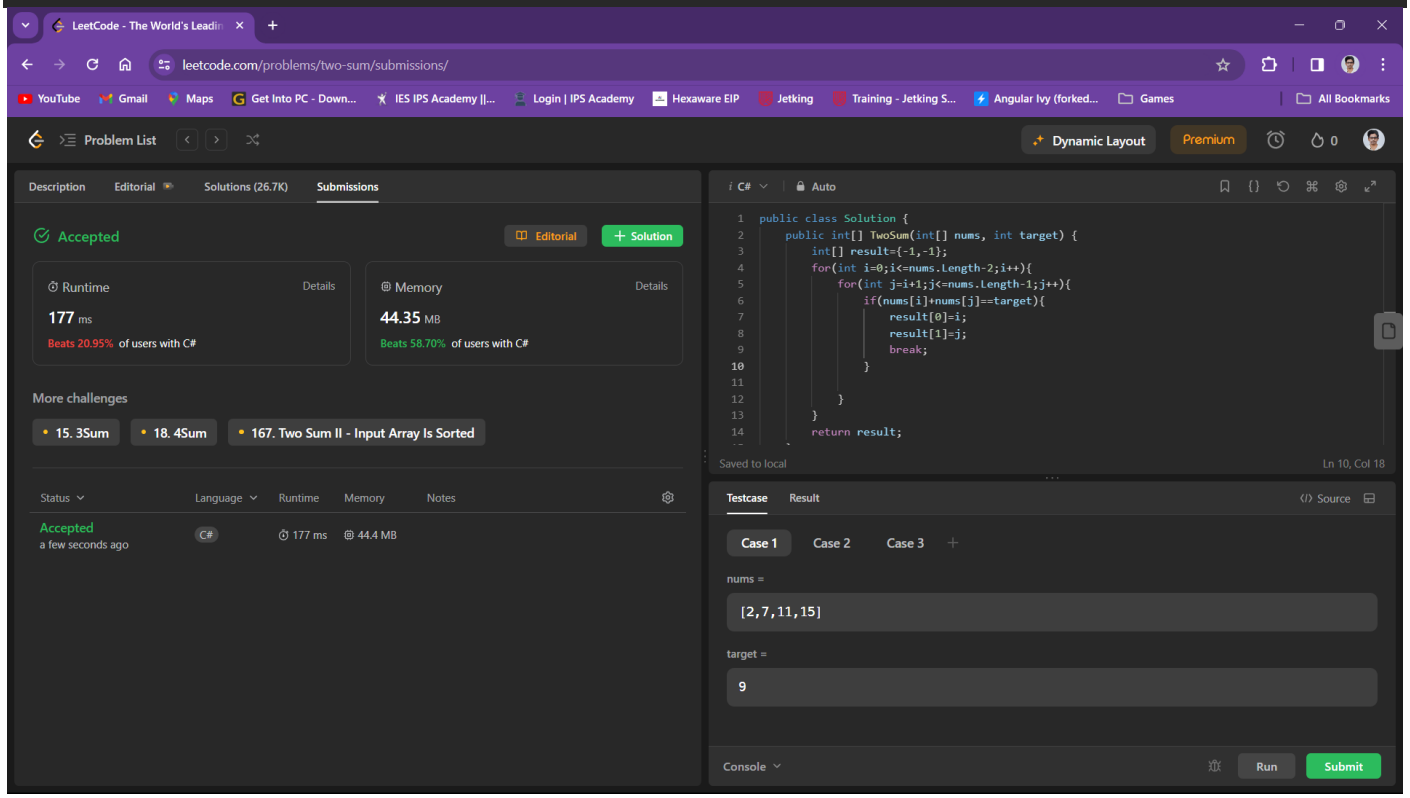
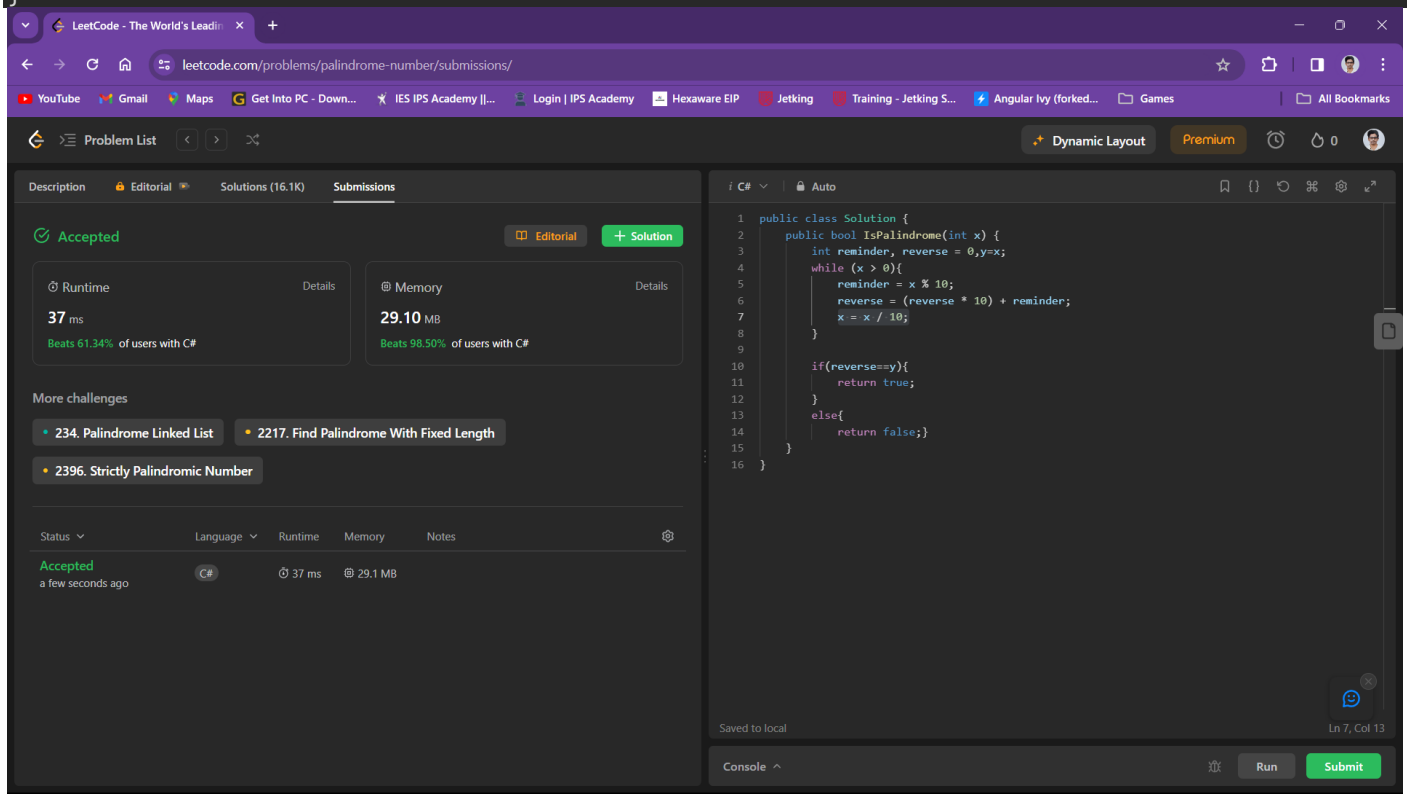
1.Two Sum

```
public class Solution {
    public int[] TwoSum(int[] nums, int target) {
        int[] result={-1,-1};
        for(int i=0;i<=nums.Length-2;i++){
            for(int j=i+1;j<=nums.Length-1;j++){
                if(nums[i]+nums[j]==target){
                    result[0]=i;
                    result[1]=j;
                    break;
                }

            }
        }
        return result;
    }
}
```



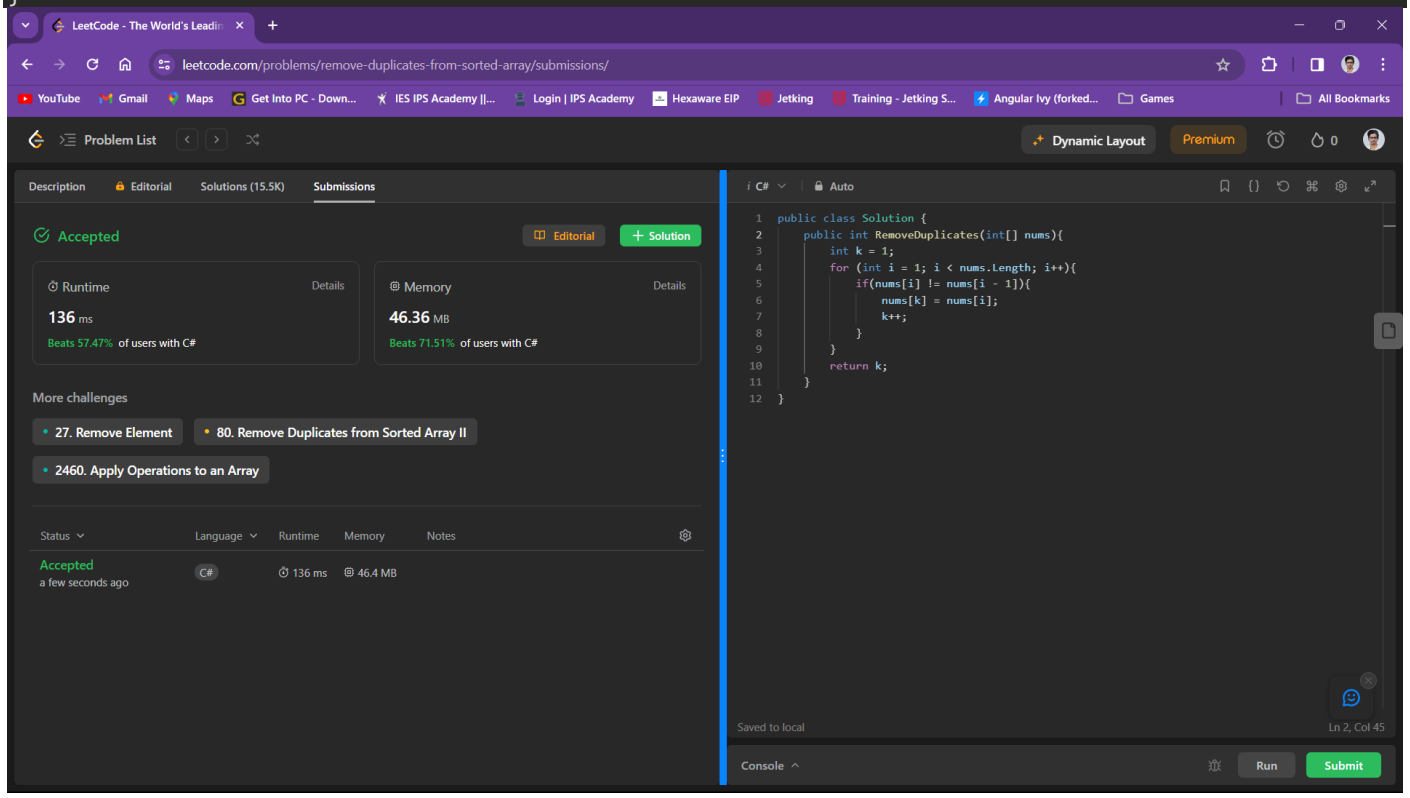
2. Palindrome Number

```
public class Solution {
    public bool IsPalindrome(int x) {
        int reminder, reverse = 0,y=x;
        while (x > 0){
            reminder = x % 10;
            reverse = (reverse * 10) + reminder;
            x = x / 10;
        }

        if(reverse==y){
            return true;
        }
        else{
            return false;}
    }
}
```



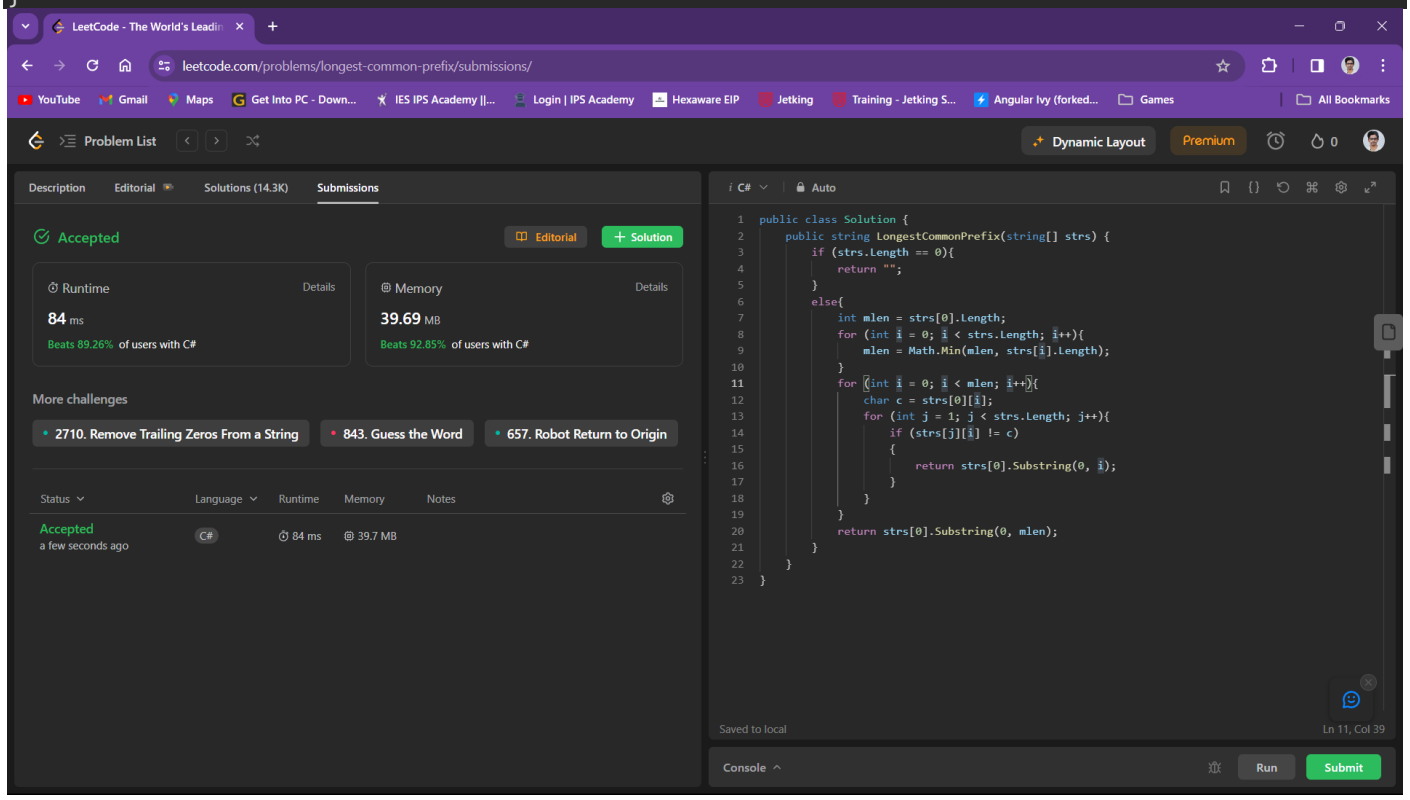
3. Remove Duplicates from Sorted Array

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



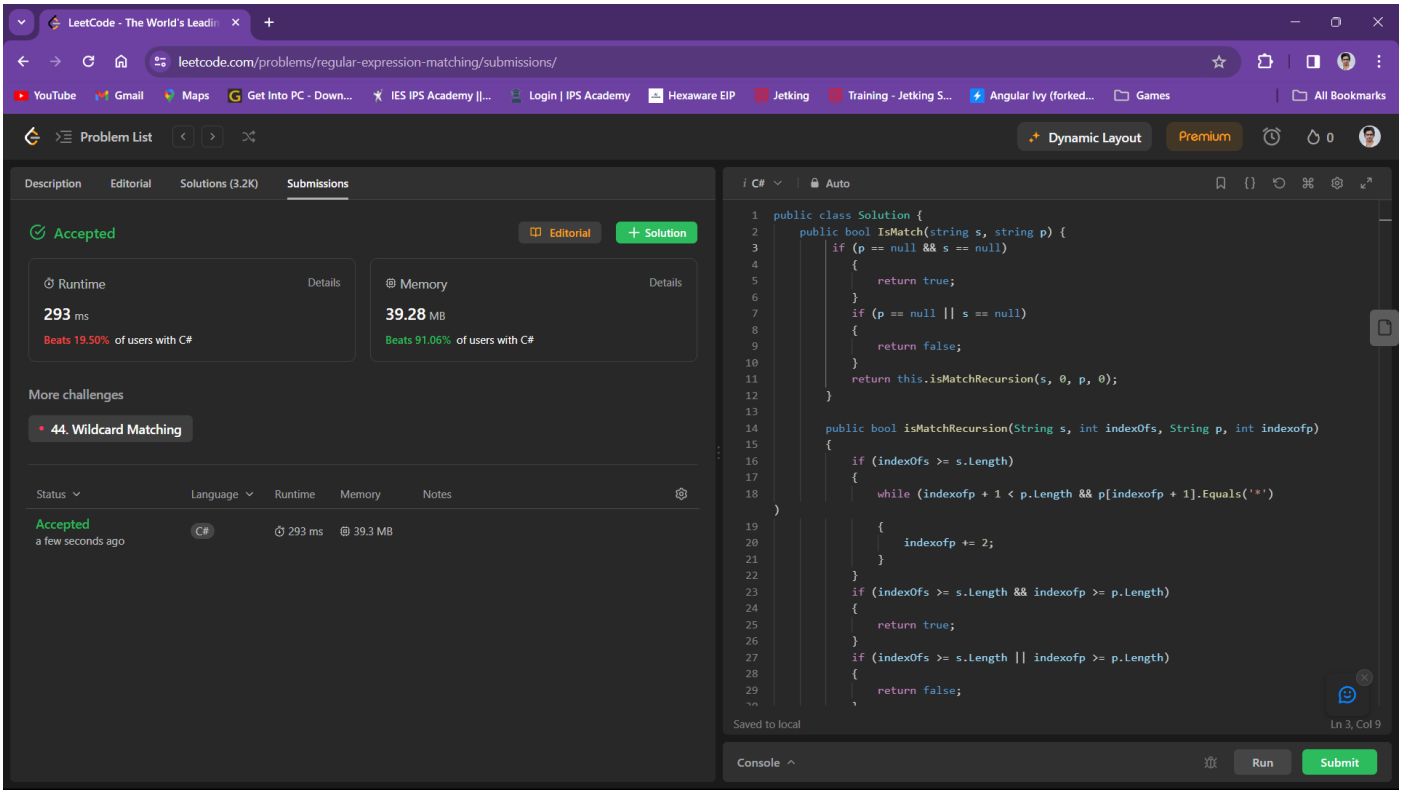
4.Longest Common Prefix

```csharp
public class Solution {
    public string LongestCommonPrefix(string[] strs) {
        if (strs.Length == 0){
            return "";
        }
        else{
            int mlen = strs[0].Length;
            for (int i = 0; i < strs.Length; i++){
                mlen = Math.Min(mlen, strs[i].Length);
            }
            for (int i = 0; i < mlen; i++){
                char c = strs[0][i];
                for (int j = 1; j < strs.Length; j++){
                    if (strs[j][i] != c)
                    {
                        return strs[0].Substring(0, i);
                    }
                }
            }
            return strs[0].Substring(0, mlen);
        }
    }
}
```



5. Regular Expression Matching

```
public class Solution {
    public bool IsMatch(string s, string p) {
         if (p == null && s == null)
            {
                return true;
            }
            if (p == null || s == null)
            {
                return false;
            }
            return this.isMatchRecursion(s, 0, p, 0);
        }

        public bool isMatchRecursion(String s, int indexOfs, String p, int indexofp)
        {
            if (indexOfs >= s.Length)
            {
                while (indexofp + 1 < p.Length && p[indexofp + 1].Equals('*'))
                {
                    indexofp += 2;
                }
            }
            if (indexOfs >= s.Length && indexofp >= p.Length)
            {
                return true;
            }
            if (indexOfs >= s.Length || indexofp >= p.Length)
            {
                return false;
            }

            var next = indexofp + 1 >= p.Length ? ' ' : p[indexofp + 1];

            if (next.Equals('*'))
            {
                if (s[indexOfs].Equals(p[indexofp]) || p[indexofp].Equals('.'))
                {
                    return this.isMatchRecursion(s, indexOfs + 1, p, indexofp)

                        || this.isMatchRecursion(s, indexOfs, p, indexofp + 2);
                }

                return this.isMatchRecursion(s, indexOfs, p, indexofp + 2);
            }

            if (s[indexOfs].Equals(p[indexofp]) || p[indexofp].Equals('.'))
            {
                return this.isMatchRecursion(s, indexOfs + 1, p, indexofp + 1);
            }

            return false;
    }
}
```


LeetCode - The World's Leadin
leetcode.com/problems/regular-expression-matching/submissions/
YouTube
Gmail
Maps
Get Into PC - Down...
IES IPS Academy ||...
Login | IPS Academy
Hexaware EIP
Jetking
Training - Jetking S...
Angular Ivy (forked...
Games
All Bookmarks
Problem List
Dynamic Layout
Premium
Description
Editorial
Solutions (3.2K)
Submissions
Accepted
Editorial
Solution
Runtime
Details
293 ms
Beats 19.50% of users with C#
Memory
Details
39.28 MB
Beats 91.06% of users with C#
More challenges
44. Wildcard Matching
Status
Language
Runtime
Memory
Notes
Accepted
a few seconds ago
C#
293 ms
39.3 MB
C#
Auto
public class Solution {
public bool IsMatch(string s, string p) {
if (p == null && s == null)
{
return true;
}
if (p == null || s == null)
{
return false;
}
return this.isMatchRecursion(s, 0, p, 0);
}
public bool isMatchRecursion(String s, int indexOfs, String p, int indexofp)
{
if (indexOfs >= s.Length)
{
while (indexofp + 1 < p.Length && p[indexofp + 1].Equals('*')
)
{
indexofp += 2;
}
}
if (indexOfs >= s.Length && indexofp >= p.Length)
{
return true;
}
if (indexOfs >= s.Length || indexofp >= p.Length)
{
return false;
Saved to local
Ln 3, Col 9
Console
Run
Submit