Teleporting Tater Tech

Input File: *standard input*
Output File: *standard output*

Time limit	Memory limit
1 second	256 MB

Statement

Farmer John has ventured to a galaxy far, far away to find a mythical $N \times M$ grid covered in peculiar potato and teleporter squares. John has deployed a specially designed *Tatertech*^{TM} robot in the top-left square of this grid to collect potatoes for his upcoming *Potato Pyramid Project*. However, the robot was designed to collect potatoes and withstand the quantum side-effects of teleportation, not to be intelligent. As such, the robot may only move a single square down or to the right at a time and it does so in a uniformly random manner.

If the robot passes over a potato square, it will collect a single potato. If the robot passes over a teleporter square, it will teleport to a uniformly random position on the grid. It could even teleport back to the same square and teleport again! All other squares are empty and the robot may pass over them with no effect.

Due to the behaviour of teleporters, the robot may pass over the same square multiple times. The effect of the square will be triggered each time the robot passes over or teleports onto it.

The bottom-right square is never a teleporter square. Therefore, the robot will eventually become stuck there. Farmer John is a cautious man. He wants to check if he will likely collect sufficient potatoes for his plans. What is the expected number of potatoes the robot will collect before it becomes stuck?

Input

The first line of input contains the integer dimensions of the grid N and M. N lines follow, each containing a string of M characters. A 'P' indicates a potato square. A 'T' indicates a teleporter square. A '.' indicates an empty square.

Output

Output a single real number: the expected number of potatoes the robot will collect. Your output will be accepted if it is within 10^{-5} of the judge's output.

Sample Input 1

```
2 2
..
PP
```

Sample Output 1

```
1.5
```

Sample Input 2

```
2 2
TP
PP
```

Sample Output 2

```
1.666667
```

Sample Input 3

```
1 4
PT.P
```

Sample Output 3

```
2.5
```

Explanation

For Sample 1, the Tatertech robot has two possible paths. If it moves right then down, it collects a single potato. If it moves down then right, it collects 2 potatoes. On average, the robot collects 1.5 potatoes.

For Sample 2, when the robot is deployed in the top-left square it immediately teleports. If it teleports back to square (1,1), it will only teleport again. Therefore there are only 3 unique possibilities each with a $\frac{1}{3}$ probability of occurring. If it lands in square (1,2) or (2,1) it will collect 2 potatoes. If it lands on square (2,2) it will only collect a single potato. Therefore the expected number of potatoes is $\frac{1}{3}(2+2+1) = \frac{5}{3} \approx 1.666667$. Note: any value between 1.666657 and 1.666677 would also be accepted.

For Sample 3, the robot always collects at least 2 potatoes as it starts and ends on a potato square. However, when the robot teleports, it has a $\frac{1}{3}$ chance of collecting an additional potato and teleporting again. This means the expected number of potatoes is $2 + \frac{1}{3}(1 + \frac{1}{3}(1 + \frac{1}{3}(1 + ...)))$ which simplifies to 2.5.

Constraints

- $1 \leq N, M \leq 1000$
- The bottom-right square will not be a teleporter square.

Subtasks

- For Subtask 1 (25 points), There are no teleporter squares.
- For Subtask 2 (35 points), $N = 1$ and there is exactly 1 teleporter square.
- For Subtask 3 (40 points), No additional constraints apply.

Note: To output your result to the correct number of decimal places, you may wish to use `cout << fixed << setprecision(6) << answer << "\n";` **or** `printf("%.6f\n" answer);`