[CS 11] Prac 10l – Only a Flesh Wound

Sub[CS 11] Practice 10 ❤

✓ **Points:** 150 (partial)

Time limit: 12.0s

Memory limit: 1G

Author: kvatienza (Kevin Atienza)

Problem type

Allowed languages
~~NONE~~, py3

Problem Statement

On their way to the Bridge of Death, King Arthur and his knights discovered that the land is filled with traps. One wrong step and the adventure is over!

We can model the land ahead as an $r \times c$ grid. Each cell is either *safe* or *trapped*. Luckily, the traps are not well hidden, so King Arthur and his knights know exactly which cells are trapped.

In this grid, they start at the bottom-left cell and need to go to the top-right cell. (These two cells are both safe.) Luckily, because the group has a king and some knights, there are several ways they can move across the grid:

- Move one cell up.
- Move one cell right.
- Move one cell up and two cells right. (The intermediate cells are skipped over.)
- Move one cell right and two cells up. (The intermediate cells are skipped over.)
- Move one cell down and two cells right. (The intermediate cells are skipped over.)
- Move one cell left and two cells up. (The intermediate cells are skipped over.)

A move is only valid if it doesn't land them outside the grid or onto a trapped cell.

How many distinct ways are there to go from the bottom-left cell to the top-right cell?

Task Details

Your task is to implement a function called `num_ways`. This function has a single parameter, a `tuple`/`list` of r `str`s representing the grid. Each element consists of c characters represents a row. Each character is either:

- `.` for a safe cell;
- `X` fro a trapped cell.

The function must return an `int` denoting the number of paths from the bottom-left cell and the top-right cell.

Restrictions

(See 10a for more restrictions)

For this problem in particular:

- The following symbols are allowed: `map`, `filter`.
- The following import is allowed: `cache` and `lru_cache` from `functools`.
- The source code limit is 1000.
- At most 2 other functions may be defined, aside from the required one.
- You may only nest up to 3 loops or comprehensions anywhere in the code.

Example Calls

Example 1 Function Call

```
num_ways([
    '...',
    '.X.',
    '...',
])
```
Copy

Example 1 Return Value

```
12
```
Copy

Example 2 Function Call

```
num_ways((
    '.....',
    '.....',
    '.XXX.',
    '.....',
    '...X.',
))
```
Copy

Example 2 Return Value

```
392
```
Copy

Example 3 Function Call

```
num_ways((
    '.......',
    '.......',
    '.......',
    '.......',
    '.......',
    '.......',
    '.......',
))
```
Copy

Example 3 Return Value

```
296256
```
Copy

Example 4 Function Call

```
num_ways((
    '.....X.',
    '....X.X',
    '.....X.',
    '.......',
    '.......',
    '.......',
    '.......',
))
```
Copy

Example 4 Return Value

```
0
```
Copy

Constraints

- The function `num_ways` will be called at most 2 times.
- $1 \leq r, c \leq 60$
- The top-right and bottom-left cells are safe.

Scoring

- You get 90 ❤ points if you solve all test cases where:
  - $r, c \leq 7$
- You get 60 ❤ points if you solve all test cases.

Clarifications

Report an issue

No clarifications have been made at this time.