**Aim: Implement Greedy search algorithm for any of the following application**

**Prim's Minimal Spanning Tree Algorithm**

import java.util.\*;

import java.lang.\*;

import java.io.\*;

class MST {

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; // Make key 0 so that this vertex is

parent[0] = -1; // First node is always root of MST

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)

{

MST t = new MST();

int graph[][] = new int[][] { { 0, 2, 0, 6, 0 },

{ 2, 0, 3, 8, 5 },

{ 0, 3, 0, 0, 7 },

{ 6, 8, 0, 0, 9 },

{ 0, 5, 7, 9, 0 } };

t.primMST(graph);

}

}

OUTPUT:

Edge Weight

0 - 1 2

1 - 2 3

0 - 3 6

1 - 4 5

