**Strings - Handling strings and common string functions**

**Functions - parameter passing, return**

**1. Count Vowels and Consonants**

**Problem:**

**Write a C# program to count the number of vowels and consonants in a given string.**

| **using System;  public class Solution {  // Method to check if a character is a vowel  public static bool CheckVowel(char ch) {  ch = char.ToLower(ch);  return ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u';  }   public static void Main() {  // Prompt user for string input  Console.WriteLine("Enter String: ");  string givenString = Console.ReadLine();    int vowelCount = 0;    foreach (char ch in givenString) {  if (CheckVowel(ch)) {  vowelCount++;  }  }   //printing result  Console.WriteLine("Number of Vowels is {0}" , vowelCount);  Console.WriteLine("Number of Consonants is {0}" , (givenString.Length - vowelCount));  } }** |
| --- |

**2. Reverse a String**

**Problem:**

**Write a C# program to reverse a given string without using any built-in reverse functions.**

| **using System;  public class Solution {  // Method to check if a character is a vowel  public static string ReverseString(string givenString) {  string revString = "";  for (int i = givenString.Length - 1 ; i>=0 ; i--) {  revString += givenString[i];  }  return revString;  }   public static void Main() {  // Prompt user for string input  Console.WriteLine("Enter String: ");  string givenString = Console.ReadLine();    //printing result  Console.WriteLine("reverse string is {0}" , ReverseString(givenString));  } }** |
| --- |

**3. Palindrome String Check**

**Problem:**

**Write a C# program to check if a given string is a palindrome (a string that reads the same forward and backward).**

| **using System;  public class Solution {  // Method to Reverse String  public static string ReverseString(string givenString) {  string revString = "";  for (int i = givenString.Length - 1 ; i >= 0 ; i--) {  revString += givenString[i];  }  return revString;  }    // Method to check palindrome  public static bool CheckPalindrome(string givenString) {  return givenString == (ReverseString(givenString));  }    public static void Main() {  // Prompt user for string input  Console.WriteLine("Enter String: ");  string givenString = Console.ReadLine();    //printing result  Console.WriteLine("String is palindrome? {0}" , CheckPalindrome(givenString));  } }** |
| --- |

**4. Remove Duplicates from a String**

**Problem:**

**Write a C# program to remove all duplicate characters from a given string and return the modified string.**

| **using System;  class Solution {  // Method to remove Duplicates   public static string RemoveDuplicates(string givenString) {  bool[] characterAscii = new bool[256];  string result = "";    foreach (char ch in givenString) {  if (characterAscii[ch] == false) {  characterAscii[ch] = true;  result += ch;  }  }   return result;  }    public static void Main() {  // Prompt user for string input  Console.WriteLine("Enter String: ");  string givenString = Console.ReadLine();    // Printing result  Console.WriteLine("Removed duplicate string is: {0}", RemoveDuplicates(givenString));  } }** |
| --- |

**5. Find the Longest Word in a Sentence**

**Problem:**

**Write a C# program that takes a sentence as input and returns the longest word in the sentence.**

| **using System;  public class Solution {  // Method to check longest word  public static string LongestWord (string givenString) {  string longest = "";   int longestLength = 0;  string [] words = givenString.Split(' ');  for (int i = 0 ; i < words.Length ; i++) {  if (words[i].Length > longestLength) {   longest = words[i];  longestLength = words[i].Length;   }  }  return longest;  }    public static void Main() {  // Prompt user for string input  Console.WriteLine("Enter String: ");  string givenString = Console.ReadLine();    //printing result  Console.WriteLine("Longest word in string is {0}" , LongestWord(givenString));  } }** |
| --- |

**6. Find Substring Occurrences**

**Problem:**

**Write a C# program to count how many times a given substring occurs in a string.**

| **using System;  public class Solution {  public static int CountSubstringOccurrences(string str, string substr) {  int count = 0;  int strLen = str.Length, substrLen = substr.Length;   for (int i = 0; i <= strLen - substrLen; i++) {  int j;  for (j = 0; j < substrLen; j++) {  if (str[i + j] != substr[j]) {  break;  }  }  if (j == substrLen) {   count++;  //i += substrLen - 1; //if want to count non-overlapping substring occurrence  }   }  return count;  }   public static void Main() {  Console.Write("Enter the main string: ");  string givenString = Console.ReadLine();   Console.Write("Enter the substring: ");  string substr = Console.ReadLine();   Console.WriteLine("Occurrences of {0} in {1}: {2}" , substr , givenString , CountSubstringOccurrences(givenString, substr));  } }** |
| --- |

**7. Toggle Case of Characters**

**Problem:**

**Write a C# program to toggle the case of each character in a given string. Convert uppercase letters to lowercase and vice versa.**

| **using System;  public class Solution {  public static string ToggleCase(string givenString) {  string toggledString = "";  foreach (char ch in givenString) {  if ((int) ch >=97) {  toggledString += (char) ((int) ch - 32) ;  }  else {  toggledString += (char) ((int) ch + 32) ;  }  }  return toggledString;  }   public static void Main() {  Console.Write("Enter the main string: ");  string givenString = Console.ReadLine();   Console.WriteLine("Toggle Case String is {0}" , ToggleCase(givenString));  } }** |
| --- |

**8. Compare Two Strings**

**Problem:**

**Write a C# program to compare two strings lexicographically (dictionary order) without using built-in compare methods.**

**Example Input:**

**String 1: "apple"**

**String 2: "banana"**

**Expected Output:**

**"apple" comes before "banana" in lexicographical order**

| **using System;  public class Solution {  // Method to compare strings  public static string CompareStrings(string str1, string str2) {  int len1 = str1.Length, len2 = str2.Length;  int minLength = Math.Min(len1, len2);    for (int i = 0; i < minLength; i++) {  if (char.ToLower(str1[i]) < char.ToLower(str2[i])) {  return $"{str1} comes before {str2}";   }  if (char.ToLower(str1[i]) > char.ToLower(str2[i])){  return $"{str2} comes before {str1}";  }  }   // If all compared characters are equal, compare lengths  if (len1 < len2) return $"{str1} comes before {str2}";   if (len1 > len2) return $"{str2} comes before {str1}";    return "Both strings are equal.";   }   public static void Main() {  // Prompt user to enter 2 strings  Console.Write("Enter the string 1: ");  string givenString1 = Console.ReadLine();    Console.Write("Enter the string 2: ");  string givenString2 = Console.ReadLine();   Console.WriteLine("{0} in lexicographical order" , CompareStrings(givenString1, givenString2));  } }** |
| --- |

**9. Find the Most Frequent Character**

**Problem:**

**Write a C# program to find the most frequent character in a string. Example Input:**

**String: "success"**

**Expected Output:**

**Most Frequent Character: 's'**

| **using System;  public class Solution {  // Method to find the most frequent character  public static char CheckFrequentCharacter(string givenString) {  int[] characterCount = new int[256];   char mostFrequentChar = givenString[0];  int maxCount = 0;   // Count occurrences of each character  foreach (char ch in givenString) {  characterCount[ch]++;  if (characterCount[ch] > maxCount) {  maxCount = characterCount[ch];  mostFrequentChar = ch;  }  }  return mostFrequentChar;  }    public static void Main() {  // Prompt user for string input  Console.Write("Enter a string: ");  string givenString = Console.ReadLine();   char result = CheckFrequentCharacter(givenString);   // Print result   Console.WriteLine("Most Frequent Character in the string is: '{0}'", result);  } }** |
| --- |

**10. Remove a Specific Character from a String**

**Problem:**

**Write a C# program to remove all occurrences of a specific character from a string. Example Input:**

**String: "Hello World"**

**Character to Remove: 'l'**

**Expected Output:**

**Modified String: "Heo Word"**

| **using System;  public class Solution {  // Method to Remove specific character  public static string RemoveSpecificCharacter (string givenString , char ch) {   string modifiedString = "";   // skip specific character  foreach (char stringChar in givenString) {  if (stringChar != ch) {  modifiedString += stringChar;  }  }  return modifiedString;  }    public static void Main() {  // Prompt user for string input  Console.Write("Enter a string: ");  string givenString = Console.ReadLine();    Console.Write("Enter Character to remove: ");  char ch = Console.ReadKey().KeyChar;   // Print result   Console.WriteLine("\nModified string is: {0}", RemoveSpecificCharacter(givenString , ch));  } }** |
| --- |

**11. Write a C# program that accepts two strings from the user and checks if the two strings are anagrams of each other (i.e., whether they contain the same characters in any order).**

| **using System;  public class Solution {  // Method to check if two strings are anagrams  public static bool CheckAnagrams(string givenString1, string givenString2) {  int len1 = givenString1.Length , len2 = givenString2.Length;  if (len1 != len2) return false;    int[] charCount = new int[256];    // Count character occurrences in str1  for (int i = 0 ; i < len1 ; i++) {  charCount[(int)(givenString1[i])] += 1;  }    // Subtract character occurrences in str2  for (int i = 0 ; i < len2 ; i++) {  charCount[(int)(givenString2[i])] -= 1;  }   // If all values in charCount are zero, they are anagrams  for (int i = 0; i < 256; i++) {  if (charCount[i] != 0) return false;  }   return true;  }   public static void Main() {  // prompt user for 2 string input  Console.WriteLine("Enter first string: ");  string givenString1 = Console.ReadLine();    Console.WriteLine("Enter second string: ");  string givenString2 = Console.ReadLine();    // Check for anagram and print result  if (CheckAnagrams(givenString1 , givenString2)) {  Console.WriteLine("The two strings are anagrams");  }  else {  Console.WriteLine("The two strings are NOT anagrams");  }  } }** |
| --- |

**12. Write a replace method in C# that replaces a given word with another word in a sentence:**

| **using System;  public class Solution {  // Method to replace a word in a sentence  public static string ReplaceWord(string sentence, string target, string replacement) {  int strLen = sentence.Length, targetLen = target.Length;  string result = "";    for (int i = 0; i < strLen; i++) {  int j;    // Checking whether substring matches target  for (j = 0; j < targetLen; j++) {  if (i + j >= strLen || sentence[i + j] != target[j]) {  break;  }  }   // If a full match found  if (j == targetLen) {  result += replacement;  i += targetLen - 1;   } else {  result += sentence[i];   }  }    return result;  }   public static void Main() {  // Prompt user for sentence, target and replacement input  Console.WriteLine("Enter Sentence: ");  string sentence = Console.ReadLine();    Console.WriteLine("Enter the word to replace: ");  string target = Console.ReadLine();   Console.WriteLine("Enter the replacement word: ");  string replacement = Console.ReadLine();    // Print result   Console.WriteLine("Modified string is: {0}", ReplaceWord(sentence, target, replacement));  } }** |
| --- |