Universidad Tecnológica Nacional Facultad Regional Avellaneda										
Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos										
Materia: Laboratorio de Programación II										
Apellido:				Fecha:	18/07/2019					
Nombre:				Docente(2):	F. Dávila / D. Boullon					
División:				Nota(2):						
Legajo:				Firma(2):						
Instancia(1):	**PP**		**RPP**		**SP**		**RSP**		**FIN**	

(1) Las instancias válidas son: 1er Parcial (**PP**), Recuperatorio 1er Parcial (**RPP**), 2do Parcial (**SP**), Recuperatorio 2do Parcial (**RSP**), Final (**FIN**). Marque con una cruz.
(2) Campos a ser completados por el docente.

IMPORTANTE:

- Guardar el proyecto en el **disco D:**. Ante un corte de energía o problema con el archivo de corrección, el alumno será responsable de que el proyecto sea recuperable.
- **2 (dos) errores en el mismo tema anulan su puntaje.**
- **Errores de conceptos de POO anulan el punto.**
- **Cada tema vale 1 (un) punto (Herencia, Generics, Test Unitarios, etc.). La correcta documentación también será evaluada.**
- **Se deberán tener al menos el 60% bien de los temas a evaluar según la instancia para lograr la aprobación.**
- Colocar sus datos personales en el nombre del proyecto principal, colocando: Apellido.Nombre.AñoCursada. Ej: Pérez.Juan.2018. No sé corregirán proyectos que no sea identificable su autor.
- **Salvo que se indique lo contrario, TODAS** las clases deberán ir en una Biblioteca de Clases llamada Entidades.
- No se corregirán exámenes que no compilen.
- **Reutilizar** tanto código como crean necesario.

Al finalizar, colocar la carpeta de la Solución completa en un archivo ZIP que deberá tener como nombre Apellido.Nombre.AñoCursada.zip y dejar este último en el Escritorio de la máquina. Luego presionar el botón de la barra superior, **colocar un mensaje** y presionar *Aceptar*. **Aguardar a que el profesor indique que el examen fue copiado de forma correcta.** Luego retirarse del aula.

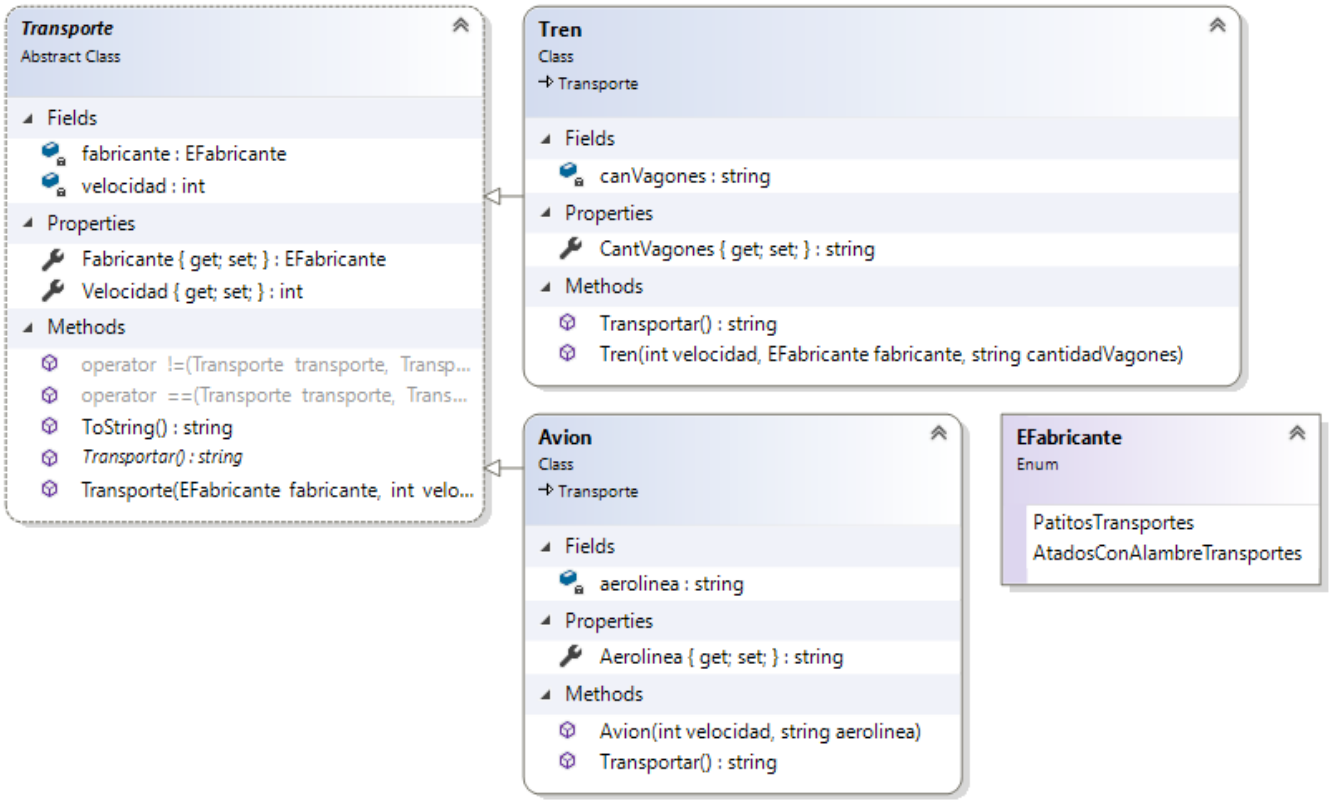
---

*TIEMPO MÁXIMO PARA RESOLVER EL EXAMEN **90** MINUTOS.*

---

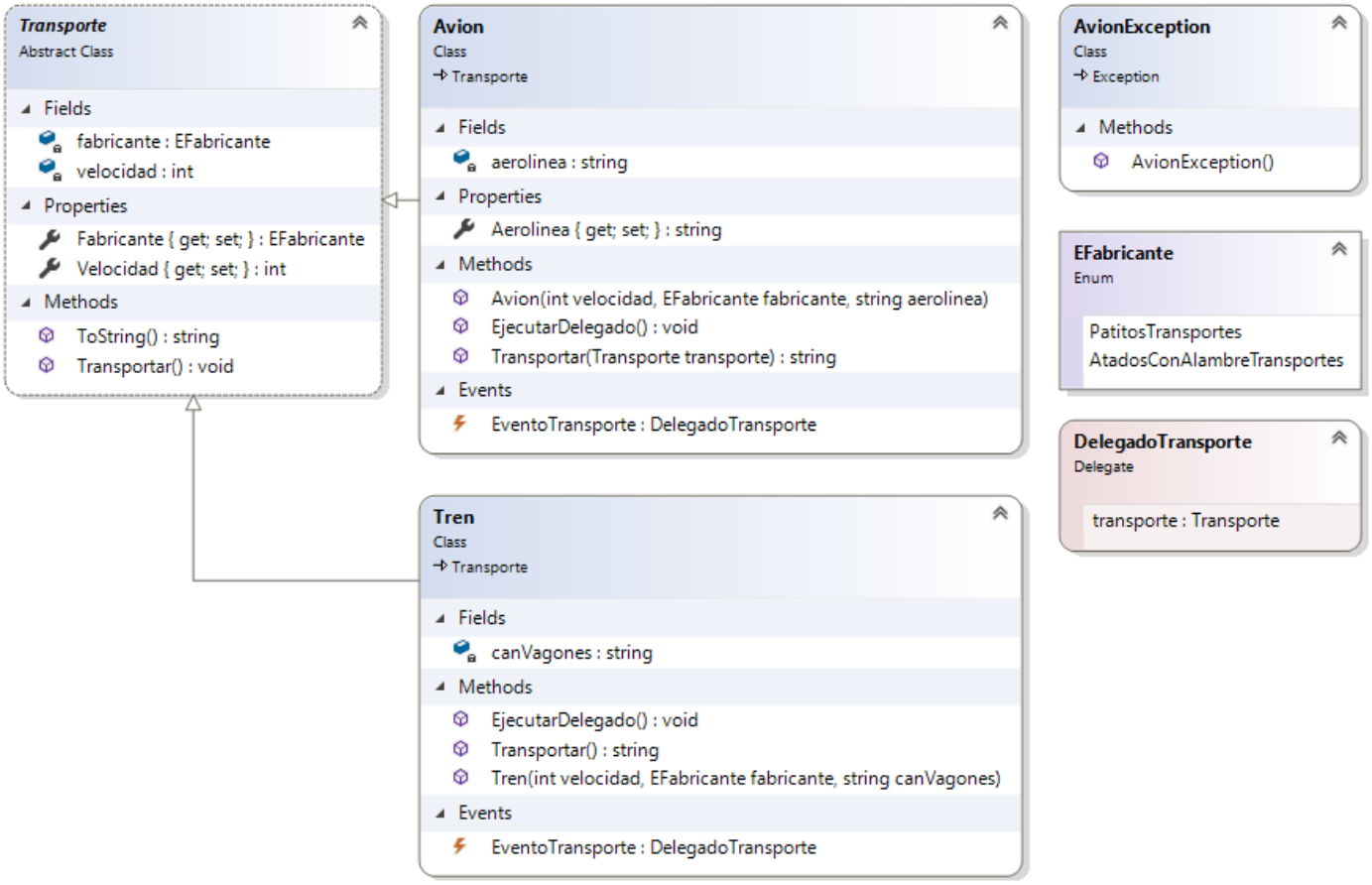
1. Renombrar la solución a "20190718–[Apellido].[Nombre]" siendo el Apellido y Nombre del alumno.
2. Los proyectos están creados, solo falta crear el Test Unitario para el final y el recuperatorio del Segundo Parcial.
3. Completar y generar los elementos necesarios para cumplir con el siguiente diagrama de clases:

RPP:



RSP y Final:

4. **<u>Transporte</u>**: **clase abstracta** con 2 atributos privado**.**



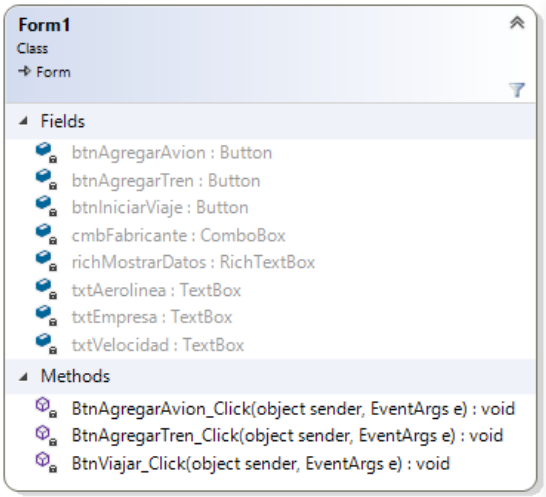
   a. **RPP, RSP y Final => Transportar**: método abstracto.
   b. **RPP y Final => ToString:** Sobrecarga que retorna: " \n.{velocidad}kph \n. {fabricante}" (usar String.Format). Ej:
      " .150kph
      .PatitosTransportes"
   c. **RPP y Final =>** Constructor protegido que recibe velocidad y fabricante.
   d. **RPP y Final =>**Aplicar encapsulamiento a todos los atributos.

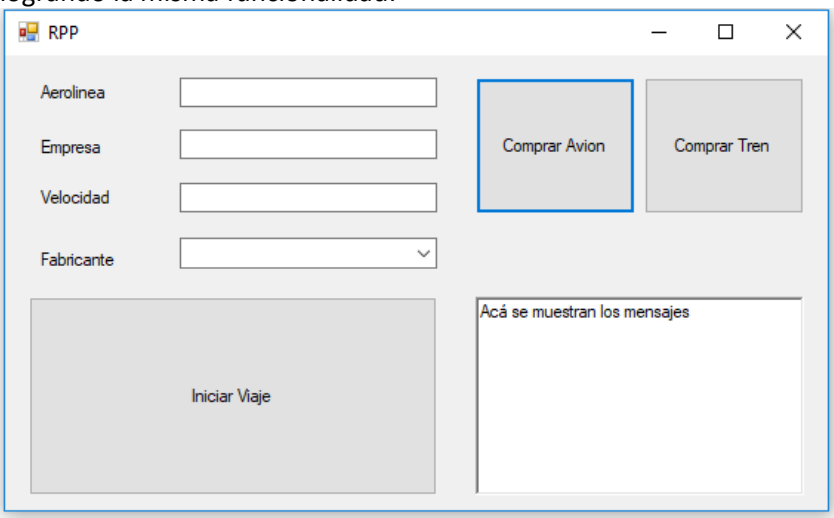
e. **RPP, RSP, Final => Definir enum EFabricante con "PatitoTransportes", "AtadosConAlambreTransportes".**
f. **RSP y Final =>** DelegadoTransporte: Delegado que retorna void y recibe Transporte.

5. **RSP y Final => DelegadoTransporte**: retorna string y recibe transporte.
6. **Avion**:
   a. **RPP => Transportar** simula el transporte de pasajeros y retorna un mensaje "Vuela avioncito".
   b. **RSP y Final => EjecutarEventos:** ejecuta el evento y se pasa a si mismo como parametro (...Invoke(this)).
   c. **RPP y Final =>** Dos aviones serán iguales si tienen la misma aerolínea y fabricante.
   d. **RPP y Final =>** hereda de **Tranporte** y agrega un atributo privado.
   e. **RPP y Final =>** Tiene un solo constructor que recibe velocidad, aerolinea y fabricante.
   f. **RPP y Final =>** Conversión implícita a string retornando toda su información utilizando **StringBuilder**.
   g. **RPP y Final =>** Aplicar encapsulamiento a todos los atributos.
   h. **RSP y Final => Transportar** valida si el evento tiene manejadores (cuando el evento no tiene manejadores es igual a null) y caso contrario lanza un AvionException con el mensaje "No pudo volar".
   i. **RSP y Final =>** Evento del tipo DelegadoTransporte llamado Evento.

7. **Tren**:
   a. **RPP y Final =>** hereda de **Transporte**.
   b. **RSP y Final => EjecutarEventos:** ejecuta el evento y se pasa a si mismo como parametro (...Invoke(this)).
   c. **RPP y Final =>** Tiene un solo constructor que recibe fabricante, velocidad y cantidad de vagones.
   d. **RPP y Final =>** Tiene una conversión explícita a string que retorna toda su información (**usar StringBuilder**).
   e. **RPP y Final =>** Aplicar encapsulamiento a todos los atributos.
   f. **RPP, RSP, Final => Transportar** simula el transporte de pasajeros y retorna un mensaje "Corre trencito ".
   g. **RSP y Final =>** Evento del tipo DelegadoTransporte llamado Evento.

8. **RPP =>** Generar el siguiente formulario, logrando la misma funcionalidad:





a. Los objetos se instancian en el constructor.
b. El nombre de la terminal es "Terminales La Marmota".
c. El título del formulario serán los datos del alumno.
d. **Comprar avión** agrega aviones a la lista, siempre que no se encuentre en la misma y muestra un MessageBox indicando si se pudo agregar.
e. **Comprar tren** agrega tren a la lista y muestra un MessageBox indicando si se pudo agregar.
f. **Iniciar Viaje** itera la lista y guarda el retorno del método transportar en un stringbuilder. Al finalizar muestra el resultado en el rich box de salida.

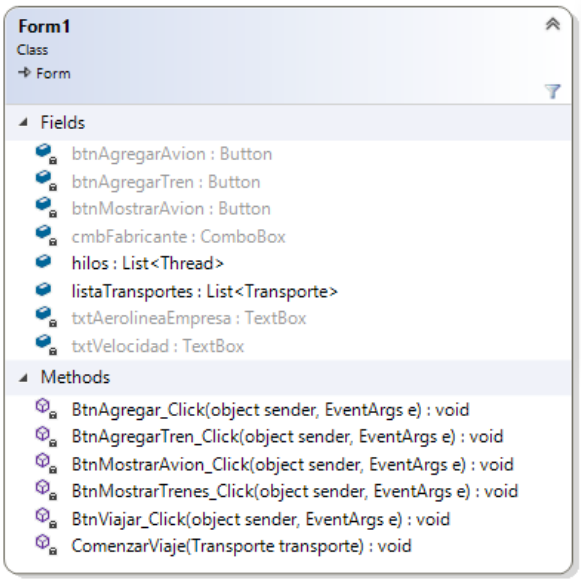
**importante - Código para cargar comboBox con productos:*

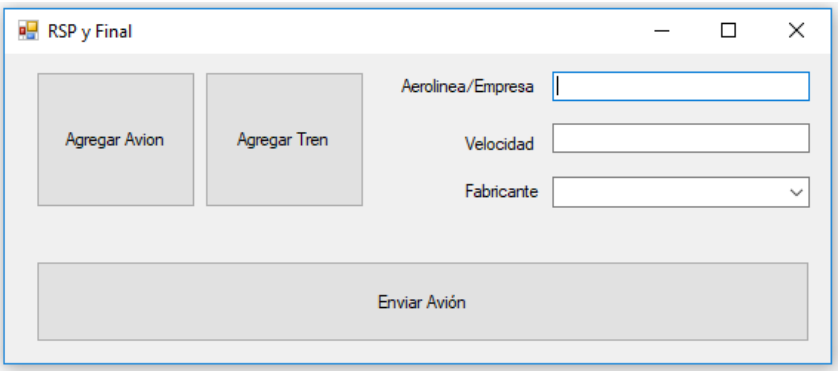
```
this.cmbFabricante.DataSource = Enum.GetNames(typeof(EFabricante ));
```

**importante - Código para leer del databox y parsear a EFabricante:*

```
EFabricante fabricante;
Enum.TryParse(this.cmbFabricante.SelectedItem.ToString(), out fabricante);
```

9. **RSP y Final =>** Tendran disponible este formulario y deberán lograr la siguiente funcionalidad:





   a. Tiene un atributo privado List<Thread> hilos, que colecciona todos los hilos que se generen.
   b. Manejar todas las excepciones.
   c. **ComenzarViaje:** método que recibe un objeto del tipo Transporte y muestra en un Messagebox con el mensaje retornado por el método **Transportar.**
   d. El botón **Iniciar Viaje Tren** lee los datos del form y agregar al **Evento** el manejador "TrenDAO.Guardar" y "ComenzarViaje" y ejecutar el método **EjecutarEvento** en un hilo nuevo.
   e. El botón **Guardar Tren** usa TrenDAO para guardar un tren en la base de datos.
   f. El botón **Guardar Avion** usa AvionXml para guardar un avión en un archivo de xml.
   g. Cuando se cierra la aplicación abortar todos los hilos.
   h. Catchear todas las posibles excepciones y mostrar los errores en un MessageBox.

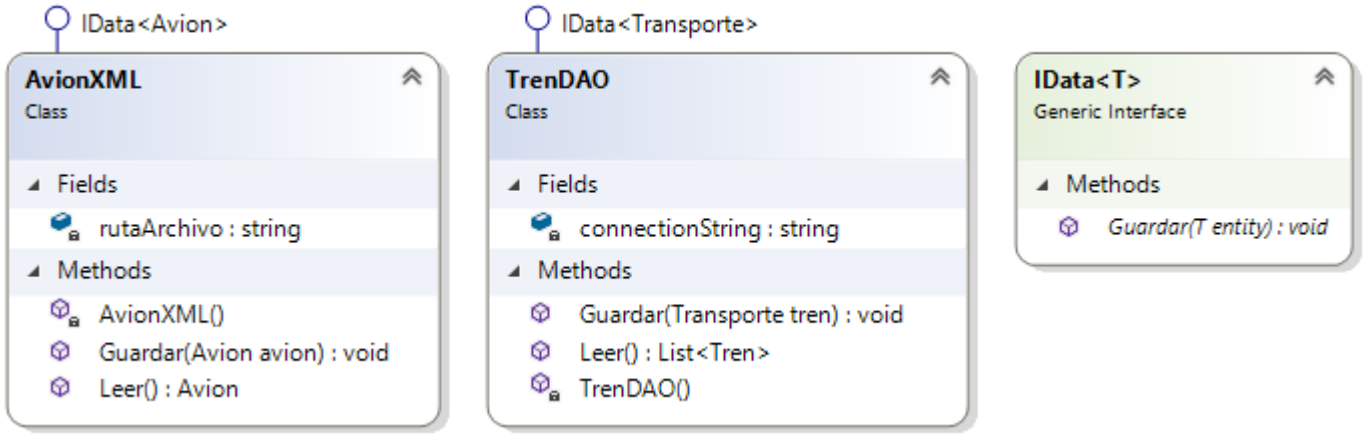
   **importante - Código para cargar comboBox con productos:*

```
this.cmbFabricante.DataSource = Enum.GetNames(typeof(EFabricante ));
```

   **importante - Código para leer del databox y parsear a EFabricante:*

```
EFabricante fabricante;
Enum.TryParse(this.cmbFabricante.SelectedItem.ToString(), out fabricante);
```

RSP y Final => Proyecto Datos:



10. **RSP y Final =>** Clase **IData<T>**:
    a. Guardar, metodo sin implementar que recibe T.
11. **RSP y Final =>** Clase **AvionXML**:
    a. rutaArchivo: estático, contiene la ruta al archivo Aviones.xml en el escritorio, se le da valor en el constructor de clase.
    b. Posee un constructor el cual es estático (de clase).
    c. Guardar recibe serializa un objeto Avion en XML en el escritorio.

d. Leer deserializar el objeto.

12. **RSP y Final =>** Clase **TrenDAO**:
    a. connectionString: atributo estático que contiene el connection string, se le da el valor en el **constructor de clase**.
    b. Posee un constructor el cual es estático (de clase).
    c. Guardar guarda un objeto del tipo tren en la base de datos.
    d. Leer Lee de la base de datos y devuelve una lista de trenes.

Ejecutar un script en una nueva base de datos llamada Viajes:

```
USE [Viajes]
GO
/****** Object:  Table [dbo].[trenes]    Script Date: 7/8/2019 1:04:10 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[trenes](
	[cantidad_vagones] [int] NULL,
	[fabricante] [varchar](50) NULL,
	[velocidad] [int] NULL
) ON [PRIMARY]
GO
```

13. **RSP y Final =>** Agregar un proyecto de Test Unitario y testear los métodos Guardar de AvionXml y .