Disjoint Sets - Kruskal's MST (A03)

Buatlah sebuah program yang mengimplementasikan algoritma MST Kruskal. Program menerima input sebuah graph, dan mengeluarkan bobot MST-nya.

Spesifikasi Input

Input diawali dengan bilangan bulat V *(1≤ V ≤ 100,000)*, yaitu jumlah vertex dari sebuah Graph yang akan dicari MST-nya. Vertex-vertex pada graph dinomori dari 0 hingga (V-1). Kemudian input dilanjutkan dengan bilangan bulat E (V-1 ≤ E ≤ *250,000)* yang menyatakan jumlah edge dari Graph tersebut. Sebanyak E baris selanjutnya, masing-masing berisi tiga buah bilangan integer v_1 v_2 w yang menunjukan v_1 terhubung dengan v_2 dengan bobot w.

Spesifikasi Output

Output berupa sebuah bilangan bulat yang menunjukan bobot MST dari graph yang diberikan..

Contoh Input

```
9
14
0 1 4
1 2 8
2 3 7
3 4 9
4 5 10
3 5 14
2 5 4
2 8 2
8 6 6
8 7 7
6 7 1
1 7 11
0 7 8
6 5 2
```

Contoh Output

```
37
```

Hint:
Anda dapat mengimplementasikan 4 buah kelas untuk menyelesaikan permasalahan ini. Kelas-kelas tersebut adalah:

- Kelas Edge
- Kelas Graph
- Kelas DisjointSets (sama dengan kelas DisjointSets pada modul)
- Kelas Tester

```
1  public class Edge implements Comparable<Edge>{
2     private int vertex1;
3     private int vertex2;
4     private int weight;
5
6     Edge(int vertex1, int vertex2, int weight){
7        this.vertex1=vertex1;
8        this.vertex2=vertex2;
9        this.weight=weight;
10    }
11
12    public int getV1(){
13       return this.vertex1;
14    }
15
16    public int getV2(){
17       return this.vertex2;
18    }
19
20    public int getWeight(){
21       return this.weight;
22    }
23
24    //Membuat Edge dapat dibandingkan berdasarkan weight-nya
25    //Digunakan untuk sort edge pada Kelas Graph
26    public int compareTo(Edge anotherEdge){
27       int anotherEdgeWeight=((Edge) anotherEdge).getWeight();
28       return this.getWeight()-anotherEdgeWeight;
29    }
30 }
```

Kelas edge ini meng-implement *Comparable* dan di dalamnya dibuat method compareTo. Method ini akan mengembalikan nilai negatif jika weight dari edge **this** lebih kecil daripada weight dari edge **anotherEdge**. Dengan demikian kita dapat memanfaatkan Arrays.sort yang dimiliki oleh Java.

```
import java.util.Arrays;
public class Graphs{
   int maxNumOfEdge;
   int currNumOfEdge;
   int numOfVertices;
   Edge[] edges;

   Graphs(int maxEdge, int numOfVertices){

   }

   public void sortEdge(){
      Arrays.sort(this.edges);
   }

   public boolean addEdge(int v1, int v2, int weight){
   }

   public boolean addEdge(Edge e){
   }

   public int totalWeight(){
      //Method untuk menghitung bobot total
      //dalam sebuah Graph
   }

   public Graphs mstKruskal(){
      //Implementasikan algoritma MST KRUSKAL
   }
}
```