Codeforces Round #359 (Div. 1)

A. Robbers' watch

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

Robbers, who attacked the Gerda's cab, are very successful in covering from the kingdom police. To make the goal of catching them even harder, they use their own watches.

First, as they know that kingdom police is bad at math, robbers use the positional numeral system **with base** 7. Second, they divide one day in n hours, and each hour in m minutes. Personal watches of each robber are divided in two parts: first of them has the smallest possible number of places that is necessary to display any integer from 0 to $n - 1$, while the second has the smallest possible number of places that is necessary to display any integer from 0 to $m - 1$. Finally, if some value of hours or minutes can be displayed using less number of places in base 7 than this watches have, the required number of zeroes is added at the beginning of notation.

Note that to display number 0 section of the watches is required to have at least one place.

Little robber wants to know the number of moments of time (particular values of hours and minutes), such that all digits displayed on the watches are **distinct**. Help her calculate this number.

Input

The first line of the input contains two integers, given in the decimal notation, n and m ($1 \leq n, m \leq 10^9$) — the number of hours in one day and the number of minutes in one hour, respectively.

Output

Print one integer in decimal notation — the number of different pairs of hour and minute, such that all digits displayed on the watches are distinct.

Examples

input
```
2 3
```
output
```
4
```

input
```
8 2
```
output
```
5
```

Note

In the first sample, possible pairs are: $(0: 1)$, $(0: 2)$, $(1: 0)$, $(1: 2)$.

In the second sample, possible pairs are: $(02: 1)$, $(03: 1)$, $(04: 1)$, $(05: 1)$, $(06: 1)$.

B. Kay and Snowflake

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

After the piece of a devilish mirror hit the Kay's eye, he is no longer interested in the beauty of the roses. Now he likes to watch snowflakes.

Once upon a time, he found a huge snowflake that has a form of the tree (connected acyclic graph) consisting of n nodes. The root of tree has index 1. Kay is very interested in the structure of this tree.

After doing some research he formed q queries he is interested in. The i-th query asks to find a centroid of the subtree of the node v_i. Your goal is to answer all queries.

Subtree of a node is a part of tree consisting of this node and all it's descendants (direct or not). In other words, subtree of node v is formed by nodes u, such that node v is present on the path from u to root.

Centroid of a tree (or a subtree) is a node, such that if we erase it from the tree, the maximum size of the connected component will be at least two times smaller than the size of the initial tree (or a subtree).

Input

The first line of the input contains two integers n and q ($2 \leq n \leq 300\,000$, $1 \leq q \leq 300\,000$) — the size of the initial tree and the number of queries respectively.

The second line contains n - 1 integer $p_2, p_3, ..., p_n$ ($1 \leq p_i \leq n$) — the indices of the parents of the nodes from 2 to n. Node 1 is a root of the tree. It's guaranteed that p_i define a correct tree.

Each of the following q lines contain a single integer v_i ($1 \leq v_i \leq n$) — the index of the node, that define the subtree, for which we want to find a centroid.

Output

For each query print the index of a centroid of the corresponding subtree. If there are many suitable nodes, print any of them. It's guaranteed, that each subtree has at least one centroid.

Example

input

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

output

```
3
2
3
6
```

Note

The first query asks for a centroid of the whole tree — this is node 3. If we delete node 3 the tree will split in four components, two of size 1 and two of size 2.

The subtree of the second node consists of this node only, so the answer is 2.

Node 3 is centroid of its own subtree.

The centroids of the subtree of the node 5 are nodes 5 and 6 — both answers are considered correct.

C. Optimal Point

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

When the river brought Gerda to the house of the Old Lady who Knew Magic, this lady decided to make Gerda her daughter. She wants Gerda to forget about Kay, so she puts all the roses from the garden underground.

Mole, who lives in this garden, now can watch the roses without going up to the surface. Typical mole is blind, but this mole was granted as special vision by the Old Lady. He can watch any underground objects on any distance, even through the obstacles and other objects. However, the quality of the picture depends on the Manhattan distance to object being observed.

Mole wants to find an *optimal* point to watch roses, that is such point with **integer coordinates** that the maximum Manhattan distance to the rose is minimum possible.

As usual, he asks you to help.

Manhattan distance between points $(x_1,\ y_1,\ z_1)$ and $(x_2,\ y_2,\ z_2)$ is defined as $|x_1 - x_2| + |y_1 - y_2| + |z_1 - z_2|$.

Input

The first line of the input contains an integer t t $(1 \le t \le 100\,000)$ — the number of test cases. Then follow exactly t blocks, each containing the description of exactly one test.

The first line of each block contains an integer n_i $(1 \le n_i \le 100\,000)$ — the number of roses in the test. Then follow n_i lines, containing three integers each — the coordinates of the corresponding rose. Note that two or more roses may share the same position.

It's guaranteed that the sum of all n_i doesn't exceed $100\,000$ and all coordinates are not greater than 10^{18} by their absolute value.

Output

For each of t test cases print three integers — the coordinates of the optimal point to watch roses. If there are many optimal answers, print any of them.

The coordinates of the optimal point may coincide with the coordinates of any rose.

Examples

input

```
1
5
0 0 4
0 0 -4
0 4 0
4 0 0
1 1 1
```

output

```
0 0 0
```

input

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

output

```
3 5 9
3 5 9
```

Note

In the first sample, the maximum Manhattan distance from the point to the rose is equal to 4.

In the second sample, the maximum possible distance is 0. Note that the positions of the roses may coincide with each other and with the position of the optimal point.

D. Kay and Eternity

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

Snow Queen told Kay to form a word "eternity" using pieces of ice. Kay is eager to deal with the task, because he will then become free, and Snow Queen will give him all the world and a pair of skates.

Behind the palace of the Snow Queen there is an infinite field consisting of cells. There are n pieces of ice spread over the field, each piece occupying exactly one cell and no two pieces occupying the same cell. To estimate the difficulty of the task Kay looks at some squares of size $k \times k$ cells, with corners located at the corners of the cells and sides parallel to coordinate axis and counts the number of pieces of the ice inside them.

This method gives an estimation of the difficulty of some part of the field. However, Kay also wants to estimate the total difficulty, so he came up with the following criteria: for each x ($1 \leq x \leq n$) he wants to count the number of squares of size $k \times k$, such that there are exactly x pieces of the ice inside.

Please, help Kay estimate the difficulty of the task given by the Snow Queen.

Input

The first line of the input contains two integers n and k ($1 \leq n \leq 100\,000$, $1 \leq k \leq 300$) — the number of pieces of the ice and the value k, respectively. Each of the next n lines contains two integers x_i and y_i ($-10^9 \leq x_i, y_i \leq 10^9$) — coordinates of the cell containing i-th piece of the ice. It's guaranteed, that no two pieces of the ice occupy the same cell.

Output

Print n integers: the number of squares of size $k \times k$ containing exactly $1, 2, ..., n$ pieces of the ice.

Example

input

```
5 3
4 5
4 6
5 5
5 6
7 7
```

output

```
10 8 1 4 0
```

E. Travelling Through the Snow Queen's Kingdom

time limit per test: 1.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Gerda is travelling to the palace of the Snow Queen.

The road network consists of n intersections and m bidirectional roads. Roads are numbered from 1 to m. Snow Queen put a powerful spell on the roads to change the weather conditions there. Now, if Gerda steps on the road i at the moment of time less or equal to i, she will leave the road exactly at the moment i. In case she steps on the road i at the moment of time greater than i, she stays there forever.

Gerda starts at the moment of time l at the intersection number s and goes to the palace of the Snow Queen, located at the intersection number t. Moreover, she has to be there at the moment r (or earlier), before the arrival of the Queen.

Given the description of the road network, determine for q queries l_i, r_i, s_i and t_i if it's possible for Gerda to get to the palace on time.

Input

The first line of the input contains integers n, m and q ($2 \le n \le 1000$, $1 \le m, q \le 200\,000$) — the number of intersections in the road network of Snow Queen's Kingdom, the number of roads and the number of queries you have to answer.

The i-th of the following m lines contains the description of the road number i. The description consists of two integers v_i and u_i ($1 \le v_i, u_i \le n$, $v_i \ne u_i$) — the indices of the intersections connected by the i-th road. It's possible to get both from v_i to u_i and from u_i to v_i using only this road. Each pair of intersection may appear several times, meaning there are several roads connecting this pair.

Last q lines contain the queries descriptions. Each of them consists of four integers l_i, r_i, s_i and t_i ($1 \le l_i \le r_i \le m$, $1 \le s_i, t_i \le n$, $s_i \ne t_i$) — the moment of time Gerda starts her journey, the last moment of time she is allowed to arrive to the palace, the index of the starting intersection and the index of the intersection where palace is located.

Output

For each query print "`Yes`" (without quotes) if Gerda can be at the Snow Queen palace on time (not later than r_i) or "`No`" (without quotes) otherwise.

Example

input

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

output

```
Yes
Yes
Yes
No
No
Yes
```