Ece Karasu - 29467

HOMEWORK 5

Question 1

E is the starting vertex (E is the first known vertex, others are unknown). In each iteration we should pick the vertex with the minimum dv (the path to vertex v created with the known vertices) and make it known. Then, we should update its dw and pw (the last vertex that caused a change to dw). An update should be done if the following rule is satisfied:

- dw > dv + Cv,w

(for example, we changed the info of C from (∞ (?)) to (3 (E)) since (∞ > 0+3).)

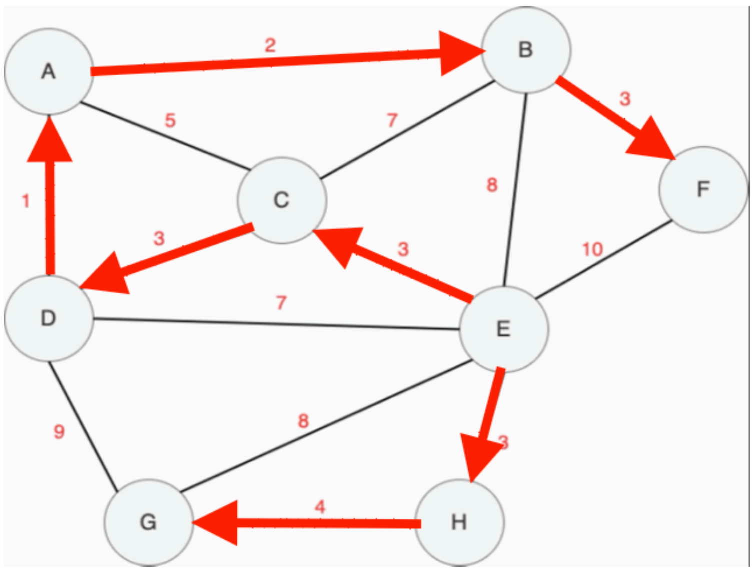
A	B	C	D	E	F	G	H
∞ (?)	∞ (?)	∞ (?)	∞ (?)	0 (E)	∞ (?)	∞ (?)	∞ (?)
∞ (?)	8 (E)	3 (E)	7 (E)		10 (E)	8 (E)	3 (E)
8 (C)	8 (E)		6 (C)		10 (E)	8 (E)	3 (E)
8 (C)	8 (E)		6 (C)		10 (E)	7 (H)	
7 (D)	8 (E)				10 (E)	7 (H)	
	8 (E)				10 (E)	7 (H)	
	8 (E)				10 (E)		
					10 (E)		

According to the table above, the path is E, C, H, D, A, G, B, F.

Question 2

In Prim's Algorithm, we grow a tree starting from a single vertex (E). In each iteration, we add an edge (u,v) if it has the minimum cost and connects u (in the tree) to v (not in the tree).

- Start from E. Then, we go the edge(E,C) which has the minimum cost and connects E and C (E is in the tree, C is not in the tree).
- Now, C is also in the tree. Then, we go the edge(C,D) which has the minimum cost and connects C and D (C is in the tree, D is not in the tree).
- Now, D is also in the tree. Then, we go the edge(D,A) which has the minimum cost and connects D and A (D is in the tree, A is not in the tree).
- Now, A is also in the tree. Then, we go the edge(A,B) which has the minimum cost and connects A and B (A is in the tree, B is not in the tree).
- Now, B is also in the tree. Then, we go the edge(B,F) which has the minimum cost and connects B and F (B is in the tree, F is not in the tree).
- Now, F is also in the tree. Then, we go the edge(E,H) which has the minimum cost and connects E and H (E is in the tree, H is not in the tree).
- Now, H is also in the tree. Then, we go the edge(H,G) which has the minimum cost and connects H and G (H is in the tree, G is not in the tree).
- Now, G is also in the tree. All vertices are connected.



Ece Karasu - 29467

Question 3

In Kruskal's Algorithm, we grow a forest where each vertex is a single tree at the beginning. In each iteration we select the edge with the smallest weight (for example, start with edge (A,D)) and add it to our forest if it does not create a cycle.

1- Connect A and D (cost = 1).

2- Connect A and B (cost = 2).

3- Connect B and F (cost = 3).

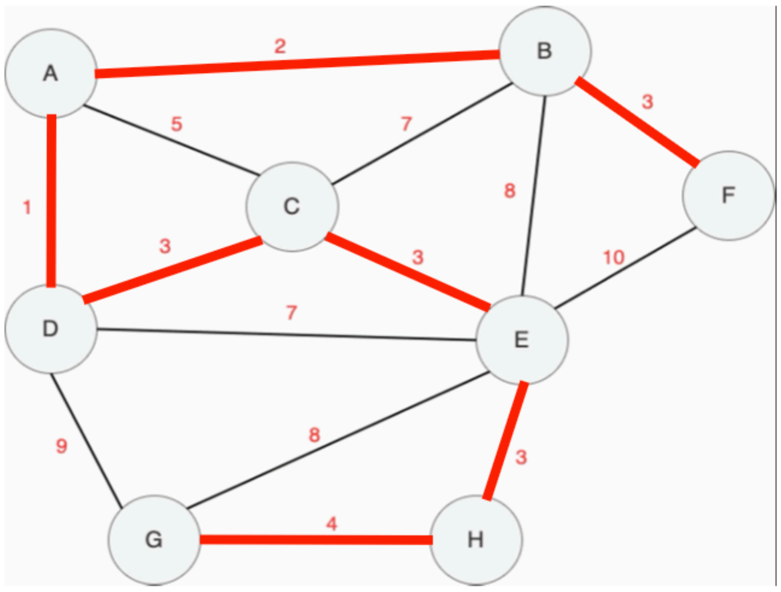
4- Connect D and C (cost = 3).

5- Connect C and E (cost = 3).

6- Connect E and H (cost = 3).

7- Connect H and G (cost = 4).

8- All vertices are connected.



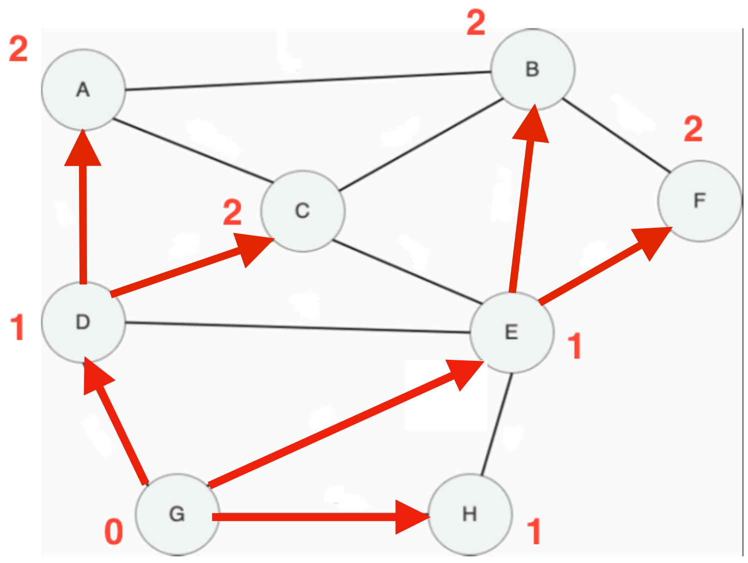
Ece Karasu - 29467

Question 4

Our starting vertex is G. The shortest path from G to G is 0. In each iteration we find the adjacent vertices of the current vertex and label them as the number of edges in the adjacent vertex's shortest path.

- The starting vertex is G. The shortest path from G to G is 0.
- Adjacent vertices to G are D, E, and H. Label them as 1.
- Adjacent vertices to D are A, C, and E. Label A and C as 2. Do not change E since 1 edge is already shorter than 2 edges.
- Adjacent vertices to E are B, C, and F. Label B and F as 2. Changing C is not necessary since it is already labeled as 2.
- No vertices are left. The shortest path from G to every other vertices is found.

- G to G => 0 edge
- G to D => 1 edge
- G to E => 1 edge
- G to H => 1 edge
- G to A => G -> D -> A => 2 edges
- G to C => G -> D -> C => 2 edges
- G to B => G -> E -> B => 2 edges
- G to F => G -> E -> F => 2 edges



Ece Karasu - 29467

Question 5

In topological sort, in each iteration we pick the vertex with in-degree 0 (it means that there is 0 edges coming to that vertex). We print it out. Then, we remove it. (degrees of other vertices should be decreased by 1). This process should be implemented until no vertex is left. If there are more than one vertices with in-degree 0, we can pick arbitrarily.

1- Start with S since it is the only vertex with in-degree 0.

2- After removing S, the next vertex with in-degree 0 becomes G.

3- After removing G, the next vertices with in-degree 0 become D and H. (it is an arbitrary choice, I picked D)

4- After removing D, the next vertices with in-degree 0 become H and A. (I picked H)

5- After removing H, the next vertex with in-degree 0 becomes A.

6- After removing A, the next vertices with in-degree 0 become B and E. (I picked B)

7- After removing B, the next vertex with in-degree 0 becomes E.

8- After removing E, the next vertex with in-degree 0 becomes I.

9- After removing I, the next vertex with in-degree 0 becomes F.

10- After removing F, the next vertex with in-degree 0 becomes C.

11- After removing C, the next vertex with in-degree 0 becomes T.

12- No vertex is left.

The result: S, G, D, H, A, B, E, I, F, C, T.