0200. 岛屿数量

ITCharge 大约 2 分钟

- 标签：深度优先搜索、广度优先搜索、并查集、数组、矩阵
- 难度：中等

题目链接

- 0200. 岛屿数量 - 力扣

题目大意

描述：给定一个由字符 `'1'`（陆地）和字符 `'0'`（水）组成的的二维网格 $grid$。

要求：计算网格中岛屿的数量。

说明：

- 岛屿总是被水包围，并且每座岛屿 由水平方向和/或竖直方向上相邻的陆地连接形成。
- 此外，你可以假设该网格的四条边均被水包围。
- $m == grid.length$。
- $n == grid[i].length$。
- $1 \leq m, n \leq 300$。
- $grid[i][j]$ 的值为 `'0'` 或 `'1'`。

示例：

- 示例 1：

```py
输入：grid = [
  ["1","1","1","1","0"],
  ["1","1","0","1","0"],
  ["1","1","0","0","0"],
  ["0","0","0","0","0"]
]
输出：1
```

- 示例 2：

```py
输入：grid = [
  ["1","1","0","0","0"],
  ["1","1","0","0","0"],
  ["0","0","1","0","0"],
  ["0","0","0","1","1"]
]
输出：3
```

解题思路

如果把上下左右相邻的字符 `'1'` 看做是 `1` 个连通块，这道题的目的就是求解一共有多少个连通块。

使用深度优先搜索或者广度优先搜索都可以。

思路 1：深度优先搜索

1. 遍历 $grid$。
2. 对于每一个字符为 `'1'` 的元素，遍历其上下左右四个方向，并将该字符置为 `'0'`，保证下次不会被重复遍历。
3. 如果超出边界，则返回 0。
4. 对于 (i, j) 位置的元素来说，递归遍历的位置就是 $(i - 1, j)$、$(i, j - 1)$、$(i + 1, j)$、$(i, j + 1)$ 四个方向。每次遍历到底，统计数记录一次。
5. 最终统计出深度优先搜索的次数就是我们要求的岛屿数量。

思路 1：代码

```py
class Solution:
    def dfs(self, grid, i, j):
        n = len(grid)
        m = len(grid[0])
        if i < 0 or i >= n or j < 0 or j >= m or grid[i][j] == '0':
            return 0
        grid[i][j] = '0'
        self.dfs(grid, i + 1, j)
        self.dfs(grid, i, j + 1)
```

```
        self.dfs(grid, i - 1, j)
        self.dfs(grid, i, j - 1)

    def numIslands(self, grid: List[List[str]]) -> int:
        count = 0
        for i in range(len(grid)):
            for j in range(len(grid[0])):
                if grid[i][j] == '1':
                    self.dfs(grid, i, j)
                    count += 1
        return count
```

思路 1：复杂度分析

- **时间复杂度**：$O(m \times n)$。其中 m 和 n 分别为行数和列数。
- **空间复杂度**：$O(m \times n)$。