|  |  |
| --- | --- |
| Programming fundamental  Project Report | Tic Tak Toe  In C++ |

# Programming Project Report: Tic-Tac-Toe Game

## Title Page:

Project Title: Tic-Tac-Toe Game

Author: Abdul Haseeb ,Aqdas Munir & Subhan Khalid

Institution: PAF-IAST

Course/Subject: Programming Fundamentals

Date of Submission: 01-01-25

## Table of Contents

1. Abstract

2. Introduction

3. Objectives

4. Requirements

5. System Design

6. Implementation

7. Testing

8. Results and Analysis

9. Conclusion

10. References

## Abstract

This project report presents the development of a console-based Tic-Tac-Toe game using C++. The game allows two players to compete against each other, alternating turns to place their marks on a 3x3 grid. The program determines the winner or declares a draw based on the standard rules of Tic-Tac-Toe. The project demonstrates fundamental programming concepts such as arrays, loops, conditionals, and functions.

## Introduction

Tic-Tac-Toe is a classic two-player game that is simple yet engaging. The objective of the game is to align three marks (X or O) in a row, column, or diagonal on a 3x3 grid. This project implements the game using C++ programming language, focusing on providing a user-friendly console interface and adhering to the standard rules of the game.

The report outlines the process of designing and implementing the game, highlighting the key features, challenges, and outcomes.

## Objectives

1. To develop a fully functional console-based Tic-Tac-Toe game.

2. To practice and demonstrate proficiency in C++ programming.

3. To implement game logic and error handling effectively.

4. To provide a fun and interactive experience for two players.

## Requirements

Functional Requirements:

- The program should display a 3x3 grid.

- Players should take turns to input their choices.

- The program should check for a win or a draw after each turn.

- The game should terminate when a winner is determined or if the grid is full.

Non-Functional Requirements:

- The program should provide clear instructions and error messages.

- It should handle invalid inputs gracefully.

- The code should be modular and easy to maintain.

Hardware/Software Requirements:

- Operating System: Windows/Linux/MacOS

- Compiler: GCC or any C++ compiler

- Development Environment: Visual Studio Code, Dev-C++, or any preferred IDE

## System Design

Architecture:

The system follows a modular design with functions handling specific tasks such as:

1. Displaying the game board.

2. Taking player input.

3. Checking game status (win/draw).

**Flowchart:**

1. Start - Initialize the game board and variables.

2. Display Board - Show the current state of the grid.

3. Player Turn - Prompt the current player to input their choice.

4. Update Board - Validate and update the board based on the input.

5. Check Status - Determine if there is a winner or a draw.

6. Repeat - Alternate turns until the game ends.

7. End - Announce the result.

## Implementation

The game is implemented in C++ using:

- Arrays: A 2D array represents the 3x3 game board.

- Loops: Used to iterate through the grid for checking conditions.

- Functions:

- display\_board(): Displays the current state of the game.

- player\_turn(): Handles player input and updates the board.

- gameover(): Checks for a win or draw condition.

- Control Statements: if-else statements ensure correct game logic.

## Results and Analysis

The game successfully meets the project objectives. Players can interact with the game seamlessly, and all scenarios (win, draw, invalid input) are handled effectively. The modular structure of the code ensures readability and maintainability.

Challenges Faced:

- Handling invalid inputs without disrupting the flow of the game.

- Ensuring the game logic covers all possible winning conditions.

## Conclusion

The Tic-Tac-Toe game implemented in C++ demonstrates fundamental programming skills and provides an interactive experience for users. The project achieves its objectives and serves as a solid foundation for future enhancements.