Structured Programming in FORTRAN

LOUIS A. HILL JR.

Structured Programming in FORTRAN

LOUIS A. HILL JR.

Arizona State University

Prentice-Hall, Inc.

Englewood Cliffs, New Jersey 07632

5306394

Library of Congress Cataloging in Publication Data

Hill, Louis A., Jr. (date)
Structured programming in FORTRAN.

(Prentice-Hall software series)
Includes index.
1. FORTRAN (Computer program language)
2. *Structured programming. I. Title. II. Series.*
QA76.73.F25H54 001.64'24 80-39567
ISBN 0-13-854612-6

© 1981 by Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632

All rights reserved. No part of this book
may be reproduced in any form or
by any means without permission in writing
from the publisher.

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

Editorial/production supervision and interior design:
Nancy Milnamow and Marianne Thomma Baltzell
Cover design: Jorge Hernandez
Manufacturing buyer· Joyce Levatino and Gordon Osbourne

Prentice-Hall International, Inc., *London*
Prentice-Hall of Australia Pty. Limited, *Sydney*
Prentice-Hall of Canada, Ltd., *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*

Preface

To the Student

This book has been class-tested by over 5,000 students from nearly every department on the Arizona State University campus. It is written to satisfy your present and future needs by helping you through various levels of programming skill and serving you as a ready reference.

Chapter exercises are chosen to clarify your thinking and to reinforce your knowledge; most exercises have answers provided in the appendix. Words for the glossary have been carefully chosen to help you communicate in the computer center and computer community, as well as in book-related activities. The index is comprehensive. Boldface entries in the index point to elements of the book which summarize the rules of FORTRAN 77.

The ultimate objective of this book is to help you develop sufficient knowledge and skill to use computers to provide humankind with pertinent information and to free time for creative endeavors.

To the Instructor

This book is designed to support a broad spectrum of teaching techniques. Several chapters and sections may easily be omitted or reordered to suit your style.

It is my intention that all FORTRAN, unless specifically noted (rare), conform to ANSI X3.9-1978, FORTRAN 77. In general, when Full Language is not specified, Subset Language is implied. Because the ANSI Standard is written by and for experienced programmers, the ANSI format is not incorporated directly into this text. Rather, the format for explaining FORTRAN 77 is based upon the results of extensive experimentation, by using different techniques with students from essentially every academic discipline. Although I have gone to great lengths to insure accuracy, I am aware that some errors have probably not been detected. Your comments, corrections, and suggestions are most welcome.

An Instructor's Manual is available to facilitate your use of this text. A Typical Course Schedule in the Instructor's Manual lists subjects for each lecture and shows text sections that the student should read and/or study before coming to class. A quiz and laboratory schedule is included, with suggested point values.

Lesson plans in the Instructor's Manual state objectives, give a summary lecture outline, and present a detailed lecture outline keyed to transparencies or slides.

Laboratory projects in the text, designed for very early use, are ample for several semesters. Individual laboratory projects allow each student to write a different computer program, yet these programs are accurately and quickly graded with aids presented in the second half of the Instructor's Manual.

ACKNOWLEDGEMENTS

This text is the outgrowth of three previous versions which were class-tested and critiqued by over 5,000 students and fifty laboratory instructors. From the large group who provided extraordinary help, the following deserve special mention: George C. Beakley, III, Carolyn Brown, Hugh Bynum, David Eyestone, Clayton Knight, John Krobock, Kathy Levandowsky, Peter K. Piascik, and Rajiv Sachdeva.

The critique and criticism of the reviewers, Dr. Lionel E. Deinel, Jr. and Dr. Brian Kernighan, was extremely helpful and especially appreciated. Linda Thompson did an excellent and timely job of copyediting. The late Marianne Baltzell and Nancy Milnamow were most pleasant and helpful in expediting the book through the production stage. Karl Karlstrom and Steve Cline, my computer science editors, have been consistently supportive. To all of them, my thanks.

Special thanks are due to Dr. Robert Lovell and Dr. James Wilson, professors who gave excellent critiques from their experience gained while classroom-testing the material. Thanks are also due to Dr. George C. Beakley, Jr., for his consistent encouragement and support.

For more than a decade, I have been able to count on my son, David A. Hill (presently a systems programmer), for critique, enlightening suggestions, and lively discussion on relevant but sometimes obscure points. He has added appreciably to the quality of this book.

Louis A. Hill Jr.

此为试读，需要完整PDF请访问：www.ertongbook.com

Contents

PREFACE xv

Chapter 1 OVERVIEW—A LOOK AHEAD 1

1-1 Situations You Can Expect to Encounter 1
1-2 Effect of Your Personal Background 2
1-2.1 Experience Level 2
1-2.2 Area of Interest or Expertise 2
1-2.3 Age 3
1-3 How this Text Relates to Your Future 3
1-4 Confronting the Computer 3
1-4.1 What the Computer Reads From 4
1-4.2 What the Computer Writes On 7
1-4.3 Basic Machine Configuration 8
1-5 Insights to Expedite Your Learning 8
1-5.1 Running a FORTRAN Program 8
1-5.2 What Kind of Trouble Can You Expect? 10
1-5.3 Saving Yourself Time and Trouble 12
1-5.4 Proofreading and Turn-Around Time 13
1-6 Objectives 13
1-6.1 Structured Programming 14
1-6.2 Module; Modular 14
1-6.3 Systems Approach 14
1-7 Exercises 14

Chapter 2 PROGRAMMING BY CONCEPT AND EXAMPLE 17

2-1 *Reading Program IEQPRT* 18
- 2-1.1 *Definition Sheet* 20
- 2-1.2 *Solutions to* i = Prt 21
- 2-1.3 *First Solution to Program IEQPRT* 21
- 2-1.4 *Aids Used to Formulate Program IEQPRT* 24
- 2-1.5 *Modifying IEQPRT into EEQPRT* 26
- 2-1.6 *DO-WHILE Variations* 31
- 2-1.7 *Enhancing Readability* 31

2-2 *Program CUB* 32
- 2-2.1 *Understanding the Problem* 32
- 2-2.2 *Evolution of Human-Oriented Flow Diagram* 33
- 2-2.3 *Summary* 35
- 2-2.4 *Alternate Approach* 35
- 2-2.5 *A Simple WRITE Statement to Output a Title* 35
- 2-2.6 *Basic Concepts of FORMAT Statements* 36
- 2-2.7 *Typical FORMAT Statement for WRITE (LW, 1003)* 38
- 2-2.8 *Typical Statements to Input Data* 42
- 2-2.9 *Typical Output* 44

2-3 *Program COMP* 44
- 2-3.1 *Definition Sheet* 45
- 2-3.2 *Human-Oriented Flow Diagram* 46
- 2-3.3 *Machine-Oriented Flow Diagram* 50
- 2-3.4 *Coding* 52
- 2-3.5 *Comment* 53

2-4 *A Program that Calls a Subroutine* 54
- 2-4.1 *Genesis of the Program* 54
- 2-4.2 *Program TFINDS* 55
- 2-4.3 *Subroutine FINDS* 56

2-5 *Exercises* 62

Chapter 3 FORMULATION PROCEDURES 64

3-1 *Problem Definition* 64

3-2 *A Formal Definition Sheet* 65
- 3-2.1 *Composition* 65
- 3-2.2 *How Definition Sheets Are Developed* 65

3-3 *Diagrams and Charts* 66
- 3-3.1 *Flow Diagrams* 66
- 3-3.2 *Modified Nassi-Shneiderman Charts* 67

3-4 *Human-Oriented (Gross) Diagrams and Charts* 67
- 3-4.1 *Human-Oriented Flow Diagrams* 68
- 3-4.2 *Human-Oriented Modified Nassi-Shneiderman Charts* 68
- 3-4.3 *Constructing Diagrams and Charts During the Development of Program* 69

3-5 *Machine-Oriented (Detailed) Flow Diagrams and Charts* 69
3-6 *Program Tracing* 71
3-7 *Coding in FORTRAN 77* 75
3-8 *Exercises* 76

Chapter 4 ESSENTIAL RULES OF FORTRAN PROGRAMMING 85

4-1 *Formula Translation* 85
4-2 *Three Kinds of FORTRAN Syntactic Items* 86
4-2.1 *A Valid Integer (Fixed Point) Constant* 87
4-2.2 *A Valid Real (Floating Point) Constant* 87
4-2.3 *Variables and Variable Names* 88
4-2.4 *Type* 89
4-2.5 *Scientific Notation* 89
4-2.6 *Exceptions* 91
4-3 *Storage of Data—A Conceptual Model* 91
4-4 *Declaration of Variables* 92
4-4.1 *Explicit Type Statements* 92
4-4.2 *IMPLICIT Type Statements* 93
4-4.3 *Combination of Type Statements* 93
4-4.4 *When to Declare Variables* 94
4-5 *Expressions* 94
4-5.1 *Exponentiation* 95
4-5.2 *Scientific Notation versus Exponentiation* 96
4-5.3 *Hierarchy of Computations* 96
(a) Negative Exponentiation 97
(b) Sequential Exponentiation 98
(c) Delineation of Denominators 99
(c) Compound Division 99
4-6 *Arithmetic Assignment (Replacement) Statements* 99
4-6.1 *Mixed Mode* 101
4-6.2 *Integer Computations* 102
4-6.3 *Built-In Library Functions* 104
4-7 *Three Control Statements* 105
4-7.1 *The CONTINUE Statement, Continuation Lines, and Comment Lines* 106
4-7.2 *STOP and END* 106
4-8 *Comments on the FORTRAN Language* 107
4-9 *Exercises* 107

Chapter 5 CONTROL OF FLOW 112

5-1 *Fundamental Control* 112
5-1.1 *Unconditional GO TO Statement* 112
5-1.2 *Logical and Relational Operators and Expressions* 113

5-1.3 *Logical IF Statement* 115
5-1.4 *More about Logical Expressions* 118
5-2 *Structured Programs and Modules* 119
5-3 *Block IF Mini-Module* 120
5-3.1 *IF-THEN* 122
5-3.2 *IF-THEN-ELSE* 123
5-3.3 *IF-ELSE IF-ELSE IF-ELSE* 124
5-4 *DO-WHILE Mini-Modules* 126
5-4.1 *The Classical DO-WHILE* 126
5-4.2 *Modified DO-WHILE Mini-Module* 129
5-5 *Compound Mini-Modules* 130
5-6 *Exercises* 131

Chapter 6 INPUT/OUTPUT **133**

6-1 *Formal Rules for I/O* 133
6-1.1 *Executable Statements* 133
6-1.2 *FORMAT Statements* 134
6-1.3 *Format Specifications* 136
6-1.4 *Format Rules* 139
6-1.5 *Mixed Mode in Input/Output* 143
6-2 *Print-Control* 146
6-2.1 *Use in Business* 146
6-2.2 *Use in Design* 146
6-2.3 *Use to Monitor Programs* 147
6-2.4 *A Specific Example of Print Control* 147
6-3 *END and ERR Used in Control Information Lists* 148
6-4 *List-directed I/O* 150
6-5 *Exercises* 153

Chapter 7 SYNTHESIS AND EXAMPLES **155**

7-1 *Program SIMLR—Simple Linear Regression* 155
7-1.1 *Understanding the Problem of Relating Data Pairs* 155
7-1.2 *Developing Program SIMLR* 157
7-1.3 *Applicability of SIMLR as a Subroutine* 158
7-2 *Program QUAD—Roots of the Quadratic Equation* 160
7-2.1 *Understanding and Developing a Solution to the Quadratic Equation* 160
7-2.2 *Spacing of Output* 161
7-2.3 *Formulating the Program* 161
7-2.4 *Choosing Test Data by "Reverse" Formulas* 163
7-3 *Program RATE—Determining Unknown Compound Interest Rates* 166
7-3.1 *Understanding the Problem of Unknown Interest Rate* 166
7-3.2 *A Solution with Program RATE* 167

7-3.3 Control of the Looping Process 168
7-3.4 Choosing Test Data by Book Examples 173
7-4 Program CUES—Solving for Roots By Iteration 173
7-4.1 The Iteration Process 173
7-4.2 Convergence of the Iterative Process (Optional) 175
7-4.3 Controlling an Iteration Process 175
7-4.4 Test Data 178
7-5 Program CUBIC—Roots of Cubic Equation by Cardan's Method (Optional) 179
7-5.1 Type Complex and the Cube Root 179
7-5.2 Two Subroutines for Cube Root 181
7-5.3 Cardan Solution of the Cubic Equation 182
7-6 Exercises 182

Chapter 8 PROGRAM LOOPING WITH DO-LOOPS 186

8-1 Introduction to the DO-Loop 186
8-1.1 DO-Loop Represented by Human-Oriented Flow Diagrams 186
8-1.2 DO-Loop Represented by Machine-Oriented Flow Diagram 186
8-1.3 DO-Loop Coding and Restrictions 189
8-1.4 DO-Loop Represented by Modified Nassi-Shneiderman Chart 190
8-2 DO-Loop Examples 190
8-2.1 DO-Loop Using Counter Within the Loop 191
8-2.2 DO-Loop With Counter Not Used in Loop 191
8-2.3 Counting, Summing and Accumulative Multiplication 191
8-2.4 Obtaining Special Sequences 192
8-2.5 "Artificial" Type Real Loop Indices 193
8-2.6 Special Considerations with Regard to DO-Loops 196
8-2.7 DO's in Combination with Block IF's 198
8-3 Program LOOP 198
8-4 Halt Methods 206
8-4.1 Error Halt or Control Information List Specifier END 206
8-4.2 Constant Count Halt 212
8-4.3 Variable Count Halt 212
8-4.4 Constant DO-Loop Halt 214
8-4.5 Variable DO-Loop Halt 214
8-4.6 Trailer Card Halt 215
8-5 Exercises 215

Chapter 9 SINGLE-DIMENSIONAL ARRAYS 217

9-1 The Concept of Single-Dimensional Arrays 217
9-2 DIMENSION Statement 217
9-3 Array Elements; Subscripted Variables 218
9-4 I/O of Single-Dimensional Arrays 219

9-4.1 *Reading the Whole Array at One Time* 220
9-4.2 *DO-Loop Used to Read an Array* 221
9-4.3 *The Implied-DO Loop* 222
9-4.4 *Mixed I/O Forms* 223
9-4.5 *Length of Implied-DO Established Within READ* 224
9-4.6 *Multiple Entity Implied-DO Lists* 224
9-5 *Bubble-Sort Algorithm Using Single-Dimensional Arrays* 224
9-5.1 *Understanding the Bubble-Sort Algorithm* 225
9-5.2 *Development of Program BBSRT1* 226
9-5.3 *Coding and Output for Program BBSRT1* 229
9-6 *A Search Program* 231
9-6.1 *Formulation of Binary Search* 233
9-6.2 *Program SEARCH* 233
9-7 *Subroutine — Best Straight Line Through Set of Data Pairs (Optional)* 237
9-8 *Exercises* 238

Chapter 10 THE DATA STATEMENT, ARRAYS, AND NESTED DO-LOOPS **241**

10-1 *The DATA Statement* 241
10-2 *Program WIND* 243
10-2.1 *Illustration of Longhand Solution for Wind Force on Buildings* 243
10-2.2 *Formulating the Computer Program* 244
10-2.3 *Summary of Program WIND* 247
10-3 *Arrays (Tables or Matrices)* 247
10-3.1 *Dimensions* 249
10-3.2 *Array Elements; Subscripted Variables* 252
10-3.3 *Exceeding Array Limits* 253
10-3.4 *Protection Against Exceeding Array Limits in DO-Loops* 254
10-4 *Nested DO-Loops* 255
10-4.1 *A Simple Example* 255
10-4.2 *Some Typical Uses of Double-Nested DO-Loops* 258
10-5 *I/O of Arrays* 262
10-6 *Exercises* 268

Chapter 11 EXAMPLES INVOLVING TWO-DIMENSIONAL ARRAYS **272**

11-1 *Program DATAM* 272
11-1.1 *Formulation of Program DATAM* 272
11-1.2 *Critique of Output* 274
11-1.3 *Modifications to Program DATAM* 274
11-2 *Program WINDZ* 278
11-3 *Sorting Routines* 280

此为试读，需要完整PDF请访问：www.ertongbook.com

11-3.1 Program BBSRT 2 280
11-3.2 Program BBSRT 3 282
11-4 Matrix Multiplication 286
11-4.1 Generalized Rules of Matrix Multiplication 287
11-4.2 A Numerical Example of Matrix Multiplication 289
11-4.3 Programming Matrix Multiplication 290
11-5 Exercises 292

Chapter 12 UTILIZING CHARACTERS **293**

12-1 Using Subset FORTRAN 77 293
12-1.1 'Fundamental Rules, Program CHRBAS 293
12-1.2 Counting Letters, Program ALPHA1 297
12-1.3 Counting Words, Program CNRBG1 300
12-1.4 Sorting a List Containing Alphanumerics, Program CHSORT 302
12-2 Variations of Output, Program VAROUT 302
12-2.1 Format Identifiers 302
12-2.2 Full Language FORTRAN 77 305
12-2.3 Calculator Mode 307
12-3 Using Full Language FORTRAN 77 with Substrings 307
12-3.1 Fundamental Rules, Programs CONCAT and LEN 307
12-3.2 Counting Letters, Program ALPHA2 310
12-3.3 Counting Words, Program CNRBG2 310
12-3.4 Program SEEK 311
12-4 Using Hollerith (Optional) 311
12-4.1 Fundamental Rules 312
12-4.2 Simple Examples, Programs DSAF and UNIA 316
12-4.3 Counting Letters, Program ALPHA3 318
12-5 Exercises 320

Chapter 13 RECAPITULATION AND SYNTHESIS **321**

13-1 Formal FORTRAN 77 321
13-1.1 Summary of Syntactic Items 321
13-1.2 Summary of Data Types 322
13-1.3 Order and Classification of Statements 322
13-1.4 Assignment Statements: Order of Evaluation and Conversion of Type 322
13-2 Style and Good Practice 324
13-2.1 Supportive Documentation 324
13-2.2 Coding 325
13-3 Problem Formulation 327
13-3.1 Input/Output; Link 327
13-3.2 Starting in the Middle 327
13-4 Input/Output 327

13-4.1 *Input Considerations* 328
13-4.2 *Using a Layout to Format Output* 328
13-4.3 *Print-Control* 329
13-5 *Program Verification* 329
13-5.1 *Choosing Test Data* 329
13-5.2 *Typical Errors* 329
13-5.3 *Tracing Techniques* 331

Chapter 14 SUBROUTINES **332**

14-1 *Construction and Use of Subroutines* 332
14-1.1 *Form of a Subroutine* 333
14-1.2 *Calling a Subroutine* 334
14-1.3 *Actual versus Dummy Arguments* 335
14-1.4 *RETURN* 336
14-1.5 *Nested CALLS* 336
14-1.6 *Elimination of Error* 336
14-2 *A Simple Example; Subroutine DEMO* 337
14-3 *A More Useful Example; Subroutine QUAD* 340
14-4 *More About Arguments* 344
14-4.1 *Adjustable Dimensioning* 344
14-4.2 *BLANK COMMON* 346
14-4.3 *Comparisons Between COMMON and Adjustabl Dimensioning* 346
14-4.4 *Two Forms of Common and the Block Data Subprogram* 348
14-5 *Subroutine PLOT, a Review* 350
14-5.1 *Basic Considerations* 350
14-5.2 *Formulation* 351
14-5.3 *Toward a Computer Solution* 353
14-5.4 *Testing Subroutine PLOT* 357
14-5.5 *Generalizing Subroutine PLOT* 357
14-6 *Text Editing* 361
14-6.1 *Subroutine FNDCHR* 362
14-6.2 *Subroutine NAMLOC* 366
14-7 *Segmenting for Ease and Flexibility* 371
14-7.1 *Saving Time* 372
14-7.2 *Systems Approach* 373
14-8 *Exercises* 376

Chapter 15 FUNCTIONS **381**

15-1 *Referencing a Function* 381
15-2 *Intrinsic Functions* 386
15-3 *Function Subprograms (External Function)* 386

15-3.1 *Formulating Function Subprogram AREA2* 387
15-3.2 *Programming Function Subprogram AREA2* 387
15-3.3 *Testing Function Subprogram AREA2* 389
15-4 *Statement Function* 390
15-5 *Program FOURIE and Dummy Procedure Calls* 394
15-5.1 *Top-Down Formulation* 395
15-5.2 *Coding the Programs* 397
15-6 *Exercises* 398

Chapter 16 ADDITIONAL FEATURES 403

16-1 *Full Language Extensions* 403
16-1.1 *Miscellaneous* 403
(a) PARAMETER Statement 403
(b) EQUIVALENCE Statement 404
(c) Complex Variables 408
(d) Double Precision Variables 408
(e) Integer Constant Expression 410
16-1.2 *Arrays* 410
16-1.3 *DO-Loops* 412
16-1.4 *The DATA Statement* 413
16-2 *Statements That Are Now Being Phased Out* 413
16-2.1 *The Arithmetic (Three-Way) IF Statement* 413
16-2.2 *The ASSIGN Statement and the Assigned GO TO Statement* 416
16-2.3 *The Computed GO TO Statement* 418
16-3 *Additional Statements* 421
16-3.1 *The PAUSE Statement* 421
16-3.2 *Additional Statements Related to Procedures* 422
(a) The ENTRY Statement 421
(b) The Alternate Return 421
(c) The SAVE Statement 423
16-4 *Exercises* 423

Chapter 17 ADDITIONAL INPUT/OUTPUT 425

17-1 *Simple File Management Using Subset FORTRAN 77* 425
17-1.1 *Terminology* 425
17-1.2 *Input/Output Statements and Control Information Lists* 428
17-1.3 *Special, Processor-Dependent Information Required* 429
17-1.4 *Program SOMNUM, Output Some Numbers to External File* 432
17-1.5 *Program AVG, Obtain Average of Numbers Taken from an External File* 433
17-1.6 *Program PAY1, Simplistic Payroll Program Using Tape, Disk, Card and High-Speed Printer* 434

17-1.7 Program PAY2, Simplistic Payroll Program Using Several External Files, including a Remote Terminal 436

17-2 Additional Edit Descriptors 441

17-2.1 Logical, P(Scale Factor), BZ and BN Editing 441

17-2.2 Positional (T, TL, TR) and Optional Plus Sign (S, SP, SS) Editing 448

17-2.3 Numeric Editing Summarized, Including Full Language 450

17-3 Additional Formatting Techniques Available in Full Language Only 451

17-3.1 Program COLON, Optional Format Termination 451

17-3.2 Program ADJFMT, Adjust Format Within the Program 453

17-3.3 Program FMTINT, Format Obtained from Input Data 455

17-3.4 Program ENCODE, Using Internal Files for Encode and Decode Processes 456

17-4 File Management With Full Language FORTRAN 77 457

17-4.1 Input/Output Statements and Corresponding Control Information Lists 458

17-4.2 Program PAY3, Simplistic Payroll Program Using Features of Full Language FORTRAN 77 459

17-4.3 The Auxiliary INPUT/OUTPUT Statement INQUIRE 459

17-5 Exercises 459

Appendix A LABORATORY PROJECTS 471

Objective of Laboratory Projects 471

Laboratory Project 1 472

Laboratory Project 2 477

Laboratory Project 3 480

Laboratory Project 4 486

Laboratory Project 5 487

Laboratory Project 6 489

Laboratory Project 7 495

Laboratory Project 8 495

The MOD Function 496

Appendix B ANSWERS TO EXERCISES 498

GLOSSARY 512

INDEX 518

Chapter 1

Overview - A Look Ahead

Welcome to an exciting and profitable study—one that can greatly increase your contribution to society and add new dimension to your personality. Not only will the computer save you hours and energy, it will eventually free you from mundane tasks so that you will have time to be creative. The study of computer programming also can increase your ability to define and organize problems in other spheres of your life.

It does take work, as well as courage and stick-to-itiveness—courage to overcome fear of the unknown, and stick-to-itiveness to overcome frustration when a "perfect" computer does what you say and not what you mean. You will face the ever present temptation to forget your ultimate "human" goals and to concentrate upon pleasing an uncompromising impersonal machine. However, if you work and persevere, you will soon be using the computer to truly optimize attainment of your goals.

As in any new subject, you will encounter new vocabulary (or words used in an unfamiliar context), new rules, and new techniques. A glossary is provided to give you ready definitions (words in italics are in the glossary); by the end of your study, the full meaning of these terms will become clear.

1-1 SITUATIONS YOU CAN EXPECT TO ENCOUNTER

In a first computer course, you will encounter a new set of problems. You will have to define each project for yourself and work carefully in defining a program. You will be dependent not only upon the machine, but upon other human beings every time you work toward a solution. You will be constantly beckoned by the siren's call to other, "better," "faster," and "easier" ways of doing things which all too often lead to hours of extra effort.

The computer is a machine—a very, very fast one. It is error-free almost beyond comprehension. But it does only what it is told to do and only when you ask in a "perfect" way. The computer will not read your mind; it will not do what you mean. In addition, the rules for its use must be followed exactly. A comma or a period in the wrong place will probably lead to a program that does not function properly, if at all.

Probably your biggest frustration will be that your work is not solely dependent upon you. In almost every case you will have to depend upon other people who run the machine and handle your program and subsequent results. You will always be dependent upon the availability of a computer and constrained by the *turn-around time* associated with each run that you make. To enjoy programming, you need to allow time in case things—often not of your own doing—go wrong. In general, you should start work early enough so that you will have sufficient slack time to overcome these almost certain untoward occurrences. Many novice programmers feel that everything is under control and do not allow contingency time; in the long run they cost themselves excessive time and effort.

In every computer center there are people willing to suggest other ways to write your program. Unfortunately, these excursions into "other ways" can be very time-consuming distractions. If you are tempted to try another "better, faster, or easier" way of doing things, at least be sure you understand the process involved. Blind use of any computer technique is certain to cause more trouble than it overcomes.

1-2 EFFECT OF YOUR PERSONAL BACKGROUND

Student concern about lack of experience and background when first learning to program a computer generally falls into three categories: experience level, area of interest or expertise, and age.

1-2.1 Experience Level

Experience has convinced the author that it is inconsequential whether you are a high school junior, college junior, graduate student, or professional. Students with no mathematics beyond high school algebra, as well as those who have completed courses in partial differential equations, have consistently mastered programming techniques. Prior industrial or educational experience does not seem to have any major effect on your ability to learn computer programming. The primary advantage to be gained from prior experience is an ability to satisfactorily define a project. Yet students with little background, either academic or professional, often formulate exciting problems that are extremely interesting and practical.

1-2.2 Area of Interest or Expertise

No one comes to a study of computer programming with built-in handicaps. A variety of experience lends unexpected benefits, since stereotyped concepts often found among civil engineers, botanists, nurses, wildlife biologists, psychologists, electronic technicians or business people tend to melt away as they work together with computer programming as a common bond. The computer not only provides a powerful tool for your specialty, it is also an important vehicle for enhancing communication among students and practitioners of the various disciplines.

此为试读，需要完整PDF请访问：www.ertongbook.com