

4. 运行结果

输入: $n=3$ $k=2$

输出: 6

▶ 例 130 房屋染色

1. 问题描述

有 n 个房子在一直线上。需要给房屋染色, 分别有红色、蓝色和绿色。每个房屋染不同的颜色费用不同, 需要设计一种染色方案, 使得相邻的房屋颜色不同, 并且费用最少。返回最少的费用。

2. 问题示例

费用通过一个 $n \times 3$ 的矩阵给出。例如, $\text{cost}[0][0]$ 表示房屋 0 染红色的费用, $\text{cost}[1][2]$ 表示房屋 1 染绿色的费用。所有费用都是正整数。

输入 $[[14, 2, 11], [11, 14, 5], [14, 3, 10]]$, 输出 10, 也就是 3 个房子分别为蓝色、绿色和红色, $2 + 5 + 3 = 10$ 。

输入 $[[1, 2, 3], [1, 4, 6]]$, 输出 3, 也就是两个房子分别为绿色和蓝色, $2 + 1 = 3$ 。

3. 代码实现

参数 costs: $n \times 3$ 矩阵

返回值: 整数, 刷完所有房子最少花费

class Solution:

```
def minCost(self, costs):
    n = len(costs)
    if n == 0:
        return 0
    INF = 0x7fffffff
    f = [costs[0], [INF, INF, INF]]
    for i in range(1, n):
        for j in range(3):
            f[i&1][j] = INF
            for k in range(3):
                if j != k:
                    f[i&1][j] = min(f[i&1][j], f[(i+1)&1][k] + costs[i][j])
    return min(f[(n-1)&1])
```

主函数

```
if __name__ == '__main__':
    generator = [[14, 2, 11], [11, 14, 5], [14, 3, 10]]
    solution = Solution()
    print("输入: ", generator)
    print("输出: ", solution.minCost(generator))
```