All the following problems are review problems. They are not difficult.

The scopes are CLRS Chapters 22 - 26, and KT Chapters 3, 4, and 7.

Each problem deserves 3 points.

1. For each of the following three statements, decide whether it is true of false. If it is true, give a short explanation. If it is false, give a counterexample.

   (a) Let G be an arbitrary connected, undirected graph with a distinct cost $c(e)$ on every edge e. Suppose e^* is the cheapest edge in G; that is, $c(e^*) < c(e)$ for every edge $e \neq e^*$. Then there is a minimum spanning tree T of G that contains the edge e^*.

   (b) Suppose we are given an instance of the Minimum Spanning Tree Problem on a graph G, with edge costs that are all positive and distinct. Let T be a minimum spanning tree for this instance. Now suppose we replace each edge cost c_e by its square c_e^2, thereby creating a new instance of the problem with the same graph but different costs.

   True of false? T must still be a minimum spanning tree for this new instance.

   (c) Suppose we are given an instance of the Shortest $s-t$ Path Problem on a directed graph G. we assume that all edge costs are positive and distinct. Let P be a minimum-cost $s-t$ path for this instance. Now suppose we replace each edge cost c_e by its square, c_e^2, thereby creating a new instance of the problem with the same graph but different costs.

   True of false? P must still be a minimum-cost $s-t$ path for this new instance.

2. Suppose you are given a connected graph G, with edge costs that are all distinct. Prove that G has a unique minimum spanning tree.

3. Let $G = (V,\ E)$ be an (undirected) graph with costs $c_e \geq 0$ on the edge $e \in E$. Assume you are given a minimum-cost spanning tree T in G. Now assume that a new edge is added to G, connecting two nodes $v,\ w \in V$ with cost c. (Note the new edge may not be in T.)

   Given an efficient algorithm to test if T remains the minimum-cost spanning tree with then new edge added to G (but not to the tree T).

   (a) Make your algorithm run in time $O(|E|)$.

   (b) Can you do it in $O(|V|)$ time?

   You can assume we represent T using an adjacency list.

4. To assess how "well-connected" two nodes in a directed graph are, one can not only look at the length of the shortest path between them, but can also count the *number* of shortest paths.

   This turns out to be a problem that can be solved efficiently, subject to some restrictions on the edge costs. Suppose we are given a directed graph $G = (V,\ E)$, with costs on the edges; the costs may be positive or negative, but every cycle in the graph has strictly positive cost. We are also given two nodes $v,\ w \in V$. Give an efficient algorithm that computes the number of shortest $v - w$ paths in G. (The algorithm should not list all the paths; just the number suffices.)

5. Consider the following problem. You are given a flow network with **unit-capacity** edges: It consists of a directed graph $G = (V,\ E)$, a source $s \in V$, and a sink $t \in V$; and $c_e = 1$ for every $e \in E$. You are also given a parameter k.

   The goal is to delete k edges so as to reduce the maximum $s - t$ flow in G by as much as possible. In other words, you should find a set of edges $F \subseteq E$ so that $|F| = k$ and the maximum $s - t$ flow in $G' = (V,\ E \setminus F)$ is as small as possible subject to this.

   Give a polynomial-time algorithm to solve this problem.

6. Let $G = (V,\ E)$ be a flow network with source s, sink t, and integer capacities. Suppose that we are given a maximum flow in G.

   (a) Suppose that the capacity of a single edge $(u,\ v) \in E$ is increased by 1. Give an $O(V + E)$-time algorithm to update the maximum flow.

   (b) Suppose that the capacity of a single edge $(u,\ v) \in E$ is decreased by 1. Give an $O(V + E)$-time algorithm to update the maximum flow.

7. Suppose a directed graph G represents a communication network. The maximum number of node-disjoint paths from node s to t is called $s - t$ *connectivity*. The $s - t$ *vulnerability* is the minimum number of nodes (besides $s,\ t$) whose removal disconnects s from t. We know that

   Claim 1. *The $s - t$ connectivity equals the $s - t$ vulnerability.*

   (You do not have to to prove Claim 1.) Based on Claim 1, work on the following problem:

   The **edge connectivity** of an **undirected graph** is the minimum number k of edges that must be removed to disconnect the graph. For example, the edge connectivity of a tree is 1, and the edge connectivity of a cycle chain of vertices is 2. Show how the edge connectivity of an undirected graph $G = (V,\ E)$ can be determined by running a maximum-flow algorithm on at most $|V|$ flow networks, each having $O(V)$ vertices and $O(E)$ edges.

   (Hint: The edge connectivity is the minimum of the maximum number of vertex-disjoint paths from s to t.)

8. (**School bus driver assignment**.)

   A bus company has n morning runs and n afternoon runs that it needs to assign to its n drivers. (Each driver has one morning run and one afternoon run.) The runs are of different duration. If the total duration of the morning and afternoon runs assigned to a driver is more than a specified number D, the driver receives a premium payment for each hour of overtime. The company would like to assign the runs to the drivers to minimize the total number of overtime hours.

   (a) Formulate this problem as a matching problem.

   (b) Suppose that we arrange the morning runs in the nondecreasing order of their duration and the afternoon runs in the non-increasing order of their duration. Show that if we assign each driver i to the ith morning run and the ith afternoon run, we obtain the optimal assignment. (Hint: Use the Exchange Argument to prove this.)

9. (**Escape problem.**)

   An $n \times n$ **grid** is an undirected graph consisting of n rows and n columns of vertices. We denote the vertex in the ith row and the jth column by $(i,\ j)$. All vertices in a grid have exactly four neighbors, except for the boundary vertices, which are the points $(i,\ j)$ for which $i = 1,\ i = n,\ j = 1,$ or $j = n$.

   Given $m \leq n^2$ starting points $(x_1,\ y_1), (x_2,\ y_2), \ldots, (x_m,\ y_m)$ in the grid, the escape problem is to determine whether or not there are m vertex-disjoint paths from the starting points to any m different points on the boundary.

   (a) Consider a flow network in which vertices, as well as edges, have capacities. That is, the total positive flow entering any given vertex is subject to a capacity constraint. Show that determining the maximum flow in a network with edge and vertex capacities can be reduced to an ordinary maximum-flow problem on a flow network of comparable size.

   (b) Describe an efficient algorithm to solve the escape problem. (You do not have to analyze its running time.)