ETH Zürich
Institute of Theoretical Computer Science
Prof. Angelika Steger, Prof. Emo Welzl
Dr. Johannes Lengler
Nemanja Škorić, Miloš Trujić

FS 2017

Algorithmen und Wahrscheinlichkeit
Programming Exercises 1

Deadline: 10:15 @ 9.3.2017

Exercise 1 – *MST*

The goal of this exercise is to implement a Minimum Spanning Tree algorithm. This will be done by solving the following problem. You are given a connected undirected *weighted* graph G on the vertex set $\{0, \ldots, n-1\}$ where each edge $\{x, y\} \in E(G)$ has a non-negative integer w assigned to it. You are required to find the weight of a subgraph of G that satisfies the following properties:

- the subgraph contains all n vertices of G,
- the subgraph contains *exactly one* path between any two vertices of the graph,
- the subgraph is of minimum total weight (sum of the weights of all edges in it) among all such subgraphs.

Input The first line of the input file contains an integer $1 \leq t \leq 20$ denoting the number of test cases that follow. Each of the t test cases is described as follows.

- It starts with a line containing two integers `n m`, separated by space, denoting the number of vertices of G and number of edges of G such that $0 \leq n \leq 10^3$ and $n - 1 \leq m \leq 10^6$.
- The next m lines contain three integers `u v w`, separated by space, indicating that $\{u, v\}$ is an edge of the graph G of weight w, where $0 \leq u < v \leq n - 1$ and $0 \leq w \leq 10^3$.

Output For each test case output one line with the weight of the subgraph described above.

Points This exercise gives 1 point.

Sample Input

```
2
3 3
0 1 5
1 2 1
0 2 1
5 7
0 1 0
1 2 3
2 3 3
3 4 1
0 4 5
0 2 2
1 4 2
```

Sample Output

```
2
5
```

Exercise 2 – *Important Stops*

You were hired by Swiss Federal Railways to do an important job. Given a map with n cities, denoted by $\{0, 1, \ldots, n-1\}$, and a map of *bi-directional* rail segments between them, your task is to decide at which stops should the passengers be checked if they have a valid ticket. Since each city has its corresponding unique stop, we refer to stops and cities interchangeably. The process of ticket control takes time and human resources, thus SBB decided that the passengers should be checked at a stop (city) i only if the stop is *very important*. A stop $a \in \{0, \ldots, n-1\}$ is very important if there exist two different cities $b, c \in \{0, \ldots, n-1\} \setminus \{a\}$ such that the following is satisfied:

- City b is reachable by train from city c (and the other way around).
- For a train to go from city b to city c it must go through city a.

Input The first line of the input file contains an integer $1 \leq t \leq 20$ denoting the number of test cases that follow. Each of the t test cases is described as follows.

- It starts with a line containing two integers `n m`, separated by space, denoting the number of stops and the number of rail segments such that $1 \leq n \leq 10^4$ and $1 \leq m \leq 10^4$.
- The next m lines describe m different rail segments. Each line consists of two integers `a b`, separated by space, which denote that there is a rail segment between a and b. It is guaranteed that $0 \leq a \neq b \leq n-1$.

Output For each test case output one line containing the id's of all the very important stops (cities) separated by space and sorted by the increasing value of the id. In case there are no important stops for a particular test case, output -1.

Points This exercise gives 1 point.

Sample Input

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

Sample Output

```
0 5
-1
```