**13.You are climbing a staircase. It takes n steps to reach the top. Each time you can either climb 1 or 2 steps. In how many distinct ways can you climb to the top?**

**Examples:**

**Input: n=4 Output: 5**

**Input: n=3 Output:3**

**Aim:**

To find how many distinct ways you can climb to the top of a staircase with n steps, if you can climb either 1 or 2 steps at a time.

**Algorithm:**

This is a classic Dynamic Programming problem.  
It follows the Fibonacci pattern:

* If n = 1, there's only 1 way: [1]
* If n = 2, there are 2 ways: [1+1], [2]
* For n >= 3:

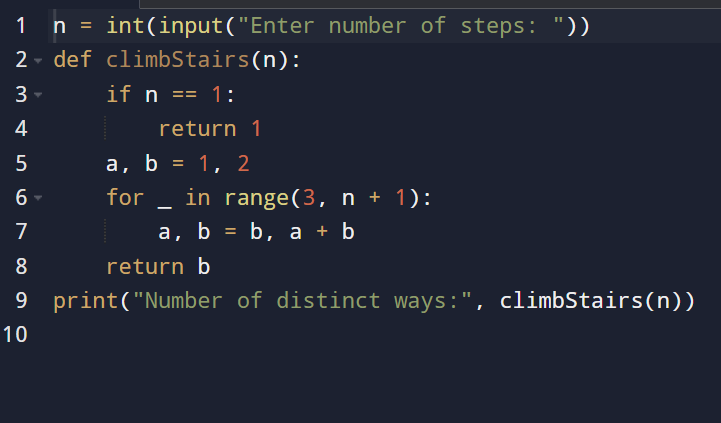
ways(n)=ways(n−1)+ways(n−2)ways(*n*)=ways(*n*−1)+ways(*n*−2)

This is because:

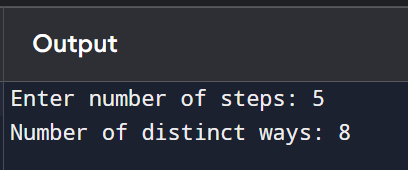
* To reach step n, you must come from step n-1 or n-2
* So, total ways = ways to reach n-1 + ways to reach n-2

Use two variables to store previous results to keep space complexity minimal.

**Code:**

****

**Input and output:**

****

**Result:** given Climbing Stairs – Count Distinct Ways to Reach the Top is successfully executed and output is verified

**Performance analysis:**

* **Time Complexity: O(n)*O*(*n*)**
* **Space Complexity: O (1)*O*(1)**