A. Increasing Sequence

time limit per test: 1 second
memory limit per test: 64 megabytes
input: standard input
output: standard output

A sequence $a_0, a_1, ..., a_{t-1}$ is called increasing if $a_{i-1} < a_i$ for each i: $0 < i < t$.

You are given a sequence $b_0, b_1, ..., b_{n-1}$ and a positive integer d. In each move you may choose one element of the given sequence and add d to it. What is the least number of moves required to make the given sequence increasing?

Input

The first line of the input contains two integer numbers n and d ($2 \le n \le 2000$, $1 \le d \le 10^6$). The second line contains space separated sequence $b_0, b_1, ..., b_{n-1}$ ($1 \le b_i \le 10^6$).

Output

Output the minimal number of moves needed to make the sequence increasing.

Sample test(s)

input

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

output

```
3
```

B. Jumping Jack

time limit per test: 1 second
memory limit per test: 64 megabytes
input: standard input
output: standard output

Jack is working on his jumping skills recently. Currently he's located at point zero of the number line. He would like to get to the point x. In order to train, he has decided that he'll first jump by only one unit, and each subsequent jump will be exactly one longer than the previous one. He can go either left or right with each jump. He wonders how many jumps he needs to reach x.

Input

The input data consists of only one integer x ($-10^9 \le x \le 10^9$).

Output

Output the minimal number of jumps that Jack requires to reach x.

Sample test(s)

input
2
output
3

input
6
output
3

input
0
output
0

C. How Many Squares?

time limit per test: 2 seconds
memory limit per test: 64 megabytes
input: standard input
output: standard output

You are given a 0-1 rectangular matrix. What is the number of squares in it? A square is a solid square frame (border) with linewidth equal to 1. A square should be at least 2×2. We are only interested in two types of squares:

1. squares with each side parallel to a side of the matrix;
2. squares with each side parallel to a diagonal of the matrix.

```
For example the following matrix contains only one square of the first type:
0000000
0111100
0100100
0100100
0111100
```

```
The following matrix contains only one square of the second type:
0000000
0010000
0101000
0010000
0000000
```

Regardless of type, a square must contain at least one `1` and can't touch (by side or corner) any foreign `1`. Of course, the lengths of the sides of each square should be equal.

How many squares are in the given matrix?

Input

The first line contains integer t ($1 \le t \le 10000$), where t is the number of test cases in the input. Then test cases follow. Each case starts with a line containing integers n and m ($2 \le n, m \le 250$), where n is the number of rows and m is the number of columns. The following n lines contain m characters each (`0` or `1`).

The total number of characters in all test cases doesn't exceed 10^6 for any input file.

Output

You should output exactly t lines, with the answer to the i-th test case on the i-th line.

Sample test(s)

input

```
2
8 8
00010001
00101000
01000100
10000010
01000100
00101000
11010011
11000011
10 10
1111111000
1000001000
1011001000
1011001010
1000001101
1001001010
1010101000
1001001000
1000001000
1111111000
```

output

```
1
2
```

input

```
1
12 11
11111111111
10000000001
10111111101
10100000101
10101100101
10101100101
10100000101
10100000101
10111111101
10000000001
11111111111
00000000000
```

output

```
3
```

D. A Simple Task

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

Given a simple graph, output the number of simple cycles in it. A simple cycle is a cycle with no repeated vertices or edges.

Input

The first line of input contains two integers n and m ($1 \le n \le 19$, $0 \le m$) – respectively the number of vertices and edges of the graph. Each of the subsequent m lines contains two integers a and b, ($1 \le a, b \le n$, $a \ne b$) indicating that vertices a and b are connected by an undirected edge. There is no more than one edge connecting any pair of vertices.

Output

Output the number of cycles in the given graph.

Sample test(s)

input

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

output

```
7
```

Note

The example graph is a clique and contains four cycles of length 3 and three cycles of length 4.

E. Forward, march!

time limit per test: 1 second
memory limit per test: 64 megabytes
input: standard input
output: standard output

Jack has become a soldier now. Unfortunately, he has trouble with the drill. Instead of marching beginning with the left foot and then changing legs with each step, as ordered, he keeps repeating a sequence of steps, in which he sometimes makes the wrong steps or — horror of horrors! — stops for a while. For example, if Jack uses the sequence 'right, left, break', when the sergeant yells: 'Left! Right! Left! Right! Left! Right!', Jack first makes a step with the right foot, then one with the left foot, then he is confused and stops for a moment, then again - this time according to the order - starts with the right foot, then uses the left foot, then - to the sergeant's irritation - he stops to catch his breath, to incorrectly start with the right foot again... Marching this way, Jack will make the step that he is supposed to in the given moment in only one third of cases.

When the officers convinced him he should do something about it, Jack decided to modify the basic sequence of steps that he repeats. However, in order not to get too tired, he has decided that the only thing he'll do is adding any number of breaks in any positions of the original sequence (a break corresponds to stopping for the duration of one step). Of course, Jack can't make a step on the same foot twice in a row, if there is no pause between these steps. It is, however, not impossible that the sequence of steps he used so far is incorrect (it would explain a lot, actually).

Help Private Jack! Given the sequence of steps he keeps repeating, calculate the maximal percentage of time that he can spend marching correctly after adding some breaks to his scheme.

Input

The first line of input contains a sequence consisting only of characters 'L', 'R' and 'X', where 'L' corresponds to a step with the left foot, 'R' — with the right foot, and 'X' — to a break. The length of the sequence will not exceed 10^6.

Output

Output the maximum percentage of time that Jack can spend marching correctly, **rounded down to exactly six digits after the decimal point**.

Sample test(s)

input

```
X
```

output

```
0.000000
```

input

```
LXRR
```

output

```
50.000000
```

Note

In the second example, if we add two breaks to receive LXXRXR, Jack will march: LXXRXRLXXRXRL... instead of LRLRLRLRLRLRL... and will make the correct step in half the cases. If we didn't add any breaks, the sequence would be incorrect — Jack can't step on his right foot twice in a row.