CS 101 Fall 2022 - Quiz 7-A

21/11/2022 - 25 Minutes **Name:** **ID number:**

---

1. (7 points) True or False

For each statement, decide whether it is true (**T**) or false (**F**). Write your answers in the table below.

(a)	(b)	(c)	(d)	(e)	(f)
T	F	F	F	T	T

(a) (1') Let $G = (V, E)$ be a connected undirected graph. If $e \in E$ is an edge such that $w(e) = \min\{w(e') \mid e' \in E\}$, then e may belong to some minimum spanning tree of G.

(b) (1') Let $G = (V, E)$ be a connected undirected graph and T be its minimum spanning tree. Suppose that we add a new vertex v together with a set of some edges $E_v = \{\{v, x\} \mid x \in V\}$ incident to v. To update the minimum spanning tree, we just need to find the edge in E_v with minimum weight, and add it to T.

(c) (1') If the graph contains multi-edges (i.e. two or more edges that are incident to the same two vertices), then the Kruskal's algorithm will fail to find the minimum spanning tree.

(d) (1') Let $G = (V, E)$ be an undirected graph. If $|E| = \Theta(|V|)$, the time complexity of the Prim's algorithm (edges stored in adjacency lists) with a binary heap is asymptotically **worse** than that of the Kruskal's algorithm.

(e) (1') A graph may have multiple minimum spanning trees. For each minimum spanning tree T of a graph G, there is a way to sort the edges of G in Kruskal's algorithm so that the algorithm returns T.

(f) (2') Given a connected undirected graph $G = (V, E)$, the following algorithm can find a minimum spaning tree of G.

Algorithm 1 Maybe-MST

```
Sort the edges into nonincreasing order of edge weights w
T ← E
for e ∈ E, taken in nonincreasing order by weight do
    if T \ {e} is a connected graph then
        T ← T \ {e}
    end if
end for
return T
```

2. (5 points) Dynamic MST

Let $G = (V, E)$ be a connected undirected graph and T is a minimum spanning tree we have computed. Suppose that we decrease the weight of one edge $e = \{u, v\}$ **that is not in** T. How quickly can you update the minimum spanning tree? Design an algorithm that finds the new minimum spanning tree based on T which we have computed. Your algorithm should run in $O(|V|)$ time. Describe your

algorithm in **pseudocode** or **natural language**, and give its time complexity. You don't have to prove its correctness.

Solution: On the tree T, we first perform DFS or BFS starting at u to find the heaviest edge e_h on the path from u to v. If $w(e) < w(e_h)$, update T by $T \leftarrow T \setminus \{e_h\} \cup \{e\}$. Otherwise just do nothing. The algorithm takes $\Theta(|V|)$ time since it runs DFS or BFS on the tree **(not the graph)**.

3. (4 points) Does Greedy Work?

There are n pieces of wood, the i-th of which has length l_i. At each time, one can choose two pieces of wood with length a and b respectively, and use $\max\{a, b\}$ units of glue to glue them into one. The new piece of wood is of length $a + b$. Xiao Wang wants to find minimum amount of glue needed to glue all the pieces of wood into one. He came up with a greedy algorithm: Take two pieces of wood with minimal length each time and glue them into one, and repeat that until there is only one piece of wood left.

Do you think this algorithm can always give the optimal solution? If so, give a proof. If not, provide a counterexample. A counterexample should contain the input, the solution given by the greedy algorithm, and the optimal solution.

Solution: No. A counterexample is $l = \langle 1, 2, 2, 3 \rangle$, for which the greedy algorithm will use 10 units of glue and the optimal will use 9 units.

The greedy algorithm:

- Glue the ones with length 1 and 2 into one with length 3, consuming 2 units of glue.
- Glue the ones with length 2 and 3 into one with length 5, consuming 3 units of glue.
- Glue the ones with length 3 and 5 into one with length 8, consuming 5 units of glue.

Optimal:

- Glue the ones with length 2 and 2 into one with length 4, consuming 2 units of glue.
- Glue the ones with length 1 and 3 into one with length 4, consuming 3 units of glue.
- Glue the ones with length 4 and 4 into one with length 8, consuming 4 units of glue.