Documentación del Proceso

Introducción

Este documento explica los pasos seguidos para desarrollar la tarea de programación orientada a objetos en C#. Se trabajó con la jerarquía de clases de vehículos, aplicando conceptos como herencia, encapsulación y polimorfismo. En particular, se detallan los cambios realizados en la clase base Vehiculo y las modificaciones específicas en AutoDeCombustion y Motocicleta.

Punto 1: Derivar la clase AutoDeCombustion de la clase Vehiculo

Para crear un modelo realista de un auto a combustión, definí la clase AutoDeCombustion como una subclase de Vehiculo. Esto permite que herede atributos y métodos de la clase base, como Marca, Modelo, Acelerar, Frenar, Encender y Apagar.

Public class AutoDeCombustion : Vehiculo

Además, agregué atributos específicos como:

: Representa la cantidad de combustible disponible en el tanque.

Capacidad Tanque: Indica la capacidad total del tanque de combustible.

Tipo Combustible: Define si el auto usa gasolina o diésel.

Para que el vehículo gaste combustible de forma realista, sobrescribí el método Acelerar, de manera que al incrementar la velocidad, se reduzca el nivel de combustible. También sobrescribí Frenar para simular un pequeño consumo de combustible al frenar.

```
Public override void Acelerar(int cuanto)
{
   Base  Acelerar(cuanto);
   Nivel Combustible -= cuanto / 10; // Gasta combustible al acelerar
   Console  WriteLine("Nivel de combustible: {0}%", Nivel de combustible);
}
```

Esto hace que AutoDeCombustion tenga un comportamiento más cercano a la realidad, ya que acelerar o frenar tiene un impacto en el consumo de combustible.

Punto 2: Encapsular la propiedad velocidad y sobrescribir el método Frenar en Motocicleta

Para mejorar la seguridad del código y controlar mejor los accesos a los atributos, encapsulé la propiedad velocidad en la clase Vehiculo. Ahora, solo puede modificarse dentro de la propia clase o por sus clases derivadas mediante protected set:

```
Private int velocidad = 0;

Public int Velocidad
{
   Get { return velocidad; }
```

```
    Protected set { velocidad = value; }
}
```

Luego, en Motocicleta, sobrescribí el método Acelerar para hacer que la moto incremente su velocidad más rápido que un auto, dado que las motocicletas suelen ser más ágiles.

```
Public override void Acelerar(int cuanto)
{
    Base.Acelerar(cuanto * 2); // La motocicleta acelera el doble de rápido
}
```

También sobrescribí el método Frenar, agregando un mensaje adicional al frenar:

```
Public override void Frenar(int cuanto)
{
    Base.Frenar.
```