Wilson T. Price

Elements of FORTRAN IV Programming

SECOND EDITION

FORTRAN IV PROGRAMMING

SECOND EDITION

Rinehart Press / Holt, Rinehart and Winston
San Francisco

Library of Congress Cataloging in Publication Data

Price, Wilson T
Elements of Fortran IV programming.

Published in 1969 under title: Elements of basic Fortran IV programming, as implemented on the IBM 1130/1800 computers.
Includes bibliographical references and index.
1. FORTRAN (Computer program language) I. Title.
QA76.73.F25P74 1975 001.6'425 74-22150
ISBN 0-03-089502-2

Elements of FORTRAN IV Programming, Second Edition, by Wilson T. Price

PRINTED IN THE UNITED STATES OF AMERICA

8 9 140 9 8 7 6 5 4 3

ELEMENTS OF

Wilson T. Price

MERRIT COLLEGE,

OAKLAND, CALIFORNIA

ELEMENTS OF FORTRAN IV PROGRAMMING

Preface

As was the first, this second edition has been written specifically for the student with average ability who desires or requires a basic working knowledge of Fortran. Perhaps the most significant aspect in which it differs from many of the excellent Fortran books now available is its slow and deliberate pace and its sequencing of topics to allow virtually immediate access to the computer. This is accomplished through use of a basic integer subset of the language in Chapter 1. The topics covered, which amply provide the beginner sufficient background to write simple programs, include (1) simple expressions and the arithmetic statement, (2) the READ and WRITE statements, (3) the FORMAT statement using the I and X data descriptors, and (4) the GO TO statement. Thus after only a few lessons the student can begin writing simple yet meaningful programs and the computer then becomes, in a sense, a teaching machine from the very beginning. This approach is in contrast to that of many books which discuss the general characteristics of computers and extensive details of Fortran constants, variables, and expressions, the Fortran arithmetic statement, and input/output before the first program can be run. Indeed, it is the author's contention that much extended discussion on computer characteristics is lost to the average student if he has no frame of reference. The approach used in this text provides that frame by presenting a basic subset of the language in Chapter 1; then while the student is writing and running programs he can be studying more advanced topics such as the IF statement and the nature of the compiling process. Taken in this manner, the latter assumes much more significance.

The attempt in this revision was to retain the strong features of the first edition and to expand and upgrade those areas which were generally considered weak. The primary changes include (1) utilization of the American National Standard (ANS) Fortran as the basic standard for the book, (2) use of an integer subset in Chapter 1, (3) inclusion of a chapter on alphabetic data manipulation (Chapter 8), (4) the generalization of disk and tape I/O to include both ANS sequential processing techniques and IBM 360/370 direct-access facilities, (5) the expansion and broadened scope of programming problems, and (6) the inclusion of exercises with answers at the end of each chapter.

Three appendixes in this book include (1) a suggested subroutine which can be used to terminate program processing for use in Chapter 1, (2) a list of relevant features in ANS Fortran which are not included in ANS Basic Fortran, and (3) a description of card coding, card fields, and using the card punch to prepare Fortran programs.

ACKNOWLEDGEMENTS

Many of the notions and techniques used in this book have come, either directly or indirectly, from the suggestions of colleagues, students, and friends. In addition to the many who contributed to the first edition, the following have contributed their expertise to this second edition:

James L. Boettler, Talladega College
Harry Caughren, Merritt College
Amos T. Hathaway, The Citadel
Colonel Oren L. Herring, Jr., The Citadel
Claude E. LaBarre, State University of New York
Marie Palmer, Edinboro State College
Jesse K. Peckenham, Merritt College
Richard Theil
James H. Westmoreland, University of Tennessee

To these and all others who participated, I extend my sincere thanks.

Oakland, California WILSON T. PRICE

CONTENTS

EXAMPLE PROGRAMS

Introduction

CHAPTER CONTENTS

THE MODERN DIGITAL COMPUTER

Electronic digital computers fall into two broad categories: special purpose and general purpose. As the name implies, special-purpose machines are designed to perform a single type of function, such as military aircraft fire control or process control for oil refineries. General-purpose computers, through their versatile languages, can be used for virtually any type of application. Although computer design will vary from one manufacturer to another, the basic components common to most modern general-purpose digital computers are (1) storage, (2) the arithmetic or logic unit, (3) the control unit, (4) input/output, and (5) auxiliary storage.

STORAGE

Computer *storage*, the single component that is probably most glamorized, is often referred to as memory analogous to the human memory. This popular representation can be misleading, because the storage unit is better compared to a filing cabinet in which information can be stored and retrieved in an orderly manner. For this reason, the term *storage* will be used exclusively.

ARITHMETIC UNIT

It is the function of the *arithmetic unit* to perform basic arithmetic and certain logical operations as required by the program. These include the common arithmetic operations of addition, subtraction, multiplication, and division as well as special shifting functions and Boolean operations. To these ends the arithmetic unit includes special storage locations commonly referred to as arithmetic registers. These together with electronic switches and other components are analogous to the gears, wheels, and registers of a desk calculator.

CONTROL UNIT

The task of directing operations within the computer is the function of the automatic *control unit*. This portion of the computer can be considered analogous to the combination of a traffic officer and automatic telephone switchboard. It obtains instructions from storage, interprets them, and makes certain that they are carried out as required. These functions involve opening and closing appropriate circuits, starting and stopping input/output devices, and, in general, directing flow of information in the computer.

INPUT/OUTPUT

Means for communicating with computers has always been a formidable problem to computer designers because all input/output devices involve some type of mechanical linkage, and, in general, mechanical methods are relatively slow. This has undoubtedly contributed to the many and varied means used to get information into and out of a computer. The most common devices for input/output of data are punched cards, magnetic tape, punched paper tape, and the computer typewriter. Printers capable of printing entire lines at a time are also commonly used for output.

AUXILIARY STORAGE

In almost all computer systems, the problem of insufficient storage for some applications is almost inevitable. Simply to obtain a computer

with a larger storage unit is not always the appropriate solution for a number of reasons, one of which is high cost. As a result, most computers are presently available with *auxiliary storage* devices. These are commonly magnetic tape, disk, or drum devices and have storage capabilities many times greater than the main storage of the computer. However, a disadvantage of auxiliary storage devices is their relatively slow operating speeds. They are commonly used to store special systems programs for achieving virtually automatic transition from one job to the next, for storing commonly used programs, and for storing large information files.

THE COMPUTING SYSTEM

These basic components, which are common to most computers, are generally interconnected as shown in Figure 1. Information flow is (1)

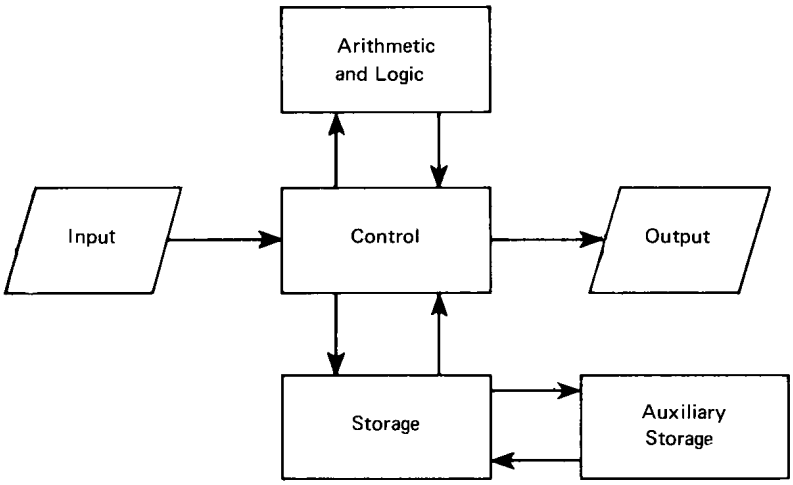

FIGURE 1 Schematic diagram of a computer system.

from input through control to storage, (2) from storage through control to the arithmetic section and back to storage through control, (3) from storage through control to output, and (4) from main storage to auxiliary storage and back to main storage.

These are the elementary concepts and components common to all digital computers. At the present time there are electronic computers in use with the capacity for storing only a few hundred characters of information and with minimal input/output capabilities. On the other hand, the newest and largest computer systems store several million pieces of information and utilize many input/output devices, as well as extra equipment including auxiliary storage facilities.

COMPUTER SOFTWARE

OPERATING SYSTEMS

Early computers required considerable attention from the operator. That is, in running a series of jobs the operator would ready the computer, load one job (deck of cards constituting the program of instructions), and start the computer executing the program. If error conditions occurred which stopped the program, the machine would frequently remain idle while the operator investigated the problem. Furthermore, after the program was completed the computer would often

be idle during the preparation for the next job. Overall the machine would be waiting for operator action a significant portion of the time. With the vastly improved computing speeds and capabilities of newer computers and the increased demand for computer use, this primitive type of manual control became totally impractical. Special supervisory types of programs quickly came into being for the purpose of providing automatic or semiautomatic control over many of the machine functions. These programs improved computer utilization and relieved the operator of many mundane activities. We now know these types of systems as *operating systems*. Briefly an operating system is a set of programs, resident in the computer system, designed to maximize the amount of work the computer system can do.

High computing speed, large storage capacities, and sophisticated input/output capabilities are three of the many hardware features of present-day computers. To most users, however, programming systems or *software* (including diverse languages and operating systems) supplied by the manufacturer are just as important. For, in general, these systems relieve the user of extensive, detailed programming and operational responsibilities, allowing for much more efficient utilization of the computer. Since the present computers were designed specifically to work with operating systems, their machine capabilities are used much more efficiently than those of earlier computer systems.

With the extensive use of magnetic tape and especially of quick access storage devices such as magnetic disk, the utilization of operating systems has become commonplace. The key to an operating system is the supervisor program (sometimes called a monitor or an executive program). The supervisor remains in storage at all times and maintains control, directly or indirectly, while the computer is in use (see Figure 2). In addition, an operating system consists of other special control programs, systems service programs, and processing programs. Through the use of system libraries, these are integrated into the comprehensive operating system, which may be controlled by the user through special job control commands. The overall nature of the operating system in general and the supervisor in particular is illustrated in the schematic representation of Figure 3.

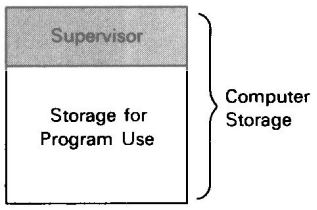

FIGURE 2 The supervisor in storage.

MULTIPROGRAMMING AND TIME SHARING

Through the use of operating systems and special hardware features of the modern computer, the total amount of useful work performed (throughput) has been greatly increased over earlier systems. Another means for increasing the usage of the machine is through *multiprogramming*. With the storage-resident supervisor concept, we have been introduced to the notion of having two programs in storage at the same time: the supervisor for overall system control and the problem program for performing the data-processing function. In multiprogramming, this concept is carried one step further by placing two or more problem programs in storage and executing them concurrently. Although two or more programs may reside in storage simultaneously, the computer is capable of executing only one instruction at a time. Thus at any given time only one of the programs has control of the computer and is executing instructions. Simultaneous execution of two programs with one central processing unit (CPU) is impossible.

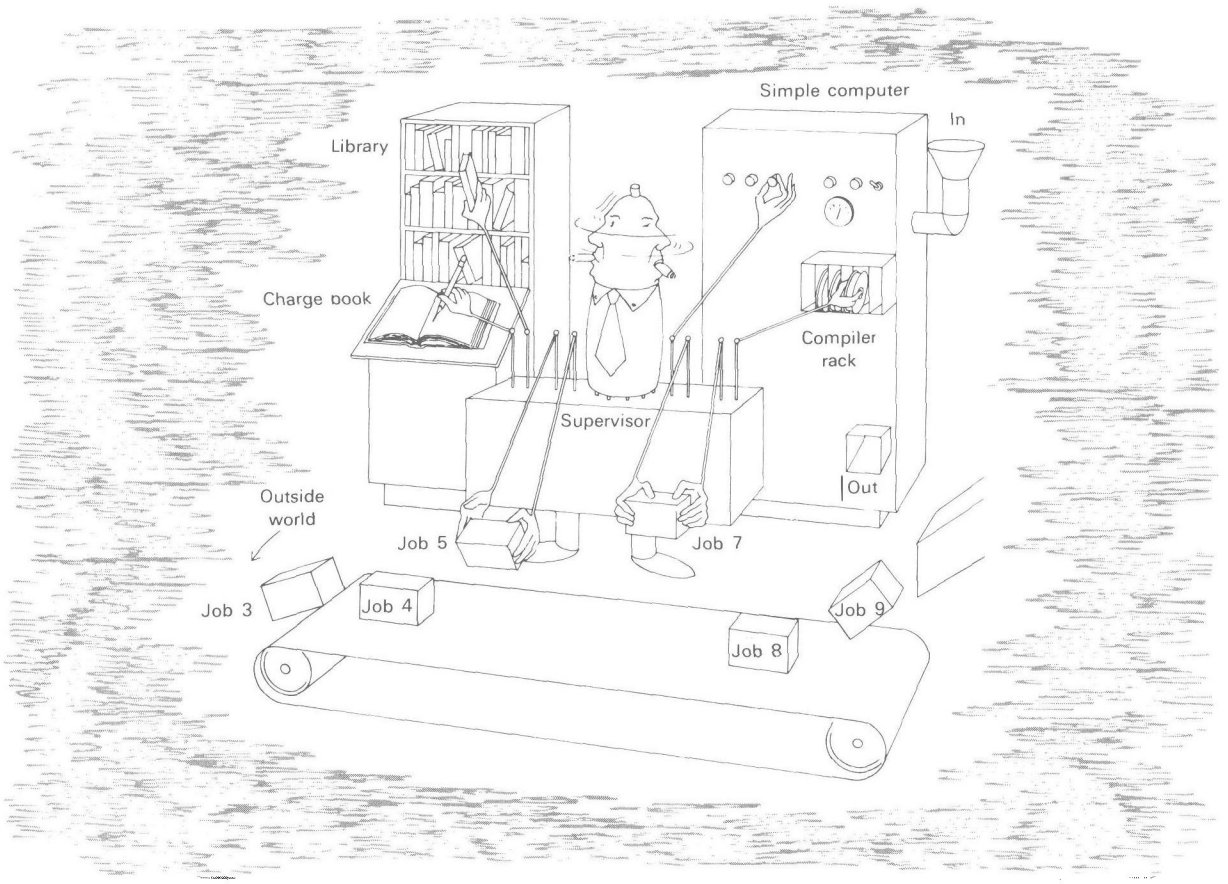

FIGURE 3 Animated representation of an operating system.

With multiprogramming, one program has a higher priority than the other. For instance, if Program 1 in Figure 4 had the higher priority then it would have control of the computer. When input/output operations are being performed for Program 1 (for instance, a card is being read), the computer must wait for this relatively slow mechanical operation to be completed. During this wait time, control can temporarily be passed to the lower-priority Program 2. When the input or output operation is completed, control will then be returned, via the supervisor, to the higher-priority Program 1.

Another commonly used technique employed to serve the computer user is *time sharing*. Generally speaking, time sharing refers to the allo-

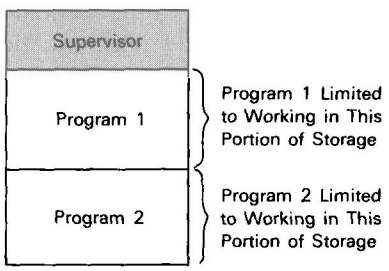

FIGURE 4 Two programs in storage—multiprogramming.

cation of computer resources in a time-dependent fashion to several programs simultaneously in storage. The principal notion of a time-sharing system is to provide a large number of users direct access to the computer for problem solving. The user thus has the ability to "converse" directly with the computer for problem solving (hence the terms *conversational* or *interactive* computing). In multiprogramming the principal consideration is to maximize utilization of the computer; in time sharing it is, in a sense, to maximize efficiency of each computer user and keep him/her busy. Figure 5 illustrates the notion of a time-shared system. Each user has

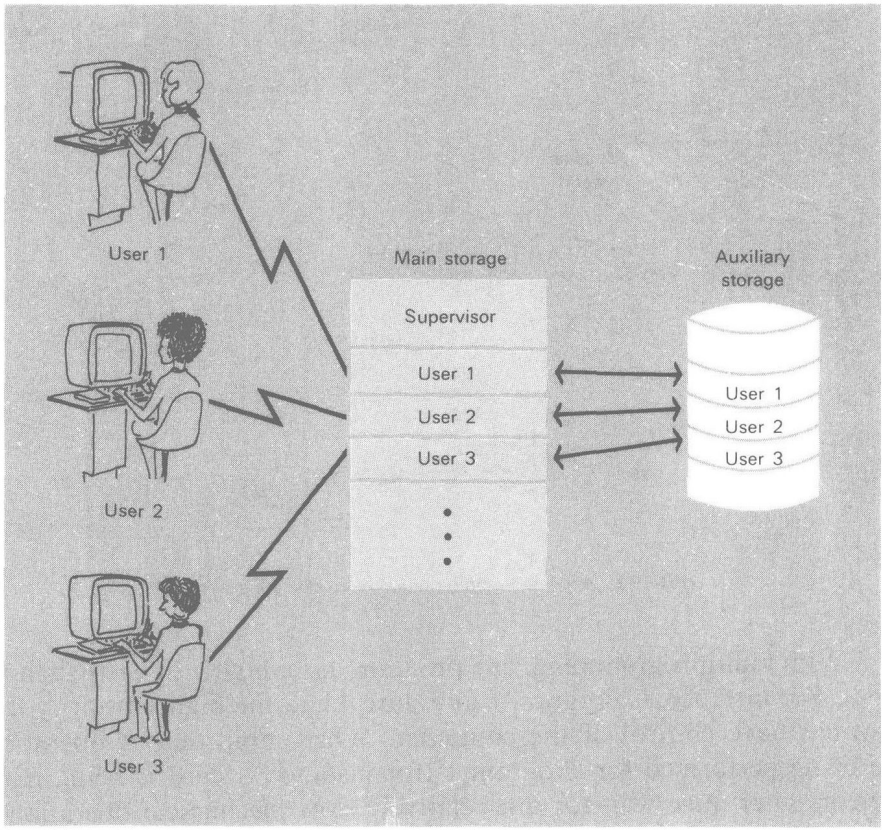

FIGURE 5 Time-sharing environment.

his/her own communications terminal, portion of storage, and auxiliary storage. In contrast to multiprogramming where programs are executed on a priority basis, with time sharing the CPU time is divided among the users on a scheduled basis. Each program is allocated its slice of the CPU time (commonly measured in milliseconds) based on some predetermined scheduling basis beginning with the first program and proceeding through the last. Upon completing the cycle, it is begun again so that an individual user scarcely realizes that someone else is also using the computer.

Time-sharing terminals have come into common use in both education and business for the purpose of programming. In fact, many computer systems include special interactive Fortran programming systems for use on a time-sharing operation.