

Universidad Tecnológica Nacional

Facultad Regional Avellaneda



Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos

Materia: Laboratorio de Programación II

Apellido:		Fecha:	10/05/2018
Nombre:		Docente ⁽²⁾ :	F. Dávila
División:	2°C	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 </div>	<div style="display: flex; justify-content: space-around;"> FIN </div>

(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:

- **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.Departamento. 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.

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

```
// Genero un oficina nuevo
Oficina oficina = new Oficina(2, Departamentos.A, new Jefe("Fede", "Dávila", "12345678", new
DateTime(2015, 03, 20)));

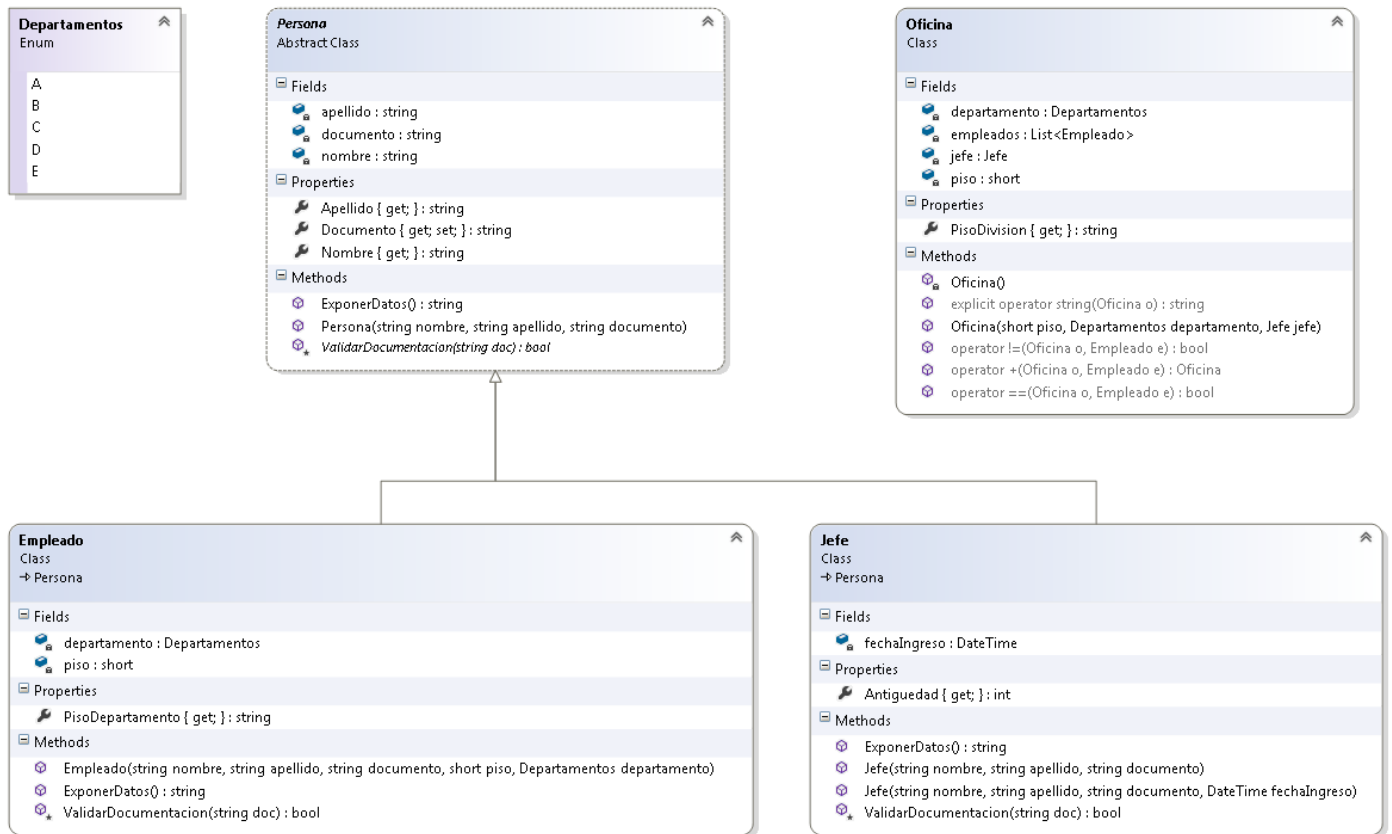
// Genero empleados...
Empleado a1 = new Empleado("Juan", "López", "22-3333-2", 2, Departamentos.A);
Empleado a2 = new Empleado("José", "Martínez", "23-3343-6", 2, Departamentos.B);
Empleado a3 = new Empleado("María", "Gutiérrez", "22-3333-2", 2, Departamentos.A);
Empleado a4 = new Empleado("Marta",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

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

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

`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** `PisoDepartamento` retornará un string con el siguiente formato: `X°Z`, siendo X el piso que se encuentra cursando y Z el departamento 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 Empleado:

- `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 Jefe:

- `Antigüedad` devolverá la cantidad de tiempo, en días, desde la fecha de ingreso del jefe hasta la actualidad.
- `ValidarDocumentacion` dará como válido cuando tenga 8 caracteres.

9. Tener en cuenta dentro de Oficina:

- El **constructor privado** será el único lugar donde se instanciará la lista de empleados.
- El **operador explícito** retornará los datos de la oficina y todos sus empleados, utilizando `StringBuilder` para compilar dicha información.
- El **operador** `==` entre **Oficina** y **Empleado** informará true si el empleado pertenece al mismo Piso y División que el oficina.
- El **operador** `+` entre **Oficina** y **Empleado** agregará al empleado a la oficina siempre y cuando su Piso y División coincidan.

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

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

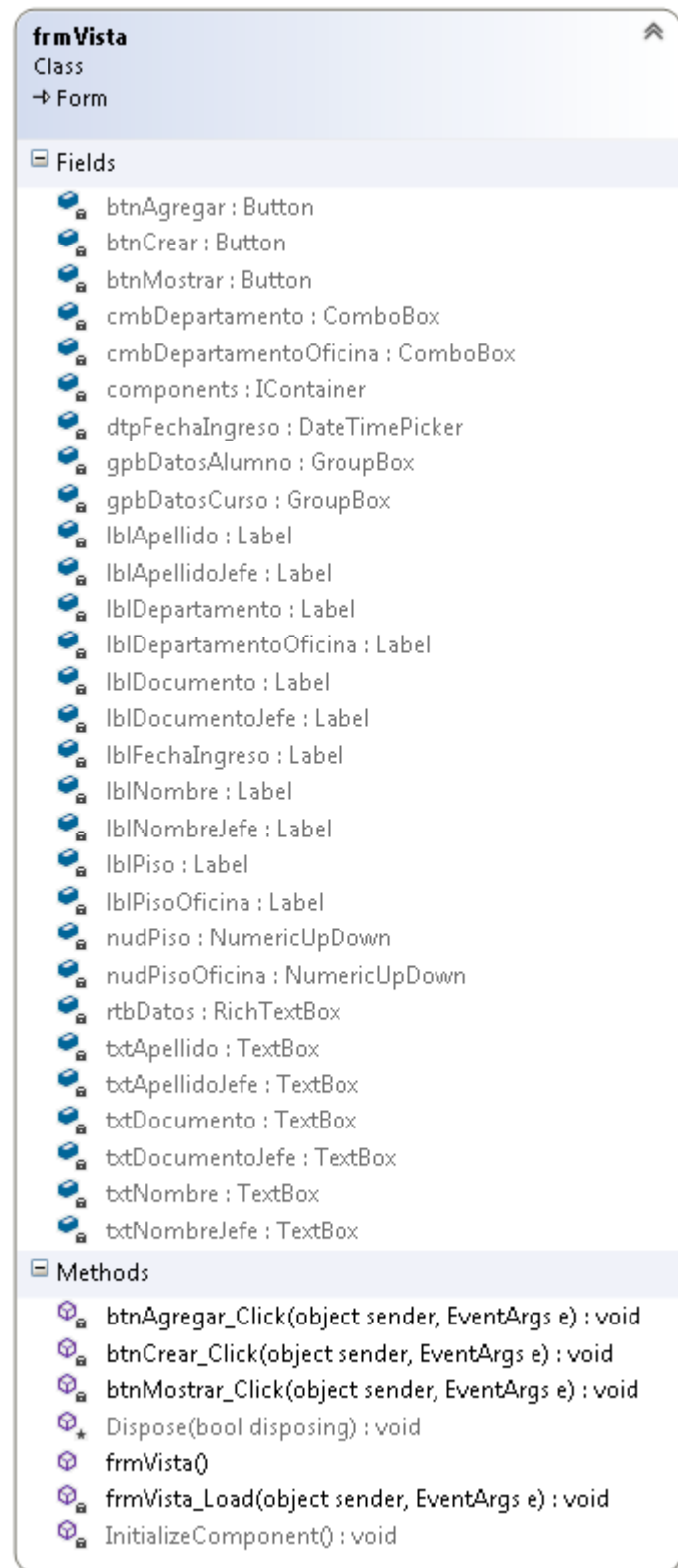
12. Siendo los elementos a utilizar `GroupBox`, `Button`, `ComboBox`, `RichTextBox`, `Label`, `DateTimePicker`, `NumericUpDown`, `TextBox`.
13. Con el botón **btnCrearOficina** se instanciará una nueva Oficina con todos los datos cargados. Caso contrario dar un error con un `MessageBox` (sólo botón OK, ícono de error).
14. Con el botón **btnMostrar** se mostrará en el `RichTextBox` **rtbDatos** todos los datos del oficina.
15. Para agregar empleados a una oficina se utilizará el botón **btnAgregar**.
16. Para cargar los `ComboBox` **cmbDepartamentoOficina** y **cmbDepartamento** utilizar el siguiente código:

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


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

```
Departamentos dptos;  
Enum.TryParse<Departamentos>(cmbDepartamentoOficina.SelectedValue.ToString(), out dptos);
```

18. Respetar los nombres de todos los elementos.
19. 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.