Evaluación Práctica en PC (Parcial 1) - Tema: A1 Auto

Criterios de Evaluación: correcto funcionamiento (compilación/ejecución sin errores), uso de convenciones, formateo, estándares y buenas prácticas.

Consignas

Cree una **carpeta** con su **ApellidoNombre** que deberá contener una **solución .NET** y todo lo solicitado.
Los nombres de la/los solución/proyectos/namespaces deben tener de prefijo su "ApellidoNombre." Ej PerezJuan.X. El entregable debe estar zipeado y nombrado como ApellidoNombre.zip

Dentro de la carpeta cree la **solución .NET** conteniendo los proyectos:

- ApellidoNombre.Dominio del tipo Biblioteca de Clases que debe contener los puntos 1, 2 y 3
- ApellidoNombre.Consola del tipo Aplicación de Consola

1. **Agregar la siguiente clase Vehiculo:**

```
public abstract class Vehiculo {
	public string Patente { get; set; }
	public int Ruedas { get; set; }
	public int Modelo { get; set; }

	protected Vehiculo(string patente, int ruedas, int modelo) {
		this.Patente = patente;
		this.Ruedas = ruedas;
		this.Modelo = modelo;
	}

	public override string ToString() {
		return $"Patente: {Patente} Ruedas: {Ruedas} Modelo: {Modelo}";
	}
}
```

2. **Crear una clase Auto que herede de Vehiculo y tenga:**
   a. Una propiedad **Color** de tipo String.
   b. Un constructor que reciba **color** y todos los datos del **Vehiculo**, el que debe llamar al constructor de la clase superior e inicializar **Color** / la propiedad correspondiente con el argumento recibido.
   c. Una sobreescritura del método ToString, invocando al método de la clase superior agregando el color.

3. **Crear una clase ListaVehiculo en el mismo proyecto que contenga dos métodos estáticos:**
   a. BuscarPatenteLinq que recibe una lista de objetos **Vehiculo** y una **patente** y retorne el objeto que contenga la **patente** buscada / coincida con el argumento buscado/recibido. Implementar usando LINQ.
   b. BuscarPatenteIterativa: es una función equivalente a la anterior pero debe ser implementada iterando sobre la lista que se recibe como argumento.

4. **En el Proyecto de Aplicación de Consola realizar lo siguiente:**
   a. Dentro del main crear una lista de objetos **Vehiculo**.
   b. Dentro de la misma almacenar tres objetos **Auto**, uno de ellos debe ser el siguiente:
   `new Auto("ABC111", 4, 1999, "Azul");`
   c. Llamar a BuscarPatenteLinq y BuscarPatenteIterativa con la lista creada previamente y la **patente** "ABC111".
   d. Mostrar por pantalla los datos del objeto recuperado en el punto anterior, utilizando el método ToString.

Evaluación Práctica en PC (Parcial 1) - Tema: A2 Moto

Criterios de Evaluación: correcto funcionamiento (compilación/ejecución sin errores), uso de convenciones, formateo, estándares y buenas prácticas.

Consignas

Cree una **carpeta** con su **ApellidoNombre** que deberá contener una **solución .NET** y todo lo solicitado.
Los nombres de la/los solución/proyectos/namespaces deben tener de prefijo su "ApellidoNombre." Ej PerezJuan.X. El entregable debe estar zipeado y nombrado como ApellidoNombre.zip

Dentro de la carpeta cree la **solución .NET** conteniendo los proyectos:

- ApellidoNombre.Dominio del tipo Biblioteca de Clases que debe contener los puntos 1, 2 y 3
- ApellidoNombre.Consola del tipo Aplicación de Consola

1. **Agregar la siguiente clase Vehiculo:**

```
public abstract class Vehiculo {
     public string Patente { get; set; }
     public int Ruedas { get; set; }
     public int Modelo { get; set; }

     protected Vehiculo(string patente, int ruedas, int modelo) {
          this.Patente = patente;
          this.Ruedas = ruedas;
          this.Modelo = modelo;
     }

     public override string ToString() {
          return $"Patente: {Patente} Ruedas: {Ruedas} Modelo: {Modelo}";
     }
}
```

2. **Crear una clase Moto que herede de Vehiculo y tenga:**
   a. Una propiedad **Color** de tipo String.
   b. Un constructor que reciba **color** y todos los datos del **Vehiculo**, el que debe llamar al constructor de la clase superior e inicializar **Color** con el argumento recibido.
   c. Una sobreescritura del método ToString, invocando al método de la clase superior agregando el color.

3. **Crear una clase ListaVehiculo en el mismo proyecto que contenga dos métodos estáticos:**
   a. BuscarPatenteLinq que recibe una lista de objetos **Vehiculo** y una patente y retorne el objeto que contenga la patente buscada. Implementar usando LINQ.
   b. BuscarPatenteIterativa: es una función equivalente a la anterior pero debe ser implementada iterando sobre la lista que se recibe como argumento.

4. **En el Proyecto de Aplicación de Consola realizar lo siguiente:**
   a. Dentro del main crear una lista de objetos **Vehiculo**.
   b. Dentro de la misma almacenar tres objetos **Moto**, uno de ellos debe ser el siguiente:
   `new Moto("123ABC", 2, 1999, "Amarillo");`
   c. Llamar a BuscarPatenteLinq y BuscarPatenteIterativa con la lista creada previamente y la patente "123ABC".
   d. Mostrar por pantalla los datos del objeto recuperado en el punto anterior, utilizando el método ToString.