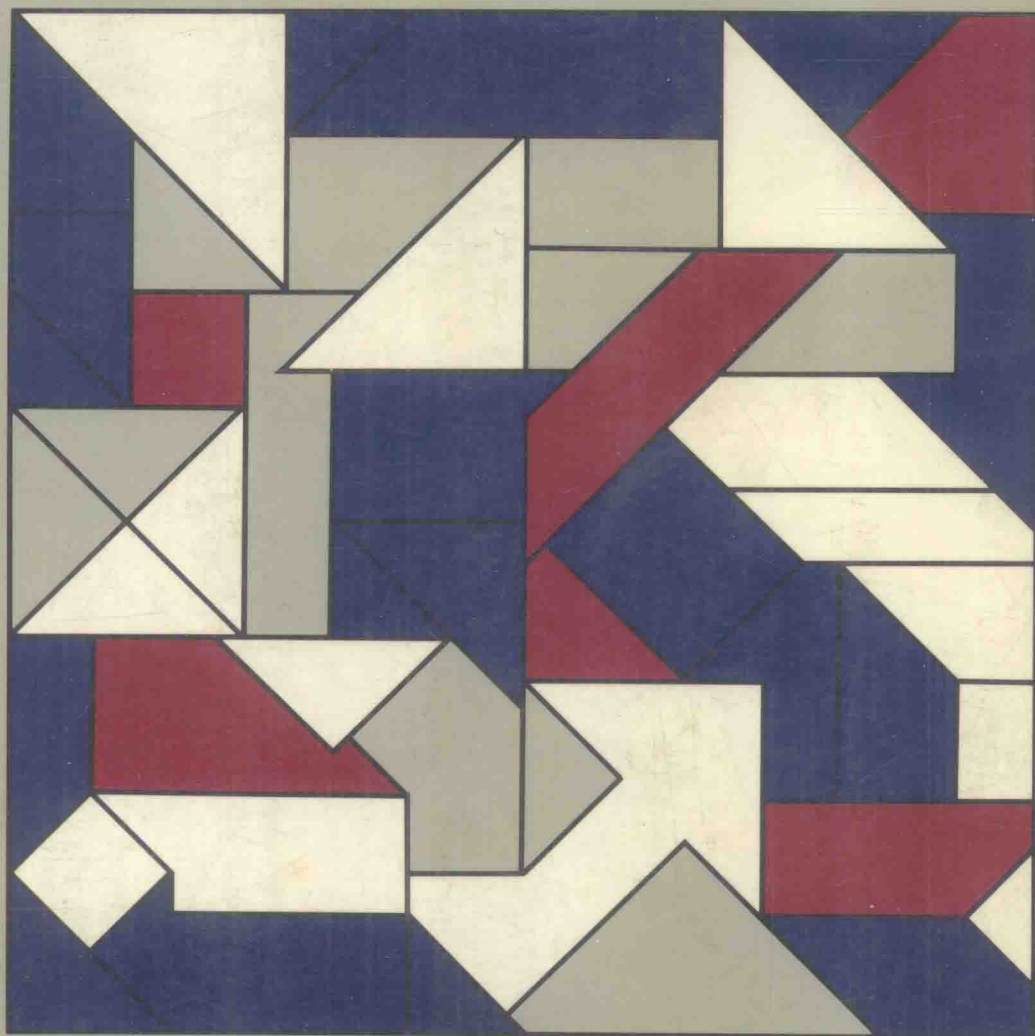


PASCO STRUCTURE AND STYLE



RICHARD LAMB

PASCAL STRUCTURE AND STYLE

RICHARD LAMB



The Benjamin/Cummings Publishing Company, Inc.
Menlo Park, California • Reading, Massachusetts •
Don Mills, Ontario • Wokingham, U.K. • Amsterdam •
Sydney • Singapore • Tokyo • Mexico City • Bogota •
Santiago • San Juan •

Sponsoring Editor: Alan Apt
Production Supervisor: Larry Olsen
Book and Cover Designer: Rodelinde Albrecht
Artist: Robert Bausch

Designs at the opening of Chapters 1, 2, 3, and 5 by Robert Bausch.
Designs at the opening of Chapters 4, 6 through 19 by Jan Arthur Lee.
Cover drawing by Robert Bausch.

Photo Credits:

Figure 1.1 from Iowa State University.
Figure 1.2 from Intel Corp.
Figure 1.3 from IBM Corp. and Apple Computer, Inc.
Figure 1.6 from Historical Pictures Service, Chicago.

Copyright © 1986 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

Lamb, Richard (Richard C.)

Pascal structure and style.

(Benjamin/Cummings series in structured programming)

Includes index.

1. PASCAL (Computer program language) 2. Structured programming. I. Title. II. Series.

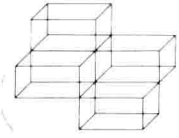
QA76.73.P2L345 1986 005.13'3 85-18511

ISBN 0-8053-5830-7

BCDEFGHIJ-DO-89876

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

PASCAL: STRUCTURE AND STYLE



Benjamin/Cummings Series in Structured Programming

- R. Billstein, S. Libeskind, J. Lott, *MIT Logo for the Apple* (1985)
G. Booch, *Software Engineering with Ada* (1983)
D. M. Etter, *Structured FORTRAN 77 for Engineers and Scientists* (1983)
D. M. Etter, *Problem Solving with Structured FORTRAN 77* (1984)
D. M. Etter, *WATFIV: Structured Programming and Problem Solving* (1985)
P. Helman, R. Veroff, *Intermediate Problem Solving and Data Structures* (1985)
A. Kelley, I. Pohl, *A Book on C* (1984)
M. Kittner, B. Northcutt, *BASIC: A Structured Approach* (1985)
R. Lamb, *Pascal: Structure and Style* (1985)
S. Perry, *Beginning COBOL: An Interactive and Structured Approach* (1985)
W. Savitch, *Pascal: An Introduction to the Art and Science of Programming* (1984)
R. Sebesta, *Structured Assembly Language Programming for the VAX 11* (1984)
R. Sebesta, *Structured Assembly Language Programming for the PDP-11* (1984)

*To my mother and father,
who have given me much,
especially a relationship with my Father,
who has given me everything.*

Preface

Teaching computer programming has changed rapidly in the past few years, in large part because of the widespread availability and usefulness of computers in society. Scientists and mathematicians are no longer the only people who are using computers, and computer literacy is quickly becoming the most necessary skill that college graduates need to find a job. Many students who are not computer science majors are taking introductory computer classes in order to familiarize themselves with the field. In fact, many students have already taken computer programming courses in high school.

Unique Approach

This creates a problem. Most text books for the introductory programming course assume that students have had no experience in front of a computer and must learn everything for the first time. Other programming books assume the reader is fluent in one language and attempt to teach the reader in a minimum amount of time how to program in a second language. However, no current text book on the Pascal language assumes a mixed audience. This book assumes that the students using it will include students who have never studied any programming language before as well as students who have taken one or more programming courses in BASIC. This book does not assume that the student has a prior knowledge of BASIC, so the material will be easily understood by the novice. However, the BASIC programmer often starts with a disadvantage because of habits she or he has learned while programming with the limitations of BASIC, especially the unstructured dialects of BASIC. The book provides a Hints for BASIC Programmers section in each chapter to help encourage the good aspects and to help correct the bad aspects of prior programming experience in BASIC.

Emphasis on Examples

The current trend in computer science education focuses on teaching abstract problem-solving skills combined with the syntax and programming techniques of a particular language. The approach of this book follows this trend. Each

chapter contains material that extends the student's knowledge of the syntax of Pascal as well as discussion about and complete examples of problem solving using the language elements covered in the discussion. The programming examples are interesting and are taken from common uses of the computer. The problems are discussed thoroughly; each is solved through rigorous application of the top-down approach to algorithmic development through the use of pseudocode. The problem is analyzed and developed in parts and then brought together, so the student is able to see each step in the process from problem statement to solution.

Important Features

Programming with Style: Chapter 5 is entirely devoted to development of a consistent and elegant programming style. This same consistent style is exemplified in the programs and program fragments throughout the text.

Common Errors and Debugging: Each chapter contains a discussion of syntax and execution errors that are commonly encountered after the introduction of the new material. This section also contains defensive programming hints to aid in program debugging.

Hints for BASIC Programmers: In each of the first 15 chapters, a section is devoted entirely to a comparison and contrast of the elements in BASIC that translate into Pascal. Often, the BASIC programmer's experience is a disadvantage, and bad habits need to be explicitly discouraged.

Writing Large Programs: Chapter 17, Writing and Documenting Large Programs, is a unique chapter on the issues involved in the development of large programs.

Programming Examples: Except for Chapter 1, 2, and 5, each chapter contains at least one and often two or more complete programming examples. These special sections give the problem statement, the initial main algorithm, algorithms for subtasks, and the individual procedures, often refined several times. The sections include the final program to be executed and actual input and output files.

Algorithmic Development: Each programming example illustrates fully the concept of top-down algorithmic development through the use of pseudocode. The pseudocode is informal English, allowing students to have a model of algorithmic development without being handicapped by unfamiliar formal structures.

Exercises and Self Tests: Each chapter contains many Self Tests at the ends of sections as well as Exercises at the end of the chapter. The answers to the Self Tests are given at the end of the book. The Exercises contain error-checking or analysis problems as well as a wide range of programming problems.

Instructor's Guide: Answers to Exercises and discussion about them are included in the *Instructor's Guide*.

All the programs and program fragments have been executed on a DEC-20 mainframe and are written in Standard Pascal as defined in the *Pascal User Manual and Report*, Second Edition (Jensen and Wirth, Springer-Verlag, 1975).

Organization

Each chapter is meant to contain enough material for one to three lectures in a normal semester or quarter course. (Usually, one-quarter courses will not finish the book.) The first chapter introduces computers and the need for programming languages. The second chapter introduces Pascal and the building blocks of the language.

The third and fourth chapters introduce variables and procedures, respectively. The first control structure taught is the procedure, in order to emphasize program breakdown and good design. The sixth chapter continues with the emphasis on procedural development as value and variable parameters are introduced. In this way, complete and consistent parameterization of procedures is emphasized and exemplified.

The fifth chapter, Programming with Style, is a detour from the introduction of the syntax of Pascal to the issue of *style*. This chapter collects many programming proverbs and motivates the student to write consistent, elegant code that is well commented and structured. Material in this chapter is referred to repeatedly throughout the text as a reminder.

Chapters 7, 8, 9, and 11 focus on other control structures and their use. The FOR loop, IF statement, REPEAT and WHILE loops, and the CASE statement are introduced, with particular emphasis on which control structure is most appropriate in particular situations.

Chapter 10 does for input and output issues what Chapter 5 does for the issue of style. It collects what has been mentioned up to that point and discusses how to use multiple input and output files.

Chapter 12 begins the discussion of data types and data structures. Chapter 12 introduces the notion of a type. Chapters 13 and 14 introduce arrays and multidimensional arrays. Chapter 15 discusses the concept of a record and illustrates the multipurpose data structure, the array of records. Optional Chapter 16 details the use of structured files and sets.

Chapter 17 provides a unique approach to the issue of writing large programs. It focuses on the trade-off between the complexity of the control structure and the complexity of the data structure needed to solve a large problem, and it outlines steps to make each of the control and data structures appropriate to the given problem. Chapter 17 considers a large problem, writing a database manager, and outlines its data structure and control structures.

Chapters 18 and 19 complete the discussion of the Pascal language with the topics of recursion and pointers. The disadvantages as well as the advantages of recursive control and data structures are presented.

Acknowledgments

Many individuals contributed to the development of this text. First, I am grateful to the members of the Department of Computer Science at Stanford University. Many of the ideas presented here came as a result of learning and teaching there. I would especially like to thank Denny Brown, Arthur Keller, Jock MacKinlay, Rich Pattis, and Stuart Reges for much of what I know about teaching computer programming. Arthur Keller and Stuart Reges were especially instrumental in my decision to write this book and its early development.

I would also like to express my gratitude to those who reviewed the manuscript and who gave very helpful comments at various stages in its development: James Gips, Boston College; Peter Flanagan, Boston University; Michael Main, University of Colorado; Tom Lynch, Northwestern University; Tom Bailey, University of Wyoming; Dave Lamb and Martin Widyono, Stanford University.

I also am indebted to individuals in Stanford Christian Fellowship who have played a part in the development of this text. Dave Lamb, Rob Perkins, and Martin Widyono have been helpful as teaching assistants and fellow instructors in developing examples throughout the text. Thanks also go to the many graders and other helpers with whom I have worked: Nate Bacon, Sue Dante, Ruben DiRado, Michael Dittmar, Kevin Gillette, Bill Glad, Curtis Gruenler, Mark Kennedy, Michael O'Brien, Rae Lee Olson, Katie Stock, and Kathy Wolfe. The text, like teaching itself, has been something of a team effort, and I am grateful to Sue Dante, Tom Nygren, Alissa Riper, Liz Argetsinger, and Mark Hornbostle for help in the typing of the manuscript. Greg Read provided helpful nontechnical advice throughout the development process. Finally, I am grateful for the prayers of the Christian Fellowships at Stanford University and University of California, Santa Cruz, as I began and completed this project.

Richard Lamb

Brief Contents

-
1. Introduction to Programming 1
 2. Introduction to Pascal 21
 3. Simple Number Crunching: Variables 39
 4. Extending the Language: Procedures 71
 5. Programming with Style 99
 6. Procedures, Parameters, and Functions 119
 7. Definite Repetition 151
 8. Conditional Execution 195
 9. Conditional Repetition 239
 10. Input and Output 281
 11. Conditional Execution: CASE 303
 12. Extending the Language: Types 323
 13. Storing Data: Arrays 345
 14. Multidimensional Arrays 393
 15. Storing Data: Records 419
 16. More Data Structures: Files and Sets 473
 17. Writing and Documenting Large Programs 497
 18. Recursion 517
 19. Pointers and Lists 541

Contents

1. Introduction to Programming 1

Adding Machines, Washing
Machines, and Computers, 2
Computer Hardware, 5
Why Pascal? 7
Algorithmic Refinement and
Pseudocode, 9
Program Execution, 12

SELF TEST, 17
Pascal and BASIC, 18
Hints for BASIC Programmers, 18
Summary, 19
New Terms, 20
Exercises, 20

2. Introduction to Pascal 21

Building Blocks, 22
Reserved Words, 22
Special Symbols, 23
Numbers, 23
Character Strings, 24
Identifiers, 25
SELF TEST, 26
Separators, 26
Program Syntax, 27

SELF TEST, 28
Statements, 28
Empty Statements, 30
Common Errors and Debugging, 31
Hints for BASIC Programmers, 34
Summary, 35
New Terms, 35
Exercises, 36

3. Simple Number Crunching: Variables 39

Variable Declarations, 40
Assignments and Expressions, 42
SELF TEST, 44

SELF TEST, 45
Writing Results, 45
SELF TEST, 47

Constants, 48	Programming Example 3.2, 60
SELF TEST, 49	Common Errors and Debugging, 63
Reading Values from Input, 49	Hints for BASIC Programmers, 65
SELF TEST, 53	Summary, 66
Standard Functions, 54	New Terms, 67
Programming Example 3.1, 56	Exercises, 67

4. Extending the Language: Procedures 71

The Procedure Call, 72	Common Errors and Debugging, 91
SELF TEST, 75	Hints for BASIC Programmers, 92
The Procedure Declaration, 75	Summary, 94
Programming Example 4.1, 77	New Terms, 95
Programming Example 4.2, 84	Exercises, 95
Global and Local Variables, 89	

5. Programming with Style 99

Program Style, 101	<i>Field Widths</i> , 108
<i>Indentation and Blank Lines</i> , 101	<i>Files and Input/</i>
<i>Capitalization</i> , 102	<i>Output Procedures</i> , 110
<i>Identifiers</i> , 103	User Friendliness, 111
<i>Constants</i> , 104	Common Errors and Debugging, 112
<i>Comments</i> , 104	Hints for BASIC Programmers, 112
<i>Procedures</i> , 106	Summary, 112
<i>Semicolons before ENDS</i> , 107	New Terms, 113
Output Style, 108	Exercises, 114

6. Procedures, Parameters, and Functions 119

Value Parameters, 122	Variables and Parameters, 128
SELF TEST, 125	<i>Local Versus Global</i> , 128
Variable Parameters, 126	<i>Parameters Versus</i>
SELF TEST, 128	<i>Global Variables</i> , 129

Variable Versus Value Parameters, 132
 Blocks, Scope and Activations, 133
 SELF TEST, 135
 Programming Example 6.1, 136
 Functions, 140
 Character Variables, 141

Programming Example 6.2, 142
 Common Errors and Debugging, 145
 Hints for BASIC Programmers, 146
 Summary, 147
 New Terms, 147
 Exercises, 147

7. Definite Repetition 151

Repetition, 152
 SELF TEST, 154
 Compound Statements, 154
 How Does It Work? 156
 SELF TEST, 159
 Nested Repetition, 159
 SELF TEST, 162
 Iteration, 162
 SELF TEST, 163
 Programming Example 7.1, 164

More on Character Variables, 167
 Programming Example 7.2, 169
 Programming Example 7.3, 174
 Programming Example 7.4, 177
 Common Errors and Debugging, 183
 Hints for BASIC Programmers, 188
 Summary, 190
 New Terms, 190
 Exercises, 190

8. Conditional Execution 195

IF-THEN-ELSE, 196
 SELF TEST, 200
 Nesting, 200
 SELF TEST, 205
 Assertions (Optional), 205
 Boolean Operators
 and Expressions, 207
 SELF TEST, 210
 Boolean Variables and Constants, 210
 Boolean Functions, 214
 SELF TEST, 216

Efficiency and Style, 216
 Random Numbers and
 Decision Making, 218
 SELF TEST, 220
 Programming Example 8.1, 220
 Common Errors and Debugging, 229
 Hints for BASIC Programmers, 232
 Summary, 234
 New Terms, 235
 Exercises, 235

9. Conditional Repetition 239

The WHILE Loop, 241

SELF TEST, 242

Eoln and Eof, 242

SELF TEST, 247

Invariant Assertions (Optional), 247

Programming Example 9.1, 248

Programming Example 9.2, 253

The REPEAT Loop, 259

SELF TEST, 260

Comparative Advantages: FOR,

WHILE, REPEAT, 261

Programming Example 9.3, 263

Common Errors and Debugging, 271

Hints for BASIC Programmers, 273

Summary, 275

New Terms, 276

Exercises, 276

10. Input and Output 281

Multiple Input and Output Files, 283

SELF TEST, 286

Programming Example 10.1, 286

Programming Example 10.2, 289

Common Errors and Debugging, 295

Hints for BASIC Programmers, 297

Summary, 297

New Terms, 298

Exercises, 298

11. Conditional Execution: CASE 303

The CASE Statement, 304

SELF TEST, 307

Programming Example 11.1, 307

When to Use CASE or IF, 311

SELF TEST, 312

Increasing the Power of CASE, 312

Programming Example 11.2, 314

Common Errors and Debugging, 319

Hints for BASIC Programmers, 319

Summary, 320

New Terms, 321

Exercises, 321

12. Extending the Language: Types 323

Enumerated Type Declarations, 324

SELF TEST, 326

Use of Enumerated Types, 327

SELF TEST, 331

- Subrange Types, 332
 - SELF TEST, 335
- Programming Example 12.1, 335
- Debugging and Types, 339
 - SELF TEST, 339
- Types and Data Abstraction, 340
- Common Errors and Debugging, 340
- Hints for BASIC Programmers, 342
- Summary, 342
- New Terms, 342
- Exercises, 343

13. Storing Data: Arrays 345

- Introduction Arrays, 346
- Array Declarations, 348
 - SELF TEST, 351
- Use of Arrays: Reading and Initializing, 351
- Use of Arrays:
 - Sequential Searching, 353
 - SELF TEST, 354
- Use of Arrays:
 - Procedure ReadWord, 354
 - SELF TEST, 356
- Array Indexing, 356
- Use of Arrays: Bubble Sort, 358
 - SELF TEST, 360
- Use of Arrays: Binary Search, 360
 - SELF TEST, 368
- Packed Arrays, 368
 - SELF TEST, 371
- Programming Example 13.1, 371
- Programming Example 13.2, 376
- Common Errors and Debugging, 385
- Hints for BASIC Programmers, 387
- Summary, 389
- New Terms, 389
- Exercises, 389

14. Multidimensional Arrays 393

- Declaring Multidimensional Arrays, 394
- Accessing Multidimensional Arrays, 397
 - SELF TEST, 398
- Programming Example 14.1, 398
- Programming Example 14.2, 405
- Common Errors and Debugging, 410
- Hints for BASIC Programmers, 411
- Summary, 413
- Exercises, 413

15. Storing Data: Records 419

- Declarations, 422
- Usage, 424
 - SELF TEST, 426
- Arrays of Records, 426

SELF TEST, 429
 Using Large Data Structures, 429
 Sorting, 435
 SELF TEST, 436
 Programming Example 15.1, 436
 Variant Record Structures, 446

Programming Example 15.2, 449
 Common Errors and Debugging, 467
 Hints for BASIC Programmers, 468
 Summary, 468
 New Terms, 469
 Exercises, 469

16. More Data Structures: Files and Sets 473

Files, 474
 Text Files Reconsidered, 475
 SELF TEST, 475
 Two Procedures: Get and Put, 476
 SELF TEST, 476
 File Parameters, 477
 SELF TEST, 477
 Files of Other Types, 477
 Programming Example 16.1, 479

Sets, 482
 SELF TEST, 484
 Programming Example 16.2, 485
 Programming Example 16.3, 489
 Common Errors and Debugging, 493
 Summary, 494
 New Terms, 494
 Exercises, 494

17. Writing and Documenting Large Programs 497

Data Structure and
 Stepwise Refinement, 499
 SELF TEST, 502
 Control Structures and
 Stepwise Refinement, 502
 SELF TEST, 504
 The Trade-Off: Control and
 Data Structures, 504

Programming Example 17.1, 506
 Common Errors and Debugging, 513
 Summary, 515
 New Terms, 515
 Exercises, 516

18. Recursion 517

Single Procedure Recursion, 518
 SELF TEST, 520
 SELF TEST, 523

Programming Example 18.1, 524
 Programming Example 18.2, 525
 SELF TEST, 527