695. Max Area of Island

最大岛屿的面积

Medium 3731 114 Add to List Share

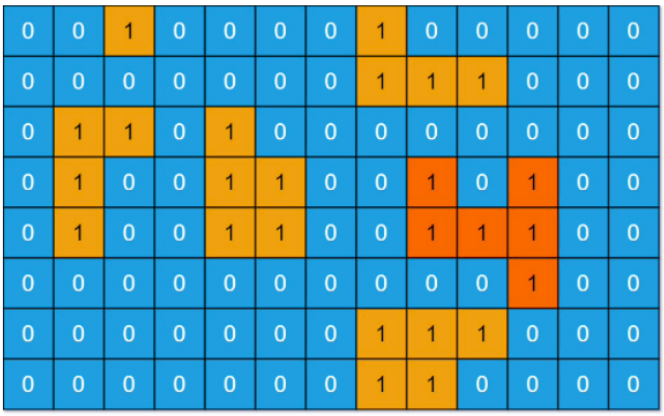
You are given an `m x n` binary matrix `grid`. An island is a group of `1`'s (representing land) connected **4-directionally** (horizontal or vertical.) You may assume all four edges of the grid are surrounded by water.

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`.

Medium

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
```

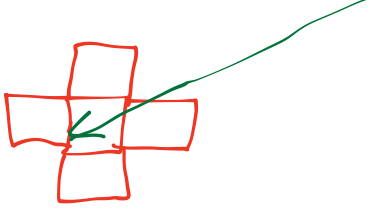
A. Use DFS to expand spot in ↑ ← → ↓ 4 directions

dfs( ) is performed on a single spot

```
def dfs(self,row:int,col:int,visited:set,grid: List[List[int]]) -> int:
        if not (0<= row < len(grid) and 0<=col < len(grid[0]) and
                (row,col) not in visited and
                grid[row][col] ):

                # make sure the pos of this spot grid[row][col] is inside and havn't beed visited before and it's value is 1
            return 0
        visited.add((row,col)) # log every visited spot
        return (1+ self.dfs(row+1,col,visited,grid) +self.dfs(row,col+1,visited,grid)
                + self.dfs(row-1,col,visited,grid) + self.dfs(row,col-1,visited,grid)  )
    # if its a valid spot([1]) , expand towards 4 directions and add upon all 1's
    # then we get the area of any island related to this spot
```

if valid, Area +1



∴ Dfs has the ability to generate the Area of the island expanded from any spot.

To get the Max Area

- perform dfs on each spot

⇓

```
def maxAreaOfIsland(self, grid: List[List[int]]) -> int:
    # iterate all sopts in the grid
    visited = set()
    [redacted]
    return max( self.dfs(r,c,visited,grid)
                for r in range(len(grid))
                for c in range(len(grid[0])))
```

Or

```
areas=[]
for r in
    for c in
        areas.append( self.dfs(r,c,visited,grid))
return max(areas)
```