Computer Game

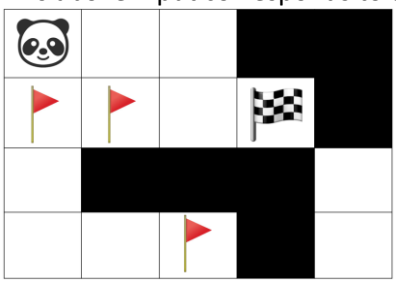
Mr. Panda is playing a computer game where his character is in a room that can be described as a 2D grid with **R** rows and **C** columns. Each cell in the grid represents exactly one of the following

- The start point of Mr. Panda's character, denoted by 'S' in the input.
- The goal point of Mr. Panda's character, denoted by 'T' in the input.
- A checkpoint, denoted by 'C' in the input.
- A wall, denoted by '#' in the input. (Mr. Panda's character cannot pass through walls.)
- Empty cell, denoted by '.' in the input. (Mr. Panda's character can pass through empty cells.)

One possible example of an input is the following:

```
S..##
CC.T#
.###.
..C#.
```

This above input corresponds to the following layout:



The objective of the game is for Mr. Panda's character to get from the start point to the goal point in the **shortest number of moves possible**, while **passing through every checkpoint at least once**. In each move, Mr. Panda's character can move exactly one cell to the left, up, right or down. However, Mr. Panda's character **is not allowed to move into a cell that has a wall,** but he can **pass through every non-wall cell** as many times as he wants. Given these constraints, can you help Mr. Panda calculate what is the shortest number of moves it can use to complete the game?

Hint: Consider all possible orders of visiting the checkpoints (**permutations**). For each order, Mr. Panda will move between 2 consecutive checkpoints using the shortest number of moves possible.

Input

The first line of input contains 2 integers **R** and **C** where **R** is the number of rows and **C** is the number of columns in the grid. The next **R** rows each contain **C** characters describing the layout of the room.

Output

The minimum number of moves Mr. Panda's character needs to complete the game.

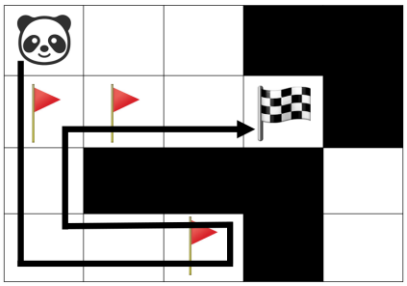
Limits

- $1 \leq \mathbf{R}, \mathbf{C} \leq 200$
- There will be **at least 1** and **at most 8** checkpoints in the room, exclusive of the start and goal point.
- It is guaranteed that it is possible to reach all the checkpoints and the goal point from the start point.

Sample Input (**computergame1.in**)	Sample Output (**computergame1.out**)
4 5 S..## CC.T# .###. ..C#.	12

Explanation

The shortest number of moves needed is 12 as shown in the picture below. The sequence of moves is down, down, down, right, right, left, left, up, up, right, right, right.



Notes:

1. You should develop your program in the subdirectory **ex4** and use the skeleton java file provided. You should not create a new file or rename the file provided.
2. You are free to define your own helper methods and classes (or remove existing ones).
3. Please be reminded that the marking scheme is:
   a. Public Test Cases (1%) - 1% for passing **all** test cases, 0% otherwise
   b. Hidden Test Cases (1%) - Partial scoring depending on test cases passed
   c. Manual Grading (1%)
      i. Overall Correctness (correctness of algorithm, severity of bugs)
      ii. Coding Style (meaningful comments, modularity, proper indentation, meaningful method and variable names)
4. Your program will be tested with a time limit of not less than **2 sec** on Codecrunch.

Skeleton File – Computergame.java

You are given the below skeleton file `Computergame.java`. You should see a non-empty file when you open the skeleton file. Otherwise, you might be in the wrong working directory.

```
/**
 * Name         :
 * Matric. No   :
 * PLab Acct.   :
 */
import java.util.*;
public class Computergame {
    private void run() {
        //implement your "main" method here
    }
    public static void main(String[] args) {
        Computergame newComputergame = new Computergame();
        newComputergame.run();
    }
}
```