LOVELY PROFESSIONAL UNIVERSITY

COURSE NAME : Programming in Java

COURSE CODE: CSE310

PROJECT NAME: TIC TAC TOE GAME

SUBMITTED TO: Dr. Bhimasen Moharana

1. Ashish Gurjar - 12112941
2. K.chaitanya Naga Venkata Subrahmanyam - 12105619
3. Keshav kumar - 12114361

CONTRIBUTION:

Roll No.	Name	Contribution
4	ASHISH GURJAR	GUI Part, Class extended part
9	K. CHAITANYA NAGA VENKATA SUBRAHMANYAM	GUI Part, Main package, Report.
21	KESHAV KUMAR	GUI Part, class Extended, Report

CONTENT:

INTRODUCTION:

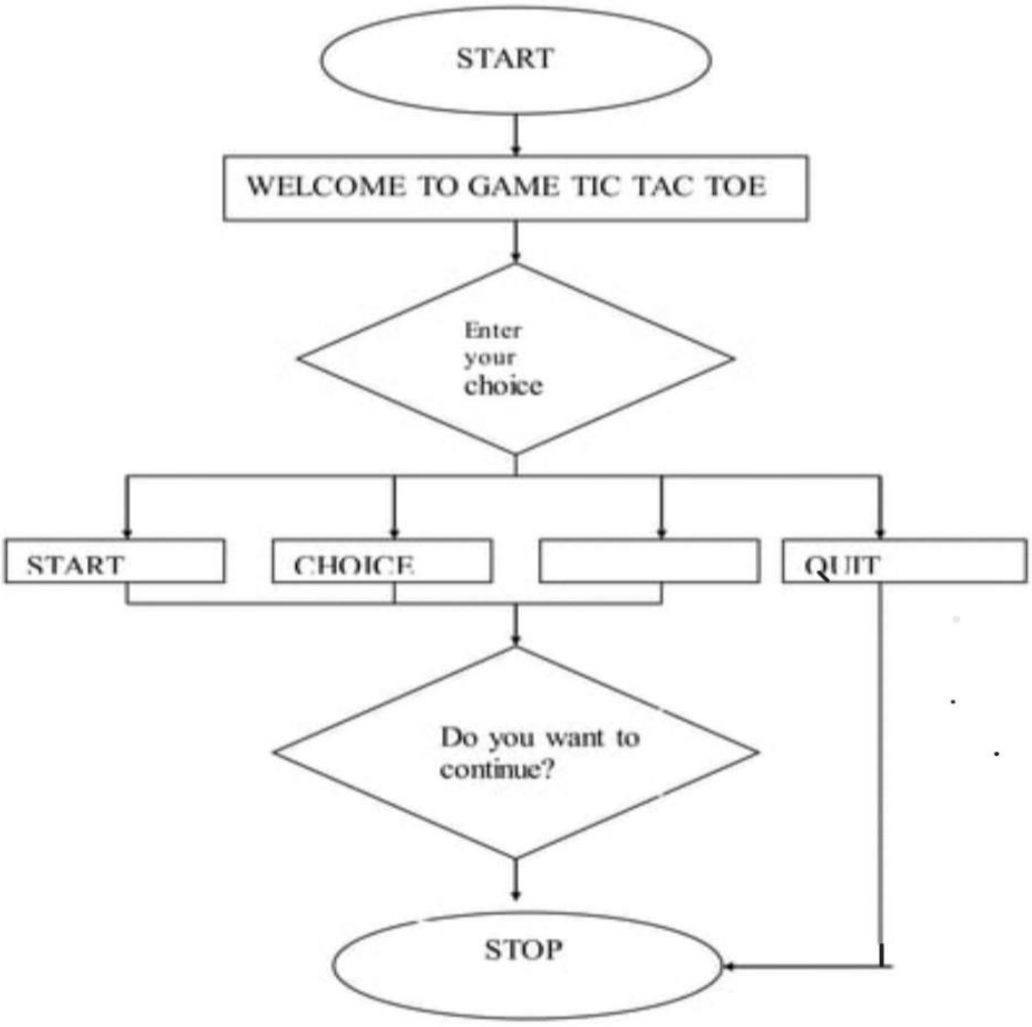
It’s a GUI-based project used with the swing library to organize all the elements that work under the TIC TAC TOE game.

Tic Tac Toe is a classic game that has been enjoyed by people of all ages for decades. It's a two-player game that is played on a 3x3 grid, where each player takes turns placing their symbol (usually X's or O's) on the board.

The objective is to get three of your symbols in a row, either horizontally, vertically, or diagonally, before your opponent does.

It may seem simple, but Tic Tac Toe requires strategic thinking and careful planning to outsmart your opponent

DATA FLOW DIAGRAM:



WORKING OF TIC TAC TOE GAME:

Tic Tac Toe is a two-player game typically played on a 3x3 grid, although larger grids can be used as well. Here are some unique points on how the game works:

1. The game starts with an empty grid.

2. One player uses "X" and the other uses "O".

3. Players take turns placing their symbol in an empty space on the grid.

4. The objective is to get three of your symbols in a row horizontally, vertically, or diagonally.

5. If a player gets three in a row, they win the game.

6. If the grid is filled with no winner, the game is a tie.

7. Players can block their opponent's moves by placing their symbol in a strategic location.

8. The first player to make a move is usually chosen randomly.

9. The game is easy to learn and can be played by people of all ages.

Tic Tac Toe is a great game for developing critical thinking and strategic planning skills

PROJECT DESCRIPTION:

"This is a Java program that implements the game of Tic Tac Toe. The program uses the Swing library to create the GUI components, and the game logic is implemented in the TicTacToe class.

The program creates a JFrame object that contains a title panel and a button panel. The title panel contains a JLabel that displays the text "Tic-Tac-Toe", while the button panel contains 9 JButtons that represent the cells of the game board.

The program uses a boolean variable player1_turn to keep track of which player's turn it is. When a button is clicked, the actionPerformed()""method of the TicTacToe class is called, which checks which button was clicked and updates the game board accordingly.

The program also has a check() method that checks whether any player has won the game. The method

checks all possible winning combinations of X's and O's on the game board, and if a winning combination is found, it calls the xWins() or oWins() method, which displays a message indicating which player has won."

CODE AND OUTPUT:

MAIN :

```
import java.awt.*;

import java.awt.event.*;

import java.util.*;

import javax.swing.*;

public class main {

   public static void main(String[] args) {

      TicTacToe tictactoe = new TicTacToe();

   }

}
```

The main class simply creates an instance of the TicTacToe class, which contains most of the logic and GUI components.

CLASS EXTENDED PART

```
class TicTacToe implements ActionListener {

   Random random = new Random();

   JFrame frame = new JFrame();

   JPanel title_panel = new JPanel();

   JPanel button_panel = new JPanel();

   JLabel textfield = new JLabel();

   JButton[] buttons = new JButton[9];

   boolean player1_turn;

   TicTacToe() {
```

The TicTacToe class extends the ActionListener interface, which means it must implement the actionPerformed() method. This method is called every time a button is clicked,

and it checks which button was clicked and updates the corresponding symbol and turn.

GUI PART

```
TicTacToe() {


    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    frame.setSize(800, 800);

    frame.getContentPane().setBackground(new Color(50, 50, 50));

    frame.setLayout(new BorderLayout());

    frame.setVisible(true);



    textfield.setBackground(new Color(25, 25, 25));

    textfield.setForeground(new Color(25, 255, 0));

    textfield.setFont(new Font("Ink Free", Font.BOLD, 75));
```

```
textfield.setHorizontalAlignment(JLabel.CENTER);

textfield.setText("Tic-Tac-Toe");

textfield.setOpaque(true);



title_panel.setLayout(new BorderLayout());

title_panel.setBounds(0, 0, 800, 100);



button_panel.setLayout(new GridLayout(3, 3));

button_panel.setBackground(new Color(150, 150, 150));



for (int i = 0; i < 9; i++) {

   buttons[i] = new JButton();

   button_panel.add(buttons[i]);

   buttons[i].setFont(new Font("MV Boli", Font.BOLD, 120));

   buttons[i].setFocusable(false);

   buttons[i].addActionListener(this);

}
```

```
    title_panel.add(textfield);
    frame.add(title_panel, BorderLayout.NORTH);
    frame.add(button_panel);

    firstTurn();
}

@Override
public void actionPerformed(ActionEvent e) {

    for (int i = 0; i < 9; i++) {
        if (e.getSource() == buttons[i]) {
            if (player1_turn) {
                if (buttons[i].getText() == "") {
                    buttons[i].setForeground(new Color(255, 0, 0));
                    buttons[i].setText("X");
```

```
                player1_turn = false;
                textfield.setText("O turn");
                check();
            }
        } else {
            if (buttons[i].getText() == "") {
                buttons[i].setForeground(new Color(0, 0, 255));
                buttons[i].setText("O");
                player1_turn = true;
                textfield.setText("X turn");
                check();
            }
        }
    }
}
public void firstTurn() {
```

```
    try {
        Thread.sleep(2000);
    } catch (InterruptedException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }

    if (random.nextInt(2) == 0) {
        player1_turn = true;
        textfield.setText("X turn");
    } else {
        player1_turn = false;
        textfield.setText("O turn");
    }
}

public void check() {
```

```
// check X win conditions
if ((buttons[0].getText() == "X") &&
      (buttons[1].getText() == "X") &&
      (buttons[2].getText() == "X")) {
   xWins(0, 1, 2);
}
if ((buttons[3].getText() == "X") &&
      (buttons[4].getText() == "X") &&
      (buttons[5].getText() == "X")) {
   xWins(3, 4, 5);
}
if ((buttons[6].getText() == "X") &&
      (buttons[7].getText() == "X") &&
      (buttons[8].getText() == "X")) {
   xWins(6, 7, 8);
}
if ((buttons[0].getText() == "X") &&
```

```
        (buttons[3].getText() == "X") &&
        (buttons[6].getText() == "X")) {
    xWins(0, 3, 6);
}
if ((buttons[1].getText() == "X") &&
        (buttons[4].getText() == "X") &&
        (buttons[7].getText() == "X")) {
    xWins(1, 4, 7);
}
if ((buttons[2].getText() == "X") &&
        (buttons[5].getText() == "X") &&
        (buttons[8].getText() == "X")) {
    xWins(2, 5, 8);
}
if ((buttons[0].getText() == "X") &&
        (buttons[4].getText() == "X") &&
        (buttons[8].getText() == "X")) {
```

```
    xWins(0, 4, 8);
}
if ((buttons[2].getText() == "X") &&
        (buttons[4].getText() == "X") &&
        (buttons[6].getText() == "X")) {
    xWins(2, 4, 6);
}
// check O win conditions
if ((buttons[0].getText() == "O") &&
        (buttons[1].getText() == "O") &&
        (buttons[2].getText() == "O")) {
    oWins(0, 1, 2);
}
if ((buttons[3].getText() == "O") &&
        (buttons[4].getText() == "O") &&
        (buttons[5].getText() == "O")) {
    oWins(3, 4, 5);
```

```
}
if ((buttons[6].getText() == "O") &&
      (buttons[7].getText() == "O") &&
      (buttons[8].getText() == "O")) {
   oWins(6, 7, 8);
}
if ((buttons[0].getText() == "O") &&
      (buttons[3].getText() == "O") &&
      (buttons[6].getText() == "O")) {
   oWins(0, 3, 6);
}
if ((buttons[1].getText() == "O") &&
      (buttons[4].getText() == "O") &&
      (buttons[7].getText() == "O")) {
   oWins(1, 4, 7);
}
if ((buttons[2].getText() == "O") &&
```

```
            (buttons[5].getText() == "O") &&
            (buttons[8].getText() == "O")) {
        oWins(2, 5, 8);
    }
    if ((buttons[0].getText() == "O") &&
            (buttons[4].getText() == "O") &&
            (buttons[8].getText() == "O")) {
        oWins(0, 4, 8);
    }
    if ((buttons[2].getText() == "O") &&
            (buttons[4].getText() == "O") &&
            (buttons[6].getText() == "O")) {
        oWins(2, 4, 6);
    }
}
public void xWins(int a, int b, int c) {
    buttons[a].setBackground(Color.GREEN);
```

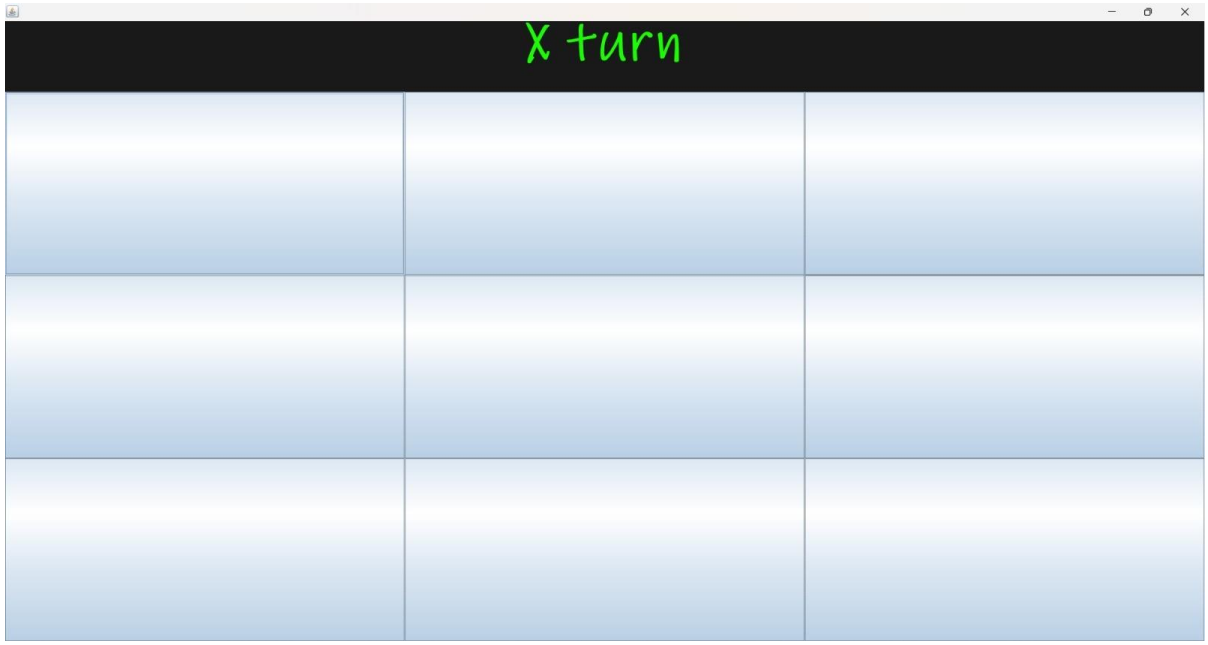
```
    buttons[b].setBackground(Color.GREEN);
buttons[c].setBackground(Color.GREEN);
    for (int i = 0; i < 9; i++) {
        buttons[i].setEnabled(false);
    }
    textfield.setText("X wins");
}
public void oWins(int a, int b, int c) {
    buttons[a].setBackground(Color.GREEN);
    buttons[b].setBackground(Color.GREEN);
    buttons[c].setBackground(Color.GREEN);

    for (int i = 0; i < 9; i++) {
        buttons[i].setEnabled(false);
    }
    textfield.setText("O wins");
}
```

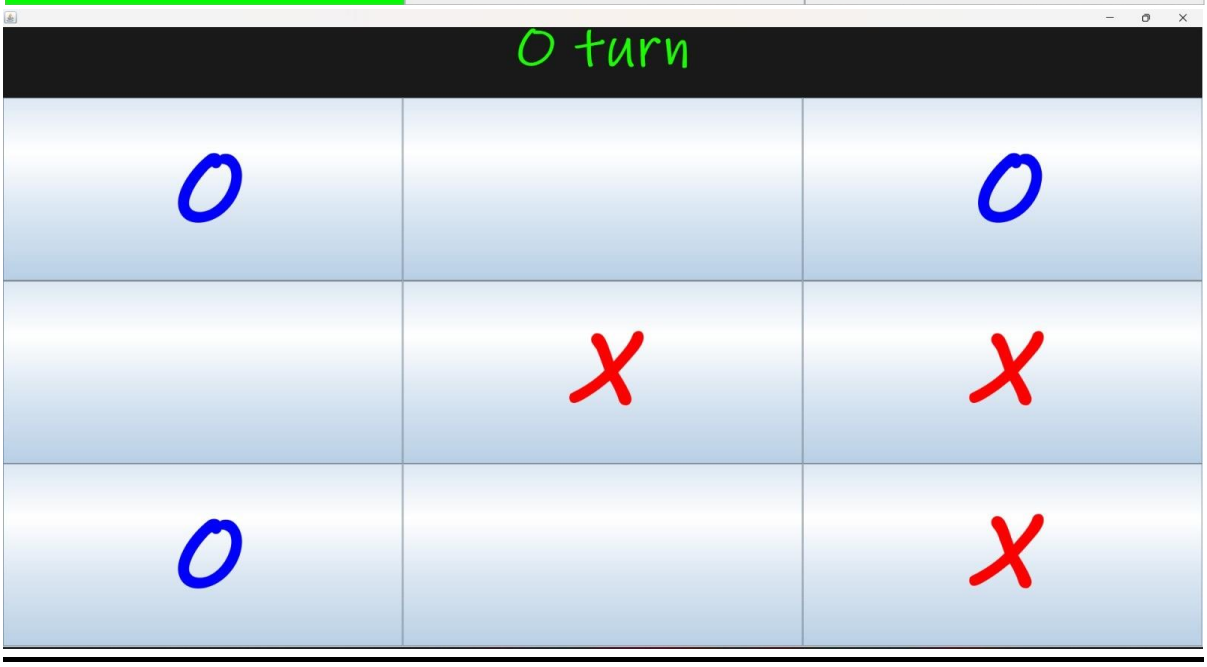
}

The constructor of the TicTacToe class initializes the GUI components, such as the frame, panels, labels, and buttons. It also sets the font, color, and layout of the components. The firstTurn() method randomly selects which player goes first and sets the turn label accordingly. The check() method checks all possible win conditions by comparing the text of each button and calls the xWins() or oWins() methods with the winning indices if a win is found. The xWins() and oWins() methods change the color of the winning buttons and display the winning message.

OUTPUT:




X wins
O
X
X
O
X
O


O turn
O
O
X
X
O
X

CONCLUSION:

The Tic Tac Toe game implemented in Java provides a simple yet fun user experience. The game mechanics are well-defined and the graphical user interface is user-friendly.

The code implements the necessary logic to determine a winner or a draw, and the game can be played by two players taking turns using X and O markers. Overall, the game is a great exercise in object-oriented programming and can be expanded upon to add additional features or improve the user interface.

THE END