C# Interview Questions and Answers:

1. Find the repeated number of characters in a String.

using System;

using System.Collections.Generic;

public class HelloWorld

{

public static void Main(string[] args)

{

Console.Write("Enter a string: ");

string input = Console.ReadLine();

// Dictionary to store character counts

Dictionary<char, int> charCount = new Dictionary<char, int>();

foreach (char c in input)

{

if (char.IsWhiteSpace(c)) continue; // Skip spaces

if (charCount.ContainsKey(c))

charCount[c]++;

else

charCount[c] = 1;

}

Console.WriteLine("\nRepeated characters:");

foreach (var pair in charCount)

{

if (pair.Value > 1)

Console.WriteLine($"'{pair.Key}' appears {pair.Value} times");

}

}

1. Reverse String:

Case 1 : Using Array.Reverse();

using System;

class Program

{

static void Main()

{

Console.Write("Enter a string: ");

string input = Console.ReadLine();

// Convert to char array and reverse

char[] charArray = input.ToCharArray();

Array.Reverse(charArray);

// Convert back to string

string reversed = new string(charArray);

Console.WriteLine($"Reversed string: {reversed}");

}

}

Case2: Using Loop

using System;

class Program

{

static void Main()

{

Console.Write("Enter a string: ");

string input = Console.ReadLine();

string reversed = "";

// Loop from end to start

for (int i = input.Length - 1; i >= 0; i--)

{

reversed += input[i];

}

Console.WriteLine($"Reversed string: {reversed}");

}

}

Casr2:Using Linq

using System;

using System.Linq;

class Program

{

static void Main()

{

Console.Write("Enter a string: ");

string input = Console.ReadLine();

string reversed = new string(input.Reverse().ToArray());

Console.WriteLine($"Reversed string: {reversed}");

}

}

Case 3: Using Aggregate() method of Linq

string reversed = input.Aggregate("", (acc, c) => c + acc);

3.1. **Count Number of Characters in a String**

This includes all characters: letters, digits, punctuation, and spaces.

using System;

class Program

{

static void Main()

{

Console.Write("Enter a string: ");

string input = Console.ReadLine();

int charCount = input.Length;

Console.WriteLine($"Total characters: {charCount}");

}

}

2. **Count Number of Words in a Sentence**

Words are typically separated by spaces. We'll split the string and count the resulting parts.

**class Program**

**{**

**static void Main()**

**{**

**Console.Write("Enter a sentence: ");**

**string sentence = Console.ReadLine();**

**// Split by spaces and remove empty entries**

**string[] words = sentence.Split(new[] { ' ' }, StringSplitOptions.RemoveEmptyEntries);**

**int wordCount = words.Length;**

**Console.WriteLine($"Total words: {wordCount}");**

**}**

**4.Part 1: Count How Many Times a Particular Character Appears**

**using System;**

**class Program**

**{**

**static void Main()**

**{**

**Console.Write("Enter a string: ");**

**string input = Console.ReadLine();**

**Console.Write("Enter the character to count: ");**

**char target = Console.ReadLine()[0];**

**int count = 0;**

**foreach (char c in input)**

**{**

**if (c == target)**

**count++;**

**}**

**Console.WriteLine($"Character '{target}' appears {count} times.");**

**}**

**}**

1. Write a C# Program in a string if a particular char repeated 1st time in that string print char its position very next appearance print and come out

Solution:· Takes a string and a character as input.

1. Finds the **first occurrence** of that character.
2. Then finds the **very next appearance** of the same character.
3. Prints both the character and the position of the second appearance.
4. Exits immediately after printing.

**using System;**

**class Program**

**{**

**static void Main()**

**{**

**Console.Write("Enter a string: ");**

**string input = Console.ReadLine();**

**Console.Write("Enter the character to search: ");**

**char target = Console.ReadLine()[0];**

**int firstIndex = -1;**

**for (int i = 0; i < input.Length; i++)**

**{**

**if (input[i] == target)**

**{**

**if (firstIndex == -1)**

**{**

**// First time the character appears**

**firstIndex = i;**

**}**

**else**

**{**

**// Second time it appears — print and exit**

**Console.WriteLine($"Character '{target}' repeated at position {i}.");**

**return;**

**}**

**}**

**}**

**Console.WriteLine($"Character '{target}' does not repeat in the string.");**

**}**

**}**

1. **Check a string is Palindrome:**

C**heck if a String is a Palindrome**

string input = "madam";

string reversed = new string(input.Reverse().ToArray());

if (input == reversed)

Console.WriteLine("Palindrome");

else Console.WriteLine("Not a palindrome");

🔢 3. **Count Vowels and Consonants**

**string input = "Interview";**

**int vowels = 0, consonants = 0;**

**foreach (char c in input.ToLower())**

**{**

**if ("aeiou".Contains(c))**

**vowels++;**

**else if (char.IsLetter(c))**

**consonants++;**

**}**

**Console.WriteLine($"Vowels: {vowels}, Consonants: {consonants}");**

🔁 4. **Remove Duplicate Characters from a String**

string input = "programming";

string result = "";

foreach (char c in input)

{

if (!result.Contains(c))

result += c;

}

Console.WriteLine(result); // Output: "progamin"

5. **Find the First Non-Repeating Character**

string input = "swiss";

foreach (char c in input)

{

if (input.IndexOf(c) == input.LastIndexOf(c))

{

Console.WriteLine($"First non-repeating character: {c}");

break;

}

}

🔄 6. **Swap Two Characters in a String**

string input = "hello";

char[] chars = input.ToCharArray();

// Swap first and last characters

char temp = chars[0];

chars[0] = chars[chars.Length - 1];

chars[chars.Length - 1] = temp;

Output : **Console.WriteLine(new string(chars)); // Output: "oellh"**

🧠 7. **Check if Two Strings Are Anagrams**

tring str1 = "listen";

string str2 = "silent";

bool isAnagram = str1.Length == str2.Length &&

str1.OrderBy(c => c).SequenceEqual(str2.OrderBy(c => c));

Console.WriteLine(isAnagram ? "Anagram" : "Not anagram");

**What Is an Anagram?**

An **anagram** is a word or phrase formed by **rearranging the letters** of another word or phrase, using **all the original letters exactly once**.

🧠 **Simple Definition:**

If two words have the **same letters** in a **different order**, they are anagrams.

✅ **Examples of Anagrams**

| **Word 1** | **Word 2** | **Are They Anagrams?** | **Why?** |
| --- | --- | --- | --- |
| listen | silent | ✅ Yes | Same letters, rearranged |
| race | care | ✅ Yes | Same letters, different order |
| hello | world | ❌ No | Different letters |
| evil | live | ✅ Yes | Same letters, different order |
| night | thing | ✅ Yes | Same letters, rearranged |

**string str1 = "listen";**

**string str2 = "silent";**

**bool isAnagram = str1.Length == str2.Length &&**

**str1.OrderBy(c => c).SequenceEqual(str2.OrderBy(c => c));**

**Console.WriteLine(isAnagram ? "Anagram" : "Not anagram");**