

TIC-TAC-TOE

A PROJECT REPORT

Submitted by

RAVIN S (2303811724321087)

in partial fulfillment of requirements for the award of the course

CGB1201 – JAVA PROGRAMMING

in

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by
AICTE, New Delhi)

SAMAYAPURAM – 621 112

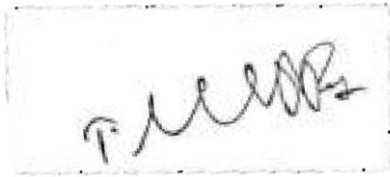
DECEMBER, 2024

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on “ **TIC-TAC-TOE**” is the bonafide work of **RAVIN S (2303811724321087)** who carried out the project work during the academic year 2024 - 2025 under my supervision.



Signature

Dr. T. AVUDAIAPPAN M.E., Ph.D.,

HEAD OF THE DEPARTMENT,

Department of Artificial Intelligence,

K. Ramakrishnan College of Engineering,

Samayapuram, Trichy -621 112.



Signature

Mrs. S. GEETHA M.E.,


SUPERVISOR,

Department of Artificial Intelligence,

K. Ramakrishnan College of Engineering,

Samayapuram, Trichy -621 112.

Submitted for the viva-voce examination held on 3.12.24



INTERNAL EXAMINER



EXTERNAL EXAMINER

DECLARATION

I declare that the project report on “**TIC-TAC-TOE** ” is the result of original work done by me and best of my knowledge, similar work has not been submitted to “**ANNA UNIVERSITY CHENNAI**” for the requirement of Degree of **BACHELOR OF TECHNOLOGY**. This project report is submitted on the partial fulfillment of the requirement of the award of the **CGB1201 – JAVA PROGRAMMING**.

S. Ravin

Signature

RAVIN S

Place: Samayapuram

Date: 3/12/2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and indebtedness to our institution, “**K. Ramakrishnan College of Technology (Autonomous)**”, for providing us with the opportunity to do this project.

I extend our sincere acknowledgement and appreciation to the esteemed and honourable Chairman, **Dr. K. RAMAKRISHNAN, B.E.**, for having provided the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director, **Dr. S. KUPPUSAMY, MBA, Ph.D.**, for forwarding our project and offering an adequate duration to complete it.

I would like to thank **Dr. N. VASUDEVAN, M.TECH., Ph.D.**, Principal, who gave the opportunity to frame the project to full satisfaction.

I thank **Dr.T.AVUDAIAPPAN, M.E.,Ph.D.**, Head of the Department of **ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**, for providing her encouragement in pursuing this project.

I wish to convey our profound and heartfelt gratitude to our esteemed project guide **Mrs.S.GEETHA M.E.**, Department of **ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**, for her incalculable suggestions, creativity, assistance and patience, which motivated us to carry out this project.

I render our sincere thanks to the Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress

VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards.

MISSION OF THE INSTITUTION

- Be a centre of excellence for technical education in emerging technologies by exceeding the needs of industry and society.
- Be an institute with world class research facilities.
- Be an institute nurturing talent and enhancing competency of students to transform them as all- round personalities respecting moral and ethical values.

VISION AND MISSION OF THE DEPARTMENT

To excel in education, innovation and research in Artificial Intelligence and Data Science to fulfill industrial demands and societal expectations.

Mission 1: To educate future engineers with solid fundamentals, continually improving teaching methods using modern tools.

Mission 2: To collaborate with industry and offer top-notch facilities in a conducive learning environment.

Mission 3: To foster skilled engineers and ethical innovation in AI and Data Science for global recognition and impactful research.

Mission 4: To tackle the societal challenge of producing capable professionals by instilling employability skills and human values.

PROGRAM EDUCATIONAL OBJECTIVES (PEOS)

PEO 1: Compete on a global scale for a professional career in Artificial Intelligence and Data Science.

PEO 2: Provide industry-specific solutions for the society with effective communication and ethics.

PEO 3: Hone their professional skills through research and lifelong learning initiatives.

PROGRAM OUTCOMES

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

- **PSO 1:** Capable of working on data-related methodologies and providing industry-focussed solutions.
- **PSO2:** Capable of analysing and providing a solution to a given real-world problem by designing an effective program.

ABSTRACT

This project implements a console-based Tic-Tac-Toe game in Java, allowing two players to compete on a 3x3 grid by alternately placing 'X' and 'O'. The program initializes the grid, validates inputs, updates the board, and checks for a winner or draw after each move. It demonstrates the use of Java concepts such as arrays for board representation, methods for modularity, loops for game flow, and conditionals for decision-making. Exception handling ensures smooth gameplay by managing invalid inputs. While the game effectively handles two-player functionality, future enhancements like a single-player mode with AI and a graphical interface could further improve it. This project serves as a foundation for understanding Java programming and basic game logic.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No.
	ABSTRACT	viii
1	INTRODUCTION	1
	1.1 INTRODUCTION	1
	1.2 OBJECTIVE	1
2	PROJECT METHODOLOGY	2
	2.1 PROPOSED WORK	2
	2.2 BLOCK DIAGRAM	2
3	JAVA PROGRAMMING CONCEPTS	3
	3.1 OBJECT-ORIENTED PROGRAMMING(OOP)	3
	3.2 JAVA SWING FOR GUI DEVELOPMENT	3
4	MODULE DESCRIPTION	4
	4.1 BOARD INITIALIZATION	4
	4.2 PRINT BOARD	4
	4.3 INPUT VALIDATION	4
	4.4 TURN MANAGEMENT	4
	4.5 WINNER CHECKING	4
5	CONCLUSION	5
	REFERENCES	6
	APPENDICES	7
	APPENDIX A – SOURCE CODE	7
	APPENDIX B – SCREEN SHOTS	11

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

This Java program implements a simple **Tic Tac Toe** game using the **Swing** library for the graphical user interface (GUI). The game is designed with a 3x3 grid of buttons, allowing two players to alternately play as 'X' and 'O'. Players can click on the grid to make their moves, and the program detects wins or draws dynamically. A message box announces the winner or a draw, and the game resets for a new round. The `ButtonClickListener` class handles user interactions, while methods like `checkWin()` and `isDraw()` ensure gameplay logic.

1.2 OBJECTIVE

The objective of this project is to design and develop a console-based Tic-Tac-Toe game in Java. The game provides an interactive platform for two players to compete by placing their marks ('X' or 'O') on a 3x3 grid. It ensures smooth gameplay with input validation, turn management, and winner detection. This project aims to demonstrate the practical application of Java concepts like arrays, loops, and conditionals.

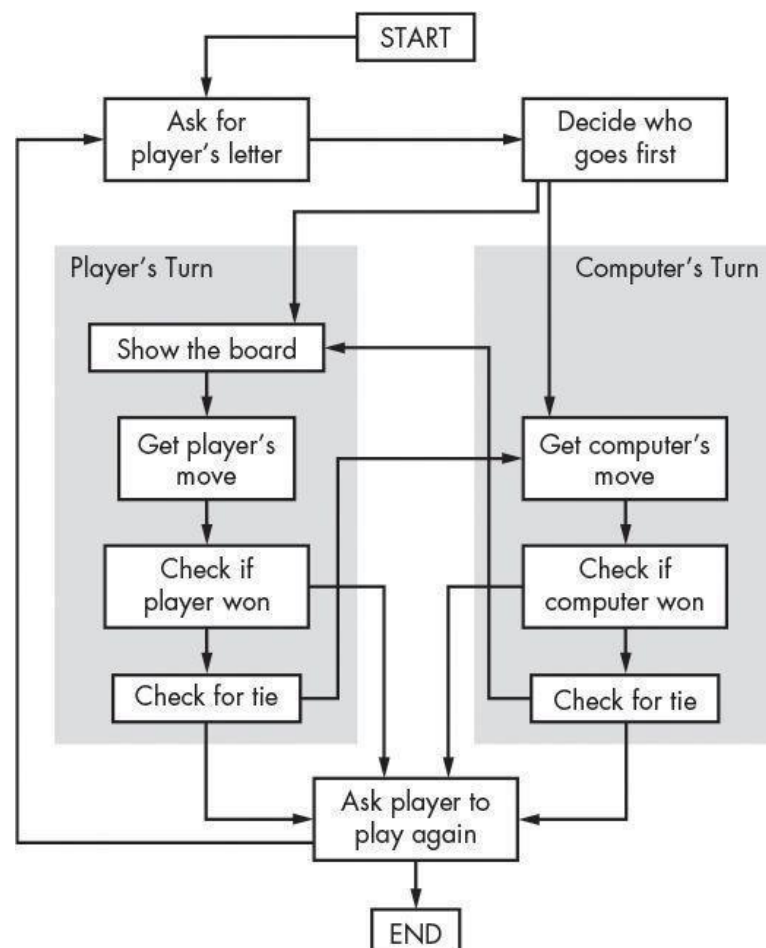
CHAPTER 2

PROJECT METHODOLOGY

2.1 PROPOSED WORK

The Tic-Tac-Toe game is designed as a console-based application with a modular structure to ensure efficient gameplay. It begins with board initialization and displays the grid for user interaction. Players take turns selecting slots, with input validation ensuring valid moves. The program checks for a winner or draw after each move and updates the grid accordingly. The game flow is managed through a loop until a game-ending condition is met. Future enhancements could include AI- based single-player mode and a graphical interface for improved usability.

2.2 BLOCK DIAGRAM



CHAPTER 3

JAVA PROGRAMMING CONCEPTS

3.1 OBJECT-ORIENTED PROGRAMMING (OOP)

Class Design:

The TicTacToeSwing class encapsulates the logic and GUI of the game.

Encapsulation:

Private fields (frame, buttons, currentPlayer) ensure data is protected from external interference.

Abstraction:

Methods like createAndShowGUI(), checkWin(), and resetGame() abstract the implementation details.

Inner Classes:

The ButtonClickListener is a private nested class, encapsulating event-handling logic specific to each button.

3.2 SWING

Swing Components:

JFrame, JButton, and JOptionPane are used to create and manage the graphical user interface.

Layouts:

GridLayout is used to arrange buttons in a 3x3 grid.

Event Handling:

ActionListener is used to detect button clicks and trigger appropriate actions

CHAPTER 4

MODULE DESCRIPTION

4.1 BOARD INITIALIZATION

This module initializes the 3x3 game board as a 1D array, assigning numbers to represent available slots. It prepares the board for gameplay.

4.2 PRINT BOARD

Displays the current state of the board in a visually clear format after each turn, showing occupied slots and available positions.

4.3 INPUT VALIDATION

Ensures the player's input is valid, checking that it falls within the range and the selected slot is unoccupied, prompting re-entry if invalid.

4.4 TURN MANAGEMENT

Manages player turns, alternates between 'X' and 'O', and ensures smooth gameplay by switching turns after each valid move.

4.5 WINNER CHECKING

Evaluates the board after every move to check rows, columns, and diagonals for a winning combination or determines if the game ends in a draw.

CHAPTER 5

CONCLUSION

The Tic-Tac-Toe game in Java successfully demonstrates the application of core programming concepts like arrays, loops, conditionals, and modular methods. It provides an interactive two-player experience, efficiently managing input validation, gameplay flow, and winner detection. The program is designed with clarity and simplicity, ensuring ease of understanding and usability.

The game effectively handles edge cases, such as invalid inputs and slot availability, through exception handling. Its modular structure ensures maintainability and opens up possibilities for future enhancements, such as adding a single-player mode with AI, replay functionality, or a graphical interface.

In summary, this project achieves its goal of showcasing Java fundamentals while delivering a functional game. It serves as a strong foundation for understanding programming logic and can be extended to develop more advanced features.

REFERENCES:

Official Java Documentation

<https://docs.oracle.com/javase/>

GeeksforGeeks - Java Programming Tutorials

<https://www.geeksforgeeks.org/java/>

Java Tutorials by Tutorialspoint

<https://www.tutorialspoint.com/java/>

Java Programming for Beginners - Programiz

<https://www.programiz.com/java-programming>

Learn Java by W3Schools

<https://www.w3schools.com/java/>

Codecademy Java Course

<https://www.codecademy.com/learn/learn-java>

Java Programming Tutorials by Javatpoint

<https://www.javatpoint.com/java-tutorial>

APPENDICES

APPENDIX-A SOURCE CODE

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class TicTacToeSwing {

    private JFrame frame;
    private JButton[][] buttons = new JButton[3][3];
    private char currentPlayer = 'X';

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new TicTacToeSwing().createAndShowGUI());
    }

    // Create and display the game GUI
    public void createAndShowGUI() {
        frame = new JFrame("Tic Tac Toe");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(new GridLayout(3, 3));

        // Initialize the grid buttons
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                buttons[i][j] = new JButton(" ");
                buttons[i][j].setFont(new Font("Arial", Font.BOLD, 40));
                buttons[i][j].setFocusPainted(false);
            }
        }
    }
}
```



```

        buttons[i][j].addActionListener(new ButtonClickListener(i, j));
        frame.add(buttons[i][j]);
    }
}

frame.setSize(400, 400);
frame.setVisible(true);
}

// Handle button click events
private class ButtonClickListener implements ActionListener {
    private int row, col;

    public ButtonClickListener(int row, int col) {
        this.row = row;
        this.col = col;
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        if (buttons[row][col].getText().equals(" ")) {
            buttons[row][col].setText(String.valueOf(currentPlayer));
            if (checkWin()) {
                JOptionPane.showMessageDialog(frame, "Player " + currentPlayer + "
wins!");
                resetGame();
            } else if (isDraw()) {
                JOptionPane.showMessageDialog(frame, "It's a draw!");
                resetGame();
            } else {

```

```

        currentPlayer = (currentPlayer == 'X') ? 'O' : 'X';
    }
} else {
    JOptionPane.showMessageDialog(frame, "Invalid move. Try again.");
}
}
}
}

```

// Check if a player has won

```

private boolean checkWin() {
    for (int i = 0; i < 3; i++) {
        if (buttons[i][0].getText().equals(String.valueOf(currentPlayer)) &&
            buttons[i][1].getText().equals(String.valueOf(currentPlayer)) &&
            buttons[i][2].getText().equals(String.valueOf(currentPlayer))) {
            return true;
        }
        if (buttons[0][i].getText().equals(String.valueOf(currentPlayer)) &&
            buttons[1][i].getText().equals(String.valueOf(currentPlayer)) &&
            buttons[2][i].getText().equals(String.valueOf(currentPlayer))) {
            return true;
        }
    }
    if (buttons[0][0].getText().equals(String.valueOf(currentPlayer)) &&
        buttons[1][1].getText().equals(String.valueOf(currentPlayer)) &&
        buttons[2][2].getText().equals(String.valueOf(currentPlayer))) {
        return true;
    }
    if (buttons[0][2].getText().equals(String.valueOf(currentPlayer)) &&
        buttons[1][1].getText().equals(String.valueOf(currentPlayer)) &&

```

```

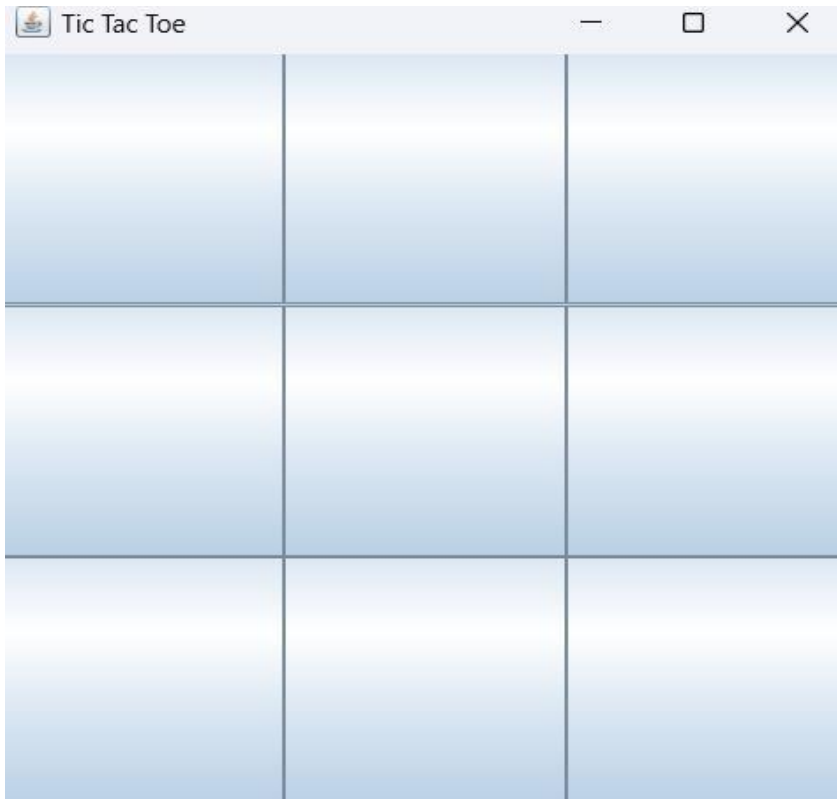
        buttons[2][0].getText().equals(String.valueOf(currentPlayer))) {
            return true;
        }
        return false;
    }

    // Check if the game is a draw
    private boolean isDraw() {
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                if (buttons[i][j].getText().equals(" ")) {
                    return false;
                }
            }
        }
        return true;
    }

    // Reset the game
    private void resetGame() {
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                buttons[i][j].setText(" ");
            }
        }
        currentPlayer = 'X';
    }
}

```

APPENDIX-B SCREEN SHOT



X		X
	O	
O	X	

