Experiment 6

Student Name: Nitil Jakhar
UID: 22BCS17300
Branch: CSE
Section/Group: NTPP_IOT-602/A
Semester: 6th
Date of Performance: 3/02/2
Subject: AP 2

1. **Aim: Dynamic Programming**
2. **Objective:**
   1. Maximum Subarray
   2. Maximum Product Subarray
3. **Code:**

1. **Maximum Subarray:**

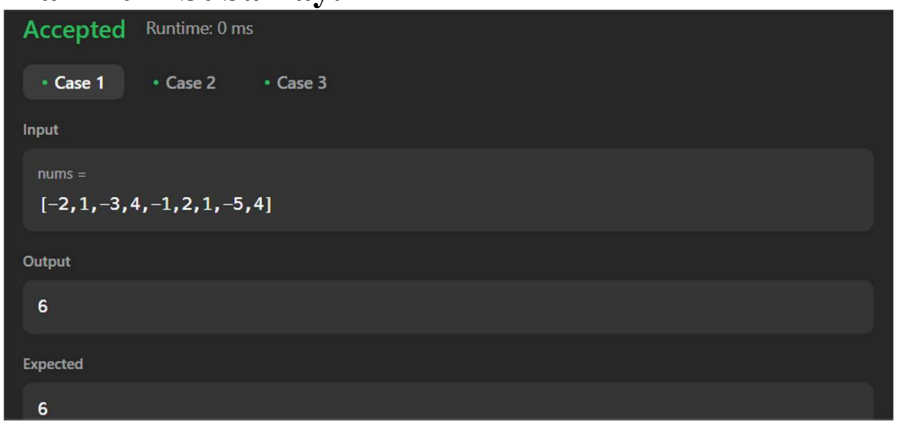
```
from typing import List

class Solution:
    def maxSubArray(self, nums: List[int]) -> int:
        max_sum = float('-inf')  # Initialize max_sum with the smallest possible value
        current_sum = 0  # Tracks the sum of the current subarray

        for num in nums:
            current_sum = max(num, current_sum + num)  # Choose to start a new subarray or continue
            max_sum = max(max_sum, current_sum)  # Update max_sum if the current sum is greater

        return max_sum
```

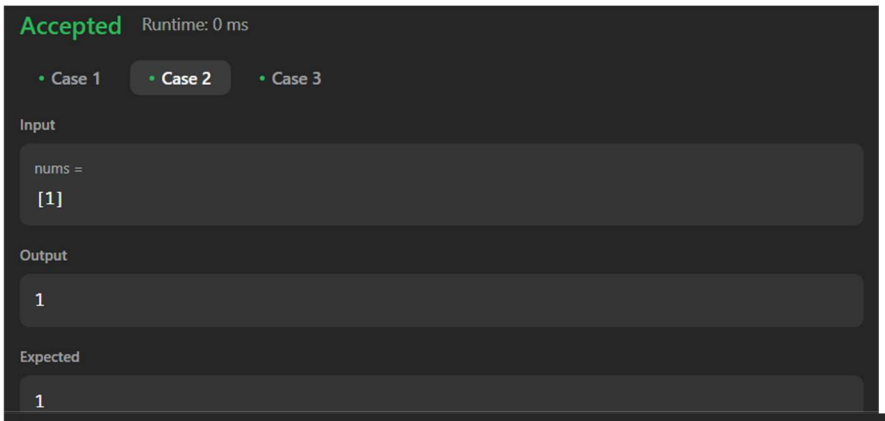
2. **Maximum Product Subarray:**

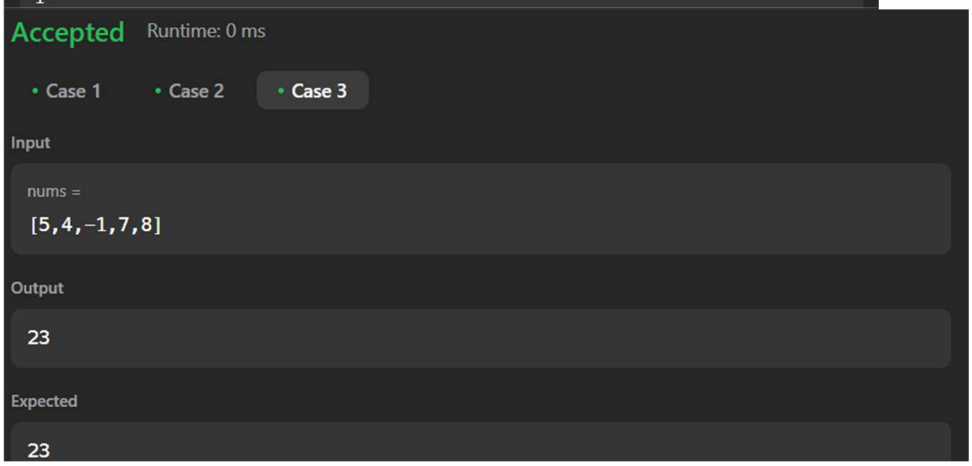
```
from typing import List

class Solution:
    def maxProduct(self, nums: List[int]) -> int:
        if not nums:
            return 0

        max_product = nums[0]
```

```
    min_product = nums[0]
    result = nums[0]

    for i in range(1, len(nums)):
        num = nums[i]

        # If num is negative, swap max and min products
        if num < 0:
            max_product, min_product = min_product, max_product

        # Calculate max/min product at the current position
        max_product = max(num, max_product * num)
        min_product = min(num, min_product * num)

        # Update the result
        result = max(result, max_product)

    return result
```

4. **Output:**
    1. **Maximum Subarray:**







2. Maximum Product Subarray:

5. Learning Outcome

1) Learn how to find the maximum product subarray efficiently.
2) Understand the role of negative numbers in product calculations.
3) Solve the problem in **O(n)** time using a single pass.
4) Handle edge cases like zeros and negative numbers.
5) Apply dynamic programming concepts to track min and max products.