ALGORITHMS FOR PROGRAMMING CONTESTS
WS15/16 - Week 11

Chair for Efficient Algorithms (LEA), TU München

Prof. Dr. Harald Räcke

Moritz Fuchs, Philipp Hoffmann, Christian Müller, Chris Pinkau, Stefan Toman

This problem set is due by

Wednesday, 13.01.2016, 6:00 a.m.

Try to solve all the problems and submit them at

https://judge.in.tum.de/conpra/

This week's problems are:

All students are invited to join this contest. If you do not have an account yet, register on the website given above or write a message to conpra@in.tum.de, we have additional accounts.

Sample solutions, statistics and small prices for the winners will be given on Wednesday, 13.01.2016, at 12:00 p.m. in room MI 00.08.038.

There will be 6 points awarded for each problem solved for contestants of the practical course. We will only use the usual 28 points for computing the total number of points, everything else is a bonus.

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.

This page is intentionally left blank.

Problem A
Rescue Mission

Back at home, Lea watches a suspense-packed film about an ancient clan of ninjas. Soon, the main character, "Thunderfist" Shen, is in dire trouble: all other ninjas of his clan have been captured by the evil ninja Karai. Now, he has to infiltrate the enemy base to free them.

The enemy fortress is a twisted maze of winding corridors and dark alleyways, overshadowed by looming watch-towers. Scattered throughout the enemy base are several dungeons where his friends are kept. Whenever a ninja reaches one of the dungeons, he can free all ninjas there. When freed, they either stay where they are and go into hiding or an arbitrary amount of them helps to free the other ninjas. Once all ninjas have been freed, they can rise up and kill all the guards, but until then they have to stay undetected by all means. So, to move through the evil fortress, a ninja has to distract the guards along the way or silently sneak past them. Remember, ninjas always work alone, so if two ninjas sneak through the same corridor, both have to distract the guards independently.

Every time one of the ninjas has to sneak past a guard, there is a chance that the guard raises an alarm and the ninjas' plan is foiled.

Lea wants to know if she can plan the rescue better than Shen did in the film. Can you tell her the minimum total amount of times the ninjas have to sneak past guards undetected to free all captives?

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 consists of a line containing three integers n, m, and d, where n is the number of rooms in the enemy base (indexed from 1 to n), m the number of connections between those rooms and d is the amount of dungeons. Shen starts in room 1. d lines follow. The i-th line contains an integer v_i which denotes that room v_i is a dungeon that contains captured ninjas. m lines follow. The j-th line contains three integers v_j, w_j and g_j. v_j and w_j each denote a room in the enemy base, g_j is the amount of times a ninja has to sneak past a guard to move along that corridor.

Output

For each test case, output one line containing "Case #i: x" where i is its number, starting at 1, and x is the minimum amount of times the ninjas have to sneak past guards. Each line of the output should end with a line break.

Constraints

- $1 \leq t \leq 50$
- $1 \leq d \leq n \leq 500$
- $1 \leq m \leq 125000$
- $1 \leq v_i, v_j, w_j \leq 500$ for all $1 \leq i, j \leq d$
- $1 \leq g_i, g_j \leq 100$
- The graph is connected.
- There are more than d captives in each dungeon.

Sample Input

```
2
5 5 2
2
4
1 2 8
1 3 1
3 4 1
4 5 2
2 5 2

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

Sample Output

```
Case #1: 6
Case #2: 8
```

Sample Input

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

8 10 7
7
5
6
1
3
2
4
8 1 2
1 3 1
3 6 7
8 4 6
6 2 6
4 5 1
1 7 3
1 8 2
6 5 10
7 7 8
```

Sample Output

```
Case #1: 25
Case #2: 26
```

Problem B
Scout the Castle

When the weather cools and winter is around the corner, times are hard. Especially when there is war and a new king pops up at every occasion. Right now, King Davos is planning to take the capital for himself. He just needs to find the right angle of attack, lest he be trapped and burned alive or worse.

The capital is surrounded by water to the east and by land on the remaining three sides. King Davos, afraid of water, because you never know what lurks beneath, plans to attack the city from land. He and his army are north of the city walls right now. To get a better picture, he wants to send scouts around the city. However, he needs to be careful not to be discovered.

The city is surrounded by various villages and some impassable terrain like mountains. He has drawn a map of the city and the terrain around it. The scouts should now circle the city without getting too far away (otherwise they won't be able to scout anything). However, to not draw attention, at most one scout may pass through every village.

Since King Davos has never been good with staying undetected, he sends a raven with the map and his instructions to his advisors. By some mistake, the raven ends up with Lea who is instantly intrigued and tries to figure out the number of scouts King Davos can send.

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 three integers w h and n, the width w and height h of the map which is a grid, and the number of impassable locations n. All following coordinates are 1-based.

The next line contains four integers c_x c_y c_w c_h meaning that the city occupies a square starting at grid position (c_x, c_y) of width c_w and height c_h in the grid, extending to the right and downward.

n lines follow, describing the impassable locations. Each line contains four integers x_i y_i w_i h_i, describing an impassable square starting at grid position (x_i, y_i) of width w_i and height h_i in the grid, extending to the right and downward.

The scouts will sneak along the shore until they enter the grid from the east, above the city. They may leave the grid again on the east side, below the city. They can move horizontally or vertically between passable terrain, but may of course not enter the city. Each passable grid location corresponds to a village, i.e. may only be passed by at most one scout.

Output

For each test case, print a line containing "Case #i: x" where i is its number, starting at 1, and x is the maximal number of scouts that King Davos can send without being detected. Each line of the output should end with a line break.

Constraints

- $1 \le t \le 20$
- $2 \le w \le 5000$
- $3 \le h \le 5000$
- $0 \le n \le 100$
- $1 \le c_x, c_y \le 51$
- $c_x < c_x + c_w = w + 1$
- $c_y < c_y + c_h < h + 1$
- $1 \le h - (c_y + c_h - 1) \le 50$
- $1 \le x_i < x_i + w_i \le w + 1$ for all $1 \le i \le n$.

- $1 \le y_i < y_i + h_i \le h + 1$ for all $1 \le i \le n$.
- An impassable location will never overlap with the city or another impassable location.

Sample Input

```
3
5 6 1
3 4 3 1
1 1 1 1

4 6 4
3 3 2 2
3 2 2 1
2 3 1 1
1 5 1 1
3 6 2 1

8 10 5
8 3 1 6
2 2 5 1
2 3 1 3
6 3 1 1
4 4 1 1
4 5 4 1
```

Sample Output

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

Sample Input

```
7
7 7 0
4 4 4 3

4 7 0
3 2 2 4

7 6 0
4 3 4 2

3 7 2
2 4 2 3
1 7 1 1
2 1 2 3

12 12 1
5 5 8 6
4 3 4 1

11 9 2
5 3 7 4
7 8 4 1
9 7 2 1

5 8 2
3 2 3 4
4 8 2 1
2 3 1 3
```

Sample Output

```
Case #1: 1
Case #2: 1
Case #3: 2
Case #4: 0
Case #5: 2
Case #6: 1
Case #7: 1
```

Problem C
Students

There are some changes going on at the TUM (Thomas Underwood University Markistan). Students have been rebelling all over the campus: lecture halls have been barricaded, mensa food has been refused, chairs have been tipped over... Lea has been following the news about the situation in Markistan with great interest. When she found out what the actual reason behind all this ruckus is, she wanted to help because it seemed to her that the problem could be solved with ease: all German documents published by the TUM must no longer use the word "`student`", but rather "`studierender`". The same is true for "`studentin`"/"`studierende`" and "`studenten`"/"`studierende`". So, Lea now wants to write a short program to do this, but to save precious time and work, she decides to only replace the patterns: "`entin`" will be replaced by "`ierende`", "`enten`" by "`ierende`", and "`ent`" by "`ierender`". (In particular, this means that "`enten`" should not be replaced by "`ierenderen`", and "`entin`" should not be replaced by "`ierenderin`".) Please help Lea and the TUM so that all students may once again attend their courses peacefully.

Input

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

Each test case starts with an integer n, the number of lines of the text. n lines follow, containing the text.

Output

For each test case, output "Case #i:" where i is its number, starting at 1. Beginning in the next line, output the input text modified by replacing the above mentioned patterns.

Constraints

- $t = 1$
- $1 \leq n \leq 1000$
- The input text contains at most 100000 characters.
- The input text contains only the lowercase alpha-numeric characters "a" to "z" or "0" to "9", or ".", ",", ":", "!", "?", "-", a space, or a line break.

Sample Input

```
1
3
ich bin ein student
a potent element went through a vent
ein zentner enten enthaelt keine studenten
```

Sample Output

```
Case #1:
ich bin ein studierender
a potierender elemierender wierender through a vierender
ein zierenderner ierende ierenderhaelt keine studierende
```

This page is intentionally left blank.

Problem D
Jetpack Jumps

Recently, Lea found a new game on the app store of her choice - "Jetpack Jumps". Since then, she spent countless hours jumping from platform to platform with her jetpack.

It works like this: Lea starts on the leftmost of an infinite series of platforms. She now jumps to the right one platform at a time, trying to get as far as possible (and break all the highscores) without falling into the bottomless pit between the platforms.

The distance between the platforms always increases by 1 meter, i.e. the second platform comes 1 meter after the first, the third comes 2 meters after the second, and so on. Lea herself can only jump a single meter, but can extend her jump by using her jetpack. With the jetpack, Lea can jump as far as she wants to. However, every meter that she uses the jetpack uses up fuel - the longer the jump, the more fuel she needs. Thus, for a jump of x meters, she needs $(x-1)^2$ liters of fuel (since she has to accelerate upwards first).

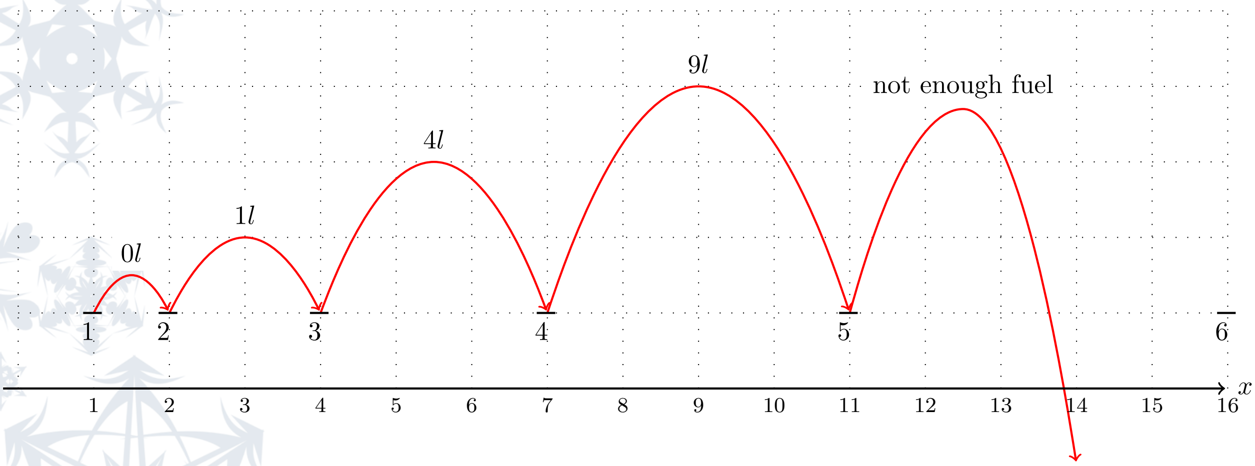

Figure D.1: Illustration of the sample input, case 2.

Lea starts with a set amount of fuel. Assuming she plays perfectly, can you tell her which platform she can reach before she eventually runs out of fuel and tumbles down into the darkness (or has to buy the DLC for additional fuel)?

Input

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

Each test case consists of a single line containing an integer f, the amount of fuel Lea starts with (in liters).

Output

For each test case, print a line containing "Case #i: x" where i is its number, starting at 1 and x is the index of the platform Lea can reach (Lea starts on platform 1). Each line of the output should end with a line break.

Constraints

- $1 \leq t \leq 20$.
- $1 \leq f \leq 10^{24}$

Sample Input	Sample Output
4 5 15 35 100	Case #1: 4 Case #2: 5 Case #3: 6 Case #4: 8

Sample Input	Sample Output
8 64 6 26 53 14 87 1 5	Case #1: 7 Case #2: 4 Case #3: 5 Case #4: 6 Case #5: 5 Case #6: 7 Case #7: 3 Case #8: 4

Problem E
Beer Pipes

After a stressful work day, Lea enjoys a nice cold beverage while sitting on her couch in front of the TV. Like most of the people from the region she comes from, she usually enjoys a beer on these occasions. And after having a few sips of the exquisite golden liquid, she contemplates the work that is put behind brewing such a masterpiece. Thus, she decides to visit the **BIER** (**B**rewery of **I**nternational **E**xcellence and **R**elevance), one of the many local breweries, on the next day to learn a bit more about the process behind her favourite beverage. Apart from all the usual brewery tour, she meets Mr. Barley Hops, the CEO of **BIER**. Recognising Lea, he says (in a heavy German accent) "Guten Tag my dear Fräulein Lea. I have heard about you and your problem solving skills, maybe you can help us? The workers installed new pipes for the Bier. They were so drunk, every pipe has a different shape und we don't know how much Bier we can pump into the pipes." Lea immediately sees the problem: the beer is poured into a pipe on one end of the brewery and exits at one valve at the other end in a great cauldron. In between there is a whole system of pipes that are connected in a seemingly chaotic fashion and are all shaped very differently. The question at stake is to come up with the highest amount of beer that can be put through the system so that the beer cauldron at the end is filled with as much beer as possible. Unfortunately, Lea is very busy right now, so she wants you to take a look at the problem. Make sure you can help Mr. Hops because he will grant you a lifetime supply of **BIER** beer if your solution is optimal!

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 valves $\{v_1, \ldots, v_n\}$ that connect the pipes, and m the number of pipes in the system. m lines follow where line i consists of three integers a_i, b_i and k_i, and a double x_i, denoting that there is a pipe that connects the valves v_{a_i} and v_{b_i} whose cross section has the shape of a regular polygon with k_i sides and side length x_i. If k_i is equal to 0, then the pipe is cylindrical with radius x_i.

The maximal amount of beer that can flow through a pipe is measured by the area of its cross section.

The first valve, where the beer enters the pipe system, is v_1, the exit of the pipe system, at the large beer cauldron, is v_n.

Beer in the pipes can flow in both directions.

Output

For each test case, output one line containing "Case #i: y" where i is its number, starting at 1, and y is either the maximal amount of beer that can be poured into v_1 with an absolute error of up to 10^{-8}, or "impossible" if that amount is 0.

Constraints

- $1 \leq t \leq 20$
- $3 \leq n \leq 1000$
- $1 \leq m \leq 2000$
- $1 \leq a_i, b_i \leq n$ for all $1 \leq i \leq m$
- $3 \leq k_i \leq 20$ or $k_i = 0$ for all $1 \leq i \leq m$
- $1 < x_i \leq 100$ for all $1 \leq i \leq m$

Sample Input	Sample Output
2 4 5 1 2 0 3.3 1 3 3 1.5 2 3 0 2.2 2 4 5 4.1 3 4 4 2.5 3 2 1 2 0 1.2 1 2 4 2	Case #1: 35.17122510390053 Case #2: impossible

Sample Input	Sample Output
3 7 9 4 1 3 7.1373647190400025 1 2 4 3.619952082374767 7 4 4 3.9229060952100747 1 5 5 5.149342554193151 6 7 0 5.9289433379154115 7 3 3 4.480317767968532 5 7 4 9.175216600972856 3 5 3 5.809582781390198 3 5 6 1.5833251223822389 10 10 9 1 0 6.103567585805486 2 9 4 1.2239623087641212 3 3 3 5.74365487996366 4 3 5 5.509203206150823 5 1 5 4.490092275486687 1 5 3 3.8459000369208165 5 6 0 3.254413686940226 2 1 4 5.257332360634563 1 1 3 3.274553532434247 6 7 4 3.260082839139213 10 5 4 8 5 7.461085908138405 8 10 0 8.000823060758666 8 1 6 4.696393660473783 1 9 6 6.339265904299603 8 3 4 8.535902037766256	Case #1: 61.00890428988396 Case #2: impossible Case #3: 57.30346357618359