Problem 2123: Minimum Operations to Remove Adjacent Ones in Matrix

Problem Information

Difficulty: Hard
Acceptance Rate: 42.16%
Paid Only: Yes
Tags: Array, Graph, Matrix

Problem Description

You are given a **0-indexed** binary matrix `grid`. In one operation, you can flip any `1` in `grid` to be `0`.

A binary matrix is **well-isolated** if there is no `1` in the matrix that is **4-directionally connected** (i.e., horizontal and vertical) to another `1`.

Return _the minimum number of operations to make_`grid` _**well-isolated**_.

Example 1:

![](https://assets.leetcode.com/uploads/2021/12/23/image-20211223181501-1.png)

Input: grid = [[1,1,0],[0,1,1],[1,1,1]] **Output:** 3 **Explanation:** Use 3 operations to change grid[0][1], grid[1][2], and grid[2][1] to 0. After, no more 1's are 4-directionally connected and grid is well-isolated.

Example 2:

![](https://assets.leetcode.com/uploads/2021/12/23/image-20211223181518-2.png)

Input: grid = [[0,0,0],[0,0,0],[0,0,0]] **Output:** 0 **Explanation:** There are no 1's in grid and it is well-isolated. No operations were done so return 0.

Example 3:

![](https://assets.leetcode.com/uploads/2021/12/23/image-20211223181817-3.png)

Input: grid = [[0,1],[1,0]] **Output:** 0 **Explanation:** None of the 1's are 4-directionally connected and grid is well-isolated. No operations were done so return 0.

Constraints:

* `m == grid.length` * `n == grid[i].length` * `1 <= m, n <= 300` * `grid[i][j]` is either `0` or `1`.

Code Snippets

C++:

```
class Solution {
public:
int minimumOperations(vector<vector<int>>& grid) {

}
};
```

Java:

```
class Solution {
public int minimumOperations(int[][] grid) {

}
}
```

Python3:

```
class Solution:
def minimumOperations(self, grid: List[List[int]]) -> int:
```