SECOND EDITION

Fundamentals of FORTRAN Programming

Robert C. Nickerson

FUNDAMENTALS OF FORTRAN PROGRAMMING

Second Edition

Robert C. Nickerson
San Francisco State University

WINTHROP PUBLISHERS, INC.
Cambridge, Massachusetts

Library of Congress Cataloging in Publication Data

Nickerson, Robert C.
Fundamentals of FORTRAN programming.

Includes index.
1. FORTRAN (Computer program language) I. Title.
QA76.73.F25N49 1979 001.6'424 79-19423
ISBN 0-87626-301-5

© 1980, 1975 by Winthrop Publishers, Inc.
17 Dunster Street, Cambridge, Massachusetts 02138

All rights reserved. No part of this book may be reproduced in any form or by any means without permission in writing from the publishers. Printed in the United States of America.

10 9 8 7 6 5 4 3 2 1

PREFACE

The major objective of the second edition of this text remains the same as that of the first edition; that is, to provide a carefully paced introduction to computer programming and the FORTRAN language for students with only a minimal mathematical background. However, with this edition much of the material has been updated to reflect changes in the FORTRAN language and to incorporate current thinking on programming methodology. The revised text covers not only the 1966 versions of ANS FORTRAN but also additional features of FORTRAN 77 (the 1978 version of ANS FORTRAN) and some features of WATFIV-S. This new material has been presented in such a way that it may be covered or omitted at the discretion of the instructor. These topics include format-free I/O, block IF statements, WHILE loops, character data processing, and other features.

In addition to the new language elements, structured programming methodology has been incorporated into the text. The problem-solving process is emphasized from the beginning. Basic program structure is discussed early with complete chapters devoted to decision logic (Chapter 5) and loop control (Chapter 6). The programming process is discussed in detail in Chapter 8 where program development, structure, style, testing, and documentation are examined. The use of subprograms in developing large programs is explained in Chapter 12. The overall approach is to develop in the student the ability to think systematically through the solution to a problem and to implement the solution correctly and understandably in FORTRAN.

A number of other features of the text have been added or revised with this edition. These include the following:

- Both batch and interactive (time-sharing) systems and programming features are discussed. This facilitates the use of the text with either type of system.
- Chapter 3 covers both formatted and format-free I/O. The instructor need only select those sections that apply to the I/O system being used. In general, subsequent chapters are not dependent upon a knowledge of one or the other system.

- The basic I/O discussed in Chapter 3 is the minimum that is necessary for simple numeric input and output. Additional features, including nonnumeric I/O, are presented in a later chapter (Chapter 7), which may be covered at any time after Chapter 3.
- More examples and illustrative programs are included in this edition. As with the first edition, the examples are nonmathematical in nature and are oriented toward applications that the students can readily understand.
- Additional algorithms are discussed including algorithms for logical and character data processing and for array manipulation (e.g., searching and sorting).
- The number of programming problems has been substantially increased. Most problems still require only a minimal mathematical background and emphasize nontechnical areas including business and social science. In addition, a number of problems have been added that will be of interest to math, science, and engineering students.
- Flowcharting is now covered in an appendix rather than in the main text. This allows the instructor to discuss this material at the time that he or she feels is most appropriate. All flowcharts are keyed to programs in the text.

The text is organized into three parts. The first part, consisting of Chapters 1 and 2, introduces the background material necessary to understand programming and FORTRAN. Chapters 3 through 8, comprising the second part, present the fundamental elements of FORTRAN and develop basic programming methodology. Advanced FORTRAN features and programming are discussed in the third part, which consists of Chapters 9 through 13. The five appendices cover differences between various versions of FORTRAN, FORTRAN-supplied functions, the operation of the IBM 029 keypunch, exponential forms, and flowcharting.

Chapters 1, 2, and 13 contain review questions and computer exercises that emphasize important topics in these chapters. Programming problems are provided at the end of Chapter 3 through 12. The problems range in difficulty from relatively easy to very difficult and challenging. Test data is provided with most problems. An instructor's manual is available that contains teaching suggestions, alternative course schedules, answers to review questions, test questions and answers, and overhead transparency masters for a number of illustrations from the text.

Many of the ideas for the revisions in the text came from reviews by users of the first edition. I am grateful for their thoughtful com-

此为试读,需要完整PDF请访问: www.ertongbook.com

ments and suggestions. I would also like to acknowledge the contribution of my students who provided much feedback as I class-tested the text material.

Finally, I would like to thank my wife, Betsy. Not only did she turn my often illegible scratchings into a typed manuscript, but she also served as editor, critic, and friend throughout the long writing process.

R. C. N.

CONTENTS

Chapter 1

COMPUTERS, PROGRAMS, AND DATA

Computers are devices that are used to solve problems. The process that people go through to prepare a computer to solve a problem is called programming. In this book we explain a particular type of computer programming. However, before we can understand the programming process in detail we must be familiar with a few background ideas. In this chapter we discuss the basic concepts that are necessary to begin studying programming.

1-1. COMPUTERS

We can call any calculating device a "computer." For example, adding machines, slide rules, and pocket calculators are all "computers" because each calculates or computes. However, we usually do not use the word "computer" for these devices. Instead we call something a *computer* if it has three distinguishing characteristics.

First, a computer is electronic; that is, it calculates by electrical means. We aren't concerned with how this is done in this book but there is an important consequence of this characteristic. Because computers calculate by electrical means, they can perform their calculations at a very high speed. Thus, modern computers are able to do hundreds of thousands of operations (such as additions and subtractions) in a second.

The second distinguishing characteristic of computers is that they have the ability to "remember" things. We call the things that a com-

puter can remember *data*. Data† is facts, figures, numbers, and words that are important to the problem being solved by the computer. A computer can remember, or, more correctly, *store* data for immediate use or for use in the future.

The final characteristic that distinguishes a computer from other calculating devices is its ability to receive and follow a set of instructions that tells it how to solve a problem. Such a set of instructions is called a *program*. The program is prepared by a person, called a *programmer*, who is familiar with the different things that a computer can do. Once the program is prepared it is given to the computer in a form that the computer can understand and *stored* along with the data to be used by the program. Then the instructions in the program are performed, or *executed*, automatically by the computer.

In this book we are concerned with the process of preparing computer programs. The preparation process is called *programming* and this book explains one commonly used way of programming computers. However, before we can describe programming in detail we need to have an understanding of the physical organization of a computer.

1-2. ORGANIZATION OF A COMPUTER

As we have seen, computers are distinguished from other calculating devices by the electronic, data storage, and stored program characteristics explained in the previous section. However, the physical organization of a computer is more complex than we might think from this discussion. One way of viewing the organization of a computer is shown in Fig. 1-1. In this diagram, boxes represent different components of the computer and lines with arrowheads show the paths that data and instructions can take within the computer. In this section we describe each of the computer components shown in Fig. 1-1. Figures 1-2 and 1-3 show actual computers with many of the components discussed in this section.

There are three basic parts to any computer—an input device, a central processing unit, and an output device. An *input device* is a mechanism that accepts data in some form from outside of the computer and transforms the data into an electronic form that is understandable to the computer. The data that is accepted is called *input data*, or simply *input*. For example, one of the most common forms for conveying input data to a computer is punched cards. An example

†The word *data* is most correctly used as a plural noun. The singular of data is *datum*. However, people commonly use the word data in a singular rather than plural sense. We will usually follow the common practice in this book.

此为试读,需要完整PDF请访问: www.ertongbook.com

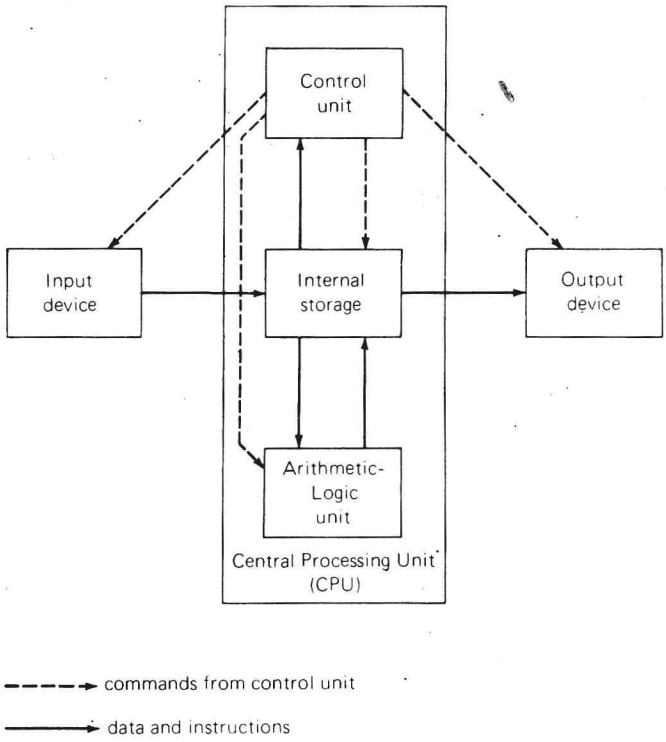

FIGURE 1-1 Organization of a computer.

of a punched card is shown in Fig. 1-4. Data is recorded on the card by punching various patterns of holes in the cards. (We will describe punched card data in detail in a later section of this chapter.) An input device for punched cards is designed to recognize what data is represented by the patterns of holes in a card and to transform the data into an electronic representation understandable to the central processing unit. Such a device is called a *card reader*. (See Fig. 1-2.)

An *output device* performs the opposite function of an input device. An output device receives data in an electronic form from the central processing unit and then transforms the data into a form that can be used outside of the computer. The resulting data is the *output data*, or simply the *output*, from the computer. For example, one common form of output is a printed document or report. Figure 1-5 shows an example of such printed output. Data from the central processing unit is transmitted electronically to a device called a *printer*. (See Fig. 1-2.) In the printer, the data is transformed into printed symbols and a paper copy of the output data is produced.

Instead of punching input data into cards and reading the data with

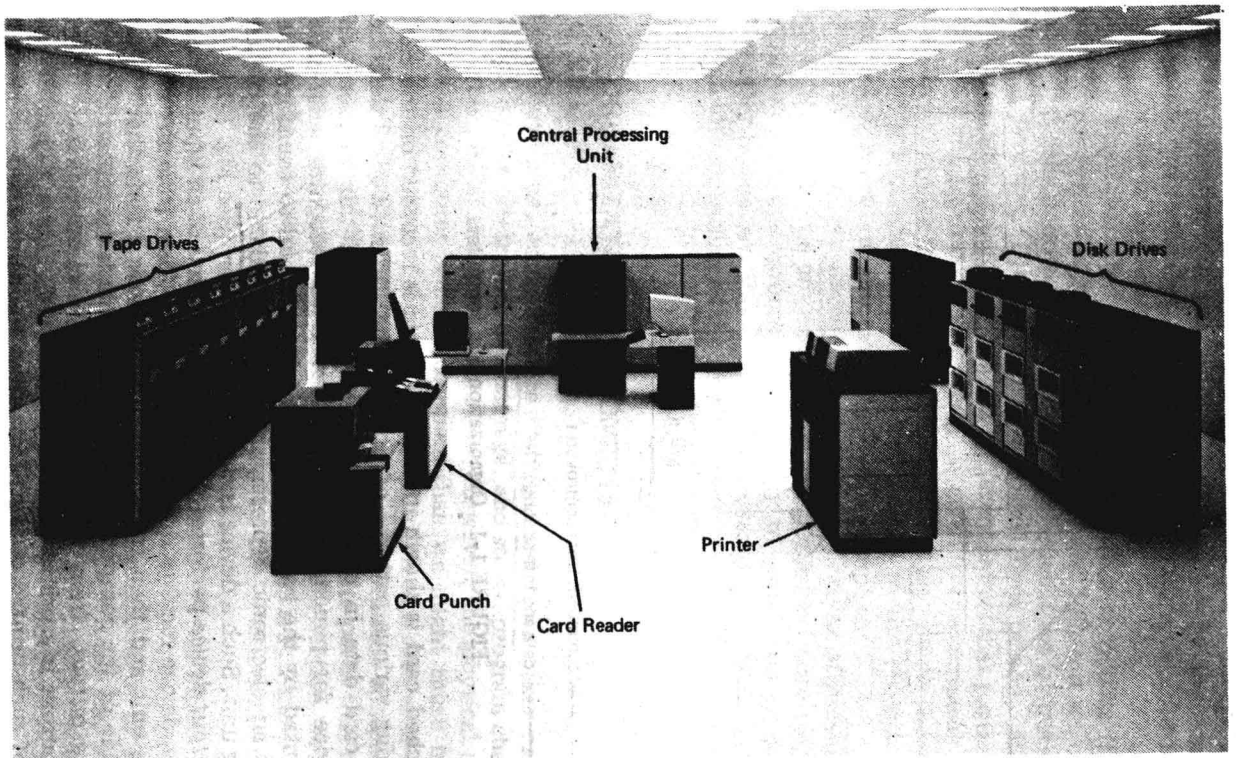

FIGURE 1-2 A typical batch computer system. This is an IBM System/370. (Courtesy of IBM Corp.)

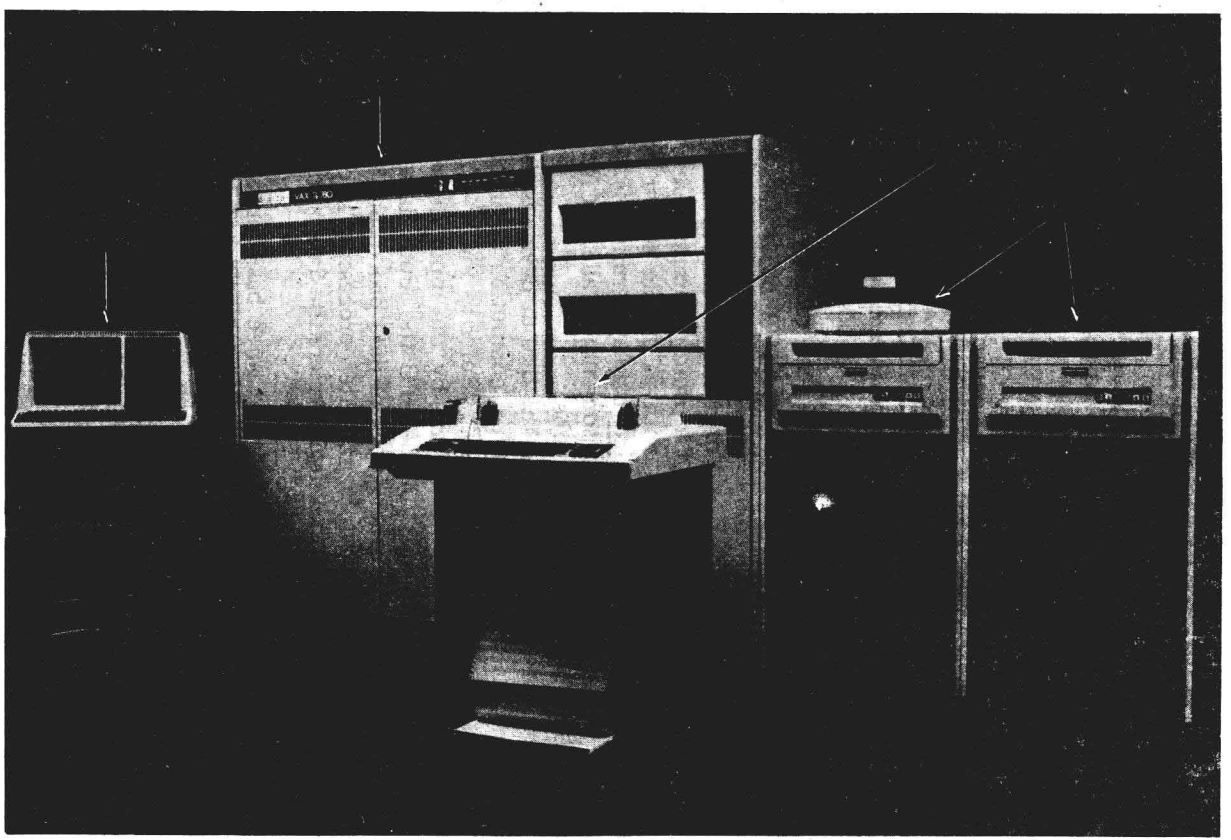

FIGURE 1-3 A typical interactive (time-sharing) computer system. This is a DEC VAX 11/780. (Courtesy of Digital Equipment Corp.)

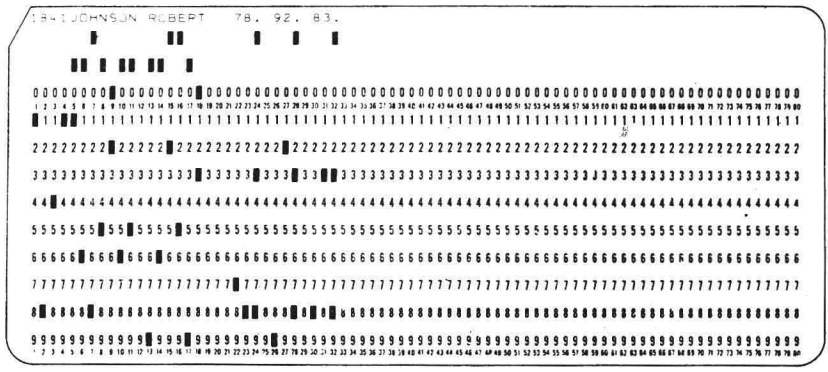

FIGURE 1-4 Punched card data.

a card reader, input is sometimes typed directly into the computer using a typewriter-like keyboard. Often a printing mechanism is attached to the keyboard and the computer uses this to print the output. Such a combined input/output (or *I/O*) device is commonly called a *terminal*. Figure 1-3 shows an example of a printing terminal. Another type of terminal combines a keyboard with a TV-like screen on which the output is displayed. Such a video display terminal is also shown in Fig. 1-3. This type of terminal is sometimes called a *CRT*, for cathode ray tube (another name for a TV tube).

Between the input device and the output device is the component of the computer that does the actual computing or processing. This is the *central processing unit* or *CPU*. (Refer to Fig. 1-1.) Input data is transformed into an electronic form by the input device and sent to the central processing unit. In the CPU the input data is stored and used in calculations or other types of data processing to produce the solution to the desired problem. After processing is completed, the results are sent to the output device where the data is transformed into the final output.

In the central processing unit are three basic units that work to-

TEST SCORE ANALYSIS

NUMBER	STUDENT NAME	TEST 1	TEST 2	TEST 3	TOTAL	AVERAGE
1841	JOHNSON ROBERT	78.	92.	83.	253.	84.3
1906	SMITH MARY	100.	95.	97.	292.	97.3
2133	ANDERSON RICHARD	65.	72.	57.	194.	64.7
2784	WILSON ALEX	73.	69.	78.	220.	73.3
2895	BOYD DEAN	42.	56.	47.	145.	48.3
3047	EMERY ELIZABETH	91.	100.	92.	283.	94.3
3260	COLE JAMES	75.	78.	73.	226.	75.3
3335	GUINN DOROTHY	86.	82.	74.	242.	80.7
3819	JONES ED	71.	85.	78.	234.	78.0

FIGURE 1-5 Printed output.

gether to perform the functions of the CPU. These are the internal storage, the arithmetic-logic unit, and the control unit.

The *internal storage* is the "memory" of the computer. It is in this device that data for processing is stored. In addition, instructions that make up the program are stored in the internal storage. The internal storage of a computer is not the same as human memory. However, like human memory, data can be put into the computer's internal storage and then recalled at some time in the future.

The *arithmetic-logic unit* performs arithmetic and logical operations. Computers can do the basic arithmetic tasks that a human can do. That is, a computer can add, subtract, multiply, and divide. However, more complicated calculations, such as finding the square root of a number, usually are not built into the computer. Instead, such complex processing must be performed by an appropriate sequence of basic arithmetic operations. The logical operations that a computer can do usually are limited to comparing two numbers to determine if they are equal or if one is greater than or less than the other. Complex logical processing, as is required in a computer that plays a game such as chess, is accomplished by long sequences of these simple operations.

The final unit in the CPU is the *control unit.* The function of the control unit is to tell the other units in the computer what to do. It does this by following the instructions in the program. The program, as we have seen, is kept in the computer's internal storage. During processing, each instruction in the program is brought from the internal storage to the control unit. The control unit analyzes the instruction, then gives commands to the other units based on what the instruction tells the control unit to do. The execution of one instruction may involve actions in any of the other components of the computer. (See Fig. 1-1.) After executing an instruction, the next instruction in the programmed sequence is brought to the control unit and executed. This continues until all the instructions in the program have been executed.

As an example of a simple computer program, let's assume that we wish to solve the problem of finding the sum of two numbers using a computer. To do this the computer must get the two numbers to be added from the input device, then the numbers must be added using the arithmetic-logic unit, and finally the sum must be sent to the output device so that we can see the result. Thus, a computer program to solve this problem would involve three instructions:

1. Read two numbers.
2. Add the numbers.
3. Write the result.

The programmer would enter these instructions into the computer using the computer's input device. For example, the instructions

may be punched in cards just as input data is punched in cards and read with a card reader. Or the instructions could be keyed into a terminal. The input device would then transfer the instructions to the computer's internal storage.

On a signal to start (perhaps from the person operating the computer) the first instruction in the program would be brought from the internal storage to the control unit. Execution of the instructions would then proceed as follows:

1. Read two numbers. The control unit examines this instruction and issues commands to the other units that cause the instruction to be executed. For this instruction, the control unit issues a command to the input device that causes the two numbers to be transferred from the input device to the internal storage. The second instruction is then brought to the control unit.
2. Add the numbers. This instruction causes the control unit to issue three commands. The first, to the internal storage, causes the two numbers to be sent to the arithmetic-logic unit. Then a command is given to the arithmetic-logic unit to add the two numbers. Finally, the arithmetic-logic unit is commanded to return the result to the internal storage. The last instruction is then brought to the control unit.
3. Write the result. This instruction causes the control unit to issue a command to the internal storage to send the result to the output device. Then a command is given to the output device to write the result.

So far the only input/output devices that we have described are card readers, printers, and terminals. Many computers use other types of I/O devices. Usually several input devices and several output devices are attached to the computer at one time.

A common output device is the ***card punch***. (See Fig. 1-2.) When this unit is used for output, the results from computer processing are punched into cards. The punched card output might then be used for input to another program.

Other forms of input and output are magnetic tape and magnetic disk. *Magnetic tape* is much like tape recorder tape; data is recorded on the surface of the tape by patterns of magnetic spots. A *magnetic tape drive* is a device that records data on magnetic tape and also retrieves data from the tape (Fig. 1-2). A *magnetic disk* resembles a phonograph record. However, like tape, the disk surface records data by magnetic spot patterns. A *magnetic disk drive* is a device for recording data on magnetic disks and for retrieving data from disks. (See Figs. 1-2 and 1-3.)

Tape and disk drives are both input and output devices; that is,

此为试读,需要完整PDF请访问: www.ertongbook.com