70. Climbing Stairs

Total Accepted: **85396** Total Submissions: **239803** Difficulty: **Easy**

> 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?

Analysis:
List the first several terms:
n = 1, fn = 1
n = 2, fn = 2 (1+1, or 2)
n = 3, fn = 3 (1+1+1, or 1+2, or 2+1)
n = 4, fn = 5 (1+1+1+1, or 1+2+1, or 1+1+2, or 2+1+1, or 2+2)
…..
we found that the No. of ways at n equals the No. of ways at n-1 plus that at n-2. Thus this is a Fibonacci question.

Following are the code:

```
3  class Solution(object):
4      def climbStairs(self, n):
5          fn = [0, 1, 2]
6          i = 3
7          while i<=n :
8              fn.append(fn[i-2] + fn[i-1] )
9              i += 1
10         return fn[n]
11
12 s = Solution()
13 for i in range(5):
14     res = s.climbStairs(i)
15     print(res)
```

Time complexity is O(n) space complexity is O(1)