Simplified BASIC Programming for Microcomputers

Gerald A. Silver & Myrna Silver

SIMPLIFIED BASIC PROGRAMMING FOR MICROCOMPUTERS

GERALD A. SILVER
Los Angeles City College

MYRNA SILVER

1817

HARPER & ROW, PUBLISHERS, New York
Cambridge, Philadelphia, San Francisco,
London, Mexico City, São Paulo, Sydney

Sponsoring Editor: **John Willig**
Project Editor: **David Nickol**
Designer: **Michel Craig**
Production: **Delia Tedoff**
Compositor: ComCom Division of Haddon Craftsmen, Inc.
Printer and Binder: R. R. Donnelley & Sons Company
Art Studio: Vantage Art, Inc.

SIMPLIFIED BASIC PROGRAMMING FOR MICROCOMPUTERS

Library of Congress Cataloging in Publication Data

Silver, Gerald A.
Simplified BASIC programming for microcomputers.

Includes index.
1. Basic (Computer program language).
2. Microcomputers—Programming. I. Silver, Myrna. II. Title.
QA76.73.B3S543 1984 001.64'24 83-22561
ISBN 0-06-046162-4

PREFACE

The past decade has seen an explosive growth in home and small-business computers and the use of BASIC. The utility of BASIC has far exceeded the expectations of its creators. Today BASIC is widely used in homes as well as in education, business, industry, and science because of its innate simplicity and ample processing power.

BASIC comes close to being the ideal language for the home computer as well as the classroom. It is easily learned and is suitable for both small and large computers. BASIC is well on its way to being the de facto standard of hobbyists and users of small computers.

This book is an easy-to-understand introduction to BASIC. It gives the reader a firm grasp of the fundamental tools needed to use a computer. This book includes several significant features to help you learn the language.

1. Clear, nontechnical language is used to convey the fundamental concepts of the language. Ideas are presented in terms that the reader will already understand or can grasp easily. Advanced programming concepts and difficult terms have been kept to a minimum and are explained clearly.

2. A nonmathematical approach to the language is used. Although programs of some mathematical sophistication are presented, the reader is not required to learn mathematics as a prerequisite to learning BASIC.

3. The book uses a modular approach. Each chapter is self-contained, discussing a different part of the programming process. Each BASIC statement is presented in an individual unit, with examples, explanatory material, and exercises.

4. The subject matter is introduced as the reader needs it, moving from the simple to the complex. This allows the reader to begin writing working programs early in his or her study of the language.

5. A list of rules at the end of each language unit brings together all the many do's

and don'ts related to each command. This facilitates referencing easy-to-forget details and helps the reader improve his or her programming skills.

6. Learning is reinforced by the inclusion of exercises at the end of each language unit. The reader can immediately apply new knowledge and test his or her understanding of that command.

7. A diverse set of sample programs are carefully documented and explained. They illustrate statement usage, fundamental algorithms, and program logic. The diverse nature of the examples demonstrates the graphic, mathematical, text processing, and interactive capabilities of the language.

8. The reader is introduced to flowcharting and program debugging in the appendices. This will be useful throughout the reader's study of the language.

9. Unless otherwise noted, all programs and examples in this book have been run and tested on Microsoft BASIC. Microsoft is one of the most common versions of BASIC found on microcomputers. However, the reader will find that even if his or her system does not have Microsoft BASIC, virtually all of the programs and examples in this book will still run on his or her system. The language concepts are presented in a generic form, without tying them to specific pieces of equipment or language implementations.

An important feature of this book is the appendices, which are devoted to error prevention, program debugging, and documentation. Many illustrations are included, both to help the reader avoid errors in the first place and to enable him or her to find and correct those that will inevitably occur.

The authors wish to thank the following firms who have kindly provided assistance in preparing this manuscript: Vector Graphic, Inc.; Apple Computer, Inc.; and Radio Shack, a division of Tandy Corporation.

GERALD A. SILVER
MYRNA SILVER

CONTENTS

PART ONE EXPLANATORY TEXT

CHAPTER 1
INTRODUCTION TO BASIC PROGRAMMING

Let's begin by dispelling a few myths about computers: "You must have an extensive background in electronics, mathematics, or some other technical subject to understand computers." "Computers are complicated, mystical machines, which are only for business or scientific use."

Nothing could be further from the truth! Computers can be fun and entertaining.

Directing the computer to produce the results you want requires only three qualities: a basic skill in typing, a familiarity with a computer language (such as BASIC), and an understanding of programming principles. (And, of course, even the typing skills can often be delegated to someone else.) So if you don't know a bit from a byte, or an arc sine from a stop sign, don't be worried; you can still have a lot of fun with BASIC and your small computer.

WHAT IS BASIC?

BASIC stands for Beginner's All-purpose Symbolic Instruction Code, which is the name of a computer language; for convenience, it is usually called just BASIC. BASIC is one of the many computer languages that have appeared during the past several decades to facilitate communications between people and computers. It is the means a hobbyist or a professional programmer uses to direct a computer through the various steps necessary to get results.

In many respects, BASIC is like a foreign language. A Frenchman who speaks no English must be given directions in French if he is to be expected to understand them. Similarly, since the computer cannot "speak English," the user must learn its language in order to communicate with it.

And, like French, computer languages have rules of grammar, spelling, order, and

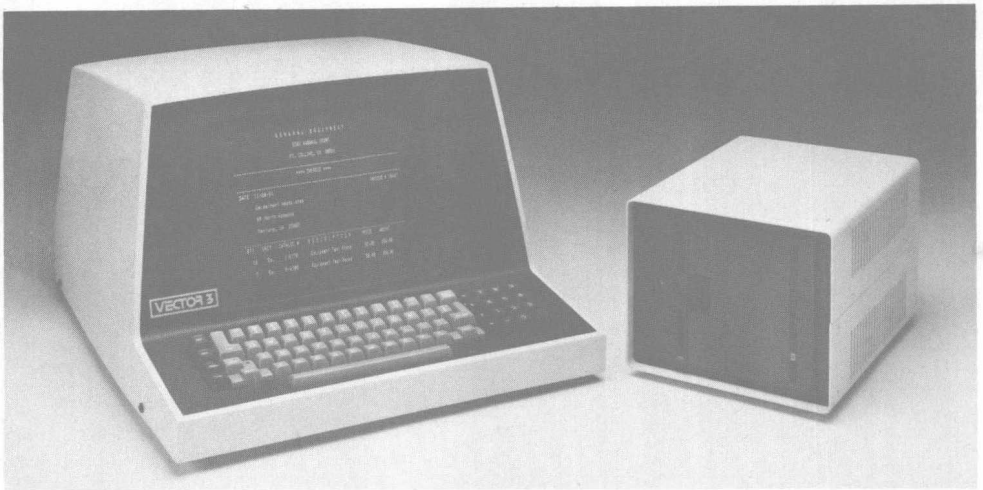

FIGURE 1.1 Microcomputer systems are widely used in business applications. (Vector Graphics, Inc.)

so on. If the directions are to be understood by the computer (or the Frenchman), the words and phrases must follow the rules.

When computers were developed in the 1940s and 1950s, it became evident that one of the biggest problems was in human-machine communication. Although the computer was capable of processing problems very quickly and accurately, the actual task of directing it was complex and difficult.

The computer understands instructions in only one language—machine language—which is composed of a combination of 1s and 0s, such as

0110 0010 1001 0001 1010

(A simplified explanation of these symbols is that they represent "on" and "off." They tell the computer which circuits to turn on and which to turn off.) Writing programs in machine language is obviously a cumbersome way for human beings to communicate with machines. In order to simplify programming, interpreter languages were developed. An interpreter is a program that remains in the computer and translates instructions into the machine language which the computer can execute.

The first computer languages were designed for use by professional programmers, scientists, and engineers, and they were relatively complicated. That hobbyists, secretaries, or even grade school children might do programming was never considered. What was needed was a simple language the professional and nonprofessional programmer could use.

John G. Kemeny and Thomas E. Kurtz, working under a National Science Foundation grant at Dartmouth College, developed a relatively simple language, which they named BASIC. Dartmouth students found BASIC easy to learn and use; and before long, many schools, colleges, and business firms became interested in the language.

BASIC uses only about 20 fundamental words. This simplifies the task of learning

FIGURE 1.2 A wide variety of games and hobby programs are available on small computers. (Radio Shack/Tandy Corp.)

the language. Today almost every home and small-business computer is equipped with a BASIC interpreter.

ADVANTAGES AND LIMITATIONS OF BASIC

BASIC is an easily learned language. It includes only a few words and concepts, being composed of readily recognizable English words and algebraic symbols. BASIC is an interactive language, which makes it very versatile.

BASIC is a powerful language, possessing good mathematical and alphabetic capabilities. It is used by the professional and nonprofessional programmer for a variety of applications—for instruction in schools and colleges, for design and engineering by engineers, mathematicians, and statisticians, among others. It is also used by business people for programming inventory, financial, accounting, marketing, and other problems (Fig. 1.1).

BASIC's usefulness is further expanded by an extensive library of ready-made programs. These programs, written by manufacturers or users, are stored in the computer's system library and can be called into use by any user. This saves a user much time and effort, since he or she can process data with these programs without taking the time and effort to write a program.

The libraries of programs vary from one computer manufacturer to another. They include statistical, financial, engineering, and mathematical procedures. Some libraries have many games and demonstration programs, such as Star Trek, Bandit, and Blackjack (Fig. 1.2).

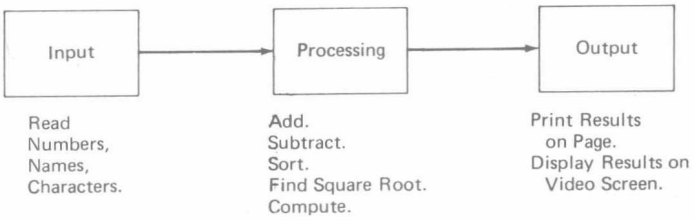

FIGURE 1.3 Data processing cycle.

HOW COMPUTERS "THINK" AND CARRY OUT INSTRUCTIONS

To solve a problem, a computer must be given a clear set of instructions and the data to be operated on. This set of instructions, called a *program,* is written by a *programmer*, who may be a student, an engineer, or a hobbyist. The data are the numbers or words that the machine is to process.

The program, or set of instructions, directs the computer to perform various tasks in a predetermined sequence. The machine may be directed to read in numbers or words, to rearrange numbers or words, to calculate sums and products, and so on. And it may be instructed to print out the results in a usable form.

The entry of data or information into the computer is called *input,* and the manipulation of data is called *processing.* The communicating of results to the user is called *output.* Figure 1.3 illustrates the input-processing-output sequence.

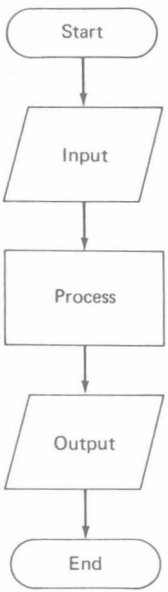

FIGURE 1.4 Batch processing.

The sequence of steps followed by the computer is thus

INPUT → PROCESSING → OUTPUT

In order for the machine to perform these tasks, the programmer must

1. write a program (give a clear set of instructions)
2. prepare the data (arrange the numbers or words to be processed)
3. enter the program into the computer
4. enter the data into the computer

The computer responds by

1. receiving the program
2. receiving the data
3. processing the data according to the instructions in the program
4. outputting results

A computer may output results in many ways. For example, it may type out information on a line printer, punch data into paper tape, display information on a video screen, or record it on magnetic tape.

WHAT IS INTERACTIVE PROGRAMMING?

Programs and data can be processed by the computer in two ways: *batch processing* and *interactive programming.* In batch processing, the program and data are entered into the computer at once and run without any further intervention or direction from the programmer. The computer operates on the data exactly as instructed by the program and outputs the results as directed. This is shown in Fig. 1.4.

In interactive programming, the program and data are entered at once, or in steps, while the computer is executing the problem. The programmer interacts with the computer, giving and receiving information, while the processing proceeds. This is illustrated in Fig. 1.5.

Interactive programming differs from batch processing in several ways. In interactive programming,

1. the program and data can be entered in parts
2. the computer can request more information or data while it is processing the data
3. the instructions in the program can be changed by the programmer during the run of the program to direct the computer to perform different steps than were originally planned
4. the programmer is in direct communication with the computer at all times

Interactive programming has numerous advantages. It provides the programmer with a fast, flexible means of solving a problem. It allows the programmer to process a

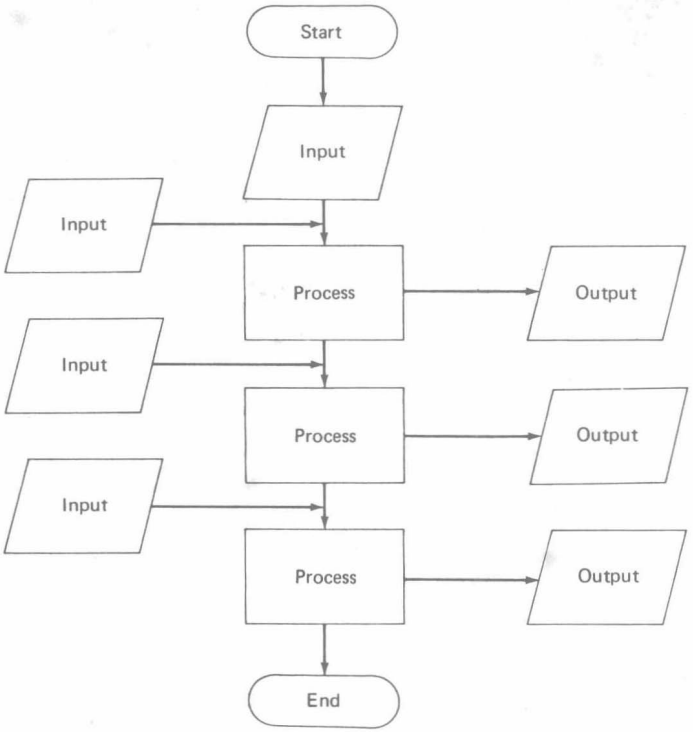

FIGURE 1.5 Interactive programming.

FIGURE 1.6 Interactive programming allows the programmer to alter program flow. (Apple Computers, Inc.)

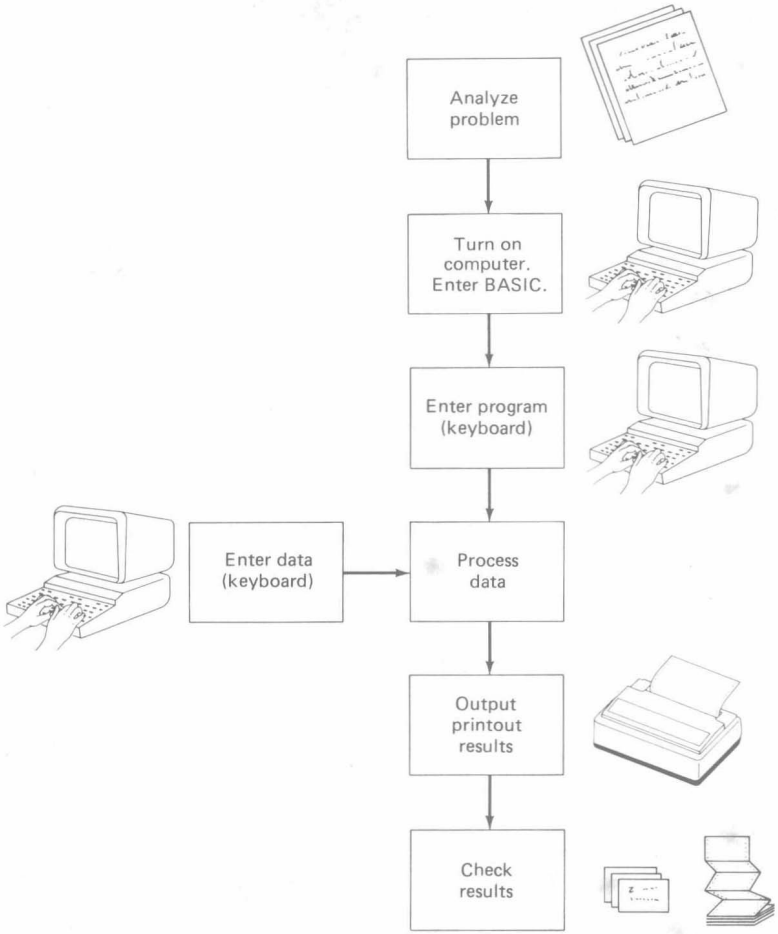

FIGURE 1.7 Computer session.

problem step by step and check each step as he or she goes along (Fig. 1.6). It allows him/her to change instructions during processing and add or delete data or perform other procedures on the data after observing the results of processing.

EXAMPLE OF HOW THE COMPUTER IS USED TO SOLVE A PROBLEM

Sue Smith wants to determine the average of the scores her classmates have achieved in her English literature class. Sue proceeds to solve the problem in the steps listed in Fig. 1.7.

1. Analyze the problem. After some study Sue writes a program that directs the computer to read the group of 20 scores, find their sum, divide by 20, and print out the answer. It takes Sue about half an hour to write down the set of instructions. Now she is ready to enter the information into the computer. The set of instructions is shown in Fig. 1.8.

```
10 REM PROGRAM AVERAGES 20 NUMBERS
20 LET T=0
30 LET C=0
40 INPUT S
50 LET C=C+1
60 LET T=T+S
70 IF C=20 THEN 90
80 GOTO 40
90 LET A=T/C
100 PRINT
110 PRINT
120 PRINT "THE CLASS AVERAGE IS: "; A
130 END
```

FIGURE 1.8 Sample program (BASIC).

2. Turn on computer and enter BASIC. This step involves preparing the computer to receive instructions in BASIC. Since Sue is using a Radio Shack TRS-80 (Fig. 1.9), she will turn on the power switch and press the ENTER key. The machine will respond by displaying the word READY on the televisionlike screen before her.

3. Type in the program. Sue now keyboards her program. Each instruction she has written down is typed line by line. Sue then types in END as the last instruction, which

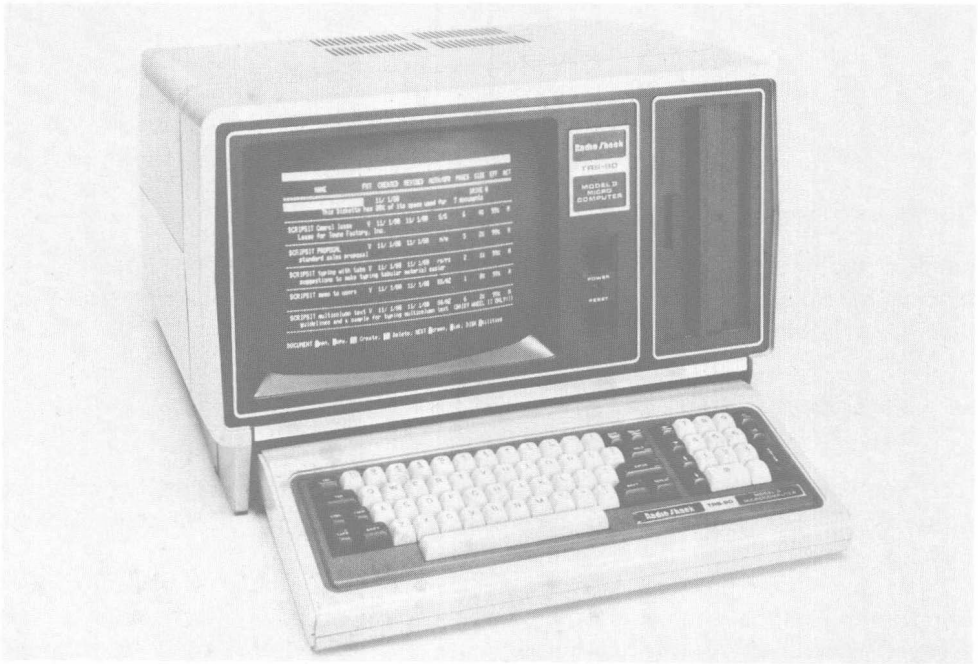

FIGURE 1.9 TRS-80 model II microcomputer. (Radio Shack/Tandy Corp.)