70. Climbing Stairs

Easy,
Dynamic Programming.

You are climbing a stair case. It takes n steps to reach to the top.

Each time you can either climb 1 or 2 steps. In how many distinct ways can you climb to the top?

Note: Given n will be a positive integer.

Example 1:

```
Input: 2
Output: 2
Explanation: There are two ways to climb to the top.
1. 1 step + 1 step
2. 2 steps
```

Example 2:

```
Input: 3
Output: 3
Explanation: There are three ways to climb to the top.
1. 1 step + 1 step + 1 step
2. 1 step + 2 steps
3. 2 steps + 1 step
```

解法

这道题是一道简单的动态规划的题。
问题是每次可以选择跨一个台阶和两个台阶，问到某一个台阶一共有多少种可能性。
所以状态转换方程是，前一个台阶的可能性+前两个台阶可能性。
dp[i] = dp[i-1]+dp[i-2];

Java

```java
class Solution {
    public int climbStairs(int n) {
        int[] dp = new int[n+2];
        if(n==0 || n==1 || n==2) return n;
        for(int i=0; i<n+2; i++) dp[i]=0;
        dp[1]=1;
        dp[2]=2;
        for(int i=0; i<n;i++){
            dp[i+2]=dp[i+1]+dp[i];
        }
        return dp[n+1];
    }
}
```