Lab 1: Tasks on C# Basics Concepts

Name: Gondrala Mani Sai

Id number: 2100031545

1. Write a C# code to implement the simple calculator?

TASK1: It's required to create a simple calculator with addition and subtraction operations for two integer numbers

For example, how to find the sum of given integer values **a** and **b**. You have a skeleton code:

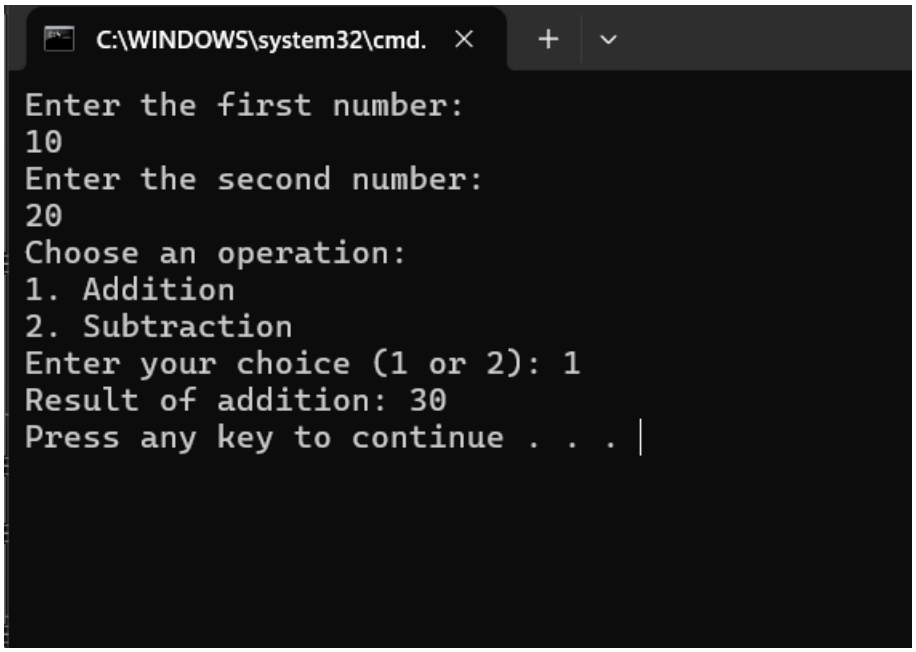
```
public static int Add(int a, int b)
{
    //TODO Delete line below and write your own solution
    throw new NotImplementedException();
}
```

Solution:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab_1
{
    internal class Program
    {
        public static int Add(int a, int b)
        {
            return a + b;
        }

        public static int Subtract(int a, int b)
        {
            return a - b;
        }

        public static void Main(string[] args)
        {
            int num1, num2;
            Console.WriteLine("Enter the first number:");
            if (!int.TryParse(Console.ReadLine(), out num1))
            {
                Console.WriteLine("Invalid input. Please enter a valid
integer.");
                return;
            }

            Console.WriteLine("Enter the second number:");
```

```
            if (!int.TryParse(Console.ReadLine(), out num2))
            {
                Console.WriteLine("Invalid input. Please enter a valid
integer.");
                return;
            }

            Console.WriteLine("Choose an operation:");
            Console.WriteLine("1. Addition");
            Console.WriteLine("2. Subtraction");
            Console.Write("Enter your choice (1 or 2): ");

            int choice;
            if (!int.TryParse(Console.ReadLine(), out choice))
            {
                Console.WriteLine("Invalid choice. Please enter 1 or 2.");
                return;
            }

            switch (choice)
            {
                case 1:
                    Console.WriteLine($"Result of addition: {Add(num1, num2)}");
                    break;
                case 2:
                    Console.WriteLine($"Result of subtraction: {Subtract(num1,
num2)}");
                    break;
                default:
                    Console.WriteLine("Invalid choice. Please enter 1 or 2.");
                    break;
            }
        }
    }

}
```

Output:



TASK2: For a given integer *n* calculate the value which is equal to:

1. squared number, if its value is strictly positive;
2. modulus of a number, if its value is strictly negative;
3. zero, if the integer n is zero.

Example

```
n = 4      result =  16
n = -5     result =  5
n = 0      result = 0
```

Solution

```
public static int CalculateValue(int n)
{
    if (n > 0)
    {
        return n * n; // squared number
    }
    else if (n < 0)
    {
        return Math.Abs(n); // modulus of a number
    }
    else
    {
        return 0; // zero
    }
}

public static void Main(string[] args)
{
    int n;

    Console.WriteLine("Enter an integer:");
    if (!int.TryParse(Console.ReadLine(), out n))
    {
        Console.WriteLine("Invalid input. Please enter a valid integer.");
        return;
    }

    int result = CalculateValue(n);
    Console.WriteLine($"Result: {result}");
}
```

Output:

```
C:\WINDOWS\system32\cmd.

Enter an integer:
20
Result: 400
Press any key to continue . . .
```

TASK3: Find the maximum integer, that can be obtained by numbers of an arbitrary three-digit positive integer *n* permutation (100<=n<=999).

Example

`n = 165  result =  651`

Solution

```
public static int MaxPermutation(int n)
{
    // Convert the integer to a string to make it easier to work with its digits
    string number = n.ToString();

    // Convert the string to an array of characters for manipulation
    char[] digits = number.ToCharArray();

    // Sort the array of digits in descending order
    Array.Sort(digits);
    Array.Reverse(digits);

    // Convert the sorted array of digits back to a string
    string result = new string(digits);

    // Convert the string representation of the number back to an integer
    return int.Parse(result);
}

public static void Main(string[] args)
{
    int n;

    Console.WriteLine("Enter a three-digit positive integer:");
    if (!int.TryParse(Console.ReadLine(), out n) || n < 100 || n > 999)
    {
        Console.WriteLine("Invalid input. Please enter a valid three-digit
positive integer.");
        return;
    }

    int result = MaxPermutation(n);
    Console.WriteLine($"Maximum permutation: {result}");
}
```

Output:

```
C:\WINDOWS\system32\cmd.

Enter a three-digit positive integer:
739
Maximum permutation: 973
Press any key to continue . . .
```