

编程题_README

1.题目简介

一个机器人位于一个 $m \times n$ 网格的左上角，机器人每次只能向下或者向右移动一步。机器人试图达到网格的右下角。问总共有多少条不同的路径？输入为网络的尺寸 m 和 n ，输出为路径的数量，动态规划法求解。



2.我的思路

在本题中，机器人从 $(0, 0)$ 点出发，走到 $(m - 1, n - 1)$ 终点，并且只能向右和向下移动；由于机器人移动的规律性，我们就可以发现，到达每一个位置只能从两个方向：即从上方或者从左方移动一格到达指定位置；所以我们采用动态规划的思想，首先确定 $dp[i][j]$ 的含义：从左上角 $(0, 0)$ 点移动到右下角 (i, j) 点所有的不同路径的条数；接下来就要动态规划的递推公式，本题中，到达 (i, j) 的方法有两种：从上方向下移动一格或从左方向右移动一格；所以这是就可以确定状态转移方程： $dp[i][j] = dp[i - 1][j] + dp[i][j - 1]$ ；接着进行 dp 数组的初始化，发现第一行和第一列的所以位置只能有一种到达方式：即一直向右走到目标位置或者一直向下走到目标位置，故 $dp[i][0] = dp[0][j] = 1$ ；这样我们就确定的本题动态规划的大致流程，通过遍历得出最后的答案。(这个时候我们要注意到，由于我们的 $dp[i][0]$ 和 $dp[0][j]$ 初始化的值已经设定为 1 了，遍历就需要从第二行和第二列开始，不能从第一行第二列开始，这样会出错。即遍历的下标起点应该从 1 开始)

3.主要函数说明

除了主函数外本题所用最重要的一个函数：

```
int uniquePaths(int m, int n)
{
    vector<vector<int>> dp(m, vector<int>(n));
    for (int i = 0; i < m; i++)
        dp[i][0] = 1;
```

```

    for (int j = 0; j < n; j++)
        dp[0][j] = 1;
    for (int i = 1; i < m; i++)
        for (int j = 1; j < n; j++)
            dp[i][j] = dp[i - 1][j] + dp[i][j - 1];
    return dp[m - 1][n - 1];
}

```

本函数的作用为通过动态规划的方法求解不同路径的总数，其输入为网格的层数 m 和 n ，返回值是从左上角到右下角不同路径的总数。在该函数中，我们首先对所定义的 dp 数组进行初始化，即 $dp[i][0] = dp[0][j] = 1$ (上文已经分析过了)，然后我们就开始从第二行和第二列进行遍历，使用到动态规划的状态转移方程 $dp[i][j] = dp[i - 1][j] + dp[i][j - 1]$ ，最后得出不同路径的结果并且 `return`。

4.五个测试用例

用例 1:

输入: $m = 3, n = 7$

输出: 28

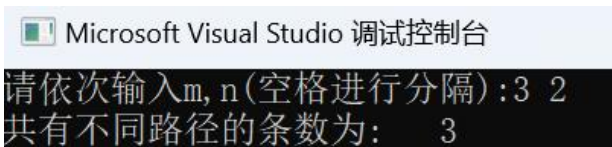


Microsoft Visual Studio 调试控制台
请依次输入m, n(空格进行分隔): 3 7
共有不同路径的条数为: 28

用例 2:

输入: $m = 3, n = 2$

输出: 3

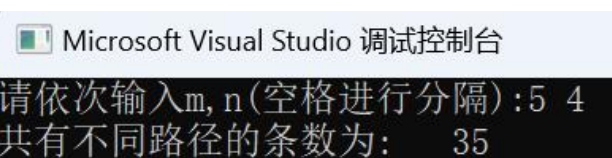


Microsoft Visual Studio 调试控制台
请依次输入m, n(空格进行分隔): 3 2
共有不同路径的条数为: 3

用例 3:

输入: $m = 5, n = 4$

输出: 35

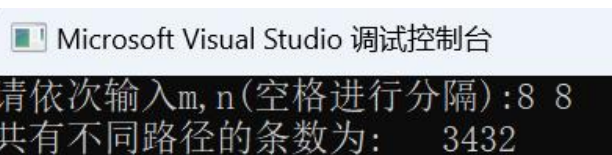


Microsoft Visual Studio 调试控制台
请依次输入m, n(空格进行分隔): 5 4
共有不同路径的条数为: 35

用例 4:

输入: $m = 8, n = 8$

输出: 3432



Microsoft Visual Studio 调试控制台
请依次输入m, n(空格进行分隔): 8 8
共有不同路径的条数为: 3432

用例 5:

输入: $m = 4, n = 9$

输出: 165



Microsoft Visual Studio 调试控制台
请依次输入m, n(空格进行分隔): 4 9
共有不同路径的条数为: 165