COMP2396B Object-oriented programming and Java

Assignment 4: A one-player Tic-Tac-Toe Game Due Date: 18th November 2025 23:59

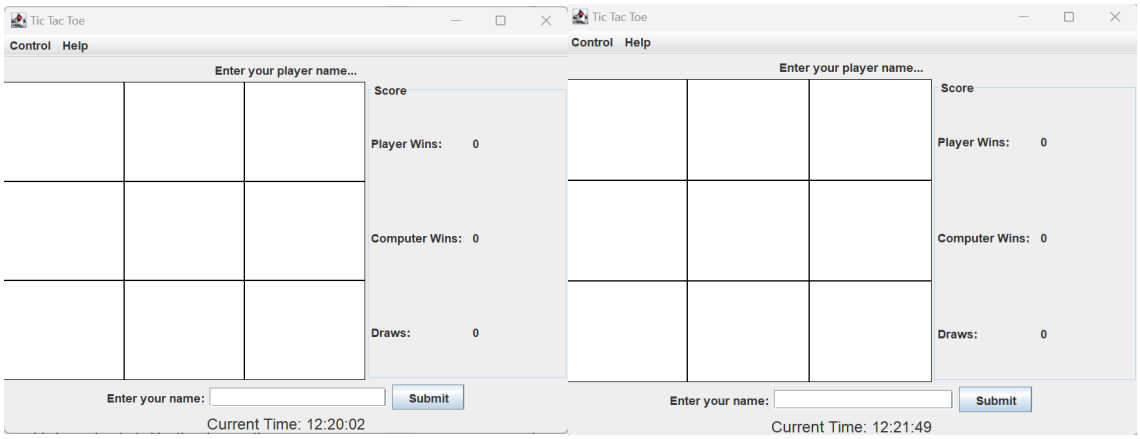
This assignment tests your understanding on GUI programming.

In this assignment, you are going to implement a local one-player Tic-Tac-Toe Game (the user plays with random events generated by the computer). The environment required to implement the game is Eclipse.

Initial setting:

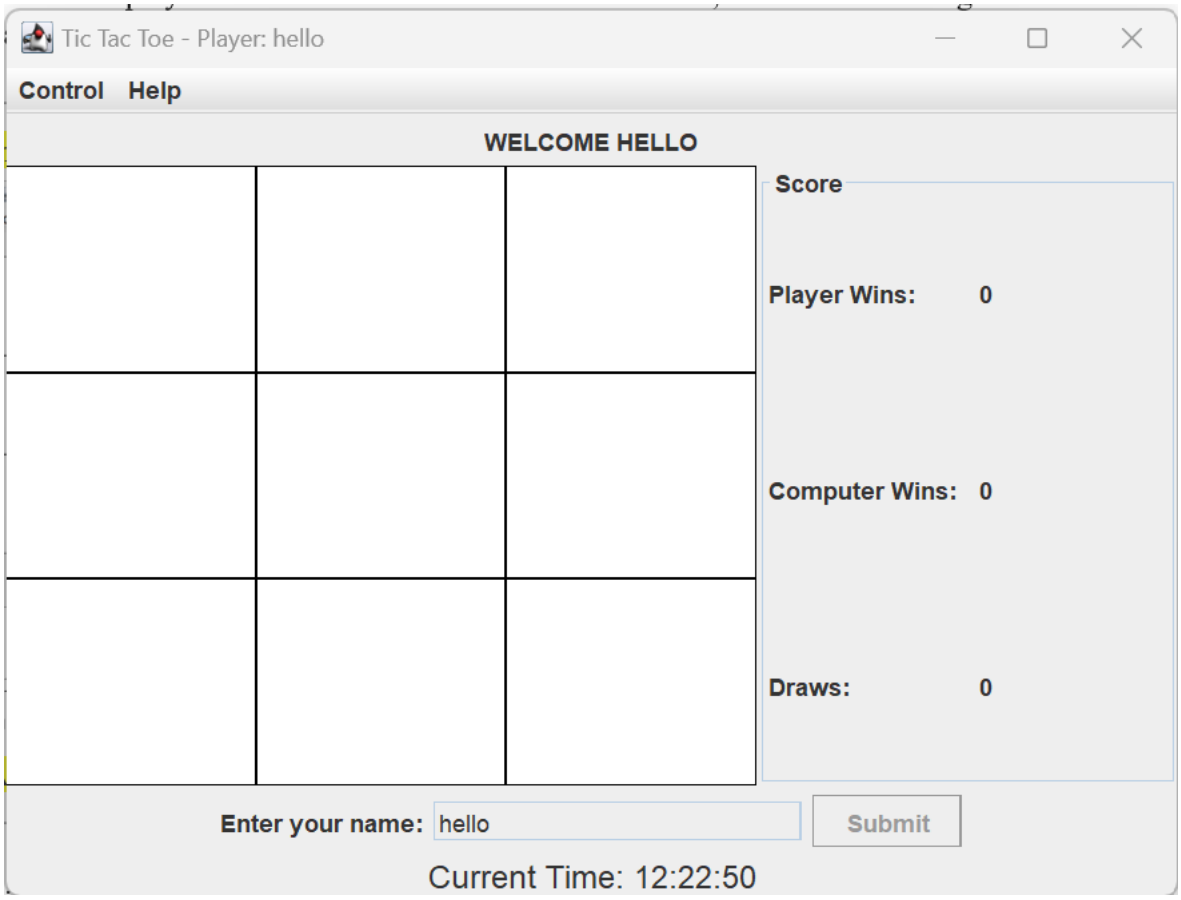
When the game starts, the player needs to input his/her name first (i.e., the player cannot make any move if he/she does not enter his/her player's name) (Fig. 1). The player is not allowed to re-input his/her names again once he/she has submitted his/her names (i.e., the textboxes and the submit buttons should be disabled). In addition, his/her name should be displayed in the frame's title (i.e., change from Tic Tac Toe to Tic Tac Toe-Player: (player's name)) and the message title (located below the menu bar) should change from "Enter your player name…" to "WELCOME (player's name)" (Fig. 2). Moreover, you need to display the current time at the end of the window, the time will change with the game. And you need to record the number of wins between the player and the computer on the right, and the number of draws. Clicking the window's close button (the "X" on the frame) immediately terminates the application. The program exits right away (equivalent to Control → Exit). An unfinished round is abandoned and no win/loss/draw is recorded.

Fig. 1



After player enters his/her name:

Fig. 2



After entering the player's name, the game would always be started by the player rather than the computer (the player's mark with a "x")'s first move (i.e., the computer player (the player's mark with a "o") cannot make its first move until the player makes his/her first move). If the player's move is valid, his/her move would be marked as a "x" on the 3 x 3 board.

Besides, the player is not allowed to make the next move until the computer moves. The computer will make its movement after 2 seconds of the player's movement. The movement of the computer would be mark as a "o" on the 3 x 3 board.

For the Computer Player

After the player makes a valid move, the computer will act 2 seconds later. The computer selects its move uniformly at random from all currently empty cells. If no empty cells remain and no winner has been determined, the round ends in a draw.

For the Turn Messages

- After the player makes a valid move, the message changes to: "Valid move, waiting for your opponent."
- After the computer completes its move, the message changes to: "Your opponent has moved, now is your turn."

These two messages must strictly alternate according to the turn order. The messages must not change on invalid moves.

For the marks "x" and "o", you can either use images or texts to represent them.

Fig. 3

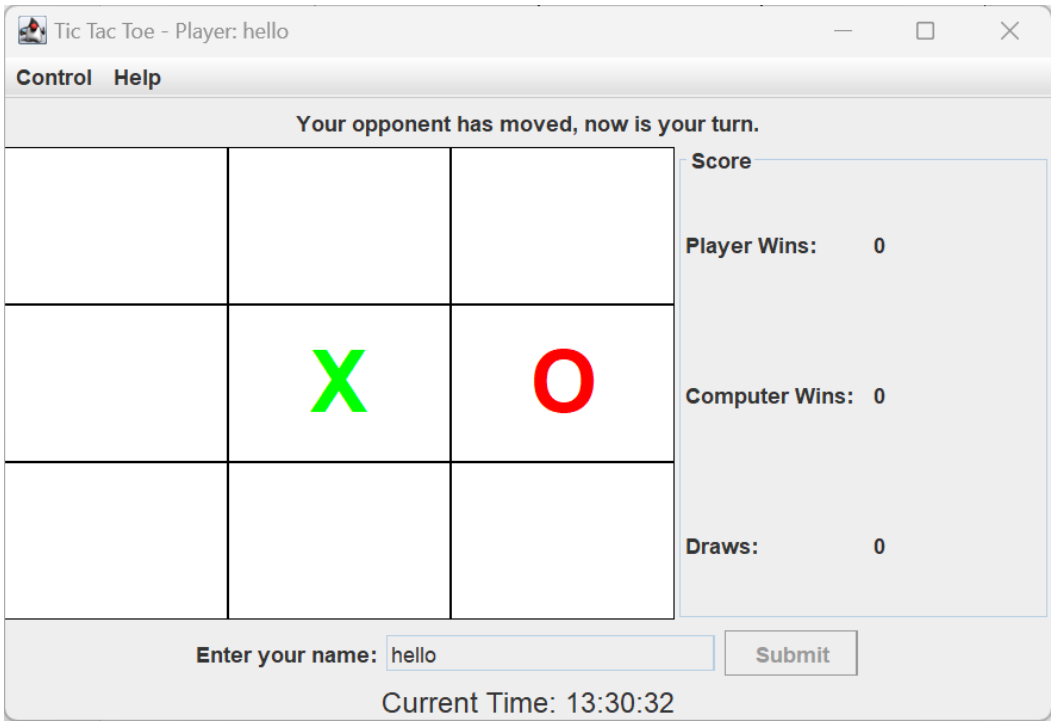
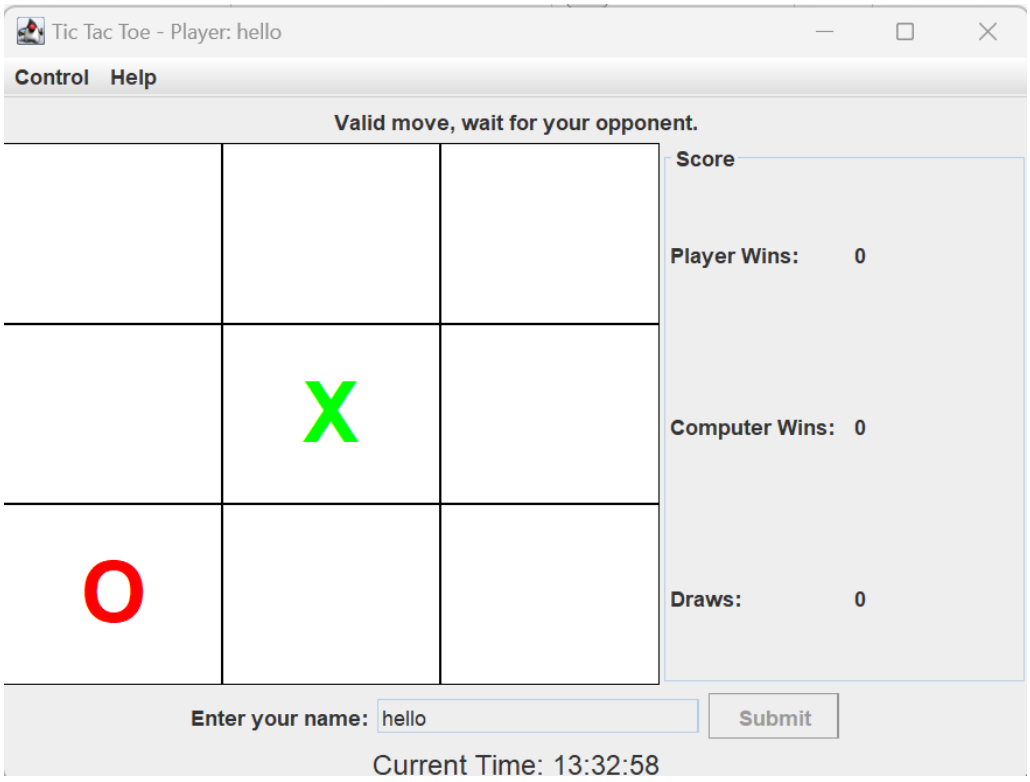


Fig. 4



Criteria for a valid move:

- The move is not occupied by any mark.
- The move is made in the player's turn.

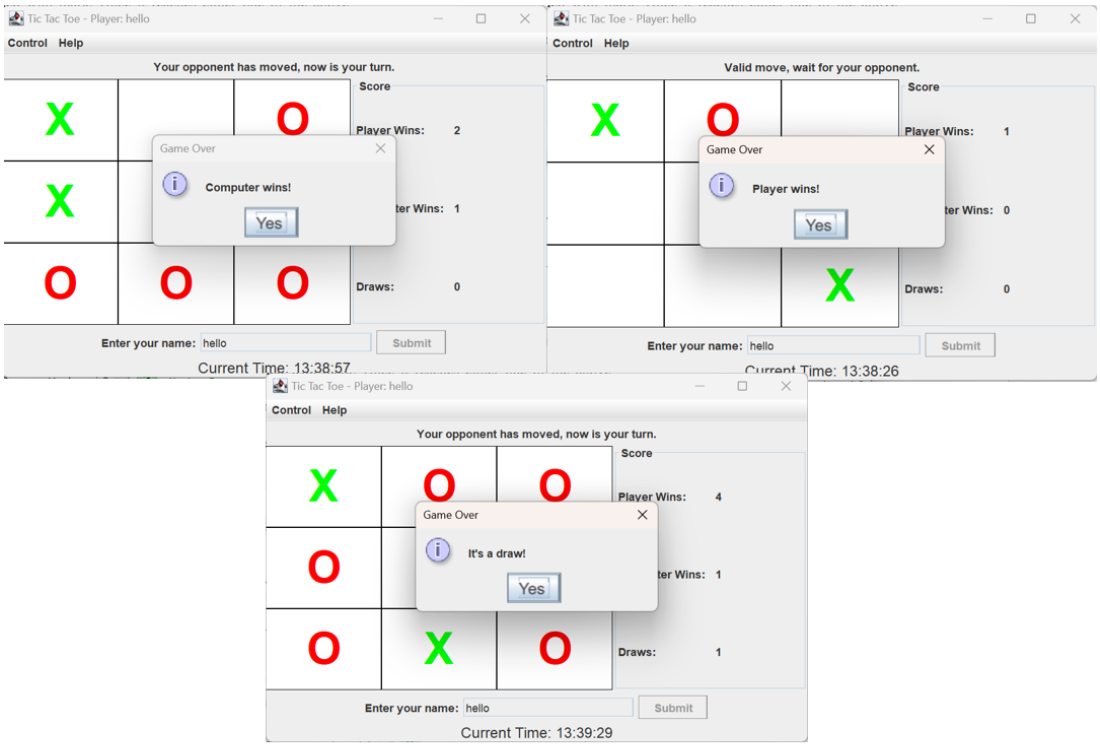
- The move is made within the 3 x 3 board.

The game would continue and switch among the opposite player until it reaches either one of the following conditions:

- Player wins.
- Computer wins.
- Draw.

The winning condition is that when there is any row, column or diagonal that is filled with the same mark (e.g., the player would win if there is any row, column or diagonal that is filled with “x”, the computer would win if there is any row, column or diagonal that is filled with “o”). The game will draw if no players satisfy the winning condition after all the board location is filled with mark. Once it reaches either one of the above conditions, Message Dialog would be displayed (the content of the Message Dialog would be different, based on the condition reached), the player cannot make further move and can click “Yes” to return to the main interface. The following screen captures show the Message Dialog displayed when the player/the computer wins (Fig. 5) or the game is draw (Fig. 5 below). And meanwhile the right JLabel will display the scores of the player and the computer together with the times of the draws. When the user clicks the “Yes” button in the dialog, the game will restart.

Fig.5



Besides, as you can see in the screen captures, there is a JMenuBar which consists of 2 JMenu, named Control and Help (located above the message title). In the JMenu of Control, it consists of a JMenuItem, named Exit while in the JMenu of Help, it consists

of a JMenuItem, named Instruction (Fig. 6 & 7). When the player clicks “Exit”, he/she would exit from the game and the game would be terminated immediately. When the player clicks “Instruction”, a Dialog Frame consists of some game information would be displayed (Fig. 8).

Fig. 6

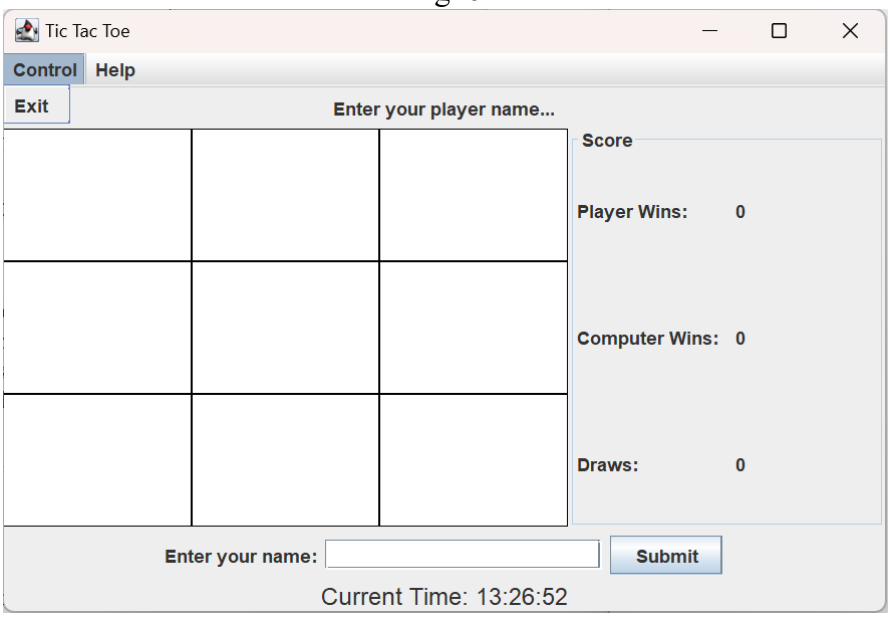


Fig. 7

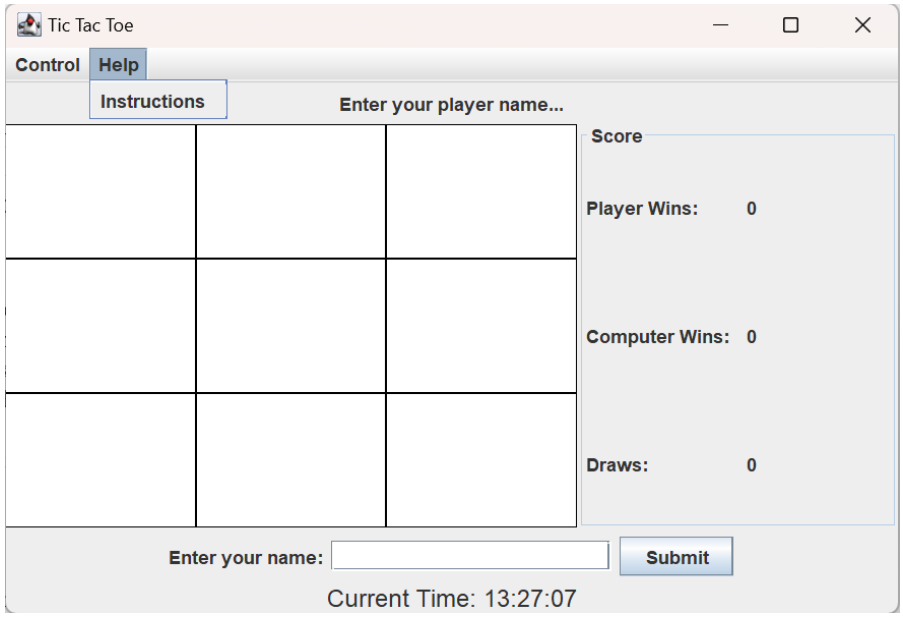
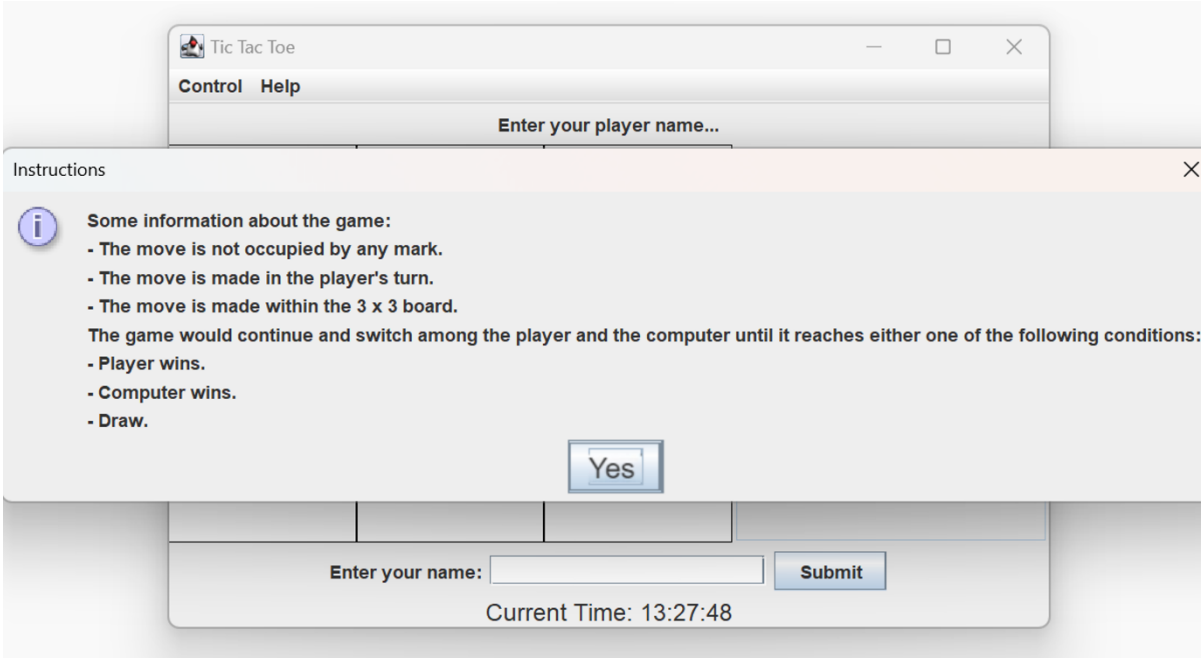


Fig. 8



You are required to write JavaDoc and JavaDoc Scope is listed as below(Required)

1.Every non-private class or interface (file-level type).

2.All public and protected constructors and methods, including event handlers and overridden methods.

3.Include @param, @return, and @throws where applicable.

JavaDoc for private members is optional but recommended if the logic is non-trivial. Trivial getters/setters do not need JavaDoc unless they contain additional logic.

Important notes for the assignment:

1. You can have your own design, but you must include the GUI components as shown in the above screen captures and all the functionalities described in this document should be implemented. To ensure your program has implemented all necessary functions, please refer to the marking scheme below as for your references.

2. This assignment will be marked by features (Your code would not be investigated). You are required to write JavaDoc

3. for all non-private classes and non-private class member functions. Programs without JavaDoc will lead to mark deduction. However, you don't need to generate JavaDoc htmls. Just write comment blocks in your source program.

4. You need to record a demo video of the program. The video should start before you run the program and include all the features your program.

5. After completing the assignment, please submit all files (including demo videos and java files) in a single compressed file (in .zip) to Moodle. Late submission is NOT allowed. Do NOT submit .class files.

6. You will get 0 mark if:
   - You submit .class files instead of .java source files, or
   - You submit java source files that are downloaded from the Internet, or
   - You submit java source files from your classmates, or
   - You submit java source files from friends taken this course last year.

6. Usage of LLM:
   i. Usage of LLMs is strictly prohibited for ALL questions in this assignment.
   ii. All answers must be your own work. When LLM is forbidden for a question, any form of consultation with LLM with regards to that question is treated as plagiarism.

7. You will get 0 mark if:
   - You submit .class files instead of .java source files, or
   - You submit java source files that are downloaded from the Internet, or
   - You submit java source files from your classmates, or
   - You submit java source files from friends taken this course last year.

8. You should also submit all other files (e.g. Image file of the images you used in the. game) that you used in the game under the correct directory). Or else, marks will be deducted.

You are encouraged to keep an optional GitHub private repo for the assignment as a record. Please refer to the guidelines on Moodle for more detail.

Marking Scheme:

Correct implementation of GUI components: - 1 JMenuBar which consists of 2 JMenu which each JMenu consists of its corresponding JMenuItem (6 marks) - 1 message title (2 marks) - 1 3 x 3 tic-tac-toe board (5 marks) - 1 textbox for player's entering his/her name (2 marks) - 1 submit button for submitting the player's name (2 marks) - Three JLabels for recording the scores (3 marks) - 1 Timer for displaying the current time. (2 marks) - 1 JOptionPane for displaying the winner. (2 marks)	Total 24 marks
Correct functionality of the game: - Implementation of restricting players to make their move before they submit their names (5 marks) - Implementation of restricting players to enter and submit their names more than ONCE (5 marks) - Implementation of updating the frame title after players submit their names (5 marks) - Implementation of correct message title after players submit their names and make a valid move (3 marks each, total 6 marks) - Implementation of correct score recording (3 marks each, total 9 marks)	Total 66 marks

- Implementation of displaying the current time (2 marks each)	
- Implementation of correct switching between player and computer player after one of them makes a valid move (2 marks) - Implementation of the game is started by Player's move (5 marks) - Implementation of display player's mark on the board (for both player and the computer) when one of the players makes a valid move (5 marks) - Implementation of NOT display player's mark on the board (for both players) if the player makes an invalid move (5 marks) - Implementation of the 3 conditions: Player wins, Computer wins and Draw (3 marks each, total 9 marks) - Implementation of the functionality of Help (2 marks) - Implementation of the functionality of Restart (2 marks) - Implementation of the functionality of Exit (4 marks)	
JavaDoc	Total 10 marks