Problem 3122: Minimum Number of Operations to Satisfy Conditions

Problem Information

Difficulty: Medium
Acceptance Rate: 41.31%
Paid Only: No
Tags: Array, Dynamic Programming, Matrix

Problem Description

You are given a 2D matrix `grid` of size `m x n`. In one **operation** , you can change the value of **any** cell to **any** non-negative number. You need to perform some **operations** such that each cell `grid[i][j]` is:

* Equal to the cell below it, i.e. `grid[i][j] == grid[i + 1][j]` (if it exists). * Different from the cell to its right, i.e. `grid[i][j] != grid[i][j + 1]` (if it exists).

Return the **minimum** number of operations needed.

Example 1:

Input: grid = [[1,0,2],[1,0,2]]

Output: 0

Explanation:

All the cells in the matrix already satisfy the properties.

Example 2:

Input: grid = [[1,1,1],[0,0,0]]

Output: 3

Explanation:

The matrix becomes `[[1,0,1],[1,0,1]]` which satisfies the properties, by doing these 3 operations:

* Change `grid[1][0]` to 1. * Change `grid[0][1]` to 0. * Change `grid[1][2]` to 1.

Example 3:

Input: grid = [[1],[2],[3]]

Output: 2

Explanation:

There is a single column. We can change the value to 1 in each cell using 2 operations.

Constraints:

* `1 <= n, m <= 1000` * `0 <= grid[i][j] <= 9`

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:
```