Problem 2267: Check if There Is a Valid Parentheses String Path

Problem Information

Difficulty: Hard
Acceptance Rate: 39.87%
Paid Only: No
Tags: Array, Dynamic Programming, Matrix

Problem Description

A parentheses string is a **non-empty** string consisting only of `'('` and `')'`. It is **valid** if **any** of the following conditions is **true** :

* It is `()`. * It can be written as `AB` (`A` concatenated with `B`), where `A` and `B` are valid parentheses strings. * It can be written as `(A)`, where `A` is a valid parentheses string.

You are given an `m x n` matrix of parentheses `grid`. A **valid parentheses string path** in the grid is a path satisfying **all** of the following conditions:

* The path starts from the upper left cell `(0, 0)`. * The path ends at the bottom-right cell `(m - 1, n - 1)`. * The path only ever moves **down** or **right**. * The resulting parentheses string formed by the path is **valid**.

Return `true` _if there exists a**valid parentheses string path** in the grid._ Otherwise, return `false`.

Example 1:

![](https://assets.leetcode.com/uploads/2022/03/15/example1drawio.png)

Input: grid = [["(","(","("],[")","(",")"],["(","(",")"],["(","(",")"]] **Output:** true **Explanation:** The above diagram shows two possible paths that form valid parentheses strings. The first path shown results in the valid parentheses string "()(())". The second path shown results in the valid parentheses string "((()))". Note that there may be other valid parentheses string paths.

Example 2:

![](https://assets.leetcode.com/uploads/2022/03/15/example2drawio.png)

Input: grid = [[")",")"],["(","("]] **Output:** false **Explanation:** The two possible paths form the parentheses strings "))(" and ")((". Since neither of them are valid parentheses strings, we return false.

Constraints:

* `m == grid.length` * `n == grid[i].length` * `1 <= m, n <= 100` * `grid[i][j]` is either `'('` or `')'`.

Code Snippets

C++:

```
class Solution {
public:
bool hasValidPath(vector<vector<char>>& grid) {

}
};
```

Java:

```
class Solution {
public boolean hasValidPath(char[][] grid) {

}
}
```

Python3:

```
class Solution:
def hasValidPath(self, grid: List[List[str]]) -> bool:
```