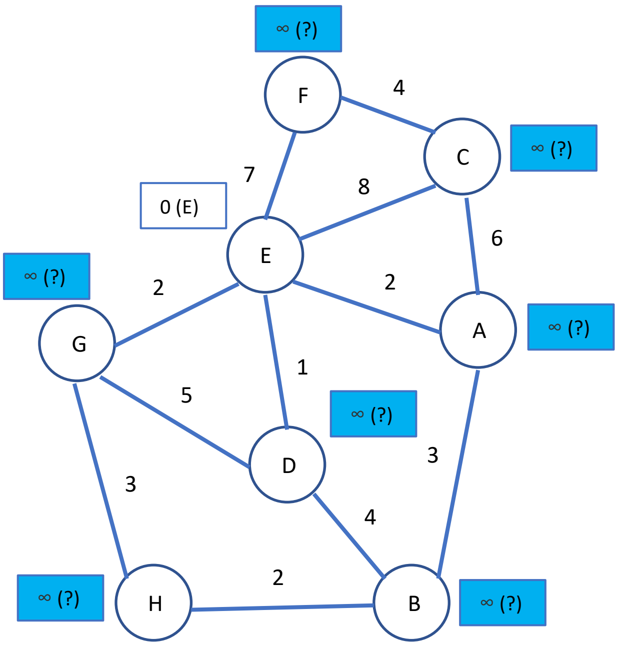
Şevki Aybars Türel 28238
Homework 4 Solutions

Question 1 Solution

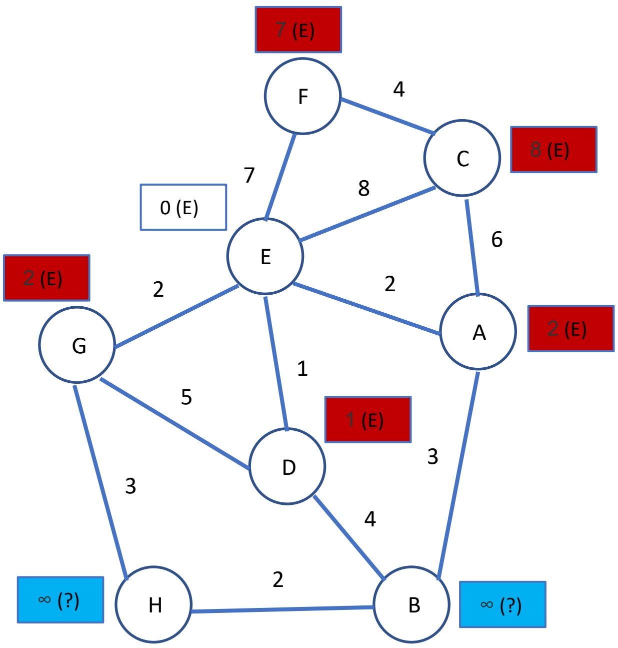
- Blue cells represent the cells that were not visited in that iteration.
- Red cells represent the cells that were visited and updated cells in that iteration.
- Yellow cells represent the cells that were visited but not updated because the new path did not change the smallest path.
- White cells represent the visited cells.

Initial State



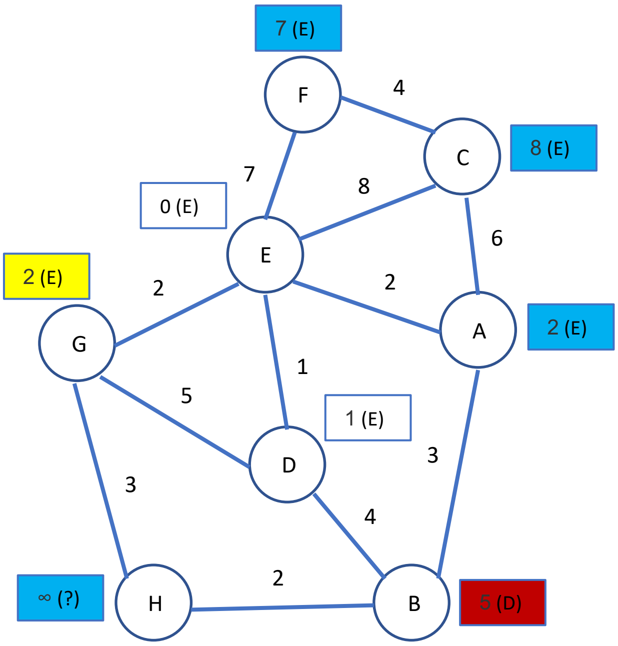
E vertex is marked as visited because it is starting vertex and it points itself. Other cells marked as blue because at the initial states that are not visited yet.

1. Iteration



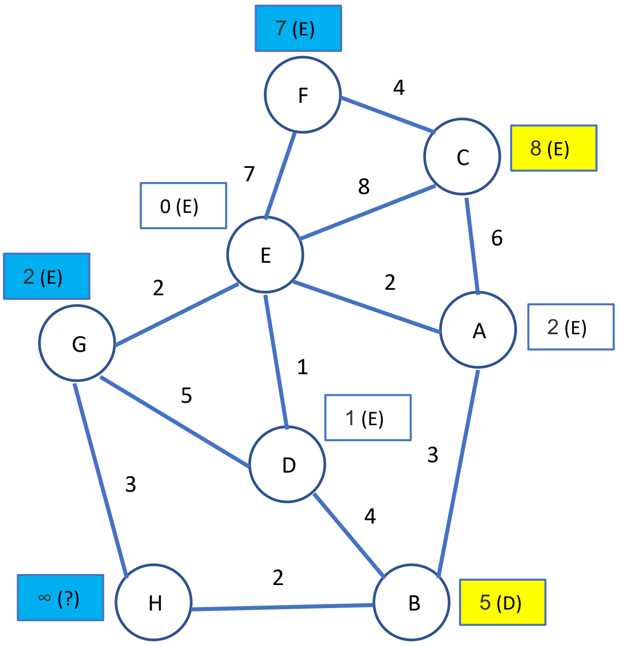
Red cells are the current adjacent to E vertex and all of them updates shortest path and blue ones did not change.

2. Iteration



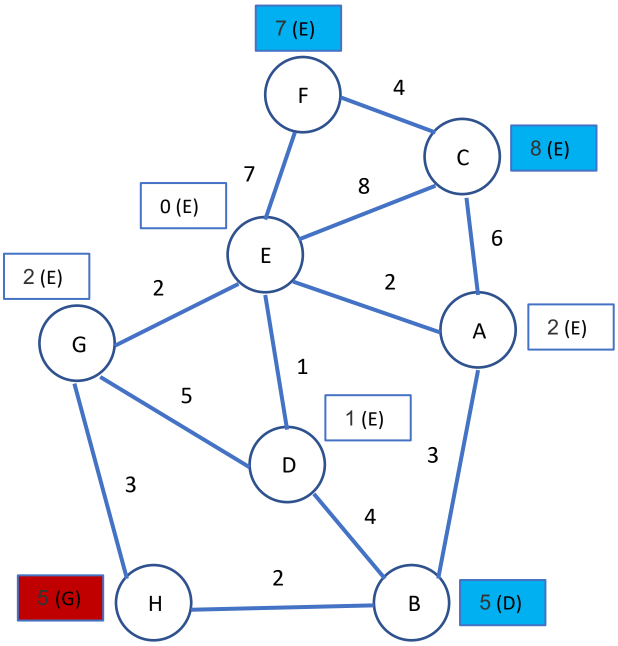
D is the smallest vertex that has not been visited yet. It marked as visited node and current adjacent nodes to D are visited. It only updates B vertex and it did not change the G vertex's smallest path.

3. Iteration



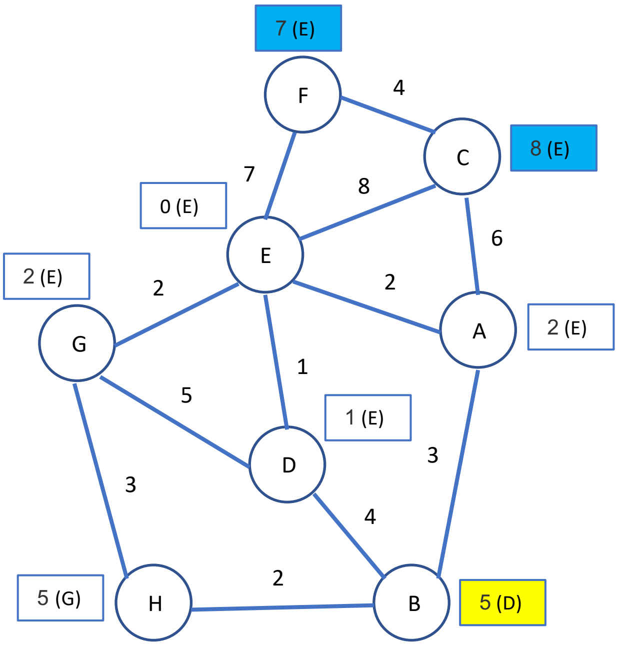
A is the smallest vertex that has not been visited yet. It did not change the shortest path that goes to A's adjacent vertices.

4. Iteration



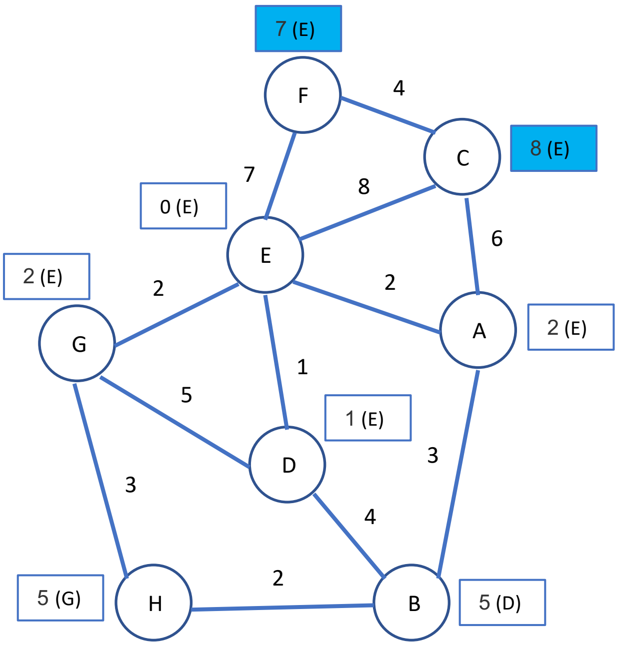
G is the smallest vertex that has not been visited yet. It marked as visited node and current adjacent nodes to G are visited. It only updates H vertex and it did not visit visited nodes (white cells).

5. Iteration



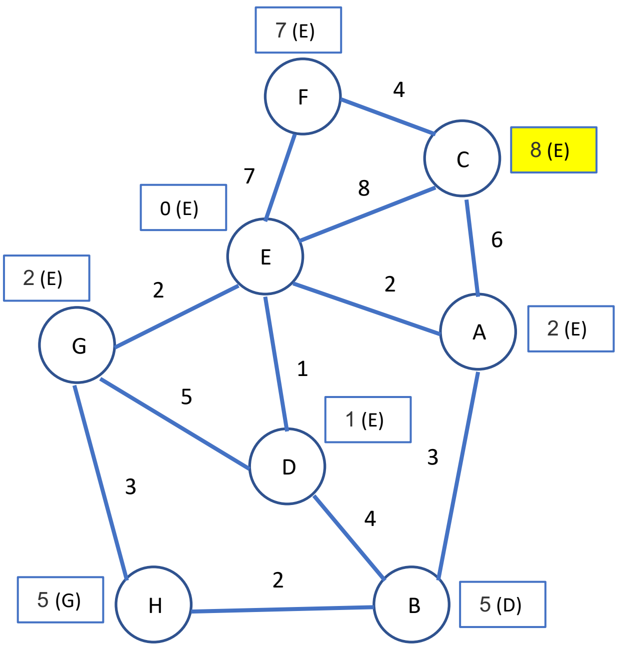
H is the smallest vertex that has not been visited yet. It marked as visited node and current adjacent nodes to H are visited. It did not update any vertices because the adjacent vertices have the smallest value.

6. Iteration



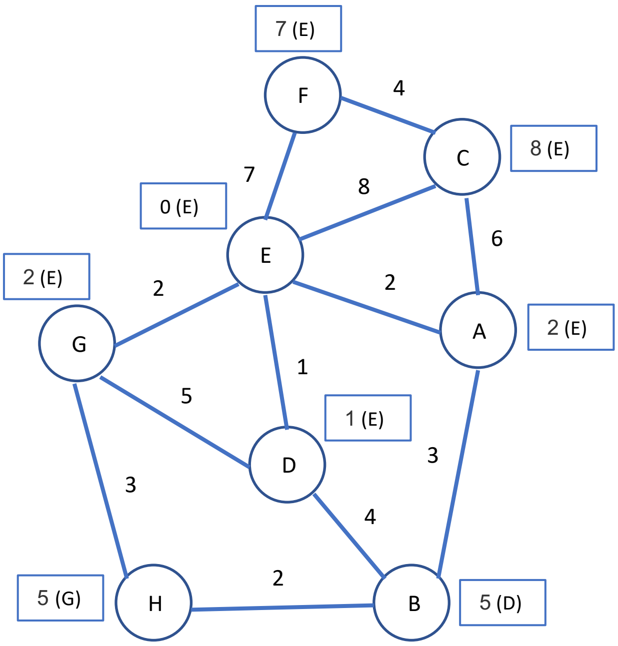
B is the smallest vertex that has not been visited yet. It marked as visited node. It did not visit any adjacent because all of them are visited.

7. Iteration



F is the smallest vertex that has not been visited yet. It marked as visited node. It did not visit any adjacent because there were not any vertex updated.

Last Iteration and Final Version



All vertices are visited and algorithm stops here.

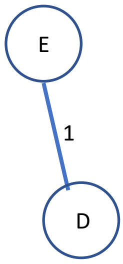
Vertex	Distance	From
A	2	E
B	5	D
C	8	E
D	1	E
E	0	E
F	7	E
G	2	E
H	5	G

Result:

Shortest path to A = E -> A
Shortest path to B = E -> D -> B
Shortest path to C = E -> C
Shortest path to D = E -> D
Shortest path to E = E -> E
Shortest path to F = E -> F
Shortest path to G = E -> G
Shortest path to H = E -> G -> H

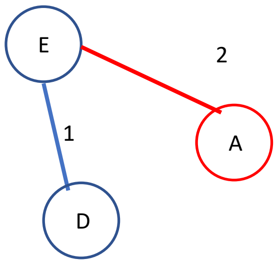
Question 2 Solution

1. Iteration

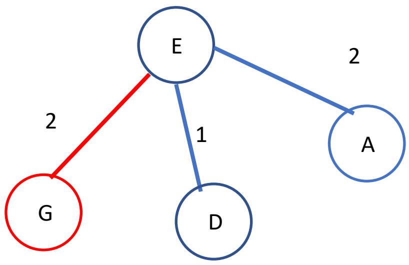


The smallest edge was chosen. Every iteration we will get the smallest edge that are connected the vertices in the tree.

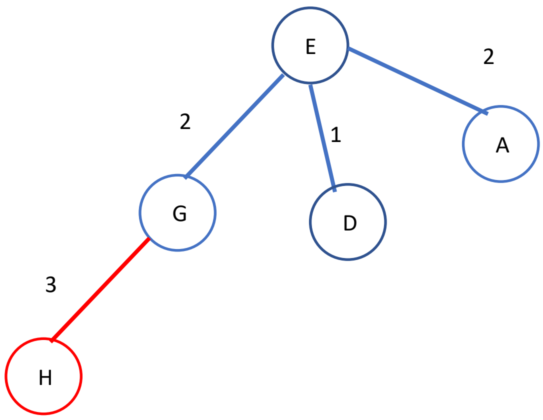
2. Iteration



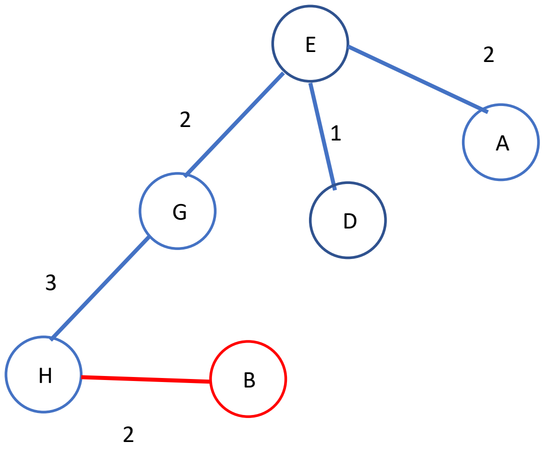
3. Iteration



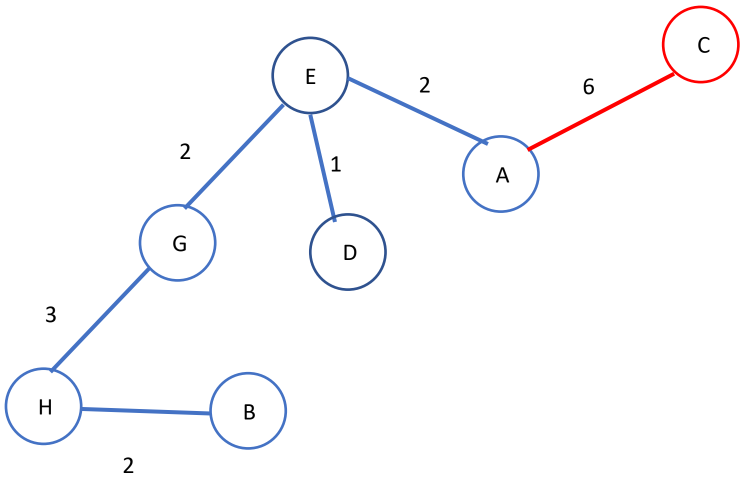
4. Iteration



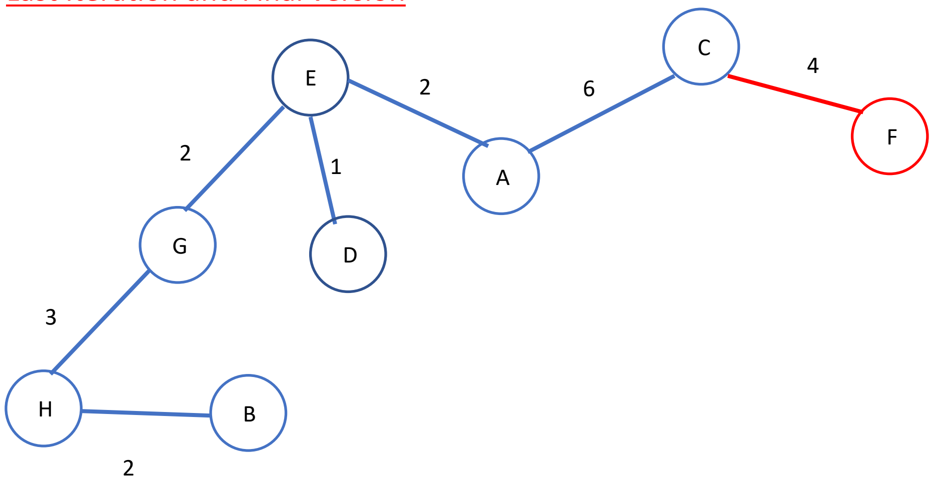
5. Iteration



6. Iteration

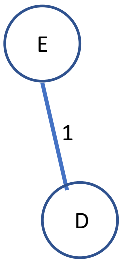


Last Iteration and Final version



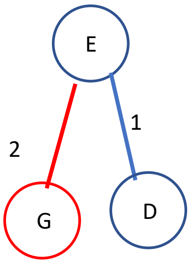
Question 3 Solution

1. Iteration

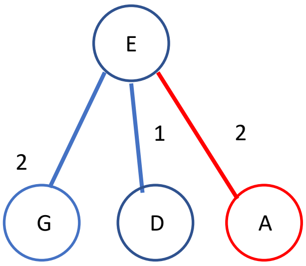


The smallest edge was chosen. Every iteration we will get the smallest edge and it does not have to connected to current vertices in the tree.

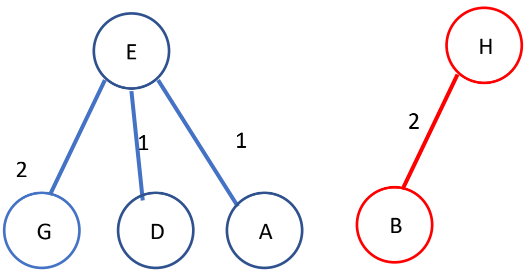
2. Iteration



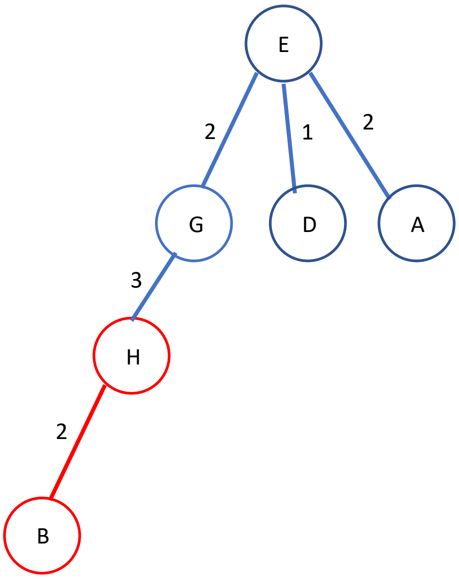
3. Iteration



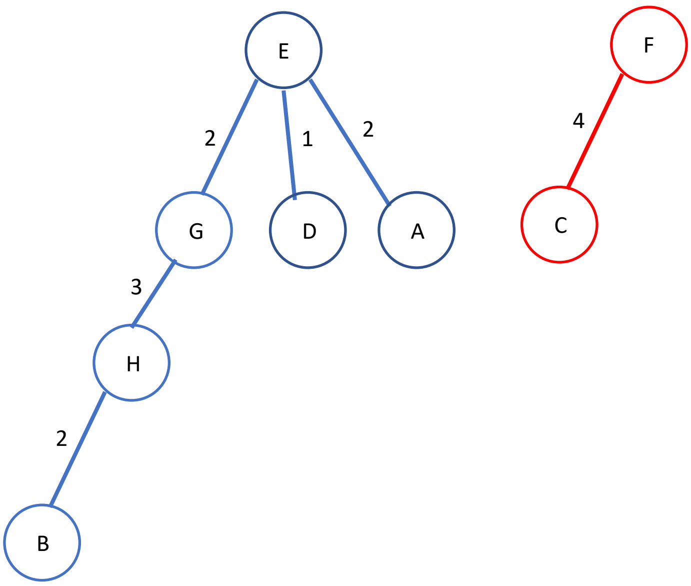
4. Iteration



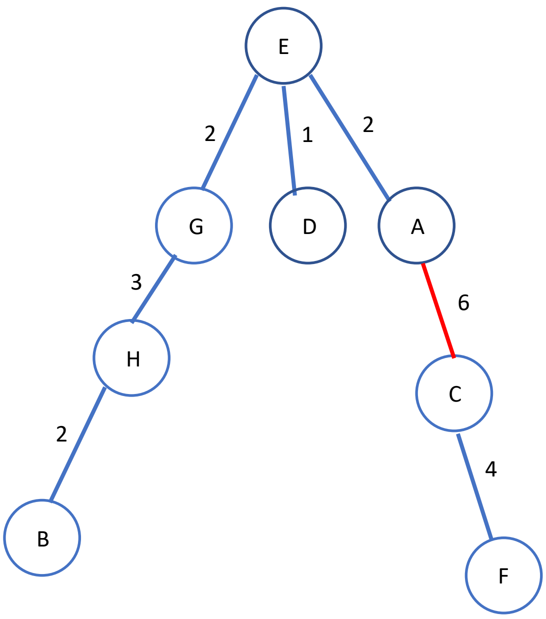
5. Iteration



6. Iteration

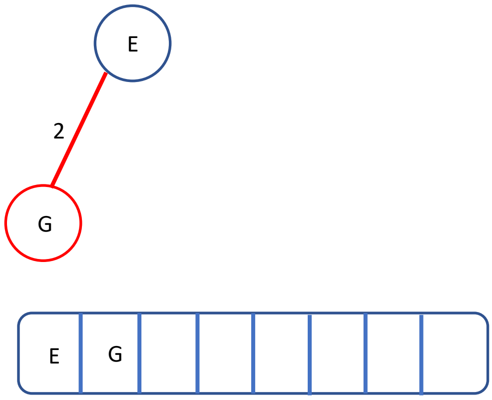


Last Iteration and Final Version



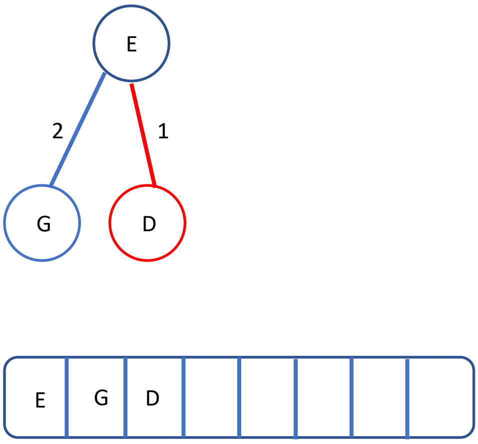
Question 4 Solution

1. Iteration

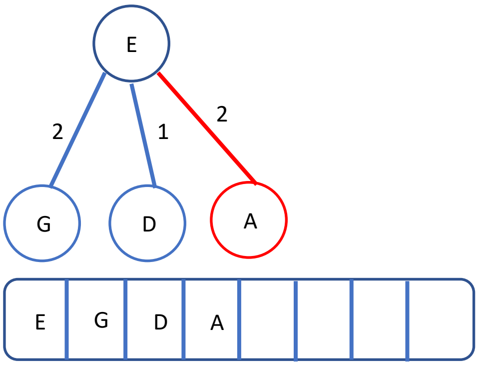


In every iteration, I will get the adjacent vertices that are adjacent to selected vertex and every iteration I will queue these vertices. If current vertex did not have any further adjacent vertex, I will dequeue that vertex and I will continue the checking adjacent vertices for new top element.

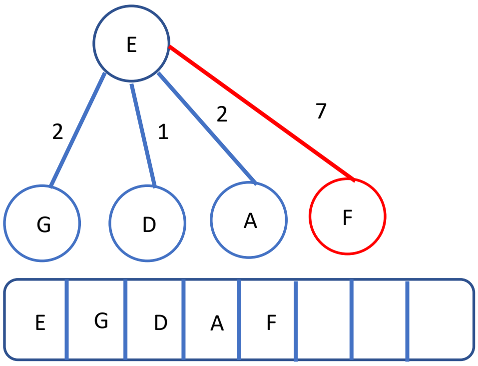
2. Iteration



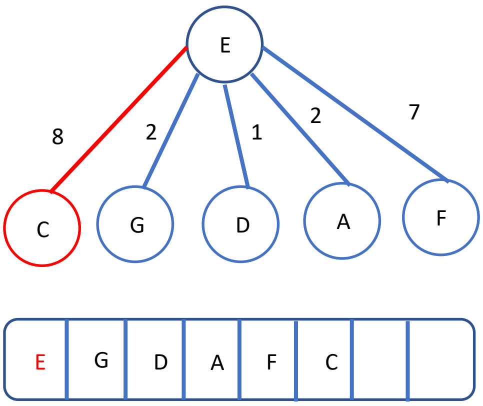
3. Iteration



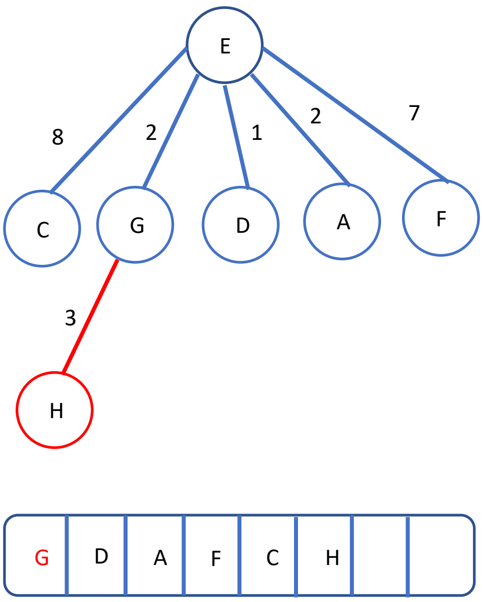
4. Iteration



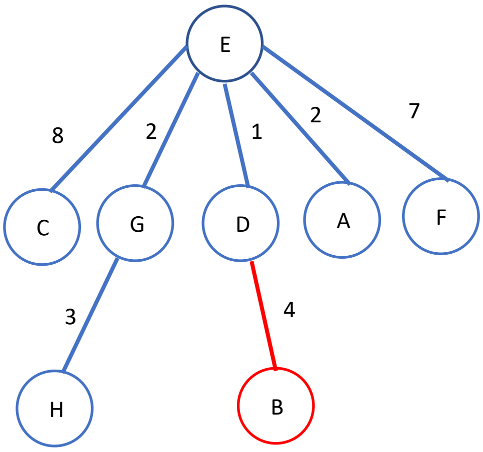
5. Iteration



6. Iteration



7. Iteration



D	A	F	C	H	B		

In these iterations, I dequeue the elements and check there is adjacent vertex. If there is no dequeue element one by one until the queue gets empty.

8. Iteration

A	F	C	H	B			

9. Iteration

F	C	H	B				

10. Iteration

C	H	B					

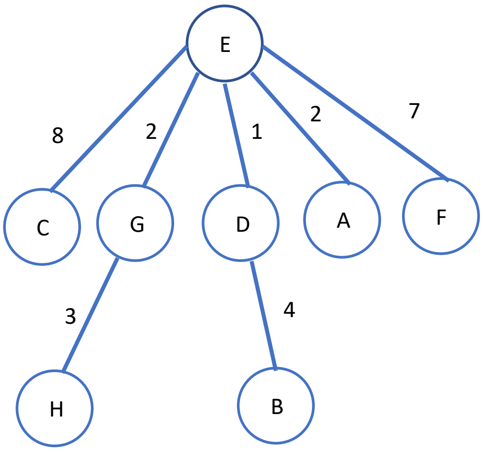
11. Iteration

H	B						

12. Iteration

B							

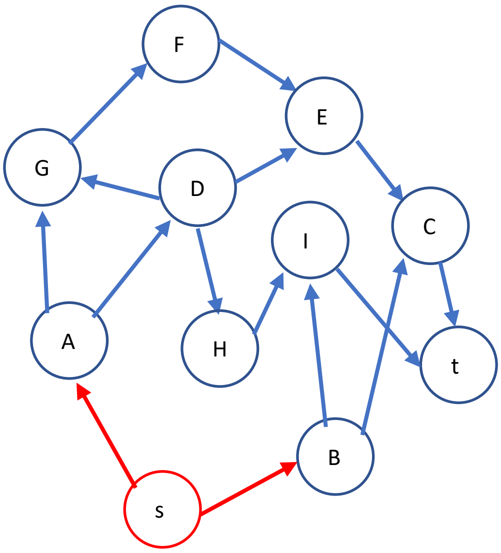
Final Version



E -> G -> D -> A -> F -> C -> H ->B

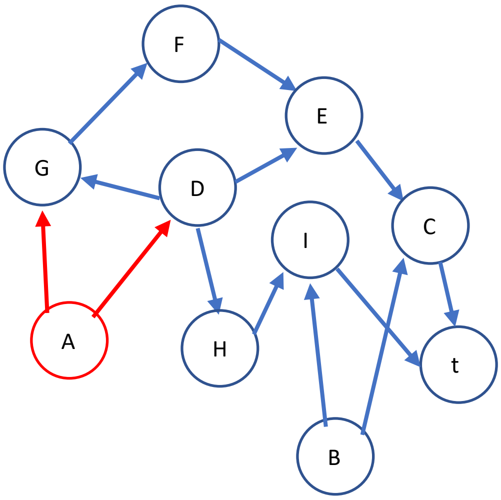
Question 5 Solution

1. Iteration



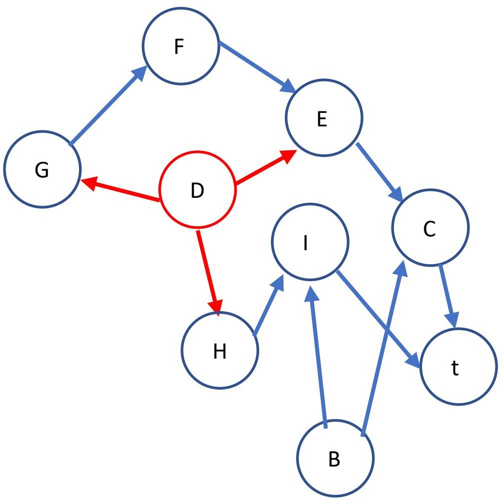
s

2. Iteration



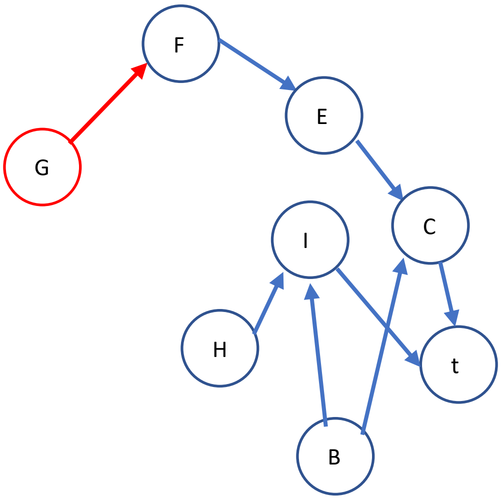
s -> A

3. Iteration



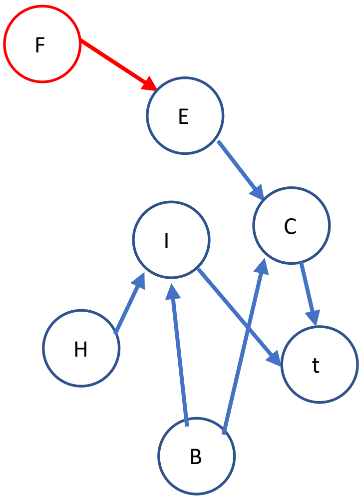
s -> A -> D

4. Iteration



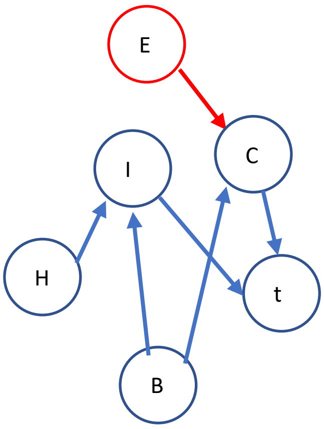
s -> A -> D -> G

5. Iteration



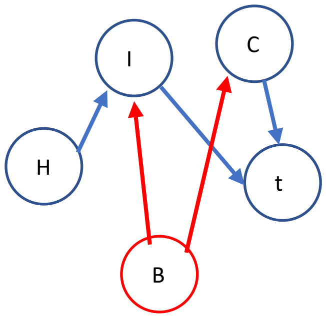
s -> A -> D -> G -> F

6. Iteration



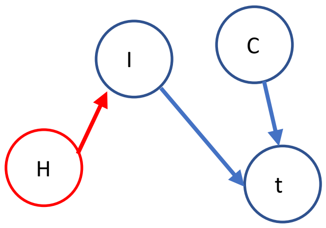
s -> A -> D -> G -> F -> E

7. Iteration



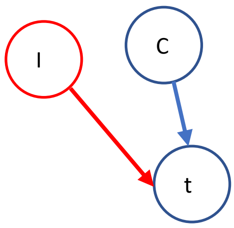
s -> A -> D -> G -> F -> E-> B

8. Iteration



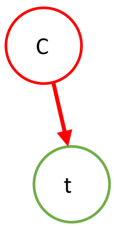
s -> A -> D -> G -> F -> E-> B -> H

9. Iteration



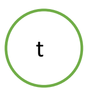
s -> A -> D -> G -> F -> E-> B -> H -> I

10. Iteration



s -> A -> D -> G -> F -> E-> B -> H -> I -> C

11. Iteration



s -> A -> D -> G -> F -> E-> B -> H -> I -> C-> t