Shri Vile Parle Kelavani Mandal's

Institute Of Technology, Dhule

Department of Information Technology

Design and Analysis of Algorithm lab

Assignment

Student Name	Roll No	PRN	Batch	Class
Vedika Kamlakar Shirude	56	2254491246056	S4	Second year

Kruskal's algorithm: Kruskal's algorithm is a greedy approach to finding a minimum spanning tree (MST) in a graph. It starts with each vertex as a separate tree (a forest of trees) and iteratively adds the smallest edge that doesn't form a cycle until all vertices are connected in a single tree (the MST).

Minimum Spanning Tree (MST): If the graph is given then minimum spanning tree is found out by considering all vertices should be included and the sum of weights of edged should be minimum.

Algorithm:

Step 1 : Initialization:

Begin with a forest of nn trees, where each tree is a single vertex. Maintain a list of edges sorted by weight.

Step 2 : Edge Selection:

Iterate over edges in ascending order of weight. For each edge, check if adding it to the MST would form a cycle.

Step 3 : Cycle Detection:

If the two vertices of an edge belong to the same tree, adding this edge would form a cycle, so skip it. Otherwise, include the edge in the MST and merge the two trees

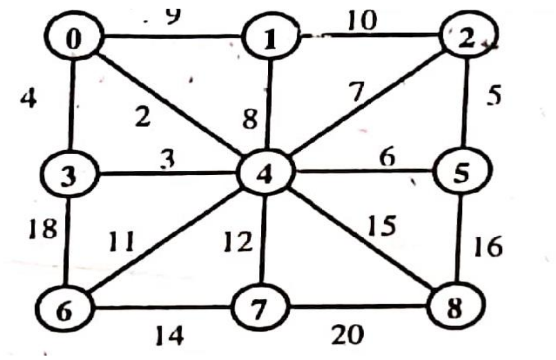
Step 4 : Merging Trees:

When adding an edge between two vertices from different trees, merge these trees into one by updating their tree representatives.

Step 5 : Termination:

Continue until $n-1$n−1 edges have been added to the MST (where nn is the number of vertices).

Problem to Obtain the minimum cost spanning tree for the following graph using Kruskal's algorithm.



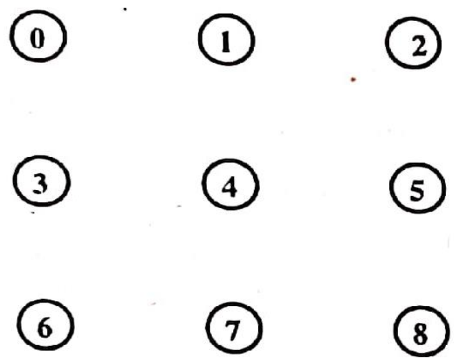
Solution:

Initially we take a forest of 9 trees, with each tree consisting of a single vertex.

All the edges are examined in increasing order of their weights.

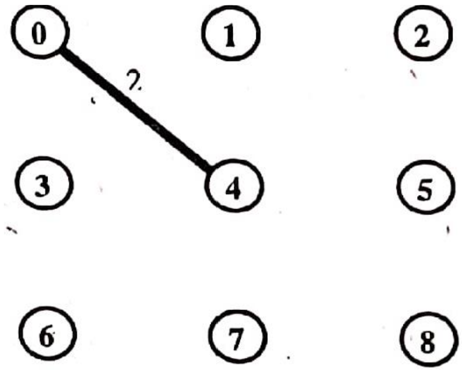
Edge	Weight
Edge 0-4	2
Edge 3-4	3
Edge 0-3	4
Edge 2-5	5
Edge 4-5	6
Edge 2-4	7
Edge 1-4	8
Edge 0-1	9
Edge 1-2	10
Edge 4-6	11
Edge 4- 7	12
Edge 6-7	14
Edge 4-8	15

Step 1 : Plot the vertices of graph without edges.



Initially 9 trees in the forest.

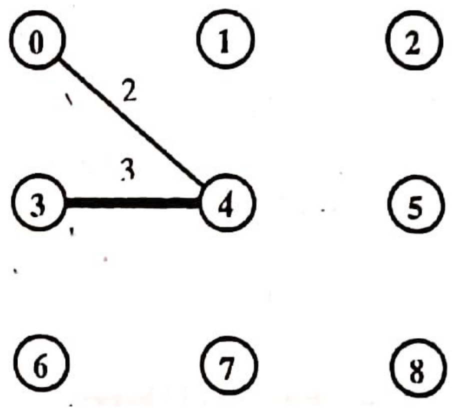
Step 2 :



Tree {0} and {4} joined by edge (0,4) to form tree {0,4}

Now 8 trees left in the forest.

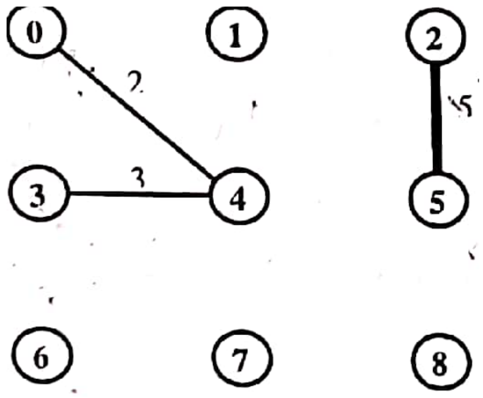
Step 3 :



Trees {0,4} and {3} joined by edge {3,4} to form tree {0,4,3}

Now 7 trees left in the forest.

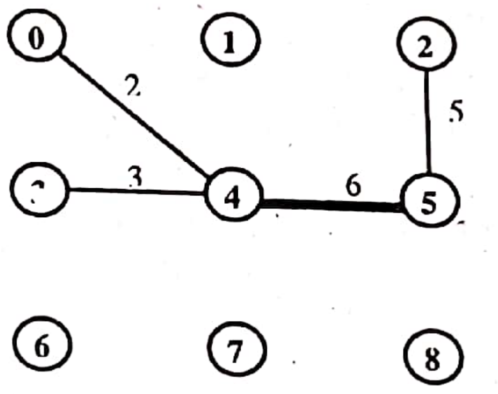
Step 4 :



Tree {2} and {5} joined by edge (2,5) to form tree {2,5}

Now 6 trees left in the forest.

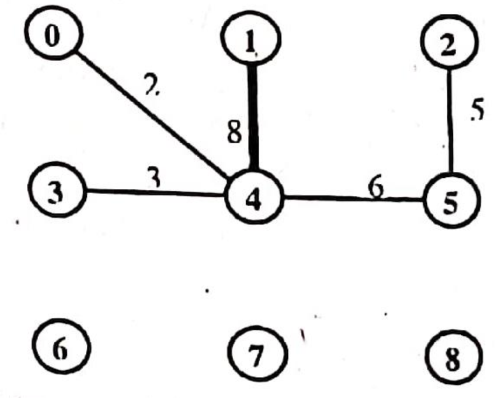
Step 5 :



Trees {0,4,3} and {2,5} joined by edge (4,5) to form tree {0,4,3,2,5}

Now 5 trees left in the forest.

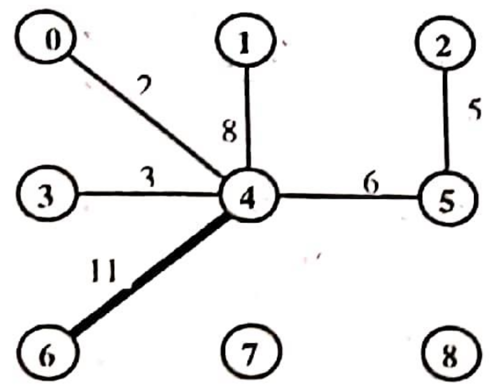
Step 6:



Trees {0,4,3,2,5} and {1} joined by edge (1,4) to form tree {0,4,3,2,5,1}

Now 4 trees left in the forest.

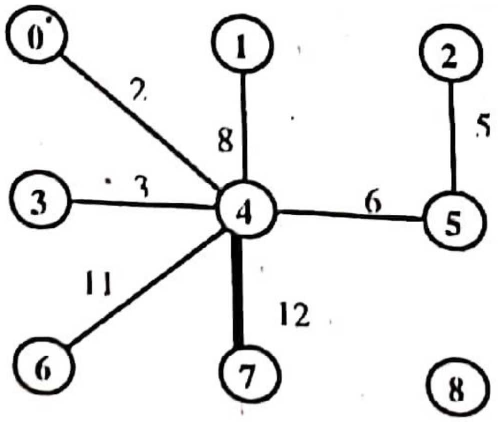
Step 7:



Trees { 0,4,3,2,5,1} and { 6} joined by edge (4,6) to form tree {0,4,3,2,5,1,6}

Now 3 trees left in the forest.

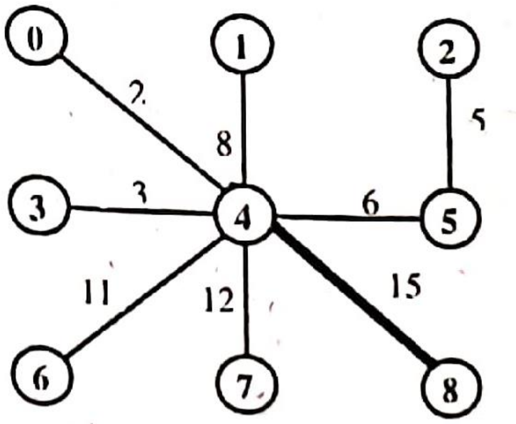
Step 8:



Trees {0,4,3,2,5,1,6} and {7} joined by edge(4,7) to form tree {0,4,3,2,5,1,6,7}

Now 2 trees left in the forest.

Step 9:



Trees {0,4,3,2,5,1,6,7} and {8} joined by edge (4,8) to form tree {0,4,3,2,5,1,6,7,8}

Now only 1 tree left and it is the Minimum spanning tree.

Minimum Spanning Tree is:

Edges included in this spanning tree is (0,4), (3,4), (2,5), (4,5), (4,6), (1,4), (4,7), (4,8)

Weight of this spanning tree is 2+3+5+6+8+11+12+15=62

❖ Explanation:

Forest of Trees: Initially, each vertex is its own tree. As edges are added, these trees merge into larger trees until a single tree remains (the MST).

Greedy Approach: Kruskal's algorithm selects the smallest available edge at each step without reconsidering previously rejected edges.

Edge Selection: Iterates over edges in ascending order of weight, ensuring that each added edge contributes to forming the MST without introducing cycles.

Cycle Detection: Crucial to avoid creating cycles in the MST. If adding an edge between two vertices already in the same tree would form a cycle, that edge is skipped

❖ Conclusion:

Kruskal's algorithm is effective for finding a minimum spanning tree in a graph by iteratively adding the smallest edge that doesn't create a cycle until all vertices are connected in a single tree. It is efficient due to its greedy nature and the use of data structures like Union-Find to manage the forest of trees and detect cycles. This algorithm guarantees the creation of a minimum spanning tree with $n-1$n−1 edges, where nn is the number of vertices in the graph.