**CST-239 Activity 4**

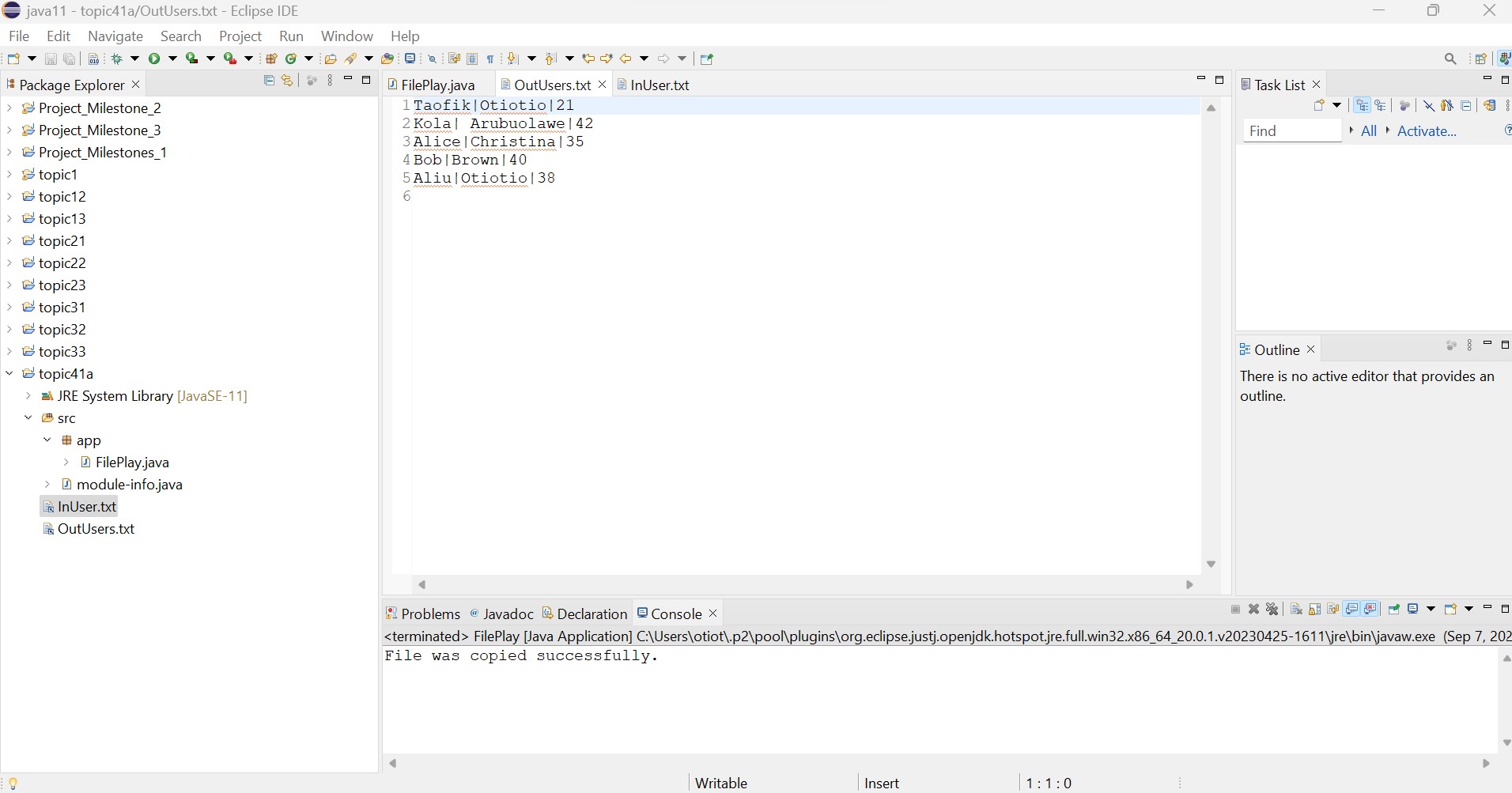
**Sept 8 ,2023**

**Taofik Otiotio**

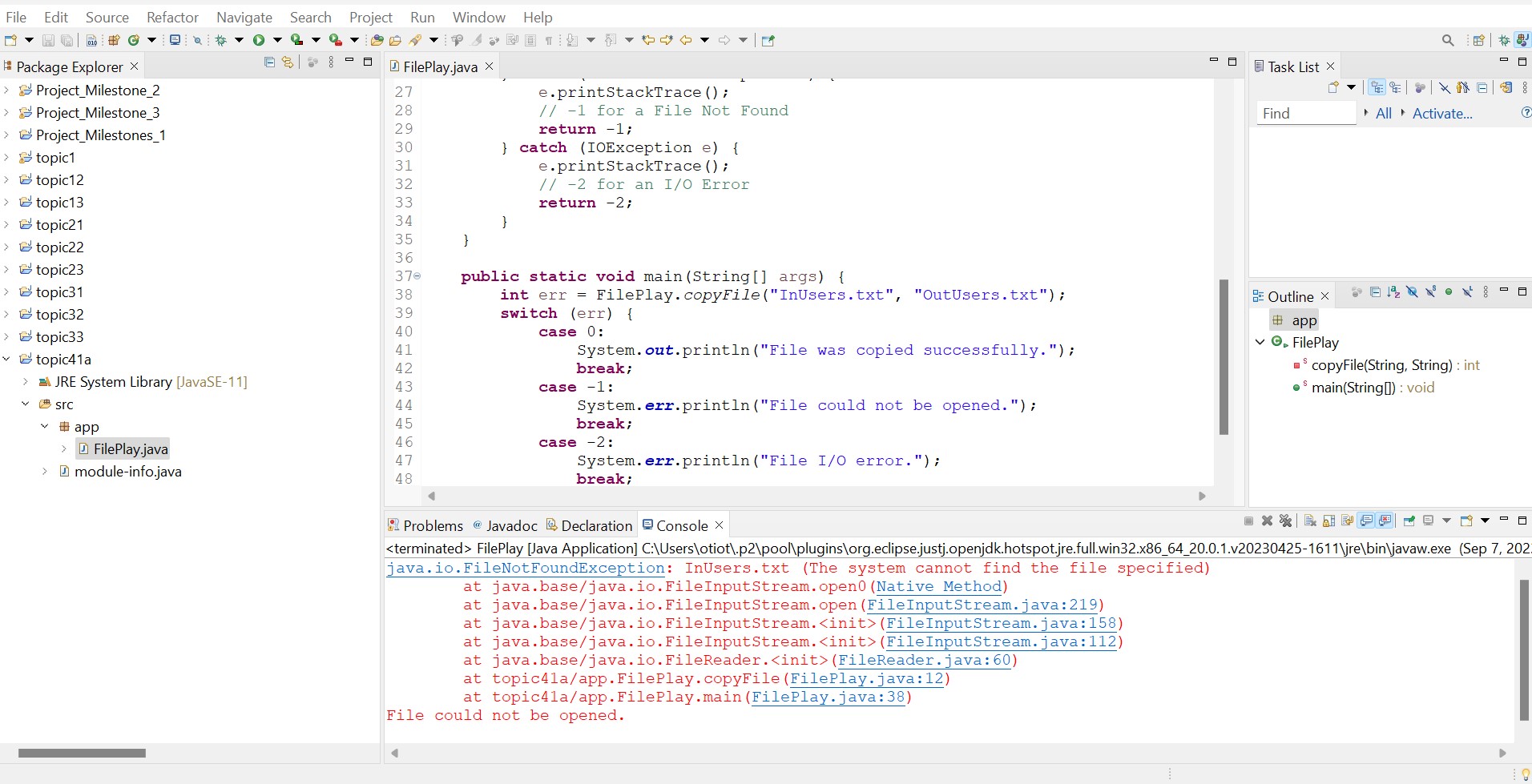
**Part 1a:**

* Create a new Java project named "topic4-1a."
* Create a file named "InUsers.txt" with user records.
* Implement a **copyFile()** method to copy the input file to the output file.
* Handle exceptions for file operations and return error codes.
* Display error messages based on error codes in the **main()** method.
* Run the application and capture console output and file contents.

**Take a screenshot of the final console output and the resulting output file**



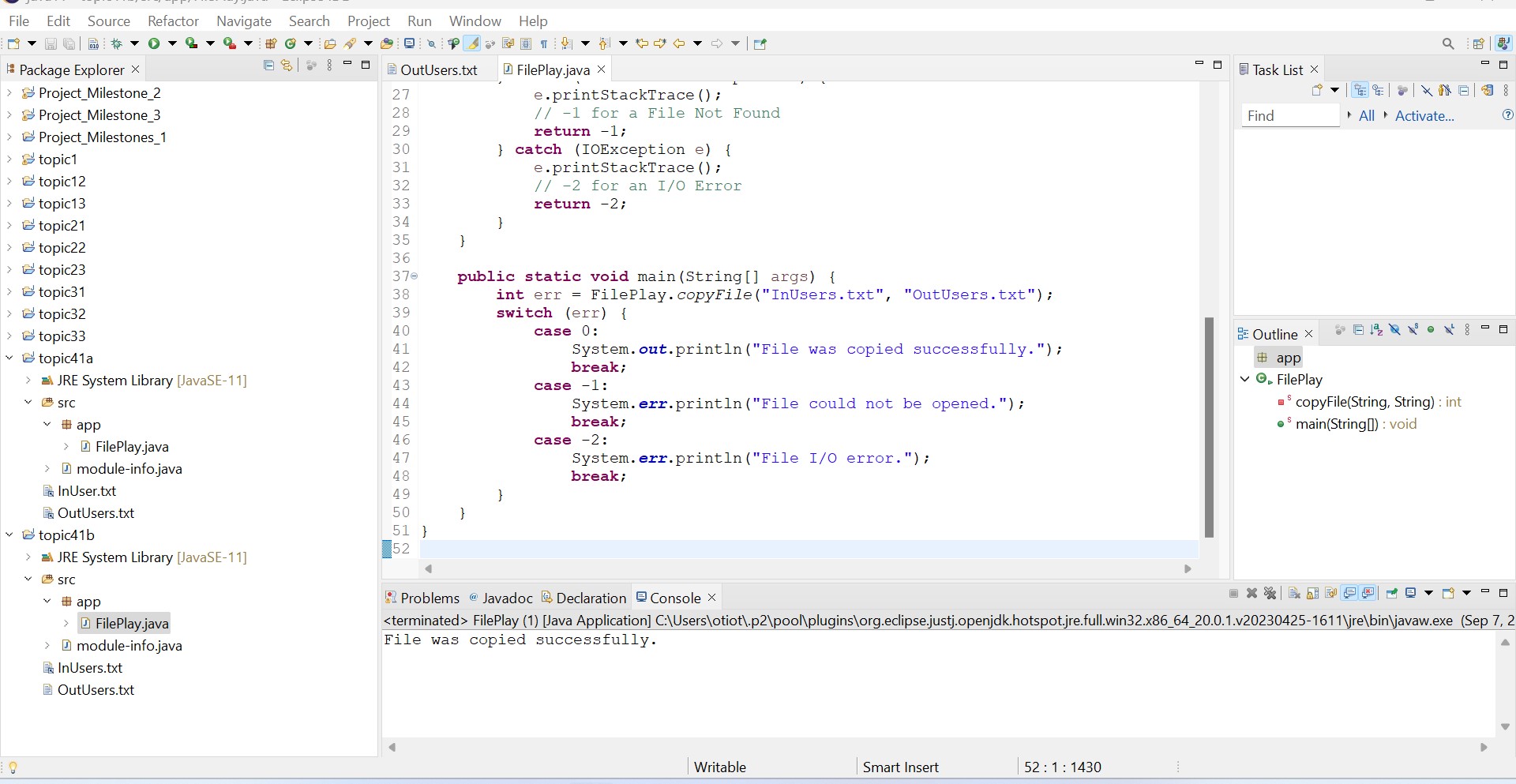
**Take a screenshot of the final console output that includes the exception stack trace and error message.**

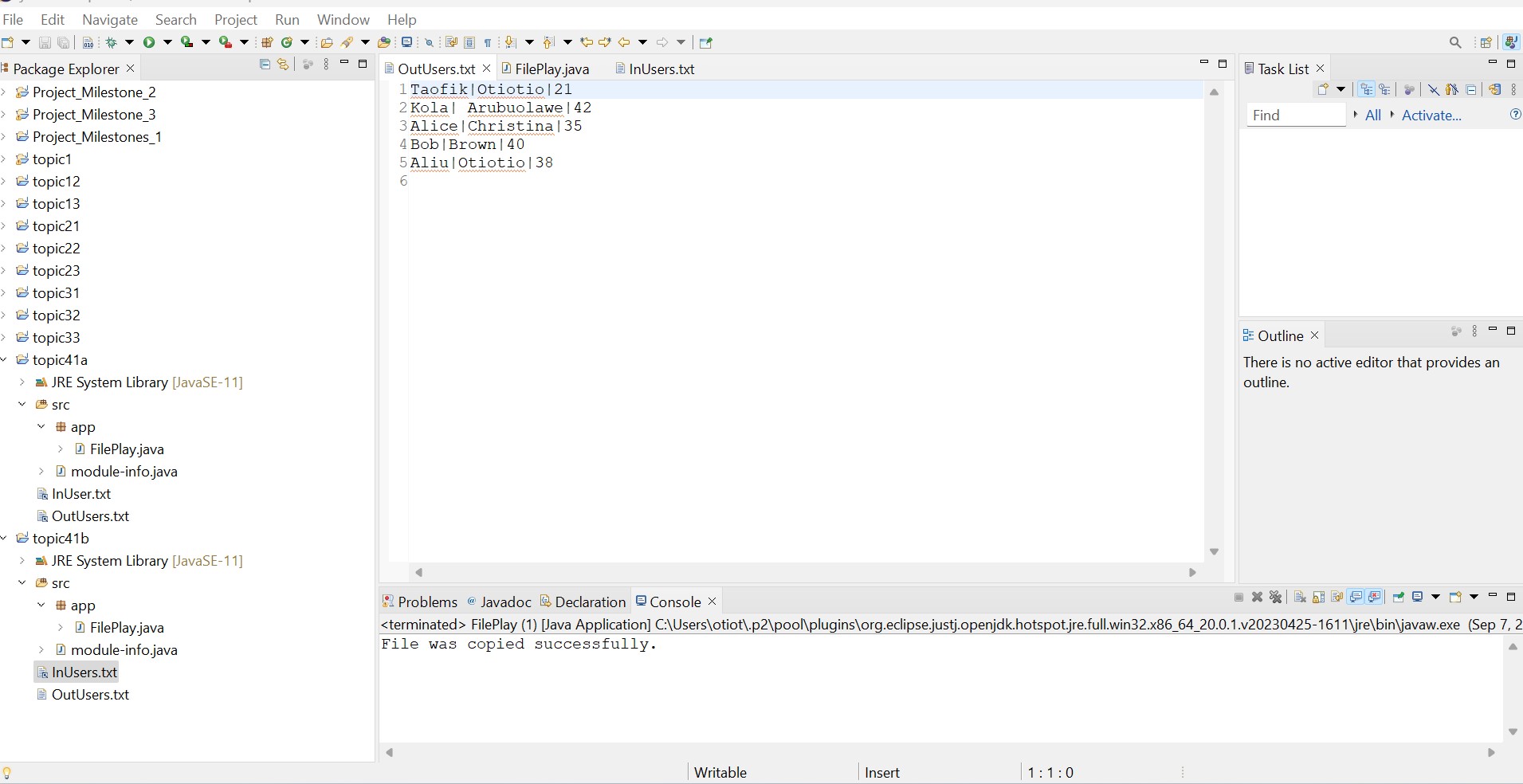


**Part 1b:**

* Create a new Java project named "topic4-1b."
* Copy the code from Part 1a to this project.
* Update file I/O to use **BufferedReader** and **BufferedWriter** for efficiency.
* Run the application and capture console output and file contents.

**Take a screenshot of the final console output and the resulting output file**

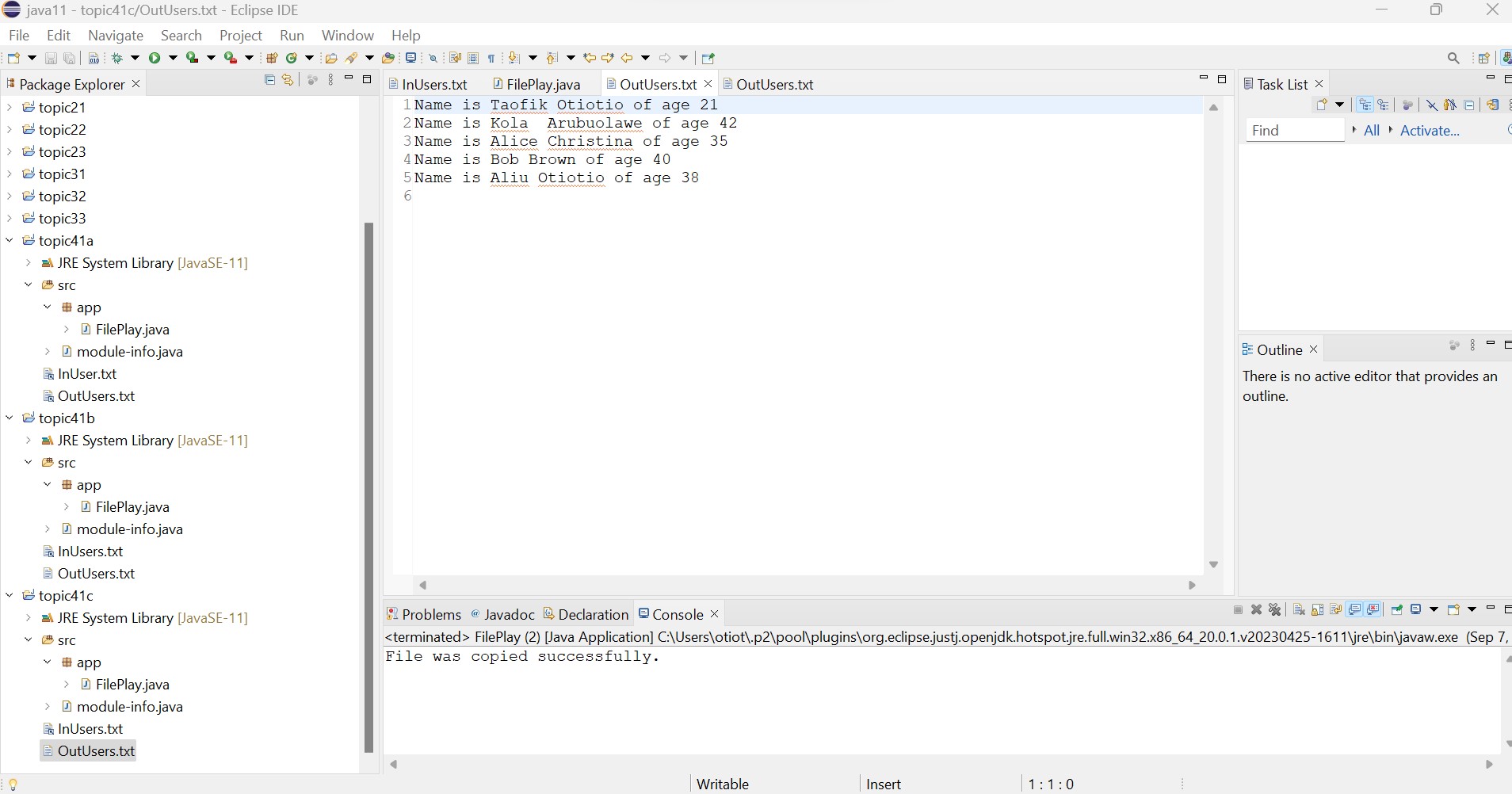




**Part 1c:**

* Create a new Java project named "topic4-1c."
* Copy the code from Part 1b to this project.
* Modify the file read and write logic to tokenize data and format output.
* Run the application and capture console output and file contents.

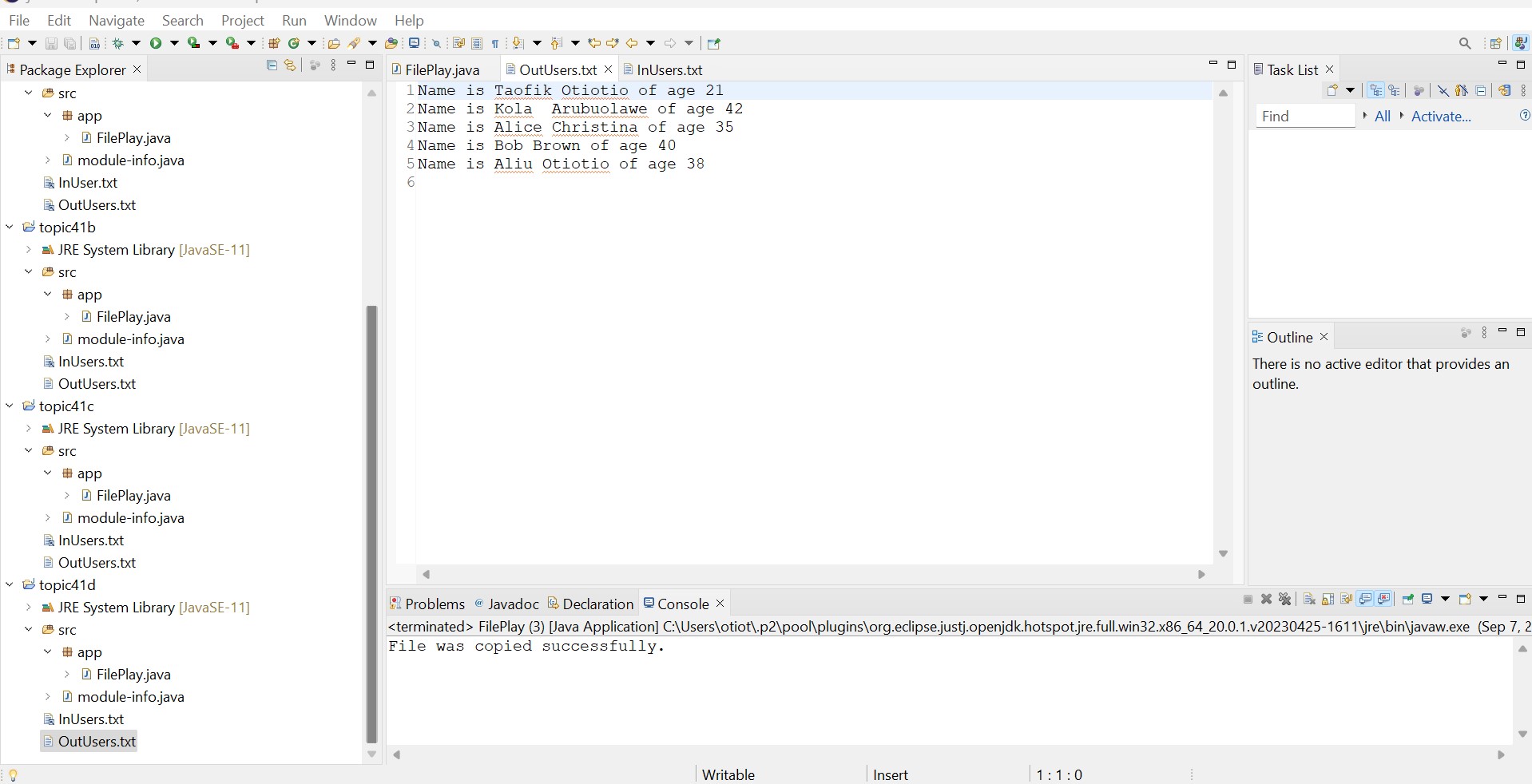
**Take a screenshot of the final console output and the resulting output file**



**Part 1d:**

* Create a new Java project named "topic4-1d."
* Copy the code from Part 1c to this project.
* Remove exception handling in the **copyFile()** method and use **throws** clause.
* Change the return type of **copyFile()** to **void**.
* Implement try-catch blocks in the **main()** method to handle exceptions.
* Run the application and capture console output and file contents.

**Take a screenshot of the final console output and the resulting output file**



**Part 2: Reading and Writing JSON Files**

**THEORY OF OPERATION**

### **DemoJsonFiles Class:**

**saveToFile() Method**:

This method takes three parameters: **filename** (the name of the JSON file), **car** (a **Car** object to be saved), and **append** (a boolean indicating whether to append to the existing file. It creates a **FileWriter** and **PrintWriter** to write to the specified file. Uses the Jackson **ObjectMapper** to convert the **Car** object into a JSON string and writes the JSON string to the file

**readFromFile() Method**:

This method takes one parameter: **filename** (the name of the JSON file to read from. It initializes an empty **ArrayList** to store **Car** objects and It uses Jackson's **ObjectMapper** to parse the JSON data from the file. It reads the file line by line using a **Scanner**. For each line, it deserializes the JSON data into a **Car** object and adds it to the **ArrayList**. Finally, it returns the **ArrayList** containing the **Car** objects read from the file.

**Main Method**

An array of **Car** objects is created, representing different cars. Each **Car** object in the array is serialized to a JSON file using the **saveToFile()** method, and the **append** parameter is set to **true** to append each car's data to the file. Then, the **readFromFile()** method is called to read the JSON data from the file, which is deserialized into an **ArrayList** of **Car** objects. A loop iterates through the **ArrayList** and prints details of each car (year, make, model, odometer, engine liters) to the console.

### **Car Class:**

### This class represents a car with properties such as **year**, **make**, **model**, **odometer**, and **engineLiters**. It provides both a default constructor and a parameterized constructor to initialize car objects. Getter and setter methods are included for each property to allow access and modification.

### **Overall Operation:**

The code demonstrates the complete process of serializing **Car** objects to JSON format and writing them to a file (**saveToFile()**). It also illustrates how to read JSON data from a file, deserialize it back into **Car** objects (**readFromFile()**), and use the objects in your Java application. The **main()** method creates sample **Car** objects, saves them to a JSON file, reads them back, and displays the car details on the console.

**screenshots of application in operation.**

