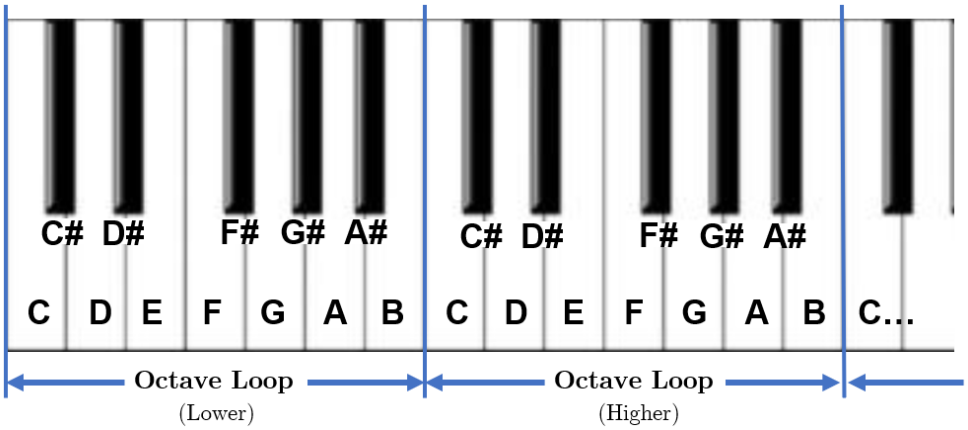
Problem A. Chord

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Lvat is studying music. In this week, he has to figure out what chords are. First of all, he needs to learn the two kinds of most basic chords - the major triad chord and the minor triad chord.

In music theory, every 12 successive notes form a loop (or formally, an **octave**). Practically, we denote each note in an octave from low to high as `C`, `C#`, `D`, `D#`, `E`, `F`, `F#`, `G`, `G#`, `A`, `A#` and `B`. The pitch distance between every two adjacent notes is the same, which is called a **semitone**. Specifically, the pitch distance between note `C` in the higher octave and note `B` in the adjacent lower octave is also one semitone. To conceptualize this, we can mark them on the piano keyboard, one key one note, as the figure shown below. As mentioned, the pitch distance between every two adjacent key is one semitone:



Now Lvat can easily make out the two kinds of most basic chords. You can think of a chord simply as a group of different notes. Specifically, the major triad chord and the minor triad chord are both composed of three different notes, and we denote them from low to high as N_1, N_2 and N_3:

- If the pitch distance between N_1 and N_2 is exactly 4 semitones, and that between N_2 and N_3 is exactly 3 semitones, we call the group of notes a **major triad chord**.
- If the pitch distance between N_1 and N_2 is exactly 3 semitones, and that between N_2 and N_3 is exactly 4 semitones, we call the group of notes a **minor triad chord**.

For example, consider a group of notes of $<N_1 =$ `C`, $N_2 =$ `E`, $N_3 =$ `G`$>$ in the same octave. Note `C` and note `E` have a pitch distance of 4, while note `E` and note `G` have a pitch distance of 3, so it is a major triad chord.

Next consider a group of notes of <`A`, `C`, `E`>, where note `A` is in the lower octave, and note `C` and `E` are in the adjacent higher octave. Note `A` and note `C` have a pitch distance of 3, while note `C` and note `E` have a pitch distance of 4, so it is a minor triad chord.

Now Lvat needs you to help him decide which groups of notes are major triads and which ones are minor triads.

Input

The first line is a single number T ($T \leq 2000$), indicating the number of groups of notes.

For the following T lines, each line contains a group of three notes N_1, N_2 and N_3 (denoted as described before), separated by a single space. It is guaranteed that the three notes are given from low to high, and the pitch distance between N_1 and N_2 is no more than 11 semitones, so is N_2 and N_3.

Output

Output T lines.

The i-th line indicates the answer for the i-th group of notes. If the i-th group constitutes a major triad chord, output "`Major triad`" (without quotation marks), or if it constitutes a minor triad chord, output "`Minor triad`" (without quotation marks), otherwise output "`Dissonance`" (without quotation marks).

Example

standard input	standard output
5 C E G A C E B D F# C F A E D C	Major triad Minor triad Minor triad Dissonance Dissonance

Problem B. Problem Select

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

HOJ is an online judge system for students in HIT to practice. As the administrator, you would always setup contests by selecting problems from the problem set. Problem IDs are integers that can be phased by url. For example, the corresponding problem ID of url `http://acm.hit.edu.cn/problemset/1001` is 1001.

Now the task is, given n urls, print the smallest k problem IDs in increasing order.

Input

The first line of the input contains one integer T ($1 \leq T \leq 10$), indicating the number of test cases.

For each test case, the first line contains two integers, n ($1 \leq n \leq 1000$) and k ($1 \leq k \leq n$).

For the next n lines, each line contains a string indicating the problem's url. The problem ID is guaranteed to be in range $[1, 10000]$ and unique in each case.

Output

Output T lines.

For each test case, you should output k numbers in a line, and there shouldn't be any spaces at the end of the line.

Example

standard input	standard output
2 3 2 http://acm.hit.edu.cn/problemset/1003 http://acm.hit.edu.cn/problemset/1002 http://acm.hit.edu.cn/problemset/1001 4 1 http://acm.hit.edu.cn/problemset/1001 http://acm.hit.edu.cn/problemset/2001 http://acm.hit.edu.cn/problemset/3001 http://acm.hit.edu.cn/problemset/501	1001 1002 501

Problem C. String Game

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Clair and Bob play a game. Clair writes a string of lowercase characters, in which Bob sets the puzzle by selecting one of his favorite subsequence as the key word of the game. But Bob was so stupid that he might get some letters wrong.

Now Clair wants to know whether Bob's string is a subsequence of Clair's string and how many times does Bob's string appear as a subsequence in Clair's string. As the answer may be very large, you should output the answer modulo $10^9 + 7$.

Input

The first line is Clair's string (whose length is no more than 5000), and the second line is Bob's string (whose length is no more than 1000).

Output

Output one line, including a single integer representing how many times Bob's string appears as a subsequence in Clair's string. The answer should modulo $10^9 + 7$.

Examples

standard input	standard output
eeettt et eeettt te	9 0

Problem D. Trie

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 seconds
Memory limit:	256 megabytes

Ken is learning the Trie. He learned that Trie is a rooted tree to store some strings. There is a character on each edge of a Trie. He uses $S(i)$ to denote the string concatenated by the characters on the path from the root to node i.

Now he needs to deal with two kinds of operations:

- **attach**: attach a new mark (different from any marks that already exist) to a set of Trie nodes once at a time. Note that one node may be attached to multiple marks.
- **query**: ask for how many mark properties a certain node matches. We say a node matches a mark property if and only if there exists a node j that has the corresponding mark and $S(j)$ is a suffix of $S(i)$. Different marks corresponds to different mark properties.

Please help him handle these queries.

Input

The first line contains a single integer T ($1 \le T \le 5$), indicating the number of test cases. Then T test cases follows.

The first line of each test case contains a single integer n ($1 \le n \le 10^5$), indicating the number of Trie nodes.

Lines through the second to the n-th ($n-1$ lines in total) describe Ken's Trie. The i-th line contains an integer u ($1 \le u < i$) and a lowercase latin letter c, which means that the father of node i is node u and the character on that edge is c. It is guaranteed that for each node, letters on edges connecting the node and its children are distinct.

The next line contains a single integer m ($1 \le m \le 10^5$), indicating the number of operations.

The next m lines describe all the operations. Each line describle one operation, and it is formatted in one of the two following formats, depends on the kind of it:

- "1 k x_1 x_2 ... x_k" - attach a new mark to the node set $\{x_1, x_2, \ldots, x_k\}$, where all the x_i are distinct.
- "2 x" - query for how many mark properties node x can match.

Output

For each test case, output the answer for each query operation, one answer in a line.

Example

standard input	standard output
1 6 1 a 1 b 1 c 3 a 3 b 7 1 2 2 3 2 5 1 2 3 4 2 6 2 1 1 2 1 2 2 6	1 2 0 3

Problem E. Shorten the Array

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Alice has an array a. The array has N positive numbers. She thinks the array is too long, so she wants to shorten the array. She decides to shorten the array via the following operation: every time she will choose an index i ($1 \le i < n$) which $a_i > 0$ and $a_{i+1} > 0$. She will delete a_i and a_{i+1} and use $(a_i \bmod a_{i+1})$ or $(a_{i+1} \bmod a_i)$ to replace them.

For example, for array $[3, 2, 4, 5]$, if Alice choose $i = 2$, she can change the array to $[3, 2, 5]$ or $[3, 0, 5]$. Alice wants you to tell her the shortest possible length of the array after several options.

Input

The first line contains a single integer T ($1 \le T \le 10$), indicating the number of test cases.

For each test cases:

The first line contains one integer N ($2 \le N \le 10^6$), indicating the size of the array.

The next line contains N integers a_i ($a_i \le 10^9$), representing the array.

Output

Output T lines.

The i-th line contains a single integer, representing the answer of i-th test case.

Example

standard input	standard output
2 4 1 1 1 1 4 2 3 4 5	2 1

Note

For the first sample, Alice first choose $i = 1$ to change the array to $[0, 1, 1]$, then choose $i = 2$ to change the array to $[0, 0]$, which is the best result she can reach.

Problem F. Queue

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

First day of school!

n primary school freshmen are queuing up and each of them owns a student number. Teacher Mr. Cycle finds that the queue is unordered. In order to make the queue more orderly, he has to choose some pairs of students and exchange their positions.

However, Mr. Cycle is a lazy dog and impatient with making plans, so he asks for Miss Ke's help. Specifically, Miss Ke gives Mr. Cycle a plan list of m position pairs, and each pair (p_1, p_2) means that Mr. Cycle can exchange the student at position p_1 and the student at position p_2 in the current queue (positions of the queue are numbered from 1). Mr. Cycle trusts Miss Ke a lot, so he exchanges one by one strictly in the order of the plan list, and writes down the queuing chaotic value V_i after i-th exchange for every $0 \le i \le m$, where V_0 is the chaotic value of the initial queue.

Now Mr. Cycle wants to know the minimal queuing chaotic value $V = \min_{0 \le i \le m}\{V_i\}$ in advance, so that he can stop exchanging at a proper time.

If we denote the student number of the student at position i as s_i, the queuing chaotic value of a certain student queue equals the number of inversions, that is, the number of pairs (i, j) such that $1 \le i < j \le n$ and $s_i > s_j$.

Input

The first line contains one integer T ($1 \le T < 10$) indicating the number of test cases. Then T test cases follows.

The first line of each test case contains one integer n ($1 \le n \le 10^5$) indicating the number of students.

The second line contains n integers indicating the students number, the i-th integer indicates the student number of the student at the i-th position in the initial queue ($0 \le s_i \le 10^5$).

Then third line contains one integer m ($1 \le m \le 1000$) indicating the number of position pairs in the plan list.

For the following m lines, each line describes a position pair (p_{i1}, p_{i2}) ($1 \le p_{i1} \le p_{i2} \le N$, $p_{i2} - p_{i1} \le 100$).

Output

Output T lines.

For each case, output one line containing one integer: the minimal queuing chaotic value V.

Example

standard input	standard output
1 5 5 4 3 2 1 2 1 5 2 4	0

Problem G. Matrix

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Bob is playing a matrix game. He needs to deal with a matrix of n rows and m columns, satisfying the properties below:

- Both rows and columns are numbered from 1
- All the elements in the matrix have only two values: 0 and 1
- All the elements equals 0 initally

To play the game, Bob can apply flip(i, j) operations to the matrix. This operation can flip all elements whose row number is a multiple of i **and** column number is a multiple of j (flipping an element means change its value from v to $1-v$). Bob is very bold when playing games. He always performs flip operations on all the positive integer pairs (i, j).

After finishing all the operations, Bob wants to know how many elements which equals 1 there are in the matrix.

Input

The first line is a single number T, indicating the number of test cases.

In the following T lines, the i-th line contains two integers n, m, representing the number of rows and columns of the i-th matrix, respectively.

It is guaranteed that $1 \leq T \leq 10$ and $1 \leq n, m \leq 10^{18}$.

Output

T lines, the i-th line a single integer - the answer of the i-th matrix.

Example

standard input	standard output
2 1 1 2 3	1 1

Problem H. Curious

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Xiao Ming is very curious about number theory and one day his teacher gives him a very strange homework. He is provided with n positive integers $a_1, a_2, \ldots, a_n$ and it is guaranteed that all numbers will be no greater than m. What's more, his teacher asks him k questions. Each time, the teacher gives a certain number x (also no greater than m) and ask Xiao Ming for the number of pairs (a_i, a_j) that satisfy $\gcd(a_i, a_j) = x$ where $1 \le i, j \le n$.

That is to calculate

$$\sum_{i=1}^{n}\sum_{j=1}^{n}[\gcd(a_i, a_j) = x].$$

It is very difficult for Xiao Ming to figure out the problem and he turn to you for help.

Input

The first line of the input contains one integer $T(1 \le T \le 10)$ - the number of test cases. Then T test cases follows.

The first line of each test case contains three integers n, m, k $(1 \le n, m, k \le 10^5)$ as described above.

The second line contains n integers $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le m)$.

Each of next k lines contains a single integer x.

Output

For each test case, print the answer in one line.

Example

standard input	standard output
1 5 5 5 1 2 3 4 5 1 2 3 4 5	19 3 1 1 1

Note

For $i \ne j$, (a_i, a_j) and (a_j, a_i) are considered different.

Problem I. World Tree

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2.5 seconds
Memory limit:	256 megabytes

The bravest warrior PP got injured in an important battle. To save his life, the King decides to turn to the World Tree for help. The World Tree has much life energy that can cure the injury and one must find the energy point to receive the bless of the World Tree.

On the World Tree, there are n energy points connected by $n-1$ bidirectional branches. Every energy point will have a unique energy value b_i. PP needs to start from the energy point labeled by 1 to find every energy point on the tree to cure himself. After PP arrives at some energy point i, the life energy will perceive the existence of PP and its energy value will change into a_i. This change will result in a great much life energy and PP will get healed with it.

Specifically, the exact energy value he can get at a point is the product of the energy value at the point and the sum of the energy value of all points he has not visited. That is $a_i \times (\sum_j b_j)$ where point i is the current point and j iterates over all the points that has not been visited.

The principle PP chooses the path is that PP only considers the directions that he is able to get more energy. That is to say, PP will only return to the previous energy point when there is no more energy points in other directions.

PP need you to help him find the best path to get the most life energy value from the World Tree.

Input

The first line of the input contains one integer T ($1 \le T \le 10$), indicating the number of test cases.

Then T test cases follow.

The first line of each test case contains a single integer n ($1 \le n \le 10^5$) - the number of energy points.

The second line contains n integers $b_1, b_2, \ldots, b_n$ ($1 \le b_i \le 100$) - the energy value before changing. The third line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 100$) - the energy value after changing.

Each of the next $n-1$ lines contains two integers x, y ($1 \le x, y \le n$), indicating the two energy point connected by the branch.

Output

Output T lines. For each test case, output a single integer in one line, indicating the maximal life energy value PP can get.

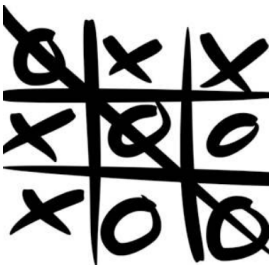
Example

standard input	standard output
1 3 1 2 3 3 4 5 1 2 1 3	27

Problem J. Situation

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Alice and Bob are really bored today, they want to play Tic-tac-toe.



The rules are basically the same as the general Tic-tac-toe chess. Two players, Alice and Bob, take turns to play on a 3×3 board. Alice goes first. The goal is to full fill their own chess pieces in a certain row, a certain column or a certain diagonal to form a continuous connection.

In our problem, the rules are somewhat different. Even if one player has reached a row (or a column or diagonal) connection, the game will continue until all the nine position are placed. Finally, we denote the number of connections of Alice minus the number of connections of Bob as the final score of the game. Alice wants to maximize the total score, while Bob wants to minimize it.

Due to some mysterious power, there have some chess pieces on the chessboard already. Alice and Bob want you to help calculate the final score starting with the given situation.

You can assume that Alice and Bob are both very smart.

Input

The first line of the input contains one integer T ($1 \leq T \leq 40000$), indicating the number of test cases.

For each test case, the first line is an integer 0 or 1 where 1 means it is Alice's turn to play and 0 means it is Bob's turn to play.

In the next three lines, each line contains 3 characters representing the situation of the chessboard:

- '`.`' represents a vacant position;
- '`O`' represents Alice's chess piece has been placed;
- '`X`' represents Bob's chess piece has been placed.

Output

Output T lines.

For each test case, output one line containng one integer to represent the final score starting with the given situation.

Example

standard input	standard output
2 0 .OO X.O OXO 1 XXX XXX XXX	2 -8

Problem K. Forager

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Forager is a pixel style 2D adventure game, players can do farming, do fishing, kill monsters and cut trees etc. In addition, some temples will appear in some lands, you can explore them and get a lot of items. Mr. D is interested in this game recently and has no intention to train on algorithm.

Today, Mr. D goes to The Frost Temple in *Forager* and finds lots of ray sources, turning stones and gem stones. The Frost Temple can be considered as an $n \times m$ map, each grid may be empty, ray source, turning stone, gem stone or wall. The characteristics of each item are listed below.

- **Empty grid.** The ray can go through it directly, multiple rays are allowed to go through it.
- **Ray source.** A ray source can shoot a beam of ray in a certain direction (up, down, left or right), and also **can** be passed like an empty gird.
- **Turning stone.** It can absorb light from all directions to it and shoot it in a certain direction (up, down, left or right). It costs you some value to rotate the shoot direction 90 degrees **clockwise**.
- **Gem stone.** It has a lot of treasures but the value of a gem stone can be taken if and only if there is a beam of ray to it, and it does **not** disappear after you take its value. It can stop ray from all directions, which means that the ray **can't** go through it. The value of a gem stone can only be taken **at most once**.
- **Wall.** It can stop ray from all directions, it means that the ray **can't** go through it.

Mr. D can rotate some turning stones (if needed) and get the value of some gem stones. He has not trained for algorithm for a long time, but he wants to get the value as much as possible, so he is asking for your help.

Given the map of The Frost Temple, you should tell Mr. D the maximum value he can earn.

Note: Mr. D has infinite value.

Input

First line contains a single integer $T(1 \le T \le 20)$ representing the number of test cases.

For each case:

The first line contains four integers n, m, k, l $(1 \le n, m \le 50, 0 \le k, l \le n \times m)$, represents the size of the Frost Temple, the number of gem stones and the number of turning stones.

The following n lines, each line contains m characters, represents the map of the Frost Temple. For each character:

- '`.`' represents the empty grid.
- '`U`', '`D`', '`L`', '`R`' represents there's a ray source, and the direction of it (up, down, left or right).
- '^' (the character above '`6`' on the keyboard), '`v`', '`<`', '`>`' represents there's a turning stone, and the initial direction of it (up, down, left or right), its cost is given in the following l lines.
- '`x`' represents there's a gem stone, its value is given in the following k lines.
- '`#`' represents there's a wall.

The following k lines, each line contains three integers x_i, y_i, v_i ($1 \le x_i \le n, 1 \le y_i \le m, 0 \le v_i \le 10^9$), representing that the value of (x_i, y_i) is v_i.

The following l lines, each line contains three integers x_i, y_i, z_i ($1 \le x_i \le n, 1 \le x_i \le m, 0 \le c_i \le 10^9$), representing that the cost of (x_i, y_i) is c_i.

It is guaranteed that all the positions appear in the last $k + l$ lines are distinct.

Output

For each test, output an integer represents the maximum value Mr. D earned.

Example

standard input	standard output
5 1 5 1 0 R...x 1 5 3 1 5 1 1 x.R.v 1 1 5 1 5 1 1 5 1 1 x.R.v 1 1 1 1 5 5 3 5 1 4 >L#R> .###. <.x.> 3 3 100 1 1 3 1 5 1 3 1 0 3 5 0 3 5 1 5 .v<.. R.>.> .>..x 3 5 100 1 2 999 1 3 999 2 3 1 2 5 3 3 2 999	3 4 0 99 97

Note

For the second case, you can rotate $(1, 5)$ then you get $-1 + 5 = 4$ value.

For the last case, the optimal solution is rotating $(2, 3)$ 3 times or rotating $(2, 5)$ 1 time, then you get $-1 \times 3 + 100 = 97$ value.

Problem L. Swimmer

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

One day, n of swimmers hold a wonderful competition in a swimming pool.

There are n lanes in the swimming pool numbered from 1 to n , and the their lengths are all m meters. Swimmers are also numbered from 1 to n, and swimmer i occpies lane i. The swimmers will start at position 0, and turn back immediately when arriving at the turn line at position m.

They will keep swimming. The speed of swimmers may be different. The speed of the swimmer i is x_i meters per second.

You need to answer q questions: each question gives two integers p and k, asking for the position of swimmer k after p seconds.

Input

The first line contains three integers n, m, q ($1 \le n \le 10^6, 1 \le m \le 10^9, 1 \le q \le 10^6$), indicating the number of lanes, the length of lanes and the number of questions, respectively.

The second line contains n integers $x_1, x_2, \ldots, x_n$ ($1 \le x_i \le 10^9$), indicating the speed of each swimmer.

Each of the following q lines contains two integers p_i, k_i ($0 \le p_i \le 10^9, 1 \le k_i \le n$) representing the i-th question.

Output

Output q lines. Each line contains a single integer, which is the answer of the i-th question.

Example

standard input	standard output
1 3 2 1 5 1 7 1	1 1

Problem M. Upanishad

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Bob has a sequence $\{a_i\}$ of length n. He has to answer q queries. The i-th query gives two positive integers l_i and r_i. Bob first need to write down all the elements which have even number of occurrences in the subsequence $a_{l_i}, a_{l_i+1}, \ldots, a_{r_i}$ (note that each element should be written for only once). After that, he need to calculate the XOR sum of the elements he writes down as the answer to the query.

We define XOR sum of a sequence $x_1, x_2, \ldots x_m$ equals to $x_1 \operatorname{xor} x_2 \operatorname{xor} \ldots \operatorname{xor} x_m$, where xor is the operator of exclusive bitwise OR. If the sequence is empty, the XOR sum is 0.

Please help Bob to deal with these queries.

Input

The first line contains two integers n, q - the number of elements in the sequence and the number of queries.

The second line contains n positive integers - the sequence $\{a_i\}$.

Each of the following q lines describes a query by a pair of integers l_i and r_i ($1 \le l_i \le r_i \le n$).

It is guaranteed that $n \le 500000, q \le 500000, a_i \le 10^9$. The number of test cases is no more than 10.

Output

Output q lines. Each line contains one integer. The i-th line answers the i-th query.

Example

standard input	standard output
4 2 1 2 4 2 1 3 1 4	0 2
3 2 1 1 1 1 3 2 3	0 1

Problem N. Expressway

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Bob likes his girlfriend very much, but unfortunately she doesn't live with him. Bob lives in building 1, while his girlfriend lives in building n. From time to time, Bob would visit his girlfriend at her house. Bob and his girlfriend live in a very big city. The whole city can be seen as a big undirected graph with n buildings and m streets. Each street connects two different buildings.

However, the evil spirit did not like lovers anymore. Every time Bob tried to see his girlfriend, he would blow up a street. The street wouldn't be fixed until the next day, and Bob couldn't pass the street that time. So he wanted to know the shortest distance from his home to his girlfriend home when the street was blown up.

Input

The first line contains three integers $n(2 \leq n \leq 10^5)$, $m(1 \leq m \leq 2 \times 10^5)$ and $q(1 \leq q \leq 2 \times 10^5)$: the number of buildings, the number of streets and the number of queries.

Each of the next m lines contains three integers u_i, v_i and $w_i(1 \leq u_i, v_i \leq n, u_i \neq v_i, 1 \leq w_i \leq 10^9)$ indicating the i -th street that connect vertices u_i and v_i with length w_i.

Each of the next q lines contains an integer $e_i(1 \leq e_i \leq m)$, which asks the shortest path from 1 to n if street e_i is broken.

Output

Output q lines.

Each line contains an integer representing the length of the shortest path from 1 to n if street e_i is broken. If there is no path from 1 to n, output -1.

Examples

standard input	standard output
5 6 3 1 2 3 4 5 19 1 3 19 2 4 20 5 4 19 4 5 11 6 1 4	42 -1 -1
5 10 5 1 2 58 1 3 91 2 5 85 3 5 27 2 5 73 1 4 86 1 4 65 5 2 85 1 5 33 2 5 73 3 8 8 5 9	33 33 33 33 118
8 14 3 8 7 2 1 8 6 7 6 2 1 7 10 6 5 2 1 6 14 5 4 2 1 5 18 4 3 2 1 4 22 3 2 2 1 3 26 2 1 2 1 2 30 8 10 4	6 6 6