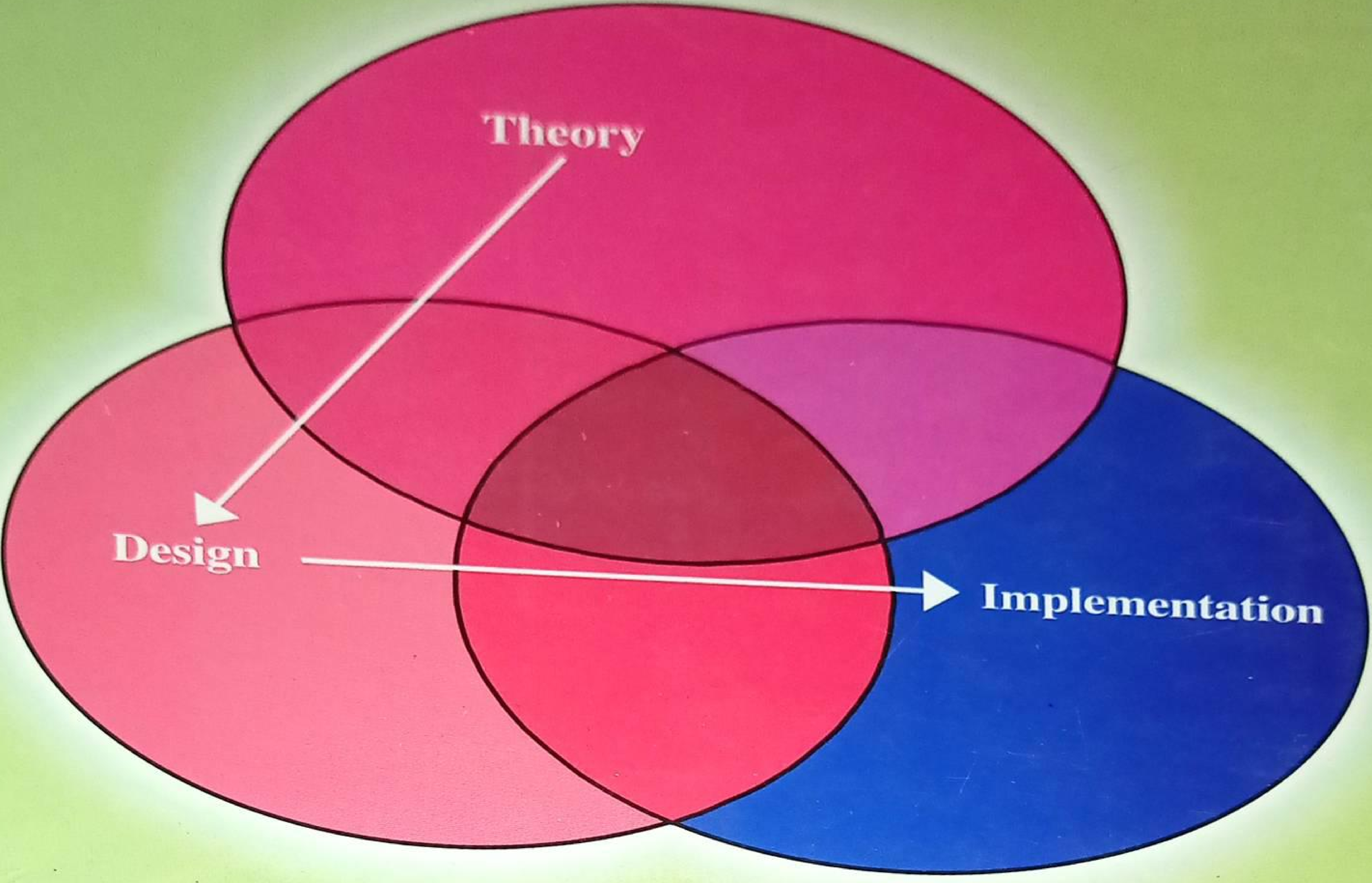
A Simplified Text-cum-Workbook on

Data Structures & Algorithms

Theory, Design and Implementation Using C



R. S. Salaria

Table of Contents

Preface
Acknowledgments

Chapter 1 : Overview of Data Structures 1-18

1.1 INTRODUCTION 2
1.2 BASIC TERMINOLOGY OF DATA ORGANIZATION 2
1.3 CONCEPT OF A DATA TYPE 4
1.3.1 Primitive Data Type 4
1.3.2 Abstract Data Type 5
1.3.3 Polymorphic Data Type 5
1.4 DATA STRUCTURE DEFINED 6
1.5 DESCRIPTION OF VARIOUS DATA STRUCTURES 6
1.5.1 Arrays 7
1.5.1.1 Linear Arrays 7
1.5.1.2 Two-dimensional Arrays 8
1.5.2 Linked Lists 9
1.5.2.1 Linear Linked List 9
1.5.2.2 Doubly Linked List 10
1.5.2.3 Circular Linked List 11
1.5.3 Stacks 11
1.5.4 Queues 12
1.5.5 Trees 13
1.5.6 Heaps 13
1.5.7 Graphs 14
1.5.8 Hash Tables 15
1.6 COMMON OPERATIONS ON DATA STRUCTURES 16
1.7 A QUICK REVIEW 16
1.8 REVIEW EXERCISES 17

Chapter 2 : Introduction to Program Design 19-60
2.1 INTRODUCTION 20
2.2 PROGRAM DEVELOPMENT LIFE CYCLE 20
2.2.1 Defining the Problem 21
2.2.2 Designing the Program 21
2.2.2.1 Program Design Tools 22
2.2.3 Coding the Program 23
2.2.4 Testing and Debugging the Program 23
2.2.5 Documenting the Program 24
2.2.1 Implementing and Maintaining the Program 24
2.3 INTRODUCTION TO ALGORITHMS 25
2.3.1 A Typical Example 25
2.3.2 Algorithm Description 26
2.3.3 Sub-Algorithms 28
2.4 STRUCTURED PROGRAMMING CONSTRUCTS 29
2.4.1 Sequential Programming Construct 30
2.4.2 Selection Programming Construct 30
2.4.2.1 Single Alternative 31
2.4.2.2 Double Alternatives 31
2.4.2.3 Multiple Alternative 32
2.4.3 Iterative Programming Construct 33
2.4.3.1 For Construct 33
2.4.3.2 While Construct 34
2.4.3.3 Do – while Construct 35
2.5 ALGORITHM COMPLEXITY 35
2.5.1 Space Complexity 36
2.5.2 Time Complexity 38
2.5.3 Time-space Trade-off 38
2.5.4 Expressing Space and Time Complexity 39
2.6 Big Oh Notation 39
2.6.1 Categories of Algorithms 41
2.6.2 Limitations of Big Oh Notation 42
2.7 CONCEPT OF RECURSION 42
2.7.1 Recursion Defined 42
2.7.2 Depth of Recursion 43
2.7.3 Examples of Some Recursive Functions 43
2.7.3.1 Factorial Function 43
2.7.3.1.1 Recursive Function and Run Time Stack 47
2.7.3.2 Positive Exponential Power Function 50
2.7.3.3 Fibonacci Numbers 52
2.7.3.4 Counting Problem 54
2.7.3.5 Minmax Problem 55

2.7.4 Recursion, Iteration or . . .? 57
2.8 A QUICK REVIEW 58
2.9 REVIEW EXERCISES 59
2.10 PROGRAMMING EXERCISES 60

Chapter 3 : Arrays & Matrices 61-92

3.1 INTRODUCTION 62
3.2 LINEAR ARRAYS 62
3.2.1 Representing Linear Arrays in Memory 63
3.2.2 Operations on Linear Arrays 64
3.2.2.1 Traversal Operation 64
3.2.2.1.1 Analysis of Traversal Operation 65
3.2.2.2 Search Operation 66
3.2.2.2.1 Linear Search 66
3.2.2.2.1.1 Analysis of Linear Search 67
3.2.2.2.2 Binary Search 67
3.2.2.2.2.1 Analysis of Binary Search 69
3.2.2.3 Insert Operation 69
3.2.2.3.1 Inserting at Given Position 70
3.2.2.3.2 Inserting in a Sorted Array 70
3.2.2.3.3 Analysis of Insert Operation 71
3.2.2.4 Delete Operation 71
3.2.2.4.1 Deleting from Given Position 71
3.2.2.4.2 Deleting from a Sorted Array 72
3.2.2.4.3 Analysis of Delete Operation 72
3.2.2.5 Sort Operation 72
3.2.2.5.1 Bubble Sort Method 73
3.2.2.5.1.1 Analysis of Bubble Sort Method 74
3.2.2.6 Merge Operation 76
3.2.2.6.1 Analysis of Merge Operation 77
3.2.3 Applications of Linear Arrays 78
3.2.4 Limitations of Linear Arrays 78
3.3 TWO-DIMENSIONAL ARRAYS 78
3.3.1 Representing Two-dimensional Array in Memory 79
3.4 MATRICES 81
3.4.1 Special Matrices 82
3.4.1.1 Representing Special Matrices in Memory 83
3.4.2 Sparse Matrices 87
3.4.2.1 Representing Sparse Matrices 88
3.4.2.1.1 Array Representation 88
3.4.2.1.2 Linked Representation 89

3.5 A QUICK REVIEW 89
3.6 REVIEW EXERCISES 90
3.7 PROGRAMMING EXERCISES 90

Chapter 4 : Linked Lists 93-138

4.1 INTRODUCTION 94
4.2 IMPLEMENTING LISTS 94
4.2.1 Array Implementation of Lists 94
4.2.2 Linked Implementation of Lists 95
4.3 LINKED LIST DEFINED 96
4.3.1 Linear Linked List 96
4.3.1.1 Representation of Linear Linked List 97
4.3.1.2 Operations on Linear Linked Lists 97
4.3.1.2.1 Creating an Empty List 97
4.3.1.2.2 Traversing a List 98
4.3.1.2.2.1 In-Order Traversal 98
4.3.1.2.2.2 Reverse-Order Traversal 98
4.3.1.2.3 Searching an Element 99
4.3.1.2.3.1 List is Unsorted 100
4.3.1.2.3.2 List is Sorted 100
4.3.1.2.3.3 Auxiliary Search Procedure 101
4.3.1.2.4 Inserting an Element 101
4.3.1.2.4.1 Inserting at the Beginning of the List 102
4.3.1.2.4.2 Inserting at the End of the List 102
4.3.1.2.4.3 Inserting after the Given Element of the List 103
4.3.1.2.5 Deleting an Element 104
4.3.1.2.4.1 Deleting from the Beginning of the List 104
4.3.1.2.4.2 Deleting from the End of the List 105
4.3.1.2.4.3 Deleting after the Given Element of the List 106
4.3.1.2.6 Reversing a List 106
4.3.1.2.7 Deleting Entire List 107
4.3.2 Doubly Linked List 108
4.3.2.1 Representation of Doubly Linked List 109
4.3.2.2 Operations on Doubly Linked Lists 109
4.3.2.2.1 Creating an Empty List 109
4.3.2.2.2 Traversing a List 109
4.3.2.2.2.1 In-Order Traversal 110
4.3.2.2.2.2 Reverse-Order Traversal 110
4.3.2.2.3 Searching an Element 110
4.3.2.2.4 Inserting an Element 111
4.3.2.2.4.1 Inserting after a Given Element 111

4.3.2.2.4.2 Inserting before a Given Element 112
4.3.2.2.5 Deleting an Element 112
4.3.2.2.5.1 Deleting after a Given Element 113
4.3.2.2.5.2 Deleting before a Given Element 113
4.3.2.2.6 Deleting Entire List 114
4.3.3 More Types of Lists 114
4.3.3.1 Circular Linked List 115
4.3.3.2 Header Linked Lists 115
4.4 APPLICATIONS OF LINKED LISTS 117
4.4.1 Polynomial Manipulation 117
4.4.1.1 Addition of Polynomials 118
4.4.1.2 Multiplication of Polynomials 122
4.4.2 Representing Sparse Matrices 127
4.5 A QUICK REVIEW 136
4.6 REVIEW EXERCISES 136
4.7 PROGRAMMING EXERCISES 137

Chapter 5 : Stacks 139-170

5.1 INTRODUCTION 140
5.2 OPERATIONS ON STACKS 141
5.3 REPRESENTATION OF A STACK IN MEMORY 142
5.3.1 Representing a Stack using an Array 142
5.3.1.1 Creating an Empty Stack 143
5.3.1.2 Testing Stack for Underflow 143
5.3.1.3 Testing Stack for Overflow 144
5.3.1.4 Push Operation 145
5.3.1.5 Pop Operation 145
5.3.1.6 Accessing Top Element 146
5.3.2 Representing a Stack using a Linked List 147
5.3.2.1 Creating an Empty Stack 148
5.3.2.2 Testing Stack for Underflow 148
5.3.2.3 Testing Stack for Overflow 149
5.3.2.4 Push Operation 149
5.3.2.5 Pop Operation 149
5.3.2.6 Accessing Top Element 150
5.3.2.7 Dispose a Stack 150
5.4 MULTIPLE STACKS 151
5.4.1 Representing Two Stacks 151
5.4.2 Representing More than two Stacks 151
5.5 APPLICATIONS OF STACKS 152
5.5.1 Parenthesis Checker 152

5.5.2 Mathematical Notation Translation 15
5.5.2.1 Infix Notation 15
5.5.2.2 Polish (Prefix) Notation 15
5.5.2.3 Reverse Polish (Postfix) Notation 15
5.5.2.4 Evaluating Mathematical Expressions 15
5.5.2.5 Infix-to-Postfix Procedure 15
5.5.2.6 Evaluating Expression in Postfix Notation 16
5.5.3 Quicksort Algorithm 16
5.6 A QUICK REVIEW 16
5.7 REVIEW EXERCISES 16
5.8 PROGRAMMING EXERCISES 16

Chapter 6 : Queues 171-19

6.1 INTRODUCTION 17
6.2 QUEUE DEFINED 16
6.3 OPERATIONS ON QUEUES 17
6.4 REPRESENTATION OF QUEUES IN MEMORY 17
6.4.1 Representing a Queue using an Array 17
6.4.1.1 Implementation of Operations on a Linear Queue 17
6.4.1.1.1 Creating an Empty Linear Queue 17
6.4.1.1.2 Testing a Linear Queue for Underflow 17
6.4.1.1.3 Testing a Linear Queue for Overflow 17
6.4.1.1.4 Enqueue Operation on Linear Queue 18
6.4.1.1.5 Dequeue Operation on a Linear Queue 18
6.4.1.1.6 Accessing Front Element 18
6.4.1.2 Limitations of a Linear Queue 18
6.4.1.3 Implementation of Operations on a Circular Queue 18
6.4.1.3.1 Testing a Circular Queue for Overflow 18
6.4.1.3.2 Enqueue Operation on a Circular Queue 18
6.4.1.3.3 Dequeue Operation on a Circular Queue 18
6.4.2 Representing a Queue using a Linked List 18
6.4.2.1 Implementation of Operations on a Linear Queue 18
6.4.2.1.1 Creating an Empty Queue 18
6.4.2.1.2 Testing Queue for Underflow 18
6.4.2.1.3 Testing Queue for Overflow 18
6.4.2.1.4 Enqueue Operation 1
6.4.2.1.5 Dequeue Operation 18
6.4.2.1.6 Accessing Front Element 1
6.4.2.1.7 Disposing a Queue 18
6.4 MULTIPLE QUEUES 18
6.6 DEQUE 19

6.7 PRIORITY QUEUE 191
6.7.1 Representing a Priority Queue in Memory 191
6.7.1.1 Linear Linked List Representation 191
6.7.1.2 Multiple Queue Representation 192
6.7.1.3 Heap Representation of a Priority Queue 192
6.8 APPLICATIONS OF QUEUES 193
6.9 A QUICK REVIEW 193
6.10 REVIEW EXERCISES 194
6.11 PROGRAMMING EXERCISES 194

Chapter 7 : Trees **195-240**

8.1 INTRODUCTION 196
7.2 TREE DEFINED 199
7.3 TREE TERMINOLOGY 199
7.4 BINARY TREES 200
7.4.1 Properties of Binary Trees 201
7.5 TOURNAMENT TREES 203
7.6 BINARY SEARCH TREES 204
7.6.1 Representation of a Binary and Binary Search Tree 204
7.6.1.1 Array Representation 205
7.6.1.2 Linked Representation 207
7.6.2 Common Operations on Binary and Binary Search Trees 208
7.6.2.1 Creating an Empty Binary (Search) Tree 209
7.6.2.2 Traversing a Binary (Search) Tree 209
7.6.2.2.1 Pre-Order Traversal of a Binary (Search) Tree 211
7.6.2.2.2 In-Order Traversal of a Binary (Search) Tree 212
7.6.2.2.3 Post-Order Traversal of a Binary (Search) Tree 214
7.6.2.2.4 Level-Order Traversal of a Binary (Search) Tree 215
7.6.2.3 Determining Height a Binary (Search) Tree 219
7.6.2.4 Determining Number of Nodes/Elements 220
7.6.2.5 Determining Number of Internal/Non-leaf Nodes 220
7.6.2.6 Determining Number of External/Leaf Nodes 221
7.6.2.7 Determining Mirror Image 22
7.6.2.8 Removing Binary (Search) Tree from Memory 22
7.6.2.9 Inserting a New Element 22
7.6.2.10 Searching an Element 22
7.6.2.11 Finding Smallest Node 22
7.6.2.12 Finding Largest Node 22
7.6.2.13 Deleting a Node 2
7.7 AVL TREES 2
7.7.1 Representation of an AVL Tree 2

7.7.2 Height of an AVL Tree 227
7.7.3 Operations on an AVL Tree 228
7.7.3.1 Insertion of a Node 228
7.7.3.1.1 Rotations 229
7.7.3.2 Deletion of a Node 231
7.8 THREADED BINARY TREES 231
7.8.1 Representing a Threaded Binary Tree in Memory 232
7.8.2 Operations on a Threaded Binary Tree 232
7.8.2.1 In-order Traversal of Threaded Binary Tree 233
7.8.2.2 Insertion in a Threaded Binary Search Tree 234
7.8.2.3 Finding Largest Element a Threaded Binary Search Tree 235
7.8.2.4 Deleting An Element from a Threaded Binary Search Tree 236
7.8.2.5 Deleting a Threaded Binary Search Tree 237
7.9 A QUICK REVIEW 237
7.10 REVIEW EXERCISES 238
7.11 PROGRAMMING EXERCISES 239

Chapter 8 : Heaps **241-258**

8.1 INTRODUCTION 242
8.2 REPRESENTING A HEAP IN MEMORY 243
8.3 OPERATIONS ON HEAPS 243
8.3.1 Deleting an Element From Heap 243
8.3.2 Inserting an Element into Heap 247
8.4 APPLICATIONS OF HEAPS 250
8.4.1 Priority Queues 250
8.5 BUILDING A HEAP 253
8.6 A QUICK REVIEW 256
8.7 REVIEW EXERCISES 256
8.9 PROGRAMMING EXERCISES 257

Chapter 9 : Graphs **259-310**

9.1 INTRODUCTION 260
9.2 GRAPH TERMINOLOGY 261
9.3 REPRESENTATION OF GRAPHS 263
9.3.1 Adjacency Matrix Representation 264
9.3.2 Adjacency List Representation 265
9.3.2.1 Implementation of Adjacency List in C 267
9.4 OPERATIONS ON GRAPHS 26
9.4.1 Creating an Empty Graph 26
9.4.2 Entering Graph Information 26

9.4.3 Outputting a Graph 269
9.4.4 Deleting a Graph 269
9.4.5 Traversal 270
9.4.5.1 Breadth-First Search 270
9.4.5.2 Depth-First Search 274
9.5 APPLICATIONS OF GRAPHS 279
9.5.1 Topological Sort 279
9.5.2 Minimum Spanning Tree 285
9.5.3 Finding Shortest Paths 291
9.5.3.1 Shortest Path for Given Source and Destination 292
9.5.3.2 Shortest Paths Among All-Pair of Vertices 298
9.5.4 Transitive Closure of a Directed Graph 304
9.6 ARTICULATION POINTS, BRIDGES, AND BICONNECTED COMPONENTS 306
9.7 STRONGLY CONNECTED COMPONENTS 307
9.8 EULERIAN TOUR 308
9.9 HAMILTONIAN TOUR 308
9.10 A QUICK REVIEW 308
9.11 REVIEW EXERCISES 308
9.12 PROGRAMMING EXERCISES 310

Chapter 10 : Hash Tables & Hashing 311-344

10.1 INTRODUCTION 312
10.2 DIRECT ADDRESS TABLES 313
10.2.1 Initializing a Direct-address Table 314
10.2.2 Searching an Element in Direct-address Table 314
10.2.3 Inserting a New Element in Direct-address Table 315
10.2.4 Deleting an Element from a Direct-address Table 315
10.3 HASH TABLE 316
10.4 WHAT IS A HASH FUNCTION? 317
10.4.1 Different Hash Functions 318
10.4.1.1 Division Method 318
10.4.1.2 Multiplication Method 319
10.4.1.3 Midsquare Method 320
10.4.1.4 Folding Method 320
10.5 RESOLVING COLLISIONS 321
10.5.1 Collision Resolution by Separating Chaining 321
10.5.1.1 Initializing a Chained-hash-table 322
10.5.1.2 Searching an Element in Chained-hash-table 323
10.5.1.3 Inserting a New Element in Chained-hash-table 323
10.5.1.4 Deleting an Element from a Chained-hash-table 324
10.5.2 Collision Resolution by Open Addressing 328

10.5.2.1 Linear Probing 32
10.5.2.2 Quadratic Probing 33
10.5.2.3 Double Hashing 33
10.6 REHASHING 34
10.7 A QUICK REVIEW 34
10.8 REVIEW EXERCISES 34

Chapter 11 : Sorting Algorithms 345-37

11.1 INTRODUCTION 34
11.2 SELECTION SORT 34
11.2.1 Analysis of Selection Sort 35
11.3 INSERTION SORT 35
11.3.1 Analysis of Insertion Sort 35
11.4 BUCKET SORT 35
11.4.1 Analysis of Bucket Sort 36
11.5 MERGE SORT 36
11.5.1 Analysis of Merge Sort 36
11.6 QUICK SORT 36
11.6.1 Analysis of Quick Sort 37
11.7 HEAP SORT 37
11.7.1 Analysis of Heap Sort 37
11.8 A QUICK REVIEW 374
11.9 REVIEW EXERCISES 37
1110 PROGRAMMING EXERCISES 37

INDEX 377-38