C#: LAB MANUAL

1. Program to add two numbers using command line arguments.

```
using System;
class AddNumbers
{
   static void Main(string[] args)
   {
      if (args.Length < 2)
      {
   Console.WriteLine("Please enter two values");
         return;
      }

     int num1 = int.Parse(args[0]);
      int num2 = int.Parse(args[1]);
      int sum = num1 + num2;
      Console.WriteLine("Sum: " + sum);
   }
}
```

Output:

Sum: 30

2. Program to demonstrate the use of methods and operators.

```
using System;
class Arithmetic
{
    // Arithmetic methods
    static int Add(int a, int b)
    {
        return a + b; // Addition
    }
    static int Subtract(int a, int b)
    {
        return a - b; // Subtraction
    }
    static int Multiply(int a, int b)
    {
        return a * b; // Multiplication
    }
    static double Divide(int a, int b)
    {
        return (double)a / b; // Division
    }
    static int Modulo(int a, int b)
    {
        return a % b; // Modulus
    }
```

```
    static void Main(string[] args)
    {
        int num1 = 20;
        int num2 = 6;

        Console.WriteLine("Arithmetic Operations:");
        Console.WriteLine($"{num1} + {num2} = {Add(num1, num2)}");
        Console.WriteLine($"{num1} - {num2} = {Subtract(num1, num2)}");
        Console.WriteLine($"{num1} * {num2} = {Multiply(num1, num2)}");
        Console.WriteLine($"{num1} / {num2} = {Divide(num1, num2)}");
        Console.WriteLine($"{num1} % {num2} = {Modulo(num1, num2)}");
    }
}
```

Output:

Arithmetic Operations:

20 + 6 = 26

20 - 6 = 14

20 * 6 = 120

20 / 6 = 3.3333333333333335

20 % 6 = 2

3. C# Program to demonstrate operations on an array list.

```
using System;

using System. Collections;

namespace arraylist
{
   internal class Program
   {
      static void Main(string[]  args)
      {
         ArrayList a = new ArrayList();


         a.Add(10);
         a.Add(3.14);
         a.Add("hello");
         a.Add(true);

         Console.WriteLine("After adding elements:");
         foreach (var item in a)
         {
            Console.WriteLine(item);
         }
         Console.WriteLine("Access elements");
         Console.WriteLine("First element a[0]: " +a[0]);
         Console.WriteLine("Second element a[1]: " +a[1]);


         Console.WriteLine(" after inserting new element at position 2");
         a.Insert(2, 20);
         foreach (var item in a)
         {
            Console.WriteLine(item);

         }

         a[4] = "false";
         Console.WriteLine(" after modifying the fourth element ");
         foreach (var item in a)
```

```
        {
            Console.WriteLine(item);
        }

        Console.WriteLine(" after removing  second element ");
        a.Remove(3.14);
        foreach (var item in a)
        {
            Console.WriteLine(item);

        }
        Console.WriteLine("does the list contains the word hello:?");

        if (a.Contains("hello"))
            Console.WriteLine("Yes,  exists at index "
                              + a.IndexOf("hello"));
        else
            Console.WriteLine("No, doesn't exists");


        Console.WriteLine("\nSize of the list after removal: " + a.Count);

        Console.WriteLine("Capacity: " + a.Capacity);
        a.Clear();

        Console.WriteLine("after clearing total elements:" + a.Count);


    }
  }

}
```

OUTPUT:

```
After adding elements:
10
3.14
hello
True
```

Access elements
First element a[0]: 10
Second element a[1]: 3.14

after inserting new element at position 2
10
3.14
20
hello
True

after modifying the fourth element
10
3.14
20
hello
false

after removing second element
10
20
hello
false

does the list contains the word hello:?
Yes, exists at index 2

Size of the list after removal: 4
Capacity: 8
after clearing total elements:0

4. Program to demonstrate string operations

```
using System;
class StringFunctionsDemo
{
    static void Main()
    {
        Console.Write("Enter a string: ");
        string str = Console.ReadLine();

        Console.WriteLine("\n--- String Functions ---");

        // Length
        Console.WriteLine("Length: " + str.Length);

        // ToUpper and ToLower
        Console.WriteLine("Uppercase: " + str.ToUpper());
        Console.WriteLine("Lowercase: " + str.ToLower());

        // Substring
        if (str.Length >= 5)
            Console.WriteLine("Substring (0 to 4): " + str.Substring(0, 5));
        else
            Console.WriteLine("String too short for substring example.");

        // Replace
```

```
        Console.WriteLine("Replace 'a' with '*': " + str.Replace('a', '*'));

        // Contains
        Console.Write("Enter a word to search in the string: ");
        string word = Console.ReadLine();
        Console.WriteLine($"Contains \"{word}\"? " + str.Contains(word));

        // IndexOf
        Console.Write("Enter a character to find its position: ");
        char ch = Console.ReadLine()[0];
        int index = str.IndexOf(ch);
        if (index >= 0)
            Console.WriteLine($"Character '{ch}' found at index: {index}");
        else
            Console.WriteLine($"Character '{ch}' not found.");

        // Trim
        string padded = "   " + str + "   ";
        Console.WriteLine("Original with spaces: '" + padded + "'");
        Console.WriteLine("After Trim(): '" + padded.Trim() + "'");
    }
}
```

OUTPUT:

Enter a string: Hello World

--- String Functions ---

Length: 11

Uppercase: HELLO WORLD

Lowercase: hello world

Substring (0 to 4): Hello

Replace 'a' with '*': Hello World

Enter a word to search in the string: World

Contains "World"? True

Enter a character to find its position: o

Character 'o' found at index: 4

Original with spaces: ' Hello World '

After Trim(): 'Hello World'

5. **Program to demonstrate both default and parameterized constructors using a Student class.**

```
using System;
class Student
{
    public int rollNo;
    public string name;

    // Default constructor
    public Student()
    {
        rollNo = 0;
        name = "Unknown";
    }

    // Parameterized constructor
    public Student(int r, string n)
    {
        rollNo = r;
        name = n;
    }

    // Display method
    public void Display()
```

```
    {
        Console.WriteLine("Roll No: " + rollNo);
        Console.WriteLine("Name   : " + name);
    }
}

class Program
{
    static void Main()
    {
        // Using default constructor
        Console.WriteLine("Student 1 (Default Constructor):");
        Student s1 = new Student();
        s1.Display();

        Console.WriteLine();

        // Using parameterized constructor
        Console.WriteLine("Student 2 (Parameterized Constructor):");
        Console.Write("Enter roll number: ");
        int r = int.Parse(Console.ReadLine());

        Console.Write("Enter name: ");
        string n = Console.ReadLine();
```

```
        Student s2 = new Student(r, n);
        s2.Display();
    }
}
```

OUTPUT:

```
Student 1 (Default Constructor):
Roll No: 0
Name   : Unknown

Student 2 (Parameterized Constructor):
Enter roll number: 101
Enter name: Ananya
Roll No: 101
Name   : Ananya
```

6. Program to demonstrate multilevel inheritance using person, employee & manager.

```
using System;


class Person
{
   public string name;
   public int age;


   public void GetPersonDetails()
   {
      Console.Write("Enter name: ");
      name = Console.ReadLine();


      Console.Write("Enter age: ");
      age = int.Parse(Console.ReadLine());
   }


   public void ShowPersonDetails()
   {
      Console.WriteLine("Name: " + name);
      Console.WriteLine("Age: " + age);
   }
}
```

```
class Employee : Person
{
    public int empId;
    public string department;

    public void GetEmployeeDetails()
    {
        Console.Write("Enter employee ID: ");
        empId = int.Parse(Console.ReadLine());

        Console.Write("Enter department: ");
        department = Console.ReadLine();
    }

    public void ShowEmployeeDetails()
    {
        Console.WriteLine("Employee ID: " + empId);
        Console.WriteLine("Department : " + department);
    }
}

class Manager : Employee
{
    public string project;
```

```
    public void GetManagerDetails()
    {
        Console.Write("Enter project name: ");
        project = Console.ReadLine();
    }

    public void ShowManagerDetails()
    {
        Console.WriteLine("Project     : " + project);
    }
}

class Program
{
    static void Main()
    {
        Manager mgr = new Manager();

        Console.WriteLine("--- Enter Manager Details ---");
        mgr.GetPersonDetails();
        mgr.GetEmployeeDetails();
        mgr.GetManagerDetails();

        Console.WriteLine("\n--- Manager Information ---");
        mgr.ShowPersonDetails();
```

```
        mgr.ShowEmployeeDetails();
        mgr.ShowManagerDetails();
    }
}
```

Output:

```
--- Enter Manager Details ---
Enter name: Arjun
Enter age: 35
Enter employee ID: 101
Enter department: Sales
Enter project name: Market Expansion

--- Manager Information ---
Name: Arjun
Age: 35
Employee ID: 101
Department : Sales
Project     : Market Expansion
```

7. Program to demonstrate method overloading

```
using System;
class AreaCalculator
{
    // Area of a circle: π * r²
    public double Area(double radius)
    {
        return Math.PI * radius * radius;
    }

    // Area of a triangle: ½ * base * height
    public double Area(double b, double h)
    {
        return 0.5 * b * h;
    }

    // Area of a square: side²
    public int Area(int side)
    {
        return side * side;
    }
}

class Program
{
```

```
    static void Main()
    {
        AreaCalculator ac = new AreaCalculator();

        // Circle with radius 5
        double circleArea = ac.Area(5.0);

        // Triangle with base 4 and height 6
        double triangleArea = ac.Area(4.0, 6.0);

        // Square with side 3
        int squareArea = ac.Area(3);

        Console.WriteLine("Area of Circle: " + circleArea);
        Console.WriteLine("Area of Triangle :" + triangleArea);
        Console.WriteLine("Area of Square: " + squareArea);
    }
}
```

OUTPUT:

Area of Circle : 78.53981633974483

Area of Triangle : 12

Area of Square : 9

8. **Program to overload the + operator to add two objects of a Complex class.**

```
using System;
class Complex
{
    public int real;
    public int imag;

    // Constructor
    public Complex(int r, int i)
    {
        real = r;
        imag = i;
    }

    // Overload + operator
    public static Complex operator +(Complex c1, Complex c2)
    {
        return new Complex(c1.real + c2.real, c1.imag + c2.imag);
    }

    // Display method
```

```
    public void Display()
    {
        Console.WriteLine($"{real} + {imag}i");
    }
}

class Program
{
    static void Main()
    {
        // Input for first complex number
        Console.WriteLine("Enter first complex number:");
        Console.Write("Real part: ");
        int r1 = int.Parse(Console.ReadLine());
        Console.Write("Imaginary part: ");
        int i1 = int.Parse(Console.ReadLine());

        // Input for second complex number
        Console.WriteLine("\nEnter second complex number:");
        Console.Write("Real part: ");
        int r2 = int.Parse(Console.ReadLine());
```

```
        Console.Write("Imaginary part: ");
        int i2 = int.Parse(Console.ReadLine());

        // Create objects
        Complex c1 = new Complex(r1, i1);
        Complex c2 = new Complex(r2, i2);

        // Add using overloaded +
        Complex result = c1 + c2;

        // Display result
        Console.WriteLine("\nFirst Complex Number: ");
        c1.Display();

        Console.WriteLine("Second Complex Number: ");
        c2.Display();

        Console.WriteLine("Sum of Complex Numbers: ");
        result.Display();
    }
}
```

OUTPUT:

Enter first complex number:

Real part: 3

Imaginary part: 4

Enter second complex number:

Real part: 5

Imaginary part: 6

First Complex Number:

3 + 4i

Second Complex Number:

5 + 6i

Sum of Complex Numbers:

8 + 10i