5. Implement Greedy search algorithm for Prim's Minimal Spanning Tree Algorithm

Java Code :

```
class PrimsMST11 {
      private int V, graph[][];

    PrimsMST11(int V, int graph[][]) {
        this.V = V;
        this.graph = graph;
    }

      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 primMSTf() {
            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];
                        }
            }

            System.out.println("\n\n\nPrim's Minimum Spanning Tree:\nEdge
\tWeight");
        int minimumCost = 0;
            for (int i = 1; i < V; i++){
                  System.out.printf("%d -- %d == %d\n", parent[i], i,
graph[i][parent[i]]);
            minimumCost += graph[i][parent[i]];
        }
        System.out.printf("Minimum Cost: %d", minimumCost);
      }
      public static void main(String ar[])
      {
```

```
            int graph[][] = new int[][] {
                  { 0, 4, 0, 0, 0, 0, 0, 8, 0},
                  { 4, 0, 8, 0, 0, 0, 0,11, 0},
                  { 0, 8, 0, 7, 0, 4, 0, 0, 2},
                  { 0, 0, 7, 0, 9,14, 0, 0, 0},
                  { 0, 0, 0, 9, 0,10, 0, 0, 0},
                  { 0, 0, 4,14,10, 0, 2, 0, 0},
                  { 0, 0, 0, 0, 0, 2, 0, 1, 6},
                  { 8,11, 0, 0, 0, 0, 1, 0, 7},
                  { 0, 0, 2, 0, 0, 0, 6, 7, 0}
              };
            PrimsMST11 primsMST = new PrimsMST11(9, graph);
            primsMST.primMSTf();
      }
}
```