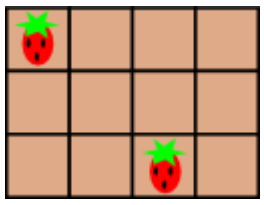
N. Cakeminator

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a rectangular cake, represented as an $r \times c$ grid. Each cell either has an evil strawberry, or is empty. For example, a 3×4 cake may look as follows:



The cakeminator is going to eat the cake! Each time he eats, he chooses a row or a column that does not contain any evil strawberries and contains at least one cake cell that has not been eaten before, and eats all the cake cells there. He may decide to eat any number of times.

Please output the maximum number of cake cells that the cakeminator can eat.

Input

The first line contains two integers r and c ($2 \le r, c \le 10$), denoting the number of rows and the number of columns of the cake. The next r lines each contains c characters — the j-th character of the i-th line denotes the content of the cell at row i and column j, and is either one of these:

- '`.`' character denotes a cake cell with no evil strawberry;
- '`S`' character denotes a cake cell with an evil strawberry.

Output

Output the maximum number of cake cells that the cakeminator can eat.

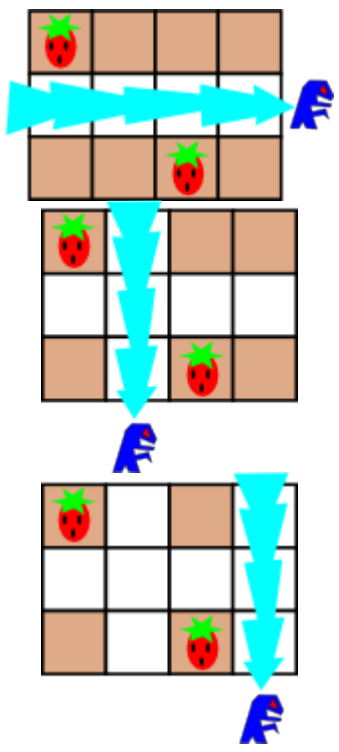
Examples

input	Copy
3 4 S... .... ..S.	

output	Copy
8	

Note

For the first example, one possible way to eat the maximum number of cake cells is as follows (perform 3 eats).



→ **Attention**

The package for this problem was not updated by the problem writer or Codeforces administration after we've upgraded the judging servers. To adjust the time limit constraint, a solution execution time will be multiplied by 2. For example, if your solution works for 400 ms on judging servers, then the value 800 ms will be displayed and used to determine the verdict.

Assiut University Training - Newcomers

Public

Participant

→ **About Group**

ICPC Assiut community

Group website

→ **Group Contests**

- Sheet #10 (General Hard)
- Sheet #9 (General medium)
- Sheet #8 (General easy)
- Sheet #7 (Recursion)
- Sheet #6 (Math - Geometry)
- Sheet #5 (Functions)
- Sheet #4 (Strings)
- Contest #3.1
- Sheet #3 (Arrays)
- Contest #2
- Sheet #2 (Loops)
- Contest #1
- Sheet #1 (Data type - Conditions)

Sheet #8 (General easy)

Finished