695. Max Area of Island

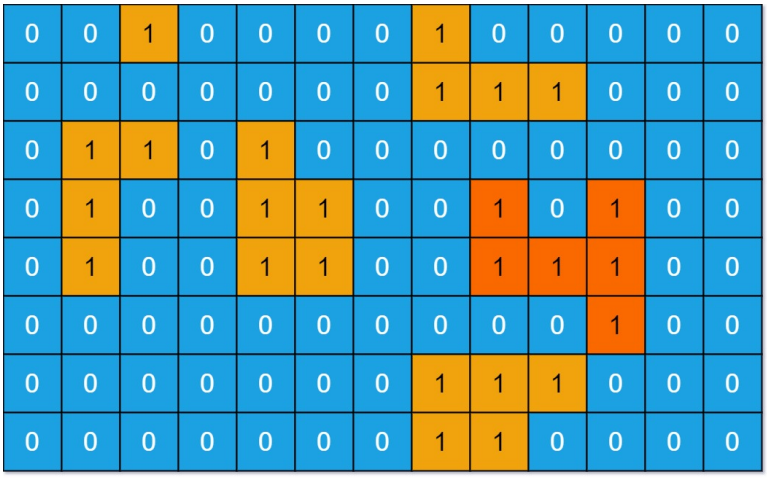
Created	@November 29, 2022 4:21 PM
Difficulty	Medium
LC Url	https://leetcode.com/problems/max-area-of-island/
Importance	
Tag	DFS Island
Reference	https://labuladong.github.io/algo/4/31/107/

You are given an `m x n` binary matrix `grid`. An island is a group of `1`'s (representing land) connected **4-directionally** (horizontal or vertical.) You may assume all four edges of the grid are surrounded by water.

The **area** of an island is the number of cells with a value `1` in the island.

Return *the maximum **area** of an island in* `grid`. If there is no island, return `0`.

Example 1:




```
Input: grid = [[0,0,1,0,0,0,0,1,0,0,0,0,0],[0,0,0,0,0,0,0,1,1,1,0,0,0],[0,1,1,0,1,0,0,0,0,0,0,0,0],[0,1,0,0,1,1,0,0,1,0,1,0,0],[0,1,0,0,1,1,0,0,1,1,1,0,0],[0,0,0,0,0,0,0,0,0,0,1,0,0],[0,0,0,0,0,0,0,1,1,1,0,0,0],[0,0,0,0,0,0,0,1,1,0,0,0,0]]
Output: 6
Explanation: The answer is not 11, because the island must be connected 4-directionally.
```

Example 2:

```
Input: grid = [[0,0,0,0,0,0,0,0]]
Output: 0
```

Constraints:

- `m == grid.length`
- `n == grid[i].length`
- `1 <= m, n <= 50`
- `grid[i][j]` is either `0` or `1`.

Solution

这题的大体思路和之前完全一样，只不过 `dfs` 函数淹没岛屿的同时，还应该想办法记录这个岛屿的面积。

我们可以给 `dfs` 函数设置返回值，记录每次淹没的陆地的个数，

```
class Solution:
    directions = [(1, 0), (-1, 0), (0, 1), (0, -1)]

    def maxAreaOfIsland(self, grid: List[List[int]]) -> int:
        # 记录岛屿的最大面积
        res = 0
        m = len(grid)
        if m == 0:
            return res
        n = len(grid[0])

        for i in range(m):
            for j in range(n):
                if grid[i][j] == 1:
                    # 淹没岛屿，并更新最大岛屿面积
                    res = max(res, self.dfs(grid, i, j))

        return res

    def is_valid(self, grid, i, j):
        """
        判断是否超出索引边界
```

```
        """
        m, n = len(grid), len(grid[0])
        if 0 <= i < m and 0 <= j < n:
            return True
        return False

    def dfs(self, grid, i, j):
        """
        淹没与 (i, j) 相邻的陆地，并返回淹没的陆地面积
        """
        if not self.is_valid(grid, i, j):
            # 超出边界
            return 0

        if grid[i][j] == 0:
            # 已经是海水了
            return 0

        # 将 (i, j) 变成海水
        grid[i][j] = 0

        cnt = 1
        for direction in self.directions:
            cur_i, cur_j = i + direction[0], j + direction[1]
            cnt += self.dfs(grid, cur_i, cur_j)

        return cnt
```