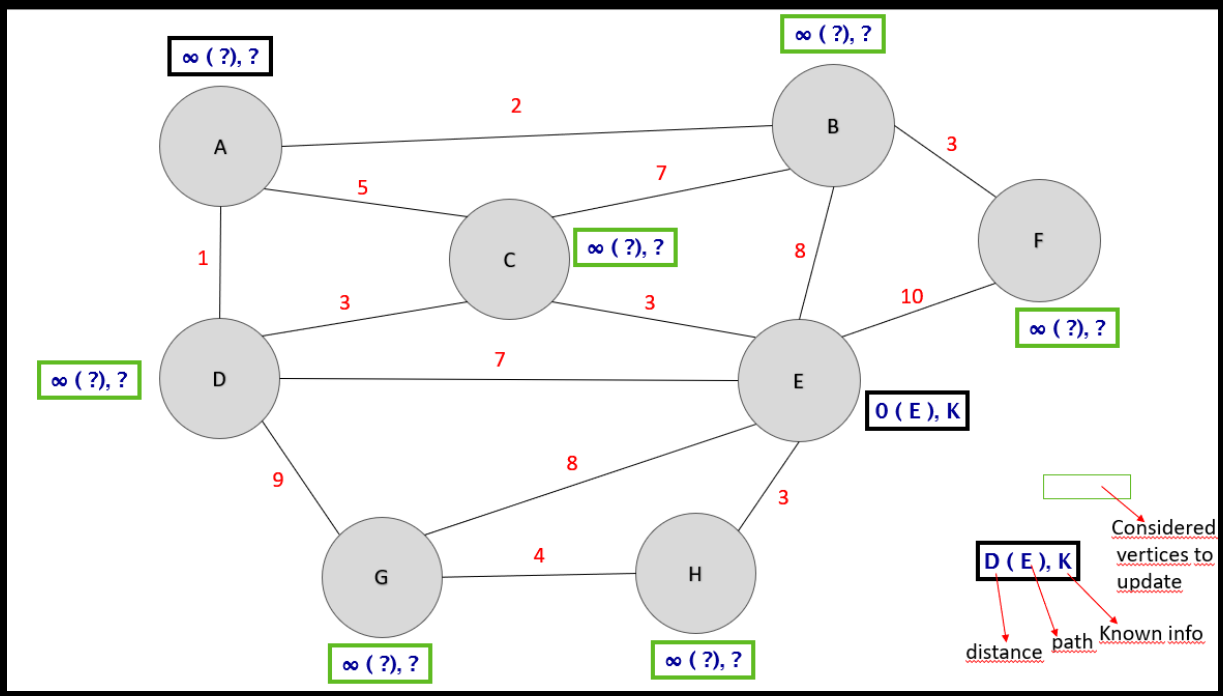
Oytun Kuday Duran 28357

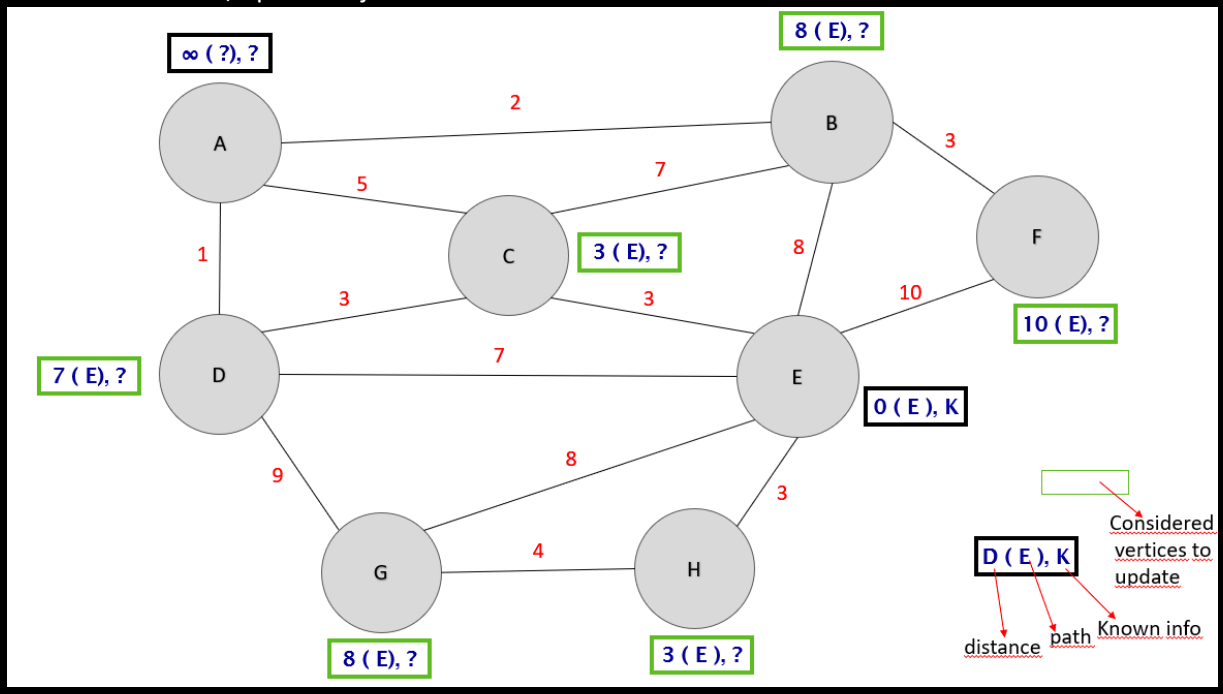
CS 300 HW

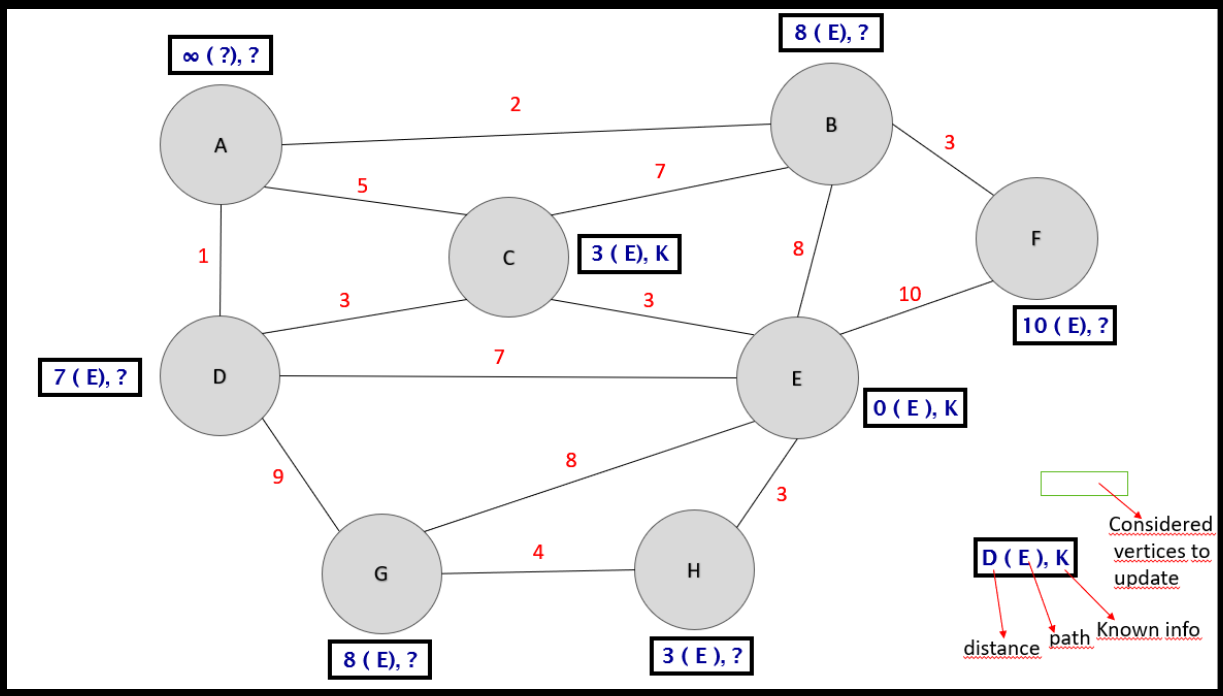
The steps are created in powerpoint. I couldn't directly copy them because Word couldn't succesfully convert them. They are screenshots from my powerpoint slides.

QUESTION 1

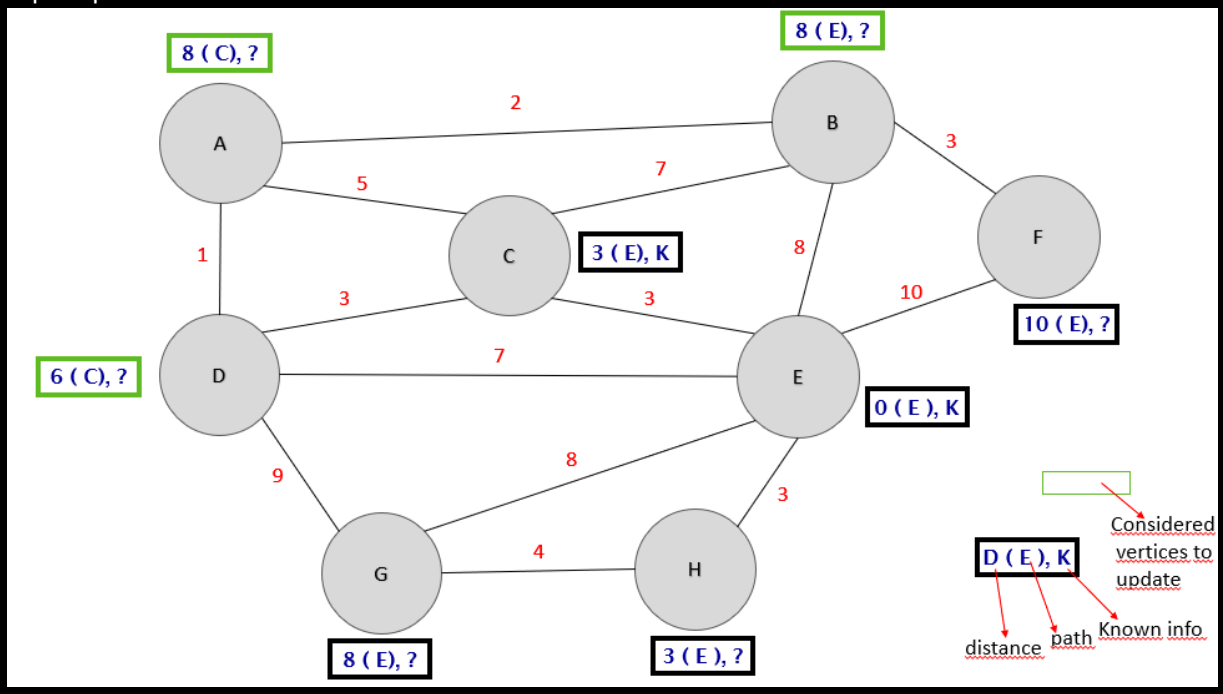


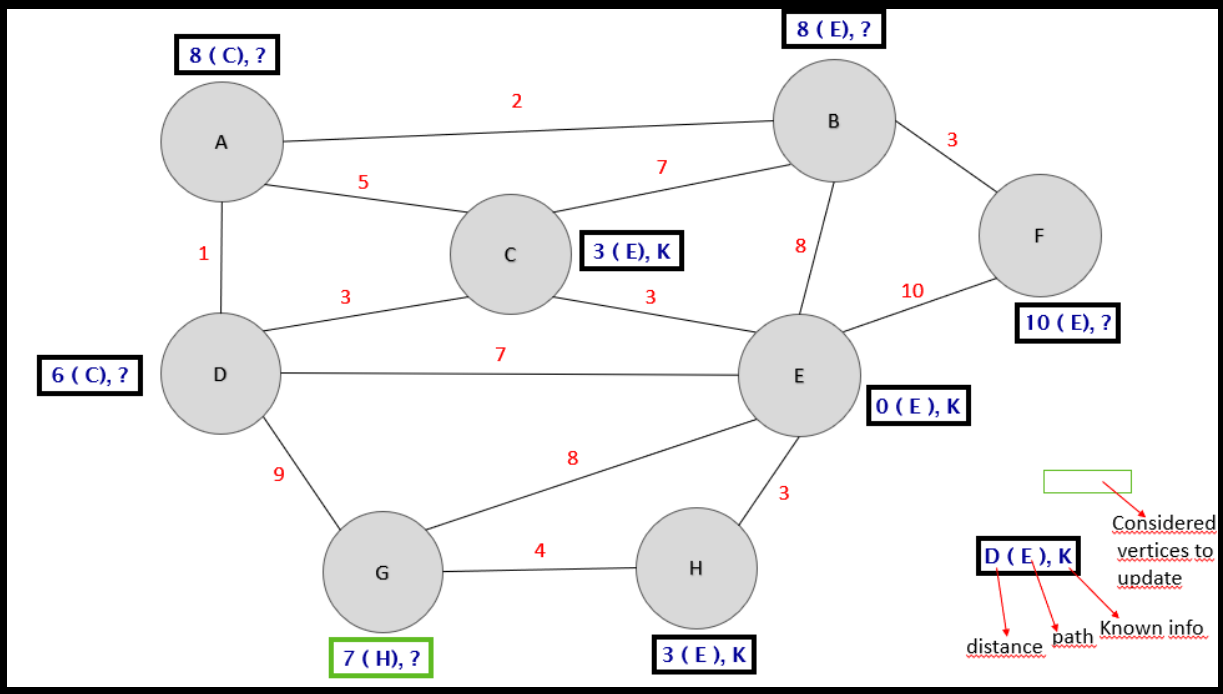
Start with vertex E, update adjacent vertices



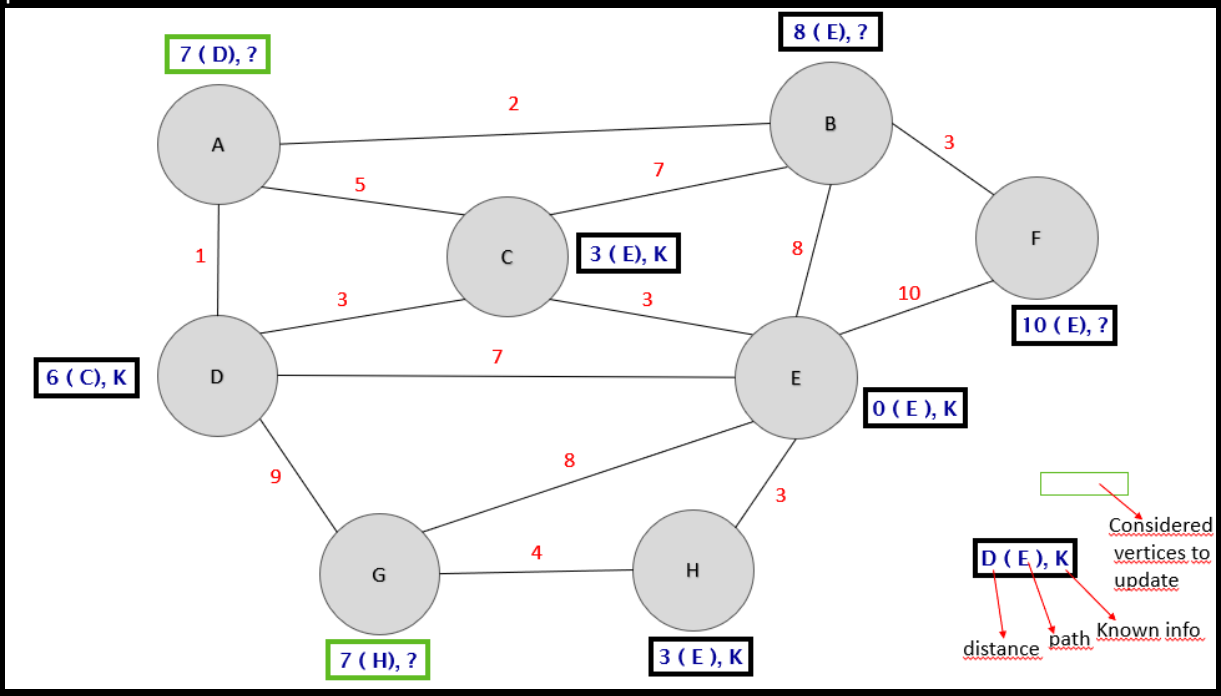


Among all, vertex C and H has the smallest distance so make it known, update adjacent vertices and repeat process

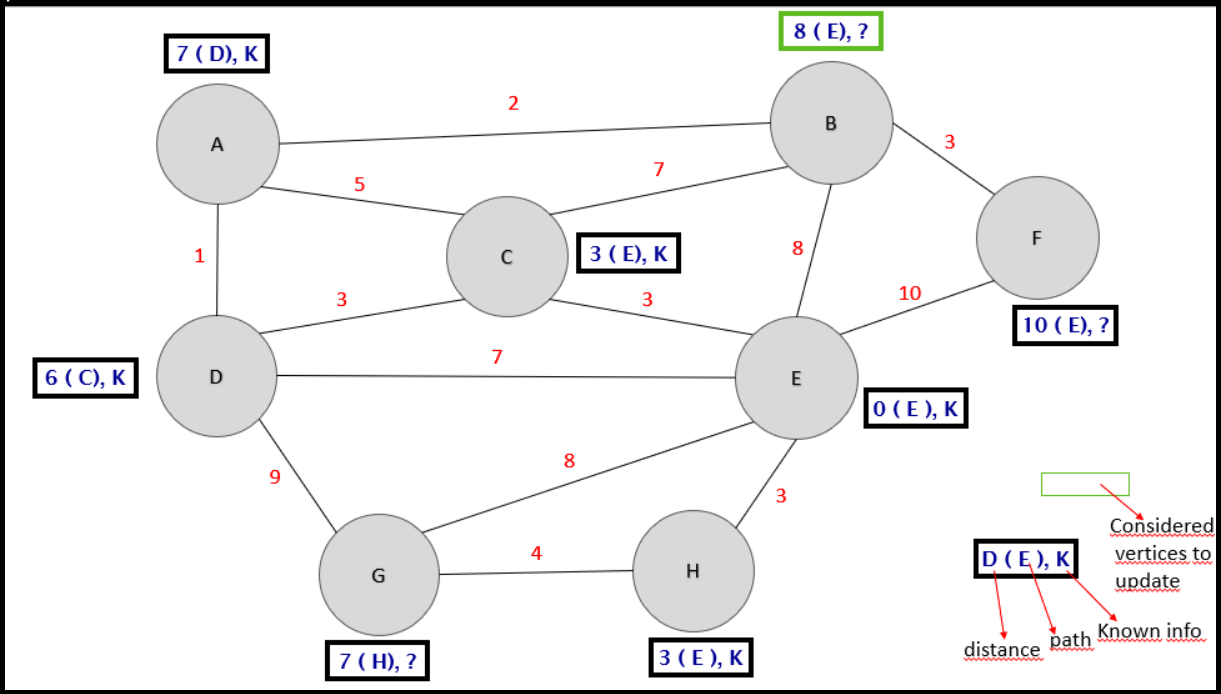




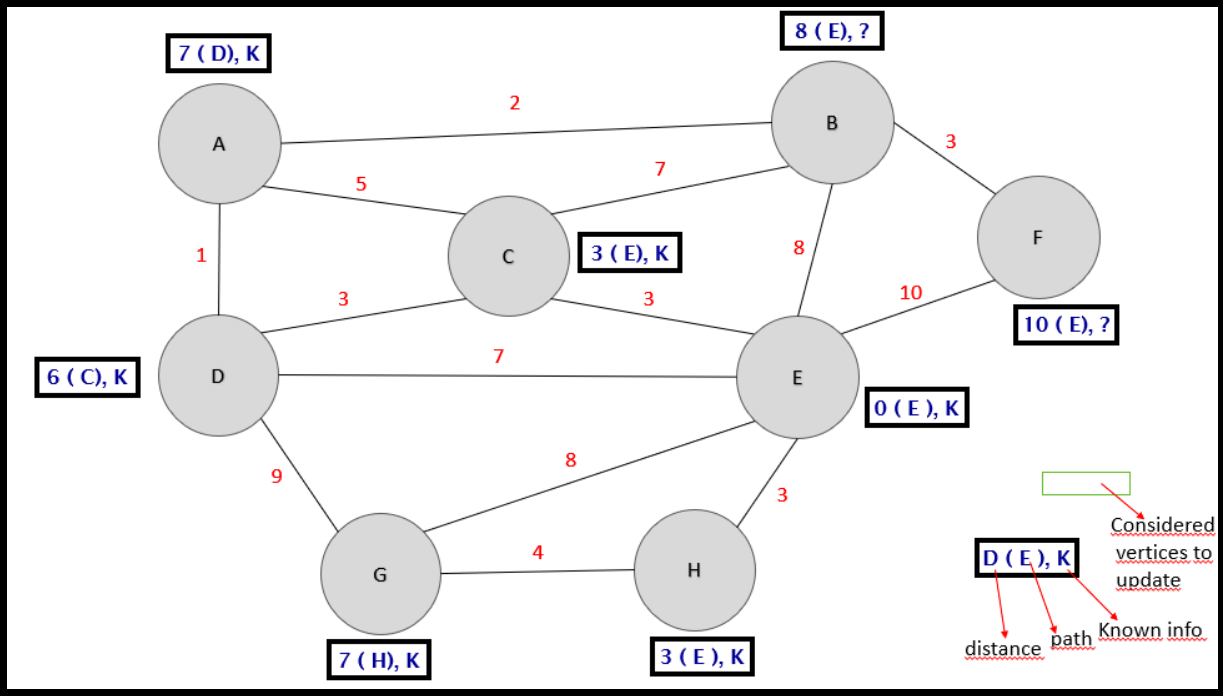
Among all, vertex H has the smallest distance so make it known, update adjacent vertices and repeat process



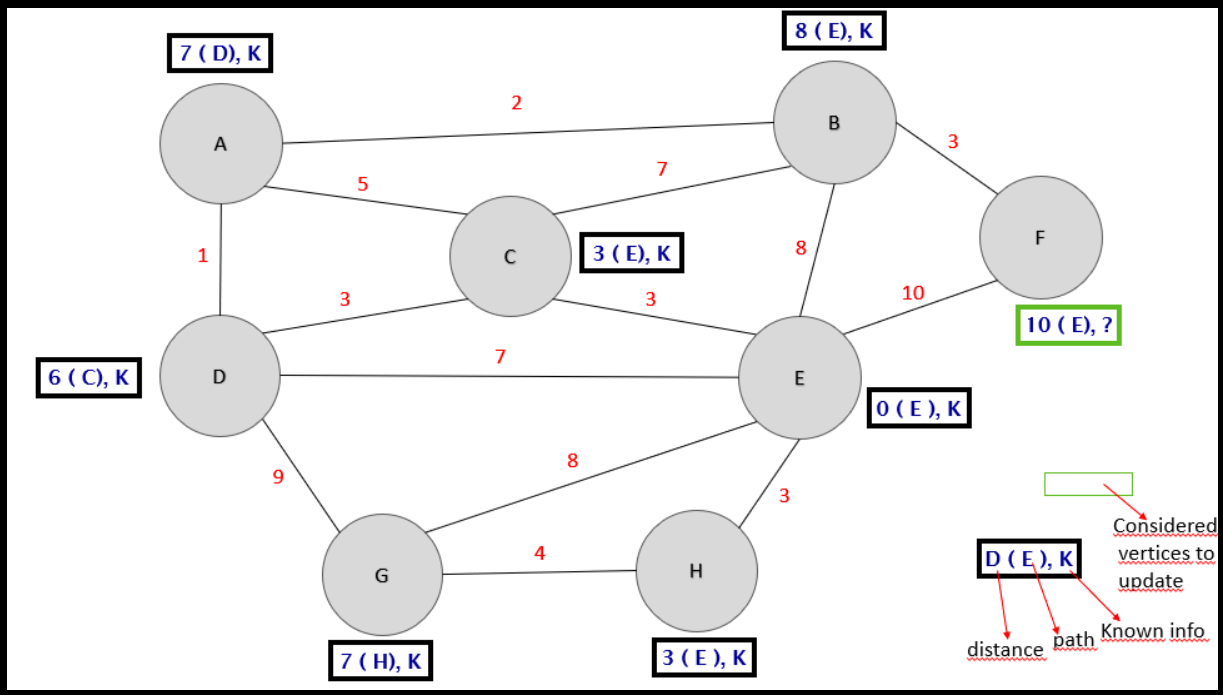
Among all, vertex D has the smallest distance so make it known, update adjacent vertices and repeat process



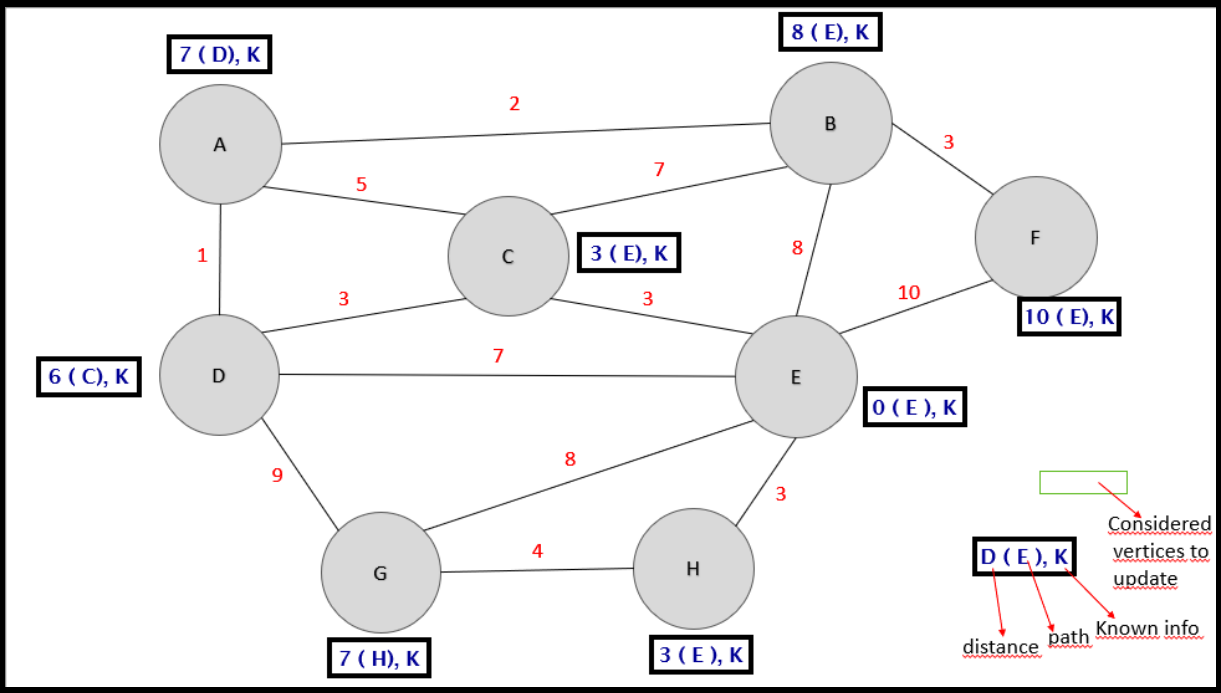
Among all, vertex A has the smallest distances so make it known, update adjacent vertices and repeat process



Among all, vertex G has the smallest distances so make it known, update adjacent vertices and repeat process



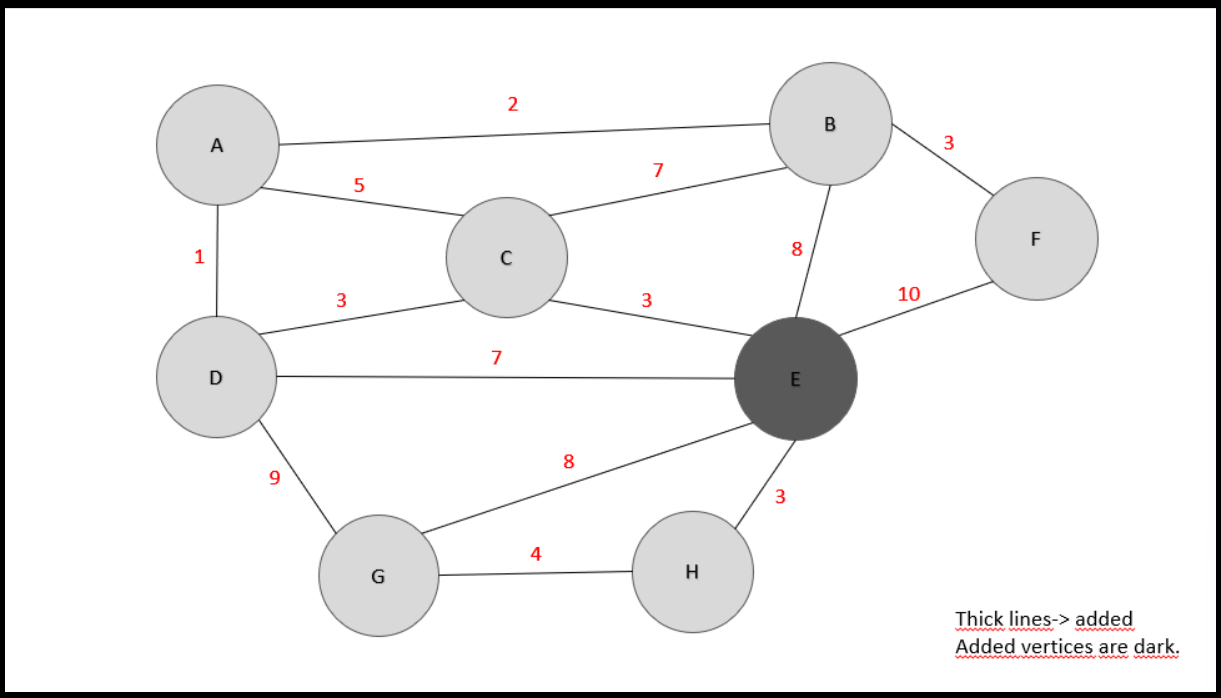
Among all, vertex B has the smallest distance so make it known, update adjacent vertices and repeat process



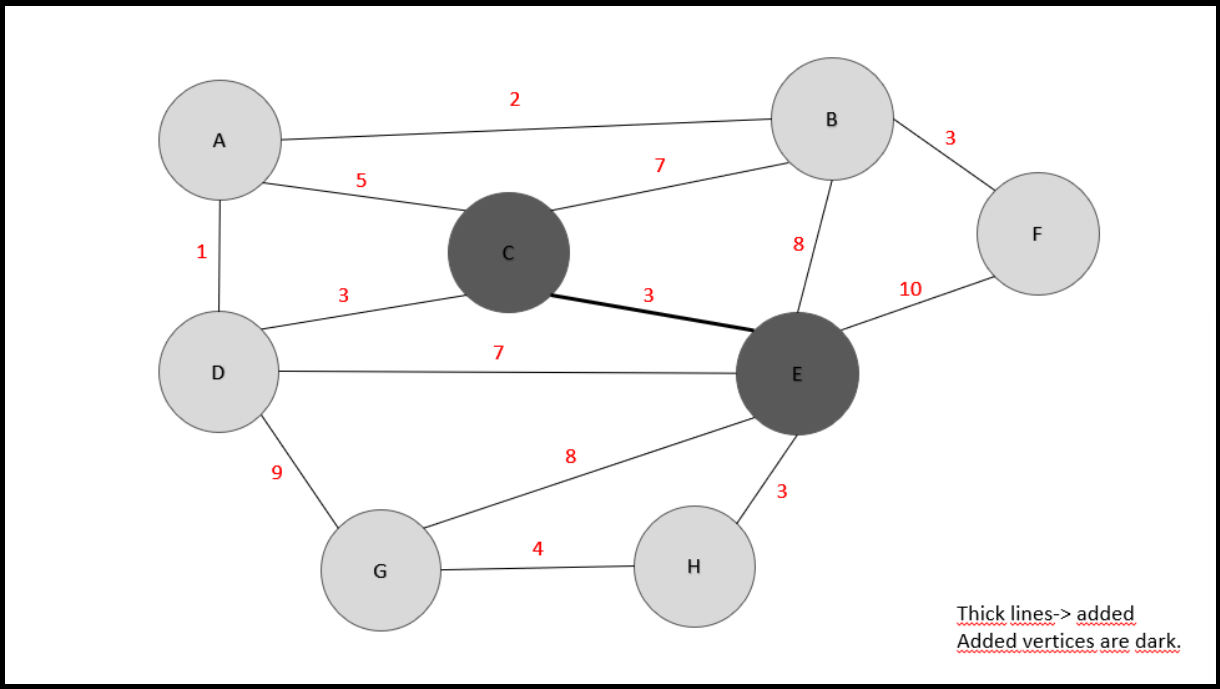
Among all, vertex F has the smallest distances so make it known, update adjacent vertices and repeat process (It ends after this iteration)

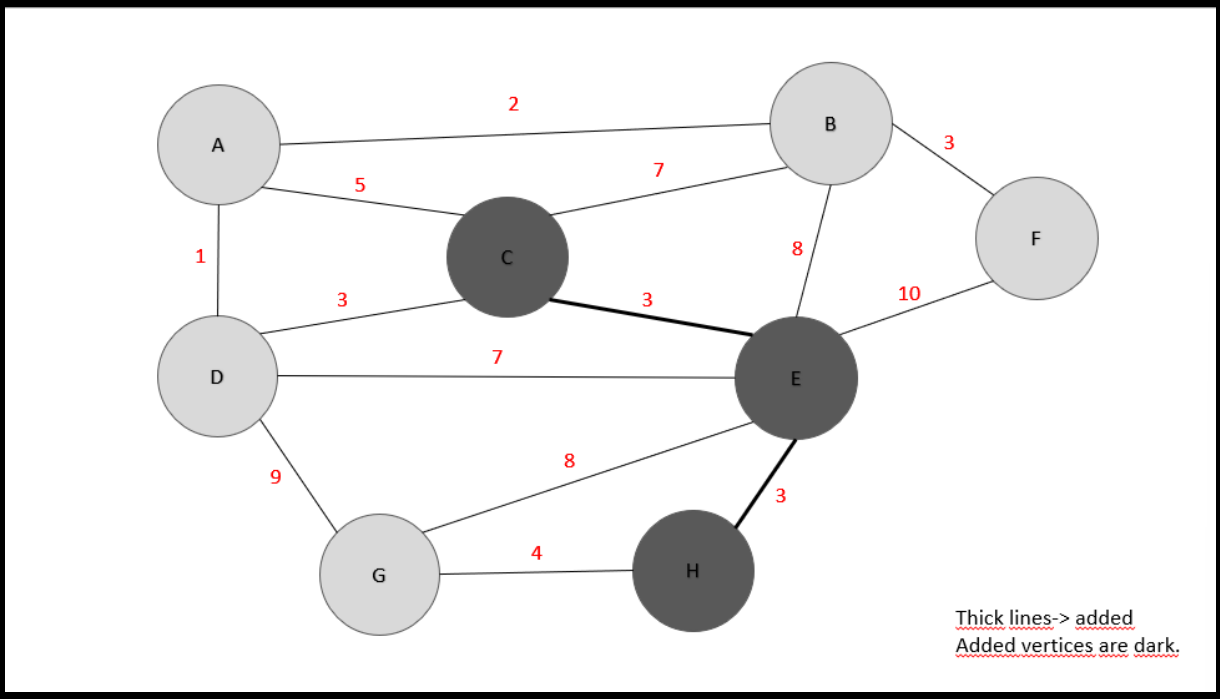
QUESTION 2

(Added = Known)

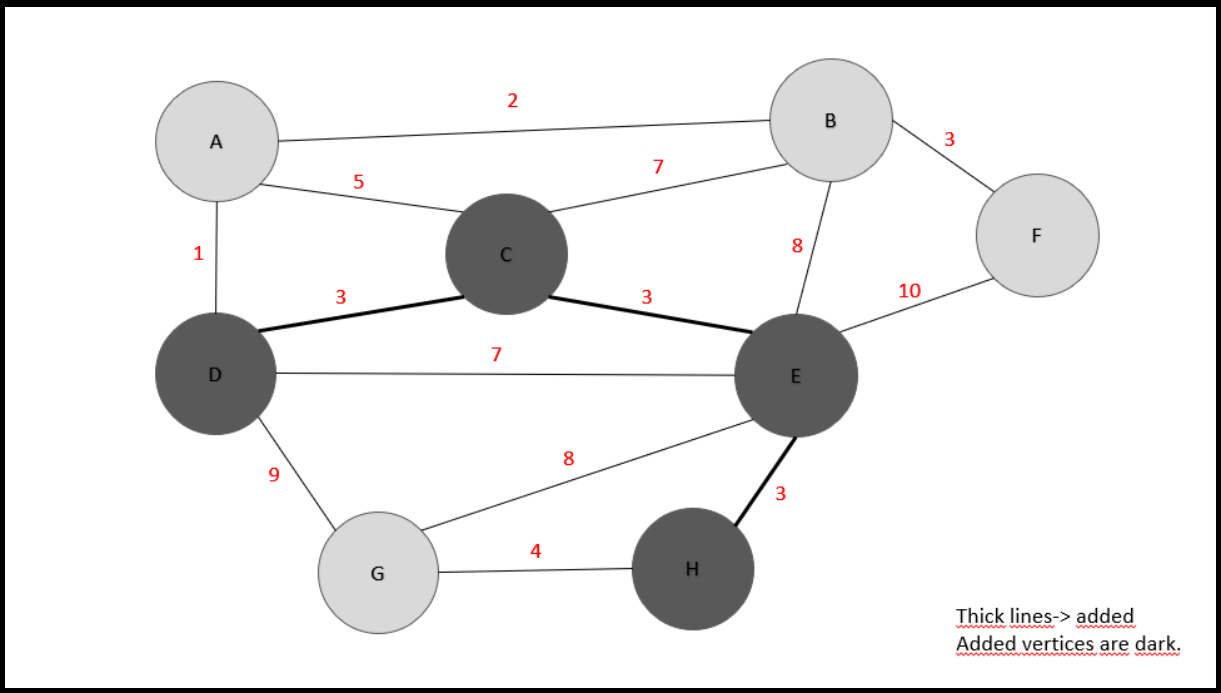


Starting by making vertex E known. The edge with lowest weight that connects an unknown and known vertex is the edge (C,E) so connect it and mark C as known.

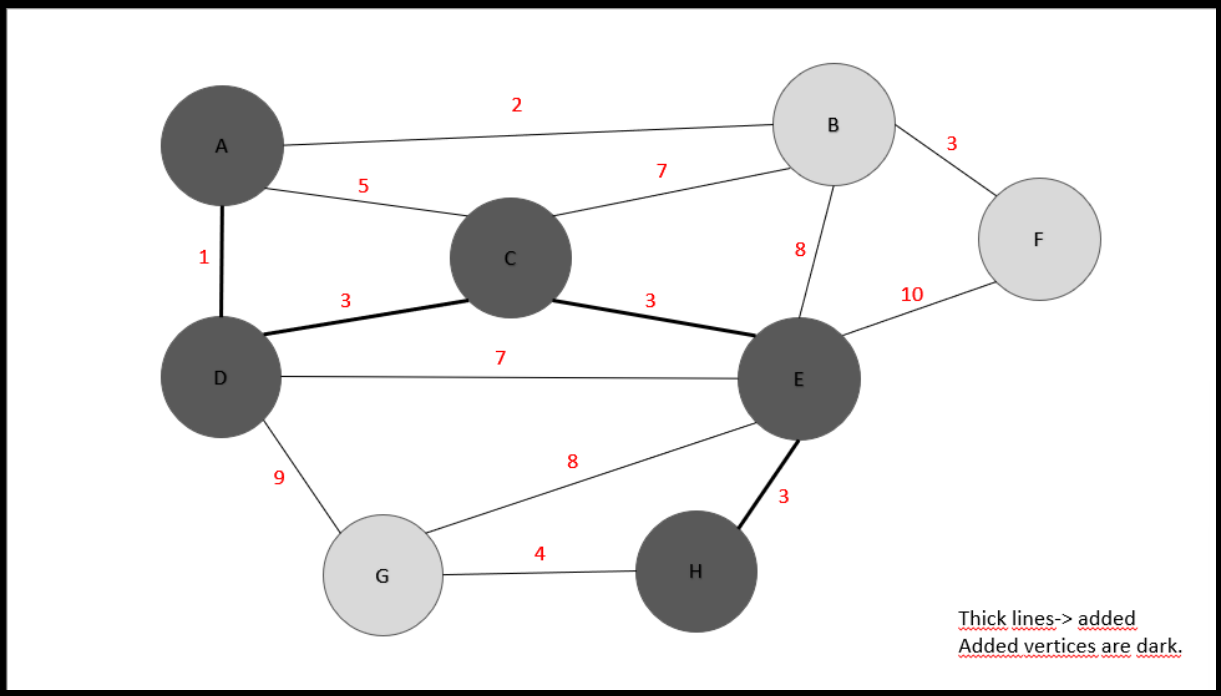




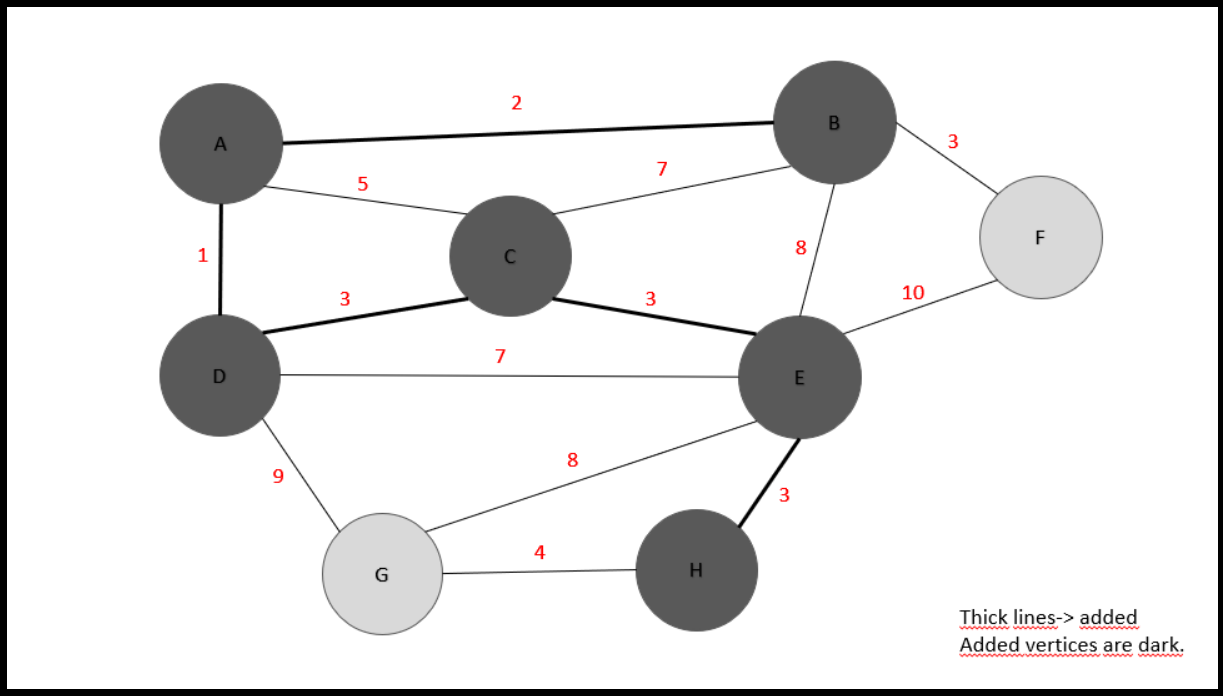
The edge with lowest weight that connects an unknown and known vertex is the edge (H,E) so connect it and mark H as known.



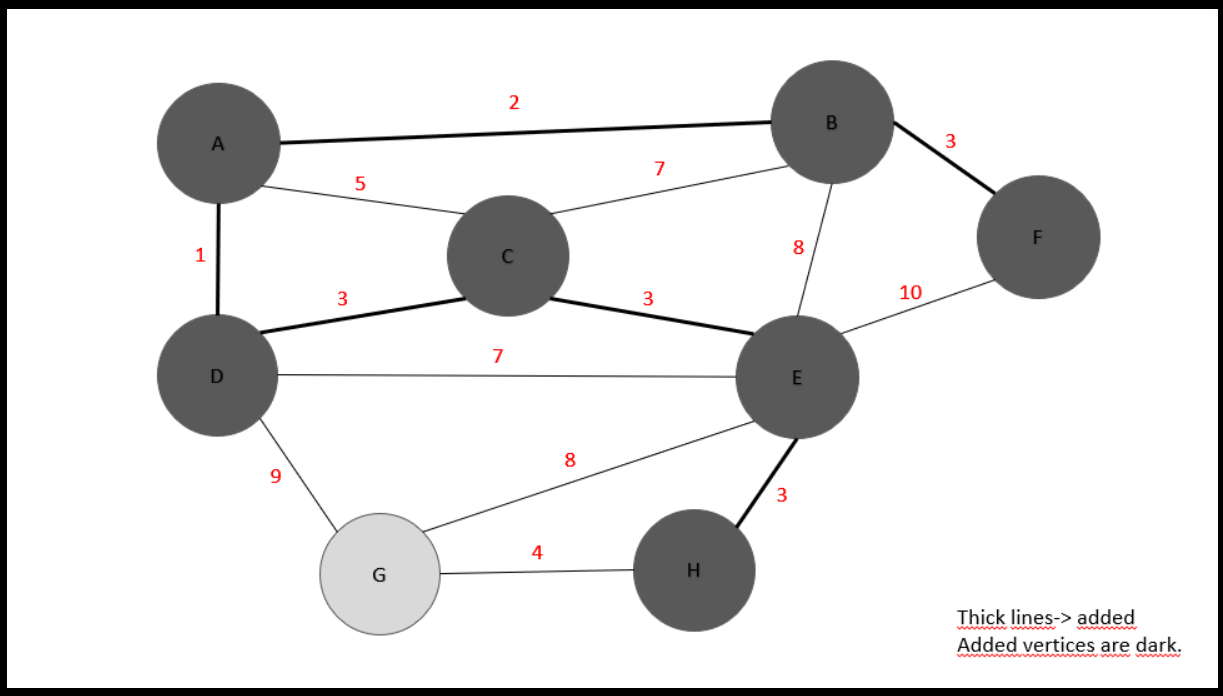
The edge with lowest weight that connects an unknown and known vertex is the edge (D,C) so connect it and mark D as known.



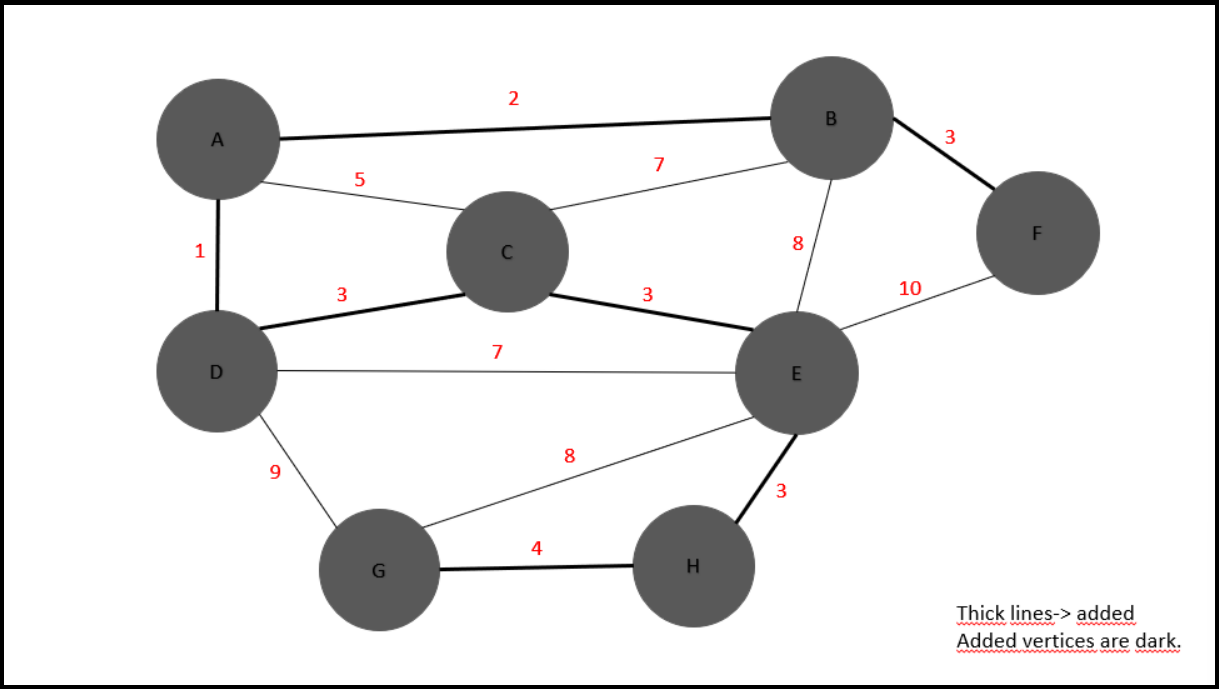
The edge with lowest weight that connects an unknown and known vertex is the edge (D,A) so connect it and mark A as known.



The edge with lowest weight that connects an unknown and known vertex is the edge (A,B) so connect it and mark B as known.



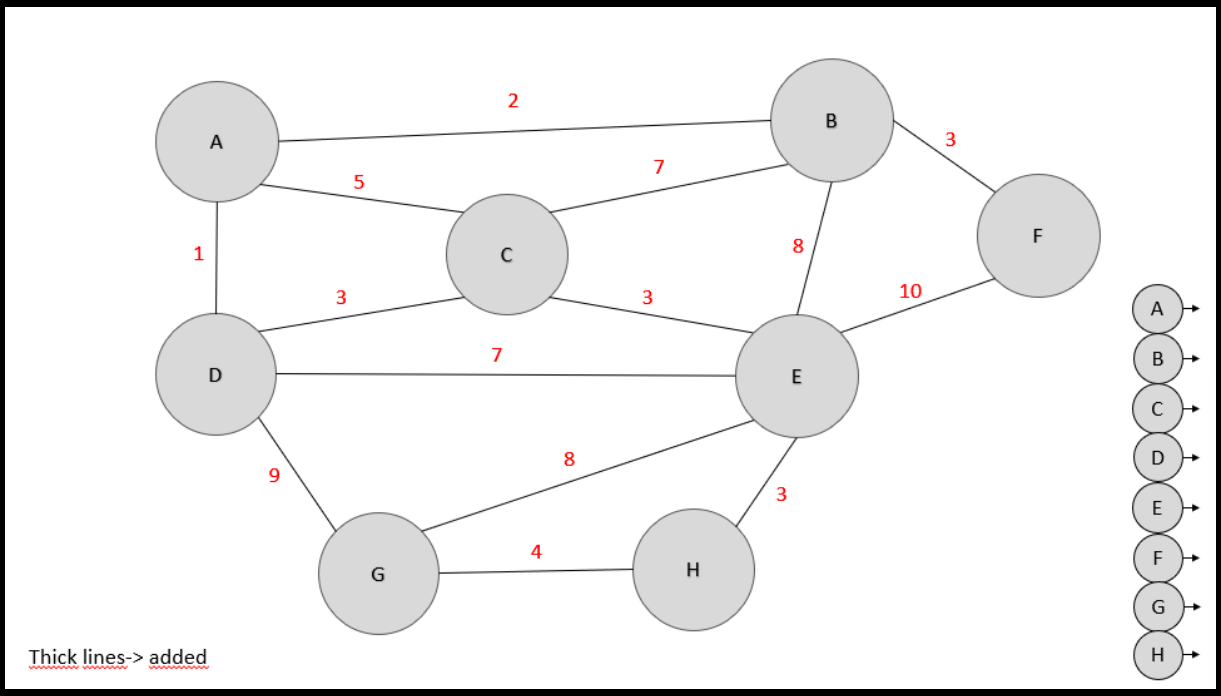
The edge with lowest weight that connects an unknown and known vertex is the edge (B,F) so connect it and mark F as known.



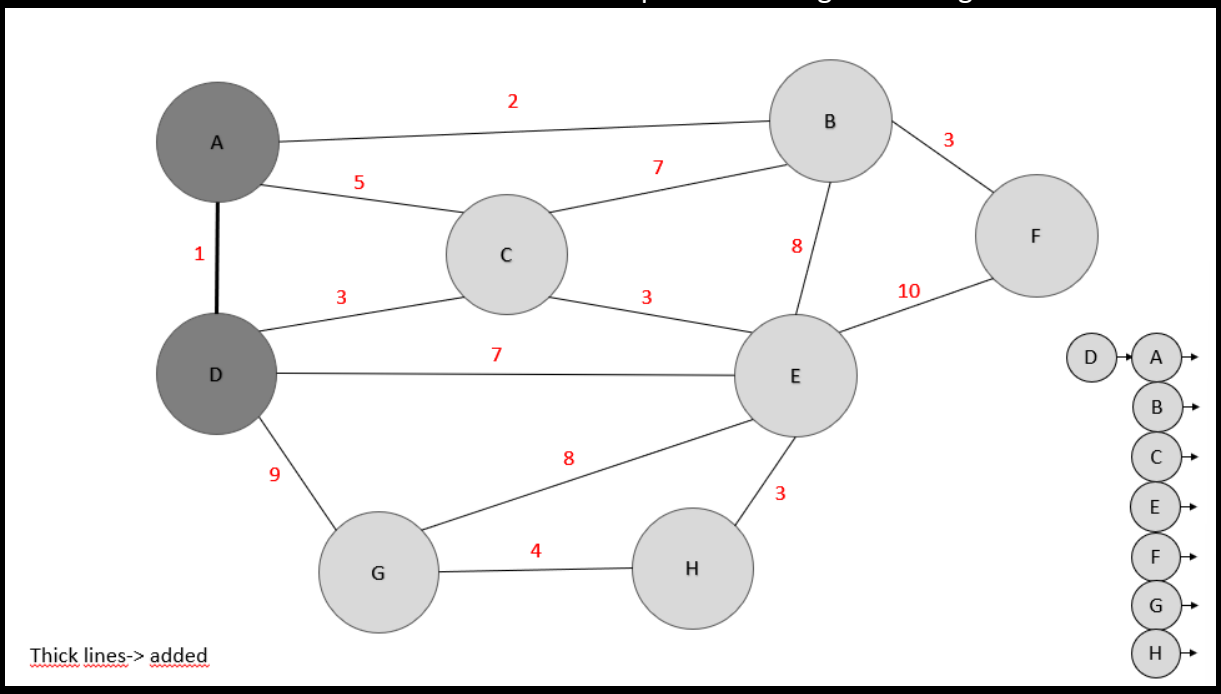
The edge with lowest weight that connects an unknown and known vertex is the edge (G,H) so connect it and mark G as known.

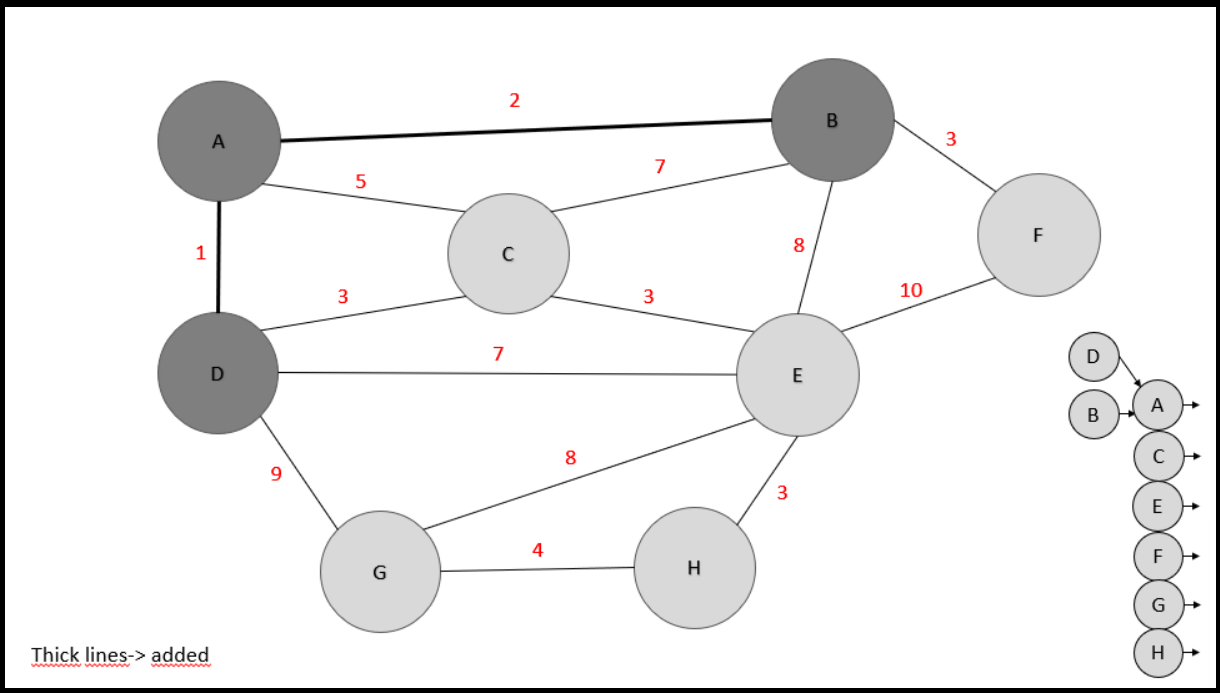
QUESTION 3

(Maintaining a forest on RHS)

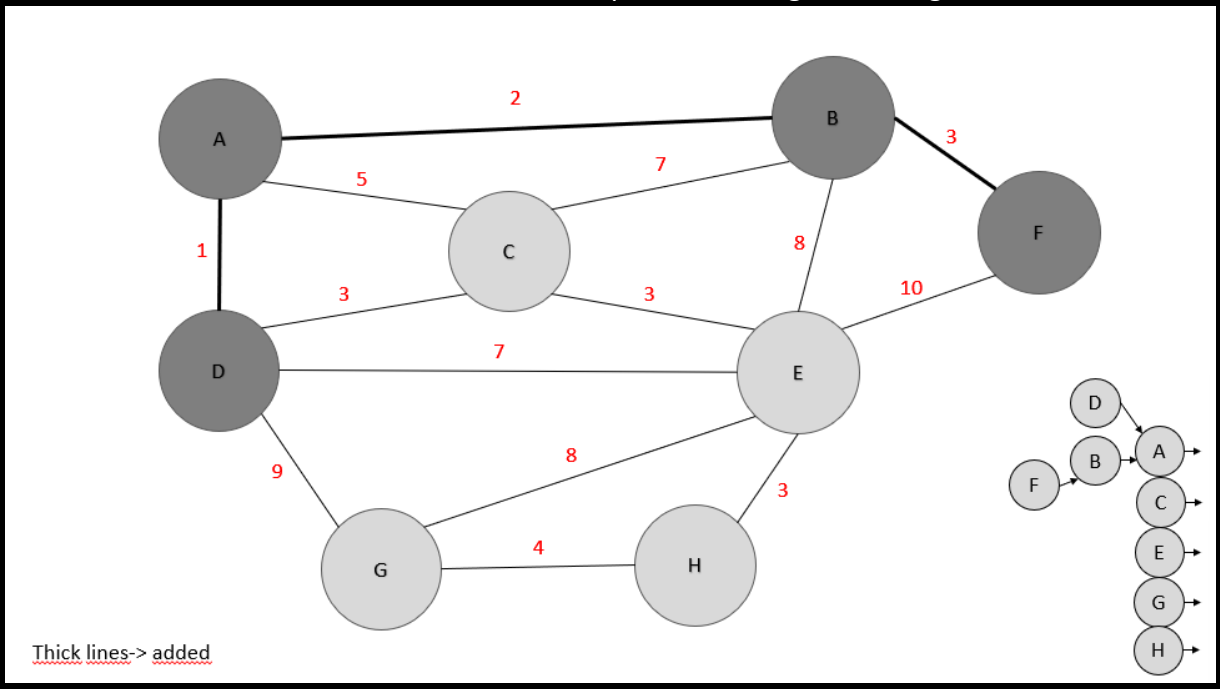


Edge with smallest weight that doesn't cause a cycle is (A,D). Union A and D on right. We also be sure that there is no loop when adding a new edge

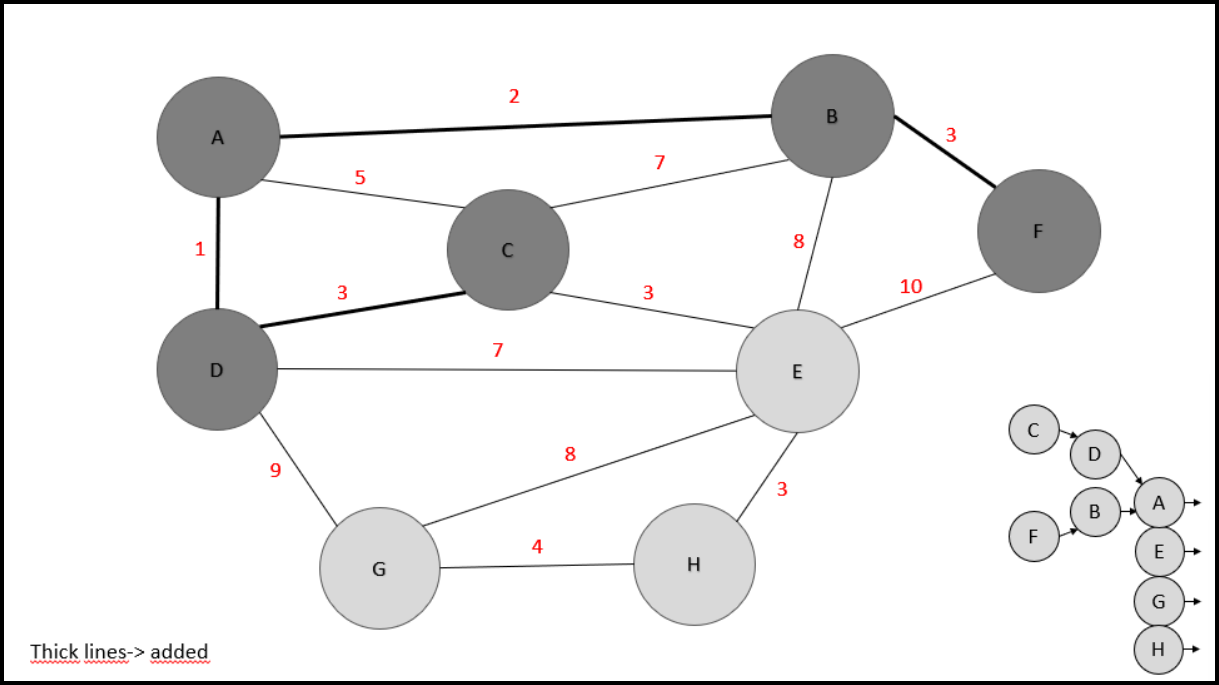




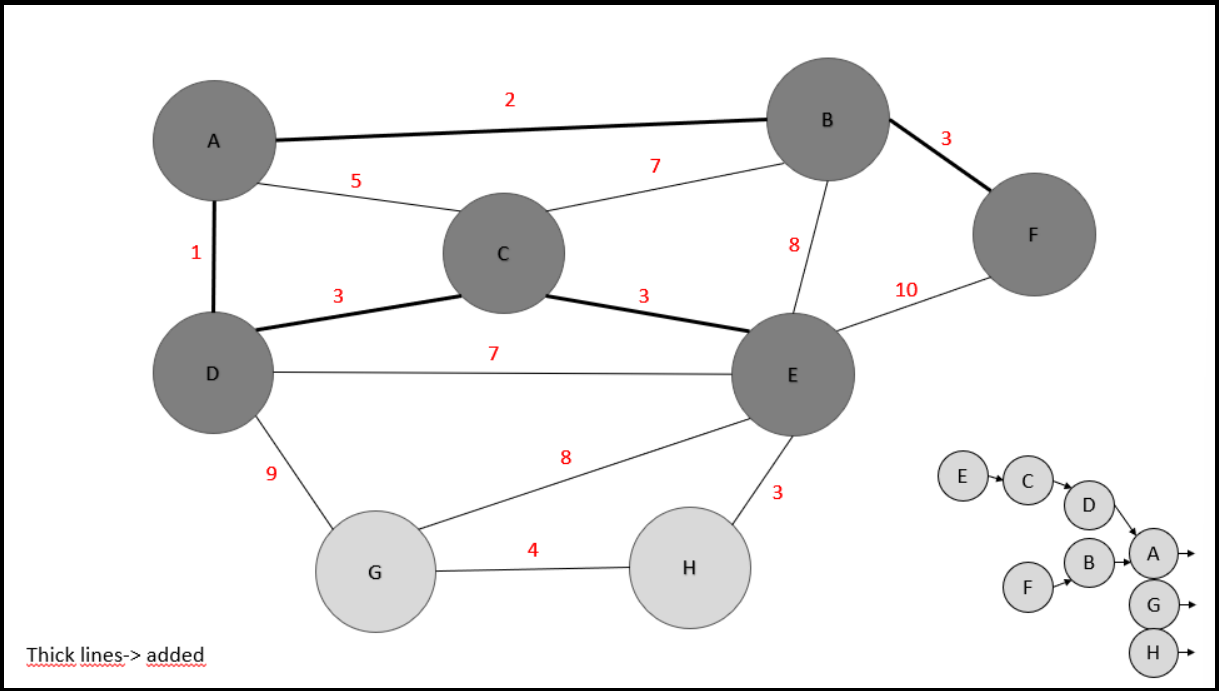
Edge with smallest weight that doesn't cause a cycle is (A,B). Union A and B on right. We also be sure that there is no loop when adding a new edge



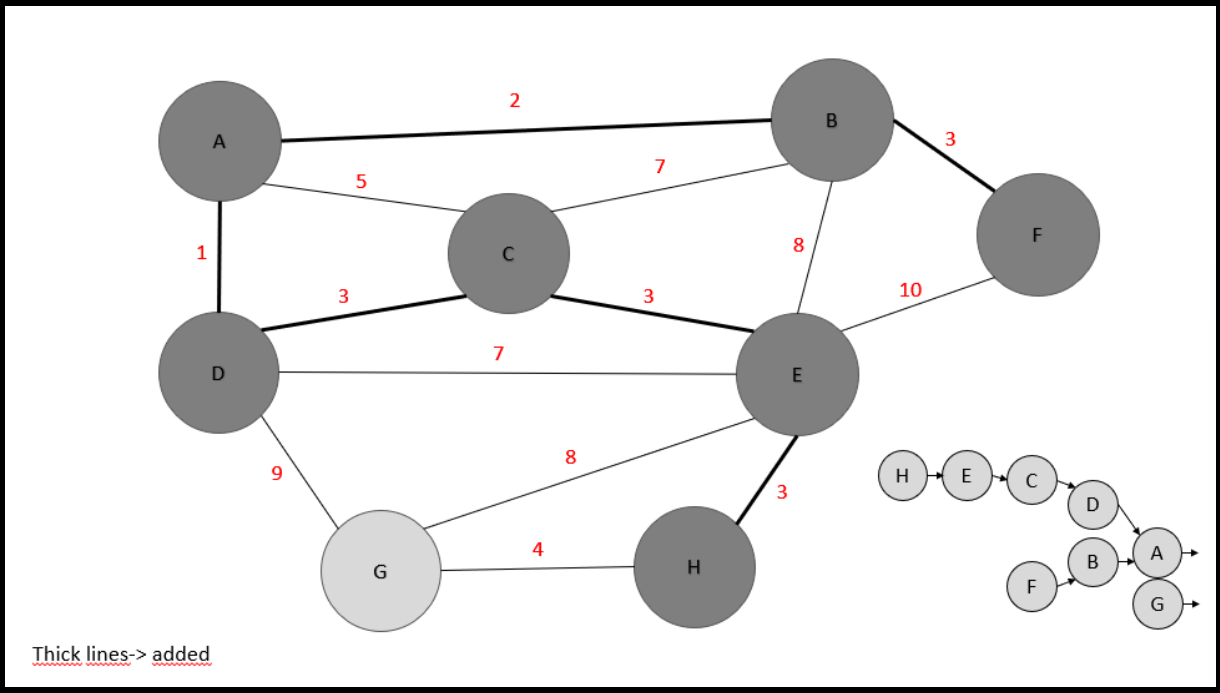
Edge with smallest weight that doesn't cause a cycle is (F,B). Union F and B on right. We also be sure that there is no loop when adding a new edge



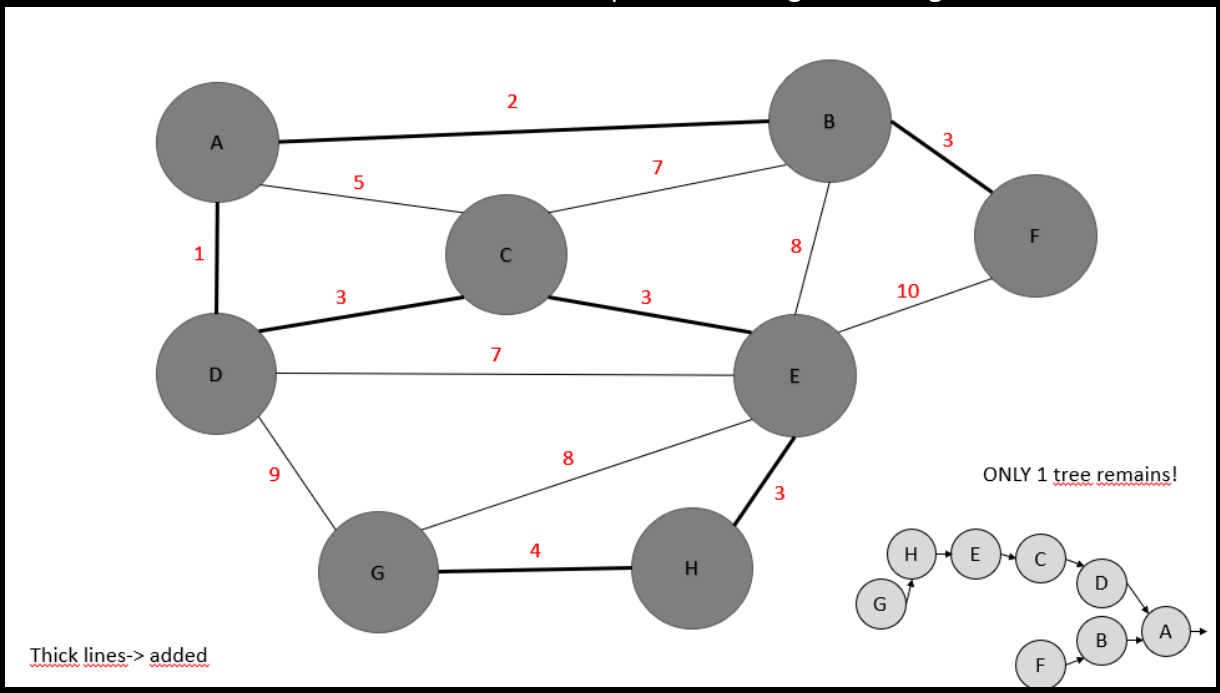
Edge with smallest weight that doesn't cause a cycle is (D,C). Union D and C on right. We also be sure that there is no loop when adding a new edge



Edge with smallest weight that doesn't cause a cycle is (C,E). Union C and E on right. We also be sure that there is no loop when adding a new edge

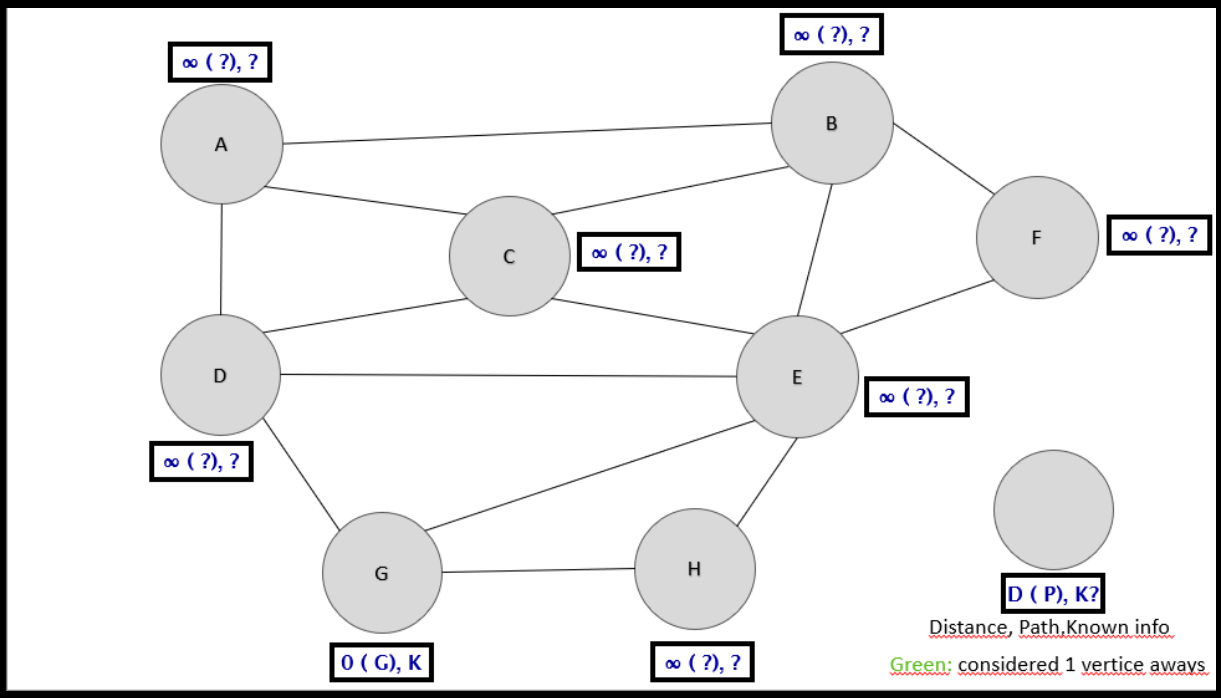


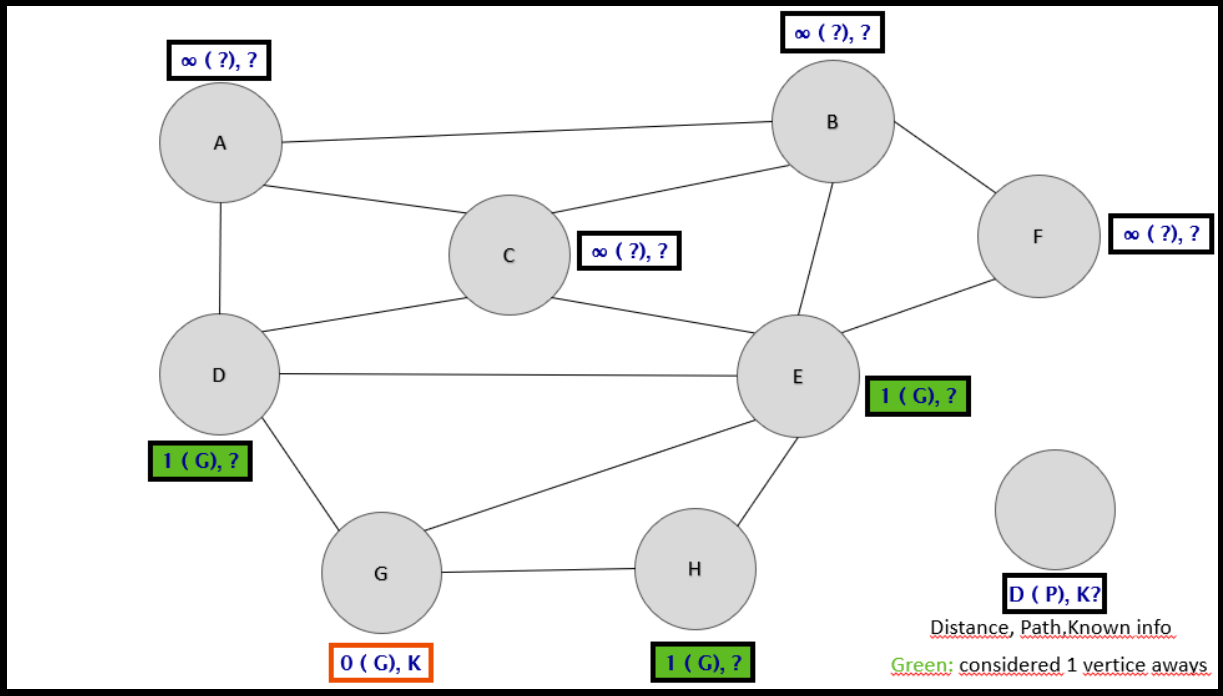
Edge with smallest weight that doesn't cause a cycle is (E,H). Union E and H on right. We also be sure that there is no loop when adding a new edge



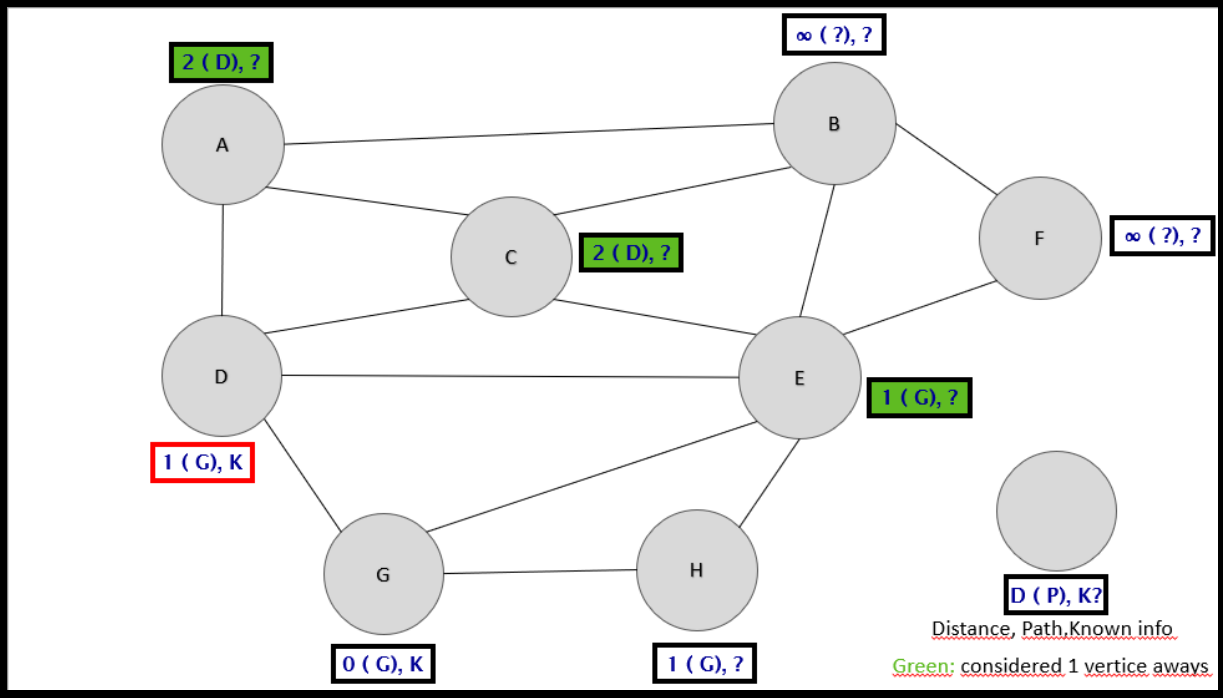
Edge with smallest weight that doesn't cause a cycle is (G,H). Union G and H on right. We also be sure that there is no loop when adding a new edge. We finish since only 1 tree remains in our forest.

QUESTION 4

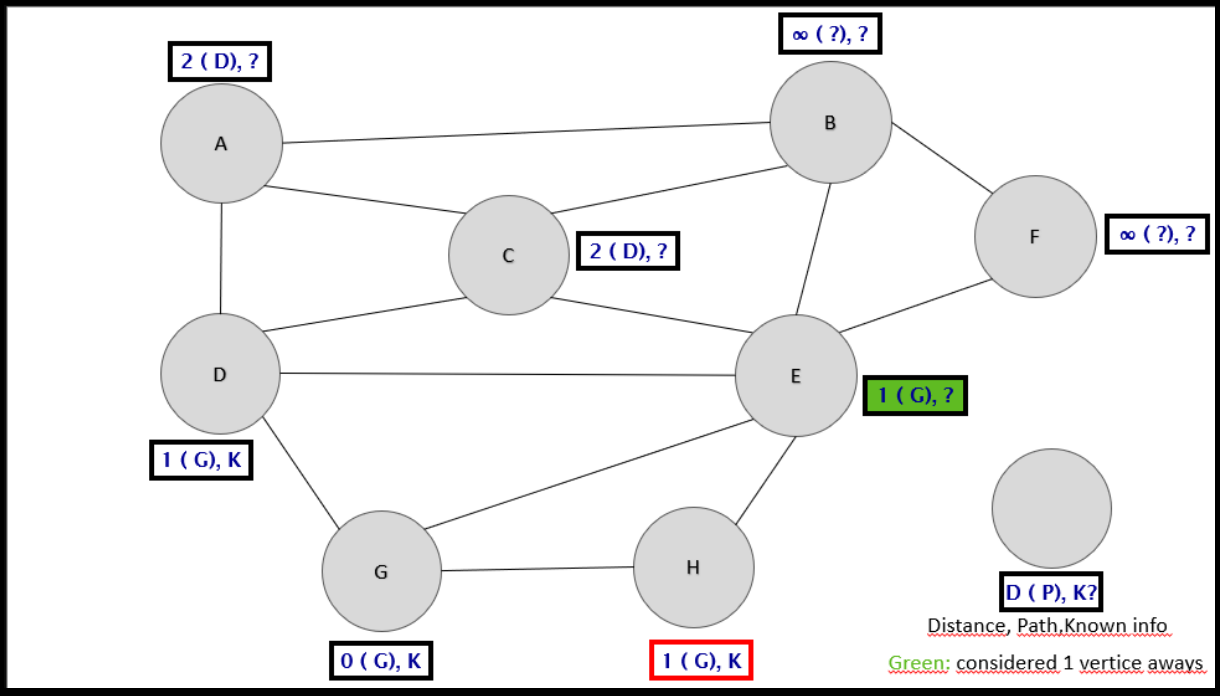




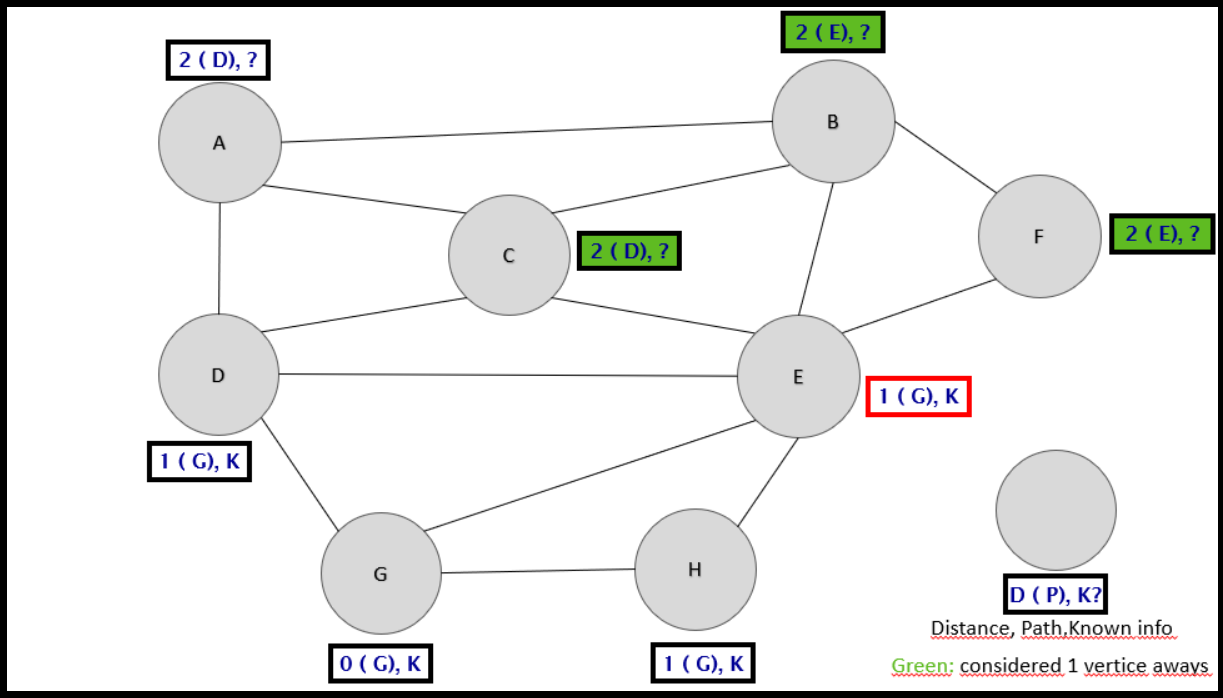
At start, mark G as known and set all vertices adjacent to its distances as 1.



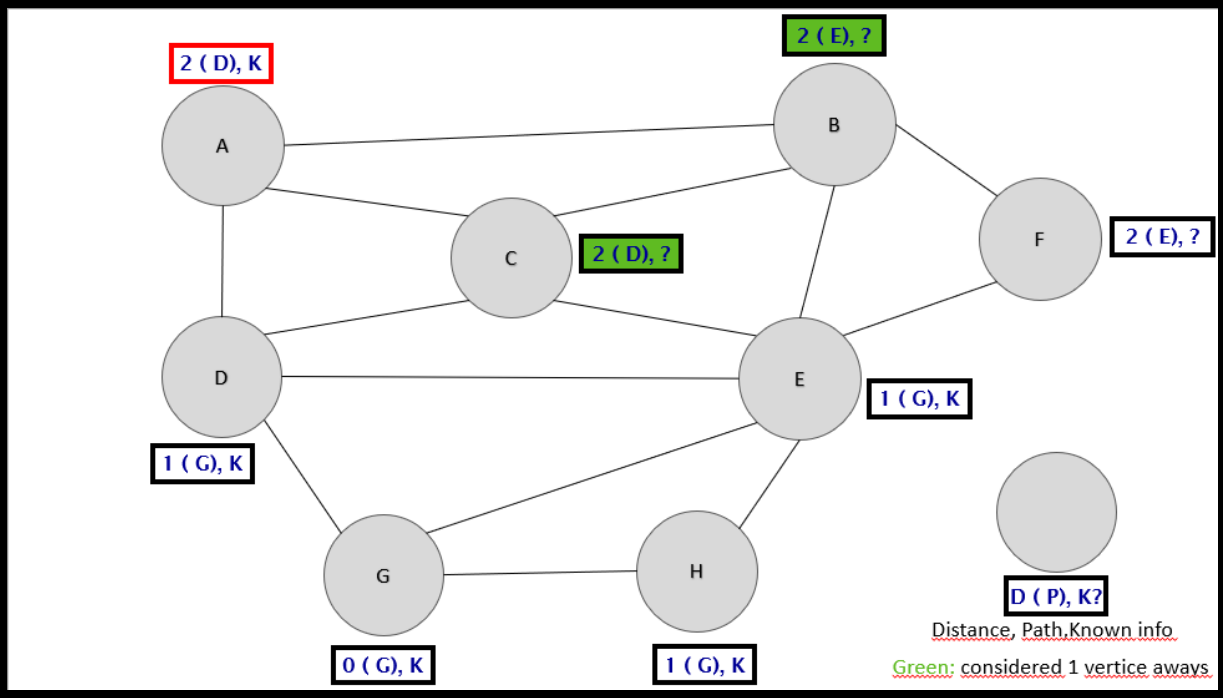
Mark D as known and set all unknown vertices adjacent to its distances as 2 if it is lesser than their current distance.



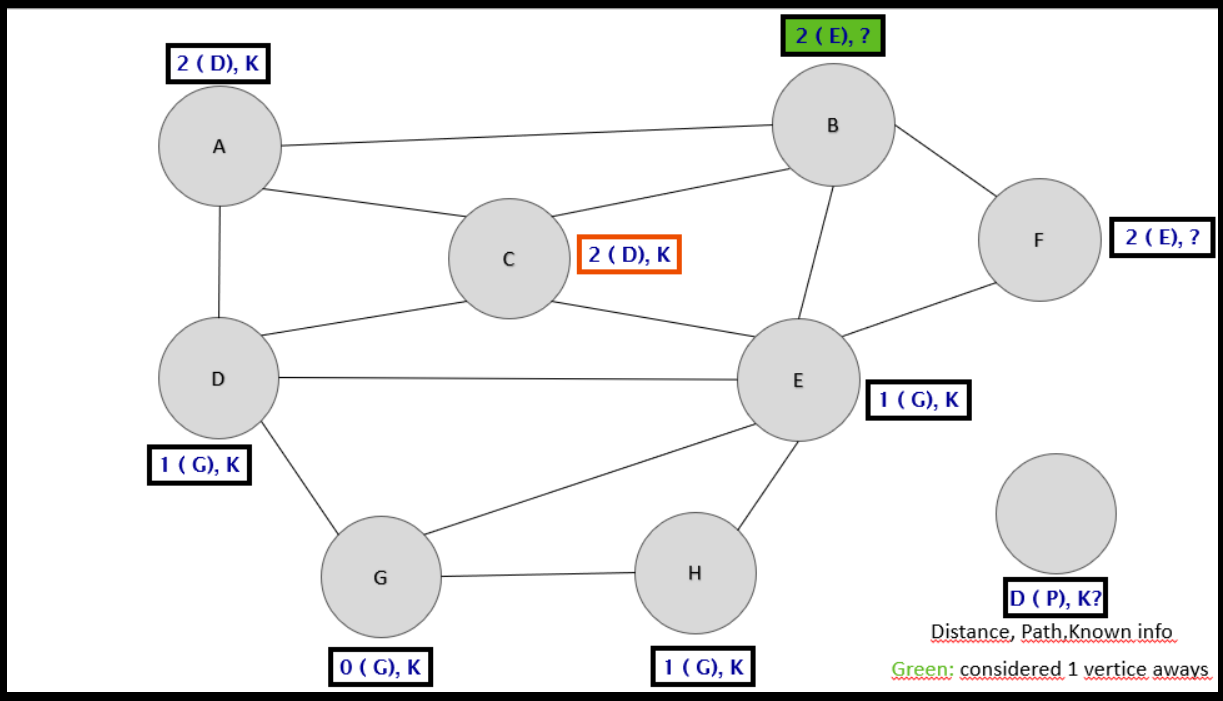
Mark H as known and set all unknown vertices adjacent to its distances as 2 if it is lesser than their current distance.



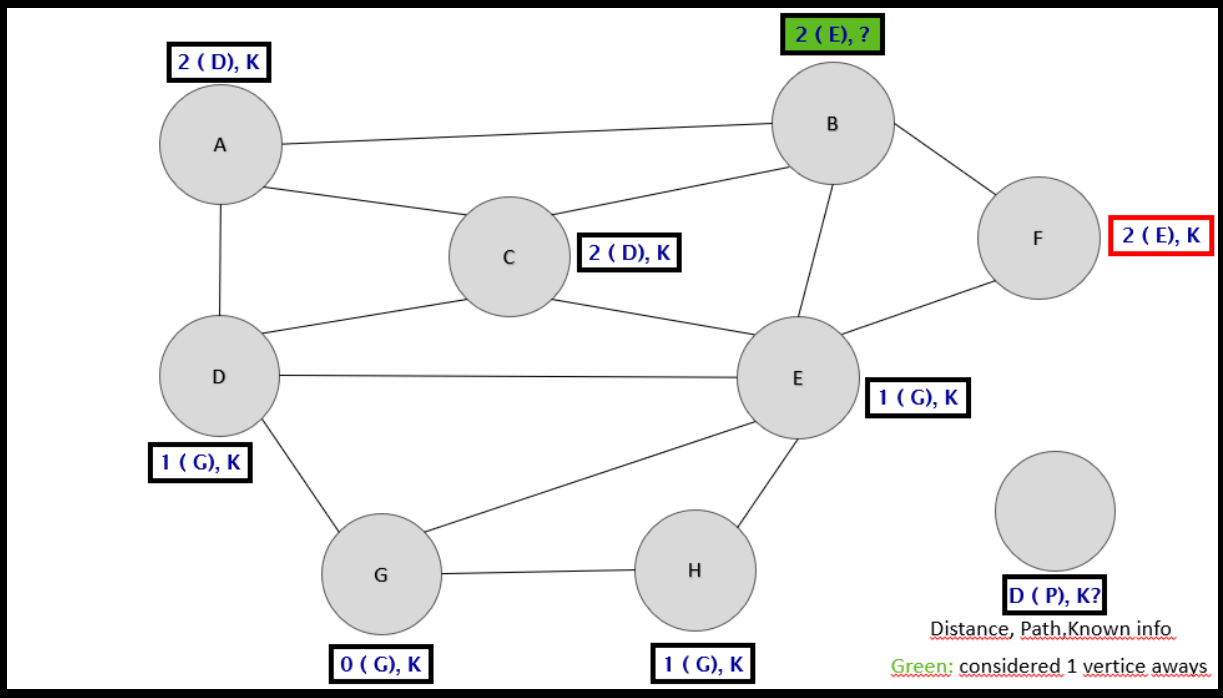
Mark E as known and set all unknown vertices adjacent to its distances as 2 if it is lesser than their current distance.



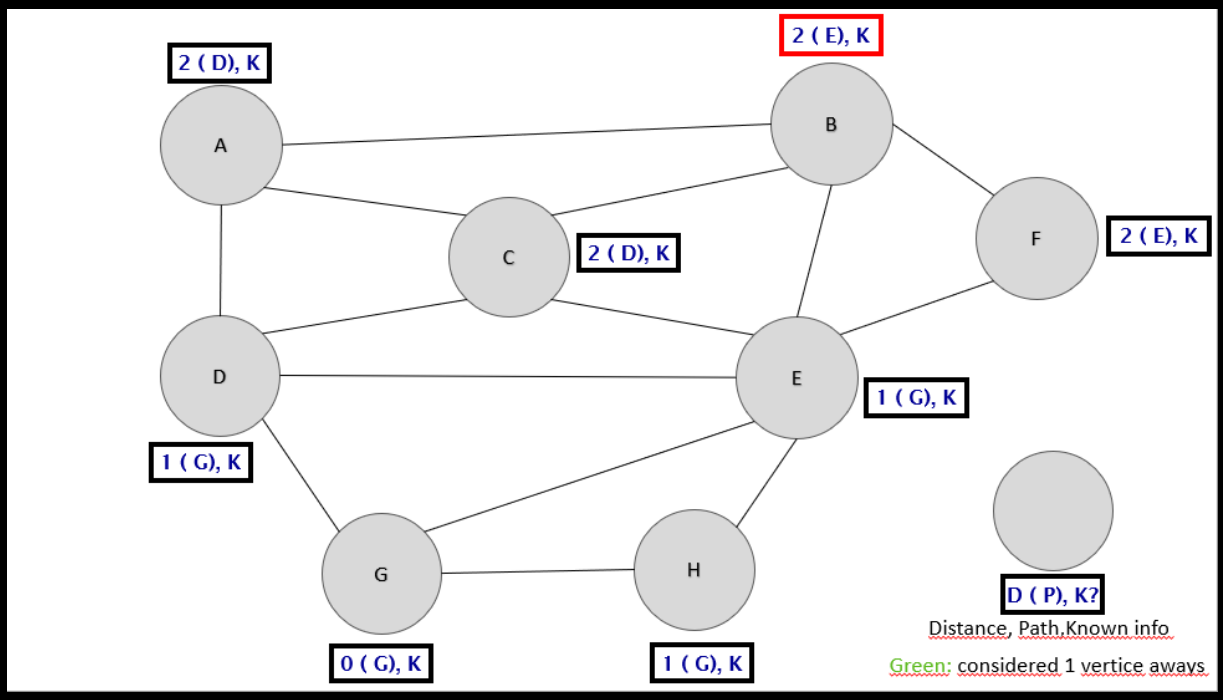
Mark A as known and set all unknown vertices adjacent to its distances as 3 if it is lesser than their current distance.



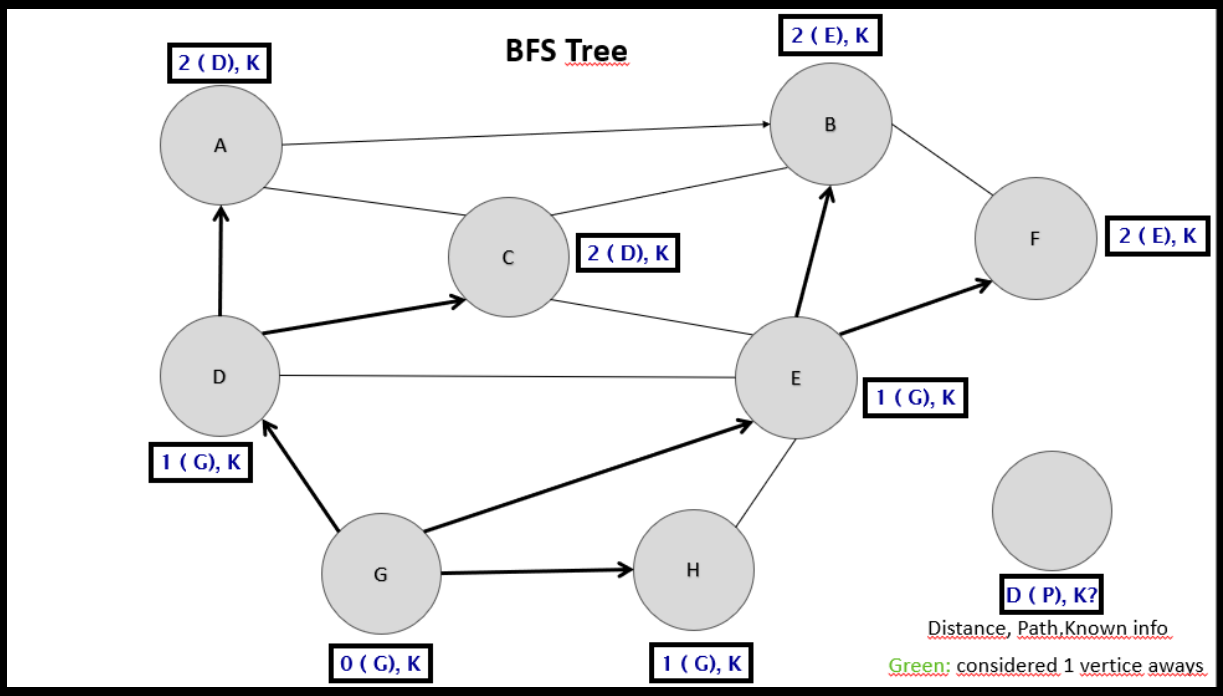
Mark C as known and set all unknown vertices adjacent to its distances as 3 if it is lesser than their current distance.



Mark F as known and set all unknown vertices adjacent to its distances as 3 if it is lesser than their current distance.



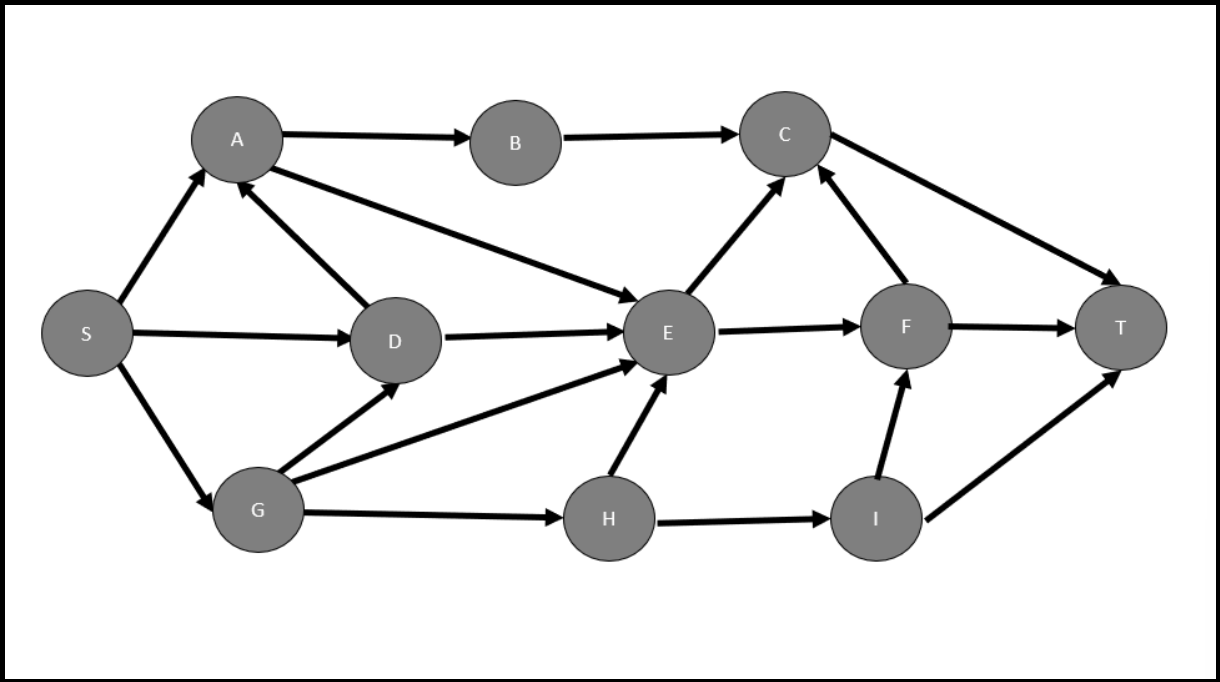
Mark B as known and set all unknown vertices adjacent to its distances as 3 if it is lesser than their current distance.

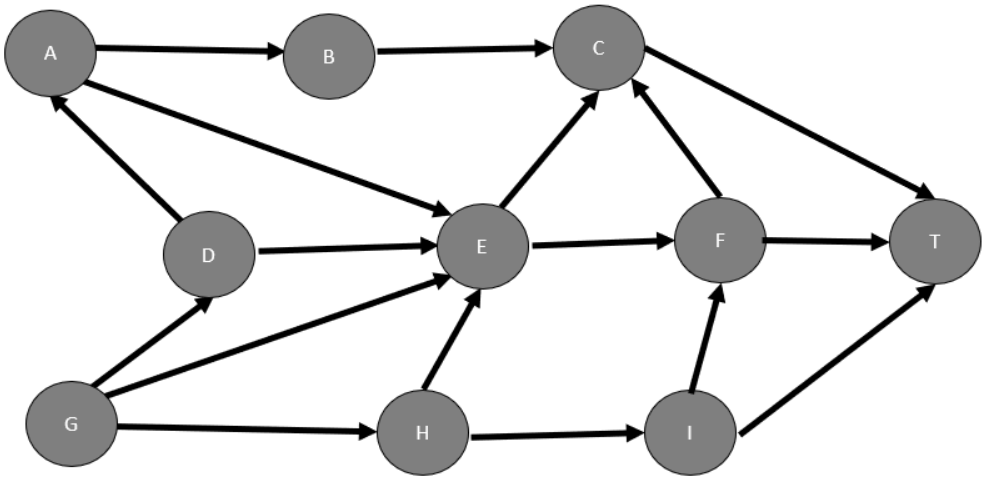


Here is the BFS tree structure. I made this to clearly show the paths in a visual way.

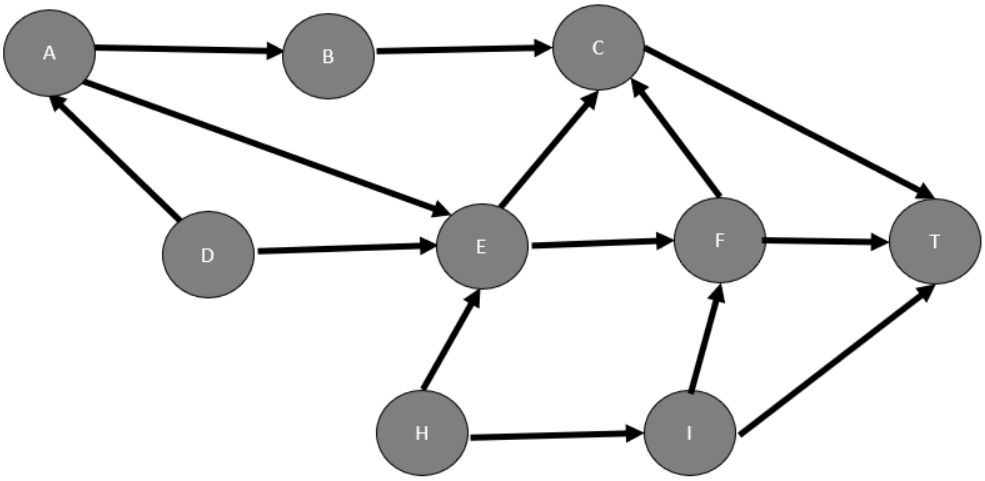
QUESTION 5

Repeatedly select a vertex with indegree 0 (If more than 1, then pick random.) Remove the vertex and print it.

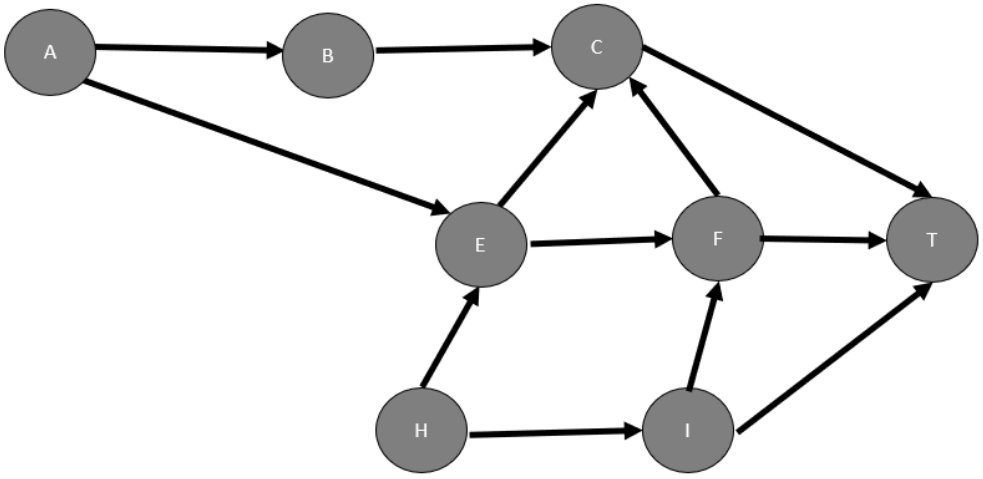




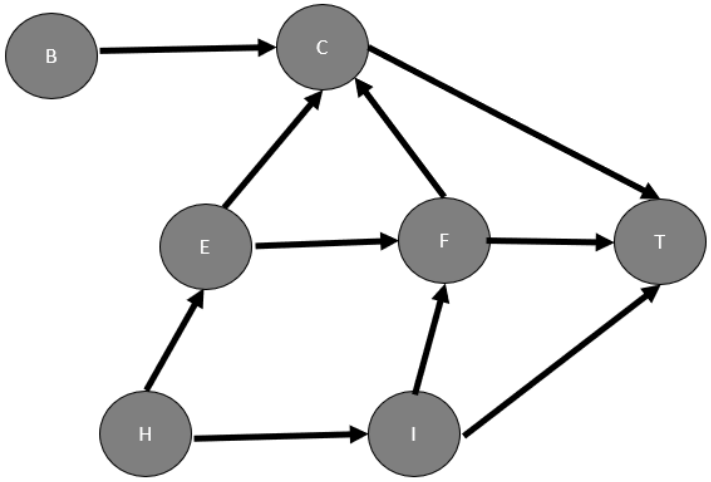
S



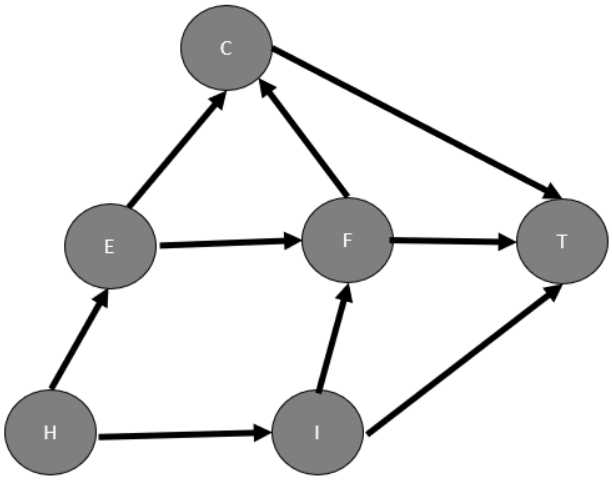
S,G



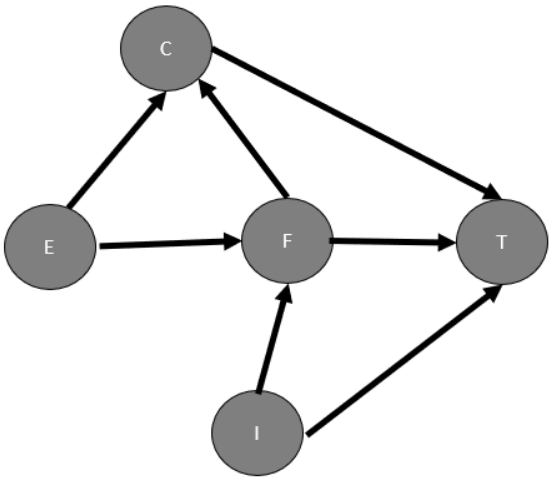
S,G,D



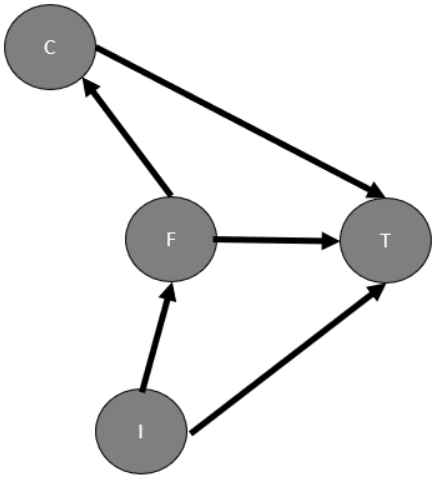
S,G,D,A



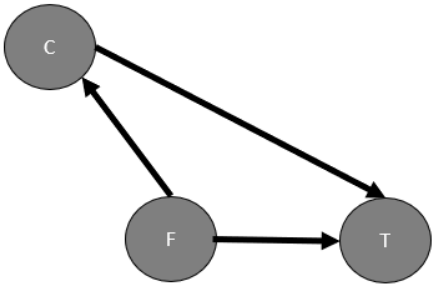
S,G,D,A,B



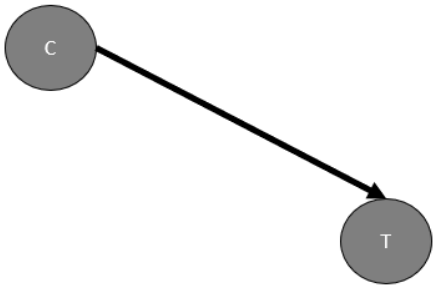
S,G,D,A,B,H



S,G,D,A,B,H,E



S,G,D,A,B,H,E,I



S,G,D,A,B,H,E,I,F

S,G,D,A,B,H,E,I,F,C

S,G,D,A,B,H,E,I,F,C,T