Experiment 8

1) Prim's algorithm

Step a) Initialize the minimum spanning tree with a vertex chosen at random.

b) Find all the edges that connect the tree with a new vertices ~~chosen at random~~. find the minimum and add it to the tree.

c) Keep repeating step (b) until we get a minimum spanning tree

PseudoCode

```
T = ϕ
U = {1}
while (U ≠ V)
    let (u,v) be the lowest cost edge such that
            u∈U and v∈ V-U
    T = T U {(u,v)}
    U = U U {v}
```

Complexity:-

Time Complexity = O(E log V)

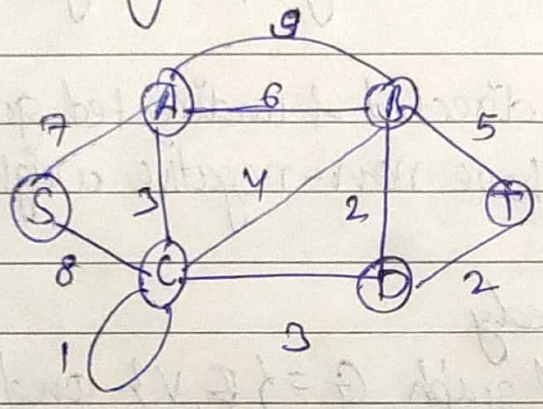
Applications

i) Laying cables of electrical wiring
ii) In network designed
iii) To make protocols in network cycles.

Prim's Algorithm to find MST

⇒ To find minimum cost spanning tree uses the greedy approach.

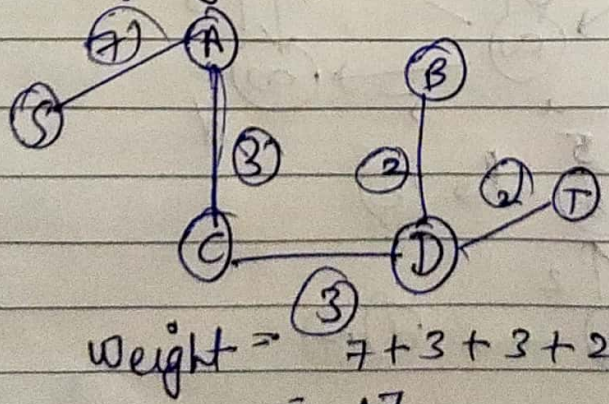
⇒ Prim's algo., in contrast with kruskal's algo., treats the nodes as a single tree and keeps on adding new nodes to the spanning tree from the given graph.



Step 1) Remove all loops and parallel edges.
(In case of Parallel edges, keep the one which has the least cost associated and remove all other)

Step 2) Choose any arbitrary node as root node.

Step 3) Check outgoing edges and select the one with less cost.



Weight = 7 + 3 + 3 + 2 + 2
= 17

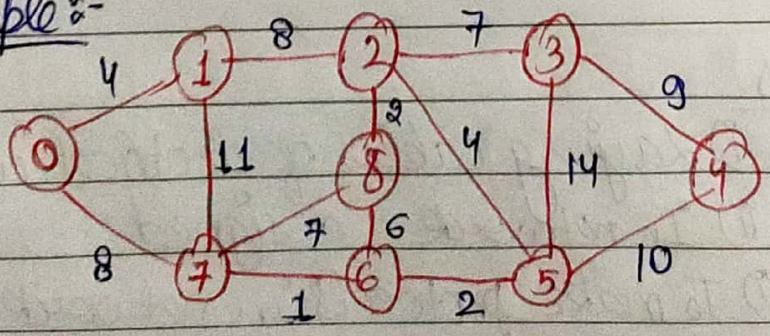
2) Kruskal's Algorithm

Steps (a) Sort all the edges from low weight to high
(b) Take the edge with the lowest weight and add it to the spanning tree. If adding the edge created a cycle, then reject this edge.
(c) Keep adding edges until we reach all vertices.

Pseudocode

```
A = ϕ
for each vertex v ∈ G.V:
    Make-SET(v)
for each edge (u,v) ∈ G.E ordered by increasing
    order by weight(u,v):
    if FIND-SET(u) ≠ FIND SET(v):
        A = A ∪ {(u,v)}
        UNION(u,v)
return A
```

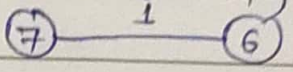
Example :-



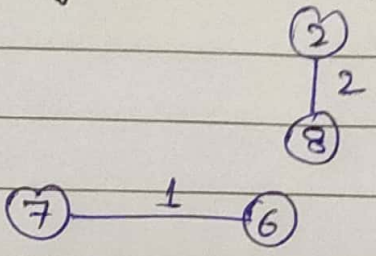
MST contain (V-1) edges. Here V = 9 so edges are 8

→ Sort the weights and pick all edges one by one from sorted list of edges

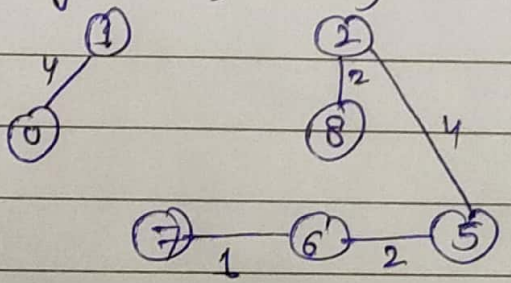
1) Pick edge 7-6 : No cycle is formed, include it



2) Pick edge 8-2 : No cycle is formed, include it

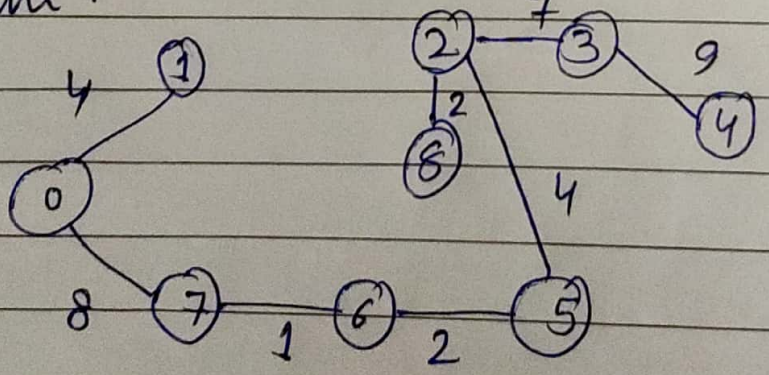


3) Similarly, pick edge 6-5, 0-1, 2-5 bcz. no cycle is formed



4) Pick edge 8-6 : Since including this edge results in cycle discard it. Similary edge 7-8 and so on..... skip when we get (V-1) edges (since no. of edges equals (V-1) the algo. stops)

5) Resultant :-



Time complexity :- $O(E \log V)$