https://course.acciojob.com/idle?question=b676f879-c0b8-4cf6-b2ba-6e896496cb94

- MEDIUM
- Max Score: 40 Points
-
-
-
-
-

Minimum Spanning Tree

Given a weighted, undirected and connected graph of V vertices and E edges. The task is to find the sum of weights of the edges of the Minimum Spanning Tree.

Input Format

First line contains the number of vertices `V` in the graph

Second line contains the number of edges `E` in the graph

The next `E` lines contains the edges along with their weights in the format

`source destination weight`

Output Format

Print the sum of weights of all the edges in the minimum spanning tree

Example 1

Input

```
5
7
```

```
0 1 2
0 3 6
1 2 3
1 3 8
1 4 5
2 4 7
3 4 9
```

Output

```
16
```

Explanation

Edge Weight 0 - 1 2 1 - 2 3 0 - 3 6 1 - 4 5

Example 2

Input

```
3
3
0 1 5
1 2 3
0 2 1
```

Output

```
4
```

Explanation

Edge Weight 0 - 2 1 1 - 2 3

Constraints

`2 <= V <= 1000`

`V-1 <= E <= (V*(V-1))/2`

`1 <= w <= 1000`

Topic Tags

- Graphs
- Trees

My code

```
// n java
import java.util.*;

public class Main {

  static int minNode(int[] dist, boolean[] mst) {
        int val = Integer.MAX_VALUE, p = -1;

        int n = dist.length;

        for(int i = 0 ; i < n; i++) {
            if(mst[i] == false && dist[i] < val) {
                val = dist[i];
                p = i;
            }
        }

        return p;
    }
  static int primMST(int g[][], int V) {
    // your code here
          int n = V;
          int dist[] = new int[n];
            boolean mst[] = new boolean[n];

            int parent[] = new int[n];

            for(int i = 0; i < n; i++) {
                dist[i] = Integer.MAX_VALUE;
                mst[i] = false;
            }

            dist[0] = 0;
            parent[0] = -1;

            for(int i = 0; i < n-1; i++) {
                int u = minNode(dist, mst);

                mst[u] = true;

                // for all nbrs, update parent and dist
                for(int v = 0; v < n; v++) {
                    if(mst[v] == false && g[u][v] != 0 && g[u][v] < dist[v]) {
                        dist[v] = g[u][v];
                        //parent[v] = u;
                    }
                }
            }

            int ans = 0;
            for(int i = 1 ; i < n; i++) {
                ans += dist[i];
```

```
          }

          return ans;

  }



  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    int vertices = sc.nextInt(), edges = sc.nextInt();
    int[][] graph = new int[vertices][vertices];
    for (int i = 0; i < edges; i++) {
      int src = sc.nextInt(), dest = sc.nextInt(), dist = sc.nextInt();
      graph[src][dest] = dist;
      graph[dest][src] = dist;
    }
    sc.close();

    System.out.println(primMST(graph, vertices));
  }
}
```