CS101 Algorithms and Data Structures

Fall 2019

Homework 7

Due date: 23:59, November 10, 2019

1. Please write your solutions in English.

2. Submit your solutions to gradescope.com.

3. Set your FULL Name to your Chinese name and your STUDENT ID correctly in Account Settings.

4. If you want to submit a handwritten version, scan it clearly. Camscanner is recommended.

5. When submitting, match your solutions to the according problem numbers correctly.

6. No late submission will be accepted.

7. Violations to any of above may result in zero score.

1   (3') Minimum Spanning Tree and Topological Sort

Each question has one or more correct answer(s). Select all the correct answer(s). For each question, you get 0 point if you select one or more wrong answers, but you get 0.5 point if you select a non-empty subset of the correct answers.

Note that you should write you answers of section 1 in the table below.

Question 1	Question 2	Question 3
CD	**D**	**AD**

Question 1. *Topological Sort can be carried out on what kinds of graphs:*

(A) All weight acyclic graphs

(B) All directed graphs

(C) All (Directed) Trees

(D) All unweight directed acyclic graphs

(E) All cyclic directed graphs

Question 2. *Consider the following algorithm that attempts to compute a minimum spanning tree of a connected undirected graph G with distinct edge costs. First, sort the edges in decreasing cost order (i.e., the opposite of Kruskal's algorithm). Initialize T to be all edges of G. Scan through the edges (in the sorted order), and remove the current edge from T if and only if it lies on a cycle of T.*
Which of the following statements is true?

(A) The output of the algorithm will never have a cycle, but it might not be connected.

(B) The algorithm always outputs a spanning tree, but it might not be a minimum cost spanning tree.

(C) The output of the algorithm will always be connected, but it might have cycles.

(D) The algorithm always outputs a minimum spanning tree.

Question 3. *You are given a connected undirected graph G with distinct edge costs, in adjacency list representation. You are also given the edges of a minimum spanning tree T of G. This question asks how quickly you can recompute the MST if we change the cost of a single edge. Which of the following are true? [RECALL: It is not known how to deterministically compute an MST from scratch in $O(m)$ time, where m is the number of edges of G.]*

(A) Suppose $e \in T$ and we increase the cost of e. Then, the new MST can be recomputed in $O(m)$ deterministic time.

(B) Suppose $e \notin T$ and we increase the cost of e. Then, the new MST can be recomputed in $O(m)$ deterministic time.

(C) Suppose $e \in T$ and we decrease the cost of e. Then, the new MST can be recomputed in $O(m)$ deterministic time.

(D) Suppose $e \notin T$ and we decrease the cost of e. Then, the new MST can be recomputed in $O(m)$ deterministic time.

2   (4')True or False

The following statements may or may not be correct. In each case, give a counterexample (if it isn't correct). Always assume that the graph $G = (V, E)$ is undirected. Do not assume that edge weights are distinct unless this is specifically stated.

- If graph G has more than $|V| - 1$ edges, and there is a unique heaviest edge, then this edge cannot be part of a minimum spanning tree.

  Answer:

  False. Consider the following graph:
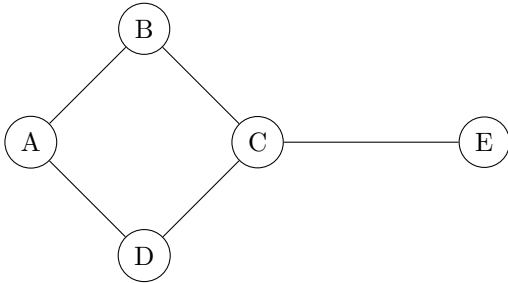
  Even if the edge $C - E$ is the heaviest, it must still be part of the MST since it's the only edge between C and E and the MST must contains an edge between those two vertices.

- If e is part of some MST of G, then it must be a lightest edge across some cut of G.

  Answer:

  True. If e is not the lightest edge across some cut of G, then there must be another lightest edge e' across the cut of G, then e doesn't belong to the MST, which is contradictory. Thus e must be a lightest edge across some cut of G.

- Prim's algorithm works correctly when there are negative edges.

  Answer:

  True. That algorithm always adds the lightest edge between the unfinished MST and other unvisited vertices, negative edges can also be compared with their values so Prim's algorithm works fine with negative edges.

3   (18') Minimum Spanning Tree

Given the following undirected, weighted graph, compute two minimum spanning trees.
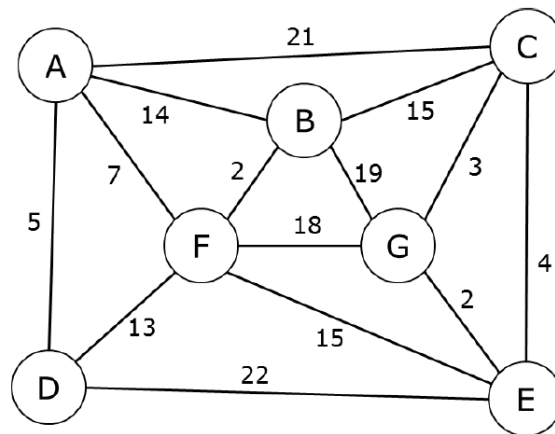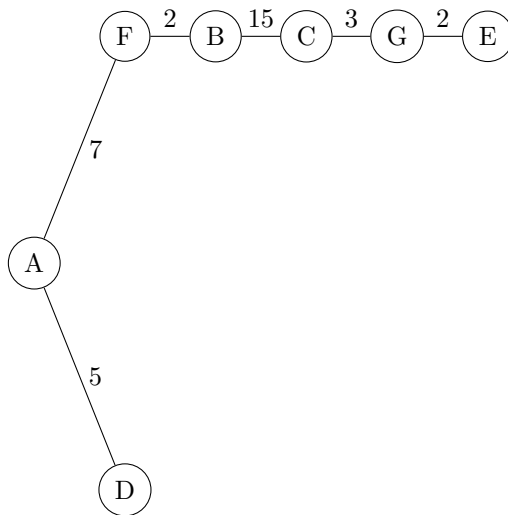

(1)(5') Step through Prim's algorithm to calculate a minimum spanning tree starting from vertex **A**. Show your steps in the table below. As the algorithm proceeds, cross out old values and write in new ones, from left to right in each cell. If during your algorithm two unvisited vertices have the same distance, use alphabetical order to determine which one is selected first. As an example, Row B is filled already.

Vertex	Distance	From(Vertex)
A	-	-
B	~~14~~ 2	~~A~~ F
C	~~21~~ 15	~~A~~ B
D	5	A
E	~~22~~ ~~15~~ ~~4~~ 2	~~D~~ ~~F~~ ~~C~~ G
F	7	A
G	~~18~~ 3	~~F~~ C

(2)(2') Draw the MST you get by this algorithm:

Answer:

(3)(9') Step through Kruskal's algorithm to calculate a minimum spanning tree of the graph. Show your steps in the table below, including the disjoint sets at each iteration. If you can select two edges with the same weight, select the edge that would come alphabetically last (e.g., select E—F before B—C. Also, select A—F before A—B).

Edge Added	Edge Cost	Disjoint Sets
-	-	(A)(B)(C)(D)(E)(F)(G)
E-G	2	(A)(B)(C)(D)(EG)(F)
B-F	2	(A)(BF)(C)(D)(EG)
C-G	3	(A)(BF)(CEG)(D)
A-D	5	(AD)(BF)(CEG)
A-F	7	(ABDF)(CEG)
E-F	15	(ABCDEFG)

(4)(2') Draw the MST you get by this algorithm:

Answer: