1. Second Largest Element in an Array

Given an array of positive integers arr[] of size n, the task is to find second largest distinct element in the array.

Note: If the second largest element does not exist, return -1.

Examples:

Input: arr[] = [12, 35, 1, 10, 34, 1]

Output: 34

Explanation: The largest element of the array is 35 and the second largest element is 34.

Input: arr[] = [10, 5, 10]

Output: 5

Explanation: The largest element of the array is 10 and the second largest element is 5.

Input: arr[] = [10, 10, 10]

Output: -1

Explanation: The largest element of the array is 10 there is no second largest element.

Solution:

```
using System;


public class HelloWorld
{
   static int SecondLargest(int[] arr)
   {
      int largest = int.MinValue;
      int secondLargest = int.MinValue;


      foreach (int num in arr)
      {
         if (num > largest)
```

```
        {
            secondLargest = largest;
            largest = num;
        }
        else if (num > secondLargest && num != largest)
        {
            secondLargest = num;
        }
    }

    // If there's no second largest distinct element, return -1
    if (secondLargest == int.MinValue)
    {
        return -1;
    }

    return secondLargest;
}

public static void Main(string[] args)
{
    Console.WriteLine("Enter the number of elements in the array:");
    int n = int.Parse(Console.ReadLine());

    int[] arr = new int[n];
    Console.WriteLine("Enter the elements of the array:");

    for (int i = 0; i < n; i++)
```

```
    {
        arr[i] = int.Parse(Console.ReadLine());
    }

    int result = SecondLargest(arr);

    if (result == -1)
    {
        Console.WriteLine(-1);
    }
    else
    {
        Console.WriteLine("The second largest distinct element is: " + result);
    }
  }
}
```

2. Maximum consecutive one's (or zeros) in a binary array

Given a binary array, find the count of a maximum number of consecutive 1s present in the array.

Examples :

Input: arr[] = {1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1}

Output: 4

Explanation: The maximum number of consecutive 1's in the array is 4 from index 8-11.

Input: arr[] = {0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1}

Output: 1

Explanation: The maximum number of consecutive 0's in the array is 1 from index 0-1.

Solution:

```
using System;

public class HelloWorld
{
   static int MaxConsecutiveOnes(int[] arr)
   {
      int maxCount = 0;
      int currentCount = 0;

      foreach (int num in arr)
      {
         if (num == 1)
         {
            currentCount++;
            maxCount = Math.Max(maxCount, currentCount);
         }
         else
         {
            currentCount = 0;
         }
      }

      return maxCount;
   }

   public static void Main(string[] args)
   {
      Console.WriteLine("Enter the number of elements in the array:");
```

```
        int n = int.Parse(Console.ReadLine());

        int[] arr = new int[n];
        Console.WriteLine("Enter the elements of the array (0s and 1s):");

        for (int i = 0; i < n; i++)
        {
            arr[i] = int.Parse(Console.ReadLine());
        }

        int result = MaxConsecutiveOnes(arr);
        Console.WriteLine("The maximum number of consecutive 1s is: " + result);
    }
}
```

3. Missing and Repeating in an Array

Given an unsorted array of size n. Array elements are in the range of 1 to n. One number from set {1, 2, ...n} is missing and one number occurs twice in the array. Find these two numbers.

Examples:

Input: arr[] = {3, 1, 3}

Output: Missing = 2, Repeating = 3

Explanation: In the array, 2 is missing and 3 occurs twice

Input: arr[] = {4, 3, 6, 2, 1, 1}

Output: Missing = 5, Repeating = 1

Solution:

```
using System;

public class HelloWorld
```

```
{
  static void FindMissingAndRepeating(int[] arr, int n, out int missing, out int repeating)
  {
    // Calculate the sum of the first n natural numbers
    int sumN = n * (n + 1) / 2;
    int sumSquareN = n * (n + 1) * (2 * n + 1) / 6;

    int sumArr = 0, sumSquareArr = 0;

    foreach (var num in arr)
    {
      sumArr += num;
      sumSquareArr += num * num;
    }

    // Calculate the differences
    int diffSum = sumN - sumArr;
    int diffSquareSum = sumSquareN - sumSquareArr;

    // Derive missing and repeating from the differences
    missing = (diffSum + diffSquareSum / diffSum) / 2;
    repeating = missing - diffSum;
  }
  public static void Main(string[] args)
  {
    Console.WriteLine("Enter the number of elements in the array:");
    int n = int.Parse(Console.ReadLine());
```

```
        int[] arr = new int[n];
        Console.WriteLine("Enter the elements of the array:");

        for (int i = 0; i < n; i++)
        {
            arr[i] = int.Parse(Console.ReadLine());
        }

        FindMissingAndRepeating(arr, n, out int missing, out int repeating);

        Console.WriteLine("Missing number: " + missing);
        Console.WriteLine("Repeating number: " + repeating);
    }
}
```

4. Find duplicate characters in a string

Example:

Here, you can see we have passed "google" as a string, and we got a result as "og". It means these two characters (og)are duplicates in the word "google".

Solution:

```
using System;
using System.Collections.Generic;

class Program
{
    static string FindDuplicateCharacters(string str)
    {
```

```
    var charCount = new Dictionary<char, int>();
    var duplicates = new HashSet<char>();

    // Count the occurrences of each character
    foreach (char ch in str)
    {
        if (charCount.ContainsKey(ch))
        {
            charCount[ch]++;
            duplicates.Add(ch);
        }
        else
        {
            charCount[ch] = 1;
        }
    }

    // Convert the duplicates set to a string
    return new string(string.Join("", duplicates).ToCharArray());
}

static void Main()
{
    Console.WriteLine("Enter a string:");
    string input = Console.ReadLine();

    string result = FindDuplicateCharacters(input);
    Console.WriteLine("Duplicate characters: " + result);
```

```
    }
}
```

5. Get all unique characters in a string

Example:

you can see the output as "gole". This means after removing duplicate characters from the word "google," we get the "gole" which are unique chars.

Solution:

```
using System;
using System.Collections.Generic;


class Program
{
    static string GetUniqueCharacters(string str)
    {
        var seenChars = new HashSet<char>();
        string uniqueChars = "";

        foreach (char ch in str)
        {
            if (!seenChars.Contains(ch))
            {
                uniqueChars += ch;
                seenChars.Add(ch);
            }
        }

        return uniqueChars;
```

```
    }

    static void Main()
    {
        Console.WriteLine("Enter a string:");
        string input = Console.ReadLine();

        string result = GetUniqueCharacters(input);
        Console.WriteLine("Unique characters: " + result);
    }
}
```

6. Reverse each word of the sentence (string)

example: "My name is vasu" output->"yM eman si usav"

Solution:

```
using System;

class Program
{
    static string ReverseWords(string sentence)
    {
        string[] words = sentence.Split(' ');
        for (int i = 0; i < words.Length; i++)
        {
            char[] wordArray = words[i].ToCharArray();
            Array.Reverse(wordArray);
            words[i] = new string(wordArray);
        }
```

```
        return string.Join(" ", words);
    }

    static void Main()
    {
        Console.WriteLine("Enter a sentence:");
        string input = Console.ReadLine();

        string result = ReverseWords(input);
        Console.WriteLine("Reversed words: " + result);
    }
}
```