

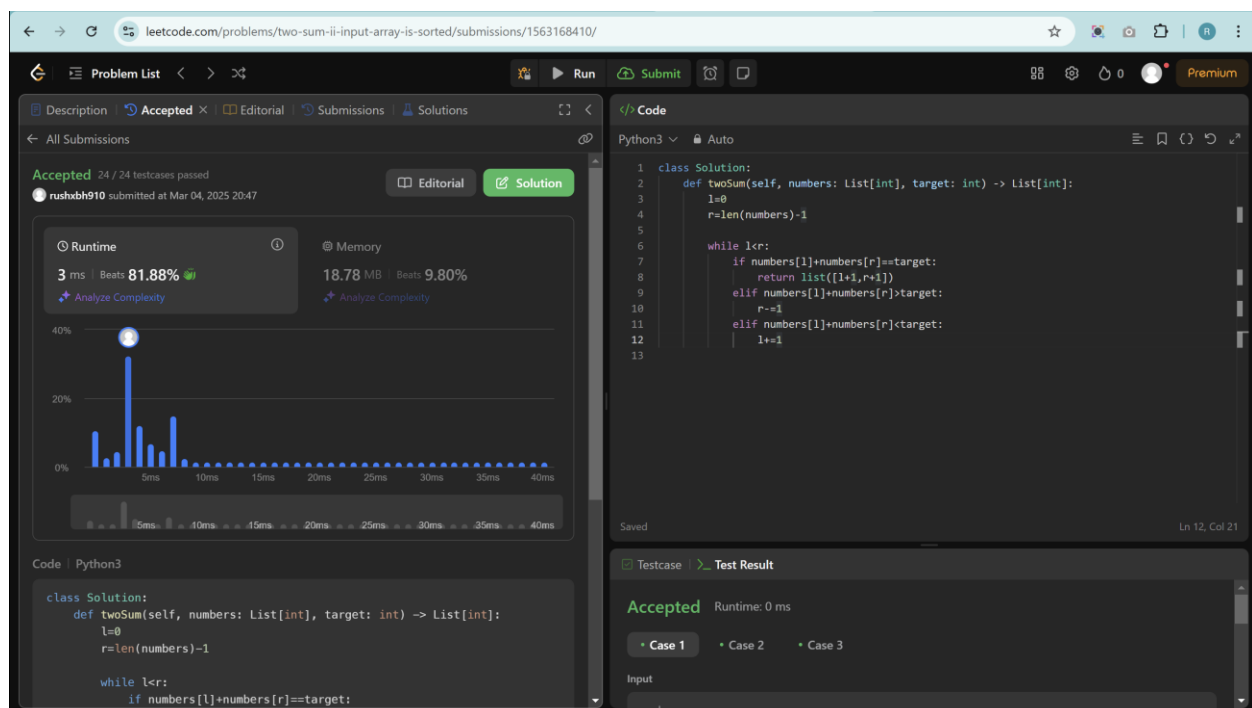
## Homework 1

### Week 1

#### 1. Two Sum II - Input Array Is Sorted:

```
class Solution:
    def twoSum(self, numbers: List[int], target: int) -> List[int]:
        l=0
        r=len(numbers)-1

        while l<r:
            if numbers[l]+numbers[r]==target:
                return list([l+1,r+1])
            elif numbers[l]+numbers[r]>target:
                r-=1
            elif numbers[l]+numbers[r]<target:
                l+=1
```



The screenshot displays a LeetCode submission for the 'twoSum' problem. The submission is 'Accepted' with a runtime of 3 ms and memory usage of 18.78 MB. A bar chart shows the runtime distribution, with a peak at 3 ms. The code editor shows a Python solution using a two-pointer approach. The test result panel shows the input as [2, 7, 11, 15] and target as 9, with the output [1, 2] matching the expected result.

**Runtime Performance:**

- Runtime: 3 ms | Beats: 81.88%
- Memory: 18.78 MB | Beats: 9.80%

**Code (Python3):**

```
class Solution:
    def twoSum(self, numbers: List[int], target: int) -> List[int]:
        l=0
        r=len(numbers)-1

        while l<r:
            if numbers[l]+numbers[r]==target:
```

**Test Result:**

- Case 1: Input: numbers = [2, 7, 11, 15], target = 9. Output: [1, 2]. Expected: [1, 2].

## 2. Products of array discluding self:

```
class Solution:
    def productExceptSelf(self, nums: List[int]) -> List[int]:
        output = [1] * len(nums)

        left = 1
        for i in range(len(nums)):
            output[i] *= left
            left *= nums[i]

        right = 1
        for i in range(len(nums) - 1, -1, -1):
            output[i] *= right
            right *= nums[i]

        return output
```

Accepted 24 / 24 testcases passed  
 Submitted at Mar 04, 2025 21:04

Runtime: 23 ms | Beats 64.51%  
 Memory: 23.09 MB | Beats 97.91%

Code | Python3

```
class Solution:
    def productExceptSelf(self, nums: List[int]) -> List[int]:
        output = [1] * len(nums)

        left = 1
        for i in range(len(nums)):
            output[i] *= left
            left *= nums[i]

        right = 1
        for i in range(len(nums) - 1, -1, -1):
            output[i] *= right
            right *= nums[i]

        return output
```

Testcase | Test Result

Accepted Runtime: 0 ms

Case 1 Case 2

Input

nums = [1,2,3,4]

Output

[24,12,8,6]

### 3. Sort Colors

```
class Solution:
    def sortColors(self, nums: List[int]) -> None:
        """
        Do not return anything, modify nums in-place instead.
        """
        zeros, ones, n = 0, 0, len(nums)
        for num in nums:
            if num == 0:
                zeros += 1
            elif num == 1:
                ones += 1

        for i in range(0, zeros):
            nums[i] = 0

        for i in range(zeros, zeros + ones):
            nums[i] = 1

        for i in range(zeros + ones, n):
            nums[i] = 2
```

The screenshot displays a LeetCode submission page for a problem. The interface is divided into several sections:

- Problem List:** Located at the top left, it includes navigation icons and tabs for Description, Editorial, Submissions, Solutions, and Accepted.
- Accepted Submissions:** A section showing the submission status as "Accepted" with 89/89 testcases passed. It lists the user "rushabh910" and the submission time "Mar 04, 2025 22:23".
- Runtime and Memory:** A performance summary showing a runtime of 0 ms (Beats 100.00%) and memory usage of 17.77 MB (Beats 52.80%).
- Bar Chart:** A chart showing the distribution of runtime performance across different percentiles.
- Code Editor:** A Python3 code editor showing the solution code. The code is a class `Solution` with a method `sortColors` that sorts an array of integers in-place based on the number of zeros, ones, and twos.
- Testcase and Test Result:** A section showing the test results for the submission. It indicates that the submission is "Accepted" with a runtime of 0 ms. The input is `nums = [2,0,1]` and the output is `[0,1,2]`.

```
class Solution:
    def sortColors(self, nums: List[int]) -> None:
        """
        Do not return anything, modify nums in-place instead.
        """
        zeros, ones, n = 0, 0, len(nums)
        for num in nums:
            if num == 0:
                zeros += 1
            elif num == 1:
                ones += 1
        for i in range(0, zeros):
            nums[i] = 0
        for i in range(zeros, zeros + ones):
            nums[i] = 1
        for i in range(zeros + ones, n):
            nums[i] = 2
```