Description | Accepted | Editorial | Solutions | Submissions

80. Remove Duplicates from Sorted Array II

Solved

Medium | Topics | Companies

Given an integer array `nums` sorted in **non-decreasing order**, remove some duplicates in-place such that each unique element appears **at most twice**. The **relative order** of the elements should be kept the **same**.

Since it is impossible to change the length of the array in some languages, you must instead have the result be placed in the **first part** of the array `nums`. More formally, if there are `k` elements after removing the duplicates, then the first `k` elements of `nums` should hold the final result. It does not matter what you leave beyond the first `k` elements.

Return `k` *after placing the final result in the first* `k` *slots of* `nums`.

Do **not** allocate extra space for another array. You must do this by **modifying the input array** in-place with O(1) extra memory.

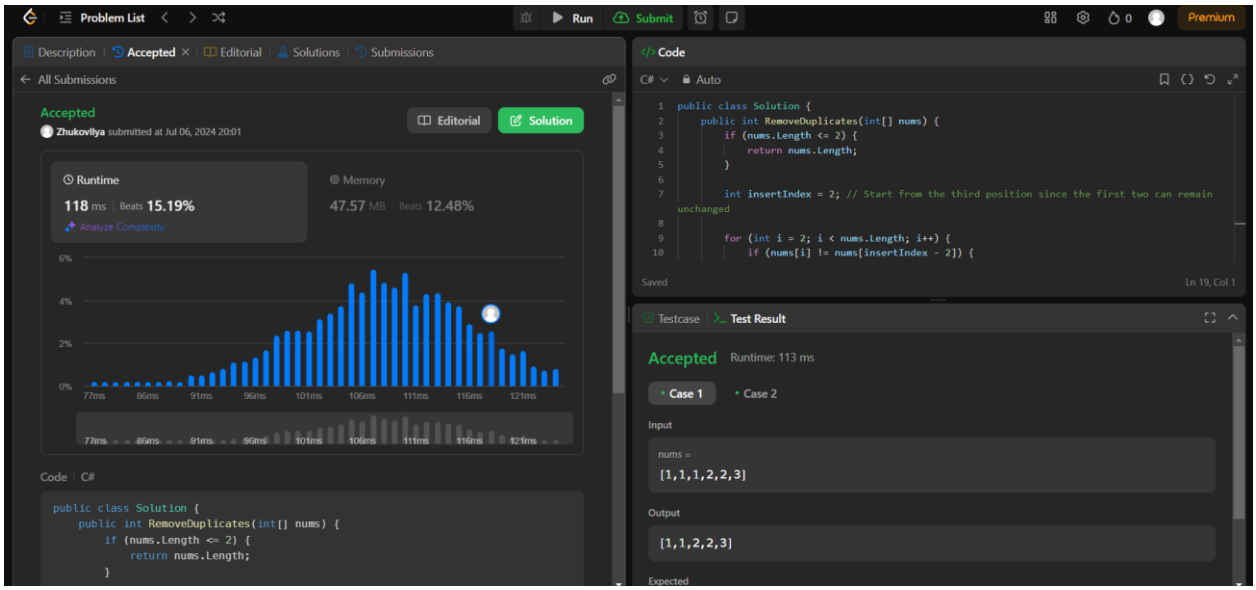
Custom Judge:

The judge will test your solution with the following code:

```
int[] nums = [...]; // Input array
int[] expectedNums = [...]; // The expected answer with correct length

int k = removeDuplicates(nums); // Calls your implementation

assert k == expectedNums.length;
for (int i = 0; i < k; i++) {
    assert nums[i] == expectedNums[i];
}
```



Код:

```
public class Solution
{
    public int RemoveDuplicates(int[] nums)
    {
        if (nums.Length <= 2)
```

```
        {
            return nums.Length;
        }

        int insertIndex = 2; // Start from the third position since the first two
can remain unchanged

        for (int i = 2; i < nums.Length; i++)
        {
            if (nums[i] != nums[insertIndex - 2])
            {
                nums[insertIndex] = nums[i];
                insertIndex++;
            }
        }

        return insertIndex;
    }
}
```