

Data Structures And Algorithms Made Easy

-To All My Readers

**By
Narasimha Karumanchi**

 **Concepts**  **Problems**  **Interview Questions**

Copyright© 2017 by *CareerMonk.com*

All rights reserved.

Designed by *Narasimha Karumanchi*

Copyright© 2017 CareerMonk Publications. All rights reserved.

All rights reserved. No part of this book may be reproduced in any form or by any electronic or mechanical means, including information storage and retrieval systems, without written permission from the publisher or author.

Acknowledgements

Mother and Father, it is impossible to thank you adequately for everything you have done, from loving me unconditionally to raising me in a stable household, where your persistent efforts and traditional values taught your children to celebrate and embrace life. I could not have asked for better parents or role-models. You showed me that anything is possible with faith, hard work and determination.

This book would not have been possible without the help of many people. I would like to express my gratitude to all of the people who provided support, talked things over, read, wrote, offered comments, allowed me to quote their remarks and assisted in the editing, proofreading and design. In particular, I would like to thank the following individuals:

- *Mohan Mullapudi*, IIT Bombay, Architect, dataRPM Pvt. Ltd.
- *Navin Kumar Jaiswal*, Senior Consultant, Juniper Networks Inc.
- *A.Vamshi Krishna*, IIT Kanpur, Mentor Graphics Inc.
- *Cathy Reed*, BA, MA, Copy Editor

–*Narasimha Karumanchi*
M-Tech, IIT Bombay
Founder, *CareerMonk.com*

Preface

Dear Reader,

Please hold on! I know many people typically do not read the Preface of a book. But I strongly recommend that you read this particular Preface.

It is not the main objective of this book to present you with the theorems and proofs on *data structures* and *algorithms*. I have followed a pattern of improving the problem solutions with different complexities (for each problem, you will find multiple solutions with different, and reduced, complexities). Basically, it's an enumeration of possible solutions. With this approach, even if you get a new question, it will show you a way to *think* about the possible solutions. You will find this book useful for interview preparation, competitive exams preparation, and campus interview preparations.

As a *job seeker*, if you read the complete book, I am sure you will be able to challenge the interviewers. If you read it as an *instructor*, it will help you to deliver lectures with an approach that is easy to follow, and as a result your students will appreciate the fact that they have opted for Computer Science / Information Technology as their degree.

This book is also useful for *Engineering degree students* and *Masters degree students* during their academic preparations. In all the chapters you will see that there is more emphasis on problems and their analysis rather than on theory. In each chapter, you will first read about the basic required theory, which is then followed by a section on problem sets. In total, there are approximately 700 algorithmic problems, all with solutions.

If you read the book as a *student* preparing for competitive exams for Computer Science / Information Technology, the content covers *all the required topics* in full detail. While writing this book, my main focus was to help students who are preparing for these exams.

In all the chapters you will see more emphasis on problems and analysis rather than on theory. In each chapter, you will first see the basic required theory followed by various problems.

For many problems, *multiple* solutions are provided with different levels of complexity. We start with the *brute force* solution and slowly move toward the *best solution* possible for that problem. For each problem, we endeavor to understand how much time the algorithm takes and how much memory the algorithm uses.

It is recommended that the reader does at least one *complete* reading of this book to gain a full understanding of all the topics that are covered. Then, in subsequent readings you can skip directly to any chapter to refer to a specific topic. Even though many readings have been done for the purpose of correcting errors, there could still be some minor typos in the book. If any are found, they will be updated at www.CareerMonk.com. You can monitor this site for any corrections and also for new problems and solutions. Also, please provide your valuable suggestions at: Info@CareerMonk.com.

I wish you all the best and I am confident that you will find this book useful.

—Narasimha Karumanchi
M-Tech, IIT Bombay
Founder, CareerMonk.com

Other Books by Narasimha Karumanchi








-  IT Interview Questions
-  Data Structures and Algorithms for GATE
-  Data Structures and Algorithms Made Easy in Java
-  Coding Interview Questions
-  Peeling Design Patterns
-  Elements of Computer Networking
-  Data Structures and Algorithmic Thinking with Python

Table of Contents

1. Introduction-----	13
1.1 Variables -----	13
1.2 Data Types -----	13
1.3 Data Structures -----	14
1.4 Abstract Data Types (ADTs) -----	14
1.5 What is an Algorithm? -----	14
1.6 Why the Analysis of Algorithms? -----	15
1.7 Goal of the Analysis of Algorithms -----	15
1.8 What is Running Time Analysis? -----	15
1.9 How to Compare Algorithms -----	15
1.10 What is Rate of Growth? -----	15
1.11 Commonly Used Rates of Growth -----	16
1.12 Types of Analysis -----	17
1.13 Asymptotic Notation -----	17
1.14 Big-O Notation [Upper Bounding Function] -----	17
1.15 Omega- Ω Notation [Lower Bounding Function] -----	19
1.16 Theta- Θ Notation [Order Function] -----	19
1.17 Important Notes -----	20
1.18 Why is it called Asymptotic Analysis? -----	20
1.19 Guidelines for Asymptotic Analysis -----	20
1.20 Simplyfying properties of asymptotic notations -----	22
1.21 Commonly used Logarithms and Summations -----	22
1.22 Master Theorem for Divide and Conquer Recurrences -----	22
1.23 Divide and Conquer Master Theorem: Problems & Solutions -----	23
1.24 Master Theorem for Subtract and Conquer Recurrences -----	23
1.25 Variant of Subtraction and Conquer Master Theorem -----	24
1.26 Method of Guessing and Confirming -----	24
1.27 Amortized Analysis -----	25
1.28 Algorithms Analysis: Problems & Solutions -----	25
2. Recursion and Backtracking -----	36
2.1 Introduction -----	36
2.2 What is Recursion? -----	36
2.3 Why Recursion? -----	36
2.4 Format of a Recursive Function -----	36
2.5 Recursion and Memory (Visualization) -----	37
2.6 Recursion versus Iteration -----	38
2.7 Notes on Recursion -----	38
2.8 Example Algorithms of Recursion -----	38
2.9 Recursion: Problems & Solutions -----	38

2.10 What is Backtracking?-----	39
2.11 Example Algorithms of Backtracking -----	40
2.12 Backtracking: Problems & Solutions -----	40
3. Linked Lists-----	42
3.1 What is a Linked List?-----	42
3.2 Linked Lists ADT-----	42
3.3 Why Linked Lists? -----	42
3.4 Arrays Overview -----	43
3.5 Comparison of Linked Lists with Arrays & Dynamic Arrays -----	44
3.6 Singly Linked Lists-----	44
3.7 Doubly Linked Lists -----	49
3.8 Circular Linked Lists -----	54
3.9 A Memory-efficient Doubly Linked List -----	59
3.10 Unrolled Linked Lists -----	60
3.11 Skip Lists -----	65
3.12 Linked Lists: Problems & Solutions -----	67
4. Stacks -----	87
4.1 What is a Stack? -----	87
4.2 How Stacks are used -----	87
4.3 Stack ADT-----	88
4.4 Applications-----	88
4.5 Implementation-----	88
4.6 Comparison of Implementations-----	93
4.7 Stacks: Problems & Solutions-----	93
5. Queues -----	109
5.1 What is a Queue? -----	109
5.2 How are Queues Used? -----	109
5.3 Queue ADT-----	109
5.4 Exceptions -----	110
5.5 Applications-----	110
5.6 Implementation-----	110
5.7 Queues: Problems & Solutions -----	115
6. Trees -----	120
6.1 What is a Tree?-----	120
6.2 Glossary -----	120
6.3 Binary Trees -----	121
6.4 Types of Binary Trees -----	122
6.5 Properties of Binary Trees -----	122
6.6 Binary Tree Traversals -----	124
6.7 Generic Trees (N-ary Trees)-----	143
6.8 Threaded Binary Tree Traversals (Stack or Queue-less Traversals)-----	149
6.9 Expression Trees -----	154

6.10 XOR Trees	156
6.11 Binary Search Trees (BSTs)	157
6.12 Balanced Binary Search Trees	171
6.13 AVL (Adelson-Velskii and Landis) Trees	172
6.14 Other Variations on Trees	186
7. Priority Queues and Heaps	190
7.1 What is a Priority Queue?	190
7.2 Priority Queue ADT	190
7.3 Priority Queue Applications	191
7.4 Priority Queue Implementations	191
7.5 Heaps and Binary Heaps	192
7.6 Binary Heaps	193
7.7 Heapsort	198
7.8 Priority Queues [Heaps]: Problems & Solutions	199
8. Disjoint Sets ADT	210
8.1 Introduction	210
8.2 Equivalence Relations and Equivalence Classes	210
8.3 Disjoint Sets ADT	211
8.4 Applications	211
8.5 Tradeoffs in Implementing Disjoint Sets ADT	211
8.8 Fast UNION Implementation (Slow FIND)	212
8.9 Fast UNION Implementations (Quick FIND)	215
8.10 Summary	217
8.11 Disjoint Sets: Problems & Solutions	217
9. Graph Algorithms	219
9.1 Introduction	219
9.2 Glossary	219
9.3 Applications of Graphs	222
9.4 Graph Representation	222
9.5 Graph Traversals	225
9.6 Topological Sort	231
9.7 Shortest Path Algorithms	233
9.8 Minimal Spanning Tree	238
9.9 Graph Algorithms: Problems & Solutions	241
10. Sorting	259
10.1 What is Sorting?	259
10.2 Why is Sorting Necessary?	259
10.3 Classification of Sorting Algorithms	259
10.4 Other Classifications	260
10.5 Bubble Sort	260
10.6 Selection Sort	261
10.7 Insertion Sort	261

10.8 Shell Sort	263
10.9 Merge Sort	264
10.10 Heap Sort	266
10.11 Quick Sort	266
10.12 Tree Sort	268
10.13 Comparison of Sorting Algorithms	268
10.14 Linear Sorting Algorithms	268
10.15 Counting Sort	269
10.16 Bucket Sort (or Bin Sort)	269
10.17 Radix Sort	270
10.18 Topological Sort	270
10.19 External Sorting	270
10.20 Sorting: Problems & Solutions	271
11. Searching	282
11.1 What is Searching?	282
11.2 Why do we need Searching?	282
11.3 Types of Searching	282
11.4 Unordered Linear Search	282
11.5 Sorted/Ordered Linear Search	282
11.6 Binary Search	283
11.7 Interpolation Search	284
11.8 Comparing Basic Searching Algorithms	285
11.9 Symbol Tables and Hashing	285
11.10 String Searching Algorithms	285
11.11 Searching: Problems & Solutions	285
12. Selection Algorithms [Medians]	305
12.1 What are Selection Algorithms?	305
12.2 Selection by Sorting	305
12.3 Partition-based Selection Algorithm	305
12.4 Linear Selection Algorithm - Median of Medians Algorithm	305
12.5 Finding the K Smallest Elements in Sorted Order	305
12.6 Selection Algorithms: Problems & Solutions	306
13. Symbol Tables	313
13.1 Introduction	313
13.2 What are Symbol Tables?	313
13.3 Symbol Table Implementations	313
13.4 Comparison Table of Symbols for Implementations	314
14. Hashing	315
14.1 What is Hashing?	315
14.2 Why Hashing?	315
14.3 HashTable ADT	315
14.4 Understanding Hashing	315

14.5 Components of Hashing	316
14.6 Hash Table	316
14.7 Hash Function	317
14.8 Load Factor	318
14.9 Collisions	318
14.10 Collision Resolution Techniques	318
14.11 Separate Chaining	318
14.12 Open Addressing	318
14.13 Comparison of Collision Resolution Techniques	320
14.14 How Hashing Gets $O(1)$ Complexity?	320
14.15 Hashing Techniques	321
14.16 Problems for which Hash Tables are not suitable	321
14.17 Bloom Filters	321
14.18 Hashing: Problems & Solutions	322
15. String Algorithms	332
15.1 Introduction	332
15.2 String Matching Algorithms	332
15.3 Brute Force Method	332
15.4 Rabin-Karp String Matching Algorithm	333
15.5 String Matching with Finite Automata	334
15.6 KMP Algorithm	335
15.7 Boyer-Moore Algorithm	338
15.8 Data Structures for Storing Strings	338
15.9 Hash Tables for Strings	338
15.10 Binary Search Trees for Strings	338
15.11 Tries	339
15.12 Ternary Search Trees	340
15.13 Comparing BSTs, Tries and TSTs	344
15.14 Suffix Trees	344
15.15 String Algorithms: Problems & Solutions	347
16. Algorithms Design Techniques	355
16.1 Introduction	355
16.2 Classification	355
16.3 Classification by Implementation Method	355
16.4 Classification by Design Method	356
16.5 Other Classifications	357
17. Greedy Algorithms	358
17.1 Introduction	358
17.2 Greedy Strategy	358
17.3 Elements of Greedy Algorithms	358
17.4 Does Greedy Always Work?	358
17.5 Advantages and Disadvantages of Greedy Method	358

17.6 Greedy Applications	359
17.7 Understanding Greedy Technique	359
17.8 Greedy Algorithms: Problems & Solutions	361
18. Divide and Conquer Algorithms	367
18.1 Introduction	367
18.2 What is the Divide and Conquer Strategy?	367
18.3 Does Divide and Conquer Always Work?	367
18.4 Divide and Conquer Visualization	367
18.5 Understanding Divide and Conquer	368
18.6 Advantages of Divide and Conquer	368
18.7 Disadvantages of Divide and Conquer	369
18.8 Master Theorem	369
18.9 Divide and Conquer Applications	369
18.10 Divide and Conquer: Problems & Solutions	369
19. Dynamic Programming	382
19.1 Introduction	382
19.2 What is Dynamic Programming Strategy?	382
19.3 Properties of Dynamic Programming Strategy	382
19.4 Can Dynamic Programming Solve All Problems?	382
19.5 Dynamic Programming Approaches	382
19.6 Examples of Dynamic Programming Algorithms	383
19.7 Understanding Dynamic Programming	383
19.8 Longest Common Subsequence	385
19.9 Dynamic Programming: Problems & Solutions	387
20. Complexity Classes	415
20.1 Introduction	415
20.2 Polynomial/Exponential Time	415
20.3 What is a Decision Problem?	415
20.4 Decision Procedure	416
20.5 What is a Complexity Class?	416
20.6 Types of Complexity Classes	416
20.7 Reductions	418
20.8 Complexity Classes: Problems & Solutions	420
21. Miscellaneous Concepts	422
21.1 Introduction	422
21.2 Hacks on Bit-wise Programming	422
21.3 Other Programming Questions	426
References	432