2100030320

M.BHAVANA

EPAM- LAB_2

1. Write a C# code to implement the Tasks on Looping Statements?

TASK1: For a positive integer *n* calculate the *result* value, which is equal to the sum of the odd numbers in *n*

Example

```
n = 1234    result = 4 (1 + 3)
n = 246     result = 0
```

Code:

```
// Task 1: Calculate sum of odd numbers in n
1 reference
static int CalculateSumOfOddNumbers(int n)
{
    int result = 0;
    while (n > 0)
    {
        int digit = n % 10;
        if (digit % 2 != 0)
        {
            result += digit;
        }
        n /= 10;
    }
    return result;
}
0 references
class Program
{
    0 references
    static void Main()
    {
        Console.Write("Enter a positive integer for Task 1: ");
        int n = int.Parse(Console.ReadLine());
        int result = CalculateSumOfOddNumbers(n);
        Console.WriteLine("Task 1 Result: " + result);
```

Output:

```
Microsoft Visual Studio Debu
Enter a positive integer for Task 1: 1234
Task 1 Result: 4
```

TASK2: For a positive integer *n* calculate the result value, which is equal to the sum of the "1" in the binary representation of *n*.

Example

```
n = 14(decimal) = 1110(binary)        result =  3
n = 128(decimal) = 1000 0000(binary) result  =  1
```

```
// Task 2: Count the number of "1"s in binary representation of n
1 reference
static int CountOnesInBinaryRepresentation(int n)
{
    int result = 0;
    while (n > 0)
    {
        result += n % 2;
        n /= 2;
    }
    return result;
}
```

```
0 references
class Program
{
    0 references
    static void Main()
    {
        // Task 2
        Console.Write("Enter a positive integer for Task 2: ");
        int task2Input = int.Parse(Console.ReadLine());
        int task2Result = CountOnesInBinaryRepresentation(task2Input);
        Console.WriteLine("Task 2 Result: " + task2Result);
```

Output:

```
Enter a positive integer for Task 2: 14
Task 2 Result: 3
```

TASK3: For a positive integer *n*, calculate the result value equal to the sum of the first *n* Fibonacci numbers Note: Fibonacci numbers are a series of numbers in which each next number is equal to the sum of the two preceding ones: 0, 1, 1, 2, 3, 5, 8, 13... (F0=0, F1=F2=1, then F(n)=F(n-1)+F(n-2) for n>2)

Example

```
n = 8     result = 33
n = 11    result = 143
```

```
// Task 3: Calculate sum of the first n Fibonacci numbers
3 references
static int fib(int n)
{
    int s = 0;
    if (n == 0 || n == 1) return 0;
    if (n == 2) return 1;
    return fib(n - 1) + fib(n - 2);
}
1 reference
static int SumOfFib(int n)
{
    int sum = 0;
    for (int i = 0; i < n; i++)
    {
        sum += fib(i);
    }
    return sum;
}
```

In main method:

```
// Task 3
Console.Write("Enter a positive integer for Task 3: ");
int n = int.Parse(Console.ReadLine());
int res = int.Parse(Console.ReadLine());
if (res == SumOfFib(n))
    Console.WriteLine("true");
else Console.WriteLine("false");
```

Output:

```
Enter a positive integer for Task 3: 8
33
false

C:\Users\Bhavana\Documents\EPAM\LAB2_QUEST
To automatically close the console when de
le when debugging stops.
Press any key to close this window
```

1. Write a C# code to implement the Tasks on Arrays?

TASK 1: In a given array of integers ***nums*** swap values of the first and the last array elements, the second and the penultimate etc., if the two exchanged values are even

Example

```
{ 10 , 5, 3, 4}                    =>  {4, 5, 3, 10}
```

```
{100, 2, 3, 4, 5}                    => {100, 4, 3, 2, 5}
{100, 2, 3, 45, 33, 8, 4, 54} => {54, 4, 3, 45, 33, 8, 2, 100}
```

```
using System;
0 references
class Program
{
    0 references
    static void Main()
    {
        // Task 1
        int[] arr1 = { 10, 5, 3, 4 };
        SwapEvenIndexedElements(arr1);
        Console.WriteLine("Task 1 Result: " + string.Join(", ", arr1));

        int[] arr2 = { 100, 2, 3, 4, 5 };
        SwapEvenIndexedElements(arr2);
        Console.WriteLine("Task 1 Result: " + string.Join(", ", arr2));

        int[] arr3 = { 100, 2, 3, 45, 33, 8, 4, 54 };
        SwapEvenIndexedElements(arr3);
        Console.WriteLine("Task 1 Result: " + string.Join(", ", arr3));
```

```
static void SwapEvenIndexedElements(int[] nums)
{
    for (int i = 0; i < nums.Length / 2; i += 2)
    {
        if (nums[i] % 2 == 0 && nums[nums.Length - 1 - i] % 2 == 0)
        {
            // Swap even-indexed elements if both are even
            int temp = nums[i];
            nums[i] = nums[nums.Length - 1 - i];
            nums[nums.Length - 1 - i] = temp;
        }
    }
}
```

```
Microsoft Visual Studio Debu
Task 1 Result: 4, 5, 3, 10
Task 1 Result: 100, 2, 3, 4, 5
Task 1 Result: 54, 2, 3, 45, 33, 8, 4, 100
```

TASK 2: In a given array of integers ***nums*** calculate integer ***result*** value, that is equal to the distance between the first and the last entry of the maximum value in the array.

Example

```
{4, 100!, 3, 4}                result = 0
{5, 50!, 50!, 4, 5}           result = 1
{5, 350!, 350, 4, 350!} result = 3
{10!, 10, 10, 10, 10!}    result = 4
```

```
using System;
0 references
class Program
{
    0 references
    static void Main()
    {
        int[] arr4 = { 4, 100, 3, 4 };
         CalculateDistanceToMax(arr4);

        int[] arr5 = { 5, 50, 50, 4, 5 };
        CalculateDistanceToMax(arr5);

        int[] arr6 = { 5, 350, 350, 4, 350 };
        CalculateDistanceToMax(arr6);

        int[] arr7 = { 10, 10, 10, 10, 10 };
        CalculateDistanceToMax(arr7);
```

```
static void CalculateDistanceToMax(int[] nums)
{
    int max = int.MinValue;
    int maxIndexFirst = -1;
    int maxIndexLast = -1;

    for (int i = 0; i < nums.Length; i++){
        if (nums[i] > max) {
            max = nums[i];
            maxIndexFirst = i; }
        if (nums[i] == max)
            maxIndexLast = i;
    }
    if (maxIndexFirst != -1 && maxIndexLast != -1)
    {
        int difference = maxIndexLast - maxIndexFirst;
        Console.WriteLine($"Highest number: {max}, Difference between indices: {difference}");
    }
    else
    {
        Console.WriteLine("Array is empty or does not contain a highest number.");
    }
}
```

```
Highest number: 100, Difference between indices: 0
Highest number: 50, Difference between indices: 1
Highest number: 350, Difference between indices: 3
Highest number: 10, Difference between indices: 4
```

TASK 3: In a predetermined two-dimensional integer array

(square matrix) ***matrix*** insert 0 into elements to the left side of the main diagonal, and 1 into elements to the right side of the diagonal.

Example

```
{{2, 4, 3, 3},            {{2, 1, 1, 1},
 {5, 7, 8, 5},     =>     {0, 7, 1, 1},
 {2, 4, 3, 3},               {0, 0, 3, 1},
 {5, 7, 8, 5}}             {0, 0, 0, 5}}
```

```
using System;
0 references
class Program
{
    0 references
    static void Main()
    {
        int[,] matrix = {
            {2, 4, 3, 3},
            {5, 7, 8, 5},
            {2, 4, 3, 3},
            {5, 7, 8, 5}
        };
        ModifyMatrix(matrix);
        Console.WriteLine("Task 3 Result:");
        PrintMatrix(matrix);
    }
```

```
1 reference
static void ModifyMatrix(int[,] matrix)
{
    int n = matrix.GetLength(0);

    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            if (j < i)
            {
                matrix[i, j] = 0;
            }
            else if (j > i)
            {
                matrix[i, j] = 1;
            }
        }
    }
}
```

```
1 reference
static void PrintMatrix(int[,] matrix)
{
    int rows = matrix.GetLength(0);
    int cols = matrix.GetLength(1);

    for (int i = 0; i < rows; i++)
    {
        for (int j = 0; j < cols; j++)
        {
            Console.Write(matrix[i, j] + " ");
        }
        Console.WriteLine();
    }
}
```

```
Task 3 Result:
2 1 1 1
0 7 1 1
0 0 3 1
0 0 0 5

C:\Users\Bhavana\Documents\EPAM\lab2_que
To automatically close the console when
le when debugging stops.
Press any key to close this window . . .
```

2. Write a C# code to implement the Tasks on Functions?

TASK 1: **Create function** ***IsSorted*****, determining whether a given** ***array*** **of integer values of arbitrary length is sorted in a given** ***order*** **(the order is set up by enum value** ***SortOrder*****). Array and sort order are passed by parameters. Function does not change the array**

```
using System;

0 references
class Program
{
    0 references
    static void Main()
    {
        // Task 1
        int[] arr1 = { 5, 17, 24, 88, 33, 2 };
        SortOrder sortOrder1 = SortOrder.Ascending;
        Console.WriteLine($"Task 1 Result: IsSorted - {IsSorted(arr1, sortOrder1)}");

 static bool IsSorted(int[] arr, SortOrder sortOrder)
 {
     if (sortOrder == SortOrder.Ascending)
     {
         for (int i = 0; i < arr.Length - 1; i++)
         {
             if (arr[i] > arr[i + 1])
             {
                 return false;
             }
         }
     }
     else if (sortOrder == SortOrder.Descending)
     {
         for (int i = 0; i < arr.Length - 1; i++)
         {
             if (arr[i] < arr[i + 1])
             {
                 return false;
             }
         }
     }

     return true;
 }
```

```
8 references
enum SortOrder
{
    Ascending,
    Descending
}
```

```
Microsoft Visual Studio Debu

Task 1 Result: IsSorted - False
Task 2 Result: Transformed Array - 5, 17, 24, 88, 33, 2
Task 3 Result: MultArithmeticElements - 6160
Task 4 Result: SumGeometricElements - 175

C:\Users\Bhavana\Documents\EPAM\lab2_question3\bin\Debug\net8.0\lab2_question3
To automatically close the console when debugging stops, enable Tools->Options
le when debugging stops.
Press any key to close this window . . .
```

TASK 2: Create function ***Transform***, replacing the value of each element of an integer ***array*** with the sum of this element value and its index, only if the given ***array*** is sorted in the given ***order*** (the order is set up by enum value ***SortOrder***). Array and sort order are passed by parameters. To check, if the array is sorted, the function ***IsSorted*** from the Task 1 is called.

Example

```
For {5, 17, 24, 88, 33, 2} and "ascending" sort order values in the array
do not change;
For {15, 10, 3} and "ascending" sort order values in the array do not
change;
For {15, 10, 3} and "descending" sort order the values in the array change
to {15, 11, 5}
```

```
class Program
{
    0 references
    static void Main()
    {
        int[] arr2 = { 5, 17, 24, 88, 33, 2 };
        SortOrder sortOrder2 = SortOrder.Ascending;
        Transform(arr2, sortOrder2);
        Console.WriteLine($"Task 2 Result: Transformed Array - {string.Join(", ", arr2)}");
```

```
1 reference
static void Transform(int[] arr, SortOrder sortOrder)
{
    if (IsSorted(arr, sortOrder))
    {
        for (int i = 0; i < arr.Length; i++)
        {
            arr[i] += i;
        }
    }
}
```

```
Microsoft Visual Studio Debu

Task 1 Result: IsSorted - False
Task 2 Result: Transformed Array - 5, 17, 24, 88, 33, 2
Task 3 Result: MultArithmeticElements - 6160
Task 4 Result: SumGeometricElements - 175

C:\Users\Bhavana\Documents\EPAM\lab2_question3\bin\Debug\net8.0\lab2
To automatically close the console when debugging stops, enable Tool
le when debugging stops.
Press any key to close this window . . .
```

TASK 3: Create function ***MultArithmeticElements***, which determines the multiplication of a given number of first ***n*** elements of arithmetic progression of real numbers with a given initial element of progression ***a(1)*** and progression step ***t***. a(n) is calculated by the formula a(n+1) = a(n) + t.

Example

```
For a(1) = 5, t = 3, n = 4  multiplication equals to 5*8*11*14 = 6160
```

```
1 reference
static double MultArithmeticElements(double a, double t, int n)
{
    double result = 1;
    for (int i = 0; i < n; i++)
    {
        result *= a;
        a += t;
    }
    return result;
}
```

```
using System;

0 references
class Program
{
    0 references
    static void Main()
    {
        double a1 = 5;
        double t1 = 3;
        int n1 = 4;
        Console.WriteLine($"Task 3 Result: MultArithmeticElements - {MultArithmeticElements(a1, t1, n1)}");
```

```
Task 1 Result: IsSorted - False
Task 2 Result: Transformed Array - 5, 17, 24, 88, 33, 2
Task 3 Result: MultArithmeticElements - 6160
Task 4 Result: SumGeometricElements - 175

C:\Users\Bhavana\Documents\EPAM\lab2_question3\bin\Debug\net8.0\
To automatically close the console when debugging stops, enable
le when debugging stops.
Press any key to close this window . . .
```

TASK 4: Create function ***SumGeometricElements***, determining the sum of the first elements of a decreasing geometric progression of real numbers with a given initial element of a progression ***a(1)***) and a given progression step ***t***, while the last element must be greater than a given ***alim***. an is calculated by the formula a(n+1) = a(n) * t, 0<t<1.

```
using System;

0 references
class Program
{
    0 references
    static void Main()
    {
        double a2 = 100;
        double t2 = 0.5;
        double alim = 20;
        Console.WriteLine($"Task 4 Result: SumGeometricElements - {SumGeometricElements(a2, t2, alim)}");
    }

    1 reference
    static double SumGeometricElements(double a, double t, double alim)
    {
        double sum = 0;
        while (a > alim)
        {
            sum += a;
            a *= t;
        }
        return sum;
    }
```

```
Task 1 Result: IsSorted - False
Task 2 Result: Transformed Array - 5, 17, 24, 88, 33, 2
Task 3 Result: MultArithmeticElements - 6160
Task 4 Result: SumGeometricElements - 175

C:\Users\Bhavana\Documents\EPAM\lab2_question3\bin\Debug\net8.0\
To automatically close the console when debugging stops, enable
le when debugging stops.
Press any key to close this window . . .
```