

# DOCUMENTACION

Juan Daniel Barrera Osuna 0905-24-14875



programación I

Partiendo del código base que nos ha proporcionado modifique algunos métodos de la base principal y agregue unos arreglos convenientes para mi al momento de hacer pruebas como lo puede ser el siguiente

```
p1bpoo - p1bpoo.MisClases.Vehiculo - frenar()
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace p1bpoo.MisClases
{
    8 references
    internal class Vehiculo
    {
        2 references
        public string Color { get; set; }
        2 references
        public string Modelo { get; }
        2 references
        public int Year { get; }
        protected int velocidad = 0;
        4 references
        public int Velocidadactual => velocidad; //solo sirve para conocer la velocidad actual, pero no modificarla
        //La verdad es solo para el ciclo while para subir la velocidad sin estar llamando a cada rato
    }
}
```


Hice otra característica llamada “Velocidadactual” que únicamente me da el valor de la velocidad que está restringida y que es visible únicamente por los métodos que derivan de la clase base.

Agregue como método base “frenar”

```
10 references
public virtual void frenar()
{
    while (velocidad > 0)
    {
        velocidad -= Math.Min(10, velocidad);
        Console.WriteLine("Has disminuido la velocidad a {0} KMS / Hora", velocidad);
        System.Threading.Thread.Sleep(1000);
    }
}
```

Me parecía poco ilógico que la velocidad disminuya y aumente tan rápido por eso se priva la velocidad y los métodos de acelerar y desacelerar están limitados a cierta cantidad por cada llamada, pero también pensé que si agregaba la función frenar era lo mismo que desacelerar pero llamando la función hasta llegar a 0, así que investigue un poco y con esas líneas de código consigo un que la función que disminuye la velocidad de 10 en 10 se frene con un intervalo, suspendiendo la función por un momento y reanudándola con cierto tiempo de intervalo, este intervalo esta determinado por la línea “System.Threading.Thread.Sleep(1000)” la cantidad dentro de paréntesis determina el tiempo, mas o menos estimo que 1000 equivalen a un segundo.

Thread.Sleep(1000); // **Sleeps for 1000 milliseconds (or 1 second)** In this snippet, the Sleep method is called with a parameter of 1000, representing the sleep duration in milliseconds. So, the current thread will be paused for a second before it resumes its tasks. 30/11/2023

 Stackify  
<https://stackify.com/c-sleep-a-detailed-guide>  
**C# Sleep: A Detailed Guide - Stackify**

Ahora para heredar los métodos y agregar las diferentes funciones como agregar combustible o disminuir el combustible cuando se acelera solo llame al método base y sobrescribí que la variable de combustible disminuya una unidad.

Para el camión agregue la opción de agregar carga conforme se decida en el código, aunque este método no esta ligado a la velocidad que tiene el camión, no se me ocurrió como podría hacerlo la verdad

```
3 references
public void cargar(int cuanto)
{
    carga += cuanto;
}
0 references
public void descargar(int cuanto)
{
    carga -= cuanto;
}
3 references
public int NivelCarga()
{
    Console.WriteLine("Nivel de carga: {0}", carga);
    return carga;
}
```

El int cuanto es lo que nos permite determinar cuanta carga le agregaremos o le quitaremos a la carga actual y nivel de carga solo imprime el nivel de carga que se tiene.

Para la motocicleta agregue una función que en nuestro contexto es común, escuchar a los motociclistas pitar pidiendo permiso o cualquier cosa

```
public void hacerpippip()
{
    Console.WriteLine("Pip Pip (sonidos de bocina)");
}
```

Por lo que agregue este código de acá que imprime la oración “pip pip (sonidos de vocina)”

Ahora para el program.cs agregue un switch que nos permite probar cada clase de forma individual así no se ve todo de forma continua en la consola, esto lo aprendí en c++ y es igual en c#.

```
static void Main(string[] args)
{
    while (true)
    {
        Console.WriteLine("Seleccione una prueba para realizar:");
        Console.WriteLine("1. Prueba Carro de Combustion");
        Console.WriteLine("2. Prueba Carro Electrico");
        Console.WriteLine("3. Prueba Camion");
        Console.WriteLine("4. Prueba Motocicleta");
        Console.WriteLine("5. Salir");
        Console.Write("Ingrese su opción: ");
        string opcion = Console.ReadLine();

        switch (opcion)
        {
            case "1":
                PruebaCarroCombustion();
                break;
            case "2":
                PruebaCarroElectrico();
                break;
            case "3":
                PruebaCamion();
                break;
            case "4":
                PruebaMotocicleta();
                break;
            case "5":
                return;
            default:
                Console.WriteLine("Opción no válida. Intente de nuevo.");
                break;
        }
    }
}
```

De esta forma cada prueba que realice a las clases las agrupe en funciones y las mande a llamar con este switch.

Para cada prueba de la clase seguí la misma lógica, mostrar la información de partida y luego la información que pueda cambiar

```
static void PruebaCarroCombustion()
{
    Console.WriteLine("-----");
    Console.WriteLine("Prueba Carro de Combustion");
    CarroCombustion miCarroCombustion = new CarroCombustion(2021, "Azul", "Toyota");

    miCarroCombustion.InformacionVehiculo();
    Console.WriteLine("-----");
    miCarroCombustion.NivelCombustible();
    Console.WriteLine("-----");
    while (miCarroCombustion.Velocidadactual < 35)
    {
        miCarroCombustion.acelerar();
    }
    Console.WriteLine("-----");
    miCarroCombustion.frenar();
    Console.WriteLine("-----");
    miCarroCombustion.NivelCombustible();
    Console.WriteLine("-----");
    miCarroCombustion.cargarCombustible();
    miCarroCombustion.cargarCombustible();
    miCarroCombustion.cargarCombustible();
    Console.WriteLine("Se han cargado 3 litros de combustible");
    Console.WriteLine("-----");
    miCarroCombustion.NivelCombustible();
    Console.WriteLine("-----");
}
```

Imprime algunos guiones para separar los datos y que se distingan entre todos.