A. Poker Shuffle

Description

Jason is not only an ACMer, but also a poker nerd. He is able to do a perfect shuffle. In a perfect shuffle, the deck containing K cards, where K is an even number, is split into equal halves of K/2 cards which are then pushed together in a certain way so as to make them perfectly interweave. Suppose the order of the cards is (1, 2, 3, 4, …, K-3, K-2, K-1, K). After a perfect shuffle, the order of the cards will be (1, 3, …, K-3, K-1, 2, 4, …, K-2, K) or (2, 4, …, K-2, K, 1, 3, …, K-3, K-1).

Suppose K=2^N and the order of the cards is (1, 2, 3, …, K-2, K-1, K) in the beginning, is it possible that the A-th card is X and the B-th card is Y after several perfect shuffles?

Input

Input to this problem will begin with a line containing a single integer T indicating the number of datasets.

Each case contains five integer, N, A, X, B, Y. 1 <= N <= 1000, 1 <= A, B, X, Y <= 2^N.

Output

For each input case, output “Yes” if it is possible that the A-th card is X and the B-th card is Y after several perfect shuffles, otherwise “No”.

Sample Input

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

Sample Output

```
Case 1: Yes
Case 2: Yes
Case 3: No
```

B. Good firewall

Description

Professor X is an expert in network security. These days, X is planning to build a powerful network firewall, which is called Good Firewall (a.k.a., GFW). Network flows enter in the GFW will be forwarded or dropped according to pre-established forwarding policies.

Basically, a forwarding policy **P** is a list of IP subnets, {ip_subnet_1, …, ip_subnet_n}. If **P** is enabled in GFW, a network flow **F** with source and destination IP address both located in **P** can be accepted and forwarded by GFW, otherwise F will be dropped by GFW.

You may know that, an IP address is a 32-bit identifier in the Internet, and can be written as four 0~255 decimals. For example, IP address **01111011.00101101.00000110.01001110** can be expressed as **123.45.6.78**. An IP subnet is a block of adjacent IP address with the same binary prefix, and can be written as the first IP address in its address block together with the length of common bit prefix. For example, IP subnet **01111011.00101101.00000100.00000000/22** (**123.45.4.0/22**) is an IP subnet containing 1024 IP addresses, starting from 123.45.4.0 to 123.45.7.255. If an IP address is in the range of an IP subnet, we say that the IP address is located in the IP subnet. And if an IP address is located in any IP subnet(s) in a policy **P**, we say that the IP address is located in the policy **P**.

How will you design the GFW, if you take charge of the plan?

Input

The input file contains no more than 32768 lines. Each line is in one of the following three formats:

E id n ip_subnet_1 ip_subnet_2 … ip_subnet_n
D id
F ip_src ip_dst

The first line means that a network policy $\mathbf{P}_{\mathbf{id}}$ (1<=id<=1024) is enabled in GFW, and there are **n** (1<=n <=15) IP subnets in $\mathbf{P}_{\mathbf{id}}$. The second line means that policy $\mathbf{P}_{\mathbf{id}}$ (which is already enabled at least once) is disabled in GFW. The last line means that a network flow with source and destination IP address is entered in GFW, and you need to figure out whether GFW is going to forward (F) or drop (D) this flow:

1. If the source and destination IP address both are located in one of enabled policy group $\mathbf{P}_{\mathbf{id}}$, GFW will forward this flow.
2. Otherwise GFW will drop this flow. That is, if the source or destination IP address is not located in any of enabled policy group, or they are only located in different enabled policy group(s), GFW will drop it.

IP subnets can be overlapped. An IP address may or may not be located in any policy group, and can also be located in multiple policy groups.

In the global routing table, most of the IP subnets have at least 2^8 IP addresses, and at most 2^24 IP addresses. In our dataset, every IP subnet has a prefix length between 8 and 24.

Output

For each ‘F’ operation, output a single ‘F’ (forward) or ‘D’ (drop) in a single line. Just see the sample output for more detail.

Sample Input

```
E 1 2 123.45.4.0/22 123.45.8.0/22
F 123.45.4.1 123.45.8.1
F 123.45.8.1 123.45.4.1
E 2 1 123.45.6.0/24
D 1
F 123.45.6.123 123.45.6.234
F 123.45.8.1 123.45.4.1
```

Sample Output

```
F
F
F
D
```

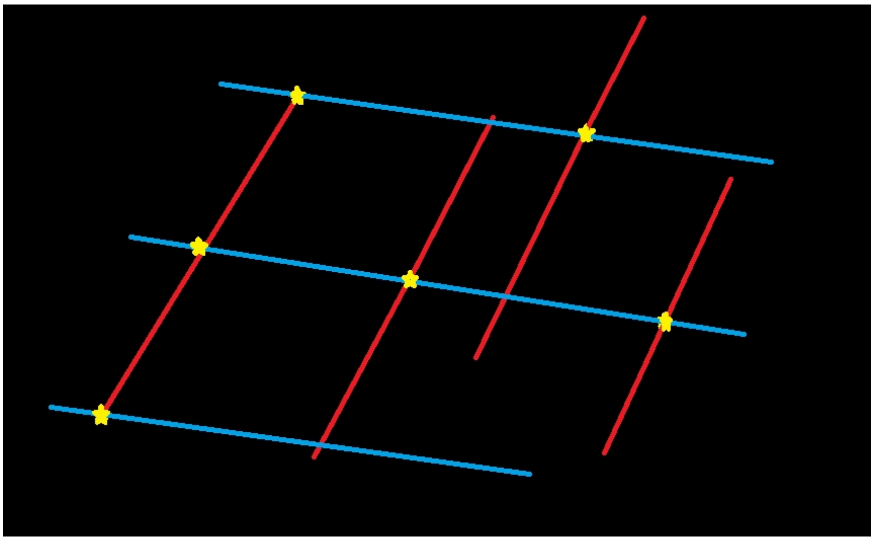
C. Sky

Description

Little Qiang is watching the night sky after completing his math homework. After staring boringly at the sky for a while, he observes that actually many stars are lying on identical lines, so that a few number of parallel lines are suffice to cover all the stars in the sky. Being passionate about both astronomy and math, he decides to find out the minimum number of parallel lines to achieve the full coverage of stars. Moreover, the stars appear and disappear from time to time, making the problem more interesting.

Specifically, we assume that the stars are numbered from 1 to N. Initially, there are no stars in the sky. Then the sky will be refreshed for M rounds. For each round, one star will appear/disappear, and Little Qiang will compute again the result. Note that the same star could appear/disappear for more than once.

Unfortunately, Little Qiang has Trypophobia such that he cannot insist looking at the sky for a long period to finish the entire task. What a pity! Can you help him?



(The figure shows a concrete snapshot of the sky, where 3 parallel lines (the blue lines) are sufficed to cover all the 6 stars.)

Input

There are multiple test cases. For each test case, there will be two integers N (1<=N<=500) and M (1<=M<=1000) in the first line as stated above. Each of the following N lines has two integers x and y (-1000000 < x, y < 1000000), indicating the coordinates of each star (from 1 to N). Then M lines follows, representing M rounds of refreshes of the sky. Each of the M lines has two integers a and b, where a = 0/1 means that b-th (1<=b<=N) star appears/disappears in the sky. Your program should proceed to the end of file.

You can assume that an appeared star would not appear again before its next disappearance, and a disappeared star would not disappear again before its next appearance.

Output

For each test case, output M lines, where each line contains an integer describing the required result. Please output a blank line after each case.

Sample Input

```
4 6
0 0
0 1
0 2
1 1
0 1
0 3
0 2
0 4
1 2
1 1
```

Sample Output

```
1
1
1
2
2
1
```

D. Cut the cake

Description

MMM got a big big big cake, and invited all her M friends to eat the cake together. Surprisingly one of her friends HZ took some (N) strawberries which MMM likes very much to decorate the cake (of course they also eat strawberries, not just for decoration). HZ is in charge of the decoration, and he thinks that it's not a big deal that he put the strawberries on the cake randomly one by one. After that, MMM would cut the cake into M pieces of sector with equal size and shape (the last one came to the party will have no cake to eat), and choose one piece first. MMM wants to know the probability that she can get all N strawberries, can you help her? As the cake is so big, all strawberries on it could be treat as points.

Input

First line is the integer T, which means there are T cases.

For each case, two integers M, N indicate the number of her friends and the number of strawberry.

(2 < M, N <= 20, T <= 400)

Output

As the probability could be very small, you should output the probability in the form of a fraction in lowest terms. For each case, output the probability in a single line. Please see the sample for more details.

Sample Input

```
2
3  3
3  4
```

Sample Output

```
1/3
4/27
```

E. Theme section

Description

It's time for music! A lot of popular musicians are invited to join us in the music festival. Each of them will play one of their representative songs. To make the programs more interesting and challenging, the hosts are going to add some constraints to the rhythm of the songs, i.e., each song is required to have a 'theme section'. The theme section shall be played at the beginning, the middle, and the end of each song. More specifically, given a theme section E, the song will be in the format of 'EAEBE', where section A and section B could have arbitrary number of notes. Note that there are 26 types of notes, denoted by lower case letters 'a' - 'z'.

To get well prepared for the festival, the hosts want to know the maximum possible length of the theme section of each song. Can you help us?

Input

The integer N in the first line denotes the total number of songs in the festival. Each of the following N lines consists of one string, indicating the notes of the i-th (1 <= i <= N) song. The length of the string will not exceed 10^6.

Output

There will be N lines in the output, where the i-th line denotes the maximum possible length of the theme section of the i-th song.

Sample Input

```
5
xy
abc
aaa
aaaaba
aaxoaaaaa
```

Sample Output

```
0
0
1
1
2
```

F. Stone

Description

Tang and Jiang are good friends. To decide whose treat it is for dinner, they are playing a game. Specifically, Tang and Jiang will alternatively write numbers (integers) on a white board. Tang writes first, then Jiang, then again Tang, etc... Moreover, assuming that the number written in the previous round is X, the next person who plays should write a number Y such that 1 <= Y - X <= k. The person who writes a number no smaller than N first will lose the game. Note that in the first round, Tang can write a number only within range [1, k] (both inclusive). You can assume that Tang and Jiang will always be playing optimally, as they are both very smart students.

Input

There are multiple test cases. For each test case, there will be one line of input having two integers N (0 < N <= 10^8) and k (0 < k <= 100). Input terminates when both N and k are zero.

Output

For each case, print the winner's name in a single line.

Sample Input

```
1 1
30 3
10 2
0 0
```

Sample Output

```
Jiang
Tang
Jiang
```

G. Tsp

Description

MGG is a poor truckman. One day he is asked to deliver packages for customers. There are n customers, where each customer specifies a location to pick up his/her package, and a location to deliver the package. Customers are labeled from 1 to n. For customer i, we denote the pickup location as i+, and the delivery location as i-. To deliver package for customer i, Mo must visit i+ before visiting i-.

However, the truck Mo drives has only a rear door, so the truck works as a stack: the last picked up package must be delivered first. If there are two packages i and j in the truck, and i is picked up before j, then i cannot be delivered unless j is delivered.

Mo knows all the coordinates of pickup and delivery locations. However, due to the censorship of the Great Fire Wall, Mo cannot visit any location more than once. What's worse, there are additional restrictions:

- Mo can't go to location j+ from i+ if j > i;
- Mo can't go to location j- from i- if j < i;
- If Mo has visited location i-, location j+ will be removed from the world and cannot be visited any more if j < i.

Now Mo can choose any location to start. He wants to choose a shortest path to deliver packages for all the n customers. Please find the shortest path for him.

Input

There are multiple test cases.
For each test case, the first line is an integer n, 0 < n <= 100.
Then n lines follow. There are 4 integers in each line. The integers in the i-th line indicate the x and y coordinates of i+, and the x and y coordinates of i-, respectively.

The distance between two locations is the Euclidean distance between them.

Input is terminated by end-of-file.

Output

For each test case, output the path in a line.
It is guaranteed that there is a unique solution for each test case.

Sample Input

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

```
5
5 0 6 0
2 0 5 0
7 0 1 0
6 0 9 0
4 0 6 0
```

Sample Output

```
1+ 1- 3+ 2+ 2- 3-
2+ 1+ 1- 2- 3+ 3- 5+ 4+ 4- 5-
```

H. Network

Description

With the help of the Ares L, the Emperor of Law and some stars of the future, General M succeeded in setting foot on the mountain, and had set up a kingdom, known as ACMSYSU. In order to reward these founding heroes, General M decided to share her network.

General M has manufactured a router. One can use this network if the Euclidean distance between his/her house and the router is not greater than d. Because of General M's excellent technology, the distance between her house and the router can be infinite.

There are n heroes in the kingdom besides General M. General M is not an eccentric person, so she hopes everyone can use the network. Because the farther the router is away from her house, the worse the signal, General M wants the router being as close to her house as possible. Please help General M place the router so that the distance between her house and the router is minimized, and every one can use the router. You just need to tell General M the minimum distance.

Input

The input will consist of multiple test cases.

For each case, the first line contains 3 integers x, y and d, which represent that the General M's house locates at the point (x, y) and one can use the network if his/her distance to the router is not greater than d.

The second line is a positive integer n, the number of heroes of the kingdom.

Then n lines follow. Each line has 2 integers x_i and y_i, indicating that the i-th hero locates at point (x_i, y_i)。

All integers in the input satisfy that their absolute values are less than or equal to 10^5, and 1<=n<=10^3.

Output

For each case, output a real number which indicates the minimum distance between the General M's house and the router, to two decimal places.

If it is impossible to share a network which can cover everyone, output an 'X' (without the single quote).

Sample Input

```
0 0 5
2
5 3
5 -3
0 0 1
2
5 3
5 -3
```

Sample Output

```
1.00
X
```

I. Bell

Description

What? MMM is learning Combinatorics!?

Looks like she is playing with the bell sequence now:

bell[n] = number of ways to partition of the set {1, 2, 3, ..., n}

e.g. n = 3:

{1, 2, 3}

{1} {2 3}

{1, 2} {3}

{1, 3} {2}

{1} {2} {3}

so, bell[3] = 5

MMM wonders how to compute the bell sequence efficiently.

Input

T -- number of cases

for each case:

    n (1 <= n < 2^31)

Output

for each case:

    bell[n] % 95041567

Sample Input

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

Sample Output

```
1
2
5
15
52
203
```

J. Flyer

Description

The new semester begins! Different kinds of student societies are all trying to advertise themselves, by giving flyers to the students for introducing the society. However, due to the fund shortage, the flyers of a society can only be distributed to a part of the students. There are too many, too many students in our university, labeled from 1 to 2^32. And there are totally N student societies, where the i-th society will deliver flyers to the students with label A_i, A_i+C_i,A_i+2*C_i,⋯A_i+k*C_i (A_i+k*C_i<=B_i, A_i+(k+1)*C_i>B_i). We call a student "unlucky" if he/she gets odd pieces of flyers. Unfortunately, not everyone is lucky. Yet, no worries; there is at most one student who is unlucky. Could you help us find out who the unfortunate dude (if any) is? So that we can comfort him by treating him to a big meal!

Input

There are multiple test cases. For each test case, the first line contains a number N (0 < N <= 20000) indicating the number of societies. Then for each of the following N lines, there are three non-negative integers A_i, B_i, C_i (smaller than 2^31, A_i <= B_i) as stated above. Your program should proceed to the end of the file.

Output

For each test case, if there is no unlucky student, print "DC Qiang is unhappy." (excluding the quotation mark), in a single line. Otherwise print two integers, i.e., the label of the unlucky student and the number of flyers he/she gets, in a single line.

Sample Input

```
2
1 10 1
2 10 1
4
5 20 7
6 14 3
5 9 1
7 21 12
```

Sample Output

```
1 1
8 1
```

K. Sword

Description

Satoshi, who has strong obsessions with Chinese domestic RPG, will never miss the marvelous game, Gu Jian Qi Tan II, a.k.a., the Legend of Ancient Sword II, because of its gorgeous graphics, picturesque characters and lively design.

Recently, Satoshi has some troubles in the new battle mode of the game, and seeks you for help to solve the problems.

Specifically, in the new instant-action mode, Satoshi can control N roles, and his goal is to kill all M tiny monsters to declare victory. The tiny monsters are too fragile to survive from any skills of any roles. In every second, Satoshi can choose to operate one single role once, or do nothing. However, to make his series of operations magnificent, he would not use same role in any two consecutive seconds, i.e., the role used in i-th second cannot be used in the (i+1)-th second anymore.

For each role, we know the number of skills he/she owns, as well as the LPT (loop time) of each skill, which indicates that the skill could be performed in every LPT seconds starting from the beginning (e.g., if the LPT of a skill is 3, Satoshi could cast the skill at 3s, 6s, 9s, ..., so on and so forth). Moreover, the information of whether a skill can reach (touch) a certain monster is also provided (i.e., whether a monster is within the attacking range of a specific skill). Your task is to help Satoshi use the minimum amount time to win the game; ties are broken by preferring fewer number of operations, which is counted by the number of seconds in which Satoshi chooses to operate a role instead of doing nothing.

Note that Satoshi could choose a certain role, and of course no roles, in a specific second. And when he operates a role in a specific second, he can cast all the available skills (subject to the the LPT constraints) if he wants. Time begins with the 1st second.

Input

The integer R in the first line of input indicates the total number of test cases. For each test case, there are two integers N (1 <= N <= 10) and M (1 <= M <= 200) as stated above, in the first line. Then N blocks of input follow. For the i-th block, the first line contains an integer S_i (1 <= S_i <= 300) denoting the number of skills that the i-th role owns. Each of the the next S_i lines in the i-th block has M+1 integers, i.e., the LPT (1 <= LPT <= 6) of the i-th role, and M Boolean values (0/1) where the j-th Boolean value indicates if the j-th monster is within the attacking range of the i-th role.

Output

For each test case, print the minimum number of seconds T by which Satoshi could win the game, as well as the corresponding minimum number of operations that Satoshi has to perform (in T seconds), in a single line; if it is impossible for Satoshi to win, output nothing but a zero in a single line.

Sample Input

```
4

1 1
1
1 1

1 1
1
3 1

2 2
2
1 1 0
2 1 0
1
1 1 0

1 2
3
2 1 0
5 0 1
6 1 1
```

Sample Output

```
1 1
3 1
0
5 2
```