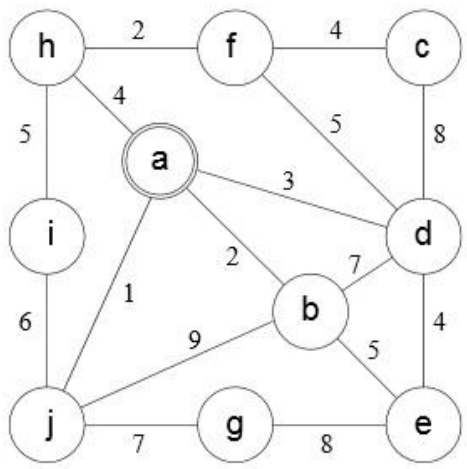
CS 325

Homework 5

Problem 1 (3 points)

Demonstrate Prim's algorithm on the graph below by showing the steps in subsequent graphs as shown in Figures 23.5 on page 635 of the text. What is the weight of the minimum spanning tree? Start at vertex a.



Problem 2 (6 points)

Consider an undirected graph G=(V,E) with nonnegative edge weights $w(u,v) \geq 0$. Suppose that you have computed a minimum spanning tree G, and that you have also computed shortest paths to all vertices from vertex $s \in V$. Now suppose each edge weight is increased by 1: the new weights $w'(u,v) = w(u,v) + 1$.

(a) Does the minimum spanning tree change? Give an example where they change or provide a proof showing they cannot change.

(b) Do the shortest paths change? Give an example where they change or provide a proof showing they cannot change.

Problem 3 (4 points)

In a problem (called the bottleneck-path), you are given a graph G with edge weights, two vertices s and t and a particular weight W; your goal is to find a path from s to t in which every edge has at least weight W.

(a) Describe an efficient algorithm to solve this problem.

(b) What is the running time of your algorithm?

Problem 4 (5 points)

Below is a list of courses and prerequisites for a fictitious CS degree.

Course	Prerequisite
CS 101	None
CS 160	None
CS 161	CS 160
CS 162	CS 160
CS 201	MTH 111
CS 225	MTH 111, CS 162
CS 261	CS 225
CS 271	CS 161, CS 162
CS 301	CS 161, CS 201, CS261
CS 325	MTH 201, CS 225, CS 261
MTH 111	None
MTH 201	MTH 111

(a) Draw a directed acyclic graph (DAG) that represents the precedence among the courses.

(b) Give a topological sort of the graph.

(c) If you are allowed to take multiple courses at one time as long as there is no prerequisite conflict, find an order in which all the classes can be taken in the fewest number of terms.

(d) Describe an efficient algorithm for finding the length of the longest path in the DAG. Determine that length? What does that length represent?

Problem 5 (12 points)

Suppose there are two types of professional wrestlers: "Babyfaces" ("good guys") and "Heels" ("bad guys"). Between any pair of professional wrestlers, there may or may not be a rivalry. Suppose we have n wrestlers and we have a list of r pairs of rivalries.

(a) Give pseudocode for an efficient algorithm that determines whether it is possible to designate some of the wrestlers as Babyfaces and the remainder as Heels such that each rivalry is between a Babyface and a Heel. If it is possible to perform such a designation, your algorithm should produce it.

(b) What is the running time of your algorithm?

(c) Implement your algorithm by writing a program named "wrestlers".

Input: Input is read in from a file **specified in the command line at run time**. The file contains the number of wrestlers n (can be odd or even), followed by their names, the number of rivalries r, and rivalries listed in pairs. *Note: The file contains only one list of wrestlers and one list of rivalries*

Output: Results are outputted to the terminal.

- Yes, if possible followed by a list of the Babyface wrestlers and a list of the Heels (order of the wrestlers does not matter).
- No, if impossible.

Sample Input file1:	**Sample Input file2:**
6 Bear Maxxx Killer Knight Duke Samson 6 Bear Samson Bear Duke Killer Bear Samson Duke Killer Duke Maxxx Knight	6 Bear Maxxx Killer Knight Duke Samson 5 Bear Samson Killer Bear Samson Duke Killer Duke Maxxx Knight

Sample Output 1: Impossible	**Sample Output 2:** Yes Possible Babyfaces: Bear Maxxx Duke Heels: Killer Knight Samson Note: there are other possible solutions.