Problem-4

Aim of the Experiment :

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?

Constraints:

$1 <= n <= 45$

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.

2. Algorithm :

Step 1: If number of step to reach top is 0 or 1 then

```
if (n == 0 || n == 1) {
        return 1;
}.
```

Step 2: Initialise vector<int> dp of size n+1.

Step 3: Assign dp[0] = 1 and dp[1] = 1.

Step 4: Initialize i=2 and repeat Step 5 until i<=n.

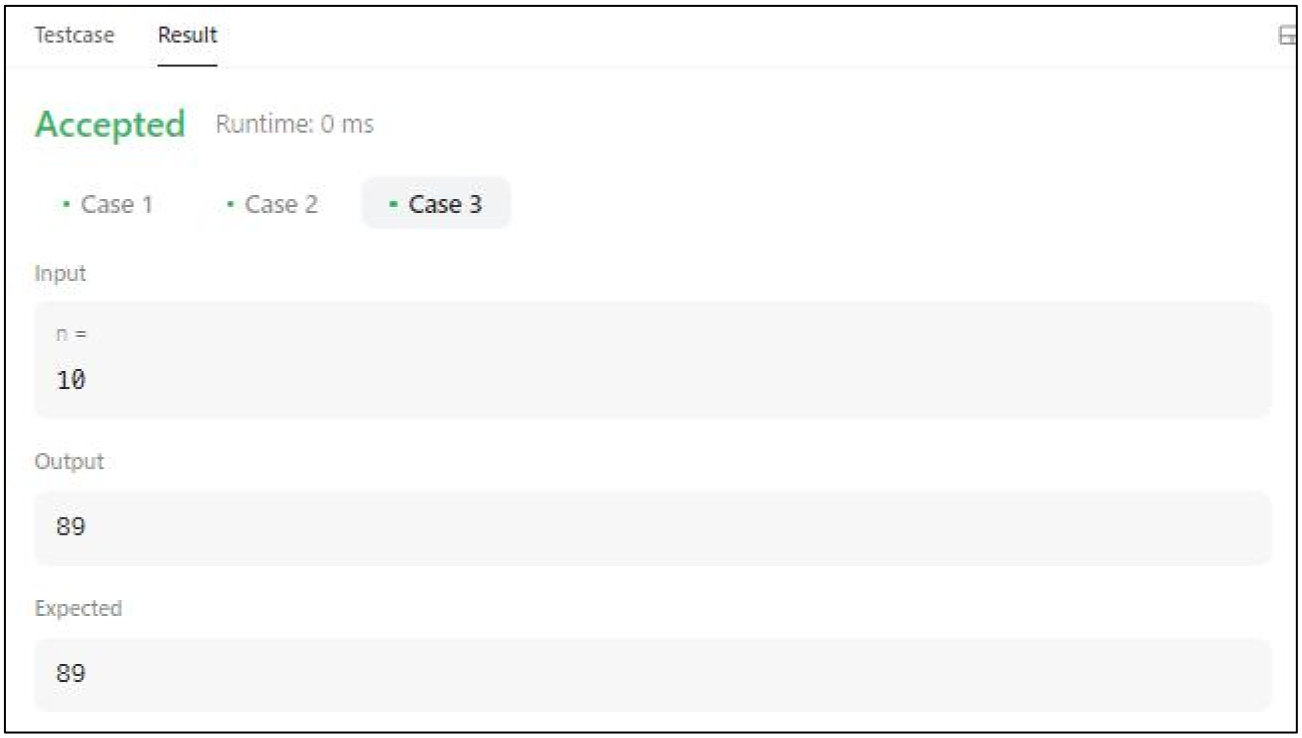
Step 5: dp[i] = dp[i -1] + dp[i -2]

Step 6: Return the value of dp[n].

3. Source Code for Experiment :

```
class Solution {

public:

    int climbStairs(int n) {
        if (n == 0 || n == 1) {
            return 1;
        }

        vector<int> dp(n+1);
        dp[0] = dp[1] = 1;

        for (int i = 2; i <= n; i++) {
            dp[i] = dp[i-1] + dp[i-2];
        }

        return dp[n];
    }
};
```

4. Result/Output :



```
n =
10
```

```
89
```

```
89
```

Learning outcomes (What I have learnt):

1. I learnt about the vector data structure in C++.

2. I learnt about the dynamci programming approach.

3. I learnt about memoization and tabulation approaches in DP.

4. I learnt about how to store values in a vector.

5. I learnt about applications of dynamic programming.