

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;
    }
}
```

The screenshot shows the LeetCode interface for the 'Two Sum' problem. The left sidebar contains the 'Problem List' and 'Submissions' tabs. The main content area shows the problem description, the accepted solution, and the runtime details. The right sidebar shows the test cases and the 'Run' button.

Accepted Editorial + Solution

Runtime Details
177 ms
Beats 20.95% of users with C#

Memory Details
44.35 MB
Beats 58.70% of users with C#

More challenges

- 15. 3Sum
- 18. 4Sum
- 167. Two Sum II - Input Array Is Sorted

Status Language Runtime Memory Notes

Accepted
a few seconds ago C# 177 ms 44.4 MB

Testcase **Result** </> Source

Case 1 **Case 2** **Case 3** +

nums =
[2, 7, 11, 15]

target =
9

Console Run Submit

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;}  
    }  
}
```

The screenshot shows a web browser with the URL `leetcode.com/problems/palindrome-number/submissions/`. The page displays the submission details for the 'Palindrome Number' problem. The submission is marked as 'Accepted' with a green checkmark. The runtime is 37 ms, and the memory usage is 29.10 MB. The code is written in C# and is displayed in the editor on the right. The code is as follows:

```
1 public class Solution {  
2     public bool IsPalindrome(int x) {  
3         int reminder, reverse = 0,y=x;  
4         while (x > 0){  
5             reminder = x % 10;  
6             reverse = (reverse * 10) + reminder;  
7             x = x / 10;  
8         }  
9  
10        if(reverse==y){  
11            return true;  
12        }  
13        else{  
14            return false;}  
15        }  
16    }
```

The page also shows a list of more challenges, including '234. Palindrome Linked List', '2217. Find Palindrome With Fixed Length', and '2396. Strictly Palindromic Number'. The submission status is 'Accepted' and it was submitted 'a few seconds ago'.

3. Remove Duplicates from Sorted Array

```
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;  
    }  
}
```

The screenshot shows a web browser with the LeetCode website open. The URL is leetcode.com/problems/remove-duplicates-from-sorted-array/submissions/. The page displays the submission details for the problem "Remove Duplicates from Sorted Array". The submission is marked as "Accepted". The runtime is 136 ms, which beats 57.47% of users with C#. The memory usage is 46.36 MB, which beats 71.51% of users with C#. The code is written in C# and is displayed in the editor on the right. The code is as follows:

```
1 public class Solution {  
2     public int RemoveDuplicates(int[] nums){  
3         int k = 1;  
4         for (int i = 1; i < nums.Length; i++){  
5             if(nums[i] != nums[i - 1]){  
6                 nums[k] = nums[i];  
7                 k++;  
8             }  
9         }  
10        return k;  
11    }  
12 }
```

The browser's address bar shows the URL. The page has a dark theme. The submission details are shown on the left, and the code editor is on the right. The code is written in C# and is displayed in the editor on the right. The code is as follows:

4.Longest Common Prefix

```
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);
        }
    }
}
```

LeetCode - The World's Leading... x +

leetcode.com/problems/longest-common-prefix/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 0 0

Description Editorial Solutions (14.3K) Submissions

Accepted Editorial + Solution

Runtime Details Memory Details

84 ms Beats 89.26% of users with C#

39.69 MB Beats 92.85% of users with C#

More challenges

2710. Remove Trailing Zeros From a String 843. Guess the Word 657. Robot Return to Origin

Status Language Runtime Memory Notes

Accepted a few seconds ago C# 84 ms 39.7 MB

```
1 public class Solution {
2     public string LongestCommonPrefix(string[] strs) {
3         if (strs.Length == 0){
4             return "";
5         }
6         else{
7             int mlen = strs[0].Length;
8             for (int i = 0; i < strs.Length; i++){
9                 mlen = Math.Min(mlen, strs[i].Length);
10            }
11            for (int i = 0; i < mlen; i++){
12                char c = strs[0][i];
13                for (int j = 1; j < strs.Length; j++){
14                    if (strs[j][i] != c)
15                    {
16                        return strs[0].Substring(0, i);
17                    }
18                }
19            }
20            return strs[0].Substring(0, mlen);
21        }
22    }
23 }
```

Saved to local Ln 11, Col 39

Console Run Submit

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 index0fs, String p, int indexofp)
    {
        if (index0fs >= s.Length)
        {
            while (indexofp + 1 < p.Length && p[indexofp + 1].Equals('*'))
            {
                indexofp += 2;
            }
        }
        if (index0fs >= s.Length && indexofp >= p.Length)
        {
            return true;
        }
        if (index0fs >= s.Length || indexofp >= p.Length)
        {
            return false;
        }

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

        if (next.Equals('*'))
        {
            if (s[index0fs].Equals(p[indexofp]) || p[indexofp].Equals('.'))
            {
                return this.IsMatchRecursion(s, index0fs + 1, p, indexofp)

                    || this.IsMatchRecursion(s, index0fs, p, indexofp + 2);
            }

            return this.IsMatchRecursion(s, index0fs, p, indexofp + 2);
        }

        if (s[index0fs].Equals(p[indexofp]) || p[indexofp].Equals('.'))
        {
            return this.IsMatchRecursion(s, index0fs + 1, p, indexofp + 1);
        }

        return false;
    }
}
```

LeetCode - The World's Leading
leetcode.com/problems/regular-expression-matching/submissions/

YouTube Gmail Maps Get Into PC - Down... IES IPS Academy [L... Login | IPS Academy Hexaware EIP Jetking Training - Jetking S... Angular Ivy (forked... Games All Bookmarks

Problem List Dynamic Layout Premium 0

Accepted

Editorial + Solution

Runtime

293 ms

Beats 19.50% of users with C#

Memory

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	

i C# Auto

```
1 public class Solution {
2     public bool IsMatch(string s, string p) {
3         if (p == null && s == null)
4         {
5             return true;
6         }
7         if (p == null || s == null)
8         {
9             return false;
10        }
11        return this.IsMatchRecursion(s, 0, p, 0);
12    }
13
14    public bool isMatchRecursion(String s, int indexofs, String p, int indexofp)
15    {
16        if (indexofs >= s.Length)
17        {
18            while (indexofp + 1 < p.Length && p[indexofp + 1].Equals('*'))
19            {
20                indexofp += 2;
21            }
22        }
23        if (indexofs >= s.Length && indexofp >= p.Length)
24        {
25            return true;
26        }
27        if (indexofs >= s.Length || indexofp >= p.Length)
28        {
29            return false;
30        }
31    }
32 }
```

Saved to local

Ln 3, Col 9

Console Run Submit