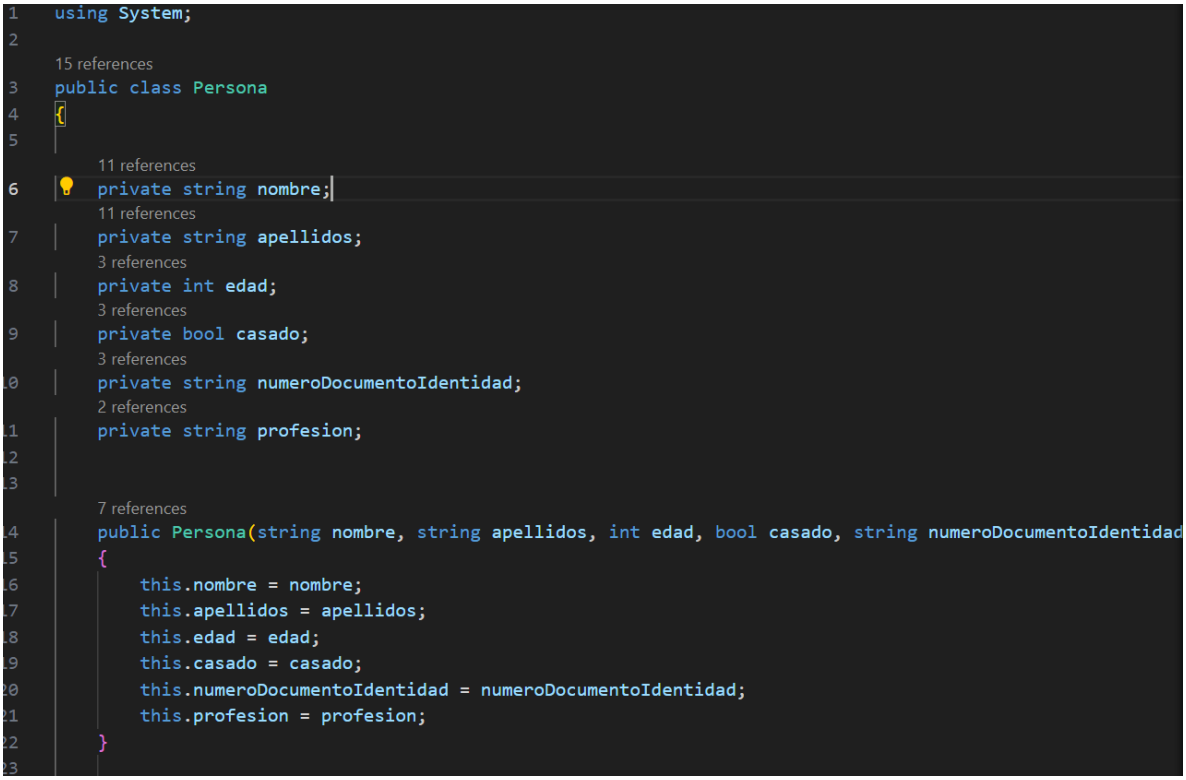
1. Considera estás desarrollando un programa donde necesitas trabajar con objetos de tipo Persona. Define una clase Persona, pero en este caso considerando los siguientes atributos de clase: nombre (String), apellidos (String), edad (int), casado (boolean), numeroDocumentoIdentidad(String) y 3 metodos como acciones diferentes por persona de acuerdo a una profesión. Define un constructor y los métodos para poder establecer y obtener los valores de los atributos. Mínimo 7 personas diferentes con acciones diferentes.



```
using System;

15 references
public class Persona
{

    11 references
    private string nombre;
    11 references
    private string apellidos;
    3 references
    private int edad;
    3 references
    private bool casado;
    3 references
    private string numeroDocumentoIdentidad;
    2 references
    private string profesion;


    7 references
    public Persona(string nombre, string apellidos, int edad, bool casado, string numeroDocumentoIdentidad
    {
        this.nombre = nombre;
        this.apellidos = apellidos;
        this.edad = edad;
        this.casado = casado;
        this.numeroDocumentoIdentidad = numeroDocumentoIdentidad;
        this.profesion = profesion;
    }
```

```
23
                0 references
24          public void EstablecerNombre(string nombre) => this.nombre = nombre;
            0 references
25      public string ObtenerNombre() => this.nombre;
26
            0 references
27      public void EstablecerApellidos(string apellidos) => this.apellidos = apellidos;
            0 references
28      public string ObtenerApellidos() => this.apellidos;
29
            0 references
30      public void EstablecerEdad(int edad) => this.edad = edad;
            0 references
31      public int ObtenerEdad() => this.edad;
32
            0 references
33      public void EstablecerCasado(bool casado) => this.casado = casado;
            0 references
34      public bool ObtenerCasado() => this.casado;
35
            0 references
36      public void EstablecerNumeroDocumentoIdentidad(string numeroDocumentoIdentidad) => this.numeroDocument
            0 references
37      public string ObtenerNumeroDocumentoIdentidad() => this.numeroDocumentoIdentidad;
38
```

```
38
39
            7 references
40      public string RealizarAccion()
41      {
42          switch (profesion)
43          {
44              case "Doctor":
45                  return $"{nombre} {apellidos} está atendiendo a un paciente.";
46              case "Ingeniero":
47                  return $"{nombre} {apellidos} está diseñando un nuevo proyecto.";
48              case "Profesor":
49                  return $"{nombre} {apellidos} está enseñando a sus alumnos.";
50              case "Abogado":
51                  return $"{nombre} {apellidos} está defendiendo a un cliente en la corte.";
52              case "Artista":
53                  return $"{nombre} {apellidos} está creando una nueva obra de arte.";
54              case "Cocinero":
55                  return $"{nombre} {apellidos} está preparando una deliciosa comida.";
56              case "Músico":
57                  return $"{nombre} {apellidos} está tocando en un concierto.";
58              default:
59                  return $"{nombre} {apellidos} está realizando una actividad.";
60          }
61      }
62  }
```

```
63
    0 references
64  public class Program
65  {
        0 references
66      public static void Main(string[] args)
67      {
68
69          Persona persona1 = new Persona("Juan", "Pérez", 30, true, "12345678", "Doctor");
70          Persona persona2 = new Persona("María", "Gómez", 28, false, "87654321", "Ingeniero");
71          Persona persona3 = new Persona("Luis", "Martínez", 35, true, "23456789", "Profesor");
72          Persona persona4 = new Persona("Ana", "López", 40, true, "34567890", "Abogado");
73          Persona persona5 = new Persona("Carlos", "Sánchez", 25, false, "45678901", "Artista");
74          Persona persona6 = new Persona("Laura", "Ramírez", 32, true, "56789012", "Cocinero");
75          Persona persona7 = new Persona("Pedro", "Hernández", 29, false, "67890123", "Músico");
76
77
78          Console.WriteLine(persona1.RealizarAccion());
79          Console.WriteLine(persona2.RealizarAccion());
80          Console.WriteLine(persona3.RealizarAccion());
81          Console.WriteLine(persona4.RealizarAccion());
82          Console.WriteLine(persona5.RealizarAccion());
83          Console.WriteLine(persona6.RealizarAccion());
84          Console.WriteLine(persona7.RealizarAccion());
85      }
86  }
```

2. Crea una clase Cuenta con los métodos ingreso, reintegro y transferencia. La clase contendrá un constructor por defecto, un constructor con parámetros y los métodos getters y setters para mostrar e ingresar.

```
using System;

7 references
public class Cuenta
{
    4 references
    private string numeroCuenta;
    4 references
    private string titular;
    8 references
    private decimal saldo;

    0 references
    public Cuenta()
    {
        numeroCuenta = "00000000";
        titular = "Desconocido";
        saldo = 0;
    }

    2 references
    public Cuenta(string numeroCuenta, string titular, decimal saldo)
    {
        this.numeroCuenta = numeroCuenta;
        this.titular = titular;
        this.saldo = saldo;
    }

    0 references
    public string ObtenerNumeroCuenta() => numeroCuenta;
    0 references
    public void EstablecerNumeroCuenta(string numeroCuenta) => this.numeroCuenta = numeroCuenta;
```

```
23      public string ObtenerNumeroCuenta() => numeroCuenta;
        0 references
24      public void EstablecerNumeroCuenta(string numeroCuenta) => this.numeroCuenta = numeroCuenta;
25
        0 references
26      public string ObtenerTitular() => titular;
        0 references
27      public void EstablecerTitular(string titular) => this.titular = titular;
28
        2 references
29      public decimal ObtenerSaldo() => saldo;
30
        2 references
31      public void Ingreso(decimal cantidad)
32      {
33          if (cantidad > 0)
34          {
35              saldo += cantidad;
36          }
37      }
38
        1 reference
39      public void Reintegro(decimal cantidad)
40      {
41          if (cantidad > 0 && cantidad <= saldo)
42          {
43              saldo -= cantidad;
44          }
45      }
```

```
        1 reference
39      public void Reintegro(decimal cantidad)
40      {
41          if (cantidad > 0 && cantidad <= saldo)
42          {
43              saldo -= cantidad;
44          }
45      }
46
        1 reference
47      public void Transferencia(Cuenta cuentaDestino, decimal cantidad)
48      {
49          if (cantidad > 0 && cantidad <= saldo)
50          {
51              saldo -= cantidad;
52              cuentaDestino.Ingreso(cantidad);
53          }
54      }
55  }
56
    0 references
57  public class Program
58  {
        0 references
59      public static void Main(string[] args)
60      {
61          Cuenta cuenta1 = new Cuenta("12345678", "Juan Pérez", 1000);
62          Cuenta cuenta2 = new Cuenta("87654321", "María Gómez", 500);
```

```
0 references
public class Program
{
    0 references
    public static void Main(string[] args)
    {
        Cuenta cuenta1 = new Cuenta("12345678", "Juan Pérez", 1000);
        Cuenta cuenta2 = new Cuenta("87654321", "María Gómez", 500);

        cuenta1.Ingreso(200);
        cuenta1.Reintegro(100);
        cuenta1.Transferencia(cuenta2, 300);

        Console.WriteLine($"Saldo cuenta 1: {cuenta1.ObtenerSaldo()}");
        Console.WriteLine($"Saldo cuenta 2: {cuenta2.ObtenerSaldo()}");
    }
}
```

3. Crea una clase Contador con los métodos para incrementar y decrementar el contador. La clase contendrá un constructor por defecto, un constructor con parámetros, y los métodos getters y setters.

```
using System;

6 references
public class Contador
{
    6 references
    private int valor;

    1 reference
    public Contador()
    {
        valor = 0;
    }

    1 reference
    public Contador(int valorInicial)
    {
        valor = valorInicial;
    }

    2 references
    public int ObtenerValor() => valor;

    0 references
    public void EstablecerValor(int valor)
    {
        this.valor = valor;
    }

    3 references
    public void Incrementar()
```

```
3  public class Contador
24     public void Incrementar()
26         valor++;
27     }
28
       1 reference
29     public void Decrementar()
30     {
31         valor--;
32     }
33 }
34
       0 references
35 public class Program
36 {
       0 references
37     public static void Main(string[] args)
38     {
39         Contador contador1 = new Contador();
40         Contador contador2 = new Contador(10);
41
42         contador1.Incrementar();
43         contador1.Incrementar();
44         contador1.Decrementar();
45
46         contador2.Incrementar();
47
48         Console.WriteLine($"Valor contador 1: {contador1.ObtenerValor()}");
49         Console.WriteLine($"Valor contador 2: {contador2.ObtenerValor()}");
50     }
```

4. Crea una clase Libro con los métodos préstamo, devolución y ToString. La clase contendrá un constructor por defecto, un constructor con parámetros y los métodos getters y setters.

```
using System;

public class Libro
{
    private string titulo;
    private string autor;
    private bool prestado;

    public Libro()
    {
        titulo = "Desconocido";
        autor = "Desconocido";
        prestado = false;
    }

    public Libro(string titulo, string autor)
    {
        this.titulo = titulo;
        this.autor = autor;
        prestado = false;
    }

    public string ObtenerTitulo() => titulo;
    public void EstablecerTitulo(string titulo) => this.titulo = titulo;

    public string ObtenerAutor() => autor;
    public void EstablecerAutor(string autor) => this.autor = autor;

    public bool EstaPrestado() => prestado;

    public void Prestamo()
    {
        if (!prestado)
        {
            prestado = true;
        }
        else
        {
            Console.WriteLine("El libro ya está prestado.");
        }
    }

    public void Devolucion()
    {…
    }

    public override string ToString()
    {
        return $"Título: {titulo}, Autor: {autor}, Prestado: {prestado}";
    }
}

public class Program
{
    public static void Main(string[] args)
    {
        Libro libro1 = new Libro("Cien años de soledad", "Gabriel García Márquez");
        Console.WriteLine(libro1.ToString());
```

```
    public override string ToString()
    {
        return $"Título: {titulo}, Autor: {autor}, Prestado: {prestado}";
    }
}

public class Program
{
    public static void Main(string[] args)
    {
        Libro libro1 = new Libro("Cien años de soledad", "Gabriel García Márquez");
        Console.WriteLine(libro1.ToString());

        libro1.Prestamo();
        Console.WriteLine(libro1.ToString());

        libro1.Devolucion();
        Console.WriteLine(libro1.ToString());
    }
}
```

5. Crea una clase Fracción con métodos para sumar, restar, multiplicar y dividir fracciones.

```
using System;

public class Fraccion
{
    private int numerador;
    private int denominador;

    public Fraccion(int numerador, int denominador)
    {
        if (denominador == 0)
        {
            throw new ArgumentException("El denominador no puede ser cero.");
        }
        this.numerador = numerador;
        this.denominador = denominador;
        Simplificar();
    }

    private void Simplificar()
    {
        int gcd = ObtenerMCD(numerador, denominador);
        numerador /= gcd;
        denominador /= gcd;
    }

    private int ObtenerMCD(int a, int b)
    {
        while (b != 0)
        {
            int temp = b;
```

```
    {
        while (b != 0)
        {
            int temp = b;
            b = a % b;
            a = temp;
        }
        return Math.Abs(a);
    }

    public Fraccion Sumar(Fraccion otra)
    {
        int nuevoNumerador = numerador * otra.denominador + otra.numerador * denominador;
        int nuevoDenominador = denominador * otra.denominador;
        return new Fraccion(nuevoNumerador, nuevoDenominador);
    }

    public Fraccion Restar(Fraccion otra)
    {
        int nuevoNumerador = numerador * otra.denominador - otra.numerador * denominador;
        int nuevoDenominador = denominador * otra.denominador;
        return new Fraccion(nuevoNumerador, nuevoDenominador);
    }

    public Fraccion Multiplicar(Fraccion otra)
    {
        int nuevoNumerador = numerador * otra.numerador;
        int nuevoDenominador = denominador * otra.denominador;
        return new Fraccion(nuevoNumerador, nuevoDenominador);
    }
```

```
    public Fraccion Dividir(Fraccion otra)
    {
        if (otra.numerador == 0)
        {
            throw new DivideByZeroException("No se puede dividir por cero.");
        }
        int nuevoNumerador = numerador * otra.denominador;
        int nuevoDenominador = denominador * otra.numerador;
        return new Fraccion(nuevoNumerador, nuevoDenominador);
    }

    public override string ToString()
    {
        return $"{numerador}/{denominador}";
    }
}

public class Program
{
    public static void Main(string[] args)
    {
        Fraccion fraccion1 = new Fraccion(1, 2);
        Fraccion fraccion2 = new Fraccion(1, 3);

        Fraccion suma = fraccion1.Sumar(fraccion2);
        Fraccion resta = fraccion1.Restar(fraccion2);
        Fraccion multiplicacion = fraccion1.Multiplicar(fraccion2);
        Fraccion division = fraccion1.Dividir(fraccion2);
```

```
public class Program
{
    public static void Main(string[] args)
    {
        Fraccion fraccion1 = new Fraccion(1, 2);
        Fraccion fraccion2 = new Fraccion(1, 3);

        Fraccion suma = fraccion1.Sumar(fraccion2);
        Fraccion resta = fraccion1.Restar(fraccion2);
        Fraccion multiplicacion = fraccion1.Multiplicar(fraccion2);
        Fraccion division = fraccion1.Dividir(fraccion2);

        Console.WriteLine($"Suma: {suma}");
        Console.WriteLine($"Resta: {resta}");
        Console.WriteLine($"Multiplicación: {multiplicacion}");
```