A. Chess Tourney

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

Berland annual chess tournament is coming!

Organizers have gathered $2 \cdot n$ chess players who should be divided into two teams with n people each. The first team is sponsored by BerOil and the second team is sponsored by BerMobile. Obviously, organizers should guarantee the win for the team of BerOil.

Thus, organizers should divide all $2 \cdot n$ players into two teams with n people each in such a way that the first team always wins.

Every chess player has its rating r_i. It is known that chess player with the greater rating always wins the player with the lower rating. If their ratings are equal then any of the players can win.

After teams assignment there will come a drawing to form n pairs of opponents: in each pair there is a player from the first team and a player from the second team. Every chess player should be in exactly one pair. Every pair plays once. The drawing is totally random.

Is it possible to divide all $2 \cdot n$ players into two teams with n people each so that the player from the first team in every pair wins **regardless** of the results of the drawing?

Input

The first line contains one integer n ($1 \le n \le 100$).

The second line contains $2 \cdot n$ integers $a_1, a_2, \dots a_{2n}$ ($1 \le a_i \le 1000$).

Output

If it's possible to divide all $2 \cdot n$ players into two teams with n people each so that the player from the first team in every pair wins regardless of the results of the drawing, then print "`YES`". Otherwise print "`NO`".

Examples

input

```
2
1 3 2 4
```

output

```
YES
```

input

```
1
3 3
```

output

```
NO
```

B. Luba And The Ticket

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

Luba has a ticket consisting of 6 digits. In one move she can choose digit in any position and replace it with arbitrary digit. She wants to know the minimum number of digits she needs to replace in order to make the ticket lucky.

The ticket is considered lucky if the sum of first three digits equals to the sum of last three digits.

Input

You are given a string consisting of 6 characters (all characters are digits from 0 to 9) — this string denotes Luba's ticket. The ticket can start with the digit 0.

Output

Print one number — the minimum possible number of digits Luba needs to replace to make the ticket lucky.

Examples

input

```
000000
```

output

```
0
```

input

```
123456
```

output

```
2
```

input

```
111000
```

output

```
1
```

Note

In the first example the ticket is already lucky, so the answer is 0.

In the second example Luba can replace 4 and 5 with zeroes, and the ticket will become lucky. It's easy to see that at least two replacements are required.

In the third example Luba can replace any zero with 3. It's easy to see that at least one replacement is required.

C. Two TVs

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

Polycarp is a great fan of television.

He wrote down all the TV programs he is interested in for today. His list contains n shows, i-th of them starts at moment l_i and ends at moment r_i.

Polycarp owns two TVs. He can watch two different shows simultaneously with two TVs but he can only watch one show at any given moment on a single TV. If one show ends at the same moment some other show starts then you can't watch them on a single TV.

Polycarp wants to check out all n shows. Are two TVs enough to do so?

Input

The first line contains one integer n ($1 \le n \le 2 \cdot 10^5$) — the number of shows.

Each of the next n lines contains two integers l_i and r_i ($0 \le l_i < r_i \le 10^9$) — starting and ending time of i-th show.

Output

If Polycarp is able to check out all the shows using only two TVs then print "YES" (without quotes). Otherwise, print "NO" (without quotes).

Examples

input

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

output

```
YES
```

input

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

output

```
NO
```

D. Driving Test

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

Polycarp has just attempted to pass the driving test. He ran over the straight road with the signs of four types.

- speed limit: this sign comes with a positive integer number — maximal speed of the car after the sign (cancel the action of the previous sign of this type);
- overtake is allowed: this sign means that after some car meets it, it can overtake any other car;
- no speed limit: this sign cancels speed limit if any (car can move with arbitrary speed after this sign);
- no overtake allowed: some car can't overtake any other car after this sign.

Polycarp goes past the signs consequentially, each new sign cancels the action of all the previous signs of it's kind (speed limit/overtake). It is possible that two or more "no overtake allowed" signs go one after another with zero "overtake is allowed" signs between them. It works with "no speed limit" and "overtake is allowed" signs as well.

In the beginning of the ride overtake is allowed and there is no speed limit.

You are given the sequence of events in chronological order — events which happened to Polycarp during the ride. There are events of following types:

1. Polycarp changes the speed of his car to specified (this event comes with a positive integer number);
2. Polycarp's car overtakes the other car;
3. Polycarp's car goes past the "speed limit" sign (this sign comes with a positive integer);
4. Polycarp's car goes past the "overtake is allowed" sign;
5. Polycarp's car goes past the "no speed limit";
6. Polycarp's car goes past the "no overtake allowed";

It is guaranteed that the first event in chronological order is the event of type 1 (Polycarp changed the speed of his car to specified).

After the exam Polycarp can justify his rule violations by telling the driving instructor that he just didn't notice some of the signs. What is the minimal number of signs Polycarp should say he didn't notice, so that he would make no rule violations from his point of view?

Input

The first line contains one integer number n $(1 \le n \le 2 \cdot 10^5)$ — number of events.

Each of the next n lines starts with integer t $(1 \le t \le 6)$ — the type of the event.

An integer s $(1 \le s \le 300)$ follows in the query of the first and the third type (if it is the query of first type, then it's new speed of Polycarp's car, if it is the query of third type, then it's new speed limit).

It is guaranteed that the first event in chronological order is the event of type 1 (Polycarp changed the speed of his car to specified).

Output

Print the minimal number of road signs Polycarp should say he didn't notice, so that he would make no rule violations from his point of view.

Examples

input

```
11
1 100
3 70
4
2
3 120
5
3 120
6
1 150
4
3 300
```

output

```
2
```

input

```
5
1 100
3 200
2
4
5
```

output

```
0
```

input

```
7
1 20
2
6
4
6
6
2
```

output

```
2
```

Note

In the first example Polycarp should say he didn't notice the "speed limit" sign with the limit of 70 and the second "speed limit" sign with the limit of 120.

In the second example Polycarp didn't make any rule violation.

In the third example Polycarp should say he didn't notice both "no overtake allowed" that came after "overtake is allowed" sign.

E. Fire in the City

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

The capital of Berland looks like a rectangle of size $n \times m$ of the square blocks of same size.

Fire!

It is known that $k+1$ blocks got caught on fire ($k+1 \le n \cdot m$). Those blocks are centers of ignition. Moreover positions of k of these centers are known and one of these stays unknown. All $k+1$ positions are distinct.

The fire goes the following way: during the zero minute of fire only these $k+1$ centers of ignition are burning. Every next minute the fire goes to all neighbouring blocks to the one which is burning. You can consider blocks to burn for so long that this time exceeds the time taken in the problem. The neighbouring blocks are those that touch the current block by a side or by a corner.

Berland Fire Deparment wants to estimate the minimal time it takes the fire to lighten up the whole city. Remember that the positions of k blocks (centers of ignition) are known and $(k+1)$-th can be positioned in any other block.

Help Berland Fire Department to estimate the minimal time it takes the fire to lighten up the whole city.

Input

The first line contains three integers n, m and k ($1 \le n, m \le 10^9$, $1 \le k \le 500$).

Each of the next k lines contain two integers x_i and y_i ($1 \le x_i \le n$, $1 \le y_i \le m$) — coordinates of the i-th center of ignition. It is guaranteed that the locations of all centers of ignition are distinct.

Output

Print the minimal time it takes the fire to lighten up the whole city (in minutes).

Examples

input

```
7 7 3
1 2
2 1
5 5
```

output

```
3
```

input

```
10 5 1
3 3
```

output

```
2
```

Note

In the first example the last block can have coordinates $(4, 4)$.

In the second example the last block can have coordinates $(8, 3)$.

F. Guards In The Storehouse

time limit per test: 1.5 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Polycarp owns a shop in the capital of Berland. Recently the criminal activity in the capital increased, so Polycarp is thinking about establishing some better security in the storehouse of his shop.

The storehouse can be represented as a matrix with n rows and m columns. Each element of the matrix is either `.` (an empty space) or `x` (a wall).

Polycarp wants to hire some guards (possibly zero) to watch for the storehouse. Each guard will be in some cell of matrix and will protect every cell to the right of his own cell and every cell to the bottom of his own cell, until the nearest wall. More formally, if the guard is standing in the cell (x_0, y_0), then he protects cell (x_1, y_1) if all these conditions are met:

- (x_1, y_1) is an empty cell;
- either $x_0 = x_1$ and $y_0 \le y_1$, or $x_0 \le x_1$ and $y_0 = y_1$;
- there are no walls between cells (x_0, y_0) and (x_1, y_1). **There can be a guard between these cells, guards can look through each other.**

Guards can be placed only in empty cells (and can protect only empty cells). The *plan* of placing the guards is some set of cells where guards will be placed (of course, two plans are different if there exists at least one cell that is included in the first plan, but not included in the second plan, or vice versa). Polycarp calls a plan *suitable* if there is **not more than one** empty cell that is not protected.

Polycarp wants to know the number of suitable plans. Since it can be very large, you have to output it modulo $10^9 + 7$.

Input

The first line contains two numbers n and m — the length and the width of the storehouse ($1 \le n, m \le 250$, $1 \le nm \le 250$).

Then n lines follow, ith line contains a string consisting of m characters — ith row of the matrix representing the storehouse. Each character is either `.` or `x`.

Output

Output the number of suitable plans modulo $10^9 + 7$.

Examples

input
```
1 3
.x.
```
output
```
3
```

input
```
2 2
xx
xx
```
output
```
1
```

input
```
2 2
..
..
```
output
```
10
```

input
```
3 1
x
.
x
```
output
```
2
```

Note

In the first example you have to put at least one guard, so there are three possible arrangements: one guard in the cell $(1, 1)$, one guard in the cell $(1, 3)$, and two guards in both these cells.

G. Shortest Path Problem?

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You are given an undirected graph with weighted edges. The length of some path between two vertices is the bitwise xor of weights of all edges belonging to this path (if some edge is traversed more than once, then it is included in bitwise xor the same number of times). You have to find the minimum length of path between vertex 1 and vertex n.

Note that graph can contain multiple edges and loops. It is guaranteed that the graph is connected.

Input

The first line contains two numbers n and m ($1 \le n \le 100000$, $n - 1 \le m \le 100000$) — the number of vertices and the number of edges, respectively.

Then m lines follow, each line containing three integer numbers x, y and w ($1 \le x, y \le n$, $0 \le w \le 10^8$). These numbers denote an edge that connects vertices x and y and has weight w.

Output

Print one number — the minimum length of path between vertices 1 and n.

Examples

input

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

output

```
2
```

input

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

output

```
0
```