Problem 1559: Detect Cycles in 2D Grid

Problem Information

Difficulty: Medium
Acceptance Rate: 51.67%
Paid Only: No
Tags: Array, Depth-First Search, Breadth-First Search, Union Find, Matrix

Problem Description

Given a 2D array of characters `grid` of size `m x n`, you need to find if there exists any cycle consisting of the **same value** in `grid`.

A cycle is a path of **length 4 or more** in the grid that starts and ends at the same cell. From a given cell, you can move to one of the cells adjacent to it - in one of the four directions (up, down, left, or right), if it has the **same value** of the current cell.

Also, you cannot move to the cell that you visited in your last move. For example, the cycle `(1, 1) -> (1, 2) -> (1, 1)` is invalid because from `(1, 2)` we visited `(1, 1)` which was the last visited cell.

Return `true` if any cycle of the same value exists in `grid`, otherwise, return `false`.

Example 1:

Input: grid = [["a","a","a","a"],["a","b","b","a"],["a","b","b","a"],["a","a","a","a"]] **Output:** true **Explanation:** There are two valid cycles shown in different colors in the image below:

Example 2:

Input: grid = [["c","c","c","a"],["c","d","c","c"],["c","c","e","c"],["f","c","c","c"]] **Output:** true **Explanation:** There is only one valid cycle highlighted in the image below:

Example 3:

Input: grid = [["a","b","b"],["b","z","b"],["b","b","a"]] **Output:** false

Constraints:

* `m == grid.length` * `n == grid[i].length` * `1 <= m, n <= 500` * `grid` consists only of lowercase English letters.

Code Snippets

C++:

```
class Solution {
public:
bool containsCycle(vector<vector<char>>& grid) {

}
};
```

Java:

```
class Solution {
public boolean containsCycle(char[][] grid) {

}
}
```

Python3:

```
class Solution:
def containsCycle(self, grid: List[List[str]]) -> bool:
```