UNIVERSITETI® METROPOLITAN TIRANA

FACULTY OF COMPUTER SCIENCES AND IT
Software Engineering Program

Object Oriented Programming

Final Project Documentation
Games using Java

Submitted by: Ajna Nushi
Submitted to: Ph.D Evis Plaku
Msc. Megi Tartari

I. Abstract

I chose for the topic of my project to create games: Tic Tac Toe and Snake Game, because not only do I master my skills in object-oriented programming, but it is also entertaing. The user-friendly design ensures accessibility for all players, aligning with my goal of providing a seamless and enjoyable gaming experience. This project allows me to combine technical skill development with the creative challenges of game design, driven by a deep passion for these classic games. Additionally, I aim to foster a sense of community engagement by encouraging players to share their achievements and experiences with these beloved classics.

II. Key Funcionalities

First, when you run the code, a window will appear that represents a menu from which you can choose one of two games: Tic Tac Toe or the snake game.

- **Main Menu:**

Displays a main menu with options for Snake Game and Tic Tac Toe.
Uses buttons for user interaction.
Game Launch:
Launches the selected game (Snake or Tic Tac Toe) when the corresponding button is clicked.
GUI Styling:
Styles the GUI elements, including buttons and labels.

- **Tic Tac Toe:**

GUI Setup:
Sets up a graphical user interface for the Tic Tac Toe game using Swing components.
Initializes buttons in a 3x3 grid for the game board.

Game Logic:
Implements game logic to handle player turns (X and O).
Checks for win conditions (horizontal, vertical, diagonal).
Checks for a tie if no one wins.

Score Tracking:
Tracks and displays the scores for Player X and Player O.

Restart and Back Buttons:
Implements restart functionality to reset the game.
Implements a "Go Back" button to return to the main menu.

- **Snake Game:**

Game Setup:

Sets up a graphical user interface for the Snake game using Swing components.
Initializes the snake, apple, and other game parameters.

Snake Movement:
Implements snake movement based on user input (arrow keys).
Updates the snake's position and direction.

Apple Generation:
Generates a new apple at a random location when the snake eats one.

Collision Detection:
Checks for collisions with the snake's body and game borders.
Ends the game if a collision is detected.

Score Tracking:
Tracks and displays the score (number of apples eaten).

Game Over Screen:
Displays a game over screen with the final score.
Allows the option to retry the game.

Retry and Back Buttons:
Allows the player to retry the game after a game over.
Implements a "Go Back" button to return to the main menu.

III. Methodology/Technical Part

1. Object-Oriented Design:
The code follows an object-oriented design, organizing functionality into classes and utilizing instances of these classes as objects.
Each game has its dedicated class (TicTacToe and GamePanel for Snake) to encapsulate game logic, GUI components, and event handling.

2. GUI Design using Swing:
Swing components (JFrame, JPanel, JButton, JLabel) are extensively used to create graphical user interfaces for both games.
Components are customized in terms of size, layout, font, color, and behavior.

3. Event-Driven Programming:
Event listeners, such as ActionListener and KeyListener, are implemented to handle user interactions and events.
Button clicks, key presses, and timer events trigger corresponding actions in the games.

4. Game Logic (Tic Tac Toe):

Implements game logic to handle player turns, check for win conditions, and detect ties.
Uses arrays of JButton components to represent the Tic Tac Toe grid.

5. Game Logic (Snake):
Manages the snake's movement, apple generation, collision detection, and scoring.
Utilizes a timer to update the game state at regular intervals, creating animation.
Implements a retry mechanism for restarting the game after a game over.

6. Code Reusability:
Common functionality such as restarting a game, handling scores, and returning to the main menu is modularized and reused across both games.

7. Styling and Aesthetics:
The GUI components are styled to create visually appealing interfaces.
Fonts, colors, and layout are carefully chosen to enhance the user experience.

8. Main Method:
The GameMenu class contains the main method, serving as the entry point for the program.
It creates an instance of the main menu, allowing users to choose between the two games.

9. Random Number Generation:
Random class is used to generate random numbers for determining the first turn in Tic Tac Toe and for placing apples in the Snake game.

10. Flow Control:
Flow control constructs (if, switch, for, while) are used to control the execution flow based on game events, user input, and game state.

11. Lambda Expressions:
Lambda expressions are utilized for concise event handling in certain cases, such as the restart button in Tic Tac Toe.

12. Layout Management:
Layout managers (BorderLayout, GridLayout, FlowLayout) are employed to organize and arrange GUI components in a structured manner.

IV. Summary

The provided Java project implements two simple games - Tic Tac Toe and Snake - with graphical user interfaces using Swing. It demonstrates fundamental object-oriented programming concepts, GUI design, and event-driven programming. The code is organized into classes for each game, facilitating code modularity and reusability.

The things I can improve in my project:

- To add a text file to make it possible to save the scoress for each game.
- Implement a high-score system.
- Improving the visual appeal of the user interface with more refined graphics, layouts, and styling.
- Make an option to choose the level of difficulty.