

**Universidad Tecnológica Nacional
Facultad Regional Avellaneda**



Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos

Materia: Laboratorio de Programación II

Apellido:		Fecha:	02/08/2018
Nombre:		Docente ⁽²⁾ :	F. Dávila / H. Dillon
División:		Nota ⁽²⁾ :	
Legajo:		Firma ⁽²⁾ :	
Instancia ⁽¹⁾ :	PP	RPP	SP
		RSP	FIN
			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 proyecto debe ser 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 apretar **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. Crear una nueva base de datos llamada **final-20180802** y dentro de la misma crear la tabla "Archivo" ejecutando el script sql otorgado.
2. General:
 - a. Utilizar la teoría de encapsulamiento en **todas** las clases.
3. Clase Almacenador
 - a. Crear un constructor que reciba y asigne el/los atributos de la misma.
 - b. La clase debe ser abstracta.
 - c. Crear un método abstracto llamado MostrarArchivos que retorne void.
4. Interfaz IAlmacenable

- a. Crear la interfaz genérica IAlmacenable con los métodos bool Guardar(V elemento) y T Leer(string path)
5. Clase Archivo:
- a. Sobreescribir el método ToString para mostrar los valores de sus atributos. Utilizar String.Format.
 - b. Agregar el operador explicit para retornar el contenido del archivo.
6. Clase DiscoElectronico:
- a. Deberá heredar de Almacenador e implementar IAlmacenable.
 - b. El método Guardar deberá insertar un archivo en la base de datos.
 - c. El método Leer recibirá el nombre de la tabla a consultar. Deberá leer y retornar todos los archivos de la base de datos.
 - d. Tanto en Leer como en Guardar capturar y relanzar las excepciones.
 - e. El método MostrarArchivos por el momento sólo deberá recorrer la lista de archivos y por cada uno simular un retardo de 5 segundos.
 - f. Agregar un constructor que reciba la capacidad y en el cual se deberá cargar la lista a partir de los datos guardados en la base.
 - g. El constructor privado inicializará la lista. Por defecto la capacidad será 5.
 - h. Sobrecargar el operador + para agregar un archivo a la lista siempre y cuando no supere la capacidad, caso contrario lanzará una excepción con el mensaje "El disco está lleno!".
7. Clase ArchiveroFisico:
- a. Deberá heredar de Almacenador e implementar IAlmacenable.
 - b. Crear un constructor que reciba y asigne el/los atributos de la misma.
 - c. El método MostrarArchivos lanzará una excepción del tipo NotImplementedException.
 - d. El método Guardar deberá guardar un objeto de tipo Archivo en un archivo de texto en la ubicación definida en el atributo pathArchivos.
 - e. El método Leer recibirá el nombre de un archivo y deberá retornar su contenido.
 - f. Tanto en Leer como en Guardar capturar y relanzar las excepciones.
8. Formulario:
- a. En el evento Load del formulario instanciar el DiscoElectrónico y el ArchiveroFisico del Form con capacidad para 3 archivos c/uno.
 - b. Controlar excepciones al guardar/leer en archivos y base de datos mostrando mediante un MessageBox cuando ocurra alguna.
 - c. En el manejador del botón AlmacenarElectronico se deberá:
 - i. instanciar un archivo a partir de los datos obtenidos de los controles del formulario.
 - ii. Agregar el archivo a la lista del DiscoElectrónico siempre y cuando haya capacidad.
 - iii. Si se pudo agregar a la lista, guardarlo también en la base de datos.
 - iv. Finalmente limpiar el contenido de los controles del formulario.
 - d. En el manejador del botón AlmacenarFisico se deberá:
 - i. Instanciar un archivo a partir de los datos obtenidos de los controles del formulario.
 - ii. Guardarlo en un archivo de texto
 - iii. Finalmente limpiar el contenido de los controles del formulario.
 - e. En el manejador del botón LeerFisico se deberá, a partir del nombre ingresado en txtNombreArchivo, recuperar el contenido del archivo y mostrarlo en el rtbContenido.
 - f. Agregar en la clase Almacenador un evento llamado MostrarInfo el cual recibirá un string y retornará void.
 - g. Agregar también en la clase Almacenador un método llamado DispararEvento que recibirá un archivo por parámetro e invocará al evento MostrarInfo con los datos del archivo.
 - h. En el manejador del botón LeerElectronico se deberá:
 - i. Asociar el manejador del formulario MostrarArchivo al evento MostrarInfo de la clase DiscoElectronico.
 - ii. Ejecutar en un hilo el método MostrarArchivos de la clase DiscoElectronico.
 - i. Modificar el método MostrarArchivos de la clase DiscoElectrónico para que ejecute el método DispararEvento por cada archivo de la lista.
 - j. Antes de cerrar, en el evento FormClosing, abortar el hilo del formulario en caso de que siga vivo.