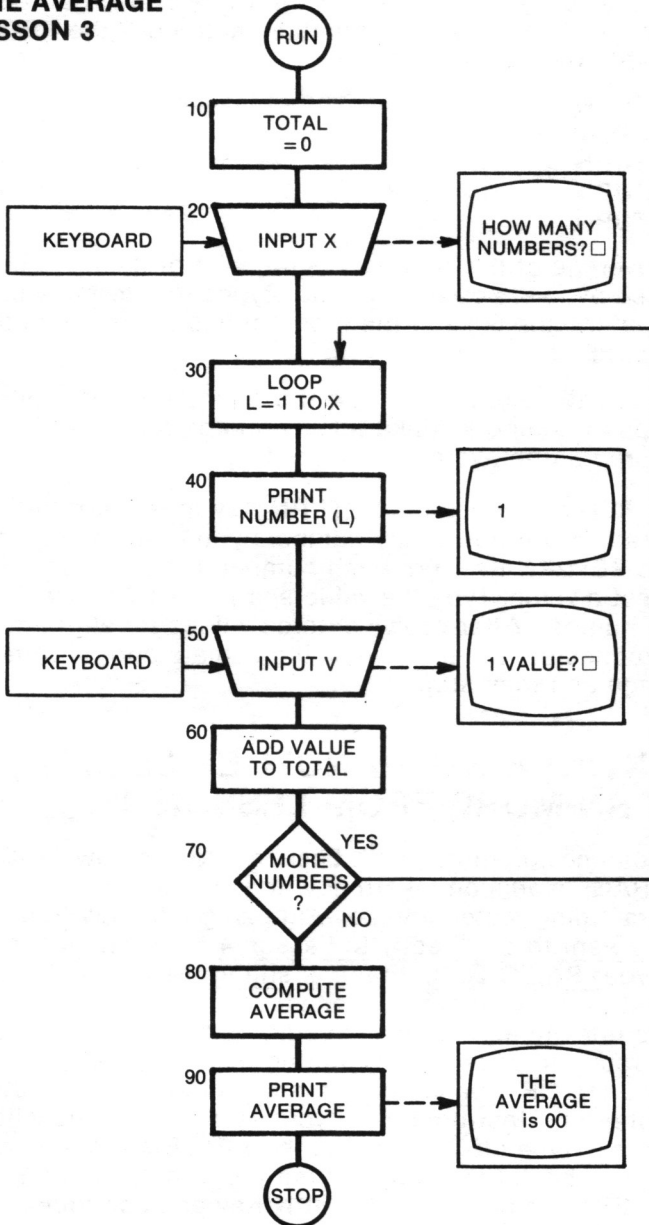
HOW "FIND THE AVERAGE" WORKS
(See flow chart on preceding page)

Type **LIST** to list program on screen.

Line 10 sets the variable T equal to zero. The letter T is used to keep track of the sum of the entered values and it's set to zero each time the program is started.

Line 20 is an input statement and first prints **HOW MANY NUMBERS?** and then sets X equal to the answer you type on the keyboard.

Line 30 is the beginning statement of a FOR-NEXT LOOP. For each of the numbers to be averaged, we have to input their value and add this value to a total. In Line 20, we have determined that there will be X amount of numbers to be entered. In Line 30, we used a variable L to keep track of which entry we are up to. We start with entry number **1** or **L = 1**, and then we go on to entry number **2** or **L = 2**. We continue until we reach entry number **X** (or the amount of numbers we have established in Line 20). Line 70 is a statement that automatically keeps changing the value of L.

Line 40 prints the current value of L. A comma is used so that the next time something is printed on the screen, it will be spaced on the same line as L.

Line 50 is an input statement that prints **VALUE?** on the screen and sets **V** equal to the answer you type on the keyboard.

Line 60 sets the variable T equal to the old total plus the value (V) that you just typed on Line 50. This line keeps a running sum of all the numbers you have typed since the program began.

Line 70 is the completion of the FOR-NEXT LOOP that was started in Line 30. The NEXT L statement tells the computer to take the current value of L and add 1 to it. This becomes the new value of L. It compares this new L with X in Line 30. If L has a value greater than X, all the values have been entered and the LOOP is over. If not, the computer automatically uses this new L (which is the old L plus 1) and goes back to the beginning of the LOOP (Line 30).

Line 80 is the formula for computing the average of several numbers where A is the average, T is the sum of all the numbers, and X is how many numbers are averaged together. The computer sets variable A equal to T divided by X (A = T/X).

Line 90 prints **"THE AVERAGE IS"** on the screen along with the value of A.

DICTIONARY

This dictionary contains detailed and comprehensive definitions on all of the BASIC keywords introduced in Lesson 3. If you wish to learn more about these keywords and their variations, continue reading this dictionary section. If you wish to go on to Lesson 4, simply type **CLOAD**, press the **RETURN** Key and follow instructions. The Imagination Machine will begin loading with Lesson 4.

FOR-NEXT-STEP — creates a program loop. Allows 1 or more statements to be repeatedly executed.

General Form:

FOR < variable > = < 1st value>**TO** <end >**STEP** < increment>
    <statements>
NEXT < variable>
THE **FOR** STATEMENT OPENS A LOOP, AND THE **NEXT** STATEMENT CLOSES THE LOOP.

VARIABLE — can be any numerical variable name, such as A, Price, T, Cost.

1ST VALUE — This is the first value assigned to the variable.

INCREMENT — Amount to add or subtract to variable each time loop is completed.

END VALUE — Final value to be used for variable.

NEXT — Tells computer to add the increment to the current value of variable and compare the result to the END value of the FOR statement. If this result surpasses the END value, the loop is completed and program goes on to the statement following the NEXT statement. If not, it uses this new value for the variable and goes back to the first statement after the opening FOR statement.

Example:

10 FOR L = 1 to 5 STEP 1
20 PRINT L
30 NEXT L
40 STOP

The above will do Statement 20 first with L = 1. Statement 30 says NEXT L. This causes the present value of L to have the increment added to it and the result compared to the END value. Since the result has not surpassed the END value, it is assigned to L and the program goes back to Statement 20 with this new value of L. When Statement 30 finally has L + increment surpass the END value, the program goes on to Statement 40.

* * * *

If the END VALUE is less than the first value the step must be negative.

10 FOR L = 16 TO 1 STEP −3
prints the value of L as the program loops: 16, 13, 10, 7, 4, 1.
20 PRINT L
30 NEXT L

* * * *

If the word STEP and a value after it are ommitted, the computer assumes a step value of + 1.

10 FOR J = 1 TO 20
I will go from 1, 2, 3 ... to 18, 19, 20
20 PRINT J
30 NEXT J

* * * *

You can exit from a loop early by having an assignment statement in the loop that changes the loop variable.

```
10 FOR P = 1 TO 20
20 PRINT P: D = P*P
30 IF D > 100 THEN P = 20
40 NEXT P
```

When P reaches 11, D will be 121 so Statement 30 will
change P to be 20. Then when Statement 40 is executed, it
will end the loop. P = 11 will be the last value done.

<p align="center">* * * *</p>

Variables can be used to set up the first value, the end value
and the increment.

```
10 INPUT A, B, C
20 FOR D = A TO B STEP C
30 PRINT D
40 NEXT D
```

MULTIPLE OR NESTED LOOPS

FOR-NEXT loops can be inside another FOR NEXT loop

```
┌───For J = 1 TO 20
│ ┌─ FOR B = 12 TO 18
│ │     Statements
│ └─ NEXT B
└───NEXT J
```

Note the imaginary dotted lines connecting the opening (for)
and closing (next) of a loop cannot cross.

An Incorrect Program would be

```
 ┌─FOR J = 1 TO 20
┌┼─FOR B = 12 TO 18
││              Incorrect
│└─ NEXT J
└── NEXT B
```

<p align="center">* * * *</p>

You can have loops within loops.

Example:

```
FOR J = 1 TO 20
FOR B = 12 TO 18
NEXT B
FOR K = 3 TO −6 STEP −1
NEXT K
NEXT J
```

SPECIAL NOTE TO APF BASIC

In APF BASIC it is not necessary to put the variable name after the word NEXT. APF BASIC assumes the loop you are closing is the last one opened and totally ignores the variable name after the word NEXT.

```
10  FOR J = 1 TO 20
20  FOR B = 12 TO 18
30  NEXT          Closes "B" Loop
40  NEXT          Closes "J" Loop
```

* * * *

STEP — Establishes the increment to be added to or subtracted from the current value of the variable each time the FOR-NEXT Loop is completed. Can be positive or negative number

Example: **FOR L = 10 TO 50 STEP 5**
Example: **FOR L = 50 TO 10 STEP −10**

or can also be a variable which causes the value of the variable in the FOR-NEXT Loop to be changed each time the corresponding NEXT statement is executed.

Example: **FOR X = 1 TO 10 STEP A**

* * * *

TO — Establishes the end value in the FOR-NEXT Loop.

Example: **FOR J = 1 TO 50**

Lesson 4

COUNTING MACHINE

With this program you can see how LOOPS can be used in counting with various increments.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 3 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 4. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 4. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 4.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 4, follow the loading instructions on Chapter 5, page 32. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 36. This should position your BASIC TUTOR tape for Lesson 4.

*(Note: On some machines the exact number on the digital tape counter may vary slightly from the number we have indicated. Don't worry. As long as your audio track is correct and you push the **RETURN** Key when told to do so on the audio track, your lesson will load perfectly.)*

REMEMBER

1. This Lesson has a special input monitoring mode which is in control of the machine. It will help you if you make a mistake, but it will also prevent you from typing in extraneous material not in your program. If you attempt to type in such material, the machine will give you an error message **THIS LINE NOT IN PROGRAM.**

2. If you have a question on any of the BASIC words, remember to use the dictionary mode. Simply hold down the **SHIFT** Key and press **?**. A list of BASIC keywords will appear. Type in the keyword, hold down the **SHIFT** Key once again and hit the **?** key. BASIC TUTOR will give you a brief definition of that word.
3. If you do not wish to type in the entire program at this time, or you merely want to see the program run, hit **ESC** Key (the Escape Key) and the **RETURN** Key and the computer will automatically type in the program for you.

ENTER COUNTING MACHINE

Enter this program into your computer by typing the 10 statements below. Copy these instructions and press the **RETURN** Key at the end of each line. If you wish you can automatically enter this program by pressing the **ESC** Key.

```
10    INPUT "START",A
20    INPUT "END",B
30    INPUT "SKIP",S
40    FOR L=A TO B STEP S
50    PRINT " ";L;
60    NEXT L
70    PRINT
80    INPUT "GO AGAIN ( 0=NO,1=YES )",K
90    IF K=1 THEN  GOTO 10
100    STOP
```

RUN COUNTING MACHINE

After your program is entered, just type **RUN** and press the **RETURN** Key. The program will ask you for a **STARTING** number **?** and an **ENDING** number **?** It also shows the **"SKIP"** or how many numbers you want to skip between counts. For example, try the following:

Type **RUN** and press the **RETURN** Key.

START ? Type **1**   press **RETURN** Key
END ?   Type **9**   press **RETURN** Key
SKIP ?   Type **2**   press **RETURN** Key

What you should see on your TV screen is the following:

1   3   5   7   9

64

What you are seeing is the starting number printed out on the left, the ending number printed out on the right and the computer automatically skips 2 numbers to get the next result, skips two again and again and finally prints the ending number on the right.

Now let's go again:

Press 1 for YES and press the **RETURN** Key.

START ? Type **20** press **RETURN** Key
END ? Type **1** press **RETURN** Key
SKIP ? Type **−3** press **RETURN** Key

What you will see on the screen is the following:

20   17   14   11   8   5   2

Note — the starting number can be higher than the ending number but you must show a negative skip number. Try it again using a small number for **START** and a large number for **END** with small **SKIP** in between and watch the computer automatically printout the results on the screen. You will see it even works with decimals.

CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 4?

You can now continue with Lesson 4 to see how Counting Machine works and learn how to write certain programming statements, or you can go directly to Lesson 5. If you want to go directly to Lesson 5 (Decision Maker) and your BASIC TUTOR tape is still in the machine, type **CLOAD**, follow the instructions on the screen and the computer will auto- ·matically begin loading. If you want to go directly to Lesson 5 (Decision Maker) and your BASIC TUTOR tape is not in the machine, see page 32, Chapter 5 for instructions. If you want to continue with Lesson 4, type **LIST** and press the **RETURN** Key which will bring back the listing of your program (or you can even type **RUN** and press the **RETURN** Key and continue running the program).

COUNTING MACHINE
LESSON 4

RUN

Line		
	KEYBOARD	10 INPUT A → START ?
	KEYBOARD	20 INPUT B → END ?
	KEYBOARD	30 INPUT C → SKIP ?

40 LOOP L = A TO B STEP S

50 PRINT NUMBER → 1 2 3 4

60 MORE ? — YES → ADVANCE TO NEXT NUMBER

NO

70 PRINT

80 INPUT K — KEYBOARD → GO AGAIN (0 = NO, 1 = YES)?

90 K = 1 ?

100 STOP
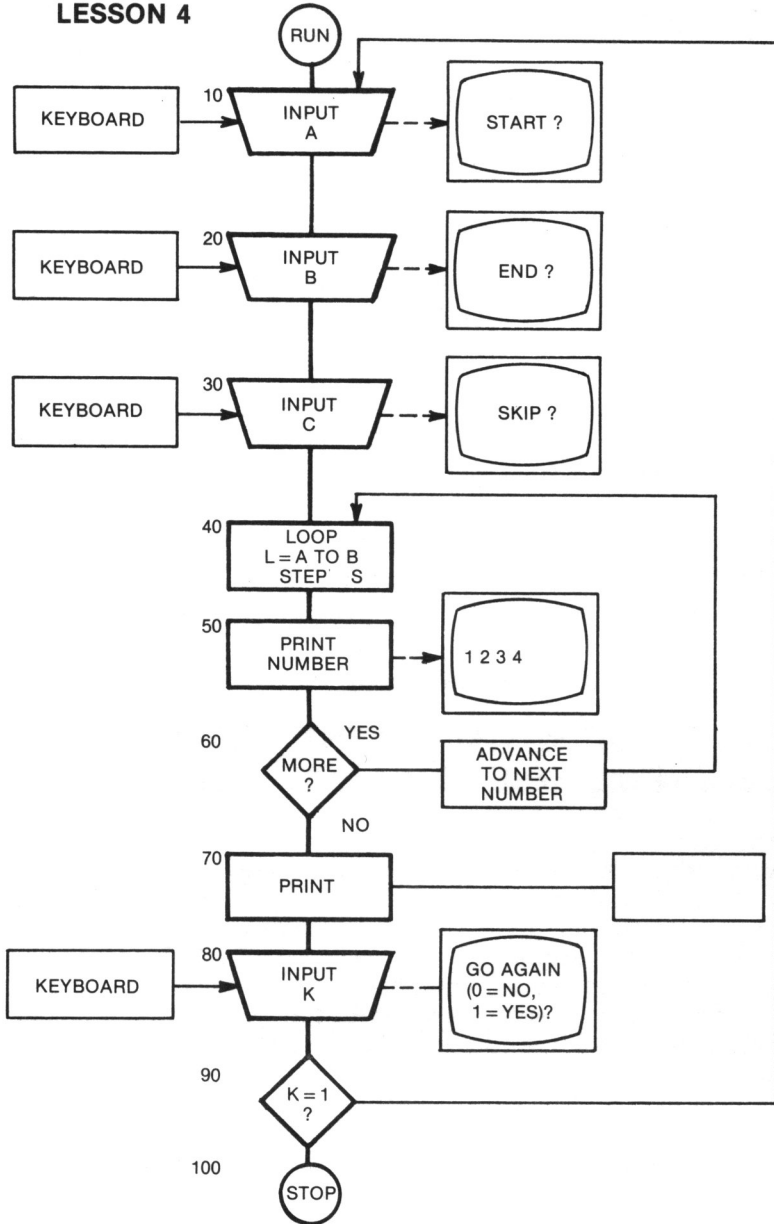
HOW COUNTING MACHINE WORKS
(See flow chart on preceding page)

Type **LIST** to list program on screen.

Line 10 is an input statement. It prints the word **START** and waits for an input. The input value is assigned to the variable A. This will be the first value we use in our counting LOOP.

Line 20 is another input statement and assigns a value to the variable B, the ending number we want to use in our counting LOOP.

Line 30 is an input statement that takes a value and assigns it to the letter S. This value is the SKIP value. That is, it is this value that we will add (or subtract) to the variable L in the LOOP.

Line 40 is the opening statement of a FOR-NEXT LOOP. Since variables are used for the STARTING and ENDING and SKIP values, the computer will replace the variable names by their numerical values. The first time L will be assigned the numerical value of A, which was inputted in Line 10. The FOR-NEXT LOOP will continue until L has a value the same as B's numerical value. L will change by adding or subtracting the value of S (the SKIP). This will occur each time the NEXT statement in Line 60 is executed.

Line 50 prints a space (indicated by **PRINT" "**) and the current value of L. This space is for readability between the numbers (or values of L). The first time Line 50 is executed, L will have the same numerical value as A. The next time, it will have the value of the old L ± the value of S.

Line 60 is the close of a FOR-NEXT LOOP that was started in Line 40. It automatically checks the current value of L to see if we have reached the end value of L in the FOR LOOP. If it hasn't, it adds the value of S to the current value of L to create a new L and goes back to the beginning of the FOR LOOP in Line 40. If L has surpassed the end value, the program goes on to Line 70.

Line 70 is a PRINT statement. When nothing follows the word **PRINT** it moves the cursor back to the beginning of the next line.

Line 80 is another input statement that puts the words **GO AGAIN (0 = NO, 1 = YES)** on the screen and waits for an input from the keyboard. This input is then assigned to the variable K.

Line 90 is an IF-THEN statement that tells the computer if the variable K has the value of 1 to go back to Line 10 and start the program all over again. If K has any other value assigned to it than 1, the program continues to Line 100.

Line 100 tells the program to STOP.

Lesson 5

DECISION MAKER

KEY$

Decision Maker is an ESP program, where you ask the computer a question and it gives you a **YES**, **NO**, or **MAYBE** answer. It shows the BASIC function **KEY$ (0)** which enables the computer to read a character from the keyboard *without* the **RETURN** Key having to be pressed to tell the computer you're finished entering.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 4 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 5. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 5. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 5.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 5, follow the loading instructions on Chapter 5, page 32. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 49. This should position your BASIC TUTOR tape for Lesson 5.

REMEMBER

1. This Lesson has a special input monitoring mode which is in control of the machine. It will help you if you make a mistake, but it will also prevent you from typing in extraneous material not in your program. If you attempt to type in such material, the machine will give you an error message **THIS LINE NOT IN PROGRAM.**

69

2. If you have a question on any of the BASIC words, remember to use the dictionary mode. Simply hold down the **SHIFT** Key and press **?**. A list of BASIC keywords will appear. Type in the keyword, hold down the **SHIFT** Key once again and hit the **?** key. BASIC TUTOR will give you a brief definition of that word.
3. If you do not wish to type in the entire program at this time, or you merely want to see the program run, hit **ESC** Key (the Escape Key) and the **RETURN** Key and the computer will automatically type in the program for you.

ENTER DECISION MAKER

Enter this program into your computer by typing the following 11 programming statements. Copy these instructions and press the **RETURN** Key at the end of each line. If you wish, you can automatically enter this program by pressing the **ESC** Key.

```
10    FOR L=70 TO 90 STEP 10
20    PRINT
30    IF   KEY$ ( 0 )="D" GOTO L
40    IF   KEY$ ( 0 )=" " STOP
50    NEXT L
60    GOTO 10
70    PRINT "YES": GOTO 100
80    PRINT "NO": GOTO 100
90    PRINT "MAYBE": GOTO 100
100    IF   KEY$ ( 0 )="D" GOTO 100
110    GOTO 20
```

RUN DECISION MAKER

After you have entered your program, type **RUN**, press the **RETURN** Key, and the screen will clear. Ask yourself a question out loud, "DOES THE IMAGINATION MACHINE LOVE ME?" Press the **D** key on the keyboard, and the computer will respond with a **YES**, **NO**, or **MAYBE**. Remember to release the **D** key after each question you ask. Ask yourself another question and press the **D** key — see the ESP powers of the Imagination Machine. To stop your program, press the space bar and the cursor will reappear on the screen.

CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 5?

You can now continue with Lesson 5 to see how Decision Maker works and learn how to write certain programming statements, or you can go directly to Lesson 6. If you want to go directly to Lesson 6 (Guessing Game) and your BASIC TUTOR tape is still in the machine, type **CLOAD**, follow the instructions on the screen and the computer will automatically begin loading. If you want to go directly to Lesson 6 (Guessing Game) and your BASIC TUTOR tape is not in the machine, see page 32, Chapter 5 for instructions. If you want to continue with Lesson 5, type **LIST** and press the **RETURN** Key which will bring back the listing of your program (or you can even type **RUN** and press the **RETURN** Key and continue running the program).

HOW DECISION MAKER WORKS
(See flow chart on following page)

Type **LIST** to list program on screen.

Line 10 is the beginning of a FOR-NEXT LOOP. We go through the LOOP with the variable L first having a value of 70, then 80, then 90. This is accomplished because we have instructed the computer to "STEP 10" between 70 and 90.

Line 20 is a PRINT statement which will bring the cursor to the next line.

Line 30 is an IF-THEN statement that makes use of the KEY$(0) function. It instructs the computer to execute the KEY$(0) function. The KEY$(0) function looks at the main keyboard to see if any keys are being pressed and returns with the value of any key pressed. If no key is pressed, it returns with no value. In Line 30 we see that if the KEY$(0) function returns with the value of the "D" key, it tells the computer to go to Line L. Since we are in a LOOP, L could be either 70, 80, or 90. The value of L is really totally dependent upon when you press the D key. If KEY$(0) returns with any other value than D, the computer goes on to Line 40.

DECISION MAKER
LESSON 5

(RUN)

10
```
LOOP
L = 70 TO 90
STEP 10
```

CLEAN SCREEN

20
```
PRINT
```
- - - - →

KEYBOARD →
30
```
INPUT
KEYBOARD
```

```
KEY$(0)
= "D"
```

70
```
L = 70
```
→
```
PRINT
YES
```

80
```
L = 80
```
→
```
PRINT
NO
```

90
```
L = 90
```
→
```
PRINT
MAYBE
```

NO
```
KEY$(0)
= " "
```
YES

```
KEY(0)
= "0"
```
YES

NO

```
GOTO 20
```

40
(STOP)
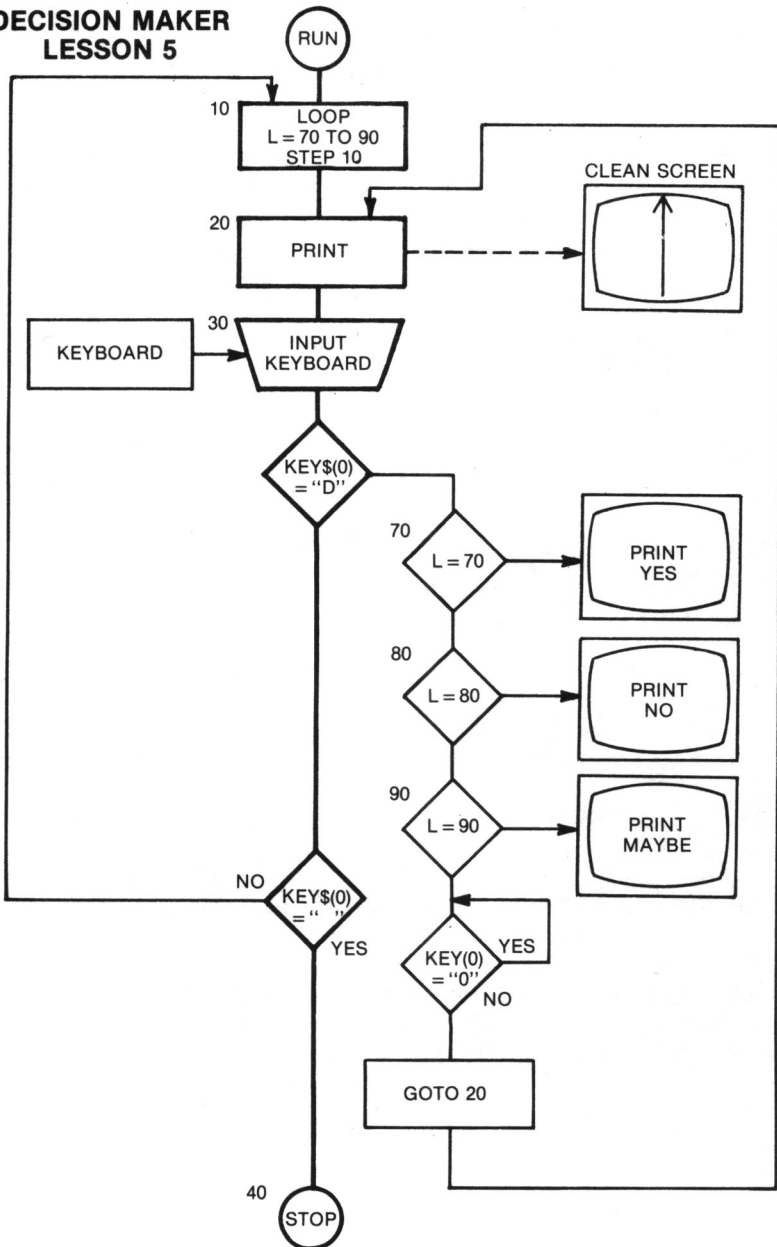
Line 40 is a second IF-THEN statement that again calls for the execution of the KEY$(0) function. If the KEY$(0) function returns a value of the space bar key, then the program is directed to stop. The space bar key is shown in the program by " ". If KEY$ returns any other value, then it goes on to Line 50.

Line 50 is a NEXT statement that closes the FOR LOOP opened in Line 10. After the first time, L has a value of 70 which is not the end limit so a new L with the value of 80 will be set up so that the program goes back to Line 20, then goes to 30 and 40 and calls upon the KEY$(0) function again.

Line 60 says go back to Line 10 after we have run through our LOOP with L equal to 70, 80, and finally 90. We want to restart the LOOP from the beginning.

Line 70 is a PRINT statement and can only be accessed from Line 30 if the KEY$(0) function returns the value D and at the same time L had a value of 70. Then it prints **YES** on the screen and goes to 100.

Line 80 is another PRINT statement and can only be accessed by Line 30 if the KEY$(0) function returned the value of D and at the same time L had a value of 80. It prints **NO** on the screen and goes to 100.

Line 90 is a PRINT statement that prints **MAYBE** on the screen if L has a value of 90 and then goes to 100.

Line 100 is another IF-THEN statement that again executes the KEY$(0) function. Before returning back to the FOR LOOP, we must make sure that the user has removed his finger from the D key. Therefore, Line 100 says that if the KEY$(0) has a value of D, go to the beginning of Line 100 and executes the KEY$(0) function again until the D key is no longer pressed. Then it goes to Line 110.

Line 110 directs the computer to go to Line 20 which starts all over again.

DICTIONARY

This dictionary contains detailed and comprehensive definitions on all of the BASIC keywords introduced in Lesson 5. If you wish to learn more about these keywords and their variations, continue reading this dictionary section. If you wish to go on to Lesson 6, simply type **CLOAD**, press the **RETURN** Key and follow instructions. The Imagination Machine will begin loading with Lesson 6.

KEY$(0) — is a BASIC function that, when executed, goes and looks at the main keyboard and returns with the value of a key pressed. If no key is pressed, KEY$(0) is a null. A null can be indicated by " ". If the **A** Key is pressed, KEY$(0) = "A", if **B** is pressed, KEY$(0) = "B", etc. The main difference from an input statement is KEY$ reads the keyboard and returns a result without waiting for the RETURN Key to be pressed. Also, KEY$ does not put the character pressed onto the screen but only returns the value into the computer program where it can be looked at or assigned to a variable.

Example:

10 PRINT KEY$(0);
20 IF KEY$(0) < > "   " THEN GOTO 10

The above program will keep printing the value returned by KEY$(0). (If no key is pressed, the returned value is called null.) It then looks at KEY$ again, and if it is not a space character (the space bar being pressed), it goes back to 10.

* * * *

KEY$(1) — similar to KEY$(0) except it reads a key from the right hand controller of the MP1000.

The Numeric Keys (0-9) return their number

CL — returns "?"
EN — returns "!"

The Joystick has 4 positions that can be returned

UP — returns **"N"**
DOWN — returns **"S"**
RIGHT — returns **"E"**
LEFT — returns **"W"**

KEY$(2) — like KEY$(1) except returns a key from left-hand controller.

* * * *

The value returned by KEY$ can be assigned to a letter variable.

Example:

```
10  DIM A$(1)
20  A$ = KEY$(0)
30  PRINT A$;
40  IF A$ < > "   " THEN GOTO 20
```

* * * *

Since KEY$ instantly looks at the keyboard and returns a value, sometimes you might want to keep looking until a key is pressed.

Example:

```
10  DIM A$(1)
20  A$(1) = KEY$(0):  IF A$(1) = "  " THEN GOTO 20
30  PRINT A$
```

Line 20 will wait until a key is pressed before it continues to line 30.

Lesson 6

GUESSING GAME

INT, RND

In this program the computer will create a random number and you will try to guess what it is.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 5 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 6. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 6. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 6.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 6, follow the loading instructions on Chapter 5, page 32. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 62. This should position your BASIC TUTOR tape for Lesson 6.

REMEMBER

1. This Lesson has a special input monitoring mode which is in control of the machine. It will help you if you make a mistake, but it will also prevent you from typing in extraneous material not in your program. If you attempt to type in such material, the machine will give you an error message **THIS LINE NOT IN PROGRAM.**
2. If you have a question on any of the BASIC words, remember to use the dictionary mode. Simply hold down the **SHIFT** Key and press **?**. A list of BASIC keywords will appear. Type in the keyword, hold down the **SHIFT** Key once again and hit the **?** key. BASIC TUTOR will give you a brief definition of that word.

3. If you do not wish to type in the entire program at this time, or you merely want to see the program run, hit **ESC** Key (the Escape Key) and the **RETURN** Key and the computer will automatically type in the program for you.

ENTER GUESSING GAME

Enter this program into your computer by typing these six instructions on your keyboard. Remember to press the **RETURN** Key after you finish typing each line.

Try to be accurate, and don't worry about making an error. If you type the wrong key or leave something out, BASIC TUTOR will show you how to make the correction.

```
10  X= INT ( RND ( 0 )*10+1 )
20     INPUT "GUESS A NUMBER",G
30     IF G=X PRINT "RIGHT": STOP
40     IF G>X PRINT "LESS"
50     IF G<X PRINT "MORE"
60     GOTO 20
```

RUN GUESSING GAME

After you've entered your program into the computer, type **RUN** and press the **RETURN** Key.

The computer has created a random number between 1 and 10. Try to guess what it is, type your number on the keyboard, and press the **RETURN** Key.

If you guessed the right number, the computer printed "**RIGHT**" and stopped. If not, the computer gave you a hint. Try again until you get the right answer.

After you've guessed right, type **RUN** and press **RETURN** Key to run your program again. Each time your program runs, the computer picks a random number between 1 and 10.

Run this program several times and see how the computer continues asking for your guess until you get the right answer.

CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 6?

You can now continue with Lesson 6 to see how Guessing Game works and learn how to write certain programming statements, or you can go directly to Lesson 7. If you want to go directly to Lesson 7 (Coloring Book) and your BASIC TUTOR tape is still in the machine, type **CLOAD**, follow the instructions on the screen and the computer will automatically begin loading. If you want to go directly to Lesson 7 (Coloring Book) and your BASIC TUTOR tape is not in the machine, see page 32, Chapter 5 for instructions. If you want to continue with Lesson 6, type **LIST** and press the **RETURN** Key which will bring back the list of your program (or you can even type **RUN** and press the **RETURN** Key and continue running the program).

HOW GUESSING GAME WORKS

After the program has stopped, type **LIST** then press the **RETURN** Key. The program shows us how the computer can pick numbers at random, input information, and make decisions.

Line 10 sets the variable X equal to a random number between 1 and 10. The computer has built-in function that when called upon returns a random number between 0 and .99. Line 10 is a formula that converts this random number to a number between 1 and 10. First, it calls upon the RND function. Next the result is multiplied by 10. This gives us a number between 0 and 9.9. To this number we add 1, which now gives us a number between 1 and 10.9. We now use the function called INT — which means the integer portion. The integer portion of a number is the number to the *left* of the decimal. Therefore X gets an integer value between 1 and 10.

Line 20 prints **"GUESS A NUMBER?"** on the screen. When you type a number on the keyboard and press **RETURN**, the variable G is set equal to that number. If you type **3** then G = 3, etc.

GUESSING GAME
LESSON 6

RUN

10	PICK A NUMBER BETWEEN 1 AND 10

GUESS A NUMBER

KEYBOARD

20	INPUT G

30	G = X RIGHT ?	Y → PRINT	RIGHT

STOP

40	G > X TOO BIG ?	Y → PRINT	LESS

50	G < X TOO SMALL ?	Y → PRINT	MORE

60	GUESS AGAIN
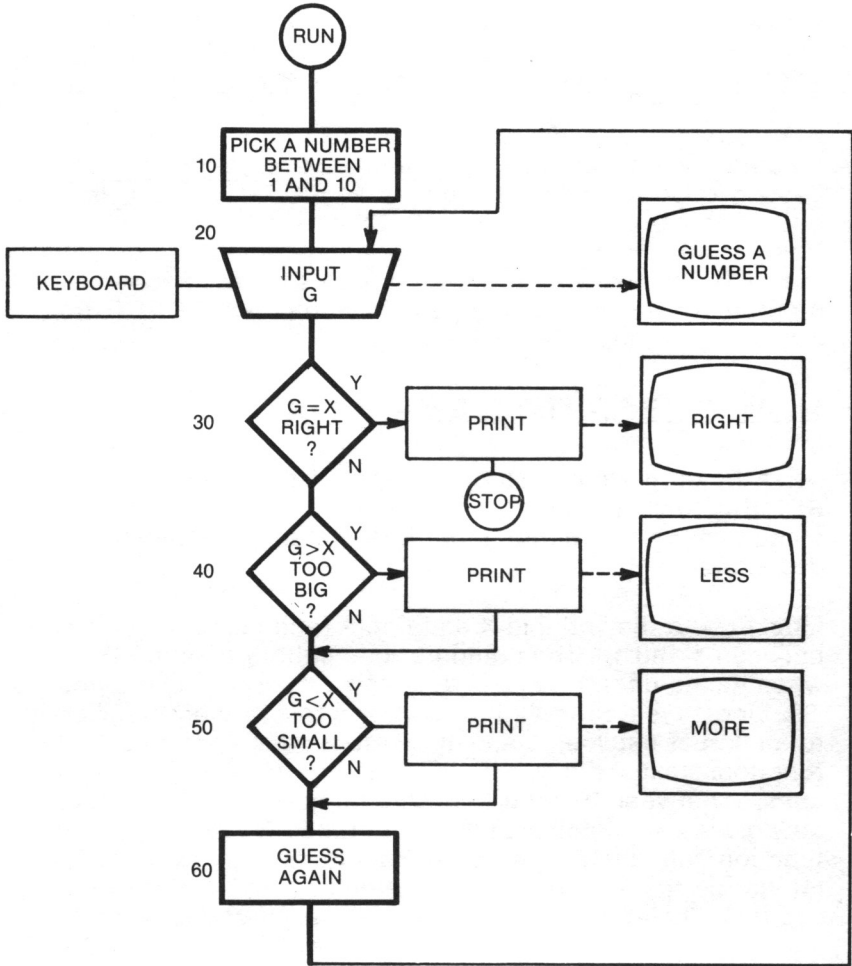
Line 30 is an IF-THEN statement and says that if G = X print **"RIGHT"** on the screen and then stop the program. If G (your guess) equals X (the computer's number) then you have guessed right. If they are not the same go on to Line 40.

Line 40 prints **"LESS"** on the screen if your guess is more than (>) X.

Line 50 prints **"MORE"** if your guess is less than (<) the answer.

Line 60 goes back to Line 20 so that you can input another guess.

DICTIONARY

This dictionary contains detailed and comprehensive definitions on all of the BASIC keywords used in Lesson 6. If you wish to learn more about these keywords and their variations, continue reading this dictionary section. If you wish to go on to Lesson 7, simply type **CLOAD**, press the **RETURN** Key and follow instructions. The Imagination Machine will begin loading with Lesson 7.

INT (expr) — returns the integer portion of the expression. The expression can be a number, a numeric variable, or a calculation. The integer portion will be the digits to the left of the decimal point.

INT (3.14) = 3
INT (7.685) = 7
INT (.999) = 0
INT (− 3.6) = − 3

PRINT INT(7.1*8.2)
58
B = INT(− 3.2 + 4.9)

RND(0) — creates a random number between .00 and .99. This random number can be printed, assigned to a variable, or used anywhere a numeric value can be used.

10 FOR I = 0 TO 100
20 PRINT RND(0),
30 NEXT I

To obtain a random number other than in the range of .00 to .99, simply multiply RND(0) by a number to set the range.

Examples:

A = RND(0)*7   will give a random number
                                    between 0 and 6.93

A = RND(0)*10 + 1   will give a random number
                                    between 1 and 10.9

A = RND(0)*RND(0)   will give a random number
                                    between 0 and .9801

A = RND(0)/100   will give a random number
                                    between .0000 and .0099

Use the INT function to obtain integer numbers

A = INT(RND(0)*10)   will give a random number
                                    between 0.00 and 9.00

Lesson 7

COLORING BOOK
SHAPE, COLOR, PLOT

This program shows you how the Imagination Machine uses the words, COLOR, SHAPE, and PLOT to draw on the screen. You can repeat the program as many times as you like and see many of the possible combinations.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 6 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 7. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 7. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 7.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 7, follow the loading instructions on Chapter 5, page 32. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 76. This should position your BASIC TUTOR tape for Lesson 7.

ENTER COLORING BOX

Now enter the program. You can type each line in the program or you can press the **ESC** Key and the program will be entered automatically. To enter COLORING BOX yourself, just type each line shown below, and press the **RETURN** Key after each line. Note that Line 10 is wider than the screen. The computer will automatically move down to the next row on the screen as you type. Don't press RETURN till you finish typing the complete line. If you make a mistake, BASIC TUTOR will show you how to correct it.

```
10    FOR L=1 TO 32: PRINT : NEXT L
20    INPUT "COLOR ( 0-7 )",C
30    INPUT "SHAPE ( 0-15 )",S
40    COLOR =C: SHAPE =S
50    PLOT 2,1
60    FOR X=11 TO 31
70    FOR Y=3 TO 11
80    PLOT X,Y
90    NEXT Y
100   NEXT X
110   INPUT "MORE( 1=YES )",K
120   IF K=1 THEN GOTO 10
```

RUN COLORING BOX

After you have entered your program, type **RUN** and press
the **RETURN** Key.

COLORING BOX begins by asking you to select one of the
eight colors. The choices are listed below, with their
numbers. Type the number of your favorite color and press
the **RETURN** Key.

0 = light green	4 = white
1 = yellow	5 = light blue
2 = dark blue	6 = purple
3 = red	7 = orange

Now select a shape to be drawn from the following chart:

84
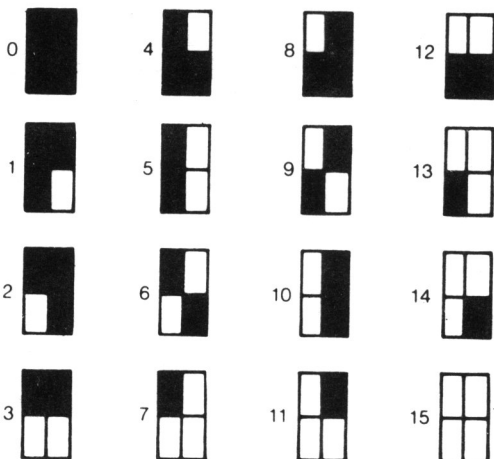
After you've selected a color and a shape, the program will plot this shape near the top left of the screen. A large rectangle in the center of your screen will then be filled in by repeatedly plotting this shape over and over. The large square is 21 columns wide and 9 rows high.

To see another shape or color, just type **1** and press the **RETURN** Key and the programs repeats. There are 128 combinations of the 8 colors and 16 shapes, so try several and see how your computer can create patterns and colors on the screen.

After you've finished experimenting with this program, just type **0** and press the **RETURN** Key and the program will stop.

CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 7?

You can now continue with Lesson 7 to see how Coloring Box works and learn how to write certain programming statements, or you can go directly to Lesson 8. If you want to go directly to Lesson 8 (Sketch Pad) and your BASIC TUTOR tape is still in the machine, type **CLOAD**, follow the instructions on the screen and the computer will automatically begin loading. If you want to go directly to Lesson 8 (Sketch Pad) and your BASIC TUTOR tape is not in the machine, see page 32, Chapter 5 for instructions. If you want to continue with Lesson 7, type **LIST** and press the **RETURN** Key which will bring back the listing of your program (or you can even type **RUN** and press the **RETURN** Key and continue running the program).

HOW COLORING BOX WORKS

Stop your program by selecting **0** and pressing the **RETURN** Key.

Type **LIST** and press the **RETURN** Key again. The complete program is now listed on your screen. Compare the steps in this program with the flow chart diagram.

Line 10 clears the screen by printing 32 blank lines. A PRINT with nothing after it simply prints a blank line. We use a FOR-NEXT Loop to repeat PRINT (the blank) 32 times.

Line 20 prints **COLOR (0-7)?** and sets the variable C to the number typed in.

Line 30 prints **SHAPE (0-15)?** and sets the variable S to the number typed in.

Line 40 sets COLOR and SHAPE to the numbers typed in Lines 20 and 30. COLOR and SHAPE are 2 system keywords in BASIC which contain the number codes to be used in a PLOT statement.

Line 50 plots the shape you selected in the color selected at the location 2,1. This places it on the screen so you can see the color and shape you selected. PLOT 2,1 places the box in column #2 and row #1. This is really the 3rd column and 2nd row because the first column and row are numbered 0.

Lines 60 and 100 form the loop that moves the pattern to the right. This is the outer loop of a nested FOR-NEXT LOOP.

Lines 70 and 90 form the loop that moves the pattern from top to bottom. This is the inner loop of a nested FOR-NEXT LOOP.

Line 80 plots the shape in the color selected at the location set by horizontal and vertical loops. It will be repeatedly executed with the values of X & Y being changed by the two FOR-NEXT loops.

Line 110 prints **MORE (1 = YES)?** and sets K = to the value typed in.

Line 120 looks at the value of K typed in Line 110. If K equals 1 then it directs the program back to Line 10. If K is anything but 1, it goes on and ends the program.

COLORING BOOK
LESSON 7

RUN

10 FOR LOOP PRINT 32 BLANK LINES TO CLEAR SCREEN

KEYBOARD → **20** INPUT C

COLOR (0-7)?

KEYBOARD → **30** INPUT S

SHAPE (0-15)?

40 SET COLOR, SHAPE

50 PLOT SINGLE BOX WITH SHAPE & COLOR

60 FOR LOOP FOR X COORDINATE

70 FOR LOOP FOR Y COORDINATE

80 PLOT X, Y

90 NEXT VALUE OF Y

100 NEXT VALUE OF X

KEYBOARD → **110** INPUT K

MORE (1 = YES)?

120 K = 1

YES

NO

STOP
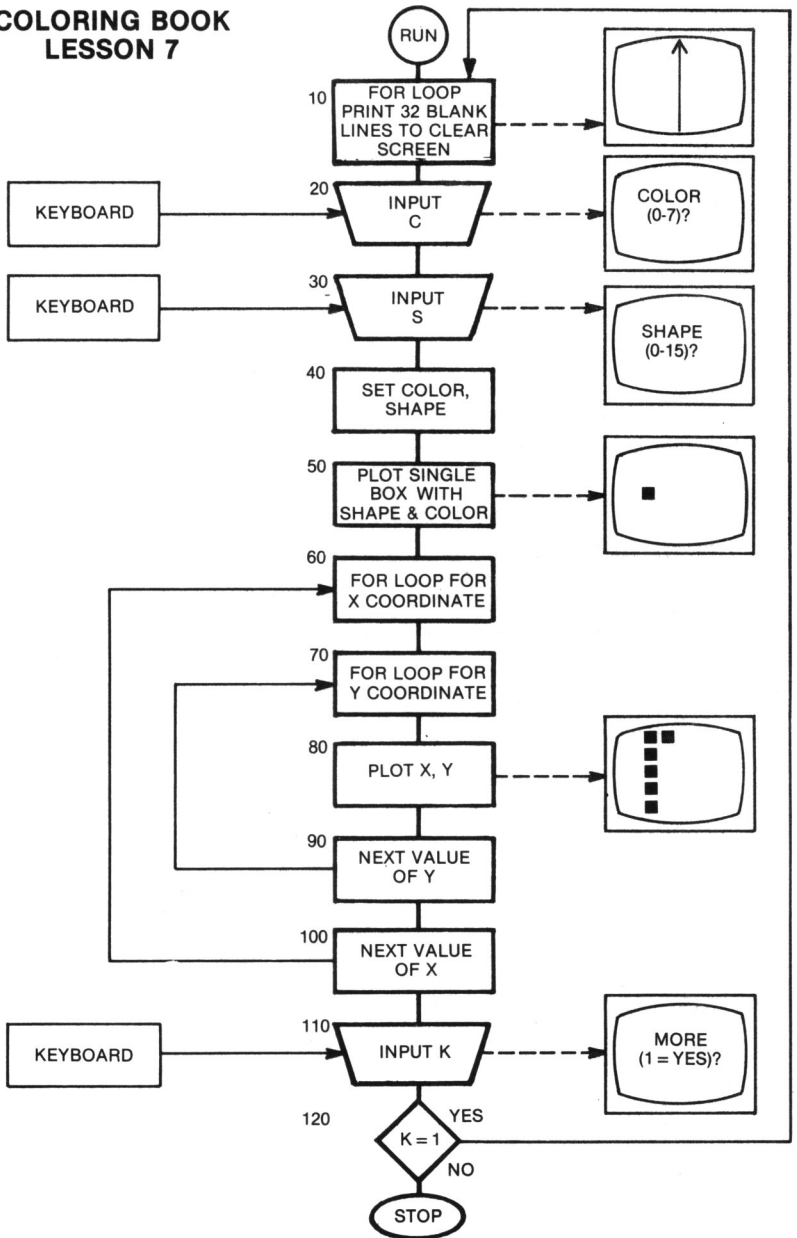
See how the program begins first by clearing the screen. Then your choice of color and shape are inputted into the program from the keyboard.

The screen is divided into 512 boxes. Each box has a row and a column number. The shape you selected is now plotted in the color you picked.
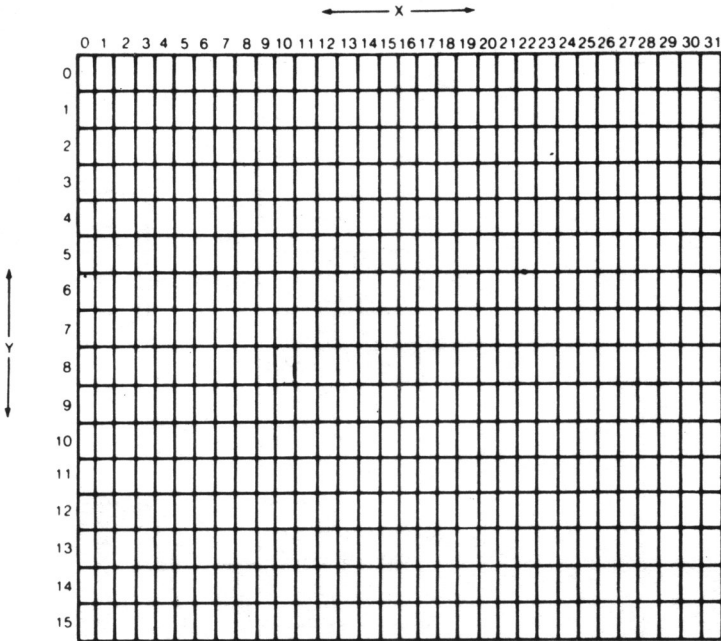
First this shape is plotted at location 2,1 (near the top of the screen). Then this shape is plotted over and over again to fill up a large section of the screen.

Two loops are used to fill the rectangle. One loop forms a vertical column and the other loop moves this column horizontally to the right repeatedly to fill the rectangle.

Finally, the program prompts you by asking if you want more. If you do, it repeats from the beginning; if not, it stops.

DICTIONARY

This dictionary contains detailed and comprehensive definitions on all of the BASIC keywords introduced in Lesson 7. If you wish to learn more about these keywords and their variations, continue reading this dictionary section. If you wish to go on to Lesson 8, simply type **CLOAD**, press the **RETURN** Key and follow instructions. The Imagination Machine will begin loading with Lesson 8.

SHAPE — sets the shape to be drawn on the screen in either a PLOT, HLIN or VLIN Statement.

To get a higher resolution each cursor position can be divided into 4 cells. By selecting the appropriate shapes you can draw a picture with much higher resolution (64 x 32).

The shape table shows the number for each cursor shape.

Shape is set equal to a constant or expression. Its value is used in the next plot or line command. Modulo 16 arithmetic is performed so shape can have a value greater than 16.

Modulo 16 arithmetic uses the remainder of the number divided by 16.

89

Since only one color can be assigned to a cursor position the cells that are lit will all be the same color and the remaining cells will be black.

SHAPE   6   BLACK   YELLOW

COLOR   1   YELLOW   BLACK

COLOR — sets the color to be drawn on the screen in either a PLOT, HLIN or VLIN Statement.

Any cursor position can be any one of eight colors at any time by assigning it a color number from 0 to 7. Setting COLOR = expression determines the color to be used in the next line or plot command. Expression can be a constant or expression. Modulo arithmetic is performed so the expression can be greater than > 7.

0 = Light Green	4 = White
1 = Yellow	5 = Light Blue
2 = Dark Blue	6 = Purple
3 = Red	7 = Orange

This means you can have eight different colors on the screen at the same time.

PLOT — draws a shape on the screen. The screen is divided into 512 boxes. Each box has a row and column number.

PLOT column, row — draws on the screen in column and row specified with color and shape previously set.

PLOT 0,0 — is in the upper left corner of the screen.

PLOT 15,31 — is in the lower right corner of the screen. Both SHAPE and COLOR must be set before PLOT is used to fill spaces on the screen.

PLOT 10,8

Means place the shape in the 10th column in the 8th row.
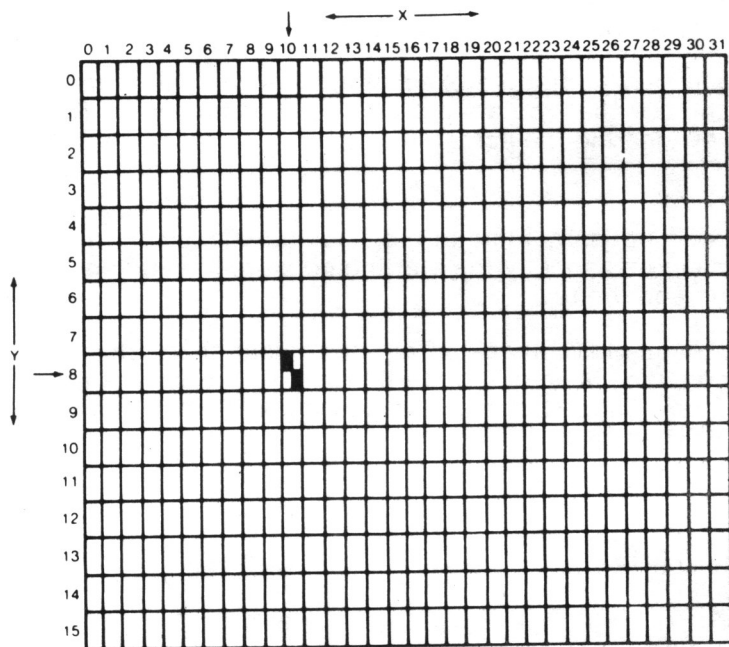
To place shape 6 on the screen in yellow and black in the position X = 10,Y = 8 the command would be:

COLOR = 1:SHAPE = 6:PLOT 10,8

PLOTTING A SHAPE ON ANOTHER SHAPE

If you plot a shape on top of another shape, the second shape is added to the first shape to give a new shape. The new shape takes on the new color.

Example:

10 FOR I = 1 TO 32: PRINT: NEXT I
20 COLOR = 4: SHAPE = 3
30 PLOT 10,10
40 SHAPE = 5: COLOR = 5
50 PLOT 10,10

Line 50 will plot a box at Location 10,10 with color 6, but the shape will be the old shape of 3 combined with the new shape of 5. The shape plotted with be 7.

The only way to plot the exact shape you want is to make sure the box that you are plotting into has no shape. This is done by clearing the screen first.

LINE

To draw a horizontal line of the selected shape use the HLIN command and specify the starting column, the ending column and the row.

Note: If you plot a shape on top of another shape the second shape is added to the first shape; it does not replace it. The new shape takes on the new color. Plotting a 9 on top of a 6 gives you a 15.

Example:

5 CALL 17046
```
10 SHAPE=15
20 FOR I=0 to 7
30 COLOR=I
40 HLIN I,31−I, I
50 HLIN I,31−I, 15−I
60 VLIN I,15−I, I
70 VLIN I, 15−I,31−I
80 NEXT I
90 GOTO 90
```

Lesson 8

SKETCH PAD

ASC, CALL, DIM, NULL, STRING, VARIABLE

This program shows you how to draw on your TV with your Imagination Machine. You will use the left hand controller to move a colored square on the screen. All eight colors are available and may be drawn on the screen at once.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 7 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 8. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 8. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 8.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 8, follow the loading instructions on Chapter 5, page 32. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 90. This should position your BASIC TUTOR tape for Lesson 8.

ENTER SKETCH PAD

Type in the following 13 program statments. Remember to press the **RETURN** Key after each statement.

```
10    DIM H$( 1 )
20    COLOR =1: SHAPE =15
30  X=16:Y=8:  CALL  17046
40  H$=  KEY$  ( 2 )
50    IF  H$=" ! "  STOP
60    IF  H$=" ? "  THEN    GOTO  30
70    IF  H$=" N "  THEN  Y=Y-1
80    IF  H$=" S "  THEN  Y=Y+1
90    IF  H$=" E "  THEN  X=X+1
100    IF  H$=" W "  THEN  X=X-1
110  H=  ASC  ( H$ )-48
120    IF  H>-1  IF  H<8  THEN    COLOR =H
130    PLOT  X,Y:  GOTO  40
```

RUN SKETCH PAD

After you've entered this program, type **RUN** and press
RETURN. A black screen will appear with a yellow rectangle
in the center.

Use the left hand controller only with this program. Move the
joystick (knob) up, down, left or right and see how the box
on the screen follows your movement. Now press any
number between zero and seven and see the color change.

Here's the complete list of colors and their numbers:

0 = light green	4 = white
1 = yellow	5 = light blue
2 = dark blue	6 = purple
3 = red	7 = orange

When you wish to clear the screen, just press the **CL** (clear)
button on the hand control and the screen will clear with a
single box left on it in the last color you entered. To stop the
program, press the **FIRE** button or the **EN** (enter) key on the
left controller.

CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 8?

You can now continue with Lesson 8 to see how Sketch Pad works and learn how to write certain programming statements, or you can go directly to Lesson 9. If you want to go directly to Lesson 9 (Kaleidoscope) and your BASIC TUTOR tape is still in the machine, type **CLOAD**, follow the instructions on the screen and the computer will automatically begin loading. If you want to go directly to Lesson 9 (Kaleidoscope) and your BASIC TUTOR tape is not in the machine, see page 32, Chapter 5 for instructions If you want to continue with Lesson 8, type **LIST** and press the **RETURN** Key which will bring back the list of your program (or you can even type **RUN** and press the **RETURN** Key and continue running the program).

HOW SKETCH PAD WORKS

Stop your program by pressing the **EN** Key. List it by typing **LIST** and pressing **RETURN**. With this program we use the left hand controller to give inputs to the computer, and based upon these inputs, the program takes a different action. This is similar to other programs where we have used the main typewriter keyboard to give inputs.

The controller gives letter-type codes into the computer similar to the main keyboard.

Moving the joystick in four directions creates the letters, N, S, E, W. The **CL** key inputs a code for a question mark(**?**). Pressing the **EN** key or **FIRE** button gives a code for an exclamation mark (**!**). Similarly, pressing any of the number buttons (0-7) gives the code for that number. Any of these inputs from the hand controller will be assigned to the variable H$ and used by the program.

SKETCH PAD
LESSON 8

```
        ( RUN )
           │
   10  ┌─────────────┐
       │ DIMENSION   │
       │   H $       │
       └─────────────┘
           │
   20  ┌─────────────┐
       │ SET COLOR   │
       │ AND SHAPE   │
       └─────────────┘
           │
   30  ┌─────────────┐
       │  CENTER     │ ─ ─ ─ ─ ─ ─►
       │ CURSOR AND  │
       │CLEAR SCREEN │
       └─────────────┘
           │
┌──────────┐   ┌─────────────┐
│  HAND    │──►│   INPUT     │
│ CONTROL  │   │DIRECTION OR │
└──────────┘ 40│  NUMBER     │
           │
   50      ◇ FIRE ──Y──►( STOP )
           │
           │ N
   60      ◇ CL
           │
           │ N
   70      ◇ N  Y──►┌─────────────┐
   80    W   E      │  ADJUST     │
   90      ◇        │  X AND Y    │
  100        S  N   └─────────────┘
           │
  110      ◇ φ-7 ──Y──►┌─────────────┐
  120      │           │  CHANGE     │
           │ N         │  COLOR      │
           │           └─────────────┘
  130  ┌─────────────┐
       │   PLOT      │ ─ ─ ─ ─ ─ ─►
       │   X, Y      │
       └─────────────┘
```

Screen diagram (30): dimensions 8, 16, 16, 32

Screen diagram (130): X, Y
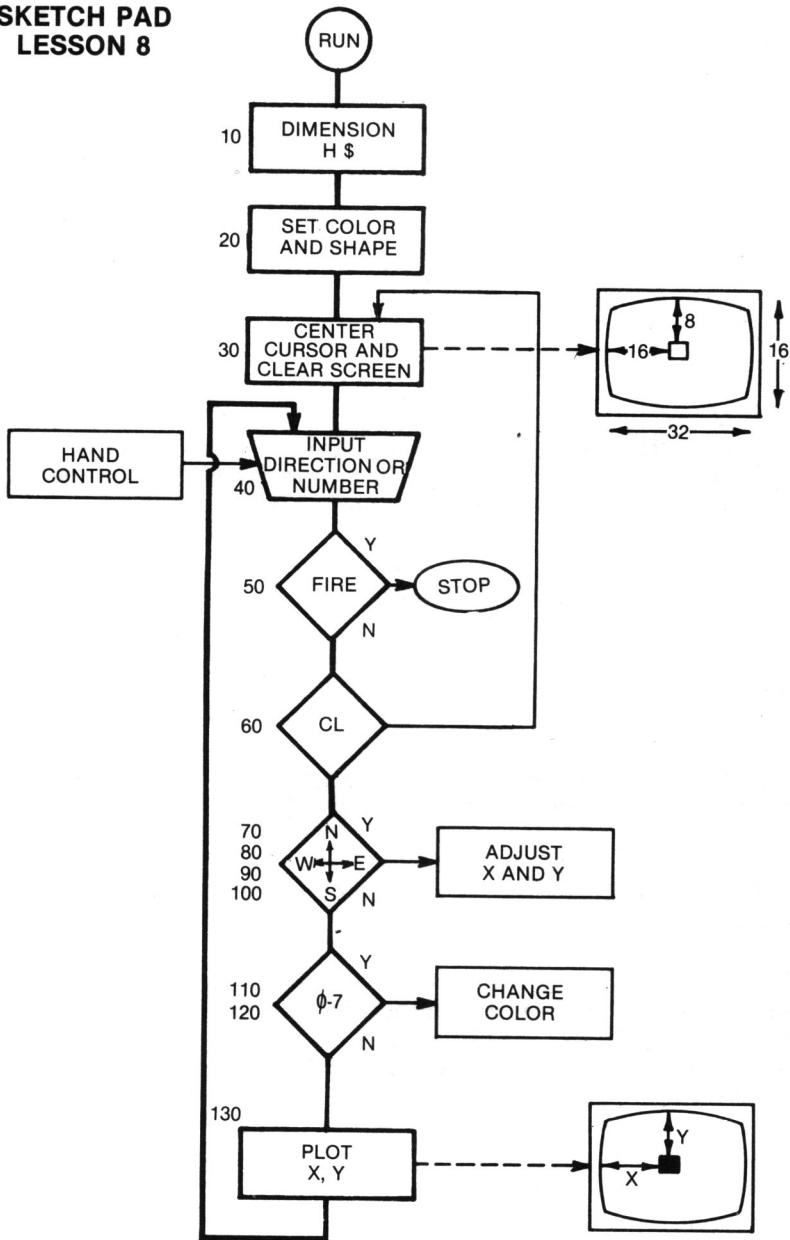
Line 10 is called the dimension statement. Whenever we want to assign letter values (as opposed to a number value) to a variable, we use what is known as a string variable name. This is like any other variable name except that the last letter of the name is a $. H$ in Line 10 is the string variable. In order to use a string variable in a program, we have to inform the Imagination Machine what will be the maximum amount of letter characters the variable will contain so the computer will reserve enough memory space. This is done in the dimension statement. Line 10 tells the computer to reserve memory space for the variable H$.

Line 20 assigns to the system variable COLOR a value of 1 (which is the Imagination Machine code for yellow). It also assigns shape number 15 for the PLOT statement (which will be a small rectangle).

Line 30 assigns variables **X** and **Y** the coordinates of where we will start plotting on the screen. These coordinates are approximately in the center of the screen. Line 30 also introduces a BASIC statement that says **CALL 17046**. BASIC is a language that uses English-type words such as PRINT to tell the computer what we want done. When the computer sees a word like PRINT, it goes to an internal procedure of steps to execute this PRINT command. It turns out that there are a number of internal procedures that are useful but cannot be accessed by using a keyword. Instead, we allow a general BASIC statement CALL   which allows you to access a procedure by a number instead of a keyword. CALL 17046, in this case calls upon a procedure of steps inside the computer that clears the screen to all black. 17046 is the number for this procedure. (See dictionary section of this Lesson for additional numbers.)

Line 40 sets the string variable H$ to have a value equal to the letter sent by the left hand controller.

Line 50 looks at the value that string variable H$ was assigned in Line 40. If it is equal to an exclamation point (!), then the EN or FIRE button on the controller was pressed and we direct the program to stop.

Line 60 looks at the value that string variable H$ was assigned in Line 40. If its value was a question mark (?) then the CL button was pressed on the controller and it tells the program to go back to Line 30 which will clear the screen and initialize X and Y to their starting coordinates in the center of the screen.

Line 70 looks at the value of the string variable H$ that was assigned in Line 40. If it is equal to a value of "N" then the joystick on the controller was pushed up and we want to change the Y coordinate of the square so it moves up on the screen. Since as we go up on the screen Y decreases, we subtract 1 from the present value of Y.

Line 80 again looks at the value assigned to H$. If it is equal to "S" then the joystick on the controller was pushed down and we want to change the coordinate of Y so that it moves down on the screen. As we go down on the screen Y increases, we add 1 to the present value of Y.

Lines 90 and 100 similarly look at the value assigned to H$. If the value is "E" then the joystick was pushed to the right and we want to change the X coordinate of the square so that it moves to the right. Since as we go to the right X increases, we add 1 to the present value of X. In Line 100 the computer looks for the "W" value. This means the joystick was moved to the left and since X decreases as we move to the left, we subtract 1 from the present value of X.

Line 110 further examines the input from the controller to see if a number key was pressed. We use a function command known as **ASC**. This function (ASC) returns the ASCII (pronounce) code for alphanumeric characters (an alphanumeric character is any letter A-Z, any number, and all punctuation). Each alphanumeric character has its own code number. (*See dictionary of this Lesson for all ASCII codes.*) Since the ASCII code for the numbers 0-9 are 48-57 respectively, we take the ASCII code of the character in H$

and subtract 48. We assign this value to the number variable
H which we look at further in Line 120.

Line 120 has two **IF** statements. We want to see if the value
of H is between 0 and 7, which would mean that the
character in H$ was one of the number keys. If it is in this
range, then we assign to the system variable COLOR the
value of H. This would be the next color used in the PLOT
statement of Line 130.

Line 130 plots a rectangle with shape of 15 at location X,Y.
X and Y were initialized in Line 30 and are varied in Line 70
through 100 depending on the movement of the joystick. The
color for this PLOT was initialized in Line 20 to be yellow
but can be varied by Lines 110 and 120 depending upon the
inputs. It then directs the program to go back to Line 40
which looks at the input from the controller again.

DICTIONARY

This dictionary contains detailed and comprehensive
definitions on all of the BASIC keywords introduced in
Lesson 8. If you wish to learn more about these keywords
and their variations, continue reading this dictionary section.
If you wish to go on to Lesson 9, simply type **CLOAD**, press
the **RETURN** Key and follow instructions. The Imagination
Machine will begin loading with Lesson 9.

LETTER OR STRING VARIABLES — stores letters or words.

A variable is used to store letter or word values as opposed
to number values. Any variable name that ends in a $ can be
used to have a value which is letter or words. When we
show a value of a letter variable we enclose it in a pair of
quotes.

Examples:

A$ = "BILL"   Variable A$ will have a value of BILL.
DAY$ = "MONDAY" Variable DAY$ will have a value of
                            MONDAY.
K$ = "Y"   Variable K$ will have a value of Y.

All the rules of number variable names apply to the letter variable names except the last character is to be a $.

The maximum number of characters a letter variable can have is 100 (similar to the maximum value of a number variable is 999999999.9999).

A$ = "ABC....HBDKQR" maximum is 100 letters

* * * *

All letter variables must be dimensioned (DIM) in a program before they can be used.

DIM — tells the computer to reserve space in its memory for letters, words, or tables of numbers.

DIM A$(5) dimensions (tells computer) to reserve enough memory to hold up to a 6 character value for A$. The amount of space you reserve is one more than the number used in the DIM statement because the computer starts its count at 0. DIM CAT$(1) reserves 2 memory spaces for CAT$.

After a DIMA$(5) statement is read by the computer, the variable A$ can be assigned any combination of up to 6 characters.

* * * *

A$ = "HAT" , A$ = "DOG" , A$ = "HOUSE" , etc.

The DIM statement also sets initial values of the string variable to contain nothing (see NULL page 00).

STRING VARIABLES AND ASSIGNMENT STATEMENTS

When a string variable is used in an Assignment Statement, it takes the first character of the value on the right side of the equals, and assigns it to the first position of the letter variable on the left. Next, it takes the 2nd character and assigns it to the 2nd position.

A way to view the storage of string variable values is that the dimension (DIM) statement reserves a series of memory positions or boxes to contain the value of the variable.

☐ ☐ ☐ ... ☐

1st 2nd 3rd   Last   Series of memory boxes
Box Box Box   Box   to contain values.
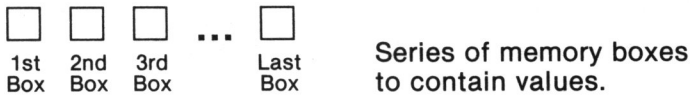
Each letter of the word value is placed in it's own box. The computer takes the first letter of the value and assigns it in the first box; the second letter goes in the second box etc.

As an example:

10 DIM A$(3) —
DIMENSIONS 4 memory location for variable A$

☐   ☐   ☐   ☐

20 A$ = "DEFG" —
Assignment of letter variable A$

1st position of A$ gets "D"

D   ☐   ☐   ☐

2nd position of A$ gets "E"

D   E   ☐   ☐

3rd position of A$ gets "F"

D   E   F   ☐

4th position of A$ gets "G"

D   E   F   G

* * * *
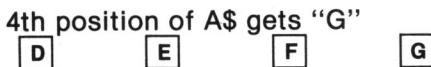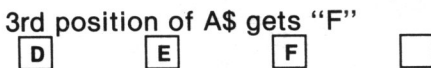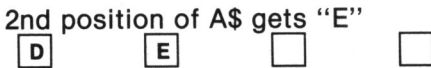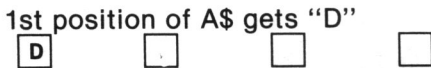
If the number of characters assigned to a letter variable is less than the number of character positions, the remaining positions stay unchanged.

10 DIM B$(3)
20 B$ = "DEFG" — | D | | E | | F | | G |

Assignment of letter variable B$ (4 places).

30 B$ = "XYZ"

1st position of B$ gets "X"

| X | | E | | F | | G |

2nd position of B$ gets "Y"

| X | | Y | | F | | G |

3rd position of B$ gets "Z"

| X | | Y | | Z | | G |

4th position of B$ retains its previous value.

| X | | Y | | Z | | G |

* * * *

In any assignment statement, the last values assigned will always be the new value.

String variables can be used in input, comparison and IF Statements.

```
 10  DIM NAME$(5)
 20  INPUT NAME$
 30  IF NAME$ = "GEORGE" THEN GOTO 100
 40  STOP
100  PRINT "HI GEORGE"
```

* * * *

A$ means to use the string variable A$ starting in its first position. APF BASIC allows you to select other starting positions than the first. For Example A$(6) would mean to use A$ with its starting point at the 7th position. As a further example look at the following.

102

```
10  DIM A$(20)
15  A$ = "                    "
20 INPUT "FIRST NAME", A$(0)
30  INPUT "LAST NAME",A$(10)
40  PRINT A$
```

Type **RUN** and then press the **RETURN** Key. It will ask for first name, then last name. Back inputs are stored in A$ and with different starting positions. Then A$ is printed completely by Line 40.

NULL — A NULL is defined as a character that is nothing. It can't be printed on the screen. When strings are initially dimensioned, each position is automatically assigned the code for a NULL character.(The ASCII code for a NULL is 0).

Example:

10 DIM A$(20): REM the DIM statement sets each position of A$ to have a NULL value.
**20 INPUT "FIRST NAME",A$(0):**REM input goes into A$ starting at position 0.
**30 INPUT "LAST NAME",A$(10)**REM input goes into A$ starting at position 10.
40 PRINT A$

Try the above with your name.

Type **RUN** and then press the **RETURN** Key. It will ask for **first name**, then **last name**, and then it prints both names.

If you type in John for the first name and Jones for the second name, Line 40 will print **JOHNJONES** without spaces. This is because we only filled the first 4 positions allotted for the first name (we allowed 11). The remaining 6 contain NULLS. When A$ was printed, those 6 NULLS do not show up on the screen, and the last name was printed right next to the first name. Try adding Line 15 as follows:

15A$ = "   "

* * * *

STRING CONCATENATION — Since assignment to strings can be done by pointing to a particular character position, the concatenation or joining of 2 strings into 1 can easily be accomplished.

Example:

```
10  DIM FIRST$ (7), LAST$ (7), WHOLE$ (16)
20  INPUT "FIRST", FI$
30  INPUT "LAST", LA$
35  WH$ (0) = LA$:  WH$ (8) = ",":  WH$ (9) = FI$
40  PRINT WH$
```

Try running the above and see what WH$ is.

ASC — The ASC function converts a character string variable to the ASCII integer code.

ASC("A") — creates the ASCII code for the Letter A (which is 65).

ASC(A$) — will produce the ASCII value of the first character in A$.

ASC(A$(4)) — will produce the ASCII value of the fifth character in A$.

Example:

```
10  DIM A$(5)
20  A$ = "ABCDEF"
30  PRINT ASC(A$)
40  PRINT ASC(A$(4))
50  FOR I = 2 TO 5
60  PRINT ASC(A$(I)),
70  NEXT I
```

**CHR$** — The CHR$ function produces an ASCII character from an ASCII Code.

PRINT CHR$(65) — Prints the character whose ASCII code is 65. This will be the letter A.

A$(4) = CHR$(72) — The letter H (ASCII Code 72) is assigned to the variable A$, in the 5th position.

LEN — A function which returns the length (or number of non null characters) of a string variable.

LEN(A$) — Returns with the length of the string A$.

```
10 DIM A$(10)
20 A$ = "ABCD"
30 PRINT LEN (A$)
```

Result is 4

A complete list of the ASCII codes used to place characters on the screen is shown on the following page.

Decimal	ASCII Character	Decimal	ASCII Character
64	@	32	
65	A	33	!
66	B	34	,,
67	C	35	#
68	D	36	$
69	E	37	%
70	F	38	&
71	G	39	'
72	H	40	(
73	I	41	)
74	J	42	*
75	K	43	+
76	L	44	,
77	M	45	-
78	N	46	.
79	O	47	/
80	P	48	0
81	Q	49	1
82	R	50	2
83	S	51	3
84	T	52	4
85	U	53	5
86	V	54	6
87	W	55	7
88	X	56	8
89	Y	57	9
90	Z	58	:
91	[	59	;
92	/	60	<
93	]	61	=
94	↑	62	>
95	←	63	?

CALL X — tells the computer to call a "MACHINE LANGUAGE" ROUTINE that starts at memory location X.

In this manual we talk about the computer language BASIC. Inside the computer is another language called "INTERNAL MACHINE LANGUAGE." When we use words like PRINT, LIST, etc., the computer breaks them down into routines written in machine language to get the function done. It is beyond the scope of this manual to go any further into machine language program except with reference to the CALL Statement.

Additional routines inside the computer can be gotten at through the CALL Statement. The most useful are

CALL 17046 — this will clear the screen to all black.
CALL 34040 — turns tape deck motor and audio on.
CALL 34061 — turns tape deck motor off.
CALL 17006 — creates a small beep noise.
CALL 17026 — creates a combination whistle beep noise.

(For more information write for a free technical reference manual.)

Lesson 9

KALEIDOSCOPE

Some video artists use computers to create very beautiful and interesting designs. With this program, you will create a colorful pattern that constantly changes and never repeats.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 8 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 9. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 9. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 9.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 9, follow the loading instructions on Chapter 5, page 32. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 105. This should position your BASIC TUTOR tape for Lesson 9.

ENTER KALEIDOSCOPE

Begin by typing the following instructions and remember to press the **RETURN** Key after each line. Notice that line 90 is wider than the screen. Don't press the **RETURN** Key until you've typed all of this instruction.

```
10    SHAPE =15
20    CALL 17046
30    X= INT ( RND ( 0 )*16 )
40    Y= INT ( RND ( 0 )*8 )
50    PLOT 16-X,8+Y
60    PLOT 16+X,8-Y
70    PLOT 16-X,8-Y
80    PLOT 16+X,8+Y
90    IF  RND ( 0 )=.1 THEN   COLOR = RND ( 0 )*8
100   IF  KEY$ ( 0 )=" "  STOP
110   GOTO 30
```

RUN KALEIDOSCOPE

After your program has been entered, type **RUN** and press
RETURN Key. On a color TV, you will see a pattern of
squares that change rapidly as new colors are added.

The secret (or the art) in creating programs like this that are
fun to watch is in deciding how the pattern shall be formed
and how often the colors shall change. Your program uses
random numbers to modify both pattern and color.

You can stop the program at any time by pressing the space
bar below the keyboard. To start again, type **RUN** and press
RETURN, as before. Each time the program starts, the
screen is cleared and a new pattern is generated.

CONTINUE WITH ANOTHER LESSON, OR
LEARN MORE FROM LESSON 9?

You can now continue with Lesson 9 to see how
Kaleidoscope works and learn how to write certain
programming statements, or you can go directly to Lesson
10. If you want to go directly to Lesson 10 (Interest/
Depreciation/Calculator) and your BASIC TUTOR tape is still
in the machine, type **CLOAD**, follow the instructions on the
screen and the computer will automatically begin loading.
If you want to go directly to Lesson 10 (Interest/
Depreciation/Calculator) and your BASIC TUTOR tape is not
in the machine, see page 32, Chapter 5 for instructions.

If you want to continue with Lesson 9, type **LIST** and press the **RETURN** Key which will bring back the listing of your program (or you can even type **RUN** and press the **RETURN** Key and continue running the program).

HOW KALEIDOSCOPE WORKS
(See flow chart on following page)

Stop your program by pressing the space bar. List it on the screen by typing **LIST** and pressing the **RETURN** Key.

The computer creates random numbers automatically. You can use this capacity to build patterns that seem to change by themselves. In this program, random numbers are used to pick a position to place a colored square on the screen. We use this position to symetrically place 4 squares on the screen — then we pick another random number to place 4 more squares on the screen.

These patterns are added continuously to the image. To make the pattern more interesting, the color is modified periodically. As with the position of the squares, there's no way to know exactly how often the colors will change or what color will be next.

While the exact appearance of designs like this can't be predicted, the aspects that make them interesting or pleasing can be controlled by the programmer or artist.

Line 10 sets the system variable called SHAPE to have a value of 15 — which selects a solid rectangle — as the SHAPE for the next PLOT command.

Line 20 clears the screen to black using the CALL command.

Line 30 sets X to a random number between 0 and 15. The RND function gives a value between 0 and .99 and multiply that by 16 which gives a number between 0 and 15.8400. Then we use the INT function to make this number into an integer between 0 and 15.

Line 40 sets Y to a random number between 0 and 7 (similar to the procedure followed in Line 30).

111

KALEIDOSCOPE
LESSON 9

RUN

10	SELECT SHAPE
20	CLEAR SCREEN

30	PICK A
40	RANDOM X AND Y

50 PLOT

60 PLOT

70 PLOT

80 PLOT

90 COLOR CHANGE ?

PICK RANDOM COLOR

KEYBOARD

100 INPUT KEYBOARD

SPACE BAR — Y — STOP

N

110 REPEAT
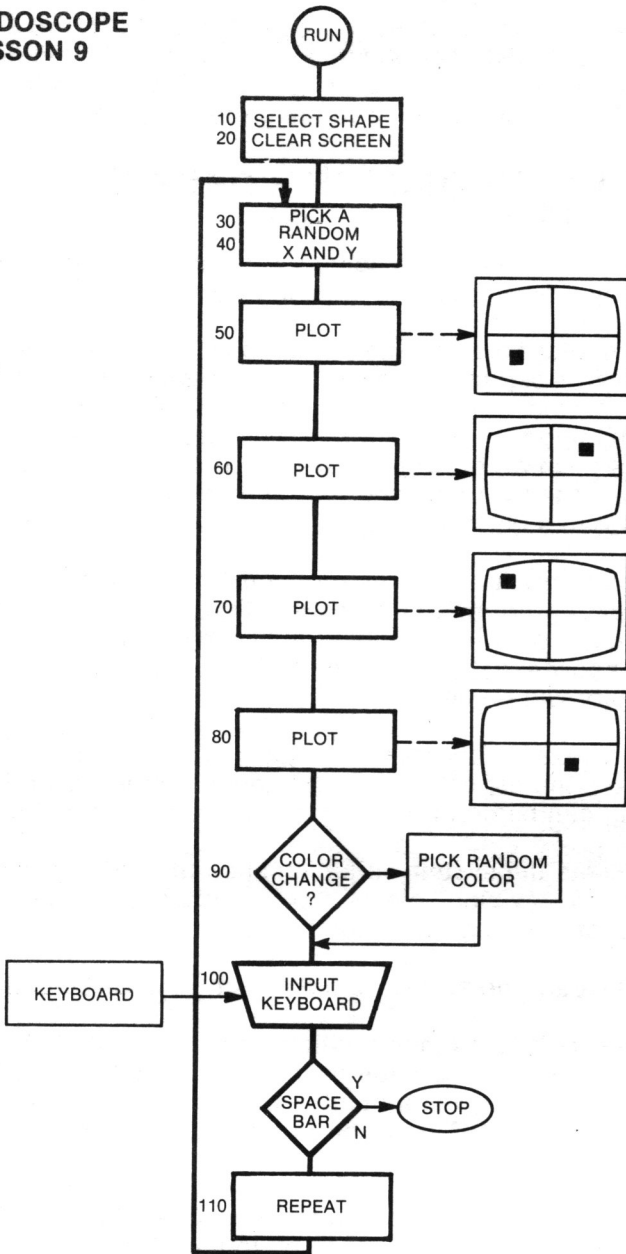
Line 50 PLOTS a box in the lower left section of the screen, using the values of X and Y set in lines 30 and 40. Since 16 is the horizontal center of the screen, we subtract X from 16 which gives us a location somewhere on the left side of the screen. To Y we add 8 which gives us a location on the lower one half of the screen. Therefore, the PLOT in Line 50 is somewhere in the lower left hand corner of the screen.

Line 60 PLOTS a box in the upper right section of the screen. It too uses the values of X and Y set in Lines 30 and 40. Again, since 16 is the horizontal center of the screen, by adding the X value to 16, we get a location somewhere on the right side of the screen. To Y, we subtract 8 which gives us a location in the upper half of the screen. Therefore, the PLOT in Line 60 is somewhere on the upper right hand side of the screen.

Line 70 PLOTS a box in the upper left section. This is done in a similar manner to Lines 50 and 60.

Line 80 PLOTS a box in the lower right section. Again, see Lines 50 and 60.

Line 90 again calls for the RND function. If this function produces a number equal to .1 we then will change the system variable COLOR to a new number which is selected randomly.

Line 100 stops the program if the character pressed on the keyboard is a space bar.

Line 110 sends the program back to Line 30 to produce a new X and Y for the four PLOT statements.

Lesson 10

INTEREST / DEPRECIATION / CALCULATOR

PRINT USING

With this program you can compare interest rates on loans or investments and even see the effects of inflation. The formula uses monthly compounding and rounds off to the next whole penny in calculating the figures.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 9 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 10. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 10. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 10.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 10, follow the loading instructions on Chapter 5, page 32. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 119. This should position your BASIC TUTOR tape for Lesson 10.

ENTER INTEREST / DEPRECIATION / CALCULATOR

Type in each line of the following 11 program statements. Remember to press the **RETURN** Key after each line.

```
1    REM       ...INTEREST...
10   CALL 17046
20   INPUT "PRINCIPAL ( $ )",T
30   INPUT "YEARLY INTEREST ( % )",I
40   INPUT "NUMBER OF MONTHS",N
50   PRINT : PRINT "  MO.      INTEREST TOTAL"
60   FOR L=1 TO N
70   A=I*T/1200
80   T=T+A
90   PRINT USING 100,L,A,T
100 :### ###,###.## #,###,###.##
110   NEXT L
```

Eight Spaces

RUN INTEREST / DEPRECIATION / CALCULATOR

Type **RUN** and press the **RETURN** Key to run your program.
It begins by asking for the principal in dollars. Enter **12000**
as the number of dollars you would like to invest and press
the **RETURN** Key. Do <u>not</u> type a **$** symbol or a **,** in a number
you enter.

Now the computer is requesting the interest rate in percent.
Enter **13.6** as your yearly interest rate and press the **RETURN**
Key.

Select **12** as the number of months you wish to examine and
press the **RETURN** Key again. The program will
automatically compute the interest earned and the total
investment for each of the next 12 months. If you've used
the numbers we suggested, you will see that the interest
paid on the last month would be $153.94 and that your
investment has grown to $13,737.67.

The program stops automatically after printing the results.
To enter another calculation, type **RUN** and press the
RETURN Key.

Here's another example to show you how your program can calculate the effects of inflation. Just consider inflation to be a "negative interest" and use the same equations. To see how much your $12,000 would be worth 24 months from now with a yearly inflation rate of 10%, just enter **12000** for the principal, **−10** for the yearly interest, and **24** for the months.

As the final figures show, the value of your investment will drop $82.49 in the final month and reach a low of $9,816.54 after two years. The moral is clear: invest your money at high enough interest to offset the effect of inflation.

CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 10?

You can now continue with Lesson 10 to see how Interest / Depreciation / Calculator works and learn how to write certain programming statements, or you can go directly to Lesson 11. If you want to go directly to Lesson 11 (Expressway) and your BASIC TUTOR tape is still in the machine, type **CLOAD**, follow the instructions on the screen and the computer will automatically begin loading. If you want to go directly to Lesson 11 (Expressway) and your BASIC TUTOR tape is not in the machine, see page 32, Chapter 5 for instructions. If you want to continue with Lesson 10, type **LIST** and press the **RETURN** Key which will bring back the listing of your program (or you can even type **RUN** and press the **RETURN** Key and continue running the program).

HOW INTEREST / DEPRECIATION / CALCULATOR WORKS
(See flow chart on following page)

Type **LIST** and press the **RETURN** Key to see your program. All the calculations are done in Lines 70 & 80 where the new total is set equal to the old total plus the interest for that month. This is the formula for compound interest on a monthly basis. In this program, the yearly interest is divided by 1200 to get the monthly interest in a decimal format.

The rest of this program is necessary for prompting the user, collecting information, and printing the results on the screen.

INTEREST / DEPRECIATION
LESSON 10

```
                    ( RUN )
                       │
                 ┌───────────┐
            10   │   CLEAR   │
                 │  SCREEN   │
                 └───────────┘
                       │
            20
            30                        ┌─────────────────────┐
┌──────────┐  40   ╱─────────╲        │ PRINCIPAL ($)?■     │
│ KEYBOARD │─────▶│  INPUT   │ ─ ─ ─ ▶│ YEARLY INTEREST     │
└──────────┘       │   DATA   │        │   (%)?■             │
                   ╲─────────╱         │ NUMBER OF           │
                       │               │ MONTHS?■            │
                       │               └─────────────────────┘
                 ┌───────────┐         ┌─────────────────────┐
            50   │  PRINT    │ ─ ─ ─ ▶ │ INTEREST TOTAL      │
                 │  TITLE    │         │ MO. INTEREST        │
                 └───────────┘         │ TOTAL               │
                       │               └─────────────────────┘
                 ┌───────────┐
            60   │  LOOP     │◀──────────┐
                 │   L       │           │
                 └───────────┘           │
                       │                 │
                 ┌───────────┐           │
            70   │ COMPUTE   │           │
                 │ INTEREST  │           │
                 └───────────┘           │
                       │                 │
                 ┌───────────┐           │
            80   │ADD INTEREST│          │
                 │ TO TOTAL  │           │
                 └───────────┘           │
                       │                 │
            90   ┌───────────┐   ┌────────────────────────┐
           100   │  PRINT    │ ─▶│ $110.00   $12110.00    │
                 │USING FORMAT│   └────────────────────────┘
                 └───────────┘
                       │
           YES         ◇
                      ╱ ╲
                   ╱ MORE ╲
           120     ╲  ?   ╱
                      ╲ ╱
                       │ NO
                    (STOP)
```
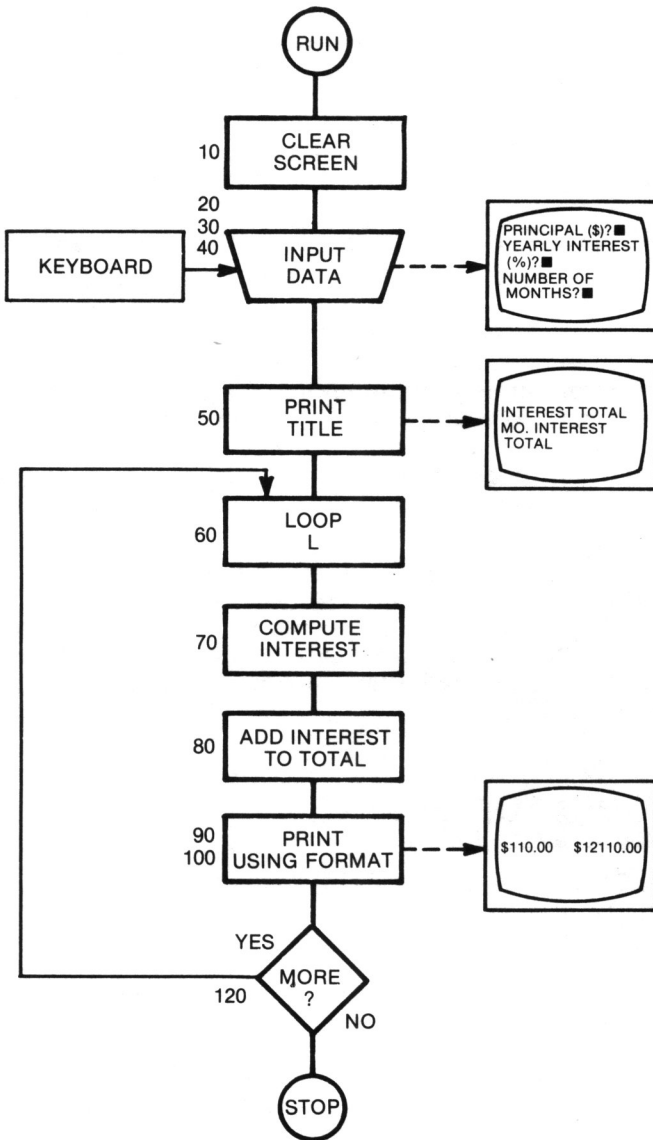
Line 10 is a call statement that clears the screen and leaves it black.

Line 20 prints **"PRINCIPAL ($) ?"** and sets variable T equal to the number you type on the keyboard.

Line 30 prints **"YEARLY INTEREST (%) ?"** and sets I equal to the number you type on the keyboard.

Line 40 prints **"NUMBER OF MONTHS ?"** and sets N equal to the number you type on the keyboard. This is the number of months we wish to do the calculation for.

Line 50 prints the headings for your printout. This is especially useful if you have a hard copy printer and would like a table that's easy to read. You can see that it prints the month number in the first column, the interest received in that month, and then the total amount you have accumulated.

Lines 60 thru 100 work together as a loop. The loop repeats once for every month you have requested. In the second example, N was set to 24 and this loop repeated 24 times.

The instructions inside the loop (lines 70, 80 and 90) are performed each time the loop repeats.

Line 70 computes the interest for this time period and sets it equal to the variable A. The equation is the formula for the interest for 1 month.

Line 80 adds this interest to the previous total and creates a new accumulated total.

Line 90 print the results. PRINT USING means to print in a particular format. We want to put the values of L, A & T using the format specified in Line 100 which is:

1. PRINT L (the number of the month) USING three places.
2. PRINT A (the monthly interest) USING three places, a comma, three more places, a decimal point, and two places.

119

3. PRINT T (the total) USING one place, a comma, three places, a comma, three more places, a comma, a decimal point, and two places.

In this way, the answer will be displayed in an even column of figures. Room on the screen has been set aside so that the months can go to 999, the interest can go to 999,999.99, and the total can reach 9,999,999.99.

Line 100 — notice the first character after the Line # is a colon (:). This tells the computer that this line is a definition for a PRINT USING statement. The pound symbol or number symbol (#) is used to set up a format of how we want values printed. PRINT USING allows us to specify where we want values printed on a line, and if we want a comma (,) or a dollar sign ($) used. It is a very useful statement for printing results of money calculations.

Line 110 completes the loop so that the program will repeat until all months have been calculated and displayed on the screen.

DICTIONARY

This dictionary contains detailed and comprehensive definitions on all of the BASIC keywords introduced in Lesson 10. If you wish to learn more about these keywords and their variations, continue reading this dictionary section. If you wish to go on to Lesson 11, simply type **CLOAD**, press the **RETURN** Key and follow instructions. The Imagination Machine will begin loading with Lesson 11.

PRINT USING — allows you to print values (number or strings) in a specific format that you can specify.

General format:

PRINT USING   format, value 1, value 2. . .

The format specifies how to and where to place the values. It may be a program statement, # or a string variable name. The format uses the number sign symbol to set up fields (1 field for each value to be printed).

120

The pound sign or (#) is used in the format definition for each character or digit. Pound symbols are grouped together fields. The PRINT USING statement takes the list of values and places them respectively into each field that is defined.

Example:

10: ### ##.#
20 PRINT USING 10, 123, 45.64

Line 10 is a format definition to be used in a PRINT USING statement. It shows 3 pound symbols which comprised the first field. The fields must always be distinguished by having a blank space around them. Next it has 2 pound signs (#), a decimal point and a third pound sign (#). This is the second field. The first field will display a number with up to 3 digits and have no decimal places. The second field will display a 2-digit number plus a single decimal place. Line 20 says to print (using the format specified in Line 10) and have the value of 123 printed in the first field and 45.64 printed in the second field.

* * * *

Another example:

10: ## ##.# ##.## ##.### ##.####
20 A = 1/3
30 PRINT USING 10, A, A, A, A, A,
RUN

The computer will print:

.3   .33   .333   .3333

WORDS CAN BE MIXED IN THE FIELDS IN THE FORMAT SPECIFIED

Example:

10: FIRST ##### SECOND #####
20 PRINT USING 10, 100, 200
RUN

The computer will print:

FIRST 100 SECOND 200

* * * *

COMMAS CAN BE INTERMIXED WITH A FIELD DEFINITION

Example:

10: AMOUNT ###,###.##
20 A = 12345.67
30 PRINT USING 10, A
RUN

The computer will print

AMOUNT 12, 345.67

Note commas are displayed on the screen

* * * *

DOLLAR SYMBOLS ($) CAN BE PART OF THE FIELD
SPECIFIED

Example:

10: AMOUNT $$###,###.##
20 A = 10000
30 B = 6500
40 PRINT USING 10, A
50 PRINT USING 10, B

We use a double dollar sign ($$) which specified that a
floating dollar sign is to be printed — will float so that it is
always just to the left of the most significant digit.

THE FORMAT SPECIFIER CAN BE CONTAINED AS THE
VALUE OF A STRING VARIABLE

Example:

10 DIM A $ (10)
20 A$ = "AMOUNT ####"
30 AMOUNT = 200

40 PRINT USING A$,AMT
RUN

The computer will print:

AMOUNT 200

               * * * *

THE FIELD OVERFLOW

If a value is too great to be printed in a specified field, the computer continuously prints a series of asterisks (*****) instead of the value.

10: AMOUNT ##
20 PRINT USING 10,500
RUN

Amount is going to show with 2 asterisks.

               * * * *

THE ASTERISK SIGN CAN BE USED IN A FIELD SPECIFIER TO INDICATE YOU WANT UNUSED PLACES OF THE FIELD FILLED-IN.

Example:

10: AMOUNT**####
20 FOR J = 10 TO 3000 STEP 800
30 PRINT USING 10,J
40 NEXT

The computer will print:

AMOUNT****10
AMOUNT***810
AMOUNT**1610
AMOUNT*ᵃ2410

Lesson 11

EXPRESSWAY

MUSIC, PEEK

One of the more interesting things you can do with your personal computer is design your own video games: we selected EXPRESSWAY as a game program, because it's easier to enter and understand and it's designed with full color, sound effect, and a scorekeeper.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 10 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 11. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 11. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set·to begin Lesson 11.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 11, follow the loading instructions on Chapter 5, page 0. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 136. This should position your BASIC TUTOR tape.for Lesson 11.

ENTER EXPRESSWAY

Type in the 11 programming steps. Remember to press the **RETURN** Key after each programming statement.

```
10  X=15:  SHAPE  =15:T=0
20    COLOR  =1
30    PLOT   RND  ( 0 )*32,15
40    IF   KEY$  ( 1 )="W"  THEN  X=X-1
50    IF   KEY$  ( 1 )="E"  THEN  X=X+1
60    IF   KEY$  ( 1 )="N"  STOP
70    COLOR  =6
80    PLOT  X,5
90    IF   PEEK  ( 704+X )=159  MUSIC  "1":T=T+1
100   PRINT  T
110    GOTO  20
```

RUN EXPRESSWAY

Type **RUN** and press the **RETURN** Key. Use the right hand control to move the purple line. Move the knob left and right and see if you can avoid hitting the yellow squares. Adjust the volume on your TV to hear the beep when you hit something.

Notice your score at the left. The program is counting the total number of hits and printing this number on each new line at the bottom of the screen.

For EXPRESSWAY to be a real game, it must have a way to win and a way to lose. We leave the rules up to your imagination. Some ideas are, for a start, you might try to see how long you can run your program without hitting anything. Another idea would be the opposite: see how many points you can get in the shortest time.

Stop the program by pushing the knob to the north position. To start it again, just type **RUN** and press the **RETURN**. Each time you run your program, the score is reset to zero.

CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 11?

You can now continue with Lesson 11 to see how Expressway works and learn how to write certain programming statements, or you can go directly to Lesson 12. If you want to go directly to Lesson 12 (Time Machine) and your BASIC TUTOR tape is still in the machine, type **CLOAD**, follow the instructions on the screen and the computer will automatically begin loading. If you want to go directly to Lesson 12 (Time Machine) and your BASIC TUTOR tape is not in the machine, see page 32, Chapter 5 for instructions. If you want to continue with Lesson 11, type **LIST** and press the **RETURN** Key which will bring back the listing of your program (or you can even type **RUN** and press the **RETURN** Key and continue running the program).

HOW EXPRESSWAY WORKS
(See flow chart on following page)

Push the knob on the right hand control forward to stop your program. Type **LIST** and press the **RETURN** Key to list the program on the screen. Each line in your program is also shown in the flow chart diagram.

Now we will look at each instruction and see how this program uses the scrolling effect to create a game.

Line 10 is actually three separate instructions, separated by a colon (:). The letter X is a VARIABLE used to store the left to right position on the screen of the purple line. This sets X to 15, or the center of the screen.

The shape to be plotted is 15, a solid rectangle. This shape is used in plotting the purple line and the yellow obstacles.

The third instruction sets T to zero. This will keep count of the number of times you hit an obstacle.

Line 20 sets the COLOR to 1 which is yellow.

Line 30 plots a yellow box. It will be in the 15th row down (at the bottom row) and its horizontal position is selected by a random number. The left/right position is a random number between zero and 31 or somewhere between the left and right edges of the screen.

RUN

10   INITIALIZE

20
30   PLOT YELLOW SQUARE

40
50   INPUT DIRECTION LEFT OR RIGHT

HAND CONTROL

60   N   YES   STOP

NO

70
80   PLOT PURPLE SQUARE

90   PURPLE COVERS YELLOW ?   YES   INCREASE TOTAL SCORE

NO

100   PRINT OLD SCORE MOVE SCREEN UP

110   START OVER
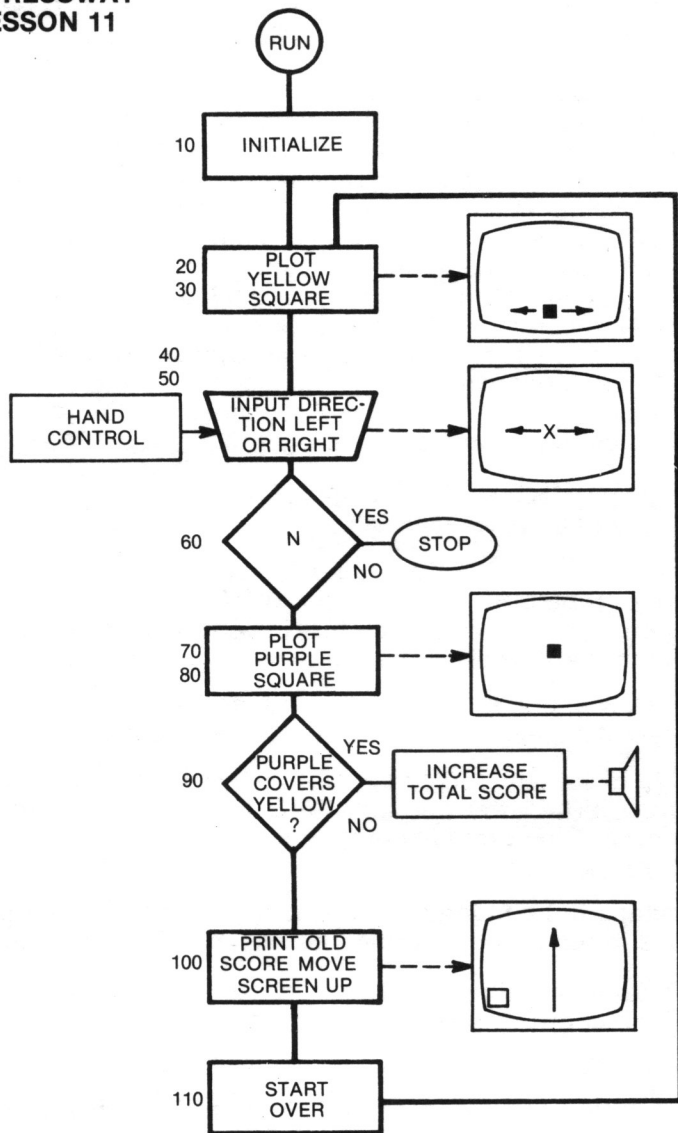
Line 40 decreases X if the knob on the right hand control is pushed to the left. X is the horizontal position and was initialized in Line 10.

Line 50 increases X if the knob is pushed to the right. X is the horizontal position and was initialized in Line 10.

Line 60 stops the program if the knob is pushed forward.

Line 70 changes the COLOR being plotted to purple.

Line 80 plots a purple rectangle on row five, with the left/right position set by X.

Line 90 determines if the purple rectangle we just plotted will intersect with one of the yellow obstacles. If so, we will create a music note and add 1 to **T**. T keeps track of the number of times you hit the obstacle. The PEEK instruction allows you to look inside the computer's memory. Line 90, we PEEK at a location which has an address of 704 + the value of X. This particular memory location contains the coded information for what appears on the screen right below where we plotted the purple square. We compare the value returned by the PEEK instruction to 159 – 159 is a code that would be returned if a yellow box was to be shown on the screen. If the code is 159, then there will be a yellow box below the purple one and they will intersect. Another keyword used in this line is MUSIC. It is how we make musical notes come through your TV speaker. If the intersection does occur, we make a musical note and then add 1 to the score (variable T).

Line 100 prints the score at the bottom of the screen. This also causes the screen to move up 1 line. It causes the screen to SCROLL. Since we've plotted the various yellow and purple boxes on the screen, when scrolling occurs, it causes them to shift up 1 line giving the effect that they are moving.

Line 110 goes back to Line 20 and begins the process all over again.

DICTIONARY

This dictionary contains detailed and comprehensive definitions on all of the BASIC keywords introduced in Lesson 11. If you wish to learn more about these keywords and their variations, continue reading this dictionary section.

If you wish to go on to Lesson 12, simply type **CLOAD**, press the **RETURN** Key and follow instructions. The Imagination Machine will begin loading with Lesson 12.

<center>* * * *</center>

All information that the computer has, whether it is program statements, values of variables, or information to be displayed on the screen is contained as numeric codes in memory locations.

PEEK — examines a location in the computer memory and returns the numeric value contained in that memory location. This value can be printed, assigned to a variable or used like any other numeric value.

Details of what is contained where in memory are available in the Imagination Machine Technical Reference Manual for those interested.

For BASIC TUTOR we will just concern ourselves with examining locations which contain codes for what is to appear on the TV screen.

There are 512 boxes on the TV screen. There are 512 corresponding memory locations, each containing a code for what will appear on the screen.

These 512 memory locations have addresses of 512-1023. 512 is the memory address that contains a code for what will appear in the upper left corner of the screen. Memory location 513 corresponds to the screen box just to the right of 512, etc. Memory location 1023 corresponds to the low right hand corner of the screen.

The value returned by a PEEK instruction will be some number between 0 and 255. This number can be printed, assigned to a number variable, or used anywhere a number variable can.

PRINT PEEK(512) — will print to the screen the value of memory location 512.
Reset the machine, press **EN** on the controller. The cursor will appear. Type:

PRINT PEEK(512)

The computer will print: **128**

128 is a code that corresponds to the black square on the screen that is in location 512 at this time.

CODE USED IN SCREEN MEMORY

Decimal	Will Appear On Screen	Decimal	Will Appear On Screen
0	@	32	
1	A	33	!
2	B	34	"
3	C	35	#
4	D	36	$
5	E	37	%
6	F	38	&
7	G	39	'
8	H	40	(
9	I	41	)
10	J	42	*
11	K	43	+
12	L	44	,
13	M	45	-
14	N	46	.
15	O	47	/
16	P	48	0
17	Q	49	1
18	R	50	2
19	S	51	3
20	T	52	4
21	U	53	5
22	V	54	6
23	W	55	7
24	X	56	8
25	Y	57	9
26	Z	58	:
27	[	59	;
28	\	60	<
29	]	61	=
30	↑	62	>
31	←	63	?

64-127	Same as 0-63 but reverse video.
128-143	Shape 0-15 in green color.
144-154	Shape 0-15 in yellow color.
160-175	Shape 0-15 in dark blue color.
176-191	Shape 0-15 in red color.
192-207	Shape 0-15 in white color.
208-223	Shape 0-15 in light blue color.
224-234	Shape 0-15 in purple color.
240-255	Shape 0-15 in orange color.

* * * *

MUSIC plays notes in the TV speaker.

Using the MUSIC command, you can play music on your computer using numbers. The numbers 1 through 7 play the musical scale.

To play one octive higher, place a multiply sign (*) in front of the number, and to play one octave lower, place a divide (/) sign in front of the number.
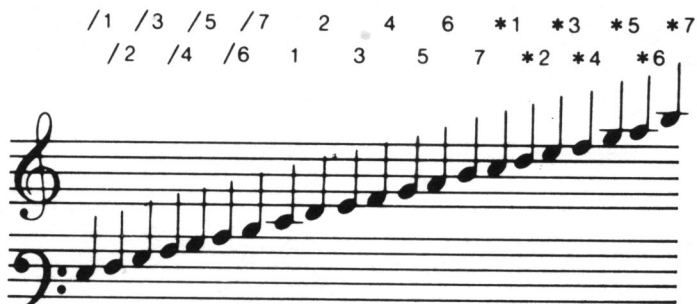
You can also play sharps and flats by typing a " + " or " − " sign in front of the note.

The number zero (0) is used to hold the previous note.

Leaving a space between numbers creates a pause in the music.

The sequence of numbers and notes to be played is placed inside a pair of quotes following the word **MUSIC**.

Example:

MUSIC"3212333"
MUSIC"3212333000"

* * * *

If the music plays too quickly you can slow it down by placing a **0** between each note.

MUSIC"302010203030"

* * * *

Spaces create a different sound.

MUSIC"3 2 1 2 3 3 3"

* * * *

If you want to use the same sequence of notes over and over you can assign them to a letter variable.

10 DIM A$(5)
20A$ = "1234"
30 MUSIC A$

Some other examples are:

Example 1:

10 FOR I = 1 TO 10
20 MUSIC "/1/2/3/4/5/6/71234567*1*2*3*4*5*6*7"
30 NEXT I

Example 2:

Let's try a song.

10 MUSIC"32123 3 3 2 2 2 3 5 5 32123 3 3 3 2 2 321"

* * * *

Two or more strings can be played together by separating them with a comma after the world **MUSIC**.

Example:

Using A$, B$

10 DIM A$(6),B$(6)
20 A$ = "5552332"
30 B$ = "77665"
40 MUSIC A$, B$

Lesson 12

TIME MACHINE

POKE

If you've been wanting a digital stop watch, this program will do the trick. In prints the hours, minutes, and seconds on the screen and updates the display ten times a second.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 11 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 12. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 12. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 12.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 12, follow the loading instructions on Chapter 5, page 32. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 152. This should position your BASIC TUTOR tape for Lesson 12.

ENTER TIME MACHINE

Type the following 10 program statements. Remember to press the **RETURN** Key after each line. Notice that Line 90 is longer than the screen. The computer automatically goes down to the next line, but do not press the **RETURN** Key until you have finished entering that line statement.

```
10   H=0:M=0:S=0
20    FOR N=1 TO 16: PRINT : NEXT N
30   S=S+.1
40    IF S=60 THEN S=0:M=M+1
50    IF M=60 THEN M=0:H=H+1
60   POKE 40960,2: POKE 40961,0
70   PRINT "HOURS      ";H
80   PRINT "MINUTES    ";M
90   PRINT USING "SECONDS ##.#",S
100   IF KEY$ ( O )<>" " GOTO 30
```

RUN TIME MACHINE

This program couldn't be easier to run. Just type **RUN** and press the **RETURN** Key. To stop the clock, press the space bar on your keyboard. To start the clock again, run your program.

CONTINUE WITH TIME MACHINE OR GO ON TO TAPE 2 which contains Lessons 13-22. Stop your program by pressing the space bar. Type **LIST** then press the **RETURN** Key to list it.

HOW TIME MACHINE WORKS
(See flow chart on following page)

The computer takes a certain amount of time to perform each instruction in your programs. In this program Lines 30 through 100 take almost exactly one tenth of a second. In designing TIME MACHINE, we selected these instructions so that the program itself can be used as a clock.

Each time the computer loops back to Line 30, about .1 second has passed. The seconds, minutes, and hours are then updated and the results printed on the screen.

**TIME MACHINE
LESSON 12**

RUN

10 | SET TO ZERO
HOURS MINUTES
AND SECONDS

20 | CLEAR
SCREEN

30 | ADD .1
SECONDS

40 | 60
SECONDS
? — Y → ADD
ONE MINUTE
N

50 | 60
MINUTES
? — Y → ADD
ONE HOUR
N

60 | START
NEW SCREEN

70
80
90 | PRINT
TIME → HOURS 0
MINUTES 0
SECONDS 0

100 | KEYBOARD → INPUT
KEYBOARD

SPACE
BAR — Y → STOP
N
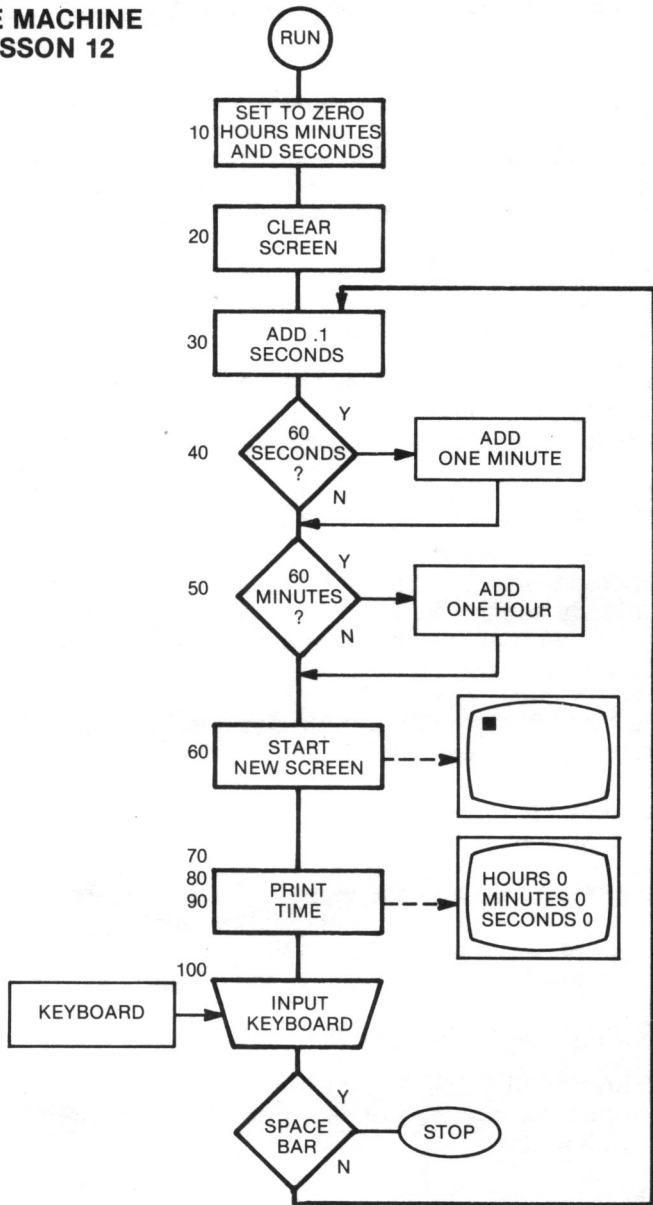
Line 10 sets variables H, M, and S to zero. These variables are used to keep track of the hours, minutes, and seconds.

Line 20 clears the screen by printing 16 blank lines.

Line 30 advances the clock by adding .1 to S. S is the symbol for number of seconds. Every time we do Line 30 we will add 1/10th of a second to the counter S.

Line 40 looks at the value of S. If it has reached 60 then it is time to increment the minute counter (M) and clears seconds counter back to zero.

Line 50 looks at the M (minutes counter). If it has reached 60, then it is time to reset (or clear) the minutes counter to zero and add 1 to the H (hours) counter.

Line 60 introduces a BASIC command called POKE. POKE allows us to change the contents of a memory location. Normally each time we do a print statement the position on the screen where to do the next print statement is automatically set by the computer. This usually is set to be the beginning of the next line unless changed by a semicolon or comma.

You see that when you do a print, the position moves down the screen. Line 60 uses 2 POKE instructions which change the position that will be used in the next print statement. Instead of allowing the computer to print where it wants to we can force it to print it where we want it to.

Line 70 prints hours and the value of H, etc. Where it starts to print was set by Line 60. It also automatically sets where the next print statement occurs and that will be directly below this one.

Line 80 prints **"MINUTES"** and the value of M.

Line 90 prints **"SECONDS"** and then the number of seconds with one decimal place. It uses a PRINT USING statement to accomplish this.

Line 100 sends the computer back to Line 30 if the keyboard is not equal to a space (" ").

DICTIONARY

This dictionary contains detailed and comprehensive definitions on all of the BASIC keywords introduced in Lesson 12. If you wish to learn more about these keywords and their variations, continue reading this dictionary section. If you wish to go on to Lesson 13, simply type **CLOAD**, press the **RETURN** Key and follow instructions. The Imagination Machine will begin loading with Lesson 13.

POKE — place a number in a memory location.

POKE 784,65 places the number 65 in memory location 784. 784 is one of the memory locations that stores a code of a character that is to be displayed on the TV. We have changed this memory location to have the code for the letter A and this will appear on the screen. A complete set of codes and what appears on the screen is in Lesson 11. The memory locations which correspond to the image for the screen are from 512 to 1023.

SOME USEFUL ROUTINES

You might find that there are some routines or functions not built into the BASIC interpreter that you need. Most can be implemented using either PEEKS/POKES/CALLS in machine language or with subroutines written in BASIC.

"PRINT AT"

If you want to print anywhere on the screen, use a routine that changes the cursor pointer.
The position of the cursor is stored in 2 memory locations — 40960 and 40961.

```
10  GOTO 100
20  REM: ROUTINE TO MOVE CURSOR POSITION TO
VALUE OF CU.
25  POKE 40960, INT (CU/256):  POKE 40961, CU:  RETURN
100  FOR I = 1 TO 32: PRINT: NEXT: REM CLEAR SCREEN
TO GREEN
```

```
110  INPUT "LINE AND COLUMN TO START PRINT", L, C
120  CU = 512 + L*32 + C:  GOSUB 25:  PRINT "HI"
125  INPUT "MORE", K:  IF K  <>  TO 0 GOTO 100
```

Line 100 will clear the screen to all green.

Line 110 asks for a horizontal line number (L), and a vertical column number (C) where you want to start printing at. It converts these to the actual memory location on the screen. 512 is the top corner of the screen so we add to it the number of lines (L) times 32 (32 characters per line), and add the column number.

Line 25. We enter line 25 with the variable CU having the memory location we want to change the cursor pointers to have. Since the cursor pointer is a double memory location (it takes 2 memory locations) we have to break CU into 2 numbers. The most significant number (into 40960) is the number of 256's contained in CU. We get this by taking the integer portion of CU/256. Into 40961 we have to POKE the remainder of dividing 256 into CU. The POKE instruction automatically does this. So, into 40961 effectively goes CU — INT(CU/256)*256.

Run the program and enter various numbers for L (between 0 and 15), and C (between 0 and 31). After their entry you will see the word **"HI"** printed on the screen. The program next says **MORE?**, and just press the **RETURN** Key to run again.

Chapter 6
TAPE TWO PROGRAMS (lessons 13-22)
Introduction to Tape Two

The second half of BASIC TUTOR, Lessons 13 thru 22, builds on the programming skills you developed in Chapter 5. There are games to test your memory, your logic, your skill, and your luck; programs to help you create your own electronic music and video art; even programs that show you how to use formulas, and one that will tutor you in math.

The lessons on this second tape are recorded just like programs you purchase or record yourself. We have eliminated the special input monitoring mode and the BASIC on-screen dictionary of keywords. Instead, we have used that space in the computer's memory to utilize the longer programs used in Lessons 13 thru 22. When a lesson is loaded, you can run it immediately. You do not have to type in the steps or even press the Escape Key for immediate entry. Once you have loaded the lesson, you are ready to run it. We have assumed that you have learned the elementary typing procedures correctly in Lessons 1-12.

When you run these programs, you'll see that there are more action-programs and that all lessons contain more content. Just because they are longer and do more interesting things, however, doesn't mean that these programs are difficult or hard to understand. They use many of the BASIC words you already know, now put together in a series of simple instructions. In other words, we use the foundation of what you have learned in Chapter 5 and build upon it.

Loading Instructions for Tape 2

1. Type **CLOAD** and press **RETURN** Key.
2. Wait for response and follow directions on screen.
3. Rewind tape all the way. Press **PLAY** then press **RETURN** Key.
4. Adjust volume control on computer console and listen to audio introduction on Lesson 13.
5. Press digital tape counter and set it back to 000.
6. Wait for **OK** to appear on left side of screen, and then type **RUN** and press **RETURN** Key.

Remember, once you have loaded the program you have no more typing to do for program execution. After you type **RUN** and press **RETURN** the computer is all set to begin the program.

Lesson 13

ROCK / SHEARS / PAPER

REM

This electronic version of a classic game is easy and fun as you play against the computer and try to guess a winning combination. You will select one of three choices and then will win, lose, or tie the round, depending on the action of the computer.

In this program, you will use a full screen scoreboard to see the moves and total the results. Begin by following these instructions and loading Lesson 13 from the cassette:

1. Type **CLOAD** and press the **RETURN** Key.
2. Wait for response and follow directions on screen.
3. Rewind tape all the way. Press **PLAY** then press the **RETURN** Key.
4. Adjust volume control on computer console and listen to audio introduction on Lesson 13.
5. Press digital tape counter and set it back to 000.
6. Wait for **OK** to appear on left side of screen, and then type **RUN** and press the **RETURN** Key.

RUN ROCK / SHEARS / PAPER

After you type **RUN** and press the **RETURN** Key, the screen will clear and you will see the program format.

```
1 = ROCK      2 = SHEARS    3 = PAPER
SCORE:
     HUMAN: 0
COMPUTER: 0
         TIE: 0
PRESS THE NUMBER OF YOUR CHOICE
YOUR PICK IS >>->
MY PICK IS >>->
```

Press **1, 2** or **3** on the main keyboard to signal your choice of ROCK, SHEARS or PAPER. The computer will next choose one of these and its pick will be shown below yours on the screen. Adjust the volume control on your TV screen to hear the score and watch the results displayed on the screen.

If your guess is the same as the computer's, the outcome will be a tie. Any other combination will decide a winner. Here are the rules of the game:

1. ROCK breaks SHEARS
2. SHEARS cut PAPER
3. PAPER wraps ROCK
4. The same pick of yours and the computer is a tie.

If you pick ROCK and the computer picks SHEARS, you will win because the ROCK breaks the SHEARS. If you pick ROCK and the computer picks PAPER, however, you would lose because PAPER wraps the ROCK. If you pick SHEARS, you would win if the computer picks PAPER since SHEARS cut PAPER; however, you would lose if the computer picks ROCK since the ROCK would break the SHEARS. If you selected PAPER, therefore, you would win against ROCK but lose to SHEARS.

As you can see there's no sure win and any guess you pick could be a winner or a loser. Run the game a few rounds and see if your score is any higher than the computer's. Remember, each time you press an entry (1, 2, or 3) it is shown on the screen, along with the computer's pick and the total score. To play another round, just enter **1, 2** or **3** AGAIN and watch the score change. Notice how this program changes some of the words and numbers from the screen while keeping others the same.

HOW ROCK / SHEARS / PAPER WORKS

After setting the scores to zero at the beginning of the game, the computer repeats this program endlessly. You can stop the program by pressing the **BREAK** Key. Look at the flow chart diagram below and see how this program is organized into sections.

The computer generates a random number between 1 and 3 to create its guess. In this particular program, there's no special logic or design for creating the computer's choice.

144

RUN

LIST 1,6

INITIALIZE

LIST 40,10

DISPLAY
SCORE BOARD

```
1 = ROCK
2 = SHEARS
3 = PAPER
SCORE:
 HUMAN: 0
 COMPUTER: 0
 TIE: 0
```

LIST 140,9

KEYBOARD

INPUT
YOUR CHOICE
(K $)

YOUR CHOICE
(1,2,3)?

PRINT
CHOICE

ROCK SHEARS
OR PAPER

LIST 230,7

PICK
COMPUTER'S
CHOICE

PICKED

PRINT
CHOICE

ROCK SHEARS
OR PAPER

LIST 300,12

COMPUTE
RESULTS

WIN

YOU WIN

LOOSE

I WIN

TIE

A TIE
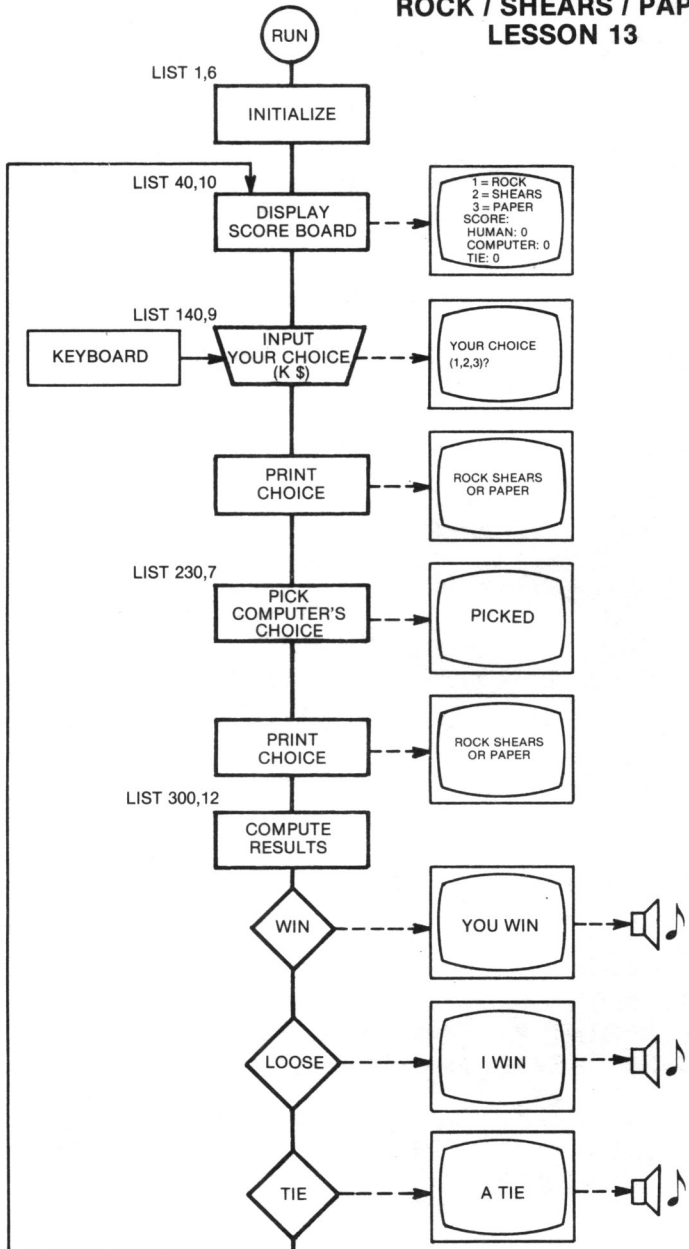
CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 13?

You can now continue with Lesson 13 to see how Rocks/Shears/Paper works and learn how to write certain programming statements, or you can go directly to Lesson 14. If you want to go directly to Lesson 14 (Probability) and your BASIC TUTOR tape is till in the machine, type **CLOAD**, follow the instructions on the screen and the computer will automatically begin loading. If you want to go directly to Lesson 14 (Probability) and your BASIC TUTOR tape is not in the machine, see page 32, Chapter 5 for instructions.

```
1   REM    ...ROCK.....
2   REM    ...SHEARS...
3   REM    ...PAPER....
10   DIM K$(1)
20  H=0:C=0:T=0
30   FOR L=1 TO 32: PRINT : NEXT L
40   REM    ...SCORE BOARD...
50   POKE 40960,2: POKE 40961,0
60   PRINT
70   PRINT " 1=ROCK   2=SHEARS   3=PAPER"
80   PRINT
90   PRINT "SCORE:"
100   PRINT "     HUMAN: ";H
110   PRINT " COMPUTER: ";C
120   PRINT "       TIE: ";T
130   PRINT
140   REM    ...YOUR CHOICE...
150   PRINT "PRESS THE NUMBER OF YOUR CHOICE"
160   PRINT
170   PRINT "YOUR PICK IS >>-> ";
180  K$= KEY$ (0)
190   IF K$="1" PRINT "ROCK  ": GOTO 230
200   IF K$="2" PRINT "SHEARS": GOTO 230
210   IF K$="3" PRINT "PAPER ": GOTO 230
220   GOTO 180
230   REM    ...COMPUTER'S CHOICE...
240  X= INT ( RND (0)*3+1)
```

```
250   PRINT "MY PICK IS >>->    ";
260   IF X=1 PRINT "ROCK   "
270   IF X=2 PRINT "SHEARS"
280   IF X=3 PRINT "PAPER "
290   PRINT
300   REM    ...RESULTS...
310 Y= ASC (K$)-48
320   IF Y=X GOTO 400
330   IF Y=1 IF X=3 GOTO 380
340   IF Y=2 IF X=1 GOTO 380
350   IF Y=3 IF X=2 GOTO 380
360   PRINT "              YOU WIN"
370 H=H+1: MUSIC "55": GOTO 40
380   PRINT "               I WIN"
390 C=C+1: MUSIC "11": GOTO 40
400   PRINT "              A TIE"
410 T=T+1: MUSIC "/1/1": GOTO 40
```

Stop your program by pressing **BREAK**. List the first few
instructions by typing **LIST 1,6** and pressing the **RETURN**
Key. This will list starting with the first instruction, the first
6 programming statements in the Lesson.

LIST 1,6 - The first 3 lines do absolutely nothing when the
program is run. They are REMARKS that we use to put in
comments, titles of program or directions in a program to
make it easier for us to read and understand. They are
BASIC command words called REMARK statements (REM)
and are useful when you read the listing. The computer sees
REM and looks for the next number statement after it. In
other words, the computer knows the REM statement is
there, and lists it on the screen when you call for a LIST, but
ignores execution of all remaining words after the REM and
just goes on to the next numbered statement.

We chose a variable K$ to store the number of your choice
that you will type on the keyboard in Line 10. We are going
to use the K$ function so it is not necessary to press the
RETURN each time you make your choice. Since you only
type one key at a time, this variable is dimensioned to hold
one character.

Three other variables are used to store the number of total points for the human, the computer, and tie scores. Picking H, C, and T for these variables makes the program easy to follow. Each are initialized to 0. Printing 32 blank lines clears the screen — this occurs in Line 30.

LIST 40, 10 - Now type **LIST 40, 10** and press the **RETURN** Key — the computer will now list the 10 programming statements starting with Line 40. Here is the section of the program that prints the scoreboard. Note the REM statement in Line 40. The two POKE instructions in Line 50 start the printing at the top left corner of the screen. We POKE the memory location that keeps track of where the cursor is positioned to. By starting at this position each time the scoreboard is printed, the words on the screen are always printed in the same place. Even though the scoreboard is printed many times, it will appear stationary. After each round of play, we return to this point in the program and print the scoreboard — the effect we generate is that many words do not appear to change even though the scoreboard is printed on the screen every time you make a choice.

LIST 140,9 - Line 140 begins the section where your choice is taken from the keyboard and used in the program. The term KEY$(0) simply means "look at the keyboard" and we set K$ equal to the key that you press.

The words **YOUR PICK IS** is printed on the screen, followed by the word describing your choice. Depending on the letter stored in K$, the program will print **ROCK, SHEARS,** or **PAPER.** This is done at Line 190-210 with 3 IF statements. If any are true, the correct word is printed and the program is sent to the next section, Line 230. If they are not true, Line 220 sends the computer back to 180 to look for an input. If you press any other key than **1, 2** or **3,** the computer simply is directly back again to Line 180. Depress the **5** key — you will note that nothing happens.

LIST 230,7 - Now the computer makes its choice by picking a random number. In Line 240 we set X equal to the integer (whole number, not a fraction) of a number between one and three. RND (0).3 + 1 produces a random number between 1-3.9; therefore, the integer portion is between 1 and 3.

The words **MY PICK IS** are printed, followed by **ROCK,**
SHEARS, or **PAPER** depending on the value of X. As before,
3 IF instructions determine which word is printed on the
screen.

LIST, 300,12 - Now you see how the results of the match are
calculated. Instructions in this section simply state the rules
of the game to the computer through a series of IF
statements so that it can automatically print the right
answer and keep score.

Line 310 is an instruction that sets Y equal to the ASCII
code number of the keys you typed earlier and subtracts 48.
ASCII code for 1, 2 or 3 are 49, 50, and 51 so Y becomes a
number between 1 and 3 like X is. If Y (your choice) is the
same as X (the computer's choice) there's a tie and the
program goes to Line 400.

The next 3 IF statements tell the computer which
combinations mean that the computer wins. If Y = 1 (ROCK)
and X = 3 (PAPER), the computer wins and goes to Line 380.
Similarly, if Y = 2 (SHEARS) and X = 1 (ROCK), the computer
wins and goes to Line 380 again. Finally, if Y = 3 and X = 2,
SHEARS cut PAPER and the victorious computer again goes
to Line 380.

As you might suspect, Line 380 prints **I WIN** on the screen.
The next line adds 1 to C, the computer score, and BEEPS
two notes ("11") through the TV speaker. Then it goes back
to Line 40 and starts another round.

Notice that the computer can win three ways. If none of
these are true, however, the program doesn't go to Line 380.
Instead it reaches Line 360 and the words **YOU WIN** are
printed. At Line 370 the human's score is increased and two
notes ("55") are BEEPED in the TV speaker. The computer
then goes back for another round to Line 40.

The other possibility is a tie, where you and the computer
both pick the same choice. If Y = X the program jumps to
Line 400 where **A TIE** is printed. In Line 410, the number of
ties is increased, and two low octave notes ("/1/1") are
BEEPED.

DICTIONARY

This dictionary contains detailed and comprehensive definitions on all of the BASIC keywords introduced in Lesson 13. If you wish to learn more about these keywords and their variations, continue reading this dictionary section. If you wish to go on to Lesson 14, simply type **CLOAD**, press the **RETURN** Key and follow instructions. The Imagination Machine will begin loading with Lesson 14.

REM is a Remark Statement

Begin an instruction with REM whenever you want to add a remark, title, or comment to your program. Everything in a REM statement line after the word REM is ignored by the computer.

1 REM MY FIRST COMPUTER
10 INPUT "YOUR NUMBER", N:REM GET THE OPERATORS NUMBER

When you list the program it shows what the computer will put on the screen but also shows you (the programmer) what you want the input statement to stand for.

Since anything on a line after the word REM is ignored by the computer, do *NOT* put statements in the same line with a REM at the beginning.

Example:

10 REM THIS IS A TEST: PRINT "HI"

Now try **RUN**.

You will see that the computer ignores the execution or this statement. Now type **LIST**, you will see that the computer remembers the statement and lists it but will not run it.

Lesson 14

PROBABILITY

ARRAYS

This program simulates the rolling of dice and draws a bar graph to show you the results. The random number generator in the computer is used to pick a number betweer one and six for each die. All the possible combinations — from snake eyes to boxcars — are recorded in the memory and displayed on the screen.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 13 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 14. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 14. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 14.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 14, follow the loading instructions on Chapter 5, page 32. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 11. This should position your BASIC TUTOR tape for Lesson 14.

RUN PROBABILITY

When you run this program, the computer throws the dice. It shows their values, the total of the dice and the total times (the number of times the dice have been thrown). It also prints a scoreboard on the screen and draws a graph, adding a box to the appropriate throw total (the total of the dice). The computer continues to throw the dice and continues to add a box to the appropriate graph segment until any line in the graph reaches the right edge of the screen. Then the program stops and the cursor appears at the top left.

After your program has stopped, run it again by typing **RUN** and pressing the **RETURN** Key. You will notice that the 7 often "wins" the bar graph race and that the 2 and the 12 never seem to come up very often. This isn't because the dice are fixed; instead, the odds or the probabilities of rolling 7 and rolling snake eyes are very different.

There are 36 different number combinations with 2 dice and only 1 of these combinations (1 and 1) will total 2.

The probability of rolling a 7 is much higher because there are 6 different ways to make a 7 with 2 dice. They are: 6 and 1, 5 and 2, 4 and 3, and 3 and 4, 2 and 5, and 1 and 6. Since there are 6 ways to roll a 7 and only one combination that will equal 2, you will roll a 7 much more frequently.

Here's a complete table of the totals for 2 dice, the number of combinations that will create this total, and the odds. If you run your program several times, you will see that the graph will tend to match this probability.

Total	Combinations	Odds	Probability
2	1	1:36	.028
3	2	2:36	.083
4	3	3:36	.111
5	4	4:36	.139
6	5	5:36	.167
7	6	6:36	.194
8	5	5:36	.167
9	4	4:36	.139
10	3	3:36	.111
11	2	2:36	.083
12	1	1:36	.028

HOW PROBABILITY WORKS

Look at the flow chart diagram and see how this program is organized into sections. First, the program initializes the variables we will use. Then the screen is cleared and formatted for our output display with the numbers of the possible throws (2-12) printed on the left edge.

The next three sections repeat over and over as each throw of the dice is simulated by the computer. The dice are "rolled" by using a random number generator. The scoreboard is printed with the number on each die, the total of both die, and the number of tries that have been rolled so far. Each roll of the dice is then plotted on the graph.

Finally, the program checks to see if the last box plotted on the graph is next to the right edge of the screen. If so, the program stops. If not, the program loops back and rolls the dice again.

CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 14?

You can now continue with Lesson 14 to see how Probability works and learn how to write certain programming statements, or you can go directly to Lesson 15. If you want to go directly to Lesson 15 (Temperature Converter) and your BASIC TUTOR tape is still in the machine, type **CLOAD**, follow the instructions on the screen and the computer will automatically begin loading. If you want to go directly to Lesson 15 (Temperature Converter) and your BASIC TUTOR tape is not in the machine, see page 32, Chapter 5 for instructions.

RUN

INITIALIZE

LIST 1,6 → FORMAT SCREEN

LIST 60,5 → ROLL DICE

LIST 110,6 → PRINT SCORE

LIST 160,4 → PLOT GRAPH

DONE ? → YES → STOP

NO

Screen 1 (FORMAT SCREEN):
```
2
3
4
5
6
7
8
9
10
11
12
```

Screen 2 (PRINT SCORE):
```
PROBABILITY CURVE
DIE 1:3
DIE 2:4

TOTAL: 7
TRIES: 1
```
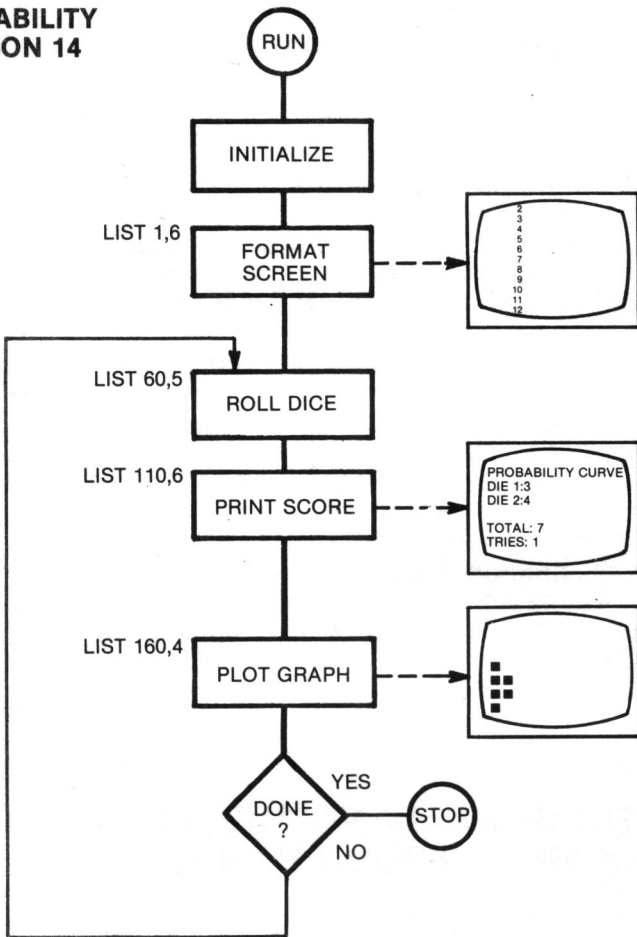
```
1   REM    ...PROBABILITY...
10  DIM X(12):T=0
20  SHAPE =4: COLOR =7
30  FOR N=1 TO 32: PRINT : NEXT N
40  FOR N=2 TO 9: PRINT " ";N: NEXT N
50  PRINT "10": PRINT "11": PRINT "12";
60  REM    ...ROLL DICE...
70  A= INT ( RND (0)*6+1)
80  B= INT ( RND (0)*6+1)
90  T=T+1
100 X(A+B)=X(A+B)+1
110 REM    ...PRINT SCORE...
120 POKE 40960,2: POKE 40961,0
130 PRINT "    ...PROBABILITY CURVE..."
135 PRINT
140 PRINT "DIE #1:";A;"  DIE #2:";B;"  TOTAL:";A+B
150 PRINT : PRINT "   TOTAL # THROWS:";T
160 REM    ...PLOT GRAPH...
170 PLOT X(A+B)+2,A+B+3
180 IF X(A+B)>28 STOP
190 GOTO 60
```

LIST 60,5 - These five instructions roll the dice and keep
track of the results. The letters A and B are each set to a
random number between 1 and 6. A is the roll of the first die
and B is the roll of the other. The number of tries (T) is
increased by 1. Line 100 makes use of the ARRAY X we set
up in Line 10. A + B is the total of the 2 dice. Therefore, the
variable in the ARRAY X that we want to add 1 to is X(A + B).
If a 2 and a 3 are rolled, for example, X(5) is increased by 1.
Type **LIST 110,6** to see the next section.

LIST 110,6 - Here's the section of your program that prints
the score.

The 2 poke instructions (Line 120) start all printing at the top
left corner of the screen. The print instructions print the text
in quotes, followed by the value of the letters. Extra spaces
are used to line up the scoreboard and make it look nice.
Now type **LIST 160,4** to see the last section of the program.

LIST 160,4 - Here's the section that plots the graph. The PLOT instruction uses 2 numbers to draw the SHAPE in the COLOR specified. The first number moves the PLOT to the right and the second number moves it down. X of A + B contains the total number of times this dice pair has occurred. It started at 0 and gets increased by 1 each time that number combination occurs; to its value, we have to add 2 since we started plotting a bar in column #2. If X(5) = 4, for example, the number 5 has been rolled 4 times and the PLOT will move 4 spaces to the right (plus 2 spaces to allow for the die numbers in the left margin).

Wait for your program to stop automatically, or press the **BREAK** Key to stop the program. Now press both the **RETURN** Key and the **REPT** Key simultaneously and hold them down while the screen is cleared.

Type **LIST 1,6** and press the **RETURN** Key. You have listed the first section of the program on the screen.

LIST 1,6 - This section of the program dimensions a numeric variable X to hold 12 different numbers. X is a collection or ARRAY of values that are grouped together by a common name. This is the first time we have used a DIM statement for numbers. Actually we are informing the computer we will have 12 numeric variables. They are X(1), X(2), X(3), etc.

X is the common variable name and the number in parenthesis is the index.

We will use these 12 variables to keep count of each of the possible dice throws — i.e., X(3) keeps track of the total number of times a 3 is thrown.

T is the total number of rolls and is initialized to 0.

SHAPE and COLOR are set to select a small orange square for drawing the graph. Line 30 clears the screen by doing 32 print statements. At Line 40 and 50 we print the vertical axis of the graph. It will be numbers 2-9 then 10, 11 and 12. Note the " " in Line 40 puts 2-9 in their most significant column.

In other words, the **9**
is under the **10**
11
12

Similarly we use A + B + 3 to get the horizontal position. The PLOT will move down the same number of spaces as the total of the roll (A + B) (plus 3 spaces to allow for the headings at the top of the graph).

In this way, a single orange box is added to the graph for each roll of the dice.

At Line 180 we see if any box we have added to has reached the right side of the screen. We compare its total number of throws to 28.

If the number of spaces to the right is greater than 28, the graph has reached the right edge of the screen and the program stops. If not, the computer goes to Line 60 and rolls the dice again.

DICTIONARY

This dictionary contains detailed and comprehensive definitions on all of the BASIC keywords introduced in Lesson 14. If you wish to learn more about these keywords and their variations, continue reading this dictionary section. If you wish to go on to Lesson 15, simply type **CLOAD**, press the **RETURN** Key and follow instructions. The Imagination Machine will begin loading with Lesson 15.

ARRAYS OR COLLECTION OF INFORMATION — Arrays allow us to work with groups of information. An array of variables has a common name, but each of its members can have a different value.

As an example, if we want to deal with the grades of 20 students in a class as a group, we would call the group "Students" and each student would have a different value for his grade. The students can be distinguished from each other by their name, but in BASIC saying Student Harry = 85 would be too complex, so instead, we given each student a number. In BASIC we would say

157

STUDENT(1) = 85
STUDENT(2) = 92
STUDENT(20) = 76
etc.

* * * *

The name of the group (or array) is STUDENT.
The first member has a grade of 85.
The second member has a grade of 92.
The 20th member has a grade of 76.

In BASIC, when you use an array, you must preceed it with a DIMENSION Statement which sets up how many members there will be.

Example:

10 DIM STUDENT(20)

This informs the computer it will have an ARRAY called STUDENT with 21 members. The first member is 0 and the last is 20 (Note start at 0 not 1).

* * * *

As we said before, the way we distinguish each member of ARRAYS is by giving each of the members a number. These are called the index (or subscript) of an array. The subscript is in a parenthesis (   ) and follows the name of the ARRAY.

Example:

DIM X(10) ... X is the name of the array
X(1) = 100 ... array X subscript 1 assigned the value of 100
X(2) = 250 ... array X subscript 2 assigned the value of 250

* * * *

In BASIC the subscript of an ARRAY must always be a number or a numeric value.

The subscript can be referenced by any variable that has a numeric value.

X(J)

J = 4 This is array X and the subscript is equal to the numeric value of J which is 4. X(J) is the same as X(4).

Another example:

```
10  DIM STUDENT(5)
20  FOR J = 1 TO 5
30  PRINT "STUDENT";J;"SCORE";
40  INPUT STUDENT(J)
50  NEXT J
RUN
```

You enter the scores 100, 95, 87, 89 and 56 and you will see the following printed:

```
STUDENT 1 SCORE? 100
STUDENT 2 SCORE? 95
STUDENT 3 SCORE? 87
STUDENT 4 SCORE? 89
STUDENT 5 SCORE? 56
```

In the above, you can see how powerful arrays become because you can refer to each member by a numeric variable.

* * * *

TWO DIMENSIONED ARRAYS

APF BASIC allows two dimensioned arrays. This means that an array can have two subscripts.

In the previous example, besides a grade, an age for the student would also be given. THESE ARE TWO VALUES OF INFORMATION WE WANT TO KEEP FOR EACH STUDENT.

As an example:

```
10  DIM STUDENT 20,2
20  FOR J = 1 TO 20
30  PRINT "STUDENT";J
40  INPUT "GRADE", STUDENT OF J,1
50  INPUT "AGE", STUDENT OF J,2
60  NEXT J
```

Line 10 dimensions or reverses space for a group called STUDENT, which will have a total of 20 members. Each member will have two values of information. The first subscript is the student number. The second tells whether you are dealing with the student's grade or age. If the second subscript is a 1, then his grade is being referenced; if it is a 2, then his age is being referenced.

STUDENT(3,2) — Age of Student #3
STUDENT(3,1) — Grade of Student #3
STUDENT(5,1) — Grade of Student #5
STUDENT(6,1) — Grade of Student #6
STUDENT(17,2) — Age of Student #17

You can see that the second subscript could be 3 or more, and we could group more information.

* * * *

DIM C(5,8) — dimensions memory to hold a table of numbers that is ~~five~~ *Six* rows by nine columns.

After dimensioning, the variable C could hold a total of ~~40~~ *54* (~~5 x 8~~ *6 x 9*) values:

C(1,2), C(5,2), C(3,8), etc.

Note: In APF BASIC, 1 is added to all dimensions automatically. Thus the words or strings will hold one more character than the number used in dimensioning them. Tables of numbers hold one additional row and column because zero can be used: C(0,0), C(0,8), C(5,0), etc.

* * * *

Letter variables can also be used as an array.

DIM B$(4,12) — dimensions memory to hold ~~four~~ *five* words of up to 12 letters each.

With letter arrays, the second number exactly equals the memory space required.

After dimensioning memory, you could write:

B$(1) = "MONDAY", B$(2) = "TUESDAY", B$(3) = "PAYDAY", etc.

160

USING ARRAYS FOR SORTING

```
10   REM THIS IS A SIMPLE SORT PROGRAM USING
     ARRAYS
20   REM IT IS USEFUL FOR RELATIVELY SMALL NUMBER
     OF VALUES
30   REM BUT IT SHOWS HOW ARRAYS CAN BE USED
40   DIM A(10):  REM WILL ALLOW UP TO 11 NUMBERS TO
     BE SORTED
50   FOR K = 1 TO 32:  PRINT:  NEXT
60   INPUT "HOW MANY NUMBERS",N
65   N = N − 1
70   FOR J = 0 TO N
80   PRINT "ENTRY #"; J + 1;
90   INPUT "      ",A(J)
100  NEXT J
110  REM NOW RESORT ARRAY A
120  FOR K = 0 TO N − 1
130  FOR J = K + 1 TO N
140  IF A(J)K > A(K) THEN
     TEMP = A(K):A(K) = A(J):A(J) = TEMP
150  NEXT:NEXT
160  FOR K = 0 TO N:  PRINT A(K):  NEXT.
```

Lesson 15

TEMPERATURE CONVERTER

ON-GOTO, ON-GOSUB

With this program you can quickly and easily convert temperatures between the American and the metric systems. A menu selection helps you pick the conversion you want and your answer will be calculated and displayed automatically. Numbers are rounded off or adjusted to the nearest degree.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 14 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 15. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 15. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 15.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 15, follow the loading instructions on Chapter 5, page 32. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 22. This should position your BASIC TUTOR tape for Lesson 15.

RUN TEMPERATURE CONVERTER

After you type **RUN** and press the **RETURN** Key, the converter will display a menu. Type either a **1** or a **2**, depending on whether you know the Fahrenheit or the Celsius units for the temperature you wish to convert, and press the **RETURN** Key.

Now type the number of degrees and press the **RETURN** Key. The computer will apply the correct formula, compute the answer, and round off any fraction to the nearest degree.

To convert another temperature, just press **Y** and the
RETURN Key.

Try comparing the temperatures at which water boils (212°F)
and freezes (32°F) in both systems.

HOW TEMPERATURE CONVERTER WORKS

A computer program like this makes it easy to solve a
problem where a formula can be used to find the answer. In
this example, two formulas are written into the program.

For converting Fahrenheit to Celsius the formula is:

C = 5/9*(F − 32)

and the formula for converting Celsius to Fahrenheit is:

F = C*9/5 + 32

When you select one or two from the menu, you are
directing the computer program to one of these formulas. In
either case, the program then asks for the number of
degrees and finds the corresponding temperature by solving
the conversion formula.

An important point to remember in writing programs like this
is to prompt the user with questions that make the program
easy to understand and use.

CONTINUE WITH ANOTHER LESSON, OR
LEARN MORE FROM LESSON 15?

You can now continue with Lesson 15 to see how Temperature
Converter works and learn how to write certain programming
statements, or you can go directly to Lesson 16. If you want
to go directly to Lesson 16 (Playback) and your BASIC
TUTOR tape is still in the machine, type **CLOAD**, follow
the instructions on the screen and the computer will
automatically begin loading. If you want to go directly to

164

TEMPERATURE CONVERTER
LESSON 15
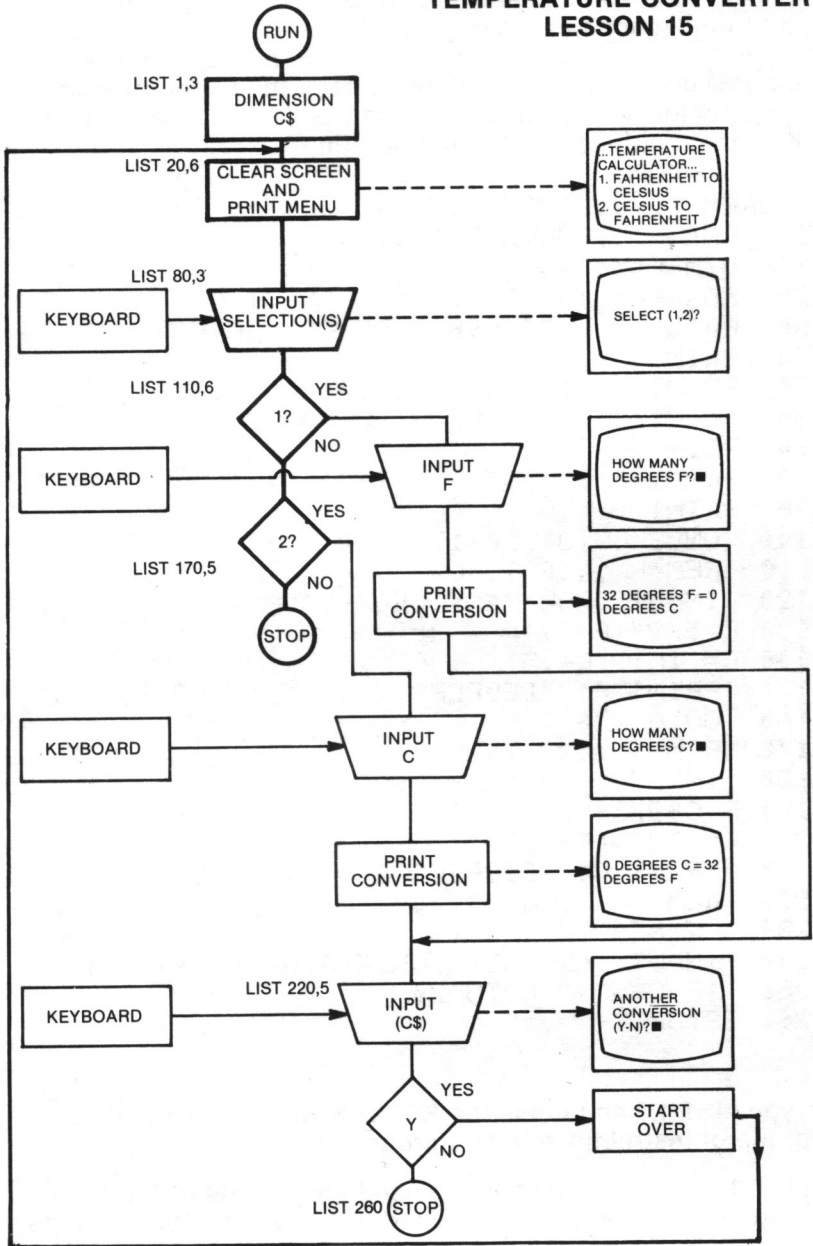
```
                              ( RUN )
                                 │
LIST 1,3          ┌──────────────────────────┐
                  │      DIMENSION           │
                  │          C$              │
                  └──────────────────────────┘
                                 │
                                 ▼
LIST 20,6   ┌──────────────────────────┐          ┌─────────────────────┐
            │     CLEAR SCREEN         │ - - - - → │ ...TEMPERATURE      │
            │         AND              │          │ CALCULATOR...       │
            │     PRINT MENU           │          │ 1. FAHRENHEIT TO    │
            └──────────────────────────┘          │    CELSIUS          │
                                 │                 │ 2. CELSIUS TO       │
                                 │                 │    FAHRENHEIT       │
                                 ▼                 └─────────────────────┘
LIST 80,3  ┌──────────┐    ┌──────────────┐        ┌─────────────────────┐
           │ KEYBOARD │ ─→ │   INPUT      │ - - - →│ SELECT (1,2)?       │
           └──────────┘    │ SELECTION(S) │        └─────────────────────┘
                           └──────────────┘
                                 │
LIST 110,6                      YES
                          ◇ 1? ─────────────┐
                           │                │
                          NO                │
           ┌──────────┐    ▼    ┌──────────┐│        ┌─────────────────────┐
           │ KEYBOARD │ ──────→ │  INPUT   ││ - - - →│ HOW MANY            │
           └──────────┘         │    F     ││        │ DEGREES F?■         │
                                └──────────┘│        └─────────────────────┘
                          YES              │
                     ◇ 2? ───┐             │
LIST 170,5            │      │  ┌──────────┐│        ┌─────────────────────┐
                     NO      └→│  PRINT   │├ - - - →│ 32 DEGREES F = 0     │
                   (STOP)      │CONVERSION││        │ DEGREES C            │
                              └──────────┘│        └─────────────────────┘
                                          │
           ┌──────────┐    ┌──────────┐   │        ┌─────────────────────┐
           │ KEYBOARD │ ─→ │  INPUT   │ - + - - →  │ HOW MANY            │
           └──────────┘    │    C     │   │        │ DEGREES C?■         │
                           └──────────┘   │        └─────────────────────┘
                                │         │
                           ┌──────────┐   │        ┌─────────────────────┐
                           │  PRINT   │ - + - - →  │ 0 DEGREES C = 32     │
                           │CONVERSION│   │        │ DEGREES F            │
                           └──────────┘   │        └─────────────────────┘
                                │ ←───────┘
LIST 220,5 ┌──────────┐    ┌──────────┐            ┌─────────────────────┐
           │ KEYBOARD │ ─→ │  INPUT   │ - - - - →  │ ANOTHER             │
           └──────────┘    │  (C$)    │            │ CONVERSION          │
                           └──────────┘            │ (Y-N)?■             │
                                │                  └─────────────────────┘
                               YES
                          ◇ Y ──────→ ┌──────────┐
                           │          │  START   │──┐
                          NO          │  OVER    │  │
LIST 260               (STOP)         └──────────┘  │
                                                    │
```

165

Lesson 16 (Playback) and your BASIC TUTOR tape is
not in the machine, see page 32, Chapter 5 for instructions.
If you want to continue with Lesson 16, type **LIST** and press
the **RETURN** Key which will bring back the listing of your
program (or you can even type **RUN** and press the **RETURN**
Key and continue running the program).

```
1   REM    ...TEMPERATURE...
2   REM    ...CONVERTER.....
10  DIM C$(1)
20  FOR L=1 TO 32: PRINT : NEXT L
30  PRINT "...TEMPERATURE CONVERTER..."
40  PRINT
50  PRINT "  1.FAHRENHEIT TO CELSIUS"
60  PRINT "  2.CELSIUS TO FAHRENHEIT"
70  PRINT
80  INPUT "     SELECT (1,2)",S
90  PRINT
100  ON S GOTO 110,170,260
110  REM    ...F TO C...
120  INPUT "HOW MANY DEGREES F",F
130 C=5/9*(F-32)
140 C= INT (C+.5)
150  PRINT F;" DEGREES F = ";C;" DEGREES C"
160  GOTO 220
170  REM    ...C TO F...
180  INPUT "HOW MANY DEGREES C",C
190 F=C*9/5+32
200 F= INT (F+.5)
210  PRINT C;" DEGREES C = ";F;" DEGREES F"
220  REM    ...MORE?...
230  PRINT
240  INPUT "ANOTHER CONVERSION (Y,N)",C$
250  IF C$="Y" GOTO 20
260  STOP
```

Type **LIST 1,3** and press the **RETURN** Key to see the first
group of instructions in this program.

LIST 1,3 - The first 2 lines are used for the title of the
program. The variable C$ is used to store the letter you type

166

when answering the question **ANOTHER CONVERSION?** Since only one letter will be used, the variable Dimension is set to one.

LINE 20,6 - These lines clear the screen and print the menu. The extra PRINT instructions are used to skip lines and make the menu look nice on the screen.

LIST 80,3 - Here in Line 80 the program sets a variable S equal to the number you select from the menu. Line 100 is an ON-GOTO or multiple GOTO statement. In this one statement the computer can look at the value of S and GOTO 1 of 3 places depending on this value. If S is equal to 1, the program is directed to Line 110. If S equals 2, the program goes to Line 170. Any other value of S will send the program to Line 260 where it will stop.

LIST 110,6 - In this section the computer converts to Celsius. Line 150 directed the program here if you selected choice #1 from the menu. First, we request a value for Fahrenheit degrees and set F to the number typed in. Then the formula is used to set C equal to the answer. In Line 140 this answer is rounded off to the nearest degree.

The word INT means integer or whole number. Since the computer rounds off by discarding fractions, 0.5 is added to C so that the answer will be given to the nearest degree.

After computing the result and printing the answer, the computer goes to Line 220.

LIST 170,5 - With these instructions, the computer follows a similar procedure as Lines 110-160 but the formula is changed and the words are reversed. The variables F and C are used to store the number of degrees typed on the keyboard and the answer.

LIST 220,5 - After finishing either conversion, the program goes to this section. If you type a **Y** the program repeats, beginning with the menu. Press any other key and the program stops.

DICTIONARY

This dictionary contains detailed and comprehensive defini-
tions on all of the BASIC keywords introduced in Lesson 15.
If you wish to learn more about these keywords and their
variations, continue reading this dictionary section. If you
wish to go on to Lesson 16, simply type **CLOAD**, press the
RETURN Key and follow instructions. The Imagination
Machine will begin loading with Lesson 16.

ON-GOTO — changes the flow in a program, depending on a
variable's value.

ON A GOTO 100,240,560 — transfers the program to one of
the line numbers, depending on the value of A.

If A has a value of 1 the program goes to Line 100, if A has a
value of 2 the program goes to Line 240, and if A has a value
of 3 or more the program goes to Line 560.

There can be up to 9 statement numbers that the GOTO can
direct the program to. If A is greater than the number of
statement numbers, it goes to the last one.

* * * *

10 A = 5
20 ON A GOTO 100,200

Since A has a value of 5 and only 2 statement numbers are
shown, it uses statement number (200).

* * * *

The computer only looks at the integer value of A. If A = 1.1,
it would go to the first statement.

10 A = 1.1
20 ON A GOTO 100,200,300

The program is directed to Line 100.

* * * *

ON-GOSUB — goes to a subroutine, depending on a variable's value.

ON A GOSUB 500,700,900 — transfers the program to one of the subroutines, depending on the value of A.

If A has a value of 1, then GOSUB 500 results; if A has a value of 2, then GOSUB 700 results; and if A has a value of 3 or more, GOSUB 900 results.

* * * *

After the subroutine gives a return, the program returns to the next statement after the ON-GOSUB.

Example:

```
10  INPUT "A NUMBER (1-9)",N
20  ON N GOSUB 100,200,300,400,500,600,700,800,900
30  GOTO 10
100 PRINT "ONE": RETURN
200 PRINT "TWO": RETURN
300 PRINT "THREE": RETURN
400 PRINT "FOUR": RETURN
500 PRINT "FIVE": RETURN
600 PRINT "SIX": RETURN
700 PRINT "SEVEN": RETURN
800 PRINT "EIGHT": RETURN
900 PRINT "NINE": RETURN
RUN
```

Lesson 16

PLAYBACK

GOSUB, RETURN, TAB

You have probably seen the small computer games that play
a tune and then you try to repeat the tune by pressing
buttons. In Lesson 16, you will see how you can create the
same game with a computer program.

After you have learned to program the computer yourself,
you will be able to copy almost any computer game — or
design and build your own — just by writing simple
instructions.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used
Lesson 15 and the cursor is on the screen, simply type
CLOAD and press the **RETURN** Key. Follow the instructions
and the computer will begin loading Lesson 16. Adjust the
volume control next to the speaker on the Imagination
Machine keyboard and listen for introduction on Lesson 16.
The audio on the tape will inform you when to press the
RETURN Key. When the word **OK** appears on your screen,
type **RUN** and press the **RETURN** Key. You are set to begin
Lesson 16.

If you have removed the BASIC TUTOR tape from the
machine and want to begin with Lesson 16, follow the
loading instructions on Chapter 5, page 32. If you type in
CLOAD and rewind the tape all the way, you should push
the digital tape counter and set the counter back to 000.
Advance the tape via the fast forward button to 33. This
should position your BASIC TUTOR tape for Lesson 16.

RUN PLAYBACK

When you type **RUN** and press the **RETURN** Key, you will
see a number at the bottom of your screen and hear a note
in the speaker. Then the number will disappear. The object
is to play the same note by typing the same number on the
keyboard. Just press the correct number key on the

PLAYBACK
LESSON 16

RUN

INITIALIZE

LIST 1,3

BEGIN
NEW SONG

LIST 30,5

LAST
NOTE
?

ADD
NEW NOTE

LIST 80,5

PLAY SONG

LIST 130,8

KEYBOARD

INPUT
SONG

ERROR
?

YES

LIST 210,6

WRONG NOTE

LIST 300,12

PRINT
&
PLAY

NO

TWEEDLE

LIST 300,12

PRINT
&
PLAY

RIGHT NOTE
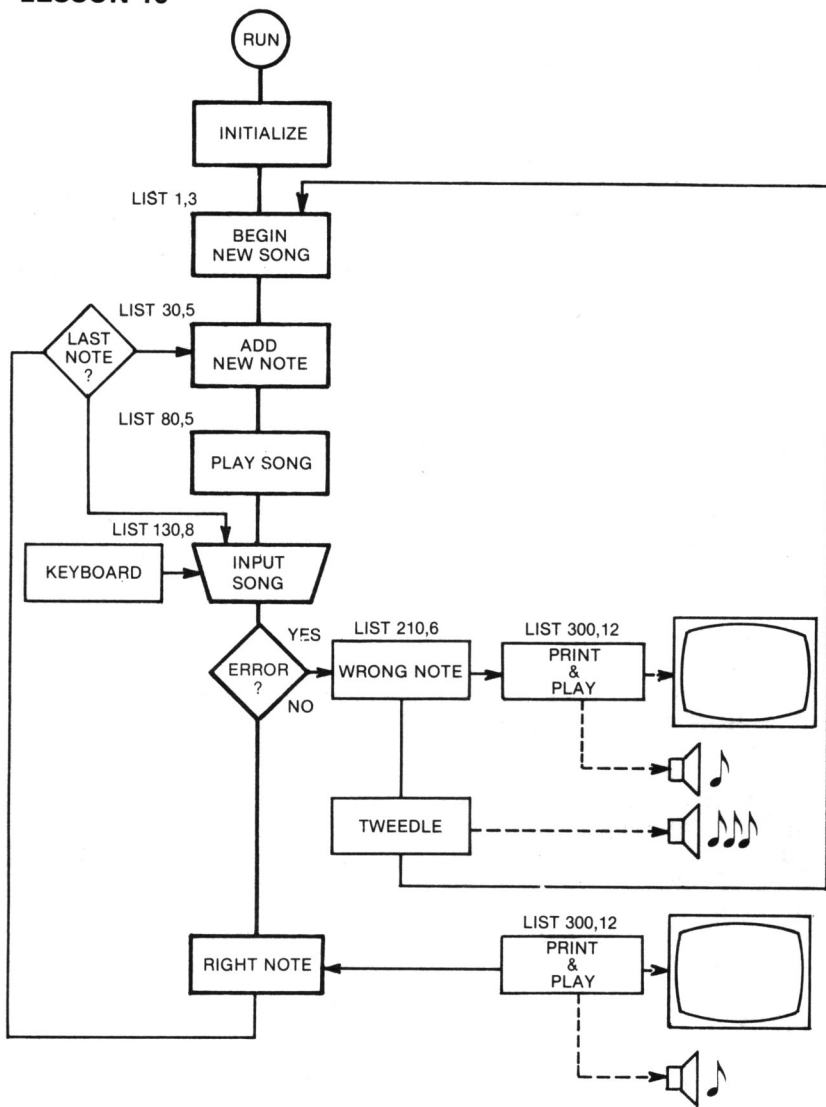
keyboard, and if you're right, the computer will play the note again and add a second note. Try to match the notes by typing their numbers on the keyboard. Very quickly you will find the length of the song increasing and becoming more difficult to remember. If (or when) you make an error, the program plays a funny tune and starts over.

If you hold a key down too long, the computer will register it as a second note. Watch the screen and when you see the note that you typed in appear on the screen, release the key. However, don't try to type in your answer too fast. The computer requires a short time to process your data and cannot respond as fast as a musical instrument.

Play with this program until you see how it works and what it does. Then look at the flow chart diagram and see if you can follow what's happening inside the computer as you run the program.

HOW PLAYBACK WORKS
(See flow chart on preceding page)

To stop the program press the **BREAK** Key. When you first run the program, it initializes the variables used and begins by picking a random number between one and eight. It uses these numbers to represent the musical scale with keys 1 through 8 on your keybaord matching an octave. Each number (or note) that the computer picks is added to the previous tune.

Next, the computer plays the tune, one note at a time. The number of each note is printed on the screen while the sound is played in the speaker. Then the screen is cleared and it is your turn to try to enter the same tune.

Notes you press on the keyboard are entered into the program as you try to copy the tune. They also appear on the screen as a number. If you make an error, the note is played and printed on the screen, then the "tweedle tune" is played and the program starts over again.

Get the note right and the right note is played and printed. If you have played the complete correct sequence of notes in the tune, the program loops back to the point where a

new note is added by the computer. If the notes are entered in the wrong sequence, the program loops back to get another note from the keyboard and starts a new tune.

There is no end to this program and it could go on forever. To stop it at any time, just press the **BREAK** Key.

CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 16?

You can now continue with Lesson 16 to see how Playback works and learn how to write certain programming statements, or you can go directly to Lesson 17. If you want to go directly to Lesson 17 (Distance/Time/Rate) and your BASIC TUTOR tape is still in the machine, type **CLOAD**, follow the instructions on the screen and the computer will automatically begin loading. If you want to go directly to Lesson 17 (Distance/Time/Rate) and your BASIC TUTOR tape is not in the machine, see page 32, Chapter 5 for instructions.

```
1   REM    ...PLAYBACK...
10  DIM K$(1): DIM S(50)
20  N=1:T=0
30  REM    ...ADD NEW NOTE...
40  FOR L=1 TO 16: PRINT : NEXT L
50  T=T+1
60  S(N)= INT ( RND (0)*8+1)
70  IF S(N)=S(N-1) GOTO 60
80  REM    ...PLAY SONG...
90  FOR N=1 TO T
100   GOSUB 300
110   NEXT N
120   FOR L=1 TO 16: PRINT : NEXT L
130   REM    ...COPY SONG...
140   FOR N=1 TO T
150   K$= KEY$ (0)
160   IF K$="" GOTO 150
170   IF  ASC (K$)-48<>S(N) GOTO 210
180   REM    ...RIGHT NOTE...
190   GOSUB 300: NEXT N
200   GOTO 30
210   REM    ...WRONG NOTE...
```

```
220  S(N)= ASC (K$)-48
230   GOSUB 300
240   MUSIC "*1-*1*1-*1*1-*1*1-*1"
250  N=T: NEXT N
260   GOTO 20
300   REM    ...PRINT & PLAY...
310   REM    ...SUBROUTINE.....
320   PRINT  TAB (S(N)*2);S(N)
330   ON S(N) GOTO 340,350,360,370,380,390,400,410
340   MUSIC "/100": RETURN
350   MUSIC "/200": RETURN
360   MUSIC "/300": RETURN
370   MUSIC "/400": RETURN
380   MUSIC "/500": RETURN
390   MUSIC "/600": RETURN
400   MUSIC "/700": RETURN
410   MUSIC "100": RETURN
```

LIST 1,3 - The first line is a remark and it is used to add a title to the program. The variable K$ is dimensioned to hold one letter. This will be used to store the key you type on the keyboard. The letter S is dimensioned to hold a string of 50 numbers. This is the tune, with S(1) being the first note, S(2) being the second note, and so forth up to a maximum of 50 notes, S(50).

The letter N is used to keep track of the position in the Array S that contains the current note being played by the computer or player. T keeps track of the total number of notes in the song. In the beginning, the position in Array S of the next note to be added or played is 1. The total number of notes played is 0.

LIST 30,5 - Here's where a new note is added to the song. First, the screen is cleared and the total number of notes in the tune is increased by one. Then a random number between 1 and 8 is stored in the appropriate position of S. If this is the first note, then N equals 1 and the note is stored in S(1).

In Line 70 a check is made to see if the new note S(N) is the same as the previous note in the song (N-1). If so, a new selection is made. This keeps the tune from repeating the same note twice in a row.

LIST 80,5 - This section plays the tune from beginning to end. We will start with the first note which is in S(1) and continue through the total number of notes with the last being S(T). Lines 90 through 110 loop for each note in the song.

GOSUB 300 means go to subroutine 300. A subroutine is a group of instructions that we repeatedly want to execute in a program. We want this set of instructions to appear in several places in the program. Instead of having to keep writing these intructions into the program, BASIC allows us to write them once and be able to go to them from anywhere in the program. After they are completed, BASIC allows the program flow to return to the next program statement after the GOSUB routine was called for. This subroutine which starts at Line 300 is used several times in the program to print each number and play each note in the tune.

After the tune is played, the screen is cleared at Line 120.

LIST 130,8 - In this section you will input numbers from the keyboard until you copy the tune correctly or make a mistake. Line 140 opens a FOR-NEXT loop that will look for the player entries for each of the notes in our tune. Each number key you type in is stored in K$. Since the computer can look at the keyboard very quickly, Line 160 checks to see that a key is actually pressed during Line 150. If no key is pressed, K$ = nothing and sends the program back to Line 150 where it waits for a key to be pressed. When a key is pressed, Line 170 will compare it to the expected note which is stored in S(N). If you type a number that isn't equal to the number of the next note, you've made a mistake and the program goes to Line 210. If your number is correct, then the subroutine at Line 300 is called to play it and print it on the screen. This loop is repeated until N = T and the tune is over. If you're successful, the program then goes to Line 30 to add a new note to the tune.

LIST 210,6 - If you goof, you'll wind up here. The note you played is substituted for the right note in the song. Then the subroutine at Line 300 is called to play it and print it on the screen.

Following the note you play on the keyboard, the MUSIC line plays a special "tweedle tune" all its own to let you know it's all over. Then the number of the next note (N) is set equal to the last note (T) and this song is finished. This is done so the NEXT N statement in Line 250 will end the loop we were in and can go on to Line 260. It is important when exiting a FOR-NEXT loop to exit with it completed. The loop variable must be equal to the end limit. Setting N = T completes the loop.

The final step sends the program back to the very beginning to start a new song.

LIST 300,12 - Here's the subroutine you've been reading about. From several places in the program, the statement "GOSUB 300" has been used. That means go to Line 300, follow these steps, and then go back to the place in the main program that you came from.

All the printing is done in Line 320. After moving (TAB) 2 spaces times the value of the note, the number of the note is printed. For example, if the note is 1, the number 1 is printed one space in from the left margin. If the note is 2, the computer spaces 4 times and prints the number 2. For 3, the computer spaces 6 times and prints 3. Finally, for the number 8, the computer spaces 16 times and prints the number 8.

Line 330 is a directory. Depending on the value of S(N), the computer will go to any one of the 8 places. Since S(N) has the value for the number of the note, our directory can send the computer to play any one of these 8 MUSIC instructions. As you might suspect, they instruct the computer to play the notes of the musical scale. Press the **RETURN** Key at the end of each statement means go back to the line following the word GOSUB that sent you to Line 300.

DICTIONARY

This dictionary contains detailed and comprehensive definitions on all of the BASIC keywords introduced in Lesson 16. If you wish to learn more about these keywords and their variations, continue reading this dictionary section. If you wish to go on to Lesson 17, simply type **CLOAD**, press the **RETURN** Key and follow instructions. The Imagination Machine will begin loading with Lesson 17.

GOSUB — allows a section of program statements to be entered only once and used many times throughout the program. Often in a program, a sequence of instructions have to be done many times. We could rewrite these statements each time or we can use a subroutine.

GOSUB 300 — transfers your program to Line 300 and remembers where program was up to before the transfer. Line 300 will begin a subroutine or series of instructions that you may use several times in your program.

RETURN — transfers the program back to the statement following the GOSUB instruction.

* * * *

A subroutine can have any group of statements in it including other GOSUBs.

Example:

```
10 GOSUB 100
20 IF A<>4 THEN PRINT A:  GOTO 10
30 END
100 FOR J = 1 TO 32:  PRINT:  NEXT J:
110 GOSUB 200
120 RETURN
200 INPUT "ENTER VALUE", A:  IF A = 10 THEN 200
210 RETURN
```

* * * *

TAB — moves the printing to the right. Similar to typewriter tab key function.

PRINT TAB(5);"HERE"

Prints the word **HERE** 5 spaces from the left edge of the screen.

If the cursor position is already passed the number of tab spaces requested, the tab is ignored. For example:

10 PRINT "THIS"; TAB(2); "IS A TEST"

The tab function is ignored since the cursor has already passed 2 spaces from the left side of the screen.

Lesson 17

DISTANCE / TIME / RATE

FORMULAS

This program shows you how your computer can help you solve a wide range of problems by using a single formula. In this example, time and speed formula along with the formula for gas mileage are used to create an interactive system.

The program begins with a menu, listing several calculations. After one is selected, the program requests specific information, applies the formula to find the answer, and prints the result.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 16 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 17. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 17. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 17.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 17, follow the loading instructions on Chapter 5, page 32. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 44. This should position your BASIC TUTOR tape for Lesson 17.

RUN DISTANCE / TIME / RATE

When you run this program, a menu is printed and the computer waits for your selection. Type any number from one to four and press the **RETURN** Key.

After clearing the screen, the computer is directed to the part of the program containing the correct formula. The data required to solve the formula is then requested. Answer

the questions on the screen by typing a number and pressing the **RETURN** Key. Notice that the questions are stated in a way that clearly shows what kind of answer is required.

After you've entered all the data, the program displays a statement showing your input and the correct result.

Select Y to request another calculation and then press the **RETURN** Key. This automatically returns you to the menu. Now pick a different selection and let the program guide you to the solution by requesting the specific data needed. Continue until you've tried all four selections. You can enter any numbers you like — even negative numbers and decimals. If your input or the answer is too large for the computer, the program will stop and the computer will print an error message. If this happens, just run the program again.

HOW DISTANCE / TIME / RATE WORKS
(See flow chart on following page)

The important features of this program are the use of a menu to allow the user to select the answer required, the input statements that clearly request the specific information, and the print out of the results. Simple formulas are used to illustrate these objectives in a working program. You could use the general format and the ideas in this lesson to create a similar program that would guide the user and create answers to any questions that can be solved by formulas.

By simply changing the formulas and the input statements, this program could be used to solve physics problems, math problems, conversions, and similar tasks.

The flowchart diagram shows a basic procedure that you can adapt to almost any application. First, a menu of choices is printed on the screen with a selection line at the bottom. Notice that the range of possible answers is printed in parenthesis — (1-4) — to guide the user. There's no rule that makes this necessary; it's just good programming practice to be clear when you request an input.

DISTANCE / TIME / RATE
LESSON 17

RUN

LIST 1,5 — INITIALIZE

LIST 100,10 — PRINT MENU AND INPUT SELECTION

```
1. DISTANCE
2. TIME
3. RATE
4. MILEAGE
SELECT (1-4) ?■
```

LIST 200,3 — CLEAR AND SELECT

LIST 300,8 — 1 — YES — TIME RATE

LIST 400,7 — 2 — YES — DISTANCE RATE

LIST 500,8 — 3 — YES — DISTANCE TIME

LIST 600,8 — 4 — YES — MILES GAL.

DISPLAY INPUT AND ANSWER

```
TIME 5
RATE 55
TRAVEL
275 MILES
```

LIST 700,4 — INPUT CHOICE

```
ANOTHER
CALCULATION
(Y-N)■ ?
```

Y — START OVER

STOP
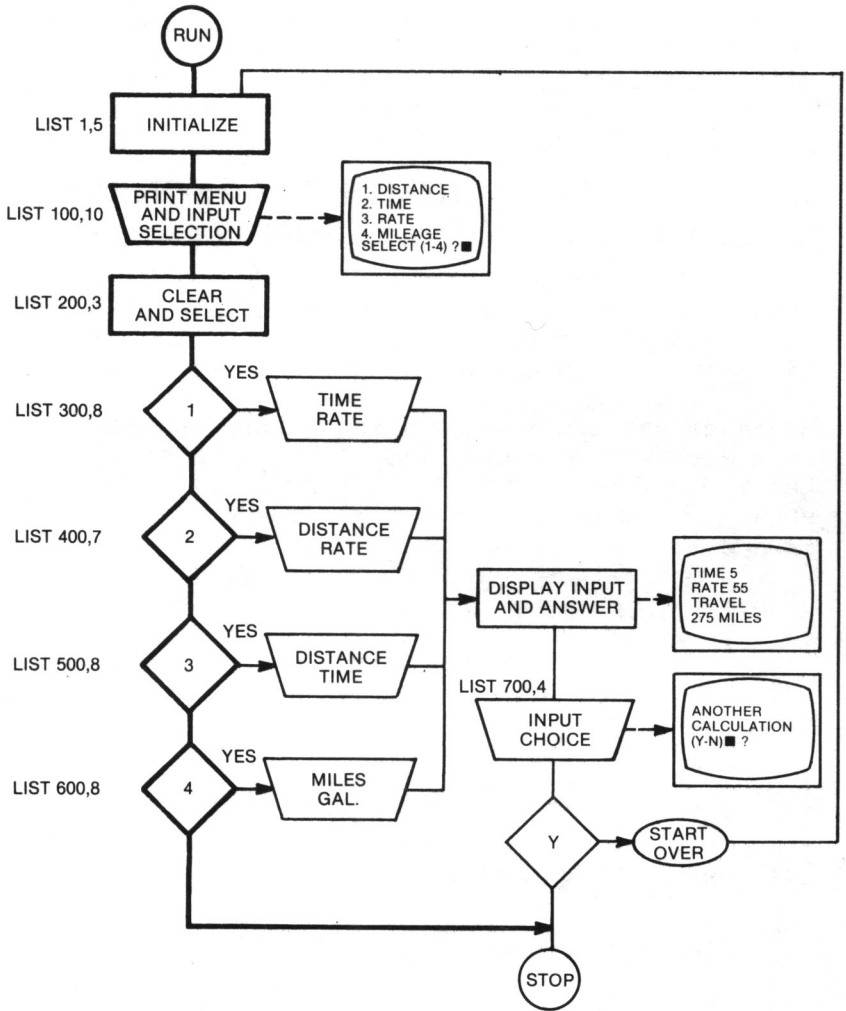
Depending on the number that's typed, the program branches to the appropriate section. In each of the four sections, the program prints specific questions and inputs the answers.

The printout displays your inputs as well as the answers in a clear statement. This acts as a double check, since any wrong input would be spotted immediately. With this format, the user can easily verify that the answer on the screen matches the input they typed.

CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 17?

You can now continue with Lesson 17 to see how Distance / Time / Rate works and learn how to write certain programming statements, or you can go directly to Lesson 18. If you want to go directly to Lesson 18 (Pilot) and your BASIC TUTOR tape is still in the machine, type **CLOAD**, follow the instructions on the screen and the computer will automatically begin loading. If you want to go directly to Lesson 18 (Pilot) and your BASIC TUTOR tape is not in the machine, see page 32, Chapter 5 for instructions.

```
1   REM     ...DISTANCE...
2   REM     ...TIME.......
3   REM     ...RATE.......
10    DIM K$(1)
20    FOR N=1 TO 32: PRINT : NEXT N
100  . REM     ...MENU...
110   PRINT
120   PRINT "     ...CAR CALCULATOR..."
130   PRINT
140   PRINT "    1. DISTANCE TRAVELED"
150   PRINT "    2. TIME REQUIRED"
160   PRINT "    3. AVERAGE SPEED"
170   PRINT "    4. GAS MILEAGE"
180   PRINT
190   INPUT "      SELECT (1-4) ",S
200   REM     ...SELECT...
210   FOR N=1 TO 32: PRINT : NEXT N
220   ON S GOTO 300,400,500,600,730
300   REM     ...DISTANCE...
310   INPUT "TIME IN HOURS ",T
320   INPUT "SPEED IN M.P.H. ",R
330   PRINT "  AT ";R;" M.P.H."
```

```
340   PRINT "   FOR ";T;" HOURS"
350   PRINT "   YOU WILL TRAVEL"
360   PRINT "   ";T*R;" MILES"
370   GOTO 700
400   REM     ...TIME...
410   INPUT "DISTANCE IN MILES ",D
420   INPUT "SPEED IN M.P.H. ",R
430   PRINT "   TRAVELING ";D;" MILES"
440   PRINT "   AT ";R;" M.P.H."
450   PRINT "   WILL TAKE ";D/R;" HOURS"
460   GOTO 700
500   REM     ...SPEED...
510   INPUT "DISTANCE IN MILES ",D
520   INPUT "TIME IN HOURS ",T
530   PRINT "   TRAVELING ";D;" MILES"
540   PRINT "   IN ";T;" HOURS"
550   PRINT "   YOU WILL AVERAGE"
560   PRINT "   ";D/T;" MILES PER HOUR"
570   GOTO 700
600   REM     ...MILEAGE...
610   INPUT "DISTANCE IN MILES ",D
620   INPUT "FUEL IN GALLONS ",G
630   PRINT "   TRAVELING ";D;" MILES"
640   PRINT "   ON ";G;" GALLONS OF FUEL"
650   PRINT "   WILL AVERAGE"
660   PRINT "   ";D/G;" MILES PER GALLON"
670   GOTO 700
700   REM     ...MORE?...
710   PRINT : INPUT "ANOTHER CALCULATION (Y,N) ",K$
720   IF K$="Y" GOTO200
730   STOP
```

Stop the program by selecting **N** after a calculation or by
pressing the **BREAK** Key.

LIST 1,5 - The lines beginning with REM are remarks, and are
used to form titles and comment sections. The variable K$ is
dimensioned to one character. This will be used later to
store the character typed on the keyboard. After clearing the
screen, the computer goes to the next section.

LIST 100,10 - Print statements display the menu of choices.
Spaces adjust the position of the lines on the screen, just
for looks. The program waits after printing the input
question until a key is typed and the **RETURN** Key is
pressed. With S equal to the number selected, the program
goes to the next section.

LINE 200,3 - After clearing the screen, the computer directs the program to one of five places, depending on the value of S. This branching technique of using an ON-GOTO allows any section of the program to be "next", based upon the value of S. Notice the fifth possible branch: if a number greater than 4 was pressed, the program goes to Line 730 where it stops.

LIST 300,8 - If $S = 1$, the program branches to this section. The formula for distance traveled is:

$$D = T*R$$

where D is distance in miles, T is time in hours, and R is rate of speed in miles per hour. After requesting T and R, the program prints the answer and goes to line 700.

LIST 400,7 - If the time required is sought, then $S = 2$ and the program branches to this section. The formula for time is:

$$T = \frac{D}{R}$$

where T is time in hours, D is distance in miles, and R is rate of speed in miles per hour. The program requests D and R, then prints the results and goes to Line 700.

LIST 500,8 - If $S = 3$ the program branches here. The formula for speed is:

$$R = \frac{D}{T}$$

where R is rate in miles per hour, D is distance in miles, and T is time in hours. Again, the formula is applied automatically as the answer is printed on the screen. After printing the results, the program goes to line 700.

LIST 600,8 - If gas mileage is selected, the program branches to line 600. The equation or formula used is:

$$\text{Miles per gallon} = \frac{M}{G}$$

186

where M is miles and G is gallons. Again, the formula is followed and the results printed on the screen. After completing this task, the computer goes to line 700.

LIST 700,4 - After completing whichever task is selected, the program prints **"ANOTHER CALCULATION (Y,N)?"** and waits for an input. If a Y is typed, the program goes to Line 20 and reprints the menu. Any other input will cause the program to stop.

DICTIONARY

This dictionary contains detailed and comprehensive definitions on all of the BASIC keywords used in Lesson 17. If you wish to learn more about these keywords and their variations, continue reading this dictionary section. If you wish to go on to Lesson 18, simply type **CLOAD**, press the **RETURN** Key and follow instructions. The Imagination Machine will begin loading with Lesson 18.

FORMULAS — Are solved directly.
You may use any formula in program. Assign variables to each quantity and the computer will automatically solve the equation. The symbols used in formulas may include:

+ addition
− subtraction
* multiplication
/ division
↑ exponentiation

If several symbols are used, multiplication and division will be done first, followed by addition and subtraction.

RULES OF PRECEDENCE

When the Imagination Machine has to evaluate a mathematical expression, it has set rules of which operations are done first. Highest precedence is exponentiation. Next is multiplication or division. Last is addition or subtraction. Where two operations of the same

Lesson 18

PILOT

HLIN, VLIN

It's easy to draw complex pictures on the screen with your computer and you can use up to 8 colors at the same time. Just watch!

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 17 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 18. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 18. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 18.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 18, follow the loading instructions on Chapter 5, page 0. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 56. This should position your BASIC TUTOR tape for Lesson 18.

RUN PILOT

This is the easiest lesson of all to run. Just type **RUN** and press the **RETURN** Key. There's no sound with this program, but you may want to adjust the color of your TV to get the best picture. If you don't have a color set, you will need to have a lot of imagination. The sky is blue, the pilot is white with green ears, the airplane is red and yellow, and the propeller is orange.

HOW PILOT WORKS

When you type **RUN** and press the **RETURN** Key the program begins by painting the sky blue. Then each section of the picture is drawn on the screen. After the picture is complete, the program goes into a loop that repeats until you stop the program by pressing the **BREAK** Key.

The program loop is used to make the propeller turn by drawing the propeller, erasing the propeller with the background color, and then drawing the propeller again.

In creating pictures or cartoons, it is important to always sketch out what you want on paper as we have done in this lesson.
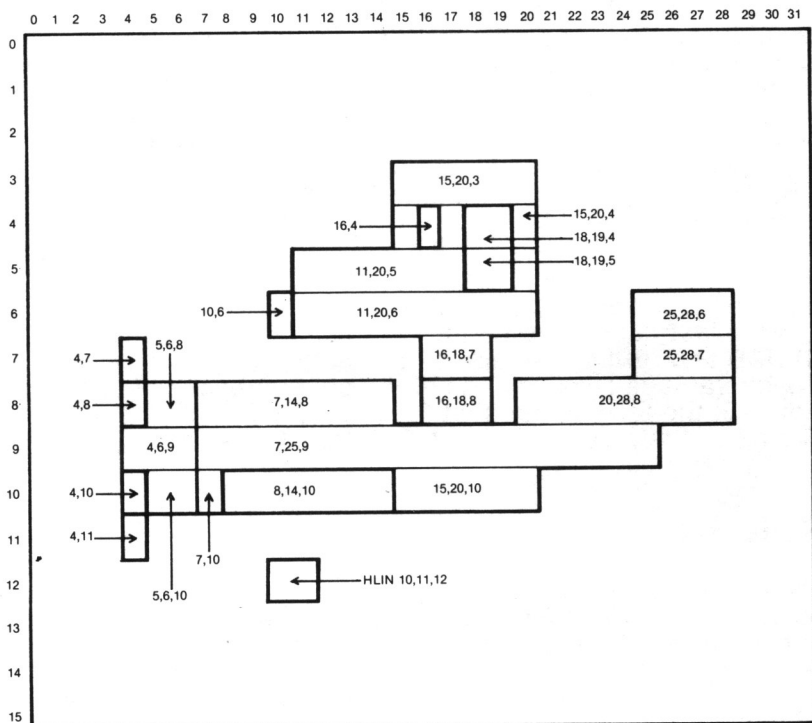
CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 18?

You can now continue with Lesson 18 to see how Pilot works and learn how to write certain programming statements, or you can go directly to Lesson 19. If you want to go directly to Lesson 19 (Math Teacher) and your BASIC TUTOR tape is still in the machine, type **CLOAD**, follow the instructions on the screen and the computer will automatically begin loading. If you want to go directly to Lesson 19 (Math Teacher) and your BASIC TUTOR tape is not in the machine, see page 32, Chapter 5 for instructions.

```
1   REM    ...PILOT...
10  SHAPE =15: COLOR =2
20  FOR Y=0 TO 31
40  VLIN 0,15,Y
50  NEXT Y
60  REM    ...AIRPLANE...
70  COLOR =1
80  HLIN25,28,6: HLIN 25,28,7: HLIN 7,14,8: HLIN 20,28,8: HLIN 7,25,9:
    PLOT 7,10: HLIN 15,20,10
90  REM    ...WING...
100  COLOR =3
110  HLIN 8,14,10
120  REM    ...WHEEL...
130  COLOR =6
140  HLIN 10,11,12
150  REM    ...ENGINE
160  COLOR =6
170  HLIN 5,6,8: HLIN 4,6,9: HLIN 5,6,10
180  REM    ...HEAD...
190  COLOR =4
200  HLIN 15,20,3: HLIN 15,20,4: HLIN 11,20,5: HLIN 11,20,6: HLIN 16,18,7
210  COLOR =0
220  PLOT 16,4: HLIN 18,19,4: HLIN 18,19,5: PLOT 10,6
230  COLOR =3
240  HLIN 16,18,8
250  REM    ...PROP...
260  COLOR =7
270  PLOT 4,8: PLOT 4,10: PLOT 4,7: PLOT 4,11
280  COLOR =2
290  PLOT 4,7: PLOT 4,11: PLOT 4,8: PLOT 4,10
300  GOTO 250
```

LIST 1,5 - SHAPE 15, a solid square is used for all drawing in this program. You can also use other shapes and create more complex patterns.

COLOR 2 (blue) — the sky — is painted by using one loop. The word VLIN draws a vertical line. There are 3 numbers or numeric variables after VLIN. The first 2 give the height of

RUN

LIST 1,5 — DRAW SKY → 2 = BLUE

LIST 60,3 — DRAW AIRPLANE → 1 = YELLOW

LIST 90,3 — DRAW WING → 3 = RED

LIST 120,3 — DRAW WHEEL → 6 = PURPLE

LIST 150,3 — DRAW ENGINE → 6 = PURPLE

LIST 180,7 — DRAW HEAD → 4 = WHITE
0 = GREEN
3 = RED

LIST 250,6 — DRAW PROP → 7 = ORANGE

ERASE PROP → 2 = BLUE
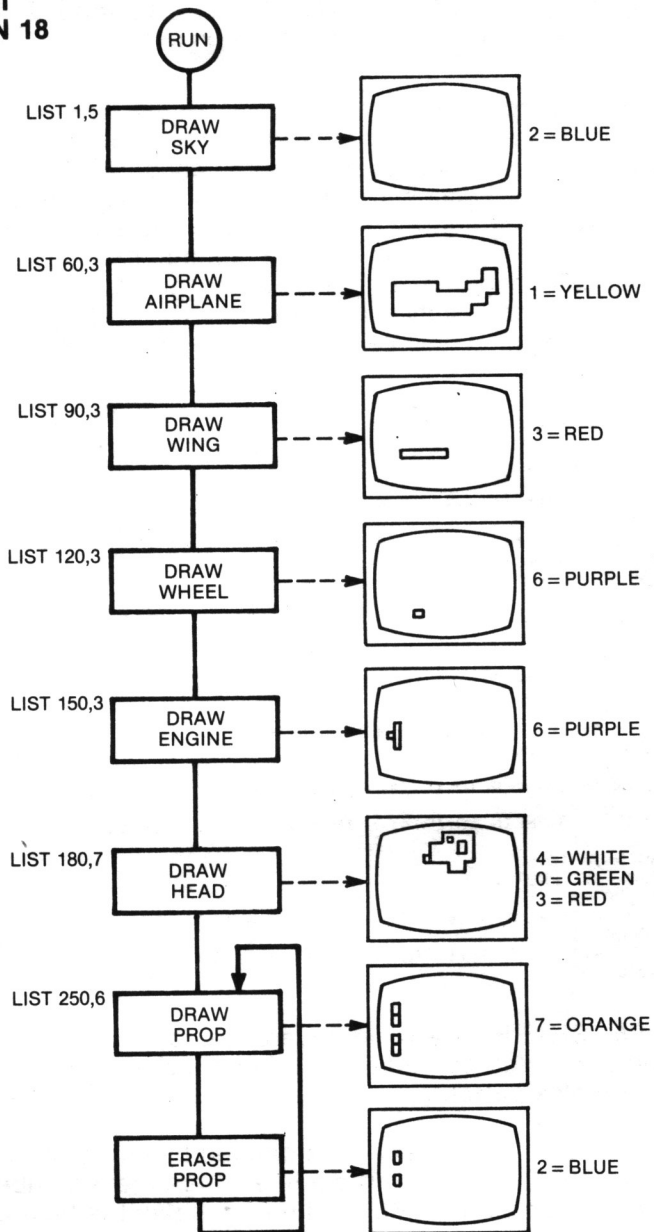
the line in terms of horizontal rows from row 0 at the top of the screen to row 15 at the bottom. The third number, in this case, is the value of Y: which moves this line from the left edge of the screen (or 0) to 31 which is the right edge.

LIST 60,3 - The body of the airplane is drawn in 7 sections — COLOR = 1 (yellow). The word HLIN draws a horizontal line. HLIN 25,28,6 forms the top line of the airplane's tail by drawing a horizontal line from column 25 to column 28 on row number 6. Look at the diagram and see where this line is located.

The next instruction draws the next line to complete the tail of the airplane. This line is drawn from column 25 to column 28 again, but on row 7. This line is located directly below the first.

Now the top of the fuselage is drawn with two lines. The first line is HLIN 7,14,8 and the second is HLIN 20,28,8. Together they draw the part of the airplane that is on row number 8.

Now one long line is used to draw the center of the airplane. HLIN 7,25,9 draws from column 7 to column 25 on row 9.

The small spot in front of the wing is only one square side. Instead of drawing a line, we use the PLOT instruction to color this square. The rest of row number 10 is filled in with HLIN 15,20,10.

LIST 90,3 - Now that you know how to draw horizontal lines, making a wing is easy. First COLOR 3 (red) is selected and then HLIN 8,14,10 draws from column 8 to column 14 on row number 10.

LIST 120,3 - The COLOR 6 (purple) wheel is drawn with one HLIN from 10 to 11 on row 12.

LIST 150,3 - Three lines are used for the engine. HLIN 5,6,8 draws the top line, HLIN 4,6,8 draws the center, and HLIN 5,6,10 draws the bottom.

LIST 180,7 - Three separate colors are used for drawing the head. First we use COLOR 4 (white) to fill in the complete head with five lines. Then COLOR 0 (green) is used for the nose, eye, and ear. Notice that we can draw over a color with another color, just like paints. The COLOR 3 (red) collar is added last on row number 8.

LIST 250,6 - Now we come to the moving part of the drawing. These six lines repeat over and over because the last line (Line 300) tells the computer to go back to the first line in this section (Line 250). Once the program reaches this section, there's no way for it to stop running unless you press the **BREAK** Key.

COLOR 7 (orange) is used first to draw the propeller. We start by drawing the part that's next to the engine, then draw the tips of the propeller. COLOR 2 (blue) is now drawn on top of the propeller to erase it. We start with the tips, and then erase the part that's next to the engine. This sequence gives the effect of the propeller turning, not just blinking on and off. The process repeats with the orange propeller alternating with the blue sky.

You can add animation like this to any drawing by writing a similar series of instructions that repeat.

DICTIONARY

This dictionary contains detailed and comprehensive definitions on all of the BASIC keywords introduced in Lesson 18. If you wish to learn more about these keywords and their variations, continue reading this dictionary section. If you wish to go on to Lesson 19, simply type **CLOAD**, press the **RETURN** Key and follow instructions. The Imagination Machine will begin loading with Lesson 19.

For more information see Lesson 7 on shape, color, plot, and screen grid.

HLIN, VLIN — allows drawing a horizontal or vertical line. SHAPE and COLOR of lines are previously set up by system variables SHAPE and COLOR.

HLIN Y1, Y2, X — draws a horizontal line from Column Y1 to Column Y2 at Row X.

VLIN X1, X2, Y — draws a vertical line from Row X1 to Row X2 at Column Y.

X, X1, X2, Y, Y1, Y2 are numerical values. They can be numbers, numerical expressions, calculations, or numerical variables.

<p style="text-align:center">* * * *</p>

Examples:

```
10  CALL 17046
20  SHAPE = 15:  COLOR = 4
30  HLIN 6,10,4
40  VLIN 2,8,20
```

<p style="text-align:center">* * * *</p>

```
10  CALL 17046
20  SHAPE = 15
30  FOR J = 0 TO 15
40  COLOR = J
50  HLIN 0,31,J
60  NEXT J
```

<p style="text-align:center">* * * *</p>

Various shapes can be used in HLIN or VLIN

```
10  CALL 17046
20  COLOR = 4:  S = 0
30  FOR J = 0 TO 31 STEP 2
40  SHAPE = S
50  VLIN 0,15,J
60  S = S + 1
70  NEXT J
```

* * * *

When lines of different colors cross each other, the intersection takes on the new color

```
10  CALL 17046:  SHAPE = 15
20  COLOR = 4:  HLIN 0,31,7
30  COLOR = 7:  VLIN 0,15,15
```

* * * *

When lines of different shapes cross, the intersection point takes a shape which is a sum of the two individual shapes.

```
10  CALL 17046
20  COLOR = 3
30  SHAPE = 1:  HLIN 0,31,7
40  SHAPE = 8:  VLIN 0,15,15
```

* * * *

If you do a line command with SHAPE = 0, then only the color will change.

```
10  CALL 17046
20  SHAPE = 7:  COLOR = 1
30  HLIN 0,31,7:  SHAPE = 0
40  FOR C = 0 TO 65
50  COLOR = C
60  HLIN 0,31,7
70  NEXT C
```

Lesson 19

MATH TEACHER

Here's a program that makes flash cards obsolete. Math Teacher and your computer are an effective educational tool for learning the arithmetic facts. The program generates an endless supply of addition, subtraction, multiplication, and division problems in all combinations. Two random numbers between 1 and 10 are picked by the computer and a problem is created and displayed.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 18 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 19. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 19. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 19.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 19, follow the loading instructions on Chapter 5, page 32. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 67. This should position your BASIC TUTOR tape for Lesson 19.

RUN MATH TEACHER

After you type **RUN** and press the **RETURN** Key, the program requests the number of problems you wish to answer. Try a low number for a start, and press the **RETURN** Key.

The computer will give you the problems one at a time. You have to supply the answers. Press the **RETURN** Key after each answer and listen to the response in the TV speaker. After answering the number of problems you requested, the computer gives you a report card. Make an "A" (all answers right) and hear a special tune.

Run this program several times and see how it helps someone to learn arithmetic.

HOW MATH TEACHER WORKS

The first step in using this program is deciding how many problems to answer. This number is typed on the keyboard and entered into the program.

Two random numbers are then generated by the computer. They can each be any number from 1 to 10. These numbers are called A and B in the program.

Another random number is generated between 1 and 4. This number is used to select what type of problem will be presented, like this:

$$1 = \text{addition}$$
$$2 = \text{subtraction}$$
$$3 = \text{multiplication}$$
$$4 = \text{division}$$

After picking a problem with the numbers A and B to be in it, the computer solves the problem and sets the answer equal to X. The problem (minus the answer) is then displayed on the screen.

With X equal to the answer, the computer now waits for a number to be typed on the keyboard. If the answer is correct, a note is played in the speaker. Get the answer wrong and a different note is played.

If this problem is the last in the series you requested, the program prints your report card. If not, it goes back and creates the next problem.

**MATH TEACHER
LESSON 19**

```
                              ( RUN )
                                 |
LIST 1,5          [  INITIALIZE  ]◄──────────────────┐
                                 |                    |
                         ┌───────────────┐  ┌──────────────┐
KEYBOARD ──────────►─────┤    INPUT      ├─ ┤ HOW MANY     │
                         │  NUMBER OF    │  │ PROBLEMS?■    │
                         │   PROBLEMS    │  └──────────────┘
                         └───────────────┘
                                 |
                         [    CLEAR      ]
                         [    SCREEN     ]
                                 |
LIST 50,5                [    SELECT     ]
                         [  VALUES FOR   ]
                         [   PROBLEM     ]
                                 |
LIST 100,11              [   CREATE      ]─ ─ ─►  ┌──────────┐
                         [   PROBLEM     ]        │ 2+3 = ?■ │
                                 |                └──────────┘
LIST 210,5               [  INPUT AND    ]
                         [    SCORE      ]
                                 |
                   Y           /MORE\              ♪
              ┌───────────────◄ ?   ├─ ─ ─►◄|
              |                \   / N
              |                  |
              |                  |                ┌────────────┐
LIST 260,7    |          [  REPORT   ]─ ─ ─►      │ YOUR SCORE │
              |          [   CARD    ]            │ RIGHT: 10  │
              |                  |                │ WRONG: 3   │
              |                  |                │ TOTAL: 13  │
              |                  |                └────────────┘
              |               / PER- \   YES
              |              <  FECT   ├─ ─ ─►◄|  ♪♪♪
              |               \SCORE?/ NO
              |                  |
              |                  |                ┌────────────┐
LIST 330,4    |          [  INPUT T$ ]─ ─ ─►      │ TRY AGAIN  │
              |                  |                │ (Y-N)?■    │
              |                  |                └────────────┘
              |                  |
              |                / Y \       ( START  )────────┘
              └───────────────<    ├─────►( OVER   )
                               \   /
                                 |
                              ( STOP )
```

199
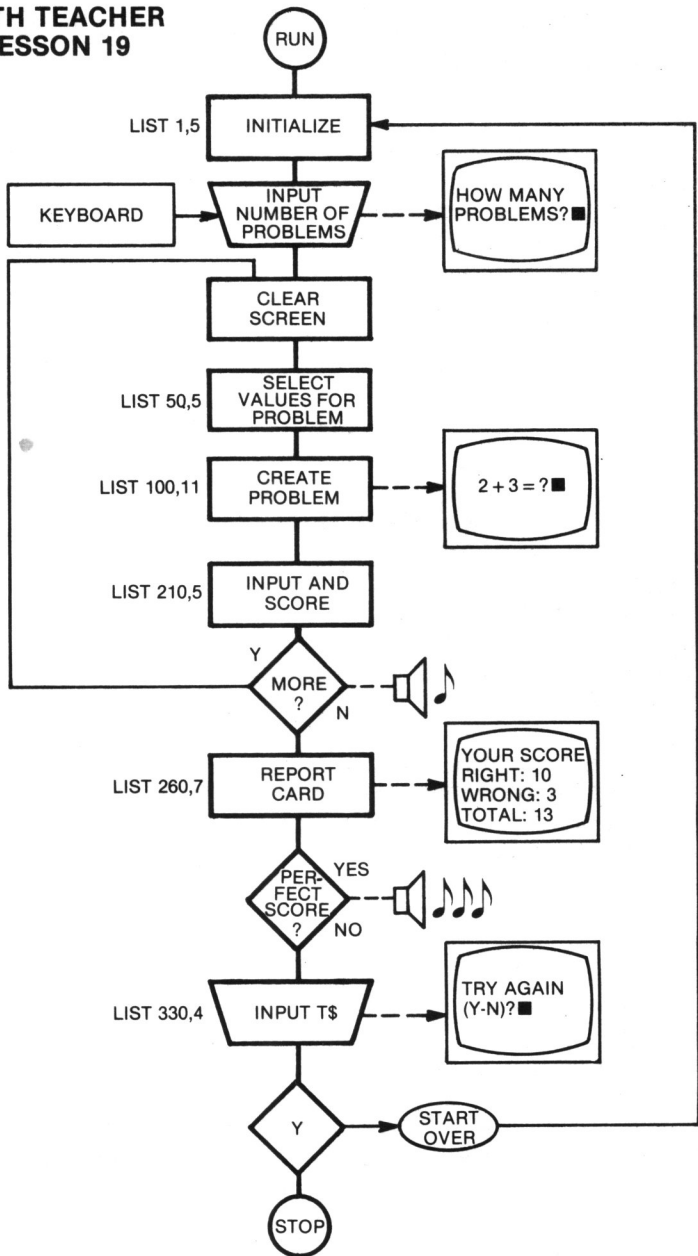
CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 19?

You can now continue with Lesson 19 to see how Math Teacher works and learn how to write certain programming statements, or you can go directly to Lesson 20. If you want to go directly to Lesson 20 (Code Breaker) and your BASIC TUTOR tape is still in the machine, type **CLOAD**, follow the instructions on the screen and the computer will automatically begin loading. If you want to go directly to Lesson 20 (Code Breaker) and your BASIC TUTOR tape is not in the machine, see page 32, Chapter 5 for instructions.

```
1   REM    ...MATH TEACHER...
10   DIM T$(1)
20 R=0:W=0
30   INPUT "   HOW MANY PROBLEMS",T
40   FOR L=1 TO 20: PRINT : NEXT L
50   REM    ...SELECT TERMS...
60   FOR P=1 TO T
70 A= INT ( RND (0)*10+1)
80 B= INT ( RND (0)*10+1)
90   IF A<B GOTO 70
100   REM    ...CREATE PROBLEMS...
110 C= INT ( RND (0)*4+1)
120   ON C GOTO 130,150,170,190
130   PRINT A;" + ";B;" = ";
140 X=A+B: GOTO 210
150   PRINT A;" - ";B;" = ";
160 X=A-B: GOTO 210
170·  PRINT A;" * ";B;" = ";
180 X=A*B: GOTO 210
190   PRINT A*B;" / ";B;" = ";
200 X=A
210   REM    ...SCORE ANSWER...
220   INPUT G
230   IF G=X THEN R=R+1: MUSIC "*1*2"
240   IF G<>X THEN W=W+1: MUSIC "/1-/1"
250   NEXT P
260   REM    ...REPORT CARD...
270   FOR N=1 TO 16: PRINT : NEXT N
```

200

```
280 PRINT : PRINT "   ...YOUR SCORE..."
290 PRINT : PRINT "     RIGHT:";R
300 PRINT "     WRONG:";W
310 PRINT "     TOTAL:";T
320 IF W=0 MUSIC "103050*10    50*100000"
330 REM    ...MORE?...
340 PRINT
350 INPUT "   TRY AGAIN (Y,N)",T$
360 IF T$="Y" GOTO 20
```

Stop your program by selecting **N** after answering the problems or by pressing the **BREAK** Key.

LIST 1,5 - The program begins by dimensioning T$ to hold one letter. This will be used later to store the key you type when answering Y or N. The letters R and W are used to store right and wrong answers. They're both set equal to 0 now. The number you type will be stored in T, the total number of problems to be presented. Notice that this T is a different variable name than T$ that was dimensioned in Line 10. Twenty blank lines are printed to clear the screen.

LIST 50,5 - The letter P is used to keep track of which problems we're on. It's set to 1, and increased each time the program returns to Line 60. The letters A and B are each set equal to a random number between 1 and 10. If A is smaller than B, two new numbers are generated because we want to avoid A-B which produces a fractional answer or A-B which produces a negative number.

LIST 100,11 - Creating problems is more difficult than solving them, at least for your computer. First, a random number between 1-4 is picked to select what kind of problem to create. Then the computer is directed to one of four different places in the program. Line 130 and 140 creates an addition problem and sets X (the answer) equal to A + B. Line 150-160 creates a subtraction problem with X equal to A − B. We know that X will be a positive number because A is larger than B (Line 90). Lines 170 and 180 create a multiplication problem. Lines 190-200 create a "pseudo-division" problem, where we try to keep the answer simple, these 2 lines will produce a division problem that will always have an answer in whole numbers or integers of number.

LIST 210,5 - Now it's your turn. Each answer you type is entered to variable G and scored at this point in the program. If your guess equals the answer (G = X), then R increases by one and the music plays. If your guess is not equal to X, then W increases by one and a low pitch note is played. The statement NEXT P tells the computer add one to P and to go back to line 60: "FOR P = 1 TO T". If P (the number of the problem we're on) is equal to T (the total number of problems you requested) then we're done, and the computer goes on to the score card. If P is not yet equal to T, the computer starts over again at Line 60.

LIST 260,7 - After you've received and answered all the problems you requested, it's report card time. The screen is cleared and the number you got right (R), the number you got wrong (W) and the total number of problems (R + W) are printed with the appropriate labels. If the number of wrong answers is zero (W = 0), you get to hear a tune. If you can think of a better reward, it goes here in the program.

LIST 330,4 - The variable that was dimensioned at the start of the program is used here to store the letter typed on the keyboard. If Key$ is Y, the program repeats. Any other answer stops it.

Lesson 20

CODE BREAKER

This lesson shows you how a program can create a popular electronic game. We've selected the game where you try to match a secret code and are given clues until you get the right answer. In this version, the computer picks the random code and scores each guess you make. You select how many digits or positions are used in the code and also select how many possible number combinations the computer will pick from. Varying these choices programs the game to match any skill level from beginner to expert.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 19 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 20. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 20. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 20.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 20, follow the loading instructions on Chapter 5, page 32. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 79. This should position your BASIC TUTOR tape for Lesson 20.

RUN CODE BREAKER

After you type **RUN** and press the **RETURN** Key, the program requests the number of positions you wish to have in the code. More positions are more difficult, so type **3** for a start and press the **RETURN** Key.

Now pick how many different numbers you wish the computer to use in selecting the code. With more numbers to choose from, the program will create a more difficult code for you to guess. Type **2** again and press the **RETURN** Key.

The computer has now created a code number that you will try to guess. There are three positions in the number and only the digits 1 and 2 are used. The answer will be a number like 122, 112, 212 or some other combination. If you only select 2 numbers for the number of different numbers you wish in the code, the computer will always use 1 and 2. Similarly, if you request 3 different numbers, then the computer will always use 1, 2, and 3, etc.

Try to guess what the code number is and press the keys on the keyboard to enter your guess. After you've typed three numbers, the computer will give you several hints to tell you how you are doing:

1. If the computer prints a blue box next to your guess, you have picked a correct number and that number is in the correct position in the code. For example, if the code number is 211 and you guess 222, the computer will score 1 blue box because the 2 is one of the digits in the code and is in the right place. After you play the game a while, you will notice that the blue box indicates a correct number in a correct position, but it does not indicate which position is correct, merely that one of them is.
2. If the computer prints an orange box, you have picked a correct number that is not in the correct position. For example, if the code number is 313 and you guess 232, the computer will score one orange box because 3 is one of the digits in the code and it is not in the position you guessed.
3. If none of the digits you guess are in the code number, the computer prints no score and moves the screen up one line so you can try again.
4. When you guess the correct code with all digits in their correct positions, the computer will score a blue box for each position and play a tune.

Try running through this program several times, each time increasing either the number of code positions or number of different numbers in the code. After you've broken each code with the correct answer, the program will ask if you wish to go again. Remember that the code gets much harder to break as you increase either the positions or the numbers.

HOW CODE BREAKER WORKS

One of the ways you can create computer games that satisfy a wide audience is to allow each user to select how difficult the game will be. In this example, it is possible to select a 3 position, 2 number code that's very easy to solve. A 5 position, 4 number code, on the other hand, is a real challenge and anything with more numbers or positions than this is for experts.
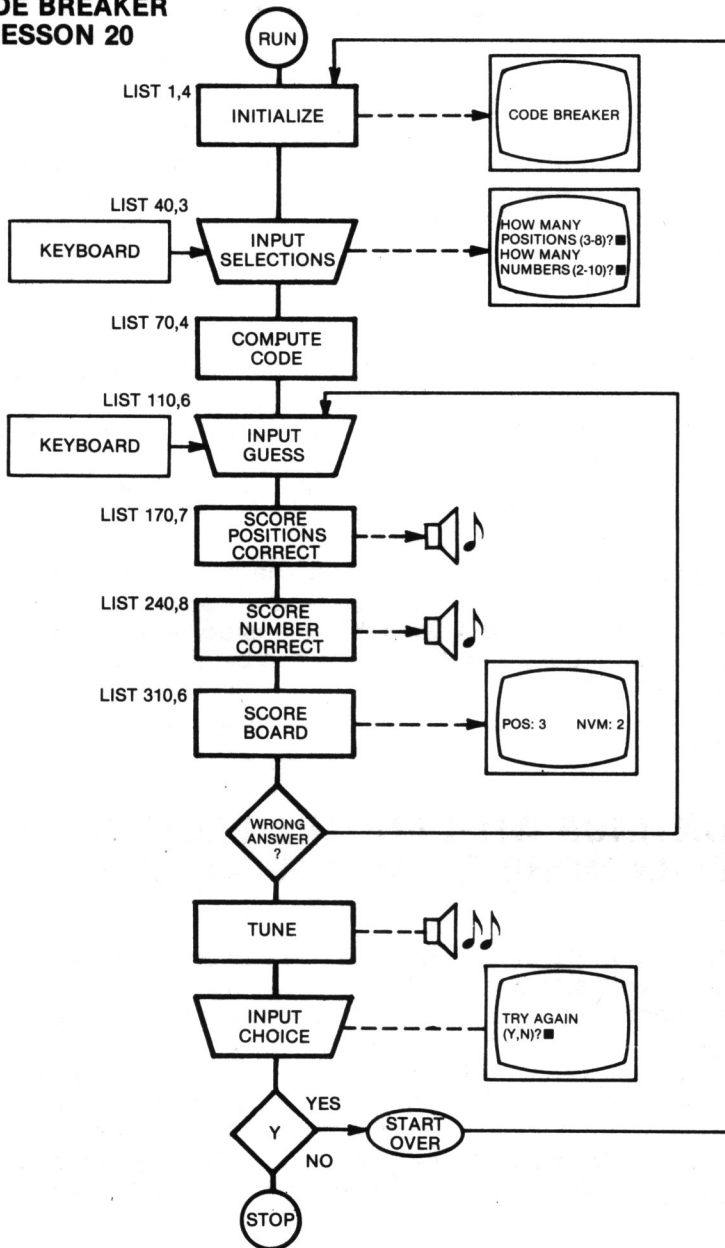
There are also several interesting variations in the problem that can be selected. For example, you might try a 6 position, 2 number code. This is a problem in binary logic. Giving flexibility like this to the program tends to create computer games that both children and adults can enjoy.

The program is not complex at all. The computer selects a code, based on the positions and numbers specified. Each guess is evaluated two ways: If the positions are correct, blue boxes are printed. If only the numbers match, orange boxes are printed. Finally, if all the numbers are in the right positions, a tune is played and the program asks if you wish to try again.

CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 20?

You can now continue with Lesson 20 to see how Code Breaker works and learn how to write certain programming statements, or you can go directly to Lesson 21. If you want to go directly to Lesson 21 (Music Box) and your BASIC TUTOR tape is still in the machine, type **CLOAD**, follow the instructions on the screen and the computer will automatically begin loading. If you want to go directly to Lesson 21 (Music Box) and your BASIC TUTOR tape is not in the machine, see page 32, Chapter 5 for instructions.

CODE BREAKER
LESSON 20

RUN

LIST 1,4 — INITIALIZE ----> CODE BREAKER

LIST 40,3 — KEYBOARD — INPUT SELECTIONS ----> HOW MANY POSITIONS (3-8)?■ HOW MANY NUMBERS (2-10)?■

LIST 70,4 — COMPUTE CODE

LIST 110,6 — KEYBOARD — INPUT GUESS

LIST 170,7 — SCORE POSITIONS CORRECT ----> ♪

LIST 240,8 — SCORE NUMBER CORRECT ----> ♪

LIST 310,6 — SCORE BOARD ----> POS: 3   NVM: 2

WRONG ANSWER ?

TUNE ----> ♪♪

INPUT CHOICE ----> TRY AGAIN (Y,N)?■

Y
YES — START OVER
NO

STOP

```
1   REM    ...CODE BREAKER...
10   DIM K$(1)
20   DIM K(9): DIM A(9)
30   FOR N=1 TO 16: PRINT : NEXT N
40   PRINT "...CODE BREAKER..."
50   INPUT "HOW MANY POSITIONS (3-7)",L
60   INPUT "HOW MANY NUMBERS (2-9)",X
70   REM    ...COMPUTE CODE...
80   FOR N=1 TO L
90 A(N)= INT ( RND (0)*X+1)
100   NEXT L
110   REM    ...INPUT GUESS...
120   FOR N=1 TO L
130 K$= KEY$ (0): IF K$="" GOTO 130
140 K(N)= ASC (K$)-48
150   PRINT " ";K$;: MUSIC "1"
160   NEXT N
170   REM    ...SCORE POSITION...
180 R=0:X=2*L: COLOR =2: SHAPE =3
190   FOR N=1 TO L
200   IF A(N)<>K(N) GOTO 230
210 R=R+1:X=X+2: MUSIC "+1"
220   PLOT X,15
230   NEXT N
240   REM    ...SCORE   NUMBER...
245 Y=-R
250   FOR P=1 TO L: FOR N=1 TO L
260   IF A(P)<>K(N) GOTO 300
270 Y=Y+1: COLOR =7: MUSIC "3"
280   IF Y>0 THEN X=X+2: PLOT X,15
290 K(N)=0:N=L
300   NEXT N: NEXT P
310   REM    ...SCORE BOARD...
320   PRINT
330   IF R<L GOTO 110
340   MUSIC "300000500030405000"
350   INPUT "TRY AGAIN (Y,N)",K$
360   IF K$="Y" GOTO 30
```

207

Stop Code Breaker by selecting **N** when the program asks if you wish to continue or by pressing the **BREAK** Key.

LIST 1,4 - The first line is the title. The variable K$ is dimensioned to hold 1 letter. Two other variables are dimensioned to hold up to 9 numbers each. We will use K for the code and X for the answer that will be typed in each turn.

LIST 40,3 - Two inputs are required. The program sets L equal to the number of positions and X equal to the number of digits. The words in quotation marks are printed on the screen to tell the user what is required and what the range of answers can be in each case.

LIST 70,4 - For each position in the code, the computer picks an integer between 1 and X, the largest number that the user has requested for this problem. Each number picked is stored in the ARRAY A in the appropriate position. The first position is A(1), the second position is A(2), the third is A(3), and so forth.

LIST 110,6 - Now a similar process is used to store the answers in ARRAY K, with K(1) equal to the first number typed in by the user, K(2) equal to the second number, and so forth. In Line 130, the number typed on the keyboard is first examined to see it's a null (""). If so, the program goes and gets another entry. In this way, the computer simply waits until a key has been pressed.

Each number typed on the keyboard is stored in ARRAY K and printed on the screen. After storing a number in ARRAY K, a BEEP is made.

LIST 170,7 - Remember that L is the number of positions in the code. Beginning with 1 and going to L, the computer compares each number in the guess — ARRAY K — with each number in the computer's code — ARRAY A. If they're not the same, the program goes to Line 230 for the next number. If they're equal, then you have a number in the correct position. A blue square in SHAPE 3, COLOR 2 is plotted on the screen and a BEEP is played in the speaker.

The variable X is used to keep track of the left-right position on the screen. The program moves X two spaces to the right after each blue box.

We will need to know the total number of correct positions. This total is stored in R, which begins as a zero and increases one for every blue square.

LIST 240,8 - Figuring out how many different numbers you have correct — regardless of position — is handled in a different way. Here, we must check each number in the entered answer against all numbers used in the generated code. This is done with a double loop. The outer loop P scans every position in the generated code. It starts with the first position. The inner loop N moves from the first position in the entered answer, to the second number in the entered answer, and so forth. For each number in A that's equal to any number in K, the program plots a COLOR 7 box and BEEPS. When the position being scanned in the outer loop is equal to L, (the number of digits in the code) the program is finished with this step.

The orange squares should be scored *only* if the same digit doesn't qualify for a blue square. This is achieved with the variable Y. Since R is the number of blue squares we put up on the screen, we set Y equal to $-R$ so that Y begins with a negative balance. The program advances Y for each number match. When Y is larger than 0, the number of blue squares has been surpassed and the program starts plotting orange squares on the screen.

LIST 310,6 - After printing a blank line to scroll the picture on the screen, the computer checks to see if R (the number of blue boxes) is less than L (the number of positions in the code). If not, the program goes to Line 110 for another guess. If so, the MUSIC in Line 340 is played indicating you have guessed the code and the offer to **TRY AGAIN** is printed on the screen. Type a **Y** and start over for a new code. Any other key stops the program.

Lesson 21

MUSIC BOX

This lesson contains a popular tune. Words are printed on the screen while the notes are played in the speaker.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 20 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 21. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 21. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 21.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 21, follow the loading instructions on Chapter 5, page 32. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 92. This should position your BASIC TUTOR tape for Lesson 21.

RUN MUSIC BOX

To run this program, just type **RUN** and press the **RETURN** Key. Adjust the volume of your TV speaker to hear the song as the words are being displayed. Run the program as many times as you like and listen to the way the notes sound. Some of the notes are short and are played quickly, and other notes are held for a longer time.

HOW MUSIC BOX WORKS
(See flow chart on following page)

Even if you haven't had any musical training, you will find that it's easy to program your computer to play notes and songs.

**MUSIC BOX
LESSON 21**
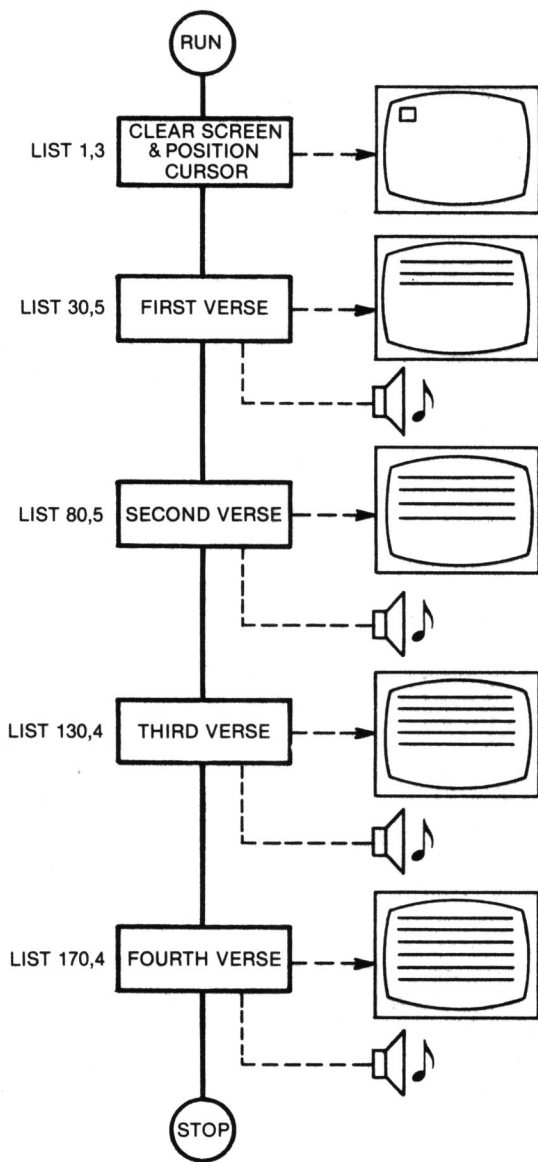
There are 7 notes in the musical scale: DO, RE, MI, FA, SO, LA and TI. Each of these notes is given a number from 1 to 7. Type the following instruction on the keyboard. Spell out the word **MUSIC** and be sure to use quotation marks (") around the numbers, exactly as shown.

MUSIC"1234567" — (1 octave scale)

Press the **RETURN** Key to hear this musical scale in the speaker.

You can shift these notes 1 octave higher by typing a multiply sign (*) in front of each note. A division sign (/) lowers the scale 1 octave. Program a 3 octave scale with this next instruction.

MUSIC"/1/2/3/4/5/6/7/1234567*1*2*3*4*5*6*7"
(3 octave scale)

As you will hear when you press the **RETURN** Key, this will play a sliding scale that covers 3 octaves.

You can also play sharps and flats by adding a plus sign (+) for sharps and a minus sign (–) for flats. If you compare your computer to a piano, the numbers by themselves play the white keys and the numbers with either a plus or a minus play the black keys.

If you add a space when typing the numbers, the computer will skip a space in the music. Try this example with 2 spaces between the notes and see how the sound changes when spaces are used. Remember to add the spaces after the last note, to use the quotation marks, and to press the **RETURN** Key when you're done.

MUSIC"3 2 1 2 3 3 3 "

There's one more trick you need to know to play music on your computer. Adding zeroes after the notes makes the notes continue playing. Try the example again, using zeroes instead of spaces between notes.

MUSIC '300200100200300300300"

You may want to repeat these two examples to hear the difference.

Now we'll use everything we have covered so far and play a tune. Remember that the divide sign (/) lowers the pitch one octave and the minus sign (−) lowers the pitch one half step (like the black keys on a piano). We'll use zeroes to make some of the notes last longer.

MUSIC"/400/40/4/400 − /60/5/50/4/40 − /4/400"

CONTINUE WITH ANOTHER LESSON, OR LEARN MORE FROM LESSON 21?

You can now continue with Lesson 21 to see how Music Box works and learn how to write certain programming statements, or you can go directly to Lesson 22. If you want to go directly to Lesson 22 (Targets) and your BASIC TUTOR tape is still in the machine, type **CLOAD**, follow the instructions on the screen and the computer will automatically begin loading. If you want to go directly to Lesson 22 (Targets) and your BASIC TUTOR tape is not in the machine, see page 32, Chapter 5 for instructions.

```
1   REM    ...MUSIC BOX...
10  FOR N=1 TO 32: PRINT : NEXT N
20  POKE 40960,2: POKE 40961,0
30  PRINT : PRINT "I'VE BEEN WORK-ING ON THE "
40  PRINT "RAIL-ROAD."
50  MUSIC "100/51/51230001000"
60  PRINT "ALL THE LIVE-LONG DAY."
70  MUSIC "400410203000000"
80  PRINT "I'VE BEEN WORK-ING ON THE"
90  PRINT "RAIL-ROAD."
100  MUSIC "100/51/512300010"
110  PRINT "JUST TO PASS THE TIME A-WAY."
120  MUSIC "33302020302000000 "
130  PRINT "CAN'T YOU HEAR THE WHIST-LE": PRINT "BLOW-ING?"
140  MUSIC "2002+12321000/5000"
150  PRINT "RISE UP SO EARLY IN THE MORN."
160  MUSIC "404411223000000 "
170  PRINT "CAN'T YOU HEAR THE CAP-TAIN": PRINT "SHOUT-ING?"
180  MUSIC "/6000/71/71/6/50001000"
190  PRINT "DI-NA BLOW YOUR HORN!"
200  MUSIC "3040302010000000"
```

LIST 1,3 - The program clears the screen by printing several blank lines. Then the printing is started in the upper left corner of the screen.

LIST 30,5 - These lines print the first verse and play the music in the TV speaker. Try to follow the tune by reading the numbers. Remember that the divide sign is used to lower the pitch 1 octave.

LIST 80,5 - These 5 lines play the second verse with words displayed. We use two lines (80 and 90) to print the words so that the word **"RAIL-ROAD"** will be placed on the next line.

LIST 130,4 - Again, we use 2 PRINT lines so that the word **"BLOW-ING"** will be placed on a line by itself. This makes the words easier to read. The space at the end of the Line 160 is used to end the note.

LIST 170,4 - The last verse is printed in the same way, with a separate PRINT for "SHOUT-ING" so that this word will be printed on a separate line. Notice the last line in the program. Here, we use a number and a zero for each of the first 4 notes, then a number and 7 zeroes for the last note in the song. The first 4 notes last for 1 beat and the last note lasts for 4 beats.

Lesson 22

TARGETS

For many people, making a computer game is more fun than playing someone elses program. Here's an example of a game program with lots of action and sound effects. You can use it as a model for writing your own program or to create versions by changing or adding to the instructions.

LOADING INSTRUCTIONS

If your tape is still in the machine after you have used Lesson 21 and the cursor is on the screen, simply type **CLOAD** and press the **RETURN** Key. Follow the instructions and the computer will begin loading Lesson 22. Adjust the volume control next to the speaker on the Imagination Machine keyboard and listen for introduction on Lesson 22. The audio on the tape will inform you when to press the **RETURN** Key. When the word **OK** appears on your screen, type **RUN** and press the **RETURN** Key. You are set to begin Lesson 22.

If you have removed the BASIC TUTOR tape from the machine and want to begin with Lesson 22, follow the loading instructions on Chapter 5, page 32. If you type in **CLOAD** and rewind the tape all the way, you should push the digital tape counter and set the counter back to 000. Advance the tape via the fast forward button to 104. This should position your BASIC TUTOR tape for Lesson 22.

Please remove BASIC TUTOR tape #2 from the Imagination Machine after your program has been loaded.

RUN TARGETS

Type **RUN** and press the **RETURN** Key. The program begins with the computer painting a blue sky on your screen. Then colored targets appear, with an orange square in the center.

Adjust the volume control on the TV to hear the score as you cover targets with the orange square. The object is to position the orange square on top of the targets; as you put the square on the target your TV will BEEP and the target will disappear. Use the right hand control knob to move the orange square around the screen.

Here's the scoring for each target:

> Blue   = 1
> Red   = 3
> Yellow = 6

Watch the score at the top of the screen. Get over 46 points and a free game. You'll have to move quickly, because the targets won't last long. If you fail to get enough points to go again, just press the **FIRE** button or the **EN** Key on the right hand controller to start a new round.

HOW TARGETS WORK

After covering the screen with rows of light blue squares to make the background, the program generates the targets and places them randomly on the screen. The next loop in the program is to look at the input for the right hand controller and then the loop will be repeated.

The program repeats this cycle and continues moving the cursor in response to the hand control. This loop will repeat until either the **FIRE** button is pressed or a target is a hit.

Pressing the **FIRE** button causes the program to restart with a new set of targets.

Hitting a target with the cursor causes the program to branch to the next section. The first thing the computer does after a target has been covered by the cursor is remove the target from the screen.

You probably noticed that the targets tend to disappear now and then. This feature is added to give the game a sense of urgency and to make it challenging to get points before the high-scoring targets are all gone.

TARGETS
LESSON 22

RUN

LIST 1,3 → INITIALIZE

LIST 100,5 → DRAW BLUE SKY

LIST 200,7 → PLACE TARGETS

HAND CONTROL → INPUT DIRECTION

LIST 300,5

LIST 350,8 → MOVE CURSOR

FIRE ? — YES → START OVER
NO

LIST 500,5 — A HIT ? — YES → ERASE TARGET
NO

COMPUTE SCORE

BLUE → ADD 1

RED → ADD 3

YELLOW → ADD 6

LIST 600,7 → PRINT SCORE → SCORE: 10

SCORE < 46 ? — YES
NO

TUNE

START OVER

219
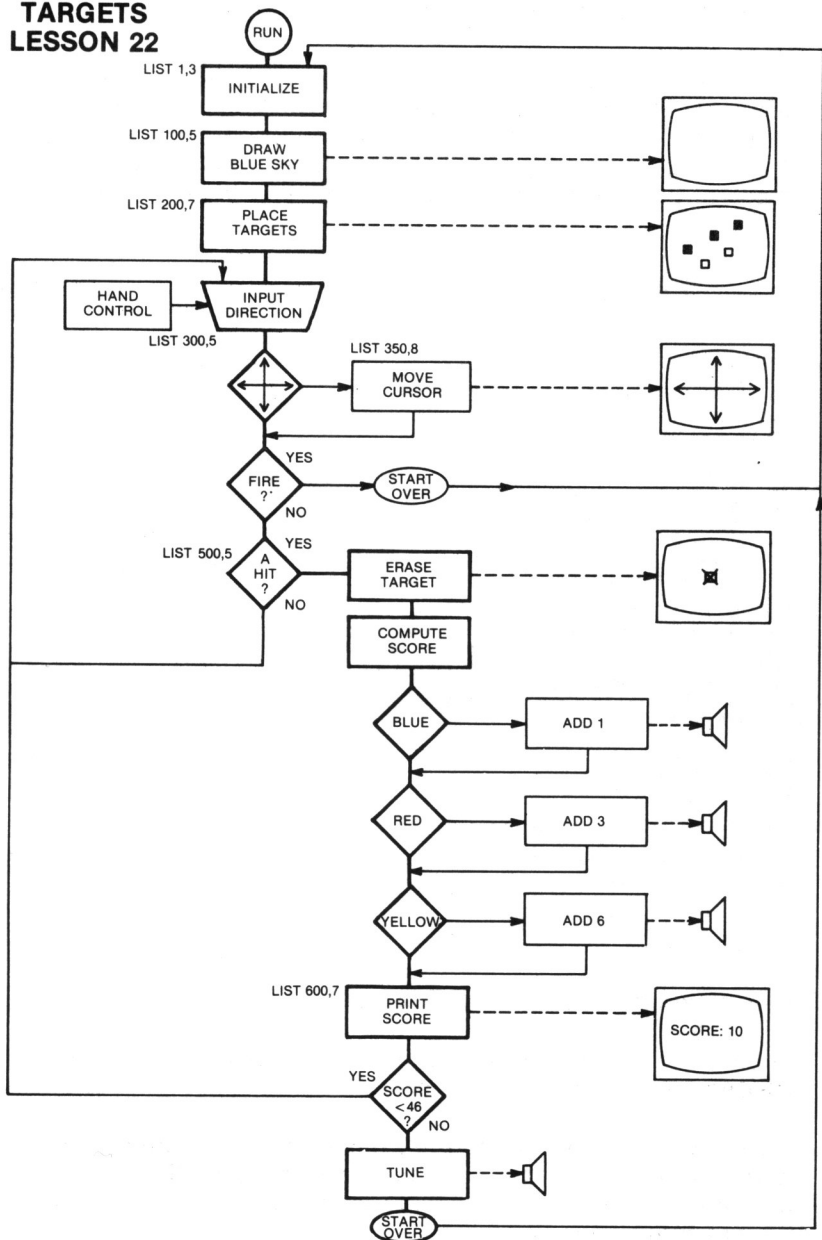
PROGRAM NOTES

This is the last lesson in BASIC TUTOR. Now that you've come this far, we suggest you study this lesson program listing and follow along with our notes we have made about the listing. This program contains music, action, color and many of the BASIC commands we have studied.

```
1   REM    ...TARGETS...
10  DIM K$(1): SHAPE =15
20  X=16:Y=8:S=0
100 REM    ...BLUE SKY...
110 COLOR =5
120 FOR A=0 TO 15
140 HLIN 0,31,A
150 NEXT A
200 REM    ...PLACE TARGETS...
210 FOR N=1 TO 25
220 A= RND (0)*32
230 B= RND (0)*16
240 C= RND (0)*3+1
250 COLOR =C: PLOT A,B
260 NEXT N
300 REM    ...MOVE CURSOR...
310 K$= KEY$ (1)
320 COLOR =5: PLOT  RND (0)*32, RND (0)*15+1: PLOT  RND (0)*32, RND (0)*15+1
330 IF K$="" GOTO 410
340 PLOT X,Y
350 IF K$="S" THEN Y=Y+1: IF Y>15 THEN Y=15: MUSIC "100"
360 IF K$="N" THEN Y=Y-1: IF Y<0 THEN Y=0: MUSIC "100"
370 IF K$="E" THEN X=X+1: IF X>31 THEN X=31: MUSIC "100"
380 IF K$="W" THEN X=X-1: IF X<0 THEN X=0: MUSIC "100"
390 IF K$="!" THEN  GOTO 20
400 IF   PEEK (512+Y*32+X)<>223 GOTO 500
410 COLOR =7: PLOT X,Y
420 GOTO 300
500 REM    ...A HIT...
510 T= PEEK (512+Y*32+X)
520 IF T=175 THEN S=S+1: MUSIC "1"
530 IF T=191 THEN S=S+3: MUSIC "3 3 3"
540 IF T=159 THEN S=S+6: MUSIC "5 5 5 5 5"
600 REM    ...SCORE CARD...
610 POKE 40960,2: POKE 40961,0
620 PRINT " SCORE:";S;" ";
630 IF S<46 GOTO 300
640 PRINT ">>-> EXTRA GAME <--<<"
650 MUSIC "103050*10  50*100000"
660 GOTO 20
```

(See flow chart on following page)

To see the first set of instructions for Targets, press the **BREAK** Key to stop the program.

LIST 1,3 - In this initialization step we set the dimensions and adjust the variables for the program. SHAPE number 15 is a solid square, used for the targets and for the orange square that you move.

The two variables X and Y are used to show the position of the cursor on the screen. Since the screen is 32 units wide, setting X = 16 positions the cursor halfway between the left and right edge of the screen. Variable Y is set to 8. This is halfway between the top and bottom because the screen is 16 units high. Later, when the cursor is plotted at the position designated by X,Y it will be placed in the center of the screen. The variable S is used to contain the score and is initialized to 0.

LIST 100,5 - Here's where the computer paints the sky light blue. There is one loop used. We will make a horizontal line from the left side of the screen (column 0) to the right side of the screen (column 31). The first time through the loop the line will occur on row 0 (A = 0), next row 1 and then through to the bottom row.

LIST 200,7 - Now the targets are placed on the screen. Another loop is run 25 times to put 25 targets at random locations. Here, variable A is used to position the targets left and right, and is set to a random number between 0 and 31. The vertical direction is set by variable B and is a random number between 0 and 15. Each time a target is plotted on the screen, the color is set to a random number between 1 and 3. One is the color yellow, two is blue, and three is red.

LIST 300,5 - This is the section that reads an input from the right hand controller as well and randomly removing some targets from the screen. Line 310 gets an input from the controller. Right after that, Line 320 sets the color to the sky (blue) and plots 2 squares randomly. If it plots on top of where a target is, then the target disappears. Line 330 checks if K$ is a null. If it is, it goes to Line 410. Line 340 plots the cursor at its current position in case it was removed by Line 320.

LIST 350,8 - This next section examines the input from the controller and takes appropriate action. Line 350 through 380 will check to see if the knob was pushed to 1 of the 4 positions and changes the cursor position (stored in X and Y) appropriately. Notice that it also checks if the cursor was attempted to be moved off the screen and makes adjustments for this. Line 390 checks if **EN** or **FIRE**

was pressed and goes back to the beginning and starts a new sound. Line 400 examines the location of the cursor to be moved on the screen. The screen occupies memory locations 512 through 1023. This PEEK instruction looks at that location and compares it to 223. 223 is the code stored in that memory location for a blue square (sky). If the PEEK shows any other number, it goes to 500.

410 plots the cursor on the screen with new X and Y values. 420 sends it back to Line 300.

LIST 500,5 - Get a hit and the computer goes to these instructions. The variable T is set to the number used by the computer in filling the square at where the new cursor location X,Y is to be plotted. We have already tested this PEEK to see that something other than light blue sky is at this location. If this location on your screen is filled with a dark blue square, then T will be equal to 175, and one is added to the score with a BEEP sent to the speaker. If T is 191, the location is colored red and 3 points are scored. Finally, if T is 159, the location is yellow and the maximum of six points are awarded.

Notice that the variable S is used to keep the score in each case.

LIST 600,7 - The computer updates the scorecard by printing the score in the upper left corner of the screen. Line 610 positions the printing and the next line prints the score. If the score is less than 46, the program returns to the main loop and continues moving the cursor in response to the hand control. A score of 46 or better gets an extra game, a tune, and an automatic restart to Line 20.

Chapter 7
CREATING YOUR OWN PROGRAMS

The first thing you need to remember when you first start to write your own programs is that your computer is virtually indestructable. That means you can type in just about anything you'd like and nothing damaging will happen to your Imagination Machine. Now maybe you won't write a perfect program at first, but our error message commentary will guide you, and you certainly will not damage your computer by attempting to create a program of your own. Remember, if you turn off the power or press **RESET**, you will lose any program or date stored in memory, but your computer will be unharmed.

The key to creating your own programs can be summed up in one word: PRACTICE. Like most new pursuits, from learning how to play baseball or a musical instrument, to writing a story or play, program writing takes practice and patience. It takes the ability to learn from your mistakes, to have fun while you're learning and to keep on applying yourself until you master a particular programming concept. Nothing beats hands-on learning, so please don't be afraid to try.

You can, however, quickly learn to write short programs that are clear and do what you want. Begin small, with just a few instructions. Start by typing in a small number of programming statements, listing those statements and running that portion of the program to test its accuracy. Then start to add other sections, one at a time. Run and test each portion of the program, so that you don't introduce errors in one section that may confuse you in another.

You might begin by designing a simple game, or by modifying one of the 22 BASIC TUTOR lessons and making it your own. Look again at the program listings for the lessons you have just completed. Try to copy some of them, modifying certain sections or statements and watch how the programs change. Don't think that to be a good programmer you have to "create" everything from scratch. Most

experienced programmers refer to existing programs and often copy or modify other programs or routines.

We might also suggest that you look through any of the numerous computer hobbyist magazines available on the newsstands. Most contain sections on program writing and give several sample programs. Copy those programs, change them around, make some mistakes, and watch how much fun you'll have! Remember, however, that these programs may be written in different variations of BASIC than APF BASIC. Make sure that you have a good understanding of APF BASIC before you attempt to copy or change any variation.

Remember, when you get confused and don't understand something in your program, you've probably done one of two things. Either you have stumbled in the BASIC language and are not using it correctly, or you have not thought out your ideas and are unsure of what you want the computer to do. You can resolve most problems with BASIC words by writing in short program sections to see how your program works. Creating flow chart diagrams, or even writing out some ideas in longhand if you prefer, will help you keep your ideas in focus.

Above all, we urge you to use your Imagination Machine and use your Imagination!

CSAVE . . . How to Save Programs to Tape

CSAVE is used to save program memory. It is not necessary to press the record button on the tape deck since that is used only to record audio. Do not delay more than 5 seconds when pressing the **PLAY/SAVE** button and the **RETURN** Key. For an 8K system CSAVE takes about 45 seconds.

During a CSAVE all keyboard functions are inoperative: CSAVE will save the entire amount of random access memory.

Saving 512 Block for Screen

In addition to the program memory. CSAVE first saves a block of 512 bytes. This will be placed on the screen first when loading so pictures or messages can be stored. There are 2 possible blocks that can be saved. The selected one is determined by the contents of memory location 41451.

If (41452) = 0 then locations 512-1023 (screen memory) are saved.
If (41452) = 255 then locations 0-511 (scratch memory) are saved.

EDIT ... How To Modify A Line

The EDIT statement is used to change a portion of a line in your program. It must be followed by the line number you want to change and the ESC Key. Then type the section of the statement you want to change. Press the **ESC** Key again and then the new characters.

EXAMPLE: **EDIT N ESC X ESC Y RETURN**
N is line - to be edited
X is text to be deleted
Y is text to replace X

EXAMPLE: **10 DIM A$(5), IN$(5)**
100 INPUT A$: PRINT IN$
200 IF IN$ = "NOT" THEN 1000
300 IF IN$ = "YES" THEN 100
EDIT 100[A$[IN$
EDIT 200[T[
EDIT 300[100[RETURN
1000 STOP

LIST

100 INPUT A$: PRINT IN$
200 IF IN$ = "NO" THEN 1000
300 IF IN$ = "YES" THEN RETURN

Chapter 8
DICTIONARY / REFERENCE GUIDE

This chapter contains a quick-check summary of terms and keywords used in BASIC. Use it as a handy reference guide or as an index to where detailed explanations are contained.

Note: **(EXPR)** means a numerical value or the result of a numerical expression (STR$) means string value.

TERM	EXAMPLE	MEANING	REFERENCE
A			
ABS(EXPR)	PRINT ABS (X) P = ABS (Y)	A function that returns with the absolute value of an expression. A number's absolute value is its value without a (+) or (−) sign	—
ARRAYS	DIM X(10) DIM Y(6,20) DIM S$(6) DIM P$(9.50)	A collection of values grouped together by a common name. Must preceed an array with Dimension Statement. Each member or subscript of an array can have a different value.	Lesson 14
ASC (STR$)	PRINT ASC ("A") D = ASC (C$) PRINT ASC (A$(4))	Converts a character string variable to the ASCII integer code. Will return with first character of Str$ unless specified.	Lesson 8
ASSIGN-MENT	PRICE = 1.3	Statement that gives (assigns) a value to a variable. Value can be a number, result of a calcula-tion, a name or a word.	Chapter 4
B			
BREAK	—	System command which allows keyboard interruption of pro-gram. Tells computer to return to immediate mode.	Chapter 3
C			
CALL EXPR	CALL 17046	Tells computer to branch to MC6800 machine language routine which starts at memory location (EXPR)	Lesson 8
CHR$(EXPR)	PRINT CHR$ (63) A$ = CHR$ (48)	Used to obtain the single char-acter represented by ASCII number code in ( )	Lesson 8
CLOAD	—	Turns on tape deck motor and audio. Will load program from tape to memory.	Chapter 5
COLOR	Color = 4 Color = A	Sets color to be drawn on screen in either PLOT, HLIN or VLIN statements. Eight colors are available.	Lesson 7

TERM	EXAMPLE	MEANING	REFERENCE
CSAVE	—	Save to tape. First saves 512 bytes which will be loaded to screen, then saves all of RAM memory.	Chapter 7

D

TERM	EXAMPLE	MEANING	REFERENCE
DATA	300 DATA 1,2,3,4,5 310 DATA ⁹JAN,⁹ ⁹FEB,⁹ ⁹MAR⁹	Specifies data to be read with a READ statement. The list may be constants or strings. List elements must be separated by commas. Data items will be read sequentially.	—
DELIMETER	PRINT "HELLO", "OUT THERE" PRINT "HELLO"; "OUT THERE"	Sets the boundaries between items we wish to be printed. BASIC delimeters are , and ;	Chapter 3
DIM	DIM A$ (10)	Tells computer to reserve space in its memory for letters, words or tables of numbers. Sets all initial values of string variables to null. All string variables must be dimensioned.	Lesson 8

E

TERM	EXAMPLE	MEANING	REFERENCE
EDIT	100 INPUT A$: PRINT IN$ EDIT 100 [A$ [IN$ LIST 100 INPUT IN$: PRINT IN$	Used to change portion of line in program. Used with ESC key.	Chapter 7
END	50 IF A＝100 GO TO 70 60 PRINT B$ 70 END	Terminates program execution and returns cursor to screen.	Chapter 3
EQUATION	COST = PRICE *NUMBER	The procedure or steps followed to achieve a result.	Chapter 4
ERROR MESSAGE	WHAT PRINT DELIMETER	Computer tells you that you've done something wrong. If they occur during program statement, statement number and error message is displayed.	Chapter 3 (plus appendix A)

F

TERM	EXAMPLE	MEANING	REFERENCE
FLOW CHARTS	—	Visual programming aid that takes program statement and shows them in sequence via pictorial representation.	Chapter 4
FOR	FOR L = 1 to 10 FOR X = Y to Z	Opens a FOR — NEXT Loop and initializes value of variable and increments it by amount in expression each time NEXT statement is executed.	Lesson 3
FORMULAS	PRINT D/T; "MILES PER HOUR"	Formulas are solved directly and you may use any formula in a program.	Lesson 17

TERM	EXAMPLE	MEANING	REFERENCE
G			
GOTO	50 IF A = 20 GO TO 100	Allows change in normal sequence of statement execution. Program jumps back or goes to specified line number	Chapter 3
GOSUB	GOSUB 100	Allows transfer of program to subroutine that can be written once but used many times in program.	Lesson 16
H			
HLIN	SHAPE = 15:COLOR = 4 HLIN 6, 10, 4 HLIN X1, X2, Y	Draws a horizontal line/or shape at a specified row on screen from column X1 to column X2 in Row Y	Lesson 18
I			
IF	IF A + B = C THEN PRINT "YOU WIN"	Tells the computer to test something. It tests a condition that follows the word IF. If it is true, then BASIC will execute statement following word THEN. If false, program will continue to next step number.	Lesson 2
IMMEDIATE MODE	PRINT "HI"	Tells computer to execute command at once rather than store it for later use.	Chapter 3
INPUT	INPUT "PRICE", P INPUT "NAME", A$ INPUT "COST", C, "MILES", M	Allows assignment to a variable by entering value from keyboard.	Lesson 1
INT(EXPR)	INT (3.14) = 3 PRINT INT (7.1*8.3) 56	Returns the integer portion of the expression. The expression can be a number, a numeric variable or a calculation.	Lesson 6
K			
KEY $(0)	10A$ = Key $ (0)	Reads the computer keyboard and looks for an input. Key $ (1) reads right controller of MP-1000. Key $ (2) reads left controller.	Lesson 5
L			
LEN (STR$)	DIM A $ (10) A$ = "ABC" PRINT LEN (A$) 3 Length is 3	Returns # of characters (length) of the string variable	Lesson 8
LET	10 LET PRICE = 1.3	Optional word in APF BASIC that can appear before variable name in assignment statement.	Chapter 4

229

TERM	EXAMPLE	MEANING	REFERENCE
LIST	LIST LIST 100 LIST 200,5	System command to display program. Program will appear in numerical order starting with lowest step number. Can also call out specific program steps or list parts of program.	Chapter 3

M

TERM	EXAMPLE	MEANING	REFERENCE
MUSIC	MUSIC "1234567"	Enables computer to play music using numbers 1 through 7.	Lesson 11
MULTI-STATEMENTS	PRINT B: INPUT C 50 PRINT A:	Allows more than one BASIC statement per line number. Must be separated by:	Lesson 2

N

TERM	EXAMPLE	MEANING	REFERENCE
NEXT	200 NEXT I 300 NEXT A,B	Used to return program execution to FOR statement in FOR — NEXT Loop so that current value of variable can be increased until it exceeds end value.	Lesson 3
NULL	DIM B$ (15)	A character that is nothing and cannot be printed on screen. When strings are initially dimensioned, each position is assigned code for a NULL character.	Lesson 8

O

TERM	EXAMPLE	MEANING	REFERENCE
ONGOSUB	ON A GOSUB 200, 400, 600	Goes to a subroutine depending on a variables value	Lesson 15
ONGOTO	ON A GOTO 100, 240, 560	Changes program flow depending on variable's value. There can be up to 9 statement numbers that GOTO can direct program to.	Lesson 15

P

TERM	EXAMPLE	MEANING	REFERENCE
PEEK(EXPR)	PRINT PEEK (512) 128	Examines location in computer memory and returns the numeric value contained there.	Lesson 11
PLOT	PLOT 2, 1 PLOT X, Y	Draws a shape on the screen in one of 512 boxes. Each box has a row and column number.	Lesson 7
POKE	POKE 784,65	Places a number in a memory location. Often used to change cursor pointer contents used in a print statement.	Lesson 12
PRECEDENCE	PRINT 4 + 6/3 6 PRINT (4 + 6)/3 3.3333	Set rules of which operations are done first in a mathematical expression. Rules of precedence can be offset by ( ) or [ ].	Lesson 1
PRINT	PRINT 2 + 3 PRINT "HELLO" PRINT A$	Tells computer to display on screen the results of arithmetic problem, something enclosed in quotation marks or a variable value.	Chapter 3

TERM	EXAMPLE	MEANING	REFERENCE
PRINT USING	10: ### ## 20 PRINT USING 10, 123	Allows you to print values (number or strings) in a specific format that you can specify.	Lesson 10

R

TERM	EXAMPLE	MEANING	REFERENCE
READ	100 DATA 37, BILL, 8, SAM 200 Read A, B$	Reads data into specified variable from DATA statement. Values are assigned in sequence starting with the first value in first DATA statement.	—
REM	30 REM THIS IS A TEST	Remark statement for adding title, comments or remarks to your programs. Computer ignores words in statement after REM	Lesson 13
RESTORE	100 DATA 37, Bill, 8, Sam 200 READ A, B$ 300 RESTORE 400 READ C, D$	Resets the data pointer so that the next READ statement begins with the first DATA statement in the program. C will have value of 37.	—
RETURN	150 GOSUB 170 160 PRINT A*B 165 STOP 170 C = RND (0) + 5 180 RETURN	When used with GOSUB — RETURN transfers program back to statement following GOSUB instruction	Lesson 16
RND (0)	A = RND (0) *7 A = RND (0) *10 + 1	Creates a random number between .00 and .99	Lesson 6
RUN	RUN	System command that tells computer to find statements it has in stored mode and execute them starting with lowest line number. Also initializes all variables back to zero.	Chapter 3

S

TERM	EXAMPLE	MEANING	REFERENCE
SHAPE	10 SHAPE = 15	Sets the shape to be drawn on screen in either a PLOT, HLIN or VLIN statement. There are 16 shapes available.	Lesson 7
SPC(EXPR)	PRINT SPC (10); "YOU WIN"	Space function that is used to insert a specified number of spaces (blank positions)	—
STEP	For X = 1 to 20 STEP 2	Used to specify the increment to be added to or subtracted from the current variable value in a FOR — NEXT Loop.	Lesson 3
STOP	50 IF A 10 GOTO 70 60 IF A 10 GOTO 80 70 STOP 8C PRINT A/B	Causes program to stop running immediately and return control to immediate mode.	Lesson 1
STORED MODE	40 INPUT "B", B 50 PRINT A*B	Statements which the computer stores for usage at a later time instead of taking immediate action and executing them at once.	Chapter 3

TERM	EXAMPLE	MEANING	REFERENCE
STRING CONCATENATION	DIM FIRST$(7), DIM LAST$(7)	Joining of two strings into one.	Lesson 8
SUBSCRIPT	DIM X(10) DIM Y(A)	Distinguishes each member of ARRAY. Must always be numbers or numeric value. Can be referenced by variable with numeric value.	Lesson 14
SYMBOLS	COST = A * B	Short cut ways to designate variables	Chapter 4
SGN	SGN(– 6) = – 1	Return with arithmetic sign of expression. Returns – 1 if EXPR is negative, 0 if EXPR is 0, + 1 if EXPR is positive.	

T

TERM	EXAMPLE	MEANING	REFERENCE
TAB	10 PRINT TAB (5); "HERE" 20 PRINT "WHY"; TAB(2) "DID THIS HAPPEN"	Moves printing to the right on screen, like a typewriter tab function. TAB ignored if cursor has already passed number of TAB spaces requested.	Lesson 16
THEN	IF K = 1 THEN GOTO 20	Second part of IF — THEN statement that will only be executed if conditions in IF portion are true. Otherwise, statement following THEN is ignored.	Lesson 2
TO	FOR J = 1 TO 50	Establishes the END value in the FOR — NEXT Loop	Lesson 3

V

TERM	EXAMPLE	MEANING	REFERENCE
VARIABLES	NAME$ = "BILL" (string variable) PRICE = 123.45 (number variable)	Symbols used to represent values. Two types are numeric variables (such as PRICE) and string variables (such as NAME$)	Chapter 4 & Lesson 8
VLIN X1, X2, Y	SHAPE = 15: COLOR = 4 VLIN 0, 15, 20	Draws a vertical line or shape at a specified row on screen from column X1 to column X2 in Row Y.	Lesson 18

APPENDIX A

ERROR MESSAGES

The following is a list of error messages. If they occur during a program statement, the statement number as well as the message is displayed.

ARITHMETIC OVERFLOW

The result of a computation is greater than 999999999.9999. This problem can be eliminated by using the IF statement to test and limit the sizes of the variables in the equation before doing the computation.

DIMENSION

Something is wrong in Dimension Statement (size is zero or greater than 99, etc.).

DIVISION BY ZERO

The result of dividing by zero is undefined. The computer cannot proceed. This problem can be prevented by checking the value of a diviser with an IF statement before doing the division.

EXPRESSION

The expression is not properly formed. Examples are PRINT PEEK 123 — parenthesis was not used around expression of a function. PRINT 234 + 3) — missing an open parenthesis. PRINT 1←2 — improper operator symbol in an expression.

EXPRESSION MISSING

An expression is expected to be located in a statement but is not found.

Example: A = (Return Key)
       For H = (Return Key)

EXPRESSION TOO LONG

You did nothing wrong. All computer languages have limits, and you just ran up against one of ours. This will occur if an expression has more than 8 nested brackets or parenthesis.

FOR — NEXT

A NEXT statement was executed when no FOR statement had been executed. Check your program. You have forgotten to supply the FOR statement or the logic of your program is incorrect. Somehow you branched to a NEXT statement without first going through its FOR statement. Fix the logic of your program before continuing.

Example: **10 NEXT J**
   RUN

IF — THEN

The comparison in the IF statement is wrong. For example, IF A*B THEN 100. Fix the statement before running the program.

ILLEGAL MASK SIZE

The format defined for a PRINT USING statement is incorrect. For example, there may be more than 8 "#"'s to the left of the decimal place, or more than 4 to the right. Correct the PRINT USING format, called a mask, before running the program.

ILLEGAL VARIABLE

This is a catchall error message when the interpretor finds something wrong with a variable. Some examples are:

A variable name starts with an illegal symbol — i.e., a number (1A = 3). You have used more than 26 variable names. You used a dimensioned variable name that was not dimensioned.

MEMORY FULL

You ran out of memory. This will usually occur in a DIM Statement. If so, then reduce the allocation in a DIM statement or shorten your program.

Memory storage is as follows:

1. A numeric variable takes 7 Bytes. A dimensioned numeric array takes 7 Bytes for each element.

   Example: DIM H (365) takes 7*365 Bytes.

2. Each line takes 2 Bytes for a line # plus 1 Byte for an end of line symbol.
3. All keywords (i.e., — PRINT, FOR, etc.) take 1 Byte.
4. Actual allowable user storage is 7166 Bytes.

A second reason for Memory Full can be due to your program's continuous execution of a FOR statement with no NEXT statement, or continuous executing a GOSUB with a RETURN.

NO — GOSUB

A RETURN statement was executed without a GOSUB having been executed. You have a logic error in your program. Make sure you do not use a GOTO or fall into a Subroutine. Fix the problem before running the program.

NO LINE # — REFERENCED

You are trying to GOTO a line number which doesn't exist in your program. Correct the statement before running your program again.

PRINT DELIMETER

There is an error in your PRINT statement. Check to see that each of the items being printed are separated by a comma or semi-colon. Correct the statement before running your program.

QUOTE MISSING

The right quote in a String Constant is missing. Correct the statement before running the program again.

READ — DATA

Either you have executed a READ statement when there is
no DATA statement, or there is an error in the DATA
statement. If there is an error in the DATA statement, correct
it before running the program again.

If you are executing a READ statement, then either you
forgot to include a DATA statement or you have tried to
READ more variables than you have data in DATA
statements. Make sure you have supplied all of the data in
the DATA statements, or that you haven't executed a READ
statement more times than you had intended.

WHAT

This is a catch-all. It usually means that one of the items in
the line should have been a keyword but wasn't recognized
as such. Carefully check the line and correct all errors
before running the program again.

> 999999999.9999

You have a constant which has more than four decimal
places or which is greater than 999999999.9999 or less than
−999999999.9999. You will have to change your program so.
that numbers outside of this range are unnecessary.

APPENDIX B
BASIC KEYWORDS

The following words can not be used as or contained
variable names

ABS	MUSIC
ASC	NEXT
CALL	ON
CHR$	OPEN
CLOAD	PEEK
CLOSE	PLOT
COLOR	POKE
CSAVE	PRINT
DATA	READ
DIM	REM
DIR	RESTORE
EDIT	RETURN
END	RND
FOR	RUN
GOSUB	SAVE
GOTO	SGN
HLIN	SHAPE
IF	SPC
INIT	STEP
INPUT	STOP
INT	TAB
KEY$	THEN
LEN	TO
LET	USING
LIST	VLIN

APPENDIX C

TROUBLESHOOTING GUIDE

We hope that we've made BASIC TUTOR easy for you to understand and virtually trouble-free to operate. However, we do realize that many people will use BASIC TUTOR not only as their introduction to programming, but to computers as well. So we've included this quick reference section on troubleshooting in case you run into some problems.

Problems in Loading Programs or Lessons from Tape

1. Check that your Imagination Machine is properly hooked up (see Owners' Manual). After power up, press Reset on the MP1000 console and the "EN" Key on either controller. You should have a clear screen with a cursor in the upper left hand corner.
2. Within the first 15 seconds of loading any program lesson, you will see the "Front Screen" appear on the T.V. It should say Basic Tutor, with the lesson # and program title. IF ANYTHING ELSE APPEARS (AN ORANGE SCREEN, MIXED LETTER PATTERNS), YOU HAVE PRESSED THE RETURN KEY AT THE WRONG TIME. Reset the system and try again. Make sure you listen to the audio from the tape system, and when you hear the "beep", press the Return Key.
3. IF YOU PRESSED THE RETURN KEY AT THE BEEP AND THE FRONT SCREEN STILL LOADS WRONG, it is possible the audio track is slightly off synchronization. Try loading again and press Return a little before or after the beep.
4. IF THE FRONT SCREEN LOADS BUT IT DOESN'T SAY "OK" WITHIN 60 SECONDS you have a bad tape load. Try again. If this repeatedly occurs, make sure the tape is in the tape lid correctly and the lid is shut properly.

Problems with Lesson 1-12

Lessons 1-12 are very unique programs. Even if you have experience with computers you can have problems. They are designed to operate under the "Basic Tutor Mode" which makes the Imagination Machine operate differently.

1. When a lesson (1-12) is first loaded from tape and "OK" appears, you are in the "Regular Mode." When you type Run and press the Return Key, you enter into the special "Basic Tutor Mode." You can always easily check which mode you are in by pressing the Return Key and the "Rept" Key simultaneously. The screen will scroll rapidly if in Regular Mode but will say "This line is not in the program" if you are in Basic Tutor Mode.

2. If you are in Regular Mode, type Run and press the Return Key to go into Basic Tutor Mode. If this does not work, you have to reload the lesson from tape and then type Run.

3. THE LIST COMMAND in regular mode: The command list can be followed by statement numbers and you can list several lines (See Chapter 3, Page 00). In Basic Tutor Mode the list command will always list all statements (and only those) you have entered for the lesson.

4. If you type Run and the lesson does not run as the manual says, you are probably in Regular Mode. The Run Command brought you into Basic Tutor Mode. Try a Run Command again. If you keyed in the lesson, it should run. If not, you will receive instructions on the screen. (A lesson will not run until you have entered all statements or you use the auto entry mode.)

5. "ESC" Key — If you press ESC and the Return Key to do auto entry of statements and the screen says WHAT, it is because you are the Regular Mode. Type Run to get into the Basic Tutor Mode, then try ESC.

6. Dictionary Mode — If you use the ? to get to the dictionary operation and the computer says "What", you are in the Regular Mode. Type Run first to get to the Basic Tutor Mode.

7. Break Key — Pressing the Break Key will usually bring you from the Basic Tutor Mode to the Regular Mode. If you still want to work with that lesson, type Run to return to the Basic Tutor Mode. If pressing "Break" does not bring you to regular mode, try pressing the Return Key first and then immediately pressing Break.

CAUTION: If you repeatedly press Break and the Return Keys, you will be putting BASIC TUTOR in yet another mode. We do not suggest that you do this, since typing LIST after this would cause your computer to list all the steps that we have entered for the special input monitoring mode, etc. If this does occur, however, press Break again to stop the listing, type Run and press Return Key to get back to BASIC TUTOR mode.

8. The CLOAD Command

If you are in the Basic Tutor Mode, typing CLOAD puts the message to the screen.

"PRESS PLAY, THEN RETURN KEY."

If you are in Regular Mode, typing CLOAD puts the message

"REWIND TAPE, PRESS PLAY, THEN RETURN KEY."

In either situation it is not necessary to rewind the tape, but instead, make sure you have advanced the tape to the beginning of a lesson and wait for the "beep" to tell you to press the Return Key.

9. RESET — Pressing the Reset Button is like turning power off/on. If you press reset, you have to reload a lesson from tape.

Problems with Lessons 13-22

1. Lessons 13-22 do not have a special Basic Tutor Mode.
2. To List a Lesson, you must be in Regular Mode. Immediately after a lesson is loaded you are in Regular Mode. If you are running the program, pressing Break returns you to a mode where you can list.

3. Keying in Statements

   If you wish to key in a lesson 13-22 from the listing in the book, it is not necessary to load the lesson from tape. When you load the tape, it does the "Keying in" for you.

4. CLOAD — When you type CLOAD for any lesson 13-22, the message "REWIND TAPE, PRESS PLAY, THEN RETURN KEY" appears. Do not rewind the tape but advance it to the beginning of a desired lesson and press Return when the audio beep occurs.