2021 Canadian Computing Olympiad
Day 1, Problem 1
Swap Swap Sort

Time Limit: 3 seconds

Problem Description

There is an array of N integers, each with a value between 1 and K. Your friend has a an algorithm that can sort this array according to any ordering of the numbers from 1 to K. The algorithm performs a sequence of **swap** operations, that exchange two **adjacent** elements of the array. The algorithm performs exactly the minimum number of such swaps to sort the array.

The desired ordering of the numbers from 1 to K is given by a **target permutation**. A target permutation is a sequence of each number from 1 to K appearing exactly once, in the same order that is desired by the corresponding ordering.

For example, the array `[1 4 2 1 2]` sorted by the target permutation $4, 1, 2, 3$ results in the array `[4 1 1 2 2]`.

You are interested in the number of swaps performed by your friend's algorithm for different target permutations. To explore this, you start with a target permutation of $1, 2, \ldots, K$ and perform Q operations on it. Each operation swaps two **adjacent** elements of the target permutation. After performing each operation, find the number of swaps your friend's algorithm would make if it was run with the current target permutation. The Q operations cumulatively change the target permutation, but do not affect the array.

Input Specification

The first line contains the three integers N, K, and Q ($1 \leq K \leq N \leq 100\,000$, $1 \leq Q \leq 1\,000\,000$).

The next line contains N integers $a_1, a_2, \ldots, a_N$ ($1 \leq a_i \leq K$) specifying the array .

The next Q lines each contains a single integer j ($1 \leq j \leq K-1$), representing the operation of swapping the elements of the target permutation at indices j and $j+1$.

For 3 of the 25 available marks, $N, Q \leq 5\,000$.

For an additional 3 of the 25 available marks, $Q \leq 100$.

For an additional 5 of the 25 available marks, $K \leq 500$.

Output Specification

For each of the Q operations, output a line containing a single integer; the answer for the current target permutation.

Sample Input

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

Output for Sample Input

```
4
2
2
```

Explanation of Output for Sample Input

The three target permutations are $1, 2, 4, 3$, then $1, 4, 2, 3$, then $4, 1, 2, 3$. For the final target permutation, your friend's algorithm uses two swaps to sort the array `[1 4 2 1 2]` to `[4 1 1 2 2]`.

2021 Canadian Computing Olympiad
Day 1, Problem 2
Weird Numeral System

Time Limit: 1.5 seconds

Problem Description

Alice enjoys thinking about base-K numeral systems (don't we all?). As you might know, in the standard base-K numeral system, an integer n can be represented as $d_{m-1}\ d_{m-2}\ \ldots\ d_1\ d_0$ where:

- Each digit d_i is in the set $\{0, 1, \ldots, K-1\}$, and
- $d_{m-1}K^{m-1} + d_{m-2}K^{m-2} + \cdots + d_1K^1 + d_0K^0 = n$.

For example, in standard base-3, you would write 15 as `1 2 0`, since $(1)\cdot 3^2 + (2)\cdot 3^1 + (0)\cdot 3^0 = 15$.

But standard base-K systems are too easy for Alice. Instead, she's thinking about **weird-base-K** systems.

A weird-base-K system is just like the standard base-K system, except that instead of using the digits $\{0, \ldots, K-1\}$, you use $\{a_1, a_2, \ldots, a_D\}$ for some value D. For example, in a weird-base-3 system with $a = \{-1, 0, 1\}$, you could write 15 as `1 -1 -1 0`, since $(1)\cdot 3^3 + (-1)\cdot 3^2 + (-1)\cdot 3^1 + (0)\cdot 3^0 = 15$.

Alice is wondering how to write Q integers, n_1 through n_Q, in a weird-base-K system that uses the digits a_1 through a_D. Please help her out!

Input Specification

The first line contains four space-separated integers, K, Q, D, and M ($2 \le K \le 1\,000\,000$, $1 \le Q \le 5$, $1 \le D \le 5001$, $1 \le M \le 2500$).

The second line contains D distinct integers, a_1 through a_D ($-M \le a_i \le M$).

Finally, the i-th of the next Q lines contains n_i ($-10^{18} \le n_i \le 10^{18}$).

For 8 of the 25 available marks, $M = K - 1 \le 400$, $K = D \le 801$.

Output Specification

Output Q lines, the i-th of which is a weird-base-K representation of n_i. If multiple representations are possible, any will be accepted. The digits of the representation should be separated by spaces. Note that 0 must be represented by a non-empty set of digits.

If there is no possible representation, output `IMPOSSIBLE`.

Sample Input 1

```
3 3 3 1
-1 0 1
15
8
-5
```

Output for Sample Input 1

```
1 -1 -1 0
1 0 -1
-1 1 1
```

Explanation of Output for Sample Input 1

We have:
$(1) \cdot 3^3 + (-1) \cdot 3^2 + (-1) \cdot 3^1 + (0) \cdot 3^0 = 15$,
$(1) \cdot 3^2 + (0) \cdot 3^1 + (-1) \cdot 3^0 = 8$, and
$(-1) \cdot 3^2 + (1) \cdot 3^1 + (1) \cdot 3^0 = -5$.

Sample Input 2

```
10 1 3 2
0 2 -2
17
```

Output for Sample Input 2

```
IMPOSSIBLE
```

2021 Canadian Computing Olympiad
Day 1, Problem 3
Through Another Maze Darkly

Time Limit: 8 seconds

Problem Description
There is a maze that is formed by connecting N rooms via $N-1$ corridors. The rooms are numbered 1 to N and each room has the shape of a circle. The corridors have the following constraints:

- Each corridor forms a connection between two distinct rooms.
- Every pair of rooms is connected by exactly one path of connected corridors.

One difficulty in navigating through this maze is that the lights are all out, so you cannot see where you are. To help with navigation, each room contains a laser pointer that initially points to a corridor. Consider the following strategy:

- Rotate the room's laser pointer clockwise until it points to another corridor.
- Follow the room's laser pointer and traverse the corridor.
- Repeat the previous two steps indefinitely.

You created Q queries to investigate this strategy. For each query, you are given an integer K and you start at room 1. Determine the final room after traversing exactly K corridors. All laser pointers will reset to their original orientation after each query.

Input Specification
The first line contains the integers N and Q ($2 \leq N \leq 800\,000$, $1 \leq Q \leq 800\,000$).

The next N lines describe the layout of the maze, with the i^{th} of these lines describing room i. Specifically, it contains an integer k, the number of corridors connecting to room i, followed by k integers, $c_1\ c_2 \ldots c_k$, indicating the rooms that these k corridors lead to, in clockwise order. Lastly, room i's laser pointer initially points to the corridor leading to room c_1.

The final Q lines describe a query, with each line containing an integer K ($1 \leq K \leq 10^{15}$).

For 4 of the 25 available marks, the i^{th} corridor forms a connection between room i and room $i+1$.

For an additional 4 of the 25 available marks, $N \leq 2000$ and $Q \leq 2000$.

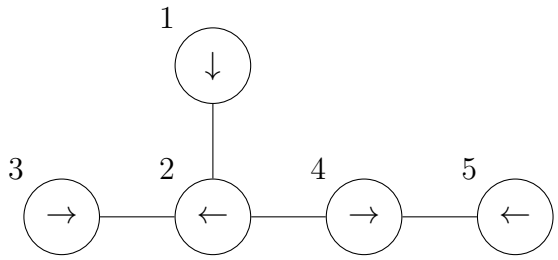
For an additional 12 of the 25 available marks, $Q = 1$.

Output Specification
Output Q lines answering the queries in order.

Sample Input
```
5 6
1 2
3 3 1 4
1 2
2 5 2
1 4
1
2
3
4
5
6
```

Output for Sample Input
```
2
1
2
4
2
3
```

Explanation of Output for Sample Input
This is the initial state of the maze.



The strategy will visit these rooms in order:

$$1, 2, 1, 2, 4, 2, 3, \ldots$$

2021 Canadian Computing Olympiad
Day 2, Problem 1
Travelling Merchant

Time Limit: 1 second

Problem Description

A merchant would like to make a business of travelling between cities, moving goods from one city to another in exchange for a profit. There are N cities and M trading routes between them.

The i-th trading route lets the merchant travel from city a_i to city b_i (in only that direction). In order to take this route, the merchant must already have at least r_i units of money. After taking this route, the total amount of money the merchant has will increase by p_i units, with a guarantee that $p_i \geq 0$.

For each of the N cities, we would like to know the minimum amount of money required for a merchant to start in that city and be able to keep travelling forever.

Input Specification

The first line contains the two integers N and M ($2 \leq N, M \leq 200\,000$).

The i-th of the next M lines contains the four integers a_i, b_i, r_i, and p_i ($1 \leq a_i, b_i \leq N$, $a_i \neq b_i$, $0 \leq r_i, p_i \leq 10^9$). Note that there may be any number of routes between a pair of cities.

For 4 of the 25 available marks, $N, M \leq 2\,000$.

For an additional 5 of the 25 available marks, $p_i = 0$ for all i.

Output Specification

On a single line, output N space-separated integers, where the i-th is the answer if the merchant were to start at city i. This answer is either the minimum amount of money, or -1 if no amount of money can be sufficient.

Sample Input

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

Output for Sample Input

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

Explanation of Output for Sample Input

Starting from city 2 with 3 units of money, the merchant can cycle between cities 2, 1, and 3.

2021 Canadian Computing Olympiad
Day 2, Problem 2

Bread First Search

Time Limit: 1 second

Problem Description
There are N towns in a network of M undirected roads. Each road connects one pair of towns. The government has decided to conduct a breadth first search, which means finding an ordering of the N towns such that if the ordering begins with r:

- Each town except for r is adjacent to another town given earlier in the order.
- The towns are given in non-decreasing order of distance to r. Here, distance means the minimum number of roads that need to be traversed to reach a town.

However, someone mistakenly did a *bread* first search. They found the ordering $1, 2, \ldots, N$ (this was obtained by sorting the towns in increasing order of bread production).

To cover up this embarrassment, the government would like to build new roads such that $1, 2, \ldots, N$ is also a possible breadth first search ordering. The new roads can be built between any two towns. What is the minimum possible number of roads that need to be built?

Input Specification
The first line contains the two integers N and M ($1 \leq N \leq 200\,000$, $0 \leq M \leq \min(200\,000, \frac{N(N-1)}{2})$).

The i-th of the next M lines contains the two integers a_i and b_i ($1 \leq a_i, b_i \leq N$), representing the two endpoints of the i-th road. It is guaranteed that $a_i \neq b_i$ and there is at most one road between any two towns.

For 5 of the 25 available marks, $N \leq 200$.

For an additional 6 of the 25 marks available, $N \leq 5\,000$.

Output Specification
On a single line, output the minimum number of roads that must be constructed.

Sample Input 1
```
2 0
```

Output for Sample Input 1
```
1
```

Explanation of Output for Sample Input 1
For $1, 2$ to be a breadth first search ordering, a road between towns 1 and 2 must be built.

Sample Input 2
```
6 3
1 3
2 6
4 5
```

Output for Sample Input 2
```
2
```

Explanation of Output for Sample Input 2
By building a road between 1 and 2 and a road between 1 and 4, the sequence of distances becomes $0, 1, 1, 1, 2, 2$ which is in non-decreasing order.

2021 Canadian Computing Olympiad
Day 2, Problem 3
Loop Town

Time Limit: 4 seconds

Problem Description

Some cities have complicated road networks that require advanced graph theory to analyze. But not Loop Town! Loop Town has a single circular road that loops around the town. It has N residents that live in N distinct houses located around the road. The town also has N offices, and each resident works at a distinct office.

The road in Loop Town has length L. The location of each building will be represented by an integer between 0 and $L - 1$. Since the road is circular, the positions 0 and $L - 1$ are adjacent. It is guaranteed that the locations of all $2N$ buildings will be distinct.

Every morning, all N residents simultaneously exit their houses onto the road. They then need to walk along the road to the entrance of the office where they work. When each resident has reached the entrance of their office, they all enter simultaneously.

However, a pandemic has now come to Loop Town, disrupting this usual routine. To prevent the spread of disease, residents must now observe social distancing while walking to work. Since the loop road is rather narrow, this means that it is far more inconvenient for two people to cross each other on their way to work (one person must temporarily step off the path to let the other pass). What is the minimum total number of crossings, assuming all the residents work together to achieve this?

Input Specification

The first line contains the two integers N and L ($1 \leq N \leq 1\,000\,000$, $1 \leq L \leq 10^9$).

The i-th of the next N lines contains the two integers a_i and b_i ($0 \leq a_i, b_i < L$), where a_i and b_i represent the locations of the i-th resident's house and office respectively. It is guaranteed that all $2N$ locations are distinct.

For 12 of the 25 available marks, $N \leq 5\,000$.

For an additional 6 of the 25 available marks, $N \leq 100\,000$.

Output Specification

On a single line, output the minimum total number of crossings.

Sample Input 1

```
3 100
10 50
```

```
30 20
60 40
```

Output for Sample Input 1

```
0
```

Explanation of Output for Sample Input 1

Since the road is circular, nobody needs to cross each other.

Sample Input 2

```
4 100
30 70
10 12
60 75
90 50
```

Output for Sample Input 2

```
1
```