CS 260 - Assignment 15

Professor Mark W. Boady

Week 9

1 Introduction

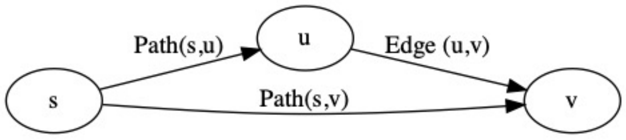
This worksheet is worth 100 points.
You may complete this worksheet using any of the following methods.

- Print out the PDF. Complete by hand. Scan the file.
- Write directly on the PDF with editing software
- Write the Answer in MS Word, Notepad, etc. You **do not** have to rewrite the questions. Just clearly label each answer in your file.

2 Assignment 15

Question 1 : 11 points

Dijkstra's algorithm for finding the shortest path in a graph is base on a property called the Triangle Inequality.



The **Triangle Inequality** states that we have two paths (`path(s,u)`, `path(sv)`) and one edge (`edge(u,v)`). These make a triangle. The longer of the two paths to v cannot be part of the shortest path.

```
function RELAX(u,v)
    global D                                   ▷ Estimated Path Distance
    global w                                   ▷ Adjacency Matrix
    if D[v] > D[u] + w[u][v] then
        D[v] = D[u] + w[u][v]
    end if
end function
```

Assume you have the following Adjacency Matrix w and D. D has the paths starting from node 0.

w=

	0	1	2	3	4
0	0	2	1	∞	8
1	∞	0	∞	∞	5
2	∞	∞	0	2	∞
3	∞	∞	∞	0	1
4	∞	∞	∞	∞	0

D=

Index	0	1	2	3	4
Value	0	2	1	∞	8

(a) (3 points) Draw the Array D after calling `relax(1,4)`.

(b) (5 points) Draw the Array D after calling `relax(2,3)`.
(Use your D from the previous part.)

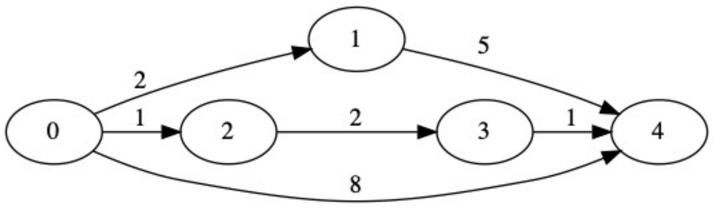
(c) (3 points) Draw the Array D after calling `relax(3,4)`.
(Use your D from the previous part.)

Question 2 : 7 points

Dijkstra's Algorithm uses repeated relaxation to find the shortest path from a start node to all other nodes.

```
function DIJKSTRA(Adjacency Matrix $M$, Start Node $a$)
    D[x] = $\infty$ for all $x$
    D[a] = 0                                        ▷ Path to Start is 0
    Make a Heap $H$ of Nodes based on D[x]
    while $H$ is not empty do
        $u$ is the Minimum of $H$
        for each node $v$ adjacent to $u$ do
            if D[v] > D[u] + M[u][v] then
                D[v] = D[u] + M[u][v]
                Fix Heap $H$
            end if
        end for
    end while
end function
```

We will walk through this algorithm for a simple graph.



We will again start at node 0.

The first distance array looks like

D=

Index	0	1	2	3	4
Value	0	∞	∞	∞	∞

(a) (2 points) We currently have 5 nodes we don't know the shortest path to.
They are listed as (distance,node) below.
$(0, 0), (\infty, 1), (\infty, 2), (\infty, 3), (\infty, 4)$
Which node has the min distance value?

(b) (2 points) There are 3 nodes that are adjacent to the node you selected in the previous question.
A node is **adjacent** if it is connect by 1 edge.
What are the three nodes that are adjacent to the node from part (a)?

(c) (3 points) Relax values in D based on u=answer from (a) and v=all 3 nodes from (b).
Draw the new array D below.

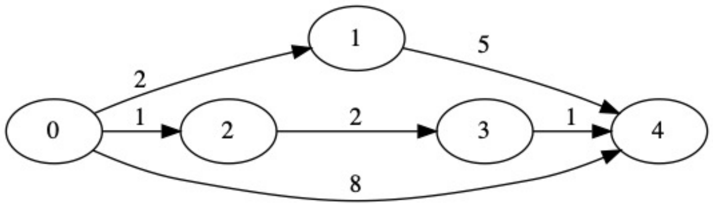
Question 3 : 7 points

You should have finished the previous question with the following distance array.

D=

Index	0	1	2	3	4
Value	0	2	1	∞	8

If you did not get this answer, go back try to figure out why.



At this point, we are sure that our value in $D[0]$ is correct. It was the closest node to the start, so 0 must be the shortest path to it.

We are not sure about any other nodes.

(a) (2 points) The current nodes in the heap are $(2, 1), (1, 2), (\infty, 3), (8, 4)$.
Of these 4 nodes, which is has the minimum estimated distance?

(b) (2 points) There is one node adjacent to your answer from (a). What is it?

(c) (3 points) Update the distance matrix based on u=answer (a) and v= answer (b).
Draw the new array D below.

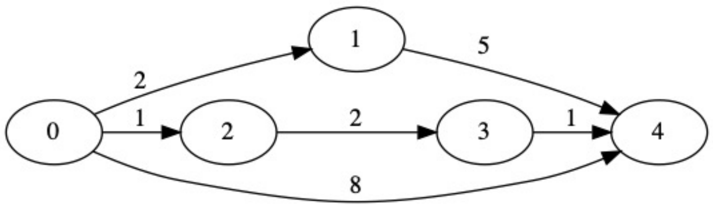
Question 4 : 7 points

You should have finished the previous question with the following distance array.

D=

Index	0	1	2	3	4
Value	0	2	1	3	8

If you did not get this answer, go back try to figure out why.



At this point, we are sure that our value in $D[0]$ and $D[2]$ are correct.

We are not sure about any other nodes.

(a) (2 points) The current nodes in the heap are $(2, 1), (3, 3), (8, 4)$.
Of these 3 nodes, which is has the minimum estimated distance? If there is a tie, use the lower node index.

(b) (2 points) There is one node adjacent to your answer from (a). What is it?

(c) (3 points) Update the distance matrix based on u=answer (a) and v= answer (b).
Draw the new array D below.

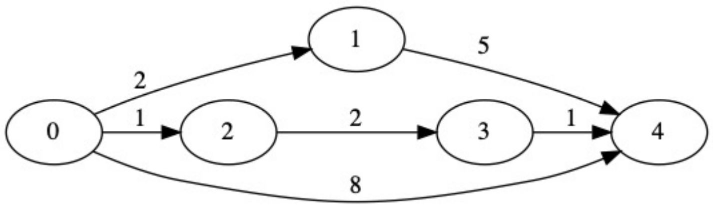
Question 5 : 9 points

You should have finished the previous question with the following distance array.

D=

Index	0	1	2	3	4
Value	0	2	1	3	7

If you did not get this answer, go back try to figure out why.



At this point, we are sure that our value in $D[0]$, $D[2]$, $D[1]$ are correct.

We are not sure about any other nodes.

(a) (2 points) The current nodes in the heap are $(3, 3), (7, 4)$.
Of these 2 nodes, which is has the minimum estimated distance? If there is a tie, use the lower node index.

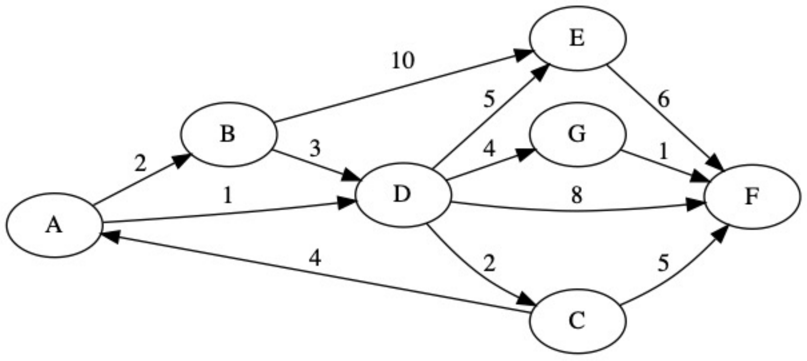
(b) (2 points) There is one node adjacent to your answer from (a). What is it?

(c) (3 points) Update the distance matrix based on u=answer (a) and v= answer (b).
Draw the new array D below.

(d) (2 points) Does your D now contain all the shortest path distances? Why?

Question 6 : 20 points

Run **Dijkstra's Algorithm** on the following graph.



The algorithm is being run with node A as the start.

Index	A	B	C	D	E	F	G
Distance	0	∞	∞	∞	∞	∞	∞

The closest node is A. Updating based on node A's adjacent nodes gives us.

Index	A	B	C	D	E	F	G
Distance	0	2	∞	1	∞	∞	∞

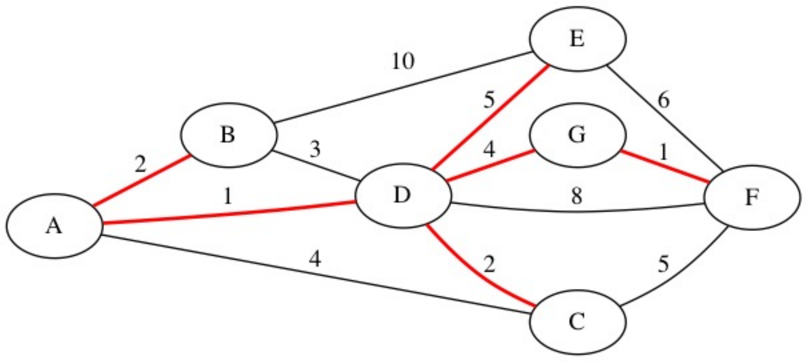
Complete execution of Dijkstra's Algorithm. Draw the distance Array after each set up updates. Write which node you picked as the minimum each time. Draw the 5 arrays required to find the shortest paths.

Question 7 : 9 points

A **Minimum Spanning Tree** is a subset of edges in a graph that has certain properties.

- **Minimum**: The set of edges has the lowest total weight to make a spanning tree.
- **Spanning**: The set of edges can be used to reach all nodes.
- **Tree**: The set of edges contains *no cycles*

The edges highlighted in bold red create a MST in the graph below.



If asked to provide the MST for this graph, we could also give the list of edges.

$$\text{MST}(G) = \{(A, B), (A, D), (D, E), (D, G), (G, F), (D, C)\}$$

(a) (3 points) If we added the edge (C,F) to the set, would the set MST(G) still be a **tree**? Why or why not?

(b) (3 points) If we replaced edge (A,B) in the MST with edge (B,D) would the set MST(G) still be a **tree**? Why or why not?

(c) (3 points) If we replaced edge (A,B) in the MST with edge (B,D) would the set MST(G) still be a **minimum spanning tree**? Why or why not?

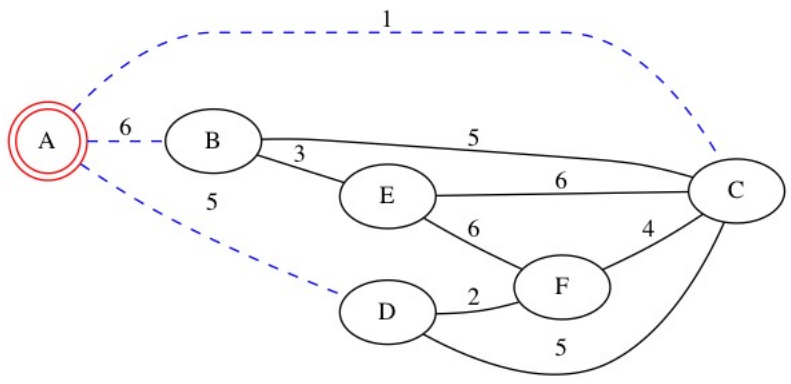
Question 8 : 7 points

In 1830, Vojtech Jarnik discovered an algorithm to find the minimum spanning tree. This algorithm was not made famous until it was rediscovered by Robert Prim in 1957. Prim's algorithm is described below.

```
function PRIM(Graph G=(V nodes, E edges), start node n)
    T = { }                                                          ▷ Set of Edges in MST
    U = { n }                                                        ▷ Set of Nodes in MST
    while $U \neq V$ do                                              ▷ While not a Spanning Tree
        (u,v) = Next Lowest Weight Edge where $u$ already in MST and $v$ is not    ▷ Use a Heap!
        T = $T \cup \{(u,v)\}$
        U = $U \cup \{v\}$
    end while
    return T                                                         ▷ Return MST edges
end function
```

When the algorithm is first called, we only know about the start node.

In this example, the algorithm is called on start node A.



The set U only has one node in it $U = \{a\}$.

We need to select the edge with the lowest weight where one side is in set U.

There are three possible edges that meet this condition $(A, C), (A, B), (A, D)$.

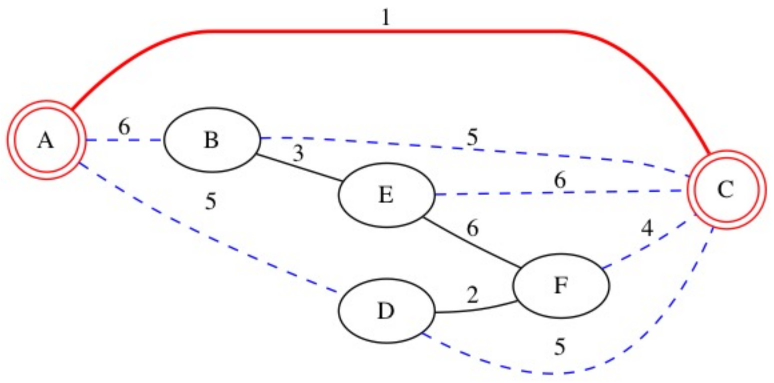
(a) (2 points) Do you feel calling this algorithm **Prim's Algorithm** is fair? Why

(b) (3 points) We only look at edges where one node is in the MST and one is not. Why does this prevent loops?

(c) (2 points) Which of the edges $(A, C), (A, B), (A, D)$ has the lowest weight?

Question 9 : 2 points

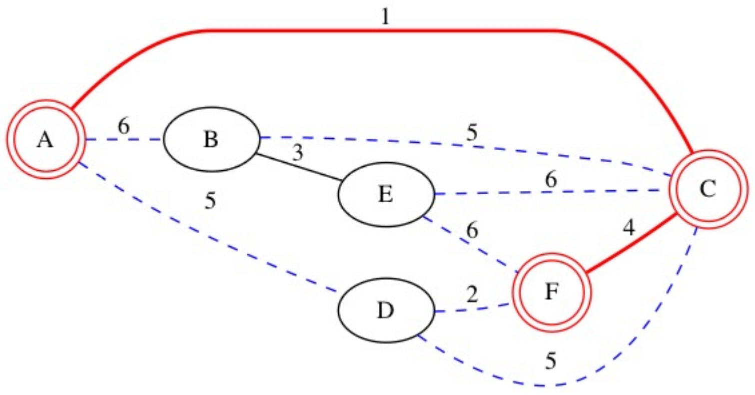
The following image shows the graph after 1 iteration of the loop. The MST so far is colored in bold red. Potential edges to select from are colored in dashed blue.



(a) (2 points) Which of the edges that connects a node in the MST to a node outside the MST has the lowest weight?

Question 10 : 2 points

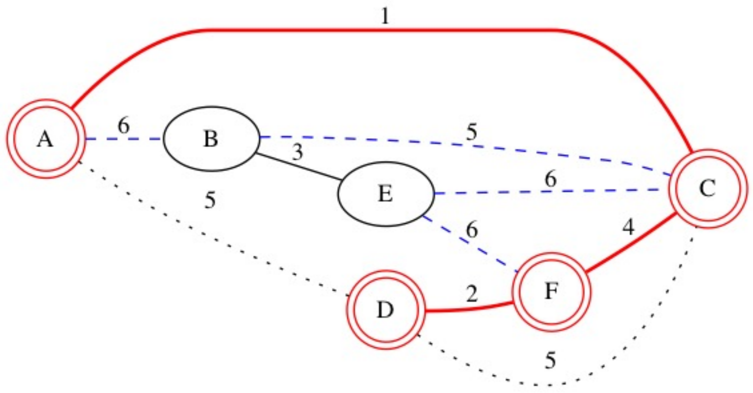
The following image shows the graph after 2 iterations of the loop. The MST so far is colored in bold red. Potential edges to select from are colored in dashed blue.



(a) (2 points) Which of the edges that connects a node in the MST to a node outside the MST has the lowest weight?

Question 11 : 2 points

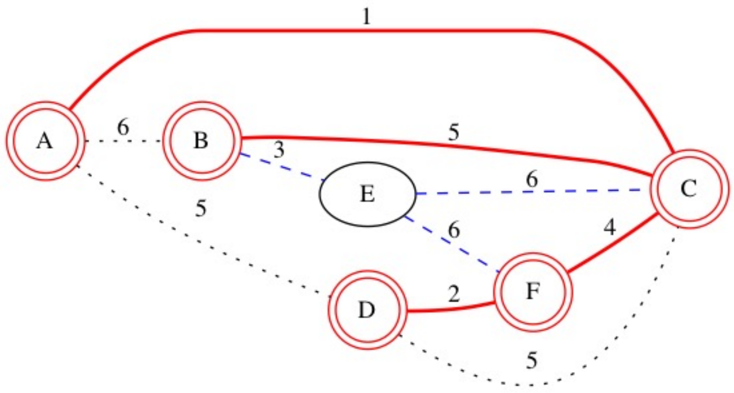
The following image shows the graph after 3 iterations of the loop. The MST so far is colored in bold red. Potential edges to select from are colored in dashed blue. Edges that create cycles are dotted.



(a) (2 points) Which of the edges that connects a node in the MST to a node outside the MST has the lowest weight?

Question 12 : 2 points

The following image shows the graph after 4 iterations of the loop. The MST so far is colored in bold red. Potential edges to select from are colored in dashed blue. Edges that create cycles are dotted.

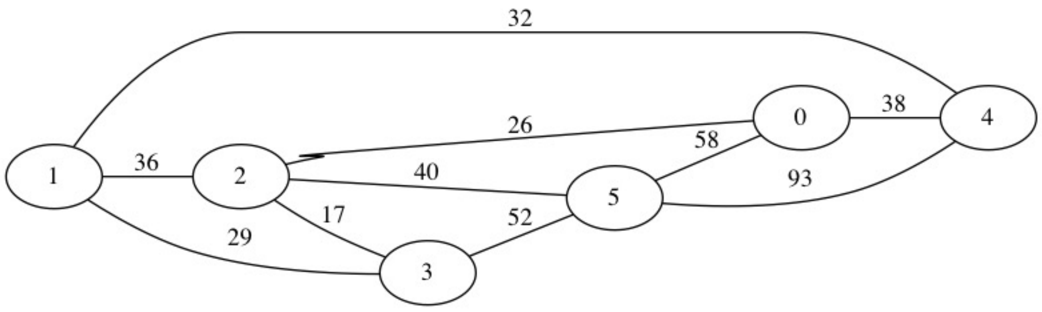


(a) (2 points) Which of the edges that connects a node in the MST to a node outside the MST has the lowest weight?

Question 13 : 15 points

Run **Prim's Algorithm** on the below graph.

1. Start at node 5
2. Write down the edges in the **order** selected.
3. Draw the MST you created.


32
26
38
58
36
40
93
17
52
29
0
1
2
3
4
5