TAREA PARA PROG07
Curso 2023-24
Comunicación con el usuario. Interfaces (Swing)

ENUNCIADO

- **La tarea no se dará por entregada si el proyecto tiene errores de sintaxis, no compila, no se ejecuta correctamente, si lo entregado no se corresponde con lo pedido o si no se ha realizado un mínimo de la tarea, es decir, si está vacía o casi.**
- En esta tarea hay que entregar:
    - Un proyecto en NetBeans llamado **Tarea_Prog07_Apellido1Apellido2Nombre** (Nombre es vuestro nombre y primer apellido), que gestione la **venta de muebles de un almacén**.
    - Un documento **PDF** llamado: **Tarea_Prog07_Apellido1Apellido2Nombre.pdf,** en el que pondremos **captura de pantallas** en la que aparezca como **fondo** vuestra conexión **al Papas** los resultados de la ejecución del ejercicio (tal y como se vería con Netbeans), para así mostrar que funcionan (debe aparecer una prueba ejecutada del ejercicio).
- Se trata de realizar un proyecto Java llamado: **Tarea_Prog07_Apellido1_Apellido2_Nombre** que gestione la **venta de muebles de un almacén**.
- Esos datos se almacenarán en un **fichero de objetos serializado**, denominado **MUEBLES.DAT**.
- Se trata de realizar un proyecto Java llamado: **Tarea_Prog07_Apellido1_Apellido2_Nombre** que gestione la venta de muebles, estos muebles se almacenarán en un **fichero serializado**, denominado **MUEBLES.DAT**. Es el mismo utilizado en la tarea de la PROG06.
- Los datos que necesitamos almacenar de los muebles son:
    - **codigo**(código del mueble) :número entero de 5 caracteres
    - **descripMueble:** cadena de 15 caracteres
    - **precioUnitario:** real: 5 enteros y 2 decimales.
    - **unidadesAlmacen:** número entero de 5 caracteres (unidades que hay de ese mueble en el almacén)
    - **unidadesMinimas:** número entero de 5 caracteres (unidades mínimas que debe de haber del mueble)
    - **tipoMueble**: 1 carácter (Valores corretos:H: Hogar/D:Despacho/C:Colegios)

- En esta tarea se realizará con un interfaz gráfico (SWING) para gestionar los muebles.
- Se realizará los siguientes paquetes:
    - **modelos**: donde se creará la clase **Muebles** serializable**,** en esta clase estarán los getter y setter de los atributos.
    - **controlador**: en este paquete realizaremos la clase **OperacionesFichero**
    - **app**: en la crearemos el formulario(JFrame) **AppGestionMueblesNombre** (donde Nombre será vuestro nombre).

Clase: OperacionesFichero

Métodos **estáticos**:

- **buscaRegistro:** que se le pasará como entrada el código del mueble, buscará en el fichero MUEBLES.DAT y nos devolverá como salida el objeto de tipo Mueble cuyo código le demos como entrada. Si no existe, devuelve null en el objeto.
  public static Mueble buscaRegistro(int codigo)

- **existeRegistro:** le pasaremos un código y nos devolverá true si existe ese código en el fichero y false si no existe.
  public static boolean existeRegistro(int codigo)

- **grabarRegistro:** le daremos como entrada un objeto de tipo Mueble y lo grabará el fichero. Hay que tener en cuenta que si el fichero ya existe hay que utilizar **MiObjectOutputStream**, para eliminar las cabeceras.
  public static boolean grabarRegistro(Mueble m)

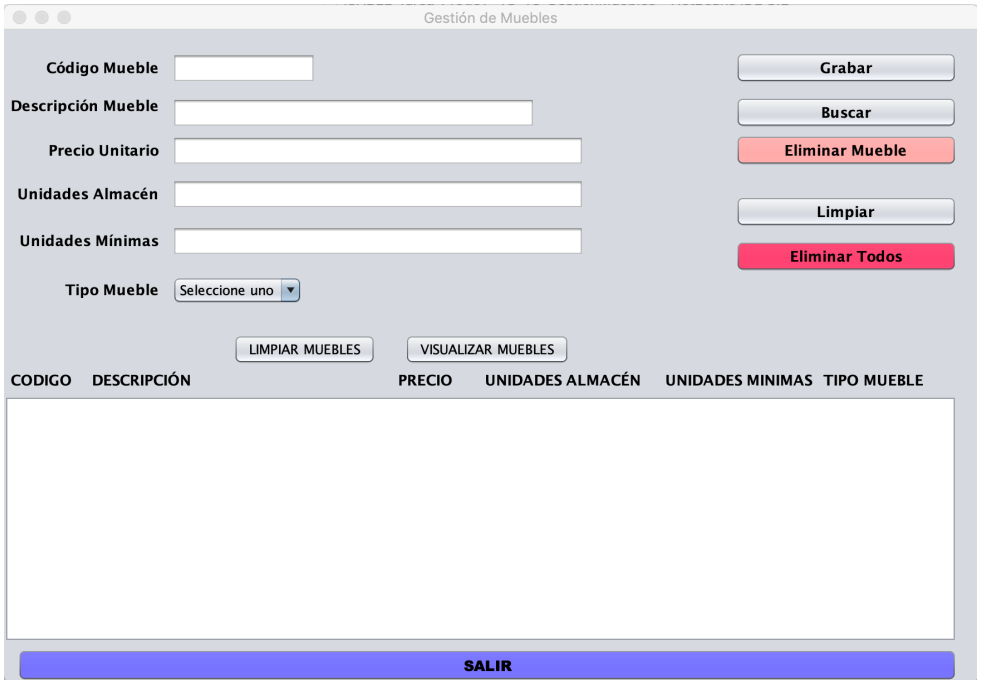
- **eliminarRegistro**: se le pasará como entrada un código.
  public static boolean eliminarRegistro(int codigo)

  Utilizaremos directamente un fichero temporal, llamado TEMPORAL.TMP, el cual lo abriremos para grabar todos los datos del fichero de MUEBLES.DAT menos el que queremos elimiar.
  Iremos leyendo del fichero de objetos MUEBLES.DAT, cada objeto leído que no sea el objeto a eliminar, lo iremos grabando en el fichero temporal.
  Cuando encontremos el objeto a eliminar no lo grabaremos en el fichero temporal.
  Cuando se acabe el fichero MUEBLES.DAT, cerraremos ambos ficheros.
  Finalmente tendremos que renombrar el fichero temporal.tmp y darle el nuevo nombre MUEBLES.DAT.

- **eliminarTodos**: método que eliminará el fichero MUEBLES.DAT.
  public static eliminarTodos()

Formulario: AppGestionMueblesNombre:

Tendrá el siguiente formato:



- Los campos serán:

  - **Código Mueble**: será un JTextField. (nombre :txtCodigo)
  - **Descripción del mueble**: será un JTextField. (nombre : txtDescripMueble)
  - **Precio unitario**: será un JTextField. (nombre : txtPrecioUnitario)
  - **Unidades almacén**: será un JTextField. (nombre : txtUnidadesAlmacen)
  - **Unidades mínimas** en almacén: será un JTextField. (nombre : txtUnidadesMinimas)
  - **Tipo mueble:** será un JComboBox, en el que se podrá elegir: Hogar/Despacho/Colegios. En el fichero almacenaremos: H, D o C (Nombre: cmbTipoMueble)
  - Botones:
    - **Grabar**: nombre: btnGrabar
    - **Buscar**: nombre: btnBuscar

        - **Limpiar**: nombre: btnLimpiar
        - **Eliminar Todo**: nombre: btnEliminarTodo
        - **Visualizar Muebles:** nombre: btnVisualizarMuebles
        - **Limpiar Muebles:** nombre: btnLimpiarMuebles
        - **Salir**: nombre: btnSalir

- **Cuando txtCodigo pierda el foco (evento: txtCodigoFocusLost):** controlaremos que el tipo de datos sea numérico, si no lo es visualizaremos el mensaje "FORMATO INCORRECTO DEL CODIGO" mediante un JOptionPane.showMessageDialog. Después controlaremos que la longitud de txtCodigo no sea mayor de 5, si es así, visualizaremos el mensaje "LONGITUD DEL CODIGO DEBE SER <=5" mediante un JOptionPane.showMessageDialog.
- **Lo mismo hay que controlar para txtPrecioUnitario, txtUnidadesAlmacen y txtUnidadesMInimas.**
- **Para txtDescripMueble:** controlaremos que la longitud no sea mayor de 15, si es así, visualizaremos el mensaje "LONGITUD DE LA DESCRIPCIÓN DEBE SER <=15" mediante un JOptionPane.showMessageDialog
- **Botón Grabar:**
    - Si el txtCodigo está vacío: visualizaremos un mensaje con JOptionPane.showMessageDialog: "CODIGO NO PUEDE ESTAR VACÍO" y el foco volverá al txtCodigo.
    - Si el codigo existe (llamaremos al método **OperacionesFichero.existeRegistro** para saberlo), visualizaremos un mensaje con JOptionPane.showMessageDialog: "CODIGO YA EXISTE" y el foco volverá al txtCodigo.
    - Pasaremos los valores de los campos al objeto con los métodos set.
    - Grabaremos el mueble en el fichero mediante el método: **OperacionesFichero.grabarRegistro**. Si se graba correctamente, visualizaremos el mensaje "MUEBLE GRABADO" mediante un JOptionPane.showMessageDialog. Si no se graba, visualizaremos el mensaje "ERROR AL GRABAR EL MUEBLE" mediante un JOptionPane.showMessageDialog.
    - Limpiaremos los campos del formulario, para ello crearemos el método **limpiarDatos**.
    - El fofo volverá al txtCodigo.
- **Botón Buscar:**
    - Si el txtCodigo está vacío: visualizaremos un mensaje con JOptionPane.showMessageDialog: "CODIGO NO PUEDE ESTAR VACÍO" y el foco volverá al txtCodigo.
    - Si no está vacío, llamaremos al método **OperacionesFichero.buscarRegistro.**
        - Si el txtCodigo **no** existe (cuando devuelva un valor null), visualizaremos un mensaje con JOptionPane.showMessageDialog: "CODIGO NO EXISTE" y el foco volverá al txtCodigo.
        - Si el txtCodigo existe, visualizaremos los datos del mueble. Para ello realizaremos en esta clase un método llamado **visualizarRegistro,** al que le daremos como entrada un objeto de tipo Mueble y visualizará cada campo en la ventana.
- **Botón Eliminar Mueble**:
    - Si el txtCodigo está vacío: visualizaremos un mensaje con JOptionPane.showMessageDialog: "CODIGO NO PUEDE ESTAR VACÍO" y el foco volverá al txtCodigo.
    - Si no está vacío, llamaremos al método **OperacionesFichero.existeRegistro:**
        - Si el txtCodigo **no** existe (, visualizaremos un mensaje con JOptionPane.showMessageDialog: "CODIGO NO EXISTE" y el foco volverá al txtICodigo.
        - Si existe, mediante un JOptionPane.showMessageDialog, preguntará si se desea realmente eliminar ese mueble, si la respuesta es afirmativa, se llamará al método **OperacionesFichero.eliminarRegistro.** Finalmente, se visualizará mediante un JOptionPane.showMessageDialog el mesaje "MUEBLE ELIMINADO".

- **Botón Limpiar**: limpiará todos los campos del mueble que están en el formulario.
- **Botón Eliminar Todo**: Mediante un JOptionPane.showMessageDialog, preguntará si se desea realmente eliminar todos los muebles, si la respuesta es afirmativa, se llamará al método **OperacionesFichero.eliminarTodos.** y se visualizará mediante un JOptionPane.showMessageDialog el mesaje "TODOS LOS MUEBLES ELIMINADOS".
- **Botón Visualizar Muebles:** visualizaremos en el txtArea todos muebles que haya en el fichero. Para ello, primero limpiamos el jTextArea, después iremos recorriendo el fichero y visualizaremos todos los

atributos de cada vino. En el TIPO MUEBLES saldrá: Hogar, Despacho o Colegios, en vez de H/D o C. Si el fichero está vació se visualizará un mensaje de error.

- **Botón Limpiar Muebles:** limpiará el área de visualización todos los vinos.

El código fuente Java debe incluir **comentarios** en cada atributo (o en cada conjunto de atributos) y método (o en cada conjunto de métodos del mismo tipo) indicando su utilidad.

CRITERIOS DE PUNTUACIÓN. TOTAL 10 PUNTOS.

Para poder empezar a aplicar estos criterios es necesario que la aplicación no tenga errores de sintaxis, compile y se ejecute correctamente. En caso de no hacerlo, la puntuación será directamente de **1,00.**

1. **Formulario AppGestionMuebles: 6 puntos**
2. **Clase : OperacionesFichero**: **2 puntos**
3. **Aspectos generales**: **2 puntos**: buen diseño proyecto, bien estructurado, etc. Originalidad de la solución: de 0 a 1 puntos. Mejoras no incluidas en las reglas mínimas pero que mejoran los requisitos mínimos : 0 a 1 punto.

RECURSOS NECESARIOS PARA REALIZAR LA TAREA.

- Ordenador personal.
- JDK y JRE de Java SE.
- Entorno de desarrollo Apache NetBeans con las funcionalidades necesarias para desarrollar y emular midlets.

CONSEJOS Y RECOMENDACIONES.

- Básate en el ejemplo **UT07_GestionEmpleadosSwing** que he dejado en Recursos Adicionales aportados por el profesor.

INDICACIONES DE ENTREGA.

- Lo que debes entregar:
  - Un proyecto en **NetBeans** llamado: **Tarea_Prog07_Apellido1_Apellido2_Nombre**
  - Además del proyecto crearemos un documento **PDF** llamado: **Tarea_Prog07_Apellido1_Apellido2_Nombre.pdf** en el que pondremos **captura de pantallas** en la que aparezca como **fondo** vuestra conexión **al Papas** y como primer plano con las capturas del código fuente y la ejecución del ejercicio (tal y como se vería con Netbeans), para así mostrar que funcionan (debe aparecer una prueba ejecutada de las diferentes opciones del ejercicio).
- Comprimir el proyecto NetBeans y el documento pdf creados en un fichero llamado: **Tarea_Prog07_Apellido1_Apellido2_Nombre.zip**
- Asegúrate que el nombre no contenga la letra ñ, tildes ni caracteres especiales extraños
- En caso de tener que realizar un segundo envío, le daremos el siguiente nombre: **Tarea_Prog07_Apellido1_Apellido2_Nombre _ENVIO2.zip**