**Problem-4**

**Aim:**

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?

1. **Problem Description :**

Problem is related to climbing a staircase and we can climb either 1 or 2 steps each time. Since the problem  
contains an optimal substructure and has overlapping sub-problems, it can be solved using bottom-up approach by dynamic programming (We can use brute force method also but it will be time consuming according to the  
time complexity of the algorithm).  
One can reach the ith step in one of the two ways :  
a) Take one step from (i – 1) th step.  
b) Take two steps from (i – 2) th step.  
We will use the tabulation method of dynamic programming ans store the values in vector after each step. When  
we reach at top of stairs then the last value in the vector is the number of ways to climb to the top.

1. **Algorithm :**

Fun climbStairs(n)

Set step1=1 and set step2=1

For loop(2 to n)

Set step2 = step2+step1

Swap(step1, step2)

1. **Source Code for Experiment :**

class Solution {

public:

    int climbStairs(int n) {

        int step1 = 1, step2 = 1;

        for(int i = 2; i <= n; i++){

            step2 += step1;

            swap(step1, step2);

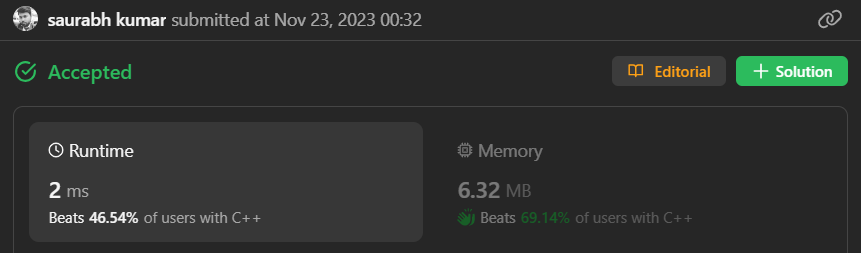
        }

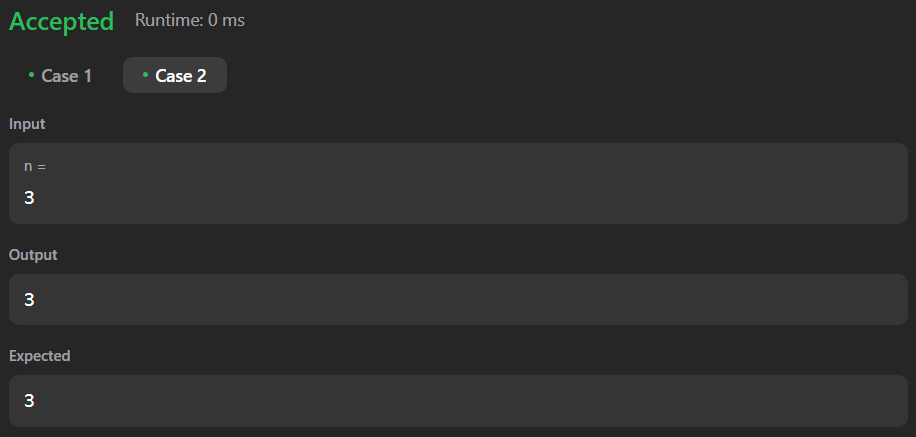
        return step1;

    }

};

1. **Result/Output :**

****

****

**Learning outcomes (What I have learnt):**

1. Learnt about the linked list data structure in C++.
2. Learnt about the fast and slow pointers approach.
3. Learnt about how to find the middle node in linked list.
4. Learnt about how to reverse the linked list.
5. Learnt about how to check palindrome linked list.