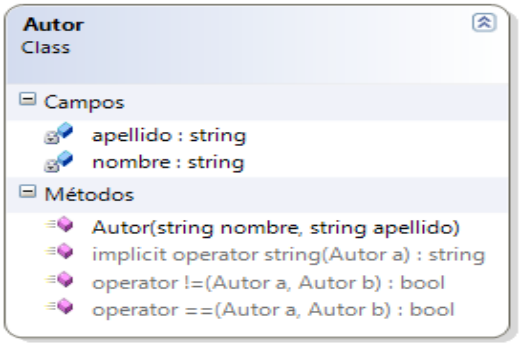
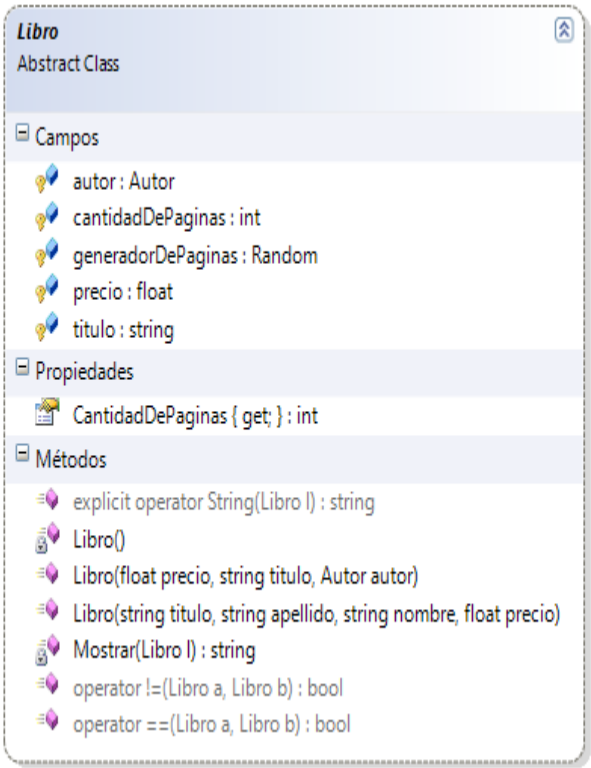
PRIMER PARCIAL – LABORATORIO II – 2020

Generar una Solución nombrada como: ***Apellido.Nombre.Division***, que contenga un proyecto de tipo *Biblioteca de Clases* (Entidades) el cual tendrá las siguientes clases:

Autor posee todos sus atributos privados, un único constructor y sobrecargas de operadores:
Igualdad (Autor, Autor). Retornará *true*, si los nombres y los apellidos son iguales, *false*, caso contrario.
Implícito. Retornará el nombre y apellido del autor que recibe como parámetro.

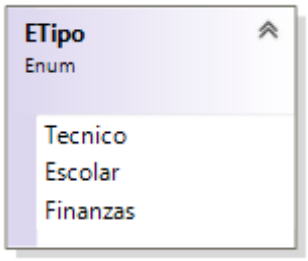


Clase abstracta **Libro**:
Todos sus atributos son protegidos.
Posee un constructor de **clase** y un constructor de instancia sobrecargado.
La propiedad (de sólo lectura) CantidadDePaginas, retornará el valor correspondiente del atributo *cantidadDePaginas*, que se inicializará en dicha propiedad, si y sólo si su valor es cero. Para inicializar dicho atributo, se utilizará el atributo estático *generadorDePaginas* (valores aleatorios entre 10 y 570). Ninguno debe de repetirse.
El método privado y de **clase** Mostrar, retornará una cadena detallando los atributos de la clase.
Sobrecarga de operadores:
Igualdad (Libro, Libro). Retornará *true*, si los títulos y los autores son iguales, *false*, caso contrario.
Explícito. Retornará el detalle completo del libro que recibe como parámetro.

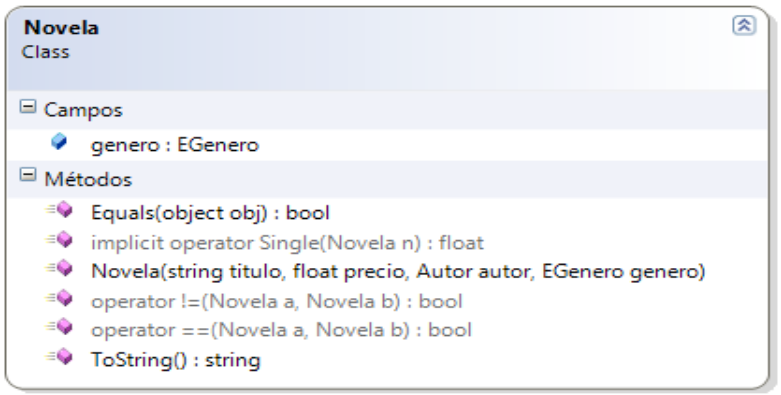


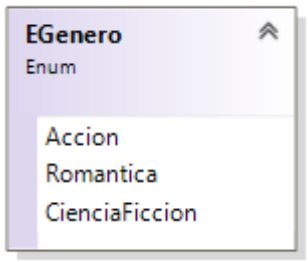
También tendrá las siguientes clases derivadas de Libro:

Manual posee un único atributo propio, que será inicializado por su **único** constructor.
Sobrecarga de operadores:
Igualdad (Manual, Manual). Retornará *true*, si los libros y los tipos son iguales, *false*, caso contrario. Reutilizar código.
Explícito. Retornará el precio del manual que recibe como parámetro.
Polimorfismo en Equals. Retornará true, si el parámetro recibido es igual a la instancia actual. Reutilizar código.
Polimorfismo en ToString, retornará una cadena conteniendo la información completa del objeto. Reutilizar código.

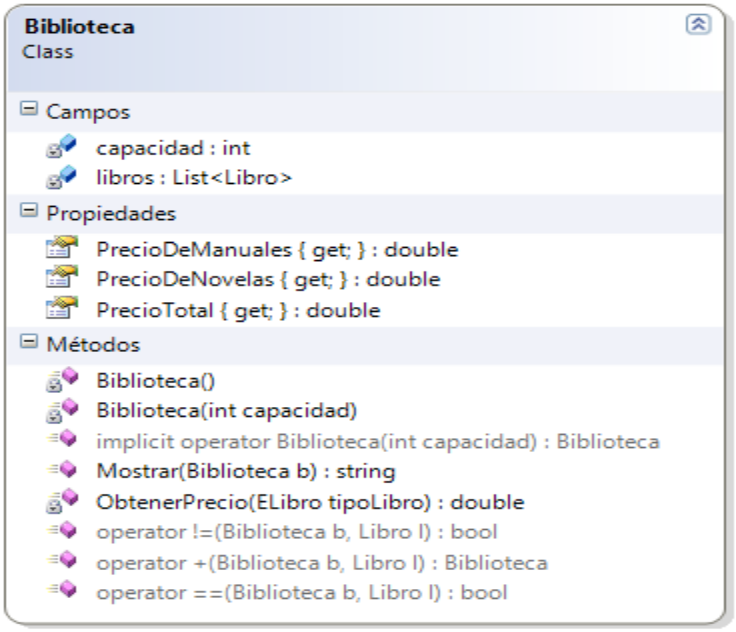


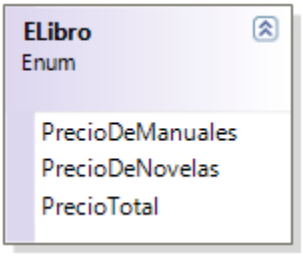
Novela posee un único atributo propio, que será inicializado por su **único** constructor.
Sobrecarga de operadores:
Igualdad (Novela, Novela). Retornará *true*, si los libros y los géneros son iguales, *false*, caso contrario. Reutilizar código.
Implícito. Retornará el precio de la novela que recibe como parámetro.
Polimorfismo en Equals. Retornará true, si el parámetro recibido es igual a la instancia actual. Reutilizar código.
Polimorfismo en ToString, retornará una cadena conteniendo la información completa del objeto. Reutilizar código.





La última clase que tendrá el proyecto será **Biblioteca**.
Dicha clase posee dos atributos, ambos privados. Uno indicará la capacidad máxima que tendrá la biblioteca para almacenar libros. El otro es una colección genérica de tipo Libro.
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 biblioteca. Reutilizar código.
El método público de **clase** Mostrar, retornará una cadena con toda la información de la biblioteca, incluyendo el detalle (completo) de cada uno de sus libros. Reutilizar código.
Sobrecarga de operadores:
Implícito, retornará una instancia de Biblioteca cuya capacidad coincida con el parámetro recibido.
Igualdad, retornará *true*, si es que el manual o la novela ya se encuentra en la biblioteca, *false*, caso contrario. Reutilizar código.
Adición, si la biblioteca posee capacidad de almacenar al menos un libro más y ese manual o novel no se encuentra en la biblioteca, lo agregará a la colección, caso contrario, informará lo acontecido. Reutilizar código.
Método privado y de instancia ObtenerPrecio, retornará el valor de la biblioteca de acuerdo con el enumerado **ELibro** que recibe como parámetro. Las propiedades públicas PrecioDeManuales, PrecioDeNovelas y PrecioTotal están asociadas al método ObtenerPrecio. Reutilizar código.





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

NOTA: Los únicos valores que podrán cambiar son los que indican la cantidad de páginas de cada libro, ya que deben ser valores aleatorios de entre 10 y 570.

Main:

```
Biblioteca miBiblioteca = 5;
Autor a = new Autor("Esteban", "Rey");
Autor b = new Autor("Joe", "Mayo");
Manual m1 = new Manual("Economia", "Domingo", "Caballo", 25f, ETipo.Finanzas);
Novela n1 = new Novela("Miseria", 63.50f, a, EGenero.Romantica);
Manual m2 = new Manual("C#", "Joe", "Mayo", 299.50f, Etipo.Tecnico);
Novela n2 = new Novela("Miseria", 205f, a, EGenero.Accion);
Novela n3 = new Novela("Miseria", 98f, a, EGenero.CienciaFiccion);
Novela n4 = new Novela("Miseria", 103.50f, b, EGenero.Accion);

miBiblioteca += m1;
//YA INGRESADO
miBiblioteca += m1;

miBiblioteca += n1;
miBiblioteca += m2;
miBiblioteca += n2;
miBiblioteca += n3;
//SIN LUGAR
miBiblioteca += n4;

Console.WriteLine();

//TRUE
Console.WriteLine(m1.Equals(m1));
//FALSE
Console.WriteLine(m1.Equals("Joe Mayo"));
//FALSE
Console.WriteLine(m1.Equals(m2));
//TRUE
Console.WriteLine(n1.Equals(n1));
//FALSE
Console.WriteLine(n1.Equals(n2));
//FALSE
Console.WriteLine(n1.Equals(n4));

Console.WriteLine(Biblioteca.Mostrar(miBiblioteca));
Console.ReadLine();
```

```
El libro ya está en la biblioteca!!!
No hay más lugar en la biblioteca!!!

True
False
False
True
False
False

Capacidad: 5
Total por manuales: 324,5
Total por novelas: 366,5
Total: 691
*****************************************
Listado de libros
*****************************************

AUTOR: Caballo - Domingo
TITULO: Economia
CANTIDAD DE PÁGINAS: 171
PRECIO: 25
TIPO: Finanzas

AUTOR: Esteban - Rey
TITULO: Miseria
CANTIDAD DE PÁGINAS: 307
PRECIO: 63,5
GENERO: Romantica

AUTOR: Mayo - Joe
TITULO: C#
CANTIDAD DE PÁGINAS: 127
PRECIO: 299,5
TIPO: Tecnico

AUTOR: Esteban - Rey
TITULO: Miseria
CANTIDAD DE PÁGINAS: 380
PRECIO: 205
GENERO: Accion

AUTOR: Esteban - Rey
TITULO: Miseria
CANTIDAD DE PÁGINAS: 47
PRECIO: 98
GENERO: CienciaFiccion
```