Contents

1 OBJECT-ORIENTED PROGRAMMING USING C++ 1

1.1 Abstract Data Types 1

1.2 Encapsulation 1

1.3 Inheritance 6

1.4 Pointers 9

1.4.1 Pointers and Arrays 12

1.4.2 Pointers and Copy Constructors 14

1.4.3 Pointers and Destructors 16

1.4.4 Pointers and Reference Variables 17

1.4.5 Pointers to Functions 20

1.5 Polymorphism 21

1.6 C++ and Object-Oriented Programming 23

1.7 The Standard Template Library 24

1.7.1 Containers 24

1.7.2 Iterators 25

1.7.3 Algorithms 25

1.7.4 Function Objects 26

1.8 Vectors in the Standard Template Library 28

1.9 Data Structures and Object-Oriented Programming 35

1.10 Case Study: Random Access File 35

1.11 Exercises 46

1.12 Programming Assignments 48

Bibliography 50

2 COMPLEXITY ANALYSIS 51

2.1 Computational and Asymptotic Complexity 51
2.2 Big-O Notation 52
2.3 Properties of Big-O Notation 54
2.4 Ω and Θ Notations 56
2.5 Possible Problems 57
2.6 Examples of Complexities 57
2.7 Finding Asymptotic Complexity: Examples 59
2.8 The Best, Average, and Worst Cases 61
2.9 Amortized Complexity 64
2.10 NP-Completeness 68
2.11 **Exercises** 71
Bibliography 74

3 LINKED LISTS 75

3.1 Singly Linked Lists 75
3.1.1 Insertion 81
3.1.2 Deletion 83
3.1.3 Search 89
3.2 Doubly Linked Lists 90
3.3 Circular Lists 94
3.4 Skip Lists 96
3.5 Self-Organizing Lists 101
3.6 Sparse Tables 106
3.7 Lists in the Standard Template Library 109
3.8 Concluding Remarks 113
3.9 Case Study: A Library 114
3.10 **Exercises** 125
3.11 **Programming Assignments** 127
Bibliography 130

4 Stacks and Queues 131

4.1 Stacks 131
4.2 Queues 139
4.3 Priority Queues 148
4.4 Stacks in the Standard Template Library 149
4.5 Queues in the Standard Template Library 149
4.6 Priority Queues in the Standard Template Library 151
4.7 Deques in the Standard Template Library 153
4.8 Case Study: Exiting a Maze 158
4.9 Exercises 165
4.10 Programming Assignments 166
Bibliography 168

5 Recursion 169

5.1 Recursive Definitions 169
5.2 Function Calls and Recursion Implementation 172
5.3 Anatomy of a Recursive Call 174
5.4 Tail Recursion 177
5.5 Nontail Recursion 178
5.6 Indirect Recursion 184
5.7 Nested Recursion 186
5.8 Excessive Recursion 186
5.9 Backtracking 190
5.10 Concluding Remarks 197
5.11 Case Study: A Recursive Descent Interpreter 198
5.12 Exercises 207
5.13 Programming Assignments 210
Bibliography 213

6 Binary Trees 214

6.1 Trees, Binary Trees, and Binary Search Trees 214
6.2 Implementing Binary Trees 219
6.3 Searching a Binary Search Tree 222

6.4 Tree Traversal 224
6.4.1 Breadth-First Traversal 225
6.4.2 Depth-First Traversal 226
6.4.3 Stackless Depth-First Traversal 233
6.5 Insertion 240
6.6 Deletion 243
6.6.1 Deletion by Merging 244
6.6.2 Deletion by Copying 246
6.7 Balancing a Tree 250
6.7.1 The DSW Algorithm 253
6.7.2 AVL Trees 256
6.8 Self-Adjusting Trees 261
6.8.1 Self-Restructuring Trees 262
6.8.2 Splaying 263
6.9 Heaps 268
6.9.1 Heaps as Priority Queues 270
6.9.2 Organizing Arrays as Heaps 271
6.10 Treaps 276
6.11 *K*-d Trees 280
6.12 Polish Notation and Expression Trees 286
6.12.1 Operations on Expression Trees 287
6.13 Case Study: Computing Word Frequencies 290
6.14 Exercises 298
6.15 Programming Assignments 302
Bibliography 306

7 MULTIWAY TREES 309
7.1 The Family of B-Trees 310
7.1.1 B-Trees 311
7.1.2 B*-Trees 321
7.1.3 B+-Trees 323
7.1.4 Prefix B+-Trees 326
7.1.5 *K*-d B-trees 327
7.1.6 Bit-Trees 334
7.1.7 R-Trees 336
7.1.8 2–4 Trees 337
7.1.9 Sets and Multisets in the Standard Template Library 353
7.1.10 Maps and Multimaps in the Standard Template Library 359

7.2 Tries 364
7.3 Concluding Remarks 373
7.4 Case Study: Spell Checker 373
7.5 Exercises 384
7.6 Programming Assignments 385
Bibliography 389

8 GRAPHS 391
8.1 Graph Representation 393
8.2 Graph Traversals 395
8.3 Shortest Paths 398
8.3.1 All-to-All Shortest Path Problem 405
8.4 Cycle Detection 408
8.4.1 Union-Find Problem 409
8.5 Spanning Trees 411
8.6 Connectivity 415
8.6.1 Connectivity in Undirected Graphs 415
8.6.2 Connectivity in Directed Graphs 418
8.7 Topological Sort 421
8.8 Networks 423
8.8.1 Maximum Flows 423
8.8.2 Maximum Flows of Minimum Cost 433
8.9 Matching 438
8.9.1 Stable Matching Problem 442
8.9.2 Assignment Problem 445
8.9.3 Matching in Nonbipartite Graphs 447
8.10 Eulerian and Hamiltonian Graphs 449
8.10.1 Eulerian Graphs 449
8.10.2 Hamiltonian Graphs 453
8.11 Graph Coloring 459
8.12 NP-Complete Problems in Graph Theory 462
8.12.1 The Clique Problem 462
8.12.2 The 3-Colorability Problem 463
8.12.3 The Vertex Cover Problem 465
8.12.4 The Hamiltonian Cycle Problem 466
8.13 Case Study: Distinct Representatives 467

8.14 Exercises 480
8.15 Programming Assignments 486
Bibliography 487

9 SORTING 491
9.1 Elementary Sorting Algorithms 492
9.1.1 Insertion Sort 492
9.1.2 Selection Sort 495
9.1.3 Bubble Sort 497
9.1.4 Comb Sort 500
9.2 Decision Trees 501
9.3 Efficient Sorting Algorithms 505
9.3.1 Shell Sort 505
9.3.2 Heap Sort 508
9.3.3 Quicksort 512
9.3.4 Mergesort 518
9.3.5 Radix Sort 521
9.3.6 Counting Sort 527
9.4 Sorting in the Standard Template Library 528
9.5 Concluding Remarks 532
9.6 Case Study: Adding Polynomials 534
9.7 Exercises 542
9.8 Programming Assignments 543
Bibliography 545

10 HASHING 548
10.1 Hash Functions 549
10.1.1 Division 549
10.1.2 Folding 549
10.1.3 Mid-Square Function 550
10.1.4 Extraction 550
10.1.5 Radix Transformation 551
10.1.6 Universal Hash Functions 551
10.2 Collision Resolution 551
10.2.1 Open Addressing 552
10.2.2 Chaining 558
10.2.3 Bucket Addressing 560
10.3 Deletion 561

10.4 Perfect Hash Functions 562
10.4.1 Cichelli's Method 563
10.4.2 The FHCD Algorithm 566
10.5 Rehashing 568
10.5.1 The Cuckoo Hashing 568
10.6 Hash Functions for Extendible Files 571
10.6.1 Extendible Hashing 571
10.6.2 Linear Hashing 574
10.7 Case Study: Hashing with Buckets 576
10.8 Exercises 586
10.9 Programming Assignments 587
Bibliography 588

11 DATA COMPRESSION 590
11.1 Conditions for Data Compression 590
11.2 Huffman Coding 592
11.2.1 Adaptive Huffman Coding 601
11.3 Run-Length Encoding 606
11.4 Ziv-Lempel Code 607
11.5 Case Study: Huffman Method with Run-Length Encoding 610
11.6 Exercises 622
11.7 Programming Assignments 622
Bibliography 624

12 MEMORY MANAGEMENT 625
12.1 The Sequential-Fit Methods 626
12.2 The Nonsequential-Fit Methods 627
12.2.1 Buddy Systems 629
12.3 Garbage Collection 636
12.3.1 Mark-and-Sweep 637
12.3.2 Copying Methods 644
12.3.3 Incremental Garbage Collection 646
12.3.4 Generational Garbage Collection 653
12.4 Concluding Remarks 657
12.5 Case Study: An In-Place Garbage Collector 658

12.6 **Exercises 667**

12.7 **Programming Assignments 668**

Bibliography 671

13 String Matching 674

13.1 Exact String Matching 674
- 13.1.1 Straightforward Algorithms 674
- 13.1.2 The Knuth-Morris-Pratt Algorithm 677
- 13.1.3 The Boyer-Moore Algorithm 685
- 13.1.4 Multiple Searches 695
- 13.1.5 Bit-Oriented Approach 697
- 13.1.6 Matching Sets of Words 700
- 13.1.7 Regular Expression Matching 707
- 13.1.8 Suffix Tries and Trees 711
- 13.1.9 Suffix Arrays 717

13.2 Approximate String Matching 719
- 13.2.1 String Similarity 720
- 13.2.2 String Matching with k Errors 726

13.3 Case Study: Longest Common Substring 729

13.4 **Exercises 738**

13.5 **Programming Assignments 740**

Bibliography 741

Appendixes

A Computing Big-O 743
- A.1 Harmonic Series 743
- A.2 Approximation of the Function $\lg(n!)$ 743
- A.3 Big-O for Average Case of Quicksort 745
- A.4 Average Path Length in a Random Binary Tree 747
- A.5 The Number of Nodes in an AVL Tree 748

B Algorithms in the Standard Template Library 749
- B.1 Standard Algorithms 749

C NP-Completeness 758
- C.1 Cook's Theorem 758

Index 771