

# Week 11 Quiz

**Due** May 28 at 23:59**Points** 10**Questions** 10**Available** May 19 at 10:00 - May 28 at 23:59 10 days**Time Limit** None**Allowed Attempts** Unlimited

## Instructions

You should attempt the quiz after the lecture and your tutorial.

- The quiz is available for a period of 10 days.
- You may attempt the quiz multiple times (if you happen to get a question wrong, you can do it again)
- Your score on the quiz will be recorded in the grade book.
- The quiz might not display equations correctly in some browsers. If you experience problems, we recommend that you use Firefox.

**Note:** you must complete at least eight of the weekly quizzes to meet one of the hurdle requirements in this subject.

## Plagiarism declaration

By submitting work for this quiz I hereby declare that I understand the University's policy on [academic integrity](https://academicintegrity.unimelb.edu.au/) (<https://academicintegrity.unimelb.edu.au/>) and that the work submitted is original and solely my work, and that I have not been assisted by any other person (collusion) apart from where the submitted work is for a designated collaborative task, in which case the individual contributions are indicated. I also declare that I have not used any sources without proper acknowledgment (plagiarism). Where the submitted work is a computer program or code, I further declare that any copied code is declared in comments identifying the source at the start of the program or in a header file, that comments inline identify the start and end of the copied code, and that any modifications to code sources elsewhere are commented upon as to the nature of the modification.

This quiz was locked May 28 at 23:59.

## Attempt History

	Attempt	Time	Score
LATEST	<a href="#">Attempt 1</a>	16 minutes	9 out of 10

Score for this attempt: **9** out of 10

Submitted May 24 at 14:50

This attempt took 16 minutes.

### Question 1

1 / 1 pts

Which one of the following statements about Floyd's algorithm running on a graph with  $V$  nodes and  $E$  edges is correct?

Correct!



The iterative (dynamic programming) version finds the shortest path between all pairs of nodes in time  $O(V^3)$



The recursive version finds the transitive closure of a graph in  $O(3^V)$  time.



The iterative (dynamic programming) version finds the shortest path between all pairs of nodes in  $O(3^E)$  time



The iterative (dynamic programming) version always finds a minimal spanning tree rooted at every node in  $O(V^3)$  time.

Well done! You are on track.

### Question 2

1 / 1 pts

For the weighted shortest path problem, let  $d_v$  be the cost of reaching the current node  $v$ . Let  $w$  be a neighbour of  $v$  and assume the edge cost is  $c_{vw}$ . Let  $d_w$  be the cost of reaching  $w$  prior to examining  $v$ .

Under which conditions is the distance  $d_w$  lowered?

Assume that any ties are broken in favour of the first path seen.

**Correct!**

☒  $d_w > d_v + c_{vw}$

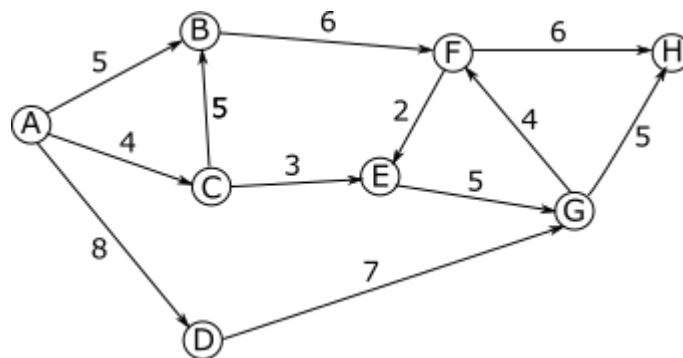
☐  $d_w > d_v$

☐  $d_w > d_v + 1$

☐  $d_v > d_w + c_{vw}$

**Question 3****1 / 1 pts**

Consider the following directed weighted graph:



Run Dijkstra's algorithm on the above graph, with source vertex A. In what order will Dijkstra visit each vertex (i.e. what is the order in which vertices are removed from the priority queue of unvisited vertices)?

☐ A, B, C, D, E, F, G, H

☐ A, C, B, D, E, F, G, H

☒ A, C, B, E, D, F, G, H

☐ A, D, C, B, F, E, H, G

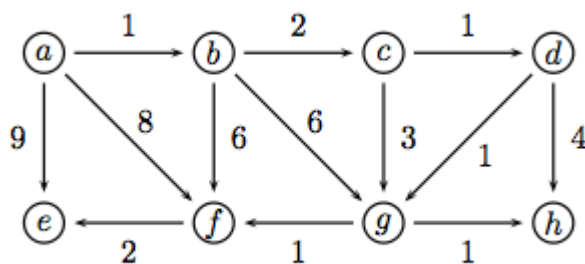
**Correct!**

When we run Dijkstra's algorithm on the given graph, we start with a vertex-cost table that assigns a cost of 0 to vertex A, and a cost of infinity to all other vertices. We first visit the source vertex, A. It's neighbours are B, C, and D. To get to B, C, and D via A incurs a cost of 5, 4, and 8, respectively. We replace the infinity cost assigned to vertices B, C, and D with 5, 4, and 8. Vertex A is marked as visited. The unvisited vertex with the least cost is currently vertex C. Dijkstra will now visit vertex C. C's unvisited neighbours are B and E, with costs (via C) of 9 and 7. We do not update B's cost as its current cost of 5 is less than 9. We update E's cost to 7. Vertex C is now marked as visited. The next vertex visited by Dijkstra is B. Vertex B, with its cost of 5, is the vertex with the smallest cost. We continue in this fashion, visiting E, D, F, G, and H, in that order.

#### Question 4

1 / 1 pts

Suppose we run Dijkstra's single-source shortest-path algorithm on the weighted directed graph below, starting from node a. When the algorithm terminates, the seven edges  $(\text{prev}[u], u)$ , with  $u$  in the set  $\{b, c, d, e, f, g, h\}$ , make up the shortest-path tree. What is the tree's weight, that is, what is the sum of its edges' weights?



Correct!

9

Correct Answers

9 (with margin: 0)

Excellent! Well done.

## Question 5

0 / 1 pts

In what circumstances might we want to use Dijkstra's algorithm to compute all-pairs shortest paths *over* the Floyd-Warshall algorithm?

☐ We have a dense graph with positive edge weights.

☐ Our graph has one or more negative edge weights.

☒ The number of edges in our graph is far greater than the number of vertices (i.e.  $|E| \gg |V|$ ).

You Answered

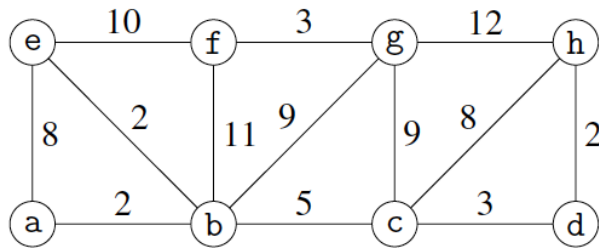
Correct Answer

☐ We have a sparse graph with positive edge weights.

Dijkstra's algorithm cannot be applied to graphs with negative edge weights. The worst case complexity of applying Dijkstra's algorithm to each vertex in a graph is  $O(|V||E| + |V|^2 \log |V|)$ . The worst case complexity of applying the Floyd-Warshall algorithm to compute all-pairs shortest paths is  $O(|V|^3)$ . For dense graphs, where the number of edges is much greater than the number of vertices, Floyd-Warshall is likely to be faster in practice. For sparse graphs, however, using Dijkstra's algorithm on each vertex is likely to be preferable. It may be helpful to think about how the relative complexities compare when  $|E| < O(|V|)$  and  $|E| > O(|V|^2)$ .

**Question 6****1 / 1 pts**

Consider the following graph:



Assuming that ties are resolved in alphabetical order, what is the shortest distance from node a to node g in the graph G when Dijkstra's algorithm is run on the graph?

**Correct!**

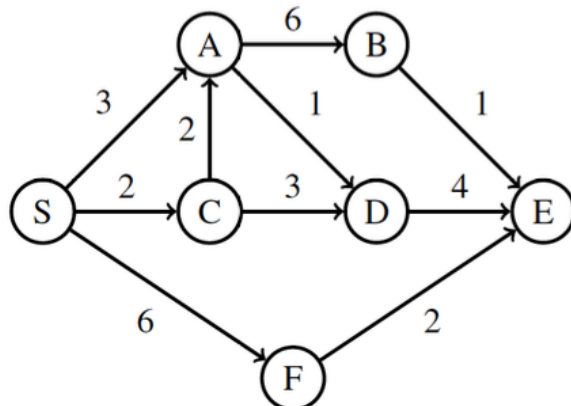
11

**Correct Answers**

11 (with margin: 0)

**Question 7****1 / 1 pts**

Run Dijkstra's algorithm on the following directed graph, starting at node S.



What is the order in which nodes get removed from the priority queue?

Correct!

☒ S, C, A, D, F, E, B

☐ S, C, A, B, E, D, F

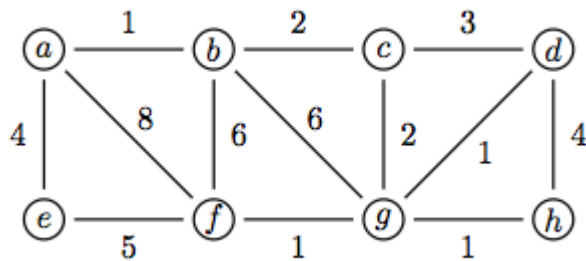
☐ S, C, D, A, F, E, B

☐ S, A, C, F, B, D, E

### Question 8

1 / 1 pts

Consider the graph below. What is the cost of its minimum spanning tree, that is, the sum of its edges' weights?



Correct!

12

Correct Answers

12 (with margin: 0)

You got that right!

### Question 9

1 / 1 pts

A connected weighted undirected graph  $G$  has 57 nodes and 194 edges. How many edges does a minimum spanning tree for  $G$  have?

Correct!

56

Correct Answers

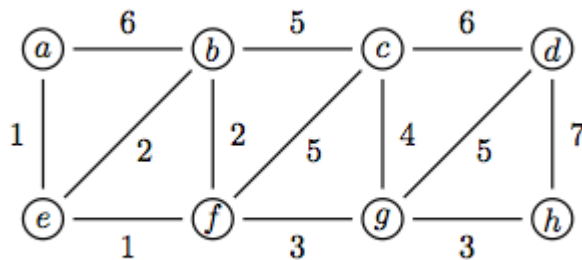
56 (with margin: 0)

Yes, too easy. For a connected undirected graph  $\langle V, E \rangle$ , any spanning tree has  $|V|-1$  edges.

### Question 10

1 / 1 pts

Consider the graph below. How many different minimum spanning trees does it have?



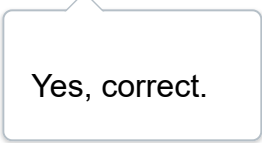
Correct!

2

Correct Answers

2 (with margin: 0)





Yes, correct.

Quiz Score: **9** out of 10