# 200. Number of Islands

## 题目描述：https://leetcode.com/problems/number-of-islands/

> 用1代表陆地，0代表大海，求一块区域中的岛屿数，岛屿是由连续的陆地组成的。
> 例如：

```
Example 1:
11110
11010
11000
00000
Answer: 1
Example 2:
11000
11000
00100
00011
Answer: 3
```

## 解题思路：

1. DFS把一个岛屿都标记，然后找下一个。
2. UnionFind
3. ###代码DFS： class Solution { public: void DFS(vector> & grid, vector> &visited, int i , int j) { if(i < 0 || i >= grid.size() || j < 0 || j >= grid[i].size()) { return; } if(visited[i][j] != 0 || grid[i][j] != '1') { return; } visited[i][j] = 1; DFS(grid, visited, i-1, j); DFS(grid, visited, i, j-1); DFS(grid, visited, i+1, j); DFS(grid, visited, i, j+1); } int numIslands(vector>& grid) { int m = grid.size(); if(m == 0) return 0; int n = grid[0].size(); if(n == 0) return n; int c = 0; vector > visited(m, vector (n, 0)); for(int i = 0; i < m; i++) { for(int j = 0; j < n; j++) { if(visited[i][j] == 0 && grid[i][j] == '1') { c++; DFS(grid, visited, i, j); } } } return c; } }; ###代码Union Find: class Solution { public: vector p; int count; void makeset(vector > grid) { count = 0; int m = grid.size(); if(m == 0) return ; int n = grid[0].size(); p.resize(m*n, 0); for(int i = 0; i < m; i++) { for(int j = 0; j < n; j++) { if(grid[i][j] == '1') count++; p[i*n+j] = i*n+j; } } return; } void MyUnion(int i, int j) { int pi = findFather(i); int pj = findFather(j); if(pi == pj) { return; } p[pi] = pj; count--; } int findFather(int i) { if(i == p[i]) return i; int t = findFather(p[i]); p[i] = t; return t; } int numIslands(vector>& grid) { int m = grid.size(); if(m == 0) return 0; int n = grid[0].size(); makeset(grid); for(int i = 0; i < m; i++) { for(int j = 0; j < n; j++) { if(grid[i][j] == '0') continue; int a = i*n+j; int b; if(i-1 >= 0 && grid[i-1][j] == '1') { MyUnion(a, a-n); } if(i+1 < m && grid[i+1][j] == '1') { MyUnion(a, a+n); } if(j-1 >= 0 && grid[i][j-1] == '1') { MyUnion(a, a-1); } if(j+1 < n && grid[i][j+1] == '1') { MyUnion(a, a+1); } } } return count; } };