Central-European Olympiad in Informatics, CEOI 2020, Day 2 (IOI, Unofficial Mirror Contest, Unrated)

A. The Potion of Great Power

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

Once upon a time, in the *Land of the Shamans*, everyone lived on the *Sky-High Beanstalk*. Each shaman had a unique identifying number i between 0 and $N-1$, and an altitude value H_i, representing how high he lived above ground level. The distance between two altitudes is the absolute value of their difference.

All shamans lived together in peace, until one of them stole the formula of the world-famous *Potion of Great Power*. To cover his/her tracks, the *Thief* has put a *Curse* on the land: most inhabitants could no longer trust each other...

Despite the very difficult circumstances, the *Order of Good Investigators* have gained the following information about the *Curse*:

- When the *Curse* first takes effect, everyone stops trusting each other.
- The *Curse* is unstable: at the end of each day (exactly at midnight), one pair of shamans will start or stop trusting each other.
- Unfortunately, each shaman will only ever trust at most D others at any given time.

They have also reconstructed a log of who trusted whom: for each night they know which pair of shamans started/stopped trusting each other.
They believe the *Thief* has whispered the formula to an *Evil Shaman*. To avoid detection, both of them visited the home of one of their (respective) trusted friends. During the visit, the *Thief* whispered the formula to the *Evil Shaman* through the window. (Note: this trusted friend did not have to be home at the time. In fact, it's even possible that they visited each other's houses – shamans are weird.)

Fortunately, whispers only travel short distances, so the *Order* knows the two trusted friends visited (by the *Thief* and the *Evil Shaman*) must live very close to each other.

They ask you to help with their investigation. They would like to test their suspicions: what if the *Thief* was x, the *Evil Shaman* was y, and the formula was whispered on day v? What is the smallest distance the whispered formula had to travel? That is, what is the minimum distance between the apartments of some shamans x' and y' (i.e. $\min\left(|H_{x'} - H_{y'}|\right)$), such that x' was a trusted friend of x and y' was a trusted friend of y on day v?

They will share all their information with you, then ask you a number of questions. You need to answer each question immediately, before receiving the next one.

Interaction

The interaction will begin with a line containing N, D, U and Q ($2 \le N \le 100000$, $1 \le D \le 500$, $0 \le U \le 200000$, $1 \le Q \le 50000$) – the number of shamans, the maximum number of trusted friends a shaman can have at any given point, the number of days, and the number of questions.

On the next line N space separated integers will follow, the ith ($1 \le i \le N$) of which being H_{i-1} ($0 \le H_{i-1} \le 10^9$), the altitude of shaman $i-1$.

On the next U lines there will be two integers each, on the ith ($1 \le i \le U$) A_i and B_i ($0 \le A_i, B_i < N$ and $A_i \ne B_i$), which represents a pair of shamans who started or stopped trusting each other at the end of day $i-1$. That is, if A_i and B_i trusted each other on day $i-1$, they did not trust each other on day i, or vice versa. Read all of these integers.

The interactor now will ask you Q question, so the following interaction should happen Q times:

1. Read 3 integers describing the current query: x, y and v ($x \ne y$, $0 \le x, y < N$ and $0 \le v \le U$), where x is the suspected *Thief*, y is the suspected *Evil Shaman*, and v is the suspected day..
2. Then print the answer to this query on a single line, i.e. you should print the minimum distance the whispered formula had to travel from some trusted friend x' of x to a trusted friend y' of y.
    - *In case someone trusted both x and y (i.e. $x' = y'$), you should print 0.*
    - *If x or y had no trusted friends, print 10^9.*

After printing each line do not forget to output end of line and flush the output. Otherwise, you will get `Idleness limit exceeded`. To do this, use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;
- `flush(output)` in Pascal;
- `stdout.flush()` in Python;

- see documentation for other languages.

Scoring

Subtask	Points	Constraints
1	0	samples
2	17	$Q, U \leq 1000$
3	14	$v = U$ for all questions
4	18	$H_i \in \{0, 1\}$ for all shamans i
5	21	$U, N \leq 10000$
6	30	no additional constraints

Example

input

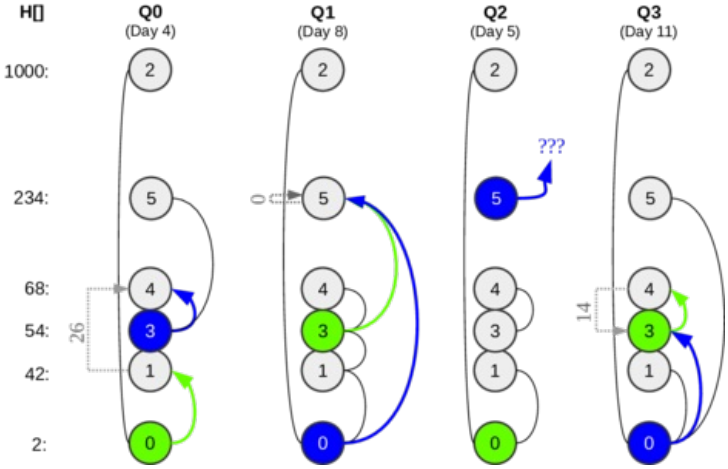
```
6 5 11 4
2 42 1000 54 68 234
0 1
2 0
3 4
3 5
3 5
1 3
5 3
0 5
3 0
1 3
3 5
0 3 4
3 0 8
0 5 5
3 0 11
```

output

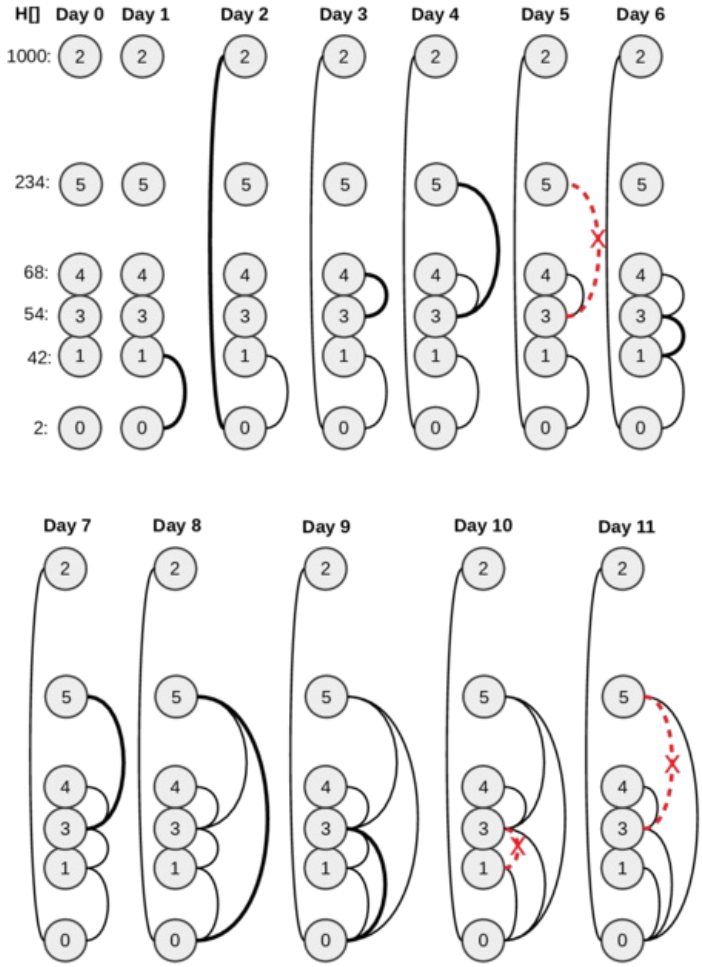
```
26
0
1000000000
14
```

Note

Example queries:



Evolution of friendships:



B. Spring cleaning

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Spring cleanings are probably the most boring parts of our lives, except this year, when Flóra and her mother found a dusty old tree graph under the carpet.

This tree has N nodes (numbered from 1 to N), connected by $N-1$ edges. The edges gathered too much dust, so Flóra's mom decided to clean them.

Cleaning the edges of an arbitrary tree is done by repeating the following process: She chooses 2 different leaves (a node is a leaf if it is connected to exactly one other node by an edge), and cleans every edge lying on the shortest path between them. If this path has d edges, then the cost of cleaning this path is d.

She doesn't want to harm the leaves of the tree, so she chooses every one of them **at most once**. A tree is cleaned when all of its edges are cleaned. The cost of this is the sum of costs for all cleaned paths.

Flóra thinks the tree they found is too small and simple, so she imagines Q variations of it. In the i-th variation, she adds a total of D_i extra leaves to the **original** tree: for each new leaf, she chooses a node from the **original** tree, and connects that node with the new leaf by an edge. Note that some nodes may stop being leaves during this step.

For all these Q variations, we are interested in the minimum cost that is required to clean the tree.

Input

The first line contains two space-separated integer, N and Q ($3 \leq N \leq 10^5$, $1 \leq Q \leq 10^5$) – the number of nodes the tree has and the number of variations.

Each of the next $N-1$ lines contains two space-separated integers u and v denoting that nodes u and v are connected by an edge ($1 \leq u, v \leq N$).

The next Q lines describe the variations. The first integer in the ith line is D_i ($1 \leq D_i \leq 10^5$). Then D_i space-separated integers follow: if the jth number is a_j, it means that Flóra adds a new leaf to node a_j ($1 \leq a_j \leq N$). We may add more than one leaf to the same node. $\sum_1^Q D_i \leq 10^5$ i.e. the sum of D_i in all varations is at most 10^5.

After each variation, Flóra restarts and adds extra leaves to the **original** tree.

Output

You should print Q lines. In the i-th line, print a single integer: the minimum cost required to clean the i-th variation of the tree. If the tree cannot be cleaned, print -1.

Scoring

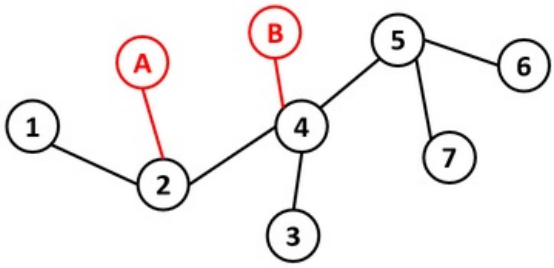
Subtask	Points	Constraints
1	0	samples
2	9	$Q = 1$, there is an edge between node 1 and i for every i $(2 \leq i \leq N)$, Flóra can't add extra leaf to node 1
3	9	$Q = 1$, there is an edge between node i and $i + 1$ for all $(1 \leq i < N)$, Flóra can't add extra leaf to node 1 nor node N
4	16	$N \leq 20000, Q \leq 300$
5	19	the original tree is a perfect binary tree rooted at node 1 (i.e. each internal node has exactly 2 children, and every leaf has the same distance from the root)
6	17	$D_i = 1$ for all i
7	30	no additional constraints

Example

input
7 3 1 2 2 4 4 5 5 6 5 7 3 4 1 4 2 2 4 1 1
output
-1 10 8

Note

The following picture shows the second variation. A possible solution is to clean the path between leaves $1 - 6$, $A - 7$ and $B - 3$.



You can download the above example and an additional (bigger) sample input here: https://gofile.io/d/8QlbsS

C. Chess Rush

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

The mythic world of Chess Land is a rectangular grid of squares with R rows and C columns, R being greater than or equal to C. Its rows and columns are numbered from 1 to R and 1 to C, respectively.

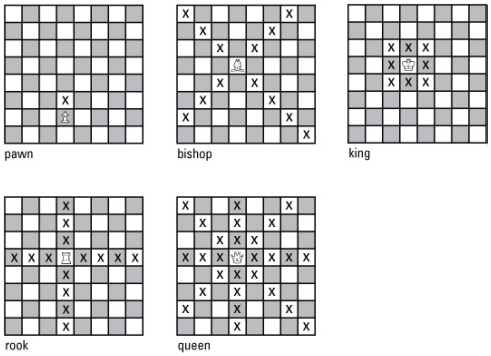
The inhabitants of Chess Land are usually mentioned as pieces in everyday language, and there are 5 specific types of them roaming the land: pawns, rooks, bishops, queens and kings. Contrary to popular belief, chivalry is long dead in Chess Land, so there are no knights to be found.

Each piece is unique in the way it moves around from square to square: in one step,

- a pawn can move one row forward (i.e. from row r to $r + 1$), without changing columns;
- a rook can move any number of columns left/right without changing rows OR move any number of rows forward/backward without changing columns;
- a bishop can move to any square of the two diagonals intersecting at its currently occupied square;
- a queen can move to any square where a rook or a bishop could move to from her position;

- and a king can move to any of the 8 adjacent squares.

In the following figure, we marked by X the squares each piece can move to in a single step (here, the rows are numbered from bottom to top, and the columns from left to right).



Recently, Chess Land has become a dangerous place: pieces that are passing through the land can get captured unexpectedly by unknown forces and simply disappear. As a consequence, they would like to reach their destinations as fast (i.e. in as few moves) as possible, and they are also interested in the number of different ways it is possible for them to reach it, using the minimal number of steps – because more paths being available could mean lower chances of getting captured. Two paths are considered different if they differ in at least one visited square.

For this problem, let us assume that pieces are entering Chess Land in a given column of row 1, and exit the land in a given column of row R. Your task is to answer Q questions: given the type of a piece, the column it enters row 1 and the column it must reach in row R in order to exit, compute the minimal number of moves it has to make in Chess Land, and the number of different ways it is able to do so.

Input

The first line contains three space-separated integers R, C, and Q ($1 \leq Q \leq 1000$, $2 \leq C \leq 1000$ and $C \leq R \leq 10^9$) – the number of rows and columns of Chess Land, and the number of questions, respectively. Then Q lines follow.

Each line consists of

- a character T, corresponding to the type of the piece in question ('P' for pawn, 'R' for rook, 'B' for bishop, 'Q' for queen and 'K' for king);
- two integers c_1 and c_R, $1 \leq c_1, c_R \leq C$, denoting that the piece starts from the c_1-th column of row 1, and has to reach the c_R-th column of row R.

Output

You have to print Q lines, the i-th one containing two space separated integers, the answer to the i-th question: the first one is the minimal number of steps needed, the second is the number of different paths available using this number of steps. Since the answer can be quite large, you have to compute it modulo $10^9 + 7$.

If it is impossible to reach the target square, output the line "0 0".

Scoring

Subtask	Points	Constraints
1	0	samples
2	8	$T \in \{$'P','R','Q'$\}$, i.e. all pieces are pawns, rooks or queens
3	15	$T =$ 'B' and $C, R \leq 100$
4	22	$T =$ 'B'
5	5	$T =$ 'K' and $C, R \leq 100$ and $Q \leq 50$
6	8	$T =$ 'K' and $C, R \leq 100$
7	15	$T =$ 'K' and $C \leq 100$
8	20	$T =$ 'K'
9	7	no additional constraints

Example

input
8 8 5 P 1 2 R 4 8 Q 2 3 B 3 6 K 5 5
output
0 0 2 2 2 5 2 2 7 393

Note

You can download the above example and an additional (bigger) sample input here: https://gofile.io/d/GDzwfC