200. Number of Islands

Given an m x n 2D binary grid grid which represents a map of '1's (land) and '0's (water), return *the number of islands*.

An **island** is surrounded by water and is formed by connecting adjacent lands horizontally or vertically. You may assume all four edges of the grid are all surrounded by water.

Example 1:

Input: grid = [

["1","1","1","1","0"],

["1","1","0","1","0"],

["1","1","0","0","0"],

["0","0","0","0","0"]

]

Output: 1

Example 2:

Input: grid = [

["1","1","0","0","0"],

["1","1","0","0","0"],

["0","0","1","0","0"],

["0","0","0","1","1"]

]

Output: 3

Constraints:

- m == grid.length

- n == grid[i].length
- 1 <= m, n <= 300
- grid[i][j] is '0' or '1'.

```
class Solution {
public:
    int numIslands(vector<vector<char>>& grid) {
        if(grid.empty() || grid[0].empty())
            return 0;

        int numIslands = 0;
        for(int i=0;i<grid.size();i++){
            for(int j=0;j<grid[0].size();j++){
                if(grid[i][j] == '1'){
                    numIslands++;
                    dfs(grid,i,j);
                }
            }
        }
        return numIslands;
    }

    void dfs(vector<vector<char>>& grid, int i, int j){
        if(i<0 || i>=grid.size() || j<0 || j>=grid[0].size()
|| grid[i][j] != '1'){
            return;
        }
        grid[i][j] = '0';
        dfs(grid, i+1, j);
        dfs(grid, i-1, j);
        dfs(grid, i, j+1);
        dfs(grid, i ,j-1);
    }
};
```