

Universidad Tecnológica Nacional Facultad Regional Avellaneda



Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos

Materia: Laboratorio de Programación II

| | | | |
|----------------------------|--|--------------------------|------------|
| Apellido: | | Fecha: | 11/07/2019 |
| Nombre: | | Docente ⁽²⁾ : | |
| División: | | Nota ⁽²⁾ : | |
| Legajo: | | Firma ⁽²⁾ : | |
| Instancia ⁽¹⁾ : | <div style="display: flex; justify-content: space-around;"> PP RPP SP RSP FIN </div> | | X |


(1) Las instancias validas son: 1^{er} Parcial (**PP**), Recuperatorio 1^{er} Parcial (**RPP**), 2^{do} Parcial (**SP**), Recuperatorio 2^{do} Parcial (**RSP**), Final (**FIN**). Marque con una cruz.

(2) Campos a ser completados por el docente.

IMPORTANTE:

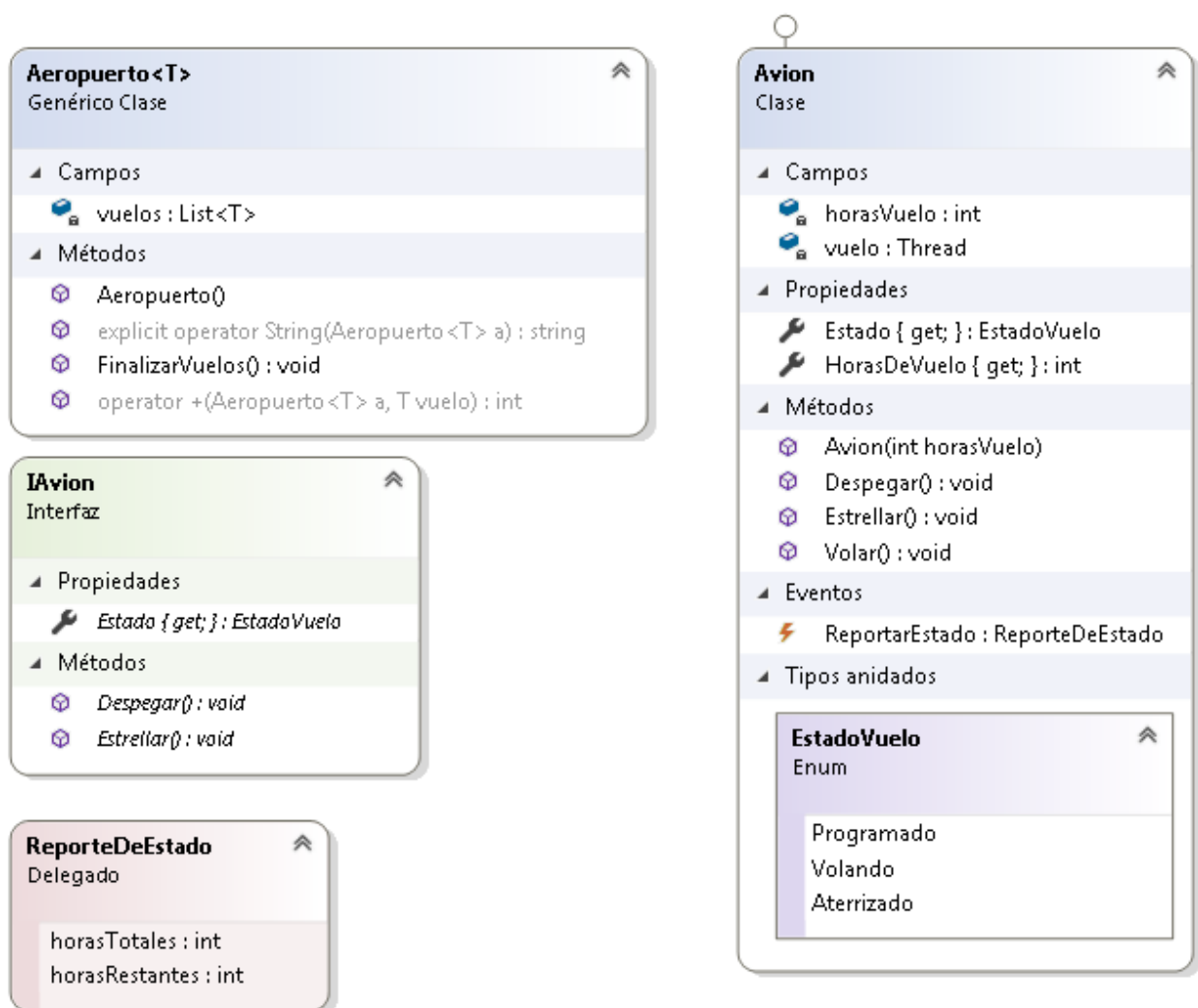
- Guardar el proyecto en el **disco D:**. Ante un corte de energía o problema con el archivo de corrección, el alumno será responsable de que el proyecto sea recuperable.
- **2 (dos) errores en el mismo tema anulan su puntaje.**
- **Errores de conceptos de POO anulan el punto.**
- **Cada tema vale 1 (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 se 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

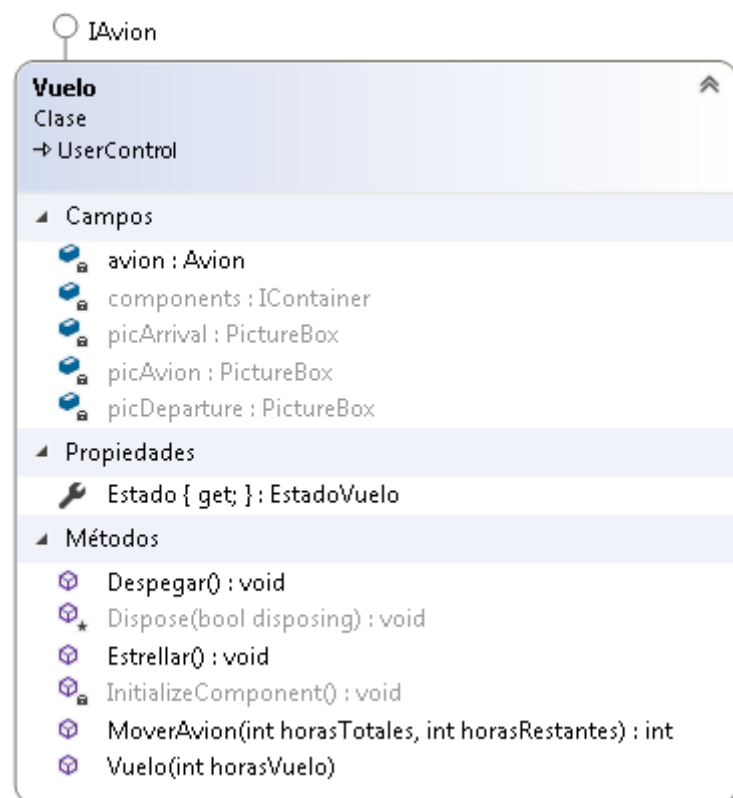
Apellido.Nombre.AñoCursada.zip y dejar este último en el Escritorio de la máquina. Luego presionar el botón  de la barra superior, **colocar un mensaje** y presionar *Aceptar*. **Aguardar a que el profesor indique que el examen fue copiado de forma correcta.** Luego retirarse del aula.

TIEMPO MÁXIMO PARA RESOLVER EL EXAMEN 110 MINUTOS.

1. Modificar el nombre tanto a la carpeta y como al proyecto entregados por el descripto anteriormente.
2. Generar un método de extensión para la clase Aeropuerto para que cada vez se agregue un vuelo se agregue su información (fecha y hora actual y cantidad de horas de vuelo programadas):
 - a. Alumnos de final: en un log.txt ubicado en el escritorio.
 - b. Alumno de segundo parcial: en base de datos.
3. Generar un proyecto llamado Entidades y colocar dentro el siguiente esquema de clases:



4. Dentro del proyecto Controles encontrar el UserControl Vuelo. Agregar lo que haga falta según este esquema:



NOTA: Tener en cuenta que Despegar, Estrellar y Estado sólo llamarán a sus iguales dentro de Avión.

Aeropuerto:

5. El tipo genérico deberá tener una restricción para el tipo IAvion.
6. El operator + agregará un vuelo a la lista y retornará a `vuelos.Count * 50`.
7. El conversor explícito retornará la cantidad de vuelos activos (volando) con el formato "El aeropuerto cuenta con X vuelos en estado Volando". Para esto utilizar la propiedad Estado de Avión.
8. FinalizarVuelos cancelará todos los hilos activos.

Avión:

9. El estado será:
 - a. Si el hilo no existe (aun no despegó) será Programado.
 - b. Si el hilo está activo, será Volando.
 - c. Si el hilo ya finalizó, será Aterrizado.
10. Despegar declarará e inicializará el hilo con el método Volar.
11. Estrellar finalizará el hilo si este está vivo.
12. Volar:
 - a. Dormirá el hilo durante 1 segundo.
 - b. Descontará una Hora Restante al vuelo.
 - c. Invocará el evento Reportar Estado con las horas totales del vuelo y las horas restantes como argumentos. El método retornará el porcentaje ya completado.

UserControl Vuelo:

13. En su constructor declarar el Avión y asociar el evento con Mover Avión.
14. Despegar hará visible el picAvion e invocará al Despegar de avión.
15. Completar código de MoverAvion (ayuda: sólo agregar código al IF, el ELSE está completo).

Formulario Aerolínea:

16. Utilizar el menú Archivo y agregar:
 - a. Los alumnos de FINAL deberán agregar los métodos para serializar y deserializar en los 2 modos.
 - b. Los alumnos de Parcial deberán serializar y deserializar en 1 modo y guardar el log en la base de datos (colocar su nombre en la columna "alumno").
 - c. TODOS: en los botones no utilizados agregar un MessageBox con el mensaje "No elegí este método".
17. Alumnos Recuperatorio:
 - a. Agregar una excepción propia llamada ErrorArchivoException.
 - b. ErrorArchivoException será la única excepción lanzada por los métodos de Serialización y Deserialización creados.
 - c. Se encapsulará en su InnerException la excepción capturada.
 - d. En el formulario se capturará y se mostrará un MessageBox con toda la información de la excepción.

Test Unitarios:

18. Agregar un test unitario que valide que lo deserializado de un archivo sea igual a lo serializado previamente.

Script Base de Datos:

19. Crear la base de datos final-20190711

```
USE [final-20190711]
GO
CREATE TABLE [dbo].[Bitacora](
    [entrada] [varchar](200) NOT NULL,
    [alumno] [varchar](100) NOT NULL
) ON [PRIMARY]
GO
```