CS161 - Homework 4
Due: Thursday July 30, 5pm

Notes:

- Many of the algorithms below can be accomplished by either modifying the graph and applying a known algorithm or slightly modifying a known algorithm. Try thinking of these *first* as they will save you a lot of work, and writing :)
- You will be graded on efficiency!
- If not specified in the problem, you may assume whatever graph representation makes your algorithm more efficient (adjacency list or adjacency matrix). State which one you are using.

Problems:

1. (5 points) Write pseudocode for an algorithm which, given an undirected graph G and a particular edge e in it, determines whether G has a cycle containing e. What is the runtime of this algorithm?
2. (10 points) Given an undirected graph G with nonnegative edge weights $w_e \geq 0$. Suppose you have calculated the minimum spanning tree of G and also the shortest paths to all nodes from a particular node $s \in V$. Now, suppose that each edge weight is increased by 1, i.e. the new weights $w'_e = w_e + 1$.
   (a) (5 points) Does the minimum spanning tree change? Give an example where it does or prove that it cannot change.
   (b) (5 points) Do the shortest paths from s change? Given an example where it does or prove that it cannot change.
3. (12 points) T/F. State whether the following are true or false **AND** give a brief, but compelling, justification of your answer. Assume graph G is undirected and connected.

(a) (3 points) If graph G has more than $|V| - 1$ edges, and there is a unique largest edge, then this edge cannot be part of the MST.

(b) (3 points) If G has a cycle with a unique heaviest edge e, then e cannot be part of the MST.

(c) (3 points) If the lightest edge in G is unique, then it must be part of every MST.

(d) (3 points) The shortest path between two nodes is necessarily part of some MST.

4. (8 points) Often there are multiple shortest paths between nodes of a graph. Write pseudocode for an algorithm that given an undirected, unweighted graph G and nodes $u, v \in V$, output the number of distinct shortest paths from u to v. What is the running time?

5. (8 points) Someone suggests to you the following algorithm for finding the shortest distance (sum of edge weights) path from node s to node t in a directed graph with some negative edges: add a large constant to each edge weight so that all the weights become positive, then run Dijkstra's algorithm starting at node s, and return the shorted path found to node t.

   Is this a valid method? Argue that it works correctly or give a counterexample.

6. (8 points) Given a directed graph $G = (V, E)$ with positive edge weights and a particular node $v_i \in V$, give an efficient algorithm for finding the shortest paths between **all pairs of nodes**, with the one restriction that these paths must all pass through v_i. Give the runtime of your algorithm. Points will be deducted for an inefficient algorithm.

7. (5 points) Given the following directed graph specified in adjacency list format:

   $A :\rightarrow B \rightarrow C$
   $B :\rightarrow E$
   $C :\rightarrow D$
   $D :\rightarrow E$
   $E :$

   List all of the topological orderings.

Extra Credit

8. (5 points) You have three containers with sizes 10 pints, 7 pints and 4 pints. The 7 and 4 pint containers start out full of water, but the 10 pint container is empty. You are allowed to pour the contents of one container into another until either a) you source container is empty or b) the destination container is full. We want to know if there is a sequence of pourings that leaves exactly 2-pints in either the 7 or 4 pint container.

   (a) Model this as a graph problem. Describe exactly how to construct the graph and what question about the graph needs to be answered.

   (b) What algorithm should be applied to this problem?