CSE200A: Competitive Programming I (Summer 2019) Reading 2

Siddharth Dhawan, Lavina Jain

June 2019

1 Graphs

1.1 Resources

1. Graph Introduction and Breadth First Search
2. Depth First Search, Directed Acyclic Graphs and Topological Sort
3. Refer to the Chapter 4 of cpbook. Read the following topics:
   - Finding Connected Components in Undirected Graph
   - Flood Fill - Labeling the Connected Components
   - Graph Edges Property Check via DFS Spanning Tree
   - Topological Sort (on a Directed Acyclic Graph)

1.2 Practice Problems

1. Party (Easy)
2. Journey (Easy)
3. Topological Sort (Easy)
4. Coloring a Tree (Medium)
5. Xor-tree (Medium)
6. Permutation Recovery (Hard)

2 Greedy Algorithms

A greedy algorithm chooses a solution based on the "best choice" at a particular instance. Greedy algorithms may not always yield an optimal solution, so it is important to check the correctness of your solution using proper proving methods. Greedy Algorithms may seem very intuitive but can be hard to prove. Implementing a greedy solution for a problem during a contest without proof of it's correctness, can result in a lot of time being wasted, in case the solution isn't correct. Therefore, it is always advisable to try proving your greedy approach before you implement it at contest time. However, there can be situations when there isn't much time to prove and you need to depend on your intuition. When you are unable to arrive at a proper proof for your algorithm, try to challenge it's correctness by coming up with possible failure cases. If you try really hard but do not succeed, there is a better chance of your solution being correct. However, without a formal proof of correctness you can never be sure.

2.1 Resources

1. Read Chapter 3 Section 3.3 Greedy, of cpbook.
2. Activity Selection Problem: Thomas H. Cormen, Chapter 16, Section 16.1
3. More.

2.2 Practice Problems

1. Twins (Easy)
2. Random Teams (Easy)
3. Karen and Coffee (Medium)
4. Producing Snow (Hard)

3 Dynamic Programming

3.1 Resources

1. Fibonacci and Shortest Paths.
2. Recitation.
3. Parenthesization, Edit Distance and Knapsack
4. Read Chapter 3 Section 3.4 Dynamic Programming, of cpbook.

3.2 Practice Problems

1. Subset Sum (Easy)
2. Knapsack (Easy)
3. CHEFSOC2 (Medium)
4. Algebra Score (Medium)
5. Riding in a Lift (Hard)

4 More Practice

If you wish to practice more problems, you can find a large collection of problems with difficulty tags, by visiting the following links:

1. Breadth First Search and Depth First Search
2. Greedy Algorithms
3. Dynamic Programming