Tic-Tac-Toe Game: A Console-based version using C Programming

Project Update of Group 2 by:

S M Uday Haider
North South University
Student ID: 2512062642
Department of Electrical & Computer Engineering

Kanij Fatema Lubna
North South University
Student ID: 2512430642
Department of Electrical & Computer Engineering

Asifur Rahman Apon
North South University
Student ID: 2512131642
Department of Electrical & Computer Engineering

Safin Ahmed
North South University
Student ID: 2511967642
Department of Electrical & Computer Engineering

Abstract

The goal of this project is to build a console based Tic- Tac- Toe game in C Program where two players can compete against each other. The key features of this project are: interchanging the moves, displaying the outcomes and declaring the result. The target is to create a user-friendly program. It is an ideal project to understand C programming in more diverse applications. This report outlines the efforts made, problems faced, and improvements expected in the future.

1. Introduction

Tic- Tac- Toe is a widely known game that was made around 1300 B.C. in ancient Egypt. It is played between two players in a 3x3 grid using the 'X' and 'O' symbol. There can be three cases of conclusion: win, loss and draw. Different fundamentals of C programming are used for making the game logic and establishing the conclusions. They are: arrays, loops, functions and conditionals. On the console, the players will be asked to input a number from 1 to 9 one after another. Each number is assigned for each block of the game and as an output, 'X' will be shown in the first player's preferred block and 'O' will be shown in the second player's preferred block.

2. Implementation Progress

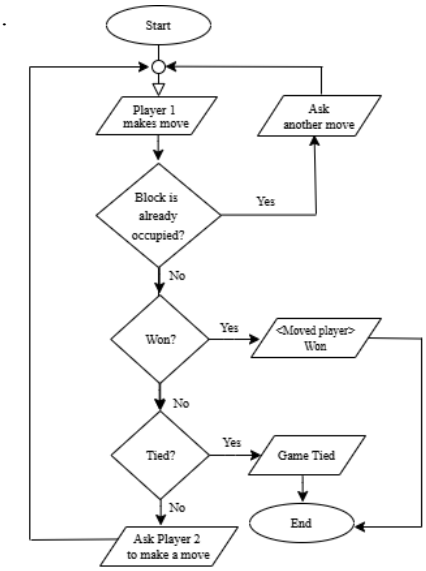
The following aspects are planned to executed:

- *Interchanging the moves:* The game board is represented by a 3x3 grid which is initialized with numbers (1-9). The board updates automatically after each valid move.
- *Displaying the outcome:* Players are asked to input their moves and the program will check if the position is already occupied or not.
- *Declaring the result:* The result will be declared as per the horizontal, vertical and diagonal blocks. The first player who will mark their signs accordingly wins the game.

3. Requirements

- *C Compiler:* GCC, Turbo C, Dev-C++.
- *Operating System:* Windows/Linux/MacOS.
- *IDE*: Code::Blocks, Visual Studio Code etc.
- A basic CPU with a small amount of RAM.

4. Flowchart



5. Code Structure

Up to this point, we have completed the following portion of the code:

```c
#include <stdio.h>

#define SIZE 3

char board        ;

void initializeBoard() {
    int num = 1;
    for (int i = 0; i < SIZE; i++) {
        for (int j = 0; j < SIZE; j++) {
            board    = num + '0';
            num++;
        }
    }
}

void displayBoard() {
    putchar('\n');
    for (int i = 0; i < SIZE; i++) {
        for (int j = 0; j < SIZE; j++) {
            putchar(' ');
            putchar();
            putchar(' ');
            if (j < SIZE - 1) putchar('|');
        }
        putchar('\n');
        if (i < SIZE - 1) printf("---|---|---\n");
    }
    putchar('\n');
}

int checkWin() {
    for (int i = 0; i < SIZE; i++) {
        if (  ) return 1;
        if (  ) return 1;
    }
    if (   ) 1;
    if (   ) return 1;
    return 0;
}

int isDraw() {
    for (int i = 0; i < SIZE; i++) {
        for (int j = 0; j < SIZE; j++) {
            if (    ) return 0;
        }
    }
    return 1;
}

void playGame() {
    int player = 1, choice, row, col;
    char mark;

    initializeBoard();

    while (1) {
        displayBoard();
        mark = (player == 1) ? 'X' : 'O';
        printf("Player %d, enter your move (1-9): ", player);
        scanf("%d", &choice);

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

        if () {
            printf("Invalid move, try again.\n");
            continue;
        }

        board = mark;

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

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

        player = (player == 1) ? 2 : 1;
    }
}

int main() {
    printf("Welcome to Tic Tac Toe!\n");
    playGame();
    return 0;
}
```

6. Issues and Solutions

Issues: This project validates inputs to allow only numbers 1 to 9 and prevent infinite loops. Turns switch only after a valid move, ensuring players don't lose turns due to invalid inputs. For a smooth game, input errors have to be handled accurately. The game should only terminate when a player wins or all blocks are filled.

Solutions: This program evaluates the input if it is a number between 1-9 and asks the players again if it is not valid. Turns will switch only after a valid move. Therefore, opting an occupied block or inputting a wrong input won't change turns. Error management should prevent infinite loops, allowing the game to run smoothly and terminate properly when someone wins or cells are filled.

7. Methodology

- *Development Approach:* The game is designed with a step-by-step approach, starting with board representation, where the playing grid is set up and displayed. The game flow ensures smooth turn-based interaction between two players. Proper input handling is essential to validate moves and prevent errors. Finally, conclusion checking helps determine the winner or if the game results in a draw.
- *Future Enhancement:* There are several ways to improve the game. Implementing AI would allow single-player mode against a computer opponent. Adding a graphical interface would make the game more user-friendly. Another potential upgrade is enabling network-based multiplayer, allowing players to compete online.

8. Conclusion

The C-based console Tic-Tac-Toe game allows two players to play interactively. It is fun, engaging, and competitive. The game sets up the board, manages inputs, and checks for win conditions. It prevents invalid moves and malpractices.This project covers loops, conditionals, arrays, and user input in C programming.It emphasizes error handling and user experience.

9. References

1. C: The Complete Reference - H. Schildt.
2. Problem Solving and Program Design in C - J. Hanly and E. Koffman.
3. C: How to Program - Deitel & Deitel.