Problem 265: Paint House II

Problem Information

Difficulty: Hard
Acceptance Rate: 56.73%
Paid Only: Yes
Tags: Array, Dynamic Programming

Problem Description

There are a row of `n` houses, each house can be painted with one of the `k` colors. The cost of painting each house with a certain color is different. You have to paint all the houses such that no two adjacent houses have the same color.

The cost of painting each house with a certain color is represented by an `n x k` cost matrix costs.

* For example, `costs[0][0]` is the cost of painting house `0` with color `0`; `costs[1][2]` is the cost of painting house `1` with color `2`, and so on...

Return _the minimum cost to paint all houses_.

Example 1:

Input: costs = [[1,5,3],[2,9,4]] **Output:** 5 **Explanation:** Paint house 0 into color 0, paint house 1 into color 2. Minimum cost: 1 + 4 = 5; Or paint house 0 into color 2, paint house 1 into color 0. Minimum cost: 3 + 2 = 5.

Example 2:

Input: costs = [[1,3],[2,4]] **Output:** 5

Constraints:

* `costs.length == n` * `costs[i].length == k` * `1 <= n <= 100` * `2 <= k <= 20` * `1 <= costs[i][j] <= 20`

Follow up: Could you solve it in `O(nk)` runtime?

Code Snippets

C++:

```
class Solution {
public:
int minCostII(vector<vector<int>>& costs) {

}
};
```

Java:

```
class Solution {
public int minCostII(int[][] costs) {

}
}
```

Python3:

```
class Solution:
def minCostII(self, costs: List[List[int]]) -> int:
```