2147. Number of Ways to Divide a Long Corridor

Solved ✓

Hard | Topics | Companies | Hint

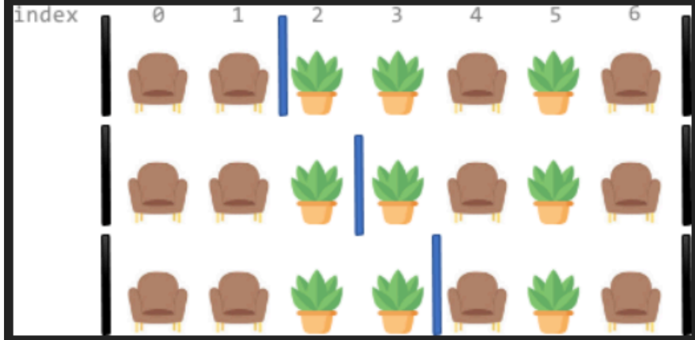
Along a long library corridor, there is a line of seats and decorative plants. You are given a **0-indexed** string `corridor` of length `n` consisting of letters `'S'` and `'P'` where each `'S'` represents a seat and each `'P'` represents a plant.

One room divider has **already** been installed to the left of index `0`, and **another** to the right of index `n - 1`. Additional room dividers can be installed. For each position between indices `i - 1` and `i` (`1 <= i <= n - 1`), at most one divider can be installed.

Divide the corridor into non-overlapping sections, where each section has **exactly two seats** with any number of plants. There may be multiple ways to perform the division. Two ways are **different** if there is a position with a room divider installed in the first way but not in the second way.

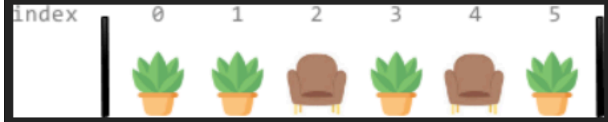
Return *the number of ways to divide the corridor*. Since the answer may be very large, return it **modulo** $10^9 + 7$. If there is no way, return `0`.

Example 1:

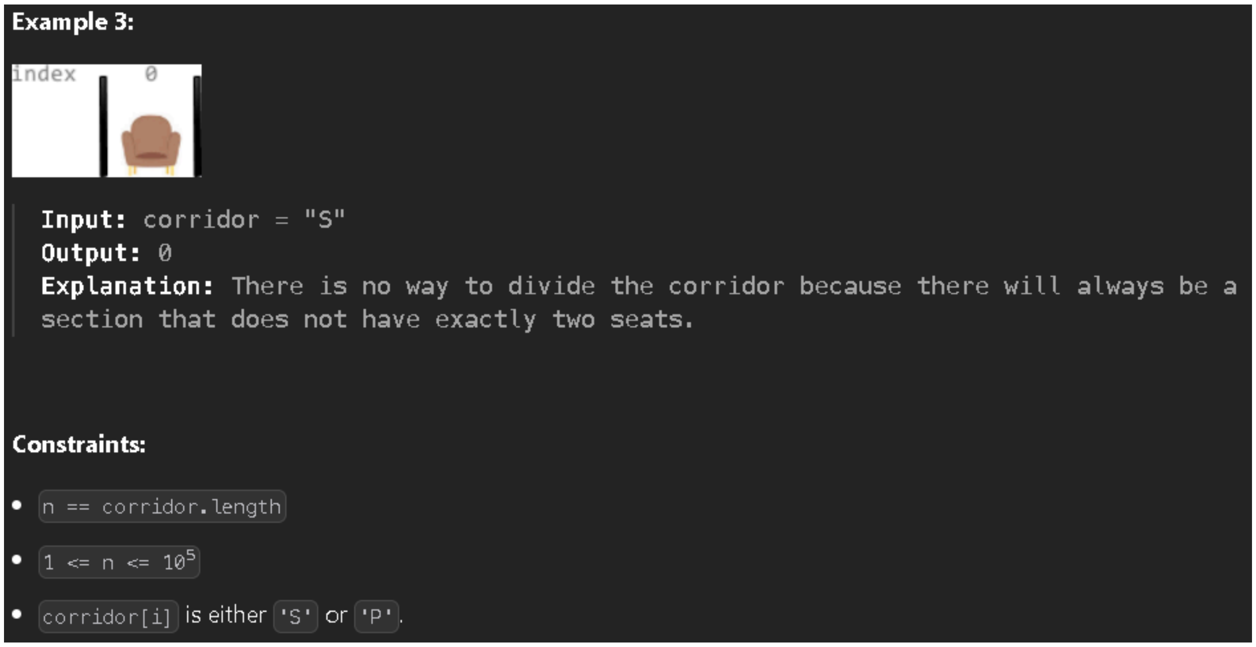


```
Input: corridor = "SSPPSPS"
Output: 3
Explanation: There are 3 different ways to divide the corridor.
The black bars in the above image indicate the two room dividers already installed.
Note that in each of the ways, each section has exactly two seats.
```

Example 2:



```
Input: corridor = "PPSPSP"
Output: 1
Explanation: There is only 1 way to divide the corridor, by not installing any
additional dividers.
Installing any would create some section that does not have exactly two seats.
```



Example 3:

> **Input:** corridor = "S"
> **Output:** 0
> **Explanation:** There is no way to divide the corridor because there will always be a section that does not have exactly two seats.

Constraints:

- `n == corridor.length`
- `1 <= n <= `10^5
- `corridor[i]` is either `'S'` or `'P'`.

Python:

```
class Solution:
    def numberOfWays(self, s):
        a = [i for i,c in enumerate(s) if c == 'S']
        res = 1
        for i in range(1,len(a) - 1,2):
            res *= a[i+1] - a[i]
        return res % (10**9+7) * (len(a) % 2 == 0 and len(a) >= 2)
```

JavaScript:

```
/**
 * @param {string} corridor
 * @return {number}
 */
var numberOfWays = function(corridor) {
    const mod = 1000000007;
    let res = 1;
    let prevSeat = 0;
    let numSeats = 0;

    for (let i = 0; i < corridor.length; i++) {
        const c = corridor.charAt(i);
        if (c === 'S') {
            numSeats += 1;
            if (numSeats > 2 && numSeats % 2 === 1) {
                res = res * (i - prevSeat) % mod;
            }
            prevSeat = i;
        }
    }

    return numSeats > 1 && numSeats % 2 === 0 ? res : 0;
};
```

Java:

```
class Solution {
    public int numberOfWays(String s) {
        long res = 1, j = 0, k = 0, mod = (long)1e9 + 7;
        for (int i = 0; i < s.length(); ++i) {
            if (s.charAt(i) == 'S') {
                if (++k > 2 && k % 2 == 1)
                    res = res * (i - j) % mod;
                j = i;
            }
        }
        return k % 2 == 0 && k > 0 ? (int)res : 0;
    }
}
```