The 13th Shandong Provincial Collegiate Programming Contest

Contest Session

June 4, 2023

Problem List

A	Orders
B	Building Company
C	Trie
D	Fast and Fat
E	Math Problem
F	Colorful Segments
G	Matching
H	Be Careful 2
I	Three Dice
J	Not Another Path Query Problem
K	Difficult Constructive Problem
L	Puzzle: Sashigane
M	Computational Geometry

This problem set should contain 13 (thirteen) problems on 19 (nineteen) numbered pages. Please inform a runner immediately if something is missing from your problem set.

Prepared by SUA Programming Contest Problem Setter Team.
https://sua.ac/

Hosted by

Problem Set Prepared by

Problem A. Orders

A factory receives n orders at the beginning of day 1. The i-th order can be described as two integers a_i and b_i, indicating that at the end of day a_i, the factory needs to deliver b_i products to the customer.

Given that the factory can produce k products each day, and at the beginning of day 1 the factory has no product in stock, can the factory complete all orders?

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 100$) indicating the number of test cases. For each test case:

The first line contains two integers n and k ($1 \le n \le 100$, $1 \le k \le 10^9$) indicating the number of orders and the number of products the factory can produce each day.

For the following n lines, the i-th line contains two integers a_i and b_i ($1 \le a_i, b_i \le 10^9$) indicating that the i-th order require the factory to deliver b_i products at the end of day a_i.

Output

For each test case output one line. If the factory can complete all orders output `Yes`, otherwise output `No`.

Example

standard input	standard output
2 4 5 6 12 1 3 6 15 8 1 3 100 3 200 4 300 6 100	Yes No

Note

For the first sample test case, the factory can produce 5 products each day.

- At the end of day 1, there are 5 products in stock so the factory can complete the 2-nd order. After delivery, there are 2 products left in stock.
- At the end of day 6, the factory produces 25 more products. There are 27 products in stock so the factory can complete the 1-st and the 3-rd order. After delivery, there are 0 products left in stock.
- At the end of day 8, the factory produces 10 more products. There are 10 products in stock so the factory can complete the 4-th order. After delivery, there are 9 products left in stock.

For the second sample test case, the factory can produce 100 products each day.

- At the end of day 3, there are 300 products in stock and the factory can complete the 1-st order. After delivery, there are 100 products left in stock.
- At the end of day 4, the factory produces 100 more products. There are only 200 products in stock so the factory cannot complete the 2-nd order.

Problem B. Building Company

You're the boss of a building company. At the beginning, there are g types of employees in the company, and different types of employees have different occupations. For the i-th type of employees, their occupation can be numbered as t_i and there are u_i employees in total.

There are n building projects in the market waiting to be undertaken. To undertake the i-th project, your company must meet m_i requirements. The j-th requirement requires that your company has at least $b_{i,j}$ employees whose occupation is $a_{i,j}$. After undertaking the project, your company will become more famous and will attract k_i types of employees to join your company. The occupation of the j-th type of employees is $c_{i,j}$ and there are $d_{i,j}$ employees in total.

You can undertake any number of projects in any order. Each project can be undertaken at most once. Calculate the maximum number of projects you can undertake.

Note that employees are not consumables. After undertaking a project the number of employees in your company won't decrease.

Input

There is only one test case in each test file.

The first line of the input first contains an integer g ($1 \le g \le 10^5$) indicating the number of types of employees in the company at the beginning. Then g pairs of integers $t_1, u_1, t_2, u_2, \cdots t_g, u_g$ follow ($1 \le t_i, u_i \le 10^9$), where t_i and u_i indicate that there are u_i employees whose occupation is t_i. It's guaranteed that for all $1 \le i < j \le g$ we have $t_i \ne t_j$.

The second line contains an integer n ($1 \le n \le 10^5$) indicating the number of projects waiting to be undertaken.

For the following $2n$ lines, each two lines describe a project.

The $(2i - 1)$-th line first contains an integer m_i ($0 \le m_i \le 10^5$) indicating the number of requirements to undertake the i-th project. Then m_i pairs of integers $a_{i,1}, b_{i,1}, a_{i,2}, b_{i,2}, \cdots, a_{i,m_i}, b_{i,m_i}$ follow ($1 \le a_{i,j}, b_{i,j} \le 10^9$) where $a_{i,j}$ and $b_{i,j}$ indicate that the company is required to have at least $b_{i,j}$ employees whose occupation is $a_{i,j}$. It's guaranteed that for all $1 \le x < y \le m_i$ we have $a_{i,x} \ne a_{i,y}$.

The $2i$-th line first contains an integer k_i ($0 \le k_i \le 10^5$) indicating the number of types of employees to join the company after undertaking the i-th project. Then k_i pairs of integers $c_{i,1}, d_{i,1}, c_{i,2}, d_{i,2}, \cdots, c_{i,k_i}, d_{i,k_i}$ follow ($1 \le c_{i,j}, d_{i,j} \le 10^9$) where $c_{i,j}$ and $d_{i,j}$ indicate that there are $d_{i,j}$ employees whose occupation is $c_{i,j}$ joining the company. It's guaranteed that for all $1 \le x < y \le k_i$ we have $c_{i,x} \ne c_{i,y}$.

It's guaranteed that neither the sum of m_i nor the sum of k_i will exceed 10^5.

Output

Output one line containing one integer indicating the maximum number of projects you can undertake.

Example

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

Note

We explain the sample test case as follows. Let (t, u) indicate u employees whose occupation is t.

First, undertake the 5-th project with no requirements. After undertaking the project, there are 2 employees, whose occupation is 3, joining the company. The company now have these employees: $\{(1,2),(2,1),(3,2)\}$.

Next, undertake the 1-st project. After undertaking the project, no employee joins the company. The company now still have these employees: $\{(1,2),(2,1),(3,2)\}$.

Next, undertake the 2-nd project. After undertaking the project, there are 2 employees, whose occupation is 3, and 1 employee, whose occupation is 2, joining the company. The company now have these employees: $\{(1,2),(2,2),(3,4)\}$.

Next, undertake the 4-th project. After undertaking the project, there are 3 employees, whose occupation is 1, joining the company. The company now have these employees: $\{(1,5),(2,2),(3,4)\}$.

As the company does not have 3 employees whose occupation is 2, we cannot undertake the 3-rd project.

Problem C. Trie

Recall the definition of a trie:

- A trie of size n is a rooted tree with n vertices and $(n-1)$ edges, where each edge is marked with a character.
- Each vertex in a trie represents a string. Let $s(x)$ be the string vertex x represents.
- The root of the trie represents an empty string. Let vertex u be the parent of vertex v, and let c be the character marked on the edge connecting vertex u and v, we have $s(v) = s(u) + c$. Here $+$ indicates string concatenation, not the normal addition operation.
- The string each vertex represents is distinct.

We now present you a rooted tree with $(n+1)$ vertices. The vertices are numbered $0, 1, \cdots, n$ and vertex 0 is the root. There are m key vertices in the tree where vertex k_i is the i-th key vertex. It's guaranteed that all leaves are key vertices.

Please mark a lower-cased English letter on each edge so that the rooted tree changes into a trie of size $(n+1)$. Let's consider the sequence $A = \{s(k_1), s(k_2), \cdots, s(k_m)\}$ consisting of all strings represented by the key vertices. Let $B = \{w_1, w_2, \cdots, w_m\}$ be the string sequence formed by sorting all strings in sequence A from smallest to largest in lexicographic order. Please find a way to mark the edges so that sequence B is minimized.

We say a string $P = p_1p_2\cdots p_x$ of length x is lexicographically smaller than a string $Q = q_1q_2\cdots q_y$ of length y, if

- $x < y$ and for all $1 \le i \le x$ we have $p_i = q_i$, or
- there exists an integer $1 \le t \le \min(x, y)$ such that for all $1 \le i < t$ we have $p_i = q_i$, and $p_t < q_t$.

We say a string sequence $F = \{f_1, f_2, \cdots, f_m\}$ of length m is smaller than a string sequence $G = \{g_1, g_2, \cdots, g_m\}$ of length m, if there exists an integer $1 \le t \le m$ such that for all $1 \le i < t$ we have $f_i = g_i$, and f_t is lexicographically smaller than g_t.

Input

There are multiple test cases. The first line of th input contains an integer T indicating the number of test cases. For each test case:

The first line contains two integers n and m ($1 \le m \le n \le 2 \times 10^5$) indicating the number of vertices other than the root and the number of key vertices.

The second line contains n integers $a_1, a_2, \cdots, a_n$ ($0 \le a_i < i$) where a_i is the parent of vertex i. It's guaranteed that each vertex has at most 26 children.

The third line contains m integers $k_1, k_2, \cdots, k_m$ ($1 \le k_i \le n$) where k_i is the i-th key vertex. It's guaranteed that all leaves are key vertices, and all key vertices are distinct.

It's guaranteed that the sum of n of all test cases will not exceed 2×10^5.

Output

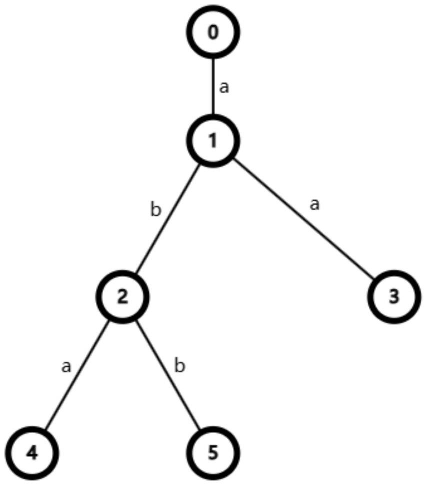
For each test case output one line containing one answer string $c_1c_2\cdots c_n$ consisting of lower-cased English letters, where c_i is the letter marked on the edge between a_i and i. If there are multiple answers strings so that sequence B is minimized, output the answer string with the smallest lexicographic order.

Example

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

Note

The answer of the first sample test case is shown as follows.



The string represented by vertex 1 is "a". The string represented by vertex 4 is "aba". The string represented by vertex 3 is "aa". The string represented by vertex 5 is "abb". So $B = \{$"a", "aa", "aba", "abb"$\}$.

Problem D. Fast and Fat

You're participating in a team competition of trail running. There are n members in your team where v_i is the speed of the i-th member and w_i is his/her weight.

The competition allows each team member to move alone or carry another team member on their back. When member i carries member j, if member i's weight is greater than or equal to member j's weight, member i's speed remains unchanged at v_i. However, if member i's weight is less than member j's weight, member i's speed will decrease by the difference of their weight and becomes $v_i - (w_j - w_i)$. If member i's speed will become negative, then member i is not able to carry member j. Each member can only carry at most one other member. If a member is being carried, he/she cannot carry another member at the same time.

For all members not being carried, the speed of the slowest member is the speed of the whole team. Find the maximum possible speed of the whole team.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains an integer n ($1 \le n \le 10^5$) indicating the number of team members.

For the following n lines, the i-th line contains two integers v_i and w_i ($1 \le v_i, w_i \le 10^9$) indicating the speed and weight of the i-th member.

It's guaranteed that the sum of n of all test cases will not exceed 10^5.

Output

For each test case output one line containing one integer indicating the maximum speed of the whole team.

Example

standard input	standard output
2 5 10 5 1 102 10 100 7 4 9 50 2 1 100 10 1	8 1

Note

The optimal strategy for the sample test case is shown as follows:

- Let member 1 carry member 4. As $w_1 > w_4$, member 1's speed remains unchanged, which is still 10.
- Let member 3 carry member 2. As $w_3 < w_2$, member 3's speed will decrease by $w_2 - w_3 = 2$ and becomes $10 - 2 = 8$.
- Member 5 shall move alone. His/Her speed is 9.

So the answer is 8.

Problem E. Math Problem

Given two positive integers n and k, you can perform the following two types of operations any number of times (including zero times):

- Choose an integer x which satisfies $0 \le x < k$, and change n into $k \cdot n + x$. It will cost you a coins to perform this operation once. The integer x you choose each time can be different.
- Change n into $\lfloor \frac{n}{k} \rfloor$. It will cost you b coins to perform this operation once. Note that $\lfloor \frac{n}{k} \rfloor$ is the largest integer which is less than or equal to $\frac{n}{k}$.

Given a positive integer m, calculate the minimum number of coins needed to change n into a multiple of m. Please note that 0 is a multiple of any positive integer.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 10^5$) indicating the number of test cases. For each test case:

The first line contains five integers n, k, m, a, b ($1 \le n \le 10^{18}$, $1 \le k, m, a, b \le 10^9$).

Output

For each test case output one line containing one integer, indicating the minimum number of coins needed to change n into a multiple of m. If this goal cannot be achieved, output -1 instead.

Example

standard input	standard output
4 101 4 207 3 5 8 3 16 100 1 114 514 19 19 810 1 1 3 1 1	11 2 0 -1

Note

For the first sample test case, initially $n = 101$. The optimal steps are shown as follows:

- Firstly, perform the second type of operation once. Change n into $\lfloor \frac{n}{4} \rfloor = 25$. This step costs 5 coins.
- Then, perform the first type of operation once. Choose $x = 3$ and change n into $4 \cdot n + 3 = 103$. This step costs 3 coins.
- Then, perform the first type of operation once. Choose $x = 2$ and change n into $4 \cdot n + 2 = 414$. This step costs 3 coins.
- As $414 = 2 \times 207$, n is a multiple of m. The total cost is $5 + 3 + 3 = 11$ coins.

For the second sample test case, perform the second type of operation twice will change n into 0. The total cost is $1 + 1 = 2$ coins.

For the third sample test case, as $n = 114 = 6 \times 19$ is already a multiple of m, no operation is needed. The total cost is 0 coins.

Problem F. Colorful Segments

Consider n segments on the number axis, where the left endpoint of the i-th segment is l_i and the right endpoint is r_i. Each segment has a color where the color of the i-th segment is c_i ($0 \le c_i \le 1$). There are two types of colors, where $c_i = 0$ indicates a red segment and $c_i = 1$ indicates a blue segment.

You need to choose some segments (you can also choose no segments at all). If any two chosen segments overlap, then they must have the same color.

Calculate the number of ways to choose segments.

We say segment i overlaps with segment j, if there exists a real number x satisfying both $l_i \le x \le r_i$ and $l_j \le x \le r_j$.

We say two ways of choosing segments are different, if there exists an integer $1 \le k \le n$ such that the k-th segment is chosen in one of the ways and is not chosen in the other.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains an integer n ($1 \le n \le 10^5$) indicating the number of segments.

For the following n lines, the i-th line contains three integers l_i, r_i and c_i ($1 \le l_i \le r_i \le 10^9$, $0 \le c_i \le 1$) indicating the left and right endpoints and the color of the i-th segment.

It's guaranteed that the sum of n of all test cases will not exceed 5×10^5.

Output

For each test case output one line containing one integer indicating the number of ways to choose segments. As the answer may be large, please output the answer modulo 998244353.

Example

standard input	standard output
2 3 1 5 0 3 6 1 4 7 0 3 1 5 0 7 9 1 3 6 0	5 8

Note

For the first sample test case, you cannot choose the 1-st and the 2-nd segment, or the 2-nd and the 3-rd segment at the same time, because they overlap with each other and have different colors.

For the second sample test case, as the 2-nd segment does not overlap with the 1-st and the 3-rd segment, you can choose them arbitrary.

Problem G. Matching

Given an integer sequence $a_1, a_2, \cdots, a_n$ of length n, we construct an undirected graph G from the sequence. More precisely, for all $1 \le i < j \le n$, if $i - j = a_i - a_j$, there will be an undirected edge in G connecting vertices i and j. The weight of the edge is $(a_i + a_j)$.

Find a matching of G so that the sum of weight of all edges in the matching is maximized, and output this maximized sum.

Recall that a matching of an undirected graph means that we choose some edges from the graph such that any two edges have no common vertices. Specifically, not choosing any edge is also a matching.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains an integer n ($2 \le n \le 10^5$) indicating the length of the sequence.

The second line contains n integers $a_1, a_2, \cdots, a_n$ ($-10^9 \le a_i \le 10^9$) indicating the sequence.

It's guaranteed that the sum of n of all test cases will not exceed 5×10^5.

Output

For each test case output one line containing one integer indicating the maximum sum of weight of all edges in a matching.

Example

standard input	standard output
3 9 3 -5 5 6 7 -1 9 1 2 3 -5 -4 -3 3 1 10 100	30 0 0

Note

For the first sample test case, the optimal choice is to choose the three edges connecting vertex 3 and 5, vertex 4 and 7, and finally vertex 8 and 9. The sum of weight is $(5 + 7) + (6 + 9) + (1 + 2) = 30$.

For the second sample test case, as all edges have negative weights, the optimal matching should not choose any edge. The answer is 0.

For the third sample test case, as there is no edge in the graph, the answer is 0.

Problem H. Be Careful 2

Little Cyan Fish has an $n \times m$ rectangle located in a plane. The top-right corner of the rectangle is at (n, m), while the bottom-left corner is at $(0, 0)$. There are k banned points inside the rectangle. The i-th banned point is located at (x_i, y_i).

Little Cyan Fish would like to draw a square inside the rectangle. However, he dislikes all the banned points, so there cannot be any banned points inside his square. Formally, Little Cyan Fish can draw a square with the bottom-left corner at (x, y) and a side length d if and only if:

- Both x and y are non-negative integers while d is a positive integer.
- $0 \le x < x + d \le n$.
- $0 \le y < y + d \le m$.
- For each $1 \le i \le k$, the following condition must **NOT** be met:
  - $x < x_i < x + d$ and $y < y_i < y + d$.

Please calculate the sum of the areas of all the squares that Little Cyan Fish can possibly draw. Since the answer could be huge, you need to output it modulo 998244353.

Input

The is only one test case in each test file.

The first line of the input contains three integers n, m and k ($2 \le n, m \le 10^9$, $1 \le k \le 5 \times 10^3$) indicating the size of the rectangle and the number of banned points.

For the following k lines, the i-th line contains two integers x_i and y_i ($0 < x_i < n$, $0 < y_i < m$) indicating the position of the i-th banned point. It is guaranteed that all the banned points are distinct.

Output

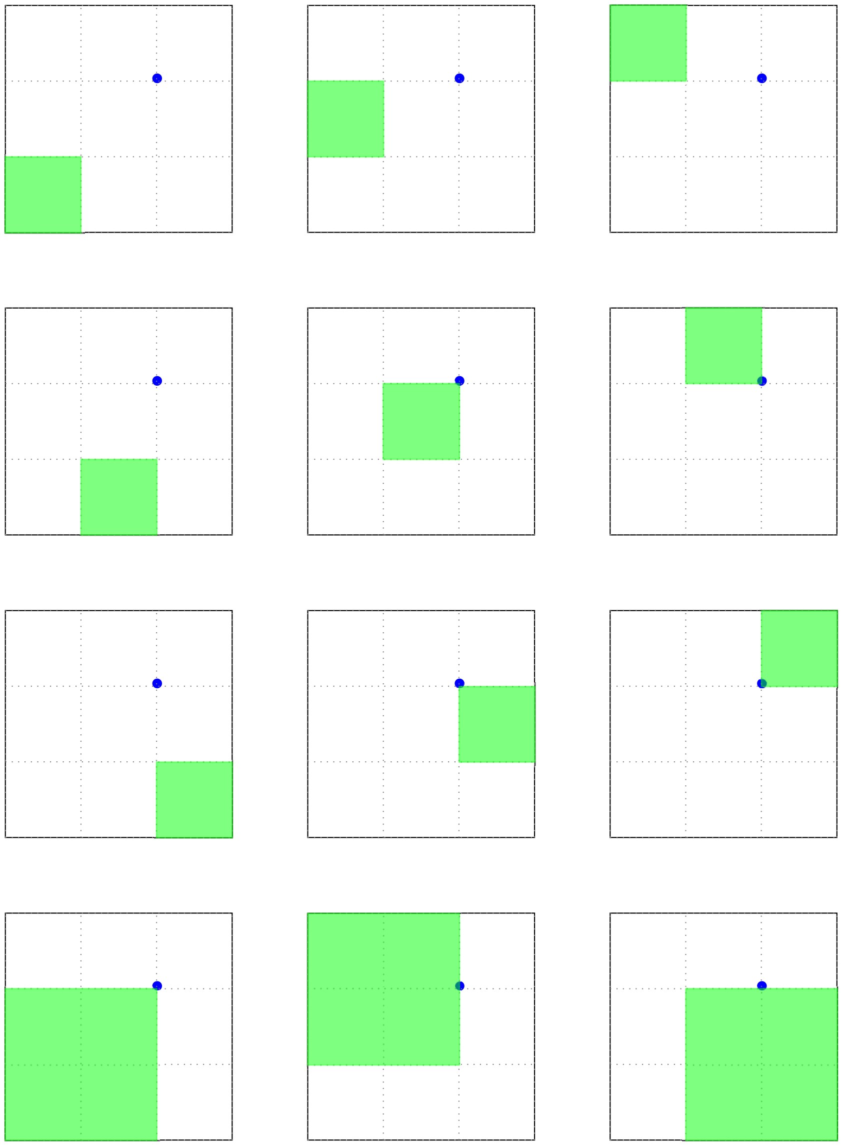
Output one line containing one integer indicating the answer modulo 998244353.

Examples

standard input	standard output
3 3 1 2 2	21
5 5 2 2 1 2 4	126

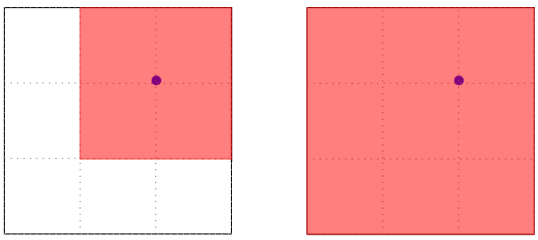
Note

For the first sample test case, Little Cyan Fish has 12 ways to draw a square, illustrated as follows.



There are 9 squares of side length 1 and 3 squares of side length 2. So the answer is $9 \times 1^2 + 3 \times 2^2 = 21$.

Note that the following plans are invalid since there's a banned point in the square.



Problem I. Three Dice

Dice are small, throwable objects with marked sides capable of landing in multiple positions. They are typically used to generate random values, especially in the context of tabletop games.

The most common dice are small cubes, with faces numbered from 1 to 6. Number n ($1 \le n \le 6$) is usually represented by a pattern of n round dots, known as pips. Moreover, the pips on the 1 and 4 faces are colored red (⚀,⚃), while those on the 2, 3, 5 and 6 faces are black (⚁,⚂,⚄,⚅).

Little Cyan Fish has three dice. One day, he threw them onto a table, and then observed the uppermost faces. He claimed that the total number of the red pips facing up was exactly A, and the total number of the black pips facing up was exactly B.

However, you find his claim doubtful. You want to verify whether it is possible to throw three dice such that the total number of red pips facing up is A, and the total number of black pips facing up is B.

Input

There is only one test case in each test file.

The first line of the input contains two integers A and B ($0 \le A, B \le 100$), indicating the total number of red pips facing up and the number of black pips facing up.

Output

Output one line. If it is possible for Little Cyan Fish to throw three dice such that the total number of red pips facing up is A, and the total number of black pips facing up is B output `Yes`. Otherwise output `No`.

Examples

standard input	standard output
4 5	Yes
3 0	Yes
1 2	No

Note

In the first example, one possible solution is ⚃,⚁,⚂.

In the second example, one possible solution is ⚀,⚀,⚀.

Problem J. Not Another Path Query Problem

What age is it that you are still solving traditional path query problems?

After reading the paper *Distributed Exact Shortest Paths in Sublinear Time*, you have learned how to solve the distributed single-source shortest paths problem in $\mathcal{O}(D^{1/3} \cdot (n \log n)^{2/3})$. To give your knowledge good practice, Little Cyan Fish prepared the following practice task for you.

Little Cyan Fish has a graph consisting of n vertices and m bidirectional edges. The vertices are numbered from 1 to n. The i-th edge connects vertex u_i to vertex v_i and is assigned a weight w_i.

For any path in the graph between two vertices u and v, let's define the value of the path as the bitwise AND of the weights of all the edges in the path.

As a fan of high-value paths, Little Cyan Fish has set a constant threshold V. Little Cyan Fish loves a path if and only if its value is at least V.

Little Cyan Fish will now ask you q queries, where the i-th query can be represented as a pair of integers (u_i, v_i). For each query, your task is to determine if there exists a path from vertex u_i to vertex v_i that Little Cyan Fish would love it.

Input

There is only one test case in each test file.

The first line contains four integers n, m, q and V ($1 \le n \le 10^5$, $0 \le m \le 5 \times 10^5$, $1 \le q \le 5 \times 10^5$, $0 \le V < 2^{60}$) indicating the number of vertices, the number of edges, the number of queries and the constant threshold.

For the following m lines, the i-th line contains three integers u_i, v_i and w_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$, $0 \le w_i < 2^{60}$), indicating a bidirectional edge between vertex u_i and vertex v_i with the weight w_i. There might be multiple edges connecting the same pair of vertices.

For the following q lines, the i-th line contains two integers u_i and v_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$), indicating a query.

Output

For each query output one line. If there exists a path whose value is at least V between vertex u_i and v_i output `Yes`, otherwise output `No`.

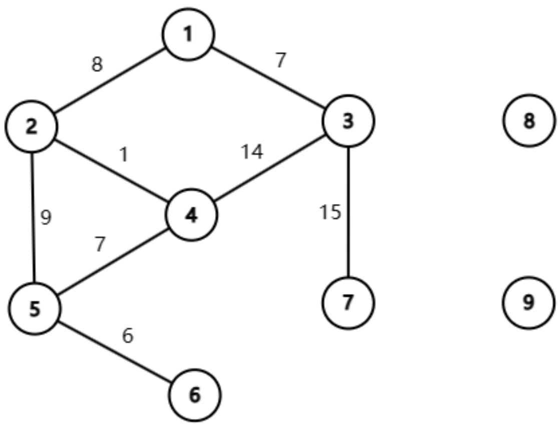
Examples

standard input	standard output
9 8 4 5 1 2 8 1 3 7 2 4 1 3 4 14 2 5 9 4 5 7 5 6 6 3 7 15 1 6 2 7 7 6 1 8	Yes No Yes No
3 4 1 4 1 2 3 1 2 5 2 3 2 2 3 6 1 3	Yes

Note

We now use & to represent the bitwise AND operation.

The first sample test case is shown as follows.



- For the first query, a valid path is $1 \to 3 \to 4 \to 5 \to 6$, whose value is $7 \,\&\, 14 \,\&\, 7 \,\&\, 6 = 6 \geq 5$.
- For the third query, a valid path is $7 \to 3 \to 4 \to 5 \to 6$, whose value is $15 \,\&\, 14 \,\&\, 7 \,\&\, 6 = 6 \geq 5$.
- For the fourth query, as there is no path between vertex 1 and 8, the answer is `No`.

For the only query of the second sample test case, we can consider the path consisting of the 2-nd and the 4-th edge. Its value is $5 \,\&\, 6 = 4 \geq 4$.

Problem K. Difficult Constructive Problem

Given a string $s_1 s_2 \cdots s_n$ of length n where $s_i \in \{‘0’, ‘1’, ‘?’\}$ and an integer k, please fill out all the ‘?’ with ‘0’ or ‘1’ such that the number of indices i satisfying $1 \le i < n$ and $s_i \ne s_{i+1}$ equals to k. Different ‘?’ can be replaced with different characters.

To make this problem even more difficult, we ask you to find the answer with the smallest possible lexicographic order if it exists.

Recall that a string $a_1 a_2 \cdots a_n$ of length n is lexicographically smaller than another string $b_1 b_2 \cdots b_n$ of length n if there exists an integer k $(1 \le k \le n)$ such that $a_i = b_i$ for all $1 \le i < k$ and $a_k < b_k$.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains two integers n and k $(1 \le n \le 10^5, 0 \le k < n)$ indicating the length of the string and the required number of indices satisfying the condition.

The second line contains a string $s_1 s_2, \cdots s_n$ $(s_i \in \{‘0’, ‘1’, ‘?’\})$.

It's guaranteed that the sum of n of all test cases will not exceed 10^6.

Output

For each test case output one line. If the answer exists output the lexicographically smallest one (you need to output the whole given string after filling out all the ‘?’ and make this string the lexicographically smallest); Otherwise output `Impossible`.

Example

standard input	standard output
5 9 6 1?010??01 9 5 1?010??01 9 6 100101101 9 5 100101101 9 3 ????????1	100100101 Impossible 100101101 Impossible 000000101

Problem L. Puzzle: Sashigane

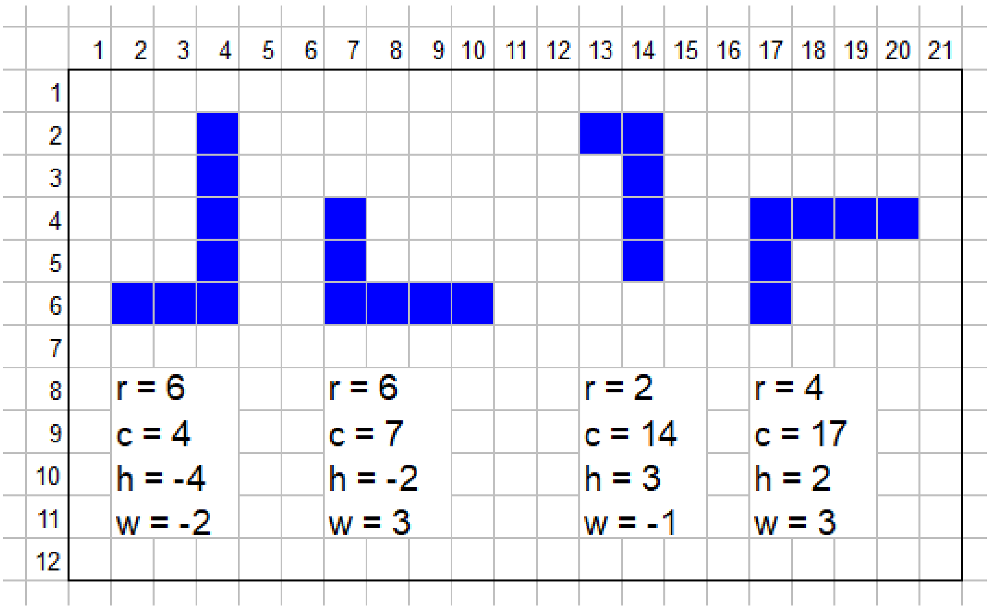
Given a grid with n rows and n columns, there is exactly one black cell in the grid and all other cells are white. Let (i, j) be the cell on the i-th row and the j-th column, this black cell is located at (b_i, b_j).

You need to cover all white cells with some L-shapes, so that each white cell is covered by exactly one L-shape and the only black cell is not covered by any L-shape. L-shapes must not exceed the boundary of the grid.

More formally, an L-shape in the grid is uniquely determined by four integers (r, c, h, w), where (r, c) determines the turning point of the L-shape, and h and w determine the direction and lengths of the two arms of the L-shape. The four integers must satisfy $1 \le r, c \le n$, $1 \le r + h \le n$, $1 \le c + w \le n$, $h \ne 0$, $w \ne 0$.

- If $h < 0$, then all cells (i, c) satisfying $r + h \le i \le r$ belong to this L-shape; Otherwise if $h > 0$, all cells (i, c) satisfying $r \le i \le r + h$ belong to this L-shape.
- If $w < 0$, then all cells (r, j) satisfying $c + w \le j \le c$ belong to this L-shape; Otherwise if $w > 0$, all cells (r, j) satisfying $c \le j \le c + w$ belong to this L-shape.

The following image illustrates some L-shapes.



Input

There is only one test case in each test file.

The first line contains three integers n, b_i and b_j ($1 \le n \le 10^3$, $1 \le b_i, b_j \le n$) indicating the size of the grid and the position of the black cell.

Output

If a valid answer exists first output `Yes` in the first line, then in the second line output an integer k ($0 \le k \le \frac{n^2-1}{3}$) indicating the number of L-shapes to cover white cells. Then output k lines where the i-th line contains four integers r_i, c_i, h_i, w_i separated by a space indicating that the i-th L-shape is uniquely determined by (r_i, c_i, h_i, w_i). If there are multiple valid answers you can print any of them.

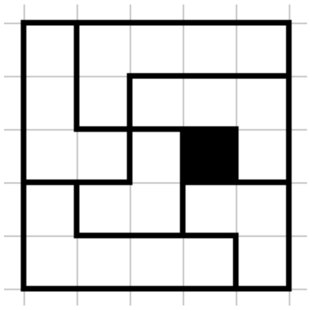
If there is no valid answer, just output `No` in one line.

Examples

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

Note

We illustrate the first sample test case as follows.



Problem M. Computational Geometry

Given a convex polygon P with n vertices, you need to choose three vertices of P, denoted as a, b and c in counter-clockwise order. There must be exactly k edges from b to c in counter-clockwise order (that is to say, a is not an endpoint of these k edges).

Consider cutting through P with segment ab and ac. Let Q be the polygon consisting of ab, ac and the k edges between b and c. It's easy to see that this polygon has $(k+2)$ edges.

Find the maximum possible area of Q.

Note that ab and ac can overlap with edges of P.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains two integers n and k ($3 \le n \le 10^5$, $1 \le k \le n-2$) indicating the number of vertices of the convex polygon P and the number of edges from b to c in counter-clockwise order.

For the following n lines, the i-th line contains two integers x_i and y_i ($-10^9 \le x_i, y_i \le 10^9$) indicating the x and y coordinate of the i-th vertex of the convex polygon P. Vertices are given in counter-clockwise order. It's guaranteed that the area of the convex polygon is positive, and there are no two vertices with the same coordinate. It's possible that three vertices lie on the same line.

It's guaranteed that the sum of n of all test cases will not exceed 10^5.

Output

For each test case output one line containing one real number indicating the maximum possible area of Q. Your answer will be considered correct if its relative or absolute error is less than 10^{-9}.

Example

standard input	standard output
3 3 1 0 0 1 0 0 1 8 3 1 2 3 1 5 1 7 3 8 6 5 8 3 7 1 5 7 2 3 6 1 1 3 1 7 1 8 1 5 6 4 6	0.500000000000 26.500000000000 20.000000000000

Note

For the first sample test case, Q is the whole triangle. Its area is 0.5.

The second and third sample test case are shown below.