

Competitive Programmer's Handbook

Antti Laaksonen

Draft July 3, 2018

Contents

Preface	ix
I Basic techniques	1
1 Introduction	3
1.1 Programming languages	3
1.2 Input and output	4
1.3 Working with numbers	6
1.4 Shortening code	8
1.5 Mathematics	10
1.6 Contests and resources	15
2 Time complexity	17
2.1 Calculation rules	17
2.2 Complexity classes	20
2.3 Estimating efficiency	21
2.4 Maximum subarray sum	21
3 Sorting	25
3.1 Sorting theory	25
3.2 Sorting in C++	29
3.3 Binary search	31
4 Data structures	35
4.1 Dynamic arrays	35
4.2 Set structures	37
4.3 Map structures	38
4.4 Iterators and ranges	39
4.5 Other structures	41
4.6 Comparison to sorting	44
5 Complete search	47
5.1 Generating subsets	47
5.2 Generating permutations	49
5.3 Backtracking	50
5.4 Pruning the search	51
5.5 Meet in the middle	54

6 Greedy algorithms	57
6.1 Coin problem	57
6.2 Scheduling	58
6.3 Tasks and deadlines	60
6.4 Minimizing sums	61
6.5 Data compression	62
7 Dynamic programming	65
7.1 Coin problem	65
7.2 Longest increasing subsequence	70
7.3 Paths in a grid	71
7.4 Knapsack problems	72
7.5 Edit distance	74
7.6 Counting tilings	75
8 Amortized analysis	77
8.1 Two pointers method	77
8.2 Nearest smaller elements	79
8.3 Sliding window minimum	81
9 Range queries	83
9.1 Static array queries	84
9.2 Binary indexed tree	86
9.3 Segment tree	89
9.4 Additional techniques	93
10 Bit manipulation	95
10.1 Bit representation	95
10.2 Bit operations	96
10.3 Representing sets	98
10.4 Bit optimizations	100
10.5 Dynamic programming	102
II Graph algorithms	107
11 Basics of graphs	109
11.1 Graph terminology	109
11.2 Graph representation	113
12 Graph traversal	117
12.1 Depth-first search	117
12.2 Breadth-first search	119
12.3 Applications	121

13 Shortest paths	123
13.1 Bellman–Ford algorithm	123
13.2 Dijkstra’s algorithm	126
13.3 Floyd–Warshall algorithm	129
14 Tree algorithms	133
14.1 Tree traversal	134
14.2 Diameter	135
14.3 All longest paths	137
14.4 Binary trees	139
15 Spanning trees	141
15.1 Kruskal’s algorithm	142
15.2 Union-find structure	145
15.3 Prim’s algorithm	147
16 Directed graphs	149
16.1 Topological sorting	149
16.2 Dynamic programming	151
16.3 Successor paths	154
16.4 Cycle detection	155
17 Strong connectivity	157
17.1 Kosaraju’s algorithm	158
17.2 2SAT problem	160
18 Tree queries	163
18.1 Finding ancestors	163
18.2 Subtrees and paths	164
18.3 Lowest common ancestor	167
18.4 Offline algorithms	170
19 Paths and circuits	173
19.1 Eulerian paths	173
19.2 Hamiltonian paths	177
19.3 De Bruijn sequences	178
19.4 Knight’s tours	179
20 Flows and cuts	181
20.1 Ford–Fulkerson algorithm	182
20.2 Disjoint paths	186
20.3 Maximum matchings	187
20.4 Path covers	190

[Click here to download full PDF material](#)