

TURBO[®] ***PASCAL***

***THEORY AND PRACTICE
OF GOOD PROGRAMMING***

GARY W. MARTIN

Gary W. Martin

SOLANO COMMUNITY COLLEGE

TURBO PASCAL[®] THEORY AND PRACTICE OF GOOD PROGRAMMING

SAUNDERS COLLEGE PUBLISHING

A Harcourt Brace College Publishers

*Fort Worth Philadelphia San Diego New York Orlando Austin San Antonio
Toronto Montreal London Sydney Tokyo*

Copyright © 1992 by Harcourt Brace & Company

All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy, recording, or any information storage and retrieval system, without permission in writing from the publisher.

Requests for permission to make copies of any part of the work should be mailed to Copyrights and Permissions Department, Harcourt Brace & Company, 8th Floor, Orlando, Florida 32887.

Text Typeface: Garamond Light
Compositor: The Clarinda Co.
Acquisitions Editor: Richard Bonacci
Manuscript Editor: Kathy Walker
Production Editor: Michael Biskup
Association: Ruth Rominger
Manager of Art and Design: Carol Bleistine
Art Editor: Karen DeLeo
Text Designer: Cheryl Solheid
Production Service: York Production Services
Director of EDP: Tim Frelick
Production Manager: Joanne Cassetti

Factial Profile and Circuit Board by Garry Gay © the Image Bank.

Printed in the United States of America

Martin: TURBO Pascal: Theory and Practice of Good Programming

0-15-592375-7

Library of Congress Catalog Card Number: 91-72931

56789 039 9876543

TURBO Pascal® is a registered trademark of Borland International, Inc.

THIS BOOK IS PRINTED ON **ACID-FREE, RECYCLED PAPER**





PREFACE


Programming with Pascal on the microcomputer has become almost synonymous with the use of Borland International's *Turbo Pascal*®. It represents a teaching environment that allows the student to master not only correct programming skills but also to gain insights into those fundamental concepts that underscore computer programming. This textbook exposes students to the essentials required of a graduate programmer as well as to the power and flexibility of Turbo Pascal, including its support of object-oriented programming.

Turbo Pascal: Theory and Practice of Good Programming introduces computer technology and the Pascal programming language assuming that the student has no previous experience with computers or computer programming or knowledge of advanced mathematics. Any student familiar with the notation used in a high school algebra course should easily be able to master all of the material contained in this book.

The examples, exercises, and programming problems found in the textbook are not specific to any one particular field of study. A variety of examples and problems relating to computer science, business data processing, mathematics, physical science, health, financial planning, and so on are presented to broaden the student's knowledge of computer application to various fields of study.

The text presents topics and language features found in entry-level college Pascal and computer science courses. It is designed as a one- or two-semester (or one to three quarters) tutorial introducing the entire Pascal language in a step-by-step manner, using theoretical discussions and practical examples. A balance is struck between beginners needs to master the syntax and their need to obtain a sound, disciplined, state-of-the-art, theoretical approach to programming. The material in this textbook meets ACM CS1 recommendations and includes material found in the Educational Testing Service's Advanced Placement Computer Science course as well.

In addition, the text serves as a reference to the powerful features and extensions of the *Turbo Pascal* implementation of the Pascal language. The book is intentionally *not* a *generic Pascal text*. All of the material presented applies strictly to the *MS/PC-DOS 5.0/5.5/6.0 versions of Borland International's Turbo Pascal*. However, adherence to standard Pascal is made whenever possible (to encourage the portability of the programs constructed in the text), except when certain important concepts in computer science are presented that standard Pascal does not support (for example, MS/PC-DOS-specific operations, strings, random access files, or object-oriented programming).



The text also introduces the student to computer science in a clear, accessible, engaging, and to-the-point manner. The concepts of structured programming are covered in detail with an emphasis on the software engineering concepts of top-down design, decomposition, modularity, procedural and data abstraction, code reusability, and separate compilation. All program examples are constructed using structured programming theory. These concepts are covered thoroughly to build the student's problem-solving and programming abilities and to establish good programming habits.

A problem-oriented approach to programming is used, emphasized by a strong, hands-on, class-tested, instructional format. A strategy of learn-while-doing is stressed throughout the text. Pascal has been long accepted as an excellent instructional language that encourages the development of good programming skills and a consistent programming style. Because of these features the textbook could be used as a self-teaching, stand-alone source for learning to program in Turbo Pascal.

FEATURES

Each chapter begins with a set of *Chapter Objectives* explicitly stating the learning objectives of that chapter. These are followed by a general overview of what is to be studied in the chapter and why. Because terminology is so important to any technical field, each new term is defined when introduced, with examples and figures to illustrate the term.

Each chapter provides a detailed presentation of a unique collection of Pascal language syntax constructs and programming methodologies, as well as numerous class-tested *Program Examples*. These *Program Examples* demonstrate the proper use of Pascal syntax and explore many of Turbo Pascal's important nonstandard extensions to the Pascal language. Programming lines critical to the implementation of each of these example programs are highlighted to draw the student's attention to them. Each chapter also has many *Exercises* to test student comprehension of the material presented as they work through it.

Each chapter also includes a series of *Tips*, *Traps*, *Tricks*, and *Rules*. The *Tips* draw the student's attention to nonobvious *programming techniques* used by professional programmers to get the most out of Pascal. The *Traps* alert the student to *potential problem areas* to be avoided by the novice Pascal programmer. The *Tricks* point out significant *unique syntax features* of Turbo Pascal that can add useful processing capabilities to a program and are not available in standard Pascal. Finally the *Rules* point out *general policies to follow* when constructing Pascal programs.

A considerable effort has been made to address the use of consistent and proper program documentation. Both *internal* and *external* documentation is stressed in all of the example programs and subprograms found in the text. Banners of asterisks set off headings and highlight the interface sections of all programs and subprograms, and comments extensively document important sections of code found in all of the program examples. These features are designed to improve the clarity of the programs presented in the text. Furthermore, *pseudocode* is used to outline the

algorithms used in program solutions, and is used extensively as a design tool in the development of programs.

Many beginning programmers may have trouble relating a program's output to the computer instructions that created it. This is especially true in the early chapters, when program output concerns formatting data types. Therefore, demonstration *executive screens* for almost all of the interactive programs presented, are shown when a program is implemented so the student can see exactly what actions will occur as a result of a program's execution. These screens show the results of programs in action.

The construction of long programs is conceptually different from the construction of short ones. Hence, all but the first few chapters contain lengthy *Chapter Sample Programs* to illustrate syntax constructs and programming concepts introduced in the chapter at hand. Each of these *Chapter Sample Programs* presents a large, detailed, modularized program, designed and implemented using the structured programming methodology. Each program is fully developed using a five-step program development process consisting of *Problem Definition*, *Program Design*, *Implementation*, *Verification*, and *Documentation*. These *Chapter Sample Programs* demonstrate to the student the program planning and development process as it might occur in a "real-world" setting.

Each chapter ends with a *Chapter Summary* listing new terms introduced and summarizing important concepts presented in the chapter. The key terms introduced in the chapter are boldface in the summary to draw attention to them. Also, all chapters include exercise sets. *Chapter Review Exercises* allow students to work through and self-test their understanding of the material presented in the chapter. In addition, a *Chapter Test* prepares students for taking in-class tests for credit. Finally, *Programming Problems* challenge students newly acquired programming skills.

A unique feature of this textbook is the inclusion of *Turbo Topics* to help students exploit many of the rich and powerful features of Turbo Pascal, without these topics detracting from the presentation of the Pascal language or conflicting with the software engineering concepts presented in the main body of the text. These topics can be introduced as an instructor sees fit or can be referenced by the interested student in addition to required class material.

The *Turbo Topics* give detailed introductions to the 5.0/5.5 and 6.0 versions of the Turbo Pascal integrated development environment, a complete command key reference for the integrated development environment, a Turbo Pascal reserved and standard identifier reference, a Turbo Pascal error message listing, a complete compiler directive listing, a listing of Turbo Pascal syntax charts, a discussion of extended key codes, a detailed discussion of files and user-defined units to create software libraries, a discussion of text colors and windows, and much more.

In addition, a demonstration software *toolbox* is included so students can create professional looking programs using basic programming skills. This toolbox introduces the student to the use of software libraries, procedural and data abstraction, and the idea of separate compilation. The Turbo Pascal code contained in the toolbox demonstrates the concrete use of many Turbo Pascal language features found in the other *Turbo Topics*. The source code of the toolbox is provided on the

Student Resource Disk so that students can observe how a toolbox is created and how each routine in the toolbox is implemented. The toolbox code can also be modified, as much as students desire, to suit their own needs.

At the end of the text three Appendices provide important reference information, such as an *Extended ASCII Character Set* table, a *MS-DOS Command Reference*, and *Answers to Selected Chapter Review Exercises*. In addition, a *Bibliography* and complete *Glossary* are included. The Glossary lists definitions of the most important terms found in the text as well as terms commonly used in computing.

ORDER OF TOPICS

Topics are all presented by building difficult concepts from simpler ones. Subprograms are introduced early, even before control structures, so that students can develop truly modularized programs from the beginning. The early introduction of subprograms helps the student to understand and use the theory of software modules abstractly, using the methodology of structured programming. Files are introduced immediately after conditional statements and loops so that students can quickly become familiar with constructing realistic practical programs for processing collections of data.

An entire chapter on recursion is presented so this important topic can be discussed and explored in detail. Unlike most Turbo Pascal textbooks, an entire chapter on object-oriented programming is presented. This chapter contains both a general description of the object-oriented programming paradigm and a detailed description of its implementation in Turbo Pascal. This chapter is extremely important because the influence of object-oriented programming will surely guide program development in the 1990s.

TO THE INSTRUCTOR

The textbook is designed to allow teaching directly from the textbook without having to write out a pseudotext on the board during each lecture. The textbook allows you to express yourself and not concentrate on hours of out-of-class preparation that amounts to nothing more than reinventing the wheel. This easy-to-use design permits you to get involved with your students and to concentrate on sharing your expert knowledge of problem solving and software development with your students.

TO THE STUDENT

To make best use of the material in this textbook, you should always read the material before it is discussed in class. Carefully study the *Examples* found in each chapter and

attempt to do the *Exercises*. This will bring out any weakness in your comprehension of the material, focus your thoughts on the material presented, and make the time you spend in class much more valuable and probably make the material covered a lot easier to digest.

It is suggested that you work through the *Chapter Review Exercises* at the end of each chapter after you have read the chapter. Resist the temptation to just read each exercise and then look up the answer to the exercise in the appendix of the text. Try to find your own solution to each exercise and only then compare it to the supplied answer to the exercise. Make notes of any confusing points, bring them to class, and ask your instructor for help.

The educationally-oriented toolbox, whose source code is on the *Student Resource Disk*, is designed to get your computer up and running in minimum time, and yet allow your programs to obtain a professional look and behavior. The toolbox can freely and legally be used in your own personal and academic projects. It is offered to further enhance your use of the textbook. However, the actual source code itself is the copyrighted property of the author. The toolbox may not be used to construct other toolboxes, or software products to be sold for profit.

SUPPLEMENTS

A *Student Resource Disk*, accompanying the text contains the debugged source code for all of the *Example Programs* and *Chapter Sample Programs* found in the text. In addition, the *Student Resource Disk* contains the source code of the toolbox unit which students can use to construct realistic and professional-quality programs using only introductory programming skills. This toolbox unit is designed to demonstrate how students can use or build their own program libraries using the concepts of procedural abstraction and separate compilation.

An *Instructor's Manual*, designed to help the instructor coordinate classroom lectures with the textbook, will be available for all instructors who adopt the textbook. The manual contains general teaching objectives, a list of learning objectives, a chapter outline, and teaching suggestions, tips, and points of emphasis for teaching the material in each chapter. The *Instructor's Manual* also contains answers to the *Exercises*, a list of key terms introduced in the chapter, answers to the *Chapter Review Exercises* and *Test Questions*, as well as solutions to selected *Programming Problems* found in each chapter. In addition, the *Instructor's Manual* contains additional exam questions to aid in the production of in-class tests.

The *Instructor's Manual* also contains a set of *Transparency Masters* of important figures from the text that can be used during lectures to create effective presentations. An *Instructor Resource Disk* will also accompany the *Instructor's Manual* and will contain the source code for the solutions of the *Exercises* found in each chapter and the source code for the solutions of the selected *Programming Problems* listed in the *Instructor's Manual*.

ACKNOWLEDGMENTS

I would like to thank all of the students enrolled in my beginning and advanced Pascal programming classes over the past three years for their assistance in the development and class testing of this textbook. Their suggestions for improving the text and their patience and help in detecting and correcting many of the errors found in the early versions of the manuscript were invaluable. In addition, I would like to thank all of the reviewers of the various versions of the manuscript for their valuable comments and suggestions for improving the text. Many of these comments and suggestions significantly influenced the final version of the textbook. These reviewers include: Margaret S. Anderson, University of Georgia; Larry Booth, Highline Community College; Rex M. Dixon Jr., Humboldt State University; Peter Falley, Farleigh Dickinson University; Roberts Fritz, American River College; Neal Gold, Lindsey Wilson College; Jim Henry, Northern Illinois University; Michael Hitt, McMurray University; Michael P. Johnson, Oregon State University; Judy Anne Kane, Nashville State Technical Institute; Randall Lechlitner, NFC America, Inc.; Robert E. Norton, San Diego Mesa College; Eleanor S. Quinlan, The Ohio State University; Deborah J. Ritchie, Moorpark College; Jesse H. Ruder Jr., Austin Community College; Lowell Stultz, Kalamazoo Valley Community College; Dale Shaffer, Lander College; and Bruce Sisko, Belleville Area College.

I would especially like to thank all of the members of the Harcourt Brace Jovanovich book team for their hard work in the preparation and production of this textbook. I would like to thank Michael Biskup, Lynne Bush, Karen DeLeo, Ruth Rominger, Tom Thompson, and David Watt. In particular I would like to thank Kathy Walker for her detailed editing of the manuscript and Cheryl Staples for her first-rate design of the text, cover, and art program. Finally, I give my sincerest thanks to my acquisitions editor, Richard Bonacci, who originally commissioned this project and without whose guidance, patience, understanding, support, encouragement, and expertise this book would not have been possible.

Gary W. Martin

PHOTO CREDITS

3, IBM. 8, © UPI/Bettmann Archive. 23, IBM. 26, IBM. 27, Intel. 31, Hewlett Packard. 31, Quantum. 34, NEC. 36, (top) IBM. 36, (bottom) Epson. 38, Hewlett Packard. 39, Hewlett Packard. 40, IBM. 41, IBM. 42, Hayes. 44, The Bettmann Archive. 87, © Tony Stone Worldwide: Mike Surowiak. 129, © Bob Daemmrich Photography. 157, Romeo Meloche. 203, © Bob Daemmrich Photography. 204, © Topham/The Image Works. 253, FPG International. 301, M. Timothy O'Keefe. 339, Photofest. 385, © Tony Stone Worldwide: John Garrett. 477, UPI/Bettmann Archive. 535, © William Taufic 1981 all rights reserved. 563, HBJ. 603, TSW © Bavaria Janicek. 639, Photofest. 701, © 1966 by Jean-Louie Swiners. 795, IBM.

PERMISSIONS

"I'm my own Grandpa," words and music by Dwight Latham and Moe Jaffe © 1947 Colgems-EMI Music Inc. Renewed 1975 Colgems-EMI Music Inc. All rights reserved. Used by permission.

"Planet's Mean Distance from Sun and Equatorial Diameter," adapted from THE RANDOM DICTIONARY OF THE ENGLISH LANGUAGE 2/e. Copyright © 1987 by Random House Inc. Reprinted by permission of Random House Inc.

REGISTERED TRADEMARKS

Epson is a registered trademark of Epson America, Inc. Hayes is a registered trademark of Hayes Microcomputer Products, Inc. Hewlett-Packard, HP, and LaserJet are trademarks of Hewlett Packard Corporation. IBM is a registered trademark of International Business Machines Corporation. Intel, 8088, 8086, 80286, 80386, 80486 are registered trademarks of Intel Corporation. MS-DOS is a registered trademark of Microsoft Corporation. NEC is a registered trademark of NEC Technologies, Inc. PC-DOS is a registered trademark of International Business Machines Corporation. Turbo Pascal is a registered trademark of Borland International, Inc.

CONTENTS

PREFACE *iii*

PART I

BEGINNING PASCAL Control Structures and Data Types

1

CHAPTER 1

ESSENTIAL COMPUTER CONCEPTS

2

CHAPTER OBJECTIVES 2

CHAPTER OVERVIEW 3

Computer Capabilities	4
Data Processing	6
Computer Hardware	8
Binary Representations of Data	11
Classification of Computer Systems	13
Computer Software	13
Five Components of a Computer System	21
Historical Development of the IBM PC line	21
IBM Compatible Microcomputer Hardware	22
Historical Development of Pascal	43
Background on Turbo Pascal	44

Chapter Summary	44
Chapter Review Exercises	46
Chapter Test	47

CHAPTER 2

CONSTRUCTING TURBO PASCAL PROGRAMS

48

CHAPTER OBJECTIVES 48

CHAPTER OVERVIEW 49

The Turbo Pascal Character Set	51
Turbo Pascal Tokens	51
Identifiers	51
Data Types	53
Basic Program Structure	57
Program Output: The Write and Writeln Statements	67
Formats for Printing Data types	68
Program Style	76

Chapter Sample Program	77
Chapter Summary	79
Chapter Review Exercises	81
Chapter Test	82
Programming Problems	84

CHAPTER 3

INPUT, CALCULATIONS, AND SUBPROGRAMS

86

CHAPTER OBJECTIVES 86

CHAPTER OVERVIEW 87

Program Input: The Read and Readln Statements	89
Calculations in Pascal	94
Assignment Statements	101
Standard Functions	102
Standard Procedures	109
Standard Units	110
Miscellaneous Turbo Functions and Procedures	113

Chapter Sample Program	116
Chapter Summary	119
Chapter Review Exercises	120
Chapter Test	121
Programming Problems	123

CHAPTER 4

STRUCTURED PROGRAMMING 128

CHAPTER OBJECTIVES 128

CHAPTER OVERVIEW 129

Introduction to Structured Programming	130
Structured Design	132
Modular Programming	134
Program Coding	139
GoTo Considered Harmful	145
The Programming Process	146
Program Maintenance	150

Chapter Sample Programs 150

Chapter Summary 153

Chapter Review Exercises 154

Chapter Test 154

CHAPTER 5

USER-DEFINED SUBPROGRAMS 156

CHAPTER OBJECTIVES 156

CHAPTER OVERVIEW 157

Introduction to User-Defined Functions	158
Introduction to User-Defined Procedures	168
A Pascal Program Containing User-Defined Subprograms	171
Physical versus Logical Order of Procedures	172
Value versus Variable Parameters	176
The Scope of Identifiers: Global versus Local Identifiers	184
The Nesting of Procedures and Functions	185
Constructing a Program Driver	189

Chapter Sample Program 190

Chapter Summary 195

Chapter Review Exercises 196

Chapter Test 197

Programming Problems 198

CHAPTER 6

SELECTION STATEMENTS 202

CHAPTER OBJECTIVES 202

CHAPTER OVERVIEW 203

Boolean Expressions	204
One-way Selection	210
Two-way Selection	216
Multi-way Selection	220

Chapter Sample Program 229

Chapter Summary 238

Chapter Review Exercises 239

Chapter Test 241

Programming Problems 244

CHAPTER 7

LOOPING STATEMENTS 252

CHAPTER OBJECTIVES 252

CHAPTER OVERVIEW 253

Program Loops	254
The While Statement	255
The For Statement	264
The Repeat Statement	274
Some Loop Comparisons	278

Chapter Sample Program 279

Chapter Summary 289

Chapter Review Exercises 289

Chapter Test 292

Programming Problems 293

CHAPTER 8

TEXT FILE PROCESSING 300

CHAPTER OBJECTIVES 300

CHAPTER OVERVIEW 301

Files	302
The Type Text	304
Text File Organization	310
The File Window (Look Ahead)	312
Writing to a Text File	314
Numeric Data within Text Files	319
Details of File Use	321

Chapter Sample Program 321

Chapter Summary 330

Chapter Review Exercises 331

Chapter Test 331

Programming Problems 332

CHAPTER 9

CREATING USER-DEFINED DATA TYPES AND SETS

CHAPTER OBJECTIVES	338
CHAPTER OVERVIEW	339

Data Types	340
User-Defined Types	341
User-Defined Enumerated Ordinal Types	342
Subrange Types	352
Sets	354
The Set Type	355
Operations On Sets	356

Chapter Sample Program	368
Chapter Summary	376
Chapter Review Exercises	377
Chapter Test	379
Programming Problems	381

PART II

ADVANCED PASCAL Data Structures

383

CHAPTER 10

STRUCTURED TYPES: ARRAYS

384

CHAPTER OBJECTIVES	384
CHAPTER OVERVIEW	385

Data Structures	386
Introduction to Arrays	387
One-Dimensional Arrays (List)	388
Multidimensional Arrays	425

Chapter Sample Program	447
Chapter Summary	461
Chapter Review Exercises	462
Chapter Test	466
Programming Problems	468

CHAPTER 11

STRUCTURED TYPES: TURBO PASCAL STRINGS

476

CHAPTER OBJECTIVES	476
CHAPTER OVERVIEW	477

String Type Declarations	478
String Assignment	483
Null Strings	486
String Comparisons	486
String Expressions	488
Turbo Pascal String Procedures and Functions	504

Chapter Sample Program	513
Chapter Summary	524
Chapter Review Exercises	525
Chapter Test	527
Programming Problems	530

CHAPTER 12

STRUCTURED TYPES: BINARY FILES

534

CHAPTER OBJECTIVES	534
CHAPTER OVERVIEW	535

Working With Files	536
Categories of Files	537
Binary Files	538
Random (Direct) Access Files	545

Chapter Sample Program	548
Chapter Summary	557
Chapter Review Exercises	558
Chapter Test	559
Programming Problems	560

CHAPTER 13

STRUCTURED TYPES: RECORDS

562

CHAPTER OBJECTIVES	562
CHAPTER OVERVIEW	563

The Record Concept	564
Declaring Record Types	565
Methods of Access to Records	567
Scope of Record Identifiers	574
Record Variants	575
Records Versus Arrays	579
Arrays of Records	581
Record Input/Output	582

Chapter Sample Program	584
Chapter Summary	597
Chapter Review Exercises	597
Chapter Test	599
Programming Problems	600

CHAPTER 14

RECURSION 602

<i>CHAPTER OBJECTIVES</i>	602
<i>CHAPTER OVERVIEW</i>	603

The Concept of Recursion	604
Implementing Recursion	605
Recursive versus Iterative Subprograms	615
Advantages and Disadvantages of Recursion	616
Indirect Recursion	618

Chapter Sample Program	623
Chapter Summary	628
Chapter Review Exercises	629
Chapter Test	630
Programming Problems	632

CHAPTER 15

SORTING AND SEARCHING 638

<i>CHAPTER OBJECTIVES</i>	638
<i>CHAPTER OVERVIEW</i>	639

Sorting Techniques	640
Efficiency of the Sorting Algorithms	660
Choosing the Appropriate Sorting Technique	662
Internal Searching Techniques	664
Efficiency of Searching Techniques	672

Chapter Sample Program	673
Chapter Summary	691
Chapter Review Exercises	691
Chapter Test	695
Programming Problems	697

CHAPTER 16

DYNAMIC DATA STRUCTURES 700

<i>CHAPTER OBJECTIVES</i>	700
<i>CHAPTER OVERVIEW</i>	701

Static versus Dynamic Data Structures	702
The Pointer Concept	703
Pascal Pointers and Dynamic Variables	703
Linked Lists	713
Doubly Linked Lists	747
Trees	751
Binary Trees	751
Turbo Pascal Pointer Functions and Procedures	769

Chapter Sample Program	771
Chapter Summary	783
Chapter Review Exercises	784
Chapter Test	786
Programming Problems	789

CHAPTER 17

OBJECT-ORIENTED PROGRAMMING (OOP) 794

<i>CHAPTER OBJECTIVES</i>	794
<i>CHAPTER OVERVIEW</i>	795

The Object-Oriented Programming Revolution	796
Object-Oriented Programming: An Overview	797
Object-Oriented Programming in Turbo Pascal	801
A Stack Object	804
Encapsulation	804
Inheritance	813
Polymorphism	817
Object Type Compatibility	824
Dynamically Allocating Objects	826

Chapter Sample Program	836
Chapter Summary	850
Chapter Review Exercises	852
Chapter Test	853
Programming Problems	854

PART III

TURBO TOPICS AND APPENDICES 857

TOPIC A

USING THE TURBO PASCAL INTEGRATED DEVELOPMENT ENVIRONMENT 858

Choosing a Compiler Format	859	Include Files	926
Starting Turbo Pascal	860	User-Defined Units	928
Sample Program Development	883		
Printing From Within the Turbo Pascal IDE	888		
TOPIC B			
TURBO PASCAL COMMAND KEYS	890	TEXT COLORS	934
Editing Keys	890		
Hot Keys	892		
TOPIC C			
TURBO PASCAL RESERVED AND STANDARD IDENTIFIERS	896	TEXT WINDOWS	938
TOPIC D			
TURBO PASCAL ERROR MESSAGES	898	TOOLBOX	940
Run-time Error Messages	898	Borland International Scholar Program	947
TOPIC E			
TURBO PASCAL COMPILER DIRECTIVES	900	APPENDICES	948
TOPIC F			
TURBO PASCAL EXTENDED DATA TYPES	902	APPENDIX 1	
		THE EXTENDED ASCII CHARACTER SET	948
TOPIC G			
TURBO PASCAL EXTENDED KEY CODES	904	APPENDIX 2	
		MS-DOS COMMAND REFERENCE	950
TOPIC H			
TURBO PASCAL SYNTAX CHARTS	906	Command Formats and Syntax	950
		APPENDIX 3	
TOPIC I			
DEVELOPING PROGRAM LIBRARIES	926	ANSWERS TO SELECTED CHAPTER REVIEW EXERCISES	952
		BIBLIOGRAPHY	968
		GLOSSARY	969
		INDEX	979
		INDEX TO TURBO TRICKS	989
		INDEX TO ILLUSTRATIVE PROGRAM EXAMPLES	991

PART

BEGINNING PASCAL CONTROL STRUCTURES AND DATA TYPES

CHAPTER 1

ESSENTIAL COMPUTER CONCEPTS

CHAPTER OBJECTIVES

After completing this chapter you should be able to

- Identify the types of operations computers can perform.
- Understand what is meant by the terms *data*, *information*, and *data processing* and how the basic data processing cycle works.
- Understand the six-part computer hardware model as it is used in the construction of modern computers.
- Understand how the binary number system is used to represent both data and programs in a computer system.
- Understand the definition and use of the terms *bit*, *byte*, and *word*.
- Understand what is meant by the computer classification terms *microcomputer*, *minicomputer*, and *mainframe computer*.
- Discern the two major categories of computer software (applications and systems) and understand the use of the terms *algorithm* and *program*.
- Identify the three categories of computer languages (machine, assembly, and high-level) and how they function in the programming of a computer.
- Comprehend how translators (assemblers, compilers, and interpreters) process source code into object code.
- Identify the five components of a computer system (hardware, programs, data, procedures, and personnel).
- Identify the specific types of IBM-compatible microcomputer hardware and software available in the marketplace.
- Understand the origins and basic features of both standard Pascal and Turbo Pascal.