A. Case of Matryoshkas

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Andrewid the Android is a galaxy-famous detective. He is now investigating the case of vandalism at the exhibition of contemporary art.

The main exhibit is a construction of n matryoshka dolls that can be nested one into another. The matryoshka dolls are numbered from 1 to n. A matryoshka with a smaller number can be nested in a matryoshka with a higher number, two matryoshkas can not be directly nested in the same doll, but there may be chain nestings, for example, $1 \to 2 \to 4 \to 5$.

In one second, you can perform one of the two following operations:

- Having a matryoshka a that isn't nested in any other matryoshka and a matryoshka b, such that b doesn't contain any other matryoshka and is not nested in any other matryoshka, you may put a in b;
- Having a matryoshka a directly contained in matryoshka b, such that b is not nested in any other matryoshka, you may get a out of b.

According to the modern aesthetic norms the matryoshka dolls on display were assembled in a specific configuration, i.e. as several separate chains of nested matryoshkas, but the criminal, following the mysterious plan, took out all the dolls and assembled them into a single large chain ($1 \to 2 \to ... \to n$). In order to continue the investigation Andrewid needs to know in what minimum time it is possible to perform this action.

Input

The first line contains integers n ($1 \le n \le 10^5$) and k ($1 \le k \le 10^5$) — the number of matryoshkas and matryoshka chains in the initial configuration.

The next k lines contain the descriptions of the chains: the i-th line first contains number m_i ($1 \le m_i \le n$), and then m_i numbers $a_{i1}, a_{i2}, ..., a_{im_i}$ — the numbers of matryoshkas in the chain (matryoshka a_{i1} is nested into matryoshka a_{i2}, that is nested into matryoshka a_{i3}, and so on till the matryoshka a_{im_i} that isn't nested into any other matryoshka).

It is guaranteed that $m_1 + m_2 + ... + m_k = n$, the numbers of matryoshkas in all the chains are distinct, in each chain the numbers of matryoshkas follow in the ascending order.

Output

In the single line print the minimum number of seconds needed to assemble one large chain from the initial configuration.

Sample test(s)

input

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

output

```
1
```

input

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

output

```
10
```

Note

In the first sample test there are two chains: $1 \to 2$ and 3. In one second you can nest the first chain into the second one and get $1 \to 2 \to 3$.

In the second sample test you need to disassemble all the three chains into individual matryoshkas in 2 + 1 + 1 = 4 seconds and then assemble one big chain in 6 seconds.

B. Case of Fugitive

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Andrewid the Android is a galaxy-famous detective. He is now chasing a criminal hiding on the planet Oxa-5, the planet almost fully covered with water.

The only dry land there is an archipelago of n narrow islands located in a row. For more comfort let's represent them as non-intersecting segments on a straight line: island i has coordinates $[l_i, r_i]$, besides, $r_i < l_{i+1}$ for $1 \le i \le n - 1$.

To reach the goal, Andrewid needs to place a bridge between each pair of **adjacent** islands. A bridge of length a can be placed between the i-th and the $(i+1)$-th islads, if there are such coordinates of x and y, that $l_i \le x \le r_i$, $l_{i+1} \le y \le r_{i+1}$ and $y - x = a$.

The detective was supplied with m bridges, each bridge can be used at most once. Help him determine whether the bridges he got are enough to connect each pair of adjacent islands.

Input

The first line contains integers n $(2 \le n \le 2\cdot 10^5)$ and m $(1 \le m \le 2\cdot 10^5)$ — the number of islands and bridges.

Next n lines each contain two integers l_i and r_i $(1 \le l_i \le r_i \le 10^{18})$ — the coordinates of the island endpoints.

The last line contains m **integer** numbers $a_1, a_2, \dots, a_m$ $(1 \le a_i \le 10^{18})$ — the lengths of the bridges that Andrewid got.

Output

If it is impossible to place a bridge between each pair of adjacent islands in the required manner, print on a single line "`No`" (without the quotes), otherwise print in the first line "`Yes`" (without the quotes), and in the second line print $n - 1$ numbers $b_1, b_2, \dots, b_{n-1}$, which mean that between islands i and $i + 1$ there must be used a bridge number b_i.

If there are multiple correct answers, print any of them. Note that in this problem it is necessary to print "`Yes`" and "`No`" in correct case.

Sample test(s)

input

```
4 4
1 4
7 8
9 10
12 14
4 5 3 8
```

output

```
Yes
2 3 1
```

input

```
2 2
11 14
17 18
2 9
```

output

```
No
```

input

```
2 1
1 1
1000000000000000000 1000000000000000000
999999999999999999
```

output

```
Yes
1
```

Note

In the first sample test you can, for example, place the second bridge between points 3 and 8, place the third bridge between points 7 and 10 and place the first bridge between points 10 and 14.

In the second sample test the first bridge is too short and the second bridge is too long, so the solution doesn't exist.

C. Case of Chocolate

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Andrewid the Android is a galaxy-known detective. Now he does not investigate any case and is eating chocolate out of boredom.

A bar of chocolate can be presented as an $n \times n$ table, where each cell represents one piece of chocolate. The columns of the table are numbered from 1 to n from left to right and the rows are numbered from top to bottom. Let's call the anti-diagonal to be a diagonal that goes the lower left corner to the upper right corner of the table. First Andrewid eats all the pieces lying below the anti-diagonal. Then he performs the following q actions with the remaining triangular part: first, he chooses a piece on the anti-diagonal and either direction 'up' or 'left', and then he begins to eat all the pieces starting from the selected cell, moving in the selected direction until he reaches the already eaten piece or chocolate bar edge.

After each action, he wants to know how many pieces he ate as a result of this action.

Input

The first line contains integers n $(1 \le n \le 10^9)$ and q $(1 \le q \le 2 \cdot 10^5)$ — the size of the chocolate bar and the number of actions.

Next q lines contain the descriptions of the actions: the i-th of them contains numbers x_i and y_i $(1 \le x_i, y_i \le n,\ x_i + y_i = n + 1)$ — the numbers of the column and row of the chosen cell and the character that represents the direction (`L` — left, `U` — up).

Output

Print q lines, the i-th of them should contain the number of eaten pieces as a result of the i-th action.

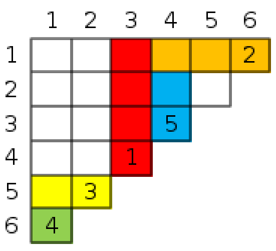
Sample test(s)

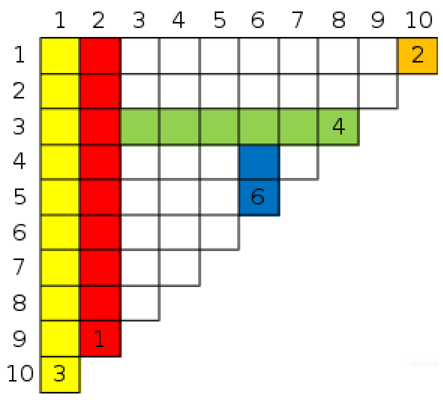
input
```
6 5
3 4 U
6 1 L
2 5 L
1 6 U
4 3 U
```
output
```
4
3
2
1
2
```

input
```
10 6
2 9 U
10 1 U
1 10 U
8 3 L
10 1 L
6 5 U
```
output
```
9
1
10
6
0
2
```

Note

Pictures to the sample tests:





The pieces that were eaten in the same action are painted the same color. The pieces lying on the anti-diagonal contain the numbers of the action as a result of which these pieces were eaten.

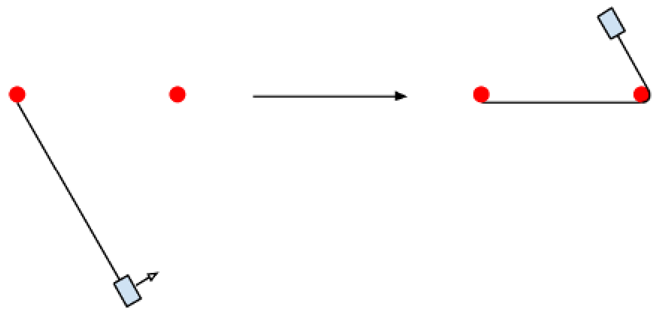
In the second sample test the Andrewid tries to start eating chocolate for the second time during his fifth action, starting from the cell at the intersection of the 10-th column and the 1-st row, but this cell is already empty, so he does not eat anything.

D. Case of a Top Secret

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Andrewid the Android is a galaxy-famous detective. Now he is busy with a top secret case, the details of which are not subject to disclosure.

However, he needs help conducting one of the investigative experiment. There are n pegs put on a plane, they are numbered from 1 to n, the coordinates of the i-th of them are $(x_i, 0)$. Then, we tie to the bottom of one of the pegs a weight on a tight rope of length l (thus, its coordinates will be equal to $(x_i, -l)$, where i is the number of the used peg). Then the weight is pushed to the right, so that it starts to rotate counterclockwise. At the same time, if the weight during rotation touches some of the other pegs, it then begins to rotate around that peg. Suppose that each peg itself is very thin and does not affect the rope length while weight is rotating around it.



More formally, if at some moment the segment of the rope contains one or more pegs in addition to the peg around which the weight is rotating, the weight will then rotate around the farthermost one of them on a shorter segment of a rope. In particular, if the segment of the rope touches some peg by its endpoint, it is considered that the weight starts to rotate around that peg on a segment of the rope of length 0.

At some moment the weight will begin to rotate around some peg, without affecting the rest of the pegs. Andrewid interested in determining the number of this peg.

Andrewid prepared m queries containing initial conditions for pushing the weight, help him to determine for each of them, around what peg the weight will eventually rotate.

Input

The first line contains integers n and m $(1 \le n, m \le 2 \cdot 10^5)$ — the number of pegs and queries.

The next line contains n integers $x_1, x_2, \ldots, x_n$ $(-10^9 \le x_i \le 10^9)$ — the coordinates of the pegs. It is guaranteed that the coordinates of all the pegs are distinct integers.

Next m lines contain the descriptions of the queries of pushing the weight, each consists of two integers a_i $(1 \le a_i \le n)$ and l_i $(1 \le l_i \le 10^9)$ — the number of the starting peg and the length of the rope.

Output

Print m lines, the i-th line should contain the number of the peg around which the weight will eventually rotate after the i-th push.

Sample test(s)

input

```
3 2
0 3 5
2 3
1 8
```

output

```
3
2
```

input

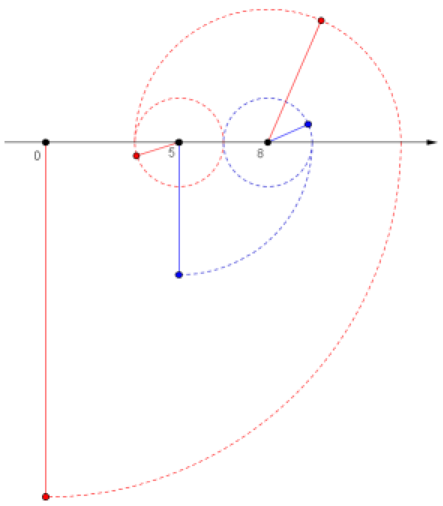
```
4 4
1 5 7 15
1 4
2 15
3 16
1 28
```

output

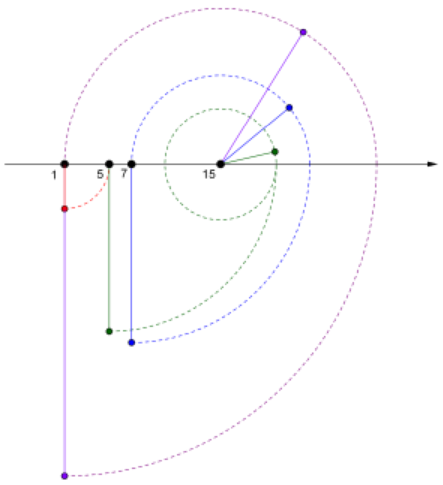
```
2
4
3
1
```

Note

Picture to the first sample test:



Picture to the second sample test:



Note that in the last query weight starts to rotate around the peg 1 attached to a rope segment of length 0.

E. Case of Computer Network

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Andrewid the Android is a galaxy-known detective. Now he is preparing a defense against a possible attack by hackers on a major computer network.

In this network are n vertices, some pairs of vertices are connected by m undirected channels. It is planned to transfer q important messages via this network, the i-th of which must be sent from vertex s_i to vertex d_i via one or more channels, perhaps through some intermediate vertices.

To protect against attacks a special algorithm was developed. Unfortunately it can be applied only to the network containing directed channels. Therefore, as new channels can't be created, it was decided for each of the existing undirected channels to enable them to transmit data only in one of the two directions.

Your task is to determine whether it is possible so to choose the direction for each channel so that each of the q messages could be successfully transmitted.

Input

The first line contains three integers n, m and q ($1 \le n, m, q \le 2 \cdot 10^5$) — the number of nodes, channels and important messages.

Next m lines contain two integers each, v_i and u_i ($1 \le v_i, u_i \le n$, $v_i \ne u_i$), that means that between nodes v_i and u_i is a channel. Between a pair of nodes can exist more than one channel.

Next q lines contain two integers s_i and d_i ($1 \le s_i, d_i \le n$, $s_i \ne d_i$) — the numbers of the nodes of the source and destination of the corresponding message.

It is not guaranteed that in it initially possible to transmit all the messages.

Output

If a solution exists, print on a single line "`Yes`" (without the quotes). Otherwise, print "`No`" (without the quotes).

Sample test(s)

input

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

output

```
Yes
```

input

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

output

```
No
```

input

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

output

```
Yes
```

Note

In the first sample test you can assign directions, for example, as follows: $1 \to 2$, $1 \to 3$, $3 \to 2$, $4 \to 3$. Then the path for for the first message will be $1 \to 3$, and for the second one — $4 \to 3 \to 2$.

In the third sample test you can assign directions, for example, as follows: $1 \to 2$, $2 \to 1$, $2 \to 3$. Then the path for the first message will be $1 \to 2 \to 3$, and for the second one — $2 \to 1$.