Green University of Bangladesh
Department of Computer Science and Engineering (CSE)
Faculty of Sciences and Engineering
Semester: (Fall, Year:2024), B.Sc. in CSE (Day)

Lab Report NO:02
Course Title: ALGORITHMS LAB
Course Code: CSE 206 Section: 231(D2)

Lab Experiment Name: Implement Kruskal's Algorithm & Implement Prim's Algorithm.

Student Details

	Name	**ID**
1.	Promod Chandra Das	231002005

Lab Date :
Submission Date :
Course Teacher's Name : **Farjana Akter Jui**

Lab Report Status
Marks: ………………………………… **Signature:**....................
Comments:... **Date:**.............................

❖ **TITLE OF THE LAB REPORT EXPERIMENT**

Implement Kruskal’s Algorithm

❖ **OBJECTIVES**

Implement Kruskal's Algorithm to find the Minimum Spanning Tree (MST) of a graph.
Key objectives include understanding the algorithm,
utilizing efficient data structures (edge list, disjoint set), sorting edges, detecting cycles,
constructing the MST, and ensuring performance optimization.
Validate correctness through diverse test cases.

❖ **PROBLEM ANALYSIS**

- **Kruskal’s Algorithm**

Kruskal’s algorithm is a minimum spanning tree algorithm that takes a graph as input and finds the subset of the edges of that graph which
• form a tree that includes every vertex.
• has the minimum sum of weights among all the trees that can be formed from the graph.

- **How Kruskal’s algorithm works**

It falls under a class of algorithms called greedy algorithms that find the local optimum in the hopes of finding
a global optimum. We start from the edges with the lowest weight and keep adding edges until we reach our
goal. The steps for implementing Kruskal’s algorithm are as follows:
• Sort all the edges from low weight to high.
• Take the edge with the lowest weight and add it to the spanning tree. If adding the edge creates a cycle,
then reject this edge.
• Keep adding edges until we reach all vertices.

- **Kruskal’s Algorithm Complexity**

The time complexity Of Kruskal’s Algorithm is: O(E log E).

Algorithm 1: Kruskal Algorithm

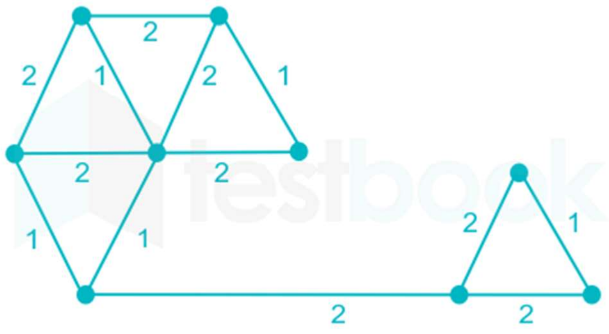
```
1 KRUSKAL(G):
2 A = ∅
3 for each vertex v ∈ G.V: do
4 MAKE-SET(v)
```

```
5 end
6 for each edge (u, v) ∈ G.E ordered by increasing order by weight(u, v): do
7 if FIND-SET(u) 6= FIND-SET(v): then
8 A = A ∪ (u, v)
9 UNION(u, v)
10 end
11 end
12 return A
```

❖ **RESULT / OUTPUT**

Find the number of distinct minimum spanning trees for a given weighted graph.

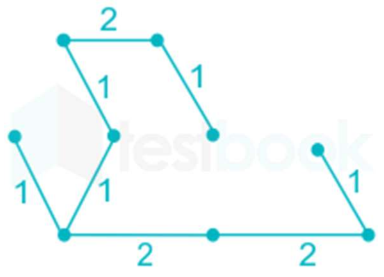
➢



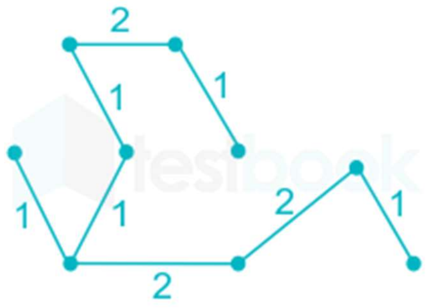
Explanation:

Total 6 minimum spanning trees are possible :

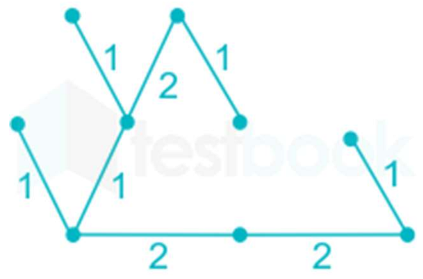
Case 1:



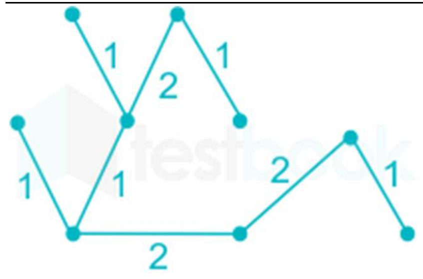
Case 2:



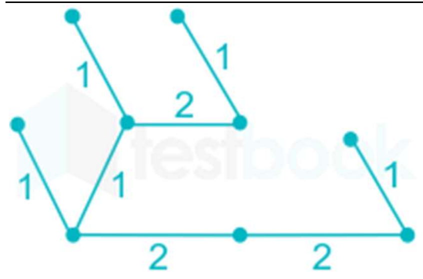
Case 3:



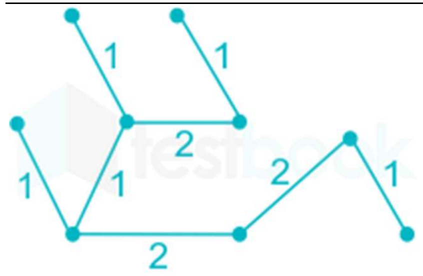
Case 4:



Case 5:



Case 6:



❖ DISCUSSION

Kruskal's Algorithm is an efficient method for finding the Minimum Spanning Tree (MST) of a connected, undirected graph. It works by sorting all edges by weight and adding them one by one, ensuring no cycles form using a Disjoint Set (Union-Find) structure. This algorithm is particularly effective for sparse graphs and has a time complexity of O(Elog E)O(E \log E)O(ElogE). Its ability to handle equal-weight edges allows for multiple distinct MSTs, making it versatile in various applications, including network design and clustering.

- **TITLE OF THE LAB REPORT EXPERIMENT**

  Implement Prim’s Algorithm

- **OBJECTIVES**

  Implement Prim’s Algorithm to find the Minimum Spanning Tree (MST) of a connected, undirected graph. Key objectives include understanding the algorithm’s greedy approach, utilizing priority queues for efficient edge selection, ensuring cycle avoidance, and achieving optimal performance with a time complexity of O(Elog V)O(E \log V)O(ElogV). Validate with diverse test cases.

- **PROBLEM ANALYSIS**
  - **Prim’s Algorithm**

Prim’s algorithm is a minimum spanning tree algorithm that takes a graph as input and fnds the subset of the
edges of that graph which
• form a tree that includes every vertex.
• has the minimum sum of weights among all the trees that can be formed from the graph.

  - **How Prim’s algorithm works**

It falls under a class of algorithms called greedy algorithms that fnd the local optimum in the hopes of fnding
a global optimum. We start from one vertex and keep adding edges with the lowest weight until we reach our
goal. The steps for implementing Prim’s algorithm are as follows:
• Initialize the minimum spanning tree with a vertex chosen at random.
• Find all the edges that connect the tree to new vertices, fnd the minimum and add it to the tree.
• Keep repeating step 2 until we get a minimum spanning tree.

  - **Prim’s Algorithm Complexity**

Prim's Algorithm: O(V2)O(V^2)O(V2) or O(Elog V)O(E \log V)O(ElogV) complexity.

Algorithm 1: Prim’s Algorithm

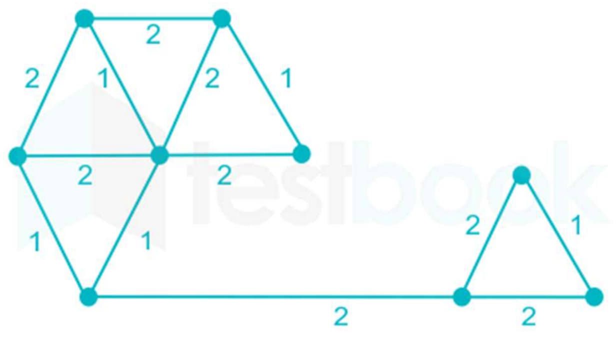
```
1 T = ∅;
2 U = 1 ;
3 while (U 6= V) do
4 let (u, v) be the lowest cost edge such that u ∈ U and v ∈ V - U;
```

```
5 T = T ∪ (u, v)
6 U = U ∪ v
7 end
```

❖ **RESULT / OUTPUT**

• Find the number of distinct minimum spanning trees for a given weighted graph.

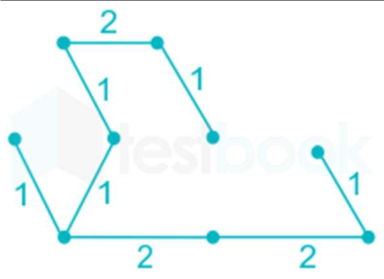
➢



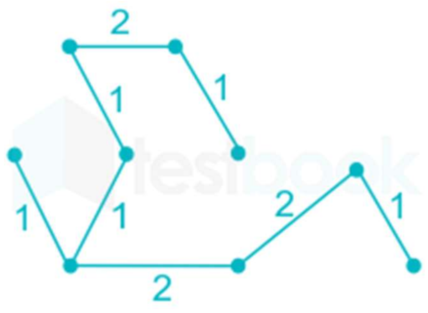
Explanation:

Total 6 minimum spanning trees are possible :

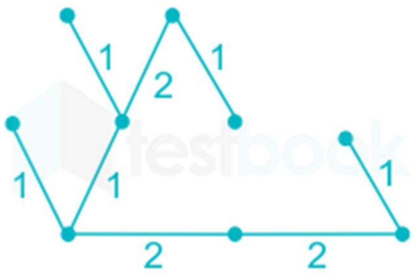
Case 1:



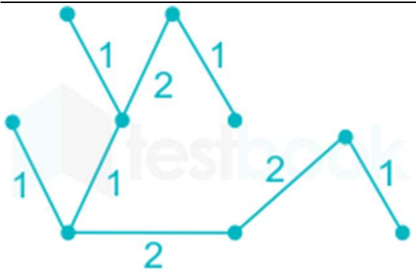
Case 2:



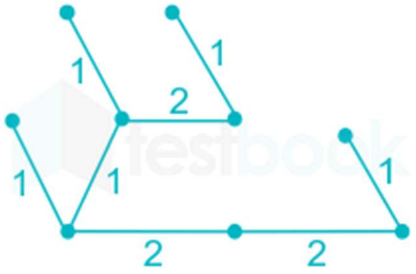
Case 3:



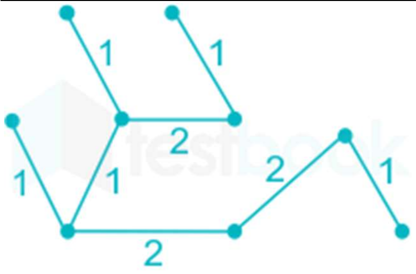
Case 4:



Case 5:



Case 6:



❖ DISCUSSION

Prim's algorithm finds the Minimum Spanning Tree (MST) of a connected, undirected graph by adding the smallest edge connecting the MST to a new vertex. It uses a greedy approach, efficiently implemented with a priority queue, making it suitable for dense graphs. It's essential in network design and clustering.