

## EXPERIMENT - 10

**Student Name:** Sandeep Kumar

**UID:** 20BCS4885

**Branch:** CSE

**Section/Group:** 603/A

**Semester:** 6<sup>th</sup> semester

**Subject:** Competitive Coding

**Aim:** To demonstrate the concept of Dynamic Programming.

**Objective:**

- a) Best time to buy and sell the stock
- b) Climbing Stairs

### Problem 1: Best time to buy and sell the stock

**Solution code:**

```
class Solution {
public:
    int maxProfit(vector<int>& prices) {
        int n = prices.size(), maxProfit = 0;
        for(int i=0; i<n-1; i++){
            for(int j=i+1; j<n; j++){
                if(prices[j] - prices[i] > maxProfit){
                    maxProfit = prices[j] - prices[i];
                }
            }
        }
        return maxProfit;
    }
};
```

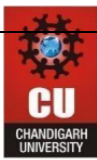
**Approach:**

Solved using Array (Two Nested Loop). Brute Force Approach.

**Complexity:**

Time Complexity:  $O(N^2)$

Space Complexity:  $O(1)$



## Output:

The screenshot shows a coding platform interface with a dark theme. At the top, there are tabs for 'Testcase' and 'Result', with 'Result' being the active tab. Below the tabs, the status 'Accepted' is displayed in green, followed by 'Runtime: 4 ms'. There are two test case buttons: 'Case 1' (active) and 'Case 2'. Under the 'Input' section, the text 'prices =' is followed by a text box containing '[7,1,5,3,6,4]'. The 'Output' section shows a text box with the value '5'. The 'Expected' section also shows a text box with the value '5'. At the bottom of the main content area, there is a link 'Contribute a testcase' with a heart icon. The footer contains a 'Console' dropdown, a settings icon, and two buttons: 'Run' and 'Submit' (highlighted in green).

## Problem 2: Climbing Stairs

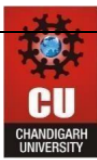
### Input Code:

```
class Solution {
public:
    int climbStairs(int n) {
        if(n<=2)
            return n;
        vector<int> dp(n+1);
        dp[0]=0;
        dp[1]=1;
        dp[2]=2;
        for(int i=3;i<=n;i++)
            dp[i]=dp[i-1]+dp[i-2];

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

### Approach:

- Using Bottom - up Approach -> Tabulation.
- Storing the values of overlapping sub - problems in a vector.



## Complexity:

Time Complexity:  $O(n)$  -> As we are traversing the vector at least 1 time.

Space Complexity:  $O(n)$  -> (Array of size  $n$ )

## Output:

The screenshot shows a coding platform interface with a dark theme. At the top, there are tabs for 'Testcase' and 'Result', with 'Result' being the active tab. Below the tabs, the status 'Accepted' is displayed in green, followed by 'Runtime: 0 ms'. There are two tabs for test cases: 'Case 1' (active) and 'Case 2'. Under 'Case 1', the 'Input' section shows 'n =' followed by a text box containing the value '2'. The 'Output' section shows a text box containing the value '2'. The 'Expected' section shows a text box containing the value '2'. At the bottom of the interface, there is a 'Console' dropdown menu, a 'Run' button, and a green 'Submit' button. A link to 'Contribute a testcase' is also visible.