

Number of Islands

● MEDIUM

■ Introduction

Given a 2D grid of '1's (land) and '0's (water), count the number of islands. An island is surrounded by water and formed by connecting adjacent land cells horizontally or vertically. This is connected components on a GRID.

■ Core Intuition

Grid as Graph

- Each cell is a NODE. Adjacent (up/down/left/right) land cells are EDGES.
- An island = a connected component of '1' cells.
- Algorithm: scan grid. On finding unvisited '1', do DFS/BFS to sink the entire island (mark as '0' or visited). Increment count.
- Four-directional movement: up(-1,0), down(+1,0), left(0,-1), right(0,+1).

■ Visual

Islands Visual

Grid: After DFS from (0,0):

1 1 0 0 0 0 0 0 ← Island 1 sunk

1 1 0 0 0 0 0 0

0 0 1 0 0 0 0 0 ← Island 2 found at (2,2), sunk

0 0 0 1 0 0 0 0 ← Island 3 found at (3,3), sunk

Total islands = 3

■ Java Implementation

Java

```
class Solution {
    int[][] dirs = {{-1,0},{1,0},{0,-1},{0,1}}; // up,down,left,right

    public int numIslands(char[][] grid) {
        if (grid == null || grid.length == 0) return 0;
        int rows = grid.length, cols = grid[0].length;
        int count = 0;

        for (int r = 0; r < rows; r++) {
```

```

for (int c = 0; c < cols; c++) {
    if (grid[r][c] == '1') {
        dfs(grid, r, c); // Sink the island
        count++;
    }
}

return count;
}

private void dfs(char[][] grid, int r, int c) {
    // Out of bounds or water — stop
    if (r < 0 || r >= grid.length || c < 0 || c >= grid[0].length
        || grid[r][c] != '1') return;

    grid[r][c] = '0'; // Mark as visited (sink land)

    for (int[] d : dirs)
        dfs(grid, r + d[0], c + d[1]);
}

```

■ Time & Space Complexity

Case	Time	Space	Why
Time	$O(m \cdot n)$	$O(m \cdot n)$	Each cell visited at most once; DFS stack $O(m \cdot n)$ worst case
Space	$O(m \cdot n)$	$O(m \cdot n)$	DFS recursion stack, BFS queue — both at most $m \cdot n$ deep

■ Common Mistakes

Avoid These

- Checking diagonal neighbors — islands connect only 4-directionally (not diagonal).
- Not resetting grid — if you modify grid to mark visited, be careful if grid is reused.
- Missing bounds check before accessing `grid[r][c]` — always validate bounds first.

■ Practice Problems

Case	Time	Space	Why
Medium	LC 200	Number of Islands	Direct application
Medium	LC 695	Max Area of Island	DFS + area count
Medium	LC 130	Surrounded Regions	DFS from boundary
Hard	LC 305	Number of Islands II	Union-Find incremental

■ Mental Model

Remember Forever

- "You're a flood that starts at each unvisited land cell. Flood fills the entire island (marks all connected land as water). Each time you start a new flood = new island."