Assignment #2 - SCC, MST

Guidelines
Below we describe the guidelines for this assignment (a question may include specific instructions).

- You are required to prove correctness and analyze the complexity of your algorithm, unless specified otherwise.
- If no specific complexity is mentioned, you should propose an algorithm with the best complexity you can think of.
- Every graph is finite and simple (with no parallel edges or self loops).

We encourage you to attend the workshops or the reception hours and to interact with the team regarding this assignment. You may also email algotechnion@gmail.com. Please keep track on the FAQ published in the course website.

Recommended approach for solving the assignment: after you think of an algorithm, write down the claims that are required for proving its correctness. Only then formalize the algorithm and the proof.

Submission instructions:
- Assignments should be submitted in pairs.
- The assignments should be put in the course box on the first floor and sent electronically in the course website. If you wrote your answers on a paper, please scan it and submit a PDF version.
- Make sure your work is readable, unreadable works will not be checked.
- In each assignment, we will check thoroughly only one or two questions. These will consist of 50% out of the overall assignment’s grade. The other questions will consist of the other 50%, and will be checked briefly.
- Changes added after publishing the assignment are marked in yellow.
Warm up question

The goal of this section is to assist you in better understanding the material before you start solving the assignment. This section is mandatory; however, your answers should be concise.

1. Let $G = (V, E)$ be a directed graph.
   Find a necessary and sufficient condition such that every DFS run over $G$ will result with a single DFS tree. Prove your answer.

Let $G = (V, E)$ be an undirected and connected graph and let $w: E \rightarrow \mathbb{R}$ be a weight function over the edges of $G$. In the next two sections a simple explanation is enough (no formal proof required). However, you are required to analyze the time complexity.

2. Propose an algorithm to compute a maximum spanning tree.
   **Update:** In order to allow you to use this algorithm during the course, we now require a full formal proof for this subsection.
   Notice that there exist solutions for this subsection which use reduction, and are very easy to prove.

3. Let $\alpha, \beta \in \mathbb{R}$.
   For this section, the weight of each edge $e \in E$ is either $\alpha$ or $\beta$. Propose an algorithm, of time complexity $O(V + E)$, to compute a MST for $G$.

Question 1 – Proof of a lemma from the lectures

Consider the algorithm we learned in the lectures for computing the strongly connected components (SCC) of a graph. Let $T_1, ..., T_k$ be the trees of the resulted DFS forest from the second DFS run, and let us denote the vertices sets of these trees by $V_1, ..., V_k$ respectively. Prove that the strongly connected components of the original graph are $V_1, ..., V_k$.

Question 2

A directed graph is called a **bypass graph** if for every pair of vertices $v, u \in V$ there exists another vertex $w \in V$, such that there exists a directed path from $v$ to $w$ and from $u$ to $w$ (it is possible that $v = w$ or $u = w$).

Note that different pairs may be associated with a different vertex $w$.

a. A **Target** in a directed graph $G$ is a vertex $t \in V$ such that for each vertex $v \in V$ there exists a directed path from $v$ to $t$. Prove that a directed graph $G$ is a bypass graph if and only if $G$ has a target.
b. Propose an algorithm that determines whether a given directed graph $G$ is a bypass graph. Your main theorem should show that your algorithm returns "true" if and only if the graph is a bypass graph.

**Question 3**

Let $G = (V, E)$ be an undirected and connected graph and let $w: E \rightarrow \mathbb{R}$ be a weight function over the edges of $G$.

a. For this section only assume $w(e) \neq w(e')$ for every two edges $e, e' \in E$, except two given edges $e_0, e_1 \in E$ who have the same weight, i.e., $w(e_0) = w(e_1)$. Prove that at most one minimum spanning tree (MST) of $G$ contains $e_0$.

*Hint:* Recall tutorial 5.

For the next subsections, we define the weight of a tree to be the weight of the "heaviest" edge in that tree, that is, $w(T) = \max_{e \in T} \{w(e)\}$.

b. Propose an algorithm, of time complexity $O(E \log V)$, to compute a MST according to the above definition.

c. Let $w(e) \neq w(e')$ for every two edges $e, e' \in E$.

Prove/Disprove: $G$ has a single MST (according to the new definition).

**Question 4**

Let $G = (V, E)$ be an undirected and connected graph and let $w: E \rightarrow \mathbb{R}_{\geq 0}$ be a non-negative weight function over the edges of $G$.

A set of edges $F \subseteq E$ is called a **Cycle Representative** if it contains at least one edge from every cycle in $G$.

The weight of a set of edges is defined to be the sum of weights over all the edges in the set, i.e., $w(F) = \sum_{e \in F} w(e)$.

Propose an algorithm, of time complexity $O(E \log V)$, to compute a cycle representative set with minimal weight.

**Good Luck!**