FUNDAMENTAL COMPUTER PROGRAMMING USING

FORTRAN 77

JARRELL
C.
GROUT

FUNDAMENTAL COMPUTER PROGRAMMING USING FORTRAN 77

Jarrell C. Grout
Department of Computer Science
Stephen F. Austin State University

Prentice-Hall, Inc. Englewood Cliffs, N.J. 07632

Library of Congress Cataloging in Publication Data

GROUT, JARRELL C.

Fundamental computer programming using Fortran 77.

Includes index.

1. FORTRAN (computer program language) 2. Electronic digital computers—Programming. I. Title.

QA76.73.F25G76 1983 001.64'24 82-20421

ISBN 0-13-335141-6

Editorial/production supervision and interior design: *Aliza Greenblatt*

Cover design: *Jeannette Jacobs*

Manufacturing buyer: *Gordon Osbourne*

". . . to Him who is able to do exceeding abundantly beyond all that we ask or think, according to the power that works within us . . ."

(Eph. 3:20 NASB)

The scripture quotation used in the dedication is from the *New American Standard Bible*, © The Lockman Foundation 1960, 1962, 1963, 1971, 1972, 1973, 1975. Used by permission.

Printed in the United States of America

10 9 8 7 6 5 4 3

ISBN 0-13-335141-6

PRENTICE-HALL INTERNATIONAL, INC., *London*
PRENTICE-HALL OF AUSTRALIA PTY. LIMITED, *Sydney*
EDITORA PRENTICE-HALL DO BRASIL, LTDA., *Rio de Janeiro*
PRENTICE-HALL CANADA INC., *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*

FUNDAMENTAL COMPUTER PROGRAMMING USING FORTRAN 77

Preface

This book is for people who want to learn computer programming and the FORTRAN 77 programming language. Its primary objective is to present the fundamental methodology of contemporary computer programming in a manner that is understandable and useful to learners. FORTRAN 77, the up-to-date standard version of the widely used Fortran language, is throughly described and illustrated to provide the means for applying the components of the methodology and practicing with them. The book meets the requirements of course CS1, Computer Programming I, of the ACM Curriculum '78 (*Communications of the ACM*, March 1979), but it is not limited to use by students of computer science; it is intended for anyone, at any age, who has at least the equivalence of a high school education and a desire to learn fundamental computer programming.

There are a number of features, in addition to readability, which distinguish this text from the other books that address the subjects of programming and FORTRAN 77 together. The major ones are listed and briefly described below.

1. There is a decided emphasis on program planning and problem-solving methods. In Chapter 2, an unusually comprehensive programming guide is presented and used to stress planning; then specific procedures for developing algorithms and program flowcharts are covered in depth. Subsequently, these are employed throughout the text to expand the learner's problem-solving skills. Top-down program design and stepwise refinement of algorithms are introduced in Chapter 2 and weaved into the text appropriately from there on. Chapter 2 is intentionally longer and more explanatory than corresponding chapters in most comparable books; yet toward the end of the chapter, the learner reads, studies, and runs a first simple program.
2. The creation of well-organized Fortran programs is encouraged and facilitated by the presentation, in Chapter 3, of a specific Fortran program organization method that emphasizes modularity at the design level and effective organization of whole programs at the coding level. The goal of the method is to make Fortran programs easier to design, document, read, understand, debug, and maintain—basically, the goal of structured

programming. This book can be used whether or not the method is adopted; its adoption, however, should be thoughtfully considered in the light of current good programming practices.

3. Well-designed input and output (i/o) is advocated, with specific i/o design instruction being given initially in Chapter 6. Printed output, especially, should be readable and understandable. Thus both explicitly formatted and implicitly formatted (i.e., list-directed) i/o are fully covered in the text; explicit formatting is not relegated to an appendix or treated as an afterthought as in many other books.
4. The GO TO problem is met head-on. Learners, and experienced programmers too, need to know when and when not to use the GO TO statement and how to use it properly when they do use it. The relevant instruction, with complete structured programming illustrations, is initiated in Chapter 4.
5. In every chapter except 1 and 10, there is at least one complete example program per chapter—completely planned, completely developed, and completely explained to the extent needed by the learner at that point. The programs are meant to be read and studied. They progress from the simple to the moderately difficult. Furthermore, they illustrate, among others, these fundamental programming techniques: linear and binary search, bubble sort (direct and indirect, numeric and alphabetic), report generation, program modification, program modularization, file building, file merging, and programming for on-line and interactive processing.
6. Programming language coverage is strictly American National Standard FORTRAN, X3.9-1978, commonly known as FORTRAN 77. The coverage is complete for the full language; no extensions are presented.
7. Instructions for using FORTRAN 77 to create good programs are integrated with the language presentation. By a good program, I mean one that is correct, well documented, readable, understandable, maintainable, produced on schedule, and reasonably efficient in terms of storage requirements and execution time.

This book can be used by individuals to learn programming and FORTRAN 77 independently; it can certainly be used to learn the same under a good teacher's guidance. Teachers should consider covering the material in one of the following three ways:

1. If the students have absolutely no background in computing, begin with Chapter 1 and cover most of the material through Chapter 11. The purpose of the first chapter is to introduce beginning students to computer

equipment, different types of computer programs, and the role of data in programming. One noteworthy point about Chapter 1 is that it provides a clear contrast between programs and data, thus alleviating a particular difficulty that beginners often have. If a time constraint occurs while going through the rest of the material, just touch lightly on or even bypass Chapter 10 on multidimensional arrays. The array chapters, 8 and 10, were purposely split to provide this flexibility and to allow the important subject of character data processing (chapter 9) to be considered sooner.

2. If the students have completed an "introduction to computers" course, or the equivalence, cover Chapters 2 through 11.
3. If the students have the background equivalence of an "introduction to computers" course and one programming course of any kind, Cover chapters 2 through 12.

Teachers, students, and independent learners alike, are encouraged to keep a copy of this book; it can serve as a long-term future reference text for computer programming and FORTRAN 77.

ACKNOWLEDGEMENTS

The late Dr. John Q. Hays, a long-time English professor and my friend, read a portion of the first draft and provided a valuable critique of my writing style. My wife Helen also read and critiqued parts of the manuscript for readability and style. My computer science colleagues Dr. John Anderson, Dr. Denis Hyams, Bill Herman, and Fred Fisher reviewed some of the early material and gave helpful recommendations. Dr. Brian Kernighan and Dr. Richard Austing performed technical reviews, providing a number of valuable suggestions. For several years, the students in my introductory computer programming classes participated in formulating the book by using versions of it in draft form. Notable contributions in the form of example program development were made in the early stages by my computer science students Brent Fodor, Chip Galloway, and Tammy Mays. Also, my non-computer science students Shiela Brookshire and Sandra and Logan Fitch reviewed significant portions of the material for readability. Stephen F. Austin State University supplied grant funds that partially supported the work leading ultimately to Chapter 3. Morris Lang, of the University Computer Center, provided a good deal of technical assistance. The *Journal of Data Education*, with prior permission from Prentice-Hall, Inc., published my initial report about the content of Chapter 3 in their October 1980 issue. An immeasurable amount of support in the

way of prayers and words of encouragement came from my family. From my heart, I thank them all.

Let me also thank you for reading and using this book. Please make me aware of any suggestions you have for improvement. Be assured that I will respond to you.

Jarrell C. Grout
Nacogdoches, Texas

Contents

Chapter

1

Computers, Programs, and Data

Before you begin to learn about computer programming, you need to have at least a fundamental familiarity with computers. It will also be helpful for you to know what a computer program is and how data are related to computers and computer programs.

If you already have this knowledge, you can probably bypass this chapter and go on to Chapter 2. If, however, you have no computer background of any kind, or if you just want to "brush up" on the concepts, you should read this chapter. It has been prepared to get you ready for computer programming by acquainting you as directly and summarily as possible with the background principles with which you need to be familiar.

1-1 WHAT COMPUTERS DO

Computers are just about the most useful machines around. Like virtually all machines, they were invented and have been developed to help people. Although they have been in widespread general use for only a relatively few years, the services they now perform are, without doubt, infinitely broader in scope than the early computer experimenters envisioned; their performance certainly far exceeds the early expectations.

It has been implied that computers are similar to human beings. In a limited sense, this is a valid analogy that can be used to help you begin to understand what

computers do. In a very simplified way, then, human beings and computers are compared in the following statement:

> Human beings act on the basis of knowledge and observation;
> computers solve problems by means of instructions and data.

The essence of this statement is depicted in Figure 1-1. You can see from the diagram—as well as from the sentence above—that instructions are like knowledge, data are like observations, and problem solutions are like actions. Keep this in mind.

A *computer* is a machine that is capable of doing the following things without intervention by a human being: receiving and storing data and instructions, performing operations on the data as prescribed by the instructions, and displaying the results in a form understandable to human beings. This definition, which is the first of many that you should learn, provides a good starting point for finding out what can really be accomplished with a computer. I want to point out that although a computer can perform the activities described free of intermediate involvement by a person, it definitely takes a person to arrange for the data to be collected, come up with the instructions, get the operations started, and do something with the results. The computer is not the "thinker" in this process; it is the "workhorse."

By considering together the computer diagram of Figure 1-1 and the definition above, you can see that problem solving is the goal of computer use. The performance of certain tasks to accomplish this goal is the service for which the computer was invented. From the human view, these tasks can usually be characterized by one or more of such terms as "repetitive," "boring," "time consuming," "complex," and "impossible." If the computer can help us with tasks such as these, it must be an invention worthy of further study by just about everyone.

Problems that can be solved advantageously with the help of a computer exist all around us. From the need of a company to write checks with which to pay

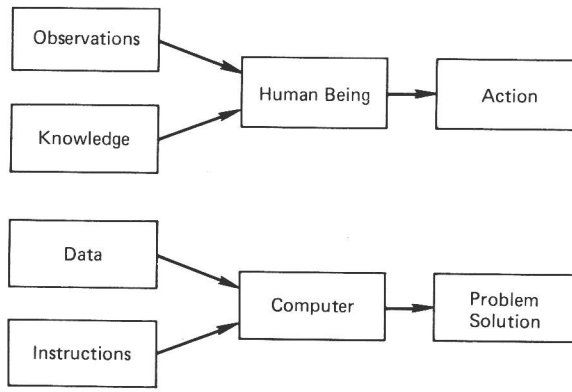

Figure 1-1. A Simplified Comparison of Human Beings and Computers

employees, to that of a scientist for analysis of some experimental data, to the individual problem of keeping personal budget records—they are everywhere. And computers are just about everywhere too.

1-2 KINDS OF COMPUTERS

There are two major classes of computers: analog and digital. In an *analog computer*, data are represented by continuously variable physical quantities such as electric current. In a *digital computer*, data are represented electronically in the form of distinct digits—ordinarily in the binary system. The digital computer is widely used throughout the world today and is usually the type meant when the term "computer" is mentioned. Similarly, from this point on, whenever I use the word "computer" I will be referring only to the digital computer because it is the type in which we are interested.

The primary characteristics that distinguish one digital computer from another are physical size and performance capabilities. Thus you will hear some referred to as "large-scale computers," others as "minicomputers," and still others as "microcomputers." Briefly stated, large-scale computers have broad capabilities and may occupy a fairly good-sized room; minicomputers ordinarily have a wide range of capabilities and can usually fit into a corner of a room; microcomputers are commonly small enough to set on the top of a desk, but they may be somewhat limited in capabilities. Figures 1-2, 1-3, and 1-4 illustrate these three kinds of digital computers.

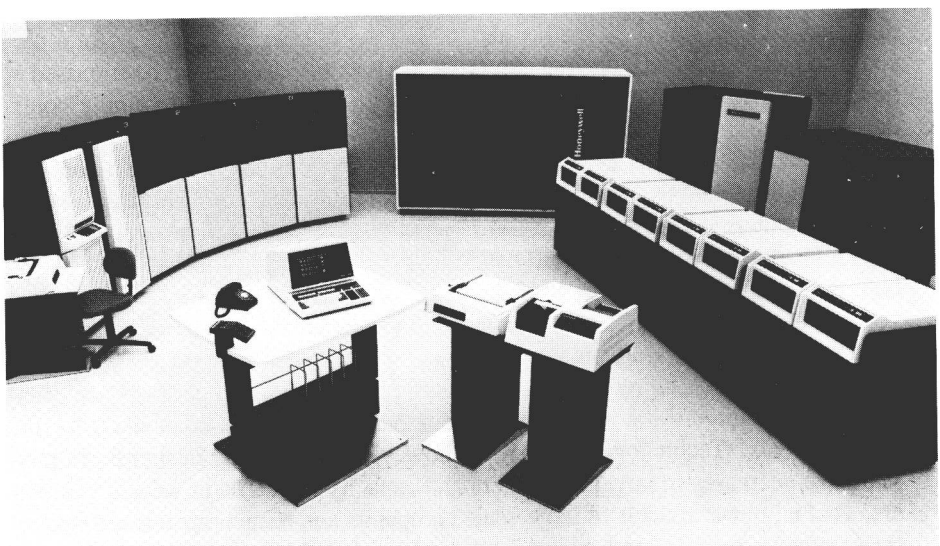

Figure 1-2. A Large-Scale Digital Computer (*Courtesy of Honeywell Information Systems, Inc.*)