# 70、 爬楼梯（Climbing Stairs）

## 题目：

假设你正在爬楼梯。需要 *n* 阶你才能到达楼顶。

每次你可以爬 1 或 2 个台阶。你有多少种不同的方法可以爬到楼顶呢？

**注意：**给定 *n* 是一个正整数。

**示例 1：**

输入： 2   
输出： 2   
解释： 有两种方法可以爬到楼顶。   
1. 1 阶 + 1 阶   
2. 2 阶

**示例 2：**

输入： 3   
输出： 3   
解释： 有三种方法可以爬到楼顶。   
1. 1 阶 + 1 阶 + 1 阶   
2. 1 阶 + 2 阶   
3. 2 阶 + 1 阶

## 解答：

|  |
| --- |
| public static int ClimbStairs(int n)  {  /\*  1、可从前面0到第一个楼梯  0 1 1 1 1 1  1    2、第二个阶梯可从0来也可以从1来  0 1 1 1 1 1  1 2  3、第三个可以从第一个和第二个来  0 1 1 1 1 1  1 2 3    4、注意第四个它可以从第二个和第三个来，另外注意想要到达第二和第三个要2步和3步，所以到第四个需要2+3步  0 1 1 1 1 1  1 2 3 5  ....以此类推  这叫dp算法。  \*/  if (n <= 0)  {  return 0;  }  if (n == 1)  {  return 1;  }  if (n == 2)  {  return 2;  }  int[] dp = new int[n + 1];  dp[0] = 0;  dp[1] = 1;  dp[2] = 2;  for (int i = 3; i <= n; i++)  {  dp[i] = dp[i - 1] + dp[i - 2];  }  return dp[n];  } |