Problem A. Artifact

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

You and your friend are playing a game called Dota 2 and you want to do a one versus one match. You are given N artifacts at the beginning of the game and you need to give some of them to your friend. These artifacts can be combined to create spells. A spell can be constructed with k artifacts a_1, a_2, ..., a_k if $k \geq 2$ and artifact a_1 can be combined with artifact a_2, artifact a_2 can be combined with artifact a_3, ..., artifact a_{k-1} can be combined with artifact a_k. Your friend is a professional player, so if he has **at least** 2 **distinct** artifacts belonging to a spell construction, he will be able to build the whole spell. Because you are not so good at this game, if your friend can build at least one spell, he will beat you.

You need to give to your friend the maximum number of artifacts, such that he will not be able to create any spell with these artifacts, having in mind that he needs only two **distinct** artifacts from a spell construction in order to create the whole spell.

Input

The first line of the input will contain the number N ($1 \leq N \leq 3000$), representing the number of artifacts, and the number M ($1 \leq M \leq 20000$), representing the number of pairs of artifacts that can be combined.

The following M lines will contain two numbers x_i ($1 \leq x_i \leq N$) and y_i ($1 \leq y_i \leq N$) each, $x_i \neq y_i$, representing that artifact x_i can be combined with artifact y_i (this does not mean that artifact y_i can be combined with artifact x_i).

Output

You should output one line containing one non-negative integer: the maximum number of artifacts that you can give to your friend, such that he will not be able to create any spell.

Example

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

Note

2 is the maximum number of artifacts that you can give to your friend.

For example, you can give to your friend artifact 3 and 5. Note that we cannot give him artifact 4 as well, because that would mean he would be able to create the spell $4, 1, 2, 3$.

Problem B. Broken Python

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Jimmy was messing around with the *eval*() function in python (he's too lazy to use a calculator), and he noticed that for some inputs he's running into issues with the recursion stack and hard-coded memory limits.

While he's desperately searching Stack Overflow for a solution to his problem, give him a hand and evaluate some expressions for him.

Input

The input consists of one line with the length $s \leq 10^5$ with integers and symbols. There are no white-space characters.

The allowed symbols are +, -, *, /, ^, (,). As a reminder, the order of the operations is PEMDAS: parenthesis ('(', ')'), exponents ('^'), multiplication ('*'), division ('/'), addition ('+'), subtraction ('-'). For operations with the same priority($+-$, $*/$) we evaluate left to right.

For every integer x in the input, $-10^9 < x < 10^9$.

If there are any expressions of the form a^b in the input, we guarantee b is a positive integer and not an expression.

Moreover, it is guaranteed that all unary operators are surrounded by parantheses.

Output

You should output the result to the given mathematical expression. We guarantee that $-10^9 < result < 10^9$ and that *result* is an integer, but the intermediate results are unbounded and could be rational numbers.

Examples

standard input	standard output
1+1-1	1
35^66*0/35	0
(1/2)*2	1

Problem C. Flash

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.5 seconds
Memory limit:	256 megabytes

Flash is a prisoner in Penguin's Cave. The Cave can be represented like a NxN matrix with the columns and lines numbered from 1 to N. Flash can be found in the cell (1,1). Flash can travel to the cell (i,j) only from the cell (i-1,j) or (i,j-1). Penguin told Flash that he will be freed only if he will solve an easy problem.

Penguin gives him all the numbers d from 1 to N+N-2 and asks Flash to start from (1, 1) and to mark all cells that can be reached. A cell (i, j) can be reached if and only if (i, j) = (1, 1) or there is another cell (i', j') that can be reached and the Manhattan distance between (i', j') and (i, j) is d.

In the end, Penguin wants to know how many cells will be marked an odd number of times.

Input

The first line of the input will contain a single number N. ($1 \leq N \leq 1.000.000.000$).

Output

The output should contain a single line with a single number representing the number of cells that will be marked by an odd number of times.

Examples

standard input	standard output
4	5
233	1974

Note

In the first example, Flash will count the cells: (1,2) , (2,1) , (2,4) , (3,3) and (4,2).

Problem D. Galleries

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

You own one of the biggest art museums in the country!

The museum consists of $1 \leq N \leq 10^5$ galleries and $1 \leq M \leq 2 * 10^5$ corridors. You may walk from a gallery to another one if there is a corridor in between them.

Each gallery has associated to it a value $1 \leq V_i \leq 10^9$. This week you decide to host a mega promotion: each visitor must only pay a cost equal to the biggest value among the galleries he decides to visit!

Now, in order to see how much money you might lose, you ask yourself $1 \leq Q \leq 10^5$ questions of type X, K and you want to know, for a person that enters the museum through gallery X, what is the minimum cost of a pass he must purchase in order to visit at least K distinct galleries(including X). It is guaranteed that for each query $1 \leq X \leq N$ and $1 \leq K \leq N$.

You may walk from a gallery to another only if there exists a corridor in between the two. You may not enter a gallery without visiting it and you can pass through the same one multiple times(though it only counts towards the answer once). You also have to stay inside of the galleries at all times(meaning you can't exit the museum and enter through another gallery).

Input

The first line of input consists of 2 numbers: N and M, the number of galleries and corridors respectively.

The next lines contains N integers, the values of the galleries.

The following M lines each contain 2 integers X and Y describing an undirected corridor between the 2 galleries.

The next line contains one number Q, the number of questions.

Each of the next Q lines describes one of the questions.

Output

Your output should consist of Q lines, the answer to the questions in the order they appear in the input. On each such line, there should be one integer representing the minimum cost of a pass.

It is guaranteed that you may reach any gallery from every other one.

Example

standard input	standard output
10 14 517439869 911708498 120170079 452675108 1 3 7 10 6 1 2 3 3 4 10 6 8 4 5 1 6 1 1 5 6 9 6 5 9 5 5 1 5 4 10 3 5 4 5 8 8 3 6	911708498 517439869 517439869 517439869 517439869 704687086 127806914 462723433 303807567 59344841 2318459

Problem E. OneZeroTree

Input file:	standard input
Output file:	standard output
Time limit:	3 seconds
Memory limit:	256 megabytes

It's Christmas time! You and your friends love Christmas, and you decide to go to see the Christmas Tree from your city fair. With this opportunity, you want to show off to your friends your programming skills.

The Christmas Tree can be represented as a Tree with $1 \leq N \leq 10^5$ vertices indexed from 1 to N (a connected graph with N vertices and $N-1$ edges). Each node can be turned on or turned off. A configuration is defined as a simple path from vertex x to vertex y ($x \leq y$) in which every vertex can be turned on or off. Two configurations are different if their associated paths are different or if there is at least one vertex that has different states in the two configurations (i.e., it is turned on in one configuration and turned off in the other one). The cost of a configuration is the number of vertices that are turned on.

For each k from 0 to N, you need to find out the number of configurations of cost equal to k. Because this number can be very large, you need to compute it modulo 998244353.

Input

The first line of the input will contain the number N ($1 \leq N \leq 10^5$), representing the number of vertices of the Tree.
The following $N-1$ lines will contain two numbers x_i ($1 \leq x_i \leq N$) and y_i ($1 \leq y_i \leq N$) each, representing an edge between x_i and y_i.

Output

You should output one line consisting of $N+1$ numbers, representing the answer for each k from 0 to N, in this order.

Example

standard input	standard output
4 1 2 2 3 2 4	10 19 12 3 0

Note

For k = 2 :

the path from 1 to 1 has 0 ways of turning on 2 vertices

the path from 2 to 2 has 0 ways of turning on 2 vertices

the path from 3 to 3 has 0 ways of turning on 2 vertices

the path from 4 to 4 has 0 ways of turning on 2 vertices

the path from 1 to 2 has 1 way of turning on 2 vertices

the path from 1 to 3 has 3 ways of turning on 2 vertices

the path from 1 to 4 has 3 ways of turning on 2 vertices

the path from 2 to 3 has 1 way of turning on 2 vertices

the path from 2 to 4 has 1 way of turning on 2 vertices

the path from 3 to 4 has 3 ways of turning on 2 vertices
In total there are 12 configurations with cost 2.

Problem F. Predictable Tree

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

Peter's tree has now come to a stable state, it doesn't give mixed signals anymore and wants to change. Our boy thinks that the best change is a look change, so he will help his friend adjust the colors in its nodes. The tree initially has colour 0 in all its nodes, but it doesn't want to change so much. Hence, Peter will colour only a path in the tree with colour 1. A colouring of a tree is defined as the sequence of colours in the nodes of the tree taken in increasing order from 1 to N.

Everyone knows that this year the number K is trending, so Peter wants a path such that the final colouring is the K-th lexicographically smallest one. Find the endpoints of this path.

Input

The first line of the input contains a number $1 \leq T \leq 10^{18}$, the number of tests.

The first line of each test contains 2 integers $1 \leq N \leq 500.000$ (the number of nodes in the tree) and $1 \leq K \leq \frac{N(N+1)}{2}$.

The following $N - 1$ lines will each contain a pair of integers (a, b) signifying that there is an edge between node a and node b $(1 \leq a, b \leq N)$.

It is guaranteed that the sum of all N is less than 500.000.

Output

The output should contain T lines, each containing an answer to a test, the endpoints of the path that gives the K-th smallest lexicographically colouring. The endpoints should be printed in **non-decreasing** order.

Example

standard input	standard output
7 3 1 1 2 1 3 3 2 1 2 1 3 3 3 1 2 1 3 3 4 1 2 1 3 3 5 1 2 1 3 3 6 1 2 1 3 5 11 3 2 5 4 1 5 1 2	3 3 2 2 1 1 1 3 1 2 2 3 2 5

Note

For the tree containing 3 nodes and edges (1,2) and (1,3) there are 6 different possible paths that each determine a different colouring.

The path (3,3) gives the colouring 001.

The path (2,2) gives the colouring 010.

The path (1,1) gives the colouring 100.

The path (1,3) gives the colouring 101.

The path (1,2) gives the colouring 110.

The path (2,3) gives the colouring 111.

Problem G. Puppetteer

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

You are given a rooted tree with N nodes, each node containing a distinct integer value p_i from the interval $[1, N]$. The root of the tree is vertex 1.

An interval $[a, b]$ is called compact for a set S if all integers from a to b are present in S, but $a - 1$ and $b + 1$ are not present in S.

For each vertex, find the number of compact intervals formed by the values in its subtree (including that vertex).

Input

The first line of the input contains a number $1 \leq T \leq 10^{18}$, the number of tests.

For each test, the first line contains an integer $1 \leq N \leq 250.000$, the number of vertices in the tree.

The next line contains N numbers $1 \leq p_i \leq N$, representing the values for each vertex from 1 to N. All values p_i are pairwise distinct.

The following $N - 1$ lines will each contain a pair of integers (a, b) signifying that there is an edge between node a and node b $(1 \leq a, b \leq N)$.

It is guaranteed that the sum of all N is less than 500.000.

Output

The output should contain T lines, containing the answer for each test, the line i having N_i numbers, representing the number of compact intervals in the subtree of each vertex from 1 to N_i.

Example

standard input	standard output
2 8 3 2 7 6 4 1 8 5 1 3 2 3 2 6 3 4 3 5 6 8 7 8 7 6 3 1 7 4 2 5 1 7 2 3 2 7 3 5 3 6 4 7	1 3 2 1 1 3 1 2 1 1 2 1 1 1 2

Problem H. Shopping Bags

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

Since Jimmy and Johnny are not allowed to leave their home, they have created a game out of boredom, but because they didn't have many things to play with in the house, they have used shopping bags. An interesting property of shopping bags is that they can fit inside other bags which can fit inside others and so on. So the 2 kids have used N bags, indexed from 1 to N, deciding for each one whether to put it in another bag or not.

After finishing the slow process of putting bags inside each other, the kids can start playing. They take turns, Jimmy being the first one to make a move. In a move, a kid can choose a bag that is still in the game and remove it together with all the bags inside it out of the game. The loser is the player that has no more bags to choose from when it is his turn.

Jimmy is a very competitive player, so he needs your help to tell him if, for a certain configuration of bags, he has a safe winning strategy or not.

Input

The first line of the input will contain one positive integer N ($1 \leq N \leq 1000$) that represents the number of shopping bags.

The next line will contain N integers $b_1, b_2, b_3, ..., b_n$, where b_i ($0 \leq b_i \leq N$) represents the bag that the i^{th} bag is directly inside in. If b_i is equal to 0 that means that the i^{th} bag isn't inside any other bag.

Output

The output should be either YES if Jimmy has a safe winning strategy or NO if he doesn't.

Examples

standard input	standard output
5 0 1 2 3 4	YES
6 0 1 2 2 0 5	NO
5 0 1 1 0 4	YES

Note

In the first example Jimmy can choose to take out bag number 1 as his first move, making Johnny lose.

In the second example, if Johnny plays optimal, Jimmy will lose no matter what his strategy is.

In the third example, Jimmy has a safe winning strategy.

Problem I. Soldiers

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

To prepare for emergencies, the king has given you the most important task: improving the defence of the country's army!

The country can be visualized as a graph with $1 \leq N \leq 10^5$ nodes and $1 \leq M \leq 2 * 10^5$ undirected edges. Each node also has a risk $1 \leq R[i] \leq 10^8$ of collapsing.

Among this country, there are also $1 \leq S \leq 2 * 10^5$ soldiers. For each of them you know the node they are situated in and also its type. All soldier types are numbers that fit within 32 bit signed integers.

In case of emergency, each soldier should have an emergency contact to help. Therefore, you must construct a matching of all soldiers such that you only match soldiers of the same type and each soldier is matched to exactly one other soldier. It is guaranteed that such matching exists.

Furthermore, you would like to minimize the sum of risks for each pair in the matching. The risk of a pair is defined as the sum of risks of all the nodes X such that if X is removed from the graph, the two paired soldiers can no longer reach each other(this includes the nodes where the soldiers are initially situated in). If 2 soldiers are situated in the same node, the risk of their pairing is the risk of that node.

Input

The first line of the input will contain 2 numbers N and M, representing the number of nodes and edges respectively.

The following line will contain N numbers, the array R.

The next M lines each contain 2 numbers X and Y, meaning there is an undirected edge between X and Y.

The next line contains S, the number of soldiers. The following S lines each contain a pair of numbers: the node in which the soldier is situated and its type.

Output

You should output one line containing one integer: the minimum risk of a valid pairing. You may assume that there always exists a valid pairing.

Example

standard input	standard output
9 10 1 2 3 4 5 6 7 8 9 1 2 1 3 2 3 2 4 4 5 5 6 5 7 6 7 7 8 7 9 6 1 1 2 1 4 2 5 1 6 1 8 2	38

Note

To obtain this answer, you should pair soldiers in nodes 1-2, 4-8 and 5-6.

For pair 1-2, the nodes we sum the risks of are 1 and 2.

For pair 4-8, the nodes we sum the risks of are 4, 5, 7 and 8.

Finally, for pair 5-6, the nodes we sum the risks of are 5 and 6.

Problem J. Statue

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	256 megabytes

Jimmy is the governor of a very important city in AGM land. He recently listened to a beautiful song called `"Vreau sa-mi fac statuie in fata portii"` (which translates to `"I want to make a statue of myself in front of the gate"`) and now Jimmy wants to make a statue of himself somewhere in the city so people can admire his face even more.

In Jimmy's city there are N ($1 \leq N \leq 10^5$) houses positioned in a 2D plane. House number i has coordinates (x_i, y_i), where $0 \leq x_i \leq 100$ and $0 \leq y_i \leq 10^9$. In addition, there are M ($1 \leq M \leq 10^5$) roads, each joining a pair of houses. The governor knows how many people travel everyday on each road. Because he wants to make as many of his people happy as possible, he wants to place the statue somewhere so that the statue can be seen by the maximum number of people.

Sadly, the city's budget forced Jimmy to make a statue that can only be seen from a distance not bigger than R ($1 \leq R \leq 10^4$). Being precautious, when counting the number of people that see the statue, Jimmy will add the number of travellers of a road only if the statue can be seen from both of the houses connected by it.

Your task is to make Jimmy and his people happy by finding the perfect spot for his statue. It should be placed in a point with integer coordinates so that the number of people that see it is maximised. Print this number.

Input

The first line of the input will contain 3 integers: N, M, R as defined above.

The following N line will contain 2 integers each: x_i and y_i, the coordinates of house i.

The last M lines each contain 3 integers a ($1 \leq a \leq N$), b ($1 \leq b \leq N$), p ($1 \leq p \leq 10^9$) meaning there is a 2 way road between house a and b with p people travelling on it.

There can be multiple edges between the same houses.

Houses cannot have roads to themselves.

Output

The output should contain the maximum number of people that can see the statue.

Example

standard input	standard output
3 3 3 4 4 7 3 4 8 1 3 3 1 2 6 2 3 8	6

Note

If the statue is placed at $(6, 4)$ then 6 people will be able to admire Jimmy.

Problem K. Trains

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

AGM, the biggest music festival in the country is being held soon and people only have $1 \leq D \leq 10^5$ days to get there! There is only one way to do so: by train. There are $1 \leq T \leq 10^5$ tracks on which you may place trains with a single condition: at any time there may be only up to one train on each of the T tracks. A train takes exactly D days to get to the destination, regardless of the track. Formally, for a train to reach AGM in time, it should be on an open track continuously for the D days (not necessarily the same one, this will be clarified further in the statement).

But life is never that easy and because of the hot weather, some of the tracks become unusable and require some time until they can be used again. There will be $0 \leq q \leq 10^5$ updates of type (a, b) which means in day a, the track b changes its state, meaning it closes down if it was open before that, or opens up if it was closed. If a track is opened on day x, it can be used starting with that day. If a track is closed on day x, that is the last day when it can be used. If a track is closed on day x and opened back up on day $x + 1$, a train may not be on it continuously, meaning that if it were to continue going on that track, on day x it should move to another one and on day $x + 1$ it should move back.

The initial states of the tracks reflect what they are like on day 0, meaning they can be updated starting even from day 1.

A track may change its state twice on the same day. In that case, it is guaranteed that the first change opens the track and the second one closes it, making the track usable for that one particular day.

But you are not helpless: you have a powerful team that can move up to one train from a track to another during every single day. To perform a move, both tracks should be open during that day. This move is done instantaneously and a train that has been moved is still considered to have been continuously moving throughout that day.

Having taken all of the above in consideration, you need to say what is the maximum number of trains that can make it in time to AGM?

Input

The first line of the input contains 2 numbers: D and T.

The second line contains T numbers $a_i \in \{0, 1\}$ describing the initial state of each track. If a_i is 1, then track i is initially open.

The third line contains the number q.

Each of the next q lines contain 2 numbers $1 \leq a \leq D$ and $1 \leq b \leq T$, describing one of the changes.

Keep in mind each day you may move at most 1 train from an open track to another open track.

Output

The output should contain a single integer: the maximum number of trains that can make it in time.

Example

standard input	standard output
7 4 1 0 1 0 4 4 1 4 3 3 2 4 4	2

Note

At first the two trains are on track 1 and 3.

On day 3 we move one train from track 1 to track 2.

On day 4 we move one train from track 3 to track 4.

Problem L. Trivial

Input file:	`standard input`
Output file:	`standard output`
Time limit:	6 seconds
Memory limit:	256 megabytes

To save the country from a dangerous virus, Domestos has reached out to you for help - create a better recording system for the country's medical system.

The current medical system uses a multiset (hence, it allows duplication) of strings to record the different patients.

Since there are no records in the multiset, Domestos has implemented two basic operations to create and get information on the different strings in the multiset. Those are:

- 1 *index character*:

  Appends the *character* to the end of the string situated on the *index*-th position. If *index* is with one bigger than the cardinal of the multiset, a new string containing only the *character* is appended to the multiset. Note that this operation does not affect the order of the other strings within the multiset. For a better understanding, please refer to the example.

- 2 *index*:

  For the string situated on the *index*-th position, say how many strings from the multiset are in the same equivalence class with it.

  For two strings to be in the same equivalence class, they need to have the same distinct palindromic substrings (a substring is a consecutive subsequence). A string is palindromic when reading it backwards is the same as forwards.

  For example, given the following multiset: {"*aaaabxa*"*aaabxaa*"*abxaaaa*"*dcxaaaa*"}, applying the operation on the second index would generate the answer 1, whereas applying it on the third one would generate the answer 2.

Since Domestos is having limited capacity, the number of 1^{st} operations is $N \leq 4 * 10^5$ and moreover the total number of operations is capped to $M \leq 8 * 10^5$.

Domestos believes that this problem is trivial for you to solve and that might climb you on the top of the standings!

Input

The first line of the input will contain the number M, representing the total number of operations.

The following M lines will contain each operation.

Output

You should output M-N lines containing the output of each 2^{nd} operation.

Example

standard input	standard output
12 1 1 a 1 1 b 1 1 a 2 1 1 2 a 2 2 1 3 a 2 3 1 2 b 2 2 1 2 a 2 2	1 1 2 1 2

Note

After the first three operations the string "aba"is created and the 2^{nd} operation performed on it outputs 1 since there is just one string.

Then, an additional string is created, resulting into having two strings in the multiset: $\{"aba"a"\}$. As those two string are not in the same equivalence class, the output of the 2^{nd} operation will be 1.

As the third string is created, the multiset will contain: $\{"aba"a"a"\}$. The third 2^{nd} operation will output 2 since the last two string are in the same equivalence class.

The last two 1^{st} operations basically equalise the second to the first string and hence the output of the last 2^{nd} operation will be also 2.