
Table of Contents

Foreword.....	vii
Preface.....	ix
1. Problem Solving.....	1
What Is an Algorithm?	1
Finding the Largest Value in an Arbitrary List	5
Counting Key Operations	6
Models Can Predict Algorithm Performance	7
Find Two Largest Values in an Arbitrary List	12
Tournament Algorithm	16
Time Complexity and Space Complexity	23
Summary	24
Challenge Exercises	25
2. Analyzing Algorithms.....	29
Using Empirical Models to Predict Performance	30
Multiplication Can Be Faster	32
Performance Classes	34
Asymptotic Analysis	36
Counting All Operations	39
Counting All Bytes	40
When One Door Closes, Another One Opens	41
Binary Array Search	42
Almost as Easy as π	44
Two Birds with One Stone	46
Pulling It All Together	50
Curve Fitting Versus Lower and Upper Bounds	52

Summary	53
Challenge Exercises	54
3. Better Living Through Better Hashing.....	57
Associating Values with Keys	57
Hash Functions and Hash Codes	62
A Hashtable Structure for (Key, Value) Pairs	64
Detecting and Resolving Collisions with Linear Probing	65
Separate Chaining with Linked Lists	71
Removing an Entry from a Linked List	74
Evaluation	77
Growing Hashtables	80
Analyzing the Performance of Dynamic Hashtables	85
Perfect Hashing	86
Iterate Over (key, value) Pairs	89
Summary	91
Challenge Exercises	92
4. Heaping It On.....	97
Max Binary Heaps	104
Inserting a (value, priority)	107
Removing the Value with Highest Priority	109
Representing a Binary Heap in an Array	112
Implementation of Swim and Sink	114
Summary	118
Challenge Exercises	118
5. Sorting Without a Hat.....	123
Sorting by Swapping	124
Selection Sort	125
Anatomy of a Quadratic Sorting Algorithm	127
Analyze Performance of Insertion Sort and Selection Sort	129
Recursion and Divide and Conquer	131
Merge Sort	137
Quicksort	141
Heap Sort	145
Performance Comparison of $O(N \log N)$ Algorithms	148
Tim Sort	149
Summary	151
Challenge Exercises	152

6. Binary Trees: Infinity in the Palm of Your Hand.....	153
Getting Started	154
Binary Search Trees	159
Searching for Values in a Binary Search Tree	165
Removing Values from a Binary Search Tree	166
Traversing a Binary Tree	171
Analyzing Performance of Binary Search Trees	174
Self-Balancing Binary Trees	176
Analyzing Performance of Self-Balancing Trees	185
Using Binary Tree as (key, value) Symbol Table	185
Using the Binary Tree as a Priority Queue	187
Summary	190
Challenge Exercises	191
7. Graphs: Only Connect!.....	195
Graphs Efficiently Store Useful Information	196
Using Depth First Search to Solve a Maze	200
Breadth First Search Offers Different Searching Strategy	207
Directed Graphs	215
Graphs with Edge Weights	223
Dijkstra’s Algorithm	225
All-Pairs Shortest Path	237
Floyd–Warshall Algorithm	240
Summary	245
Challenge Exercises	245
8. Wrapping It Up.....	249
Python Built-in Data Types	251
Implementing Stack in Python	253
Implementing Queues in Python	254
Heap and Priority Queue Implementations	255
Future Exploration	256
Index.....	259

