|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Universidad Tecnológica Nacional**  **Facultad Regional Avellaneda** | | | | | | | | | | | | |
| Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos | | | | | | | | | | | | |
| Materia: Laboratorio de Programación II | | | | | | | | | | | | |
| Apellido: |  | | | | Fecha: | | | | 07/12/2017 | | | |
| Nombre: |  | | | | Docente(2): | | | |  | | | |
| División: |  | | | | Nota(2): | | | |  | | | |
| Legajo: |  | | | | Firma(2): | | | |  | | | |
| Instancia(1): | **PP** |  | **RPP** |  | | **SP** |  | **RSP** | |  | **FIN** | X |

**(1)** Las instancias validas 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.

1. ***IMPORTANTE:***

* ***2 (dos) errores en el mismo tema anulan su puntaje.***
* ***Cada tema vale 1 (un) punto (Herencia, Generics, Test Unitarios, etc.). La correcta documentación también será evaluada.***
* ***Se deberán tener al menos el 60% de los temas de cada parcial bien para lograr la aprobación.***
* ***Errores de conceptos de POO anulan el punto.***
* Colocar sus datos personales en el nombre del proyecto principal, colocando: Apellido.Nombre.AñoCursada. Ej: Pérez.Juan.2017. No sé corregirán proyectos que no sea identificable su autor.
* **TODAS** las clases e interfaces 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 apretar **Aceptar.** Luego retirarse del aula y aguardar por la corrección.

*TIEMPO MÁXIMO PARA RESOLVER EL EXAMEN* ***100*** *MINUTOS.*

Crear un proyecto con el nombre

1. Crear un proyecto de WindowsForms y a través de su conocimiento sobre los mismos:
   1. Cambiar el título del Formulario, agregándole por CÓDIGO (sin usar las propiedades de la vista de diseño) el nombre del alumno y el examen rendido con el siguiente formato: "Final – Juan Pérez".
   2. Agregar 5 botones llamados Punto 1 hasta Punto 5 cuyo nombre será btnPunto1 a btnPunto5.
   3. Agregar los eventos y manejadores para el click del mouse a cada uno de los botones.
   4. Agregar un RichTextBox llamado rtbTextoSalida.
   5. Crear un método llamado CargarRichTextBox que reciba un string y lo asigne a rtbTextoSalida.
   6. Agregar el código para que pueda ser escrito desde un hilo:

delegate void CargarRichTextBoxCallback(string datos);

private void CargarRichTextBox(string datos)

{

if (rtbTextoSalida.InvokeRequired)

{

CargarRichTextBoxCallback d = new CargarRichTextBoxCallback(CargarRichTextBox);

this.Invoke(d, new object[] { datos });

}

else

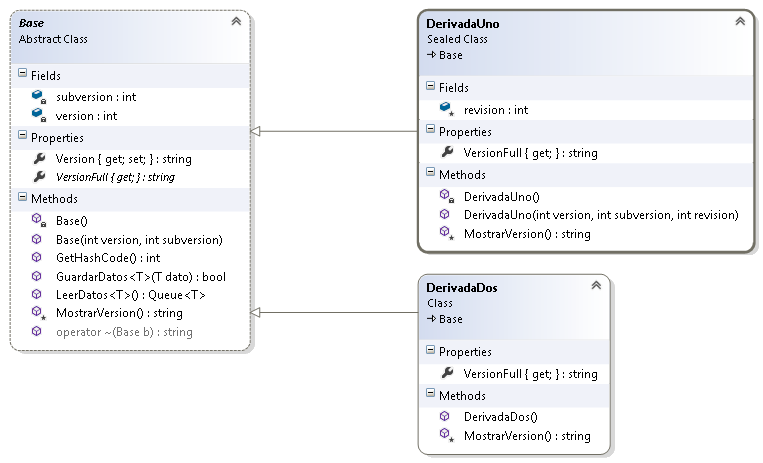
{

// Código del alumno!!!!

}

}

1. Crear una Biblioteca de clases llamada EntidadesPP y dentro:



* 1. Crear la clase Base (abstracta).
     1. Sus dos atributos son estáticos.
     2. Tendrá un constructor estático que inicializará a versión como 1 y subversión como 0.
     3. La propiedad Versión asignará el valor sólo si está validado que sea numérico, sino no asignará nada.
     4. VersionFull es de sólo lectura y abstracto.
     5. MostrarVersion es virtual y, utilizando string.Format, retornará versión.subversion (ej.: con los datos que asigna el constructor por defecto "1.0").
     6. El operador ~ invertirá los datos retornados por MostrarVersion (ej.: con los datos que asigna el constructor por defecto "0.1").
     7. GetHashCode sumará y retornará los valores de versión y subversión.
  2. Crear las clases DerivadaUno (sellada) y DerivadaDos que heredan de Base.
     1. MostrarVersion agregará los datos propios.
     2. DerivadaUno cuenta con un constructor estático.
     3. VersionFull publicará los datos de MostrarVersion.
  3. Crear un atributo llamado listaElementos del tipo List<Base>.
  4. Instanciar dicha lista en el constructor.
  5. Dentro del evento click del botón btnPunto1 agregar y completar el siguiente código, mostrando a través del RichTextBox los datos de todos los elementos:

Base derUno1 = new DerivadaUno(10, 11, 12);

DerivadaUno derUno2 = new DerivadaUno(1, 2, 3);

Base derDos1 = new DerivadaDos();

listaElementos.Add(derUno1);

listaElementos.Add(derUno2);

listaElementos.Add(derDos1);

// Generar el código para obtener todos los datos de los elementos de la lista y luego utilizar CargarRichTextBox para mostrarlos

string auxiliar = "";

this.CargarRichTextBox(auxiliar);

1. En el btnPunto2:
   1. Quienes estén haciendo el **primer parcial**:
      1. En la clase Base se deben sobrecargar al menos un operador unario y uno binario.
      2. Sobrecargar GetHashCode para base que retorne la suma de version + subversion y para derivadas que sume datos propios.
      3. Repetir el código del Punto 2 con una pila e iterandola con un while.
      4. Generar un indexador para base, siendo el indice 0 la versión y el 1 la subversion. Retornar -1 en cualquier otro subindice.
      5. A la clase Base, crearle la conversión implícita a string, retornando MostrarVersion.
      6. A las clases DerivadaUno y DerivadaDos crearle la conversión explícita a string, retornando MostrarVersion.
      7. Generar el código necesario para probar todos los métodos hechos e imprimirlos por pantalla. Utilizar StringBuilder.
   2. Quienes estén haciendo el **segundo parcial** y **final**:
      1. Colocar un MessageBox que diga "Para primer parcial" con sólo el botón de Aceptar y el icono de exclamación (Exclamation).

**FIN PRIMER PARCIAL**

1. Utilizando lo aprendido para base de datos y genéricos:
   1. Crear una base de datos llamada **exactamente** final-20171207. Correr el siguiente script dentro del SQL Management Studio / Nueva Consulta.

USE [final-20171207]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Datos](

[version] [int] NOT NULL,

[subversion] [int] NOT NULL,

[revision] [int] NULL,

[derivada] [int] NOT NULL

) ON [PRIMARY]

GO

**NOTA:** el campo "derivada" indicará 1 para objetos del tipo DerivadaUno y 2 para objetos del tipo DerivadaDos.

* 1. Crear un método estático y genérico para Base bool GuardarDatos<T>(T dato), dónde T sólo pueda ser del tipo Base.
  2. Este guardará Version y Subversion en una tabla de la base de datos creada.
  3. Crear un método estático y genérico para Base bool Queue<T> LeerDatos<T>(), dónde T sólo pueda ser del tipo Base.
  4. Este retornará una cola de elementos de Base.

1. Crear el método void EjecutarEvento(List<Base>) dentro de la clase Base:
   1. EjecutarEvento guardará en la base de datos los elementos de la lista.
   2. Luego lanzará un evento retornando una cola con los elementos presentes en la base de datos utilizando LeerDatos<T>().
   3. Lanzar un Thread que ejecute el método EjecutarEvento y pasarle como parámetro la lista listaElementos.
   4. Mostrar el resultado del evento por pantalla a través del método ManejadorEvento(Queue<Base>).