*Unidad 1*

*Introducción al acceso a datos y manejo de ficheros*

Tarea AD01

Manejando información almacenada en ficheros

[1. Enunciado de la tarea 2](#_Toc216301894)

[2. Crear una clase Usuario 4](#_Toc216301895)

[3. Implementación del menú 7](#_Toc216301896)

[4. Agregar un usuario a la lista de usuarios 8](#_Toc216301897)

[5. Borrar un usuario de la lista de usuarios 11](#_Toc216301898)

[6. Guardar la lista de usuarios en un archivo (serialización) 13](#_Toc216301899)

[7. Cargar la lista de usuarios desde el archivo (deserialización) 15](#_Toc216301900)

[8. Mostrar los usuarios en consola 19](#_Toc216301901)

[9. Exporta a fichero .txt la lista de usuarios 21](#_Toc216301902)

[10. Salir de la aplicación 23](#_Toc216301903)

[11. ¿Existe el archivo ‘user.dat’? 24](#_Toc216301904)

[12. Advertencia sobre datos no guardados antes de salir 25](#_Toc216301905)

[13. ¿Recuperar datos de disco? 26](#_Toc216301906)

[14. Resultados del programa en consola 27](#_Toc216301907)

[14.1. Inicio del programa 27](#_Toc216301908)

[14.2. Agregar un usuario 29](#_Toc216301909)

[14.3. Mostrar los usuarios en consola 30](#_Toc216301910)

[14.4. Borrar usuario 31](#_Toc216301911)

[14.5. Guardar lista (Serialización) 33](#_Toc216301912)

[14.6. Cargar lista (deserialización) 34](#_Toc216301913)

[14.7. Exportación archivo TXT 35](#_Toc216301914)

[14.8. ¿Cargar datos guardados? 36](#_Toc216301915)

[14.9. Salida del programa 38](#_Toc216301916)

### 

## Enunciado de la tarea

En esta tarea debes realizar una aplicación en Java que se utilizará para gestionar los usuarios de un comercio online. El proyecto Java creado deberá llamarse **Nombre\_Apellido1\_AD1\_E1** y deberá tener en cuenta los siguientes requisitos:

- Debes crear una clase **Usuario** que contenga los siguientes atributos:

* identificador (de tipo String)
* contraseña (de tipo String)
* dirección (de tipo String)
* año de nacimiento (de tipo int)

- El objetivo principal es serializar una lista de objetos de la clase **Usuario**, de modo que se pueda guardar en un archivo, y que de igual modo se pueda deserializar esa lista desde el archivo.

- La aplicación debe permitir lo siguiente (implementando un menú para ello):

Texto

El contenido generado por IA puede ser incorrecto.

Consideraciones:

* No olvides usar la interfaz **Serializable**.
* Asegúrate de manejar las excepciones correctamente.
* El archivo debe ser un archivo binario (**user.dat**), salvo el **user.txt** al que se exporte con la opción al efecto.
* Para borrar un usuario se deberá facilitar su identificador.
* Al ejecutarse la aplicación, se comprobará si existe el fichero**user.dat**. Si no existe, se avisará al usuario de que la aplicación no tiene datos grabados. Si existe, se cargarán los datos guardados de modo que los usuarios que hubiera en el fichero se cargarán en una lista.
* En caso de que, al ejecutar la aplicación, se modifique algún dato, y el usuario intente salir de la aplicación sin haber guardado los datos, se le avisará indicando que, si se sale sin guardar los datos, perderá los cambios realizados. Podría ser con un mensaje como:***"Ha habido cambios en el programa que todavía no se han guardado. Si desea guardarlos ejecute la opción correspondiente del menú. Si sale ahora no se guardarán. ¿Está seguro de que desea salir sin guardar?"***.
* Si se ejecuta la opción de recuperar datos, habiendo realizado cambios en el programa, se advertirá al usuario que los cambios se perderán, puesto que se cargarán los datos del fichero en la lista, y se le pedirá confirmación antes de continuar: "***Ha realizado cambios que no ha guardado en disco. Si continúa la carga del archivo se restaurarán los datos de disco y se perderán los cambios no guardados. ¿Desea continuar con la carga y restaurar los datos del archivo? (S/N).***"
* Se deben mostrar todos los mensajes informativos que se consideren oportunos.

## Crear una clase Usuario

Texto

El contenido generado por IA puede ser incorrecto.

Se crea una clase Usuario implementando el método constructor que define los cuatro atributos solicitados en la tarea, identificador (de tipo String), contraseña (de tipo String), dirección (de tipo String) y año de nacimiento (de tipo int).

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Se implementan los métodos getters y setters para obtener o establecer los atributos anteriores para cada objeto usuario creado mediante la clase Usuario.

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Importante, para poder serializar una lista de objetos de la clase Usuario, de modo que se pueda guardar en un archivo, y que de igual modo se pueda deserializar esa lista desde el archivo, la clase Usuario debe implementarse desde la interfaz Serializable.

## Implementación del menú

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

Se implementa un bucle do-while para reproducir el menú tantas veces como el String salidaPermitida distinta de "S" (Salida permitida = SI) se presenten por parte del usuario. Se deberá insertar intencionadamente la letra S para salir del programa, evitando errores de salida del programa al pulsar accidentalmente cualquier otra tecla del teclado.

## Agregar un usuario a la lista de usuarios

Texto

El contenido generado por IA puede ser incorrecto.

Una vez seleccionada la opción 1 por el usuario, se procede a solicitar los cuatro atributos que definen el objeto usuario de la clase Usuario, para terminar con la instanciación de dicho objeto usuario, así como la aplicación del método estático agregarUsuario() de la clase Usuario aportando como argumento el objeto usuario recientemente instanciado. Se captura la posible excepción en la cual el usuario del aplicativo pudiera introducir un dato no numérico para la definición del atributo ‘año de nacimiento’.

Texto

El contenido generado por IA puede ser incorrecto.

En la clase Usuario se crea una lista temporal donde se almacenarán los distintos objetos usuarios que se vayan instanciando, se trata de un atributo estático privado de la clase Usuario.

Texto

El contenido generado por IA puede ser incorrecto.

En esta imagen se presenta el método estático agregarUsuario() para incorporar a la lista temporal indicada con anterioridad cada uno de los usuarios que se vayan dando de alta. Se tiene la precaución de solicitar al usuario de la aplicación que, en caso de que el usuario a introducir ya se encuentre almacenado en la lista temporal, para ello se comprueba si el atributo identificación ya existe entre los usuarios almacenados en la lista temporal, en tal caso, el usuario deberá introducir el carácter “S” si desea actualizar el resto de los atributos. En caso de que no exista el objeto usuario en la lista temporal, se procede a añadirlo.

Se emplea un bucleo do-while tal que mientras el usuario no exista vaya comprobando en cada elemento de la lista temporal el atributo identificación, y con la ayuda de un contador.

## Borrar un usuario de la lista de usuarios

Texto

El contenido generado por IA puede ser incorrecto.

Para borrar un usuario de la lista temporal de usuarios, el usuario del programa deberá seleccionar la opción 2, acto seguido, el programa solicitará la identificación del usuario a eliminar para ejecutar el método estático de la clase Usuario borrarUsuario() pasando como argumento el atributo identificador del usuario a borrar.

Texto

El contenido generado por IA puede ser incorrecto.

Se presenta el método borrarUsuario estático de la clase Usuario, haciendo uso de un iterador para garantizar el borrado del objeto usuario contenido en la lista temporal de manera segura. Se hace uso de una variable tipo booleano para manejar el mensaje de salida por consola de la existencia del objeto usuario a eliminar en la lista de usuarios temporal.

## Guardar la lista de usuarios en un archivo (serialización)

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Para guardar la lista temporal de usuarios en un fichero ubicado en el disco duro del PC, el usuario deberá establecer la opción 3, de esta manera se conseguirá la persistencia de los datos quedando grabados en un archivo del disco duro.

Se ejecuta el método estático de la clase Usuario guardarLista() aportando como argumento el nombre del archivo en cuestión.

Texto

El contenido generado por IA puede ser incorrecto.

Se muestra el método estático de la clase Usuario guardarLista, para el cual, se emplean objetos de las clases FileOutPutStream y ObjectOutPutStream, para transformar la lista de objetos usuarios en bytes (out), y luego, estos bytes guardarlos en el archivo en cuestión (fileOut).

## Cargar la lista de usuarios desde el archivo (deserialización)

Texto

El contenido generado por IA puede ser incorrecto.

El usuario del programa seleccionará la opción 4 para cargar los datos persistentes del fichero ubicado en el disco duro del PC. Posteriormente, se ejecutará el método estático de la clase Usuario cargarListaGrabada aportando el nombre del fichero a implementar. La lista de usuarios almacenada en tal fichero se guardará en la lista de usuarios deserializada, y posteriormente, se comparará con la lista temporal por medio del método estático cargarLista() para determinar si ambas listas son iguales, además de, comprobar la existencia del fichero en cuestión, para que en caso de que no lo sean o no exista el fichero, el usuario del programa podrá determinar si desea actualizar los datos o realizar cualquier otra opción del menú.

Texto

El contenido generado por IA puede ser incorrecto.

Se presenta el método cargarListaGrabada empleándose objetos FileInputStream y ObjectInputStream, el primero para obtener los bytes del archivo guardado en el disco, y el segundo para transformar los bytes en una lista de objetos usuarios que posteriormente se almacenarán en la lista de usuarios deserializada.

Texto

El contenido generado por IA puede ser incorrecto.

Posteriormente, con el método igualmente estático compararListas(), se compararán las listas deserializada y temporal para comprobar si son iguales, para ello, primero se comprueba si ambas listas tienen el mismo número de elementos, en segundo lugar, para cada elemento se comprueba que sus atributos sean iguales, a la vez que se comprueban que los usuarios siguen el mismo orden en ambas listas.

Texto

El contenido generado por IA puede ser incorrecto.

Se muestra el método cargarLista(), mediante el cual se actualizan los objetos usuarios de la lista temporal a partir de los datos obtenidos del fichero grabado en el disco duro del PC. De manera similar al método cargarListaGrabada, se empleará un objeto FileInputStream envuelto en un objeto ObjectInputStream al cual se le pasa como argumento para obtener los bytes del archivo del disco duro y transformarlos en una lista de objetos usuarios.

## Mostrar los usuarios en consola

Texto

El contenido generado por IA puede ser incorrecto.

Cuando el usuario del programa seleccione la opción 5 se mostrarán los resultados en la consola, mostrando todos los usuarios almacenados en la lista temporal de usuarios.

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

Se presentan los métodos mostrarListaUsuarios(), estático, y toString(), para imprimir a modo de cadena de texto todos y cada uno de los objetos usuarios contenidos en la lista temporal de usuarios.

El método toString() devuelve un String cadena de caracteres mostrando todos los atributos que definen a cada usuario, según su método constructor presentado al inicio de este documento.

## Exporta a fichero .txt la lista de usuarios

Texto

El contenido generado por IA puede ser incorrecto.

Se selecciona la opción 6 para escribir en un archivo de texto cada uno de los objetos almacenados en la lista temporal. Para ello se emplea el método estático escribirTXT().

Se define como atributo estático público de la clase Usuario el nombre del archivo “user.txt” de esta manera al pulsar ENTER por defecto se usará tal archivo (de la misma manera se procede con el archivo user.dat al iniciar el programa).

Texto

El contenido generado por IA puede ser incorrecto.

El método escribirTXT estático, de la clase Usuario, emplea dos objetos para transformar los caracteres en bytes, un objeto BufferedWriter que envuelve a un FileWriter, el primero capta un flujo de caracteres y el segundo los transforma en bytes para finalmente hacerlos persistente en el fichero \*.txt en cuestión.

Previamente, se define una lista de cadenas de caracteres listaUsuariosTXT, limpia inicialmente, Strings obtenidos de aplicar el método toString() sobre cada objeto usuario almacenada en la lista temporal. Se emplea un bucle for-each para ello, y posteriormente se vuelve a utilizar para escribir cada String en el archivo .txt sumándole un salto de carro y de línea mediante el método newLine().

## Salir de la aplicación

Texto

El contenido generado por IA puede ser incorrecto.

Se selecciona la opción 0 para salir del programa, pero antes, se llamará al método cargarListaGrabada() pasando como argumento el nombre del fichero en disco, para posteriormente comprobar si la lista obtenida deserializada coincide con la lista temporal y también comprobar si ese archivo existe en el disco duro, ofreciendo la oportunidad de volver atrás, no salir del programa y poder llevar a cabo otra opción ofrecida por el menú del aplicativo.

Finalmente, si el archivo existe y ambas listas de objetos usuarios, tanto la temporal como la deserializada, son iguales, el programa sí finalizará son preguntar al usuario dado que los cambios están guardados en el fichero del disco duro.

## ¿Existe el archivo ‘user.dat’?

Texto

El contenido generado por IA puede ser incorrecto.

Al ejecutarse la aplicación, se comprueba si existe el fichero user.dat. Para el caso, se ha tenido en cuenta que si el usuario presiona ENTER se usará el nombre user.dat por defecto. Para ello, se emplea el método estático comprobarExistenciaArchivo(). Si no existe, se avisará al usuario de que la aplicación no tiene datos grabados. Si existe, se cargarán los datos guardados de modo que los usuarios que hubiera en el fichero se cargarán en una lista empleando para ello el método estático cargarLista(), y luego, con el método mostrarListaUsuarios() mostrando en consola la lista de los objetos usuarios toString() almacenados en el fichero.

## Advertencia sobre datos no guardados antes de salir

Texto

El contenido generado por IA puede ser incorrecto.

En caso de ejecutar la aplicación y modificar algún dato, si el usuario del programa intenta salir de la aplicación sin haber guardado los cambios, se le avisa indicando que los cambios no han sido guardados. Tal y como se explicó en el punto anterior sobre la selección de la opción 0, se comprueba si la lista temporal es igual a la lista deserializada, y si no es así o el archivo no existe, se le avisará al usuario del programa antes de salir.

## ¿Recuperar datos de disco?

Texto

El contenido generado por IA puede ser incorrecto.

Si se ejecuta la opción de recuperar datos, habiendo realizado cambios en el programa, se advertirá al usuario que los cambios se perderán, puesto que se cargarán los datos del fichero en la lista, y se le pedirá confirmación antes de continuar. Este punto se tiene en cuenta al aplicar la opción 4 del menú, y en cierto modo ya se ha detallado cómo se procesa este aspecto, básicamente, se compara la lista deserializada del archivo con la lista temporal de tal manera que si son diferentes se le da la opción al usuario de retroceder antes de sustituir el contenido de la lista temporal.

El usuario deberá seleccionar el carácter “S” intencionadamente para que se proceda con la carga de la lista deserializada, evitando en cierto modo, que accidentalmente el usuario pudiera pulsar otra tecla del teclado.

## Resultados del programa en consola

### Inicio del programa

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Al iniciar el programa se pregunta al usuario del programa el nombre del archivo binario. Si pulsa ENTER se asignará el nombre por defecto ‘user.dat’.

Luego, si el archivo existe se imprime en consola y acto seguido se presenta la lista de usuarios guardados en el fichero.

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

En caso de no existir se indica que no existe, para ello, se establece un nombre no existente en el directorio raíz del proyecto. Luego, se muestra el menú del programa.

### Agregar un usuario

Texto

El contenido generado por IA puede ser incorrecto.

La opción 1 agrega un usuario previa solicitud de los atributos del objeto usuario. Acto seguido vuelve a mostrarse el menú.

### Mostrar los usuarios en consola

Texto

El contenido generado por IA puede ser incorrecto.

Se selecciona la opción 5 para mostrar todos los usuarios observándose en último lugar el usuario recientemente añadido (a la lista temporal).

### Borrar usuario

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Se selecciona la opción 2 y se pasa como argumento la identificación del usuario, se aporta la identificación del usuario recientemente aportado.

Texto

El contenido generado por IA puede ser incorrecto.

Se comprueba que el usuario ha sido eliminado con éxito volviendo a mostrar la lista de usuarios y observando que en último lugar ya no aparece el usuario anterior.

### Guardar lista (Serialización)

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Se selecciona la opción 3 del menú y se muestra un mensaje tras el guardado satisfactorio.

### Cargar lista (deserialización)

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Se selecciona la opción 4 del menú y se muestra un mensaje tras la carga satisfactoria.

### Exportación archivo TXT

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Se selecciona la opción 6 para proceder con la exportación del fichero de texto, se lanza un mensaje si se ha llevado a cabo satisfactoriamente y se vuelve a mostrar el menú.

### ¿Cargar datos guardados?

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

En caso de que el usuario pretenda cargar los datos guardado en el fichero binario, primeramente, el programa comparará la lista temporal de usuarios con la lista deserializada, de tal manera que si existen cambios de la opción de volver atrás y pueda guardar cambios o sustituya los valores si es finalmente la intención del usuario.

A modo de ejemplo se creó un usuario nuevo para alterar la lista temporal y así hacerla diferente a la deserializada.

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Solo si el usuario pulsa “S” serán cuando se sustituyan los valores, de esta manera se elimina la posibilidad de pulsar cualquier otra letra accidentalmente.

### Salida del programa

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Al salir del programa, opción 0 del menú, si existen datos sin guardar se le mostrará al usuario un mensaje de salida advirtiendo de ello, solo si el usuario pulsa sobre el carácter “S” saldrá del programa.