Technische Universität München
Fakultät für Informatik
Lehrstuhl für Effiziente Algorithmen (LEA)
Prof. Dr. Ernst W. Mayr
Christian Müller, Philipp Hoffmann, Chris Pinkau, Stefan Toman

Summer Term 2015
Problem Set 2
23.04.2015

Algorithms for Programming Contests

This problem set is due by

Thursday, 30.04.2015, 6:00 a.m.

Try to solve all the problems and submit them at

`http://judge.informatik.tu-muenchen.de/`

This week's problems are:

The following amount of points will be awarded for solving the problems.

Problem	SS15N02A	SS15N02B	SS15N02C	SS15N02D	SS15N02E
Difficulty	easy	easy	medium	medium	hard
Points	4	4	6	6	8

If the judge does not accept your solution but you are sure you solved it correctly, use the "request clarification" option. In your request, include:

- the name of the problem (by selecting it in the subject field)
- a verbose description of your approach to solve the problem
- the time you submitted the solution we should judge

We will check your submission and award you half the points if there is only a minor flaw in your code.

If you have any questions please ask by using the judge's clarification form.

SS15N02A Pizza Toppings

Author: Chris Pinkau

Lea and her friend Bea are famished. It was a long day for them, and they just arrived at home. Obviously, they decide on having pizza. But on this joyous occasion they want to make the delicious oven-baked flatbread by themselves. Fortunately, there is an abundant supply of toppings, but not all combinations of toppings taste delicious together, which is why Lea and Bea want to distribute all available toppings over exactly two pizzas.

Input

The first line of the input contains an integer t. t test cases follow, each of them separated by a blank line.

Each test case starts with two integers n and m, n being the number of available pizza toppings (indexed from 1 to n), and m being the number of pairs of toppings which do not taste well together on the same pizza. m lines follow each containing two integers a_j and b_j, the indices of two toppings which cannot be put on the same pizza together.

Output

For each test case, output one line "Case #i: x" where i is its number, starting at 1, and x is either "yes" if Lea and Bea can distribute all available toppings and bake two pizzas that both taste delicious, or "no" otherwise.

Constraints

- $1 \leq t \leq 20$
- $2 \leq n \leq 10000$
- $0 \leq m \leq 25000$
- $a_j \neq b_j$ for all $1 \leq j \leq m$

Sample Data

Input

```
3
4 2
1 2
3 4

3 3
1 2
1 3
2 3

2 1
1 2
```

Output

```
Case #1: yes
Case #2: no
Case #3: yes
```

SS15N02B Networking

Author: Philipp Hoffmann

Sometimes, Lea has to attend loud and boring parties. In the beginning of such events, everybody wants to say hello to all the other people attending. This usually creates a big mess, so this year a new system was introduced: The transitive symmetric stationary greeting system (TSSGS). The system works as follows: Greetings are now transitive, that is, if Lea greets Ralph and Ralph greets Tom, then Lea is considered to have greeted Tom (thus transitive). Greetings are also symmetric, thus if Lea greets Tom (directly or via transitivity), Tom is considered to have greeted Lea. As to further reduce work, people do not move through the room but simply shout out to persons they want to greet (thus stationary). Still everybody wants to greet everybody (possibly indirectly).

This system of course reduced the work, but now everybody was shouting through the room and it soon got very loud.

Lea wants to do one more optimization: The sound level should be as low as possible while satisfying all the constraints above. A greeting between two people is as loud as the distance between them, the sound level is the sum of all greetings that take place. Help Lea with that problem by providing the lowest sound level possible.

Input

The input begins with a number t, t test cases follow, each of them separated by a blank line. Each test case begins with an integer n, the number of people at the party, n lines follow. The i-th line consists of n integers $m_{i,j}$ where $m_{i,j}$ is the distance of person i to person j. It is always the case that $m_{i,i} = 0$ and $m_{i,j} = m_{j,i}$.

Output

For each test case, output one line containing "Case #i:" where i is its number, starting at 1.

Starting in the next line, output the greetings that take place, line by line. For each greeting, output its start person i and end person j, separated by one space, such that $i < j$. Furthermore, order your output lexicographically, that is, greeting ab should appear before greeting ij if $a < i$, or $a = i$ and $b < j$. End each output with an empty line.

Constraints

- $1 \leq t \leq 20$.
- $1 \leq n \leq 150$.
- $0 \leq m_{i,j} \leq 10000$ for all $1 \leq i, j \leq n$
- $m_{i,i} = 0$ and $m_{i,j} = m_{j,i}$ for all $1 \leq i, j \leq n$

Sample Data

Input

```
2
2
0 1
1 0

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

Output

```
Case #1:
1 2

Case #2:
1 2
2 3
```

SS15N02C Road Destruction

Author: Stefan Toman

Beachistan is a country situated on a small but warm island. Lea's aunt decided to move to Beachistan some weeks ago because she hoped to have a better life there without worrying too much about money. She got a nice job in the administration of Sandington, Beachistan's capital. Unfortunately, she was wrong: It seems like money is even more important to Beachistanians than to the people at home. On her first day at work, her colleagues asked her to help them with the following problem.

Some years ago, Beachistan's government invested heavily in the country's infrastructure, and lots of new roads were built. Afterwards, people were able to get to other cities much faster, but there was one problem: Upkeeps of the road maintenance depot exploded. Therefore, the prime minister decided to close most of the roads again. He wants to shut down as many roads as possible to cut costs, but it should still be possible to reach each city from every other city. The time needed for travelling is not important since people in Beachistan tend to have lots of free time. Because he wants to be reelected in some years, he needs some nice numbers to talk about and therefore wants to close roads in such a way that the sum of the remaining roads' capacities is as big as possible. Tell him the total capacity of the roads he needs to shut down.

Input

The first line of the input contains an integer t. t test cases follow, each of them separated by a blank line.

Each test case starts with a single line containing two integers n and m. n is the number of cities in Beachistan (called city 1 to city n) and m is the number of roads between them. m lines follow describing the roads. The i-th line consists of four integers a_i, b_i, x_i, and l_i meaning that the i-th road goes from city a_i to city b_i, its length is x_i and it has l_i lanes. All roads may be used in both directions. The capacity of a road is the product of its length and the number of lanes.

Output

For each test case, output one line containing "Case #i: x" where i is its number, starting at 1, and x is the sum of the capacities of the roads to be closed or the string "impossible" if there is no way to close roads in a way that each city is still reachable from each other one. In particular, if no road should be closed but the road network is still connecting all cities with each other, output 0.

Constraints

- $1 \leq t \leq 20$

- $1 \leq n \leq 1000$
- $1 \leq m \leq 10000$
- $1 \leq a_i, b_i \leq n$ for all $1 \leq i \leq m$
- $1 \leq x_i \leq 100$ for all $1 \leq i \leq m$
- $1 \leq l_i \leq 5$ for all $1 \leq i \leq m$

Sample Data

Input

```
3
3 3
1 2 100 1
1 3 50 3
2 3 60 3

4 3
1 2 20 2
1 3 30 4
1 4 50 1

3 1
1 2 3 4
```

Output

```
Case #1: 100
Case #2: 0
Case #3: impossible
```

SS15N02D Technical Difficulties

Author: Philipp Hoffmann

Lea is on vacation and this time, she takes a city tour. She has already planned all the things that she wants to visit. Unfortunately, the public transport provider is not very reliable. Every day there are technical difficulties. In fact, exactly one connection is not usable on each day.

Lea is worried that this might make her trip impossible. For her plan to work, it must be possible to go from every station to all other stations. So from now on, every morning she looks up which connection is not working that day. But she still does not know whether this connection was needed or not! Can you provide her with a list of connections that, if not usable, make it impossible to reach every station from every other station?

Input

The first line of the input contains an integer t. t test cases follow.

Each test case starts with two integers n and m, the number of stations and connections in the public transportation network. m lines follow, describing the connections. The i-th line consists of two integers u_i and v_i, the two stations connected by the i-th connection. Connections are undirected.

Output

For each test case, print a line containing "Case #i: x" where i is its number, starting at 1, and x is a sorted, space-separated list of the indices of connections that make Lea's trip impossible if they are not usable. Each line of the output should end with a line break.

Constraints

- $1 \leq t \leq 20$.
- $1 \leq n \leq m + 1 \leq 10000$.
- $1 \leq u_i, v_i \leq n$ for all $1 \leq i \leq m$.
- The public transportation network will be connected.
- There is at most one connection between each pair of stations.
- No station is connected to itself.

Sample Data

Input

```
3
4 4
1 2
2 3
3 4
2 4

4 3
1 2
2 3
3 4

3 3
1 2
2 3
3 1
```

Output

```
Case #1: 1
Case #2: 1 2 3
Case #3:
```

SS15N02E Walking Straight Into Circles

Author: Stefan Toman

Jan L. Souman, Ilja Frissen, Manish N. Sreenivasa, and Marc O. Ernst made a survey testing whether people are able to walk in straight lines for a longer time. They left some people in a forest and some others in the Sahara desert with the task of walking in one direction for several hours. The locations of the participants were monitored by GPS. Some of the participants managed to walk in perfectly straight lines while other walked circles repeatedly. It turned out that people were able to walk straight if they can see the sun, but walk in circles on cloudy days.

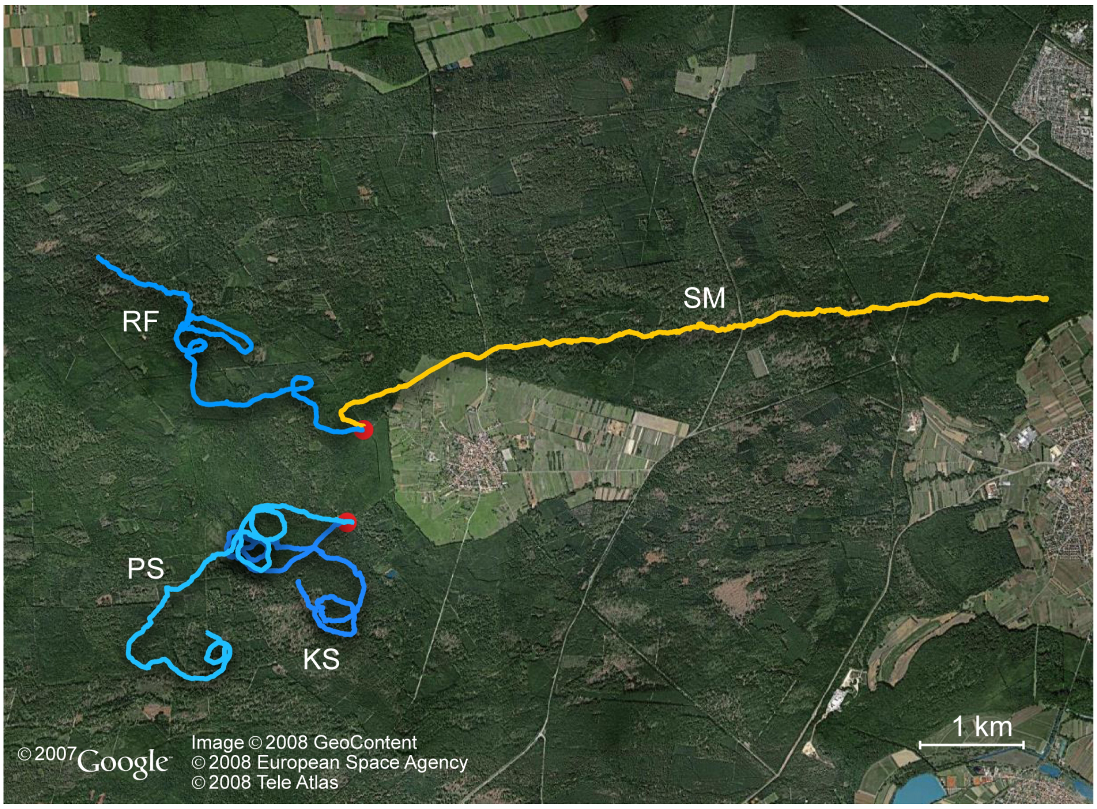

Figure 1: Paths of the participants. Yellow paths indicate that the sun was visisble, blue paths indicate clouds.

Jan Souman commented the results in the following way:

> “ *One explanation offered in the past for walking in circles is that most people have one leg longer or stronger than the other, which would produce a systematic bias in one direction. To test this explanation, we instructed people to walk straight while blindfolded, thus removing the effects of vision. Most of the participants in the study walked in circles, sometimes in extremely small ones (diameter less than 20 m).*
>
> *However, it turned out that these circles were rarely in a systematic direction. Instead, the same person sometimes veered to the left, sometimes to the right. Walking in circles is therefore not caused by differences in leg length or strength, but more likely the result of increasing*

uncertainty about where straight ahead is. Small random errors in the various sensory signals that provide information about walking direction add up over time, making what a person perceives to be straight ahead drift away from the true straight ahead direction ”

Here is another comment by Marc Ernst, Group Leader at the MPI for Biological Cybernetics:

“ *The results from these experiments show that even though people may be convinced that they are walking in a straight line, their perception is not always reliable. Additional, more cognitive, strategies are necessary to really walk in a straight line. People need to use reliable cues for walking direction in their environment, for example a tower or mountain in the distance, or the position of the sun.* ”

This is one of the problems were the story is (so far) not totally made up, but actually true. You can check the reference “Jan L. Souman, Ilja Frissen, Manish N. Sreenivasa & Marc O. Ernst: Walking straight into circles. Current Biology, 2009, doi:10.1016/j.cub.2009.07.053” at `http://www.sciencedirect.com/science/article/pii/S0960982209014791`.

Lea worked together with these researchers as a research assistant in 2009. One of the things not mentioned in this paper is that they actually left more participants in the woods. One of them is not displayed on the map since there was a slight problem with the GPS receiver. Unfortunately, this was one of the participants starting on a cloudy day and after one week he still did not show up in the institute again. Probably, he is still walking in circles somewhere in the forest.

Lea got the job of organizing the search for the lost participant. The plan is to put some researchers into the forest and wait for the participant. Since Lea does not want to lose more people in the woods, the researchers are not allowed to move. But were should they be left to minimize the number of researchers needed but still guarantee that the participant walking some circle in the forest will be seen by one of them?

Input

The first line of the input contains an integer t. t test cases follow.

Each test case starts with a line containing two space-separated integers n and m. n is the number of junctions in the forest and m is the number of paths. m lines follow describing the paths. The i-th line consists of three space-separated integers a_i, b_i, and c_i, meaning that there is a path from junction a_i to junction b_i which can be guarded by c_i researchers. Since the participant wants to find his way out of the woods, he will only use paths in the forest. He will furthermore always walk along some circle.

Output

For each test case, print a line containing “Case #i: x” where i is its number, starting at 1 and x is the minimal number of researchers needed. If there is no circle in the forest, the participant has obviously left it and went home: x should be 0 in this case. Each line of the output should end with a line break.

Constraints

- $1 \leq t \leq 20$
- $2 \leq n \leq 10000$
- $1 \leq m \leq 100000$
- $1 \leq a_i, b_i \leq n$ for all $1 \leq i \leq m$
- $1 \leq c_i \leq 100$ for all $1 \leq i \leq m$

Sample Data

Input

```
2
4 5
1 2 3
2 3 5
3 4 1
4 1 9
2 4 5

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

Output

```
Case #1: 4
Case #2: 6
```