Minombre: Juan Sebastian Peña Estevez,Mimatricula;2024-1732,Diade,clase,Lunes6pm

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;

class Program
{
   static void Main()
   {
      Doctor persona1 = new Doctor();
      persona1.Trabajar();

      Mecanico persona2 = new Mecanico();
      persona2.Trabajar();

      Chef persona3 = new Chef();
      persona3.Trabajar();

      Pintor persona4 = new Pintor();
      persona4.Trabajar();

      Cajero persona5 = new Cajero();
      persona5.Trabajar();
```

```
        Console.WriteLine("\nFin del programa...");
    }
}

class Persona
{
    public string Nombre { get; set; }
    public int Edad { get; set; }
    public string Profesion { get; set; }

    public Persona(string nombre, int edad, string profesion)
    {
        Nombre = nombre;
        Edad = edad;
        Profesion = profesion;
    }

    public virtual void Trabajar()
    {
        Console.WriteLine($"{Nombre}, {Profesion}, está trabajando.");
    }
}

class Doctor : Persona
{
    public Doctor() : base("Carlos", 35, "Doctor") { }

    public override void Trabajar()
    {
        Console.WriteLine($"{Nombre} está atendiendo a un paciente.");
    }
}

class Mecanico : Persona
{
```

```
    public Mecanico() : base("Luis", 28, "Mecánico") { }

    public override void Trabajar()
    {
        Console.WriteLine($"{Nombre} está reparando un automóvil.");
    }
}

class Chef : Persona
{
    public Chef() : base("Ana", 30, "Chef") { }

    public override void Trabajar()
    {
        Console.WriteLine($"{Nombre} está cocinando un platillo
delicioso.");
    }
}

class Pintor : Persona
{
    public Pintor() : base("Marcos", 40, "Pintor") { }

    public override void Trabajar()
    {
        Console.WriteLine($"{Nombre} está pintando un mural.");
    }
}

class Cajero : Persona
{
    public Cajero() : base("Laura", 25, "Cajero") { }

    public override void Trabajar()
    {
```

```
        Console.WriteLine($"{Nombre} está cobrando en la caja
registradora.");
    }
}
```

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;

public class Cuenta
{
    private string _titular;
    private decimal _saldo;

    // Constructor por defecto
    public Cuenta()
    {
        _titular = string.Empty;
        _saldo = 0.0m;
    }

    // Constructor con parámetros
    public Cuenta(string titular, decimal saldo)
    {
        _titular = titular;
        _saldo = saldo;
    }

    // Métodos getters
    public string GetTitular()
    {
```

```
        return _titular;
    }

    public decimal GetSaldo()
    {
        return _saldo;
    }

    // Métodos setters
    public void SetTitular(string titular)
    {
        _titular = titular;
    }

    public void SetSaldo(decimal saldo)
    {
        _saldo = saldo;
    }

    // Método para ingresar dinero
    public void Ingreso(decimal 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
cero.");
        }
    }
```

```
    // Método para reintegrar dinero
    public void Reintegro(decimal cantidad)
    {
        if (cantidad > 0 && cantidad <= _saldo)
        {
            _saldo -= cantidad;
            Console.WriteLine($"Reintegro de {cantidad} realizado. Nuevo
saldo: {_saldo}");
        }
        else
        {
            Console.WriteLine("La cantidad a reintegrar debe ser mayor que
cero y menor o igual al saldo disponible.");
        }
    }

    // Método para transferir dinero a otra cuenta
    public void Transferencia(decimal cantidad, Cuenta cuentaDestino)
    {
        if (cantidad > 0 && cantidad <= _saldo)
        {
            Reintegro(cantidad); // Realiza el reintegro en la cuenta actual
            cuentaDestino.Ingreso(cantidad); // Realiza el ingreso en la
cuenta destino
            Console.WriteLine($"Transferencia de {cantidad} a
{cuentaDestino.GetTitular()} realizada.");
        }
        else
        {
            Console.WriteLine("La cantidad a transferir debe ser mayor que
cero y menor o igual al saldo disponible.");
        }
    }
}
```

```
// Ejemplo de uso
public class Program
{
    public static void Main(string[] args)
    {
        Cuenta cuenta1 = new Cuenta("Juan", 1000);
        Cuenta cuenta2 = new Cuenta("Maria", 500);

        cuenta1.Ingreso(200);
        cuenta1.Reintegro(150);
        cuenta1.Transferencia(300, cuenta2);

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

```
 using System;


public class Contador

{

   private int _valor;
```

```
// Constructor por defecto
public Contador()
{
    _valor = 0; // Inicializa el contador en 0
}

// Constructor con parámetros
public Contador(int valorInicial)
{
    _valor = valorInicial; // Inicializa el contador con el valor proporcionado
}

// Método getter
public int GetValor()
{
    return _valor;
}

// Método setter
public void SetValor(int valor)
{
    _valor = valor;
}
```

```
    // Método para incrementar el contador
    public void Incrementar()
    {
        _valor++;
        Console.WriteLine($"Contador incrementado. Nuevo valor: {_valor}");
    }

    // Método para decrementar el contador
    public void Decrementar()
    {
        _valor--;
        Console.WriteLine($"Contador decrementado. Nuevo valor: {_valor}");
    }
}

// Ejemplo de uso
public class Program
{
    public static void Main(string[] args)
    {
        Contador contador1 = new Contador(); // Usando el constructor por
defecto
        contador1.Incrementar(); // Incrementa a 1
        contador1.Incrementar(); // Incrementa a 2
        contador1.Decrementar(); // Decrementa a 1
```

```
        Contador contador2 = new Contador(5); // Usando el constructor con
parámetros
        contador2.Decrementar(); // Decrementa a 4
        contador2.Incrementar(); // Incrementa a 5

        Console.WriteLine($"Valor del contador 1: {contador1.GetValor()}");
        Console.WriteLine($"Valor del contador 2: {contador2.GetValor()}");
    }
}
```

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
{
    public string Titulo { get; set; }
    public string Autor { get; set; }
    public bool Prestado { get; private set; }

    // Constructor por defecto
    public Libro() : this("Sin título", "Sin autor") { }

    // Constructor con parámetros
    public Libro(string titulo, string autor)
    {
        Titulo = titulo;
        Autor = autor;
        Prestado = false;
```

```
    }

    // Método para prestar el libro
    public void Prestamo()
    {
        if (!Prestado)
        {
            Prestado = true;
            Console.WriteLine($"El libro '{Titulo}' ha sido prestado.");
        }
        else
        {
            Console.WriteLine($"El libro '{Titulo}' ya está prestado.");
        }
    }

    // Método para devolver el libro
    public void Devolucion()
    {
        if (Prestado)
        {
            Prestado = false;
            Console.WriteLine($"El libro '{Titulo}' ha sido devuelto.");
        }
        else
        {
            Console.WriteLine($"El libro '{Titulo}' no está prestado.");
        }
    }

    // Método ToString
    public override string ToString() => $"Título: {Titulo}, Autor: {Autor},
Prestado: {(Prestado ? "Sí" : "No")}";
}
```

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

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

        libro1.Devolucion();
        Console.WriteLine(libro1);

        libro1.Devolucion(); // Intento de devolver un libro que no está
prestado
    }
}
```

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

```
using System;

public class Fraccion
{
    public int Numerador { get; private set; }
    public int Denominador { get; private set; }

    public Fraccion(int numerador, int denominador)
    {
        Numerador = numerador;
```

```
        Denominador = denominador == 0 ? 1 : denominador; // Evitar
división por cero
        Simplificar();
    }

    public Fraccion Sumar(Fraccion otra) =>
        new Fraccion(Numerador * otra.Denominador + otra.Numerador *
Denominador, Denominador * otra.Denominador);

    public Fraccion Restar(Fraccion otra) =>
        new Fraccion(Numerador * otra.Denominador - otra.Numerador *
Denominador, Denominador * otra.Denominador);

    public Fraccion Multiplicar(Fraccion otra) =>
        new Fraccion(Numerador * otra.Numerador, Denominador *
otra.Denominador);

    public Fraccion Dividir(Fraccion otra) =>
        new Fraccion(Numerador * otra.Denominador, Denominador *
otra.Numerador);

    private void Simplificar()
    {
        int gcd = ObtenerMCD(Math.Abs(Numerador),
Math.Abs(Denominador));
        Numerador /= gcd;
        Denominador /= gcd;
        if (Denominador < 0)
        {
            Numerador = -Numerador;
            Denominador = -Denominador;
        }
    }

    private int ObtenerMCD(int a, int b)
```

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

    public override string ToString() => $"{Numerador}/{Denominador}";
}

public class Program
{
    public static void Main(string[] args)
    {
        Fraccion f1 = new Fraccion(1, 2);
        Fraccion f2 = new Fraccion(3, 4);

        Console.WriteLine($"Fracción 1: {f1}");
        Console.WriteLine($"Fracción 2: {f2}");
        Console.WriteLine($"Suma: {f1.Sumar(f2)}");
        Console.WriteLine($"Resta: {f1.Restar(f2)}");
        Console.WriteLine($"Multiplicación: {f1.Multiplicar(f2)}");
        Console.WriteLine($"División: {f1.Dividir(f2)}");
    }
}
```