1034. 边界着色

ITCharge 大约 2 分钟

- 标签：深度优先搜索、广度优先搜索、数组、矩阵
- 难度：中等

题目链接

- 1034. 边界着色 - 力扣

题目大意

给定一个二维整数矩阵 `grid` ，其中 `grid[i][j]` 表示矩阵第 `i` 行、第 `j` 列上网格块的颜色值。再给定一个起始位置 `(row, col)` ，以及一个目标颜色 `color` 。

要求：对起始位置 `(row, col)` 所在的连通分量边界填充颜色为 `color` 。并返回最终的二维整数矩阵 `grid` 。

- 连通分量：当两个相邻（上下左右方向上）网格块的颜色值相同时，它们属于同一连通分量。
- 连通分量边界：当前连通分量最外圈的所有网格块，这些网格块与连通分量的颜色相同，与其他周围网格块颜色不同。边界上的网格块也是连通分量边界。

解题思路

深度优先搜索。使用二维数组 `visited` 标记访问过的节点。遍历上、下、左、右四个方向上的点。如果下一个点位置越界，或者当前位置与下一个点位置颜色不一样，则对该节点进行染色。

在遍历的过程中注意使用 `visited` 标记访问过的节点，以免重复遍历。

代码

```py
class Solution:
    directs = [(0, 1), (0, -1), (1, 0), (-1, 0)]

    def dfs(self, grid, i, j, origin_color, color, visited):
        rows, cols = len(grid), len(grid[0])

        for direct in self.directs:
            new_i = i + direct[0]
            new_j = j + direct[1]

            # 下一个位置越界，则当前点在边界，对其进行着色
            if new_i < 0 or new_i >= rows or new_j < 0 or new_j >= cols:
                grid[i][j] = color
                continue

            # 如果访问过，则跳过
            if visited[new_i][ne    :
                continue

            # 如果下一个位置颜色与当前颜色相同，则继续搜索
            if grid[new_i][new_j] == origin_color:
                visited[new_i][new_j] = True
                self.dfs(grid, new_i, new_j, origin_color, color, visited)
            # 下一个位置颜色与当前颜色不同，则当前位置为连通区域边界，对其进行着色
            else:
                grid[i][j] = color


    def colorBorder(self, grid: List[List[int]], row: int, col: int, color: int)
-> List[List[int]]:
        if not grid:
            return grid

        rows, cols = len(grid), len(grid[0])
        visited = [[False for _ in range(cols)] for _ in range(rows)]
```

```
        visited[row][col] = True

        self.dfs(grid, row, col, grid[row][col], color, visited)

        return grid
```