Portals

There is a cake placed inside of a labyrinth and you desperately want to eat it. You have a map of the labyrinth, which is a grid of R rows and C columns. Each grid cell contains one of the following characters:

- `#` (number sign) which denotes a wall,
- `.` (dot) which denotes an open square,
- `S` (uppercase letter s) which denotes your current position,
- `C` (uppercase letter c) which denotes the position of the cake.

You may only walk on the open squares. Additionally, the rectangular area depicted on the map is surrounded by walls from the outside with no open spaces.

In order to reach the cake faster you have acquired a portal gun from Aperture Science™, which operates as follows. At any time it can fire a portal in one of the four directions *up*, *left*, *down* and *right*. When a portal is fired in some direction, it will fly in that direction until it reaches the first wall. When this happens, a portal will be spawned on the side of that wall that faces you.

At most two portals can exist at any given time. If two portals are already placed in the labyrinth, then one of them (selected by you) will be removed immediately upon using the portal gun again. Firing a portal at a wall where there is already a portal placed will replace that portal (there may be at most one portal per side of wall). Note that there may be multiple portals placed on different sides of same wall.

Did I get this note right?

Once two portals are placed on the map you can use them to teleport yourself. When standing next to one of the portals, you can walk into it and end up at the square next to the other portal. Doing this takes as much time as moving between two adjacent squares.

You may assume that firing portals does not take time and moving between two squares (or teleporting through portals) takes one unit of time.

Task

Given the map of the labyrinth together with your starting location and the location of the cake, calculate the minimum possible time needed for you to reach the cake.

Implementation

You need to implement the function `get_cake(R, C, M)` which takes the following parameters:

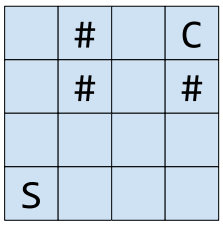
- `R` — the number of rows in the map R,
- `C` — the number of columns in the map C,
- `M` — a two–dimensional array, where `M`$[i][j]$ ($0 \leq i < R$, $0 \leq j < C$) is the cell in the i–th row and j–th column of the map.

It is guaranteed that characters `S` and `C` each appear exactly once in the map, and that it is always possible to reach the cake from your starting location.

The function has to return the minimum time that is needed to reach the cake from the starting position.

Example

Let us consider the example illustrated on the right. One quickest sequence of moves is as follows:



1. move right
2. move right, shoot one portal up, and one portal down
3. move through the bottom portal — you will appear at the location $row = 0, column = 2$
4. move one square right and reach the cake.

This takes time 4 and no quicker way is possible. Therefore function `get_cake` should return `4`.

Scoring

Subtask 1 (25 points): $0 \leq R \leq 2, 0 \leq C \leq 2$.

Subtask 2 (25 points): $0 \leq R \leq 10, 0 \leq C \leq 10$.

Subtask 3 (25 points): $0 \leq R \leq 100, 0 \leq C \leq 100$.

Subtask 4 (25 points): $0 \leq R \leq 1000, 0 \leq C \leq 1000$.

Constraints

Time limit: 1 s.

Memory limit: 256 MB.

Need to add info about graders.