**Assingment -8**

**Name-Nikhil**

**Uid-22BCS11558**

**LEETCODE:[https://leetcode.com/u/nikhil11558/](mailto:https:/leetcode.com/u/nikhil11558/)**

**Q1. Climbing Stairs**

**Source code:-**

class Solution {

public:

    int climbStairs(int n) {

           if (n <= 3) return n;

        int prev1 = 3;

        int prev2 = 2;

        int cur = 0;

        for (int i = 3; i < n; i++) {

            cur = prev1 + prev2; //3+2

            prev2 = prev1;

            prev1 = cur;

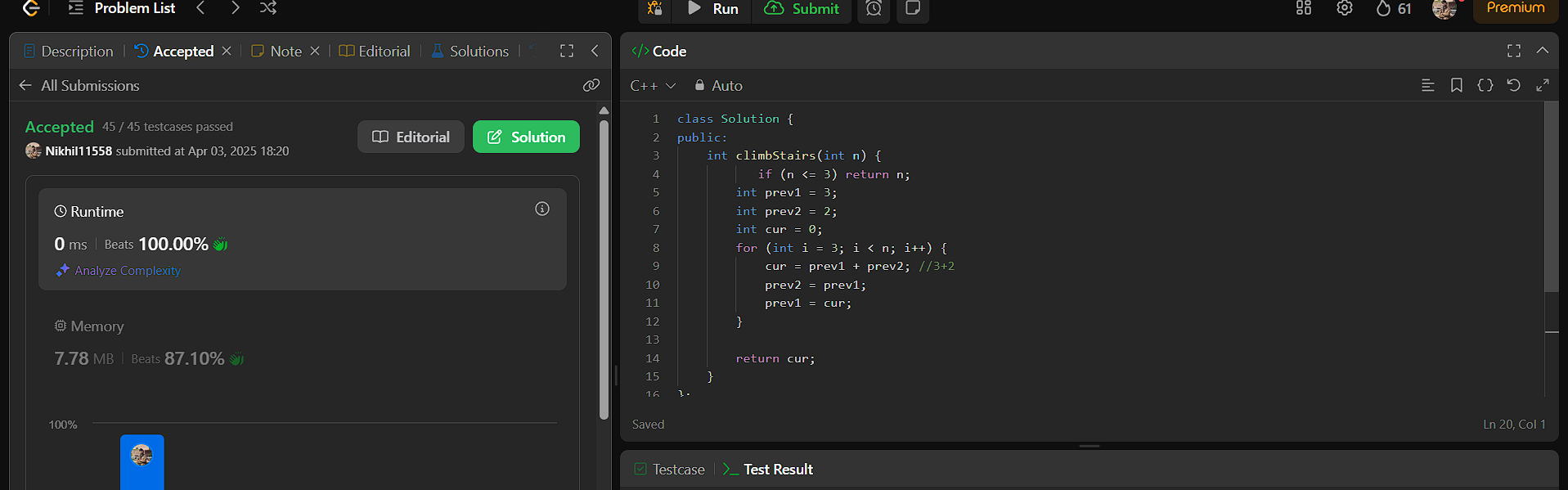
        }

        return cur;

    }

};

**Screenshot:-**



**Q2: Best Time to Buy and Sell a Stock**

**Source code:**

class Solution {

public:

    int maxProfit(vector<int>& prices) {

        int n=prices.size();

        int profit=0;

        int mini=prices[0];

        for(int i=1;i<n;i++){

            int cost=prices[i]-mini;

            profit= max(cost,profit);

            mini=min(mini,prices[i]);

            }

           if(profit<0 || profit==0){

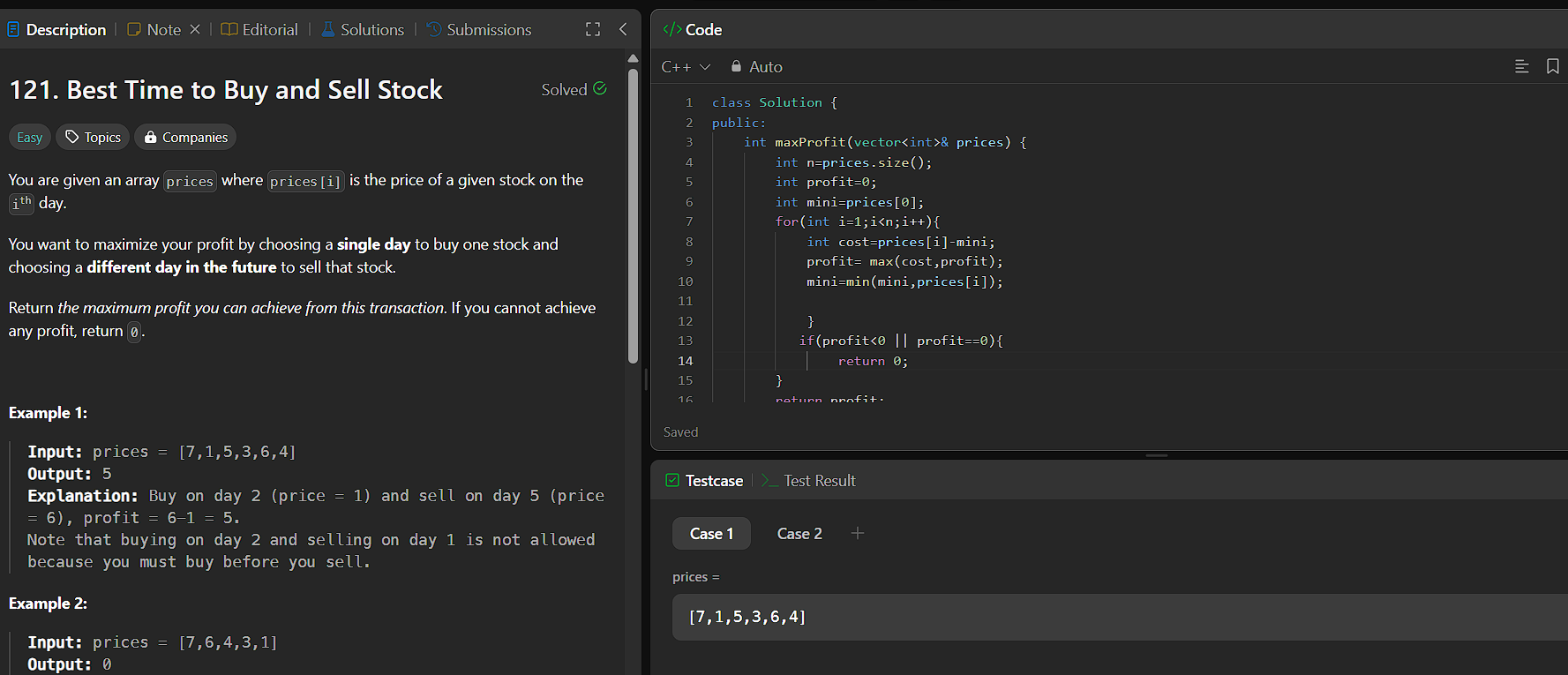
                return 0;

        }        return profit

}

};

**Screenshot:**



**Q3: Jump Game**

**Source code:**

class Solution {

public:

    bool canJump(vector<int>& nums) {

        int n=nums.size();

        int maxreach=0;

        for(int i=0;i<n;i++){

            if(i>maxreach) return false;

             maxreach=max(maxreach,nums[i]+i);

             if(maxreach>=n-1) return true;

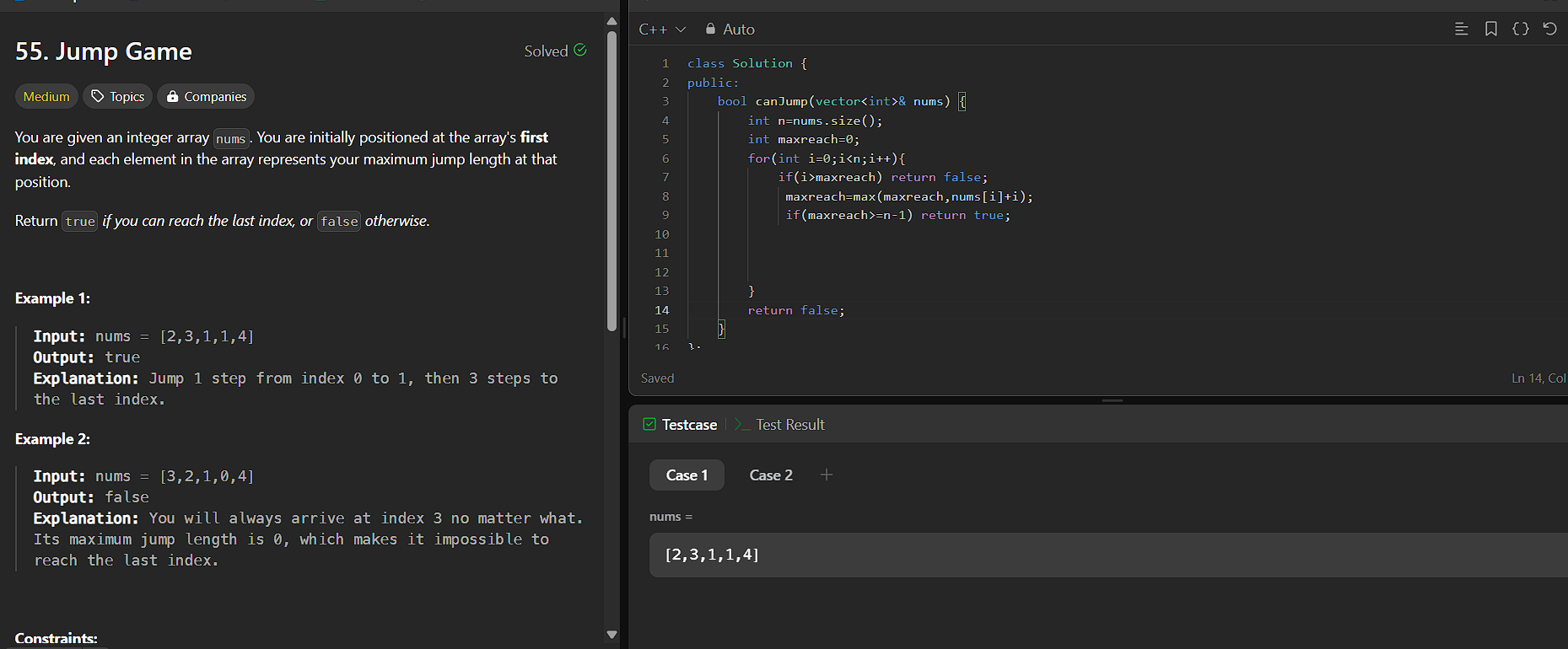
        }

        return false;

    }

};

**Screenshot:**



**Q4.[Maximum Product Subarray](https://leetcode.com/problems/maximum-product-subarray/)**

**Source code:**

class Solution {

public:

    int maxProduct(vector<int>& nums) {

        int suffix=1, prefix=1;

        int n=nums.size();

        int ans=INT\_MIN;

        for(int i=0;i<n;i++ ){

            if(prefix==0) prefix=1;

            if(suffix==0) suffix=1;

           prefix\*=nums[i];

            suffix\*=nums[n-i-1];

            ans=max(ans,max(prefix,suffix));

        }

        return ans;

    }

};

**Screenshot:**

