VTECH®

Pre Computer POWER PAD™

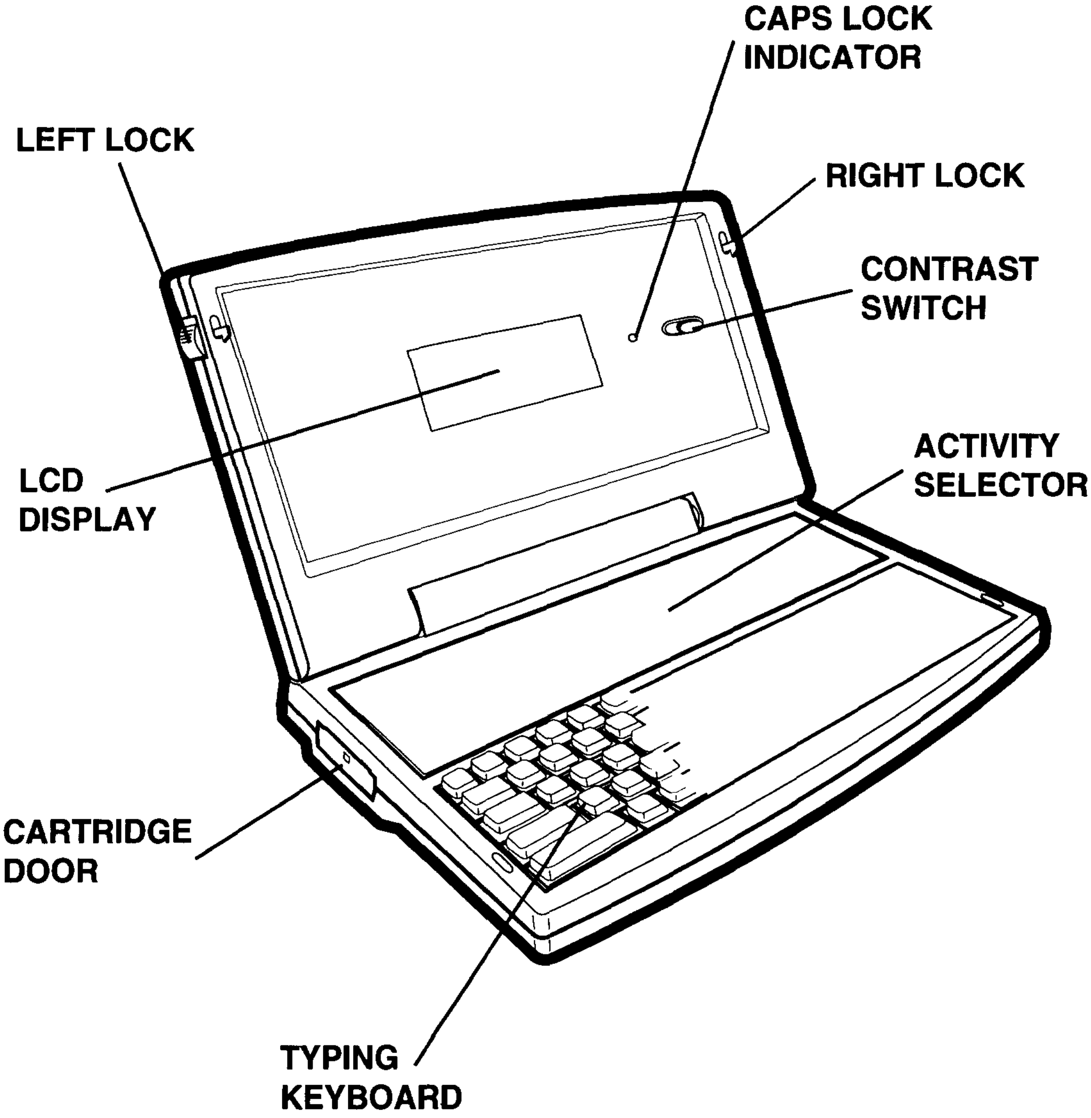
COURSE BOOK

CONTENTS

CHAPTER 1 INTRODUCTION

Thank you for buying the **PRECOMPUTER POWER PAD™** learning toy by VTECH®! We at VTECH® are committed to providing the best possible products to entertain and educate your child. Should you have any questions or concerns, please call our Consumer Services Department at (800)521-2010.

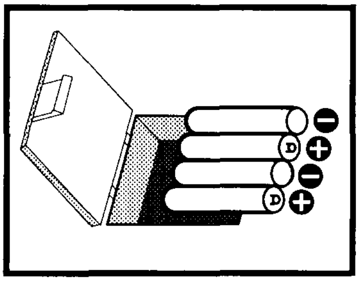
This top of the line learning toy features a deluxe 4 line LCD screen and a standard sized computer style keyboard that introduces the user to the look and feel of a real laptop computer. 35 activities teach spelling, vocabulary, tenses, comparatives, superlatives, synonyms, antonyms, grammar, basic math and advanced math. 1000 trivia questions, BASIC programming, built-in calculator and an 80,000 word spell check round out the curriculum. **PRECOMPUTER POWER PAD™** is the learning toy that has it all!



CHAPTER 2 POWER SOURCE

PRECOMPUTER POWER PAD™ operates on 4 “AA” size batteries or a 9V ⎓ 300 mA +-◐-- AC adaptor.

BATTERY INSTALLATION:



1) Make sure the unit is turned **OFF.**
2) Locate the battery cover on the bottom of the unit and open it.
3) Insert 4 “AA” (UM3 or LR6) batteries as illustrated in the diagram. **DO NOT USE RECHARGEABLE BATTERIES.**
4) Replace the battery cover.

NOTE: We recommend the use of brand-new alkaline batteries for best results.

AC ADAPTOR CONNECTION:

Use a standard 9 volt ⎓ 300 mA +-◐-- AC adaptor.

1) Make sure the unit is turned **OFF.**
2) Locate the adaptor jack on the side of the unit and plug the adaptor in.
3) Plug the adaptor into a wall outlet.
4) Turn the unit **ON.**

NOTE: Do not leave the adaptor plugged in for long periods of time if the unit is not in use.

In the U.S.A.: Many retailers carry this type of adaptor. However, if you are unable to find one locally, the VTECH® AC Adaptor is perfectly suited to operate your **PRECOMPUTER POWER PAD™**. To order just send $10.00 plus $5.00 for shipping and handling (Illinois residents please add 8% sales tax) to:

Adaptor c/o
VTECH INDUSTRIES, INC.
380 West Palatine Road
Wheeling, IL 60090-5831

DISPLAY CONTRAST SWITCH:

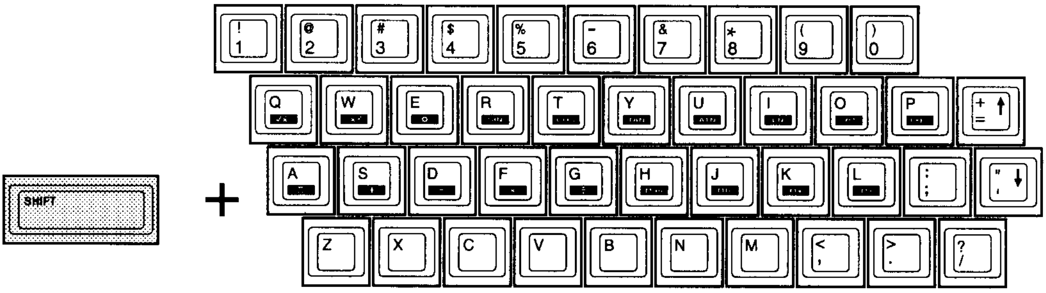
PRECOMPUTER POWER PAD™ has a high/low contrast switch to adjust for different lighting conditions. Slide the switch to the left or right to adjust the screen's contrast.

AUTOMATIC SHUT OFF:

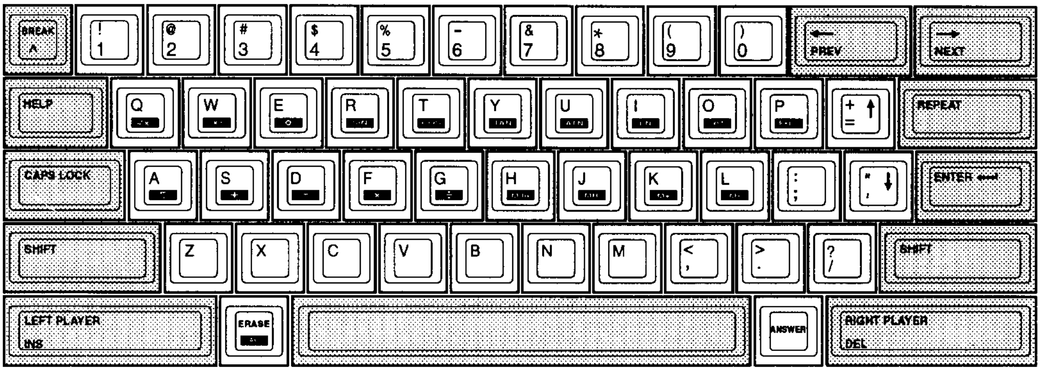
PRECOMPUTER POWER PAD™ will turn itself **OFF** automatically if there is no input for approximately 15 minutes.

CHAPTER 3 THE KEYBOARD:

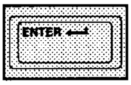
As you can see on the diagram below, the keys on this keyboard resemble and function like the keys on a typewriter or computer. The letters and numbers are in the same positions and capital letters and the symbols above the numbers are made by holding the **SHIFT** key down while pressing the appropriate key.



The keyboard on **PRECOMPUTER POWER PAD™** combines the operation of a standard computer keyboard with several other functions described below.

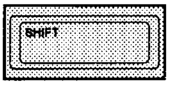


ENTER KEY:



Press this key after you have typed in an answer to a question. **PRECOMPUTER POWER PAD™** will then determine whether your answer is correct or not.

SHIFT KEY:

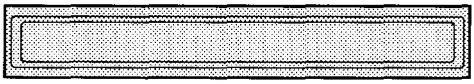


Press this key at the same time as any letter, number or symbol key to get a capital letter or a symbol that appears on the top half of keys with two items printed on them. For example: if you press the **SHIFT** key and the number 4 key, the $ symbol will appear on the screen.

CAPS LOCK KEY:

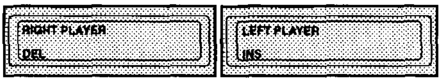
Press this key to generate a continuous series of capital letters without having to press the **SHIFT** key each time. Press once to activate this key. Press it again to release it.

SPACE BAR:



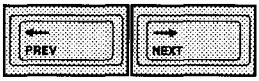
Press this key to insert one space between words or numbers in games or activities where you must type sentences or phrases. Each time you press the Space Bar, another space will be added.

RIGHT PLAYER/ LEFT PLAYER KEYS:



Pressing these two keys on the lower right and left corners of the keyboard determine which player will answer when operating in the two player mode. In activities with the two player mode, the left player's name and the right player's name will flash alternately. The first player to press his or her appropriate **LEFT** or **RIGHT PLAYER** button will get the opportunity to answer the question first.

CURSOR KEYS:

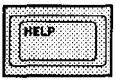


Press the NEXT cursor key to move the cursor position to the right and press the PREV cursor key to move the cursor position to the left. The short blinking line on the screen, which is the cursor, indicates where the next letter will appear. On **PRECOMPUTER POWER PAD™** , these cursor keys also allow you to scroll left or right on the visible screen so that you can make corrections to words you've already typed.

Another great function of these keys is that they can control the scrolling speed of questions in the **TRIVIA** and **STORY PROBLEMS** activities. Continuously pressing the NEXT key makes the scrolling faster and continuously pressing the PREV key makes the scrolling slower.

NOTE: These keys must be pressed in succession in order to control the scrolling speed properly. If you just press the key once, it will not work. You must continuously press the key at least 5-7 times until you visibly see the question speeding up or slowing down.

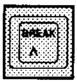
HELP KEY:



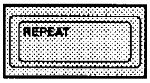
Press the **HELP** key in the **TRIVIA** activities and **STORY PROBLEMS** when you need help with a question. Three multiple choice answers will appear for you to choose from.

In "WHAT AM I?" press **HELP** key will display the word's clue.

NOTE: The **HELP** key does not function in the two-player mode.

BREAK KEY: 

Press the **SHIFT** and **BREAK** keys when you want to interrupt a program in **BASIC** while it is running.

REPEAT KEY: 

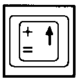
Press this key when you wish to repeat a question in the Trivia questions and **STORY PROBLEMS.**

ANSWER KEY:

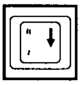
Press this key to find the correct answer to a question.

ERASE/AC KEY:

Press this key to change an entry prior to pressing the **ENTER** key. The AC key is used to cancel all numbers in the calculator mode.

UP CURSOR 

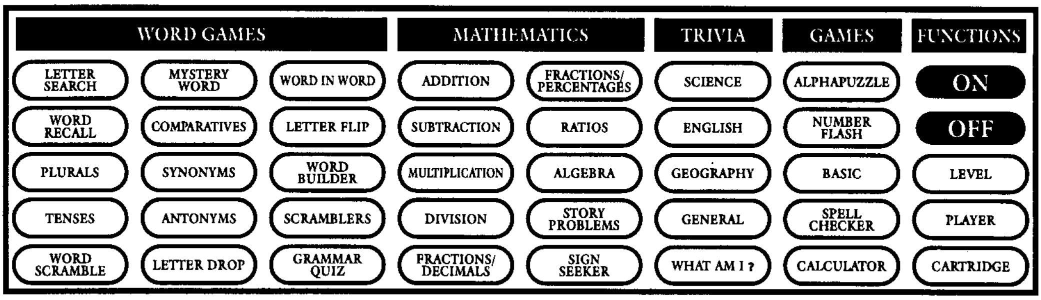
Use this key to move your cursor to the previous line of text or to move letters up vertically in the **ALPHAPUZZLE**. The "+/=" symbols can be used in **PRECOMPUTER POWER PAD**$^{\text{TM}}$'s **CALCULATOR** and **BASIC** function.

DOWN CURSOR 

Use this key to move your cursor to the next line of text or to move letters down vertically in the **ALPHAPUZZLE.** The " " / ' " symbol can be used in **PRECOMPUTER POWER PAD**$^{\text{TM}}$'s **BASIC** function.

THE ACTIVITY SELECTOR:

The activity selector is a membrane keyboard which allows you to access the functions and activities easily with just the touch of a finger.



WORD GAMES			MATHEMATICS		TRIVIA	GAMES	FUNCTIONS
LETTER SEARCH	MYSTERY WORD	WORD IN WORD	ADDITION	FRACTIONS/ PERCENTAGES	SCIENCE	ALPHAPUZZLE	ON
WORD RECALL	COMPARATIVES	LETTER FLIP	SUBTRACTION	RATIOS	ENGLISH	NUMBER FLASH	OFF
PLURALS	SYNONYMS	WORD BUILDER	MULTIPLICATION	ALGEBRA	GEOGRAPHY	BASIC	LEVEL
TENSES	ANTONYMS	SCRAMBLERS	DIVISION	STORY PROBLEMS	GENERAL	SPELL CHECKER	PLAYER
WORD SCRAMBLE	LETTER DROP	GRAMMAR QUIZ	FRACTIONS/ DECIMALS	SIGN SEEKER	WHAT AM I ?	CALCULATOR	CARTRIDGE

ON KEY:

Press the **ON** key to turn the unit on.

OFF KEY:

Press the **OFF** key to turn the unit off.

LEVEL KEY:

PRECOMPUTER POWER PAD™ provides 4 levels of challenge in most of the activities. When you first enter an activity, it will automatically begin in Level 1. If you would like to change the level while in the middle of an activity, simply press the **LEVEL** key and the number of the level that you would like to change to.

PLAYER KEY:

Press the **PLAYER** key to change from the one player mode to the two player mode and vice versa. If you press the **PLAYER** key during a game it will reset and begin again.

BASIC KEY:

This key enables you to access the **BASIC** program within the **PRECOMPUTER POWER PAD™**.

SPELL CHECKER KEY:

Press the **SPELL CHECKER** key to check the spelling of a word. Type in the word and **PRECOMPUTER POWER PAD™** will verify if it is spelled correctly or not. If the word is misspelled, use the PREV and NEXT arrow keys to "scroll" through the list of possible choices for the correct word.

CALCULATOR KEY:

Press this key to transform the unit into a functional calculator.

CARTRIDGE KEY:

Press this key after you have inserted a **PRECOMPUTER POWER PAD™** compatible cartridge to make use of the activities on that cartridge.

(NOTE: CARTRIDGES ARE SOLD SEPARATELY.)

THE ACTIVITY KEYS:

There are 32 learning games in **PRECOMPUTER POWER PAD**™. They can be accessed with just a touch of one of the 32 touch sensitive keys on the Activity Selector.

HOW TO PLAY:

To begin an activity or game, first press the **ON** key to turn the unit on. Choose an activity and press its key on **PRECOMPUTER POWER PAD**™'s Activity Selector. Activities automatically begin in Level 1 in the one player mode.

TWO PLAYER MODE:

To enter the two player mode, press the **PLAYER** key then the number 2. Type in the name of the person who will operate the **RIGHT PLAYER** key and press **ENTER.** Follow the same procedure for the **LEFT PLAYER** key. In the two player mode, the right and left player's names will flash back and forth on the screen. The player who presses his/her player button first will have the chance to answer the question.

The level of play and player mode will remain the same from one activity to another unless you change the setting.

NOTE: The following games and functions do not offer a 2 Player Mode:

WHAT AM I?

ALPHAPUZZLE

NUMBER FLASH

BASIC

SPELL CHECKER

CALCULATOR

SCORING:

In the one player mode for most activities there will be five questions in one round. You will receive 20 points for a correct answer on the first try, 15 points on the second try and 10 points on the third try. 5 points will be deducted for each incorrect answer.

THE DISPLAY:

PRECOMPUTER POWER PAD™ features a large dot matrix liquid crystal display (LCD). It can display 4 lines X 20 characters on screen.

CHAPTER 4 ACTIVITIES AND GAMES

WORD GAMES

LETTER SEARCH

Insert the missing letter to complete the word. A word will appear on the screen with one letter missing. Type in the letter that will complete the word and press **ENTER.**

For each question you will get three chances to answer correctly. If you enter a wrong answer, your last input will be shown on the screen for your reference.

WORD RECALL

Learn the spelling of new words as they appear on the screen. A word will move fast from right to left on the bottom row of the screen then disappear. Type the word on the bottom row and press the **ENTER** key.

For each question you will get three chances to answer correctly. If you enter a wrong answer, your last input will be shown on the screen for your reference.

PLURALS

A singular form of a noun will appear on the viewing screen. Type in the plural form of that word and press the **ENTER** key.

TENSES

Using your knowledge of verb forms, provide the present tense, past tense, or past participle. Two forms of a verb will appear on the screen. Type in the missing form of that verb and press the **ENTER** key.

WORD SCRAMBLE

A scrambled word will appear on the viewing screen. Rearrange the letters, type the correct word on the bottom row and press the **ENTER** key.

MYSTERY WORD

Guess the word that **PRECOMPUTER POWER PAD**™ is thinking of! Keep choosing letters until you discover the mystery word. But choose carefully because you only have ten chances to guess the mystery word.

The series of dashes that appear on the bottom row of the screen represent the number of letters in the word. The # symbols on the second row represent the number of chances you have remaining to guess the correct letters. Each incorrect letter you guess will appear on the third row. Each letter you guess correctly will appear in its appropriate place in the word on the bottom row.

COMPARATIVES

Using your knowledge of adjectives, provide the comparative or the superlative adjective of the words displayed on screen. Two adjective forms will appear on the screen with either the comparative or superlative form missing. Type in the correct answer and press the **ENTER** key.

SYNONYMS

Provide a word which has a similar meaning to **PRECOMPUTER POWER PAD**$^{\text{TM}}$'s reference word. A word will appear on the third row of the viewing screen. Type in a word which has the same meaning and press the **ENTER** key. The number of letters in the synonym that **PRECOMPUTER POWER PAD**$^{\text{TM}}$ wants you to enter is indicated by the short dashes on the bottom row of the viewing screen. One letter in the correct word is given as a clue.

NOTE: These clues are provided because, although there is more than one correct answer to this kind of question, only one answer could be programmed into **PRECOMPUTER POWER PAD**$^{\text{TM}}$.

ANTONYMS

Provide a word that has the opposite meaning of **PRECOMPUTER POWER PAD**$^{\text{TM}}$'s reference word. A word will appear on the third row of the viewing screen. Type in a word that has the opposite meaning and press the **ENTER** key. The number of letters of the correct antonym is indicated by the short dashes on the bottom row of the viewing screen.

NOTE: These clues are provided because, although there is more than one correct answer to this kind of question, only one answer could be programmed into **PRECOMPUTER POWER PAD**$^{\text{TM}}$.

LETTER DROP

Insert the given letter into its correct position in the word. An incomplete word will appear on the bottom row with its missing letter above it. Use the PREV or NEXT cursor keys to move the letter left or right. When the letter is located above the correct position, press the **ENTER** key.

WORD IN WORD

Take out a letter from the given word to form a new word. A word will appear on the screen. Move the arrow left or right by using the PREV or NEXT cursor keys until it is above the letter that you would like to take out and press **ENTER**.

LETTER FLIP

"Flip" two letters to make a pair of words correct. Two words with the same number of letters will appear on the viewing screen with one letter from each word flipped. To make both words correct, highlight the incorrect pair of letters using the cursor keys and press the **ENTER** key to "flip" them to their correct position.

WORD BUILDER

An incomplete word will appear on the screen. Groups of letters will start to scroll above the incomplete word. When the group of letters that will correctly complete the word is directly above the space in that word, press the **ENTER** key.

SCRAMBLERS

Rearrange the words to form a correct sentence. A group of words in random order will appear on the viewing screen. Find the first word of the sentence, move the cursor to the first letter of that word, and press the **ENTER** key. Do the same for the second word, third word and so on. Each word, in the order that you choose them, will appear on the bottom half of the viewing screen.

GRAMMAR QUIZ

Find the incorrect word and change it to form a grammatically correct sentence. A sentence will appear on the screen with one word containing a grammatical error. Find the incorrect word by pressing the PREV and NEXT key until it stops at the incorrect one, then press the **ENTER** key. If you are right, 3 words will appear on the screen. Choose the word that would make the sentence grammatically correct and press its corresponding number.

MATHEMATICS

ADDITION

An addition problem will appear on the viewing screen. Type in the correct answer using the number keys and press **ENTER.**

SUBTRACTION

A subtraction problem will appear on the viewing screen. Type in the correct answer using the number keys and press **ENTER.**

MULTIPLICATION

A multiplication problem will appear on the viewing screen. Type in the correct answer using the number keys and press **ENTER.**

DIVISION

A division problem will appear on the viewing screen. Type in the correct answer using the number keys and press **ENTER.**

FRACTIONS/DECIMALS

A fraction will appear on the viewing screen. Convert that fraction into a decimal or vice versa, type in the answer and press **ENTER.**

FRACTIONS/PERCENTAGES

A fraction will appear on the viewing screen. Convert that fraction into a percentage or vice versa, type in the answer and press **ENTER.**

RATIOS

One complete fraction and one incomplete fraction will appear on the screen. Type in the missing numerator or denominator to make both fractions equal and press **ENTER.**

ALGEBRA

An algebraic equation will appear on the screen. Type in the answer using the number keys and press **ENTER.**

STORY PROBLEMS

In this activity **PRECOMPUTER POWER PAD**™ challenges you to find answers to math problems that are presented as story problems instead of numerical equations. Read each question that appears on the viewing screen and type in the answer from the keyboard. Then press the **ENTER** key.

SIGN SEEKER

Find the appropriate sign from the horizontal scrolling list to correctly complete the mathematical equation. A mathematical equation with one symbol missing will appear on the bottom row of the screen with a continuing series of > = symbols in Level 1, < > = symbols in Level 2 and + - x ÷ symbols in Levels 3 and 4 scrolling horizontally above. When the proper symbol reaches the position to correctly complete the equation on the bottom, press the **ENTER** key and the symbol will fall into place.

TRIVIA

The trivia categories in **PRECOMPUTER POWER PAD™** include **SCIENCE, ENGLISH, GEOGRAPHY,** and a **GENERAL** category. There is also a crossword puzzle trivia game called **WHAT AM I?**

Select the **TRIVIA** category you want by pressing the corresponding key. Read each question as it scrolls across the viewing screen, type in the correct answer and press the **ENTER** key.

Press the **HELP** key to select from three multiple choice answers if you need help answering a question. 5 points will be deducted when you use the **HELP** function.

WHAT AM I?

This is a fun crossword puzzle type game. A crossword puzzle will appear on the screen. Press the PREV or NEXT cursor keys to go to the word you want to guess and press **ENTER** to reveal this word's clue. When you have finished reading the clue, press **ENTER** again to go back to the puzzle screen, type in the correct answer and press **ENTER**. There are 10 words for each puzzle.

GAMES

ALPHAPUZZLE

Use the cursor keys to put a series of scrambled letters into alphabetical order. Depending on the level you choose, two to four rows of scrambled letters will appear on the screen with one open space to allow for letter position shifting. Use the PREV and NEXT cursor keys as well as the ↑ and ↓ vertical cursor keys to shift letters back and forth or up and down to put the entire series of letters into proper alphabetical order. No score is given for **ALPHAPUZZLE**.

NUMBER FLASH

Test your memory skills as an increasing series of numbers appear on the screen. One single digit number will appear briefly on the far left side of the third row before disappearing. Key in that number key. Then the first number and a second single digit number will appear briefly on the screen before disappearing. Type in those two numbers. Continue this process until you make a mistake or until the round is over.Your score in **NUMBER FLASH** will be based on the total number of digits you correctly remember and enter.

CHAPTER 5 BASIC

INTRODUCTION

ARE COMPUTERS IMPORTANT?

Computers are all around us. Like cars, phones and TVs, computers help us make our lives easier. Millions of computers are used every day. Doctors use computers to treat patients. Police use computers to track criminals. Stores use computers to add up prices quickly and so on.

WHAT ARE COMPUTERS?

A computer is a tool that helps us to do things better and faster. They work without getting tired. They operate at very high speed. They follow instructions perfectly. And they never forget, like we sometimes do.

The electronic computer has been around now since the late 1940's. The first ones were huge electronic mazes of wires and tubes. They were so expensive so that only big businesses and governments could afford to buy them and keep them running. The early computers required large amounts of electrical energy to run them and generated vast quantities of heat. More importantly they required a specialized team of experts called programmers to tell them what to do.

This is a far cry from how we see a computer nowadays. We see them mostly as watches, calculators, microwave ovens, VCR's, phones, TV's, CD's, and personal computer workstations. This is a little different from the computers used in businesses which are larger and require more space to operate.

People today generally work with a very small version of a computer called a microcomputer. A microcomputer is a computing device where the central processor and memory (RAM) are contained on a very small integrated circuit "chip" called a microprocessor. The central processor is the device that performs arithmetic and makes decisions. The memory is where information in the form of numbers and letters is stored. A microcomputer therefore contains both a microprocessor and some ways to communicate with a person.

All microcomputers have a keyboard for you to type in your instructions. This is called "input". Some have individual keyboards, like a typewriter; others have plastic membranes, like microwave ovens.

All microcomputers have a device which is used to communicate back to a person. This is called "output". Some microcomputers use your TV set, while others require a special type of display called a "monitor". Some use paper in a device called a "printer" and some use a liquid crystal display (LCD) which is the same display that is used on some time pieces.

WHAT IS BASIC?

BASIC is a computer language. The name **BASIC** stands for Beginner's All-purpose Symbolic Instruction Code. It was developed by professors at Dartmouth College in the mid-1960s. It was designed for students who had no previous experience in programming computers.

BASIC is basic; however there are differences in the language based on the computer that uses it in much the same way that English is different depending on which country it is spoken in (Great Britain or the United States) or which part of the country the language is spoken in (East Coast, Midwest, South or the West Coast).

The differences in the language will depend on the computer that is being used and on the computer that produced the language for the computer manufacturer. Microsoft **BASIC**, so-called because it was developed by the Microsoft Corporation of Belleview, Washington, is found on Apple II+, IBM PC and PC jr., Laser Computers, Commodore VIC20 and Commodore 64 and others. Other computers such as the Timex Sinclair ZX80 and the Texas Instruments 99/4 use versions of **BASIC** similar to **MS-BASIC**.

WHAT IS A PROGRAM?

A computer program is a series of instructions that tells a computer what tasks you want it to perform. Computer programs are written in a programming language. **BASIC** is one of the languages that a human and a computer understand and so **BASIC** can be used to write a program that a computer can carry out for you.

WHY SHOULD I LEARN TO USE A COMPUTER?

The future belongs to the people who understand how to use computers. Computers can do many things if they are given instructions. For a particular job to be done, by a person or a machine, we must specify the steps which must be done to do the job. The computer becomes an extension of our capabilities. To do this requires us to learn how to talk to our computer.

This chapter will help you to get started. All you need is your **PRECOMPUTER POWER PAD**™ and some time.

WHAT IS THIS CHAPTER FOR?

This chapter is a step-by-step guide for understanding programs and learning to use the computer language called **BASIC**.

WHO IS THIS CHAPTER FOR?

This guide is written for students and adults who haven't used a computer before. With a little time and effort you will discover that there is nothing difficult about learning to talk to a computer.

WHAT IF I MAKE A MISTAKE?

Don't worry about making a mistake. The more mistakes you make the more you will learn. This is called discovery! The computer doesn't care how many mistakes you make. There is nothing you can do to damage the computer – except smashing it. Experimenting is a very good way to make your programs do exactly what you want.

THE KEYBOARD

1. The numeric (top row) keys are used for entering numbers. The letter keys are used for entering letters.
2. The cursor is a special symbol on the display that indicates where the next character that is pressed on the keyboard will be entered in the computer. The character used on the **PRECOMPUTER POWER PAD**™ is the blinking block symbol, " █ ".
3. The **SHIFT** key is used with another key to enter a character into the computer. For example to enter the "+" character, you must press **SHIFT** and the key marked "+ =". If you just press the "+ =" key then the "=" character will be entered.
4. The PREV key and the NEXT key are used to position the cursor left or right, respectively. They do not erase any character that is displayed, but you can type over anything in their positions.

5. To enter a space, simply press the **SPACE** bar. Each time you press the **SPACE** bar, one space is inserted on the display line.

6. The **INS** and **DEL** keys are used to insert or delete one character at a time on the display where the cursor is positioned. The **INS** key shifts all characters on the display one character to the right to make room for the new character. You can then type in the character in the space provided. Be sure to press the **INS** key and the character together each time you want to insert.

   The **DEL** key is used to delete one character at a time on the display where the cursor is positioned. The **DEL** key will shift all characters on the display one position to the left. Thereby erasing the character under the cursor.

7. Before anything on the display is actually sent, you must press the **ENTER** key. This key acts much like a Return key on a typewriter. You press all the keys that comprise a statement to the **PRECOMPUTER POWER PAD**™ learning toy and then press **ENTER**. **PRECOMPUTER POWER PAD**™ will take it from there.

8. Every time **BASIC** is selected, it is in the overwrite mode. That is, the character you type will appear on the LCD no matter whether there is a character at the cursor position or not.

   When the **INS** key is pressed once, it will be in insert mode. All characters to the right of the cursor position will be shifted one character to the right to make room for the new character once it is input through the keyboard. Pressing the **INS** key again will go to the overwrite mode again.

9. The "^" character on the BREAK ^ key (which is used only in the **BASIC** program) is used to tell the computer you want to raise the number preceding it to a power of the number following it. More on arithmetic later.

   The **BREAK** key is used as a **BREAK** which means it can do one or two things. When you are typing a line, but before you press **ENTER**, it will erase all that you have typed; or, if you are running a **BASIC** program, you can interrupt it by pressing this key. You can restart a program by typing **CONT** and pressing the **ENTER** key.

   When using the **INPUT** command, **PRECOMPUTER POWER PAD**™ will ask you to supply it with some data from the keyboard. **PRECOMPUTER POWER PAD**™ tells you it needs input with a prompt. The prompt is the "?" symbol and it appears on the display indicating that the computer is waiting for you to type something. The letters and numbers that you press appear on the display.

THE DISPLAY

The **PRECOMPUTER POWER PAD**™ learning toy uses a Liquid Crystal Display (LCD). It is 80 characters wide but you can only see 20 characters at a time. The PREV and NEXT are used to scroll the 20 character display window left or right, respectively.

In **BASIC**, you will need to press the **ENTER** key to see the next printed statement that your program is displaying. You need to do this so that the output doesn't scroll off the end of the line.

GETTING STARTED

The activity lets you type in your own programs and run them. Follow these simple steps:

1) Turn on **PRECOMPUTER POWER PAD™.**
2) Choose the **BASIC** function.

There are two modes available; Command and Run.

Type in this program:

```
10 PRINT "HI THERE"              press the ENTER key
```

This is the usual way to write a program. Type in the program lines and remember to press the **ENTER** key. Each line starts with a number and is followed by a statement. **PRECOMPUTER POWER PAD™** stores the line with the other lines into memory. Later you execute the program by typing the command, **RUN**. Don't forget to press the **ENTER** key after typing **RUN**. The program that you have typed into memory will begin. The results appearing on the LCD display will be the message: **HI THERE.**

There is a short cut.

Enter the following (no line number in front):

```
PRINT "HI THERE"
```

This time **PRINT** was used as a command and the results appeared immediately on the display after the **ENTER** key was pressed. The computer executes the command right away, without waiting for you to type **RUN.** When you do this, the statements are not saved for future re-use. They are executed immediately and discarded. This is not recommended for creating programs but highly recommended for use as a calculator.

Here are some other commands that can be used:

NEW

This command clears the memory of any **BASIC** statements that have been previously entered.

LIST

This command displays each line of your program starting with the lowest line number. Each time you press **ENTER**, the next lines are displayed. You can stop this by pressing the **SHIFT** and **BREAK** keys. If you enter the command with a line number after the word, i.e., **LIST** 50, **PRECOMPUTER POWER PAD**™ will list the statement at line 50.

RUN

This command instructs **PRECOMPUTER POWER PAD**™ to begin executing each **BASIC** statement with a line number that was typed into memory. The computer will start with the lowest line number and proceed up to the highest numbered statement.

EDIT

Use the **EDIT** command when you want to change a statement that has been typed into memory without re-typing the entire line. Just type **EDIT** and the line number and press the **ENTER** key. The statement will appear on the display. Use the **LEFT-ARROW** and **RIGHT-ARROW** keys to move the cursor. Use the **DEL** key to remove an unwanted one, or type in a new character.

You can type in line -numbered **BASIC** statements in any order. The **PRECOMPUTER POWER PAD**™ will sort them out for you and **LIST** them or **RUN** starting with the lowest numbered one to the highest numbered one. It is a good idea to number your lines in increments of 10 (10, 20, 30...) or 100 (100, 200, 300...). This way you have room to add more statements if you decide to change your program in the future.

You can insert an entirely new line in a program by using a line number that doesn't exist between two existing ones. You can delete an existing line-numbered statement by merely typing the line number and pressing the **ENTER** key. Of course, you can change an existing line by merely retyping the entire line.

AUTO

When you want to type in the program, you need to type in the line number before the statement. With the **AUTO** function, the line number will generate automatically.

e.g.	**AUTO**	generate the line no. starting from 10 with increment of 10
	AUTO 100	generate the line no. starting from 100 with increment of 10
	AUTO 100, 5	generate the line no. starting from 100 with increment 5

```
      i.e.  100
            105
             "
             "
```

To end the **AUTO** mode, press **ENTER** without input or press **Shift-Break.**

CONT

This command causes the program to resume executing after encountering the **BASIC** command called **STOP**. The program will be carried on with the next statement after **STOP**. You can type **CONT** to resume the program after you hit the **SHIFT** and **BREAK** keys.

BASIC COMPUTER TUTORIAL

1. A REAL SMALL PROGRAM.

You and Taraesa went to the local video store to rent some movies. Taraesa brought back 12 but 4 were too gory for you and so you took them back. On that trip you got carried away and came home with 7 new ones. How many are you going to watch this afternoon? You could work this out yourself but here is a small BASIC program that can do the arithmetic for you.

Type in:

```
10  PRINT  12-4+7          and press ENTER

20  END                    and press ENTER
```

Not much to it. Now type RUN and press **ENTER**. What happens? The answer, 15, appears on the display.

In BASIC, you write a series of line-numbered statements that tell **PRECOMPUTER POWER PAD™** what to do. The first statement, numbered 10, tells the computer to work out the sum of three numbers and then **PRINT** the answer in the display. The next statement, number 20, tells the computer that this is the end of the program and that it can stop **RUNNING.** However, the last statement is not necessary in this program because you only want to print out the sum of 3 numbers, therefore, it could be removed.

Remove that last line by typing 20 and press **ENTER**, then type **RUN** and press **ENTER**. What happens? That's right, you get the same results.

Suppose you typed:

```
10 PRONG 12-4+7                    and press ENTER
```

Now type RUN and press **ENTER**. What happens? You get the strange message, **"? SYNTAX ERROR IN 10"** in the display. That means you made a mistake in **BASIC** grammar and this is **PRECOMPUTER POWER PAD™** 's way of telling you this. You need to **EDIT** the line or re-type it, changing the **"PRONG"** word to **"PRINT"** and then running the program again to get the correct answer.

You can use **PRECOMPUTER POWER PAD™** as a calculator in arithmetic statements like the one above by not using a line number.

Just type:

```
PRINT 12-4+7                    and press ENTER
```

The answer will appear on the display.

Another short cut is that you could use the "?" symbol to stand for the word **"PRINT".** When you are using **BASIC** as a calculator in "command mode", think of the question mark as meaning "What is 12-4+7"? When you are using the question mark as a **"PRINT"** command in a program, the computer will replace the "?" with the word **"PRINT"** for you.

2. LET'S DO SOME ARITHMETIC.

In the previous section you did a sum of three numbers. The numbers like 1, 3, 27, 14.3, etc., are called **CONSTANTS.** The program added and subtracted the constants 12, 4 and 7. The order that you do addition is unimportant: 6+10 is the same as 10+6. In subtraction, the order **is** important: 10-6 is not the same as 6-10. So the order that you write numbers and do arithmetic operations is important. In **BASIC,** operations are from left to right.

The * symbol is used to represent multiplication. Like addition, the order of the numbers is unimportant. Here's an example. There are 2.204 pounds to a kilogram. How many pounds does a 6 kilogram parakeet weigh?

Try this:

```
10 PRINT 6*2.204
```

The symbol / is used for division. Here, like in subtraction, the order of the numbers is important since 15/3 is 5 and 3/15 is .2. How many kilograms does a 6 pound parakeet weigh?

Try this:

```
10 PRINT 6/2.204
```

You can raise a number to a power also. To do this you need the ^ sign. The expression 5^3 means 5*5*5 or 125; similarly, 3^5 means 3*3*3*3*3 or 243. There are fractional powers; for example 2^.5 is the square root of 2 or 1.414.... Here's an example: I bet you 1 doubloon and throw the dice 10 times; each time you double your money. How much have you won?

Type:

```
10 PRINT 2^10
```

When you use all the operations together, it can get a bit complicated. After a winning baseball game, your 7 person team is awarded 3 cans of pop for the game plus another 11 cans because you finished first in the league. How many cans does each player get?

Type this:

```
10 PRINT 3+11/7
```

Is it really evaluated from left to right? Yes, but was it the answer you were expecting? If the answer came out to 4.57143, the computer first divided 11 by 7 getting 1.57143 and then it added 3 to the result giving 4.57143. You probably were expecting the answer to be 2 because 3+11 is 14, then divided by 7 is 2.

You can see that expressions are not simply evaluated from left to right. It is necessary to have an exact set of rules to follow.

Here they are:

Functions	Order in which they are performed
( ) things in parentheses (brackets)	1st
^ raising to a power	2nd
* / multiplication and division	3rd
+ - addition and subtraction	4th

The computer will always look at an expression in parentheses first and do the things you put there. It could be an expression or more sets of brackets. Among the expression, raising to a power is done first, followed by multiplication and division from left to right, followed by addition and subtraction from left to right. If you're not sure how the computer will evaluate the expression, simply put brackets around the expression you want done first. Remember the brackets must always be in pairs, one right bracket for each left one. If you don't, then you will get a "? **SYNTAX ERROR**" message.

3. NUMERIC VARIABLES.

You have seen how you can use BASIC to print numbers and messages. You can also give values to a **BASIC** program yourself. The computer uses a letter to stand for a value. This is called a variable. For example LET D = 609. This statement puts the value 609 into a box in memory called "D".

Try this. Type:

10 LET A=8 *press* ***ENTER***

The computer makes a box in memory and calls it "A". Into this box it puts the number 8.

Now type:

20 LET B=10 *press* ***ENTER***

The computer makes a box called "B" and places 10 into it.

Type:

30 LET B=15 *press* ***ENTER***

Since there is a box called "B" already with a value of 10 in it, it doesn't make another box called "B" with a value of 15 in it. The number 10 is just replaced with the number 15.

Now type:

40 LET C=A+B *press* ***ENTER***

This statement is a bit more complicated. Here's how it works. First, the computer searches for a memory box called "B" and finds in it the number 15. The "+" sign tells the computer to add the numbers found in "A" and "B" together. It does that and the answer is 23. Now, where to put the answer? No problem. The "=" tells the computer to store the answer in a memory box called "C". The computer searches for a box called "C". It doesn't find one so it makes one in memory and then puts the answer into it.

Of course, if there was a memory variable called "C" that had a number in it from a previous operation, the old number would be replaced by the new one by this statement.

Now let's finish this by typing:

```
50 PRINT "FIRST NUMBER"; A

60 PRINT "SECOND NUMBER"; B

70 PRINT "THE SUM IS"; C

RUN
```

You will see the following on the display:

```
FIRST NUMBER 8

SECOND NUMBER 15

THE SUM IS 23
```

(Don't forget to press **ENTER** after you have seen each line to get the next one displayed.)

Let's take a look at the print statements on lines 50 through 70. The words in quotes are called a **STRING** or a **LITERAL.** Whatever you put between a set of quotation marks will appear on the display exactly the way you typed them. Don't forget that quotation marks come in pairs!

Next, the semicolon tells the print statement to print the next thing immediately to the right of the end of the string. This is the number 8 which is the value in memory box A.

Suppose that you asked the computer to multiply 3 million times 3 million.

Let's try. Type:

```
10 PRINT 3000000*3000000

RUN
```

The answer on the display is 9E+12. This is the computer's way of showing extremely large numbers. A number with 12 zeros to the right of the decimal point is called a scientific or exponential notation. You find the decimal point and move the number of places specified after the "E" to the right filling in with zeros. The long way to write out the above answer is 9,000,000,000,000. The procedure works in reverse for very small numbers. The number 9E-12 is .000000000009. This is a very small positive number. The number -9E+12 is a very large negative number while the number -9E-12 is a very small negative number.

4. STRING VARIABLES.

String variables are similar to the numeric variables that we have been working with so far except that the variables contain alphabetic characters (numbers and letters and symbols). The name of the memory variable always contains a $ to distinguish it as a string variable. String variables are not used in arithmetic but allow you to store things like your pet's name. Here are some examples:

VARIABLE NAMES	CONTENTS
A	675
B	2.3434
C$	CURTIS
A$	ALPHABET SOUP

You can store a string variable like this:

```
10 LET A$="ALPHABET SOUP NO 123"
```

You can have a numeric variable called A and a string variable called A$. The $ lets **BASIC** know how to tell them apart.

NOTE: For both numeric and string variables, only the first 2 characters of the variable names are valid. Also, they cannot resemble the name of any command.

Examples: **TOTAL** is interpreted as the **TO** command and **POTATO** is interpreted as variable **PO.**

5. CONSTANTS.

A constant is something that doesn't change. The computer stores it in memory just like a variable. However, since a constant doesn't change it has no variable name. The number 6 is a constant; so is 4.567 or 28967.35. "**QUICK BROWN FOX**" is a string constant.

You can store constants to a variable. That's usually done to start out a program; later the contents of the variables change. Let's see. Suppose we want to count to 10.

Type in this program:

```
New

10 N=0

20 N=N+1

30 PRINT N

40 IF N<10 THEN 20

50 PRINT "THE ANSWER IS"; N

RUN
```

The statement at line 10 stores the numeric constant zero into N, a numeric variable. In statement 20 the variable will have a numeric constant, 1, added to itself and then the sum of that operation will be stored in place of the value previously stored there.

The program in the above example contains a programming construction called a loop. A loop is one or more **BASIC** statements, usually called a set, that can be executed as many times as you want. The statement in line 40 contains an IF statement. It is called a conditional statement and will be discussed in detail under the topic called Making Decisions.

6. INPUTTING DATA

We have been putting information into the computer by typing constants into programs using statements like the following:

```
10 LET N=10
20 LET A$="NANCY LIKES CHOCOLATE CUPCAKES"
```

Another way to supply data is by using an INPUT statement.

Try this out by typing:

```
NEW

10 INPUT "GIVE ME A NUMBER"; N

20 PRINT "YOUR NUMBER IS"; N
```

The computer will print the literal following the **INPUT** command in line 10 on the display. It will prompt you for a number with the character. You can type any number of digits that you like and press **ENTER** to let the computer know you have finished. Ready? Try it, type: **RUN** and press **ENTER**.

You can also input alphabetic data into string variables like this program:

```
10 INPUT "WHAT IS YOUR NAME"; N$

20 PRINT "HI THERE "; N$
```

You can use as many input statements as you need to get all the value into your program.

Try this:

```
NEW

10 INPUT "GIVE ME A NUMBER"; N1

20 INPUT "AND ANOTHER"; N2

30 PRINT "I WILL ADD THEM"

40 C=N1+N2

50 PRINT "THE ANSWER IS"; C
```

Type RUN and press **ENTER** to see what happens.

7. MAKING DECISIONS

The **IF** statement tells the computer that it has to make a decision. It does this by comparing two numbers, or arithmetic expressions or string variables. It uses special symbols to represent a condition.

The symbol > means greater than.

The symbol < means less than.

The symbol = means equal to.

The symbol > = means greater than or equal to.

The symbol < = means smaller than or equal to

Here are some conditions:

```
5 < 10    20 > 10
```

They are both true because

5 is less than 10

20 is greater than 10

A condition is either **TRUE** or **FALSE**. The IF statement has a condition. The computer evaluates the condition and decides that if a condition is **TRUE** it will do something for the **TRUE** condition.

Take the following:

```
IF A > 90 THEN PRINT "YOU ARE SMART"
```

The A > 90 is a condition. The **PRINT "YOU ARE SMART"** is the thing it does for the true condition. In the above example, let's give A the value 80. Would the message be printed? No, it would not be printed because the condition is false (80 is less than 90).

Let's try a program to compute the area of a rectangle.

Type:

```
NEW
10 INPUT "ENTER THE LENGTH";L
20 INPUT "ENTER WIDTH";W
30 LET A=L*W
40 PRINT "THE AREA IS";A
50 INPUT "ANOTHER? (Y/N)";R$
60 IF R$="Y" THEN 10
70 PRINT "THAT'S ALL FOLKS!"
80 END
```

This program uses the **INPUT** statement to get values for numeric variables and prints a computed result. It then prompts for a string variable and makes the decision to begin again if the value typed in is a Y.

Here's another one.

Type:

```
NEW
10 INPUT "GIVE ME A NUMBER";A
20 INPUT "AND ANOTHER";B
30 PRINT "ADD, SUBTRACT, MULTIPLY OR DIVIDE"
40 PRINT "TYPE IN THE FIRST LETTER OF THE"
50 PRINT "ARITHMETIC OPERATION YOU WANT"
60 PRINT "ME TO DO FOR YOU"
70 INPUT R$
100 IF R$="A" THEN C=A+B : GOTO 200
110 IF R$="S" THEN C=A-B : GOTO 200
120 IF R$="M" THEN C=A*B : GOTO 200
130 IF R$="D" THEN C=A/B : GOTO 200
140 PRINT "YOU MADE A MISTAKE"
150 PRINT "TRY AGAIN"
160 GOTO 30
200 PRINT "THE ANSWER IS";C
300 END
```

This is a long program but it has a lot of new things in it that are worthwhile to know about. Line 70 contains an **INPUT** statement without the prompt string. Lines 100 through 130 contain two statements for the thing to do when the condition is true. One is a **LET** statement without the word **LET** (C=A+B) and the other is a GOTO statement. Both of the statements are separated from each other by the colon (: symbol). The GOTO statement tells the computer the number of the next statement to do. Line 140 is an error trap. It tells you that you have not followed instructions.

You can experiment some more with this program and put in a decision to start the program all over again.

What happens if you change line 300 to be GOTO 10? Yes, you are right. The program continues forever. You can stop it by pressing the **SHIFT** and **BREAK** keys and resume it by typing in **CONT** and press **ENTER**.

8. LOOPING

A loop is a set of one or more instructions. These instructions can be repeated as many times as you want. You can create a loop with a **GOTO** statement.

A program like the following will run forever:

```
10 INPUT "GIVE ME A NUMBER";N

20 PRINT "YOUR NUMBER IS";N

30 GOTO 10
```

You can modify the program to get out of a loop by using an IF statement to control the loop like the program that calculates the area of a rectangle.

Another way to create a loop is through the use of the **FOR** and **NEXT** statements. These statements surround the instructions that you want to repeat.

For example, type the following:

```
NEW

10 FOR N=1 TO 10

20 PRINT "HELLO"

30 NEXT N

RUN
```

How many times does the word **HELLO** get printed? You can change line 20 to print the value of **N** every time the loop repeats.

Type:

```
20 PRINT "N IS";N
```

We could always rewrite the program like this:

```
10 N=1

20 PRINT "N IS";N

30 N=N+1

40 IF N > 10 THEN END

50 GOTO 20
```

Are they the same? We can also count by 2's. Let's bring back the previous program and change line 10.

The entire program looks like this:

```
NEW

10 FOR N=1 TO 10 STEP 2

20 PRINT "N IS";N

30 NEXT N

RUN
```

The STEP 2 part changes the meaning hidden in the **NEXT** N statement to be N = N + 2 or any other number that gets put in the **STEP** part.

We can even count backwards.

Try this:

```
NEW

10 FOR N=10 TO 1 STEP -1

20 PRINT N

30 NEXT N

40 PRINT "BLAST OFF"

RUN
```

9. MORE ABOUT GROUPS - GOSUB ... RETURN

How much is 10 degrees Celsius in Fahrenheit? What is 100 degrees Fahrenheit in Celsius? Here's a program that gives you the answers. It uses the **GOSUB** and **RETURN** statements to create a group of instructions that can be executed from various parts of the program. Remember when you use a **GOSUB**, the program branches to the line number that you specify in the statements that will be executed sequentially until a **RETURN** statement is encountered. At that point the program will resume at the next statement following the **GOSUB**. Confusing? Not really, but first some more background about our problem.

The formula for converting Celsius to Fahrenheit is:

F=(9/5*C)+32

The formula for Fahrenheit to Celsius is:

C=(F-32)*5/9

Now for the program!

Type:

```
New
10 INPUT "ENTER THE CELSIUS TEMP";C
20 GOSUB 500
30 INPUT "ENTER THE FAHRENHEIT TEMP";F
40 GOSUB 600
50 END
500 F=(C*9/5)+32
510 PRINT "THE FAHRENHEIT TEMP IS";F
520 RETURN
600 C=(F-32)*5/9
610 PRINT "THE CELSIUS TEMP IS";C
630 RETURN
```

REFERENCE SECTION

The **REFERENCE SECTION** contains a brief explanation of all the commands and statements you will use in this chapter. Refer to this when you need help.

CLEAR

The CLEAR statement is used to assign more memory space for the string variables.

Example:

```
10 Clear 100
```

This command will assign 100 bytes of memory for strings. If the **CLEAR** command is not used, the computer will assume the number of bytes of memory for strings to be 50. The use of the **CLEAR** command only, will reserve the same number of bytes as the default value. However, if a value follows the **CLEAR** command, the computer will assign the number of bytes of that value. If you want to use more strings in your program, set this number to a larger one but, at the same time, you will have less space for your program.

CONT

Also see **STOP**

This command causes the program to resume executing after encountering the **BASIC** command called **STOP**. The program will carry on with the next statement after **STOP.** You can type **CONT** to resume the program after you hit the **SHIFT** and **BREAK** keys.

DIM

Line # DIM array-name (array-size).

The **DIM** command reserves space for one dimensional numeric or string arrays. The array-name may be up to 6 characters long but only the first 2 characters are valid. For a string array the last character must be a $. The array can contain up to 100 elements (0-99).

Example:

```
10 DIM FRIEND$(50)

10 DIM PRICE(19)
```

EDIT

Use the **EDIT** command when you want to change a statement that has been typed into memory without re-typing the entire line. Just type **EDIT** and line number and press the **ENTER** key. The statement will appear on the display. Use the **LEFT-ARROW** and **RIGHT-ARROW** keys to move the cursor. Use the **DEL** key to remove an unwanted character, or simply type in new characters.

Example:

```
10 INPUT B                press ENTER

EDIT 10                   press ENTER

10 INPUT B_
```

```
10 INPUT B̲
```

```
10 INPUT C_
```

You can type in line-numbered **BASIC** statements in any order. **PRECOMPUTER POWER PAD**™ will sort them out for you and **LIST** them or **RUN** them starting with the lowest numbered one. It is a good idea to number your lines in increments of 10 (10, 20, 30...) or 100 (100, 200, 300...). This way you have room to add more statements if you decide to change your program in the future.

You can insert an entirely new line in a program by using a line number that doesn't exist between two existing ones. You can delete an existing line-numbered statement by merely typing the line number and pressing the **ENTER** key. Of course, you can change an existing line by merely retyping the entire line.

END

Line # END

The **END** statement is used as the last command in a program. It stops the program.

Examples:

```
160 END
```

```
999 END
```

FOR ... TO ... STEP ... NEXT

Line# **FOR** variable=initial value **TO** final value **STEP** stepsize (optional)

Line#

Line#

Line#

Line# **NEXT** variable

The **FOR ... NEXT** statement repeats a task a set number of times without having to rewrite it. All statements between the **FOR** and **NEXT** command are repeated based on the initial value, the final value and the stepsize. If the **STEP** portion of the command is not used, it is assumed to be 1.

Example:

```
60 FOR ITEMS=1 TO 20

70 INPUT PRICE

80 SUM=PRICE+SUM

90 NEXT ITEMS
```

FUNCTION

A function is a mathematical procedure which when applied to a certain value will give a new value. We call the value in brackets ( ), the argument and the new value the result.

Example: **SQR** is the square root function. So if we type **PRINT SQR (9) (press ENTER)** we will get the answer 3.

Example:

```
Y=9 : X=SQR(Y) : PRINT X                press ENTER
```

We will get the answer 3.

In these two examples 9 is the argument, **SQR** is the function and 3 is the result.

A LIST OF NUMERIC FUNCTIONS

Function	What it does
ABS (X)	Returns the absolute (positive) Value of X
SGN (X)	Returns the sign of the argument X negative returns - 1 X positive returns + 1 X zero returns 0
SQR (X)	Returns the square root of X. X cannot be negative.
LOG (X)	Gives the natural logarithm of X, i.e., the logarithm to the base e (=2.71828). The value of the argument must be greater than zero.
EXP (X)	Gives you the value e^x , i.e., the natural antilogarithm of X.
INT (X)	Gives the greatest integer which is less than or equal to X.
RND (X)	Gives a random whole number between 1 and X. If X equals zero, **RND (X)** returns a random number between 0 and 1. X cannot be negative.
SIN (X) **COS (X)** **TAN (X)**	The argument of the trigonometrical functions is taken to be in radians (1 radian=360/2π = 57.296 degrees). The range of X is -999 < (X) < 999.
ATN (X)	This gives the result of **ARC TANGENT** in radians.

ARITHMETIC FUNCTIONS

ABS (X)

This gives the absolute (positive) value of the argument. So ABS (-7) =7.

Example:

```
PRINT ABS (7-2*4)                  press ENTER
1
```

SGN (X)

This function will give the value of +1 if X is positive, 0 if X is zero, and -1 if X is negative. So SGN(4.3)=1; SGN(0)=0; SGN(-276)=-1

Example:

```
A=6                                press ENTER
PRINT SGN (A); SGN (A-A)           press ENTER
1   0
```

INT (X)

This converts arguments which are not whole into the largest whole number below the argument. So **INT** (5.9)=5;also INT (-5.9)=-6. Note that with negative arguments, the absolute value of the result returned by **INT** will be greater than that of the argument.

Example:

```
PRINT INT (-6.7)                        press ENTER
-7
```

RND (X)

This will produce a random number between 1 and X if X is positive.

Example:

```
PRINT RND (19)                          press  ENTER
```

You will get a number between 1 and 19. **RND** (0) will give you a number between 0 and 1.

Note: X cannot be negative.

STRING FUNCTIONS

We can also use functions to act on strings. Have a look at the following:

Note: From now on, the **ENTER** key sign will be deleted for simplicity. Remember to press the **ENTER** key after each line of entry.

LEN

This function computes the length of the string argument, which must be in brackets. So if you type **PRINT LEN ("JOHN")** the computer will return the result 4. This is telling you that there are 4 characters in the string **"JOHN"**. Blank spaces have the value of characters. Thus if you put in spaces **"J O H N"**, it comes out as 7 characters.

STR$

The **STR$** function changes a number argument into a string. Let us take a look at the following example and see how it works.

Example:

```
A$=STR$(73)
```

This is the same as saying A$="73".

Here is a sample program

Example:

```
10 A$=STR$(7*3)
20 B$=A$+"BIG"
30 PRINT B$
RUN
21BIG
```

VAL

VAL works like **STR$** but in reverse. It changes a string argument into a number. Look at the following short program.

Example:

```
10 A$="33"
20 B$="20"
30 C=VAL(A$+B$)
40 D= VAL (A$) + VAL (B$)
50 PRINT C;D
RUN
3320  53
```

SUBSTRINGS

It is also possible to get substrings of strings. A substring is, as you might guess, a part of a string. For example: "ABC" is a substring of "ABCDE". The following three functions operate on substrings.

LEFT$ (A$,N)

This will return the substring from the leftmost of string A$, the first character to the Nth character.

Example:

```
10 A$="ABCDE"
20 B$=LEFT$ (A$+"FGH",6)
30 PRINT B$
RUN
ABCDEF
```

RIGHT$ (A$,N)

This will return a substring but starting from the Nth character from the end and running to the last one - the right most character in the string A$.

Example:

```
10 A$="WHY"
20 B$=RIGHT$(A$+"ME",4)
30 PRINT B$
RUN
HYME
```

MID$ (A$,M,N)

This function returns a substring of the string A$ starting from the Mth character with a length of N characters.

Example:

```
10 A$="ABCDEFGH"
20 B$=MID$(A$,2,3)
30 PRINT B$
RUN
BCD
```

ASC(A$)

The ASC statement will return the ASCII code (in decimal) for the **FIRST** character of the specified string. Brackets must enclose the string specified. Refer to the appendix for the ASCII code table (Pg.54). For example the ASCII decimal value of "X" is 88. If A$="XAB", then ASC (A$)=88.

Example:

```
10 X=ASC("ROY")
20 PRINT X
RUN
82
```

CHR$ (N)

This statement works the opposite way as the **ASC** statement. The **CHR$** statement will return the string character which corresponds to the given ASCII code. The argument may be any number from 32 to 127 or any variable expression with an integer value within that range. Brackets must be put around the argument.

Example:

```
30 PRINT CHR$(68)

RUN

D
```

GOSUB...RETURN

Line# **GOSUB** first line number of subroutine.

Line#

Line#

Line#

Line# first line number of subroutine

Line#

Line# **RETURN**

The **GOSUB** command tells the computer to **GOTO** another line number. After the process has been completed, a **RETURN** statement is used to send the computer back to the line immediately after the **GOSUB.** These statements are used to process a series of commands that are frequently used.

Example:

```
30 GOSUB 120

.

.

other program lines

.

.

120 PRINT "WELCOME TO THE"

130 PRINT "PRECOMPUTER POWER PAD™ "

140 RETURN
```

GOTO

Line # **GOTO** line number

The **GOTO** statement is used to change the normal flow of the program (which is from the lowest statement number to the highest). A **GOTO** command transfers control to the specified line.

Examples:

```
10 GOTO 130

200 GOTO 65
```

IF...THEN...ELSE

In general terms, the IF...THEN... statement is used for **CONDITIONAL BRANCHING.** It uses the general form "IF (condition) THEN (action clause)". A condition is made up of an expression, a relation and an expression.

Any BASIC expression, either numeric or string, may be used, but both expressions must be the same type.

Relations or comparisons used in the IF...THEN statement are the following:

= Equal to

< = Less than or equal to

< > Not equal to

> = Greater than or equal to

< Less than

> Greater than

Examples of how you can use conditionals:

IF....THEN A=B

IF....THEN GOTO

IF....THEN GOSUB

IF....THEN PRINT

IF....THEN INPUT

Example:

```
30 IF X >25 THEN 60
```

If the condition X >25 is true, the computer is told to jump to line 60 (Note:the GOTO is optional after THEN).

If the condition is not true, that is, if X is not greater than 25, then the computer simply carries on with the normal line number order in the program. Notice that it is not necessary to use the **ELSE** part of the command here, as this is optional.

Example:

```
10 INPUT A,B

20 IF A > B THEN 50

30 IF A < B THEN 60

40 IF A = B THEN 70

50 PRINT A; "IS GREATER THAN"; B:END

60 PRINT A; "IS LESS THAN"; B: END

70 PRINT A; "IS EQUAL TO";B

80 END

RUN

? 7

?? 3

7 IS GREATER THAN 3
```

Example:

```
40 IF P=6 THEN PRINT "TRUE" ELSE PRINT "FALSE"
```

In this example if P=6 the computer will print **TRUE**. Any other value will produce a **FALSE**. In either case the computer will carry onto the next line.

It is possible for more than one statement to follow the **THEN** or **ELSE** command, a colon separates the statements.

Example:

```
50 IF A =5 THEN PRINT "TRUE": S=S-3:

GOTO 90 ELSE PRINT "FALSE": K=K+8
```

So if A equals 5 the computer will print **TRUE**, subtract 3 from the variable S and go to line 90. If A does not equal 5 the computer will print **FALSE**, add 8 to the variable K and then carry on with the next normal line.

LOGICAL OPERATORS

Logical operators are used IF...THEN...ELSE and such statements where a condition is used to determine subsequent operations within the user program. The logical operators are: AND,OR,NOT.

For purposes of this discussion, A and B are relational expressions having only **TRUE** and **FALSE**. Logical operations are performed after mathematical and relational operations.

Operator	**Example**	**Meaning**
NOT	NOT A	If A is true, NOT A is false.
AND	A AND B	A AND B has the value true, only if A and B are both true. A AND B has the value false if either A or B is false.
OR	A OR B	A OR B has the value true if either A or B or both are true. It has the value false if both are false.

TRUTH TABLES

The following tables are called **TRUTH TABLES**. They illustrate the results of the previous logical operations with both A and B given for every possible combination of values.

TRUTH TABLE FOR "NOT" FUNCTION	
A	**NOT A**
T	F
F	T

TRUTH TABLE FOR "AND" FUNCTION		
A	**B**	**A AND B**
T	T	T
T	F	F
F	T	F
F	F	F

TRUTH TABLE FOR "OR" FUNCTION		
A	**B**	**A OR B**
T	T	T
T	F	T
F	T	T
F	F	F

Note that T = TRUE and F = FALSE.

Example:

```
10 INPUT A,B,C
20 IF A=B AND B=C THEN PRINT "A=B=C"
30 IF (NOT A=B) OR (NOT B=C) THEN 50
40 END
50 PRINT "A=B=C IS FALSE"
60 END
RUN
? 10
?? 5
?? 7
A=B=C IS FALSE
```

Moreover **AND, OR, NOT** can be used to manipulate numerical values. These operations are based on binary numbers with 1 and 0 representing **TRUE** and **FALSE** respectively.

For example:

i) NOT 1=-2 [1=binary 00000001 and -2=binary 11111110, so it just changes the 1 to 0 and 0 to 1. In other words, TRUE(1) changed to FALSE(0) and FALSE(0) is changed to TRUE(1).]

ii) 6 OR 13=15 [6=binary 00000110 and 13=binary 00001101, so, with reference to the OR truth table, 6 OR 13 =15=binary 00001111]

iii) 6 AND 13=4 [6=binary 00000110 and 13=binary 00001101, so with reference to the AND truth table, 6 AND 13=4 binary 00000100]

INPUT

Line # **INPUT** "(optional character string)"; variable 1, variable 2,....

INPUT allows the user to type in the value of a variable at the time the program is **RUN**. If an optional character string is used, this message will be printed before the question is asked. The type of data to be **INPUT** varies according to the type of the variable.

Examples:

```
35 INPUT AMOUNT
```

```
140 INPUT "WHAT IS YOUR NAME:";NAME$
```

LET

Line # **LET** variable = variable expression

The variable expression is calculated and the result is stored under the variable. The word **LET** is optional.

Examples:

```
40 LET SUM=A+B+C
```

```
25 LET AMOUNT=PRICE–DISCOUNT
```

LIST

LIST (optional line number)

LIST is used to display the active program. If the optional line number is omitted, the program will be displayed from the first line-on. If the line is longer than 20 characters, you must use —> to move to the right. To continue the **LISTING** press the **ENTER** key. To discontinue the **LISTING** press **SHIFT** and **BREAK** keys.

Examples:

```
LIST
```

```
LIST 50
```

NEW

NEW erases all program lines from the active program area.

PRINT

Line # **PRINT** expression and/or character string

The **PRINT** statement is used to produce output on the display panel. The **PRINT** command will print one or several items – either expressions or strings. Each item in the list should be separated by a comma or a semicolon.

Example:

```
80 PRINT NAME$; "IS COMING"

230 PRINT "SUM="; A+B+C
```

READ AND DATA

When it is necessary to enter a lot of information or data into the computer, using the **INPUT** statement can be very time consuming. To help out use the **READ** and **DATA** commands.

Example:

```
10 DATA 10,60,70,80,90
20 READ A,B,C,D,E
30 PRINT A;B;C;D;E
RUN
10 60 70 80 90
```

The **READ** statement consists of a list of variable names with commas between each variable.

The **DATA** statement consists of a list of expressions separated by commas. These expressions can be either numeric or strings. The **READ** statement makes the computer look up the value of its variables from the **DATA** statement. When the computer goes to **READ** first it will assign the first expression from the **DATA** list. The next time it goes to **READ** it will assign the second value–And so on. If the **READ** runs out of **DATA** you will get '? **OUT OF DATA ERROR.**'

RESTORE

If you want to use the same data later on in the program you can do so by using the **RESTORE** statement.

Example:

```
10 DATA 1,3,8,9
20 READ A,B,D
30 RESTORE
40 READ X,Y
50 PRINT A;B;D
60 PRINT X;Y
70 END
RUN
1  3   8
1  3
```

The **RESTORE** command makes subsequent **READ** statements get their values from the start of the first **DATA** statement.

Now see if you can work out what is happening here.

Example:

```
10 REM FIND AVERAGE

20 DATA 0.125,3,0.6,7

30 DATA 23,9.3,25.2,8

40 S=0

50 FOR I=1 TO 8

60 READ N

70 S=S+N

80 NEXT

90 A=S/8

100 PRINT A

RUN

9.52813
```

REM

Line # **REM** text

REM is used to add comments in your program which are ignored when the program is **RUN**.

Example:

```
10 REM**GUESSING GAME**

20 REM TEST OF SORTING
```

RUN

RUN tells the computer to begin to perform your program beginning with the lowest statement number.

STOP ... CONT

Line #   **STOP**

CONT

The **STOP** command halts the **RUNNING** of a program at that line. This is helpful for debugging. To **CONTINUE** at the next line after the **STOP** command, type **CONT** directly without a line number.

Example:

```
800 STOP
```

APPENDIX

EXAMPLE PROGRAMS

1 MOONWEIGHT

```
10 PRINT "Weight on the Moon"
20 INPUT "Your Weight in Lbs. : " ; WEIGHT
30 MOONWEIGHT = WEIGHT /6
40 PRINT "Your moon weight would"
50 PRINT "be" ; MOONWEIGHT; "pounds."
60 PRINT "Wow"
```

2 STONE, PAPER, SCISSORS

```
10 PRINT " Stone, Paper, Scissors"
20 PRINT " Type in St, P, S"
30 INPUT " and see if you win. ->"; ANSWER$
40 PRINT " You : " ; ANSWER$;
50 MYGUESS = INT (RND (0) * 3) + 1
60 IF MYGUESS = 1 THEN PRINT " Me : Stone"
70 IF MYGUESS = 2 THEN PRINT " Me : Paper"
80 IF MYGUESS = 3 THEN PRINT " Me : Scissors"
100 INPUT " To play again press Y : "; AGAIN$
110 IF AGAIN$ = "Y" OR AGAIN$ = "y" THEN 20
120 END
```

3 CANDY KISSES

```
10 PRINT "How many candy kisses"
20 PRINT "would equal your"
30 PRINT "weight if each kiss"
40 PRINT "weighs one ounce"
60 INPUT "Enter your weight." ; WEIGHT
70 INPUT "Enter a guess.";GUESS
90 AMOUNT = WEIGHT * 16
100 DIFF = ABS (AMOUNT-GUESS)
110 PRINT AMOUNT; "kisses will"
120 PRINT "equal your weight."
130 PRINT "You were "; DIFF; "off."
140 IF DIFF= 0 THEN PRINT "Wow that was right on!"
150 END
```

4 LIFE

```
10 PRINT "If an average life"
20 PRINT "span is 70 years."
30 PRINT "Then how many"
40 PRINT "generations would have"
50 PRINT "lived in 2000 years?"
60 INPUT "Enter a guess.";GUESS
70 AMOUNT = INT (2000/70)
80 DIFF = ABS (AMOUNT–GUESS)
90 PRINT "The answer was ";AMOUNT"."
100 PRINT "You were "; DIFF
110 PRINT "generations off."
120 END
```

5 THE COOLEST PERSON

```
10 PRINT "Enter the coolest"
20 INPUT "person you know.";COOL$
30 FOR COUNT = 1 TO 50
40 PRINT COOL$;" is super cool!"
50 NEXT COUNT
60 END
```

6 THE FORTUNE TELLER

```
10 DIM N$(5)
20 PRINT "The Fortune Teller"
30 INPUT "boy or girl" ;S$
40 INPUT "Pick a number between 2-5"; NUMBER
50 FOR COUNT = 1 TO NUMBER
60 IF S$ = "boy" OR S$ = "BOY" THEN 70 ELSE 90
70 INPUT "Enter a girl's name:"; N$(COUNT)
80 NEXT
85 GOTO 110
90 INPUT "Enter a boy's name:"; N$(COUNT)
100 NEXT COUNT
110 Y= INT(RND (NUMBER))
120 TG$ = N$(Y)
130 PRINT "The person of your"
140 PRINT "dreams is " ; TG$
```

7 LEARN SCHEDULE

```
10 DIM WEEK$(6)
20 DIM EVENT$(6)
30 DATA "SUNDAY","PLAY TIME"
40 DATA "MONDAY","LEARN ECOLOGY"
50 DATA "TUESDAY","LEARN ENGLISH"
60 DATA "WEDNESDAY","LEARN MUSIC"
70 DATA "THURSDAY","LEARN HISTORY"
80 DATA "FRIDAY","LEARN GEOGRAPHY"
90 DATA "SATURDAY","PLAY FOOTBALL"
100 REM *****LIST******
110 FOR W=0 TO 6
120 READ WEEK$(W)
130 READ EVENT$(W)
140 PRINT WEEK$(W),EVENT$(W)
150 NEXT W
999 END
```

8 MAD LIB'S

```
10 REM ** MAD LIB'S **
20 PRINT "Mad lib's"
30 INPUT "Adjective"; A1$
40 INPUT "Verb"; V$
50 INPUT "Adverb"; A2$
60 PRINT "Once upon a time"
70 PRINT "there was a "; A1$
80 PRINT "girl that decided to"
90 PRINT V$; "    "; A2$;" down the"
100 PRINT "hill to save her"
110 PRINT "father"
120 END
```

9 FUN WORD GAME

```
10 REM ** FUN WORD GAME **
20 PRINT "Fun word game"
30 INPUT "Enter girl's name"; GIRL$
40 INPUT "Enter boy's name"; B1$
50 INPUT "Enter a number"; NUMBER
60 INPUT "Enter boy's name"; B2$
70 INPUT "Enter a number";YEARS
80 PRINT GIRL$;" is going to"
90 PRINT "marry "; B1$; YEARS; "years"
100 PRINT "from now and ";B2$
110 PRINT "will be their"
120 PRINT "gardener. They will"
130 PRINT "also have"; NUMBER; "kids."
```

The ASCII code table

Decimal Value	Printable Character	Decimal Value	Printable Character	Decimal Value	Printable Character
32	(Space)	64	@	96	(Accent)
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	(Apostrophe)	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[	123	{
60	<	92	\	124	|
61	=	93	]	125	}
62	>	94	^	126	~
63	?	95	_ (Under score)	127	Δ

THE ERROR MESSAGES

CANNOT CONTINUE

An attempt is made to continue a program that:

1. has halted due to an error,
2. has been modified during a break in execution, or
3. does not exist.

DIVISION BY ZERO

A division by zero is encountered in an expression, or the operation of involution results in zero being raised to a negative power.

EXTRA IGNORED

More than one parameter is entered to 'INPUT' command.

ILLEGAL DIRECT

A statement that is illegal in direct mode command. Example: **INPUT**

ILLEGAL FUNCTION CALL

A parameter that is out of range is passed to a math or string function. This error may also occur as the result of:

1. a negative or unreasonably large subscript
2. a negative or zero argument with LOG
3. a negative argument to SQR
4. a negative mantissa with a non-integer exponent
5. an improper argument to MID$, LEFT$, RIGHT$.

MISSING OPERAND

The operand of some commands are missed.

NEXT WITHOUT FOR

A variable in a **NEXT** statement does not correspond to a previously executed unmatched **FOR** statement variable.

OUT OF DATA

A **READ** statement is executed when there are no **DATA** statements with unread data remaining in the program.

OUT OF MEMORY

A program is too large, had too many **FOR** loops or **GOSUB**, too many variables, or expressions that are too complicated.

OUT OF STRING SPACE

String variables have caused **BASIC** to exceed the amount of free memory remaining.

OVERFLOW

The result of a calculation is too large to be represented in the number format. If underflow occurs, the result is zero and execution continues without an error.

REDIMENSIONED ARRAY

Two **DIM** statements are given for the same array, or a **DIM** statement is given for an array after the default dimension of 10 has been established for that array.

REDO

A string is assigned to a numeric variable during the execution of the **INPUT** command.

RETURN WITHOUT GOSUB

A **RETURN** statement is encountered for which there is no previous unmatched **GOSUB** statement.

STRING FORMULA TOO COMPLEX

A string expression is too long or too complex. The expression should be broken into smaller expressions.

STRING TOO LONG

An attempt is made to create a string more than 255 characters long.

SUBSCRIPT OUT OF RANGE

An array element is referenced either with a subscript that is outside the dimensions of the array or with the wrong number of subscripts.

SYNTAX ERROR

A line is encountered that contains some incorrect sequence of characters (such as unmatched parentheses, misspelled command or statement, incorrect punctuation, etc.)

TYPE MISMATCH

A string variable name is assigned a numeric value or vice versa; a function that expects a numeric argument is given a string argument or vice versa.

CHAPTER 6 SPELL CHECKER

1. Sound Out A Word

Try in the word the way that you think it's spelled. Then press the **ENTER** key. If the spelling is correct, the screen will display **"CORRECT!"**. If it is not the **PRECOMPUTER POWER PAD™** will give a list of suggestions. Use the PREV and NEXT keys to search back and forth in the list until **"END OF LIST"** appears on the screen. Depending on how close to the correct spelling you are, the actual word you are searching for will be displayed on the screen.

If **"NO WORD FOUND!"** appears on the screen, input the word again using a different spelling.

2. USE THE QUESTION MARK KEY

Another way to find the correct spelling of a word, is to use the question mark symbol in place of the letters that you are not sure of.

NOTE: Use this function when you know how many letters are in the word.

EXAMPLE: If you want to know the spelling of **DICTIONARY** but you only remember the letters **DICT_NA_Y**, press the question mark key for the letters you don't know, like **DICT??NA?Y**, then press **ENTER**. **PRECOMPUTER POWER PAD™** will then display **DICTIONARY** on the screen.

3. USE THE PERIOD KEY

Use the period key to find the correct abbreviations for words.

EXAMPLE: If you want to learn the proper abbreviation for the word **ETCETERA,** type in the letters that you think are correct followed by the **PERIOD** Key, and press **ENTER**. If you input **"ETA."**, the unit would begin displaying abbreviations that are close to what you typed, including the correct abbreviation for **ETCETERA, "ETC."**

NOTE: Depending on the word that you input, **PRECOMPUTER POWER PAD™** will display either one word or a suggested list of words.

4. USE THE APOSTROPHE KEY

Use the **APOSTROPHE** key to find the correct contraction for words.

EXAMPLE: If you want to know the correct contraction of **"ARE NOT"**, type in the contraction (with the apostrophe) the way you think it should be spelled, and press **ENTER**. If you input **"AREN'T"**, **PRECOMPUTER POWER PAD™** would display a list of words including the correct spelling of **"AREN'T"**.

5. USE THE HYPHEN KEY - 6

This key allows you to check the spelling of hyphenated words.

EXAMPLE: If you want to find the correct way to spell "X -RAY", type it in the way you think it should be spelled including the hyphen. If you input "S-RA", **PRECOMPUTER POWER PAD™** will display a list of words which include the correct spelling "X-RAY".

CHAPTER 7 CALCULATOR

The **CALCULATOR** function turns the **PRECOMPUTER POWER PAD™** learning toy into an operational calculator with a 14 digit memory. Use the numeric keys on the keyboard and the 19 specially marked keys in the second and third row. To clear the screen, use the ERASE AC key at the bottom of the keyboard.

BASIC CALCULATOR OPERATION

The basic calculator symbols are:

S +	: Addition	or	SHIFT + + = ↑
D -	: Subtraction	or	SHIFT + - 6
F X	: Multiplication		
G ÷	: Division		

USING THE MEMORY FEATURES:

H Min	: Inputs the number to memory.
J MR	: Recalls the memory. The number saved in memory will be shown on the screen.
K M+	: Adds the number to memory.
L M-	: Subtracts the number from memory.

Example:

Input			Display
23 + 45 + 78	ENTER ↵	K M+	M = 146 146 _
ERASE AC			M = 146 _
34 - 78	ENTER ↵		M = 146 -44 _
K M+			M = 102 -44 _
ERASE AC			M = 102 _
23 - 6 + 9	ENTER ↵		M = 102 26 _
L M-			M = 76 26 _

TRIGONOMETRIC FUNCTIONS

The **CALCULATOR** has special function keys located on the second and third rows of the keyboard. Always press the function key first when using them in a problem and then enter the number they are to operate on.

Aim	Operation		Display
Sqaure root of a number	Q √x	9	sqrt 9
	ENTER ↵		3
Square of a number	W x²	3	sqr 3
	ENTER ↵		9
The constant e to the power of the **ENTER**ed number. (e=2.718282)	O eˣ	2	exp 2
	ENTER ↵		7.38906
LN function	I LN	100	ln 100
	ENTER ↵		4.60517
log 10 function	P Log 10	100	log 100
	ENTER ↵		2

TRIGONOMETRIC FUNCTIONS

All the trigonometric functions can take two forms of input, radians and degrees. If you just type in a number, the **CALCULATOR** assumes the number is in radians. To enter degrees, press the degree sign which is on the E key of the input keyboard.

Function	Operation	Display
SINE	R SIN 60	sin 60
	ENTER ↵	-0.30481
SINE	R SIN 60 E o	sin 60°
	ENTER ↵	0.866025
COSINE	T COS 60	cos 60
	ENTER ↵	-0.952413
COSINE	T COS 60 E o	cos 60°
	ENTER ↵	0.5
TANGENT	Y TAN 60	tan 60
	ENTER ↵	0.32004
TANGENT	Y TAN 60 E o	tan 60°
	ENTER ↵	1.73205

Function	Operation	Display
ARC TANGENT	U ATN 60	atn 60
	ENTER ↵	1.55413
ARC TANGENT	U ATN 60 E °	atn 60°
	ENTER ↵	--E-- (atn 60° produces an error)
π (Constant)	A π	3.14159265358979
This special constant can be entered into any problem by entering this key.		

CHAPTER 8 HOW TO USE AN EXPANSION CARTRIDGE

There is an entire library of optional expansion cartridges available for **PRECOMPUTER POWER PAD™**. To use a cartridge, follow the steps below.

1. Turn **PRECOMPUTER POWER PAD™ OFF**.
2. Insert the cartridge into the cartridge slot that is located on the left side of the unit's base.
3. Turn **PRECOMPUTER POWER PAD™ ON.**
4. Press the **CARTRIDGE** key on the Activity selector

Look for the following titles at your local retailers or order them directly from our Consumer Services Department. Just call (800)521-2010 and a service representative will be happy to help you.

Item No. 80-1410	SUPER SCIENCE
Item No. 80-1533	FAMOUS THINGS AND PLACES
Item No. 80-0989	BIBLE TRIVIA
Item No. 80-1001	FANTASY TRIVIA
Item No. 80-1002	GENERAL KNOWLEDGE II
Item No. 80-1003	SPORTS TRIVIA
Item No. 80-1531	32K RAM Memory Expansion Cartridge (Expands the memory of the BASIC programming activity.)

CHAPTER 9 CARE AND MAINTENANCE

1. Keep the unit clean by wiping with a slightly damp cloth only.
2. Do not spill liquids on the keyboard or main unit.
3. Do not expose the unit to direct sunlight.
4. Do not clean with strong cleansers.
5. Do not leave the batteries in for long periods of time when the unit is not in use.

IMPORTANT NOTE:

Creating and developing electronic learning toys is accompanied by a responsibility that we at VTECH® take very seriously. We make every effort to ensure the accuracy of the information which forms the value of our product. However, sometimes errors can occur. It is important for you to know that we stand behind our products and encourage you to call our Consumer Services Department at (800)521-2010 with any problems and/or suggestions that you might have. A service representative will be happy to help you.

Note:

This equipment generates and uses radio frequency energy and if not installed and used properly, that is, strict accordance with the manufacturer's instructions, may cause interference to radio and television reception. It has been type tested and found to comply with the limits for a Class B computing device in accordance with the specifications in Sub-part J of Part 15 FCC Rules, which are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that interference will not occur in particular installation. If this equipment does cause interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- reorient the receiving antenna
- relocate this product with respect to the receiver
- move this product away from the receiver

VTECH®

Smart Play®