

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.

```
class Persona
{

    private string nombre;
    private string apellidos;
    private int edad;
    private bool casado;
    private string numeroDocumentoIdentidad;

    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;
    }

    // Métodos obtener valores
    public string GetNombre() => nombre;
    public string GetApellidos() => apellidos;
    public int GetEdad() => edad;
    public bool GetCasado() => casado;
    public string GetNumeroDocumentoIdentidad() => numeroDocumentoIdentidad;

    // Métodos establecer valores
    public void SetNombre(string nombre) => this.nombre = nombre;
    public void SetApellidos(string apellidos) => this.apellidos = apellidos;
    public void SetEdad(int edad) => this.edad = edad;
    public void SetCasado(bool casado) => this.casado = casado;
    public void SetNumeroDocumentoIdentidad(string numeroDocumentoIdentidad) =>
this.numeroDocumentoIdentidad = numeroDocumentoIdentidad;

    // Métodos profesión
    public void Medico_AtenderPaciente()
    {
        Console.WriteLine($"{nombre} {apellidos} está atendiendo a un paciente
en la sala de emergencias.");
    }

    public void Ingeniero_DiseñarProyecto()
```

```

    {
        Console.WriteLine($"{nombre} {apellidos} está diseñando un puente para
mejorar la infraestructura de la ciudad.");
    }

    public void Profesor_EnseñarClase()
    {
        Console.WriteLine($"{nombre} {apellidos} está enseñando matemáticas a
sus alumnos.");
    }

    public void Bombero_ApagarIncendio()
    {
        Console.WriteLine($"{nombre} {apellidos} está apagando un incendio en
un edificio.");
    }

    public void Policia_Patrullar()
    {
        Console.WriteLine($"{nombre} {apellidos} está patrullando las calles
para garantizar la seguridad.");
    }

    public void Chef_Cocinar()
    {
        Console.WriteLine($"{nombre} {apellidos} está preparando un platillo
gourmet en su restaurante.");
    }

    public void Artista_PintarCuadro()
    {
        Console.WriteLine($"{nombre} {apellidos} está pintando una obra maestra
en su estudio.");
    }
}

class Program
{
    static void Main()
    {
        Persona medico = new Persona("Carlos", "González", 45, true,
"123456789");
        Persona ingeniero = new Persona("Laura", "Fernández", 38, false,
"987654321");
        Persona profesor = new Persona("Miguel", "Hernández", 50, true,
"456123789");
        Persona bombero = new Persona("Sofía", "Ramírez", 32, false,
"321654987");
        Persona policia = new Persona("Andrés", "Martínez", 40, true,
"789321654");
        Persona chef = new Persona("Elena", "Sánchez", 29, false, "159753486");
        Persona artista = new Persona("Pablo", "López", 35, false,
"753159486");

        medico.Medico_AtenderPaciente();
        ingeniero.Ingeniero_DiseñarProyecto();
    }
}

```

```
profesor.Profesor_EnseñarClase();  
bombero.Bombero_ApagarIncendio();  
policia.Policia_Patrullar();  
chef.Chef_Cocinar();  
artista.Artista_PintarCuadro();  
}  
}
```

- 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.**

```
class Cuenta
{
    private string titular;
    private double saldo;

    public Cuenta()
    {
        titular = "Desconocido";
        saldo = 0.0;
    }

    public Cuenta(string titular, double saldoInicial)
    {
        this.titular = titular;
        this.saldo = saldoInicial;
    }

    public string GetTitular() => titular;
    public void SetTitular(string titular) => this.titular = titular;

    public double GetSaldo() => saldo;
    public void SetSaldo(double saldo) => this.saldo = saldo;

    public void Ingreso(double cantidad)
    {
        if (cantidad > 0)
        {
            saldo += cantidad;
            Console.WriteLine($"Ingreso de {cantidad} realizado. Nuevo saldo:
{saldo}");
        }
        else
        {
            Console.WriteLine("La cantidad a ingresar debe ser mayor que 0.");
        }
    }

    public void Reintegro(double cantidad)
    {
        if (cantidad > 0 && cantidad <= saldo)
        {
            saldo -= cantidad;
            Console.WriteLine($"Reintegro de {cantidad} realizado. Nuevo saldo:
{saldo}");
        }
    }
}
```

```

    }
    else
    {
        Console.WriteLine("Fondos insuficientes o cantidad inválida.");
    }
}

public void Transferencia(Cuenta cuentaDestino, double cantidad)
{
    if (cantidad > 0 && cantidad <= saldo)
    {
        saldo -= cantidad;
        cuentaDestino.Ingreso(cantidad);
        Console.WriteLine($"Transferencia de {cantidad} realizada a la
cuenta de {cuentaDestino.GetTitular()}. Nuevo saldo: {saldo}");
    }
    else
    {
        Console.WriteLine("Fondos insuficientes o cantidad inválida.");
    }
}
}

class Program
{
    static void Main()
    {
        Cuenta cuenta1 = new Cuenta("Juan Pérez", 500.0);
        Cuenta cuenta2 = new Cuenta("María García", 300.0);

        cuenta1.Ingreso(200);
        cuenta1.Reintegro(100);
        cuenta1.Transferencia(cuenta2, 150);

        Console.WriteLine($"Saldo final de la cuenta de {cuenta1.GetTitular()}:
{cuenta1.GetSaldo()}");
        Console.WriteLine($"Saldo final de la cuenta de {cuenta2.GetTitular()}:
{cuenta2.GetSaldo()}");
    }
}

```

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.

```
5 6 references
6  class Contador
7  {
8      private int valor;
9
10     1 reference
11     public Contador()
12     {
13         valor = 0;
14     }
15
16     1 reference
17     public Contador(int valorInicial)
18     {
19         valor = valorInicial;
20     }
21
22     2 references
23     public int GetValor() => valor;
24     0 references
25     public void SetValor(int valor) => this.valor = valor;
26
27     2 references
28     public void Incrementar()
29     {
30         valor++;
31         Console.WriteLine($"Contador incrementado. Valor actual: {valor}");
32     }
33
34     2 references
35     public void Decrementar()
36     {
37         valor--;
38         Console.WriteLine($"Contador decrementado. Valor actual: {valor}");
39     }
40 }
41
42 0 references
43 class Program
44 {
45     0 references
46     static void Main()
47     {
48         Contador contador1 = new Contador();
49
50         Contador contador2 = new Contador(10);
51
52         contador1.Incrementar();
53         contador1.Decrementar();
54
55         contador2.Incrementar();
56         contador2.Decrementar();
57
58         Console.WriteLine($"Valor final de contador1: {contador1.GetValor()}");
59         Console.WriteLine($"Valor final de contador2: {contador2.GetValor()}");
60     }
61 }
62
63
```

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.

```
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;
        this.prestado = false;
    }

    public string GetTitulo() => titulo;
    public void SetTitulo(string titulo) => this.titulo = titulo;

    public string GetAutor() => autor;
    public void SetAutor(string autor) => this.autor = autor;

    public bool GetPrestado() => prestado;
    public void SetPrestado(bool prestado) => this.prestado = prestado;

    public void Prestamo()
    {
        if (!prestado)
        {
            prestado = true;
            Console.WriteLine($"El libro '{titulo}' de {autor} ha sido prestado.");
        }
        else
        {
            Console.WriteLine($"El libro '{titulo}' de {autor} ya está prestado.");
        }
    }

    public void Devolucion()
    {
        if (prestado)
        {
            prestado = false;
        }
    }
}
```

```

        Console.WriteLine($"El libro '{titulo}' de {autor} ha sido
devuelto.");
    }
    else
    {
        Console.WriteLine($"El libro '{titulo}' de {autor} no estaba
prestado.");
    }
}

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

class Program
{
    static void Main()
    {
        Libro libro1 = new Libro();

        Libro libro2 = new Libro("Cien años de soledad", "Gabriel García
Márquez");

        libro1.Prestamo();
        libro1.Devolucion();

        libro2.Prestamo();
        libro2.Devolucion();

        Console.WriteLine(libro1.ToString());
        Console.WriteLine(libro2.ToString());
    }
}

```


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

```
3  class Fraccion
4  {
5      private int numerador;
6      private int denominador;
7
8      0 references
9      public Fraccion()
10     {
11         numerador = 0;
12         denominador = 1;
13     }
14
15     6 references
16     public Fraccion(int numerador, int denominador)
17     {
18         if (denominador == 0)
19         {
20             throw new ArgumentException("El denominador no puede ser cero.");
21         }
22         this.numerador = numerador;
23         this.denominador = denominador;
24     }
25
26     0 references
27     public int GetNumerador() => numerador;
28     0 references
29     public void SetNumerador(int numerador) => this.numerador = numerador;
30
31     0 references
32     public int GetDenominador() => denominador;
33     0 references
34     public void SetDenominador(int denominador)
35     {
36         if (denominador == 0)
37         {
38             throw new ArgumentException("El denominador no puede ser cero.");
39         }
40         this.denominador = denominador;
41     }
42 }
```

```

36 public Fraccion Sumar(Fraccion otra)
37 {
38     int nuevoNumerador = (numerador * otra.denominador) + (otra.numerador * denominador);
39     int nuevoDenominador = denominador * otra.denominador;
40     return new Fraccion(nuevoNumerador, nuevoDenominador);
41 }
42 1 reference
43 public Fraccion Restar(Fraccion otra)
44 {
45     int nuevoNumerador = (numerador * otra.denominador) - (otra.numerador * denominador);
46     int nuevoDenominador = denominador * otra.denominador;
47     return new Fraccion(nuevoNumerador, nuevoDenominador);
48 }
49 1 reference
50 public Fraccion Multiplicar(Fraccion otra)
51 {
52     int nuevoNumerador = numerador * otra.numerador;
53     int nuevoDenominador = denominador * otra.denominador;
54     return new Fraccion(nuevoNumerador, nuevoDenominador);
55 }
56 1 reference
57 public Fraccion Dividir(Fraccion otra)
58 {
59     if (otra.numerador == 0)
60     {
61         throw new DivideByZeroException("No se puede dividir por una fracción con numerador 0.");
62     }
63     int nuevoNumerador = numerador * otra.denominador;
64     int nuevoDenominador = denominador * otra.numerador;
65     return new Fraccion(nuevoNumerador, nuevoDenominador);
66 }
67 0 references
68 public override string ToString()
69 {
70     return $"{numerador}/{denominador}";
71 }

```

```

70 class Program
71 {
72     0 references
73     static void Main()
74     {
75         Fraccion fraccion1 = new Fraccion(1, 2); // 1/2
76         Fraccion fraccion2 = new Fraccion(3, 4); // 3/4
77
78         // Sumar ;
79         Fraccion suma = fraccion1.Sumar(fraccion2);
80         Console.WriteLine($"Suma: {suma}");
81
82         // Restar ;
83         Fraccion resta = fraccion1.Restar(fraccion2);
84         Console.WriteLine($"Resta: {resta}");
85
86         // Multiplicar ;
87         Fraccion multiplicacion = fraccion1.Multiplicar(fraccion2);
88         Console.WriteLine($"Multiplicación: {multiplicacion}");
89
90         // Dividir ;
91         Fraccion division = fraccion1.Dividir(fraccion2);
92         Console.WriteLine($"División: {division}");
93     }
94 }

```