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 :) I don’t expect long answers, but be precise.

- 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.
1. **[8 points]** State whether the following statements about a graph \( G \) that is undirected and connected are true or false and justify your answer.

(a) Prim’s algorithm works correctly if \( G \) has negative edge weights.
(b) The shortest path between two nodes is always part of some MST.

2. **[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?

3. **[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 \( u, v \in V \), outputs the number of distinct shortest paths from \( u \) to \( v \). What is the running time?

4. **[5 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.

   **Hints:**
   - Any path in this problem can be seen as two parts, the part to \( v_i \) and the part from \( v_i \).
   - Look at how we determined if a graph was strongly connected.

5. **[5 points]** If a graph does not have a negative cycle, when calculating the shortest paths from a given vertex using the Bellman-Ford algorithm, we can stop early and do not need to do all \(|V| - 1\) iterations and will still have a correct answer for all the shortest paths from that vertex. Describe how to modify the Bellman-Ford algorithm to stop early when all of the distances are already correct.

6. **[6 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 are \( w'_e = w_e + 1 \).

   (a) (3 points) Does the minimum spanning tree change? Give an example where it does or prove that it cannot change.

   (b) (3 points) Do the shortest paths from \( s \) change? Given an example where it does or prove that it cannot change.