# Contents

Preface xv

Chapter Dependency Chart xviii

**PART ONE**
**Problem-Solving Techniques 1**

**1 Review of Java Fundamentals 3**

1.1 Program Structure **4**
- Packages 4
- Classes 6
- Data Fields 7
- Methods 8
- How to Access Members of an Object 10

1.2 Language Basics **11**
- Comments 11
- Identifiers and Keywords 11
- Variables 12
- Primitive Data Types 12
- References 13
- Literal Constants 14
- Named Constants 14
- Assignments and Expressions 15
- Arrays 18

1.3 Selection Statements **22**
- The *`if`* Statement 22
- The *`switch`* Statement 23

1.4 Iteration Statements **24**
- The *`while`* Statement 24
- The *`for`* Statement 25
- The *do* Statement 28

1.5 Useful Java Classes **28**
- The *`Object`* Class 29
- String Classes 29

**1.6** Java Exceptions **34**
Catching Exceptions 34
Throwing Exceptions 41
**1.7** Text Input and Output **43**
Input 43
Output 45
**1.8** File Input and Output **48**
Text Files 49
Object Serialization 57
*Summary 60* *Cautions 63*

**2 Principles of Programming and Software Engineering** **65**

**2.1** Problem Solving and Software Engineering **66**
What Is Problem Solving? 66
The Life Cycle of Software 67
What Is a Good Solution? 76
**2.2** Achieving an Object-Oriented Design **78**
Abstraction and Information Hiding 78
Object-Oriented Design 81
Functional Decomposition 83
General Design Guidelines 84
Modeling Object-Oriented Designs Using UML 85
Advantages of an Object-Oriented Approach 88
**2.3** A Summary of Key Issues in Programming **89**
Modularity 90
Modifiability 92
Ease of Use 93
Fail-Safe Programming 94
Style 100
Debugging 104
*Summary 107* *Cautions 108* *Self-Test Exercises 108*
*Exercises 108* *Programming Problems 111*

**3 Recursion: The Mirrors** **113**

**3.1** Recursive Solutions **114**
A Recursive Valued Method: The Factorial of *n* *117*
A Recursive `void` Method: Writing a String Backward 124
**3.2** Counting Things **135**
Multiplying Rabbits (The Fibonacci Sequence) 135
Organizing a Parade 137
Mr. Spock's Dilemma (Choosing *k* out of *n* Things) 140
**3.3** Searching an Array **142**
Finding the Largest Item in an Array 143
Binary Search 144
Finding the $k^{th}$ Smallest Item in an Array 148

3.4 Organizing Data **152**
The Towers of Hanoi 152
3.5 Recursion and Efficiency **156**
*Summary 162* *Cautions 163* *Self-Test Exercises 163*
*Exercises 164* *Programming Problems 170*

**4 Data Abstraction: The Walls 171**
4.1 Abstract Data Types **172**
4.2 Specifying ADTs **177**
The ADT List 178
The ADT Sorted List 183
Designing an ADT 184
Axioms (Optional) 189
4.3 Implementing ADTs **192**
Java Classes Revisited 193
Java Interfaces 201
Java Packages 204
An Array-Based Implementation of the ADT List 206
*Summary 213* *Cautions 214* *Self-Test Exercises 215*
*Exercises 215* *Programming Problems 218*

**5 Linked Lists 221**
5.1 Preliminaries **222**
Object References 222
Resizeable Arrays 228
Reference-Based Linked Lists 229
5.2 Programming with Linked Lists **234**
Displaying the Contents of a Linked List 234
Deleting a Specified Node from a Linked List 236
Inserting a Node into a Specified Position of a Linked List 238
A Reference-Based Implementation of the ADT List 244
Comparing Array-Based and Reference-Based Implementations 248
Passing a Linked List to a Method 251
Processing Linked Lists Recursively 252
5.3 Variations of the Linked List **257**
Tail References 257
Circular Linked Lists 258
Dummy Head Nodes 260
Doubly Linked Lists 260
5.4 Application: Maintaining an Inventory **264**

**5.5** The Java Collections Framework **270**
Generics 271
Iterators 272
The Java Collections Framework *List* Interface 275
*Summary 278* *Cautions 280* *Self-Test Exercises 281*
*Exercises 283* *Programming Problems 286*

**PART TWO**
**Problem Solving with Abstract Data Types** **291**

**6** **Recursion as a Problem-Solving Technique** **293**
**6.1** Backtracking **294**
The Eight Queens Problem 294
**6.2** Defining Languages **299**
The Basics of Grammars 300
Two Simple Languages 301
Algebraic Expressions 304
**6.3** The Relationship Between Recursion and Mathematical Induction **314**
The Correctness of the Recursive Factorial Method 314
The Cost of Towers of Hanoi 315
*Summary 317* *Cautions 317* *Self-Test Exercises 318*
*Exercises 318* *Programming Problems 321*

**7** **Stacks** **327**
**7.1** The Abstract Data Type Stack **328**
Developing an ADT During the Design of a Solution 328
**7.2** Simple Applications of the ADT Stack **334**
Checking for Balanced Braces 334
Recognizing Strings in a Language 338
**7.3** Implementations of the ADT Stack **339**
An Array-Based Implementation of the ADT Stack 341
A Reference-Based Implementation of the ADT Stack 343
An Implementation That Uses the ADT List 345
Comparing Implementations 347
The Java Collections Framework Class *Stack* 347
**7.4** Application: Algebraic Expressions **349**
Evaluating Postfix Expressions 349
Converting Infix Expressions to Equivalent Postfix Expressions 351
**7.5** Application: A Search Problem **354**
A Nonrecursive Solution That Uses a Stack 356
A Recursive Solution 364

**7.6** The Relationship Between Stacks and Recursion **367**

*Summary 369* *Cautions 369* *Self-Test Exercises 370*
*Exercises 371* *Programming Problems 375*

**8 Queues 381**

**8.1** The Abstract Data Type Queue **382**

**8.2** Simple Applications of the ADT Queue **384**
Reading a String of Characters 384
Recognizing Palindromes 385

**8.3** Implementations of the ADT Queue **386**
A Reference-Based Implementation 388
An Array-Based Implementation 391
An Implementation That Uses the ADT List 397
The Java Collections Framework Interface `Queue` 398
Comparing Implementations 400

**8.4** A Summary of Position-Oriented ADTs **401**

**8.5** Application: Simulation **402**

*Summary 412* *Cautions 413* *Self-Test Exercises 413*
*Exercises 414* *Programming Problems 417*

**9 Advanced Java Topics 421**

**9.1** Inheritance Revisited **422**
Java Access Modifiers 428
Is-a and Has-a Relationships 429

**9.2** Dynamic Binding and Abstract Classes **431**
Abstract Classes 435
Java Interfaces Revisited 439

**9.3** The ADTs List and Sorted List Revisited **440**
Implementations of the ADT Sorted List That Use the ADT List 441

**9.4** Java Generics **445**
Generic Classes 445
Generic Wildcards 447
Generic Classes and Inheritance 448
Generic Implementation of the Class List 451
Generic Methods 453

**9.5** Iterators **454**

*Summary 455* *Cautions 456* *Self-Test Exercises 456*
*Exercises 457* *Programming Problems 460*

**10 Algorithm Efficiency and Sorting** **463**

**10.1** Measuring the Efficiency of Algorithms **464**
The Execution Time of Algorithms 465
Algorithm Growth Rates 467
Order-of-Magnitude Analysis and Big O Notation 467
Keeping Your Perspective 473
The Efficiency of Searching Algorithms 475

**10.2** Sorting Algorithms and Their Efficiency **476**
Selection Sort 477
Bubble Sort 481
Insertion Sort 483
Mergesort 485
Quicksort 491
Radix Sort 503
A Comparison of Sorting Algorithms 505
The Java Collections Framework Sort Algorithm 506

*Summary 510* *Cautions 510* *Self-Test Exercises 511*
*Exercises 512* *Programming Problems 515*

**11 Trees** **517**

**11.1** Terminology **518**

**11.2** The ADT Binary Tree **526**
Basic Operations of the ADT Binary Tree 526
General Operations of the ADT Binary Tree 527
Traversals of a Binary Tree 530
Possible Representations of a Binary Tree 533
A Reference-Based Implementation of the ADT Binary Tree 537
Tree Traversals Using an Iterator 542

**11.3** The ADT Binary Search Tree **551**
Algorithms for the Operations of the ADT Binary Search Tree 556
A Reference-Based Implementation
of the ADT Binary Search Tree 572
The Efficiency of Binary Search Tree Operations 576
Treesort 580
Saving a Binary Search Tree in a File 581
The JCF Binary Search Algorithm 584

**11.4** General Trees **586**

*Summary 588* *Cautions 588* *Self-Test Exercises 588*
*Exercises 590* *Programming Problems 597*

**12 Tables and Priority Queues** **601**

**12.1** The ADT Table **602**
Selecting an Implementation 609
A Sorted Array-Based Implementation of the ADT Table 616
A Binary Search Tree Implementation of the ADT Table 619

**12.2** The ADT Priority Queue: A Variation of the ADT Table **621**
Heaps 625
A Heap Implementation of the ADT Priority Queue 634
Heapsort 636

**12.3** Tables and Priority Queues in the JCF **639**
The JCF *Map* Interface 639
The JCF *Set* Interface 643
The JCF *PriorityQueue* Class 647

*Summary 649* *Cautions 650* *Self-Test Exercises 650*
*Exercises 651* *Programming Problems 654*

**13 Advanced Implementations of Tables 657**

**13.1** Balanced Search Trees **658**
2-3 Trees 659
2-3-4 Trees 679
Red-Black Trees 686
AVL Trees 689

**13.2** Hashing **695**
Hash Functions 699
Resolving Collisions 701
The Efficiency of Hashing 710
What Constitutes a Good Hash Function? 713
Table Traversal: An Inefficient Operation under Hashing 715
The JCF *Hashtable* and *TreeMap* Classes 716
The *Hashtable* Class 716
The *TreeMap* Class 719

**13.3** Data with Multiple Organizations **722**

*Summary 727* *Cautions 728* *Self-Test Exercises 729*
*Exercises 729* *Programming Problems 732*

**14 Graphs 735**

**14.1** Terminology **736**

**14.2** Graphs as ADTs **739**
Implementing Graphs 740
Implementing a Graph Class Using the JCF 743

**14.3** Graph Traversals **746**
Depth-First Search 748
Breadth-First Search 749
Implementing a BFS Iterator Class Using the JCF 751

**14.4** Applications of Graphs **754**
Topological Sorting 754
Spanning Trees 757
Minimum Spanning Trees 761
Shortest Paths 765
Circuits 769
Some Difficult Problems 772
*Summary 773* *Cautions 774* *Self-Test Exercises 774*
*Exercises 775* *Programming Problems 778*

**15 External Methods 781**

**15.1** A Look at External Storage **782**
**15.2** Sorting Data in an External File **785**
**15.3** External Tables **793**
Indexing an External File 795
External Hashing 799
B-Trees 803
Traversals 813
Multiple Indexing 815
*Summary 816* *Cautions 817* *Self-Test Exercises 817*
*Exercises 817* *Programming Problems 820*

**APPENDICES**

**A A Comparison of Java to C++ 821**

**B Unicode Character Codes (ASCII Subset) 825**

**C Java Resources 826**
Java Web Sites 826
Using J2SE 826
Integrated Development Environments (IDEs) 827

**D Mathematical Induction 828**
Example 1 828
Example 2 829
Example 3 830
Example 4 831
Example 5 831
*Self-Test Exercises 832* *Exercises 832*

Glossary 835

Answers to Self-Test Exercises 855

Index 873