Shri Vile Parle Kelavani Mandal's

Institute of Technology, Dhule

Department of Information Technology

Design and Analysis of Algorithms Lab

Assignment

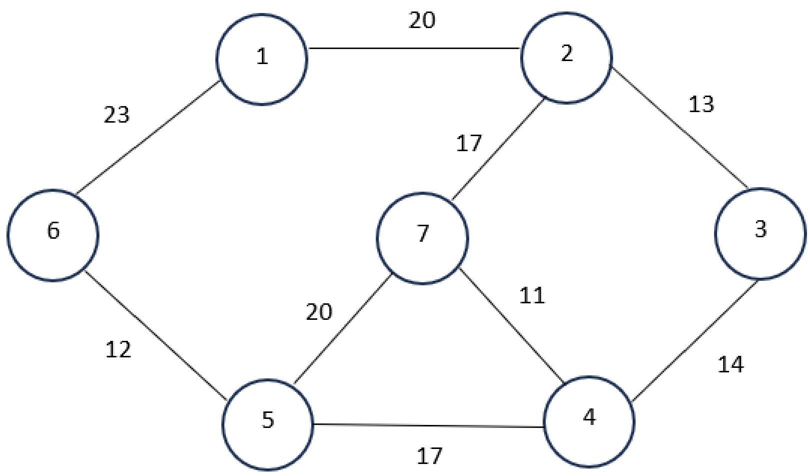
Name : Anushka Diwakar Pawar

Roll No : 50

Class : Second Year (Information Technology)

PRN : 2254491246002

Problem : Differentiate between Prim's and Kruskal's algorithm to find minimum cost spanning tree. Obtain minimum cost spanning tree for the following graph using Kruskal's Algorithm.



Solution :

Minimum Spanning Tree (MST) :

A minimum spanning tree (MST) is defined as a spanning tree that has the minimum weight among all the possible spanning trees.

Properties of a Minimum Spanning Tree :

The minimum spanning tree holds the below-mentioned Properties :

- The number of vertices (V) in the graph and the minimum spanning tree is the same.
- There is a fixed number of edges in the minimum spanning tree which is equal to one less than the total number of vertices (E = V-1).
- The minimum spanning tree should be acyclic, which means there would not be any cycle in the tree.
- The sum of the weights of the edges in the Minimum Spanning Tree is minimized among all possible spanning trees of the graph.

➢ Algorithms to find Minimum Spanning Tree :

1. Prim’s Minimum Spanning Tree Algorithm
2. Kruskal’s Minimum Spanning Tree Algorithm

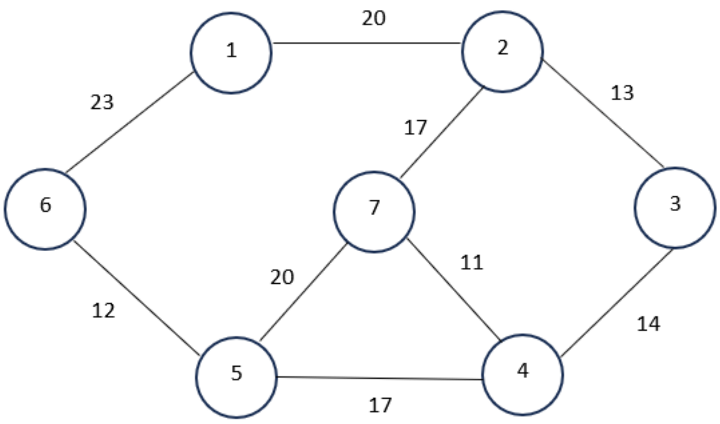
- **Difference between Prim’s Algorithm and Kruskal’s Algorithm :**

Sr. No.	**Prim’s Algorithm**	**Kruskal’s Algorithm**
1.	This algorithm begins to construct the shortest spanning tree from any vertex in the graph.	This algorithm begins to construct the shortest spanning tree from the vertex having the lowest weight in the graph.
2.	To obtain the minimum distance, it traverses one node more than one time.	It crosses one node only one time.
3.	The time complexity of Prim’s algorithm is O(V2).	The time complexity of Kruskal’s algorithm is O(E log V).
4.	In Prim’s algorithm, all the graph elements must be connected.	Kruskal’s algorithm may have disconnected graphs.
5.	When it comes to dense graphs, the Prim’s algorithm runs faster.	When it comes to sparse graphs, Kruskal’s algorithm runs faster.
6.	It prefers list data structure.	It prefers the heap data structure.

Algorithm For Kruskal's Algorithm :

1. Sort all the edges in increasing order of their weight.
2. Pick the smallest edge. Check if it forms a cycle with the spanning tree formed so far. If the cycle is not formed, include this edge. Else, discard it.
3. Repeat step 2 until there are (V-1) edges in the spanning tree.

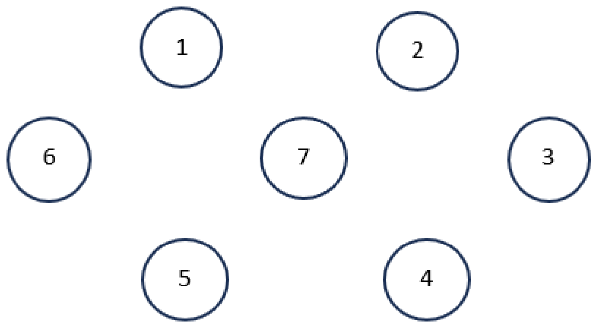
The minimum cost spanning tree for the following graph using Kruskal's Algorithm.



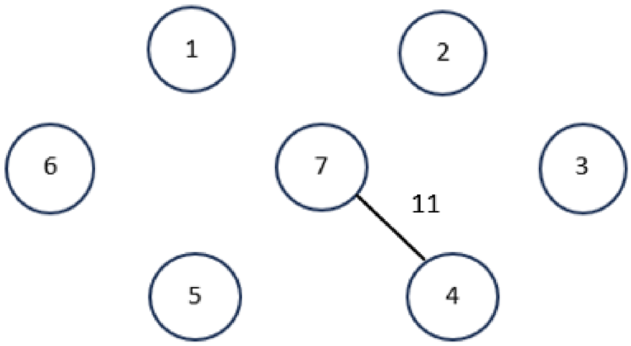
Edges in increasing order of their weight :

Edge	**Weight**
7 - 4	11
6 - 5	12
2 - 3	13
3 - 4	14
2 - 7	17
5 - 4	17
1 - 2	20
7 - 5	20
1 - 6	23

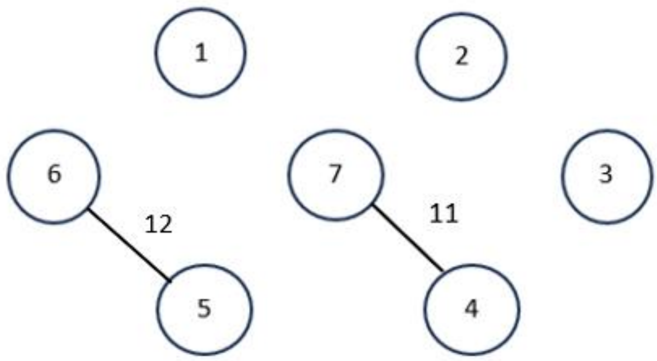
Step 1 : Plot all Vertices without edges. Initially we have 7 trees. Tree {1}, Tree {2}, Tree {3}, Tree {4}, Tree {5}, Tree {6} and Tree {7}.



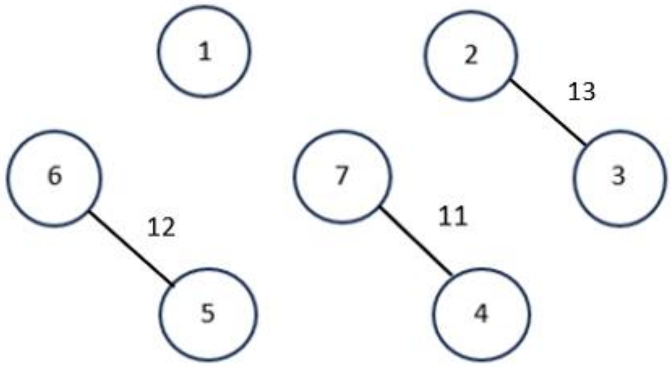
Step 2 : Edge 7 - 4 have minimum weight Hence plot edge between Tree {7} and Tree {4} by edge 7 – 4. New Tree will be {7,4}.



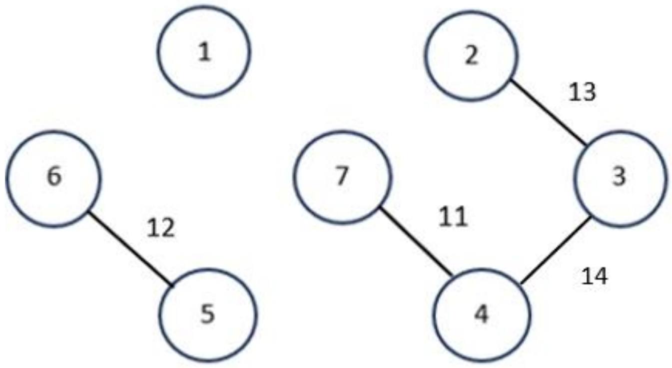
Step 3 : Next Edge 6 - 5 have minimum weight Hence plot edge between Tree {6} and Tree {5} by edge 6 – 5. New Tree will be {6,5}.



Step 4 : Next Edge 2 - 3 have minimum weight Hence plot edge between Tree {2} and Tree {3} by edge 2 – 3. New Tree will be {2,3}.

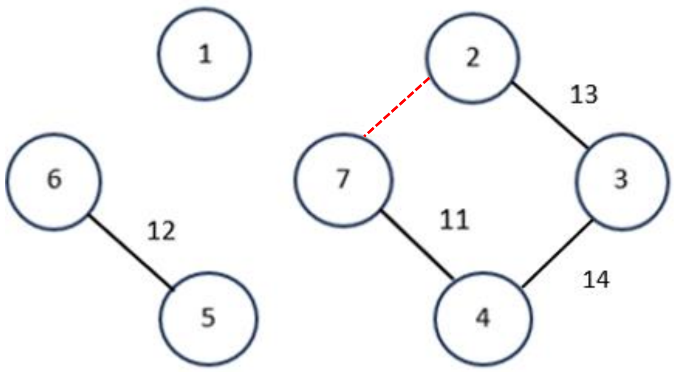


Step 5 : Next Edge 3 - 4 have minimum weight Hence plot edge between Tree {7,4} and Tree {2,3} by edge 2 – 3. New Tree will be {7,4,2,3}.

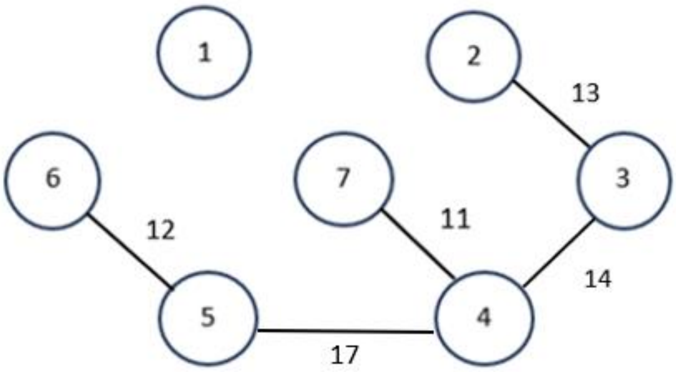


Step 6 : Next Edge 2 - 7 have minimum weight But it forms a cycle (2–3–4–7)

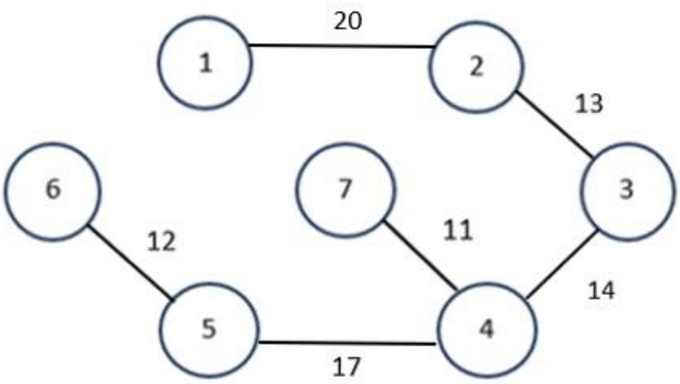
Hence reject it.



Step 7 : Next Edge 5 - 4 have minimum weight Hence plot edge between Tree {7,4,2,3} and Tree {6,5} by edge 2 – 3. New Tree will be {7,4,2,3,6,5}.

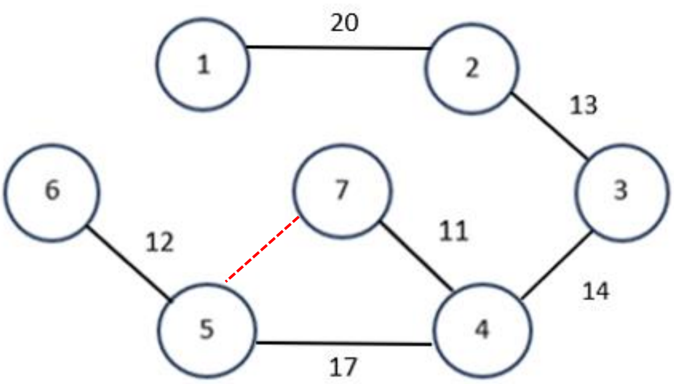


Step 8 : Next Edge 1 - 2 have minimum weight Hence plot edge between Tree {7,4,2,3,6,5} and Tree {1} by edge 2 – 3. New Tree will be {7,4,2,3,6,5,1}.

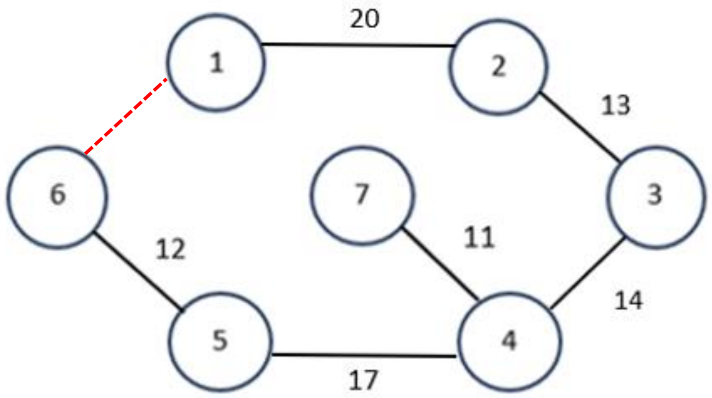


Step 9 : Next Edge 7 - 5 have minimum weight But it forms a cycle (7-5-4)

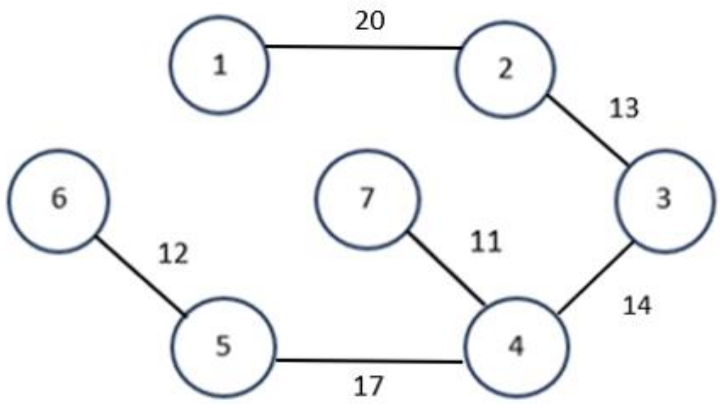
Hence reject it.



Step 10 : Next Edge 1 - 6 have minimum weight But it forms a cycle (1-2-3-4-5-6) Hence reject it.



Hence Final Minimum Spanning Tree is :



Total Minimum weight required

to visit all vertices is = (7,4) + (6,5) + (2,3) + (3,4) + (5,4) + (1,2)

= 11 + 12 + 13 + 14 + 17 + 20

= 87

Edge	Weight	Status
7 - 4	11	Considered
6 - 5	12	Considered
2 - 3	13	Considered
3 - 4	14	Considered
2 - 7	17	Not - Considered
5 - 4	17	Considered
1 - 2	20	Considered
7 - 5	20	Not - Considered
1 - 6	23	Not - Considered