Prims & Kruskal's

Algorithm....

Learn Vid Fun...

Prim's & Kruskal's Algorithms:

Prim's Algorithm and Kruskal's Algorithm are the two greedy algorithms that are used to find the minimum spanning tree (MST) of a given weighted connected undirected graph.

Steps for implementing Prim's Algorithm:

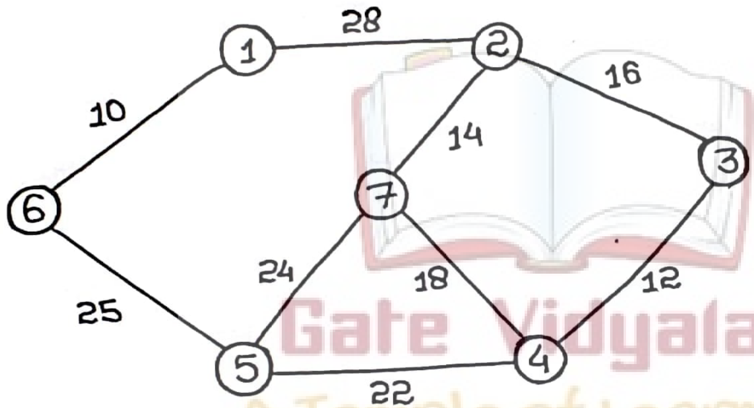
Step-1: Randomly choose any vertex. We usually select and start with a vertex that connects to the edge having least weight.

Step-2: Find all the edges that connect the tree to new vertices, then find the least weight edge among those edges and include it in the existing tree. If including that edge creates a cycle, then reject that edge.

Step-3: Keep repeating step-2 until all vertices are included and minimum spanning tree (MST) is obtained.

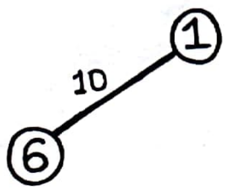
Question:

Construct the minimum spanning tree (MST) for the given graph using Prim's Algorithm-

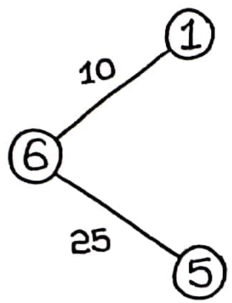


Solution:

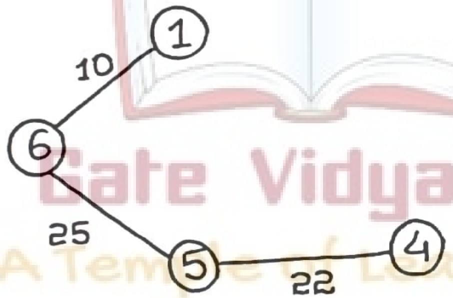
Step-1:



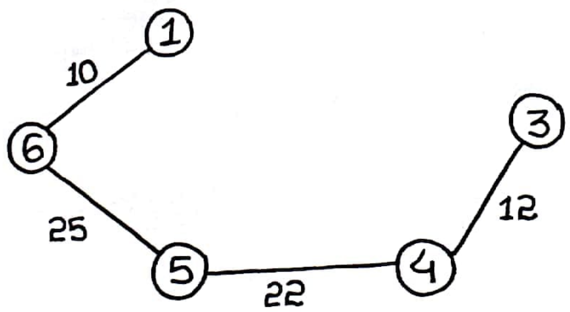
Step-2 :



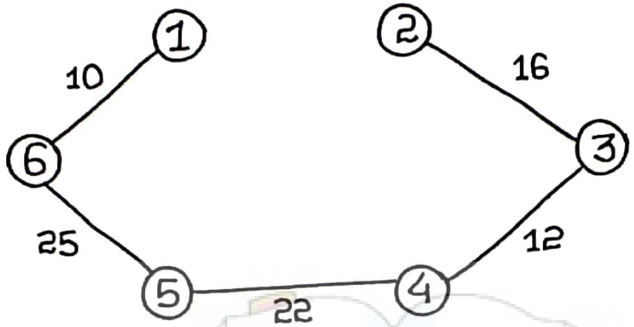
Step-3 :



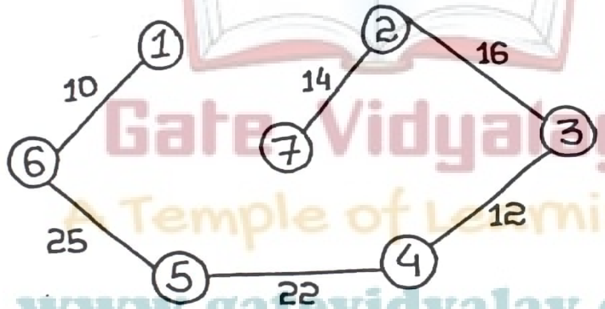
Step-4 :



Step-5:



Step-6:



Since, all vertices are included, so we stop.

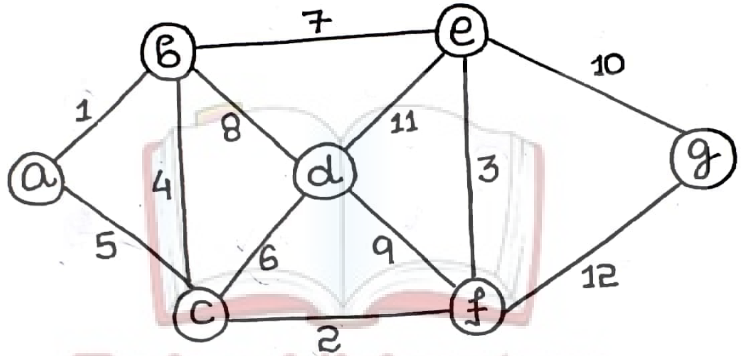
Weight of the MST = Sum of all edges weight

= 10 + 25 + 22 + 12 + 16 + 14

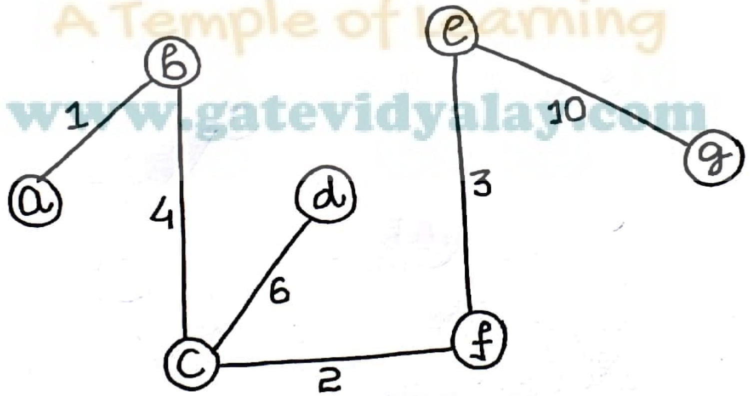
= 99 units

Question:

Construct the minimum spanning tree for the given graph using Prim's Algorithm-



Solution:



Weight of MST

= 1+4+2+6+3+10

= 26 units

MST

Question:

Consider an undirected graph with vertex set {1,2,3,4} represent as a matrix where each entry W_{ij} represents the weight of the edge {i,j}

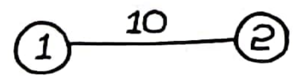
The weight of the minimum spanning tree (MST) is _____?

	1	2	3	4
1	0	10	10	50
2	10	0	40	30
3	10	40	0	20
4	50	30	20	0

Solution:

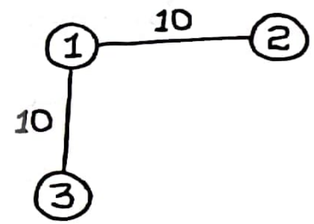
Step-1:

	1	2	3	4
1	0	10	10	50
2	10	0	40	30
3	10	40	0	20
4	50	30	20	0



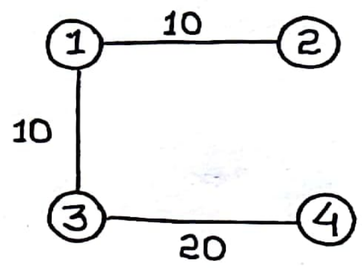
Step-2:

	1	2	3	4
1	0	10	10	50
2	10	0	40	30
3	10	40	0	20
4	50	30	20	0



Step-3:

	1	2	3	4
1	0	10	10	50
2	10	0	40	30
3	10	40	0	20
4	50	30	20	0



Weight of MST = 10 + 10 + 20
= 40 units

Time Complexity of Prim's Algorithm:

Worst case time complexity of Prim's Algorithm

$= O(E \log V)$ using Binary heap

$= O(E + V\log V)$ using Fibonacci heap

How?

If the graph is represented using adjacency list, then all the vertices of a graph can be traversed in $O(V+E)$ time using BFS. The idea is to traverse all vertices of graph using BFS and use a min heap to store the vertices not yet included in MST. Min heap is used as priority queue to get the minimum weight edge. Min heap operations like extracting minimum element and decreasing key value takes $O(\log V)$ time.

So, overall time complexity

$= O(E+V) \times O(\log V)$

$= O((E+V)\log V)$

$= O(E\log V)$

It can be improved & reduce to $O(E + V\log V)$ using Fibonacci heap.