

Universidad Tecnológica Nacional

Facultad Regional Avellaneda



Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos

Materia: Laboratorio de Programación II

Apellido:		Fecha:	08/05/2018
Nombre:		Docente ⁽²⁾ :	F. Dávila
División:	2ºD	Nota ⁽²⁾ :	
Legajo:		Firma ⁽²⁾ :	
Instancia ⁽¹⁾ :	<div style="display: flex; justify-content: space-around;"> PP X RPP </div>	<div style="display: flex; justify-content: space-around;"> SP RSP FIN </div>	

(1) Las instancias validas son: 1º Parcial (PP), Recuperatorio 1º Parcial (RPP), 2º Parcial (SP), Recuperatorio 2º Parcial (RSP), Final (FIN). Marque con una cruz.

(2) Campos a ser completados por el docente.

IMPORTANTE:

- 2 (dos) errores en el mismo tema anulan su puntaje.**
- La correcta documentación y reglas de estilo de la cátedra serán evaluadas.**
- Colocar sus datos personales en el nombre del proyecto principal, colocando: Apellido.Nombre.Division. Ej: Pérez.Juan.2D. No se corregirán proyectos que no sea identificable su autor.
- 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.
- Colocar nombre de la clase (en estáticos), **this** o **base** en todos los casos que corresponda.

TIEMPO MÁXIMO PARA RESOLVER EL EXAMEN 90 MINUTOS.

- Crear una solución con el nombre en el siguiente formato: [APELLIDO].[NOMBRE]
- Dentro crear 3 proyectos: *Entidades* (Class Library), *VistaConsola* (Console) y *VistaForm* (WindowsForms).
- Dentro del **Program**, en el **Main** de *VistaConsola*, colocar el siguiente código para probar las entidades:

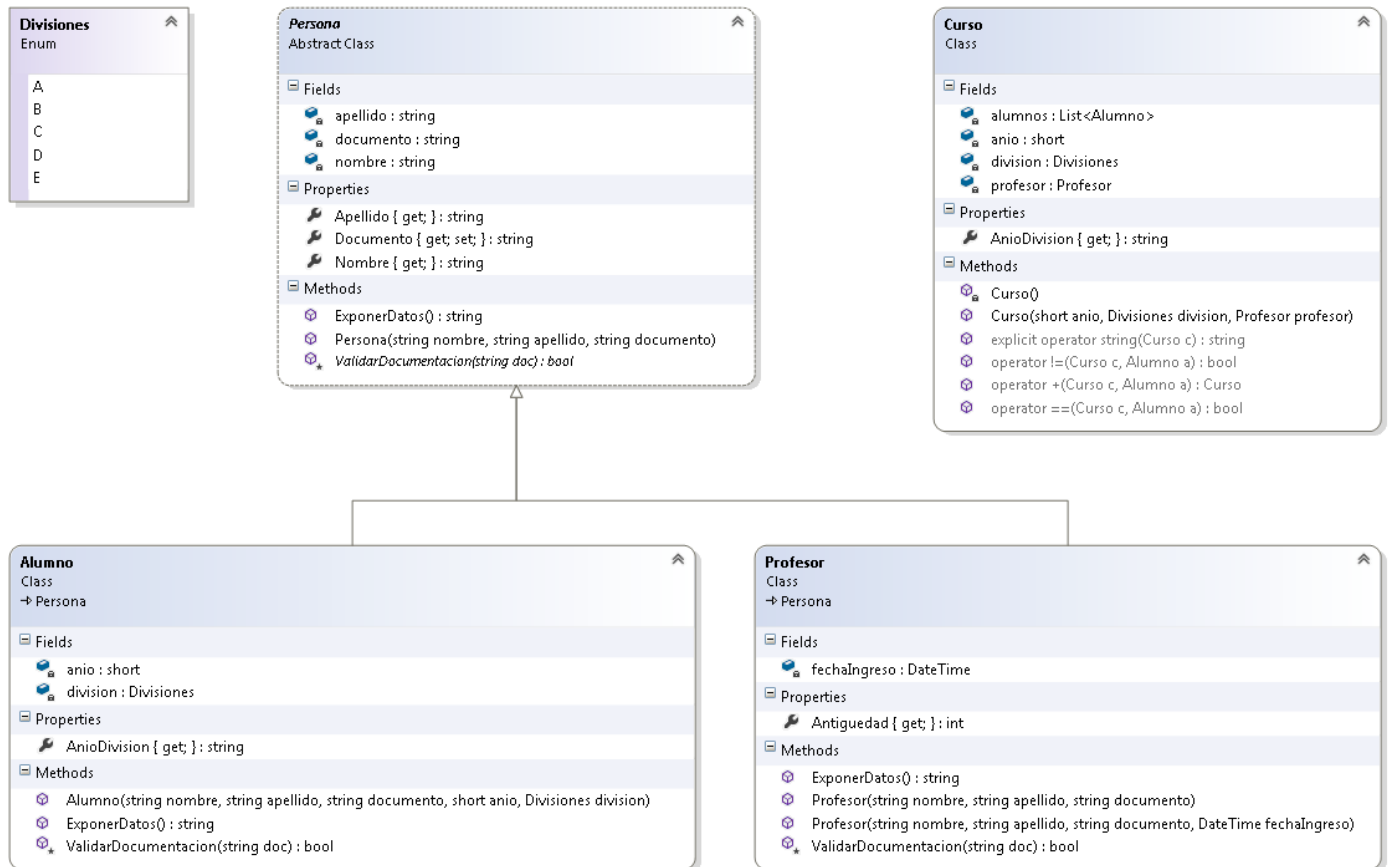
```
// Genero un curso nuevo
Curso curso = new Curso(2, Divisiones.A, new Profesor("Fede", "Dávila", "12345678", new
DateTime(2015, 03, 20)));

// Genero alumnos...
Alumno a1 = new Alumno("Juan", "López", "22-3333-2", 2, Divisiones.A);
Alumno a2 = new Alumno("José", "Martínez", "23-3343-6", 2, Divisiones.B);
Alumno a3 = new Alumno("María", "Gutiérrez", "22-3333-2", 2, Divisiones.A);
Alumno a4 = new Alumno("Marta", "Rodríguez", "23-3343-6", 2, Divisiones.A);
Alumno a5 = new Alumno("Marta", "Rodríguez", "233343126", 2, Divisiones.A);

// ... Y los agrego al curso
curso += a1;
curso += a2;
curso += a3;
curso += a4;
curso += a5;

// Imprimo los datos del curso
Console.WriteLine((string)curso);
Console.ReadKey();
```

4. Dentro de Entidades, diagramar las siguientes clases:



5. Tener en cuenta:

- ExponerDatos* retornará los datos de la clase dónde se lo coloque, utilizando `StringBuilder` para compilar dicha información.
- La **propiedad** *AnioDivision* retornará un string con el siguiente formato: `XºZ`, siendo `X` el año que se encuentra cursando y `Z` la división en letra (`A`, `B`, `C`, `D` o `E`).

6. Tener en cuenta dentro de Persona:

- ExponerDatos* tendrá implementación en todas las clases y podrá ser sobreescrita en las clases derivadas.
- ValidarDocumentacion* no tendrá implementación dentro de *Persona*.
- La **propiedad** *Documento* validará la documentación según corresponda.

7. Tener en cuenta dentro de Alumno:

- ValidarDocumentacion* dará como válido sólo documentos que tengan el siguiente formato `XX-XXXX-X` siendo las `X` números. Caso contrario retornará false y no se asignará el documento, siguiendo luego con el curso normal de la aplicación.

8. Tener en cuenta dentro de Profesor:

- Antigüedad* devolverá la cantidad de tiempo, en días, desde la fecha de ingreso del profesor hasta la actualidad.
- ValidarDocumentacion* dará como válido cuando el documento tenga exactamente 8 caracteres.

9. Tener en cuenta dentro de Curso:

- El **constructor privado** será el único lugar donde se instanciará la lista de alumnos.
- El **operador explícito** retornará los datos del curso y todos sus alumnos, utilizando `StringBuilder` para compilar dicha información.
- El **operador ==** entre *Curso* y *Alumno* informará true si el alumno pertenece al mismo Año y División que el curso.
- El **operador +** entre *Curso* y *Alumno* agregará al alumno al curso siempre y cuando su Año y División coincidan.

10. Divisiones tendrá su propio archivo, siendo también parte del namespace Entidades.

11. Por último, generar el siguiente **formulario** dentro de *VistaForm*:

12. Siendo los elementos a utilizar `GroupBox`, `Button`, `ComboBox`, `RichTextBox`, `Label`, `DateTimePicker`, `NumericUpDown`, `TextBox`.

13. Con el botón **btnCrearCurso** se instanciará un nuevo `Curso` con todos los datos cargados.

14. Con el botón **btnMostrar** se mostrará en el `RichTextBox` **rtbDatos** todos los datos del curso. De no existir el curso dar un error con un `MessageBox` (sólo botón OK, ícono de error).

15. Para agregar alumnos a un curso se utilizará el botón **btnAgregar**.

16. Para cargar los `ComboBox` **cmbDivisionCurso** y **cmbDivision** utilizar el siguiente código:

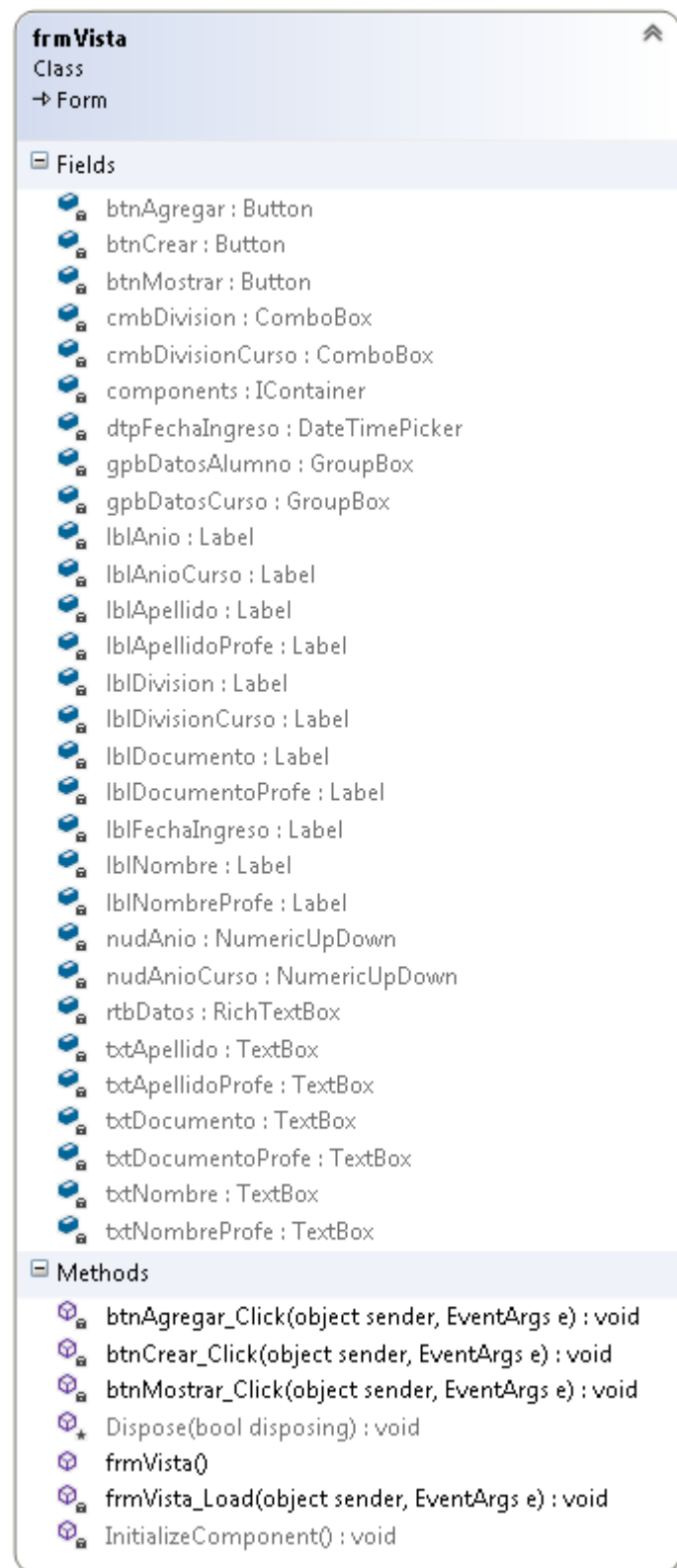
```
cmbX.DataSource = Enum.GetValues(typeof(Divisiones));
```

17. Para leer el elemento enumerado de esos combos, utilizar el siguiente código:


```
Divisiones division;  
Enum.TryParse<Divisiones>(cmbDivisionCurso.SelectedValue.ToString(), out division);
```

18. Respetar los nombres de todos los elementos.

19. De ser necesario, agregar **atributos** faltantes. El diagrama de clases del formulario será:



Al finalizar, colocar la carpeta de la Solución completa en un archivo ZIP que deberá tener como nombre Apellido.Nombre.division.zip y dejar este último en el Escritorio de la máquina.

Luego presionar el botón  de la barra superior, colocar un mensaje y apretar **Aceptar**. Finalmente retirarse del aula y aguardar por la corrección.