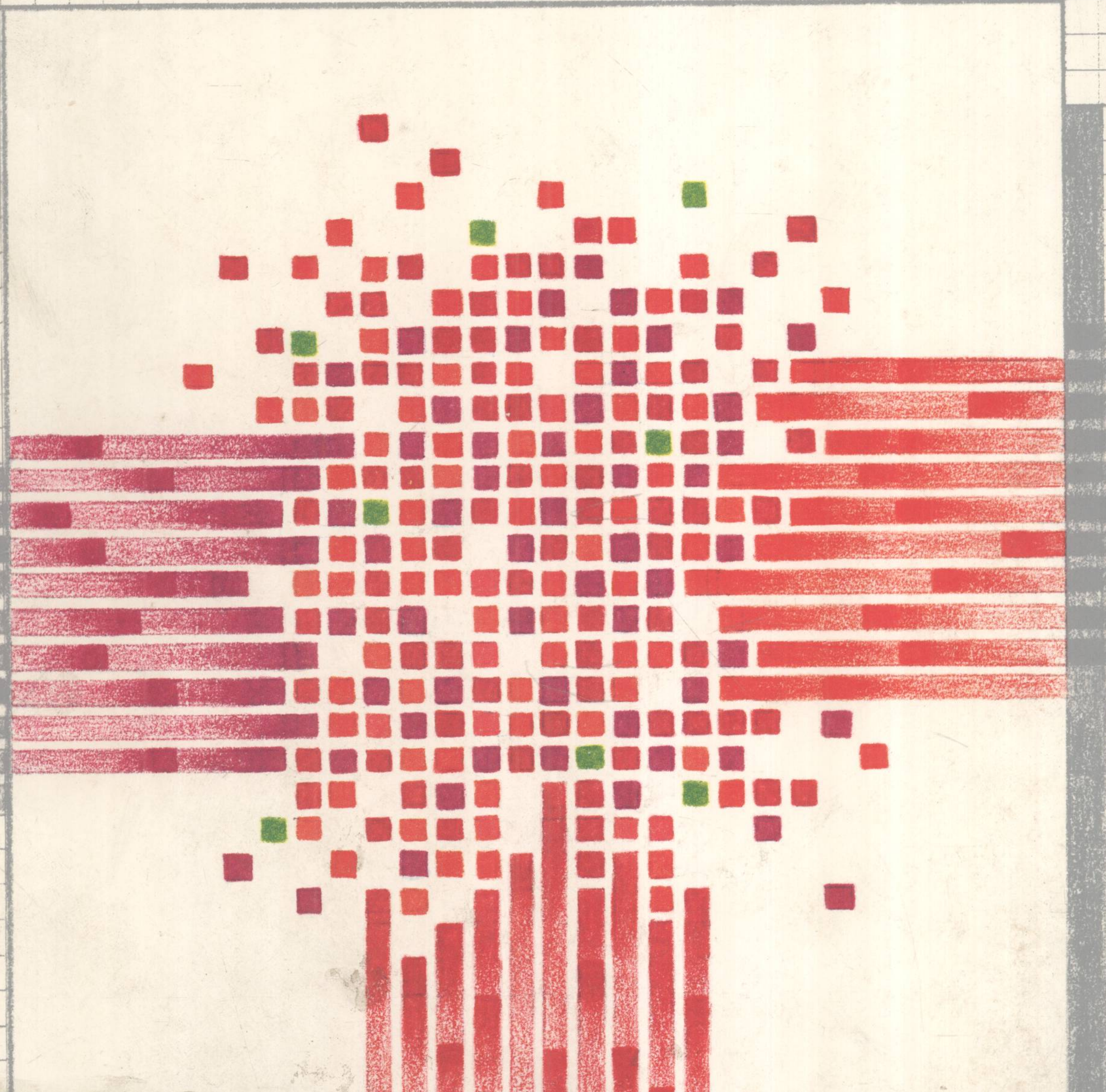
PROGRAMMING FOR MICROCOMPUTERS

Apple II BASIC

JUNE GRANT SHANE



TP312.3
S2

HOUGHTON MIFFLIN

TP312.3
S2

8566047

Programming for Microcomputers

Apple II BASIC

June Grant Shane
Indiana University

华南工学院
★
图书馆藏书

E8566047

HOUGHTON MIFFLIN COMPANY BOSTON
Dallas Geneva, Illinois Hopewell, New Jersey Palo Alto

To:

Leo C. Fay and Carl B. Smith

Apple is a registered trademark of Apple Computer Inc. This text is based on the use of a 48K Autostart Apple II with Applesoft in ROM. The list of reserved words and the list of error messages found in the appendices are reproduced through the courtesy of Apple Computer Inc.

Cover design and illustration by Ellen Clancy.

Copyright © 1983 by Houghton Mifflin Company. All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, or by any information storage or retrieval system, except as may be expressly permitted by the 1976 Copyright Act or in writing by the Publisher. Requests for permission should be addressed in writing to Permissions, Houghton Mifflin Company, One Beacon Street, Boston, Massachusetts 02108.

Printed in the U.S.A.

Library of Congress Catalog Card Number: 83-81563

ISBN: 0-395-35206-1

Programming for Microcomputers

Apple II BASIC

Preface

Are you interested in personal computers? Do you want to write your own computer programs? Are you looking for a book with nontechnical language, clear explanations, and good program models? If your answer is yes, then *Programming for Microcomputers: Apple II BASIC* is the book for you.

This book has been designed for a wide range of potential users, including students, professionals, and other individuals who are interested in learning to understand and apply a new technology. The text teaches you how to write computer programs in BASIC, which is a computer language developed expressly for people who are just learning to program. BASIC is the standard language for home computers and is also widely used in educational settings. Likewise, the Apple® II microcomputers have become popular choices for use in homes, schools, and offices.

The text can be used independently, in computer programming classes, or in computer literacy courses. It is designed for situations in which you have direct access to a computer so that you can enter the program examples featured in the text and see the results immediately. You move sequentially, one step at a time, from simple to increasingly complicated programs. You begin by writing short programs and finish with long, well-documented programs.

As the computer programs become longer and more time consuming to write, you will see the need for developing good planning skills. To help you in this planning process, the text uses a block structure and a top-down design that lets you break down a problem into smaller, more manageable parts. Grid sheets are used extensively to assist in the planning process. Their use to design and code your programs will allow you to minimize the amount of time needed to work at the computer. Moreover, the grid sheets will provide you with a permanent record of your program's development.

此为试读，需要完整PDF请访问：www.ertongbook.com

Special features

Programming for Microcomputers: Apple II BASIC is organized in an easy-to-follow format to help you learn new and exciting things and to encourage you to become a knowledgeable advocate of structured programming with top-down design, coding, and testing. The text has successfully undergone extensive field testing with junior high and secondary school students, undergraduate and graduate college students, teachers, librarians, and other professional groups.

Following is a list of the text's special features:

1. The text follows a sequential, step-by-step presentation, moving from simple to more complex programs.
2. Program examples are carefully explained and easy to follow.
3. Charts and other illustrations of different processes are included.
4. An outline of major topics appears on the first page of each chapter.
5. Each chapter begins with the specific learning objectives to be covered.
6. Each chapter ends with self-test activities for applying material that has been learned.
7. Solutions to all problems and programs are included in the text.
8. A glossary of special terms is provided.
9. Programs are shown in the form in which they are to be entered, not in LIST, or computer printout, format.
10. Structured programming using block structure and a top-down approach to program design, construction, and testing is explained and demonstrated.
11. The book is written in nontechnical language and features a highly readable style.
12. A *Grid Sheet Booklet*, which is available separately, provides a supply of removable grid sheets on which programs can be planned.

Acknowledgments

The advice and help given by Melanie Davis, Leo Fay, Eric Mielke, Sue Sanford, Carl Smith, and Dorothy Winchester, all of Indiana University, have been greatly appreciated.

June Grant Shane

Contents

1 Introduction

When computers were large, expensive machines, operating them seemed both mysterious and complex. People thought programmers were mathematical Merlins and that programming was a black art, something you and I would never understand. Yet, these machines changed our lives through their tremendous ability to perform calculations and process information.

Engineers and architects now use computers as they design and build aircraft, bridges, and buildings. Computers help medical doctors and other health professionals diagnose and treat health problems. Insurance companies, banks, newspaper and publishing companies, government agencies, and educational institutions use them to process large amounts of data efficiently. Computers help our scientists and astronauts explore outer space; our airlines check reservations; and our fast-food chains provide just what you ordered on your hamburger.

Whether you plan a career as an engineer or teacher, as a secretary, lawyer, or mechanic, chances are you'll find a computer in your future. In

此为试读，需要完整PDF请访问：www.ertongbook.com

fact, the fastest growing job market is the computer field itself. If you know how to run, operate, or fix a computer, your chances of finding a job are better than most.

Work isn't the only part of our lives computers are changing. Today, thousands and thousands of small, personal computers are in our homes. People use these machines in different ways. They use them to control home heating, get current stock market reports, make travel reservations, monitor phone calls, provide electronic mail service, play games, and improve learning skills. People also write **computer programs**—the instructions that make the computer do what they want it to do. When people write programs, they're using the computer to help them define and solve their own problems.

Writing programs is one of the most powerful ways to use a computer. It's also the best way to understand the computer and to prepare yourself for life in our computerized world. Writing programs is what this book is about.

You and computer programming

People of all ages, from first graders to retirees, are learning to program. You can, too! You don't have to be Merlin, the magician, or a mathematical wizard, or an expert typist to learn how to program. Nor do you have to understand the inner workings of the computer. Most of us learn to drive a car without knowing what goes on under the hood. We also learn the traffic rules we must follow. Well, the computer is also a machine—a machine for handling information. And you must follow rules to operate it successfully. These rules are easy to learn, and this book is here to help you.

Programming for Microcomputers provides a friendly, hands-on approach to learning how to write computer programs. It teaches you BASIC, the standard language for home computers. BASIC is an acronym for Beginner's All-purpose Symbolic Instruction Code. It was developed at Dartmouth College by John Kemeny and Thomas Kurtz. Kemeny and Kurtz developed BASIC expressly for people like you—people just learning to program.

BASIC is a very popular language, and many versions of it exist today. This book uses the Applesoft version of BASIC, which will run on your Apple II, II Plus, or IIe computer. Applesoft BASIC shares many features with other versions of BASIC, so even if your computer doesn't use Applesoft, you should be able to use this book. The concepts it teaches apply to any version of BASIC. Specific commands will differ, however. Most of the differences concern edit and format commands. Check the manual that came with your version of BASIC to find equivalent commands to those in Applesoft.

Like BASIC, *Programming for Microcomputers* is designed for you, the beginner who wants to tap the full power of the computer. Beginning programmers of all ages have used it successfully. Several special features make this book easy to use:

1. The book progresses, as does the computer, step-by-step. It takes you from the simple through the more complex steps involved in writing programs. You begin with short programs and finish with long, well-documented ones. The contents, objectives, and special terms at the beginning of each chapter clearly outline what you will learn; the self-test at the end of each chapter lets you apply your new skills. You may check your work against the answers in the back of the book. A glossary and an index, also at the end of the book, help you find concepts or terms with ease.
2. Short programs illustrate every concept you learn. This is a book to read and use *at* the computer. When you enter the sample programs, you reinforce what you are learning. You gain the deep understanding that only first-hand experience can provide. These exercises prepare you for writing your own programs.
3. The book explains the concepts underlying programming. But these explanations, like the rest of the book, are nontechnical. They are written so that you can easily grasp important ideas.
4. *Programming for Microcomputers* provides a detailed, structured approach to planning your programs—a vital, but often overlooked, part of the programming process. It provides special Program Planning and Output Grids. You'll see how these grids will save time as you plan and write your programs.
5. Finally, even the physical design of *Programming for Micro-computers* makes the book easy to use. We used a spiral binding so that the book would stay open to the page you want. And we decided to print program listings in regular type rather than computer print-out because regular type is easier to read. Each program line contains exactly as many characters as it will when you type it into your Apple, even though the lines look a little different because of the regular type. If a word is broken in a strange place, you'll notice that the line ends at forty characters, the total number that fits across the monitor of most Apple II's.

We think that you will enjoy *Programming for Microcomputers*, and that you will find programming exciting and challenging. Here's what you will learn: first, you will learn about the keyboard and the **syntax**, or rules, for writing the commands and statements that make up programs. Then, you will write short programs and learn more BASIC commands. You will learn exactly what to **input** (type on the computer keyboard) to have the **output** (what appears on the screen) appear just as you want it.

Once you have a good understanding of BASIC, you will begin to plan programs using a **top-down** approach. This approach lets you break a problem into smaller and smaller parts until all are easy to program. You will also learn to use the grid sheets to plan just how you want your material to look on the screen.

Later, you will use a top-down approach to **coding**—translating what you have planned into lines of BASIC the computer can understand. You will also use a top-down approach to test and **debug** (remove errors from) your program. By the time you get this far, you will be a bonafide programmer. And being a programmer has more benefits than the obvious one of getting the computer to do what you want. Programming may help you to define problems better and to think logically. You will learn how to plan, organize, and evaluate things in a more systematic manner.

The computer

Before we get started, a word about the computer. The computer is a fantastic machine for processing information. But, as we said before, it is still just a machine—a tool for handling information. It is simply an assembly of electronic switches that can be arranged in many patterns. Everything the computer does is based on the fact that a switch can be turned on or off! The computer responds only to specific instructions that are written following preset rules. The instructions are the programs you write, and the rules are the syntax of the programming language (in our case, BASIC).

No matter what microcomputer you are using, it will have a **central processing system**, a way of getting information into the system, a way of getting information out of the system, and a means of storing information. The Apple takes care of these functions with three pieces of equipment. The central processing system is housed in a box that looks like a typewriter. The keyboard lets you type in your instructions. This box connects to a monitor that has a **cathode ray tube**, which looks like a television screen. Your output appears on this screen. The box also connects to a **disk drive**, which stores information.

That's all there is to it! You can see already that no Merlin-like powers are necessary here—only curiosity and the desire to learn. So welcome, and let's get started!

2

Getting Started

Objectives

In this chapter you will learn how to:

- ▶ Use the keyboard to have certain functions performed
- ▶ Write commands in immediate and deferred execution
- ▶ Distinguish between alphanumeric expressions, or character strings, and numeric expressions
- ▶ Write statements following the required syntax
- ▶ Write a short program using the required syntax
- ▶ Recognize the difference in format as a program is entered, listed, and executed
- ▶ Write an algorithm
- ▶ Draw a flowchart to visualize an algorithm
- ▶ Use the following commands:
  - System commands: NEW, LIST, RUN, END, CTRL C
  - Input/output commands: PRINT
  - Format commands: HOME

Special terms

Algorithm	Left arrow key
Alphanumeric expression	Line number
Character string	Loop
Command	Monitor
Control key	Numeric expression
CRT	Process symbol
Cursor	Prompt
Debugging	Repeat key
Decision symbol	RESET
Deferred execution	RETURN
Error message	Right arrow key
Execute	Space bar
Flowchart	Statement
Immediate execution	Syntax
Insert	System comand
Keyword	Terminal symbol

Keys and their functions

The computer keyboard resembles a typewriter keyboard in many ways. As Figures 2.1 and 2.2 show, the letters of the alphabet are in their usual positions on the third, home, and bottom lines. The **RETURN key**, which is equivalent to the carriage return key on an electric typewriter, is similarly familiar. This is the key you will use most frequently. You must press it after each line that you type. Pressing RETURN signals the computer that you have completed an instruction. It also returns the **cursor** (the blinking square) to the left edge of the screen.

The **space bar,** another familiar key, functions just like a typewriter space bar. Similarly, the two shift keys let you type the top character on two-character keys. Some of these upper characters differ from those on a regular typewriter, however.

The Apple keyboard differs from that of a typewriter in other ways as well. Unless you have an Apple IIe or have modified your Apple II/II+, you can type only upper-case letters. This means you cannot use the lower-case letter L for the number *1*. The number keys are on the top row of the keyboard. Notice that the zero symbol, Ø, is slashed to distinguish it from the upper-case letter *O*.

The computer keyboard also contains unique keys. Some of these, such as the **repeat key**, enhance traditional operations. Others are essential for writing programs or for using commercial computer software. The special keys are:

← The **left arrow key** backspaces. It moves the cursor to the left.

→ The **right arrow key** moves the cursor to the right. It acts as a retype key as it moves the cursor over existing letters.

REPT When you press a key and hold down the repeat key at the same time, the character appears repeatedly.

CTRL The **control key** does not put characters on the screen. It instructs the computer to perform certain functions.

CTRL C Pressing control and C at the same time, or control and **reset** at the same time, interrupts a program.

此为试读，需要完整PDF请访问：www.ertongbook.com