Given Away

locked

Problem Submissions Discussions

Time Limit : C/C++ $(1s)$, $Java$ $(2s)$
Memory Limit : 512 MB

Your friend encountered this challenging problem and found it really annoying. But since you love problem solving so much, he gave it away to you. Let's see if trying it actually gets you any serotonin.

There is a grid with n rows and m columns. Each cell is either empty (denoted by '.') or blocked with a wall (denoted by 'x').

Two empty cells are directly connected if they share a side. Two cells $(r1,\ c1)$ (located in the row $r1$ and column $c1$) and $(r2,\ c2)$ are connected if there exists a sequence of empty cells that starts with $(r1,\ c1)$, finishes with $(r2,\ c2)$, and any two consecutive cells in this sequence are directly connected. A connected component is a set of empty cells such that any two cells in the component are connected, and there is no cell in this set that is connected to some cell not in this set.

You can do **at most one** operation on this grid. Choose a row, column or diagonal line (*see sample explanation*) of the grid and turn every cell in the selected row, column or diagonal line to a wall. The objective is to have as many connected components as possible in the final grid.

Input Format

The first line contains a single integer t — the number of test cases. Then follow t test cases.

The first line of each test contains two integers n and m — the size of the grid.

Each of the next n lines contains a string with m characters, denoting the i-th row of the grid. Each character is '.' or 'x', denoting an empty cell or a blocked one, respectively.

Constraints

$1 \le t \le 10^5$

$1 \le n, m \le 1000$

Sum of $n * m$ over all testcases does not exceed 10^6

Output Format

Print the maximum possible count of connected components in the final resulting grid after using at most one operation.

Sample Input 0

```
4
3 3
...
xxx
...
4 5
x..x.
...x.
..x..
xxx.x
1 1
.
2 3
```

```
...
x..
```

Sample Output 0

```
4
5
1
3
```

Explanation 0

- 1st test case: Using the operation on the 2nd column will result in 4 separate connected components in the 4 corners of the grid.
- 2nd test case: A diagonal operation can make a grid like the following, where the 'o' symbol respresents the new wall blocks created:

```
x.ox.
.o.x.
o.x..
xxx.x
```

- 3rd test case: There is only one cell, so at most 1 connected component is possible.
- 4th test case: A diagonal operation can make a grid like the following, where the 'o' symbol respresents the new wall blocks created:

```
.o.
x.o
```

Submissions: 7
Max Score: 100

Rate This Challenge:
☆☆☆☆☆

More

C

```c
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>

int main() {

    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    return 0;
}
```

Line: 1 Col: 1

Upload Code as File   ☐ Test against custom input   Run Code   Submit Code