

## .Net Programming-Home-Assignment-2

Sec-12

Name: Syed Hasif Alisha

Id number:2100030522

1Q) Access Modifiers(public, private, protected, internal)-

Solution:

```
using System;

public class Example
{
    public int PublicMember;
    private int PrivateMember;
    protected int ProtectedMember;
    internal int InternalMember;

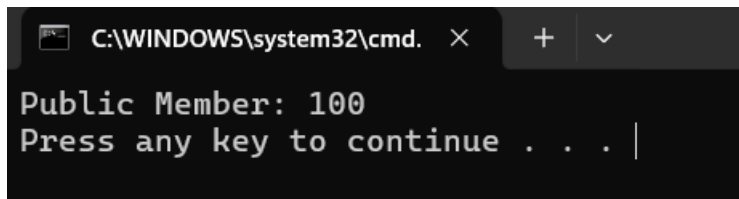
    public Example()
    {
        // Inside the class, you can access all members
        this.PublicMember = 1;
        this.PrivateMember = 2;
        this.ProtectedMember = 3;
        this.InternalMember = 4;
    }
}

public class DerivedExample : Example
{
    public DerivedExample()
    {
        this.ProtectedMember = 10;
    }
}

class Program
{
    static void Main(string[] args)
    {
        Example example = new Example();

        example.PublicMember = 100;
        Console.WriteLine("Public Member: " + example.PublicMember);
    }
}
```

Output:



```
C:\WINDOWS\system32\cmd. X + v
Public Member: 100
Press any key to continue . . . |
```

## 2Q) Use of set and get accessors

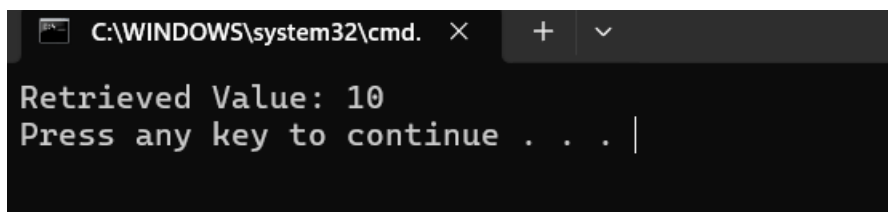
Solution:

```
using System;

public class Example
{
    private int _value;
    public int Value
    {
        get
        {
            return _value;
        }
        set
        {
            _value = value;
        }
    }
}

class Program
{
    static void Main(string[] args)
    {
        Example example = new Example();
        example.Value = 10;
        int retrievedValue = example.Value;
        Console.WriteLine("Retrieved Value: " + retrievedValue);
    }
}
```

Output:



```
C:\WINDOWS\system32\cmd. X + v
Retrieved Value: 10
Press any key to continue . . . |
```

## 3Q) Inheritance

Solution:

```
using System;

public class Animal
```

```

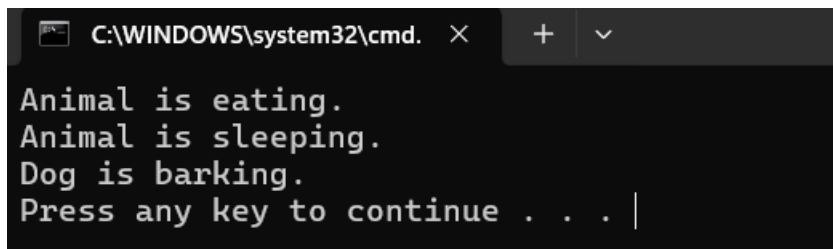
{
    public void Eat()
    {
        Console.WriteLine("Animal is eating.");
    }

    public void Sleep()
    {
        Console.WriteLine("Animal is sleeping.");
    }
}
public class Dog : Animal
{
    public void Bark()
    {
        Console.WriteLine("Dog is barking.");
    }
}

class Program
{
    static void Main(string[] args)
    {
        Dog dog = new Dog();
        dog.Eat();
        dog.Sleep();
        dog.Bark();
    }
}

```

**Output:**



A screenshot of a Windows Command Prompt window. The title bar shows the path 'C:\WINDOWS\system32\cmd.' and standard window controls. The command prompt displays the following output: 'Animal is eating.', 'Animal is sleeping.', 'Dog is barking.', and 'Press any key to continue . . . |'.

```

C:\WINDOWS\system32\cmd.
Animal is eating.
Animal is sleeping.
Dog is barking.
Press any key to continue . . . |

```

#### 4Q) Abstract classes and methods

**Solution:**

```

using System;
public abstract class Shape
{
    public abstract double Area();
    public void Display()
    {
        Console.WriteLine("This is a shape.");
    }
}
public class Rectangle : Shape
{
    public override double Area()
    {
        return 5 * 10;
    }
}

```

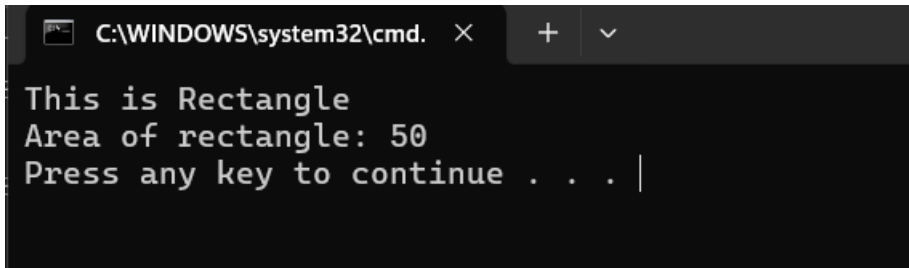
```

    }
}

class Program
{
    static void Main(string[] args)
    {
        Rectangle rectangle = new Rectangle();
        rectangle.Display();
        Console.WriteLine("Area of rectangle: " + rectangle.Area()); // Output:
    }
}

```

Output:



```

C:\WINDOWS\system32\cmd.  X  +  v
This is Rectangle
Area of rectangle: 50
Press any key to continue . . . |

```

## 5) Polymorphism (compile-time, run-time)

Solution:

```

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

namespace HA_2
{
    internal class Calculator
    {
        public int Add(int a, int b)
        {
            return a + b;
        }

        public double Add(double a, double b)
        {
            return a + b;
        }
    }
}

using System;

class Program
{
    static void Main(string[] args)
    {
        Calculator calculator = new Calculator();

        int sum1 = calculator.Add(5, 10); // Calls the int version of Add
        double sum2 = calculator.Add(3.5, 2.7); // Calls the double version of
Add

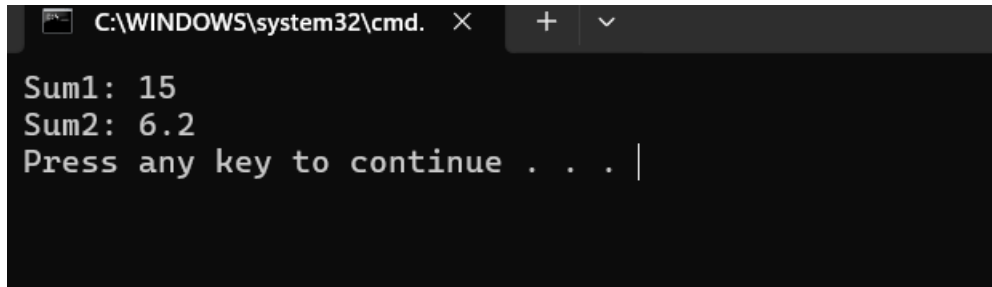
```

```

        Console.WriteLine("Sum1: " + sum1); // Output: Sum1: 15
        Console.WriteLine("Sum2: " + sum2); // Output: Sum2: 6.2
    }
}

```

**Output:**



```

C:\WINDOWS\system32\cmd.
Sum1: 15
Sum2: 6.2
Press any key to continue . . . |

```

## 6Q) Encapsulation and Abstraction

**Solution:**

```

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

namespace HA_2
{
    internal class Car
    {
        private string _brand;
        private int _year;

        public string Brand
        {
            get { return _brand; }
            set { _brand = value; }
        }

        public int Year
        {
            get { return _year; }
            set
            {
                if (value >= 1900 && value <= DateTime.Now.Year)
                    _year = value;
                else
                    throw new ArgumentException("Invalid year.");
            }
        }
    }
}

```

**Output:**

7Q) Constructors-destructors

Solution:

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

namespace HA_2
{
    internal class Car1
    {
        private string _brand;
        private int _year;

        // Parameterized constructor
        public Car1(string brand, int year)
        {
            _brand = brand;
            _year = year;
            Console.WriteLine("Constructor called. Car created.");
        }

        // Destructor
        ~Car1()
        {
            Console.WriteLine("Destructor called. Cleaning up resources.");
            // Cleanup operations go here
        }

        // Method to display car details
        public void DisplayDetails()
        {
            Console.WriteLine($"Brand: {_brand}, Year: {_year}");
        }
    }
}
```

Output: