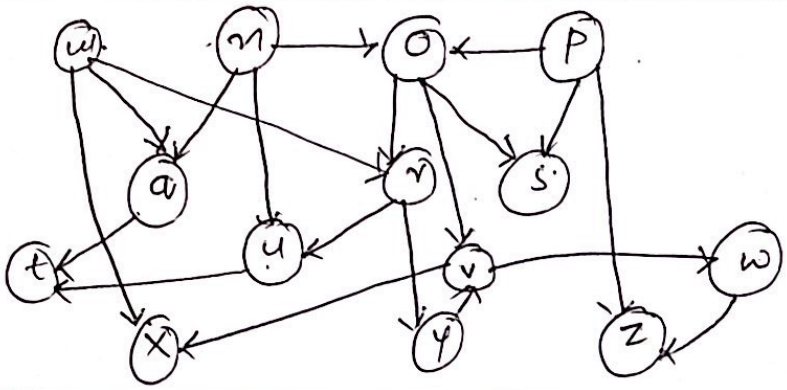
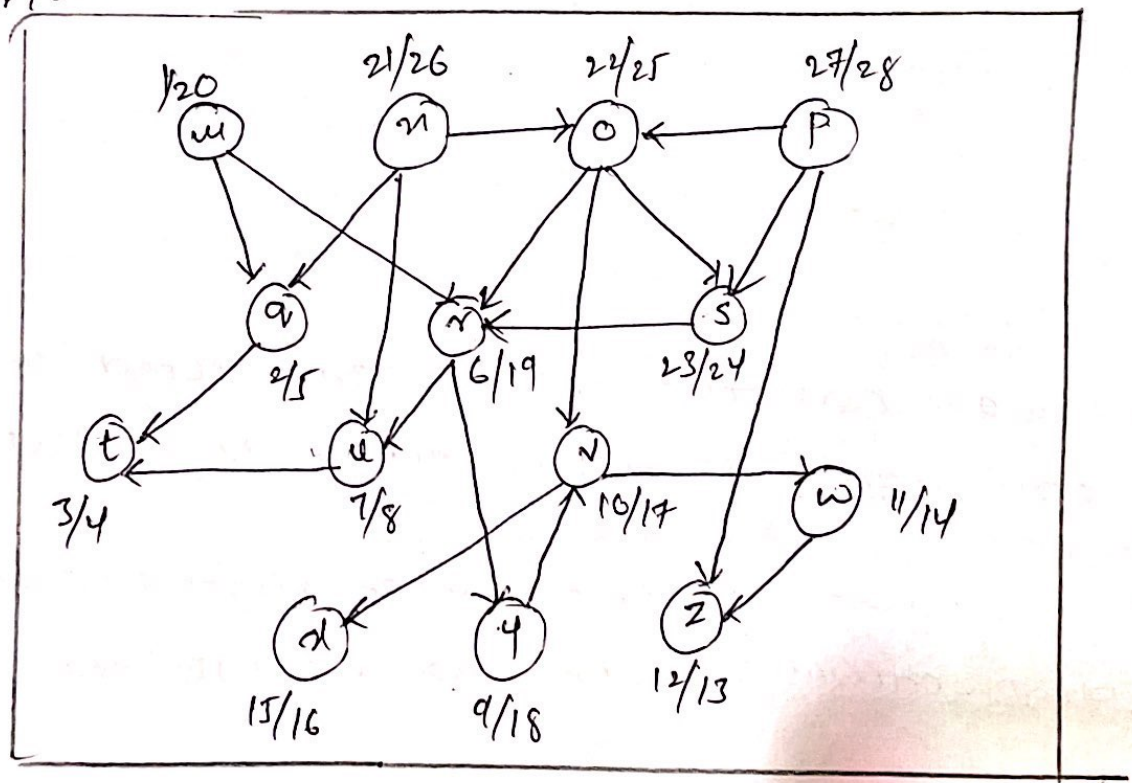
HOMEWORK #7
Advanced Algorithms

MANOJ SUVVALA
700744733

Question-1 show the ordering of vertices produced by topological sort when it is run on the DAG of below figure



Answer) From the given point DAG graph, if we run DFS on this graph we get the finishing time for each vertex as below.



Scanned with CamScanner

So, now the first step of topological sort algorithm is completed, we now order the vertices in the decreasing order of finishing time of vertices as below,

Order of vertices:

p
n
o
s
u
r
y
v
x
w
z
u
q
t

Ques-2) How can the strongly connected component of a graph change if a new edge is added?

Ans) The number of strongly connected components of a graph can change in two way if edge is added they are as follows,

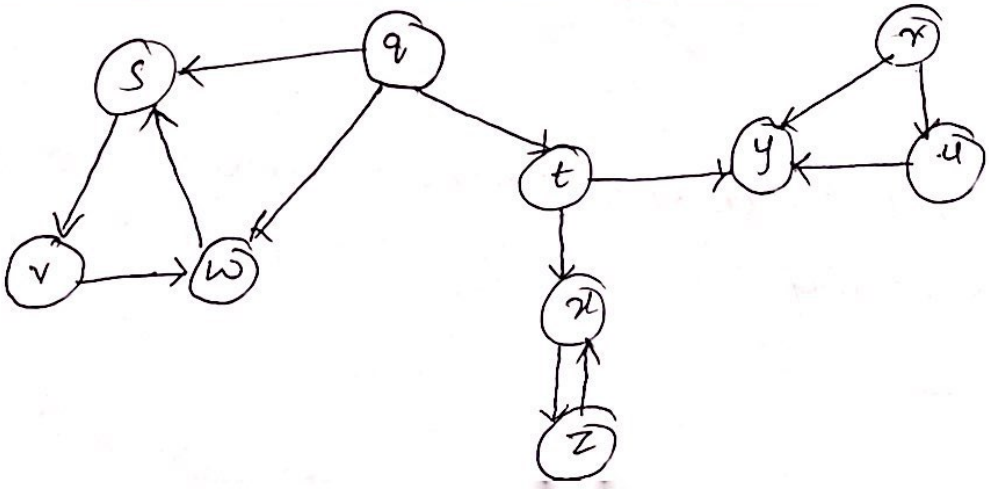
Scanned with CamScanner

i) If a new edge connects two vertices in the same strongly components, the number (count) of strongly connected components will remain the same. This is because the new edge will not create any new paths between vertices in the components.

ii) If the new edge connects two vertices that belong to different strongly connected components, the number of strongly connected components may decrease. This is because the new edge may create a path between two components, which would merge the components into one.

---

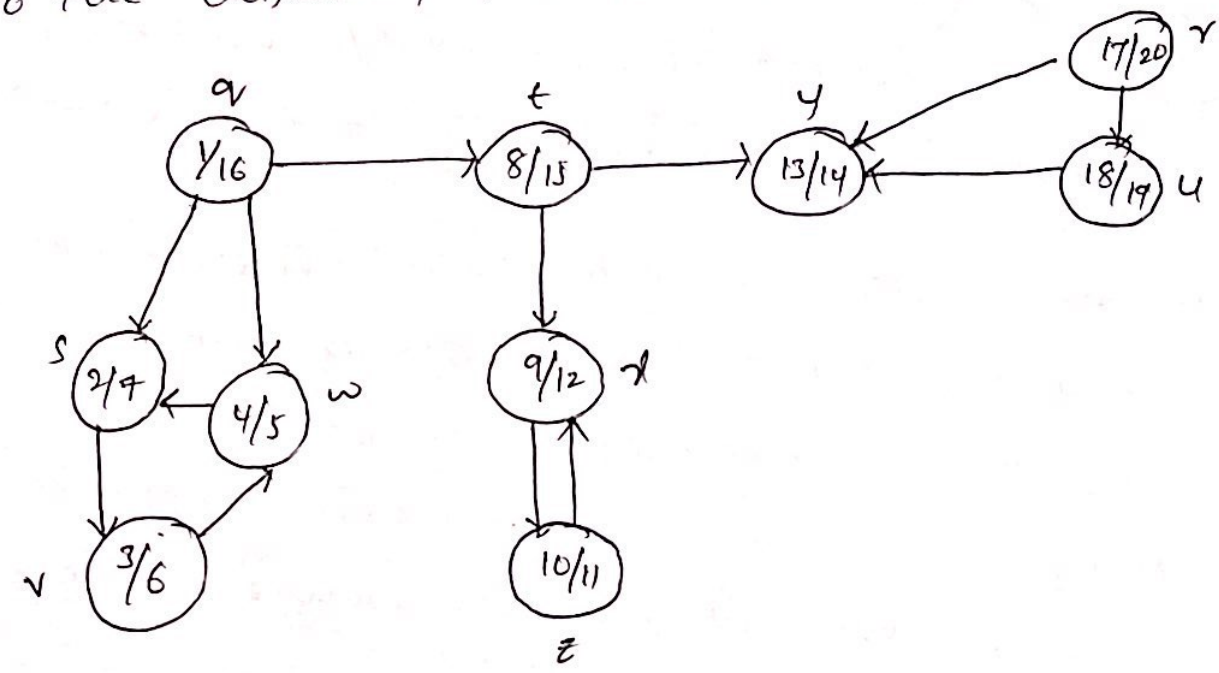
(3) Show how the procedure strongly connected components works on the graph below. Specifically show the finishing times computed in line 1 and the forest produced in line 3. Assume that loop of lines 5-7 of DFS. consider vertices in alphabetical order & that the adjacency list are in alphabetical order.



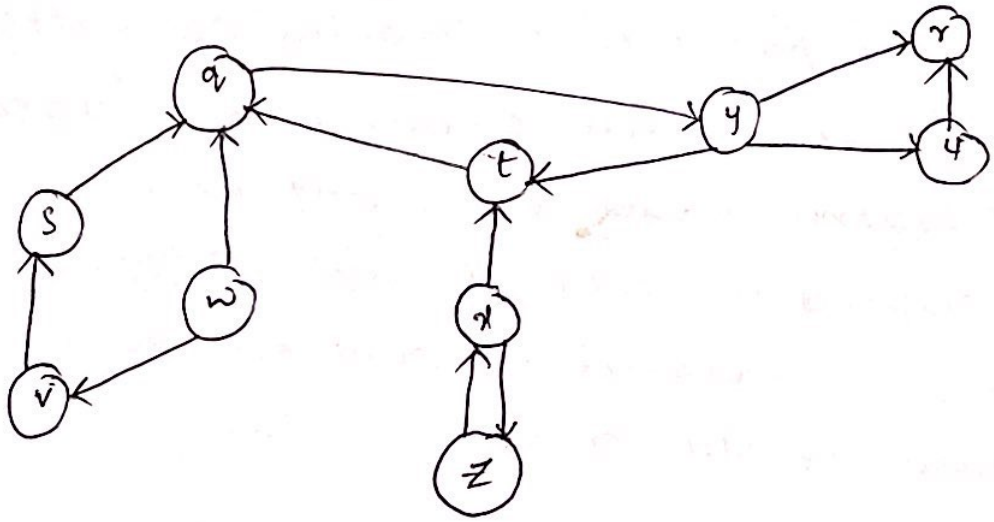
Scanned with CamScanner

sol) **step-1 : DFS**

considering the vertices in alphabetical order & also the adjacency list in alphabetical order

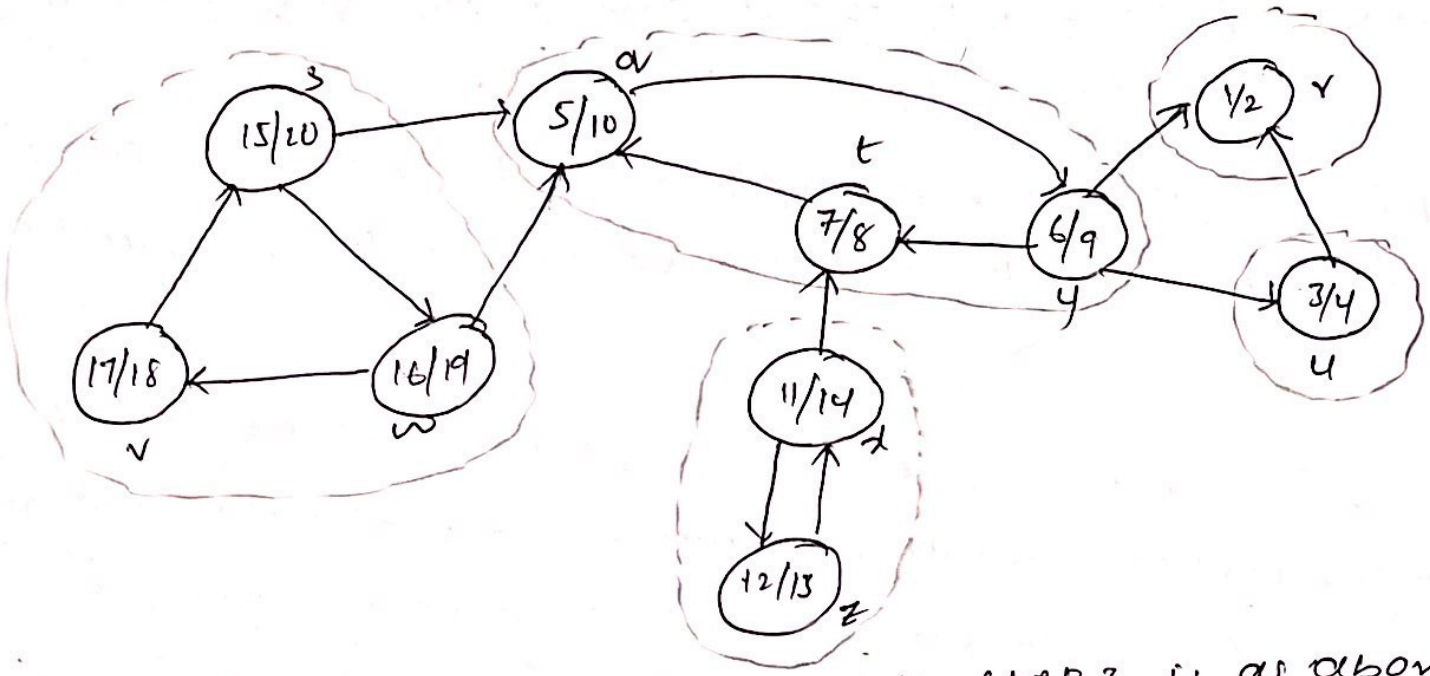


step-2 G^T



step-3 : DFS(G^T), consider the vertices in order of decreasing

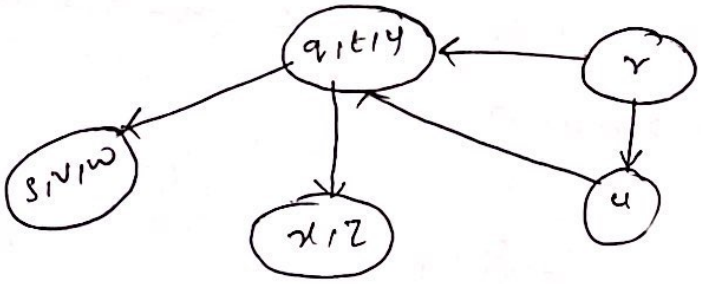
Scanned with CamScanner



so the forest produced in this step 3 is as above.

step:4: so the trees & the vertices of each tree are as below;

$$\{r\}, \{u\}, \{q,t,y\}, \{x,z\}, \{s,v,w\}$$



(4) Do you think it is possible for a minimum spanning tree to have a cycle? Justify your answer.

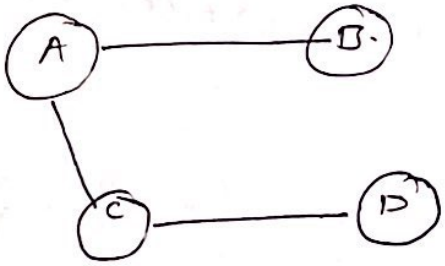
Ans) It is not a possible for a minimum spanning tree to have a cycle. It is a subset of the edges connected edge weighted undirected graph that connects all

Scanned with CamScanner

vertices together without any cycles & with the minimum possible total edge weight.

→ the minimum spanning tree does not have any cycle. It includes all the vertices with v-1 number of edges.

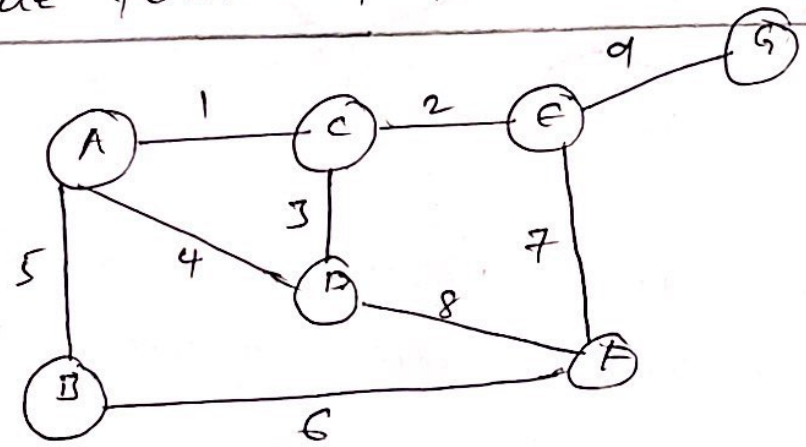
eg: If we have 4 vertices & 3 edges. To get a loop no. of vertices should be equal to no. of edges & if no. of vertex & no. of edges are equal then its not a minimum spanning tree.



No. of vertices = 4
No. of edges = 3

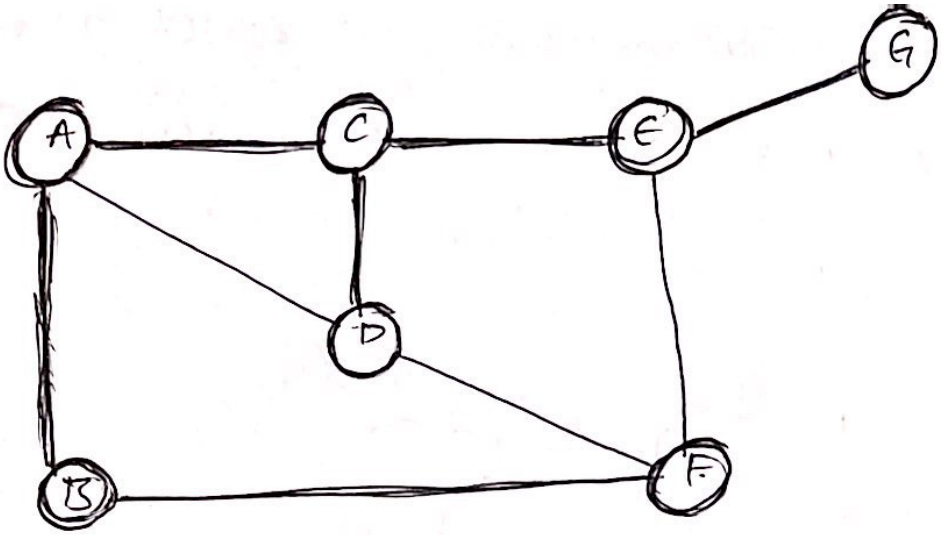
---

(5) Use prims algorithm to find a minimum spanning tree for the following graph



sol) Given graph

Scanned with CamScanner



Taking A as source.

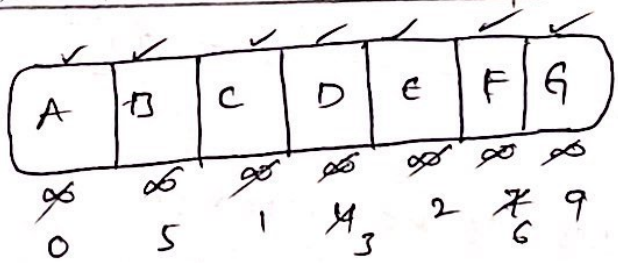
Vertex	key	π
A	~~∞~~ 0	NIL
B	~~∞~~ 5	NIL A
C	~~∞~~ 1	NIL A
D	~~∞~~ ~~4~~ 3	NIL ~~A~~ C
E	~~∞~~ 2	NIL C
F	~~∞~~ ~~7~~ 6	NIL ~~E~~ B
G	~~∞~~ 9	NIL E

No. of vertices = 7

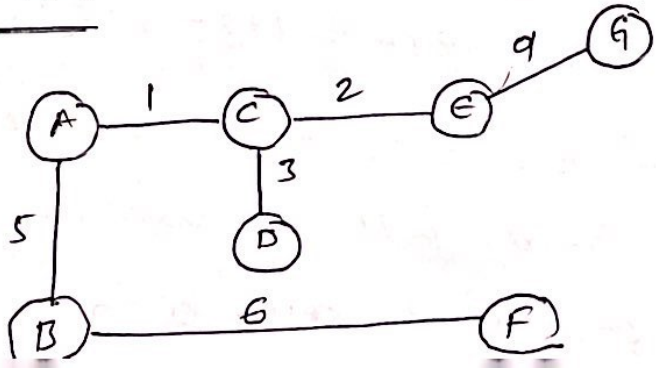
No. of edges = 6

Total cost =
1 + 2 + 3 + 5 + 6 + 9
= 26

Queue



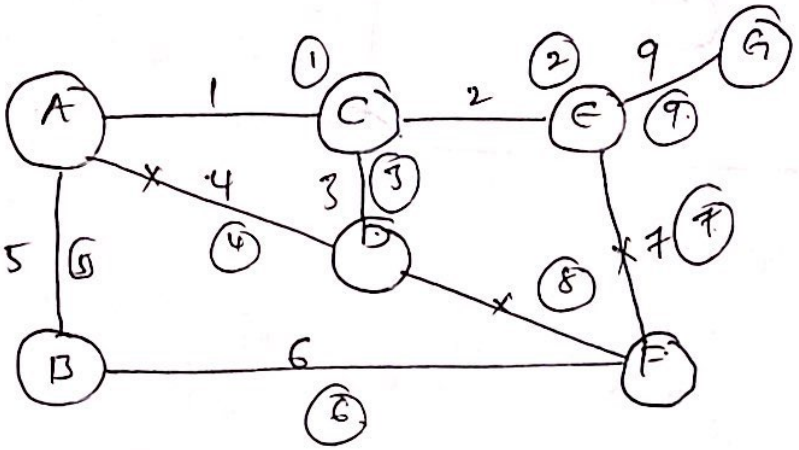
Minimum spanning tree:



Scanned with CamScanner

(6) Use kruskal's algorithm to find a minimum spanning tree for the graph given in Question.

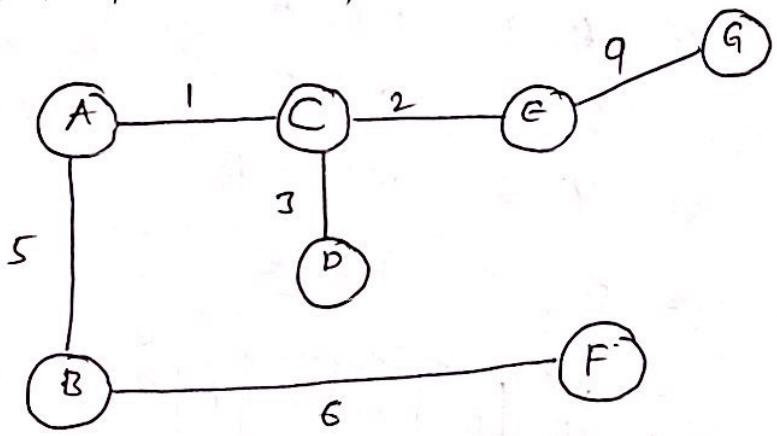
Sol) Given graph



edges sorted : 1, 2, 3, 4, 5, 6, 7, 8, 9.

So considering the edges in sorted order

Minimum spanning tree:



No. of vertices = 7

No. of edges = 6

Total cost = 1+2+3+5+6+9 = 26.

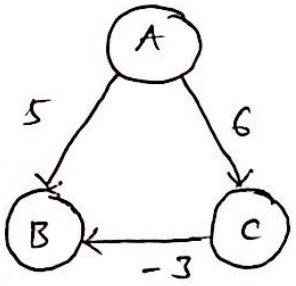
(7) Can Dijkstra's algorithm be used to find the shortest paths in a graph with some negative weights. If so prove it. Otherwise give a counter example?.

Scanned with CamScanner

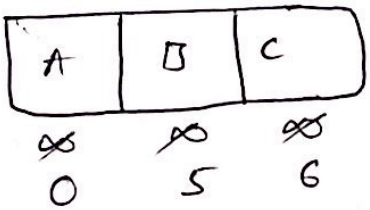
Ans) Dijkstra's Algorithm solves shortest path problem for direct weighted graph with non-negative edge weights. It is a greedy algorithm. It starts at source vertex, grows a tree, that eventually spans all vertices reachable from the source vertex. So Algorithm fails to calculate the shortest path correctly with negative weights.

---

example:



vertex	Distance	π (parent)
A	~~∞~~ 0	NIL
B	~~∞~~ 5	~~NIL~~ A
C	~~∞~~ 6	~~NIL~~ A

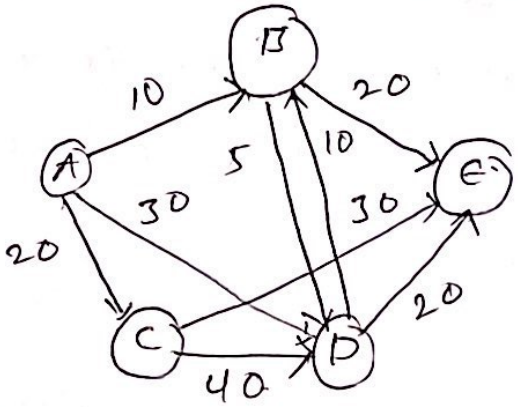


Here, Dijkstra algorithm starts from A & discovers B, C. It visits 'C' and marks as visited. Since it is marked visited the algorithm assumes the path developed this vertex (A|C) is shortest but actually the shortest path for A to C is (A-B-C) due to negative at (B-C)

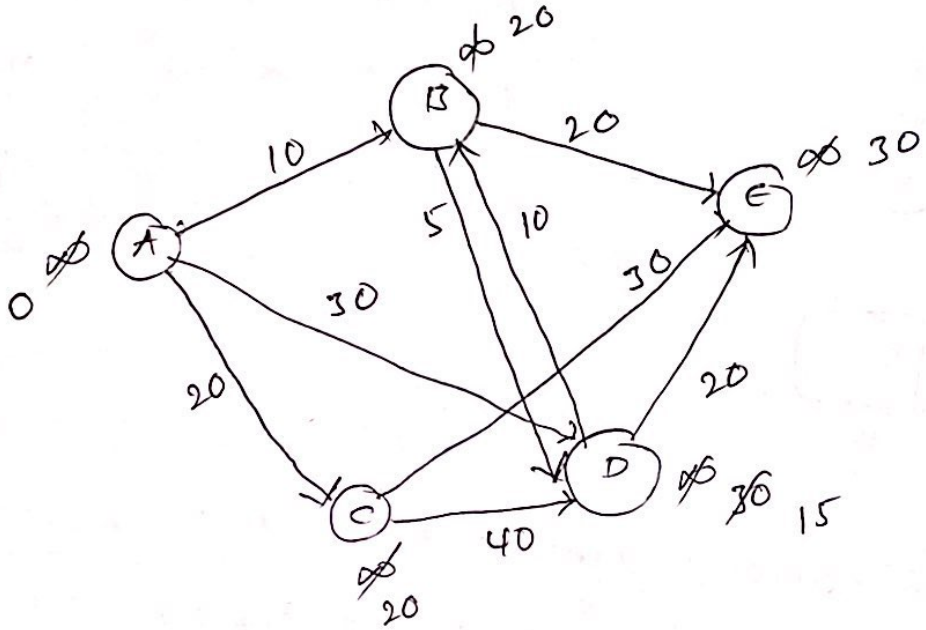
So, the Dijkstra's algorithm fails to find a shortest path.

Scanned with CamScanner

(8) Use Dijkstra's to find the shortest paths from vertex A to all other vertices for the graph given below.



Sol) Given graph is,



Given source is A.

So, from the Algorithm (Dijkstra's)

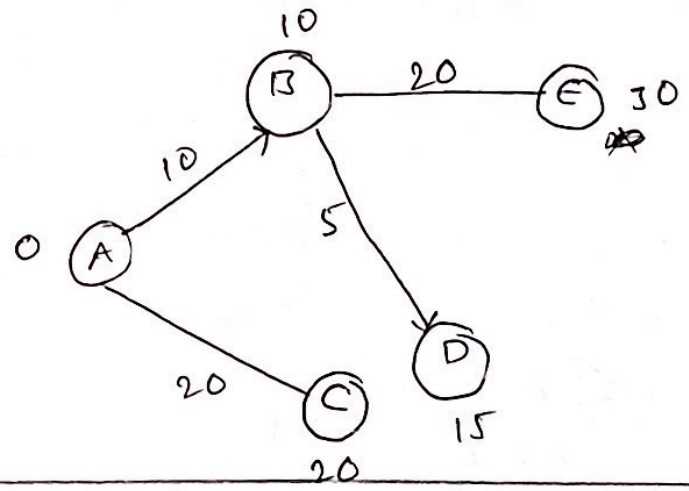
Queue =

A	B	C	D	E
∞	∞	∞	∞	∞
0	10	20	30	30
			15	

Scanned with CamScanner

Vertex	key	π
A	~~∞~~ 0	NIL
B	~~∞~~ 10	~~NIL~~ A
C	~~∞~~ 20	~~NIL~~ A
D	~~∞~~ ~~30~~ 15	~~NIL~~ ~~A~~ B
E	~~∞~~ 30	NIL B
ϕ		

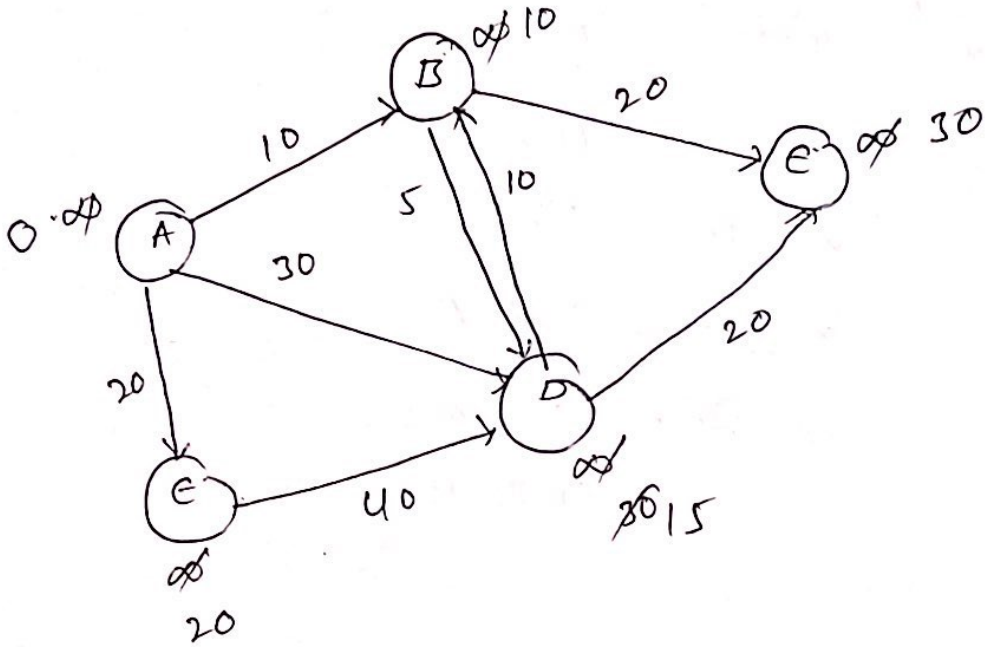
So, shortest paths from vertex A to all other vertices for the given graph is



(9) Use Bellman-Ford Algorithm to find the shortest path from vertex A to all the other vertices for the graph given in Question-8.

Sol) Given graph

Scanned with CamScanner



Given source is A,

so using Bellman-Ford algorithm,

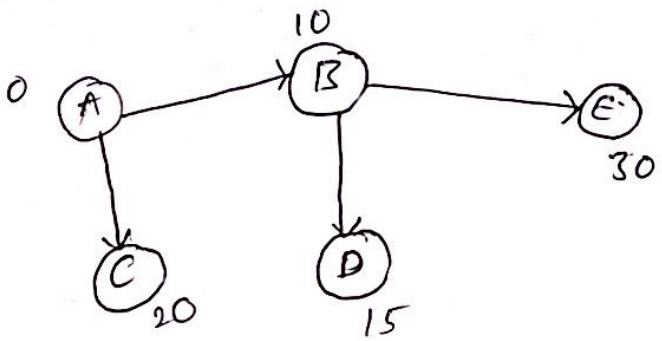
Vertex	d	π
A	∞ 0	NIL
B	∞ 10	~~NIL~~ A
C	∞ 20	~~NIL~~ A
D	∞ ~~30~~ 15	~~NIL~~ ~~A~~ B
E	∞ 30	~~NIL~~ B

$|G.V| = 5$ (no. of vertices)

so $i = 1$ to 4

i	
✓	B, C, D updated
✗	D, E updated
✓	No update
✓	No update
5	STOP!!

So, the shortest path from A to all other source vertices for graph is,



Scanned with CamScanner