

Tarea para PROG07.

Detalles de la tarea de esta unidad.

Enunciado.

Te han encargado que diseñes e implementes una aplicación llamada **GesTenis**, que se ocupará de gestionar el circuito profesional de tenis de la ATP (Asociación de Tenistas Profesionales). El programa debe ser capaz de crear tenistas y torneos y almacenar el palmarés de cada tenista junto a su puntuación. Por palmarés entenderemos aquellos torneos que un tenista ha ganado, y su puntuación irá en función del tipo de torneo ganado. Así, la **Tabla 1** muestra la relación entre los diferentes tipos de torneos a tener en cuenta y su puntuación:

TIPO DE TORNEO	PUNTUACIÓN
Grand Slam	2000
ATP World Tour Masters 1000	1000
ATP World Tour 500	500
ATP World Tour 250	250

Tabla 1: Tipo de torneos y puntuación

Para el desarrollo de la aplicación, vamos a considerar tres partes diferenciadas: el diseño e implementación de las clases **Tenista** y **Torneo**, el **diseño de la interfaz gráfica** de la aplicación y la **implementación de las funciones** que debe realizar la aplicación.

Diseño e implementación de clases(2 puntos)

Implementar la clase **Torneo** con las siguientes características **(0.75 puntos)**:

- Debe implementar el interfaz **Serializable**.
- Debe tener dos **atributos privados**: nombre (String) y puntuación (int).
- Deberá disponer del **constructor** que cree el objeto inicializando los dos atributos privados.
- Debe tener los métodos **get** y **set** para cada uno de los atributos.
- Implementar el método: **public static ArrayList<Torneo> cargar(File fichero)**. Se le pasará por parámetro un fichero que contiene los datos de los torneos y devolverá los mismos en un **ArrayList** de torneos. Si se produce algún error, el método devolverá **null**.
- Implementar el método: **public static boolean guardar(ArrayList<Torneo> lista, File fichero)**. Se le pasará por parámetro el **ArrayList** con la lista de torneos a guardar y el fichero donde se almacenarán los datos. Devolverá **true** si los datos se guardan correctamente y **false** en caso contrario.

Implementar la clase **Tenista** con las siguientes características **(1.25 puntos)**:

- Debe implementar el interfaz **Serializable**.
- Debe tener tres **atributos privados**: nombre (String), edad (int) y palmarés (ArrayList<Torneo>).
- El **constructor** de la clase debe tener los parámetros nombre y edad. El palmarés estará vacío inicialmente.
- Debe disponer de métodos **get** y **set** para los atributos nombre y edad.

- Implementar el método: **public String [] getPalmares()**. Devolverá un array con los nombres de los torneos ganados por el tenista.
- Implementar el método: **public void añadirPalmares(Torneo torneo)**. Añadirá el torneo pasado por parámetro al palmarés del tenista.
- Implementar el método: **public int getPuntuacionATP()**. Devolverá la puntuación obtenida por el tenista según su palmarés.
- Implementar el método: **public static ArrayList<Tenista> cargar(File fichero)**. Se le pasará por parámetro un fichero que contiene los datos de los tenistas y devolverá los mismos en un **ArrayList** de tenistas. Si se produce algún error, el método devolverá **null**.
- Implementar el método: **public static boolean guardar(ArrayList<Tenista> lista, File fichero)**. Se le pasará por parámetro el **ArrayList** con la lista de tenistas a guardar y el fichero donde se almacenarán los datos. Devolverá **true** si los datos se guardan correctamente y **false** en caso contrario.

Diseño de la interfaz gráfica(2 puntos)

La interfaz gráfica de la aplicación **GesTenis** se compondrá de tres ventanas. Se podrá utilizar cualquier gestor de distribución (incluido **NullLayout**). A continuación se muestra cómo debe ser el aspecto de cada una de estas ventanas:

- La ventana **Principal(1 punto)** deberá ser de tipo **JFrame** y tendrá el aspecto que muestra la **Figura 1**.

- La barra de menús contendrá los **menús Archivo y Añadir**:
 - El menú **Archivo** deberá tener los siguientes elementos o items: **inicializar, cargar**

tenistas, guardar tenistas, cargar torneos, guardar torneos y salir.

- El menú **Añadir** tendrá los siguientes elementos o ítems: **Tenista** y **Torneo**.

En la Figura 1 se observan **tres paneles** diferenciados:

- El primero para realizar la búsqueda de tenistas.
 - El segundo para mostrar los datos de los tenistas. El palmarés del tenista debe mostrarse en un componente **JComboBox**. La propiedad **editable** de los componentes de este panel debe estar a **false**.
 - El tercero para añadir torneos al palmarés de un tenista. La lista de torneos se muestra en un componente **JList**.
- La ventana **JDTenista(0.5 puntos)** será del tipo **JDialog** y tendrá el aspecto que se muestra en la **Figura 2**.



- La ventana **JD Torneo(0.5 puntos)** será del tipo **JDialog** y tendrá el aspecto que se muestra en la **Figura 3**. Los elementos del **JComboBox** se cargarán manualmente y serán los tipos de torneo que se indicaron en la **Tabla 1**.



Implementación de funciones de la aplicación(6 puntos)

A continuación se describen las características de implementación de cada una de las ventanas diseñadas en el apartado

anterior y sus componentes:

- **JDTenista (0.5 puntos):**
 - Tendrá un **atributo privado** de tipo **boolean** llamado **haPulsadoAceptar** y un **método público** con el mismo nombre que nos devolverá el valor de dicho atributo.
 - Al pulsar el botón **“Aceptar”** pondremos el valor del atributo privado **haPulsadoAceptar** a **true** y al pulsar sobre el botón **“Cancelar”** pondremos su valor a **false**. Después pondremos en ambos casos la visibilidad del diálogo a **false** (**setVisible(false)**).
 - En caso de pulsar el botón **“Aceptar”**, se comprobarán los datos de los cuadros de texto y si no son válidos se mostrará un mensaje de error y la ventana seguirá siendo visible.
 - Tendrá un **método público** nombrado **getTenista** que nos devolverá una instancia de la clase **Tenista** con los valores introducidos en los cuadros de texto.
- **JDTorneo (0.5 puntos):**
 - Tendrá un **atributo privado** de tipo **boolean** llamado **haPulsadoAceptar** y un **método público** con el mismo nombre que nos devolverá el valor de dicho atributo.
 - Al pulsar el botón **“Aceptar”** pondremos el valor del atributo privado **haPulsadoAceptar** a **true** y al pulsar sobre el botón **“Cancelar”** pondremos su valor a **false**. Después pondremos en ambos casos la visibilidad del diálogo a **false** (**setVisible(false)**).
 - En caso de pulsar el botón **“Aceptar”**, se comprobarán los datos de los cuadros de texto y si no son válidos se mostrará un mensaje de error y la ventana seguirá siendo visible.
 - Tendrá un **método público** nombrado **getTorneo** que nos devolverá una instancia de la clase **Torneo**. Para crear el torneo, hemos de tener en cuenta la puntuación del torneo según el elemento seleccionado en el **JComboBox**. La relación entre el tipo de torneo y la puntuación se puede consultar en la **Tabla 1**.
- **Principal (5 puntos):**
 - Contendrá los siguientes **atributos privados (0.1 puntos)**:
 - La **lista de tenistas** de tipo **ArrayList<Tenista>**.
 - La **lista de torneos** de tipo **ArrayList<Torneo>**.
 - Un **índice** que indique la posición en la lista del tenista que se está mostrando en pantalla.
 - El **modelo de datos** del **JComboBox** que mostrará el palmarés del tenista de tipo **DefaultComboBoxModel**.
 - El **modelo de datos** para el **JList** que mostrará la lista de torneos de tipo **ListModel**.
 - Se deberán implementar los siguientes **métodos privados**:
 - **private void inicializar()**. Inicializa la lista de tenistas y torneos, el índice y los componentes de la interfaz gráfica para que no muestren ningún dato. Para el caso del modelo de datos del **JComboBox** con el palmarés de los tenistas se creará un **DefaultComboBoxModel** vacío (llamada al constructor de esta clase). El modelo de datos del **JList** lo tomaremos de la lista de torneos para lo que tendremos que crearnos una clase (clase interna o inner class) del tipo **AbstractListModel**. Esta clase debe implementar los métodos **int getSize()** y **Object getElementAt(int index)**. (1 punto)
 - **private void mostrarDatosTenista(int indice)**. Muestra los datos del tenista en la posición **indice** de la lista. Se debe establecer el modelo del **JComboBox**

pasándole al constructor de **DefaultComboBoxModel** un array con los nombres de los torneos ganados por el tenista. **(0.75 puntos)**

- El **constructor** de la clase principal debe llamar al método **inicializar** después de la llamada a **initComponents**.
- Los métodos **ActionPerformed** que deben implementarse y su comportamiento se listan a continuación:
 - **Panel Buscar – Cuadro de texto (0.5 puntos)**: Buscará un tenista en la lista con el mismo nombre que el introducido en el cuadro de texto. Si lo encuentra establecerá el valor del atributo **índice** a la posición del tenista en la lista y llamará al método **mostrarDatosTenista** para que se muestren los datos en la aplicación. En caso de que la lista esté vacía o el tenista no se haya encontrado se mostrará un mensaje de error con **JOptionPane**.
 - **Panel Torneos – Botón Añadir (0.5 puntos)**: Añadirá el torneo seleccionado del **JList** al palmarés del tenista actual. Si no hay tenista o torneo seleccionado se mostrará un mensaje de error con **JOptionPane**.
 - **Menú Archivo – Inicializar (0.1 puntos)**: Llamará al método **inicializar**, que se encarga de inicializar todos los datos del programa.
 - **Menú Archivo – Cargar Tenistas (0.5 puntos)**: Mostrará un **JFileChooser** para seleccionar el archivo a cargar. Se mostrará un mensaje con **JOptionPane** tanto si el archivo se carga con éxito como si ocurre algún error.
 - **Menú Archivo – Guardar Tenistas (0.5 puntos)**: Mostrará un **JFileChooser** para seleccionar el archivo donde guardar los datos. Se mostrará un mensaje con **JOptionPane** tanto si el archivo se guarda con éxito como si ocurre algún error.
 - **Menú Archivo – Cargar Torneos (0.25 puntos)**: Mostrará un **JFileChooser** para seleccionar el archivo a cargar. Se mostrará un mensaje con **JOptionPane** tanto si el archivo se carga con éxito como si ocurre algún error.
 - **Menú Archivo – Guardar Torneos (0.25 puntos)**: Mostrará un **JFileChooser** para seleccionar el archivo donde se guardarán los datos. Se mostrará un mensaje con **JOptionPane** tanto si el archivo se guarda con éxito como si ocurre algún error.
 - **Menú Archivo – Salir (0.15 puntos)**: Finalizará el programa.
 - **Menú Añadir – Tenista (0.2 puntos)**: Creará y hará visible la ventana **JDTenista** con la opción **modal** a **true**. Se comprobará si el usuario pulsa el botón aceptar y se añadirá el nuevo tenista a la lista de tenistas.
 - **Menú Añadir – Torneo (0.2 puntos)**: Creará y hará visible la ventana **JD Torneo** con la opción **modal** a **true**. Se comprobará si el usuario pulsa el botón aceptar y se añadirá el nuevo torneo a la lista de torneos. Será necesario llamar al método **updateUI** del **JList** para que aparezca el nuevo torneo.