Making friends
Problem ID: 3makingfriends

Introduction

Your are a consultant of a recently started making-friend-agency. Soon you are planning to host an event where the goal is to help everyone befriend everyone else at the event. It is widely known that the friend of a friend automatically becomes your friend. Therefore it is only of importance to connect all people, once there is a friendly path (a path of friendships) between two people they will soon become friends.

Since you, as all consultants and more specifically all programmers, are lazy you want to minimize the effort you have to put in. Therefore all needed information about the participants have been collected and from this you can extract the compatability of two people. From this you have predicted how many minutes you would have to spend on introducing some pairs of people. Due to your laziness you have not predicted the number of minutes to introduce all pairs of people, only a subset of the pairs, however it is guaranteed that there is a path of predicted pairs between all pairs of people (i.e. it is possible to construct a path between each pair of people). Now the goal is to minimize your amount of work in order to connect all people.

Aims

The goals of the lab are:

- Implementing an algorithm to solve Minimal Spanning Tree. (Prim's or Kruskal's algorithms are recommended.)
- Debugging your code.
- Structuring your code in a logical fashion.
- Reason about correctness of your algorithm.
- Reason about upper bounds for time complexity.

Problem formulation

You are given a number of people and a number of undirected weighted edges between them (the weight being the number of minutes you would have to spend in order for the pair to become friends). The task is to print the number of minutes you will have to spend (in total) in order to connect all people.

Input

The first line of the input consists of two integers, N, M, the number of people at the event and the number of pairs for which you have predicted the time it would take you to make them friends. It is guaranteed that $2 \leq N \leq 10^5$ and $1 \leq M \leq \min(N \cdot (N-1)/2, 3 \cdot 10^6)$. Then follows M lines containing the description of each edge. Each of these lines contain three integers u, v, w, where $1 \leq u, v \leq N$ are the indices of the persons of the edge and $0 \leq w \leq 10^6$ is the weight of the edge, i.e. the number of minutes it would take you to make u and v friends. It is guaranteed that each pair $\{u, v\}$ occurs at most once in the input.

Output

The output should contain a single integer on a single line, the total sum of the weights of the minimal spanning tree in the graph – i.e. the miminal total number of minutes you would have to spend on connecting all people.

Examination and Points of Discussion

To pass the lab make sure you have:

- Have successfully implemented the algorithm with the correct time complexity.

- Have neat and understandable code.
- Have descriptive variable names.
- Have filled in the blanks in the report.
- Have run the check_solution script to validate your solution.

During the oral presentation you will discuss the follwoing questions with the lab assistant:

- Why does the algorithm you have implemented produce a minimal spanning tree?
- What is the time complexity, and more importantly why?
- What happens if one of the edges you have chosen to include collapses? Might there be any problems with that in real applications?
- Can you think of any real applications of MST? What would the requirements of a problem need to be in order for us to want MST as a solution?

Sample Input 1

```
3 2
1 2 3
2 3 7
```

Sample Output 1

```
10
```

Sample Input 2

```
4 4
1 2 1
2 3 4
3 4 5
1 3 7
```

Sample Output 2

```
10
```