64. 最小路径和

题目：64. Minimum Path Sum 语言: python3 英文版链接：https://leetcode.com/problems/minimum-path-sum/description/ 中文版链接：https://leetcode-cn.com/problems/minimum-path-sum/description/

题目分析

对于网格中的元素$grid[i][j]$,从最上角的元素$grid[0][0]$走到它的最短距离为：
$grid[i][j] = min(grid[i-1][j], grid[i][j-1]) + grid[i]$[j] 因此，这题的思路是：首先计算出第一行从左到右的步数。然后从第二行开始，采用动态规划的方法。

事实上，我们只需要维护一个一维dp数组即可，这个数组代表走到对应行的每个位置的最短路径。当更新某位置dp[j]的时候只需要上方路径信息dp[j]（可以看作从上向下一遍一遍访问矩阵，更新dp，那么此时的dp[j]由于还未更新，所以是上方位置的路径信息）和左侧路径信息dp[j-1]即可，所以dp数组可以不断的覆盖，而不需要维护二维dp数组。

答案

```
class Solution:
    def minPathSum(self, grid: List[List[int]]) -> int:
        r, c = len(grid), len(grid[0])
        dp = [0] * c
        for i in range(r):
            for j in range(c):
                if j == 0:
                    dp[j] = dp[j]
                elif i == 0:
                    dp[j] = dp[j - 1]
                else:
                    dp[j] = min(dp[j - 1], dp[j])
                dp[j] += grid[i][j]
        return dp[c - 1]
```