# Experiment 7

Name: Rakesh UID: 22BET10340

Branch: BE-IT Section: 22BET-IOT-703/A

Semester: 6th Subject Code: 22ITP-351

Problem 1

**AIM : Climbing Stairs**

Code:

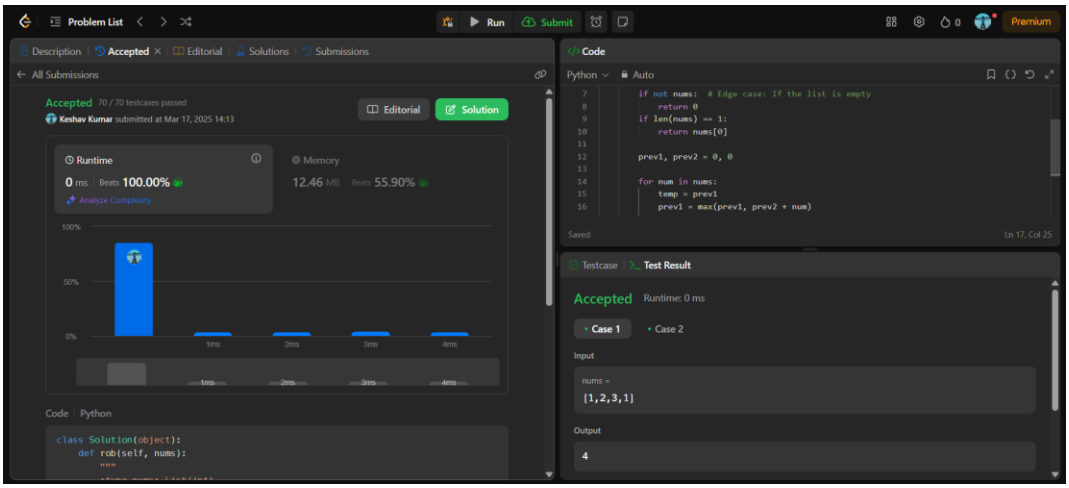
```
class Solution(object):
    def climbStairs(self, n):
        """
        :type n: int
        :rtype: int
        """
        if n <= 2:
            return n

        dp = [0] * (n + 1)
        dp[1], dp[2] = 1, 2

        for i in range (3, n + 1):
            dp[i] = dp[i - 1] + dp[i - 2]

        return dp[n]
```

Output :



Problem 2

**AIM : Maximum Subarray**

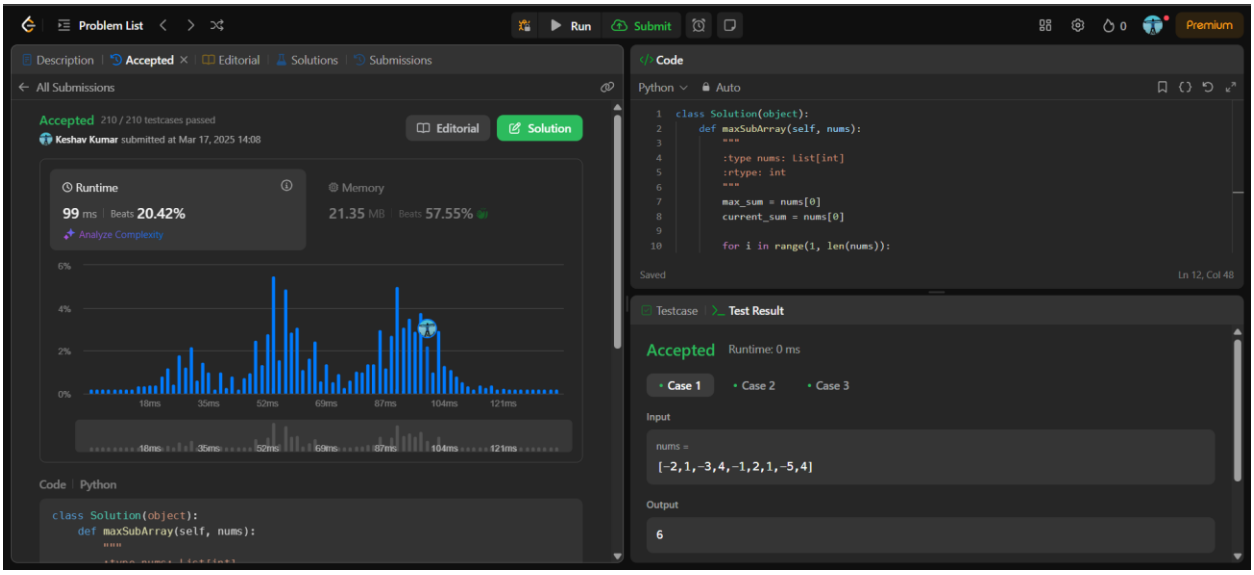
Code :

```
class Solution(object):
    def maxSubArray(self, nums):
        """
        :type nums: List[int]
        :rtype: int
        """
        max_sum = nums[0]
        current_sum = nums[0]

        for i in range(1, len(nums)):
            current_sum = max(nums[i], current_sum + nums[i])
            max_sum = max(max_sum, current_sum)

        return max_sum
```

Output:



Problem 3

**AIM : House Robber**

Code :

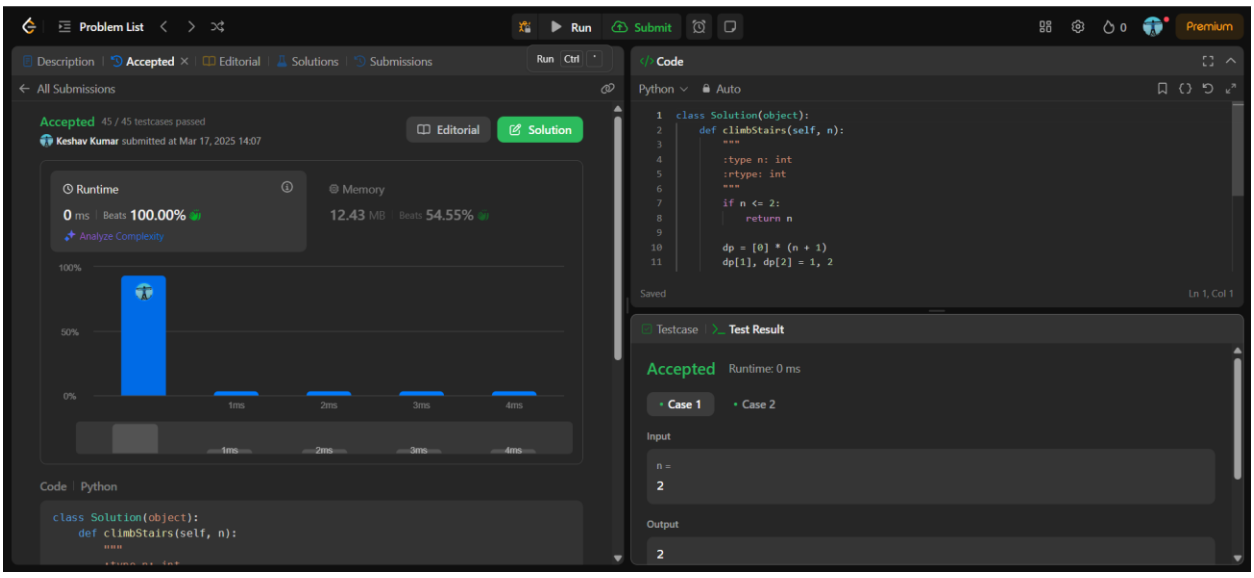
```python
class Solution(object):
    def rob(self, nums):
        """
        :type nums: List[int]
        :rtype: int
        """
        if not nums:  # Edge case: If the list is empty
            return 0
        if len(nums) == 1:
            return nums[0]

        prev1, prev2 = 0, 0

        for num in nums:
            temp = prev1
            prev1 = max(prev1, prev2 + num)
            prev2 = temp

        return prev1
```

Output :



Problem 4

**AIM : Jump Game**

**Code :**

```
class Solution(object):
    def canJump(self, nums):
        """
        :type nums: List[int]
        :rtype: bool
        """
        max_reach = 0

        for i in range(len(nums)):
            if i > max_reach:
                return False
            max_reach = max(max_reach, i + nums[i])
            if max_reach >= len(nums) - 1:
                return True

        return False
```

**Output :**