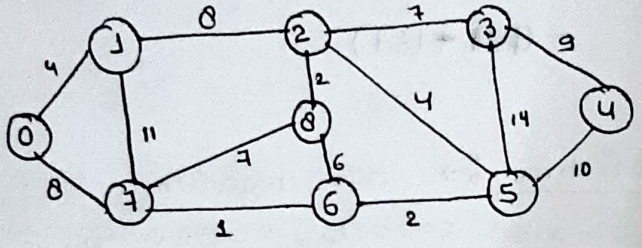
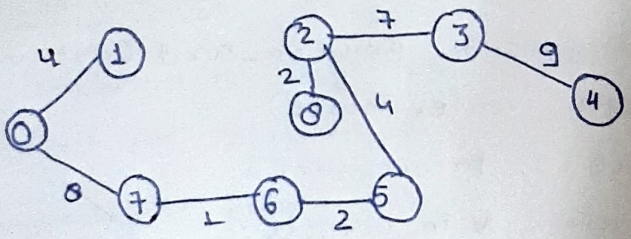
Name- Harshita Mishra
University Rollno → 2017498
Section - CST-SPL-1

TUTORIAL-06

Ques 1 :→ What do you mean by minimum Spanning Tree? What are the applications of MST?

Ans :→ A minimum spanning Tree or minimum weight spanning Tree for a weighted, connected, undirected graph is a Spanning Tree with weight less than or equal to the weight of every other spanning Tree. The weight of spanning Tree is the sum of weights given to each edge of the Spanning Tree.





$\Rightarrow 4+8+1+2+4+2+7+9 = 37$

Ques 2 :→ Please analyze the time and space complexity of Prism, Krushkal, Dijkstra and Bellman ford algorithm?

Ans :→ Dijkstra algorithm :→ This is used to Solve Single-source Shortest-paths problem on a weighted and directed graph.

Pseudo code :→ algo

```
function Dijkstra (Graph, source):
    dist [source] ← 0
Create vertex    set 0 for each vertex V in Graph:
        if V ≠ source
    dist[v] ← undefined
```

Q.add_with_priority (v, dist[v])
while Q is not empty:
u ← Q.extract_min()
for each neighbor v of u:
alt ← dist[u] + length(u,v)
if alt < dist[v]
dist[v] ← alt
Prev[v] ← u
Q.decrease_priority(v, alt)
return dist[], Prev[]

Time Complexity → For Best Case → ~~O(E)~~

Average and worst case → O((|V|+|E|) log V)

Space Complexity →

O(|V| + |E|)

② Bellman-Ford Algorithm :→ This algorithm computes shortest-paths from a single source vertex to all other vertices in a weighted directed graph.

Pseudo Code algo → function BellmanFord(list vertices, list edges, vertex source)
// distance[], predecessor[]
for each vertex v in vertices:
distance[v] := INFINITY
Predecessor[v] := null
distance[source] := 0
for(i from 1 to size(vertices)-1):
for each edge (u,v) with weight w in edges:
if distance[u] + w < distance[v]:
distance[v] := distance[u]+w
Predecessor[v] := u
for each edge (u,v) with weight w in edges:
if distance[u] + w < distance[v]:
return "Graph contains a negative-weight cycle"
return distance[], predecessor[]

Time Complexity → O(E) → for best case

for avg & worst case → O(NE)

Space Complexity →

Kruskal's Algorithm :→ It is a greedy algorithm for finding minimum Spanning Tree. It finds a minimum weight.

Pseudo Code algo → Kruskal (G):

```
    A = ∅
    for each v ∈ G.V:
        Make-set(v)
    Sort the edges in G.E in non-decreasing order by weight
    for each (u,v) in G.E ordered by weight(u,v), nondecreasing:
        if FIND-SET(u) ≠ FIND-SET(v):
            A = A ∪ {(u,v)}
            UNION(u,v)
    return A
```

Time Complexity :→ $O(E \log V)$

Space Complexity :→ $O(\log E)$

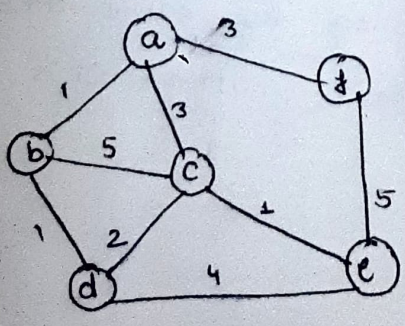
(4) **Prim's Algorithm** :→ It also finds MST in a graph and is closely related to Dijkstra algorithm. It's implementation is also based on implementing the priority queue.

Time Complexity :→ $O(V^2)$

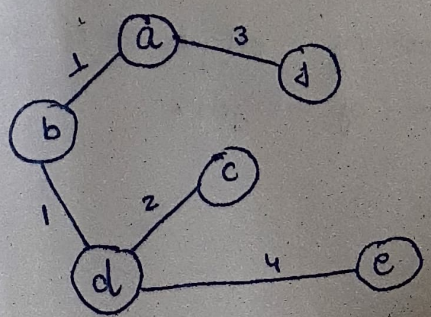
Space Complexity :→ $O(V+E)$

Ques 3:→ Apply Kruskal and Prim's algorithm on graph given on right side to compute MST and its weight.

Ans →



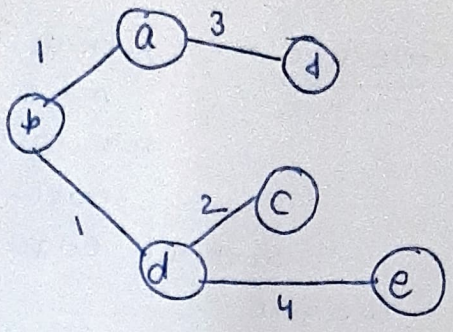
Prim's algorithm :→



⇒ 1+3+1+2+4 = 11

Kruskal's algorithm :→

u	v	weight
b	a	1
b	d	1
c	e	1
d	c	2
a	c	3
a	f	3
d	e	4
f	e	5
b	c	5



Ques4 :→ Given a directed graph (weighted). You are also given the shortest path from a source vertex 's' to a destination vertex 't'. Does the shortest path remain same in the modified graph in following cases?

(i) If weight of every edge is increased by 10 units.

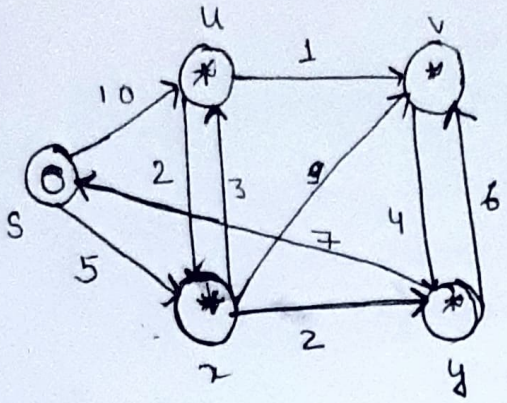
Ans:→ The shortest path may change. The reason is, there may be different number of edges in different paths from s to t. For example, Let shortest path be of weight 15 and has 5 edges. Let there be another path with 2 edges and total weight 25. The weight of the shortest path is increased by 5*10 and becomes 15+50. weight of other path is increased by 2*10 and become 25+20. So, the shortest path changes to the outer path with weight as 45.

(ii) If weight of every edge is multiplied by 10 units.

Ans:→ If we multiply all edge weights by 10, the shortest path doesn't change. The reason is simple, weights of all paths from s to t get multiplied by same amount. The number of edges on a path doesn't matter. It is like changing unit of weights.

Ques 5:→ Apply Dijkstra and Bellman algorithm on graph given on right side to compute shortest path to all nodes from node S.

Ans→



Node	shortest distance from source node
u	0
x	5
v	9
y	7

1st → s(0) u(10) v(∞) x(5) y(∞)

2nd → s(0) u(10) v(11) x(5) y(∞)

3rd → s(0) u(8) v(9) x(5) y(7)

4th → s(0) u(8) v(9) x(5) y(7)

→ graph does not have cycle.