

Unidad 1

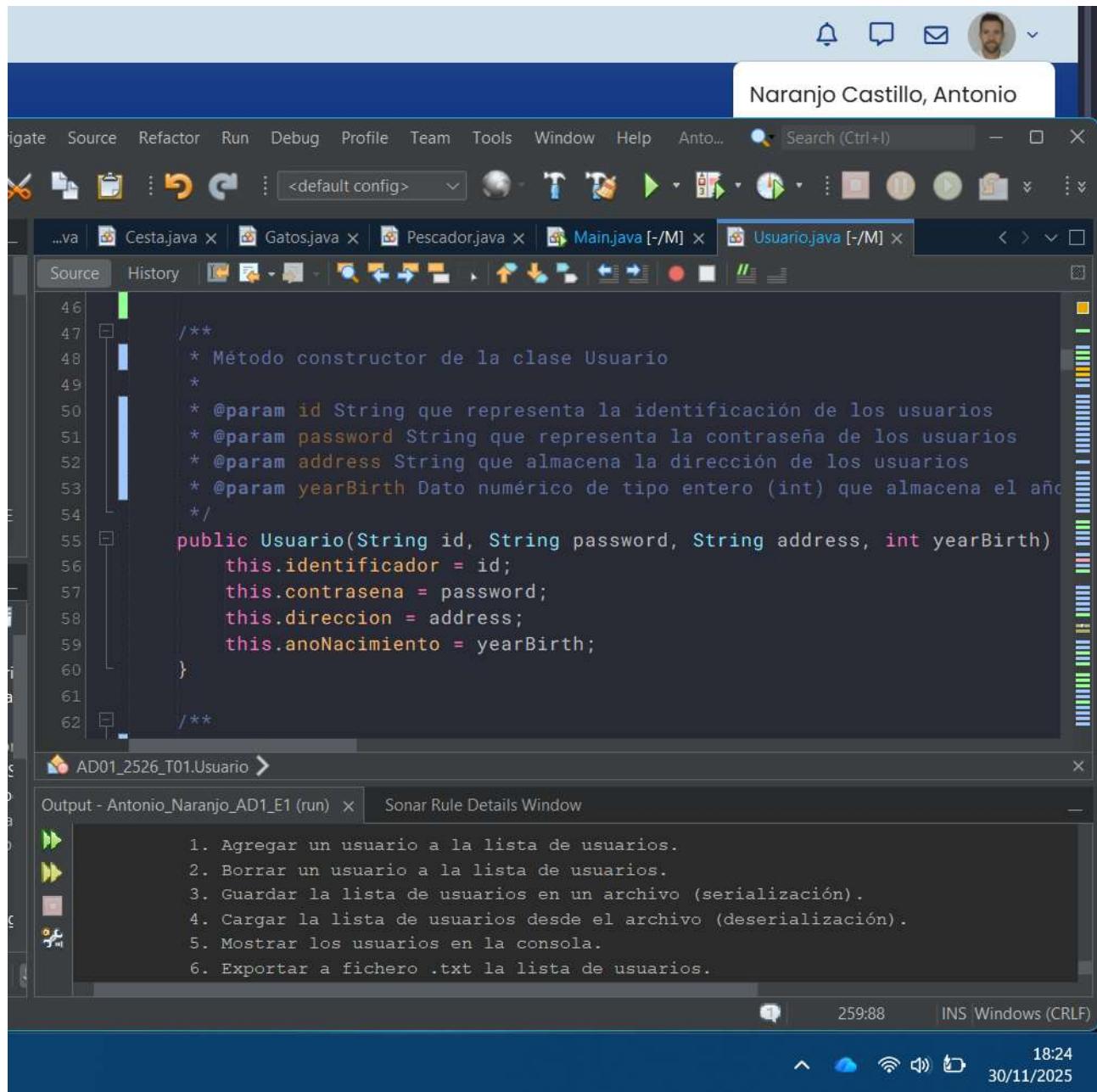
Introducción al acceso a datos y manejo de ficheros

Tarea AD01

Manejando información almacenada en ficheros

| | | |
|-----|--|----|
| 1. | Crear una clase Usuario..... | 2 |
| 2. | Implementación del menú | 5 |
| 3. | Agregar un usuario a la lista de usuarios..... | 6 |
| 4. | Borrar un usuario de la lista de usuarios..... | 9 |
| 5. | Guardar la lista de usuarios en un archivo (serialización)..... | 11 |
| 6. | Cargar la lista de usuarios desde el archivo (deserialización) | 13 |
| 7. | Mostrar los usuarios en consola | 17 |
| 8. | Exporta a fichero .txt la lista de usuarios..... | 19 |
| 9. | Salir de la aplicación | 21 |
| 10. | ¿Existe el archivo ‘user.dat’? | 22 |
| 11. | Advertencia sobre datos no guardados antes de salir | 23 |
| 12. | ¿Recuperar datos de disco? | 24 |
| 13. | Anexo. Resultados del programa en consola. | 25 |
| a. | Inicio del programa | 25 |
| b. | Agregar un usuario..... | 27 |
| c. | Mostrar los usuarios en consola | 28 |
| d. | Borrar usuario | 29 |
| e. | Guardar lista (Serialización) | 31 |
| f. | Cargar lista (deserialización)..... | 32 |
| g. | Exportación archivo TXT..... | 33 |
| h. | ¿Cargar datos guardados?..... | 34 |
| i. | Salida del programa..... | 36 |

1. Crear una clase Usuario

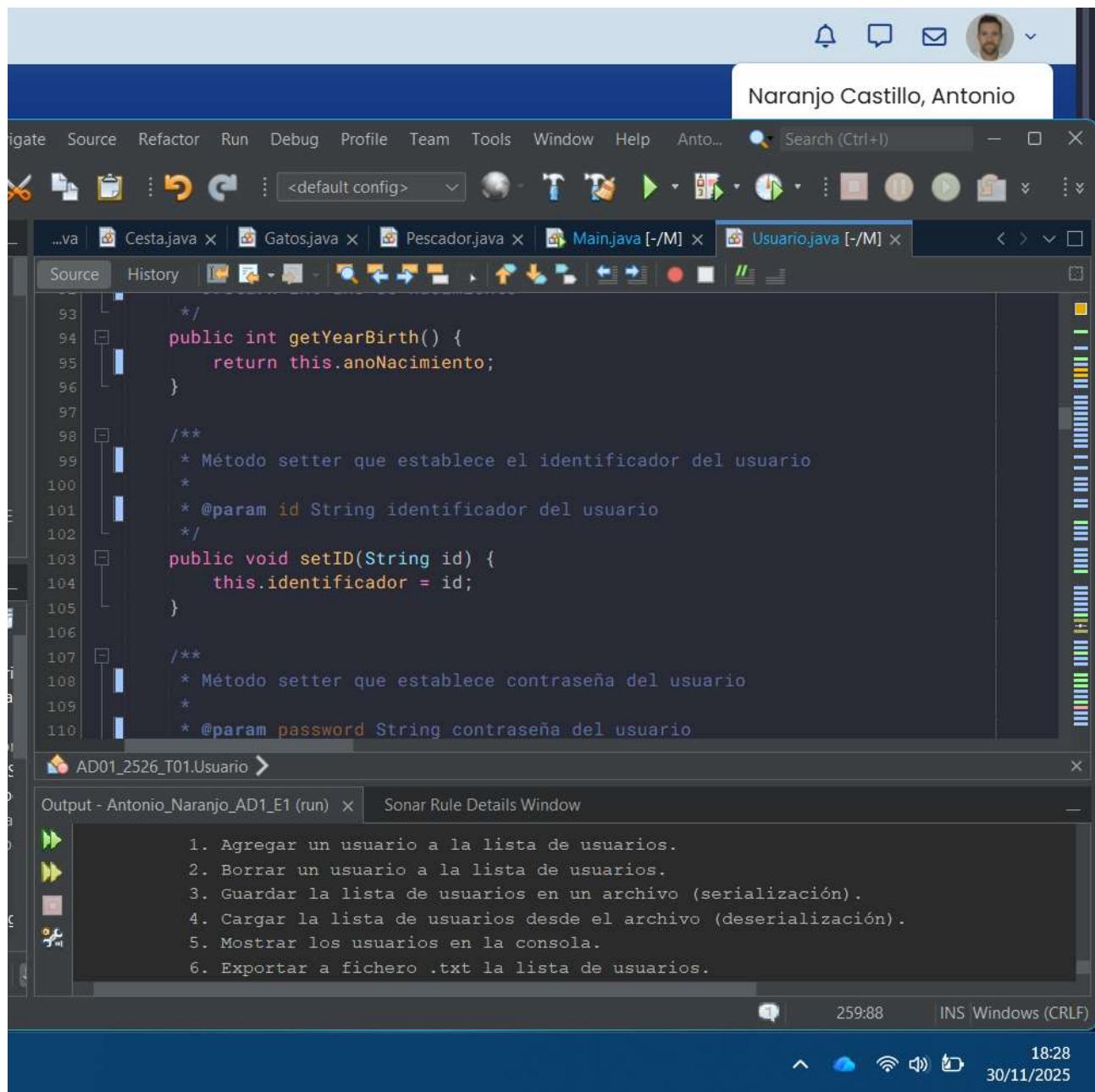


The screenshot shows an IDE interface with the following details:

- Title Bar:** Shows the user's name "Naranjo Castillo, Antonio".
- Toolbar:** Includes standard icons for file operations like Open, Save, and Print.
- MenuBar:** Lists options like Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and Auto... A search bar is also present.
- Project Explorer:** Shows files like Cesta.java, Gatos.java, Pescador.java, Main.java, and Usuario.java.
- Code Editor:** Displays the code for the Usuario.java class. The constructor is shown with annotations explaining its parameters:

```
46
47     /**
48      * Método constructor de la clase Usuario
49      *
50      * @param id String que representa la identificación de los usuarios
51      * @param password String que representa la contraseña de los usuarios
52      * @param address String que almacena la dirección de los usuarios
53      * @param yearBirth Dato numérico de tipo entero (int) que almacena el año
54      */
55     public Usuario(String id, String password, String address, int yearBirth)
56         this.identificador = id;
57         this.contrasena = password;
58         this.direccion = address;
59         this.anoNacimiento = yearBirth;
60     }
61
62     /**
```
- Output Window:** Shows a list of tasks:
 1. Agregar un usuario a la lista de usuarios.
 2. Borrar un usuario a la lista de usuarios.
 3. Guardar la lista de usuarios en un archivo (serialización).
 4. Cargar la lista de usuarios desde el archivo (deserialización).
 5. Mostrar los usuarios en la consola.
 6. Exportar a fichero .txt la lista de usuarios.
- System Tray:** Shows icons for battery, signal, and date/time (30/11/2025, 18:24).

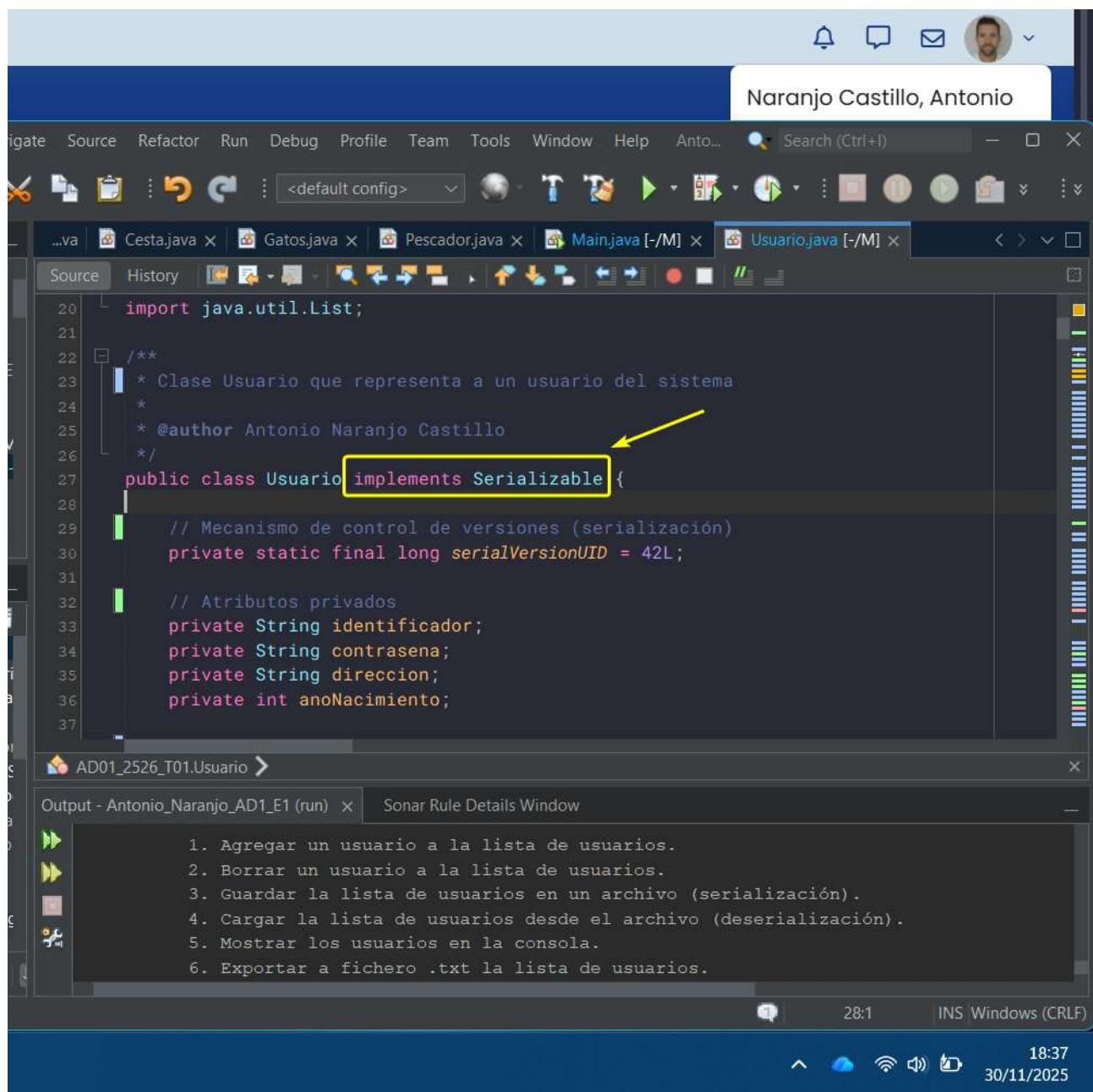
Se crea una clase Usuario implementando el método constructor que define los cuatro atributos solicitados en la tarea, identificador (de tipo String), contraseña (de tipo String), dirección (de tipo String) y año de nacimiento (de tipo int).



The screenshot shows an IDE interface with the following details:

- Title Bar:** Shows the user's name "Naranjo Castillo, Antonio".
- Menu Bar:** Includes options like Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and Auto... A search bar is also present.
- Toolbar:** Contains various icons for file operations, navigation, and tools.
- Project Explorer:** Shows files like Cestajava.java, Gatos.java, Pescador.java, Mainjava [-/M], and Usuario.java [-/M].
- Code Editor:** Displays the Java code for the `Usuario.java` class. The code includes methods for getting and setting the birth year, ID, and password, along with their corresponding Javadoc comments.
- Output Window:** Titled "AD01_2526_T01.Usuario", it lists the following steps:
 1. Agregar un usuario a la lista de usuarios.
 2. Borrar un usuario a la lista de usuarios.
 3. Guardar la lista de usuarios en un archivo (serialización).
 4. Cargar la lista de usuarios desde el archivo (deserialización).
 5. Mostrar los usuarios en la consola.
 6. Exportar a fichero .txt la lista de usuarios.
- System Tray:** Shows icons for battery, signal, and date/time (30/11/2025, 18:28).

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



```
20 import java.util.List;
21
22 /**
23  * Clase Usuario que representa a un usuario del sistema
24  *
25  * @author Antonio Naranjo Castillo
26 */
27 public class Usuario implements Serializable {implements Serializable
28
29     // Mecanismo de control de versiones (serialización)
30     private static final long serialVersionUID = 42L;
31
32     // Atributos privados
33     private String identificador;
34     private String contrasena;
35     private String direccion;
36     private int anoNacimiento;
37 }
```

Output - Antonio_Naranjo_AD1_E1 (run) x Sonar Rule Details Window

1. Agregar un usuario a la lista de usuarios.
2. Borrar un usuario a la lista de usuarios.
3. Guardar la lista de usuarios en un archivo (serialización).
4. Cargar la lista de usuarios desde el archivo (deserialización).
5. Mostrar los usuarios en la consola.
6. Exportar a fichero .txt la lista de usuarios.

28:1 INS Windows (CRLF)
18:37 30/11/2025

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.

2. Implementación del menú

The screenshot shows an IDE interface with the following details:

- Title Bar:** Shows the user's name "Naranjo Castillo, Antonio".
- Toolbar:** Includes standard icons for file operations like Open, Save, and Print.
- MenuBar:** Lists "File", "Source", "Refactor", "Run", "Debug", "Profile", "Team", "Tools", "Window", "Help".
- Project Bar:** Displays project files: "...va", "Cestajava.java", "Gatosjava.java", "Pescador.java", "Main.java [-M]", and "Usuario.java [-M]".
- Code Editor:** Displays Java code for a menu system. The code uses a do-while loop to print a menu and read user input. It includes options for adding, removing, serializing, deserializing, displaying, and exporting user data.
- Output Window:** Shows the menu options listed as 1 through 6.
- System Tray:** Shows the date and time as 30/11/2025 18:42.

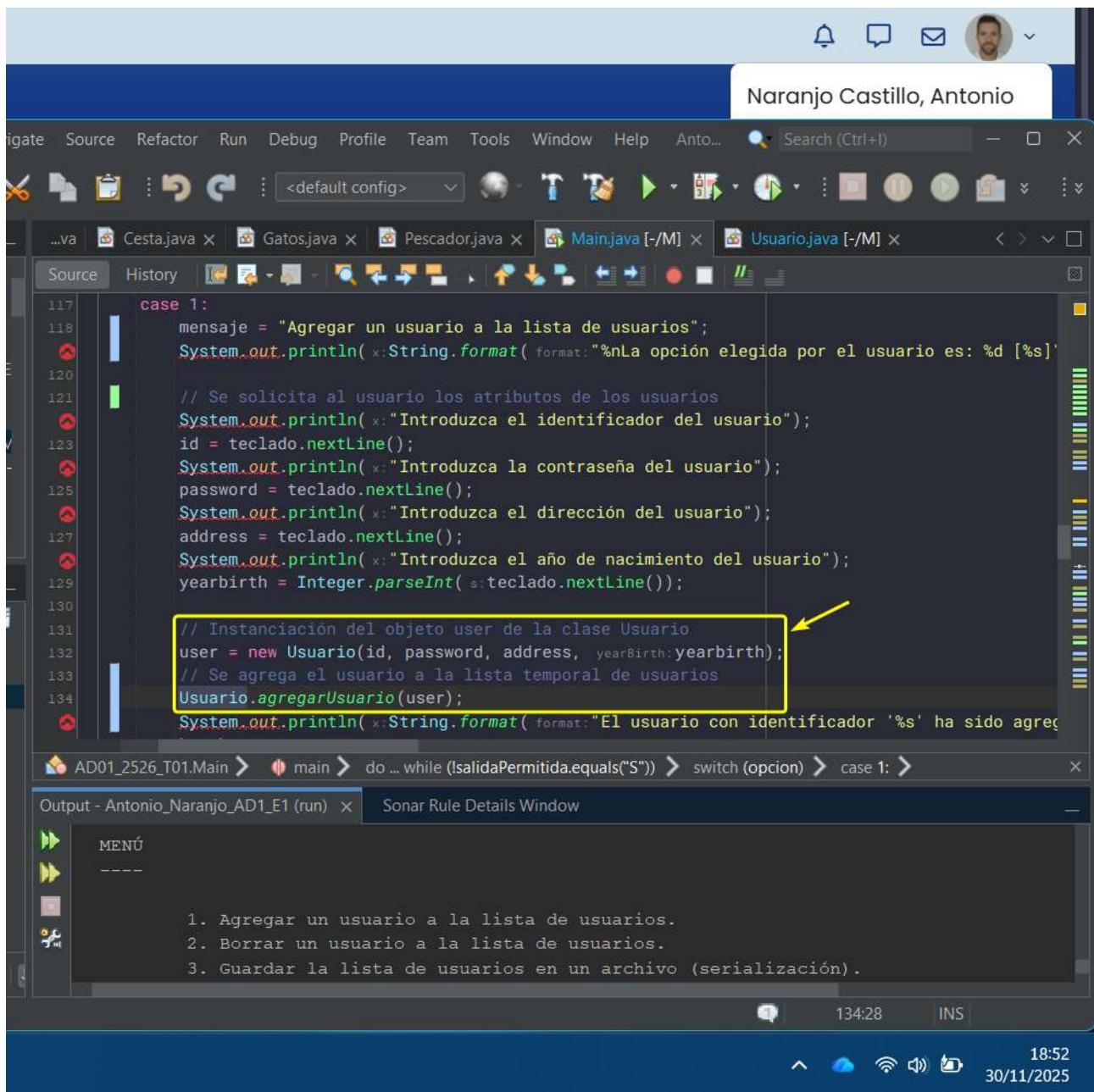
```
78 do {  
79     // Menú  
80     System.out.println( x: "\n----");  
81     System.out.println( x: "MENÚ");  
82     System.out.println( x: "----\n");  
83     System.out.println( x: "\t1. Agregar un usuario a la lista de usuarios.");  
84     System.out.println( x: "\t2. Borrar un usuario a la lista de usuarios.");  
85     System.out.println( x: "\t3. Guardar la lista de usuarios en un archivo (serialización).");  
86     System.out.println( x: "\t4. Cargar la lista de usuarios desde el archivo (deserialización).");  
87     System.out.println( x: "\t5. Mostrar los usuarios en la consola.");  
88     System.out.println( x: "\t6. Exportar a fichero .txt la lista de usuarios.\n");  
89     System.out.print( s:"\t\t--> Opción seleccionada por el usuario: ");  
90  
91     // Recoger opción seleccionada por el usuario  
92     opcion = Integer.parseInt( s:teclado.nextLine().trim());  
93  
94 } while( opcion != 0 );  
95  
96 // Salir del bucle  
97 System.out.println( x: "-----");  
98  
99 }
```

Output - Antonio_Naranjo_AD1_E1 (run) x Sonar Rule Details Window

1. Agregar un usuario a la lista de usuarios.
2. Borrar un usuario a la lista de usuarios.
3. Guardar la lista de usuarios en un archivo (serialización).
4. Cargar la lista de usuarios desde el archivo (deserialización).
5. Mostrar los usuarios en la consola.
6. Exportar a fichero .txt la lista de usuarios.

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.

3. Agregar un usuario a la lista de usuarios



The screenshot shows an IDE interface with the following details:

- Title Bar:** Shows the user's name: "Naranjo Castillo, Antonio".
- Toolbar:** Includes standard icons for file operations like Open, Save, and Print.
- Project Explorer:** Lists several Java files: ...va, Cestajava, Gatosjava, Pescador.java, Main.java, and Usuario.java.
- Code Editor:** Displays Java code. A specific section of the code is highlighted with a yellow box and a yellow arrow pointing to it. The highlighted code is:

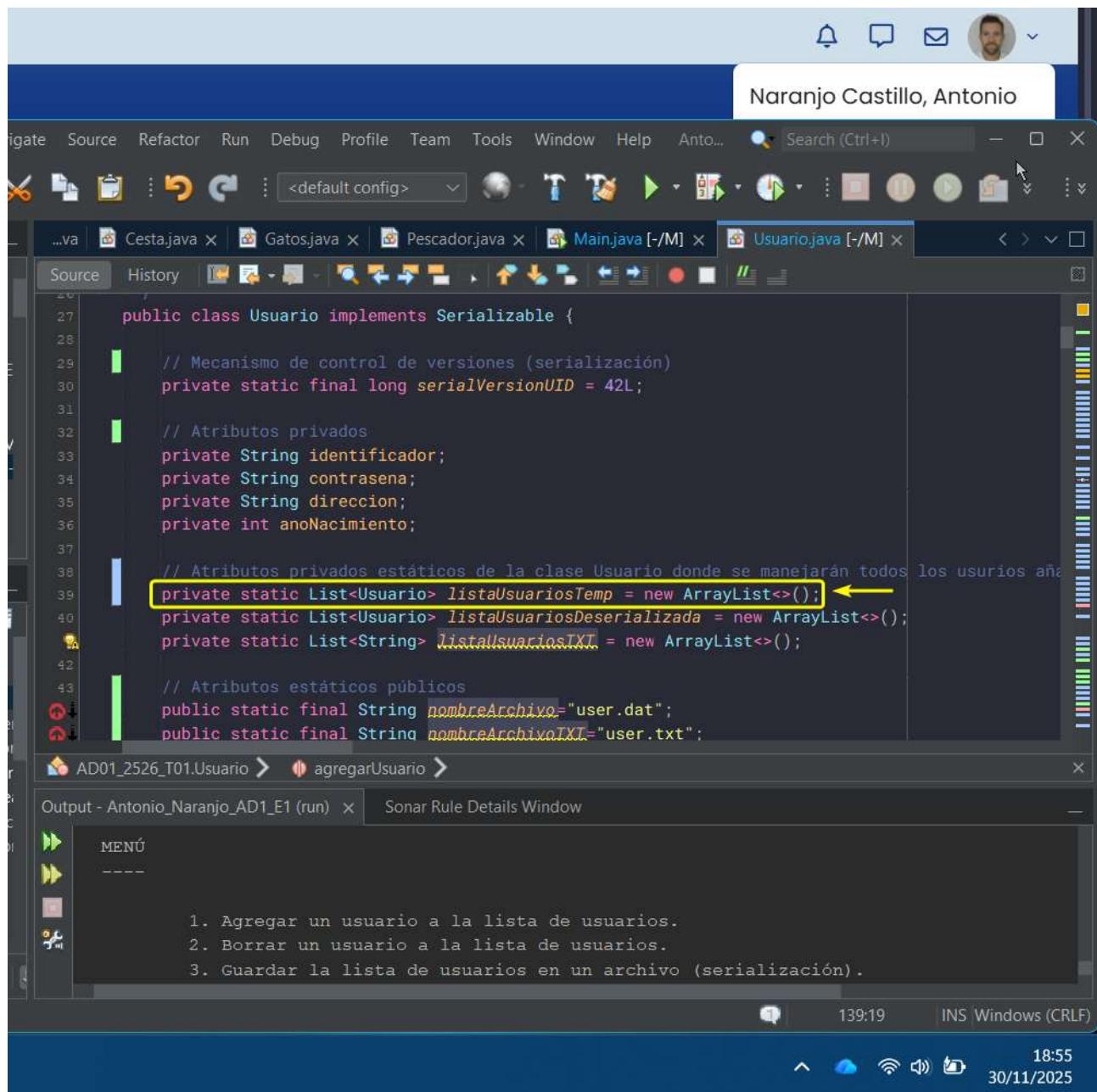

```

case 1:
    mensaje = "Agregar un usuario a la lista de usuarios";
    System.out.println(String.format("%nLa opción elegida por el usuario es: %d [%s]"));

    // Se solicita al usuario los atributos de los usuarios
    System.out.println("Introduzca el identificador del usuario");
    id = teclado.nextLine();
    System.out.println("Introduzca la contraseña del usuario");
    password = teclado.nextLine();
    System.out.println("Introduzca la dirección del usuario");
    address = teclado.nextLine();
    System.out.println("Introduzca el año de nacimiento del usuario");
    yearbirth = Integer.parseInt(teclado.nextLine());

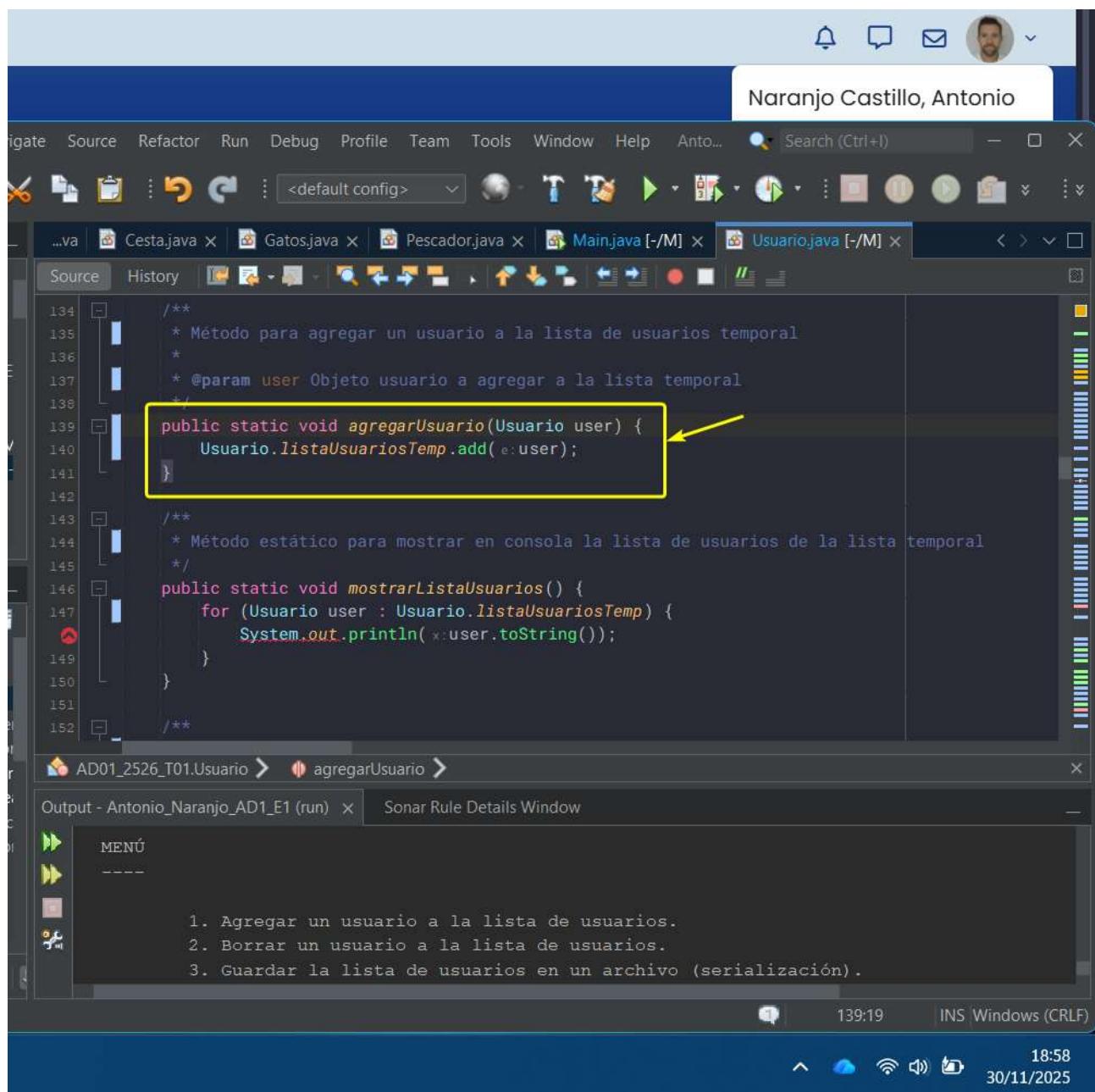
    // Instanciación del objeto user de la clase Usuario
    user = new Usuario(id, password, address, yearBirth:yearbirth);
    // Se agrega el usuario a la lista temporal de usuarios
    Usuario.agregarUsuario(user);
    System.out.println(String.format("El usuario con identificador '%s' ha sido agregado"));
      
```
- Output Window:** Shows the command line history and the output of the application execution. The output includes the menu options and the confirmation message for adding a user.
- System Tray:** Shows the date and time (30/11/2025, 18:52) and some system icons.

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 instancia de dicho objeto usuario, así como la aplicación del método estático agregarUsuario() de la clase Usuario aportando como argumento el objeto usuario recientemente instanciado.



```
public class Usuario implements Serializable {  
    // Mecanismo de control de versiones (serialización)  
    private static final long serialVersionUID = 42L;  
  
    // Atributos privados  
    private String identificador;  
    private String contrasena;  
    private String direccion;  
    private int anoNacimiento;  
  
    // Atributos privados estáticos de la clase Usuario donde se manejarán todos los usuarios añ  
    private static List<Usuario> listaUsuariosTemp = new ArrayList<>();  
    private static List<Usuario> listaUsuariosDeserializada = new ArrayList<>();  
    private static List<String> listaUsuariosTXT = new ArrayList<>();  
  
    // Atributos estáticos públicos  
    public static final String nombreArchivo="user.dat";  
    public static final String nombreArchivTXT="user.txt";  
}
```

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.



```
134 /**
135 * Método para agregar un usuario a la lista de usuarios temporal
136 *
137 * @param user Objeto usuario a agregar a la lista temporal
138 */
139 public static void agregarUsuario(Usuario user) {
140     Usuario.listaUsuariosTemp.add(e:user);
141 }
142 /**
143 * Método estático para mostrar en consola la lista de usuarios de la lista temporal
144 */
145 public static void mostrarListaUsuarios() {
146     for (Usuario user : Usuario.listaUsuariosTemp) {
147         System.out.println(user.toString());
148     }
149 }
150 /**
151 */
152 
```

AD01_2526_T01.Usuario > ● agregarUsuario >

Output - Antonio_Naranjo_AD1_E1 (run) x Sonar Rule Details Window

MENÚ

1. Agregar un usuario a la lista de usuarios.
2. Borrar un usuario a la lista de usuarios.
3. Guardar la lista de usuarios en un archivo (serialización).

139:19 INS Windows (CRLF)

18:58 30/11/2025

En esta imagen se presenta el método estático `agregarUsuario()` para incorporar a la lista temporal indicada con anterioridad a cada uno de los usuarios que se vayan dando de alta.

4. Borrar un usuario de la lista de usuarios

The screenshot shows an IDE interface with the following details:

- Title Bar:** Shows the name "Naranjo Castillo, Antonio".
- Toolbar:** Standard IDE toolbar with icons for file operations, search, and other functions.
- Project Explorer:** Shows files like Cestajava.java, Gatos.java, Pescador.java, Main.java, and Usuario.java.
- Code Editor:** Displays Java code. The relevant part is:

```
134     Usuario.agregarUsuario(user);
135     System.out.println(String.format("El usuario con identificador '%s' ha sido agregado"));
136     break;
137
138     // Borrar un usuario a la lista de usuarios
139     case 2:
140         mensaje = "Borrar un usuario a la lista de usuarios";
141         System.out.println(String.format("%nLa opción elegida por el usuario es: %d [%s]"));
142         // Se solicita al usuario el identificador del usuario a eliminar
143         System.out.println("Introduzca el identificador del usuario a eliminar");
144         id = teclado.nextLine(); 1
145         // Se llama al método estático de la clase Usuario para eliminar el usuario solicitado
146         Usuario.borrarUsuario(identificador:id); 2
147         System.out.println(String.format("El usuario con identificador '%s' ha sido borrado"));
148
149         break;
150
151     // Guardar la lista de usuarios en un archivo (serialización)
152     case 3:
```
- Output Window:** Shows the menu options:

```
MENÚ
-----
1. Agregar un usuario a la lista de usuarios.
2. Borrar un usuario a la lista de usuarios.
3. Guardar la lista de usuarios en un archivo (serialización).
```
- System Tray:** Shows the date and time (30/11/2025, 19:01), battery level, and network status.

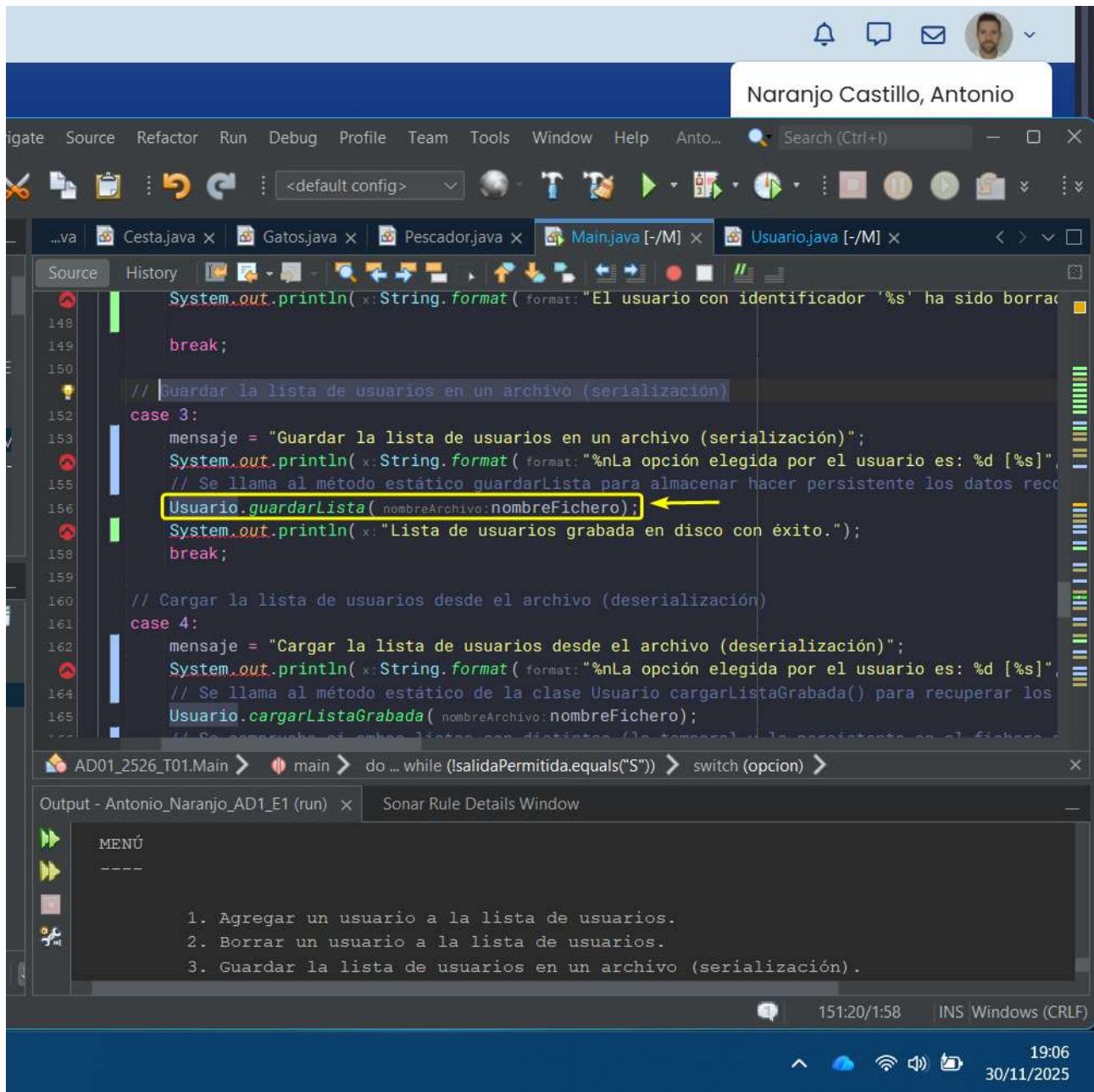
Para borrar un usuario de la lista temporal de usuarios, el usuario del programa deberá seleccionar la opción 2, acto seguido, el programa solicitará la identificación del usuario a eliminar para ejecutar el método estático de la clase Usuario borrarUsuario() pasando como argumento el atributo identificador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

The screenshot shows an IDE interface with the following details:

- Title Bar:** Shows the name "Naranjo Castillo, Antonio".
- Toolbar:** Includes standard icons for file operations like Open, Save, and Print.
- Menu Bar:** Contains options like Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and Auto... A search bar is also present.
- Project Explorer:** Shows files like ...va, Cestajava.x, Gatosjava.x, Pescador.java.x, Mainjava [-/M].x, and Usuario.java [-/M].x.
- Code Editor:** Displays Java code for the `borrarUsuario` method in the `Usuario` class. The code uses an iterator to safely remove users from a temporary list.
- Status Bar:** Shows the time as 13:19 and the operating system as Windows (CRLF).
- Bottom Status Bar:** Shows the date as 30/11/2025 and the time as 19:03.

Se presenta el método `borrarUsuario` estático de la clase `Usuario`, haciendo uso de un iterador para garantizar borrar el usuario de manera segura de la lista temporal.

5. Guardar la lista de usuarios en un archivo (serialización)



The screenshot shows an IDE interface with the following details:

- Title Bar:** Shows the user's name "Naranjo Castillo, Antonio".
- Toolbar:** Standard IDE tools like file operations, search, and navigation.
- Project Explorer:** Lists projects: ...va, Cestajava, Gatosjava, Pescador.java, Main.java [-/M] (selected), and Usuario.java [-/M].
- Code Editor:** Displays Java code in the Main.java file. The code handles user input and calls the `Usuario.guardarLista()` method. A yellow arrow points to this method call.
- Output Window:** Shows the current run configuration: AD01_2526_T01.Main > main > do ... while (!salidaPermitida.equals("S")) > switch (opcion) >.
- Output Sub-Window:** Shows the menu options:
 - MENÚ
 -
 - 1. Agregar un usuario a la lista de usuarios.
 - 2. Borrar un usuario a la lista de usuarios.
 - 3. Guardar la lista de usuarios en un archivo (serialización).
- System Tray:** Shows the date and time (15/11/2025, 19:06), battery level, and signal strength.

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.

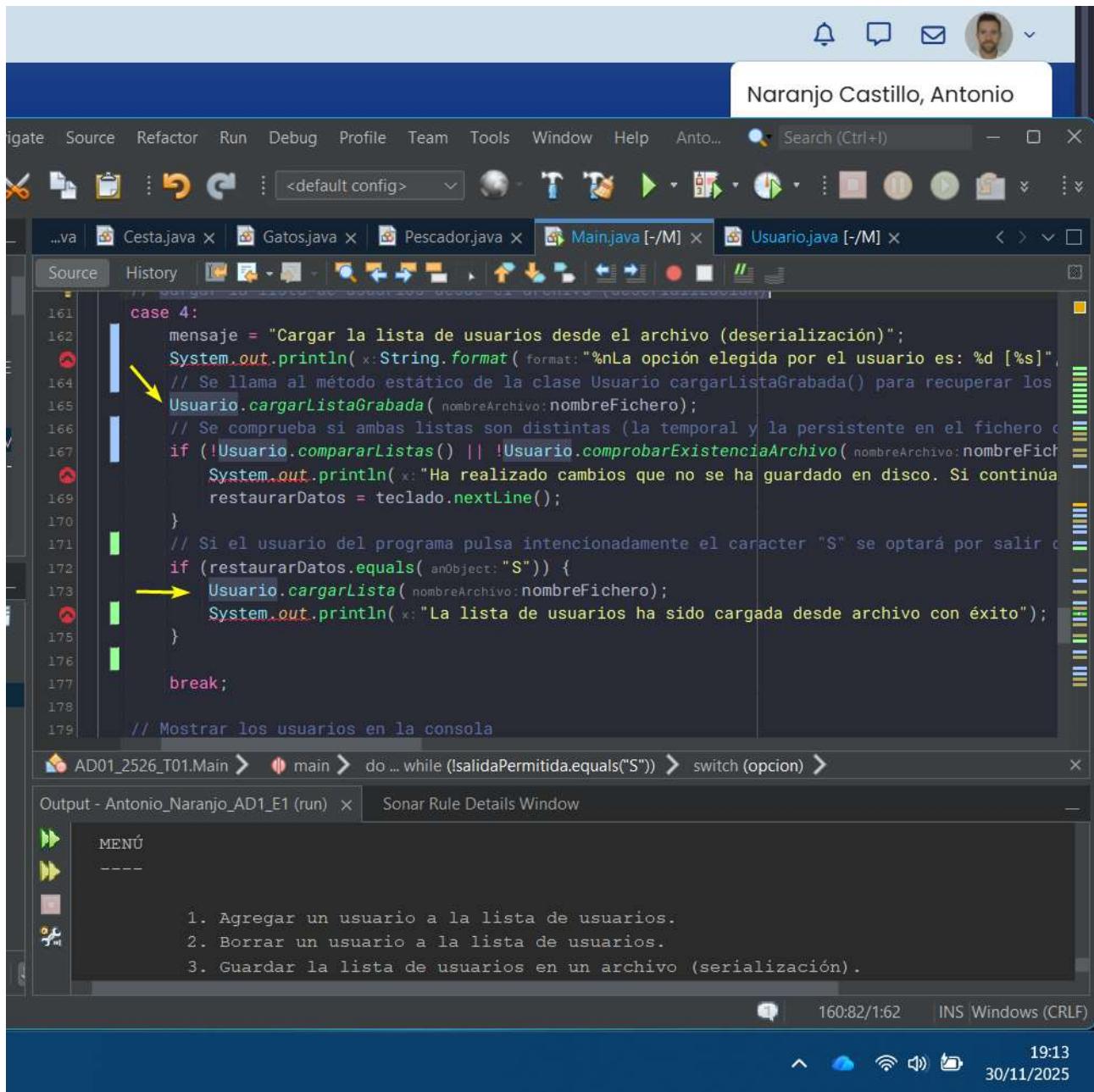
Se ejecuta el método estático de la clase `Usuario.guardarLista()` aportando como argumento el nombre del archivo en cuestión.

The screenshot shows an IDE interface with the following details:

- Title Bar:** Shows the name "Naranjo Castillo, Antonio".
- Toolbar:** Includes standard icons for file operations like Open, Save, and Run.
- Menu Bar:** Contains options like Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and Auto... A search bar is also present.
- Code Editor:** Displays Java code for a static method `guardarLista` in the `Usuario.java` class. The code uses `FileOutputStream` and `ObjectOutputStream` to serialize a list of users.
- Output Window:** Shows the command `AD01_2526_T01.Usuario > agregarUsuario >` and the output "MENÚ" followed by a list:
 1. Agregar un usuario a la lista de usuarios.
 2. Borrar un usuario a la lista de usuarios.
 3. Guardar la lista de usuarios en un archivo (serialización).
- System Tray:** Shows the date and time as 30/11/2025 at 19:08.

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).

6. Cargar la lista de usuarios desde el archivo (deserialización)



```

161
162     case 4:
163         mensaje = "Cargar la lista de usuarios desde el archivo (deserialización)";
164         System.out.println(String.format("%nLa opción elegida por el usuario es: %d [%s]", opcion));
165         // Se llama al método estático de la clase Usuario cargarListaGrabada() para recuperar los
166         // datos persistentes almacenados en el fichero
167         Usuario.cargarListaGrabada(nombreArchivo,nombreFichero);
168         // Se comprueba si ambas listas son distintas (la temporal y la persistente en el fichero)
169         if (!Usuario.compararListas() || !Usuario.comprobarExistenciaArchivo(nombreArchivo,nombreFichero))
170             System.out.println("Ha realizado cambios que no se ha guardado en disco. Si continúa");
171             restaurarDatos = teclado.nextLine();
172         }
173         // Si el usuario del programa pulsa intencionadamente el carácter "S" se optará por salir
174         if (restaurarDatos.equals("S")) {
175             Usuario.cargarLista(nombreArchivo,nombreFichero);
176             System.out.println("La lista de usuarios ha sido cargada desde archivo con éxito");
177         }
178
179         break;
180
181         // Mostrar los usuarios en la consola

```

The screenshot shows an IDE interface with the following details:

- Title Bar:** Shows the user's name: "Naranjo Castillo, Antonio".
- Toolbar:** Standard IDE tools like file operations, search, and navigation.
- Project Bar:** Lists projects: "...va", "Cestajava x", "Gatosjava x", "Pescador.java x", "Main.java [-/M] x", and "Usuario.java [-/M] x".
- Code Editor:** Displays the Java code for the `Main` class. A yellow arrow points to the line `Usuario.cargarListaGrabada(nombreArchivo,nombreFichero);`. Another yellow arrow points to the line `Usuario.cargarLista(nombreArchivo,nombreFichero);`.
- Output Window:** Shows the execution path: "AD01_2526_T01.Main > main > do ... while (!salidaPermitida.equals("S")) > switch (opcion) >". Below it, the output window displays a menu with options 1, 2, and 3.
- System Tray:** Shows the date and time: "16/08/16 19:13" and the date: "30/11/2025".

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.

The screenshot shows an IDE interface with the following details:

- Top Bar:** Shows navigation options like Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and Auto... along with a search bar labeled "Search (Ctrl+I)".
- Title Bar:** Displays the name "Naranjo Castillo, Antonio".
- Toolbars:** Includes standard icons for file operations, code navigation, and project management.
- Code Editor:** Displays Java code for a class named "Usuario.java". The code implements a static method "cargarListaGrabada" which reads a file and deserializes it into a list of users. It uses `FileInputStream` and `ObjectInputStream` to achieve this.
- Output Window:** Shows a terminal window titled "AD01_2526_T01.Usuario > agregarUsuario >". It displays the following menu:

```
MENÚ
-----
1. Agregar un usuario a la lista de usuarios.
2. Borrar un usuario a la lista de usuarios.
3. Guardar la lista de usuarios en un archivo (serialización).
```
- System Tray:** Shows the date and time as "30/11/2025 19:24".

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.

The screenshot shows an IDE interface with the following details:

- Title Bar:** Naranjo Castillo, Antonio
- Menu Bar:** Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, Auto...
- Toolbar:** Includes icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run.
- Project Explorer:** Shows files like ...va, Cestajava.x, Gatosjava.x, Pescador.java.x, Mainjava [-/M].x, and Usuario.java [-/M].x.
- Code Editor:** Displays Java code for a static method `compararListas()`. The code compares two lists of users based on their attributes: identifier, password, address, and birth year.
- Output Window:** Shows a menu with options:
 - MENÚ
 -
 - 1. Agregar un usuario a la lista de usuarios.
 - 2. Borrar un usuario a la lista de usuarios.
 - 3. Guardar la lista de usuarios en un archivo (serialización).
- System Tray:** Shows icons for battery, signal, and date/time (30/11/2025) at the bottom right.

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.

The screenshot shows an IDE interface with the following details:

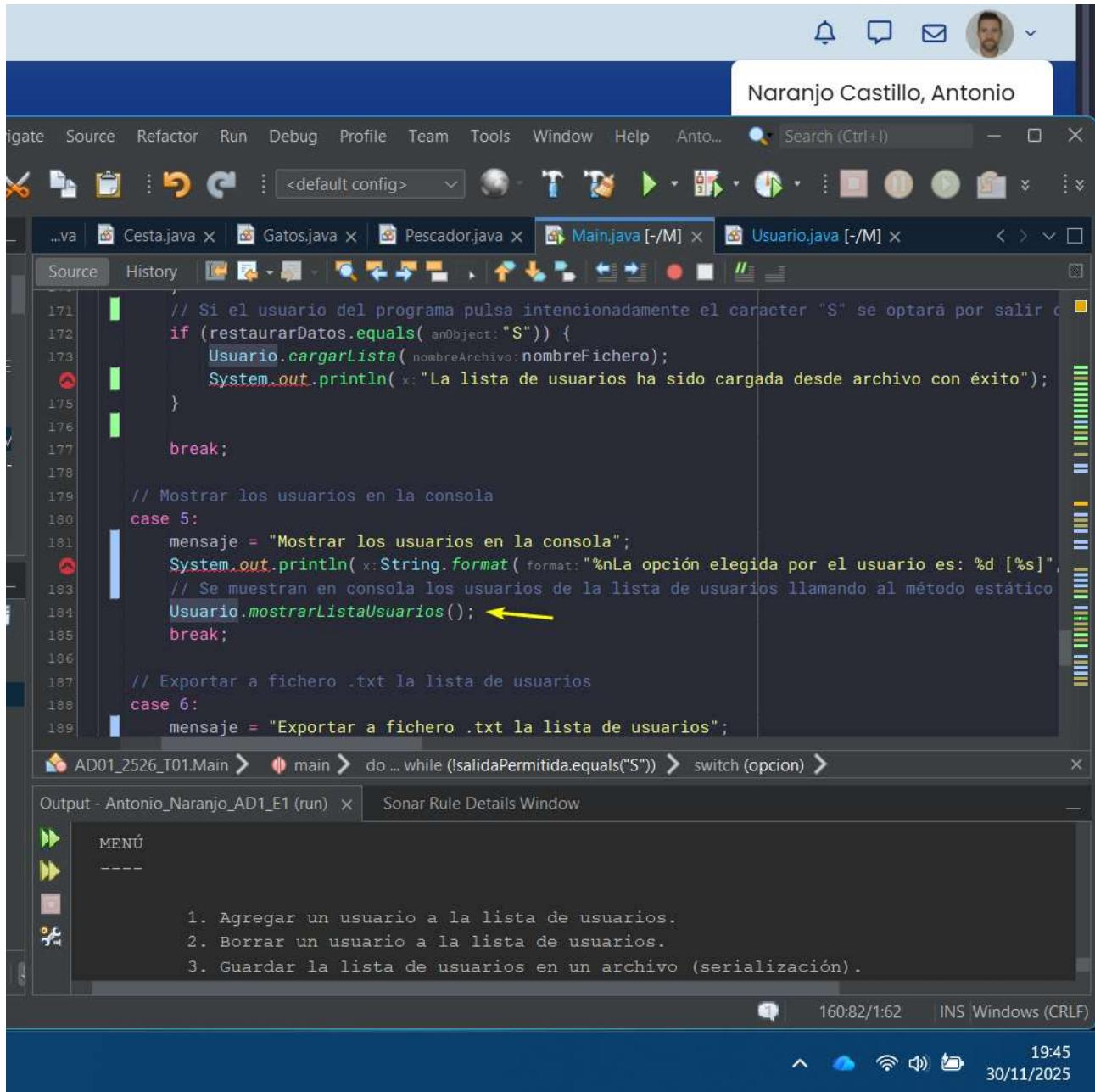
- Top Bar:** Shows navigation links like Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and Auto...; a search bar labeled "Search (Ctrl+I)"; and user information "Naranjo Castillo, Antonio".
- Toolbar:** Includes icons for file operations (New, Open, Save, Find, etc.), code navigation, and build/run.
- Project Explorer:** Shows files like ...va, Cestajava.x, Gatosjava.x, Pescador.java.x, Mainjava [-/M].x, and Usuario.java [-/M].x.
- Code Editor:** Displays the `cargarLista` method from the `Usuario.java` file. The code reads bytes from a file and deserializes them into a list of `Usuario` objects.
- Output Window:** Shows the command `AD01_2526_T01.Usuario > agregarUsuario >`. Below it, the "Output - Antonio_Naranjo_AD1_E1 (run)" tab displays a menu:

 - MENÚ
 -
 - 1. Agregar un usuario a la lista de usuarios.
 - 2. Borrar un usuario a la lista de usuarios.
 - 3. Guardar la lista de usuarios en un archivo (serialización).

- System Tray:** Shows icons for battery, signal, and date/time (19:35, 30/11/2025).

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.

7. Mostrar los usuarios en consola



The screenshot shows an IDE interface with the following details:

- Title Bar:** Shows the user's name "Naranjo Castillo, Antonio".
- Toolbar:** Standard IDE tools like file operations, search, and navigation.
- Project Explorer:** Lists files: ...va, Cestajava, Gatosjava, Pescador.java, Main.java [-/M] (selected), and Usuario.java [-/M].
- Code Editor:** Displays Java code. A yellow arrow points to the line `Usuario.mostrarListaUsuarios();` in the `case 5` block.
- Output Window:** Shows the menu options:
 - MENÚ
 -
 - 1. Agregar un usuario a la lista de usuarios.
 - 2. Borrar un usuario a la lista de usuarios.
 - 3. Guardar la lista de usuarios en un archivo (serialización).
- System Tray:** Shows the date and time (30/11/2025, 19:45), battery level, and network status.

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.

The screenshot shows an IDE interface with the following details:

- Title Bar:** Shows the user's name "Naranjo Castillo, Antonio".
- Toolbar:** Standard IDE tools like file operations, search, and run.
- Project Bar:** Lists projects: Cestajava, Gatosjava, Pescador.java, Mainjava [-/M], and Usuario.java [-/M].
- Code Editor:** Displays the `Usuario.java` source code. The code includes methods `mostrarListaUsuarios()` and `toString()`. The `toString()` method uses `String.format` to build a string representing the user object.
- Output Window:** Shows a menu with options:
 1. Agregar un usuario a la lista de usuarios.
 2. Borrar un usuario a la lista de usuarios.
 3. Guardar la lista de usuarios en un archivo (serialización).
- System Tray:** Shows the date and time as 30/11/2025 19:48.

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.

8. Exporta a fichero .txt la lista de usuarios

The screenshot shows a Java application interface with several tabs at the top: Cesta.java, Gatos.java, Pescador.java, Mainjava [-/M], and Usuario.java [-/M]. The Mainjava tab is active, displaying the following code:

```
// Exportar a fichero .txt la lista de usuarios
case 6:
    mensaje = "Exportar a fichero .txt la lista de usuarios";
    System.out.println( x:String.format( format:"%nLa opción elegida por el usuario es: %d [%s]");
    System.out.println( x:"Introduzca el nombre del archivo de texto txt para exportar la lista");
    nombreFicheroTXT = teclado.nextLine();
    // Si se pulsa ENTER se tomará el nombre del archivo por defecto, en caso contrario el usuario
    if (nombreFicheroTXT.isEmpty()) {
        nombreFicheroTXT = Usuario.nombreArchivoTXT;
    }
    // Se escribe el contenido de la lista de usuario en un archivo TXT
    Usuario.escribirTXT( nombreArchivo:nombreFicheroTXT); ← Yellow arrow here
    System.out.println( x:"Archivo de texto exportado correctamente.");
    break;

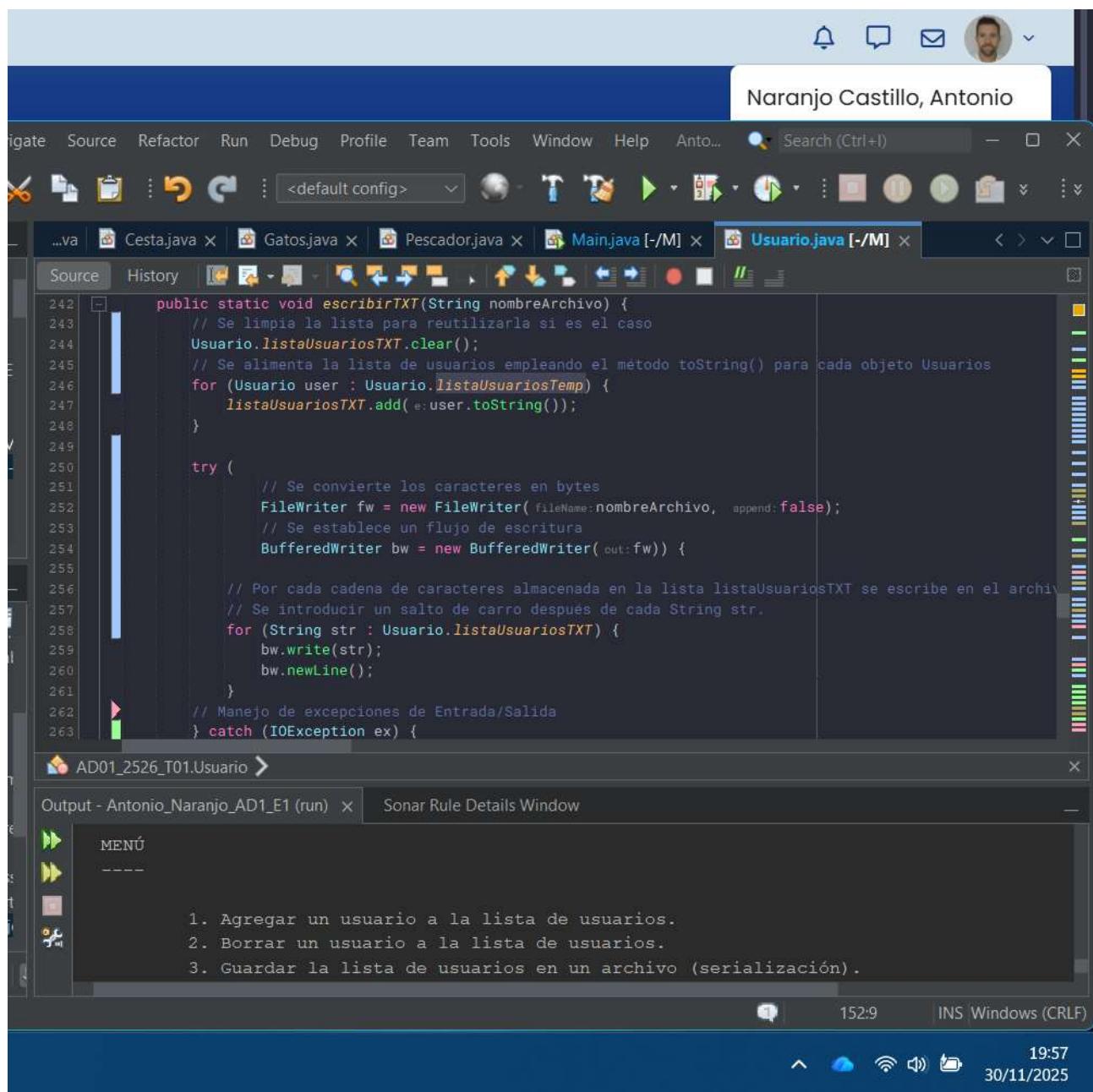
// Mostrar un mensaje de error en la consola en caso de introducir una opción no permitida
default:
    if (opcion < 0 || opcion > 6) {
```

The code is annotated with a yellow arrow pointing to the line `Usuario.escribirTXT(nombreArchivo:nombreFicheroTXT);`. Below the code, the stack trace shows the execution path: AD01_2526_T01.Main > main > do ... while (salidaPermitida.equals("S")) > switch (opcion) >. The Output window shows the menu options:

- MENÚ
-
- 1. Agregar un usuario a la lista de usuarios.
- 2. Borrar un usuario a la lista de usuarios.
- 3. Guardar la lista de usuarios en un archivo (serialización).

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).



The screenshot shows an IDE interface with the following details:

- Title Bar:** Shows the user's name "Naranjo Castillo, Antonio".
- Toolbar:** Standard IDE tools like Open, Save, Run, Debug, etc.
- Project Explorer:** Lists projects: ...va, Cestajava, Gatosjava, Pescador.java, Mainjava [-/M], and Usuario.java [-/M].
- Code Editor:** Displays the `Usuario.java` source code. The code implements a static method `escribirTXT` that serializes a list of `Usuario` objects to a file named `nombreArchivo`. It uses `FileWriter` and `BufferedWriter` to handle the output, including a loop to write each string from the list followed by a new line.
- Output Window:** Shows the command `AD01_2526_T01.Usuario >` and the output of the run: "Output - Antonio_Naranjo_AD1_E1 (run) x Sonar Rule Details Window".
- Bottom Status Bar:** Shows the time as 152:9 and date as 30/11/2025.

```

public static void escribirTXT(String nombreArchivo) {
    // Se limpia la lista para reutilizarla si es el caso
    Usuario.listaUsuariosTXT.clear();
    // Se alimenta la lista de usuarios empleando el método toString() para cada objeto Usuarios
    for (Usuario user : Usuario.listaUsuariosTemp) {
        listaUsuariosTXT.add(user.toString());
    }
    try {
        // Se convierte los caracteres en bytes
        FileWriter fw = new FileWriter(fileName:nombreArchivo, append: false);
        // Se establece un flujo de escritura
        BufferedWriter bw = new BufferedWriter(out:fw)) {

            // Por cada cadena de caracteres almacenada en la lista listaUsuariosTXT se escribe en el archivo
            // Se introducir un salto de carro después de cada String str.
            for (String str : Usuario.listaUsuariosTXT) {
                bw.write(str);
                bw.newLine();
            }
        // Manejo de excepciones de Entrada/Salida
    } catch (IOException ex) {

```

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 incialmente, 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 métod `newLine()`.

9. Salir de la aplicación

```

98
99     // Salir de la aplicación
100    case 0:
101        mensaje = "Salir de la aplicación";
102        System.out.println(String.format("%nLa opción elegida por el usuario es: %d [%s]", opcion, mensaje));
103
104        // Se carga la lista de usuarios almacenada en el archivo del disco duro
105        Usuario.cargarListaGrabada(nombreArchivo:nombreFichero);
106        // Se comparan la lista temporal y la guardada en el archivo si son distintas o bien el archivo no existe
107        if (!Usuario.compararListas() || !Usuario.comprobarExistenciaArchivo(nombreArchivo:nombreFichero))
108            System.out.println("Ha habido cambios en el programa que todavía no se han guardado.");
109            salidaPermitida = teclado.nextLine().trim();
110        } else {
111            salidaPermitida = "S";
112            System.out.println("FIN del programa.");
113        }
114    break;
115
116    // Agregar un usuario a la lista de usuarios

```

Output - Antonio_Naranjo_AD1_E1 (run) x Sonar Rule Details Window

MENÚ

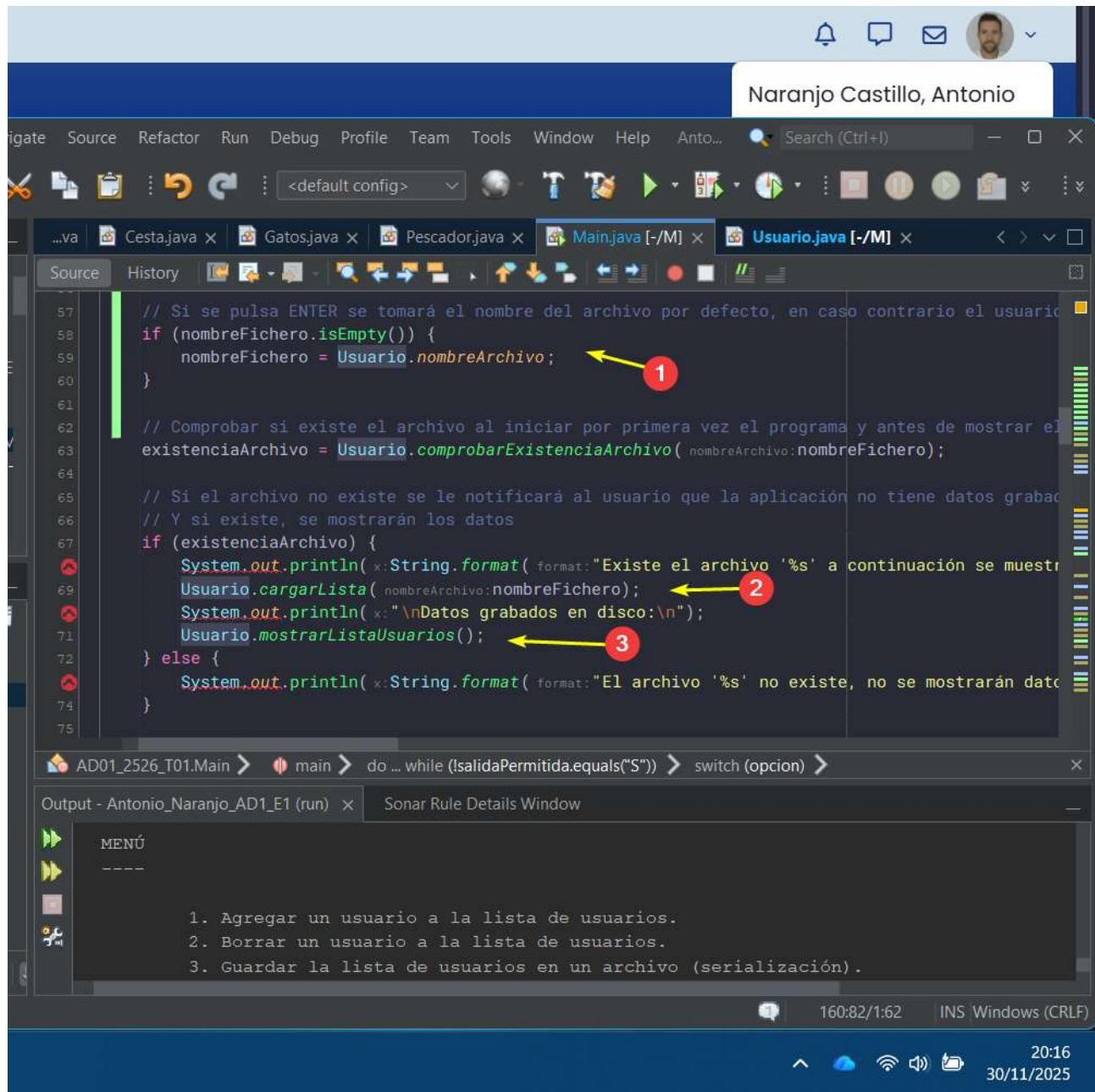
- 1. Agregar un usuario a la lista de usuarios.
- 2. Borrar un usuario a la lista de usuarios.
- 3. Guardar la lista de usuarios en un archivo (serialización).

160:82/1:62 INS Windows (CRLF) 20:07 30/11/2025

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 sin preguntar al usuario dado que los cambios están guardados en el fichero del disco duro.

10. ¿Existe el archivo ‘user.dat’?



The screenshot shows an IDE interface with the following details:

- Title Bar:** Naranjo Castillo, Antonio
- Toolbar:** Includes icons for file operations like Open, Save, and Run.
- Project Explorer:** Shows files: ...va, Cestajava.x, Gatosjava.x, Pescador.java.x, Main.java [-/M] x, and Usuario.java [-/M] x.
- Code Editor:** Displays Java code. Annotations with numbers 1, 2, and 3 point to specific lines:
 - Annotation 1 points to line 59: `nombreFichero = Usuario.nombreArchivo;` (highlighted in red).
 - Annotation 2 points to line 69: `Usuario.cargarLista(nombreArchivo:nombreFichero);` (highlighted in red).
 - Annotation 3 points to line 70: `Usuario.mostrarListaUsuarios();` (highlighted in red).
- Output Window:** Shows a menu with options:
 - MENÚ
 - 1. Agregar un usuario a la lista de usuarios.
 - 2. Borrar un usuario a la lista de usuarios.
 - 3. Guardar la lista de usuarios en un archivo (serialización).
- System Tray:** Shows icons for battery, signal, and date/time (30/11/2025, 20:16).

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.

11. Advertencia sobre datos no guardados antes de salir

```

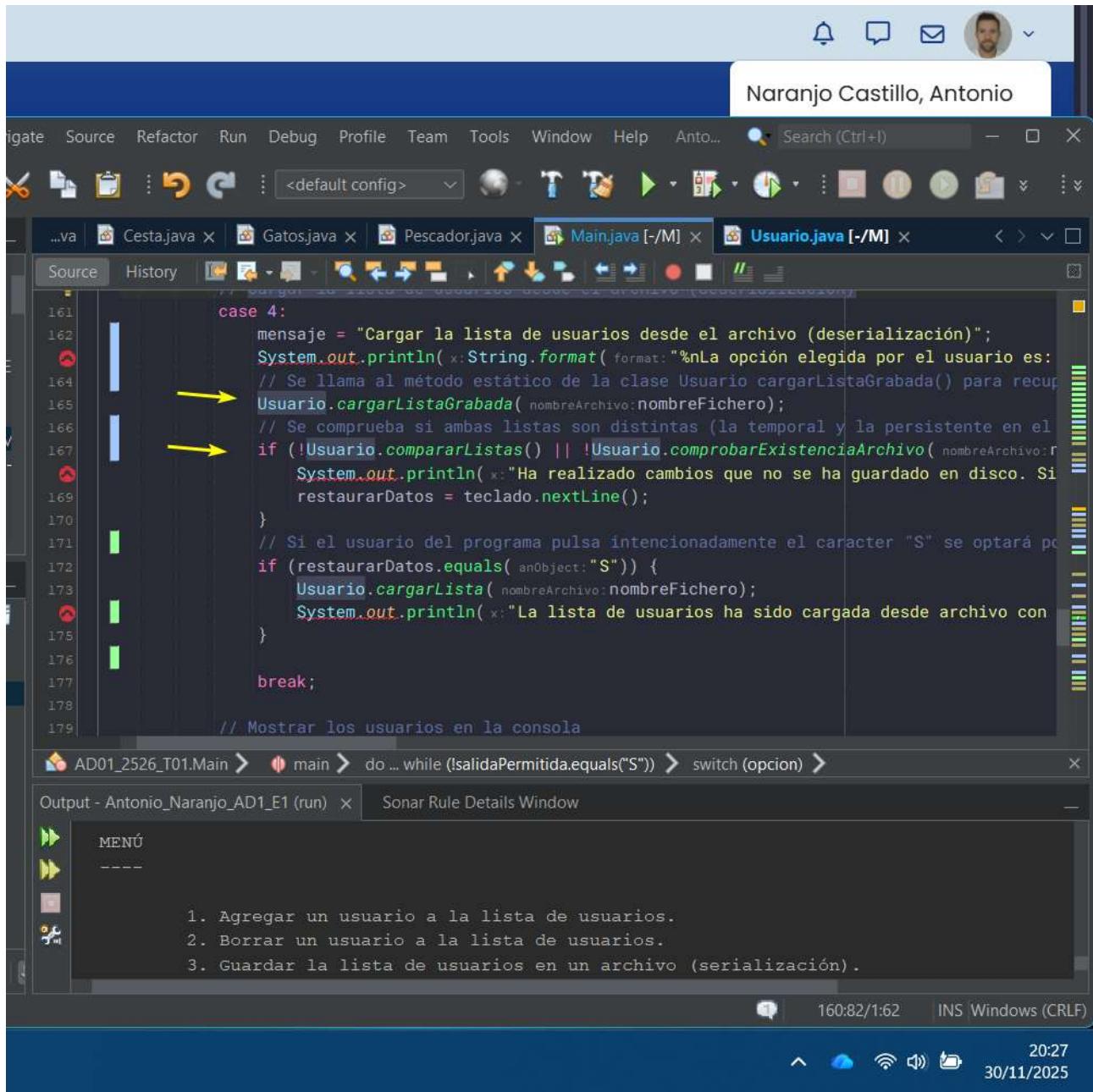
switch (opcion) {
    case 0:
        mensaje = "Salir de la aplicación";
        System.out.println(String.format("%nLa opción elegida por el usuario es:"));
        // Se carga la lista de usuarios almacenada en el archivo del disco duro
        Usuario.cargarListaGrabada(nombreArchivo:nombreFichero);
        // Se comparan la lista temporal y la guardada en el archivo si son distintas o bien
        if (!Usuario.compararListas() || !Usuario.comprobarExistenciaArchivo(nombreArchivo: nombreFichero))
            System.out.println("Ha habido cambios en el programa que todavía no se han guardado");
            salidaPermitida = teclado.nextLine().trim();
        else {
            salidaPermitida = "S";
            System.out.println("FIN del programa.");
        }
        break;
}

```

The screenshot shows the IntelliJ IDEA interface with the Main.java file open. The code above is part of a switch block for option 0. It prints a message, then loads a user list from a file using the `cargarListaGrabada` method. It then compares this with a temporary list using `compararListas`. If they differ or the file doesn't exist, it prints a message asking if the user wants to save changes before exiting. Yellow arrows point to the `cargarListaGrabada` and `compararListas` method calls.

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.

12. ¿Recuperar datos de disco?



The screenshot shows an IDE interface with the following details:

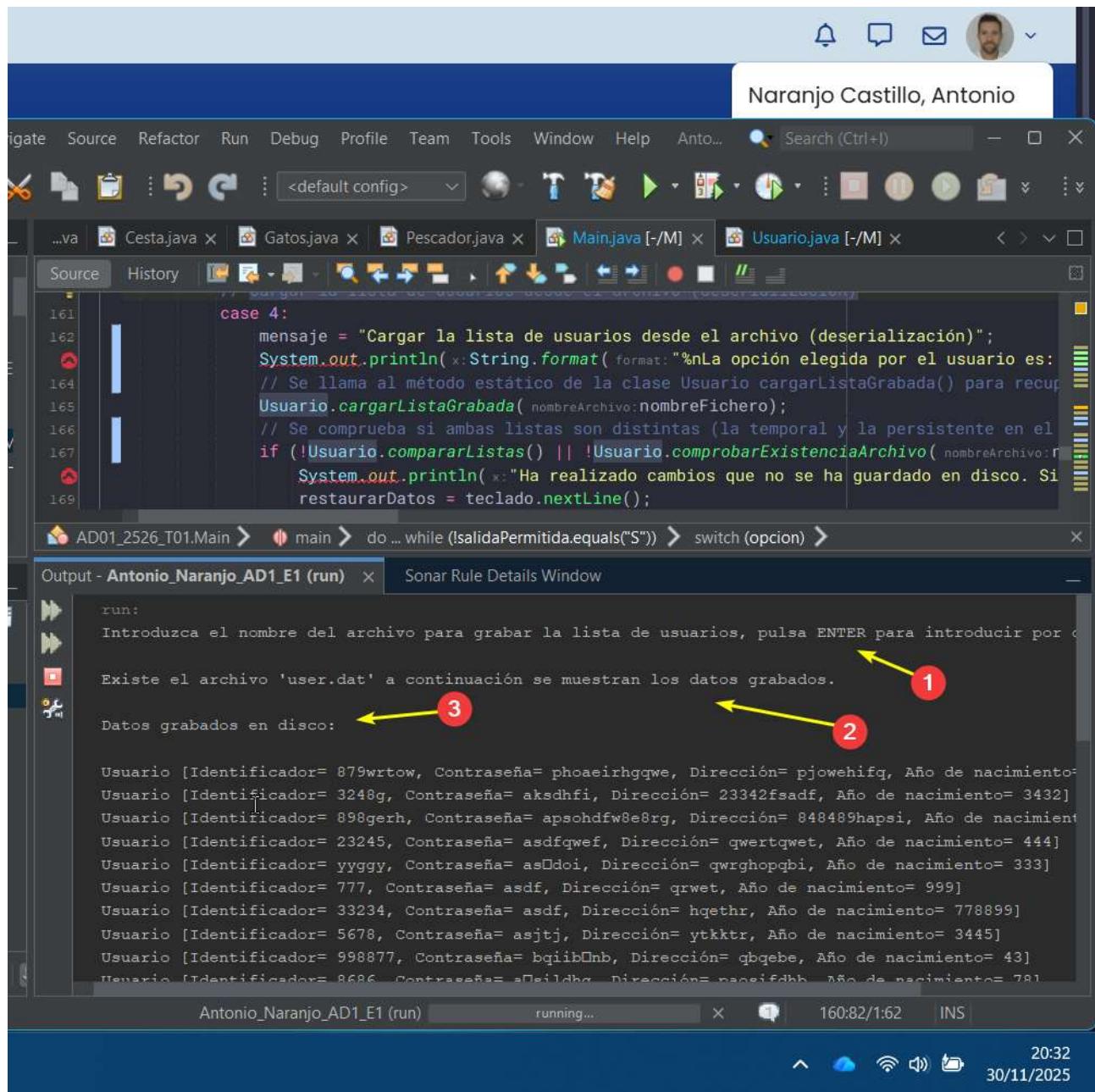
- Title Bar:** Shows the user's name: "Naranjo Castillo, Antonio".
- Toolbar:** Includes standard icons for file operations like Open, Save, Cut, Copy, Paste, and Run.
- Project Explorer:** Lists several Java files: "...va", Cestajava.java, Gatos.java, Pescador.java, Main.java [-/M], and Usuario.java [-/M].
- Code Editor:** Displays Java code for handling user input and file operations. Two yellow arrows point to the line where the method `Usuario.cargarListaGrabada()` is called. The code also includes logic to compare temporary and persistent lists and handle changes made by the user.
- Toolbars and Panels:** Includes a Source History toolbar, a Sonar Rule Details Window panel, and a status bar at the bottom showing the date and time.

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.

13. Anexo. Resultados del programa en consola.

a. Inicio del programa



The screenshot shows an IDE interface with the following details:

- Top Bar:** Shows the user's name "Naranjo Castillo, Antonio".
- Menu Bar:** Includes options like Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and Auto... A search bar is also present.
- Toolbar:** Contains various icons for file operations, code navigation, and tool integration.
- Code Editor:** Displays Java code. A specific section of the code is highlighted in green:


```

161     case 4:
162         mensaje = "Cargar la lista de usuarios desde el archivo (deserialización)";
163         System.out.println(String.format("La opción elegida por el usuario es:"));
164         // Se llama al método estático de la clase Usuario cargarListaGrabada() para recuperar la lista de usuarios
165         Usuario.cargarListaGrabada(nombreArchivo,nombreFichero);
166         // Se comprueba si ambas listas son distintas (la temporal y la persistente en el fichero)
167         if (!Usuario.compararListas() || !Usuario.comprobarExistenciaArchivo(nombreArchivo))
168             System.out.println("Ha realizado cambios que no se ha guardado en disco. Si desea restaurarlos pulse 'S' para guardarlos");
169         restaurarDatos = teclado.nextLine();
      
```
- Output Window:** Titled "Output - Antonio_Naranjo_AD1_E1 (run)". It shows the following text:


```

run:
Introduzca el nombre del archivo para grabar la lista de usuarios, pulsa ENTER para introducir por defecto 'user.dat'

Existe el archivo 'user.dat' a continuación se muestran los datos grabados. 1
Datos grabados en disco: 2
3

Usuario [Identificador= 879wrtow, Contraseña= phoaeirhggwe, Dirección= pjowehifq, Año de nacimiento= 1999]
Usuario [Identificador= 3248g, Contraseña= aksdhfi, Dirección= 23342fsadf, Año de nacimiento= 3432]
Usuario [Identificador= 898gerh, Contraseña= apschdfw8e8rg, Dirección= 848489hapsi, Año de nacimiento= 1999]
Usuario [Identificador= 23245, Contraseña= asdfqwef, Dirección= qwertqwet, Año de nacimiento= 444]
Usuario [Identificador= yyggy, Contraseña= asldoi, Dirección= qrghopqbi, Año de nacimiento= 333]
Usuario [Identificador= 777, Contraseña= asdf, Dirección= qrwet, Año de nacimiento= 999]
Usuario [Identificador= 33234, Contraseña= asdf, Dirección= hqethr, Año de nacimiento= 778899]
Usuario [Identificador= 5678, Contraseña= asjtj, Dirección= ytkktr, Año de nacimiento= 3445]
Usuario [Identificador= 998877, Contraseña= bqiibOnb, Dirección= qbqebe, Año de nacimiento= 43]
Usuario [Identificador= 8686, Contraseña= sDaildhg, Dirección= nancifdhh, Año de nacimiento= 781]
      
```
- Bottom Status Bar:** Shows the run configuration "Antonio_Naranjo_AD1_E1 (run)", status "running...", time "160:82/1:62", and date/time "20:32 30/11/2025".

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.

The screenshot shows an IDE interface with several tabs at the top: Source, History, and others. Below the tabs, there is a toolbar with various icons. The main area displays Java code in a text editor. The code is part of a switch statement, specifically case 4, which handles the loading of user lists from a file. The code includes imports for System.out.println and String.format, and calls to Usuario.cargarListaGrabada() and Usuario.compararListas(). It also checks for changes in the file and uses teclado.nextLine() to read input.

The terminal output window below shows the execution of the program. It prompts the user to enter a file name, with 'user2.dat' highlighted by a red circle labeled '1'. It then informs the user that the file does not exist and no data will be displayed, with this message highlighted by a red circle labeled '2'. Finally, it shows the menu options numbered 1 through 6, with 'MENÚ' highlighted by a red circle labeled '3'.

```
case 4:
    mensaje = "Cargar la lista de usuarios desde el archivo (deserialización)";
    System.out.println(String.format("%nLa opción elegida por el usuario es:"));
    // Se llama al método estático de la clase Usuario cargarListaGrabada() para recuperar la lista persistente en el disco
    Usuario.cargarListaGrabada(nombreArchivo:nombreFichero);
    // Se comprueba si ambas listas son distintas (la temporal y la persistente en el disco)
    if (!Usuario.compararListas() || !Usuario.comprobarExistenciaArchivo(nombreArchivo:nombreFichero))
        System.out.println("Ha realizado cambios que no se ha guardado en disco. Si desea guardarlos pulse 'S' y presione ENTER");
    restaurarDatos = teclado.nextLine();
```

Output - Antonio_Naranjo_AD1_E1 (run) x Sonar Rule Details Window

```
run:
Introduzca el nombre del archivo para grabar la lista de usuarios, pulsa ENTER para introducir por defecto el nombre temporal.
user2.dat 1
El archivo 'user2.dat' no existe, no se mostrarán datos grabados. 2
---- 3
MENÚ
----
```

1. Agregar un usuario a la lista de usuarios.
2. Borrar un usuario a la lista de usuarios.
3. Guardar la lista de usuarios en un archivo (serialización).
4. Cargar la lista de usuarios desde el archivo (deserialización).
5. Mostrar los usuarios en la consola.
6. Exportar a fichero .txt la lista de usuarios.

0 Salir de la aplicación

Antonio_Naranjo_AD1_E1 (run) running... x 160:82/1:62 INS 20:37 30/11/2025

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.

b. Agregar un usuario

The screenshot shows an IDE interface with several tabs at the top: ...va, Cestajava, Gatosjava, Pescadorjava, Main.java [-/M], and Usuario.java [-/M]. The Main.java tab is active, displaying Java code. The code includes a section for adding users:

```
System.out.println("Lista de usuarios grabada en disco con éxito.");
break;

// Cargar la lista de usuarios desde el archivo (deserialización)
case 4:
    mensaje = "Cargar la lista de usuarios desde el archivo (deserialización)";
    System.out.println(String.format("%nLa opción elegida por el usuario es:"));
    // Se llama al método estático de la clase Usuario cargarListaGrabada() para recuperar la lista de usuarios.
```

The Output window shows the execution of the application. It starts with a welcome message and then displays the user input for adding a new user:

```
0. Salir de la aplicación.

----> Opción seleccionada por el usuario: 1

La opción elegida por el usuario es: 1 [Agregar un usuario a la lista de usuarios]
Introduzca el identificador del usuario
11223344A
Introduzca la contraseña del usuario
paseoporelparque
Introduzca el dirección del usuario
Calle Cervantes, 5
Introduzca el año de nacimiento del usuario
1972
El usuario con identificador '11223344A' ha sido agregado con éxito

----
```

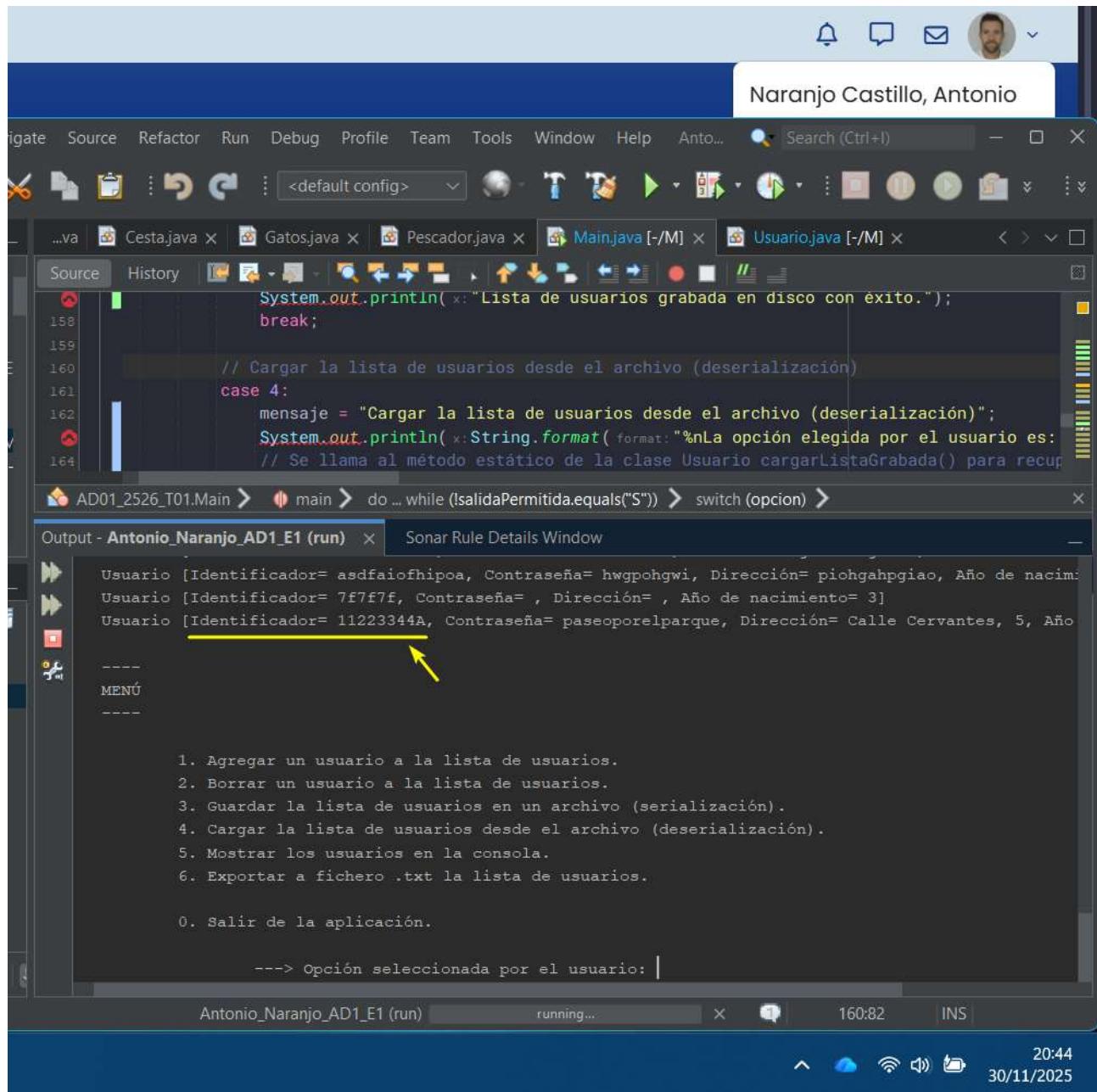
At the bottom of the output, it shows the menu again:

```
MENÚ
```

The status bar at the bottom indicates the application is running, the time is 16:08, and the date is 30/11/2025.

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

c. Mostrar los usuarios en consola



The screenshot shows an IDE interface with several tabs at the top: Source, History, and others. The main code editor displays Java code related to user management. A yellow arrow points to the output window, which shows the results of running the application. The output window title is "Output - Antonio_Naranjo_AD1_E1 (run)". It displays three user objects and a menu:

```
System.out.println("Lista de usuarios grabada en disco con éxito.");
break;

// Cargar la lista de usuarios desde el archivo (deserialización)
case 4:
    mensaje = "Cargar la lista de usuarios desde el archivo (deserialización)";
    System.out.println(String.format("%nLa opción elegida por el usuario es:"));
    // Se llama al método estático de la clase Usuario cargarListaGrabada() para recuperar la lista de usuarios desde el archivo.
```

Output - Antonio_Naranjo_AD1_E1 (run) | Sonar Rule Details Window

```
Usuario [Identificador= asdfaiofhipoa, Contraseña= hwgpohgwi, Dirección= pioghahpgiao, Año de nacimiento= 2000]
Usuario [Identificador= 7f7f7f, Contraseña= , Dirección= , Año de nacimiento= 3]
Usuario [Identificador= 11223344A, Contraseña= paseoporelparque, Dirección= Calle Cervantes, 5, Año de nacimiento= 1990]
```

MENÚ

```
1. Agregar un usuario a la lista de usuarios.
2. Borrar un usuario a la lista de usuarios.
3. Guardar la lista de usuarios en un archivo (serialización).
4. Cargar la lista de usuarios desde el archivo (deserialización).
5. Mostrar los usuarios en la consola.
6. Exportar a fichero .txt la lista de usuarios.

0. Salir de la aplicación.
```

---> Opción seleccionada por el usuario: |

Antonio_Naranjo_AD1_E1 (run) | running... | 160:82 | INS | 20:44 | 30/11/2025

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

d. Borrar usuario

The screenshot shows an IDE interface with several tabs at the top: ...va, Cestajava, Gatos.java, Pescador.java, Main.java [-/M], and Usuario.java [-/M]. The Main.java tab is active, displaying Java code related to user management. Below the tabs is a toolbar with various icons. The main workspace shows code snippets like:

```
System.out.println("Lista de usuarios grabada en disco con éxito.");
break;
// Cargar la lista de usuarios desde el archivo (deserialización)
case 4:
    mensaje = "Cargar la lista de usuarios desde el archivo (deserialización)";
    System.out.println(String.format("%nLa opción elegida por el usuario es:");
    // Se llama al método estático de la clase Usuario cargarListaGrabada() para recuperar
```

The terminal window below shows the execution of the application. It prompts the user to enter the identifier of the user to delete, and then confirms that the user with identifier '11223344A' has been deleted successfully. Arrows point to the input '11223344A' and the confirmation message.

```
Introduzca el identificador del usuario a eliminar
11223344A
El usuario con identificador '11223344A' ha sido borrado con éxito
```

The menu options listed are:

1. Agregar un usuario a la lista de usuarios.
2. Borrar un usuario a la lista de usuarios.
3. Guardar la lista de usuarios en un archivo (serialización).
4. Cargar la lista de usuarios desde el archivo (deserialización).
5. Mostrar los usuarios en la consola.
6. Exportar a fichero .txt la lista de usuarios.
0. Salir de la aplicación.

At the bottom, it says '--> Opción seleccionada por el usuario: |'.

The status bar at the bottom shows the application name 'Antonio_Naranjo_AD1_E1 (run)', status 'running...', time '16:08:22', and date '30/11/2025'. The system tray shows icons for battery, signal, and network.

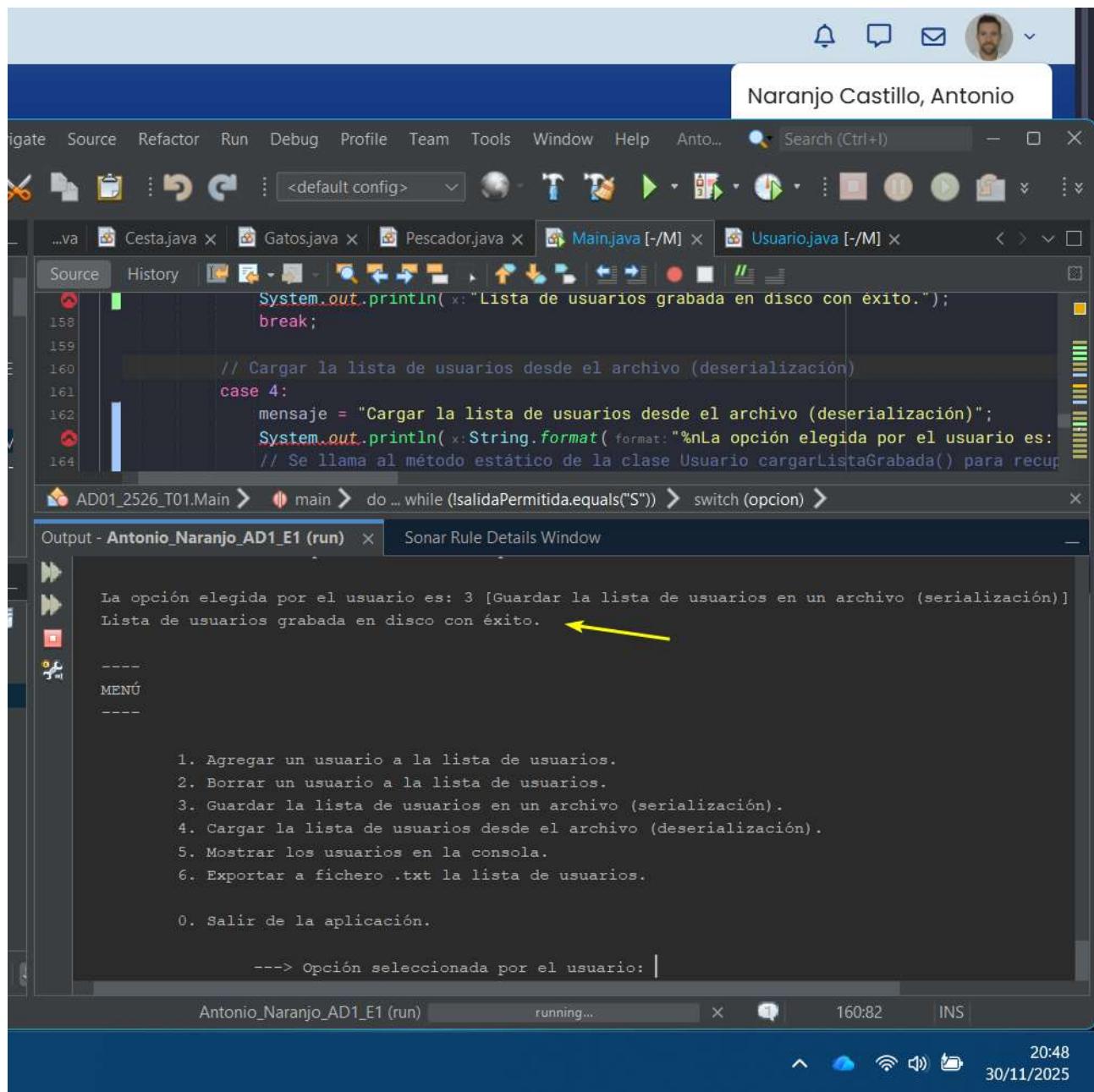
Se selecciona la opción 2 y se pasa como argumento la identificación del usuario, se aporta la identificación del usuario recientemente aportado.

The screenshot shows an IDE interface with the following details:

- Top Bar:** Shows navigation tabs like Navigate, Source, Refactor, Run, Debug, etc., and a search bar labeled "Search (Ctrl+I)".
- Title Bar:** Displays the name "Naranjo Castillo, Antonio".
- Toolbar:** Includes icons for file operations, code navigation, and other development tools.
- Project Explorer:** Shows files like Cestajava.java, Gatosjava.java, Pescador.java, Main.java, and Usuario.java.
- Code Editor:** Displays Java code for a menu system. A yellow arrow points from the output window back to the code editor, specifically highlighting the line where the user was deleted.
- Output Window:** Shows the execution of the application. It prints three user objects and then lists menu options. The last option listed is "0. Salir de la aplicación." followed by a prompt: "----> Opción seleccionada por el usuario: |".
- Status Bar:** Shows the run configuration "Antonio_Naranjo_AD1_E1 (run)", the timestamp "16:08:22", and the date "30/11/2025".

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.

e. Guardar lista (Serialización)



The screenshot shows an IDE interface with several tabs at the top: ...va, Cestajava, Gatosjava, Pescadorjava, Main.java [-/M], and Usuario.java [-/M]. The Main.java tab is active, displaying Java code. The code includes a println statement that outputs "Lista de usuarios grabada en disco con éxito." followed by a break statement. Below this, there is a switch block with case 4 handling. The output window shows the application's terminal output:

```
La opción elegida por el usuario es: 3 [Guardar la lista de usuarios en un archivo (serialización)]
Lista de usuarios grabada en disco con éxito.
```

A yellow arrow points from the text "Lista de usuarios grabada en disco con éxito." to the right. The output window also displays a menu with numbered options and a selection prompt at the bottom:

```
MENÚ
-----
1. Agregar un usuario a la lista de usuarios.
2. Borrar un usuario a la lista de usuarios.
3. Guardar la lista de usuarios en un archivo (serialización).
4. Cargar la lista de usuarios desde el archivo (deserialización).
5. Mostrar los usuarios en la consola.
6. Exportar a fichero .txt la lista de usuarios.

0. Salir de la aplicación.

----> Opción seleccionada por el usuario: |
```

The status bar at the bottom shows the application name "Antonio_Naranjo_AD1_E1 (run)", "running...", the time "16:08", and the date "30/11/2025".

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

f. Cargar lista (deserialización)

The screenshot shows an IDE interface with several tabs at the top: Start Page, Main.java, Cesta.java, Gatos.java, Pescador.java, and Main.java [-/M]. The Main.java tab is active, displaying Java code. The code includes logic for handling user input and loading user lists from files.

```
169     \xtLine();
170
171     if (ulsa intencionadamente el caracter "S" se optará por salir del programa
172         &lt;"S">) {
173         chivo:nombreFichero);
174
175         lista de usuarios ha sido cargada desde archivo con éxito.");
176 }
```

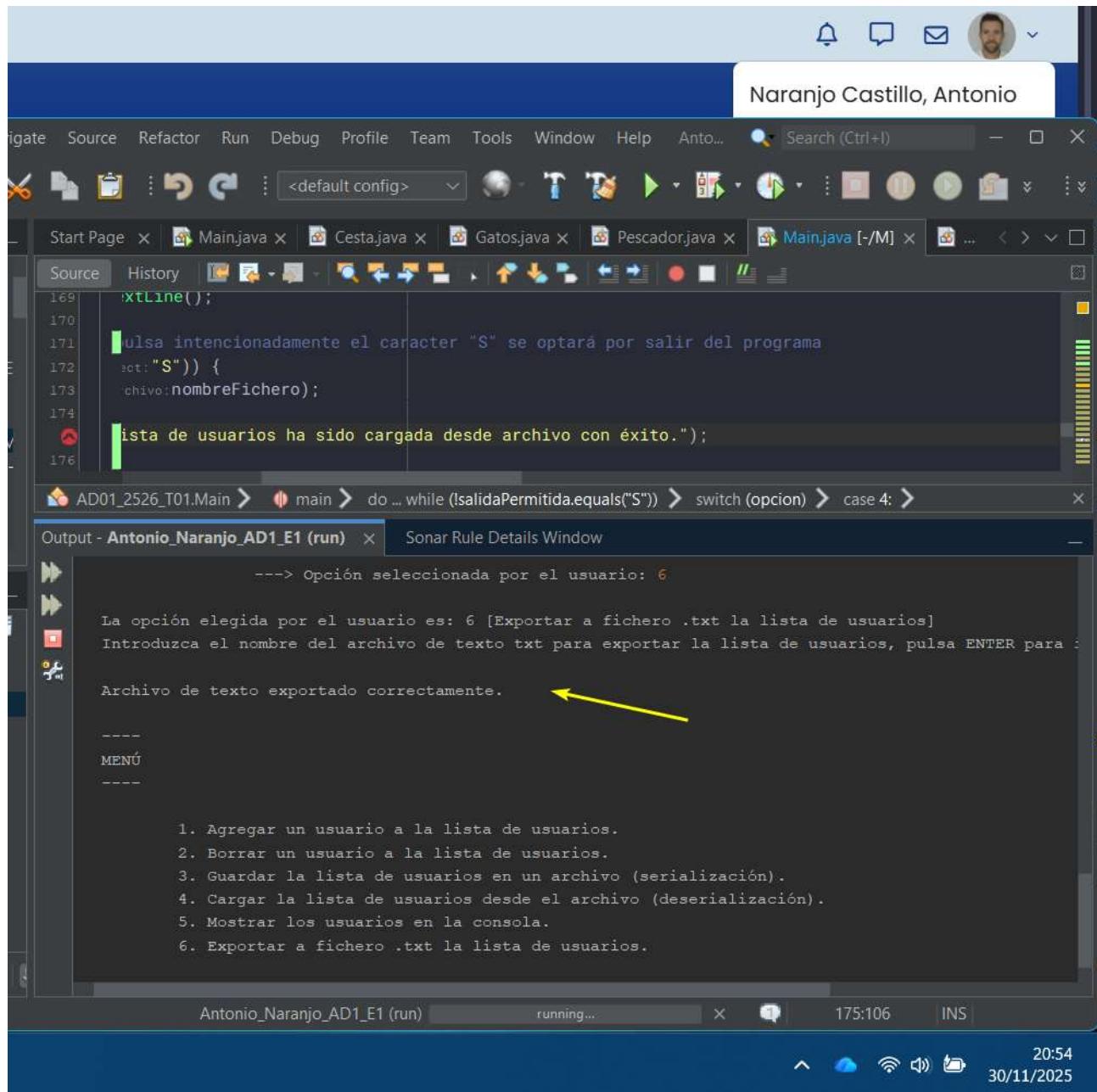
The terminal window below shows the application's output:

```
La opción elegida por el usuario es: 4 [Cargar la lista de usuarios desde el archivo (deserialización).
La lista de usuarios ha sido cargada desde archivo con éxito. ----- MENÚ
-----  
1. Agregar un usuario a la lista de usuarios.
2. Borrar un usuario a la lista de usuarios.
3. Guardar la lista de usuarios en un archivo (serialización).
4. Cargar la lista de usuarios desde el archivo (deserialización).
5. Mostrar los usuarios en la consola.
6. Exportar a fichero .txt la lista de usuarios.  
0. Salir de la aplicación.  
----> Opción seleccionada por el usuario: |
```

A yellow arrow points from the text "La lista de usuarios ha sido cargada desde archivo con éxito." in the terminal output to the corresponding line in the code editor.

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

g. Exportación archivo TXT



The screenshot shows an IDE interface with the following details:

- Title Bar:** Shows the name "Naranjo Castillo, Antonio".
- Toolbar:** Includes standard icons for file operations like Open, Save, and Run.
- Menu Bar:** Lists options like Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and Auto... A search bar is also present.
- Code Editor:** Displays Java code related to exporting user lists. The code includes logic for handling the letter 'S' to exit the program and saving the user list to a file named "nombreFichero".
- Output Window:** Titled "Output - Antonio_Naranjo_AD1_E1 (run)", it shows the following text:

```
----> Opción seleccionada por el usuario: 6

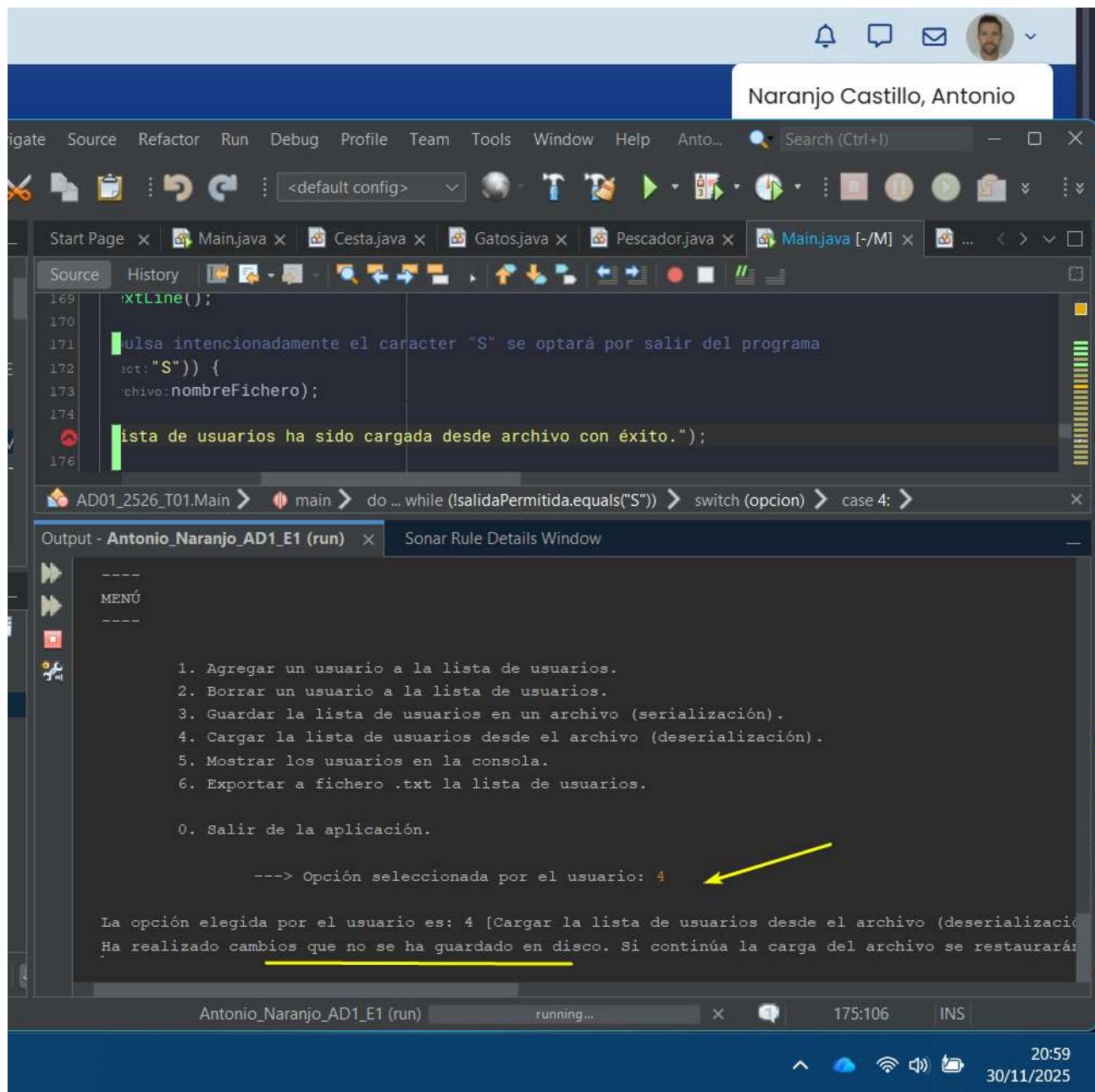
La opción elegida por el usuario es: 6 [Exportar a fichero .txt la lista de usuarios]
Introduzca el nombre del archivo de texto txt para exportar la lista de usuarios, pulsa ENTER para :

Archivo de texto exportado correctamente.
```

A yellow arrow points from the text "Archivo de texto exportado correctamente." to the right.
- Status Bar:** Shows the run configuration "Antonio_Naranjo_AD1_E1 (run)", status "running...", time "175:106", and mode "INS". It also displays system icons for battery, signal, and date/time "30/11/2025 20:54".

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.

h. ¿Cargar datos guardados?



The screenshot shows an IDE interface with the following details:

- Top Bar:** Shows the user's name "Naranjo Castillo, Antonio".
- Toolbar:** Includes standard icons for file operations like Open, Save, and Cut.
- Project Explorer:** Lists several Java files: Start Page, Main.java, Cesta.java, Gatos.java, Pescador.java, and Main.java [-/M].
- Code Editor:** Displays Java code for a menu system. It includes a switch statement where option 4 corresponds to loading users from a file.
- Output Window:** Titled "Output - Antonio_Naranjo_AD1_E1 (run)", it shows the following text:


```

MENÚ
-----
1. Agregar un usuario a la lista de usuarios.
2. Borrar un usuario a la lista de usuarios.
3. Guardar la lista de usuarios en un archivo (serialización).
4. Cargar la lista de usuarios desde el archivo (deserialización).
5. Mostrar los usuarios en la consola.
6. Exportar a fichero .txt la lista de usuarios.

0. Salir de la aplicación.

----> Opción seleccionada por el usuario: 4
      
```

 A yellow arrow points to the number 4 in the output.

Below the menu options, there is a message: "La opción elegida por el usuario es: 4 [Cargar la lista de usuarios desde el archivo (deserialización)]. Ha realizado cambios que no se ha guardado en disco. Si continúa la carga del archivo se restaurará..."
- Bottom Status Bar:** Shows the run configuration "Antonio_Naranjo_AD1_E1 (run)", status "running...", timestamp "175:106", and mode "INS".
- System Tray:** Shows icons for battery, signal, and date/time "30/11/2025 20:59".

En caso de que el usuario pretenda cargar los datos guardados 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.

The screenshot shows an IDE interface with several windows:

- Editor Window:** Displays Java code in Main.java. The code includes logic for saving user lists to files and exiting the program if 'S' is pressed.
- Output Window:** Shows terminal output. It indicates changes have been made but not saved to disk. It also displays the message "La lista de usuarios ha sido cargada desde archivo con éxito." followed by a menu and a list of options.
- Terminal Output:** The terminal output window shows:
 - "Ha realizado cambios que no se ha guardado en disco. Si continúa la carga del archivo se restaurará"
 - "S" (with a yellow arrow pointing to it)
 - "La lista de usuarios ha sido cargada desde archivo con éxito." (with a yellow arrow pointing to it)
- Bottom Status Bar:** Shows the application name "Antonio_Naranjo_AD1_E1 (run)", status "running...", time "175:106", and mode "INS".
- System Tray:** Shows icons for battery, signal, and date/time "21:03 30/11/2025".

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.

i. Salida del programa

The screenshot shows an IDE interface with a Java code editor and a terminal window.

Java Code Editor: Displays a file named Main.java with the following code:

```
169     tos = teclado.nextLine();
170
171     // Si el programa pulsa intencionadamente el carácter "S" se optará por salir del programa
172     if (tos.equals("anObject: "S")) {
173         guardarLista(nombreArchivo, nombreFichero);
174
175         println("La lista de usuarios ha sido cargada desde archivo con éxito.");
176     }
```

Terminal Window: Shows the execution of the program:

```
AD01_2526_T01.Main > main > do ... while (!salidaPermitida.equals("S")) > switch (opcion) > case 4: >
Output - Antonio_Naranjo_AD1_E1 (run) > Sonar Rule Details Window
```

```
1. Agregar un usuario a la lista de usuarios.
2. Borrar un usuario a la lista de usuarios.
3. Guardar la lista de usuarios en un archivo (serialización).
4. Cargar la lista de usuarios desde el archivo (deserialización).
5. Mostrar los usuarios en la consola.
6. Exportar a fichero .txt la lista de usuarios.

0. Salir de la aplicación.

---> Opción seleccionada por el usuario: 0

La opción elegida por el usuario es: 0 [Salir de la aplicación]
Ha habido cambios en el programa que todavía no se han guardado. Si desea guardarlos ejecute la opción S
BUILD SUCCESSFUL (total time: 55 seconds)
```

Yellow arrows highlight the message "Ha habido cambios en el programa que todavía no se han guardado. Si desea guardarlos ejecute la opción S".

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.