Art Class

Input File: *artin.txt*
Output File: *artout.txt*

Time Limit: 1 second

You are attending a highly exclusive seminar run by the famous artist known only as Leo.

Leo is said to be highly influential in the art world. In fact, he invented all the genres of art. All of them. Impressionism, post-modernism, and even cave paintings. In this seminar, Leo teaches you about his newest style of art, which he calls *post-cubist squarism*.

A post-cubist squarist artwork is painted on a rectangular grid, in which each grid cell is either coloured or blank. Two adjacent coloured cells (vertically or horizontally, not diagonally) are connected, and a *shape* is a group of coloured cells that are directly or indirectly connected to each other.

The *diameter* of a shape is the greatest distance between any two of its cells.

The *distance* between two cells is the minimal number of up, down, left or right steps through the grid to get from one to the other. It doesn't matter whether the steps go through blank cells or even part of an entirely different shape.

In a stroke of genius, Leo realised that the most important aspect of a post-cubist squarist painting is the **largest diameter of any shape**. He announces that there is a reward for whoever can determine this in his newest artwork. Of course, Leo's genius won't help you now. But what about his computer? While no one else is looking, you steal his laptop and begin to code.

Input

Your program should read from the file `artin.txt`. The file will describe a single post-cubist squarist painting.

- The first line of input will contain two space-separated integers H and W, the height and width of the painting.
- The following H lines describe the painting grid. Each of these lines will contain W characters each. Each character will be either '`X`' or '`.`', denoting a coloured or blank grid cell respectively.
- It is guaranteed that there will be at least two adjacent coloured cells.

Output

Your program should write to the file `artout.txt`. Your output file should contain one line with one integer: the largest diameter of any shape in the painting.

Sample Input 1

```
4 9
XXXX.XXXX
..XX.X...
XX.XXX..X
....X..XX
```

Sample Output 1

```
8
```

Explanation

There are three shapes in this example, labelled 1, 2 and 3 as follows:

```
2222.2222
..22.2...
11.222..3
....2..33
```

Shape 2 has the greatest diameter, which is the distance between the top-left square and the top-right square.

Sample Input 2

```
3 6
.X.X..
XX.XX.
XXXX..
```

Sample Output 2

```
5
```

Explanation

In this example, every coloured square is part of the same shape. The greatest distance is between the bottom-left square and the rightmost coloured square.

Constraints

To evaluate your solution, the judges will run your program against several different input files. All of these files will adhere to the following bounds:

- $1 \leq H \leq 1\,000$ (the height of the painting)
- $1 \leq W \leq 1\,000$ (the width of the painting)
- Note that $H \times W$ will be at least 2, as there will be at least one shape with diameter ≥ 1.

As some of the test cases will be quite large, you may need to think about how well your solution scales for larger input values. However, not all the cases will be large. In particular:

- For 40% of the marks, $1 \leq H, W \leq 40$.
- For an additional 30% of the marks, the painting will contain exactly one shape.
- For the remaining 30% of the marks, no special constraints apply.