Problem A. Suspicious Submissions

Input file:	`standard input`
Output file:	`standard output`
Time limit:	8 seconds
Memory limit:	1024 megabytes

The students have been naughty: they have been sharing codes between each other! You have the list of all their submissions, and you want to examine some pairs of codes manually for plagiarism.

Each submission is a string consisting of uppercase Latin characters. Let k be an integer. A pair of submissions $\{s, t\}$ is called *k-suspicious* if s can be obtained from t by replacing a (possibly empty) substring of length at most k with another (possibly empty) string of length at most k (and vice versa). Recall that a substring is a connected fragment of a string.

For a given integer k, how many pairs of submissions are k-suspicious?

Input

The first line of input contains the number of test cases Z ($1 \leq Z \leq 90\,000$). The descriptions of the test cases follow.

The first line of a test case contains integers n and k ($1 \leq n \leq 100\,000$, $1 \leq k \leq 1\,000\,000$) – the number of submissions and an integer parameter respectively. The following n lines describe the submissions. The i-th line contains a string s_i consisting of uppercase Latin characters – the i-th submission. It is guaranteed that submissions are pairwise different, i.e., no two strings are equal.

The total length of all submissions in a single test case does not exceed $1\,000\,000$. The sum of n in all test cases does not exceed $200\,000$. The total length of all submissions in all test cases does not exceed $2\,000\,000$.

Output

For each test case, print a single integer denoting the number of k-suspicious pairs of submissions.

Example

standard input	standard output
1 4 2 SUFFIXTREE SUSIXTREE SUFFIXTREAP SUFFIXARRAY	2

Note

There are only two pairs of 2-suspicious strings in the sample (the replaced substrings are marked bold): (SU**FF**IXTREE, SU**S**IXTREE) and (SUFFIXTRE**E**, SUFFIXTRE**AP**).

This page is intentionally left blank.

Problem B. ICFC World Finals

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

The annual International Collegiate Foosball Contest World Finals are just around the corner, and it just so happens that you're in charge of marketing. Among other things, your job is to design a banner promoting the event. To avoid unnecessary details, you have decided that the banner will only contain the competition ladder, and nothing else.

This year's ladder is a full binary tree consisting of n nodes. Each internal node has exactly two children. Each node corresponds to a match between two players; the winner of which advances to the match represented by the node's parent. The only exception is the root, representing the grand final, from which no player advances.

The banner should be rectangular, with integer side lengths. The ladder should be drawn on the banner adhering to the following rules:

- Nodes in the ladder are points on the banner with integer coordinates (relative to the banner's top left corner) and can be drawn on the banner's edges.
- Edges in the ladder are straight line segments parallel to the sides of the banner connecting nodes on the banner and cannot intersect one another.
- For each internal node, one of it's children must be drawn to the right of it, and one below it (not necessarily directly)*.
- The bounding boxes of disjoint subtrees must be disjoint †.

Input

The first line of input contains the number of test cases Z ($1 \leq Z \leq 200\,000$). The descriptions of the test cases follow.

Each test case is given in two lines, the first of which contains the number of nodes in the ladder n ($3 \leq n \leq 200\,000$). The next line contains $n-1$ integers, the i-th of which denotes the parent of the $(i+1)$-th node in the ladder. The root of the ladder is the node 1.

The sum of n over all test cases does not exceed 200 000.

Output

For every test case, print a single line containing the area of the smallest required banner.

Example

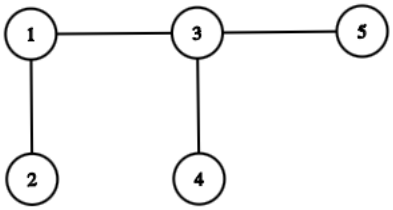
standard input	standard output
2 5 1 1 3 3 9 1 1 2 2 3 3 6 6	2 6

*One of the children must have a greater x coordinate than the parent and the other a smaller y coordinate.

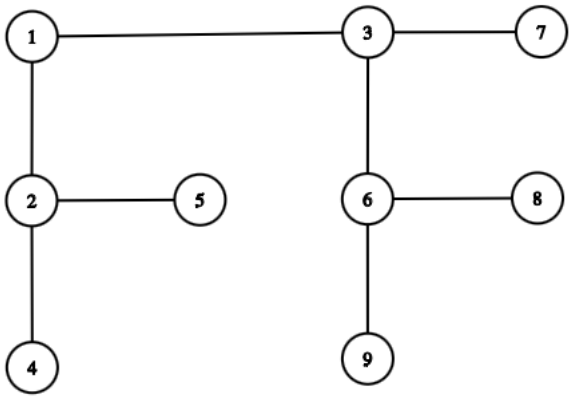
†Let $Box(T)$ be the minimal rectangle with sides parallel to sides of the banner that contains nodes in the subtree T. If two subtrees A and B are disjoint, $Box(A)$ and $Box(B)$ must also be disjoint (including their perimeter).

Note

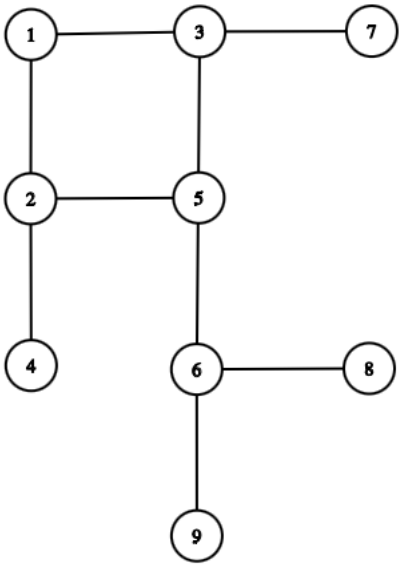
An optimal placement of nodes in the first sample.



An optimal placement of nodes in the second sample.



An **illegal** placement of nodes in the second sample. Produces the optimal area; however, the bounding boxes of subtrees rooted at nodes 3 and 2 intersect.



Problem C. Diamonds and the Genie

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

Jack is a treasure hunter who loves to explore ancient ruins. One day, he stumbled upon a hidden chamber that contained a large grid of squares. Each square had a number of diamonds embedded in it. Jack noticed a sign that said: "To claim the treasure, you must find a path from the top left corner to the bottom right corner of the grid, and collect as many diamonds as possible along the way. But beware, once you enter the grid, you can only move right or down."

Jack was intrigued by the challenge and decided to give it a try. He was about to enter the grid when he spotted a golden lamp lying near the entrance. He picked up the lamp and rubbed it, hoping for some extra luck. To his surprise, a genie popped out of the lamp and greeted him. "Hello, Master. I am the genie of the grid. I will grant you one wish, but it must be related to the grid. What do you wish for?"

Jack thought for a moment and realized that he could use the genie's help to improve his path. He asked the genie to allow him to move to the left once, so that he could collect more diamonds. The genie nodded and said: "Your wish is granted. You can now move to the left once, but only once. Use it wisely."

Jack thanked the genie and entered the grid. What is the maximum number of diamonds he can collect?

Input

The first line of input contains the number of test cases Z ($1 \leq Z \leq 500\ 000$). The descriptions of the test cases follow.

The first line of a test case contains two numbers n, m ($1 \leq n, m \leq 3\,000$) – the dimensions of the grid.

The following n lines contain m integers each, the j-th number in the i-th line $d_{i,j}$ is the number of diamonds in the j-th cell of the i-th row ($0 \leq d_{i,j} \leq 10^6$).

The sum $\sum n \cdot m$ over all test cases will not exceed $9\,000\,000$.

Output

For each test case, print in a single line the maximum number of diamonds that Jack can collect. Note that Jack does not have to move left, and that he cannot collect diamonds from one square twice.

Example

standard input	standard output
2 3 2 1 1 1 1 1 1 2 2 1 2 0 4	6 7

Note

In the first test case, one of the optimal paths is RDLDR. In the second test case, both RD and RLRD paths are optimal.

This page is intentionally left blank.

Problem D. Money in the Hat

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

Lucy's company is organizing a Christmas Lottery. There are N employees participating, and during the event, N notes will be put in a hat, one for each employee. The i-th note will be worth i dollars. Then the employees will, in a random order, draw a random note from the hat.

Or at least they should. However, Lucy does not play by the rules, so in her turn she plans to peek into the hat and take the note with the highest value.

Now she is wondering what the expected value of her note is, assuming all her colleagues do not cheat. Unfortunately, she cannot solve this problem the usual corporate way (i.e. using Excel), so she asked you for help.

Input

The first line of input contains the number of test cases Z ($1 \leq Z \leq 10^6$). The descriptions of the test cases follow.

The first and only line of a test case contains a number of employees N ($1 \leq N \leq 10^6$).

Output

It can be shown that the expected value (in dollars) of Lucy's note can be expressed as an irreducible fraction $\frac{P}{Q}$ where P, Q are integers and Q is coprime with 998244353.

For each test case, print in a single line the number $P \cdot Q^{-1} \mod 998244353$.

Example

standard input	standard output
2 2 10	249561090 390067951

Note

In the first test case, the expected value of the note is $\frac{7}{4}$.

This page is intentionally left blank.

Problem E. Gambling

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Alice and Bob are compulsive gamblers. While waiting in the casino lobby, they started flipping coins for money (obviously), until one of them goes broke (obviously).

In the beginning, Alice has a dollars and Bob has b dollars. They flip a coin and if it lands on tails, Alice gets $\min(a, b)$ dollars from Bob; otherwise, Bob gets $\min(a, b)$ dollars from Alice. If one of them loses all the money, they stop and go to the closest ATM. Otherwise, they continue the same process.

For example, if Alice has 9$ and Bob has 2$ the process might go as follows:

Coin flip	Alice	Bob	Outcome
1st	9$	2$	heads
2nd	7$	4$	heads
3rd	3$	8$	tails
4th	6$	5$	tails

After the 4th coin flip, Bob has 0$ and they stop.

Some people in the lobby started making side bets on how many coin flips it will take before one of them goes broke. You also want to participate in the betting, but first, you have to compute the expected number of times they will flip a coin in the process, so you can make informed decisions.

Input

The first line of input contains the number of test cases Z ($1 \leq Z \leq 100\,000$). The descriptions of the test cases follow.

The first and only line of a test case contains two integers a, b ($1 \leq a, b \leq 10^{18}$) – the number of dollars that Alice and Bob have, respectively.

Output

We can prove that the sought expected value is always a finite rational number. Moreover, when the value is represented as an irreducible fraction P/Q, we can show that there exists a unique integer R such that $R \cdot Q \equiv P \mod 998244353$ and $0 \leq R < 998244353$.

For each test case print in a single line this integer R.

Example

standard input	standard output
2 1 1 3 9	1 499122178

Note

In the first test case, either Alice or Bob will lose all the money after the first coin flip.

In the second test case, the expected number of coin flips is $1\frac{1}{2} \equiv 499122178 \mod 998244353$.

This page is intentionally left blank.

Problem F. Red-Blue MST

Input file:	`standard input`
Output file:	`standard output`
Time limit:	8 seconds
Memory limit:	1024 megabytes

You are given a connected undirected graph on n vertices. Each edge has a positive integer weight and is painted in one of two colors – red or blue. For each $k \in \{0, \ldots, n-1\}$, find the minimum possible weight of a spanning tree that contains exactly k red edges or decide that no such spanning tree exists.

Input

The first line of input contains the number of test cases Z ($1 \leq Z \leq 10\,000$). The descriptions of the test cases follow.

The first line of each test case contains integers n, m ($2 \leq n \leq 100\,000$, $n - 1 \leq m \leq 200\,000$) – the number of vertices and the number of edges.

Next m lines describe graph edges. Each line contains three integers u_i, v_i, w_i followed by a character c_i ($1 \leq u_i, v_i \leq n$, $u_i \neq v_i$, $1 \leq w_i \leq 10^6$, $c_i \in \{R, B\}$) – the two endpoints of an edge, its weight, and its color. The color red is denoted by R and the color blue is denoted by B.

It is guaranteed that the graph is connected. The sum of n over all test cases does not exceed 500 000. The sum of m over all test cases does not exceed 1 000 000.

Output

For each test case, print in a single line n integers $a_0, \ldots, a_{n-1}$. a_k should be the minimum possible weight of a spanning tree that contains exactly k red edges, or -1 if no such spanning tree exists.

Example

standard input	standard output
2 4 6 1 2 1 R 2 3 5 B 3 4 1 R 4 1 5 B 1 3 1 R 2 4 3 B 3 3 1 2 5 R 1 3 7 R 2 3 9 B	13 9 5 3 -1 14 12

This page is intentionally left blank.

Problem G. Road Trip

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	1024 megabytes

There are n cities in Treeland, connected by bidirectional roads. It is possible to travel from any city to any other city, but only in a single way. In other words, the road network of Treeland forms a tree.

You are planning a road trip around Treeland. Your car has a tank capacity of c liters. For each road, you know how much fuel is required to pass through it. The car can be refueled in any city, but there are no gas stations between the cities. The fuel is cheap in Treeland, so you only want to minimize the number of refueling stops to have more time admiring Treeland's scenic routes.

You are not sure which route to take yet. Answer q queries of the form: given two cities u and v, how many times do you need to refuel when traveling from the city u to the city v? Assume that you start in the city u with a full tank.

Input

The first line of input contains the number of test cases Z ($1 \leq Z \leq 10\,000$). The descriptions of the test cases follow.

The first line of a test case contains three integers n, q, and c ($2 \leq n \leq 200\,000$, $1 \leq q \leq 200\,000$, $1 \leq c \leq 10^9$) – the number of cities, the number of queries, and the tank capacity respectively.

The next $n - 1$ lines describe the road network. Each line contains integers a_i, b_i and w_i ($a_i \neq b_i, 1 \leq a_i, b_i \leq n$, $1 \leq w_i \leq c$) – the endpoint cities and the required fuel amount for the i-th road.

The final q lines describe the queries. Each line contains two distinct integers u_j and v_j ($u_j \neq v_j, 1 \leq u_j, v_j \leq n$) – the queried pair of cities.

The sum of n over all test cases does not exceed 400 000. The sum of q over all test cases does not exceed 400 000.

Output

For each query, print a single line containing the minimum possible number of refuelings.

Example

standard input	standard output
1 7 6 5 1 2 3 2 3 3 2 4 1 1 5 2 5 6 1 6 7 4 3 6 3 7 4 6 4 7 3 4 7 1	2 2 1 2 0 1

Note

For the first query in the sample, one of the optimal routes is as follows:

- start in city 3 with a full tank with 5 liters of fuel;
- drive to city 2 using 3 liters of fuel, bringing the tank's fuel down to 2 liters;
- **refuel** the car, bringing the tank up to 5 liters of fuel again;
- drive to city 1 using 3 liters and bringing the tank's fuel down to 2 liters;
- drive to city 5 using our 2 remaining liters of fuel;
- **refuel** the car for the last time;
- drive to city 6 using only one liter out of the five remaining in the tank.

Problem H. Decent Path Around Bajtów

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

The road network of Bajtów is a graph consisting of n intersections connected by m roads. Each road can be either unidirectional or bidirectional. To simplify the road policies and ease the traffic, the city administrators want to make all roads unidirectional, picking an orientation for every bidirectional road.

As Bajtów is a popular tourist destination, there was a petition to make some roads into a directed cycle so that the "Fantastic Tour Around Bajtów" could be offered as a new tourist attraction. But unfortunately, the road network turned out to have a curious property: no matter how bidirectional roads were directed, the resulting graph would never contain a cycle!

To make up for this, Bajtów authorities decided to pick an orientation such that the longest possible path in the road network would be as long as possible. This way, at least the "Decent Path Around Bajtów" could be arranged. Your goal is to calculate the maximum length of a path over all possible orientations.

Input

The first line of input contains the number of test cases Z ($1 \le Z \le 10\,000$). The descriptions of the test cases follow.

The first line of a test case contains two integers n and m ($2 \le n \le 500\,000$, $1 \le m \le 500\,000$) – the number of intersections and the number of roads respectively. The next m lines describe the road network. Each line contains a road description in one of two following formats:

- `u -> v`: a unidirectional road from u to v;
- `u -- v`: a bidirectional road between u and v.

In both cases, the values u and v are distinct integers satisfying $1 \le u, v \le n$. Each unordered pair of vertices $\{u, v\}$ appears at most once. The road network is guaranteed to have no orientations that would contain a directed cycle.

The sum of n over all test cases does not exceed $1\,000\,000$. The sum of m over all test cases does not exceed $1\,000\,000$.

Output

For each test case, print a single line containing the maximum possible number of edges in a path over all possible orientations.

Example

standard input	standard output
1 6 7 1 -> 2 1 -> 4 2 -> 3 2 -- 5 4 -> 5 5 -> 6 3 -> 6	5

This page is intentionally left blank.

Problem I. Random Remainders

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

You are given a sequence of n **random** positive integers: $a_1, a_2, \ldots, a_n$. Your task is to compute the following sum:

$$\sum_{i=1}^{n}\sum_{j=1}^{n}(a_i \bmod a_j)^2$$

Input

The first line of input contains the number of test cases: Z ($1 \leq Z \leq 1000$). The descriptions of the test cases follow.

The first line of a test case contains the length of the sequence n ($1 \leq n \leq 200\,000$). The second line contains the sequence – n positive integers separated by spaces. Each of them was pseudo-randomly generated, uniformly from range $[1, 10^{12}]$.

The sum of n over all test cases does not exceed 400 000.

Output

For each test case print the required sum modulo 998244353.

Example

standard input	standard output
1 2 3 5	13

Note

$(3 \bmod 3)^2 + (5 \bmod 3)^2 + (3 \bmod 5)^2 + (5 \bmod 5)^2 = 0 + 4 + 9 + 0 = 13$.

This page is intentionally left blank.

Problem J. Sumotonic Sequences

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	1024 megabytes

A sequence of integers is called **monotonic** if it is either nondecreasing or nonincreasing. A sequence is **sumotonic** if it can be expressed as a sum of any number of monotonic sequences, all having only non-negative values.

You are given a sequence $A = (a_1, \ldots, a_n)$ and a list of k operations. Each operation adds an arithmetic sequence to some segment of A. More precisely, for a given interval $[p, q]$ and integers s and d, the elements $a_p, a_{p+1}, \ldots, a_q$ are changed to $a_p + s, a_{p+1} + s + d, \ldots, a_q + s + (q - p) \cdot d$.

Determine if A is sumotonic, for the initial sequence and after each operation. Note that the operations are persistent (they have a permanent effect on the sequence A).

Input

The first line of input contains the number of test cases Z ($1 \leq Z \leq 10\,000$). The descriptions of the test cases follow.

The first line of each test case contains integers n and k ($2 \leq n \leq 200\,000$, $1 \leq k \leq 200\,000$) – the length of the sequence and the number of operations respectively.

The second line contains initial elements $a_1, \ldots, a_n$ of the sequence A ($1 \leq a_i \leq 2 \cdot 10^9$ for $i = 1, 2, \ldots, n$).

The last k lines contain operations, each described by four integers p, q, s, d ($1 \leq p \leq q \leq n$, $1 \leq s, d \leq 10\,000$), meaning that to the interval $[p, q]$ we add an arithmetic sequence with starting element s and difference d.

The sum of $n + k$ over all test cases does not exceed $400\,000$.

Output

For each test case, output $k + 1$ lines. The first one should contain the text "YES" if the initial sequence is sumotonic, "NO" otherwise. Each of the remaining k lines should similarly describe the state of the sequence after every operation.

Example

standard input	standard output
1 5 2 5 8 5 2 4 5 5 6 6 2 4 1 3	NO NO YES

This page is intentionally left blank.

Problem K. Bitter

Input file:	`standard input`
Output file:	`standard output`
Time limit:	10 seconds
Memory limit:	1024 megabytes

After successfully defrauding his investors, Melon Usk gathered enough money to take over Bitter Inc, and he now has big plans for the company. For a start, Melon wants to reduce Bitter's headcount, and for that, he needs to understand which pairs of employees are *similar*.

The management at Bitter Inc follows a rooted tree structure: employees are numbered from 1 to n, and each employee (apart from Melon i.e. employee 1) has exactly one direct manager, and is either directly or indirectly managed by Mr Usk himself. Employees directly managed by a given employee are *ordered*, and each employee v has an assigned project p_v. Given an employee v, we define their *team* as all employees either directly or indirectly reporting to v (including v themselves). We will say a project p is *owned* by v if all employees working on p are in v's team.

Initially, Melon considered employees x and y similar if their teams could be superimposed onto each other, exactly matching in both management structure and projects p_v that the matched employees work on. However, the resulting number of pairs of similar employees wasn't high enough (considering the number of layoffs Usk is aiming for), so he decided to relax the notion of similarity by allowing *conversions* between projects owned by the compared employees.

Formally, for an employee v, denote their team as T_v, and the projects owned by v as O_v. Consider two employees x and y and a bijection f between O_x and O_y; imagine taking all $v \in T_x$ such that $p_v \in O_x$ and setting p_v to $f(p_v) \in O_y$. If there exists f such that after the above transformation x's team becomes identical[‡] to y's team, then Melon considers x and y similar. Note that if x's and y's teams are identical without any changes and $x \neq y$, then O_x and O_y must be empty, and indeed the employees are similar according to Melon's definition.

If an employee is found to be similar to many other employees, then they may be in for an unpleasant surprise...

You are an intern, recently hired by Melon himself. Write a program that for each employee v computes the number of other employees that are similar to v.

Input

The first line of input contains the number of test cases Z ($1 \leq Z \leq 100\,000$). The descriptions of the test cases follow.

The first line of a test case contains the number of employees n ($1 \leq n \leq 400\,000$).

The second line of a test case contains n numbers p_v ($1 \leq p_v \leq n$) – the project that the respective employee is working on.

The next n lines describe the reporting structure: the i-th line starts with an integer d_i ($0 \leq d_i < n$), denoting the number of i's direct reports; it is then followed by d_i numbers $v_{1,j}$ ($1 \leq v_{1,j} \leq n$), denoting the IDs of the reports. Note that the order of reports does matter.

The total number of employees in all test cases does not exceed 800 000.

Output

For each test case, output a single line containing n numbers c_v – the number of employees (excluding v) that are similar to v.

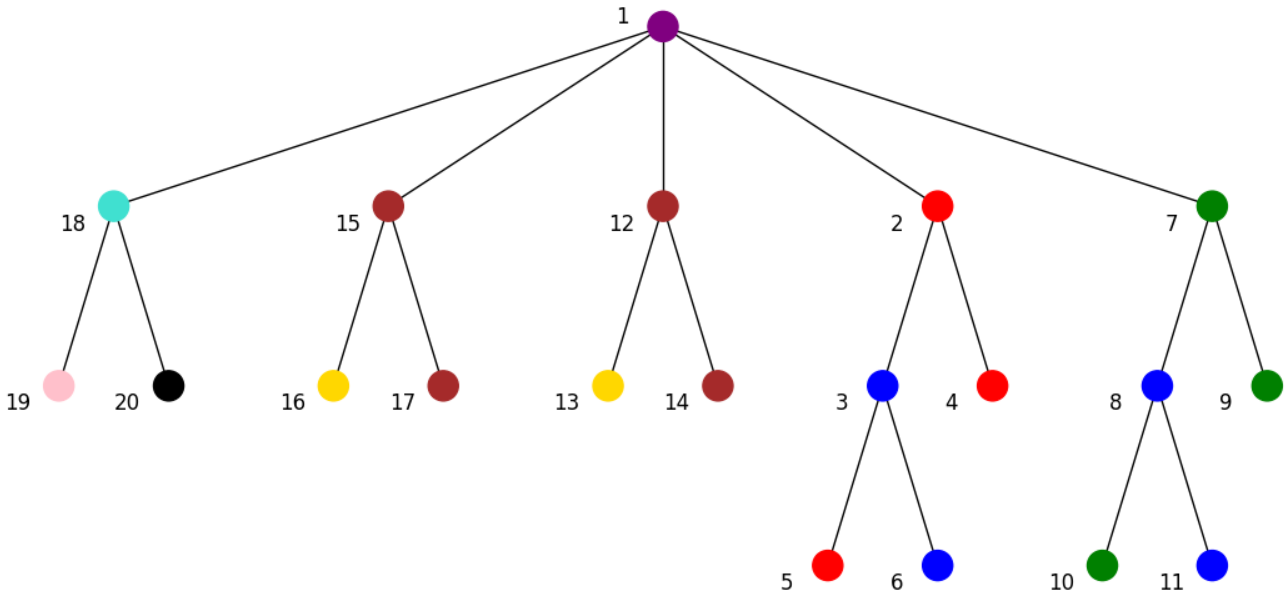
[‡]Formally, two teams are identical if there is a bijection g between them, such that for any node a and an integer k, the k-th child of a is mapped by g to the k-th child of $g(a)$ (note that the order of children matters).

Example

standard input	standard output
1 20 3 1 2 1 1 2 5 2 5 5 2 9 8 9 9 8 9 7 6 4 5 18 15 12 2 7 2 3 4 2 5 6 0 0 0 2 8 9 2 10 11 0 0 0 2 13 14 0 0 2 16 17 0 0 2 19 20 0 0	0 1 0 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1

Note

The sample test case is depicted below, where different projects are shown as different colors.



The similar pairs of employees are: $(2, 7)$, $(4, 5)$, $(6, 11)$, $(9, 10)$, $(12, 15)$, $(13, 16)$, $(14, 17)$, and $(19, 20)$.

Note that 3 and 8 are not similar, as they differ in one project (red vs green), which are not owned by 3 and 8. However, these projects are owned by 2 and 7, and indeed defining a bijection f to map from red to green shows 2 and 7 are similar. 18 is not similar to any of the employees, which can be inferred from the fact that $|O_{18}| = 3$, and the other employees with the same team structure own at most 1 project (and thus a bijection cannot exist). Employees x and y with no reports are similar to each other either

if x owns p_x and y owns p_y (which is the case for 19 and 20) or if $p_x = p_y$ (which is the case for several other pairs, e.g. 4 and 5).

This page is intentionally left blank.

Problem L. Empty Triangles

Input file:	`standard input`
Output file:	`standard output`
Time limit:	15 seconds
Memory limit:	1024 megabytes

You are given n points on the plane. For some triangles, each spanning three of these points, determine if they are empty (that is, they do not contain any of the other given points).

Input

The first line of input contains the number of test cases Z ($1 \leq Z \leq 400$). The descriptions of the test cases follow.

The first line of a test case contains two integers: the number of points n ($3 \leq n \leq 5\,000$) and the number of queries q ($1 \leq q \leq 4\,000\,000$).

The next n lines contain the coordinates of points with two integers x_i, y_i ($-10^9 \leq x_i, y_i \leq 10^9$). The points are pairwise distinct. No three points are collinear.

The final q lines describe the queries, each containing three distinct numbers a_j, b_j, c_j ($1 \leq a_j, b_j, c_j \leq n$) – the vertices of the j-th triangle.

The total number of points over all test cases does not exceed 5 000. The total number of queries over all test cases does not exceed 4 000 000.

Output

For every query, print "YES" if the triangle is empty, "NO" if it contains other points.

Example

standard input	standard output
1 9 4 0 0 4 -1 8 0 9 4 8 8 4 9 0 8 -1 4 3 1 1 3 5 3 5 7 2 4 6 2 6 8	NO YES YES NO

This page is intentionally left blank.