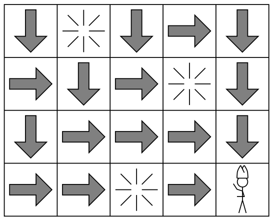
Crowd Surfing

Input File: *surfin.txt*
Output File: *surfout.txt*

Time Limit: 1 second

It is World Youth Day 2008, and you have joined the masses in Sydney flocking to see the Pope. The crowds are enormous, and you decide there is only one way that you will be able to meet His Holiness to obtain an autograph: crowd surfing.

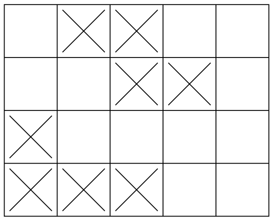
The stadium is a large rectangle with the Pope standing in the south-east corner. Your hope is the crowd will carry you above their heads from where you are standing all the way across to the Pope. Unfortunately the crowd is not very organised—some sections of the crowd will carry you south, some sections will carry you east, and some sections are so disorganised that they will just drop you back onto the ground.



As an example, consider the map above, where the stadium is broken into a 4 × 5 grid of squares. The arrows show the direction in which the crowd will carry you from each square, and the stars indicate squares in which the crowd will drop you.

Suppose you begin in the top left square. The crowd will carry you south, then east, then south, then three squares east, then south again to where the Pope is standing. On the other hand, suppose you begin in the bottom left square. In this case, the crowd will carry you two squares east and then drop you.

Your task is to decide how many starting points are *bad*, i.e., will lead you to being dropped. For the example above, there are eight bad starting points; these are indicated with crosses in the diagram below.



Input

The input will describe the stadium as a grid of squares, and will show how the crowd moves you from each square.

The first line of input will consist of two integers r and c, separated by a single space. These indicate that the stadium is broken into a grid with r rows and c columns ($1 \leq r, c \leq 1\,000$). Following this will be r lines (ordered from north to south), each containing c characters (ordered from west to east), describing the individual squares within the stadium.

Each square of the grid will be described by one of the characters ‘`v`’, ‘`>`’, ‘`*`’, or ‘`+`’:

- The character ‘v’ represents a downward pointing arrow, i.e., the crowd moves you south.
- The character ‘>’ represents a right pointing arrow, i.e., the crowd moves you east.
- The character ‘*’ represents a square in which the crowd drops you.
- The character ‘+’ represents the pope. This will appear as the final character of the final line, and will not appear anywhere else in the grid.

It is guaranteed that no arrow will point outside the grid. That is, no arrows in the rightmost column will point right, and no arrows in the bottom row will point down.

Output

Your output file must consist of a single integer giving the number of bad squares, i.e., the number of squares that lead to you being dropped.

Sample Input 1

```
4 5
v*v>v
>v>*v
v>>>v
>>*>+
```

Sample Output 1

```
8
```

Sample Input 2

```
3 4
>>vv
>>>v
>>*+
```

Sample Output 2

```
3
```

Explanation

The first example input and output files correspond to the example discussed above. In the second example, every square along the bottom row is bad (except for where the Pope is standing), and all other squares are good; therefore the answer is 3.

Scoring

The score for each input scenario will be 100% if the correct answer is written to the output file, and 0% otherwise.