Problem A. Bridges in Liyue

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

In Genshin Impact, the protagonist Traveler excels at using teleport waypoints for fast travel. Specifically, the Traveler can teleport from any location to the designated teleport waypoint.

In the Huaguang Stone Forest of Liyue, there are several extremely high peaks that are nearly impossible to climb, connected by wooden bridges. Among these peaks, only one has a teleport waypoint.

Huaguang Stone Forest

Recently, the Traveler received a commission to inspect these bridges. In order to minimize the inspection time, the Traveler hopes to find a path that **starts from the teleport waypoint**, passes through each bridge exactly once, and can **end at any peak**.

Your task is to determine whether the path the Traveler is looking for exists.

Input

The first line consists of two integers, n and m, representing the number of peaks and the number of bridges. The peak with the teleport waypoint is designated as peak 1. ($3 \le n \le 200,\ 2 \le m \le \frac{n(n-1)}{2}$)

The following m lines each contain two integers, u and v, indicating a bridge connecting peaks u and v. ($1 \le u < v \le n$)

It is guaranteed that any peak can be reached from the teleport waypoint, and there is no pair of bridges connecting the same pair of peaks.

Output

If the path exists, output "YES"; otherwise, output "NO". The checker is case-insensitive.

Examples

standard input	standard output
9 8 1 2 2 5 2 3 5 6 6 7 8 9 3 4 6 8	NO
9 10 1 2 2 5 2 3 5 6 6 7 8 9 3 4 6 8 4 7 2 9	YES
9 10 1 2 2 5 2 3 5 6 6 7 8 9 3 4 6 8 4 7 1 9	NO

Problem B. Prophecy of Fontaine

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

This problem is interactive. If you are not familiar with this type of problem, you can first check the guide. In general, you must flush the buffer of the standard output each time you interact before you can read the answer from the standard input of the judge program.`https://codeforces.com/blog/entry/45307`

Fontaine has a prophecy that reads roughly as follows: the water levels in Fontaine will rise, and all the people with sin will be drowned and dissolved, with only the Hydro Archon, Furina, remaining, weeping on her throne.

Now we know that all of this is due to a huge narwhal in Primordial Sea, All-Devouring Narwhal. When it is big enough, it will cause the Primordial Sea water to leak.

All-Devouring Narwhal

Focalors has mastered the function $f(t)$ of the All-Devouring Narwhal's size changing with time, and the minimum size k that causes the water of Primordial Sea water to leak.

Now it is known that $f(t)$ is a strictly increasing and differentiable function, and the function value at some point in $(0, 100)$ is equal to k. However, Focalors does not master the closed form of the function. She can only calculate $f(t)$ based on a certain t, and cannot derive or do other operations.

Input

The judge program for this problem does not support scientific notation.

A finite decimal k represents the minimum body size that causes the Primordial Sea to leak. ($1 \leq k \leq 100$)

Interaction Protocol

Your program needs to interact with the judge program to complete the task, output content to the judge

program using standard output, and read data from the judge program using standard input.

There are two types of content you can output to the judge program:

- ? t, where t is a finite decimal, indicating the value of $f(t)$ to be queried. $(0 \le t \le 100)$
  The judge program will respond with a finite decimal $f(t)$, with an error not exceeding 10^{-12}. $(0 \le f(t) \le 1000)$
- ! t_0, where t_0 is a finite decimal, indicating the root of $f(t) = k$. $(0 \le t_0 \le 100)$
  After sending this message to the judge program, your program should exit immediately, and the judge program will determine whether your answer is correct.

Your answer t_0 will be accepted if and only if it meets at least one of the following conditions: (t^* represents the correct answer)

- $|t_0 - t^*| \le 10^{-4}$
- $\frac{|t_0 - t^*|}{|t^*|} \le 10^{-4}$
- $|f(t_0) - k| \le 10^{-4}$
- $\frac{|f(t_0) - k|}{|k|} \le 10^{-4}$

If the interaction format is incorrect or the final answer is not accepted, the verdict will be Wrong Answer.

In addition, your program can interact up to 100 times at most, otherwise the verdict will be Partial Correct. If the interaction exceeds 100 times, 20% of the score for that test point will be deducted, and 10% of the score will be deducted for each 10 additional interaction until no points remain.

Example

standard input	standard output
50 50.41 50.055625 50.0000309449	 ? 71 ? 70.75 ? 70.7107 ! 70.7107

Problem C. Puzzle in Inazuma

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 seconds
Memory limit:	256 megabytes

During the quest Sacred Sakura Cleansing Ritual, the shrine maiden Hanachirusato takes the Traveler to the bottom of a well.

Here, there are multiple Electro lanterns, with exactly one of them having a magatama count of 1, which cannot be modified. The magatama count on the other lanterns can be modified to any integer greater than or equal to 2. Additionally, there is a torii gate with a puzzle pattern on it.

Sacred Sakura Cleansing Ritual

After modifications are made, the Traveler returns to the prayer seat with a magatama count of 1 to pray. At this point, all lanterns with a magatama count of 1 will connect to all lanterns with a magatama count of 2, all lanterns with a magatama count of 2 will connect to all lanterns with a magatama count of 3, and so on, with all lanterns with a magatama count of n connecting to all lanterns with a magatama count of $n+1$.

After the prayer, the connections between the lanterns must be exactly the same as the pattern on the torii gate. That is, if two lanterns are connected in the pattern, they must be connected after the traveler prays, and vice versa. The pattern on the torii gate does not show magatama counts and guarantees that all lanterns in the pattern are directly or indirectly connected to the lantern with a magatama count of 1. Notice the connections are undirected.

Since the traveler needs to solve similar puzzles multiple times during this quest, he/she would like you to write a program to help him/her solve the puzzle or indicate that the puzzle is unsolvable.

Input

Two integers n, m, representing the number of lanterns and the number of connections in the pattern, where the prayer seat with a magatama count of 1 is seat number 1. ($2 \leq n \leq 10^5$, $1 \leq m \leq \min\{\frac{n(n-1)}{2}, 10^5\}$)

The following m lines, each containing two integers u, v, indicate that u is connected to v. ($1 \leq u < v \leq n$)

Output

If there is a solution, output "YES", followed by a line with $n-1$ integers, representing the magatama count on each lantern other than seat number 1.

If there is no solution, output "NO".

The checker is case-insensitive.

Examples

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

Problem D. Card Game in Mondstant

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

The Genius Invocation TCG is a card game that has taken Teyvat by storm. In this game, each character card boasts three unique skills. At the outset of every turn, players acquire a set number of dice through rolls. Each die can represent one of seven elements or be a colorless die, and players can expend these dice to unleash a variety of skills.

The Cat's Tail in Mondstadt frequently hosts showdowns in the heated battle mode of the Genius Invocation TCG. This mode often features a plethora of unconventional rules, such as reducing the number of dice needed for skills or making all dice colorless.

Now, let's delve into a simplified version of the heated battle mode where all dice are colorless. Normal attacks and elemental skills require only 1 die, while elemental bursts demand 2 dice, and no charging is necessary. In this scenario, only one character occupies the battlefield.

To optimize resource utilization, the goal is to determine the number of ways to exhaust all the dice.

To be specific, starting with k dice, you can execute the following three operations until you have 0 dice remaining:

- Normal attack, consuming 1 die.
- Elemental skill, consuming 1 die.
- Elemental burst, consuming 2 dice.

Two sequences of operations are considered distinct if and only if their total numbers of operations differ or if there exists an i such that the i-th operation in the two sequences differs.

As the result could be extremely large, you only need to output the result modulo 998244353.

The Cat's Tail

Input

An integer k represents the initial number of dice. ($1 \leq k \leq 10^{15}$).

Output

An integer representing the number of ways to use all the dice, modulo 998244353.

Examples

standard input	standard output
1	2
3	12
87842506579379	893188283

Note

Some participants may not know how to calculate x^n (n being a natural number). Here is an algorithm:

Note that:

$$x^0 = 1x^n = x^{n \bmod 2} \cdot (x^2)^{\lfloor n/2 \rfloor}$$

In programming, the recursion can actually be unfolded into a loop with the following algorithm:

$POWER(x, n)$:
    $r = 1$
    while $n > 0$:
        if $n \bmod 2 == 1$:
            $r = r \times x$
        $x = x \times x$
        $n = \lfloor n/2 \rfloor$
    return r

Time Complexity: $O(\lg n)$

Problem E. Mountains in Sumeru

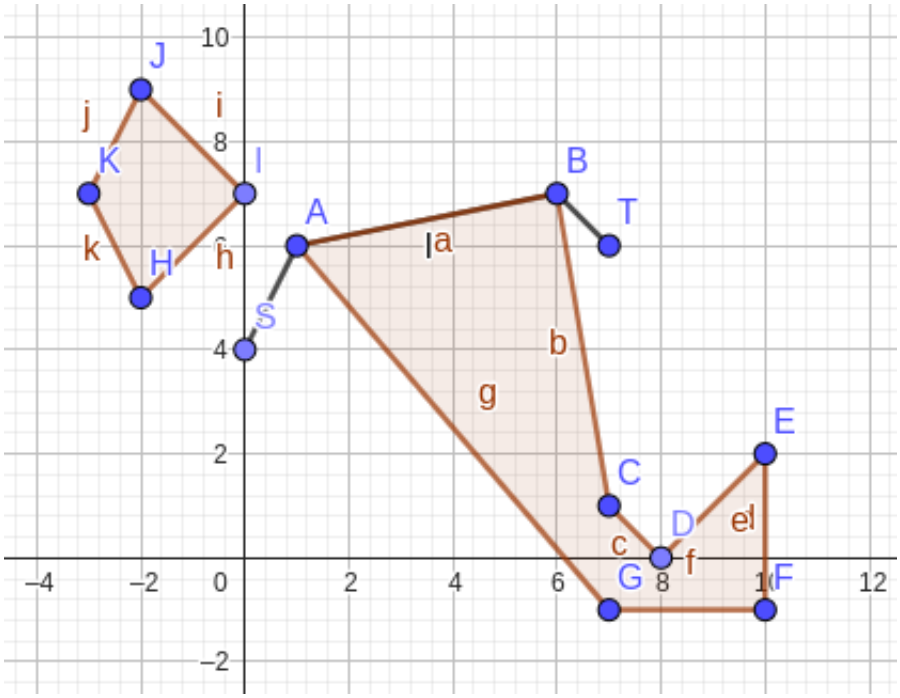
Input file:	`standard input`
Output file:	`standard output`
Time limit:	4.5 seconds
Memory limit:	256 megabytes

The Sumeru Desert is home to many mountains, most of which are unclimbable, causing inconvenience for travelers.

As is widely known, Venti from Mondstadt once leveled Mondstadt's mountainous terrain by submerging the mountain ranges into the sea, giving rise to the Golden Apple Archipelago.

However, Venti can't level these mountains, but he can assist you in ascending them **at most once**.

These mountains can be roughly considered as polygons, as depicted in the diagram below, where the vertices of the shortest path must reside on the vertices of the obstacle polygons.



The shortest path from S to T

Considering this problem might be too challenging for you, Venti has taken care of the geometric relationships. Specifically, he will provide you with information on the starting point, ending point, pairs of points between vertices of each polygon that are reachable without climbing, exactly once reachable pairs with a climb, and the distances between these pairs.

If you find it hard to grasp, don't worry. In formal terms, you are given an undirected weighted graph with edges in two colors (red and blue). Your task is to find the shortest path from the starting point to the ending point, passing through at most one blue edge, or confirm that reaching the ending point is impossible.

Venti is adorable!

Input

The first line consists of three positive integers, n, m, and k ($3 \leq n \leq 2 \cdot 10^5$, $2 \leq m + k \leq \min\{\frac{(n+1)(n+2)}{2}, 2 \cdot 10^5\}$), representing the number of vertices in the polygon (**excluding the starting point and ending point, numbered** $3, 4, ..., n+2$), the quantity of point pairs reachable without climbing (number of red edges), and the quantity of point pairs reachable exactly once with a climb (number of blue edges).

Following that, there are m lines, each containing three integers, i, j, and w ($1 \leq i < j \leq n + 2$, $0 \leq w \leq 10^4$), indicating that points i and j are reachable without climbing (red edge), with a distance of w.

Afterwards, there are k lines, each containing three integers, i, j, and w ($1 \leq i < j \leq n+2$, $0 \leq w \leq 10^4$), indicating that points i and j are reachable exactly once with a climb (blue edge), with a distance of w.

Output

An integer representing the shortest distance from the starting point 1 to the ending point 2, or -1 indicating that it is not possible to reach the destination.

Examples

standard input	standard output
3 5 2 2 3 1 3 4 2 4 5 3 1 5 3 1 4 9 1 3 10 3 5 4	8
3 0 4 2 3 1 3 4 1 4 5 1 1 5 1	-1
4 4 1 1 6 1 2 3 0 4 5 0 3 4 0 2 6 3	4