CSCI 232: Data Structures and Algorithms

Final Study Guide

Logistics

- Thursday, May 9th @ **10:00 AM – 11:50 AM** in Barnard Hall 103
- Time length: 110 minutes. This exam is designed to be completed in 60-75 minutes.
- Open notes. You are allowed to use your laptop, your IDE, any notes, slides, lecture examples, and java documentation. This exam can be completed without a laptop.
- You are NOT allowed to use the internet to access external resources (Google, Stack Overflow, W3 Schools, etc)
- The midterm exam will consist of different types of question, such as:
    - Multiple choice questions
    - True/False
    - Short answer
    - Illustrate the steps of X algorithm
    - Compute the (graph related problem)
    - Describe an algorithm that would do X
    - Given a scenario, What choice of data structure is best to use?

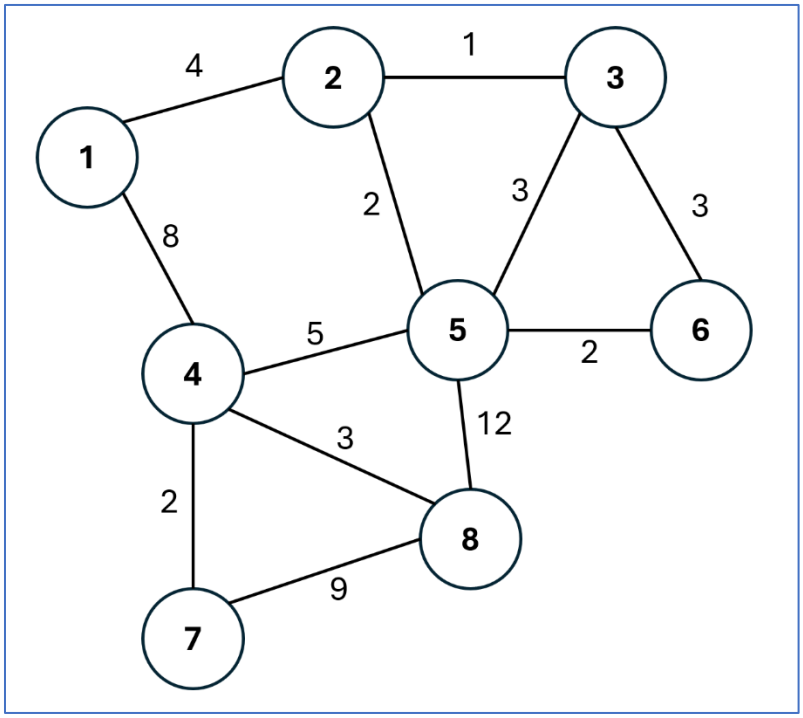
Content

The following topics are all fair game for the midterm exam.

- Graphs (Representation, Traversal)
- Minimum Spanning Tree
    - Kruskal's Algorithm
    - Primm's Algorithm
- Shortest Path
    - Dijkstra's Algorithm
    - A*
- Dynamic Programming
    - Rod Cutting
    - Change Making
    - Edit Distance
- Greedy Algorithms
    - Knapsack Problem
- Closest Pair of Points/Divide and Conquer

Sample Exam Questions

Given the following weighted, undirected graph:



1. What is the degree of vertex 5 ?

2. Please give a path that goes from 1 to 6, and contains a cycle.

3. What is the minimum spanning tree?

4. Is there another minimum spanning tree that has the same cost?

5. What is a memorization table? What is it used for?

6. What is the running time of our dynamic programming change making problem?

   Let n = target change value , m = number of coins in our denomination set

   a. O(logn)
   b. O(n * m)
   c. O(n^2)
   d. O(n^m)

7. True or False. In the fractional knapsack problem, our greedy algorithm by ratio will give us the optimal solution.

8. What is the difference between Kruskal's Algorithm and Primm's Algorithm?

9. Consider the following shortest path greedy algorithm:

   From the starting point, travel the edge with the smallest cost to the next unvisited vertex. Then from there, travel the edge with smallest cost to the next unvisited vertex. Repeat this process until we reach the finish point.

   Will this algorithm always give you the shortest path? If not, sketch a sample graph where this would not work.