

## Consigna

Generar un proyecto de tipo Biblioteca de clases (Entidades) el cual tendrá las siguientes clases:


### Fabricante


El fabricante posee todos sus atributos privados, dos constructores, propiedades públicas, y sobrecarga de operadores. A continuación, se detallan sus características:

- Ambos constructores son de instancia, uno vacío y el otro inicializará los atributos.
- Las propiedades deben ser públicas, de lectura y escritura:
  - Marca: Obtiene o establece la marca del fabricante (\_marca).
  - Pais: Obtiene o establece el país del fabricante (\_pais).
- Sobrecarga de operadores
  - Igualdad (Fabricante, Fabricante). Retornara true, si las marcas y los países son iguales, false, caso contrario.
  - Implícito. Retornara la marca y el país del fabricante que recibe como parámetro, en el siguiente formato: "marca – país", ejemplo: "RENAULT – Francia"


**Fabricante**  
Clase


Campos

 \_marca : string


 \_pais : EPais


Propiedades


 Marca { get; set; } : string


 Pais { get; set; } : EPais


Métodos

 Fabricante()

 Fabricante(string marca, EPais pais)

 implicit operator string(Fabricante fabricante) : string

 operator !=(Fabricante x, Fabricante y) : bool

 operator ==(Fabricante x, Fabricante y) : bool

**EPais**  
Enum

Italia  
Francia  
Alemania

## Clase abstracta **Vehiculo**

Posee todos sus atributos protegidos, cuatro constructores, propiedades, métodos y sobrecarga de operadores. A continuación, se detallan sus características:

- Posee uno constructor vacío, uno de clase y otros dos de instancia, estos dos se encargarán de inicializar sus atributos. Reutilizar código.
- El atributo `_generadorDeVelocidades` será inicializado en el constructor de clase.
- La propiedad `VelocidadMaxima`, retornará el valor correspondiente del atributo `velocidadMaxima`, que se inicializará en dicha propiedad, si y solo si su valor es cero.

Para inicializar dicho atributo, se utilizará el atributo estático `_generadorDeVelocidades` (valores aleatorios entre 100 y 280).

- Poseerá las propiedades de lectura y escritura:
  - Fabricante: Obtiene o establece el fabricante del vehículo (`_fabricante`).
  - Modelo: Obtiene o establece el modelo del vehículo (`_modelo`).
  - Precio: Obtiene o establece el precio del vehículo (`_precio`).
- El método privado y de clase `Mostrar()`, retornara una cadena detallando los atributos del vehículo recibido por parámetro, ejemplo:

FABRICANTE: RENAULT – Francia

MODELO: R9

VELOCIDAD MAXIMA: 249

PRECIO: \$6500

- Sobrecarga de operadores:
  - Igualdad (`Vehiculo, Vehiculo`). Retornará `true`, si los modelos y los fabricantes son iguales, `false`, caso contrario. Reutilizar código.
  - Explícito. Retornará el detalle completo del vehículo que recibe como parámetro. Reutilizar código.

**Vehiculo**

Abstract Clase

Campos

\_fabricante : Fabricante

\_generadorDeVelocidades : Random

\_modelo : string

\_precio : float

\_velocidadMaxima : int

Propiedades

Fabricante { get; set; } : Fabricante

Modelo { get; set; } : string

Precio { get; set; } : float

VelocidadMaxima { get; set; } : int

Métodos

explicit operator string(Vehiculo vehiculo) : string

Mostrar(Vehiculo vehiculo) : string

operator !=(Vehiculo x, Vehiculo y) : bool

operator ==(Vehiculo x, Vehiculo y) : bool

Vehiculo()

Vehiculo()

Vehiculo(string marca, EPais pais, string modelo, float precio)

Vehiculo(string modelo, float precio, Fabricante fabricante)

También tendrá las siguientes clases derivadas de **Vehiculo**

## Auto

Posee un único atributo privado, un constructor, una única propiedad, sobrecarga de operadores y sobreescritura de métodos. A continuación, se detallan sus características:

- Ambos constructores son de instancia, uno vacío y un constructor que será el encargado de inicializar su único atributo.
- Posee una propiedad Tipo de lectura y escritura, que retornara y asignar el valor correspondiente de \_tipo.
- Sobrecarga de operadores:

- Igualdad (Auto, Auto). Retornara true, si los vehículos y los tipos (\_tipo) son iguales, false, caso contrario. Reutilizar código.
- Explícito. Retornara el precio del auto que recibe como parámetro.
- Equals. Retornará true, si el parámetro recibido es igual a la instancia actual. Reutilizar código.
- ToString. Retornará una cadena conteniendo la información completa del objeto. Reutilizar código.

Ejemplo:

FABRICANTE: RENAULT – Francia

MODELO: R9

VELOCIDAD MAXIMA: 249

PRECIO: \$6500

TIPO: Sedan

**Auto**  
 Clase  
 ↳ Vehiculo

▲ Campos
 

\_tipo : ETipo

▲ Propiedades
 

Tipo { get; set; } : ETipo

▲ Métodos
 

Auto()  
 Auto(string modelo, float precio, Fabricante fabricante, ETipo tipo)  
 Equals(object obj) : bool  
 explicit operator float(Auto auto) : float  
 operator !=(Auto x, Auto y) : bool  
 operator ==(Auto x, Auto y) : bool  
 ToString() : string

**ETipo**  
 Enum

Deportivo  
 Sedan  
 Coupe  
 Familiar

## Moto

Posee un único atributo privado, un constructor, una única propiedad, sobrecarga de operadores y sobreescritura de métodos. A continuación, se detallan sus características:

- Ambos constructores son de instancia, uno vacío y un constructor que será el encargado de inicializar su único atributo.
- Posee una propiedad Cilindrada de lectura y escritura, que retornara y asignar el valor correspondiente de \_cilindrada.
- Sobrecarga de operadores:

- Igualdad (Moto, Moto). Retornará true, si los vehículos y las cilindradas son iguales, false, caso contrario. Reutilizar código.
- Explícito. Retornará el precio de la moto que recibe como parámetro.
- Equals. Retornará true, si el parámetro recibido es igual a la instancia actual. Reutilizar código.
- ToString. Retornará una cadena conteniendo la información completa del objeto. Reutilizar código.

Ejemplo:

FABRICANTE: RENAULT – Francia

MODELO: R9

VELOCIDAD MAXIMA: 249

PRECIO: \$6500

CILINDRADA: cc500

**Moto**  
 Clase  
 Vehiculo

Campos  
 \_cilindrada : ECilindrada

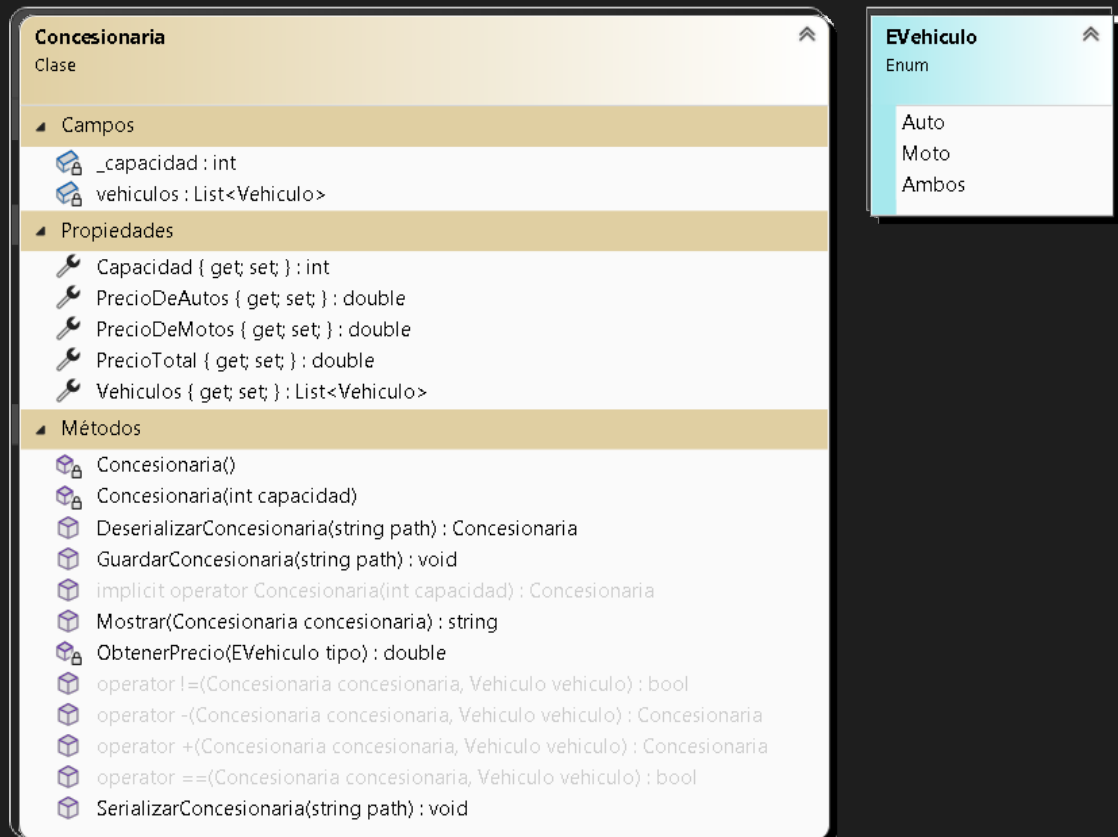
Propiedades  
 Cilindrada { get; set; } : ECilindrada

Métodos  
 Equals(object obj) : bool  
 Moto()  
 Moto(string marca, EPais pais, string modelo, float precio, ECilindrada cilindrada)  
 operator !=(Moto x, Moto y) : bool  
 operator ==(Moto x, Moto y) : bool  
 ToString() : string

**ECilindrada**  
 Enum

cc50  
 cc125  
 cc250  
 cc500

La última clase que tendrá el proyecto será **Concesionaria**



La concesionaria posee todos sus atributos privados, dos constructores, propiedades públicas, métodos y sobrecarga de operadores. A continuación, se detallan sus características:

- Ambos atributos son privados. Uno indicará la capacidad máxima que tendrá la concesionaria para almacenar vehículos. El otro es una colección de tipo Vehiculo.
- El constructor y su sobrecarga son privados. El constructor por defecto será el único que inicializará la lista genérica. La sobrecarga, inicializará la capacidad de la concesionaria. Reutilizar código.
- Poseerá las propiedades de lectura y escritura:
  - Capacidad: Obtiene o establece la capacidad de la concesionaria (`_capacidad`).
  - Vehiculos: Obtiene o establece (vacío) la lista de vehículos que posee la concesionaria (`_vehiculos`).
- Método privado y de instancia `ObtenerPrecio()`, retornará el valor de la

concesionaria de acuerdo con el enumerado EVehiculo que recibe como parámetro. Las propiedades publicas PrecioDeAutos, PrecioDeMotos y PrecioTotal están asociadas al método ObtenerPrecio(). Reutilizar código.

- El método público de clase Mostrar(), retornará una cadena con toda la información de la concesionaria, incluyendo el detalle de cada uno de sus vehículos de la concesionaria que recibe como parámetro. Reutilizar código.

Ejemplo:

Capacidad: 6

Total por autos: \$447200

Total por motos: \$1037500

Total: \$1484700

\*\*\*\*\*

Listado de Vehiculos

\*\*\*\*\*

FABRICANTE: RENAULT – Francia

MODELO: R9

VELOCIDAD MAXIMA: 196

PRECIO: \$65000

TIPO: Sedan

- Sobrecarga de operadores:
  - Implícito. Retornará una instancia de Concesionaria cuya capacidad coincida con el parámetro recibido.
  - Igualdad (Concesionaria, Vehiculo). Retornará true, si es que el auto o moto ya se encuentra en la concesionaria, false, caso contrario. Reutilizar código.
  - Adición (Concesionaria, Vehiculo). Si la concesionaria posee capacidad de almacenar al menos un vehículo mas y ese auto o moto, no se encuentra en la concesionaria, lo agregara a la colección, caso contrario informara lo acontecido con el mensaje "¡El vehículo ya está en la concesionaria!" o "¡No hay más lugar en la concesionaria!". Reutilizar código.
  - Sustracción (Concesionaria, Vehiculo), retornará una concesionaria sin el vehículo, siempre y cuando el vehículo se encuentre en el

listado, caso contrario informará lo acontecido con el mensaje "¡El vehículo no está en la concesionaria!". Reutilizar código.

- Método GuardarConcesionaria() de instancia. El método deberá guardar en un archivo de texto toda la información de la concesionaria y sus vehículos.
- Método SerializarConcesionaria() de instancia. El método deberá guardar en un archivo XML toda la información de la concesionaria y sus vehículos.
- Método DeserializarConcesionaria() de clase. El método deberá leer el archivo XML con toda la información de la concesionaria y sus vehículos.

Agregar a la solución un proyecto de tipo Aplicación de Consola (Test) y agregar el Main indicado sin modificar línea alguna:

```
Concesionaria miConcesionaria = 6;
```

```
Fabricante fabricante1 = new Fabricante("RENAULT", EPais.Francia);
```

```
Fabricante fabricante2 = new Fabricante("CITROEN", EPais.Francia);
```

```
Auto auto1 = new Auto("R9", 65000, fabricante1, ETipo.Sedan);
```

```
Auto auto2 = new Auto("C4", 285900, fabricante2, ETipo.Deportivo);
```

```
Auto auto3 = new Auto("C4", 390500, fabricante2, ETipo.Familiar);
```

```
Auto auto4 = new Auto("C4", 96300, fabricante1, ETipo.Deportivo);
```

```
Moto moto1 = new Moto("DUCATI", EPais.Italia, "MONSTER", 450000,  
ECilindrada.cc500);
```

```
Moto moto2 = new Moto("BMW", EPais.Alemania, "G 310 GS", 437500,  
ECilindrada.cc125);
```

```
Moto moto3 = new Moto("DUCATI", EPais.Italia, "MONSTER", 150000,  
ECilindrada.cc125);
```



```
miConcesionaria += auto1;
```

```
//YA INGRESADO
```

```
miConcesionaria += auto1;
```

```
miConcesionaria += moto1;
```

```
miConcesionaria += moto2;
```

```
miConcesionaria += moto3;
```

```
miConcesionaria += auto2;
```

```
miConcesionaria += auto3;
```

```
//SIN LUGAR
```

```
miConcesionaria += auto4;
```

```
//AGREGO UN LUGAR SACANDO UN VEHICULO
```

```
miConcesionaria -= auto3;
```

```
//NO ESTA MÁS EN LA CONCESIONARIA
```

```
miConcesionaria -= auto3;
```

```
miConcesionaria += auto4;
```

```
Console.WriteLine();
```

```
Console.WriteLine(Concesionaria.Mostrar(miConcesionaria));
```

```
string rutaEscritorio =
```

```
Environment.GetFolderPath(Environment.SpecialFolder.Desktop);
```

```
string rutaArchivoTxt = Path.Combine(rutaEscritorio, "Concesionaria.txt");  
miConcesionaria.GuardarConcesionaria(rutaArchivoTxt);
```

```
string rutaArchivoXml = Path.Combine(rutaEscritorio, "Concesionaria.xml");  
miConcesionaria.SerializarConcesionaria(rutaArchivoXml);
```

```
Concesionaria deserializedConcesionaria =  
Concesionaria.DeserializarConcesionaria(rutaArchivoXml);
```

```
Console.WriteLine("\n*****");  
Console.WriteLine("Objeto deserealizado:");  
Console.WriteLine("*****\n");
```

```
Console.WriteLine(Concesionaria.Mostrar(deserializedConcesionaria));
```

```
Console.ReadLine();
```

## Salida por consola

```
iEl vehículo ya está en la concesionaria!  
iNo hay más lugar en la concesionaria!  
iEl vehículo no está en la concesionaria!  
  
Capacidad: 6  
Total por autos: $447200  
Total por motos: $1037500  
Total: $1484700  
  
*****  
Listado de Vehículos  
*****  
FABRICANTE: RENAULT - Francia  
MODELO: R9  
VELOCIDAD MÁXIMA: 204  
PRECIO: $65000  
TIPO: Sedan  
  
FABRICANTE: DUCATI - Italia  
MODELO: MONSTER  
VELOCIDAD MÁXIMA: 134  
PRECIO: $450000  
CILINDRADA: cc500  
  
FABRICANTE: BMW - Alemania  
MODELO: G 310 GS  
VELOCIDAD MÁXIMA: 250  
PRECIO: $437500  
CILINDRADA: cc125  
  
FABRICANTE: DUCATI - Italia  
MODELO: MONSTER  
VELOCIDAD MÁXIMA: 226  
PRECIO: $150000  
CILINDRADA: cc125  
  
FABRICANTE: CITROEN - Francia  
MODELO: C4  
VELOCIDAD MÁXIMA: 268  
PRECIO: $285900  
TIPO: Deportivo  
  
FABRICANTE: RENAULT - Francia  
MODELO: C4  
VELOCIDAD MÁXIMA: 252  
PRECIO: $96300  
TIPO: Deportivo
```

```
*****  
Objeto deserealizado:  
*****  
  
Capacidad: 6  
Total por autos: $447200  
Total por motos: $1037500  
Total: $1484700  
  
*****  
Listado de Vehículos  
*****  
FABRICANTE: RENAULT - Francia  
MODELO: R9  
VELOCIDAD MÁXIMA: 102  
PRECIO: $65000  
TIPO: Sedan  
  
FABRICANTE: DUCATI - Italia  
MODELO: MONSTER  
VELOCIDAD MÁXIMA: 233  
PRECIO: $450000  
CILINDRADA: cc500  
  
FABRICANTE: BMW - Alemania  
MODELO: G 310 GS  
VELOCIDAD MÁXIMA: 268  
PRECIO: $437500  
CILINDRADA: cc125  
  
FABRICANTE: DUCATI - Italia  
MODELO: MONSTER  
VELOCIDAD MÁXIMA: 226  
PRECIO: $150000  
CILINDRADA: cc125  
  
FABRICANTE: CITROEN - Francia  
MODELO: C4  
VELOCIDAD MÁXIMA: 190  
PRECIO: $285900  
TIPO: Deportivo  
  
FABRICANTE: RENAULT - Francia  
MODELO: C4  
VELOCIDAD MÁXIMA: 272  
PRECIO: $96300  
TIPO: Deportivo
```

### NOTA:

Los únicos valores que podrán cambiar son los que indican la velocidad máxima de cada vehículo, ya que son valores aleatorios de entre 100 y 280.