10952   Pattern Transformations

Consider the two patterns of 'O' and 'X' below ('.' represent an empty square). We want the first pattern to be transformed into the second pattern in one time unit. During this time unit, each symbol ('O' and 'X') can move one step in any of the four directions (or remain at its current square). All movements happen simultaneously, so a symbol can move to an occupied square, if that symbol is moved to some other square. If a symbol moves from square A to B, and the symbol at B moves to A, we have a *swap*. Write a program which calculates the least number of swaps needed to transform a given pattern into another given pattern.

```
.XO..    ..XO.
..OX.    .XX..
.XX..    ..OX.
```

    To transform the first pattern above into the second one requires one swap: The two symbols in the first line are moved to the right, the 'O' in the second line must be swapped with the 'X' below. The other two 'X' are moved up and down, respectively.

Input

The first line in the input contains the number of test cases to follow (at most 20). Each test case starts with a line containing two integers, w and h ($1 \le w, h \le 8$), the width and height of the two patterns. Then follow h lines describing each row of the two patterns (the two patterns will be separated with a single space, see the sample input). The only allowed characters in the patterns will be the symbols 'O', 'X' and '.'.

Output

For each input you should output a line containing a single integer: the least number of swaps required to transform the first pattern into the second. If the transformation is not possible, output a '-1'.

Sample Input

```
3
5 3
.XO.. ..XO.
..OX. .XX..
.XX.. ..OX.
4 4
OXOX XOXO
XX.O OX.X
O..X X..O
XOXO OXOX
3 4
.X. .X.
.OX XO.
..O .O.
... ...
```

Sample Output

```
1
0
-1
```