5 Leetcode256房子涂色付钱问题

笔记本： DP Note
创建时间： 10/19/2019 11:54 PM **更新时间：** 10/23/2019 10:18 PM
作者： tanziqi1756@outlook.com

256. Paint House

难度 简单 12 收藏 分享 切换为中文 关注

题目描述 评论 (18) 题解(14) New 提交记录

There are a row of *n* houses, each house can be painted with one of the three colors: red, blue or green. 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 a `n x 3` cost matrix. For example, `costs[0][0]` is the cost of painting house 0 with color red; `costs[1][2]` is the cost of painting house 1 with color green, and so on... Find the minimum cost to paint all houses.

Note:
All costs are positive integers.

Example:

```
Input: [[17,2,17],[16,16,5],[14,3,19]]
Output: 10
Explanation: Paint house 0 into blue, paint house 1 into green, paint house 2 into blue.
             Minimum cost: 2 + 5 + 3 = 10.
```

如果直接套用之前机器人走路的思路，可以做到O(n^3)
因为这里只有三种颜色，可以做到O(n)
如果是N种颜色，就只能是O(n^3)

还是用机器人走路的二维dp
这里可以in-place，即使用原来的costs作为dp矩阵
输入[

[17,2,17],
[16,16,5],
[14,3,19]
]
第一行[17, 2, 17]为dp初始状态
第二行开始进行状态转移方程：dp[i][j] = Min(dp[i-1]) + dp[i][j]
含义：在对第i座房子，涂第j种颜色时，第i-1座房子就不能涂成第j种颜色，
因而，就找第i-1行的最小值（最小花费），
加上当前对第i座房子，涂第j种颜色时的花费。

最终的dp矩阵
17 2 17
18 33 7
21 10 37
遍历最后一行，得到最小值10

```java
class Solution {
    public int minCost(int[][] costs) {
        if( costs.length == 0 ) {
            return 0;
        }
        for( int i = 1; i < costs.length; i++ ) {
            for( int j = 0; j < costs[0].length; j++ ) {
                // For every row[i], find the minimum of row[i-1], excluding the element costs[i-1][j]
                int minCost = findMin(costs[i-1], j);
                costs[i][j] += minCost;
            }
        }

        int ans = Integer.MAX_VALUE;
        int N = costs.length - 1;
        int colors = costs[0].length;
        for( int j = 0; j < colors; j++ ) {
            if( costs[N][j] < ans ) {
                ans = costs[N][j];
            }
        }
        return ans;
    }

    private int findMin(int[] cost, int index) {
        int minCost = Integer.MAX_VALUE;
        for( int i = 0; i < cost.length; i++ ) {
            if( i != index && cost[i] < minCost ) {
                minCost = cost[i];
            }
        }
        return minCost;
    }
}
```

Leetcode 265 房子涂颜色升级版+时间优化
每次都要算除了一个数，其他元素中的最小元素，
那我只要直接记录最小元素和次小元素就行了。

```java
class Solution {
    public int minCostII(int[][] costs) {
        if (costs.length == 0) {
            return 0;
        } else if (costs[0].length == 1) {
            return costs[0][0];
        }
        int minColour = -1;
        int minCost = 0;
        int secondMinCost = 0;
        for (int[] cost : costs) {
            int tmpMinColour = -1;
            int tmpMinCost = Integer.MAX_VALUE;
            int tmpSecondMinCost = Integer.MAX_VALUE;
            for (int i = 0; i < cost.length; i++) {
                int thisMinCost = cost[i] + (i == minColour ? secondMinCost : minCost);
                if (thisMinCost < tmpMinCost) {
                    tmpSecondMinCost = tmpMinCost;
                    tmpMinCost = thisMinCost;
                    tmpMinColour = i;
                } else if (thisMinCost < tmpSecondMinCost) {
                    tmpSecondMinCost = thisMinCost;
                }
            }
            minCost = tmpMinCost;
            minColour = tmpMinColour;
            secondMinCost = tmpSecondMinCost;
        }
        return minCost;
    }
}
```