

C Programming Project Report



Project Title:

Tic Tac Toe Game in C

Course:

Programming in C (B. Tech cse)

Submitted By:

NAME : RIYA KEDIA

SAP ID: 590027051

BATCH: 48

Submitted To:

Mr. MOHSIN DAR

Date of Submission:

05/12/2025

1. Problem Statement

The project aims to develop a simple two-player **Tic Tac Toe** game using the C programming language. The game allows two users to play on a 3×3 grid by marking 'X' or 'O'. The program should determine whether Player 1 or Player 2 wins or if the game ends in a draw. It solves the problem of manually managing game states by automating the display, turn-taking, and win-checking processes.

2. Objective of the Project

- To implement a playable Tic Tac Toe game using C.
- To understand and apply concepts of arrays, loops, and conditional statements.
- To practice modular programming by dividing tasks into functions.
- To develop logical thinking through board evaluation and win detection.
- To create an interactive console-based application.

3. Software / Tools Used

- **Operating System:** Windows / Linux / macOS
 - **IDE/Compiler:** VS Code
-

4. Algorithm

1. Start the program.
 2. Initialize a 3×3 board with numbers 1–9.
 3. Display the current board to the players.
 4. Set Player 1 as 'X' and Player 2 as 'O'.
 5. Ask the current player to choose a position (1–9).
 6. Validate the input:
 - If the position is already taken, ask again.
 - Otherwise, place the symbol.
 7. Check for a winning condition:
 - Any row, column, or diagonal with the same symbol.
 8. If a player wins, display the winner and end the game.
 9. If all 9 moves are completed with no winner, declare a draw.
 10. End the program.
-

5. Pseudocode

Start

Initialize board with numbers 1-9

moves = 0

Repeat

 Display board

 player = (moves % 2) + 1

 mark = 'X' if player == 1 else 'O'

 Input choice from player

 Validate choice

 If invalid, repeat input

 Place mark on board

 moves = moves + 1

 If checkWin() is true

 Display board

 Print "Player wins"

 End

 If moves == 9

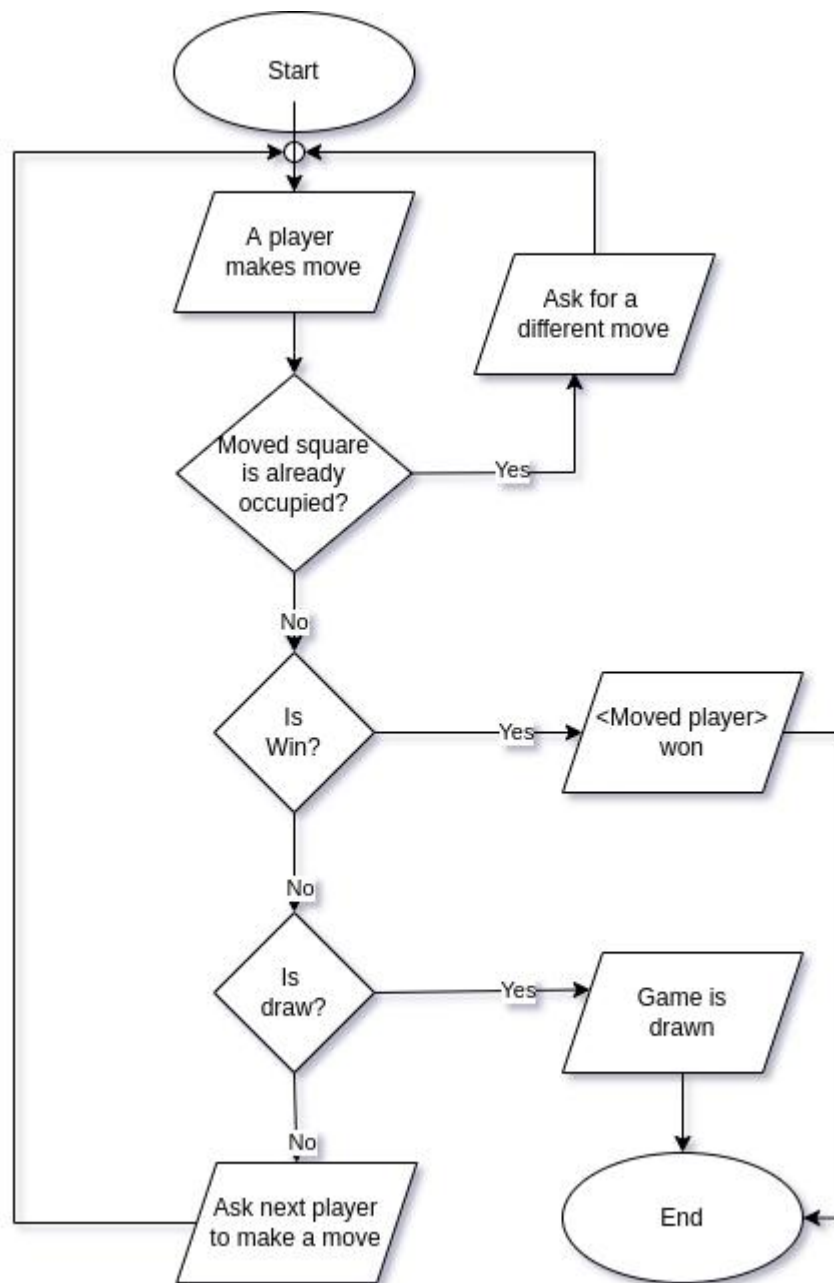
 Display board

 Print "Draw"

 End

Until false

6. FLOWCHART



7. Source Code

Below is the complete C code for the Tic Tac Toe project:

```
#include <stdio.h>
```

```
char board[3][3] = { {'1','2','3'},  
                     {'4','5','6'},  
                     {'7','8','9'} };
```

```
void displayBoard() {  
    printf("\n");  
    for(int i=0; i<3; i++) {  
        for(int j=0; j<3; j++) {
```

```

        printf(" %c ", board[i][j]);
        if(j < 2) printf("|");
    }
    if(i < 2) printf("\n---|---|---\n");
}
printf("\n");
}

```

```

int checkWin() {
    // Check rows and columns
    for(int i=0; i<3; i++) {
        if(board[i][0] == board[i][1] && board[i][1] == board[i][2])
            return 1;
        if(board[0][i] == board[1][i] && board[1][i] == board[2][i])
            return 1;
    }

    // Check diagonals
    if(board[0][0] == board[1][1] && board[1][1] == board[2][2])
        return 1;
    if(board[0][2] == board[1][1] && board[1][1] == board[2][0])
        return 1;

    return 0;
}

```

```

int main() {
    int choice, row, col, player = 1, moves = 0;
    char mark;

    while(1) {
        displayBoard();

        player = (moves % 2) + 1;
        mark = (player == 1) ? 'X' : 'O';

        printf("Player %d, enter a number: ", player);
        scanf("%d", &choice);

        row = (choice - 1) / 3;
        col = (choice - 1) % 3;

        if(choice < 1 || choice > 9 || board[row][col] == 'X' || board[row][col] == 'O') {
            printf("Invalid move! Try again.\n");
            continue;
        }

        board[row][col] = mark;
    }
}

```

```
    moves++;

    if(checkWin()) {
        displayBoard();
        printf("Player %d wins!\n", player);
        break;
    }

    if(moves == 9) {
        displayBoard();
        printf("It's a draw!\n");
        break;
    }
}

return 0;
}
```

8. Conclusion

The Tic Tac Toe project successfully demonstrates the use of C programming concepts such as loops, functions, conditionals, and arrays. The game is fully interactive and handles player turns, move validation, and win/draw detection effectively. The project helped in understanding modular program design and logical problem-solving.

9. Future Enhancements

- Add a computer vs player (AI mode) using the Minimax algorithm.
- Create a graphical version using graphics libraries.
- Add a menu system and a scoreboard.
- Implement sound effects and animations.
- Add input sanitization and replay options.