UVa Online Judge

1501 Construct the Great Wall

A defensive wall is a fortification used to protect a city or settlement from potential aggressors. From ancient to modern times, they were used to enclose settlements.

Generally, these are referred to as city walls or town walls. Even though, our ancestors decided to build a Great Wall to protect the northern borders of the Chinese Empire against invasion by various nomadic groups.

The map is given as an rectangle area of size $N \times M$. Each grid is an empty area, a city or an enemy. The Great Wall is a simple polygon build alone the edge of the grids, enclosing all the cities and keeping all the enemies out.

The Great Wall is not easy to build, so we should make the Great Wall as short as possible. Now it is your job to calculate the length of the shortest Great Wall so that it can protect all the cities from the enemies.

Input

The first line contains an integer T ($1 \leq T \leq 50$), indicating the number of test cases.

Each test case contains several lines.

The first line contains two integer H, W ($1 \leq H, W \leq 8$), indicating the number of rows and columns of the map.

The following H lines contains W chars, indicating the map. ‘`o`’ represents a city, ‘`.`’ represents a empty area and ‘`x`’ represents an enemy.

You can assume that there will be at least one city on the map.

Output

For each test case in the input, print one line: ‘`Case #`X`:` Y’, where X is the test case number (starting with 1) and Y is the length of the shortest Great Wall (‘`-1`’ if impossible).

Hint: A simple polygon is a closed polygonal chain of line segments in the plane which do not have points in common other than the common vertices of pairs of consecutive segments.

The solution for the first test case in sample is shown in Figure 1.

There is no solution for the second test case because no matter how you build the Great Wall, it will always intersects with itself. (Figure 2).

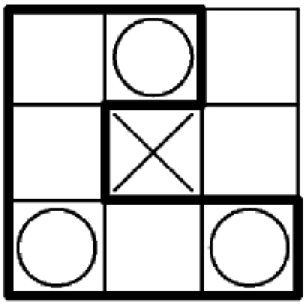

Figure.1

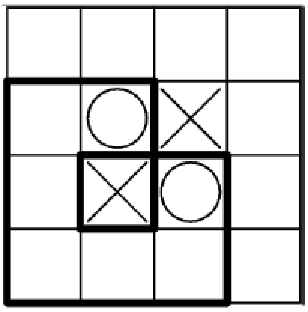

Figure.2

Sample Input

```
3
3 3
.o.
.x.
o.o
4 4
....
.ox.
.xo.
....
5 5
.ooo.
.x...
..xoo
x.xoo
.ox.x
```

Sample Output

```
Case #1: 14
Case #2: -1
Case #3: 28
```