Problem 931: Minimum Falling Path Sum

Problem Information

Difficulty: Medium
Acceptance Rate: 60.84%
Paid Only: No
Tags: Array, Dynamic Programming, Matrix

Problem Description

Given an `n x n` array of integers `matrix`, return _the**minimum sum** of any **falling path** through_ `matrix`.

A **falling path** starts at any element in the first row and chooses the element in the next row that is either directly below or diagonally left/right. Specifically, the next element from position `(row, col)` will be `(row + 1, col - 1)`, `(row + 1, col)`, or `(row + 1, col + 1)`.

Example 1:

![](https://assets.leetcode.com/uploads/2021/11/03/failing1-grid.jpg)

Input: matrix = [[2,1,3],[6,5,4],[7,8,9]] **Output:** 13 **Explanation:** There are two falling paths with a minimum sum as shown.

Example 2:

![](https://assets.leetcode.com/uploads/2021/11/03/failing2-grid.jpg)

Input: matrix = [[-19,57],[-40,-5]] **Output:** -59 **Explanation:** The falling path with a minimum sum is shown.

Constraints:

* `n == matrix.length == matrix[i].length` * `1 <= n <= 100` * `-100 <= matrix[i][j] <= 100`

Code Snippets

C++:

```
class Solution {
public:
int minFallingPathSum(vector<vector<int>>& matrix) {

}
};
```

Java:

```
class Solution {
public int minFallingPathSum(int[][] matrix) {

}
}
```

Python3:

```
class Solution:
def minFallingPathSum(self, matrix: List[List[int]]) -> int:
```