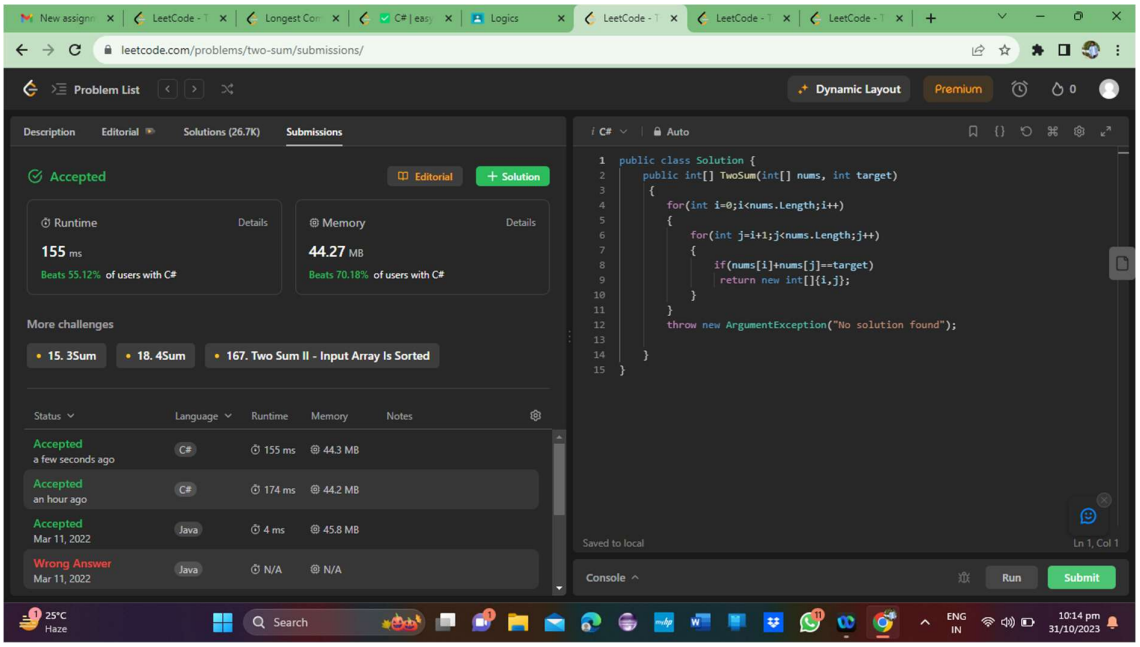
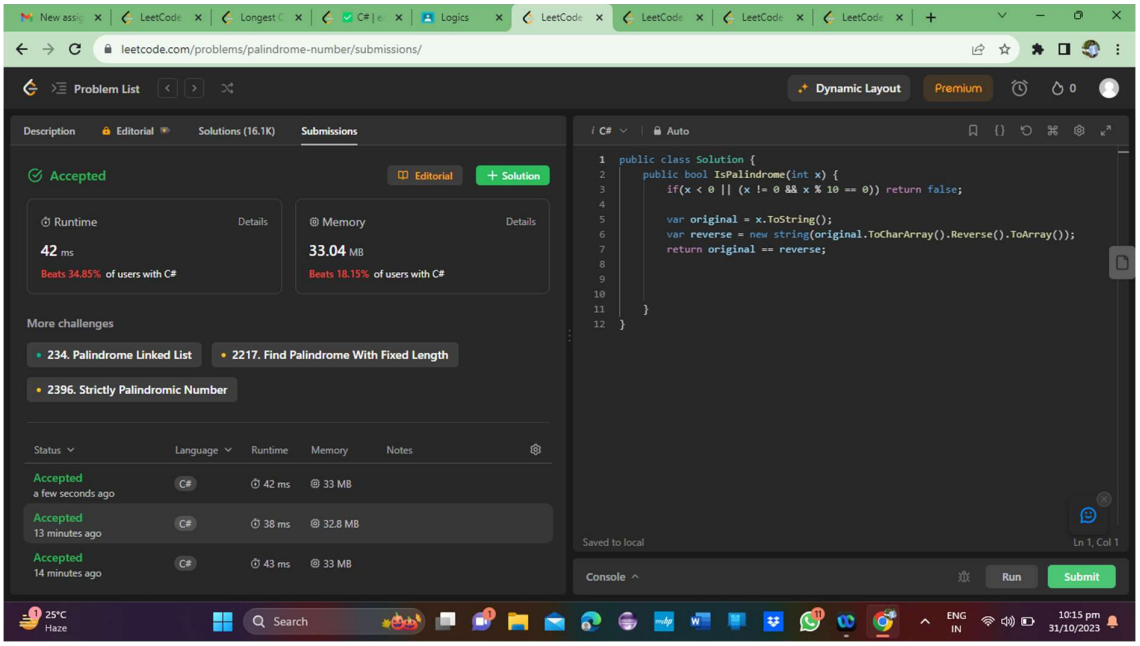
1.

```
public class Solution {
    public int[] TwoSum(int[] nums, int target)
     {
        for(int i=0;i<nums.Length;i++)
        {
            for(int j=i+1;j<nums.Length;j++)
            {
                if(nums[i]+nums[j]==target)
                 return new int[]{i,j};
            }
        }
        throw new ArgumentException("No solution found");

    }
}
```



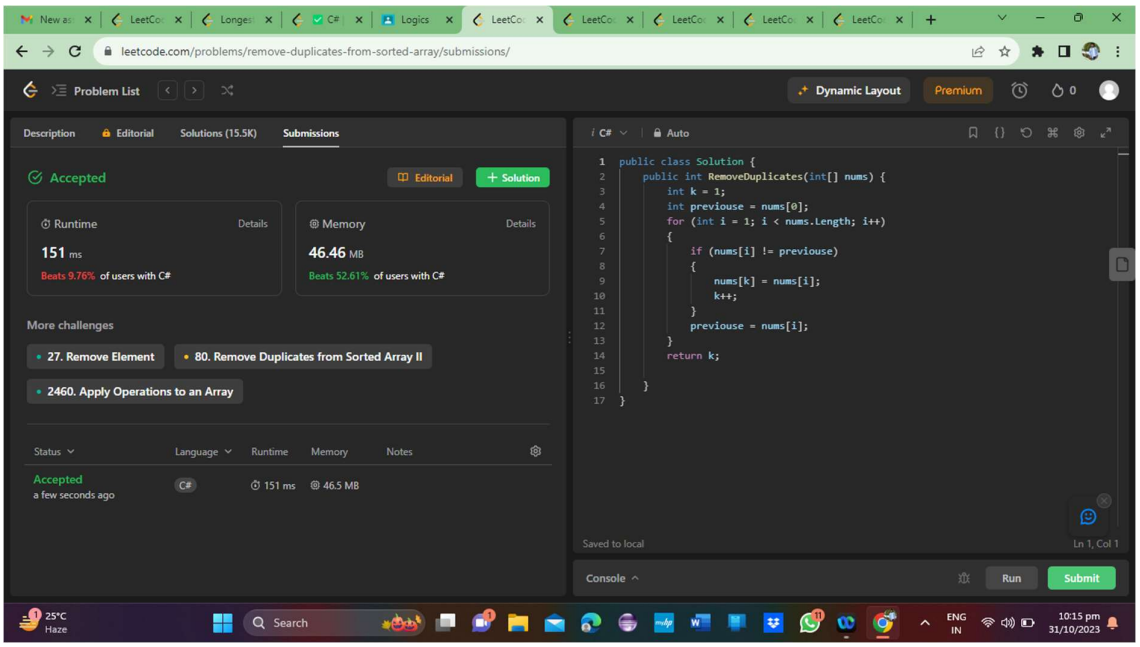
2.

```
ublic class Solution {
    public bool IsPalindrome(int x) {
        if(x < 0 || (x != 0 && x % 10 == 0)) return false;

        var original = x.ToString();
        var reverse = new string(original.ToCharArray().Reverse().ToArray());
        return original == reverse;



    }
}
```



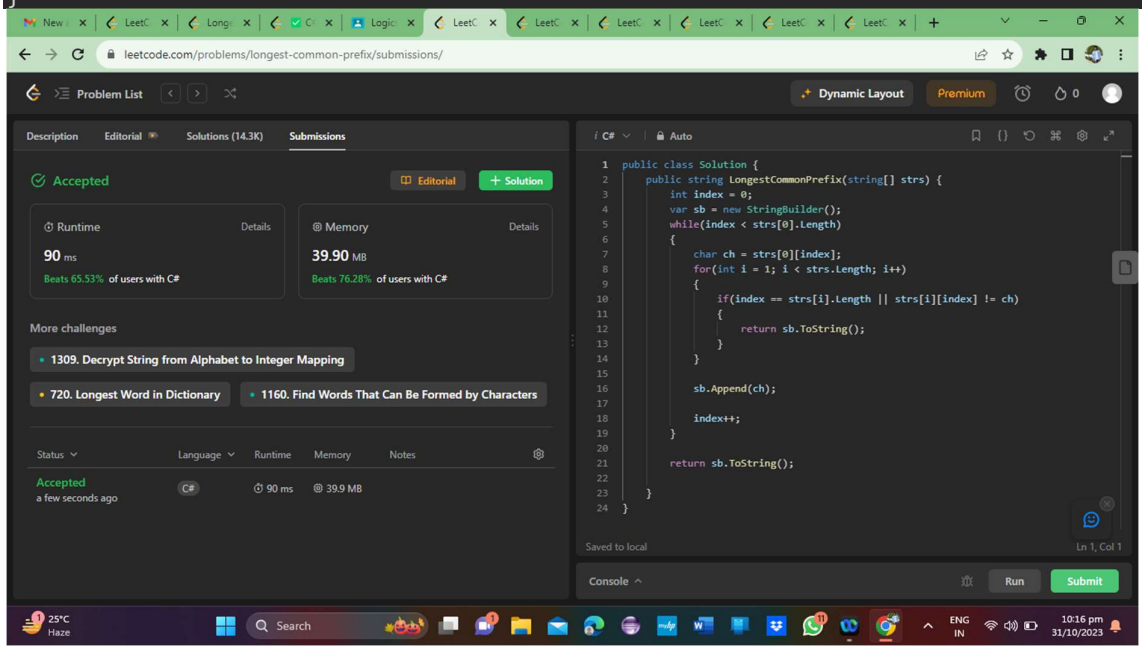
3.

```
public class Solution {
    public int RemoveDuplicates(int[] nums) {
        int k = 1;
        int previouse = nums[0];
        for (int i = 1; i < nums.Length; i++)
        {
            if (nums[i] != previouse)
            {
                nums[k] = nums[i];
                k++;
            }
            previouse = nums[i];
        }
        return k;

    }
}
```



4.

```
public class Solution {
    public string LongestCommonPrefix(string[] strs) {
        int index = 0;
        var sb = new StringBuilder();
        while(index < strs[0].Length)
        {
            char ch = strs[0][index];
            for(int i = 1; i < strs.Length; i++)
            {
                if(index == strs[i].Length || strs[i][index] != ch)
                {
                    return sb.ToString();
                }
            }

            sb.Append(ch);

            index++;
        }

        return sb.ToString();

    }
}
```



5.

```
ublic class Solution {
    public bool IsMatch(string s, string p) {
        bool[,] dp = new bool[s.Length + 1, p.Length + 1];
    dp[s.Length, p.Length] = true;

    for (int i = s.Length; i >= 0; i--){
        for (int j = p.Length - 1; j >= 0; j--){
            bool first_match = (i < s.Length &&
                                (p[j] == s[i] ||
                                 p[j] == '.'));
            if (j + 1 < p.Length && p[j+1] == '*'){
                dp[i,j] = dp[i,j+2] || first_match && dp[i+1,j];
            } else {
                dp[i,j] = first_match && dp[i+1,j+1];
            }
        }
    }
    return dp[0,0];

    }
}
```