

Problem A. Y-Shaped Knife

Input file: *standard input*
Output file: *standard output*
Time limit: 3 seconds
Memory limit: 512 mebibytes

Serious debates often arise around chocolate cakes with cherries. Everyone likes cherries, so the cake must be cut in such a way that everyone receives a piece with the same number of cherries.

Two people can easily divide a cake with one straight cut using a simple knife. Though less obvious, two straight cuts are always enough for four people. But what if you want to cut a cake for three people? We have a solution!

You are given a cake with cherries and a Y-shaped knife. Cut the cake into three parts with equal number of cherries or report that it is impossible.

Formally, you are given a set of points in a plane in general position. A Y-shaped knife is an apex point and three infinite straight rays going from the apex. The angle between any two rays equals 120° ($2\pi/3$). The knife divides the plane into three sections. You have to find the position of the apex and the rotation of the first ray such that each of the sections contains equal number of points.

Input

In the first line of input there is a single integer n , the number of points ($1 \leq n \leq 10^5$). Next n lines contain two integers each, x and y coordinates of the corresponding point. Coordinates do not exceed 10^6 by absolute value.

It is guaranteed that no two points coincide and no three points lie on the same line.

Output

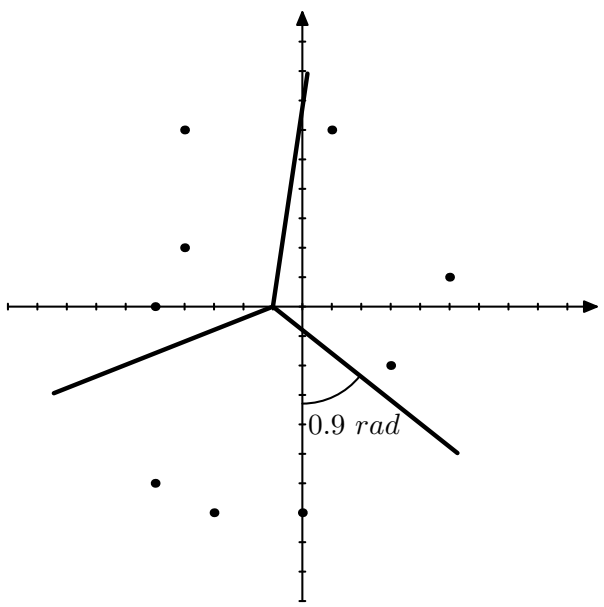
If it is impossible to make the desired cut, print “No” (without quotes).

If it is possible, print “Yes” (without quotes). Then print three real numbers x , y , and r ($|x|, |y| \leq 10^7$, $0 \leq r < 2\pi/3$), where x and y denote the coordinates of the apex and r denotes the counter-clockwise rotation of the knife in radians.

The initial position of the knife (that is, corresponding to zero rotation) is when one of the rays points down. The position when the ray points to the right corresponds to the rotation of $\pi/2$.

See “Notes” section for detailed explanation of the checker.

Example

standard input	standard output	illustration
<pre> 9 3 -2 -4 6 0 -7 -5 -6 5 1 1 6 -5 0 -3 -7 -4 2 </pre>	<pre> Yes -1 0 0.9 </pre>	

Note

All calculations are done in `long double`, an 80-bit floating-point type. Let the apex point be (x, y) with rotation r . For each input point (x_i, y_i) :

- if $|x - x_i| < 10^{-9}$ and $|y - y_i| < 10^{-9}$, the solution gets “Wrong Answer”;
- the angle of the direction from the apex point to the input point is calculated as $\text{atan2}(y_i - y, x_i - x) + \pi/2 - r$ modulo 2π , where $\text{atan2}(y, x)$ is the polar angle of point (x, y) ;
- if the absolute difference of the angle and 0, $2\pi/3$, or $4\pi/3$ is less than 10^{-9} , the solution gets “Wrong answer”;
- finally, the section which the point belongs to is calculated straightforwardly.

Problem B. Bad Doctor

Input file: *standard input*
Output file: *standard output*
Time limit: 3 seconds
Memory limit: 512 mebibytes

Alex got sick. He went to a clinic and visited n doctors. The i -th doctor said that starting with the day l_i and ending with the day r_i Alex must take k_i medicines: a_1, a_2, \dots, a_{k_i} , one pill a day of each. Medicines are numbered from 1 to m .

Of course, if several doctors tell Alex to take the same medicine at the same day, he will take only one pill of this medicine that day. At least, this is how people act in real life.

One pill of the medicine j costs c_j roubles. But Alex has a doubt: the rumors say that one of the doctors in the clinic is really bad. He doesn't know which doctor is bad, but he decided to ignore this doctor's prescription.

Your task is to find n numbers t_i : how much money Alex will spend on the pills if the i -th doctor is bad.

Input

The first line contains two integers n and m : the number of doctors and the number of medicines ($1 \leq n \leq 500\,000$, $1 \leq m \leq 500\,000$).

The second line contains m integers c_j : the cost of one pill of the j -th medicine ($1 \leq c_j \leq 1\,000\,000$).

Each of the next n lines describes doctors. The i -th of them starts with three integers l_i, r_i, k_i : the start and end days in the i -th doctor's prescription and the number of medicines he told Alex to take ($1 \leq l_i \leq r_i \leq 1\,000\,000$, $1 \leq k_i \leq m$). Then follow k_i distinct integers a_1, a_2, \dots, a_{k_i} , each from 1 to m : the medicines in the prescription.

The sum of all k_i in the input doesn't exceed $1\,000\,000$.

Output

Output n integers t_1, t_2, \dots, t_n : how much money Alex will spend on the pills if he ignores the i -th doctor's prescription.

Example

standard input	standard output
5 4 1000 100 10 1 3 4 2 2 3 4 8 3 1 2 4 6 7 2 3 4 8 9 2 1 4 2 6 3 1 2 3	8766 7564 8756 7765 6646

Problem C. Topological Ordering

Input file: *standard input*
Output file: *standard output*
Time limit: 4 seconds
Memory limit: 512 mebibytes

The topological ordering of a directed acyclic graph is a permutation of its vertices p_1, \dots, p_n such that for each arc, its source comes before its target in the permutation.

You are given a directed acyclic graph. For each pair of vertices (u, v) count the number of topological orderings such that vertex u comes before vertex v .

Input

The first line contains a single integer t , the number of test cases. Descriptions of t test cases follow.

In the first line of each test case there are two integers n and m : the number of vertices and arcs ($1 \leq n \leq 20$, $0 \leq m \leq n \cdot (n - 1)/2$).

Each of the next m lines contains two integers u_i and v_i , denoting the arc from vertex u_i to vertex v_i ($1 \leq u_i < v_i \leq n$).

There are at most 100 test cases in the input. In at most 5 test cases $n > 10$.

Output

For each test case, print n lines of n numbers each. The j -th number in the i -th line should equal the number of topological orderings where vertex j is before vertex i . In particular, it should equal 0 if $i = j$.

Example

standard input	standard output
2	0 0 0
3 2	2 0 1
1 2	2 1 0
1 3	0 0 3 1
4 2	6 0 5 3
1 2	3 1 0 0
3 4	5 3 6 0

Problem D. Bracket Euler Tour

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

You are given an undirected graph with n vertices and m edges. Each vertex has a bracket (either opening "(" or closing ")") associated with it.

You have to find an Euler tour in this graph such that its vertices (written in traversal order) form a correct bracket sequence.

A correct bracket sequence is a sequence of brackets that can be transformed into a correct arithmetic expression by inserting characters "1" and "+" between the original characters of the sequence. For example, bracket sequences "()", "(())" are correct (the resulting expressions are "(1)+(1)" and "((1+1)+1)"), while ")(" and "((" are not.

An Euler tour of an undirected graph is a cycle which visits every edge in this graph exactly once. It is allowed to visit the same vertex multiple times, though.

Input

The first line contains two integers n and m ($1 \leq n, m \leq 2 \cdot 10^5$), the number of vertices and edges respectively.

Each of the following m lines contains two integers v_i and u_i ($1 \leq v_i, u_i \leq n$), meaning that there is an undirected edge between vertices v_i and u_i . Note that self-loops and multiple edges are allowed.

The last line of the input contains a string of n round brackets, where the i -th bracket is associated with vertex i .

Output

If there is no Euler tour in the given graph that forms a correct bracket sequence, print "No" in the first line.

Otherwise, print "Yes" in the first line. In the second line, print a sequence of vertices that form an Euler tour and also a correct bracket sequence. If there are multiple solutions, print any of them.

Examples

standard input	standard output
2 2 1 2 1 2) (Yes 2 1
5 6 1 2 2 3 3 1 2 4 4 5 5 2 (())	Yes 1 2 4 5 2 3
1 1 1 1 (No

Problem E. Tree of Charge

Input file: *standard input*
Output file: *standard output*
Time limit: 4 seconds
Memory limit: 512 mebibytes

There is a rooted tree. Each vertex contains some non-negative amount of charge c_v . You have to process three kinds of queries:

- **Move the charge up:** all vertices simultaneously transfer all their charge to their direct parent. The charge from the root is not transferred anywhere. That is, if vertex v had children u_1, \dots, u_k , then its new charge becomes $c_{u_1} + \dots + c_{u_k}$ for a non-root vertex v , or $c_{u_1} + \dots + c_{u_k} + c_v$ if v is the root.
- **Move the charge down:** all vertices simultaneously transfer all their charge to their children in equal proportions. That is, if vertex v had children u_1, \dots, u_k , then the new charge of each u_i becomes c_v/k .

There is a virtual line tree of sufficient length attached to each leaf. That is, if the charge is moved down from the leaf and then moved up the same number of times, then the leaf retains its original charge.

- **Add charge to a vertex:** add a certain amount of charge to a certain vertex.

At the end, you should print the value of charge in each vertex.

Input

In the first line of input there is a single integer n , the number of vertices in the tree ($2 \leq n \leq 500\,000$). The next line contains n integers c_i , i -th of them denoting the initial charge of the tree ($0 \leq c_i < 10^9 + 7$). Each of the next $n - 1$ lines contains two integers u_i and v_i denoting the edge between vertices u_i and v_i ($1 \leq u_i, v_i \leq n$).

Next line contain a single integer q , the number of queries ($0 \leq q \leq 500\,000$). Then q lines follow with the description of queries. “Move up” query is denoted with a character “^”. “Move down” query is denoted with a character “v”. “Add charge” query is denoted with a character “+” followed by two integers v_i and x_i , meaning that you should add charge x_i to vertex v_i ($1 \leq v_i \leq n$, $0 \leq x_i < 10^9 + 7$).

It is guaranteed that the graph in the input is a tree.

Output

Print n numbers, i -th of them being the final charge of vertex i modulo $10^9 + 7$.

Formally, let the charge be p/q . Then you should print a unique number x , $0 \leq x < 10^9 + 7$, such that $p \equiv x \cdot q \pmod{10^9 + 7}$.

Examples

standard input	standard output
5 4 3 3 6 0 1 2 1 3 2 4 4 5 5 v + 1 7 ^ + 2 1 v	0 500000009 500000009 4 6
2 5 10 1 2 5 v v v ^ ^	0 5
4 0 1 0 0 1 2 1 3 1 4 2 ^ v	0 333333336 333333336 333333336

Problem F. Find a Tree

Input file: *standard input*
Output file: *standard output*
Time limit: 4 seconds
Memory limit: 512 mebibytes

Proper k -coloring of undirected graph $G(V, E)$ is a mapping $c : V \rightarrow \{1, 2, 3, \dots, k\}$ such that for each edge $(u, v) \in E$, we have $c_u \neq c_v$.

Undirected graph is k -colorable if a proper k -coloring exists for it.

Chromatic number of a graph is the smallest k such that the graph is k -colorable.

Tree is a simple acyclic undirected graph.

Alice has an undirected graph with chromatic number k , and Bob has a tree on k vertices. Bob wants to find k **different** vertices $p_1, p_2, p_3, \dots, p_k$ in Alice's graph such that for each edge (u, v) in Bob's tree, there exists an edge (p_u, p_v) in Alice's graph. Help him.

Input

The first line contains a single integer T ($1 \leq T \leq 10^6$), the number of test cases to solve. Description of T testcases follows. Each testcase is described as follows.

The first line contains three integers n, m , and k ($1 \leq n, k \leq 10^6, 0 \leq m \leq 10^6$), the number of vertices and edges of Alice's graph and its chromatic number, respectively.

Each of the next m lines contains a pair of integers u_i and v_i ($1 \leq u_i, v_i \leq n, u_i \neq v_i$) describing an edge of Alice's graph. It is guaranteed that there are no multiple edges and that Alice's graph has chromatic number exactly equal to k .

Each of the next $k - 1$ lines contains a pair of integers p_i and q_i ($1 \leq p_i, q_i \leq k, p_i \neq q_i$) describing an edge in Bob's tree. It is guaranteed that the given set of edges forms a tree.

It is guaranteed that the sum of n in all test cases, as well as the sum of m in all test cases, does not exceed 10^6 .

Output

For each testcase, output the answer in the following format.

If it is impossible to find the required k vertices in Alice's graph, print "No".

Otherwise, print "Yes" in the first line. In the second line, print k different integers p_i ($1 \leq p_i \leq n$): the numbers of vertices in Alice's graph corresponding to the respective vertices of Bob's tree. If there are several possible answers, print any one of them.

Example

standard input	standard output
3	Yes
6 6 3	3 2 1
1 2	Yes
2 3	4 1 2 3
3 1	Yes
1 4	5 4 3
2 5	
3 6	
1 2	
2 3	
4 6 4	
1 2	
1 3	
1 4	
2 3	
2 4	
3 4	
1 2	
1 3	
1 4	
5 4 3	
1 2	
3 4	
4 5	
5 3	
1 2	
2 3	



Problem G. One Root

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

For a fixed n and integers p and q such that $|p| \leq m$ and $|q| \leq m$, how many equations of the form

$$x^n + px + q = 0$$

have exactly one real root?

Input

The only line of input contains two integers n and m ($1 \leq n, m \leq 10^6$, $n \geq 2$).

Output

Print a single integer: the number of equations having exactly one real root.

Examples

standard input	standard output
2 4	5
3 5	96

Problem H. Delete the Points

Input file: *standard input*
Output file: *standard output*
Time limit: 3 seconds
Memory limit: 512 mebibytes

There are n points in the plane. The coordinates of the points are integer. The number n is always even. You can draw a square whose sides are parallel to the coordinate axes. Its vertices can have real coordinates. If there are exactly two points inside the square or on its borders, they are deleted. You have to find a way to delete all points or say that it is impossible.

Input

The first line contains a single integer n ($1 \leq n \leq 3000$), the number of points. Each of the following n lines contains two integers x_i and y_i ($0 \leq x_i, y_i \leq 10^9$), describing the coordinates of the i -th point. All points are distinct.

Output

If it is impossible to delete all points, print “No” in the first line. Otherwise, print “Yes” in the first line. In each of the next $n/2$ lines, print four real numbers: the coordinates of the opposite corners of the square. Squares should be printed in the order in which they should be drawn. If there are several possible answers, print any one of them. Real numbers should be output with no more than four digits after the decimal point. Print them in the following form: possibly unary minus, then any number of decimal digits, and then possibly a decimal point, followed by up to four decimal digits.

Examples

standard input	standard output
4 1 1 2 2 5 5 6 6	Yes 1.0 1.0 2.0 2.0 5.0 5.0 6.0 6.0
4 0 0 1 2 2 1 4 4	Yes 1.0 1.0 2.0 2.0 0.0 0.0 4.0 4.0

Problem I. Hard Times for Your Data

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

There is a storage cluster of n nodes. Each document is stored in exactly two replicas in some two nodes. Hard times are coming. Budget is shrinking. Bits of storage become very expensive, so some optimization is required. Maximum capacity was determined for each node; that is, how many replicas can be stored in it.

Since existing replicas cannot be moved, we've made a willful decision to drop some documents from the cluster to satisfy new requirements. We want to utilize the cluster completely, so after the cleanup each node should store exactly as many documents as its capacity allows.

We somehow know that it is possible to remove some documents such that each node is utilized perfectly. However, it turned out to be very hard to find exact documents to remove. Can you help us?

Input

In the first line of input, there are two integers n and m ($1 \leq n \leq 500$, $0 \leq m \leq n \cdot (n-1)/2$), the number of nodes and the number of sets of documents.

The next line contains n integers f_1, \dots, f_n ($0 \leq f_i \leq 10^9$), where f_i denotes the capacity of node i .

Each of the next m lines contains three integers u_i, v_i, c_i ($1 \leq u_i, v_i \leq n$, $u_i \neq v_i$, $1 \leq c_i \leq 10^4$). The description means that there are c_i documents with replicas in nodes u_i and v_i .

All pairs of nodes in the description of different document sets are distinct.

Output

First, print a number k : the number of pairs of nodes with common replicas after the removal.

Then print k lines, each containing three integers, in the same format as in the input: the pair of nodes and the number of documents stored in these nodes.

After the removal, each pair of nodes can not have more documents than it had before. Furthermore, the number of documents stored in the i -th node must equal f_i .

It is guaranteed that the answer exists. If there are multiple answers, print any one of them.

Example

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

Problem J. Program

Input file: *standard input*
Output file: *standard output*
Time limit: 2 seconds
Memory limit: 512 mebibytes

You are given a program operating with integer variable X , which is initially equal to 1. The program consists of n instructions of two types:

- “1 p ” ($1 \leq p \leq n$), assigns value p to variable X .
- “2 p q ” ($1 \leq p, q \leq n$, $p \neq q$), assigns value q to variable X only if the current value of X is p .

In one step, you can remove any single instruction from the program. You can't reorder instructions or add new instructions. What is the minimum number of steps required to get such a program that, after it runs, the variable X has value k ? You are asked to solve this problem for each k from 1 to n .

Input

The first line of input contains a single integer n ($2 \leq n \leq 10^6$), the number of instructions in program. The next n lines contains descriptions of instructions in the format described above.

Output

Output n integers, where i -th integer is the minimum number of steps required to make program set value i to variable X , or -1 if it is impossible.

Examples

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