MST.java

```
1       // A Java program for Prim's Minimum Spanning Tree (MST) algorithm.
2       // The program is for adjacency matrix representation of the graph
3       package com.hongchuan.app;
4       import java.util.*;
5       import java.lang.*;
6       import java.io.*;
7
8       public class MST
9       {
10              // Number of vertices in the graph
11              private static final int V=5;
12              private int totalW= 0;
13              // A utility function to find the vertex with minimum key
14              // value, from the set of vertices not yet included in MST
15              public int minKey(int key[], Boolean mstSet[])
16              {
17                      // Initialize min value
18                      int min = Integer.MAX_VALUE, min_index=-1;
19
20  3                  for (int v = 0; v < V; v++)
21  3                          if (mstSet[v] == false && key[v] < min)
22                              {
23                                      min = key[v];
24                                      min_index = v;
25                              }
26
27  1                  return min_index;
28              }
29
30              // A utility function to print the constructed MST stored in
31              // parent[]
32              public void printMST(int parent[], int n, int graph[][])
33              {
34                      // System.out.println("Edge \tWeight");
35  3                  for (int i = 1; i < V; i++)
36                          // System.out.println(parent[i]+" - "+ i+"\t"+
37                                                          // graph[i][parent[i]]);
38  1                          totalW += graph[i][parent[i]];
39                      // System.out.println(totalW);
40                      // return totalW;
41              }
42              public int getTotalW(){
43  1                  return totalW;
44              }
45              // Function to construct and print MST for a graph represented
46              // using adjacency matrix representation
47              public void primMST(int graph[][])
48              {
49                      // Array to store constructed MST
50                      int parent[] = new int[V];
51
52                      // Key values used to pick minimum weight edge in cut
53                      int key[] = new int [V];
54
55                      // To represent set of vertices not yet included in MST
56                      Boolean mstSet[] = new Boolean[V];
57
58                      // Initialize all keys as INFINITE
59  3                  for (int i = 0; i < V; i++)
60                      {
```

```
 61                        key[i] = Integer.MAX_VALUE;
 62                        mstSet[i] = false;
 63                }
 64
 65                // Always include first 1st vertex in MST.
 66                key[0] = 0;      // Make key 0 so that this vertex is
 67                                         // picked as first vertex
 68                parent[0] = -1; // First node is always root of MST
 69
 70                // The MST will have V vertices
 71 3            for (int count = 0; count < V-1; count++)
 72                {
 73                        // Pick thd minimum key vertex from the set of vertices
 74                        // not yet included in MST
 75                        int u = minKey(key, mstSet);
 76
 77                        // Add the picked vertex to the MST Set
 78                        mstSet[u] = true;
 79
 80                        // Update key value and parent index of the adjacent
 81                        // vertices of the picked vertex. Consider only those
 82                        // vertices which are not yet included in MST
 83 3                    for (int v = 0; v < V; v++)
 84
 85                                // graph[u][v] is non zero only for adjacent vertices of m
 86                                // mstSet[v] is false for vertices not yet included in MST
 87                                // Update the key only if graph[u][v] is smaller than key[v]
 88 4                            if (graph[u][v]!=0 && mstSet[v] == false &&
 89                                        graph[u][v] < key[v])
 90                                {
 91                                        parent[v] = u;
 92                                        key[v] = graph[u][v];
 93                                }
 94                }
 95
 96                // print the constructed MST
 97 1            printMST(parent, V, graph);
 98        }
 99
100        // public static void main (String[] args)
101        // {
102        //      /* Let us create the following graph
103        //      2 3
104        //      (0)--(1)--(2)
105        //      | / \ |
106        //      6| 8/ \5 |7
107        //      | /      \ |
108        //      (3)-------(4)
109        //              9               */
110        //      MST t = new MST();
111        //      int graph[][] = new int[][] {{0, 2, 0, 6, 0},
112        //                                                      {2, 0, 3, 8, 5},
113        //                                                      {0, 3, 0, 0, 7},
114        //                                                      {6, 8, 0, 0, 9},
115        //                                                      {0, 5, 7, 9, 0}};
116
117        //      // Print the solution
118        //      t.primMST(graph);
119        // }
120  }
121  // This code is contributed by Aakash Hasija
```

Mutations

[20]	1. changed conditional boundary → KILLED 2. Changed increment from 1 to -1 → KILLED 3. negated conditional → KILLED
[21]	1. changed conditional boundary → SURVIVED 2. negated conditional → KILLED 3. negated conditional → KILLED
[27]	1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
[35]	1. changed conditional boundary → KILLED 2. Changed increment from 1 to -1 → KILLED 3. negated conditional → SURVIVED
[38]	1. Replaced integer addition with subtraction → SURVIVED
[43]	1. replaced return of integer sized value with (x == 0 ? 1 : 0) → SURVIVED
[59]	1. changed conditional boundary → KILLED 2. Changed increment from 1 to -1 → KILLED 3. negated conditional → KILLED
[71]	1. changed conditional boundary → SURVIVED 2. Changed increment from 1 to -1 → KILLED 3. negated conditional → KILLED
[83]	1. changed conditional boundary → KILLED 2. Changed increment from 1 to -1 → KILLED 3. negated conditional → KILLED
[88]	1. changed conditional boundary → SURVIVED 2. negated conditional → KILLED 3. negated conditional → KILLED 4. negated conditional → KILLED
[97]	1. removed call to com/hongchuan/app/MST::printMST → SURVIVED

Active mutators

- INCREMENTS_MUTATOR
- VOID_METHOD_CALL_MUTATOR
- RETURN_VALS_MUTATOR
- MATH_MUTATOR
- NEGATE_CONDITIONALS_MUTATOR
- INVERT_NEGS_MUTATOR
- CONDITIONALS_BOUNDARY_MUTATOR

Tests examined

- com.hongchuan.app.MSTTest.testCase1(com.hongchuan.app.MSTTest) (6 ms)
- com.hongchuan.app.MSTTest.testCase2(com.hongchuan.app.MSTTest) (1 ms)

Report generated by [PIT](#) 1.4.3