In all cases explain clearly and as succinctly as possible.

1. (15 pts) In the topological sort algorithm described in class, we ran DFS on the graph $G$ and returned nodes in decreasing order of finish times. Prove that this algorithm can not produce all possible topological orderings of a graph $G$.

Note first of all that given a graph $G$, there can be several possible topological orderings to it. In order to solve the above problem you have to construct a graph $G$ and a topological ordering on its nodes such that irrespective of how DFS is run on the graph $G$, it will not generate the topological ordering that you propose.

2. (10 pts) In the strongly-connected components algorithm described in class, the second call to DFS was made on the transpose of the graph $G$. Show that the algorithm would not produce the SCC’s of $G$ if the second call to DFS was made on the same graph $G$, i.e., give an example of a directed graph $G$ and the execution of the two DFS’s and show that the second DFS does not yield the SCC’s of $G$.

3. (40 pts) Second-best minimum spanning tree. Let $G = (V, E)$ be an undirected, connected graph with weight function $w : E \rightarrow \mathbb{R}_+$. A second-best minimum spanning tree is a spanning tree that has equal or less weight than any spanning tree excepting those that are minimum spanning trees.

   (a) (10 pts) Tweak the proof given in class to show that if the weight function $w : E \rightarrow \mathbb{R}_+$ is one-to-one, i.e., edges are assigned distinct weights, then the minimum spanning tree is unique.

   (b) (10 pts) Is the above the case with second-best minimum spanning trees too? If so, prove it. If not give a counter-example.

   (c) (20 pts) Let $T$ be a minimum spanning tree of $G$. Prove that there exist edges $(u, v) \in T$ and $(x, y) \notin T$ such that $T - \{(u, v)\} + \{(x, y)\}$ is a second-best minimum spanning tree of $G$.

4. (15 pts) Assume that you have an algorithm Neg-Cycle such that given any directed, connected graph $G$ with weight function $w : E \rightarrow \mathbb{R}$, Neg-Cycle terminates by giving you a negative weight cycle (a cycle such that the sum of the edge weights on the cycle is less than 0) or FALSE if no negative weight cycle exists in $G$.

Consider a currency trader who deals in the currency of $n$ countries and makes money through arbitrage. For example, if 1 dollar is selling for 0.98 euros, and 1 euro is selling for 110 yen and 1 yen is selling for 0.0098 dollars, then the trader can convert 1 dollar into 0.98 euros, convert that into $0.98 \times 110 = 107.8$ yen, and convert it back to $107.8 \times 0.0098 = 1.05644$ dollars for a profit of 5.644 cents.

The currency trader has at any time an $n \times n$ matrix $A$ where $A[i, j]$ represents the conversion rate from $i$ to $j$. How can he use Neg-Cycle to make money?
5. (20 pts) Following is a description of the *longest monotonically increasing subsequence* problem. Given a sequence of numbers \( X = \langle x_1, x_2, \ldots, x_n \rangle \) a monotonically increasing subsequence is any sequence \( X' = \langle x_{a_1}, x_{a_2}, \ldots, x_{a_k} \rangle \) such that \( 1 \leq a_1 < a_2 < \ldots < a_k \leq n \) and \( x_{a_1} \leq x_{a_2} \leq \ldots \leq x_{a_k} \). The goal is to find the length of the longest monotonically increasing subsequence.

(a) (5 pts) We will first solve a restricted sub-case of the above problem: Given a sequence \( X = \langle x_1, x_2, \ldots, x_n \rangle \) find \( L(n) \), the length of the longest monotonically increasing subsequence that ends with \( x_n \).

   Prove that for each \( k \in \{1, \ldots, n\} \), the following relation holds:

\[
L(k) = 1 + \max_{j \in \{1, \ldots, k-1\} \text{ with } x_j \leq x_k} L(j)
\]

(b) (10 pts) Use the above formulation to write an algorithm using dynamic programming that solves for \( L(n) \) in \( O(n^2) \) time.

(c) (5 pts) Show how you can use this algorithm to find the longest monotonically increasing subsequence without the restriction that it end with \( x_n \).