ASSIGNMENT 3

**Name: Soham Makarand Limaye**

**Roll No: 222038**

**PRN: 22111058**

**Batch: B2**

**Aim:** You are given an undirected weighted graph with nodes and edges. The nodes are numbered from and to. Find the total weight of the minimum spanning tree, as well as one specific minimum spanning tree using Prim's algorithm. Note that there may be multiple different minimum spanning trees.

You need to construct any one of them.

**Theory :**

First we take the total number of vertex of graph from the user and after that we take the total vertex of graph and send to object to make partition body of the graph and using the for loop we took the value of the weight of each edge and took the source and destination node value .

**Code :**

import java.util.\*;

class Main { private static final int V = 5; int minKey(int key[], Boolean mstSet[])

{

int min = Integer.MAX\_VALUE, min\_index = -1; for (int v = 0; v < V; v++) if (mstSet[v] == false

&& key[v] < min) { min = key[v]; min\_index = v;

}

return min\_index;

}

void printMST(int parent[], int graph[][])

{

System.out.println("Edge \tWeight");

for (int i = 1; i < V; i++)

System.out.println(parent[i] + " - " + i + "\t"

+ graph[i][parent[i]]);

}

void primMST(int graph[][])

{

int parent[] = new int[V]; int key[] = new int[V];

Boolean mstSet[] = new Boolean[V];

for (int i = 0; i < V; i++) { key[i] = Integer.MAX\_VALUE; mstSet[i] =

false;

}

key[0] = 0; parent[0] = -1; for (int count = 0; count < V - 1; count++) { int u = minKey(key, mstSet); mstSet[u] = true; for (int v = 0; v < V; v++) if (graph[u][v] != 0 && mstSet[v] == false && graph[u][v] < key[v]) { parent[v] = u; key[v] = graph[u][v];

}

}

printMST(parent, graph);

}

public static void main(String[] args)

{

Main t = new Main(); int graph[][] = new

int[][] { { 0, 1, 0, 4, 1 },

{ 2, 4, 1, 8, 5 },

{ 1, 3, 0, 1, 6 },

{ 2, 4, 0, 1, 3 },

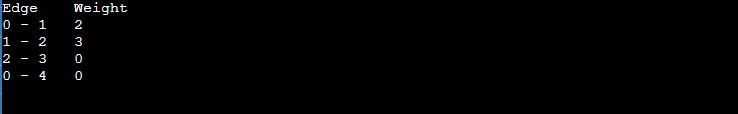
{ 0, 4, 7, 8, 0 } };

t.primMST(graph);

}

}

**Output :**



**Conclusion:**

In this program we successfully created the graph and after that we have found the minimum path to reach from source to destination .