PRIMS ALGORITHM

```
import java.util.*;
public class PrimsAlgorithm {
    static class Edge {
        int src, dest, weight;
        Edge(int src, int dest, int weight) {
            this.src = src;
            this.dest = dest;
            this.weight = weight;
        }
    }
    static int V;
    static List<List<Edge>> graph = new ArrayList<>();
    static void addEdge(int src, int dest, int weight) {
        Edge = new Edge(src, dest, weight);
        graph.get(src).add(edge);
        edge = new Edge(dest, src, weight);
        graph.get(dest).add(edge);
    }
    static void primMST() {
        boolean[] mstSet = new boolean[V];
        int[] parent = new int[V];
        int[] key = new int[V];
        Arrays.fill(key, Integer.MAX_VALUE);
```

```
        Arrays.fill(parent, -1);
        PriorityQueue<Edge> pq = new PriorityQueue<>(V,
Comparator.comparingInt(o -> o.weight));
        key[0] = 0;
        pq.offer(new Edge(-1, 0, 0));

        while (!pq.isEmpty()) {
            Edge e = pq.poll();
            int u = e.dest;
            mstSet[u] = true;
            for (Edge : graph.get(u)) {
                int v = edge.dest;
                int weight = edge.weight;
                if (!mstSet[v] && weight < key[v]) {
                    parent[v] = u;
                    key[v] = weight;
                    pq.offer(new Edge(u, v, weight));
                }
            }
        }
        System.out.println("Minimum Spanning Tree:");
        for (int i = 1; i < V; i++)
            System.out.println(parent[i] + " - " + i);
    }
    public static void main(String[] args) {
        V = 5;
        for (int i = 0; i < V; i++)
```

```
        graph.add(new ArrayList<>());
    addEdge(0, 1, 2);
    addEdge(0, 3, 6);
    addEdge(1, 2, 3);
    addEdge(1, 3, 8);
    addEdge(1, 4, 5);
    addEdge(2, 4, 7);
    addEdge(3, 4, 9);
    primMST();
  }
}
```

OUTPUT:

```
Minimum Spanning Tree:
0 - 1
1 - 2
0 - 3
1 - 4
PS C:\Users\HP\Desktop\LP2>
```