Air Traffic Control (`atc`)

Edoardo has recently been hired by the municipality of Milan as an Air Traffic Control operator. For his first week, he has been assigned by his boss to tackle the *paper-plane project*.

The paper-plane project is an effort to connect the city of Milan with a network of paper planes. To do this, Edoardo should build a number of control towers.

We can assume the city is represented by a square grid of size $N \times N$, where each cell can be either:

- Empty space, indicated with a "`.`" in input.

- An obstacle such as a mountain or building or other obstructing item, indicated with a "`#`" in input.

- A control tower, indicated with a "`T`" in input.

A control tower placed on cell (i, j) can throw a paper plane in four directions: up, down, left, right. The paper plane will travel in that direction until either another control tower or an obstacle is encountered. If a paper plane reaches a control tower, then it can be launched again from there.

The goal of this project is to connect the $(1, 1)$ tower to the (N, N) tower, which will always be present, so that it's possible to exchange paper planes between them. Help Edoardo compute the *minimum* number of towers he should construct in order to reach the goal. If such task is impossible regardless the number of additional towers built, then the whole paper-plane project is impossible.

> ☞ Among the attachments of this task you may find a template file `atc.*` with a sample incomplete implementation.

Input

The first line contains an integer N, the size of the grid. The next N lines contain N characters each. The first character of the first line indicates the position $(1, 1)$ of the grid. The last character of the last line indicates the position (N, N).

Output

You need to write a single line with an integer: the minimum number of towers that should be built to reach the goal. If it is impossible to reach the goal, write `-1`.

Constraints

- $2 \leq N \leq 200$.
- It is guaranteed that coordinates $(1, 1)$ and (N, N) always contain control towers.

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

– **Subtask 1** (0 points)   Examples.

– **Subtask 2** (10 points) $N = 2$.

– **Subtask 3** (20 points) $N = 3$.

– **Subtask 4** (50 points) $N \leq 6$ and no more than 4 additional control towers are needed.

– **Subtask 5** (20 points)   No additional limitations.

Examples

input	output
5 T.... ....# ..#.. #.... ....T	2
5 T...# ...#. ..#.. .#... #...T	-1

Explanation

In the **first sample case**, it is enough to add 2 towers. For example, a possible solution is to add the towers highlighted in red:

```
T..T.
....#
..#..
#....
...TT
```

In the **second sample case** it is impossible to connect towers at $(1, 1)$ and (N, N).