

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");
    }
}
```

The screenshot shows the LeetCode submission page for the "Two Sum" problem. The browser address bar displays "leetcode.com/problems/two-sum/submissions/". The page has a dark theme. On the left, the "Submissions" tab is active, showing a submission status of "Accepted". Performance metrics are listed: Runtime is 155 ms (beating 55.12% of users with C#) and Memory is 44.27 MB (beating 70.18% of users with C#). Below this, "More challenges" are suggested, including "15. 3Sum", "18. 4Sum", and "167. Two Sum II - Input Array Is Sorted". A table of recent submissions is visible, with columns for Status, Language, Runtime, and Memory. The first three entries are "Accepted" (C# or Java), and the last one is "Wrong Answer" (Java). On the right, the code editor shows the C# solution code, which matches the code in the first block. The code is:

```
1 public class Solution {
2     public int[] TwoSum(int[] nums, int target)
3     {
4         for(int i=0;i<nums.Length;i++)
5         {
6             for(int j=i+1;j<nums.Length;j++)
7             {
8                 if(nums[i]+nums[j]==target)
9                     return new int[]{i,j};
10            }
11        }
12        throw new ArgumentException("No solution found");
13    }
14 }
15
```

 At the bottom right of the code editor, it says "Saved to local" and "Ln 1, Col 1". Below the code editor is a "Console" area and buttons for "Run" and "Submit". The Windows taskbar is visible at the very bottom, showing the time as 10:14 pm on 31/10/2023.

2.

```
public 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;  
    }  
}
```

The screenshot displays the LeetCode interface for the 'Palindrome Number' problem. The left sidebar shows the problem description and a list of solutions. The main area shows the user's submission, which is 'Accepted'. The submission details include a runtime of 42 ms (beating 34.85% of users with C#) and a memory usage of 33.04 MB (beating 18.15% of users with C#). The code editor on the right shows the C# solution, which is identical to the one in the first block. The bottom of the screen shows the Windows taskbar with various application icons and the system clock indicating 10:15 pm on 31/10/2023.

Status	Language	Runtime	Memory	Notes
Accepted	C#	42 ms	33 MB	a few seconds ago
Accepted	C#	38 ms	32.8 MB	13 minutes ago
Accepted	C#	43 ms	33 MB	14 minutes ago

3.

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

The screenshot shows a web browser with multiple tabs open, including 'LeetCode'. The active tab displays the 'Remove Duplicates from Sorted Array II' problem page. The submission status is 'Accepted'. The runtime is 151 ms, and the memory usage is 46.46 MB. The code is written in C# and matches the code in the previous block. The browser's address bar shows the URL 'leetcode.com/problems/remove-duplicates-from-sorted-array/submissions/'. The bottom of the screen shows a Windows taskbar with various icons and the system clock indicating 10:15 pm on 31/10/2023.

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();
    }
}
```

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

Accepted

Runtime: 90 ms
Beats 65.53% of users with C#

Memory: 39.90 MB
Beats 76.28% of users with C#

More challenges

- 1309. Decrypt String from Alphabet to Integer Mapping
- 720. Longest Word in Dictionary
- 1160. Find Words That Can Be Formed by Characters

Status	Language	Runtime	Memory	Notes
Accepted a few seconds ago	C#	90 ms	39.9 MB	

```
1 public class Solution {
2     public string LongestCommonPrefix(string[] strs) {
3         int index = 0;
4         var sb = new StringBuilder();
5         while(index < strs[0].Length)
6         {
7             char ch = strs[0][index];
8             for(int i = 1; i < strs.Length; i++)
9             {
10                 if(index == strs[i].Length || strs[i][index] != ch)
11                 {
12                     return sb.ToString();
13                 }
14             }
15
16             sb.Append(ch);
17
18             index++;
19         }
20
21         return sb.ToString();
22     }
23 }
24 }
```

Console

Run Submit

25°C Haze

10:16 pm 31/10/2023

5.

```
public 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];
    }
}
```

The screenshot shows a web browser window with the URL leetcode.com/problems/regular-expression-matching/submissions/. The page displays the submission details for the 'Regular Expression Matching' problem. The submission is marked as 'Accepted' with a runtime of 69 ms and memory usage of 39.63 MB. The code is written in C# and matches the code in the previous block. The page also shows a list of other submissions and a table of submission statistics.

Accepted

Runtime: 69 ms (Beats 70.66% of users with C#)

Memory: 39.63 MB (Beats 60.11% of users with C#)

More challenges

44. Wildcard Matching

Status	Language	Runtime	Memory	Notes
Accepted	C#	69 ms	39.6 MB	a few seconds ago
Accepted	C#	67 ms	39.6 MB	4 minutes ago

```
1 public class Solution {
2     public bool IsMatch(string s, string p) {
3         bool[,] dp = new bool[s.Length + 1, p.Length + 1];
4         dp[s.Length, p.Length] = true;
5
6         for (int i = s.Length; i >= 0; i--){
7             for (int j = p.Length - 1; j >= 0; j--){
8                 bool first_match = (i < s.Length &&
9                                     (p[j] == s[i] ||
10                                      p[j] == '.'));
11                 if (j + 1 < p.Length && p[j+1] == '*'){
12                     dp[i,j] = dp[i,j+2] || first_match && dp[i+1,j];
13                 } else {
14                     dp[i,j] = first_match && dp[i+1,j+1];
15                 }
16             }
17         }
18         return dp[0,0];
19     }
20 }
21
```