Problem B. Bitwise Xor

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	1024 mebibytes

Zhong Ziqian got an integer array $a_1, a_2, \ldots, a_n$ and an integer x as birthday presents.

Every day after that, he tried to find a non-empty subsequence of this array $1 \le b_1 < b_2 < \ldots < b_k \le n$, such that for all pairs (i, j) where $1 \le i < j \le k$, the inequality $a_{b_i} \oplus a_{b_j} \ge x$ held. Here, $\oplus$ is the bitwise exclusive-or operation.

Of course, every day he must find a different subsequence.

How many days can he do this without repeating himself? As this number may be very large, output it modulo 998 244 353.

Input

The first line of the input contains two integers n and x ($1 \le n \le 300\,000$, $0 \le x \le 2^{60} - 1$). Here, n is the size of the array.

The next line contains n integers $a_1, a_2, \ldots, a_n$: the array itself ($0 \le a_i \le 2^{60} - 1$).

Output

Output one integer: the number of subsequences of Ziqian's array such that bitwise xor of every pair of elements is at least x, modulo 998 244 353.

Examples

standard input	standard output
3 0 0 1 2	7
3 2 0 1 2	5
3 3 0 1 2	4
7 4 11 5 5 8 3 1 3	35

Note

In the first example, all $2^3 - 1$ non-empty subsequences are suitable.

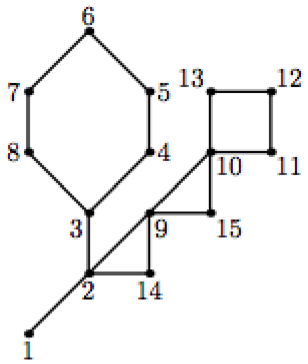
in the second example, two non-empty subsequences are not suitable, it is $b = [1, 2]$ and $b = [1, 2, 3]$, that is because $a_1 \oplus a_2 = 0 \oplus 1 = 1$ which is smaller than 2.

In the third example, $b = [1], b = [2], b = [3], b = [2, 3]$ are suitable.

Problem C. Counting Cactus

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

NEERC featured a number of problems about cactuses: **connected** undirected graphs in which every edge belongs to at most one simple cycle. Intuitively, a *cactus* is a generalization of a tree where some cycles are allowed. An example of a cactus from NEERC 2007 problem is given on the picture below.



Dreamoon has an undirected graph. Now he is wondering, how many subgraphs (subsets of edges) of his graph are cactuses? Can you help him find this value modulo 998 244 353?

Input

The first line contains two integers n and m: the number of vertices and edges in the Dreamoon's graph ($1 \le n \le 13$, $0 \le m \le \frac{n(n-1)}{2}$).

The next m lines describe edges in the graph. The i-th of these lines contains two integers a_i and b_i ($1 \le a_i, b_i \le n$, $a_i \ne b_i$), denoting an edge between vertices a_i and b_i. It is guaranteed that there are no multiple edges.

Output

Output one integer: the number of cactus subgraphs of Dreamoon's graph, modulo 998 244 353.

Examples

standard input	standard output
3 3 1 2 2 3 3 1	4
5 0	0
8 9 1 5 1 8 2 4 2 8 3 4 3 6 4 7 5 7 6 8	35

Note

Sorry, Dreamoon.

Problem E. Easy Win

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1.5 seconds
Memory limit:	256 mebibytes

V–o_o–V and LHiC are playing a game.

At first, gritukan shows them an undirected graph with n vertices where each edge has a pile of stones on it.

After that, LHiC chooses some non-empty subset of edges of this graph that forms edge-disjoint edge-simple cycles (in other words, each connected component should have an Euler circuit). If he can't (in other words, if the graph is acyclic), he loses immediately.

Otherwise, LHiC and V–o_o–V play a game of Nim with the piles on the chosen edges. V–o_o–V moves first. In a single move, a player may remove an arbitrary positive number of stones from a single pile. The player who cannot make a move loses.

Let's call a graph **good** if LHiC can't choose a non-empty subset of edge-disjoint cycles on which he will win Nim.

Gritukan asks q queries, can you help him?

There is a set of possible edges which can be picked by gritukan to form a **good** graph. Initially, this set is empty. In query i, first, an edge i connecting vertices u_i and v_i, with a pile of a_i stones on it and weight w_i, is added to the set of possible edges. After that, you should find the largest sum of weights of a **good** graph that gritukan can form using a subset of edges $1, 2, \ldots, i$.

Input

The first line contains two integers n and q: the number of vertices in the graph and the number of queries ($2 \le n \le 64$, $1 \le q \le 200\,000$).

Each of the next q lines contains four integers u_i, v_i, a_i, w_i, describing the edge added during i-th query ($1 \le u_i, v_i \le n$, $u_i \ne v_i$, $0 \le a_i < 2^{60}$, $1 \le w_i \le 10^9$).

Output

Print q lines. For the i-th query, you should print the largest sum of weights of a **good** graph that you can form using a subset of edges $1, 2, \ldots, i$.

Examples

standard input	standard output
3 3 1 2 0 1 2 3 0 1 3 1 0 2	1 2 3
6 6 1 2 1 1 2 3 1 2 3 4 1 3 4 1 1 4 5 6 1 2 6 5 1 1	1 3 6 9 11 11
5 5 1 2 0 1 2 3 1 1 3 4 2 3 4 5 4 9 5 1 7 29	1 2 5 14 42
5 1 3 5 35 35	35

Problem G. Grammarly

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

CauchySheep has a string s.

He looked at all its different non-empty substrings and added a directed edge from a to b if $|b| + 1 = |a|$ and b is a substring of a.

You need to calculate the number of simple paths starting from s in this graph, modulo 998 244 353.

Input

The first line of the input contains a string s consisting of lowercase Latin letters: the string CauchySheep has ($1 \le |s| \le 300\,000$).

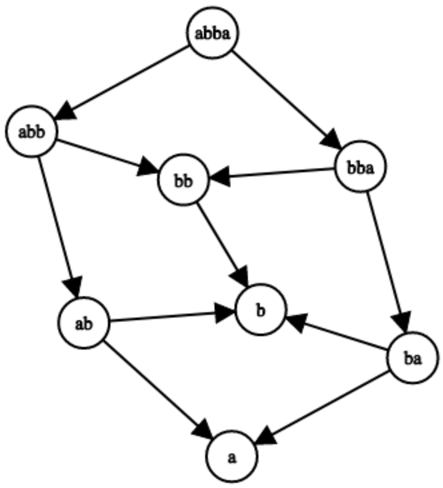
Output

Output one integer: the number of simple paths starting from s in CauchySheep's graph, modulo 998 244 353.

Examples

standard input	standard output
abba	13
benbeipo	255
iqiiiiiiqq	300
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	35

Note



Problem I. Interactive Vertex

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

This is an interactive problem.

Endagorion has a tree on n vertices, and he also showed it to you. He chooses one vertex u as a special vertex, but now he won't tell you anything about it!

Instead, you can ask him questions. For each question, you should choose a vertex x, an integer k, and k vertices $v_1, v_2, \ldots, v_k$, and he will tell you whether it is true that $\min(\text{dist}(u, v_i)) \geq \text{dist}(u, x)$. Here, $\text{dist}(p, q)$ is the number of edges in the simple path between vertices p and q in the tree.

You should guess the special vertex using at most $4\lceil \log_2 n \rceil$ queries.

Endagorion is very honest, so he wouldn't change the vertex between your queries (in other words, the interactor is not adaptive).

As the constraints are large, and flush is an expensive operation, make sure that you are not flushing too often. You may do it only once after each query.

Interaction Protocol

The interaction starts with a line containing one integer n: the number of vertices in Endagorion's tree ($2 \leq n \leq 200\,000$).

Each of the next $n-1$ lines contains two integers u and v ($1 \leq u, v \leq n$), denoting an edge between u and v. It is guaranteed that the given edges form a tree.

After that, you can make queries.

To make a query, print one line containing "`?` k" ($1 \leq k \leq n$), an integer x ($1 \leq x \leq n$) and then k distinct integers $v_1, v_2 \ldots, v_k$ ($1 \leq v_i \leq n$). Separate consecutive integers by single spaces. Then `flush` the output.

After each query, read one line with a single integer $ans \in \{0, 1\}$. If $\min(\text{dist}(u, v_i)) \geq \text{dist}(u, x)$, then ans will be equal to 1. Otherwise, ans will be equal to 0.

When you find the special vertex u ($1 \leq u \leq n$), print one line containing "`!` u", and then flush the output and terminate.

Your solution will get `Wrong Answer` or `Time Limit Exceeded` if you make more than $4\lceil \log_2 n \rceil$ queries.

Your solution will get `Idleness Limit Exceeded` if you don't print anything or forget to flush the output.

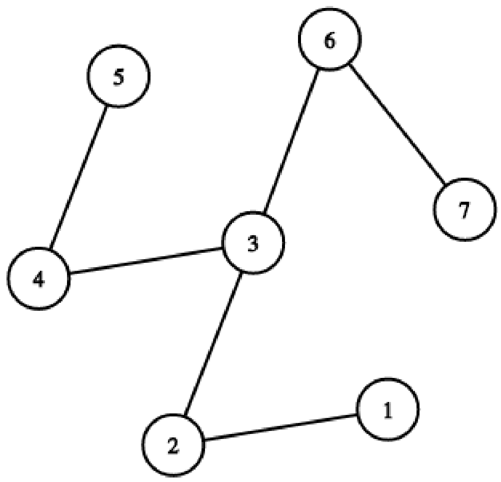
To `flush`, you need to do the following right after printing a query and a line break:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;
- `flush(output)` in Pascal;
- `stdout.flush()` in Python;
- see documentation for other languages.

Examples

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

Note



Problem K. K-pop Strings

Input file:	*standard input*
Output file:	*standard output*
Time limit:	10 seconds
Memory limit:	512 mebibytes

A substring $s[l..r]$ is a tandem repeat if $r - l + 1$ is even and $s[l \dots \frac{l+r-1}{2}] = s[\frac{l+r+1}{2} \dots r]$.

Recently Gennady came up with a method to calculate the number of tandem repeats in a string using suffix structures, and now he came up with a new type of strings based on tandem repeats. Gennady thinks that string s of length n is a K-pop string if there are no tandem repeats of length $\geq n - k$.

Help him find the number of K-pop strings consisting only of the characters '1', '2', ..., '9', 'a', 'b', ..., 'z', modulo 998 244 353.

Input

The first line of input contains two integers n and k: the required length of string and the parameter ($1 \leq n \leq 100$, $0 \leq k \leq 16$).

Output

Output one integer: the number of K-pop strings of length n for the given k, consisting only of nonzero digits and lowercase alphabetic characters, modulo 998 244 353.

Examples

standard input	standard output
1 16	35
4 0	1499400
15 5	911125634

Note

The answer for the first example is 35 because all strings of length 1 are possible: "1", "2", ..., "9", "a", "b", ..., "z".

The answer for the second example is $35^4 - 35^2$.

Problem L. Glass

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Clara is laying N rectangular pieces of grayish glass to make a window. Each piece of glass adds an integer value "gray-factor". Where two pieces of glass overlap, the gray-factor is the sum of their gray-factors.

You know the planned position for each piece of glass and these pieces of glass are placed such that the sides of each rectangle are parallel to either the x-axis or the y-axis

Calculate the total area of the resulting grayish glass window with a gray-factor of at least G.

Input

The first line of the input contains the integer N ($1 \le N \le 1000$), the number of pieces of glass. The second line of input is the integer G ($1 \le G \le 10^9$), the threshold for the gray-factor. Each of the next N lines contain five integers, representing the position of the top-left and bottom-right corners of the i-th piece of tinted glass followed by the tint-factor of that piece of glass. Specifically, the integers are placed in the order x_l, y_t, x_r, y_b, t_i, x_l, y_t, x_r y_b, g_i, where the top-left corner is at (x_l, y_t), (x_l, y_t) and the bottom-right corner is at (x_r, y_b), (x_r, y_b), and gray-factor is g_i. You can assume that $1 \le t_i \le 10^6$). The top-most, left-most coordinate where glass can be placed is $(0, 0)$ and you may assume $0 \le x_l < x_r \le 10^9$ and $0 < y_t < y_b \le 10^9$.

Output

Print the total area of the resulting grayich glass window which has a tint-factor of at least G. All output will be less than 2^{64}, and the output for some test cases will be larger than 2^{32}.

Examples

standard input	standard output
4 3 3 3 12 7 1 5 0 6 9 2 9 0 10 9 1 4 4 11 5 1	5

Problem M. New York Underground

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

New York underground consists of many stations, numbered 1 to N, and many lines connecting them, numbered 1 to M. All stations are connected, but some times you have to switch between lines to travel between two stations. This takes S minutes per switch. You can switch between lines L_1 and L_2 on station S_i if both lines stop at station S_i. The lines go back and forth all day, with the same distance between stations in both directions. Trains leave in both directions from every station every minute.

Alice and Bob are visiting New York. They use the underground a lot, but they keep feeling that they aren't always taking the fastest route. Help them find the fastest route between stations A and B.

Input

The first line of the input consists of a single integer, T, the number of test cases ($1 \le T \le 20$).

Each of the following T cases begins with a line consisting of 5 integers S, N, M, A, B, $1 \le S \le 100$, $2 \le N \le 100$, $1 \le M \le 10$, $A \ne B$, $1 \le A, B \le N$. These numbers give the number of minutes it takes to switch lines, the number of stations, the number of lines, and the start and end stations, respectively.

The next M lines of a test case describe each subway line, and has the following format.
First an integer X, the number of stations on the subway line ($1 \le X \le N$). Then X station specifications follow, separated by spaces, in the order they appear on the line. Each station specification consists of two integers, where the first number, Sn_i, is the station number, and the second number, St_i is the number of minutes from the start of the line to that station ($St_0 = 0$, $St_i > St_{i-1}$, $0 \le St_i \le 1000$, all stations numbered 1 to N will be a part of at least one line; all stations will be reachable from all other stations.

Output

For each test case, output the minimum number of minutes it takes to travel from station A to station B.

Example

standard input	standard output
3 1 5 1 1 5 5 2 0 1 2 3 5 5 10 4 15 3 4 2 1 4 3 1 0 2 2 3 5 3 2 0 3 10 4 11 1 4 2 1 4 3 1 0 2 2 4 15 3 2 0 3 1 4 2	8 9 5

Problem N. Pseudosquare

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Alice considers a *pseudosquare* as rectangle with integer sides x and $x+1$.

Given integer n, check if exists a pseudosquare with area equal to n.

Input

Input contains one integer n ($1 \le n \le 10^9$).

Output

If here are no pseudosquares with area equal to n, printf -1 -1. Otherwise print two integers — lengths of sides of the pseudosquare listed in the ascenging order.

Examples

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

Problem O. String classification

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

A string is considered "`even`" if every letter in the string appears an even number of times; the string is "`odd`" if every letter in the string appears an odd number of times.

Given a string, determine whether the string is even, odd, or neither.

Input

Input consists of a single non-empty string consisting of no more than 70 lowercase English letters.

Output

If string is even, print "`even`", if string is odd, print "`odd`". If the string is not even and is not odd, print "`neither`".

Example

standard input	standard output
cppc	even
coding	odd
contest	neither

Problem P. Window

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Given the width and height of a rectangular wall and the window-border gap (minimum distance required between the perimeter of the wall and the perimeter of the window), determine the area of the largest rectangular window that can be installed on the wall.

Input

The input contains one line with three space-separated positive integers, w, h, and d ($1 \leq w, h \leq 999$, $1 \leq d \leq 99$), representing, respectively, the wall's width, wall's height, and the minimum window- border gap amount needed.

Output

Print one integer which represents the area of the largest rectangular window that can be installed. If it is not possible to install a window, output 0.

Example

standard input	standard output
40 25 5	450
31 21 50	0