Assignment - 1

.Net Technologies

Name: Kamithkar Vinod
Course: PG DAC August 2025
PRN: 250850320040
Form No: 250500480
Date: 01-12-2025

1. Calculator Application

File Name: CalculatorApp.cs

Question: Create a simple calculator application that performs basic arithmetic operations (addition, subtraction, multiplication, and division). Take input from the user and display the result.

Code

```
using System;

class Calculator
{
    static void Main()
    {
        Console.Write( Enter first number:  );
        double a = Convert.ToDouble(Console.ReadLine());

        Console.Write( Enter second number:  );
        double b = Convert.ToDouble(Console.ReadLine());

        Console.Write( Enter operator (+ - * /):  );
        char op = Console.ReadLine()[0];

        double result = 0;
        switch (op)
        {
            case ’+’: result = a + b; break;
            case ’-’: result = a - b; break;
            case ’*’: result = a * b; break;
            case ’/’: result = b != 0 ? a / b : double.NaN; break;
            default: Console.WriteLine( Invalid operator ); return;
        }
        Console.WriteLine( Result =   + result);
    }
}
```

Sample Output

```
Enter first number: 10
Enter second number: 5
Enter operator (+ - * /): *
Result = 50
```

2. Multicast Delegate

File Name: MulticastDelegateDemo.cs

Question: Implement multicast delegates using Add and Sub two methods to perform addition and subtraction.

Code

```
using System;

public delegate void MyDelegate(int x, int y);

class Program
{
    public static void Add(int a, int b) => Console.WriteLine( Addition
        =   + (a + b));
    public static void Sub(int a, int b) => Console.WriteLine(
       Subtraction =   + (a - b));

    static void Main()
    {
        MyDelegate del = Add;
        del += Sub;
        del(20, 10);
    }
}
```

Sample Output

```
Addition = 30
Subtraction = 10
```

3. Even / Odd

File Name: EvenOddCheck.cs

Question: Write a program to check whether the entered number is even or odd.

Code

```
using System;

class EvenOdd
{
    static void Main()
    {
        Console.Write( Enter a number:  );
        int n = Convert.ToInt32(Console.ReadLine());

        if (n % 2 == 0)
            Console.WriteLine( Even Number );
        else
            Console.WriteLine( Odd Number );
    }
}
```

Sample Output

```
Enter a number: 17
Odd Number
```

4. Generic Delegate Even / Odd

File Name: GenericDelegateEvenOdd.cs

Question: Create a C# program with a generic delegate to check whether the entered number is even or odd.

Code

```
using System;

public delegate bool CheckDelegate<T>(T value);

class Program
{
    static void Main()
    {
        CheckDelegate<int> isEven = num => num % 2 == 0;

        Console.Write( Enter a number:  );
        int n = Convert.ToInt32(Console.ReadLine());

        Console.WriteLine(isEven(n) ?  Even  :  Odd );
    }
}
```

Sample Output

```
Enter a number: 24
Even
```

5. Interface – Area of Shapes

File Name: ShapesAreaProgram.cs

Question: Create interface based program to calculate area of circle, square and triangle.

Code

```
using System;

interface IShape
{
    double Area();
}

class Circle : IShape
{
    public double Radius { get; set; }
    public double Area() => 3.14 * Radius * Radius;
}

class Square : IShape
{
    public double Side { get; set; }
    public double Area() => Side * Side;
}

class Triangle : IShape
{
    public double Base { get; set; }
    public double Height { get; set; }
    public double Area() => 0.5 * Base * Height;
}

class Program
{
    static void Main()
    {
        Circle c = new Circle { Radius = 5 };
        Square s = new Square { Side = 4 };
        Triangle t = new Triangle { Base = 6, Height = 3 };

        Console.WriteLine( Circle Area =   + c.Area());
        Console.WriteLine( Square Area =   + s.Area());
        Console.WriteLine( Triangle Area =   + t.Area());
    }
}
```

Sample Output

```
Circle Area = 78.5
Square Area = 16
Triangle Area = 9
```

6. Dependency Injection – Show Employees

File Names:

- Employee.cs
- EmployeeDAL.cs
- EmployeeBL.cs
- DependencyInjectionDemo.cs

Code

```
public class Employee
{
    public int ID { get; set; }
    public string Name { get; set; }
    public string Department { get; set; }
}
```

```
using System.Collections.Generic;

public class EmployeeDAL
{
    public List<Employee> SelectAllEmployees()
    {
        return new List<Employee>
        {
            new Employee() { ID = 1, Name =  Pranaya , Department =  IT
                 },
            new Employee() { ID = 2, Name =  Kumar , Department =  HR
               },
            new Employee() { ID = 3, Name =  Rout , Department =
               Payroll  }
        };
    }
}
```

```
using System;

public class EmployeeBL
{
    private readonly EmployeeDAL _dal;

    public EmployeeBL(EmployeeDAL dal)
    {
        _dal = dal;
    }

    public void DisplayEmployees()
    {
        foreach (var e in _dal.SelectAllEmployees())
            Console.WriteLine($ ID: {e.ID}, Name: {e.Name}, Department:
                {e.Department} );
    }
}
```

```
class DependencyInjectionDemo
{
    static void Main()
    {
        EmployeeDAL dal = new EmployeeDAL();
        EmployeeBL bl = new EmployeeBL(dal);
        bl.DisplayEmployees();
    }
}
```

Sample Output

```
ID: 1, Name: Pranaya, Department: IT
ID: 2, Name: Kumar, Department: HR
ID: 3, Name: Rout, Department: Payroll
```