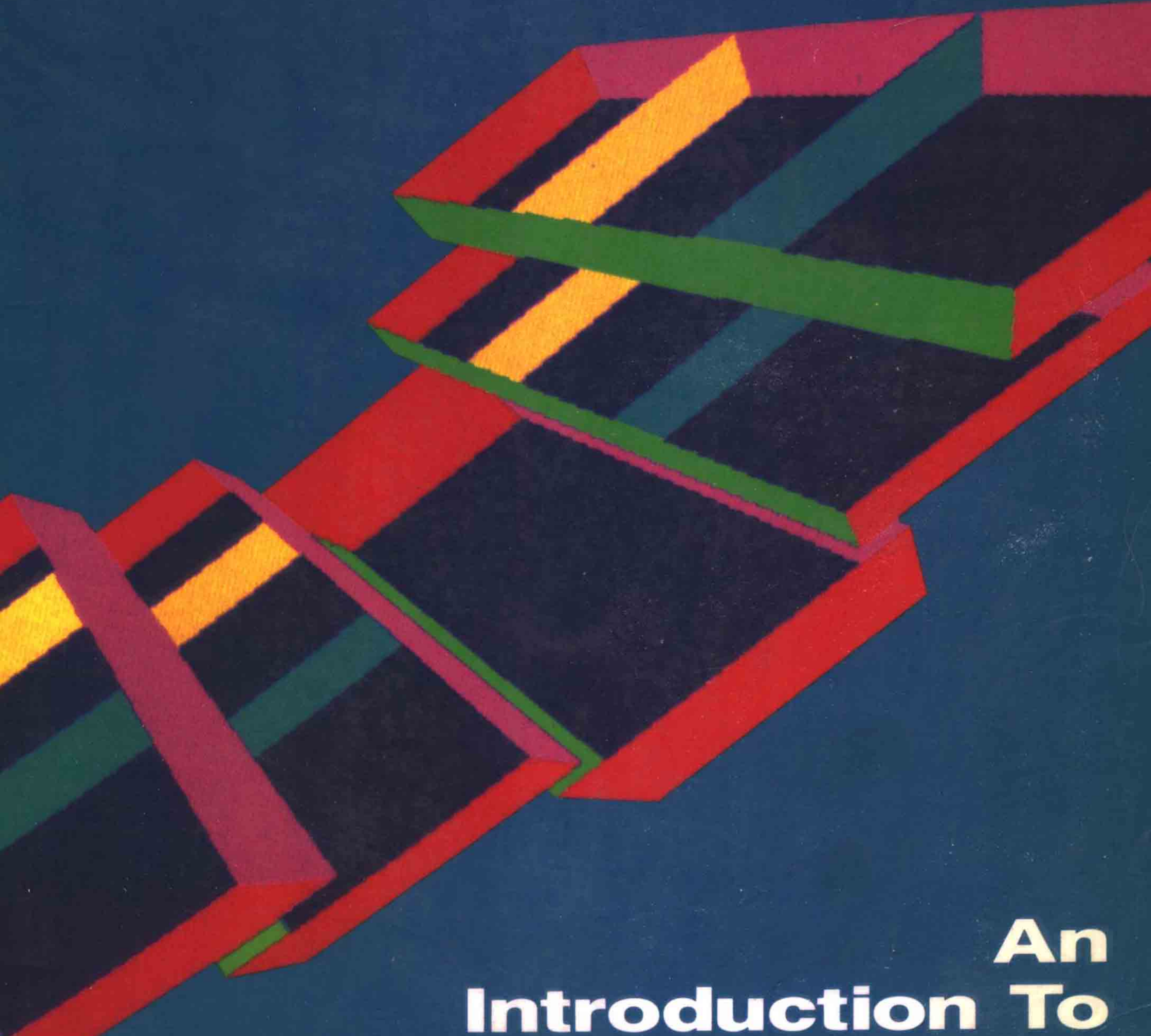
USING PASCAL



An Introduction To Computer Science I

DAVID D. RILEY

USING PASCAL An Introduction To Computer Science I

DAVID D. RILEY

University of Wisconsin
La Crosse

BOYD & FRASER PUBLISHING COMPANY
BOSTON

PREFACE

Recent developments in computer hardware and software are driving a change in introductory computer science courses. These developments range from new computer workstations that provide a productive environment for software design to new programming languages that incorporate up-to-date facilities for software engineering. The introductory computer science course must mature along with the profession. New techniques for software development must be integrated into sound pedagogical experience.

ABOUT THIS BOOK

Using Pascal: An Introduction to Computer Science I is an examination of today's computer science fundamentals, consistent with the ACM's 1984 guidelines for a CS 1 course. This text recognizes that tomorrow's software designer will require knowledge that goes beyond mere familiarity with programming languages.

This text also reflects a philosophy that students' performance is closely related to the instructor's expectations. Student feedback, based on class testing of preliminary editions of this book, suggest that this presentation is both challenging and realistic.

Since many students entering college have some familiarity with programming in high-level languages, the organization of this text is based upon programming tools and techniques, rather than the features of some programming language. These tools include intelligent use of control structures, data structures, design, good programming style, and debugging technique.

DISTINGUISHING FEATURES

Development of Tools and Techniques

Every programmer needs a tool box. This book includes numerous tools for the developing software engineer. Among these tools are syntax diagrams, debugging techniques, assertions, procedure hierarchy charts, block structure diagrams, control flow diagrams, pseudolanguages, and numerous widely-used algorithms.

Attention to Programming Style and Design

To highlight the importance of good programming style many separate "Programming with Style" sections are sprinkled throughout the text. Each describes the **how** and the **why** of some aspect of competent program design. Issues of good algorithmic design are similarly highlighted in "Designing with Wisdom" sections appearing throughout the book.

Extensive Well-Chosen Examples

Many students learn by example. Therefore, the text includes numerous carefully selected examples, averaging over two complete programs per chapter with many other algorithms and procedures. The programs have all been compiled and executed. Each major concept is illustrated by one or more examples. Programmers often model their programs from examples drawn from texts. A concerted effort has been made to choose examples that not only support the material, but serve as good models for reference use as well.

Assertions

In this text assertions are introduced immediately and used throughout. Pre- and postconditions are included in procedures. Loop invariants are shown as a means for designing loops. Assertions are among the most important documentation and design tools currently available and represent a disciplined, precise approach to software development. Students find this approach both challenging and rewarding.

Problem Definition Form

A six-part problem definition form is used to illustrate the importance of precise and complete definitions. This form includes input specifications, output specifications, error handling, and a sample program execution. Major programming examples and programming project assignments are consistently presented in this form.

Terminology

The computer scientist needs an extensive vocabulary. Terms are boldfaced and listed at the end of every chapter. An extensive glossary at the end of the text includes the definitions of boldfaced as well as other important terms.

Exercises

Each chapter contains numerous student exercises and programming projects that form a review of the chapter. Selected answers appear in the back of the book for student review. The solutions to all unanswered exercises and projects are in the Instructor's Manual.

Class Tested

Preliminary versions of this text were successfully used in classes at the University of Wisconsin–La Crosse over two semesters. Student suggestions have resulted in important improvements in pedagogy and accuracy.

Standard Pascal

All programs have been written to conform to the ISO standard. They have also been compiled and executed using Turbo Pascal™ a product of Borland International. All changes that are necessary to conform to Turbo Pascal requirements are included as comments in the code.

Turbo Pascal

Special sections, entitled "For Turbo Users," highlight the significant features of Turbo Pascal. These sections are included throughout the text to point out differences between the ISO standard and Turbo and to present Turbo extensions to the language.

ANCILLARY MATERIALS

A comprehensive instructor's support package accompanies **Using Pascal: An Instroduction to Computer Science I**. These ancillaries are available upon request from Boyd & Fraser.

Instructor's Manual

Material in the Instructor's Manual follows the organization of the text. Chapters of the Instructor's Manual include

- Lecture Outline
- General Comments
- Chapter Goals
- Solutions to Exercises
- Programming projects solutions
- Quick Challenge Questions

Included in the Instructor's Manual is an extensive collection of 135 transparency masters. These masters include selected figures, tables, and program segments from the text.

ProTest

Boyd & Fraser's fourth generation test generating program. ProTest. has been designed specifically to accompany this text.

DATA ABSTRACTION AND STRUCTURE: AN INTRODUCTION TO COMPUTER SCIENCE II

Using Pascal: An Introduction to Computer Science I is the first of a pair of texts covering the ACM recommendations for CS 1 and CS 2. The second book, entitled **Data Abstraction and Structure: An Introduction to Computer Science II**, is also available from Boyd & Fraser. This second text explores the use of data abstraction in software design. The presentation includes abstract data types, external modules, generic data types, and object-oriented design. Numerous data structures, such as stacks, queues, sequences, strings, linked lists, and trees, are examined as strategies for implementing abstractions.

ORGANIZATION

The Programming Process

From the very first chapter programming is presented as a four-step process. Chapter 1 defines this process as

1) Problem definition
2) Algorithm design
3) Program coding
4) Program maintenance

The importance of problem definitions is emphasized in the first chapter and continually reinforced throughout the text.

An Introduction to the Complete Pascal Program

Chapter 2 focuses on data. Introductory data types are presented in terms of their domain and operations, consistent with modern forms for defining abstract data types. This chapter also presents syntax diagrams, a formalism used throughout the text to define language structures. The subset of Pascal presented within Chapter 2 is sufficient for students to write elementary programs.

Program Design Techniques

Chapter 3 is an introduction to software design. Several examples illustrate top down design using a careful stepwise refinement. This chapter also explores alternative pseudolanguages and begins the study of debugging issues that continues throughout the remainder of the book.

(Control) Structured Programming

Chapter 4 begins the examination of structured programming. The topics of sequential execution and abstraction (in the form of procedures with parameters) are presented. The focal point of this presentation is top down design using these control structures. This emphasis is continued in Chapter 5.

More Structured Programming

Chapter 5 completes the survey of control structures by examining selection (in the form of the **if** instruction) and repetition (the **while** instruction.) As in Chapter 4, this presentation concentrates on the use of these forms of control in top down design. The early presentation of the basic control structures from Chapters 4 and 5 allows ample opportunity for students to exercise their programming skills.

Logic and Programming

Programming is rooted in logic. The current trends in software engineering and new language design suggest that these roots are becoming even more important. Chapter 6 is a brief look at the direct application of logic to programming in a procedural language, such as Pascal. This chapter incorporates critical material often omitted from introductory computer science texts.

The Procedure

Chapter 7 expands upon the earlier presentation of procedures by including an extensive look at intelligent use of parameter passage and scope of variables in software design. Functions and built-in procedures are also discussed in this chapter.

Input and Output of Text Files

Chapter 8 concentrates on the topic of text I/O. This chapter examines both interactive I/O and files, using the standard procedures of Pascal.

More About Selection

Chapters 9 completes the presentation of selection control structures. This chapter looks at methods of combining **if** instructions, as well as the **case** instruction. An examination of how to choose from alternate selection structures is included.

More About Repetition

Chapter 10 covers repetition, including the **repeat** instruction. Loop invariants are presented, and a straight forward six-step procedure for using invariants to design loops is utilized.

Introduction to Data Abstraction

Chapter 11 is an introduction to data abstraction, using the **const** and **type** declarations. Subranges and enumerated types are presented.

Introduction to Structured Data: The One-Dimensional Array

The structures, algorithms and applications of one-dimensional arrays are examined in Chapter 12. Included in this presentation is the **for** instruction, buffering, searching, simple sorting, strings, and table driven code.

The Multidimensional Array

The discussion of arrays is expanded to included multidimensional structures in Chapter 13. This chapter also explores using arrays as functions and methods for debugging code with arrays.

The Record Data Structure

Chapter 14 completes this introduction to structured data by examining record structures. Significant topics of this chapter include arrays of records, sorting records, the **with** statement, and variant records.

More Programming Tools

The final chapter is a collection of five major topics: recursion, pointers and dynamic data, stacks, queues, and sets. Recursive procedures and functions are both examined. Recursion is also compared to repetition, as a form of control. The dynamic data facilities of Pascal are used to illustrate linked list structures. Linked lists implementations of stacks and queues are given.

ACKNOWLEDGEMENTS

Many people have played important roles in the development of this text including Boyd & Fraser: Boston, under the direction of President Tom Walker and the incomparable production crew in California, under the direction of Becky Herrington.

I am grateful to the following reviewers for their helpful suggestions: Gary Cutler, Peter J. Gingo, John M. Lloyd, Mike Michaelson, Allen Tucker, Terry A. Ward, John A. Wenzel, and Ron Willard. I also thank my colleagues at the University of Wisconsin-La Crosse who have contributed to this effort in countless ways.

A special acknowledgement is due to all of the people involved with the IBM University Program courses. Participation in the CSF2 and CSF3 courses shaped many of my beliefs regarding an introduction to this discipline. I thank the people who developed or maintain this educational material — Debra Baker, Tony Baxter, John Bentley, Elaine Rich, and Mary Shaw. I also thank those with whom I have taught these courses, especially Albert Baker. In addition, I must express my gratitude to Nancy Morrison who has made a significant contribution to the success of this program and who is responsible for my interest and involvement.

TABLE OF CONTENTS

Preface xix

Chapter 1 - THE PROGRAMMING PROCESS

1.1 What is programming? 1
1.2 The steps of the programming process 1
1.3 Defining the problem 2
1.4 A standard problem definition form 4
1.5 Designing a solution 5
1.6 Coding the algorithm 6
1.7 Maintaining the program 8
1.8 Program documentation 8
1.9 The future of programming 9
1.10 Summary 9
Exercises 10
Programming Projects 12

Chapter 2 - THE CONCEPT OF DATA

2.1 Storage of information 19
2.2 Syntax diagrams 20
2.3 Semantics 25
2.4 The assignment instruction 26
2.5 Integer: a scalar data type 28
2.6 Integer expressions 30
2.7 More scalar types 32
2.8 Strong data typing 36
2.9 Sequences of instructions 40
Swap Algorithm
2.10 The comment 42
2.11 Standard Pascal output 45
2.12 Standard input in Pascal 47
2.13 The shell of a Pascal program 50
PoundsToKilograms Program

2.14 An example of a program 54
ReverseFourChars Algorithm and Program
2.15 The coder's view of programming 55
2.16 Errors that may occur 58
2.16 Summary 59
Exercises 60
Programming Projects 64

Chapter 3 - PROGRAM DESIGN TECHNIQUES

3.1 From definition to code 65
3.2 How to begin the design 66
First Robot Cart Algorithm
3.3 Top down design 69
Second Robot Cart Algorithm
3.4 Alternate forms for top down design 75
3.5 Another example of top down design 77
ChangeMaker Algorithm and Program
3.6 Introduction to debugging 86
3.7 Summary 86
Exercises 88
Programming Projects 89

Chapter 4 - (CONTROL) STRUCTURED PROGRAMMING

4.1 The four fundamental elements of control 95
4.2 Sequences 97
4.3 Control abstraction in algorithms 98
StickFigure Algorithm
4.4 Control abstraction in Pascal: the procedure 102
StickFigure Program
ProcessWeeklyEmployeeInfo Algorithm and Program
4.5 Parameter passage 113
WriteSquaredLine Procedure
GetPercent Procedure
AnnualInterest Procedure
4.6 An example of a design using procedures 117
FenceAndSodCalculator Algorithm and Program

4.7 Summary 120
Exercises 120
Programming Projects 122

Chapter 5 - MORE STRUCTURED PROGRAMMING

5.1 The need for more control 125
5.2 Selection: the if control structure 125
5.3 if-then and if-then-else in Pascal 128
5.4 The compound instruction 130
5.5 An example of a design using selection 131
AnalyzeThreeTemps Algorithm and Program
5.6 Repetition: the while control structure 136
CalculatingPowersOfTwo Algorithm and Program
5.7 Counter and trailer style loops 138
General counting loop Form
General trailer loop Form
5.8 An example of a design using repetition 140
WordCount Algorithm and Program
5.9 Another design 144
Rectangle/ParallelogramDisplay Algorithm and Program
5.10 Summary 154
Exercises 155
Programming Projects 158

Chapter 6 - LOGIC AND PROGRAMMING

6.1 Logical values and expressions 169
6.2 Implication and equivalence 171
6.3 Boolean data types in Pascal 172
6.4 Simplifying Boolean expressions 175
6.5 How boolean expressions are evaluated in Pascal 177
6.6 Applications of Logic 178
6.7 Robust programs 179
IntToChar Procedure
RobustDiv Procedure
6.8 Using logical assertions as program documentation 181
6.9 Debugging and the logical expression 183
6.10 The Assert procedure 183
AssertExample Program

6.11 Summary 184
Exercises 185
Programming Projects 186

Chapter 7 - THE PROCEDURE

7.1 The importance of parameters 189
CubeIt Program
CubeItWithProc Program
7.2 Parameter passage by value revisited. 192
OutputReal Algorithm and Procedure
7.3 Parameter passage by reference revisited 196
SwapChars Procedure
SortThreeChars Program
7.4 Functions 200
Floor Function
SmallerInt Function
PowerOf Function
Zero Function
Add2 Function
7.5 Standard Pascal procedures and functions. 204
IsOdd Function
7.6 Scope, visibility, and binding 205
Food Program Shell
Stooges Program
7.7 Procedure hierarchy charts. 213
7.8 An example of procedures 214
BaseballStats Program
7.9 Improving readability with procedures 219
IsValidTriangle Function
7.10 Debugging and the procedure: stubs and drivers 220
ProcessPlayer Procedure Stub
TestOutputStats Program with Test Driver
7.11 Summary 222
Exercises 223
Programming Projects 227

Chapter 8 - INPUT AND OUTPUT OF TEXT FILES

8.1 Text files 237
8.2 More about Pascal text I/O. 238

8.3 Interactive I/O 245
8.4 File processing 247
WirthMessage Program
DisplayFile Program
CopyWirthToMyCopy Program
8.5 Another file manipulation program 252
CompressBlanks Program
8.6 Summary 254
Exercises 255
Programming Projects 256

Chapter 9 - MORE ABOUT SELECTION

9.1 Review of the **if** instruction 259
9.2 Multi-way selection 261
CarCommission Function
9.3 The relationship of else-if and optional **elses** 264
CarCommission Function (2 versions)
WeeklyWage Procedure (3 versions)
9.4 The **case** instruction 270
GradePoint Function
ClassifyCar Procedure
9.5 The choice between **if** and **case** 273
TransDialLetter Procedure (2 versions)
AlphaCheck Procedure (2 versions)
WaterCondition Procedure (2 versions)
9.6 Debugging selection 277
9.7 Summary 278
Exercises 279
Programming Projects 283

Chapter 10 - MORE ABOUT REPETITION

10.1 Review of the **while** instruction 293
FibonacciNumber Program
10.2 The **repeat** loop 295
TestEvenDivisors Program
10.3 The relationship of **while** and **repeat** 297

10.4 The loop invariant 298
Factorial Algorithm (3 Versions)
Find First Nonblank Input Algorithm
Find Greatest Whole Divisor Algorithm
Find Smallest Input Algorithm
10.5 Designing loops 303
LineCount Algorithm and Program
BallBounce Algorithm and Program
10.6 Debugging loops 316
10.7 Summary 317
Exercises 318
Programming Projects 320

Chapter 11 - INTRODUCTION TO DATA ABSTRACTION

11.1 More abstraction 327
11.2 Constants 327
11.3 Types 330
11.4 Enumerated data types 333
11.5 Subrange data types 338
11.6 An example of use of data abstraction 340
DateDifference Program
11.7 Common pitfalls of Const, Type, enumerated type, and subrange 345
11.8 Summary 350
Exercises 351
Programming Projects 357

Chapter 12 - INTRODUCTION TO STRUCTURED DATA: THE ONE-DIMENSIONAL ARRAY

12.1 Why are structured types necessary? 365
SortTwo Program
SortThree Program
12.2 The concept of an array 367
12.3 Array declaration 369
12.4 Accessing the items of an array 371
12.5 Uses of vectors—random access 375
PoolInventory Program
12.6 A control structure for loop processing: the **for** loop 385
ZeroAllInventory Procedure

12.7 Uses of vectors: sequential access 390
Input Buffering Algorithm
CapCount Function
Linear Search Procedure
Interleave Procedure
ShiftRightBy1 Procedure
12.8 An example of a sorting algorithm: the insertion sort 396
InsertionSort Program
12.9 Treating arrays as aggregates 401
12.10 Strings: packed array of char 403
12.11 Introduction to table driven code 406
Converting Planet Weight Algorithm
Calculating Insurance Premium Algorithm
12.12 Summary 409
Exercises 410
Programming Projects 414

Chapter 13 - THE MULTIDIMENSIONAL ARRAY

13.1 The need for more array dimensions 425
13.2 Vectors of arrays 425
PrecedesAlphabetically Function
SelSort Procedure (selection sort of strings)
13.3 Two-dimensional arrays 432
13.4 Sequential access of matrices 436
DisplayScreen Procedure
ColumnTot Procedure
13.5 Multidimensional arrays 437
13.6 Using arrays as functions 439
EXOR Algorithm
MorseCode Program
InitializeWeight Procedure
Recommended Weight Algorithm
Output Baseball Position Algorithm
13.7 Debugging arrays 445
13.8 Summary 446
Exercises 447
Programming Projects 451

Chapter 14 - THE RECORD DATA STRUCTURE

14.1 Non-homogeneous data 461
14.2 The record 461
SimplifyFraction Program
14.3 Records with structured fields 466
14.4 Aggregate record operations 470
14.5 Arrays of records 471
ReadEmployee Procedure
ReadOneEmployee Procedure
14.6 Sorting records 474
SortEmployees Procedure
14.7 The with statement 475
14.8 Variant records (optional) 478
14.9 Summary 483
Exercises 484
Programming Projects 488

Chapter 15 - MORE PROGRAMMING TOOLS

15.1 Recursion 495
EchoReversed Program
Fib Function
15.2 A comparison of repetition and recursion 501
RecWhile Algorithm
15.3 Pointers and dynamic data 501
15.4 Linked lists 509
WriteListInts Procedure
Inserting After Algorithm
Deleting After Algorithm
15.5 EchoReversed revisited 514
EchoReversed as a Linked List Program
15.6 Stacks and queues 516
EchoReversed as a Stack Program
Stack implemented as a singly linked list Algorithm
Queue implemented as a singly linked list Algorithm
15.7 The Set data type 520
15.8 Summary 524
Exercises 524

Appendix A
Syntactic Description of Pascal in Syntax Diagrams 533

Appendix B
Pascal Reserved Words 549

Appendix C
ASCII Conversion Table 550

Appendix D
Rules of Logic 553

Appendix E
Standard Procedures and Functions 555

Appendix F
The goto Instruction and Labels 563

Solutions for Selected Exercises 565

Glossary 587

Index 601