Problem 1001. Link with Bracket Sequence II

Link has a bracket sequence of length n. Today, Link finds out that some brackets of the sequence were lost.

Of course, he wants you to calculate how many ways are there to fill the sequence so that it will be a valid bracket sequence.

Note that there are m types of brackets in Link's world.

Here's the definition of a valid bracket sequence:

· A sequence of length 0 is a valid bracket sequence.

· If A is a valid bracket sequence, x is some type of left bracket, y is the same type of right bracket, xAy is a valid bracket sequence.

· If A, B are both valid bracket sequences, then AB is also a valid bracket sequence.

Input

Each test contains multiple test cases. The first line contains the number of test cases $T(1 \le T \le 20)$. Description of the test cases follows.

The first line contains two integers $n(1 \le n \le 500), m(1 \le m < 10^9 + 7)$, which is the length of Link's bracket sequence and the types of brackets.

The second line contains n integers $a_1, a_2, \ldots, a_n(|a_i| \le m)$. The i-th integer a_i describes the i-th character in the sequence:

· If $a_i = 0$, it means the bracket in this position is lost.

· $a_i > 0$, it means the i-th character in the sequence is the a_i-th type of left bracket.

· $a_i < 0$, it means the i-th character in the sequence is the $-a_i$-th type of right bracket.

It is guaranteed that there are at most 15 test cases with $n > 100$.

Output

For each test case, output one integer in a line, which is the number of ways to fill the bracket sequence so that it is a valid bracket sequence. Since the answer can be huge, just print it modulo $10^9 + 7$.

For some reason, there could be no way to make it a valid bracket sequence.

Example Input

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

Example Output

```
3
4
135
```

Problem 1002. Link with Running

Link hates running.

Today, Link is asked to run. Roads in BIT can be described as n nodes and m **directed** edges. Link has to run from node 1 to node n. When Link is at node u_i, he can run through the i-th edge to get to node v_i. Every time he runs through the i-th edge, he spends e_i points of energy and gains p_i points of physical fitness.

As a lazy boy, Link wants to spend as little energy as possible. He is also greedy and wants to gain maximum physical fitness when spending minimum energy.

Tell Link the minimum energy min_e he needs to spend and the maximum physical fitness max_p he can gain when spending the minimum energy.

Input

Each test contains multiple test cases. The first line contains the number of test cases $T(1 \le T \le 12)$. Description of the test cases follows.

The first line contains two integers $n, m(2 \le n \le 10^5, 1 \le m \le 3 \times 10^5)$, which are the number of nodes and the number of edges.

Each of the next m lines contains four integers $u_i, v_i, e_i, p_i(1 \le u_i, v_i \le n, 0 \le e_i, p_i \le 10^9)$, describing an edge.

Output

For each test case, output min_e and max_p in a single line, separated by one space.

IT IS GUARANTEED THAT THE ANSWER EXISTS!!!

Example Input

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

Example Output

```
2 2
1 0
```

Problem 1003. Magic

A meteor shower passed through the kingdom of Woll, which led the people to speculate that it was a punishment from the gods. Many gathered at the door of the mage Hongly and prayed to him to protect them with his spells. The enthusiastic Hongly quickly granted them their blessing.

Hongly's spell requires the use of n magic towers **lined up in a straight line**, numbered as $1, 2, 3, ..., n$. Each tower requires some magic value to cast this spell. And each tower has an initial magic value of 0 . Hongly needs to add some magic ingredients to the magic tower in order to increase its magic values. When 1 unit of magic ingredients is added to a tower, it will increase the magic value of all nearby magic towers by 1 within a radius of k, in which k is called *effective radius*. For example, assuming the i-th magic tower is added with 1 unit of magic ingredients, when $k = 1$, only magic tower i will add its magic value by 1, when $k = 2$, magic tower $i - 1,\ i,\ i + 1$ will increase their magic values by 1, and so on. The effective radius for all towers is **the same**. For this spell to protect the people, the i-th magic tower needs **at least** p_i points of magic value.

However, magic is not free to be used at will, and when there are too many magic ingredients in a range, it can lead to a big explosion. So there are q restrictions among Hongly's magic towers. The i-th restriction contains three integers L_i, R_i, B_i, which means that the sum of the magic ingredients in the magic towers $[L_i, R_i]$ **cannot be greater than** B_i.

Hongly is glad to help the villagers but is worried about the explosion. Now he wants to know if there is a way to place the ingredients that will meet the magic value requirements of **all magic towers** for using the spell while avoiding an explosion. If so, he would also like to know the minimum value of magic ingredients that need to be used.

Input

Each test contains multiple test cases. The first line contains the number of test cases $T(1 \le T \le 15)$. Description of the test cases follows.

The input data of each test case has $q + 3$ lines.

The first line contains two integers: $n(1 \le n \le 10000),\ k(1 \le k \le \frac{n-1}{2})$, representing the number of magic towers and their effective radius.

The second line contains n integers: $p_1,\ p_2,\ p_3,\ ...\ p_n(1 \le p_i \le 1000)$, where p_i represents the magic value required for the i-th tower.

The third line contains an integer $q(0 \le q \le 100)$, representing the number of restrictions.

Next q lines, each line contains 3 integers:$L_i, R_i, B_i(1 \le L_i \le R_i \le n,\ 0 \le B_i \le 10000)$, have the same meaning as described above.

Output

For each test data, output one line contains an integer, representing the minimum amount of magic ingredients required. If the condition cannot be reached, output "-1" (without quotes).

Example Input

```
3
5 2
2 2 0 10 3
1
1 5 11
5 2
2 2 0 10 3
1
2 3 0
3 2
3 0 6
2
1 1 0
3 3 0
```

Example Output

```
-1
12
6
```

Problem 1004. Link with Equilateral Triangle

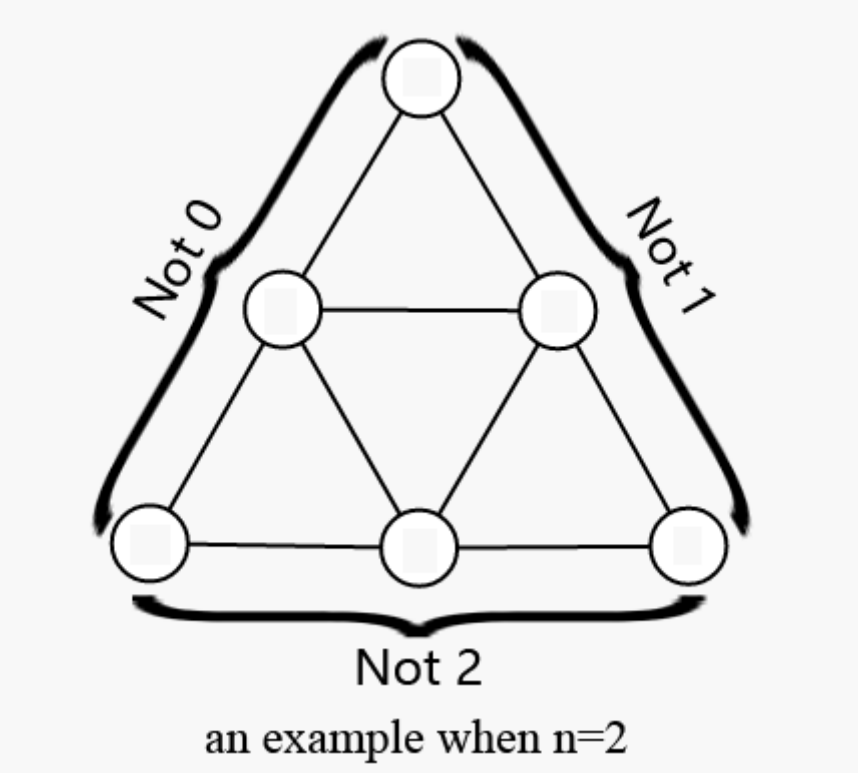
Link has a big equilateral triangle with side length n. The big triangle consists of n^2 small equilateral triangles with side length 1.

Link is going to fill numbers into each vertex of the small triangle with the following limits:

· The number filled in should be 0, 1, or 2.

· The left side of the big triangle should not be filled with 0. The right side of the big triangle should not be filled with 1. The bottom side of the big triangle should not be filled with 2.

· For each small triangle with side length 1, the sum of three vertices should not be a multiple of 3.



Link went crazy when he tried to do so because he couldn't find any triangle satisfying all conditions above. Now, he turns to you for help.

Please tell Link: Is it possible to fill the triangle so that it satisfies all conditions above?

Input

Each test contains multiple test cases. The first line contains the number of test cases T ($1 \le T \le 1000$). Description of the test cases follows.

For each test case, there is only one line containing a single integer n ($1 \le n \le 10^3$).

Output

For each test case, output Yes if it is possible to do so. Output No if it is impossible to do so.

Example Input

```
2
1
2
```

Example Output

```
No
No
```

Problem 1005. Link with Level Editor II

Link is playing a game, called **Advertising Space for Rent**.

In this game, a level consists of several worlds. Each world consists of m nodes and some directed roads. The player starts on node 1 of the first world. In each world, the player can either stay at the current node or go through exactly one road that exists in that world. After that, the player will be teleported to the next world without changing the ID of the node where he stays. If there is no next world, the game ends. The player wins if he ends on node m.

Link is editing a new level, he has already made n worlds (numbered from 1 to n) and wants to choose a **continuous subsegment** of them to form a new level. **The only limit is there shouldn't be more than k ways to win.** (Two ways are considered different if and only if the operation in some world differs.)

Link doesn't want to **discard** too many worlds. What is the **maximum** number of worlds Link can use in the new level?

Input

Each test contains multiple test cases. The first line contains the number of test cases $T(1 \le T \le 30)$. Description of the test cases follows.

The first line contains three integers n, m, k $(1 \le n \le 5 \times 10^3, 2 \le m \le 20, 1 \le k \le 10^9)$.

The following input describes the worlds from 1 to n. For each world:

The first line contains an integer l $(0 \le l \le m \times (m-1))$, which is the number of roads in this world.

The next l lines, each line contains two integers u, v $(1 \le u, v \le m, u \ne v)$, which means there is a road from node u to node v in this world.

For each world, it is guaranteed that there is no duplicate edge.

For each test case, it is guaranteed that the sum of l does not exceed 10^6.

It is guaranteed that the sum of n over all test cases does not exceed 10^5, and the sum of l over all test cases does not exceed 3×10^6.

Output

For each test case, output a single integer, which is the maximum number of worlds Link can use in the new level.

Example Input

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

Example Output

```
3
2
```

Problem 1006. BIT Subway

BIT(Beijing International Transport) subway, which can take people anywhere in a short time, is the most popular travel mode in 2050. One day, BIT subway launches a promotion as follows:

- If the total ticket price x you have spent this month is greater than or equal to 100 and you buy another ticket with $y$yuan, then you only need to pay $0.8y$yuan.

- If the total ticket price x you have spent this month is greater than or equal to 200 and you buy another ticket with $y$yuan, then you only need to pay $0.5y$yuan.

DLee is so happy that he can save more money to buy a house. However, a long time later, he notices that the real billing method is a bit different from what he thought. For example, DLee has spent 199yuan on tickets this month, he now buys a 10yuan ticket, then buys an 8yuan ticket:

- DLee thinks that he can buy only a part of the ticket instead of the whole ticket at a time. That is, for the 10yuan ticket, DLee thinks he can buy the 1.25yuan part of the ticket first and buy the 8.75yuan part of the ticket then. Under his misunderstanding, he needs to spend $199 + 1.25 * 0.8 + 8.75 * 0.5 + 8 * 0.5 = 208.375$yuan. Note that in this example, DLee has to spend 1.25yuan instead of only 1 yuan to make $x = 200$.

- The real billing method is that only if you have spent enough, you can get the discount, so it will be $199 + 10 * 0.8 + 8 * 0.5 = 211$yuan.

Now DLee wants to know in the previous months, how much difference did the billing method make.

Input

Each test contains multiple test cases. The first line contains one integer $T(1 \le T \le 10)$, which means the months DLee wants to check. Description of the months follows.

The first line contains a single integer $n(1 \le n \le 10^5)$, which means the number of tickets DLee bought in this month.

Then follows n integers $a_1, a_2, \ldots, a_n(1 \le a_i \le 200)$, a_i means the i-th ticket's price.

Output

For each month, output one line with two numbers divided by a single whitespace with three decimal places. The first number represents the cost in DLee's thought, and the second number represents the real cost.

Example Input

```
3
7
20 20 20 20 18 7 8
13
30 20 23 20 7 20 11 12 30 20 30 15 13
3
10 200 10
```

Example Output

```
110.400 111.400
213.000 216.900
196.000 215.000
```

hint

For the first case, DLee thinks the cost is: $20 + 20 + 20 + 20 + 18 + 2 + ((7 - 2) + 8) * 0.8 = 110.4$, the real cost is: $20 + 20 + 20 + 20 + 18 + 7 + 8 * 0.8 = 111.4$

Problem 1007. Climb Stairs

DLee came to a new level. What is waiting for him is a tall building with n floors, with a monster on each stair, the i-th of which has health point a_i.

DLee starts from the ground(which can be regarded as the 0-th floor), with a base attacking point a_0. He can choose to jump $1, 2, \ldots, k$ floors up or walk 1 floor down, but he cannot go to floors whose monster has a health point strictly greater than his attacking point, nor can he go to floors which had been visited. Once he comes and defeats a monster he can absorb his health point and add it to his attacking point.

Note that DLee should always be on floors $\{0, 1, 2, 3, \ldots, n\}$.

Now DLee asks you whether it is possible to defeat all the monsters and pass the level.

Input

There are T test cases.

In each test case, the first line contains three integers: $n, a_0, k (1 \leq n, a_0, k \leq 10^5)$, representing the number of floors, base attacking point, and the maximum number of floors that DLee can jump.

The second line contains n integers $a_1, \ldots, a_n (1 \leq a_i \leq 10^5)$, representing the health point of each monster.

The sum of n does not exceed 10^6.

Output

For each test case, output "YES" or "NO" to show whether it's possible to defeat all monsters.

Example Input

```
4
6 1 4
2 2 1 1 9 3
4 2 2
2 3 8 1
3 1 2
3 1 2
7 2 3
4 3 2 7 20 20 20
```

Example Output

```
YES
YES
NO
NO
```

Problem 1008. Fight and upgrade

You lead an army to fight monsters in a world where the blood of monsters is composed of physical blood hp_p and magical blood hp_m.

Your army has n soldiers, the i-th soldier has k_i attack methods, and the j-th attack method of the ith soldier causes p_{ij} points of physical damage and m_{ij} points of magic damage to the monster.

The total attack power of an army is the sum of the attack power of each soldier, and the attack power of each soldier is a weakly normalized linear combination of all attack methods of this soldier, i.e., the coefficients sum of the linear combination is less than or equal to 1. Formally, if the physical damage caused by a total attack of the army is $hurt_p$ and the magic damage is $hurt_m$, then

$$hurt_p = \sum_{i=1}^{n} \sum_{j=1}^{k_i} c_{ij} * p_{ij}$$

$$hurt_m = \sum_{i=1}^{n} \sum_{j=1}^{k_i} c_{ij} * m_{ij}$$

where c_{ij} is the weight assigned by the i-th soldier to the j-th attack, satisfying: $\forall i : \sum_{j=1}^{k_i} c_{ij} \leq 1$, $\forall c_{ij} \geq 0$.

Since you're competitive, you want to know if you can kill the monster in just 1 attack. After your army fights the r-th monster, whether or not it can be killed by your army in a single attack, it will drop an attack method that will be gained by the s_r-th soldier.

During the expedition, you will encounter q monsters in turn. For the r-th monster, it has five attributes $hp_{pr}, hp_{mr}, p_r, m_r, s_r$, which indicate the physical blood, magic blood, the physical damage and magic damage of the new attack method, the number of the soldier who gained the attack method after the battle.

Input

The input consists of multiple test cases.

The first line contains an integer T ($T = 11$) – the number of test cases.

For each test case:

The first line contains two positive integers $n, q (1 \leq n \leq 10^3, 1 \leq q \leq 10^5)$, which indicates the number of soldiers and the total number of monsters.

The next $3 * n$ rows describe the attacks of n soldiers. The $3 * i - 2$ to $3 * i$ rows describe the original attack of the i-th soldier: first an integer $k_i (1 \leq k_i \leq 10^5)$ indicating the number of attack methods of this soldier; the next k_i non-negative integer $p_{ij} (0 \leq p_{ij} \leq 10^6)$ indicating the physical damage of the j attack of the i soldier; the next k_i non-negative integer $m_{ij} (0 \leq m_{ij} \leq 10^6)$ indicating the magic damage of the j attack of the i soldier.

The next q rows, each containing five integers $hp_{pr}, hp_{mr}, p_r, m_r, s_r (0 \leq hp_p, hp_m \leq 10^9, 1 \leq s_r \leq n, 0 \leq p_r, m_r \leq 10^6,)$, indicate the *physical blood*, *magic blood* of the r-th monster, the *physical damage* of the new attack and the *magic damage* of the new attack, the *soldier number* that gained this attack method.

For a single test case, it is guaranteed that $\sum_{i=1}^{n} k_i \leq 10^5$.

Note : Not all 11 test cases are full, just make sure your time complexity can run a test case in 3s.

Output

For each test case, output q rows containing YES or NO, indicating whether the monster can be defeated in 1 attack.

Example Input

```
2
2 3
1
1
0
1
1
0
1 1 0 1 1
1 1 0 1 2
0 2 1 1 1
2 2
3
13 11 4
11 0 9
3
4 16 4
4 16 11
20 26 15 19 2
20 26 1 1 1
```

Example Output

```
NO
YES
YES
NO
YES
```

Problem 1009. Fall with Full Star

Fall loves playing games and collecting strange achievements in games. If he doesn't pass a level with full stars, he will feel uncomfortable.

Today, Fall is playing a new game. The game includes n levels without order, which means you can play the levels in any order. However, each level can only be challenged once. During the challenge, Fall may get hurt or pick up rare items. Assuming that before he challenges level i, he has power x, and after he challenges this level, his power will be increased by d_i. **After that**, if his power is higher than or equal to s_i, he will get a full star in this level.

Initially, Fall has power s_0 , he wants to find a challenge order that he can get full star levels as more as possible.

Input

The first line contains a single integer T ($1 \leq T \leq 20$), the number of test cases. For each test case:

The first line contains two integers $n, s_0 (1 \leq n \leq 1 \times 10^5, \sum n \leq 10^6, |s_0| \leq 10^{14})$.

The i-th line of the next n lines contains two integers $d_i, s_i (|d_i| \leq 10^9, |s_i| \leq 10^{14})$, representing the information about the i-th level.

Output

For each test case, output a single line containing one integer – the maximum number of full star levels that Fall can get in a single line.

Example Input

```
1
3 0
4 5
2 5
1 100
```

Example Output

```
2
```

Problem 1010. Fall with Intersection

Alice and Bob are playing a game, the rules are as follows.

Initially, there is an infinite plane with w straight lines. Alice and Bob now draw new lines in turn, m rounds in total (so there will be $w + 2m$ lines in the end). After that, these lines will produce some intersections, if the number of intersections is odd, then Bob wins, and Alice wins otherwise. If multiple lines intersect at one point, this intersection will only be calculated once.

All the straight lines should be in the form of $y = kx + b$, where k and b are integers, but there is no limitation of the range(except the first w lines).

The game has reached a white-hot stage, there have been $w + n$ lines on the plane. Now Alice and Bob are very focused and their brains are racing that all the decisions they make will be optimal.

Fall is interesting with the final result. Can you tell him who will be the winner in the end?

Input

The first line contains a single integer T ($1 \le T \le 100$), the number of test cases. For each test case:

The first line contains three integers, $w, m, n (0 \le w \le 5, 1 \le m \le 500, \sum m \le 1300, 0 \le n \le 2m)$, with the same meaning as described above.

The next $w + n$ lines, each line contains two integers $k_i, b_i (-10^6 \le k_i, b_i \le 10^6)$, representing the i-th line one the plane is $y = k_i x + b_i$.

Output

For each test case, output a single line containing one string ("Alice" or "Bob", without quotation marks), representing the winner.

Example Input

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

Example Output

```
Alice
```

Problem 1011. Link is as bear

Link, a famous bear magician in Bear Institute of Talented(BIT), has recently learned new magic.

That is, given a array a containing n elements $a_1, ..., a_n$, and Link can cast the following magic:

Link can choose two integers l, r such that $1 \leq l \leq r \leq n$, making all $a_i = xor(l, r)$ where $l \leq i \leq r$ and $xor(l, r)$ denotes the bitwise-xor($\oplus$) of all elements in $[l, r]$. More formally, $xor(l, r) = a_l \oplus a_{l+1} \oplus ... \oplus a_r$.

Link can cast this magic any time(possibly, zero) and can choose l, r arbitrarily. However, since Link has a sort of Obsessive-Compulsive Disorder(OCD), he wants all elements to become the same after his operation. Now, he wonders about the maximum of this same value.

What's more, Link finds that the given array has a weird property: there always exists at least one pair of $x, y(x \neq y)$ such that $a_x = a_y$.

Input

The first line contains an integer $T(1 \leq T \leq 3 * 10^4)$, the number of the test cases.

The first line of each test case is an interger $n(1 \leq n \leq 10^5)$, the length of the array a.

The second line of each test case containing n integers, while the i-th denoting $a_i(0 \leq a_i \leq 10^{15})$. It's guaranteed that there always exists at least one pair of $x, y(x \neq y)$ such that $a_x = a_y$.

It's also guaranteed that $\Sigma n \leq 10^6$.

Output

For each test case, output a single intergers indicating the maximum of the same value after Link's operations.

Example Input

```
2
5
10 10 10 10 10
4
1 1 2 1
```

Example Output

```
10
3
```