time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

PolandBall lives in a forest with his family. There are some trees in the forest. Trees are undirected acyclic graphs with k vertices and k - 1 edges, where k is some integer. Note that one vertex **is** a valid tree.

There is exactly one relative living in each vertex of each tree, they have unique ids from 1 to n. For each Ball i we know the id of its most distant relative living on the same tree. If there are several such vertices, we only know the value of the one with smallest id among those.

How many trees are there in the forest?

Input

The first line contains single integer n $(1 \leq n \leq 10^4)$ — the number of Balls living in the forest.

The second line contains a sequence $p_1, p_2, ..., p_n$ of length n, where $(1 \leq p_i \leq n)$ holds and p_i denotes the most distant from Ball i relative living on the same tree. If there are several most distant relatives living on the same tree, p_i is the id of one with the smallest id.

It's guaranteed that the sequence p corresponds to some valid forest.

Hacking: To hack someone, you should provide a **correct forest** as a test. The sequence p will be calculated according to the forest and given to the solution you try to hack as input. Use the following format:

In the first line, output the integer n $(1 \leq n \leq 10^4)$ — the number of Balls and the integer m $(0 \leq m < n)$ — the total number of edges in the forest. Then m lines should follow. The i-th of them should contain two integers a_i and b_i and represent an edge between vertices in which relatives a_i and b_i live. For example, the first sample is written as follows:

```
5 3
1 2
3 4
4 5
```

Output

You should output the number of trees in the forest where PolandBall lives.

Interaction

From the technical side, this problem is interactive. However, it should not affect you (except hacking) since there is no interaction.

Examples

input

```
5
2 1 5 3 3
```

output

```
2
```

input

```
1
1
```

output

```
1
```

Note

In the first sample testcase, possible forest is: `1-2 3-4-5`.

There are 2 trees overall.

In the second sample testcase, the only possible graph is one vertex and no edges. Therefore, there is only one tree.

time limit per test: 0.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Bomboslav likes to look out of the window in his room and watch lads outside playing famous shell game. The game is played by two persons: operator and player. Operator takes three similar opaque shells and places a ball beneath one of them. Then he shuffles the shells by swapping some pairs and the player has to guess the current position of the ball.

Bomboslav noticed that guys are not very inventive, so the operator always swaps the left shell with the middle one during odd moves (first, third, fifth, etc.) and always swaps the middle shell with the right one during even moves (second, fourth, etc.).

Let's number shells from 0 to 2 from left to right. Thus the left shell is assigned number 0, the middle shell is 1 and the right shell is 2. Bomboslav has missed the moment when the ball was placed beneath the shell, but he knows that exactly n movements were made by the operator and the ball was under shell x at the end. Now he wonders, what was the initial position of the ball?

Input

The first line of the input contains an integer n ($1 \leq n \leq 2 \cdot 10^9$) — the number of movements made by the operator.

The second line contains a single integer x ($0 \leq x \leq 2$) — the index of the shell where the ball was found after n movements.

Output

Print one integer from 0 to 2 — the index of the shell where the ball was initially placed.

Examples

input

```
4
2
```

output

```
1
```

input

```
1
1
```

output

```
0
```

Note

In the first sample, the ball was initially placed beneath the middle shell and the operator completed four movements.

1. During the first move operator swapped the left shell and the middle shell. The ball is now under the left shell.
2. During the second move operator swapped the middle shell and the right one. The ball is still under the left shell.
3. During the third move operator swapped the left shell and the middle shell again. The ball is again in the middle.
4. Finally, the operators swapped the middle shell and the right shell. The ball is now beneath the right shell.

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Sasha and Kolya decided to get drunk with Coke, again. This time they have k types of Coke. i-th type is characterised by its carbon dioxide concentration $\frac{a_i}{1000}$. Today, on the party in honour of Sergiy of Vancouver they decided to prepare a glass of Coke with carbon dioxide concentration $\frac{n}{1000}$. The drink should also be tasty, so the glass can contain only integer number of liters of each Coke type (some types can be not presented in the glass). Also, they want to minimize the total volume of Coke in the glass.

Carbon dioxide concentration is defined as the volume of carbone dioxide in the Coke divided by the total volume of Coke. When you mix two Cokes, the volume of carbon dioxide sums up, and the total volume of Coke sums up as well.

Help them, find the minimal natural number of liters needed to create a glass with carbon dioxide concentration $\frac{n}{1000}$. Assume that the friends have unlimited amount of each Coke type.

Input

The first line contains two integers n, k ($0 \le n \le 1000$, $1 \le k \le 10^6$) — carbon dioxide concentration the friends want and the number of Coke types.

The second line contains k integers $a_1, a_2, ..., a_k$ ($0 \le a_i \le 1000$) — carbon dioxide concentration of each type of Coke. Some Coke types can have same concentration.

Output

Print the minimal natural number of liter needed to prepare a glass with carbon dioxide concentration $\frac{n}{1000}$, or -1 if it is impossible.

Examples

input

```
400 4
100 300 450 500
```

output

```
2
```

input

```
50 2
100 25
```

output

```
3
```

Note

In the first sample case, we can achieve concentration $\frac{400}{1000}$ using one liter of Coke of types $\frac{300}{1000}$ and $\frac{500}{1000}$: $\frac{300+500}{1000+1000} = \frac{400}{1000}$.

In the second case, we can achieve concentration $\frac{50}{1000}$ using two liters of $\frac{25}{1000}$ type and one liter of $\frac{100}{1000}$ type: $\frac{25+25+100}{3 \cdot 1000} = \frac{50}{1000}$.

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Everyone knows that DNA strands consist of nucleotides. There are four types of nucleotides: "`A`", "`T`", "`G`", "`C`". A DNA strand is a sequence of nucleotides. Scientists decided to track evolution of a rare species, which DNA strand was string s initially.

Evolution of the species is described as a sequence of changes in the DNA. Every change is a change of some nucleotide, for example, the following change can happen in DNA strand "`AAGC`": the second nucleotide can change to "`T`" so that the resulting DNA strand is "`ATGC`".

Scientists know that some segments of the DNA strand can be affected by some unknown infections. They can represent an infection as a sequence of nucleotides. Scientists are interested if there are any changes caused by some infections. Thus they sometimes want to know the value of impact of some infection to some segment of the DNA. This value is computed as follows:

- Let the infection be represented as a string e, and let scientists be interested in DNA strand segment starting from position l to position r, inclusive.
- Prefix of the string $eee...$ (i.e. the string that consists of infinitely many repeats of string e) is written under the string s from position l to position r, inclusive.
- The value of impact is the number of positions where letter of string s coincided with the letter written under it.

Being a developer, Innokenty is interested in bioinformatics also, so the scientists asked him for help. Innokenty is busy preparing VK Cup, so he decided to delegate the problem to the competitors. Help the scientists!

Input

The first line contains the string s ($1 \le |s| \le 10^5$) that describes the initial DNA strand. It consists only of capital English letters "`A`", "`T`", "`G`" and "`C`".

The next line contains single integer q ($1 \le q \le 10^5$) — the number of events.

After that, q lines follow, each describes one event. Each of the lines has one of two formats:

- `1 x c`, where x is an integer ($1 \le x \le |s|$), and c is a letter "`A`", "`T`", "`G`" or "`C`", which means that there is a change in the DNA: the nucleotide at position x is now c.
- `2 l r e`, where l, r are integers ($1 \le l \le r \le |s|$), and e is a string of letters "`A`", "`T`", "`G`" and "`C`" ($1 \le |e| \le 10$), which means that scientists are interested in the value of impact of infection e to the segment of DNA strand from position l to position r, inclusive.

Output

For each scientists' query (second type query) print a single integer in a new line — the value of impact of the infection on the DNA.

Examples

input

```
ATGCATGC
4
2 1 8 ATGC
2 2 6 TTT
1 4 T
2 2 6 TA
```

output

```
8
```

```
2
4
```

input

```
GAGTTGTTAA
6
2 3 4 TATGGTG
1 1 T
1 6 G
2 5 9 AGTAATA
1 10 G
2 2 6 TTGT
```

output

```
0
3
1
```

Note

Consider the first example. In the first query of second type all characters coincide, so the answer is 8. In the second query we compare string "TTTTT..." and the substring "TGCAT". There are two matches. In the third query, after the DNA change, we compare string "TATAT..."' with substring "TGTAT". There are 4 matches.

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Bear Limak examines a social network. Its main functionality is that two members can become friends (then they can talk with each other and share funny pictures).

There are n members, numbered 1 through n. m pairs of members are friends. Of course, a member can't be a friend with themselves.

Let `A-B` denote that members `A` and `B` are friends. Limak thinks that a network is *reasonable* if and only if the following condition is satisfied: For every three **distinct** members (`X`, `Y`, `Z`), if `X-Y` and `Y-Z` then also `X-Z`.

For example: if Alan and Bob are friends, and Bob and Ciri are friends, then Alan and Ciri should be friends as well.

Can you help Limak and check if the network is reasonable? Print "`YES`" or "`NO`" accordingly, without the quotes.

Input

The first line of the input contain two integers n and m ($3 \leq n \leq 150\,000$, $0 \leq m \leq min(150\,000, \frac{n \cdot (n-1)}{2})$) — the number of members and the number of pairs of members that are friends.

The i-th of the next m lines contains two distinct integers a_i and b_i ($1 \leq a_i, b_i \leq n, a_i \neq b_i$). Members a_i and b_i are friends with each other. No pair of members will appear more than once in the input.

Output

If the given network is reasonable, print "`YES`" in a single line (without the quotes). Otherwise, print "`NO`" in a single line (without the quotes).

Examples

input

```
4 3
1 3
3 4
1 4
```

output

```
YES
```

input

```
4 4
3 1
2 3
3 4
1 2
```

output

```
NO
```

input

```
10 4
4 3
5 10
8 9
```

```
1 2
```

output

```
YES
```

input

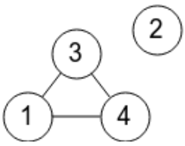
```
3 2
1 2
2 3
```

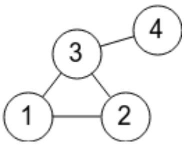
output

```
NO
```

Note

The drawings below show the situation in the first sample (on the left) and in the second sample (on the right). Each edge represents two members that are friends. The answer is "NO" in the second sample because members $(2, 3)$ are friends and members $(3, 4)$ are friends, while members $(2, 4)$ are not.





time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Yet another round on DecoForces is coming! Grandpa Maks wanted to participate in it but someone has stolen his precious sofa! And how can one perform well with such a major loss?

Fortunately, the thief had left a note for Grandpa Maks. This note got Maks to the sofa storehouse. Still he had no idea which sofa belongs to him as they all looked the same!

The storehouse is represented as matrix $n \times m$. Every sofa takes two neighbouring by some side cells. No cell is covered by more than one sofa. There can be empty cells.

Sofa A is standing to the left of sofa B if there exist two such cells a and b that $\boldsymbol{x_a < x_b}$, a is covered by A and b is covered by B. Sofa A is standing to the top of sofa B if there exist two such cells a and b that $\boldsymbol{y_a < y_b}$, a is covered by A and b is covered by B. Right and bottom conditions are declared the same way.

Note that in all conditions $\boldsymbol{A \neq B}$. Also some sofa A can be both to the top of another sofa B and to the bottom of it. The same is for left and right conditions.

The note also stated that there are cnt_l sofas to the left of Grandpa Maks's sofa, cnt_r — to the right, cnt_t — to the top and cnt_b — to the bottom.

Grandpa Maks asks you to help him to identify his sofa. It is guaranteed that there is no more than one sofa of given conditions.

Output the number of Grandpa Maks's sofa. If there is no such sofa that all the conditions are met for it then output `-1`.

Input

The first line contains one integer number d ($1 \leq d \leq 10^5$) — the number of sofas in the storehouse.

The second line contains two integer numbers n, m ($1 \leq n, m \leq 10^5$) — the size of the storehouse.

Next d lines contains four integer numbers x_1, y_1, x_2, y_2 ($1 \leq x_1, x_2 \leq n$, $1 \leq y_1, y_2 \leq m$) — coordinates of the i-th sofa. It is guaranteed that cells (x_1, y_1) and (x_2, y_2) have common side, $(x_1, y_1) \neq (x_2, y_2)$ and no cell is covered by more than one sofa.

The last line contains four integer numbers cnt_l, cnt_r, cnt_t, cnt_b ($0 \leq cnt_l, cnt_r, cnt_t, cnt_b \leq d - 1$).

Output

Print the number of the sofa for which all the conditions are met. Sofas are numbered 1 through d as given in input. If there is no such sofa then print `-1`.

Examples

input

```
2
3 2
3 1 3 2
1 2 2 2
1 0 0 1
```

output

```
1
```

input

```
3
10 10
1 2 1 1
5 5 6 5
6 4 5 4
2 1 2 0
```

output

```
2
```

input

```
2
2 2
2 1 1 1
1 2 2 2
1 0 0 0
```

output

```
-1
```

Note

Let's consider the second example.

- The first sofa has 0 to its left, 2 sofas to its right ($(1, 1)$ is to the left of both $(5, 5)$ and $(5, 4)$), 0 to its top and 2 to its bottom (both 2nd and 3rd sofas are below).
- The second sofa has $cnt_l = 2$, $cnt_r = 1$, $cnt_t = 2$ and $cnt_b = 0$.
- The third sofa has $cnt_l = 2$, $cnt_r = 1$, $cnt_t = 1$ and $cnt_b = 1$.

So the second one corresponds to the given conditions.

In the third example

- The first sofa has $cnt_l = 1$, $cnt_r = 1$, $cnt_t = 0$ and $cnt_b = 1$.
- The second sofa has $cnt_l = 1$, $cnt_r = 1$, $cnt_t = 1$ and $cnt_b = 0$.

And there is no sofa with the set $(1, 0, 0, 0)$ so the answer is `-1`.

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Boris really likes numbers and even owns a small shop selling interesting numbers. He has n decimal numbers B_i. Cost of the number in his shop is equal to the sum of costs of its digits. You are given the values c_d, where c_d is the cost of the digit d. Of course, Boris is interested in that numbers he owns have the maximum cost possible.

Recently Boris got hold of the magical artifact A, which can allow him to increase the cost of his collection. Artifact is a string, consisting of digits and '?' symbols. To use the artifact, Boris must replace all '?' with digits to get a decimal number without leading zeros (it is also not allowed to get number 0). After that, the resulting number is added to all numbers B_i in Boris' collection. He uses the artifact exactly once.

What is the maximum cost of the collection Boris can achieve after using the artifact?

Input

First line contains artifact A, consisting of digits '0'–'9' and '?' symbols ($1 \le |A| \le 1000$). Next line contains n — the amount of numbers in Boris' collection ($1 \le n \le 1000$). Next n lines contain integers B_i ($1 \le B_i < 10^{1000}$). A doesn't start with '0'.

Last line contains ten integers — costs of digits $c_0, c_1, ..., c_9$ ($0 \le c_i \le 1000$).

Output

Output one integer — the maximum possible cost of the collection after using the artifact.

Examples

input

```
42
3
89
1
958
0 0 1 1 2 2 3 3 4 4
```

output

```
4
```

input

```
?5?
4
2203
5229
276
6243
2 1 6 1 1 2 5 2 2 3
```

output

```
62
```

Note

In the second sample input, the optimal way is to compose the number 453. After adding this number, Boris will have numbers 2656, 5682, 729 and 6696. The total cost of all digits in them is equal to $18 + 15 + 11 + 18 = 62$.

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Igor the analyst fell asleep on the work and had a strange dream. In the dream his desk was crowded with computer mice, so he bought a mousetrap to catch them.

The desk can be considered as an infinite plane, then the mousetrap is a rectangle which sides are parallel to the axes, and which opposite sides are located in points (x_1, y_1) and (x_2, y_2).

Igor wants to catch all mice. Igor has analysed their behavior and discovered that each mouse is moving along a straight line with constant speed, the speed of the i-th mouse is equal to (v_i^x, v_i^y), that means that the x coordinate of the mouse increases by v_i^x units per second, while the y coordinates increases by v_i^y units. The mousetrap is open initially so that the mice are able to move freely on the desk. Igor can close the mousetrap at any moment catching all the mice that are **strictly** inside the mousetrap.

Igor works a lot, so he is busy in the dream as well, and he asks you to write a program that by given mousetrap's coordinates, the initial coordinates of the mice and their speeds determines the earliest time moment in which he is able to catch all the mice. Please note that Igor can close the mousetrap only once.

Input

The first line contains single integer n $(1 \leq n \leq 100\,000)$ — the number of computer mice on the desk.

The second line contains four integers x_1, y_1, x_2 and y_2 $(0 \leq x_1 \leq x_2 \leq 100\,000)$, $(0 \leq y_1 \leq y_2 \leq 100\,000)$ — the coordinates of the opposite corners of the mousetrap.

The next n lines contain the information about mice.

The i-th of these lines contains four integers r_i^x, r_i^y, v_i^x and v_i^y, $(0 \leq r_i^x, r_i^y \leq 100\,000$, $-100\,000 \leq v_i^x, v_i^y \leq 100\,000)$, where (r_i^x, r_i^y) is the initial position of the mouse, and (v_i^x, v_i^y) is its speed.

Output

In the only line print minimum possible non-negative number t such that if Igor closes the mousetrap at t seconds from the beginning, then all the mice are **strictly** inside the mousetrap. If there is no such t, print `-1`.

Your answer is considered correct if its absolute or relative error doesn't exceed 10^{-6}.

Formally, let your answer be a, and the jury's answer be b. Your answer is considered correct if $\frac{|a-b|}{max(1,b)} \leq 10^{-6}$.

Examples

input

```
4
7 7 9 8
3 5 7 5
7 5 2 4
3 3 7 8
6 6 3 2
```

output

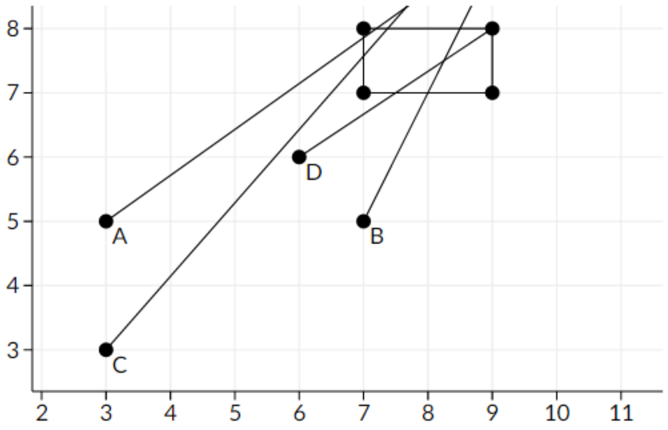
```
0.57142857142857139685
```

input
4 7 7 9 8 0 3 -5 4 5 0 5 4 9 9 -1 -6 10 5 -7 -10
output
-1

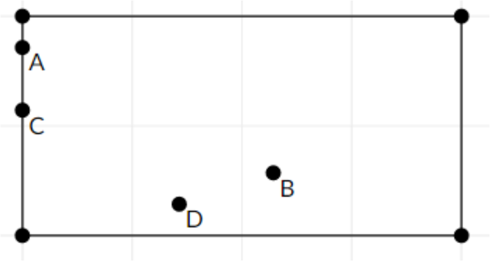
Note

Here is a picture of the first sample

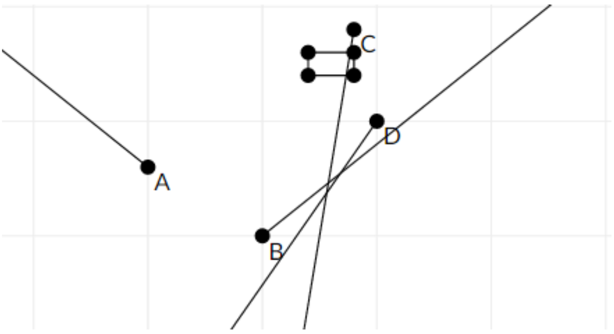
Points A, B, C, D - start mice positions, segments are their paths.



Then, at first time when all mice will be in rectangle it will be looks like this:



Here is a picture of the second sample



Points A, D, B will never enter rectangle.

Privacy Policy

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given an undirected graph, constisting of n vertices and m edges. Each edge of the graph has some non-negative integer written on it.

Let's call a triple (u, v, s) **interesting**, if $1 \le u < v \le n$ and there is a path (**possibly non-simple**, i.e. it can visit the same vertices and edges multiple times) between vertices u and v such that xor of all numbers written on the edges of this path is equal to s. **When we compute the value s for some path, each edge is counted in xor as many times, as it appear on this path.** It's not hard to prove that there are finite number of such triples.

Calculate the sum over modulo $10^9 + 7$ of the values of s over all **interesting** triples.

Input

The first line of the input contains two integers n and m ($1 \le n \le 100\,000$, $0 \le m \le 200\,000$) — numbers of vertices and edges in the given graph.

The follow m lines contain three integers u_i, v_i and t_i ($1 \le u_i, v_i \le n$, $0 \le t_i \le 10^{18}$, $u_i \ne v_i$) — vertices connected by the edge and integer written on it. It is guaranteed that graph doesn't contain self-loops and multiple edges.

Output

Print the single integer, equal to the described sum over modulo $10^9 + 7$.

Examples

input

```
4 4
1 2 1
1 3 2
2 3 3
3 4 1
```

output

```
12
```

input

```
4 4
1 2 1
2 3 2
3 4 4
4 1 8
```

output

```
90
```

input

```
8 6
1 2 2
2 3 1
2 4 4
4 5 5
4 6 3
7 8 5
```

output

```
62
```

Note

In the first example the are 6 interesting triples:

1. $(1, 2, 1)$
2. $(1, 3, 2)$
3. $(1, 4, 3)$
4. $(2, 3, 3)$
5. $(2, 4, 2)$
6. $(3, 4, 1)$

The sum is equal to $1 + 2 + 3 + 3 + 2 + 1 = 12$.
In the second example the are 12 interesting triples:

1. $(1, 2, 1)$
2. $(2, 3, 2)$
3. $(1, 3, 3)$
4. $(3, 4, 4)$
5. $(2, 4, 6)$
6. $(1, 4, 7)$
7. $(1, 4, 8)$
8. $(2, 4, 9)$
9. $(3, 4, 11)$
10. $(1, 3, 12)$
11. $(2, 3, 13)$
12. $(1, 2, 14)$

The sum is equal to $1 + 2 + 3 + 4 + 6 + 7 + 8 + 9 + 11 + 12 + 13 + 14 = 90$.