# PASCAL

An Introduction to the Art and Science of Programming

Walter J. Savitch

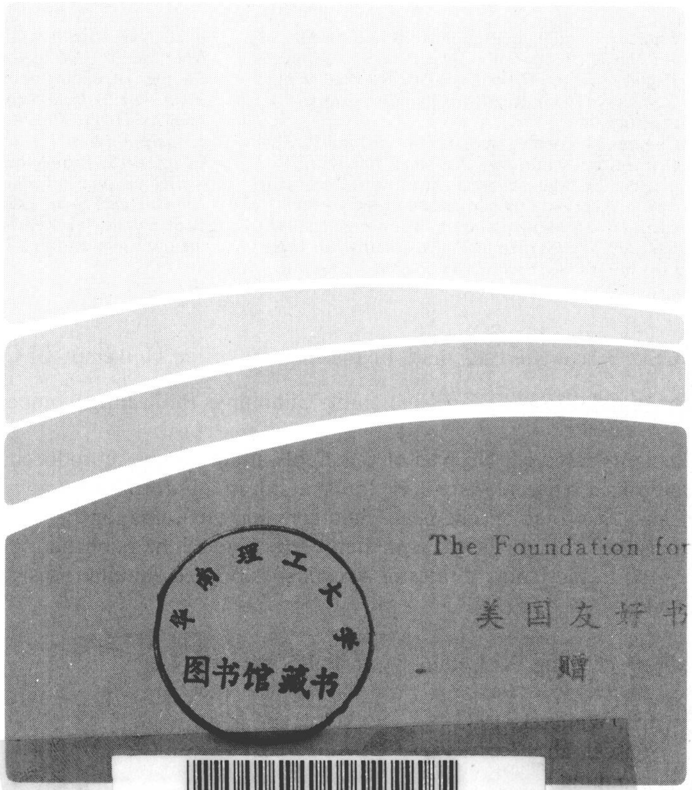
TP312
S267
(2)

9062485.
9062485

贈閱

# PASCAL

An Introduction to the Art and Science of Programming



The Foundation for Books to China
美国友好书刊基金会
赠 书

华南理工大学
图书馆藏书

E9062485

Walter J. Savitch
University of California, San Diego

The Benjamin / Cummings Publishing Co., Inc.
Menlo Park, California · Reading, Massachusetts
London · Amsterdam · Don Mills, Ontario · Sydney

*Sponsoring Editor:* Alan Apt
*Production Coordinator:* Charles Hibbard
*Book Designer:* Marilyn Langfeld
*Cover Designers:* Marilyn Langfeld and Betty Armstrong
*Cover Illustrator:* Betty Armstrong

*Acknowledgments:*

Chapter 7, ending quotation from F. P. Brooks, Jr., *The Mythical Man-Month, Essays on Software Engineering*, p. 116. Addison-Wesley Publishing Company (1975). Reprinted by permission of the publisher.
Chapter 12, opening quotation from John R. Ross, *Constraints on Variables in Syntax*, p. i, Ph.D. dissertation, Massachusetts Institute of Technology (1967). Reprinted by permission of the author.
Chapter 12 ending quotation, from Jorge Luis Borges, "The Garden of Forking Paths," in Jorge Luis Borges, *Selected Stories and Other Writings*, p. 25. New Directions Publishing Company (1964). Reprinted by permission of the publisher.
Chapter 14, ending quotation from Niklaus Wirth, *Algorithms + Data Structures = Programs*, title. Prentice-Hall (1976). Reprinted by permission of the publisher.
Chapter 15, ending quotation from B. W. Kernighan and P. J. Plauger, *The Elements of Programming Style*, 2nd ed., p. 117. McGraw-Hill Book Co. (1978). Reprinted by permission of the publisher and Bell Laboratories.

UCSD Pascal is a trademark of the Regents of the University of California.

Copyright © 1984 by The Benjamin/Cummings Publishing Company, Inc.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the publisher. Printed in the United States of America. Published simultaneously in Canada.

**Library of Congress Cataloging in Publication Data**

Savitch, Walter J., 1943–
Pascal, an introduction to the art and science of programming.

Includes bibliographies and index.
1. PASCAL (Computer program language) 2. Electronic digital computer—Programming. I. Title. II. Title: Pascal.
QA76.73.P2S28 1983 001.64′24 84-303
ISBN 0-8053-8370-0

ABCDEFGHIJ-HA-8987654

**The Benjamin / Cummings Publishing Company, Inc.**
2727 Sand Hill Road
Menlo Park, California 94025

# Preface

This book was designed for use in introductory programming courses that use the Pascal language. It can be used for courses as short as one quarter or as long as one academic year. The book includes both a thorough introduction to programming techniques and a complete description of the Pascal language. It assumes no previous knowledge of computers and no mathematics beyond high school algebra. Some additional details about the book are summarized below.

## Emphasizes Problem Solving and Programming Techniques

Chapters 1, 11, 12 and 15 are dedicated exclusively to problem solving and programming techniques. Every other chapter includes sections dedicated to such topics as top-down and other algorithm design techniques, programming style, testing techniques and debugging techniques. For ease of reference, these sections are tabulated in a separate table of contents. Additional discussion of these topics is integrated into the presentation of Pascal language constructs.

## Designed for Interactive Programming

This book can be used by those running programs in batch mode. However, the emphasis is on interactive use. All the programs were designed for use in an interactive environment.

## Many Sample Programs

Complete programs are presented very early. Thereafter, concepts are always illustrated with complete programs and procedures, rather than small fragments of code. Sample input and output are displayed along with the program text. Both long and short programs are presented. The longest program occupies four pages of text; the input and output extend to another two pages. Program displays of two or three pages are common.

## Introduces Procedures and Parameters Early

Like most newer texts on programming, procedures are introduced early, the reason being that this facilitates the teaching of modularity and top-down design.

Unlike other texts, this book adopts the approach without reservation. Not only procedures, but also a detailed discussion of parameters are presented very early, even before such basic Pascal constructs as ***if - then*** and loops. This allows virtually all the sample and exercise programs to be written using completely modular procedures. Programming without parameters invariably violates generally accepted modular design principles, and so forces students to practice techniques that must later be unlearned. Introducing parameters early eliminates the need to teach programming techniques that will later be rejected.

When we tested this approach in actual classes we found that it was not only possible to introduce parameters this early, but that the students found this approach easier than the traditional approaches. At the early stages of a programming course they have fewer new concepts to integrate and can thus concentrate without distraction on the notion of parameters.

## Self-Teaching Style

The chapters were designed so that they can be read by students before they attend lectures on the material. Material is covered completely, problems and questions are anticipated, and typical examples are given. This allows students to read ahead at their own pace and to come to lecture already primed to digest the material quickly. (This also makes the book suitable for self-teaching outside a classroom situation.)

## Covers Both Standard and UCSD Pascal

This book can be used in either a standard Pascal or a UCSD Pascal environment. All the UCSD Pascal detail is isolated into separate sections. Hence a standard Pascal user can read this book without seeing any reference to UCSD Pascal details. All the programs satisfy the ANSI/IEEE770X3.97-1983 Pascal standard (except for two short programs in optional sections about UCSD Pascal). They have all been compiled and tested using the Berkeley Pascal compiler and interpreter with the standard Pascal option. For a few programs dealing with files, alternative UCSD Pascal versions are needed and are given. However, those cases are the exceptions rather than the rule. If one programs with a view toward portability, the two dialects of Pascal are simply not that different. This does mean that the programming style in this book tends a bit more toward standard Pascal than UCSD Pascal. Still, a complete treatment of UCSD Pascal constructs is given in the appropriate chapters and not relegated to a brief appendix.

The point of including both versions of Pascal is not simply a compromise between two alternatives. One important programming technique that needs to be taught is portability. The two dialects of Pascal are close enough that they can both be taught without confusing the student. Yet they contain enough differences to illustrate portability techniques. If time permits, it is a good idea to cover some aspects of the "other" Pascal, whichever dialect that may be.

此为试读，需要完整PDF请访问：www.ertongbook.com

## Flexibility

The dependency chart on p. xxiii shows the possible orders in which the chapters may be covered without losing continuity. A bit more than the first half of the book is a core course that must be covered first. The rest of the book contains topics such as recursion, some software engineering topics, some numeric programming techniques and a substantial amount on data structures including records, files and pointers. These later chapters may be covered in almost any order, or a subset of the chapters may be chosen to form a shorter course. The chapter on text files has been divided into two parts to allow for two possibilities, either postponing the topic entirely until later in a course or briefly introducing it early and giving more detail later on. To add even more flexibility, sections with optional topics are included throughout the book.

## Self-Test and Interactive Exercises

Each chapter includes self-test exercises with answers in the back of the book. In addition to these self-test exercises, students should be encouraged to learn by trying out small test programs; that is, by "playing" with the system. Unfortunately, most beginning students have trouble thinking of things to try out. For this reason, interactive exercises are also included. They consist of suggestions for writing short programs that help give a feel for the concepts presented in the chapter. More traditional exercises are also included. The book contains more than 370 exercises of various kinds.

## Support Material

A chapter-by-chapter instructor's guide is available. All the programs in the text are available in machine readable form. Thus it is possible to have all the programs available for students to run without having to type them in.

## Acknowledgments

I have received much help and encouragement from numerous individuals and groups while preparing this book. I am very grateful to the University of Washington (Seattle) Computer Science Department and to the EECS Department at the University of California, San Diego for providing facilities and a conducive environment for writing the book. My thanks also go to the following individuals for suggestions and comments on the book (unfortunately the size of the list makes it impractical to catalogue the many important contributions that each individual made): Guy Almes, Bill Appelbe, Andrew Black, Jim Bunch, John Donald, Mike Denisevich, Patrick Dymond, Klaus Eldridge, Jim Gips, Richard Kaufmann, Alean Kirnak, Keith Muller, Vijay Rao, Robert Rother, Gary Sackett, Joe Sandmeyer, Robert Streett, Sue Sullivan, Martin Tompa, Dennis Volper, Ann Wilson and Chin Wu. Special thanks go to the many students in my programming classes who tested and helped correct preliminary versions of the book. Finally, I express my appreciation to all the individuals at Benjamin/Cummings who organized the reviewing and production of the book; in particular Charlie Hibbard, Jo Andrews and my Editor Alan Apt contributed much to the final product.

# Contents

Chapter 1
**Introduction to Computers and Problem Solving** 1

What a Computer Is 3
The Modern Digital Computer 3
The Notion of an Algorithm 6
Programs and Data 8
High Level Languages 9
The Software Hierarchy 12
Pragmatics of Running a Program 12
Designing Programs 13
Top-Down Design 13
*Summary of Terms* 16
*Self-Test and Interactive Exercises* 18
*Regular Exercises* 18
*References for Further Reading* 19

Chapter 2
**Introduction to Pascal** 21

The Pascal Language 23
The Notion of a Program Variable 23
Stepping through a Program 24
More about Assignment Statements 27
Data Types—An Introduction 28
Simple Output 31
Formatted Output 33
Input 35
Batch Processing (Optional) 37
Designing Input and Output Instructions 37
Names: Identifiers 39
Putting the Pieces Together 41
Introduction to Programming Style 42

Designing a Sample Program 43
Blaise Pascal (Optional) 45
Summary of Problem Solving and Programming Techniques 47
*Summary of Pascal Constructs* 47
*Self-Test and Interactive Exercises* 49
*Regular Exercises* 51

Chapter 3
**More Pascal and Problem Solving** 53

Naming Constants 55
Syntax Diagrams 56
Allowable Range for Integers 58
More about **real** Values 59
Type Compatibilities 61
The Types **char** and **string** 61
Arithmetic Expressions 62
Standard Functions 65
A Sample Program Using Functions 66
Comments 67
Another Sample Program 68
Designing a Sample Program 70
Testing and Debugging 74
UCSD Pascal (Optional) 77
Summary of Problem Solving and Programming Techniques 79
*Summary of Pascal Constructs* 80
*Self-Test and Interactive Exercises* 82
*Regular Exercises* 83
*References for Further Reading* 84

Chapter 4
**Procedures and Top-Down Design** 87

Simple Pascal Procedures 89
A Top-Down Design Example 89
Commenting Procedures 91
Testing Procedures 93
Top-Down and Bottom-Up Strategies 95
Variable Parameters 97
Parameter Lists 100
Choosing Parameter Names 103
Summary of Problem Solving and Programming Techniques 104
*Summary of Pascal Constructs* 104
*Self-Test and Interactive Exercises* 106
*Regular Exercises* 107

Chapter 5
**Procedures for Modular Design** 109
Local and Global Variables 111
Other Local Identifiers 113
Scope of a Local Identifier 115
Value Parameters 116
Mixed Parameter Lists 122
What Kind of Parameter to Use 124
Parameters and Global Variables 124
Self-Documenting Procedures 125
Designing a Sample Procedure 127
Summary of Problem Solving and Programming Techniques 130
*Summary of Pascal Constructs* 131
*Self-Test and Interactive Exercises* 133
*Regular Exercises* 134

Chapter 6
**Designing Programs That Make Choices** 137
The **case** Statement 139
Compound Statements 142
Nested Statements 143
**if - then - else** Statements 147
**if - then** Statements 149
Simple Boolean Expressions 150
Some Notes about Semicolons 152
More on Indenting 153
Nesting **if - then** and **if - then - else** Statements 153
UCSD Pascal—**string** Comparisons (Optional) 155
Guarded Commands—One Method for Designing Branches 155
Assertions 157
Designing a Sample Program 158
Summary of Problem Solving and Programming Techniques 163
*Summary of Pascal Constructs* 163
*Self-Test and Interactive Exercises* 165
*Regular Exercises* 166

Chapter 7
**Programming with Boolean Expressions** 169
The Type **boolean** 171
Complex Boolean Expressions 171
Evaluating Boolean Expressions 174
The **odd** Function 175
Introduction to Sets 176
Programming with Boolean Variables 178

A Longer Example 179
Boolean Input and Output 179
A Design Example 183
Iterative Enhancement 185
The Throwaway Program 185
Summary of Problem Solving and Programming Techniques 187
*Summary of Pascal Constructs* 187
*Self-Test and Interactive Exercises* 189
*Regular Exercises* 190

Chapter 8
**Looping Mechanisms** **193**

**repeat** Statements 195
Example—Testing a Procedure 196
Unrolling a Loop 196
**while** Statements 201
Infinite Loops 203
Invariant Assertions and Variant Expressions (Optional) 203
**for** Statements 207
Example—Summing a Series 211
Invariant Assertions and **for** Loops (Optional) 212
What Kind of Loop to Use 212
Example—Calendar Program 214
Debugging Loops 215
Designing a Loop Using Invariants (Optional) 219
Summary of Problem Solving and Programming Techniques 223
*Summary of Pascal Constructs* 224
*Self-Test and Interactive Exercises* 226
*Regular Exercises* 226
*References for Further Reading* 228

Chapter 9
**Functions** **229**

Use of Functions 231
A Sample Function Declaration 231
Computing Powers—A More Complicated Example 234
Side Effects 236
Functions That Change Their Minds 236
Boolean Valued Functions 237
Example—Testing for Primes 237
Some More Standard Functions (Optional) 239
UCSD Pascal—More **string** Functions and Procedures (Optional) 240
Random Number Generators 242
Using Pseudorandom Numbers 244
A Better Method for Scaling Random Numbers (Optional) 247

Standard Pascal—Functions and Procedures as Parameters (Optional) 248
Summary of Problem Solving and Programming Techniques 249
*Summary of Pascal Constructs* 250
*Self-Test and Interactive Exercises* 250
*Regular Exercises* 251
*References for Further Reading* 253

Chapter 10
**More on Data Types** **255**

Ordinal Types 257
UCSD Pascal—The Type **string** (Optional) 259
Subrange Types 259
The ***for*** and ***case*** Statements Revisited 262
The Functions **ord**, **chr**, **pred** and **succ** (Optional) 262
Enumerated Types (Optional) 264
Arrays—An Introduction to Structured Types 266
Type Declarations—A Summary 271
Type Equivalence 273
Example—Sorting an Array 273
Example—Partially Filled Array 275
Array Example with Noninteger Indexes 280
Storage Efficiency 280
Packed Arrays (Optional) 282
Standard Pascal—Packed Arrays of Characters (Optional) 283
Summary of Problem Solving and Programming Techniques 284
*Summary of Pascal Constructs* 285
*Self-Test and Interactive Exercises* 286
*Regular Exercises* 288

Chapter 11
**Program Design Methodology** **291**

Some Guidelines for Problem Solving 293
Writing Code 295
Testing and Debugging 295
Verification 302
Portability 303
Efficiency 304
Efficiency versus Clarity 306
Batch Processing 307
Coping with Large Programming Tasks 308
*Summary of Terms* 309
*Self-Test and Interactive Exercises* 310
*Regular Exercises* 311
*References for Further Reading* 312

此为试读，需要完整PDF请访问：www.ertongbook.com

Chapter 12
**Using Recursion** **313**
Example of a Recursive Function 315
A Closer Look at Recursion 317
Infinite Recursion 319
Stacks 320
Proving Termination and Correctness for Recursive Functions (Optional) 324
Recursive Procedures—A Simple Example 326
Binary Search—A More Complicated Example of Recursion 327
Towers of Hanoi—An Example of Recursive Thinking 331
Recursive versus Iterative Procedures and Functions 334
Forward Declarations (Optional) 335
Summary of Problem Solving and Programming Techniques 337
*Self-Test and Interactive Exercises* 338
*Regular Exercises* 339
*References for Further Reading* 340

Chapter 13
**Text Files and Secondary Storage** **341**
PART I 343
Text Files 343
Writing and Reading of Text Files 344
Standard Pascal 345
UCSD Pascal (Optional) 348
Avoid the Silent Program 351
*Self-Test and Interactive Exercises* 351
PART II 352
**readln** and **writeln** Reexamined 352
**eof** and **eoln** 354
Using a Buffer 357
Pragmatics of **eoln** and **eof** 358
Text Files as Parameters to Procedures 358
Portability 359
Basic Technique for Editing Text Files 360
Standard Pascal—Internal Files 361
Example—Editing Out Excess Blanks 361
Text Editing as a Programming Aid 365
Summary of Problem Solving and Programming Techniques 366
*Summary of Pascal Constructs* 366
*Self-Test and Interactive Exercises* 370
*Regular Exercises* 370

Chapter 14
**More Structured Data Types** **373**

Multidimensional Arrays 375
Grading—An Example Using a Two-Dimensional Array 377
Storage Efficiency 380
Introduction to Records 381
The Syntax of Simple Records 384
Hierarchies of Structured Types 385
The **`with`** Statement 386
Parimutuel Betting Odds—An Example Using Records 388
Standard Pascal—Use of Packed Arrays of Characters (Optional) 395
UCSD Pascal—Use of Strings in Records (Optional) 395
Variant Records (Optional) 396
More about Sets (Optional) 399
Choosing a Data Structure 403
Summary of Problem Solving and Programming Techniques 404
*Summary of Pascal Constructs* 405
*Self-Test and Interactive Exercises* 407
*Regular Exercises* 408
*References for Further Reading* 412

Chapter 15
**Solving Numeric Problems** **413**

A Hypothetical Decimal Computer 415
Binary Numerals (Optional) 419
Machine Representation of Numbers in Binary (Optional) 420
Extra Precision (Optional) 422
UCSD Pascal—Long Integers (Optional) 423
Sources of Error in **`real`** Arithmetic 423
Error Propagation 426
Example—Series Evaluation 427
Example—Finding a Root of a Function 428
Summary of Problem Solving and Programming Techniques 432
*Summary of Terms* 433
*Self-Test and Interactive Exercises* 433
*Regular Exercises* 434
*References for Further Reading* 435

Chapter 16
**More File Types** **437**

The General Notion of a File 439
File Variables 440
Standard Pascal—Opening Files 440
UCSD Pascal—Opening Files (Optional) 440

Windows and Buffer Variables 441
**put**—"Writing" to Files 442
The Procedure **write** 444
**get**—"Reading" from Files 444
The Procedure **read** 446
Example Using **put** and **get** 446
Files as Parameters to Procedures 449
Deciding What Type of File to Use 449
Standard Pascal—A File of Records Example 450
UCSD Pascal—**seek** (Optional) 451
Summary of Problem Solving and Programming Techniques 453
*Summary of Pascal Constructs* 454
*Self-Test and Interactive Exercises* 458
*Regular Exercises* 458
*References for Further Reading* 460

Chapter 17
**Dynamic Data Structures** **461**

The Notion of a Pointer 463
Pascal Pointers and Dynamic Variables 464
Manipulating Pointers 465
Nodes 468
The Pointer ***nil*** 469
Linked Lists—An Example of Pointer Use 470
**dispose** (Optional) 477
Doubly Linked Lists—A Variation on Simple Linked Lists 479
Trees 480
Implementation (Optional) 482
UCSD Pascal—**mark** and **release** (Optional) 487
Summary of Problem Solving and Programming Techniques 489
*Summary of Pascal Constructs* 490
*Self-Test and Interactive Exercises* 491
*Regular Exercises* 491
*References for Further Reading* 493

Appendix 1 The ***goto*** Statement 496
Appendix 2 Reserved Words 499
Appendix 3 Precedence of Operators 500
Appendix 4 Packed Structures 501
Appendix 5 Syntax Diagrams for Standard Pascal 502
Appendix 6 UCSD Pascal Details 517
Glossary 521
Answers to Self-Test Exercises 531
Index 543

# Chapter Sections Dedicated to Problem Solving and Programming Techniques

Chapter 1
**Introduction to Computers and Problem Solving**
What a Computer Is 3 / The Modern Digital Computer 3 / The Notion of an Algorithm 6 / Programs and Data 8 / High Level Languages 9 / The Software Hierarchy 12 / Pragmatics of Running a Program 12 / Designing Programs 13 / Top-Down Design 13 / Summary of Terms 16

Chapter 2
**Introduction to Pascal**
The Notion of a Program Variable 23 / Data Types—An Introduction 28 / Designing Input and Output Instructions 37 / Introduction to Programming Style 42 / Designing a Sample Program 43 / Summary of Problem Solving and Programming Techniques 47

Chapter 3
**More Pascal and Problem Solving**
Naming Constants 55 / A Sample Program Using Functions 66 / Comments 67 / Another Sample Program 68 / Designing a Sample Program 70 / Testing and Debugging 74 / Summary of Problem Solving and Programming Techniques 79

Chapter 4
**Procedures and Top-Down Design**
A Top-Down Design Example 89 / Commenting Procedures 91 / Testing Procedures 93 / Top-Down and Bottom-Up Strategies 95 / Choosing Parameter Names 103 / Summary of Problem Solving and Programming Techniques 104

Chapter 5
**Procedures for Modular Design**

Local and Global Variables 111 / What Kind of Parameter to Use 124 / Parameters and Global Variables 124 / Self-Documenting Procedures 125 / Designing a Sample Procedure 127 / Summary of Problem Solving and Programming Techniques 130

Chapter 6
**Designing Programs That Make Choices**

More on Indenting 153 / Guarded Commands—One Method for Designing Branches 155 / Assertions 157 / Designing a Sample Program 158 / Summary of Problem Solving and Programming Techniques 163

Chapter 7
**Programming with Boolean Expressions**

Programming with Boolean Variables 178 / A Longer Example 179 / A Design Example 183 / Iterative Enhancement 185 / The Throwaway Program 185 / Summary of Problem Solving and Programming Techniques 187

Chapter 8
**Looping Mechanisms**

Example—Testing a Procedure 196 / Unrolling a Loop 196 / Infinite Loops 203 / Invariant Assertions and Variant Expressions (Optional) 203 / Example—Summing a Series 211 / Invariant Assertions and ***for*** Loops (Optional) 212 / What Kind of Loop to Use 212 / Example—Calendar Program 214 / Debugging Loops 215 / Designing a Loop Using Invariants (Optional) 219 / Summary of Problem Solving and Programming Techniques 223

Chapter 9
**Functions**

Side Effects 236 / Boolean Valued Functions 237 / Example Testing for Primes 237 / Random Number Generators 242 / Using Pseudorandom Numbers 244 / A Better Method for Scaling Random Numbers (Optional) 247 / Summary of Problem Solving and Programming Techniques 249

Chapter 10
**More on Data Types**

Arrays—An Introduction to Structured Types 266 / Example—Sorting an Array 273 / Example—Partially Filled Array 275 / Array Example with Noninteger Indexes 280 / Storage Efficiency 280 / Summary of Problem Solving and Programming Techniques 284

Chapter 11
**Program Design Methodology**

Some Guidelines for Problem Solving 293 / Writing Code 295 / Testing and Debugging 295 / Verification 302 / Portability 303 / Efficiency 304 / Efficiency versus Clarity 306 / Batch Processing 307 / Coping with Large Programming Tasks 308 / Summary of Terms 309

Chapter 12
**Using Recursion**

Example of a Recursive Function 315 / A Closer Look at Recursion 317 / Infinite Recursion 319 / Stacks 320 / Proving Termination and Correctness for Recursive Functions (Optional) 324 / Recursive Procedures —A Simple Example 326 / Binary Search—A More Complicated Example of Recursion 327 / Towers of Hanoi—An Example of Recursive Thinking 331 / Recursive versus Iterative Procedures and Functions 334 / Summary of Problem Solving and Programming Techniques 337

Chapter 13
**Text Files and Secondary Storage**

Avoid the Silent Program 351 / Using a Buffer 357 / Portability 359 / Basic Technique for Editing Text Files 360 / Example—Editing Out Excess Blanks 361 / Text Editing as a Programming Aid 365 / Summary of Problem Solving and Programming Techniques 366

Chapter 14
**More Structured Data Types**

Grading—An Example Using a Two-Dimensional Array 377 / Storage Efficiency 380 / Hierarchies of Structured Types 385 / Parimutuel Betting Odds—An Example Using Records 388 / Choosing a Data Structure 403 / Summary of Problem Solving and Programming Techniques 404

Chapter 15
**Solving Numeric Problems**

A Hypothetical Decimal Computer 415 / Binary Numerals (Optional) 419 / Machine Representation of Numbers in Binary (Optional) 420 / Sources of Error in `real` Arithmetic 423 / Error Propagation 426 / Example—Series Evaluation 427 / Example—Finding a Root of a Function 428 / Summary of Problem Solving and Programming Techniques 432 / Summary of Terms 433

此为试读，需要完整PDF请访问：www.ertongbook.com