Quiz 8 - Graph Algorithms - 2

Due Mar 1 at 11:59pm **Points** 10 **Questions** 8
Available until Mar 2 at 11:59pm **Time Limit** None **Allowed Attempts** 2

Instructions

Instructions

This quiz will test your understanding of the material covered so far this week (MLOs).

This is an online quiz. There will be no time limit to the quiz. You can attempt the quiz twice and the best of the scores will be retained. This is open notes and open internet quiz but refrain from discussing with anybody during the exam.

Note that this test cannot be taken past the due date for any credit.

This quiz is worth 10 points.

You can view the correct answers here after the due date.

Take the Quiz Again

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	8 minutes	10 out of 10

Answers will be shown after your last attempt

Score for this attempt: **10** out of 10
Submitted Mar 1 at 10:58am
This attempt took 8 minutes.

Question 1 1 / 1 pts

Mark True/False: A spanning tree of a graph should contain all the edges of the graph.

- ○ True
- ◉ False

Question 2

1 / 1 pts

Mark True/False: A graph can have many spanning trees.

- ◉ True
- ○ False

Question 3

1 / 1 pts

Consider the graph M with 3 vertices. Its adjacency matrix is shown below.

M = [[0, 1, 1], [1, 0, 1], [1, 1, 0]]

This graph has ____ number of spanning trees.

- ○ 2
- ○ 1
- ○ 4
- ◉ 3

graph: /_\

1. /_

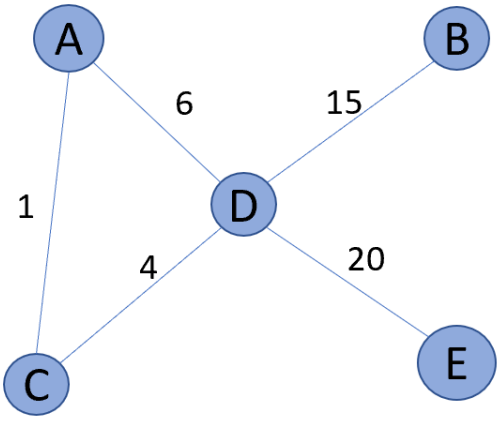
2. _\

3. /\

Question 4

1 / 1 pts

In the following graph which of the edges are part of the minimum spanning tree?

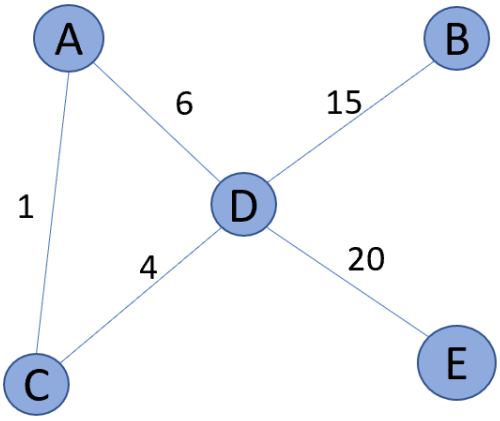


- (a-d)(d-c)(d-b)(d-e)
- (a-c)(c-d)(d-b)(d-e)
- (c-a)(a-d)(d-c)(d-b)(d-e)
- (c-a)(a-d)(d-b)(d-e)

Question 5

1 / 1 pts

What is the cost of the MST of the following graph?



- 21
- 45
- 26
- 40

Question 6

1 / 1 pts

The following pseudocode is for which of the algorithms?

```
def mystery(G):
    s <- pick a source vertex from V

    for v in V:
        dist[v] = infinity
        prev[v] = Empty

    #initalize source
    dist[v] = 0
    prev[v] = s

    #update neighbouring nodes of s
    for node in s.neighbours
```

```
        dist[v] = w(s,node)
        prev[v] = s

    while(len(visited)<len(V)):
        CurrentNode = unvisited vertex v with smallest dist[v]
        MST.add((prevNode, CurrentNode))
        for node in CurrentNode.neighbours:
            dist[node] = min(w(CurrentNode, node), dist[node])
            if dist[node] updated: prev[node] = CurrentNode
        visited.add(CurrentNode)
    return MST
```

- ○ Krushal's Algorithm
- ◉ Prim's Algorithm
- ○ Dijkstra

Question 7

1 / 1 pts

Prim's and Kruskal's algorithms to find the MST follows which of the algorithm paradigms?

- ○ Brute Force
- ○ Dynamic Programming
- ◉ Greedy Approach
- ○ Divide and Conquer

Question 8

3 / 3 pts

Which of the following is true for Prim's algorithm?

- ☑ It never accepts cycles in the MST

- [x] It can be implemented using a heap
- [x] It is a greedy algorithm

Quiz Score: **10** out of 10