**WEB WEAVERS**

**EHINMOLA ADEDEJI HARRY**

**BHU/22/04/05/0003**

**RHEMA CHOJI DAYLOP**

**BHU/22/04/05/0076**

**PELIAH OSIGBMEMHE IDEGBESOR**

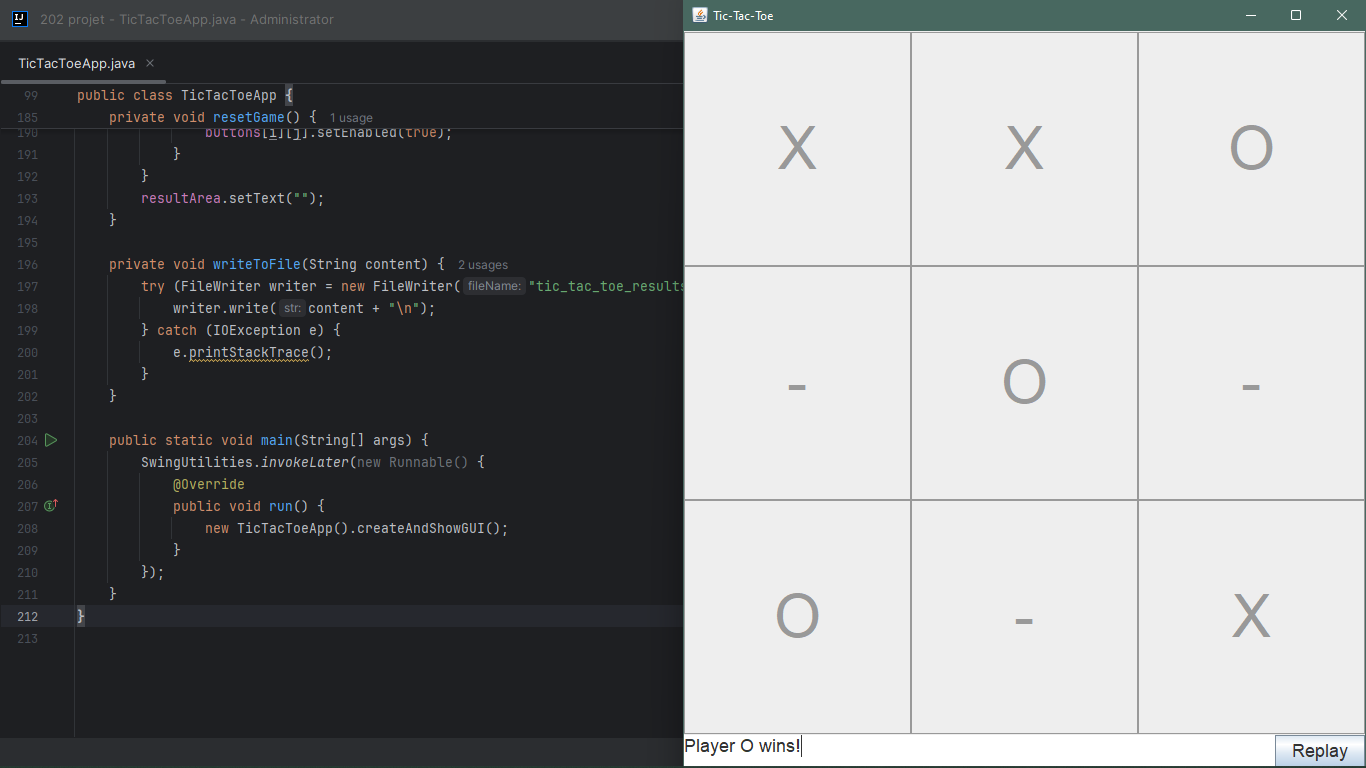
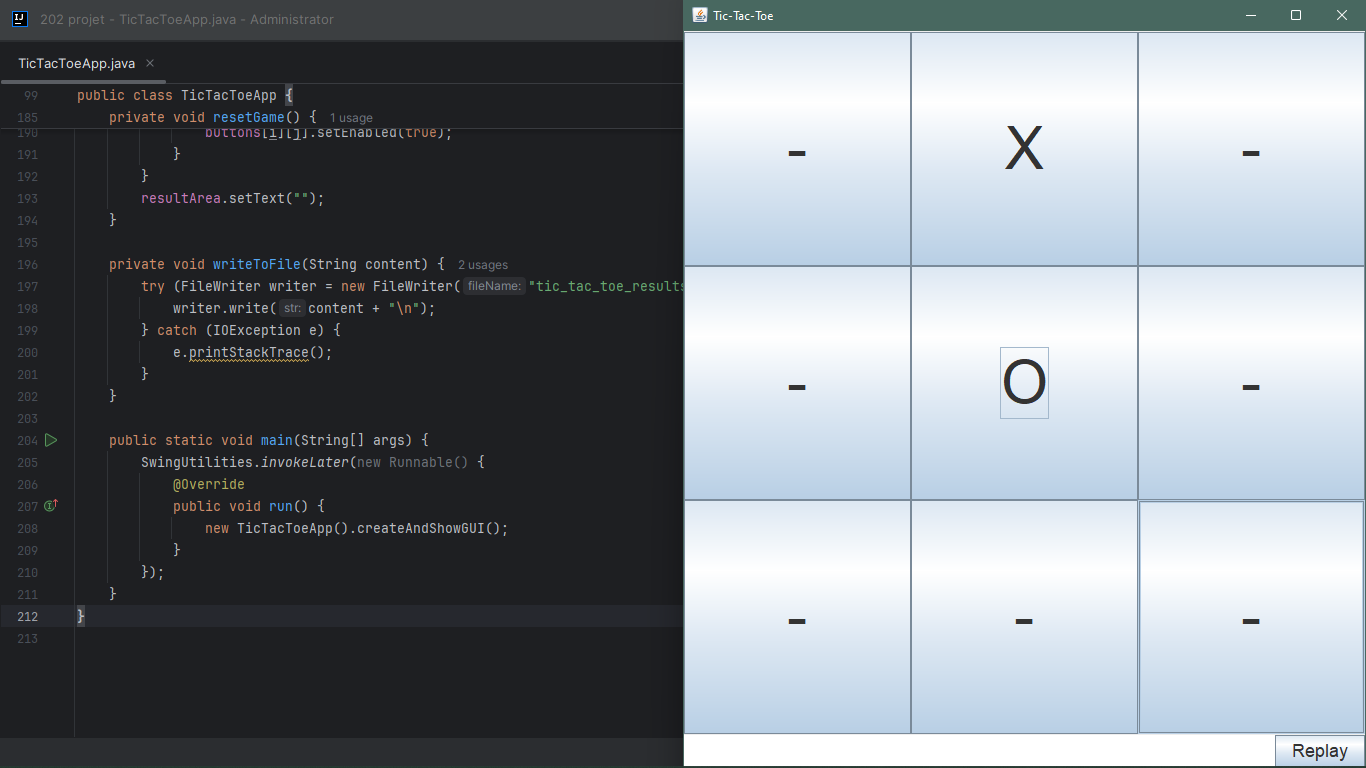
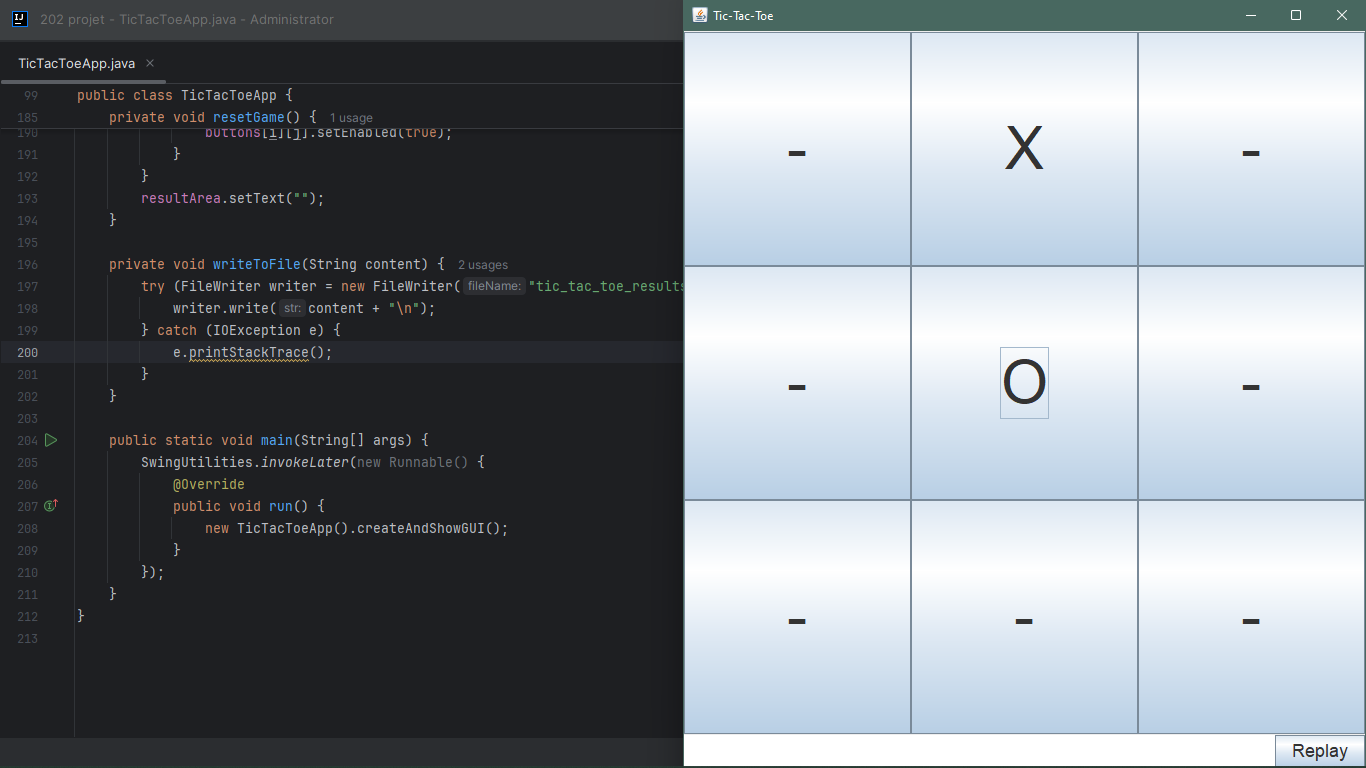
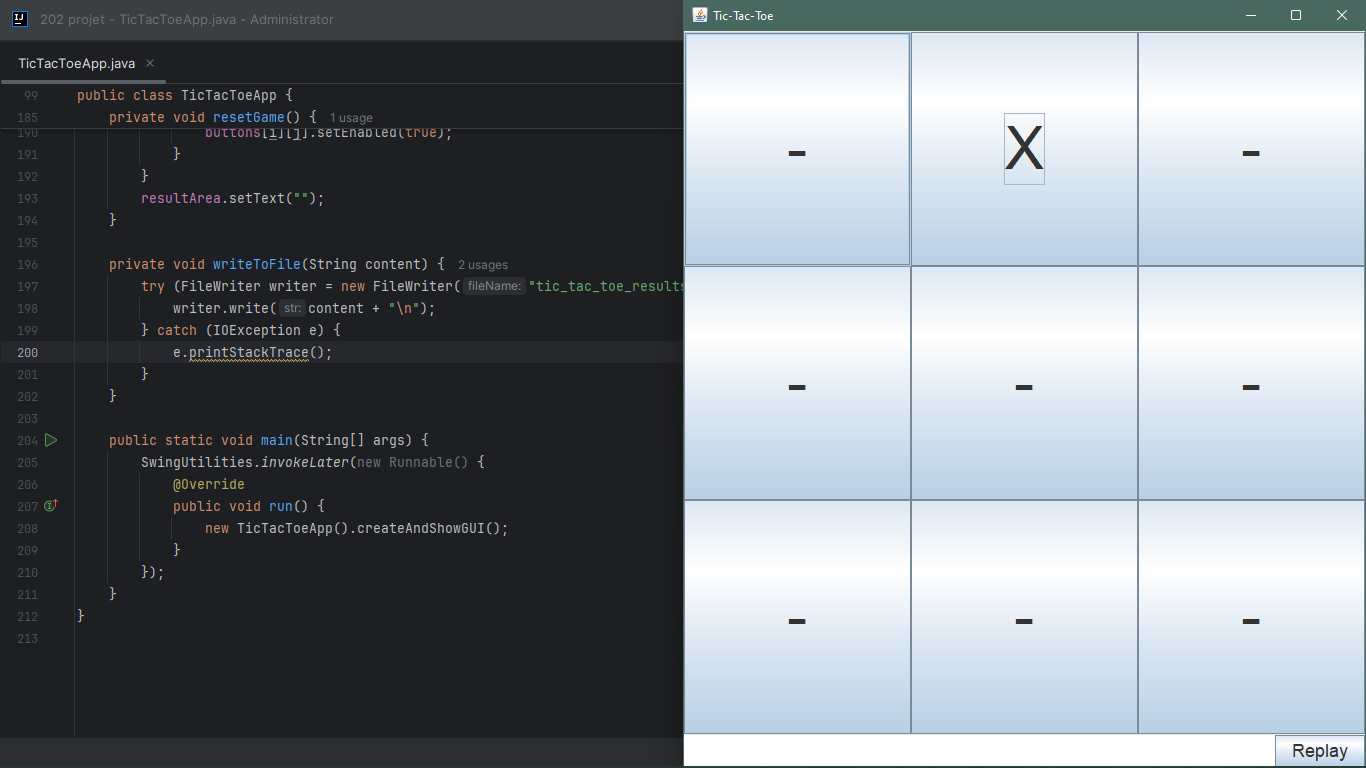
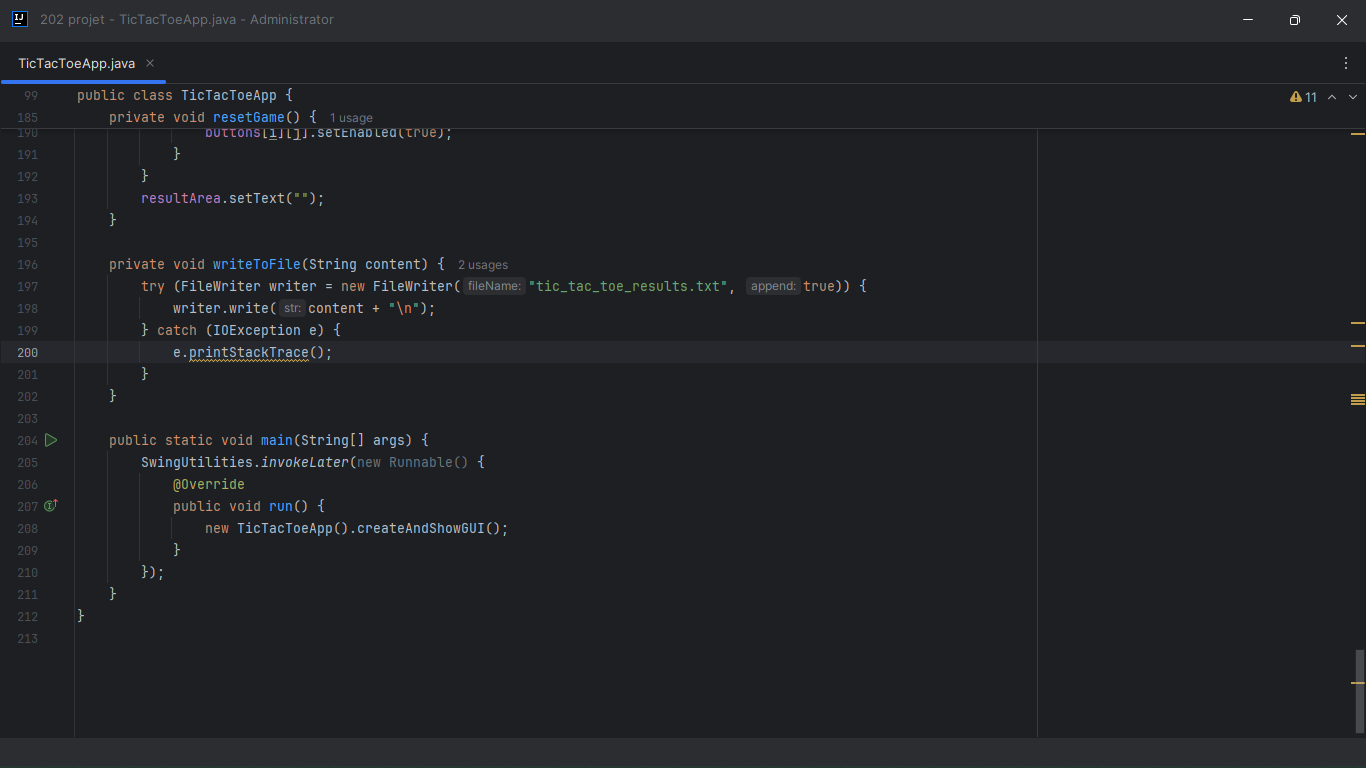
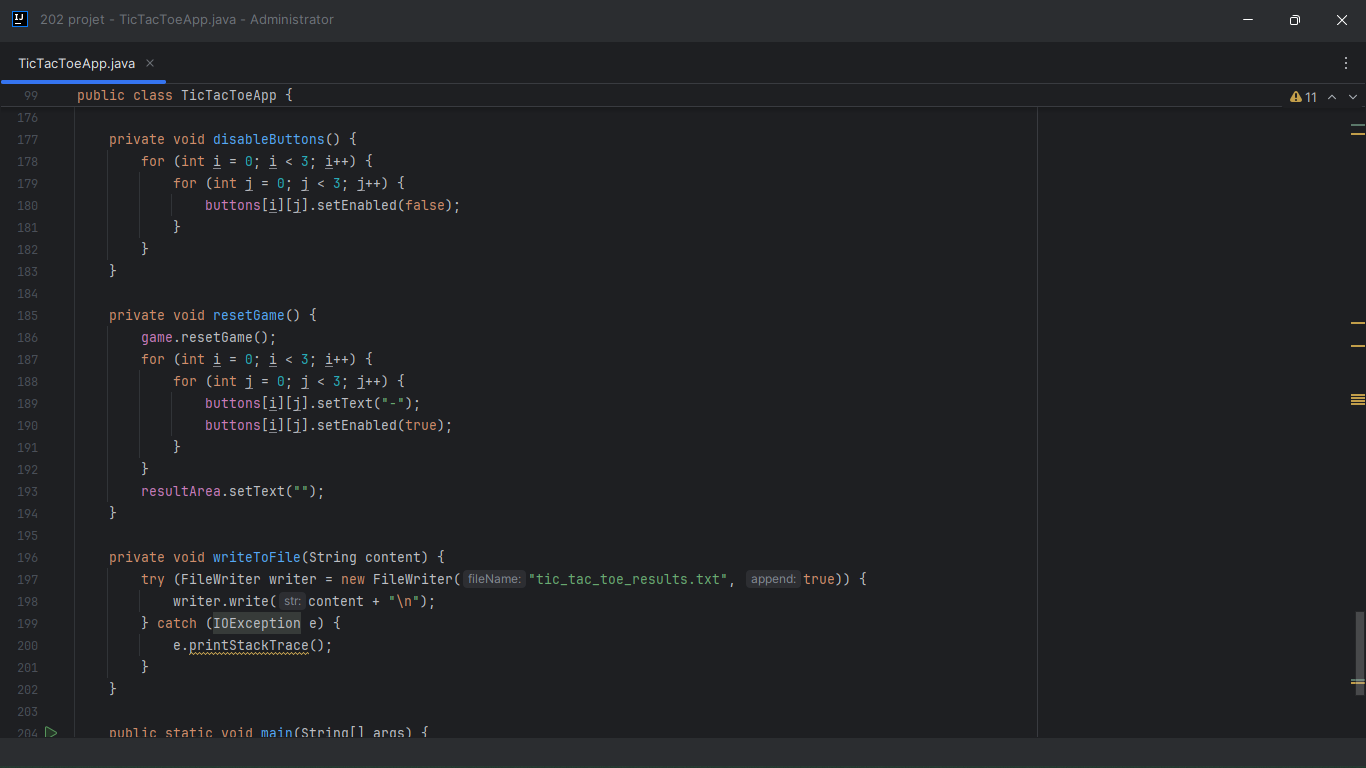
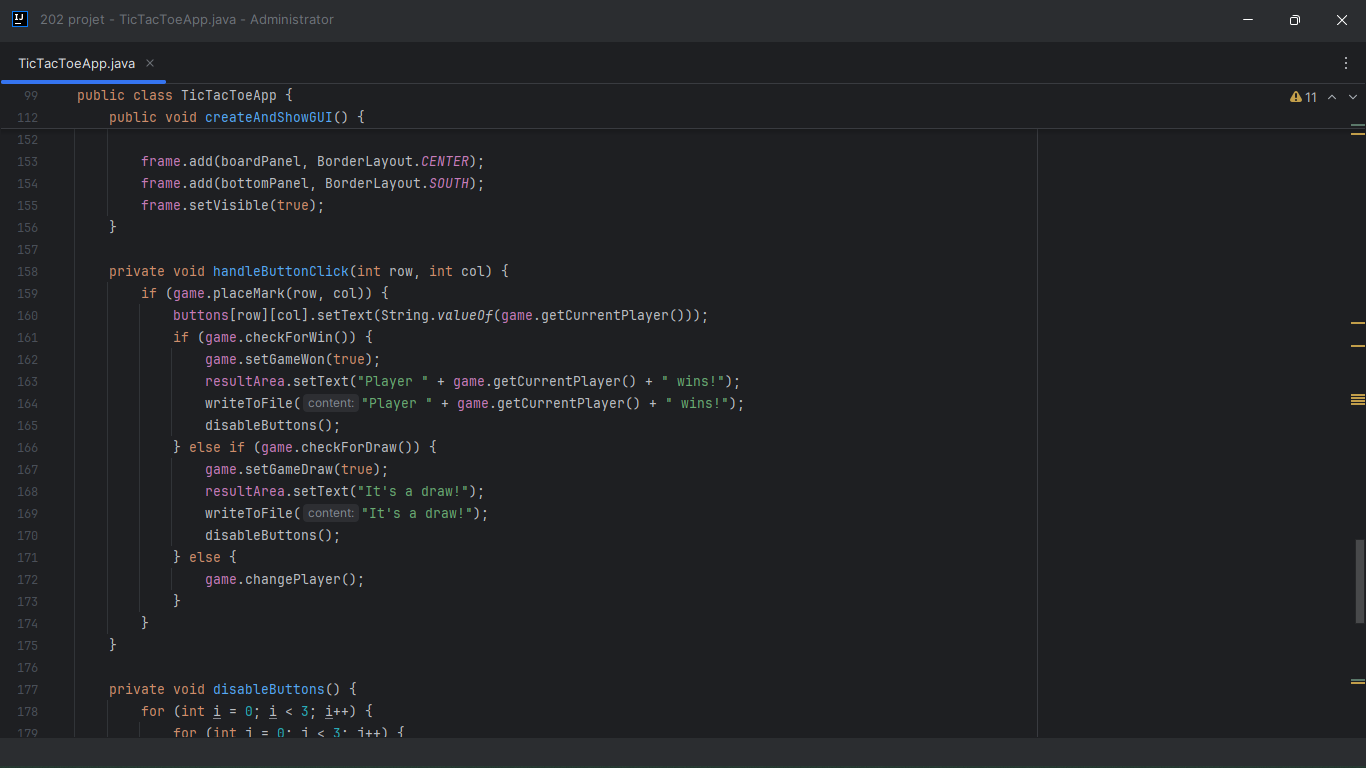
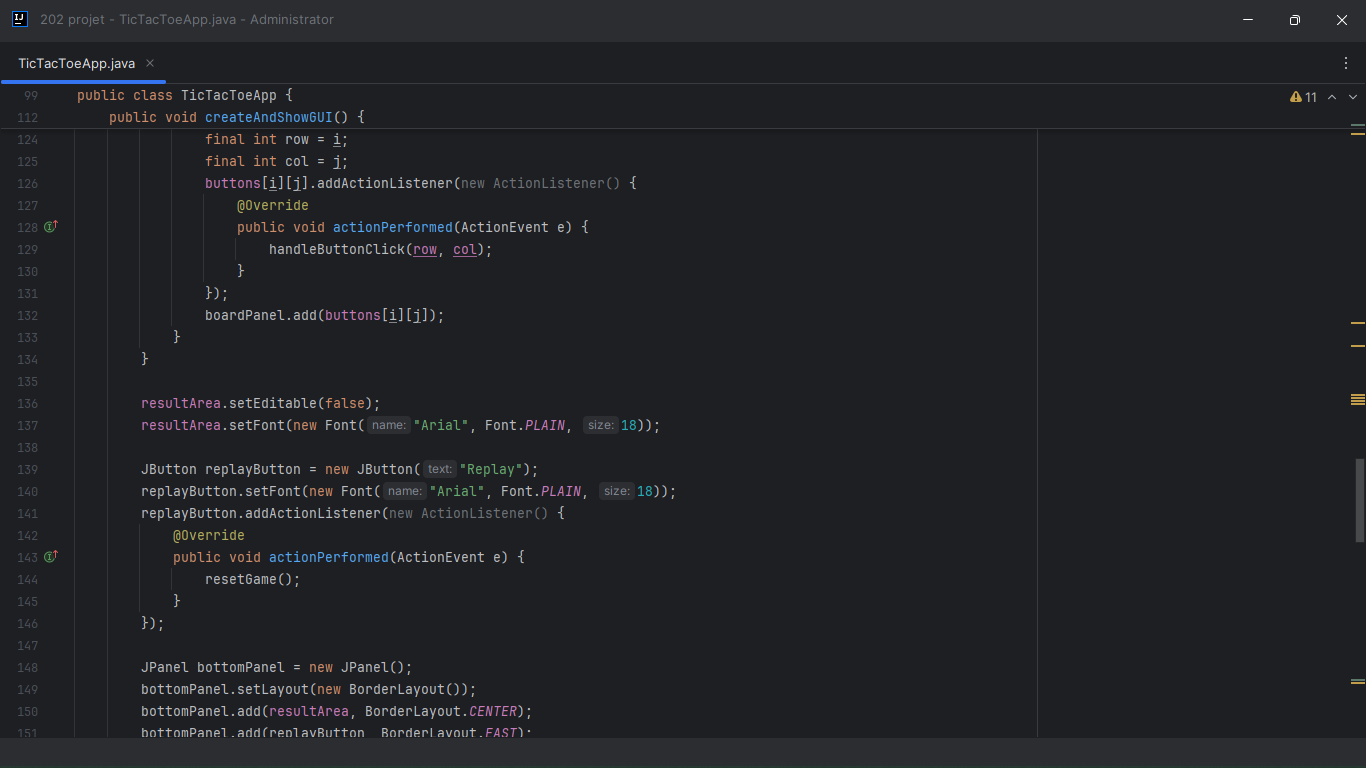
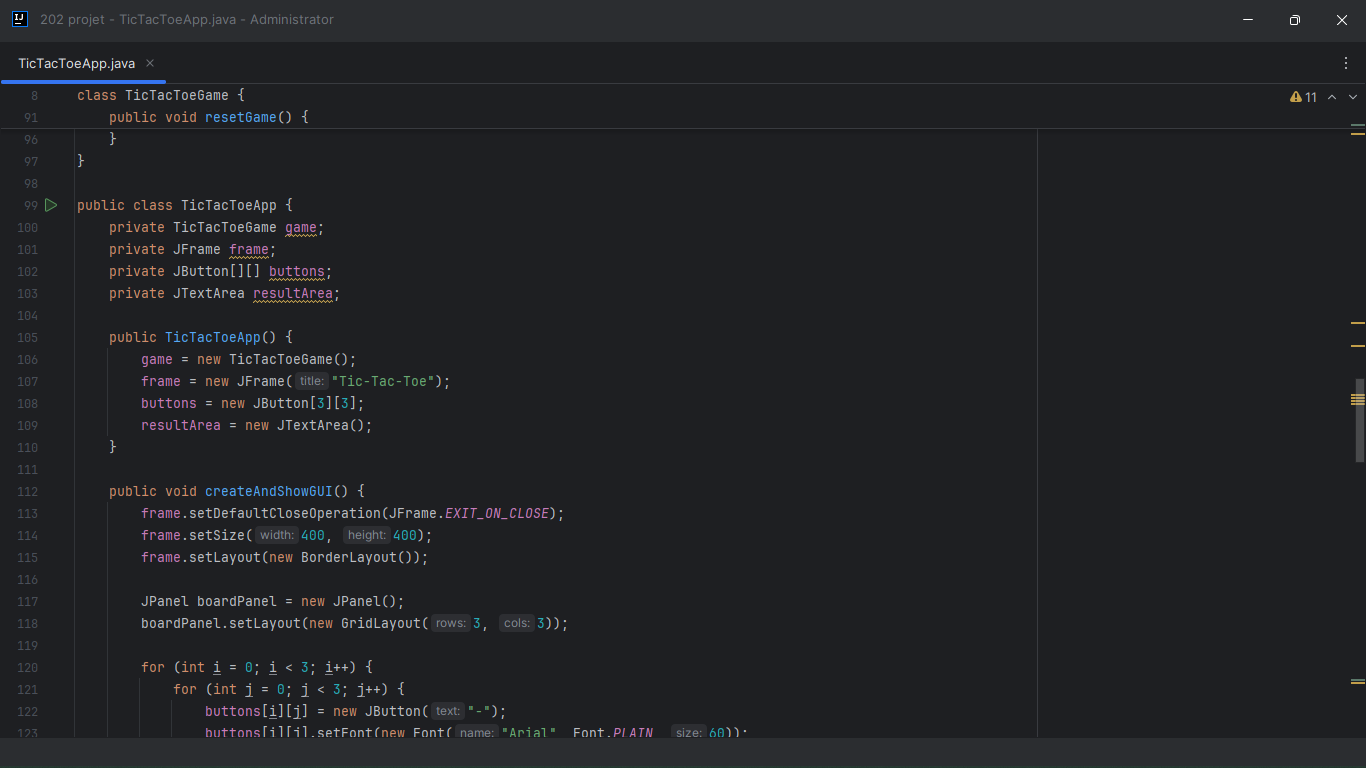
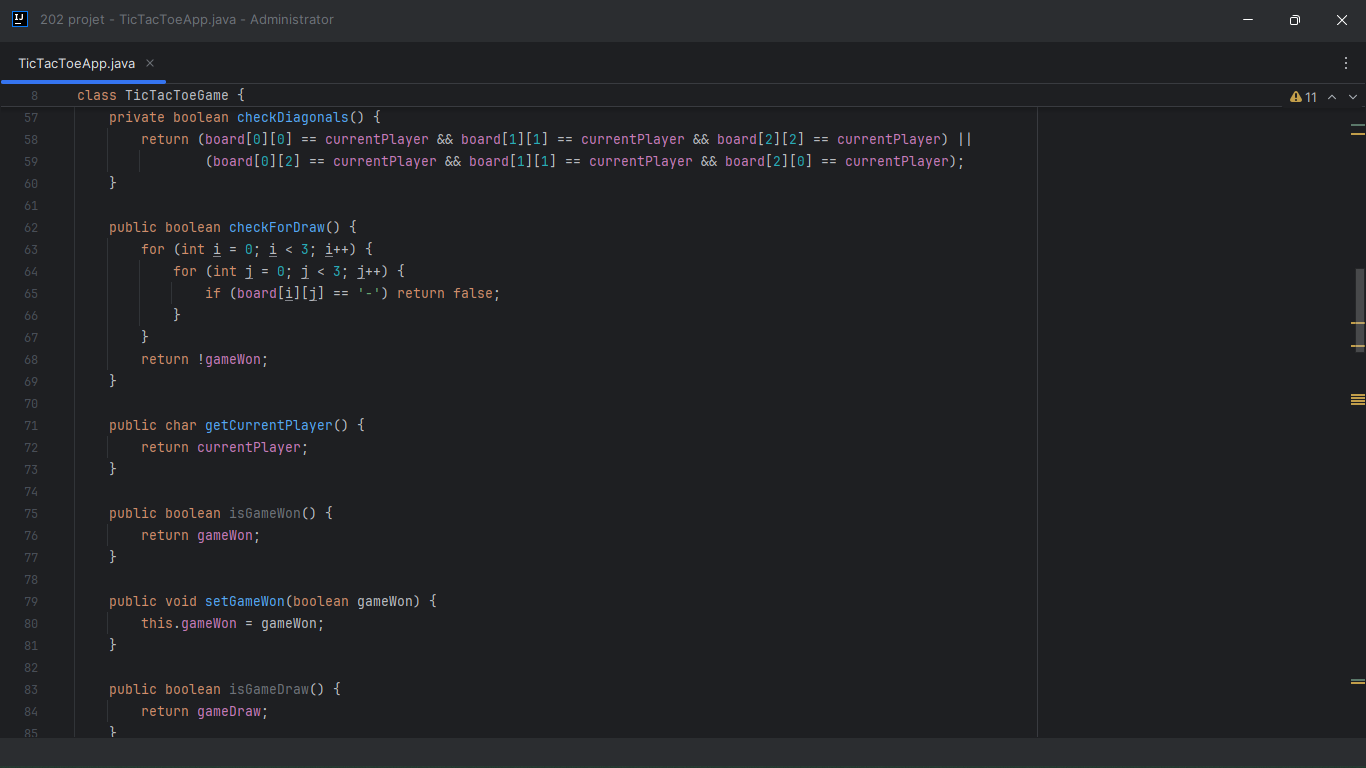
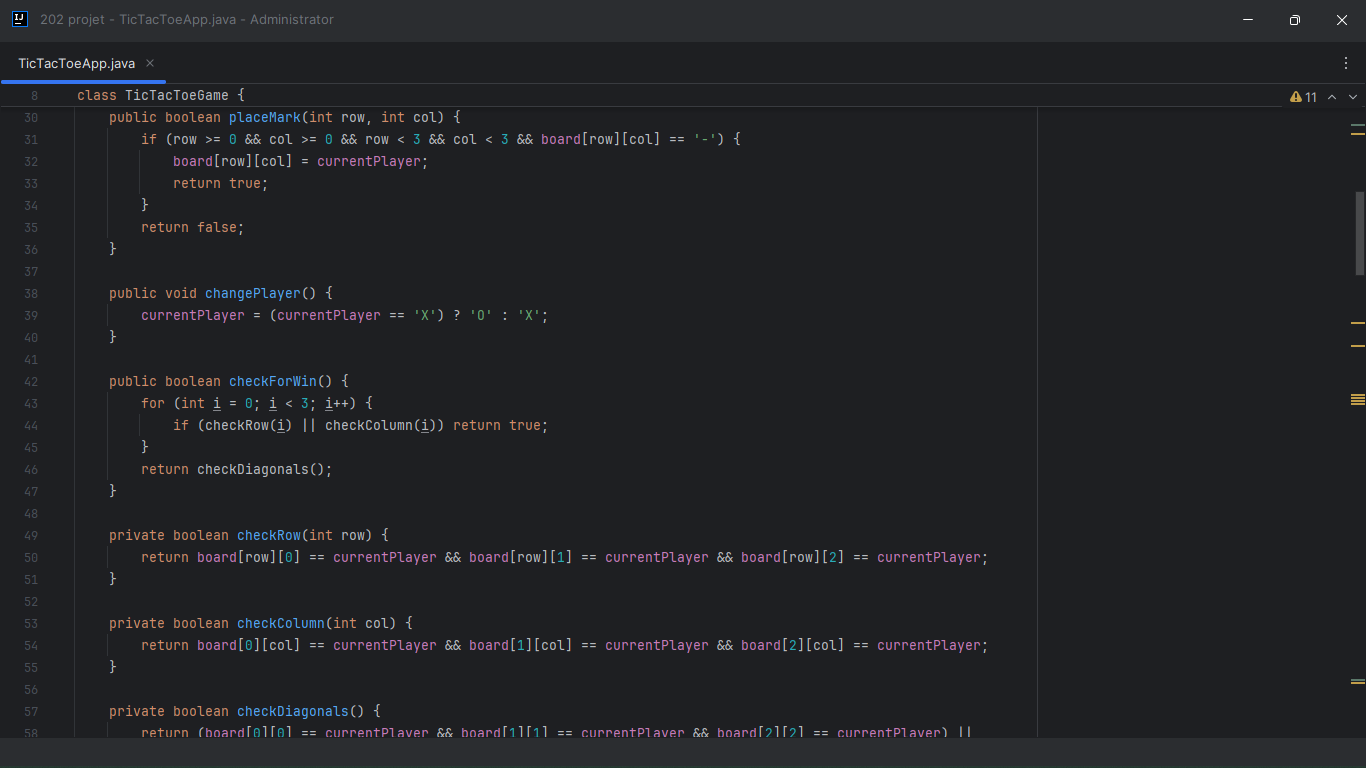
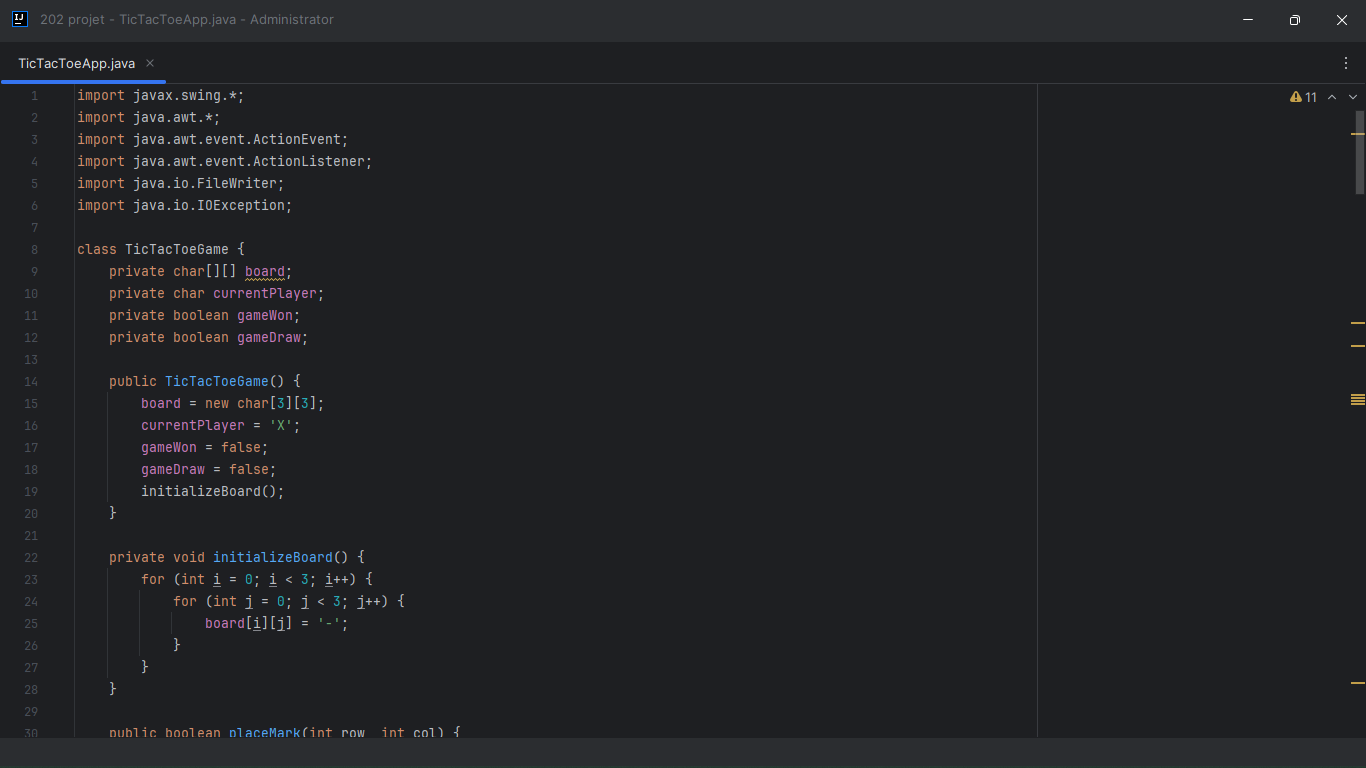
**BHU/22/04/05/0040**

Project Description: Tic-Tac-Toe Game

The Tic-Tac-Toe project is a Java-based implementation of the classic two-player game, utilizing object-oriented principles, Swing for the graphical user interface (GUI), file writing for recording game results, and exception handling for robust error management. The primary goal of the project is to create an interactive, user-friendly game where players can take turns marking a 3x3 grid with their respective symbols (X or O) and determine a winner or if the game ends in a draw. Additionally, the game allows users to replay after each round.

**Key Features:**

1. 3x3 Grid Game Board
   * The game board is a 3x3 grid where two players take turns to place their marks (X or O).
   * The board is represented using a 2D array, which is displayed on the GUI using JButton components.
2. Player Interaction:
   * Players interact with the game by clicking buttons on the grid to place their marks.
   * The game ensures that players can only place their marks on empty cells.
3. Win and Draw Detection:
   * The game checks for winning conditions after each move by evaluating rows, columns, and diagonals.
   * If a player wins, the game displays the result and disables further input on the board.
   * The game also checks for a draw if all cells are filled without any player winning.
4. Replay Functionality:
   * After each game round, players can reset the board and start a new game by clicking the "Replay" button.
   * This feature allows continuous play without restarting the application.
5. Graphical User Interface (GUI):
   * The GUI is built using Java Swing, providing a simple and interactive interface for the players.
   * The main window (JFrame) contains the game board, result display area (JTextArea), and replay button.
6. File Writing:
   * The game results (win or draw) are recorded in a text file named `tic\_tac\_toe\_results.txt`.
   * Each result is appended to the file, ensuring a log of all game outcomes.
7. Exception Handling:
   * The project includes robust exception handling to manage potential I/O errors during file writing.
   * Ensures the application remains stable and user-friendly even when unexpected errors occur.
8. Object-Oriented Design:
   * TicTacToeGame Class:
   * Encapsulates the game logic, including the board state, current player, and methods for placing marks, checking wins/draws, and resetting the game.
9. TicTacToeApp Class:
   * Manages the GUI components, handles user interactions, updates the game state, and integrates the game logic with the GUI.
10. Conclusion:
    * This Tic-Tac-Toe project showcases a well-structured and interactive application built with Java. It combines essential programming concepts such as object oriented design, GUI development, file handling, and exception management, providing a comprehensive learning experience. The replay functionality and persistent result logging enhance the user experience, making the game both enjoyable and informative.



**EHINMOLA ADEDEJI HARRY**

**BHU/22/04/05/0003**

**Core Game Logic and Algorithm Design**

Role: Harry was responsible for designing and implementing the core game logic for the Tic-Tac-Toe game.

Contributions:

Developed the TicTacToeGame class, which encapsulates the game board, current player, and game status.

Implemented methods to initialize the game board, place marks, switch players, and check for win or draw conditions.

Designed the algorithm for checking rows, columns, and diagonals for a win, and for determining a draw.

Ensured the game could reset properly for a new round by implementing the resetGame method.

Focused on maintaining clean code with appropriate encapsulation and method abstraction.

**RHEMA CHOJI DAYLOP**

**BHU/22/04/05/0076**

**Graphical User Interface (GUI) Development**

Rhema was in charge of creating the graphical user interface for the game using Swing.

Contributions:

Developed the TicTacToeApp class, which handles the creation and management of the GUI components.

Set up the main JFrame window and created a JPanel with a GridLayout to represent the game board.

Initialized the 3x3 grid of JButton components, configured their properties, and added action listeners for player interaction.

Created a JTextArea to display the game results and a JButton for replaying the game.

Ensured a user-friendly interface with clear and responsive design elements.

Integrated the core game logic with the GUI to handle player moves, update the board, and display results.

**PELIAH OSIGBMEMHE IDEGBESOR**

**BHU/22/04/05/0040**

**Exception Handling and File Writing**

Peliah focused on implementing exception handling and file writing functionality.

Contributions:

Added a method in the TicTacToeApp class to write game results to a file named tic\_tac\_toe\_results.txt.

Used FileWriter in a try-with-resources statement to ensure proper file handling and closing of resources.

Implemented exception handling to catch and manage IOException scenarios, ensuring the program could gracefully handle file writing errors.

Conducted thorough testing to ensure that game results were correctly written to the file after each round.