ETH Zurich
Institut für Theoretische Informatik
Prof. Dr. Angelika Steger
Prof. Dr. Emo Welzl
Prof. Dr. Peter Widmayer

Algorithms Lab

There are n cities in AlgoLand, named very creatively with numbers from 1 to n (no two cities have the same "name"). There are, however, no roads yet between them. Since flying cars are still not available in AlgoLand, the government is forced to construct some roads. In each of the following problems you are given a list of possible roads which can be built, and based on the specific task description you have to decide which of these roads will actually be constructed.

Exercise – *Connecting Cities*

As a first step toward building a reliable infrastructure, the people of AlgoLand built a skeleton of the road network. The skeleton of the road network is just a list of possible roads, some of which will be actually built. Moreover, the list of possible roads is such that between every two cities in AlgoLand there exists exactly one path connecting them. You want to start building as many roads as possible at the same time.

There is a constraint, though: to avoid too much disruption you do not want to build two roads ending in the same city.

Input The first line contains $1 \leq T \leq 30$, the number of test cases. For each test case, in the first line there are two numbers `n m`, the number of the cities and possible roads. It is guaranteed that $m = n - 1$. The following m lines contain the descriptions of the roads, each of them consisting of two labels `s t` ($1 \leq s < t \leq n$) denoting the cities which are connected by such road.

Points There are two test sets:

1. For the first set, worth 30 points, you may assume that the total number of cities is at most 500.
2. For the second set, worth 70 points, the total number of cities is at most 10^6.

Output For each test case output a single line with the maximum number of roads that can be built at the same time. Furthermore, you can only build a road if it is in the list of possbile roads.

Sample input

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

Sample output

```
3
```

Exercise – *Consecutive Constructions*

The basic roads in Algoland are already built, but the road network still needs to be extended.

The roads to be constructed now are in a more difficult terrain, so, if possible, you want to build them in consecutive intervals. A single construction team with all the heavy equipment can start building the next road in the same city just after finishing the previous one.

The rules are as follows:

- There will be several teams of workers operating the construction.
- Each team will start in some city u, and then build a road to a city with larger number v. After finishing the construction the team will be in v. Then it might build another road to a city with even larger number, and so on.
- To avoid disruptions each city can be visited at most once by at most one team. Starting or finishing in a city counts as a visit.
- You want to build as many roads as possible.

Input The first line contains $1 \leq T \leq 30$, the number of test cases. For each test case, in the first line there are two numbers `n m` ($1 \leq n \leq 500, 0 \leq m \leq 1000$), the number of the cities and possible roads. The following m lines contain the descriptions of the roads, each of them consisting of two labels `s t` ($1 \leq s < t \leq n$).

Output For each test case output a single line with the maximum number of roads that can be built.

Sample input

```
1
7 10
1 2
1 4
2 3
2 5
4 3
4 7
3 5
3 7
5 6
7 6
```

Sample output

```
5
```

Exercise – *Missing Roads*

A lot of roads were built in Algoland, but some of them are still missing. Since many accidents happened last year at construction sites, the government decided to decrease the risk of injury by using a a separate construction team for each road. However, as the road-building equipment makes a lot of noise when it is started, no two construction teams are allowed to start building from the same city. There is no problem though in two construction teams ending the road in the same city. A team assigned for one road can *choose* from which city will it start the construction.

There is not a lot of time and all roads must be built simultaneously, but you need to pay attention to the budget. Thus, given a construction cost for each potential road, estimate the minimum

possible cost of building at least k of them, without starting a construction of more than one road at each city. Each given possible road has a cost between 1 and 100.

Input For each test case, in the first line there are `n m k` ($1 \leq n \leq 50, 0 \leq m \leq \binom{n}{2}$, $k \leq n/2$), the number of the cities, possible roads and the number of roads to be built. The following m lines contain the descriptions of the roads and the costs. Each line consists of three numbers `s t c` ($1 \leq s < t \leq n$, $1 \leq c \leq 100$), denoting that the cost of building a road between cities s and t is c.

Output For each test case output a single line with the minimum possible cost of building at least k roads such that no two roads start the construction in the same city. Output `no` if it is impossible to achieve the goal.

Points There are two test sets:

1. For the first set, worth 50 points, you may assume that all possible roads have the same cost.
2. For the second set, worth 50 points, there are no additional constraints.

Output For each test case output a single line with the maximum number of roads that can be built.

Sample input

```
1
7 12 5
1 2 2
2 3 2
1 3 1
3 4 1
2 4 2
1 4 4
1 7 10
2 6 6
4 6 8
7 6 5
6 5 15
4 5 9
```

Sample output

```
11
```

Briefly sketch your approach for each of the three exercises.

Exercise 1) — *Connecting Cities*

Method(s): ☐ Greedy ☐ DP ☐ BFS/DFS ☐ Graph components ☐ Maximum matching ☐ Shortest paths (Dijkstra) ☐ Minimum Spanning Tree ☐ Network flows ☐ MinCost flows ☐ Delaunay triangulation ☐ LP ☐ QP ☐ Backtracking ☐ Sorting ☐ Binary search

Reasoning:

Exercise 2) — *Consecutive Constructions*

Method(s): ☐ Greedy ☐ DP ☐ BFS/DFS ☐ Graph components ☐ Maximum matching ☐ Shortest paths (Dijkstra) ☐ Minimum Spanning Tree ☐ Network flows ☐ MinCost flows ☐ Delaunay triangulation ☐ LP ☐ QP ☐ Backtracking ☐ Sorting ☐ Binary search

Reasoning:

Exercise 3) — *Missing Roads*

Method(s): ☐ Greedy ☐ DP ☐ BFS/DFS ☐ Graph components ☐ Maximum matching ☐ Shortest paths (Dijkstra) ☐ Minimum Spanning Tree ☐ Network flows ☐ MinCost flows ☐ Delaunay triangulation ☐ LP ☐ QP ☐ Backtracking ☐ Sorting ☐ Binary search

Reasoning:

Please provide some feedback on the *ACM* tutorials

What is the "muddiest point" of the ACM lectures / Which part of the ACM tutorials is the most unclear to you?

Your feedback on this threefold problem set:

Your feedback on the ACM tutorials:

Which ACM Task would you really like to have discussed in one of the remaining tutorials? (Please provide only 1 ACM task.)