Programming in BASIC for Personal Computers

Practical, easy-to-follow instructions that enable home computer users to design and operate their own effective programs for countless applications.

David L. Heiserman

David L. Heiserman

Research and Development Consultant

Programming in BASIC for Personal Computers

PRENTICE-HALL, INC., Englewood Cliffs, New Jersey 07632

Library of Congress Cataloging in Publication Data

Heiserman, David L date
Programming in BASIC for personal computers.

Includes index.

1. Basic (Computer program language) 2. Minicomputers—Programming. 3. Microcomputers—Programming. I. Title
QA76.73.B3H44 001.64'24 80-15939
ISBN 0-13-730747-0
ISBN 0-13-730739-X (pbk.)

Editorial/production supervision
by Gary Samartino and Daniela Lodes
Interior design by Gary Samartino
Cover design by Jorge Hernandes
Manufacturing buyer: Joyce Levatino

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

PRENTICE-HALL INTERNATIONAL, INC., *London*
PRENTICE-HALL OF AUSTRALIA PTY. LIMITED, *Sydney*
PRENTICE-HALL OF CANADA, LTD., *Toronto*
PRENTICE-HALL OF INDIA PRIVATE LIMITED, *New Delhi*
PRENTICE-HALL OF JAPAN, INC., *Tokyo*
PRENTICE-HALL OF SOUTHEAST ASIA PTE. LTD., *Singapore*
WHITEHALL BOOKS LIMITED, *Wellington, New Zealand*

Preface

Computer power is within the grasp of most people now living in the advanced technological societies of today. Computers are no longer limited to large rooms in industry, business, universities, and military installations. You might even know a youngster down the street who plays with a computer instead of building model airplanes or flying kites. Or maybe you have a computer sitting there on the desk in front of you.

Perhaps computer power has moved into the consumer marketplace a bit too quickly. Many owners of new personal computing systems are not prepared to handle the programming tasks and are thus facing a sort of *computer shock.* Once the novelty of running prerecorded cassette programs wears off, many people begin wondering if the whole affair was worth the investment.

"I have this dandy new computer at home. Now what can I do with it, and how do I go about doing it?" That's where this book enters the scene.

The BASIC programming language is the overwhelming choice for personal computing systems, and this book is about programming in BASIC. What you can learn here applies to most personal computers on the market—including Radio Shack's TRS-80, Apple II, and the Commodore PET. The notions apply to a number of other brands, but these three happen to be the pace-setters.

There is much more to BASIC programming than simply learning the fundamental expressions. If that were not so, you could learn

everything there is to know from the owner's manual supplied with the computer.

Of course, there is a need to master the BASIC language itself, but that is only the tool for fashioning more important things. This book emphasizes the sort of analytical thinking that lets you use the tool—the BASIC language—to transform your own ideas into workable programs.

It isn't easy to learn how to fashion custom programs, no matter what language might be at your disposal. One way to handle the situation is by considering programs from two different viewpoints: (1) Given a prepared program, analyze it to see exactly how it works and, (2) given a basic idea, transform it into a workable program. Both approaches are used in this book, with the latter being the dominant one.

The discussions and examples in this book will be far more useful and meaningful if you have access to a personal computer. Do all of the exercises for yourself and give each of them a lot of careful thought. When you begin feeling confident with a new idea, try it out for yourself, extending the notion as far as your experience will let you.

Exercises in the final section of each chapter are intended to test your understanding of the material in that chapter. Computer programming is a building-block affair, and you will have trouble working the exercises properly if you've been careless about learning the material in earlier chapters.

Learning programming can be a lot of fun. It's frustrating at times, but there is a good answer to every problem you encounter. And if you handle the lessons in the proper spirit, you are likely to learn more from your own tinkering than from the book, itself. When that becomes the case, you'll know you are on your way to becoming a proficient computer programmer—"for fun or for profit," as the old saw goes.

Columbus, Ohio DAVID L. HEISERMAN

Programming in BASIC for Personal Computers

Contents

CHAPTER 1

The System READY Status and PRINT Command

Keyboard commands introduced in this chapter:

PRINT LET

Computers are virtually useless unless there is some means for getting information into them from the outside world. In the case of small personal computers, this generally means entering information from a keyboard, cassette tape machine, or disk operating system (DOS). In some special cases, input devices can also include joystick controls, pushbuttons, and a host of interfacing circuits that translate electrical signals into computer "words."

By the same token, it is important to provide some means for presenting computer information to the outside world in an intelligible fashion. Usually, this is handled by a cathode ray tube (CRT) monitor or, in some instances, a line printer. In the first case, the computer-generated information is presented in the form of *alphanumeric characters* (combinations of ordinary letters and numerals) and special symbols on the face of the CRT. In the second case, the same sort of information is presented as typewritten *hardcopy* on a sheet of paper.

For our purposes throughout this book, assume the system is operating with the most common input/output (I/O) configura-

tion—a keyboard input linked through the computer assembly to a CRT monitor output. Information is entered into the computer, by hand, from the keyboard, and the results are displayed on a CRT monitor output.

You can, of course, learn about alternative I/O configurations—using cassette tape, disk, and line-printing machines—from the owner's manual for your own computer system. And you can be sure that the BASIC you learn using the keyboard and CRT will apply equally well to the alternative systems.

Figure 1-1 shows the keyboard layout for Radio Shack's TRS-80 home computer. Note that most of the keys are labeled with characters that are identical to those found on a common typewriter. The keyboard, in fact, is normally used as though it were a common typewriter keyboard. There are a few special-purpose keys, though, carrying labels such as BREAK, ENTER, and CLEAR. These represent special computer functions that will be described as the need arises.

Other brands of home computers, such as the Commodore PET and Apple II, have similar kinds of special-purpose keys, but they often carry slightly different labels. When it comes to citing operations that use these special keys, the convention throughout this book will be to cite the Radio Shack TRS-80 key labels. If you are using a different kind of computer, you must consult your user's manual to make the conversions. Describing the possible differences each time they arise in this book would cause needless confusion and defeat one of this book's primary tasks—showing that BASIC is easy to learn and master.

A similar sort of situation arises from the fact that different models of personal computers use a few programming steps that differ slightly from the others. Most of the BASIC programming procedures are completely *portable*; that is, the programs can be carried over directly from one kind of computer to another. But there are slight differences in the BASIC procedures and conventions, and they are going to cause some annoyances if you are not aware of those little differences.

The brand of BASIC used throughout this book is based on something now commonly called Microsoft™ BASIC. It is a slight variation of the original Dartmouth BASIC, and one tailored to the special needs of small computer systems.

Radio Shack's TRS-80 uses Microsoft BASIC, so that will be the convention established in this book. Wherever there is a chance that other machines will not operate exactly as described, you will find a note referring you to your user's manual.

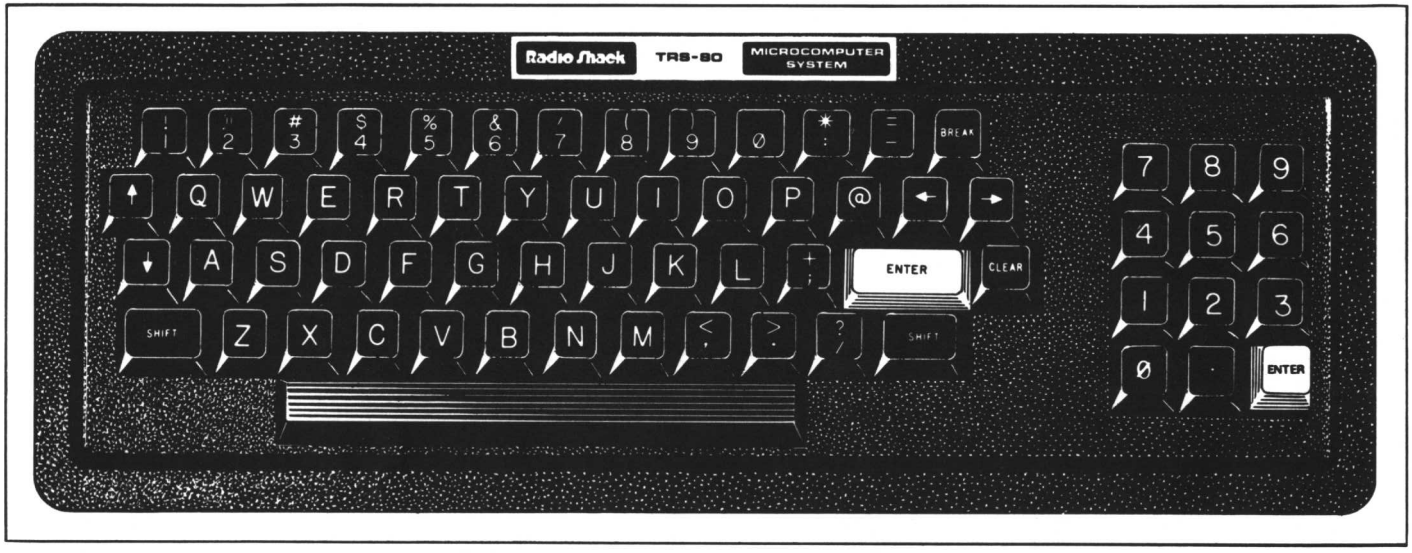

Figure 1-1 Keyboard layout for the Radio Shack TRS-80 personal computer. (Courtesy Radio Shack, a division of Tandy Corporation.)

None of this is cause for discouragement, alarm, or even confusion. The keyboard and programming differences are important, but rather slight, and it doesn't take long to get accustomed to dealing with them. Besides, it all gives you a chance to develop a special knack for dealing with the common problem of portability—a knack that will serve you quite well in the future.

1-1 The System READY Status

Although the exact procedures for turning on and initializing personal computers vary a little from one make and model to another, all schemes include some means for indicating that the system is ready to begin accepting information from the keyboard. The TRS-80, for example, generates the following message:

```
READY
>_
```

Other expressions and symbols might precede this particular message when the system is first turned on, but once this message appears on the screen, you know that the system is expecting you to begin entering new data and instructions.

The READY expression is automatically generated from within the system, and it indicates that the system has *just entered the command mode of operation.*

The *greater-than* symbol (>), properly called the *prompt* symbol, also indicates that the system is ready to accept information from the keyboard. But unlike the READY message, which appears only when the system *first enters* the command mode, the prompt symbol appear each time the system is expecting a new line of information, whether that line happens to be the first one or not.

> ***Definition.*** *Command mode*—A mode of operation whereby the computer *immediately* executes any valid operation specified by the user from the keyboard.

> *Note.* Not all systems use the *greater-than* symbol as a prompt symbol. Others might use symbols such as # or *.

Every line of information, generally made up of a computer instruction and some data, is typed into the computer one character at a time. The *underline* symbol (_), properly called the *cursor* symbol, indicates where the next keyboard character will appear on the screen. So as you type characters and spaces, the cursor moves to the right, one space at a time.

So in effect, when the computer prints READY on the screen, it is saying, "I have just entered the command mode of operation, and I'm ready to begin accepting information from the keyboard." The prompt symbol means, "I am ready to accept a new line of information." And the cursor symbol means, "Here is where the next character will appear on the screen."

Whenever the system is in the command mode and the operator is entering information, there has to be some provision for telling the computer that the line is completed. The computer itself has no way of telling when you are done entering a line of information—you have to tell it you are done.

On the TRS-80, you signal the end of a line of information by striking the ENTER key. In the command mode, striking the ENTER key signals two things: (1) the end of the command line and (2) time to execute the command.

The system responds by carrying out the specified operation and then returning the READY status message:

```
READY
>_
```

The system is then ready to start all over again. You enter a line of information and strike the ENTER key. The computer does whatever you tell it to do, then returns the READY status message again—and so on, and so on, and so on.

> *Note.* The keyboards for most computer systems have a CR or RETURN key, instead of an ENTER key. The function in any case is exactly the same.

1-2 The PRINT *"expression"* Command

Consider the following set of information and messages on the CRT:

EXAMPLE 1-1

```
READY
>PRINT "DAVID"
DAVID
READY
>_
```

The whole thing began with the system printing READY to indicate it has just entered the command mode of operation. It then printed the prompt symbol at the beginning of the next line.

The operator responded by typing a valid command, PRINT "DAVID", and striking the ENTER key.

Immediately after striking the ENTER key, the system did exactly what it was told to do–print the expression DAVID. It then returned to the command mode.

When the sequence was first started, the screen showed only the READY status message:

```
READY
>_
```

The indication is that the system had just entered the command mode (as signaled by READY) and is prepared to receive some instructions from the keyboard (as signaled by the prompt and cursor symbols).

The user then typed

```
PRINT "DAVID"
```

And before striking the ENTER key, the display looked like this:

```
READY
>PRINT"DAVID"_
```

Note that the prompt symbol remained at the beginning of the second line, but the cursor moved toward the right, still indicating where the next character or space would go. There is no more information to be entered, however.

Striking the ENTER key at this point told the computer that the command sequence from the keyboard was complete and that it was time for the computer to execute that particular command. Here is what the computer did: