Codeforces Round #354 (Div. 2)

A. Nicholas and Permutation

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

Nicholas has an array a that contains n **distinct** integers from 1 to n. In other words, Nicholas has a permutation of size n.

Nicholas want the minimum element (integer 1) and the maximum element (integer n) to be as far as possible from each other. He wants to perform exactly one swap in order to maximize the distance between the minimum and the maximum elements. The distance between two elements is considered to be equal to the absolute difference between their positions.

Input

The first line of the input contains a single integer n $(2 \leq n \leq 100)$ — the size of the permutation.

The second line of the input contains n distinct integers $a_1, a_2, ..., a_n$ $(1 \leq a_i \leq n)$, where a_i is equal to the element at the i-th position.

Output

Print a single integer — the maximum possible distance between the minimum and the maximum elements Nicholas can achieve by performing exactly one swap.

Examples

input

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

output

```
3
```

input

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

output

```
6
```

input

```
6
6 5 4 3 2 1
```

output

```
5
```

Note

In the first sample, one may obtain the optimal answer by swapping elements 1 and 2.

In the second sample, the minimum and the maximum elements will be located in the opposite ends of the array if we swap 7 and 2.

In the third sample, the distance between the minimum and the maximum elements is already maximum possible, so we just perform some unnecessary swap, for example, one can swap 5 and 2.

B. Pyramid of Glasses

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

Mary has just graduated from one well-known University and is now attending celebration party. Students like to dream of a beautiful life, so they used champagne glasses to construct a small pyramid. The height of the pyramid is n. The top level consists of only 1 glass, that stands on 2 glasses on the second level (counting from the top), then 3 glasses on the third level and so on.The bottom level consists of n glasses.

Vlad has seen in the movies many times how the champagne beautifully flows from top levels to bottom ones, filling all the glasses simultaneously. So he took a bottle and started to pour it in the glass located at the top of the pyramid.

Each second, Vlad pours to the top glass the amount of champagne equal to the size of exactly one glass. If the glass is already full, but there is some champagne flowing in it, then it pours over the edge of the glass and is equally distributed over two glasses standing under. If the overflowed glass is at the bottom level, then the champagne pours on the table. For the purpose of this problem we consider that champagne is distributed among pyramid glasses immediately. Vlad is interested in the number of completely full glasses if he stops pouring champagne in t seconds.

Pictures below illustrate the pyramid consisting of three levels.

Input

The only line of the input contains two integers n and t $(1 \le n \le 10, 0 \le t \le 10\,000)$ — the height of the pyramid and the number of seconds Vlad will be pouring champagne from the bottle.

Output

Print the single integer — the number of completely full glasses after t seconds.

Examples

input

```
3 5
```

output

```
4
```

input

```
4 8
```

output

```
6
```

Note

In the first sample, the glasses full after 5 seconds are: the top glass, both glasses on the second level and the middle glass at the bottom level. Left and right glasses of the bottom level will be half-empty.

C. Vasya and String

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

High school student Vasya got a string of length n as a birthday present. This string consists of letters '`a`' and '`b`' only. Vasya denotes *beauty* of the string as the maximum length of a **substring** (consecutive subsequence) consisting of equal letters.

Vasya can change no more than k characters of the original string. What is the maximum beauty of the string he can achieve?

Input

The first line of the input contains two integers n and k ($1 \le n \le 100\,000, 0 \le k \le n$) — the length of the string and the maximum number of characters to change.

The second line contains the string, consisting of letters '`a`' and '`b`' only.

Output

Print the only integer — the maximum beauty of the string Vasya can achieve by changing no more than k characters.

Examples

input

```
4 2
abba
```

output

```
4
```

input

```
8 1
aabaabaa
```

output

```
5
```

Note

In the first sample, Vasya can obtain both strings "`aaaa`" and "`bbbb`".

In the second sample, the optimal answer is obtained with the string "`aaaaabaa`" or with the string "`aabaaaaa`".

D. Theseus and labyrinth

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Theseus has just arrived to Crete to fight Minotaur. He found a labyrinth that has a form of a rectangular field of size $n \times m$ and consists of blocks of size 1×1.

Each block of the labyrinth has a button that rotates **all** blocks 90 degrees clockwise. Each block rotates around its center and doesn't change its position in the labyrinth. Also, each block has some number of doors (possibly none). In one minute, Theseus can either push the button in order to rotate all the blocks 90 degrees clockwise or pass to the neighbouring block. Theseus can go from block A to some neighbouring block B only if block A has a door that leads to block B and block B has a door that leads to block A.

Theseus found an entrance to labyrinth and is now located in block (x_T, y_T) — the block in the row x_T and column y_T. Theseus know that the Minotaur is hiding in block (x_M, y_M) and wants to know the minimum number of minutes required to get there.

Theseus is a hero, not a programmer, so he asks you to help him.

Input

The first line of the input contains two integers n and m ($1 \le n, m \le 1000$) — the number of rows and the number of columns in labyrinth, respectively.

Each of the following n lines contains m characters, describing the blocks of the labyrinth. The possible characters are:

- «+» means this block has 4 doors (one door to each neighbouring block);
- «-» means this block has 2 doors — to the left and to the right neighbours;
- «|» means this block has 2 doors — to the top and to the bottom neighbours;
- «^» means this block has 1 door — to the top neighbour;
- «>» means this block has 1 door — to the right neighbour;
- «<» means this block has 1 door — to the left neighbour;
- «v» means this block has 1 door — to the bottom neighbour;
- «L» means this block has 3 doors — to all neighbours except left one;
- «R» means this block has 3 doors — to all neighbours except right one;
- «U» means this block has 3 doors — to all neighbours except top one;
- «D» means this block has 3 doors — to all neighbours except bottom one;
- «*» means this block is a wall and has no doors.

Left, right, top and bottom are defined from representing labyrinth as a table, where rows are numbered from 1 to n from top to bottom and columns are numbered from 1 to m from left to right.

Next line contains two integers — coordinates of the block (x_T, y_T) ($1 \le x_T \le n$, $1 \le y_T \le m$), where Theseus is initially located.

Last line contains two integers — coordinates of the block (x_M, y_M) ($1 \le x_M \le n$, $1 \le y_M \le m$), where Minotaur hides.

It's guaranteed that both the block where Theseus starts and the block where Minotaur is hiding have at least one door. Theseus and Minotaur may be initially located at the same block.

Output

If Theseus is not able to get to Minotaur, then print -1 in the only line of the output. Otherwise, print the minimum number of minutes required to get to the block where Minotaur is hiding.

Examples

input

```
2 2
+*
*U
1 1
2 2
```

output

```
-1
```

input

```
2 3
<><
><>
1 1
2 1
```

output

```
4
```

Note

Assume that Theseus starts at the block (x_T, y_T) at the moment 0.

E. The Last Fight Between Human and AI

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

100 years have passed since the last victory of the man versus computer in Go. Technologies made a huge step forward and robots conquered the Earth! It's time for the final fight between human and robot that will decide the faith of the planet.

The following game was chosen for the fights: initially there is a polynomial

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0,$$

with yet undefined coefficients and the integer k. Players alternate their turns. At each turn, a player pick some index j, such that coefficient a_j that stay near x^j is not determined yet and sets it to **any** value (integer or real, positive or negative, 0 is also allowed). Computer moves first. The human will be declared the winner if and only if the resulting polynomial will be divisible by $Q(x) = x - k$.
Polynomial $P(x)$ is said to be divisible by polynomial $Q(x)$ if there exists a representation $P(x) = B(x)Q(x)$, where $B(x)$ is also some polynomial.

Some moves have been made already and now you wonder, is it true that human can guarantee the victory if he plays optimally?

Input

The first line of the input contains two integers n and k $(1 \le n \le 100\,000, |k| \le 10\,000)$ — the size of the polynomial and the integer k.

The i-th of the following $n + 1$ lines contain character '`?`' if the coefficient near x^{i-1} is yet undefined or the integer value a_i, if the coefficient is already known ($-10\,000 \le a_i \le 10\,000$). Each of integers a_i (and even a_n) may be equal to 0.

Please note, that it's not guaranteed that you are given the position of the game where it's computer's turn to move.

Output

Print "`Yes`" (without quotes) if the human has winning strategy, or "`No`" (without quotes) otherwise.

Examples

input

```
1 2
-1
?
```

output

```
Yes
```

input

```
2 100
-10000
0
1
```

output

```
Yes
```

input

```
4 5
?
1
?
1
?
```

output

```
No
```

Note

In the first sample, computer set a_0 to -1 on the first move, so if human can set coefficient a_1 to 0.5 and win.

In the second sample, all coefficients are already set and the resulting polynomial is divisible by $x - 100$, so the human has won.