



Table of Contents

1	Introduction to Data Structures and Algorithm	1
1.1	Introduction	1
1.2	Variables	2
1.2.1	Definition	2
1.3	Data Type	4
1.4	Data Structures	5
1.4.1	Definition	6
1.4.2	Abstract Data Types	6
1.5	Algorithm	7
1.5.1	Definition	7
1.5.2	Why it's Necessary to Analyze an Algorithm?	8
1.5.3	What is the Primary Purpose of Analyzing an Algorithm	8

1.5.4	What is Running Time Analysis	8
1.5.5	How to Compare Different Algorithms?	9
1.6	Define Rate of Growth	9
1.6.1	Commonly used Rate of Growths	10
1.7	Various Types of Analysis	11
1.7.1	Asymptotic Notation	12
1.7.2	Important Notes	13
1.7.3	Why is it called Asymptotic Notation?	13
1.7.4	Guidelines for Asymptotic Notations	14
1.8	Properties for Notations	15
1.9	Commonly Used Logarithms and Summations	15
1.9.1	Logarithms	15
1.9.2	Summations	16
1.10	Master Theorem for Divide and Conquer	16
1.10.1	Problems on Master Theorem for Divide and Conquer	17
1.11	Master Theorem for Subtract and Conquer Recurrence	17
1.12	Method of Guess and Confirm	18
1.13	Amortized Analysis	18
1.14	References	19
2	Brief about Recursion and Backtracking	21
2.1	Introduction	21
2.2	What is Recursion?	21
2.2.1	Properties	22
2.2.2	Implementation	22
2.3	Why Recursion?	23
2.3.1	Time Complexity	23
2.3.2	Space Complexity	23
2.4	Format of Recursive Function	23
2.5	Recursion and Memory (Visualization)	24
2.6	Recursion versus Iteration	26
2.7	Points to Remember about Recursion	26
2.7.1	Direct Recursion	27
2.7.2	Indirect Recursion	27
2.7.3	What is Tail Recursion?	28
2.7.4	What are Linear and Tree Recursion?	29

2.7.5	How to Convert Recursive Functions to be Tail Recursive?	30
2.7.6	How to Convert Tail Recursive Functions to Iterative Functions	31
2.8	Brief about Recursive Algorithm	32
2.9	Problems on Recursion	32
2.10	What is Backtracking?	33
2.10.1	Backtracking Algorithm	34
2.11	References	34
3	Brief about Linked List	36
3.1	Introduction	36
3.2	Array Overview	37
3.3	Linked Lists Vs Dynamic Arrays	38
3.4	Singly Linked Lists	41
3.5	What is Doubly Linked List?	46
3.6	Circular Linked Lists	53
3.7	What is the Unrolled Linked List?	53
3.8	What is the Skip List?	54
3.9	References	54
4	Everything about Stacks	56
4.1	What is a Stack?	56
4.2	How to Use stacks	57
4.2.1	Stack Pop Method	59
4.2.2	Stack Peek Method	60
4.2.3	Boolean Empty Stack	62
4.2.4	Stack Search Method	63
4.3	Stacks in ADT	64
4.4	Exceptions	64
4.5	Implementation of Stack	67
4.5.1	Stack Implementation Using an Array	67
4.5.2	Implementation of Stack Using Linked List	68
4.6	Applications of Stacks	71
4.7	Comparison of Implementation of Stacks	71
4.8	References	72

5	Detailed Overview of Queues	74
5.1	Introduction	74
5.2	Operations on Queue	76
5.2.1	How are Queues Used?	76
5.2.2	Queue Abstract Data Type	77
5.2.3	Queue as an Array	77
5.2.4	Queue as a Linked List	79
5.2.5	Exceptions in Queue	81
5.3	Queue Implementation	81
5.4	References	83
6	Everything about Trees	85
6.1	What is a Tree?	85
6.2	Glossary	86
6.3	What is a Binary Tree?	87
6.3.1	Properties of Binary Tree	87
6.3.2	Types of Binary Tree	88
6.4	Binary Tree Traversals	88
6.5	Generic Trees (N-ary Trees)	90
6.5.1	Advantages	91
6.5.2	Disadvantages	91
6.6	Threaded [Stack or Queue Less] Binary Tree Traversal	91
6.6.1	Introduction to Threaded Binary Tree	91
6.6.2	Why Do We Need Threaded Binary Tree	92
6.6.3	Types of Threaded Binary Tree	92
6.6.4	Threaded Binary Tree Traversal	93
6.7	Expression Trees	94
6.7.1	Construction of Expression Tree	95
6.8	XOR Tree	95
6.9	Binary Search Trees (BSTs)	96
6.9.1	Properties of Binary Search Tree	96
6.9.2	Operation Which can be Performed on Binary Search Tree	97
6.9.3	Advantages of Binary Search Tree	97
6.9.4	Disadvantages of Binary Search Tree	97
6.10	Balanced Binary Search Tree	98
6.10.1	Conversion of a Binary Search Tree to Balanced Binary Search Tree	98

6.10.2	What is the Difference between Binary Search Tree and Binary Tree?	99
6.11	AVL (Adelson-Velskii and Landis) Trees	100
6.11.1	Properties of AVL Tree	100
6.11.2	Height Balanced Tree	101
6.11.3	AVL Rotations	101
6.12	Other Variation in Trees	103
6.13	References	106
7	Detailed Overview of Priority Queues and Heaps	109
7.1	Introduction	109
7.2	Priority Queue ADT	111
7.2.1	Here are the Operations of the Priority Queue ADT	111
7.3	Double Ended Priority Queue	112
7.4	Applications of Priority Queues	112
7.4.1	Some of the Other Implementation Queues	113
7.5	Naïve and Usual Implementation of Priority Queues	113
7.6	Heaps and Binary Heaps	114
7.7	Binomial Tree	115
7.8	Binary Heap	116
7.8.1	Operations of Binary Heap	116
7.8.2	Advantages of Binary Heap	116
7.9	References	117
8	Detailed Overview of Disjoint Set ADT	119
8.1	Introduction	119
8.2	Equivalence Relation and Equivalence Classes	120
8.3	Disjoint Set ADT	121
8.4	Operations on Disjoint sets	121
8.5	Applications	122
8.5.1	What is a Kruskal's Algorithm?	123
8.6	Quick Find Fast Union Implementation	126
8.7	Quick Find Implementation in JAVA	128
8.8	Path Compression	128
8.9	References	129

9	Detailed Overview of Graph Algorithms	132
9.1	Introduction	132
9.2	Definition	133
9.3	Applications of Graphs	134
9.4	Graph Representation	134
9.4.1	Representation Using Sets	134
9.4.2	Adjacency Matrix	135
9.4.3	Adjacency List	135
9.5	Graph Traversals	135
9.5.1	Breadth First Search	136
9.5.2	Depth First Search	140
9.5.3	Topological Sort	140
9.5.4	Shortest Path Algorithms	141
9.5.5	Unweighted Shortest paths	141
9.5.6	Dijkstra's Algorithm	142
9.5.7	All-to-all Shortest Problem	142
9.6	Minimal Spanning Tree	142
9.6.1	Brouvka's Algorithm	144
9.6.2	Kruskal's Algorithm	144
9.6.3	Jarnik Prim's Algorithm	145
9.6.4	Dijkstra's Algorithm	145
9.7	References	145
10	Detailed Overview of Sorting	147
10.1	What is Sorting?	147
10.2	Why is Sorting Necessary?	148
10.3	Classification	148
10.4	Other Classifications	149
10.5	What is a Bubble Sort?	150
10.6	Selection Sort	152
10.6.1	How Selection Sort Works?	154
10.7	Insertion Sort	154
10.7.1	Advantages of Insertion Sorting	154
10.8	Shell Sort	157
10.9	Merge Sort	160
10.10	Heap Sort	161
10.11	Quick Sort	162

10.12 Tree Sort	165
10.13 Comparison of Various Sorting Algorithms	167
10.14 Linear Sorting Algorithms	167
10.15 Counting Sort	168
10.16 Bucket Sort	169
10.17 What is Radix Sort?	171
10.18 Topological Sort	173
10.19 External Sorting	175
10.20 References	175
11 Detailed Overview of Searching	178
11.1 What is Searching?	178
11.2 Why do We Need Searching?	178
11.3 Types of Searching	179
11.3.1 Binary Searching	179
11.3.2 Linear Search	181
11.4 Symbol Tables and Hashing	183
11.4.1 Symbol Table	183
11.4.2 Hashing	184
11.4.3 Hash Function	185
11.5 Search Algorithm	185
11.5.1 Classifications of Search Algorithms	185
11.6 References	187
12 Detailed Overview of Selection Algorithms	189
12.1 Introduction	189
12.2 Selection by Sorting	190
12.3 Partition Based Selection	192
12.4 Incremental Sorting by Selection	192
12.5 Linear Selection Algorithm – A Median of Medians	193
12.6 References	195
13 Detailed Overview of Symbol Tables	197
13.1 Introduction	197
13.2 What are Symbol Tables?	198
13.2.1 Uses of Symbol Table	199
13.2.2 Symbol Table Entries	199
13.2.3 Symbol Table Stores	199

13.2.4	Symbol Table provides the following Information to Compiler	200
13.2.5	Operations on Symbol Table	200
13.2.6	Advantages of Symbol Table	201
13.2.7	Disadvantages of Symbol Table	201
13.3	Symbol Table Implementation	202
13.4	Comparison of Symbol Table Implementations	203
13.4.1	List	203
13.4.2	Linked List	204
13.4.3	Hash Table	204
13.4.4	Binary Search Tree	204
13.5	References	205

14 Detailed Overview of Hashing 206

14.1	What is Hashing?	206
14.2	Why is Hash used?	207
14.3	Hash Table ADT	208
14.3.1	Understanding Hashing	208
14.3.2	Hash Function	209
14.3.3	Load Factor	209
14.4	Collision	210
14.4.1	Separate Chaining	210
14.4.2	What is Open Addressing?	211
14.5	Advantages and Disadvantages of Hashes	211
14.5.1	Advantages	211
14.5.2	Disadvantages	212
14.6	Uses of Hashes	212
14.7	Hash implementation in Java	213
14.7.1	Various Constructors used in Implementing Hash Tables	215
14.7.2	Various Methods used While Implementing Hash Tables	215
14.8	Hashing Techniques	216
14.8.1	Linear Probing	217
14.8.2	Quadratic Probing	218
14.8.3	Double Hashing	219
14.8.4	Bloom Filters	220
14.9	References	222

15 Detailed Overview of String Algorithms	224
15.1 Introduction	224
15.2 String Algorithms	224
15.3 Brute Force Method	227
15.4 Rabin-Karp String Matching Algorithm	229
15.5 String Matching with Finite Automata	230
15.5.1 Automate	230
15.5.2 Finite Automata	231
15.6 KMP Algorithm	233
15.7 Boyce-Moore Algorithm	235
15.8 Data Structures for Storing Strings	237
15.8.1 Storing String as a Character Array	237
15.8.2 Storing String as a Character Pointer	237
15.9 Hash Tables for Strings	238
15.10 Binary Search Trees for String	239
15.11 Tries	240
15.12 Ternary Search Tree	242
15.13 Comparing BSTs, Tries, and TSTs	243
15.13.1 Binary Search Trees	243
15.13.2 Tries	243
15.13.3 Ternary Search Tree	243
15.14 Suffix Trees	244
15.15 References	245
16 A Brief about Algorithm Design Techniques	247
16.1 Introduction	247
16.2 Classification of Algorithms Based on the Representation	249
16.3 Choosing the Right Algorithm	251
16.4 Classification of Algorithm Based on Implementation	251
16.4.1 Serial Implementation	251
16.4.2 Parallel Implementation	251
16.5 Classification of Algorithms Based on Design	252
16.6 Classification Algorithms	253
16.7 References	254
17 Detailed Overview of Greedy Algorithms	258
17.1 Introduction	258

17.2 Greedy Strategy	259
17.3 Elements of Greedy Algorithm	260
17.4 Does Greedy Always Work?	261
17.5 Advantages and Disadvantages of the Greedy Algorithm	263
17.6 Applications of Greedy Algorithm	263
17.6.1 Dijkstra's Algorithm	264
17.6.2 Huffman Coding	265
17.7 Understanding Greedy Technique	266
17.8 References	267

18 Brief about Divide and Conquer Algorithm 269

18.1 Introduction	269
18.2 Divide and Conquer Strategy	270
18.3 Does Divide and Conquer Always Works?	271
18.4 Visualization of Divide and Conquer	272
18.5 Understanding Divide and Conquer	273
18.5.1 Deployment of the Issues	273
18.5.2 Advantages	274
18.5.3 Disadvantages	275
18.6 Master Theorem	275
18.7 Divide and Conquer Applications	277
18.8 References	279

19 Brief About Dynamic Programming 282

19.1 Introduction	282
19.2 What is Dynamic Programming Strategy?	283
19.3 Properties of Dynamic Programming Strategy	284
19.4 Can Dynamic Programming Solve all the Problems?	284
19.5 Dynamic Programming Approaches	285
19.6 Examples of Dynamic Programming Algorithms	286
19.6.1 0-1Knapsack Problem	286
19.6.2 Chain Matrix Multiplication	286
19.6.3 All Pairs Shortest Path	287
19.6.4 The Floyd Warshall Algorithm: Improved All Pairs Shortest Path	287
19.7 Understanding of Dynamic Programming	287
19.8 References	288

20 Brief about Complexity Classes	290
20.1 Introduction	290
20.2 Polynomial/ Exponential Time	291
20.3 Decision Problem	292
20.3.1 Decidability	293
20.3.2 Function Problems	294
20.4 Decision Procedure	294
20.5 What is a Complexity Class?	294
20.6 Types of the Complexity Class	295
20.6.1 Complexity Class P	295
20.6.2 NP class	295
20.7 Reduction	295
20.7.1 Requirement of Reduction	296
20.7.2 Types and Applications of Reduction	296
20.8 References	297
21 Miscellaneous Concepts	298
21.1 Introduction	298
21.2 References	300
22 Appendix A: Abbreviations	302
23 Appendix B: Figures	304
24 Appendix C: Graphs & Tables	308
24.1 Graphs	308
24.2 Tables	308
Index	310