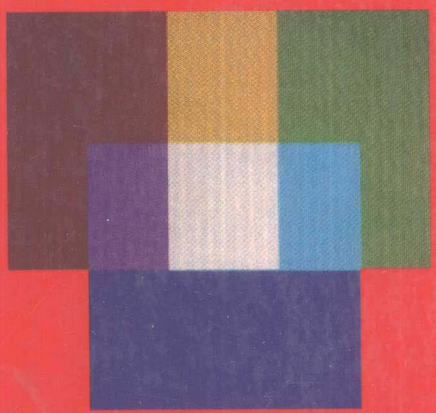
Programming The IBM Personal Computer:

Fundamentals of BASIC



Neill Graham

Programming the IBM Personal Computer:

Fundamentals of BASIC

Neill Graham

HOLT, RINEHART AND WINSTON

New York Chicago San Francisco Philadelphia
Montreal Toronto London Sydney Tokyo
Mexico City Rio de Janeiro Madrid

The cover illustration is The Additive Color Model from Miles Color Art, Tallahassee, Florida, prepared using the Digital Facsimiles process (patent pending), Center for Color Graphics, Florida State University, Tallahassee, Florida.

IBM® is a registered trademark of International Business Machines Corporation.

The photograph on page 3 is courtesy of
International Business Machines Corporation.

Address correspondence to:
383 Madison Avenue, New York, NY 10017
First distributed to the trade in 1984 by Holt, Rinehart and Winston General Book Division.

Library of Congress Cataloging in Publication Data

Graham, Neill, 1941—
Programming the IBM personal computer.

Includes index.
1. IBM Personal Computer—Programming. 2. Basic (Computer program language) I. Title. II. Title: Programming the I.B.M. personal computer.
QA76.8.I2594G726 1984 001.64'2 83–22846
ISBN 0-03-059561-4

Printed in the United States of America

Published simultaneously in Canada

4 5 6 7 039 9 8 7 6 5 4 3 2 1

CBS COLLEGE PUBLISHING
Holt, Rinehart and Winston
The Dryden Press
Saunders College Publishing

Introduction

BASIC was developed at Dartmouth in the early 1960s as an easy-to-learn language for teaching programming to beginners. The emphasis on simplicity is reflected both in the full name of the language, Beginner's All-purpose Symbolic Instruction Code, and in the acronym BASIC. During the sixties and seventies, BASIC steadily gained favor with educators, until it became the most widely used language for introductory programming courses, displacing FORTRAN which had previously enjoyed that distinction.

Another distinctive feature of BASIC was the ease of implementing it on small computers with limited amounts of memory. This led to a number of implementations on minicomputers in the sixties and seventies. When the first microcomputers, or personal computers, appeared in the mid-seventies, BASIC was already proven as both an easy-to-use and easy-to-implement language for small computers. Now BASIC is the

most widely used programming language for personal computers, and the enormous growth of personal computing in recent years has made BASIC the most widely used programming language, period.

Over the years, BASIC has been extended from a simple teaching language to one suitable for a wide range of programming tasks. While a few of the more complex extensions have moved beginners to complain that "BASIC isn't," the extended versions generally retain the ease of learning and use that characterized the original language.

This book is an introduction to BASIC for the IBM Personal Computer, or IBM PC as it's usually called. IBM PC BASIC is one of the most widely used versions of the language, not only because of the widespread use of the IBM PC, but also because of the eagerness of other computer manufacturers to make their computers compatible with the highly successful PC.

As appropriate for a book with "fundamentals" in its title, this book does not attempt to cover all of IBM PC BASIC. It does, however, cover the most useful and widely used parts; what is omitted is mainly of interest to advanced programmers. Coverage of color and graphics is brief, but the material in Chapter 15 should be sufficient to get students started on their own experiments, which are probably the best (if not the only) way to master graphics. Both BASIC 1.10 and BASIC 2.00 are covered, although some of the new features of BASIC 2.00 are sufficiently advanced to receive only brief mention or be omitted entirely.

With classroom use in mind, I have tried to make this book as independent of the BASIC reference manual as possible, but have not been able to do so completely, particularly for advanced features that warrant only brief mention here but that some students may want to explore in more detail. I suggest, therefore, that a copy of the reference manual be placed where interested students can consult it.

In my earlier book, *Programming the IBM Personal Computer: BASIC,* I used programs for such things as information retrieval and text editing to illustrate the practical applications of BASIC. These programs were necessarily lengthy, and they often involved subtle points in computer science, such as advanced searching and sorting techniques. In this book I have, for the most part, kept to simple, easy-to-understand examples. What they lack in sophistication or immediate applicability is made up for by their greater accessibility to beginners.

Finally, a few words of advice to those setting out to master BASIC. Learning to program takes practice. Nobody ever learned to program a computer just by reading a book, any more than a book ever taught anybody to ride a bicycle or play the piano. I recommend that you try as many of the programming exercises at the end of each chapter as possible. Even better, as your skills increase you should be able to find problems in your work, studies, and hobbies that you can solve with the computer.

Experiment with your computer. If you aren't sure what a particular command will cause the computer to do, try it and find out. Don't be afraid you will damage the computer (one of the most common concerns of beginners). Sure, you will damage the computer if you drop it on the floor or spill coffee in the works, but nothing you type on the keyboard will hurt. At worst, the computer will display an "error message" informing you that it doesn't understand what you are asking it to do.

Don't be afraid to make mistakes. Who ever learned to ride a bicycle without taking

any falls or to play the piano without hitting any wrong notes? Don't hesitate to try out your ideas on the computer, and don't lose any sleep over those that don't work out.

Now turn to Chapter 1 and begin your acquaintance with the IBM Personal Computer. Good luck, and have fun!

Contents

1

The IBM Personal Computer

Like every other computer, the IBM Personal Computer (or IBM PC, as it's usually called) is a machine for storing and manipulating information. A book far larger than this one would be needed to describe all the kinds of information-processing tasks to which computers have been applied. Game playing, business and personal record keeping, computer-assisted instruction, word processing (using the computer as an electronic typewriter), and control of robots are just a few of the areas in which the IBM PC has been put to work.

A computer processes information according to a set of detailed, step-by-step instructions called a *program*. Each information-processing task calls for a different program—one program for each game the computer is to play, one for each set of records it is to keep, one for each lesson it is to teach, and so on. We use the term *hardware* for the computer equipment itself and the term *software* for the programs that direct the hardware. Software is to hardware as records are to phonographs and as films are to movie projectors.

You can buy software at computer stores or through mail order, or you can write your own. Many people do some of both. Just as you may do simple repairs and

construction around the house but call professionals for the more complicated jobs, so you may want to write the simpler programs for your computer but buy the more complicated ones.

In order to write programs you need a language in which to communicate your instructions to the computer. Unfortunately, neither English nor any other human language is precise enough for the task. A similar situation arises in other areas of human endeavor, such as music, dance, knitting, chemistry, and mathematics, where special notations have to be devised for ideas that can be expressed only clumsily in English. A system of notation for giving instructions to a computer is called a *programming language*. There are nearly two hundred such languages, although nowhere near that many are in widespread use.

Some programming languages available for the IBM PC are BASIC, Pascal, COBOL, and FORTRAN. The most widely used of these is BASIC, the language described in this book. BASIC is readily available: the software needed to use one version of BASIC is built into the IBM PC, and the software needed for more advanced versions comes with the *Disk Operating System,* which almost all IBM PC owners buy anyway. As its name implies, BASIC is designed to be particularly easy for beginning programmers to learn and use. Yet in spite of its simplicity, BASIC is powerful enough for programming a wide range of applications in areas including business, education, science, and recreation.

We will begin our study of BASIC in Chapter 2. The rest of this chapter is a brief introduction to the hardware of the IBM Personal Computer and to the software needed to use BASIC. For now, don't bother trying to memorize all the details in this chapter; just read the chapter through once and go on to Chapter 2. You can come back to this chapter when you need more information on specific topics. Also note the glossary in the back of the book, which gives concise definitions of many of the terms introduced in this and other chapters.

HARDWARE

Figure 1.1 shows the main hardware components of the IBM PC. You will have no trouble recognizing the *keyboard unit,* which can be used to enter both programs and data. Behind the keyboard is the *system unit,* which is responsible for processing and storing information. Atop the system unit is the *video display,* on which the computer's output appears. To the left of the system unit is the *printer,* which we can use to produce a permanent copy of the computer's output.

The Microprocessor

In the system unit we find the heart of the computer, the microprocessor, which does all the actual data manipulation. All other devices, such as the display and the printer, work under the direction of the microprocessor. The microprocessor, in turn, works under the direction of a program, the instructions of which are in a special code called *machine language*. Machine-language programs are easy for microprocessors to execute but tedious for humans to work with, which is why most people use more human-oriented programming languages such as BASIC.

Figure 1.1 An IBM Personal Computer. The keyboard unit and the video display are easily recognized. The video display is atop the system unit, to the left of which is the line printer. The black area on the front of the system unit contains two diskette drives.

Main Memory

The data that the microprocessor is currently manipulating and the program it is currently executing are stored in *main memory*, which, like the microprocessor, is installed in the system unit. The IBM Personal Computer contains two kinds of main memory: (1) *read-only memory*, or *ROM*, the contents of which are permanent and cannot be changed by the microprocessor, and (2) *random-access memory*, or *RAM*, in which data can be stored and later retrieved. ROM is like a library book, which you can read but are not allowed to write in. RAM is like a blackboard, on which you can write whatever you need to remember, leave it as long as you need it, then erase it and write something else in its place.

ROM is sometimes referred to as *permanent memory*, since its contents are fixed when it is manufactured and never changed. The ROM for the IBM PC includes the following:

1. *A BASIC interpreter*. This is the software*, mentioned earlier, which makes available one version of BASIC. BASIC interpreters are described in more detail later in this chapter.
2. *A program for testing the computer*. This program is always executed when you first turn on the computer; after turning on the power switch you will notice a short pause while this program does its work. If the program finds anything wrong with the computer, it displays a code indicating the nature of the problem.
3. *Subroutines for communicating with input, output, and storage devices, such as*

*Software permanently installed in ROM is often referred to as *firmware*.

the keyboard and the display. A *subroutine* is a program whose execution is initiated by another program, rather than directly by the user. When the subroutine finishes its work it returns control to the program that called it. The subroutines in ROM are available for all programs to use when they need to communicate with input, output, and storage devices. This set of subroutines is called the *Basic Input-Output System* or *BIOS*.

The storage capacity of a computer memory is measured in *bytes*. Each byte of memory can store one character, such as a letter of the alphabet.* The term *kilobyte* refers to 1024 bytes and is abbreviated K or KB, so a 64K (or 64KB) memory has a capacity of 64 kilobytes and can store more than 64,000 characters. The IBM PC contains 40K of ROM. Although it can be used with as little as 16K of RAM, few machines have less than 64K of RAM, and 128K or more is common.

Auxiliary Storage

The contents of RAM are lost when the computer is turned off. Thus, we need some form of auxiliary storage for storing programs and data when the computer is not in use. Three kinds of auxiliary storage can be used with the IBM Personal Computer:

1. *Cassette tape*. Programs and data can be stored on cassette tapes using an ordinary cassette recorder, which is connected to the computer by a cable that plugs into the rear of the system unit. Because cassettes are a slow and sometimes unreliable method of storing information, cassette storage is rarely used with the IBM PC. One model of the IBM Personal Computer, the IBM PC-XT, does not provide cassette storage.
2. *Diskettes*. A diskette is a flexible plastic disk mounted in a square protective jacket. A diskette is inserted in a *diskette drive* for information to be stored on it or retrieved from it, after which it is removed from the diskette drive and stored separately. Up to two diskette drives can be installed in the system unit. These can be *single-sided* drives, which record on only one side of a diskette, or *double-sided* drives, which record on both sides of a diskette. A single-sided diskette—one recorded on only one side—can store 160K; a double-sided diskette—one recorded on both sides—can store 320K.† Diskettes are the most popular form of auxiliary storage for the IBM PC.
3. *Fixed disk*. A fixed disk is used in much the same way as diskettes; however, the fixed disk is permanently installed either in the system unit or in a separate expansion unit. A fixed disk has a much greater storage capacity than a diskette—10 megabytes, or over 10,000K—and storage and retrieval is much faster than with diskettes. The IBM PC-XT comes with one diskette drive and one fixed disk, both installed in the system unit.

*More precisely, information in a computer's memory is coded as groups of *bits*, each of which can have only two values, 0 and 1. A byte is a group of eight bits. Thus 01000001, 01000010, and 01000011, which are the standard codes for the letters A, B, and C, each occupies one byte of memory.

†By recording the data in a different format, we can store 180K on a single-sided diskette and 360K on a double-sided one.

In this book, the word *disk* is used to refer to both diskettes and fixed disks. *Diskette* and *fixed disk* are used only when we need to distinguish between the two kinds of disk storage.

Adapters

Computer circuits are constructed on *circuit boards*. The electronic components making up the circuit are mounted on the circuit board, where they are connected together by an etched pattern of copper conductors. The system unit is built around a large *system board* that contains the microprocessor, all or part of main memory, and circuits for communicating with the keyboard and a cassette recorder. Communication with and control of other devices are handled by *adapters*, smaller circuit boards that plug into "slots" (sockets) on the system board. All systems using disk storage, for example, contain an adapter for controlling diskette drives and perhaps another for controlling fixed-disk drives. Circuit boards containing additional main memory can also be plugged into the system board.

The Display and the Printer

The IBM PC can be used with either a monochrome (black and white*) or a color display, each of which requires a different adapter. For a monochrome display we need the Monochrome Display and Parallel Printer Adapter, which serves not only for the display but for the printer as well. For a color display we need the Color/Graphics Monitor Adapter. The Color/Graphics Monitor Adapter does not handle a printer, so a Parallel Printer Adapter is also needed if a printer is to be used.

The word *graphics* refers to drawings and pictures. The Color/Graphics Monitor Adapter provides highly detailed graphics; the Monochrome Display and Parallel Printer Adapter is oriented towards text rather than drawings; it provides only limited graphics that are crude in comparison with those available with the Color/Graphics Monitor Adapter.

We can think of the characters on the display as arranged in columns; the first character on each line is in column 1, the second character is in column 2, and so on. If each displayed line can contain up to 40 characters, we speak of a *40-column display*. If each line can contain up to 80 characters, we speak of an *80-column display*. The IBM PC can be set to display either 40 or 80 columns. A 40-column display allows a home television set or a low-resolution television monitor to be used as the display device. Most IBM PCs, however, are equipped with the high-resolution monochrome or color monitors needed for an 80-column display.

The Keyboard

The IBM PC keyboard is divided into three parts. On the left are 10 function keys that can be used for giving commands to programs. Each function key causes the program to

*The color of "white" varies with the display device and is usually either green or amber.

carry out a different operation, so we can control a program by pressing function keys much as we might control a machine by pressing a different button for each thing we want the machine to do.

The middle portion of the keyboard is very similar to a standard typewriter keyboard, but with some additional keys needed in many computer applications. Because the IBM PC keyboard is more crowded than its typewriter counterpart, and because a few of the keys are smaller, touch typists may find that they tend to make errors at first. (The left-hand shift key seems to be particularly troublesome; one tends to hit the adjacent backslash key, \, instead.) After a short time, however, you should get the feel of the IBM PC keyboard, after which you will probably type faster and more accurately on it than on many typewriter keyboards.

The right-hand part of the keyboard contains 15 special keys that serve two purposes. They can be used for editing—for correcting mistakes in programs and data—and 10 of them can be used as a calculator-like numeric keypad, allowing numbers to be punched in as with a calculator rather than typed on the top row of keys on the typewriter part of the keyboard.

The two most important keys are the Enter key and the backspace key. The Enter key, which is marked with an arrow that goes down and to the left, is on the right-hand side of the typewriter part of the keyboard, where the carriage-return key would be on an electric typewriter. You must press the Enter key after each command, program line, or data entry before the computer will accept the input you have typed.

Up until the time you press the Enter key, you can correct the line you are typing by using the backspace key, which is just above the Enter key and marked with an arrow pointing to the left. Pressing the backspace key once erases the last character typed; pressing it again erases the next-to-last character typed, and so on. Like most other keys, the backspace key repeats automatically when held down, so you can erase as many characters as you like by holding the backspace key down long enough. Be careful not to hold it down too long, however, and erase something you didn't mean to.

The two shift keys, each marked with an arrow pointing upward, work as on a typewriter—you hold down a shift key to type a capital letter or the upper character on a key marked with two characters. The Caps Lock key reverses the effect of the shift keys for capital and small letters. After pressing Caps Lock, letters typed without holding down the shift key are capital letters and those typed with the shift key down are small letters. Pressing Caps Lock again restores normal operation of the shift keys. Note that Caps Lock affects letters only. For keys marked with two characters, you must always hold down a shift key to type the upper character, even if the Caps Lock key has been pressed.

SOFTWARE

In order to write and execute BASIC programs, we need two pieces of software: a Disk Operating System for communicating with input, output, and storage devices and a BASIC interpreter to accept and execute BASIC programs.

The Disk Operating System

Early computers were operated by means of elaborate control panels. To get the computer to execute a particular program, you would go to the filing cabinet, get the deck of punched cards or the punched paper tape containing the program, insert it in the card or tape reader, and manipulate the switches on the control panel to get the program loaded into main memory and executed.

Eventually, computer designers discovered that a computer could operate itself to a large degree, allowing the complex control panel to be eliminated. To get the computer to execute a particular program, for instance, you could just enter the name of the program and leave it up to the computer to load the program into main memory and execute it.

The advent of more complex forms of auxiliary storage, such as disks, also made computer control essential. We could manually select the deck of cards or the paper tape containing a particular program and insert it in the card or tape reader. If several programs were punched on one tape, we had little trouble positioning the tape so that the proper program would be read. But the only way to find a program or data file on a disk is to consult the directory recorded on the disk, which only the computer can read. And only the computer can position the reading mechanism with sufficient precision to read one program or data file out of the many usually recorded on a disk.

A computer requires a program for every job it does, and the job of operating the computer system is no exception. This job is done by a program called the Disk Operating System or DOS. The word "Disk" emphasizes that a prime responsibility of the DOS (although not the only one) is to store programs and data files on disks and retrieve them upon request.

Three versions of the DOS are available for the IBM PC. The earliest of these, version 1.00, is no longer sold but some early purchasers of IBM PCs may still be using it. Version 1.10 contains improvements over version 1.00 and is currently the most widely used version. Version 2.00 contains further improvements over 1.10, particularly those needed to use fixed disks. Almost everything in this book applies equally well to all three versions.

Loading the DOS

The DOS has the responsibility of loading programs, but the DOS itself has to somehow be loaded when the computer is first turned on. This job is initiated by a short program in ROM, which loads a more elaborate loading program from the disk. The more elaborate loading program loads the DOS and transfers control to it. Thinking of the phrase "lifting yourself by your own bootstraps," people often refer to the loading program in ROM as a *bootstrap loader,* since it is the "bootstrap" by which the entire system is "brought up" or started. (The loading program that is read from the disk is also sometimes called a bootstrap loader.) The process of starting up a computer and loading the DOS is sometimes referred to as "booting the system," a phrase that must seem peculiar to those unaware of its origin.