# TIC TAC TOE GAME IN C

*Complex Computing Program (CCP)*

**NED UNIVERSITY OF ENGINEERING AND TECHNOLOGY**


**DEPARTMENT:** BCIT

**CLASS:** FSCS-D

**COURSE INSTRUCTOR:** Muhammad Abdullah

**GROUP MEMBERS:**

Maryum Mairaj (CT-170)

Taha Ahmed Mallick (CT-183)

Abdul Raheem Sheikh (CT-192)

# Project Proposal

---

**1. Project Title**

**Tic Tac Toe Multiplier – An Interactive Logic-Based Game in C**

---

**2. Project Description**

*Tic Tac Toe Multiplier* is an enhanced version of the classic Tic Tac Toe game, developed using the C programming language.

The game allows two players to compete on a standard 3×3 grid. The goal is to win by aligning three identical symbols, combining strategy with arithmetic reasoning.

The project focuses on implementing efficient control structures, modular programming, and algorithmic thinking — all within the constraints of console-based C programming.

---

**3. Project Methodology**

**a. Dataset**

Since this is a logic-based game rather than a data-driven application, there is no external dataset involved. However, data is internally stored in 1D array representing the game board and player moves. Dynamic variables store:

- Current Player
- Board states (X/O/Empty)
- Winning combination

---

**b. Tools and Technologies**

| Category | Tools/Technologies Used |
|---|---|
| Programming Language | C |
| IDE | Dev-C++ / Visual Studio Code |
| OS Platform | Windows (compatible with Linux terminals) |
| Data Structures | 2D Arrays, Loops, Conditional Statements |
| Libraries | `<stdio.h>`, `<stdlib.h>`, `<windows.h>` (only on windows machine) |

---

**c. Algorithm**

**Algorithm Overview:**

1. **Initialize** 3×3 board with empty cells.
2. **Repeat** until there isn't a decisive winner or a draw:
    - Display updated board.
    - Accept player input.
    - Validate move (check if cell is empty).
    - Update cell with player symbol.
    - Check win or draw conditions.
    - Switch turn to the next player.
3. **Declare winner, or draw if the match is a draw**.
4. Ask user if they wish to replay.

**Win Detection Logic:**
Checks all rows, columns, and diagonals using logical AND comparisons on the board matrix.

---

**d. Objectives**

- To develop a modular and optimized C program using functions, arrays, and loops.
- To integrate arithmetic logic within a game environment.
- To improve players' strategic and computational thinking.
- To understand algorithm design and complexity management in real-time execution.
- To apply validation and control mechanisms to handle erroneous inputs.

---

**e. Timeline**

| Phase | Duration | Activities |
|---|---|---|
| **Phase 1: Planning** | Week 1 | Define scope, design logic, and rules. |
| **Phase 2: Development** | Weeks 2–3 | Implement main game loop, board setup, validation. |
| **Phase 3: Testing & Debugging** | Week 4 | Test edge cases, fix logical and runtime errors. |
| **Phase 4: Optimization** | Week 5 | Improve efficiency, code refactoring. |
| **Phase 5: Documentation** | Week 6 | Prepare report, flowcharts, and final presentation. |

---

**f. Expected Outcomes**

- A fully functional and interactive C-based console game.
- Enhanced understanding of data structures, control flow, and logic implementation.
- Replay system.
- Algorithm optimized for efficiency and clarity.
- A documented, reproducible project ready for future extension (AI opponent, GUI, etc.).

**g. Goals**

- Achieve error-free and stable execution.
- Demonstrate programming competency in structured C.
- Encourage creativity and problem-solving through algorithmic design.
- Lay the groundwork for developing more advanced logic-based applications.

---

**4. Justification: Why It Is a Complex Computing Problem**

Although the concept appears simple, *Tic Tac Toe Multiplier* qualifies as a **complex computing problem** because it integrates multiple programming concepts simultaneously:

- **Dynamic Data Handling:** Real-time score updates demand efficient memory and control flow management.
- **Algorithmic Validation:** The win-detection and draw logic must work flawlessly for all 8 possible win combinations.
- **Error and Exception Handling:** Input validation, replay functionality, and invalid move detection ensure program robustness.
- **Optimization Constraints:** Designing for both performance and readability within the limitations of procedural programming makes it challenging.

Thus, the project not only tests basic coding skills but also challenges the student's ability to integrate **logic, mathematics, and algorithmic design** effectively.

---

**5. Industrialization / Commercial Product Potential**

While *Tic Tac Toe Multiplier* is primarily an educational project, it has **strong commercial and learning potential** when extended:

- **Gamification Platforms:** The multiplier-based logic can be adapted into mobile or web-based educational games that teach logical reasoning and arithmetic in an engaging format.
- **AI Integration:** Adding an AI opponent using algorithms like Minimax could elevate it into a full-fledged app for logical training.
- **Learning Software:** Schools or coding bootcamps could use it to demonstrate algorithm design, modular programming, and debugging techniques.
- **Commercial Product Idea:** With a graphical interface (e.g., C++/Python GUI or Unity), it could become a **competitive logic puzzle game** targeting young learners.