|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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: 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:***

* 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 sé 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:
   1. Utilizar la teoría de encapsulamiento en **todas** las clases.
3. Clase Almacenador
   1. Crear un constructor que reciba y asigne el/los atributos de la misma.
   2. La clase debe ser abstracta.
   3. Crear un método abstracto llamado MostrarArchivos que retorne void.
4. Interfaz IAlmacenable
   1. Crear la interfaz genérica IAlmacenable con los métodos bool Guardar(V elemento) y T Leer(string path)
5. Clase Archivo:
   1. Sobreescribir el método ToString para mostrar los valores de sus atributos. Utilizar String.Format.
   2. Agregar el operador explicit para retornar el contenido del archivo.
6. Clase DiscoElectronico:
   1. Deberá heredar de Almacenador e implementar IAlmacenable.
   2. El método Guardar deberá insertar un archivo en la base de datos.
   3. El método Leer recibirá el nombre de la tabla a consultar. Deberá leer y retornar todos los archivos de la base de datos.
   4. Tanto en Leer como en Guardar capturar y relanzar las excepciones.
   5. El método MostrarArchivos por el momento sólo deberá recorrer la lista de archivos y por cada uno simular un retardo de 5 segundos.
   6. 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.
   7. El constructor privado inicializará la lista. Por defecto la capacidad será 5.
   8. 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:
   1. Deberá heredar de Almacenador e implementar IAlmacenable.
   2. Crear un constructor que reciba y asigne el/los atributos de la misma.
   3. El método MostrarArchivos lanzará una excepción del tipo NotImplementedException.
   4. El método Guardar deberá guardar un objeto de tipo Archivo en un archivo de texto en la ubicación definida en el atributo pathArchivos.
   5. El método Leer recibirá el nombre de un archivo y deberá retornar su contenido.
   6. Tanto en Leer como en Guardar capturar y relanzar las excepciones.
8. Formulario:
   1. En el evento Load del formulario instanciar el DiscoElectrónico y el ArchiveroFisico del Form con capacidad para 3 archivos c/uno.
   2. Controlar excepciones al guardar/leer en archivos y base de datos mostrando mediante un MessageBox cuando ocurra alguna.
   3. En el manejador del botón AlmacenarElectronico se deberá:
      1. instanciar un archivo a partir de los datos obtenidos de los controles del formulario.
      2. Agregar el archivo a la lista del DiscoElectrónico siempre y cuando haya capacidad.
      3. Si se pudo agregar a la lista, guardarlo también en la base de datos.
      4. Finalmente limpiar el contenido de los controles del formulario.
   4. En el manejador del botón AlmacenarFisico se deberá:
      1. Instanciar un archivo a partir de los datos obtenidos de los controles del formulario.
      2. Guardarlo en un archivo de texto
      3. Finalmente limpiar el contenido de los controles del formulario.
   5. 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.
   6. Agregar en la clase Almacenador un evento llamado MostrarInfo el cual recibirá un string y retornará void.
   7. 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.
   8. En el manejador del botón LeerElectronico se deberá:
      1. Asociar el manejador del formulario MostrarArchivo al evento MostrarInfo de la clase DiscoElectronico.
      2. Ejecutar en un hilo el método MostrarArchivos de la clase DiscoElectronico.
   9. Modificar el método MostrarArchivos de la clase DiscoElectrónico para que ejecute el método DispararEvento por cada archivo de la lista.
   10. Antes de cerrar, en el evento FormClosing, abortar el hilo del formulario en caso de que siga vivo.