


## 645. Set Mismatch

Solved 

Easy

Topics

Companies

You have a set of integers `s`, which originally contains all the numbers from `1` to `n`. Unfortunately, due to some error, one of the numbers in `s` got duplicated to another number in the set, which results in **repetition of one** number and **loss of another** number.

You are given an integer array `nums` representing the data status of this set after the error.

Find the number that occurs twice and the number that is missing and return *them in the form of an array*.

### Example 1:

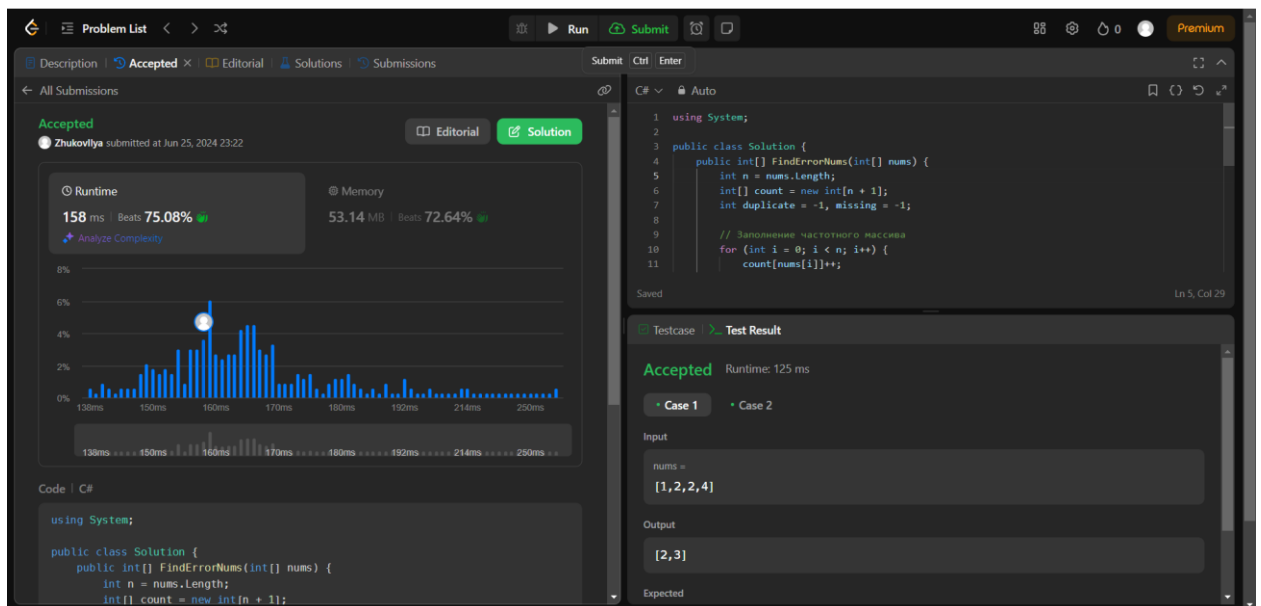
**Input:** `nums = [1,2,2,4]`

**Output:** `[2,3]`

### Example 2:

**Input:** `nums = [1,1]`

**Output:** `[1,2]`



The screenshot shows a code editor with the following C# code:

```
using System;

public class Solution {
    public int[] FindErrorNums(int[] nums) {
        int n = nums.Length;
        int[] count = new int[n + 1];
        int duplicate = -1, missing = -1;

        // Заполнение частотного массива
        for (int i = 0; i < n; i++) {
            count[nums[i]]++;
        }
    }
}
```

The test results show the solution is accepted for the provided test case:

Case	Input	Output	Expected
Case 1	<code>nums = [1,2,2,4]</code>	<code>[2,3]</code>	<code>[2,3]</code>

Код:

```
using System;

public class Solution
{
    public int[] FindErrorNums(int[] nums)
    {
        int n = nums.Length;
        int[] count = new int[n + 1];
        int duplicate = -1, missing = -1;

        // Заполнение частотного массива
        for (int i = 0; i < n; i++)
```

```
{
    count[nums[i]]++;
}

// Поиск дублирующегося и недостающего чисел
for (int i = 1; i <= n; i++)
{
    if (count[i] == 2)
    {
        duplicate = i;
    }
    else if (count[i] == 0)
    {
        missing = i;
    }
}

return new int[] { duplicate, missing };
}
```