# **Check Perfect Number**

(Perfect number = sum of proper divisors = number)

int num = 28;

int sum = 0;

for (int i = 1; i < num; i++)

{

    if (num % i == 0)

        sum += i;

}

Console.WriteLine(sum == num ? "Perfect" : "Not Perfect");

# **Fibonacci Series up to N terms**

Input = 20

Ouptut = 0, 1, 1, 2, 3, 5, 8, 13, 21

int n = 20;

int first = 0, second = 1;

        Console.WriteLine("Fibonacci Series up to " + n + " terms:");

        for (int i = 1; i <= n; i++)

        {

            Console.Write(first + " ");

            int next = first + second;

            first = second;

            second = next;

        }

# **Sum of Digits**

int num = 1234;

int sum = 0;

while (num > 0)

{

    sum += num % 10;

    num = num / 10;

}

Console.WriteLine("Sum: " + sum);

# **Count Digits in a Number**

int num = 4587;

int count = 0;

while (num > 0)

{

    count++;

    num = num / 10;

}

Console.WriteLine("Digits: " + count);

# **Armstrong Number (3-digit)**

int num = 153;

int sum = 0;

int original = num;

while (num > 0)

{

    int digit = num % 10;

    sum += digit \* digit \* digit;

    num = num / 10;

}

if (sum == original)

    Console.WriteLine("Armstrong");

else

    Console.WriteLine("Not Armstrong");

# **Check Palindrome Number**

int num = 121;

int original = num;

int reverse = 0;

while (num > 0)

{

    int digit = num % 10;

    reverse = reverse \* 10 + digit;

    num = num / 10;

}

if (original == reverse)

    Console.WriteLine("Palindrome");

else

    Console.WriteLine("Not Palindrome");

# **Reverse an Integer**

int num = 1234;

int reverse = 0;

while (num > 0)

{

    int digit = num % 10;

    reverse = reverse \* 10 + digit;

    num = num / 10;

}

# **find given number is a Prime Number**

int number = 3;

 int count = 0;

 for (int i = 1; i <= number; i++)

 {

     if (number % i == 0)

         count++;

     if (count > 2)

         break;

 }

 if (number > 1 && count == 2)

     Console.WriteLine(number + " is a Prime Number.");

 else

     Console.WriteLine(number + " is NOT a Prime Number.");

# **How to rotate an array by k positions (left)?**

int[] arr = { 1, 2, 3, 4, 5 };

int k = 1;

// Left rotate by k positions

for (int i = 0; i < k; i++)

{

    // Store first element

    int first = arr[0];

    // Shift all elements one position to the left

    for (int j = 0; j < arr.Length - 1; j++)

    {

        arr[j] = arr[j + 1];

    }

    // Move the first element to the end

    arr[arr.Length - 1] = first;

}

// Output the rotated array

Console.WriteLine("Array after left rotation:");

foreach (int num in arr)

{

    Console.Write(num + " ");

}

// Output = Array after left rotation:   2 3 4 5 1

# **How do you count the frequency of elements in an array? in c#**

int[] arr = { 1, 2, 2, 3, 4, 4, 4, 5 };

        Dictionary<int, int> frequency = new Dictionary<int, int>();

        foreach (int num in arr)

        {

            if (frequency.ContainsKey(num))

            {

                frequency[num]++;

            }

            else

            {

                frequency[num] = 1;

            }

        }

        // Output the frequencies

        Console.WriteLine("Element : Frequency");

        foreach (var pair in frequency)

        {

            Console.WriteLine($"{pair.Key} : {pair.Value}");

        }

Output =

Element : Frequency

1 : 1

2 : 2

3 : 1

4 : 3

5 : 1

# **How do you remove duplicates from an array?**

Create an empty set or list to track seen elements.

Iterate through the array.

If the element is not in the set, add it to the result.

Return the result list (with no duplicates).

 int[] arr = { 1, 2, 2, 3, 4, 4, 5 };

 List<int> result = new List<int>();

 foreach (int num in arr)

 {

     if (!result.Contains(num))

         result.Add(num);

 }

 // Convert back to array if needed

 int[] uniqueArray = result.ToArray();

 Console.WriteLine("Unique elements:");

 foreach (int num in uniqueArray)

 {

     Console.Write(num + " ");

 }

# **How do you find the max element in an array?**

Initialize max = first element of array

Loop through the array:

   If current element > max:

       Set max = current element

Return max

int[] arr = { 5, 1, 9, 3, 7 };

int max = arr[0];

for (int i = 1; i < arr.Length; i++)

{

    if (arr[i] > max)

        max = arr[i];

}

Console.WriteLine("Maximum element: " + max);

# **Move all 0s to the end**

Problem name: Move 0s to End

Problem type: Array manipulation

Core logic: Track index for non-zero; overwrite and fill 0s

Mini-code (2-3 lines or pseudocode)

 int[] arr = { 0, 5, 0, 3, 0, 2, 7 };

 int nonZeroIndex = 0;

 // Move non-zero elements to the front

 for (int i = 0; i < arr.Length; i++)

 {

     if (arr[i] != 0)

     {

         arr[nonZeroIndex] = arr[i];

         nonZeroIndex++;

     }

 }

 // Fill the rest with 0s

 for (int i = nonZeroIndex; i < arr.Length; i++)

 {

     arr[i] = 0;

 }

 // Print the updated array

 foreach (int num in arr)

 {

     Console.Write(num + " ");

 }

# **Reverse an array**

  int[] arr = {1,2,3,4,5 };

  int n = arr.Length;

  // Reverse using swap

  for (int i = 0; i < n / 2; i++)

  {

      int temp = arr[i];

      arr[i] = arr[n - 1 - i];

      arr[n - 1 - i] = temp;

  }

  // Output reversed array

  Console.WriteLine("Reversed Array:");

  foreach (int num in arr)

  {

      Console.Write(num + " ");

  }

  // Output {5, 4,3, 2, 1}

# **Sort array without Array.Sort()**

Input = int[] arr = { 7, 2, 9, 1, 5 };

Output = { 1 2 5 7 9}

int[] arr = { 7, 2, 9, 1, 5 };

// Sort and print in a single method

for (int i = 0; i < arr.Length - 1; i++)

{

    for (int j = 0; j < arr.Length - 1 - i; j++)

    {

        if (arr[j] > arr[j + 1])

        {

            int temp = arr[j];

            arr[j] = arr[j + 1];

            arr[j + 1] = temp;

        }

    }

}

// Print sorted array

foreach (int num in arr)

{

    Console.Write(num + " ");

}

# **Sort An Given Array element**

problem statement Array of Numbers it can have positive and negative  {0 ,1,2,-5,-8,10,-13} Arrange  this Like First Negative Number and Then put zero then next put All Positive Number

Input = {0, 1, 2, -5, -8, 10, -13}

Output = { -13 -8 -5 0 1 2 10 }

Program

int[] arr = { 0, 1, 2, -5, -8, 10, -13 };

int n = arr.Length;

// Bubble Sort without using Array.Sort

for (int i = 0; i < n - 1; i++)

{

    for (int j = 0; j < n - i - 1; j++)

    {

        if (arr[j] > arr[j + 1])

        {

            // Swap using temp

            int temp = arr[j];

            arr[j] = arr[j + 1];

            arr[j + 1] = temp;

        }

    }

}

// Output the sorted array

Console.WriteLine("Manually Sorted Array:");

foreach (int num in arr)

{

    Console.Write(num + " ");

}

# **Sorting of an Integer array**

int[] arr = { 5, 3, 8, 4, 2 };

int n = arr.Length;

for (int i = 0; i < n - 1; i++)

{

    for (int j = 0; j < n - i - 1; j++)

    {

        if (arr[j] > arr[j + 1]) // Swap if greater

        {

            int temp = arr[j];

            arr[j] = arr[j + 1];

            arr[j + 1] = temp;

        }

    }

}

Console.WriteLine("Sorted Array: " + string.Join(", ", arr));

# **Find Second Highest Number in Array**

// find 2nd highest array element

int[] arr = { 3, 2, 5, 4, 1 };

int first = int.MinValue, second = int.MinValue;

foreach (int num in arr)

{

    if (num > first)

    {

        second = first;

        first = num;

    }

    else if (num > second && num != first)

        second = num;

}

Console.WriteLine("Second max "+smax);

Third Max Number in Array==>>

 public static void ThirdMaxArray(int[] array)

 {

     int firstMax = int.MinValue;

     int secondMax = int.MinValue;

     int thirdMax = int.MinValue;

     foreach (int num in array)

     {

         if (num > firstMax)

         {

             thirdMax = secondMax;

             secondMax = firstMax;

             firstMax = num;

         }

         else if (num > secondMax && num != firstMax)

         {

             thirdMax = secondMax;

             secondMax = num;

         }

         else if (num>thirdMax && num !=firstMax && num !=secondMax)

         {

             thirdMax = num;

         }

     }

     if (thirdMax == int.MinValue)

     {

         Console.WriteLine("There is no third highest element.");

     }

     else

     {

         Console.WriteLine("The third highest element is: " + thirdMax);

     }

 }

# **Find Duplicate Array Element Input: [4,3,2,7,8,2,3,1]**

// find duplicate array element

int[] inputarray = { 4, 3, 2, 7, 8, 2, 3, 1 };

List<int> outputarray = new List<int>();

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

{

    for (int j = i+1; j < inputarray.Length; j++)

    {

        if(inputarray[i] == inputarray[j])

        {

            if (!outputarray.Contains(inputarray[j]))

            {

                outputarray.Add(inputarray[j]);

            }

            }

        }

    }

foreach (int i in outputarray)

{

    Console.WriteLine(i);

}

# **Missing Element in Given Array**

class Program

{

    static void Main()

    {

        int[] array = { 0, 1, 3, 4 };

        int missingNumber = FindMissingNumber(array);

        if (missingNumber != -1)

        {

            Console.WriteLine($"The missing number is: {missingNumber}");

        }

        else

        {

            Console.WriteLine("No missing number found.");

        }

    }

    static int FindMissingNumber(int[] array)

    {

        // Sort the array for efficient search

        Array.Sort(array);

        // Check for the missing number

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

        {

            if (array[i] != i)

            {

                return i;

            }

        }

        // If no missing number is found, return -1

        return -1;

    }

}