PREMIER UNIVERSITY

Department of Computer Science & Engineering

.....

REPORT

Course Code	: CSE211
Course Title	: Object Orienter Programming Lab
Assignment No.	: 01
Assignment Title	: Tic Tac Toe game in java
Date of Submission	: 15. 04.2023

Submitted To: Sourav Adhikary, Lecturer, Premier University

Submitted By:

Name	:Moumita Nag
ID	: 2104010202264
Program	: Bsc in CSE
Batch	: 40th
Section	: C

Remarks:

Title : Developing Tic-Tac-Toe Game using Java Graphics.

Description:

The objective of this assignment is to develop a Tic-Tac-Toe game using Java Graphics. I am required to design and implement a graphical user interface (GUI) version of the game that allows two players to play against each other.

Source Code :

```
package com.mycompany.tictactoe;

import java.awt.GridLayout;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JOptionPane;

import javax.swing.JPanel;


public class Tictactoe extends JPanel {

  char playerSign = 'x';

   int totalCells = 9;

   int totalRows = 3;

   int totalColumns = 3;

   JButton[] jButtons = new JButton[totalCells];
```

```
    public Tictactoe() {
GridLayout ticTacToeGridLayout = new GridLayout(totalRows,
totalColumns);
 setLayout(ticTacToeGridLayout);
 createButtons();
 }

 public void createButtons() {
       for (int i = 0; i <= 8; i++) {
       jButtons[i] = new JButton();
       jButtons[i].setText("");
       jButtons[i].addActionListener(e -> {
      JButton clickedBtn = (JButton) e.getSource();
       clickedBtn.setText(String.valueOf(playerSign));
       clickedBtn.setEnabled(false);
       if (playerSign == 'x')
                playerSign = 'o';
             else
                playerSign = 'x';
           showWinner();
          });
         add(jButtons[i]);
```

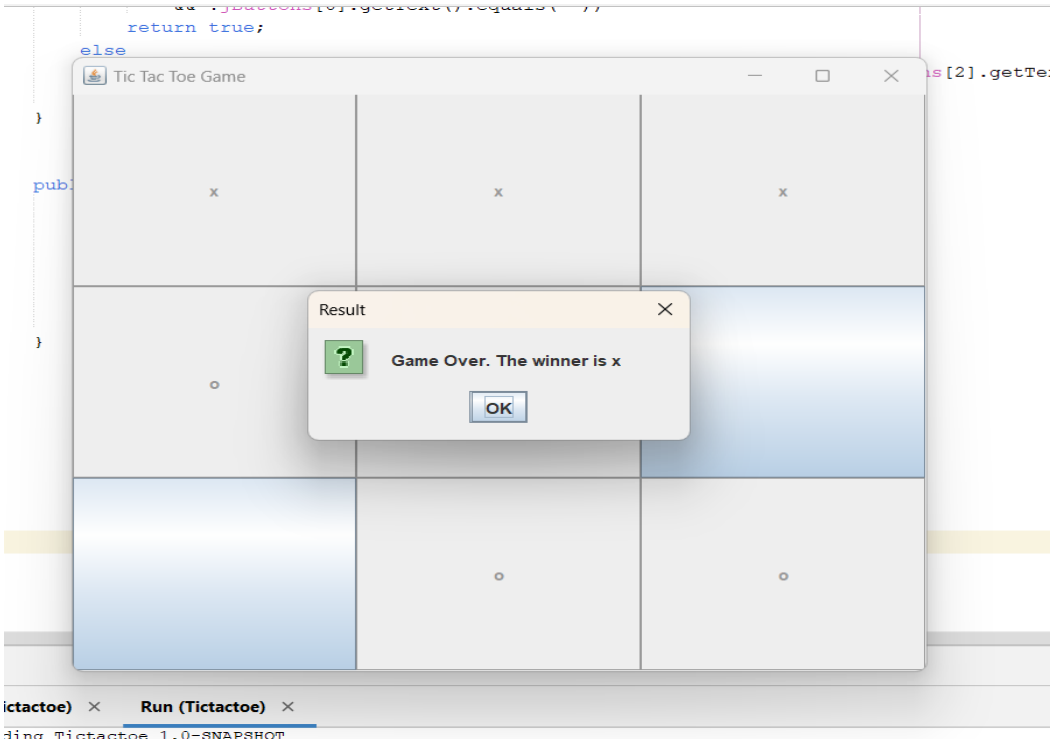
```
      }
    }
 public void showWinner() {
     if (checkForWinner()) {
     if (playerSign == 'x') playerSign = 'o';
          else  playerSign = 'x';
JOptionPane jOptionPane = new JOptionPane();
 int dialog = JOptionPane.showConfirmDialog(jOptionPane, "Game Over. " +
"The       winner is " + playerSign + " ", "Result",
JOptionPane.DEFAULT_OPTION);
       if (dialog == JOptionPane.OK_OPTION)
             System.exit(0);
       } else if (checkIfMatchDraw()) {
          JOptionPane jOptionPane = new JOptionPane();
          int dialog = JOptionPane.showConfirmDialog(jOptionPane, "Game
Draw", "Result", JOptionPane.DEFAULT_OPTION);

          if (dialog == JOptionPane.OK_OPTION)
             System.exit(0);
       }
    }

 public boolean checkIfMatchDraw() {
```

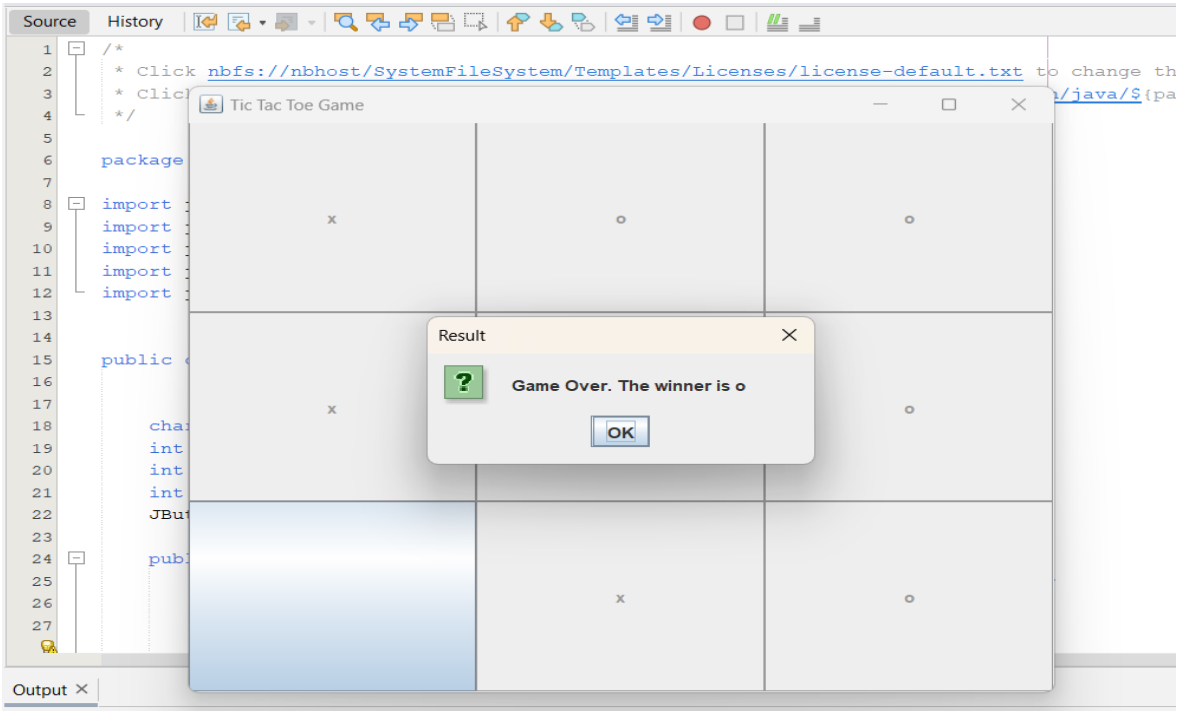
```
        boolean gridsFull = true;
        for (int i = 0; i < totalCells; i++) {
            if (jButtons[i].getText().equals("")) {
                gridsFull = false;
            }
        }
        return gridsFull;
    }
  public boolean checkForWinner() {
        return checkAllRows() || checkAllColumns() || checkTheDiagonals();
    }

    public boolean checkAllRows() {
        int i = 0;
        for (int j = 0; j < 3; j++) {
            if (jButtons[i].getText().equals(jButtons[i + 1].getText()) &&
jButtons[i].getText().equals(jButtons[i + 2].getText())
                    && !jButtons[i].getText().equals("")) {
                return true;
            }
            i = i + 3;
        }
```

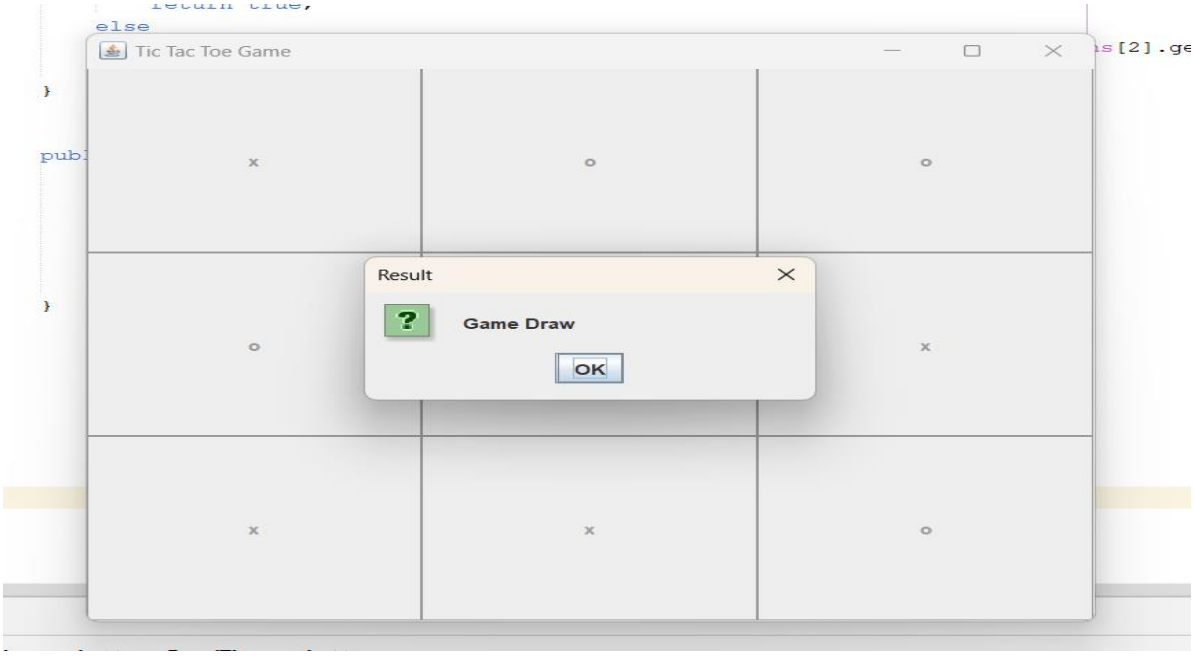
```
        return false;
    }
public boolean checkAllColumns() {
 int i = 0;
        for (int j = 0; j < 3; j++) {
       if (jButtons[i].getText().equals(jButtons[i +3].getText())&&
jButtons[i].getText().equals(jButtons[i + 6].getText())
                  && !jButtons[i].getText().equals("")) {
              return true;
        }
           i++;
        }
        return false;
       }

public boolean checkTheDiagonals() {
       if (jButtons[0].getText().equals(jButtons[4].getText()) &&
jButtons[0].getText().equals(jButtons[8].getText())
              && !jButtons[0].getText().equals(""))
           return true;
       else
           return jButtons[2].getText().equals(jButtons[4].getText()) &&
jButtons[2].getText().equals(jButtons[6].getText())
```

```
                && !jButtons[2].getText().equals("");
    }
    public static void main(String[] args) {
        JFrame jFrame = new JFrame("Tic Tac Toe Game");
        jFrame.getContentPane().add(new Tictactoe());
        jFrame.setBounds(400, 400, 600, 500);
        jFrame.setVisible(true);
        jFrame.setLocationRelativeTo(null);
    }
}
```

Output :




Source
History
Tic Tac Toe Game
Result
Game Over. The winner is o
OK
Output


Tic Tac Toe Game
Result
Game Draw
OK

Details of Implementation :

We gather the things we need to create a GUI representation of the tic tac toe game. First, we need a window that should contain a three-by-three grid. In the below example, we create a class and extend the JPanel to add it to the JFrame in the main() function. We create some class instances that we will use in the game's logic. The first variable is the playerSign that specifies the current sign (either X or O) while playing the game; the second variable, totalCells, is the total number of cells in the grid. Then we have the number of rows and columns to use. For every cell in the grid, we need a button to click and perform our turn, so we create an array of JButton with the size of totalCells. After the initialization part, in the class' constructor, we create an object of GridLayout and pass the totalRows and totalColumns values. Now we set the layout of the JPanel by calling the setLayout() function and passing the GridLayout object in it.

We use addActionListener to listen to every button's click action. We get the click button using e.getSource() and set the current playerSign as its text in the listener. We have to change the sign of the player every time, so we use a condition to change the playerSign.

To check the winner, we need to create three methods; the first method, checkAllRows(), contains a loop that loops through every row and checks the text of every button. The same goes for the checkAllColumns() that checks the text of every column in a loop. In the checkForWinner() method, we check the condition if the result of any of the three methods comes true, then we return true as a result.

Now inside the showWinner() function, we check the result of the checkForWinner() function. We show a popup using JOptionPane, and if the result of checkForWinner() is false, we check if it was a draw or not using the method checkIfMatchDraw().

Challenges and solution :

While making the game I faced some problems. As there are constructors in this game, some errors were showing regarding these. There were some logical errors , for that the program was not running. As I used GUI , there were also some problems regarding GUI during running the program. At first, the box was not showing , after that the parameters I initialized for the box was not proper. So I had to change the size of the box.

For the above problems , I had to run the program again and again. Then after overcoming the problems I got my final output of the game that was running successfully.

Conclution : The above program shows how we can make a Tic-Tac-Toe game in Java . In this game there are X and O , and either of them can be the winner . Otherwise it can be a draw game . Graphical User Interfaces is used to illustrate the game.