**TIC TAC TOE**

**CONSOLE APPLICATION**

**SUBMITTED BY:**

MANALO, AIRON G.

VALDEZ, JOHN RAE A.

BS IN COMPUTER SCIENCE 1 - B

**NOVEMBER 23, 2023**

Contents

[**PROJECT OVERVIEW** 2](#_Toc151976387)

[**PROGRESS SUMMARY** 2](#_Toc151976388)

[**TECHNICAL DETAILS:** 3](#_Toc151976389)

[**CODE SNIPPET:** 3](#_Toc151976390)

[**CONSOLE SCREENSHOTS:** 10](#_Toc151976391)

[**FUNDAMENTALS OF PROGRAMMING APPLIED:** 13](#_Toc151976392)

# **PROJECT OVERVIEW**

**Project Name**: TIC TAC TOE

**Date Started**: NOVEMBER 23, 2023

**Date Last Updated**: NOVEMBER 27, 2023

**Project Team Leader**: VALDEZ, JOHN RAE A.

**Project Team Members**: MANALO, AIRON G.

# **PROGRESS SUMMARY**

**Completed Tasks**:

**Task 1:** Create a data structure (often a 3x3 grid or array) to represent the game board on which players will place their movements.

**Task 2:** Create a function that displays the current state of the board, including the X and O positions.

**Task 3:** Create a method for players to enter their moves. Accepting input as coordinates or numerical input corresponding to grid positions could be involved.

**Task 4:** Check that the moves entered by players are correct. Handle invalid moves by asking the player to enter a valid move again.

**Task 5:** After each valid move, implement logic to switch between players (X and O).

**Task 6:** To determine if a player has won the game, create a mechanism that checks for winning conditions after each move. This entails looking for matching symbols in rows, columns, and diagonals.

**Task 7:** If all spaces on the board are occupied without a winner, implement a condition to check for a draw.

**Task 8:** Create a loop that keeps the game going until a player wins or the game ends in a tie.

**Task 9:** Allow players to restart the game after it has finished.

**Task 10:** Handle unexpected inputs or errors with grace, directing players to appropriate actions.

**Challenges Faced:**

**Challenge 1:** Console applications rely primarily on textual input, which frequently necessitates robust validation. It can be difficult to handle many forms of input errors and ensure that the program responds appropriately to incorrect inputs.

**Challenge 2**: It can be difficult to ensure that the program operates consistently across different terminal contexts and operating systems. Terminal capabilities and behaviors may influence how the program runs.

**Lessons Learned:**

- Do not rush to type the codes and make it a practice to carefully examine the encoded codes to reduce errors.

**Future Plans:**

**Next Milestone:** N/A

**Enhancements:** N/A

**Bug Fixes:** N/A

# **TECHNICAL DETAILS:**

**Technologies Used:**

Programming Language: **C#**

Framework: **N/A**

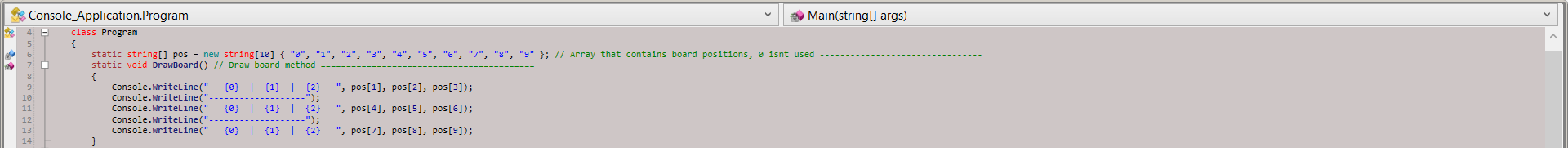
External Libraries: **N/A**

# **CODE SNIPPET:**

class Program  
    {  
        static string[] pos = **new** string[10] { "0", "1", "2", "3", "4", "5", "6", "7", "8", "9" }; // Array that contains board positions, 0 isnt used --------------------------------  
        static void **DrawBoard**() // Draw board method ==========================================  
        {  
            Console.**WriteLine**("   {0}  |  {1}  |  {2}   ", pos[1], pos[2], pos[3]);  
            Console.**WriteLine**("-------------------");  
            Console.**WriteLine**("   {0}  |  {1}  |  {2}   ", pos[4], pos[5], pos[6]);  
            Console.**WriteLine**("-------------------");  
            Console.**WriteLine**("   {0}  |  {1}  |  {2}   ", pos[7], pos[8], pos[9]);  
        }  
        static void **Main**(string[] args) // Main ==============================================  
        {  
            string player1 = "", player2 = "";  
            **int** choice = 0, turn = 1, score1 = 0, score2 = 0;  
            **bool** winFlag = **false**, playing = **true**, correctInput = **false**;  
            Console.**WriteLine**("Welcome to Tic Tac Toe! This is a two-player game where each player will take turns marking the spaces in a 3x3 grid.");  
            Console.**WriteLine**("Player 1 is 'O' and Player 2 is 'X'. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row is the winner.");  
            Console.**WriteLine**("The game board looks like this:");          
            Console.**WriteLine**("   1  |  2  |  3   ");  
            Console.**WriteLine**("-------------------");  
            Console.**WriteLine**("   4  |  5  |  6   ");  
            Console.**WriteLine**("-------------------");  
            Console.**WriteLine**("   7  |  8  |  9   ");  
            Console.**WriteLine**("Each number represents a position on the board where you can place your mark. For example, if you want to place your mark in the top left corner, you would enter '1'.");  
            Console.**WriteLine**("If a position is already taken, you'll be asked to choose another one. If all positions are filled and no player has won, the game is a draw.");  
            Console.**WriteLine**("You can pause the game at any time by entering 'P', and resume by pressing 'R'.");  
            Console.**WriteLine**("Let's get started!");  
            Console.**WriteLine**("What is the name of player 1?");  
            player1 = Console.**ReadLine**();  
            Console.**WriteLine**("Very good. What is the name of player 2?");  
            player2 = Console.**ReadLine**();  
            Console.**WriteLine**("Okay good. {0} is O and {1} is X.", player1, player2);  
            Console.**WriteLine**("{0} goes first.", player1);  
            Console.**WriteLine**("Press any key to start!");  
            Console.**ReadLine**();  
            Console.**Clear**();  
            **while** (playing == **true**)  
            {  
                **while** (winFlag == **false**) // Game loop ------------------------------------------------------  
                {  
                    **DrawBoard**();  
                    Console.**WriteLine**("");  
                    Console.**WriteLine**("Score: {0} - {1}     {2} - {3}", player1, score1, player2, score2);  
                    **if** (turn == 1)  
                    {  
                        Console.**WriteLine**("{0}'s (O) turn", player1);  
                    }  
                    **if** (turn == 2)  
                    {  
                        Console.**WriteLine**("{0}'s (X) turn", player2);  
                    }  
                    **while** (correctInput == **false**)  
                    {  
                         Console.**WriteLine**("Which position would you like to take? Press P to pause the game.");  
                        string input = Console.**ReadLine**();  
                        **if** (input.**ToUpper**() == "P")  
                        {  
                            Console.**WriteLine**("Game paused. Press R to resume the game.");  
                            **while** (Console.**ReadKey**().Key != ConsoleKey.R)  
                            {  
                                // Wait for the user to press R to resume the game  
                            }  
                            Console.**Clear**();  
                            **DrawBoard**(); // Redraw the board after resuming the game  
                            Console.**WriteLine**("");  
                            Console.**WriteLine**("Score: {0} - {1}     {2} - {3}", player1, score1, player2, score2); // Redraw the score after resuming the game  
                        }  
                        **else**  
                        {  
                            choice = **int**.**Parse**(input);  
                            **if** (choice > 0 && choice < 10)  
                            {  
                                correctInput = **true**;  
                            }  
                            **else**  
                            {  
                                continue;  
                            }  
                        }  
                    }  
                    correctInput = **false**; // Reset -------  
                    **if** (turn == 1)  
                    {  
                        **if** (pos[choice] == "X") // Checks to see if spot is taken already --------------------  
                        {  
                            Console.**WriteLine**("You can't steal positions asshole! ");  
                            Console.**Write**("Try again.");  
                            Console.**ReadLine**();  
                            Console.**Clear**();  
                            continue;  
                        }  
                        **else**  
                        {  
                            pos[choice] = "O";  
                        }  
                    }  
                    **if** (turn == 2)  
                    {  
                        **if** (pos[choice] == "O") // Checks to see if spot is taken already -------------------  
                        {  
                            Console.**WriteLine**("You can't steal positions asshole! ");  
                            Console.**Write**("Try again.");  
                            Console.**ReadLine**();  
                            Console.**Clear**();  
                            continue;  
                        }  
                        **else**  
                        {  
                            pos[choice] = "X";  
                        }  
                    }  
                    winFlag = **CheckWin**();  
                    **if** (winFlag == **false**)  
                    {  
                        winFlag = **CheckWin**();  
                        **if** (winFlag == **false**)  
                        {  
                            **if** (turn == 1)  
                            {  
                                turn = 2;  
                            }  
                            **else** **if** (turn == 2)  
                            {  
                                turn = 1;  
                            }  
                            Console.**Clear**();  
                        }  
                    }  
                }  
                Console.**Clear**();  
                **DrawBoard**();  
                **for** (**int** i = 1; i < 10; i++) // Resets board ------------------------  
                {  
                    pos[i] = i.**ToString**();  
                }  
                **if** (winFlag == **false**) // No one won ---------------------------  
                {  
                    Console.**WriteLine**("It's a draw!");  
                    Console.**WriteLine**("Score: {0} - {1}     {2} - {3}", player1, score1, player2, score2);  
                    Console.**WriteLine**("");  
                    Console.**WriteLine**("What would you like to do now?");  
                    Console.**WriteLine**("1. Play again");  
                    Console.**WriteLine**("2. Leave");  
                    Console.**WriteLine**("");  
                    **while** (correctInput == **false**)  
                    {  
                        Console.**WriteLine**("Enter your option: ");  
                        choice = **int**.**Parse**(Console.**ReadLine**());  
                        **if** (choice > 0 && choice < 3)  
                        {  
                            correctInput = **true**;  
                        }  
                    }  
                    correctInput = **false**; // Reset -------------  
                    **switch** (choice)  
                    {  
                        **case** 1:  
                            break;  
                        **case** 2:  
                            Console.**Clear**();  
                            Console.**WriteLine**("Thanks for playing!");  
                            Console.**ReadLine**();  
                            playing = **false**;  
                            break;  
                    }  
                }  
                **if** (winFlag == **true**) // Someone won -----------------------------  
                {  
                    **if** (turn == 1)  
                    {  
                        score1++;  
                        Console.**WriteLine**("{0} wins!", player1);  
                        Console.**WriteLine**("What would you like to do now?");  
                        Console.**WriteLine**("1. Play again");  
                        Console.**WriteLine**("2. Leave");  
                        **while** (correctInput == **false**)  
                        {  
                            Console.**WriteLine**("Enter your option: ");  
                            choice = **int**.**Parse**(Console.**ReadLine**());  
                            **if** (choice > 0 && choice < 3)  
                            {  
                                correctInput = **true**;  
                            }  
                        }  
                        correctInput = **false**; // Reset --------------  
                        **switch** (choice)  
                        {  
                            **case** 1:  
                                Console.**Clear**();  
                                winFlag = **false**;  
                                break;  
                            **case** 2:  
                                Console.**Clear**();  
                                Console.**WriteLine**("Thanks for playing!");  
                                Console.**ReadLine**();  
                                playing = **false**;  
                                break;  
                        }  
                    }  
                    **if** (turn == 2)  
                    {  
                        score2++;  
                        Console.**WriteLine**("{0} wins!", player2);  
                        Console.**WriteLine**("What would you like to do now?");  
                        Console.**WriteLine**("1. Play again");  
                        Console.**WriteLine**("2. Leave");  
                        **while** (correctInput == **false**)  
                        {  
                            Console.**WriteLine**("Enter your option: ");  
                            choice = **int**.**Parse**(Console.**ReadLine**());  
                            **if** (choice > 0 && choice < 3)  
                            {  
                                correctInput = **true**;  
                            }  
                        }  
                        correctInput = **false**; // Reset -----------------  
                        **switch** (choice)  
                        {  
                            **case** 1:  
                                Console.**Clear**();  
                                winFlag = **false**;  
                                break;  
                            **case** 2:  
                                Console.**Clear**();  
                                Console.**WriteLine**("Thanks for playing!");  
                                Console.**ReadLine**();  
                                playing = **false**;  
                                break;  
                        }  
                    }  
                }  
            }  
        }  
          
        static void **IsBoardFull**()  
        {  
            **throw** **new** **NotImplementedException**();  
        }  
        static **bool** **CheckWin**() // Win checker method ================================================  
        {  
            **if** (pos[1] == "O" && pos[2] == "O" && pos[3] == "O") // Horizontal ----------------------------------------  
            {  
                return **true**;  
            }  
            **else** **if** (pos[4] == "O" && pos[5] == "O" && pos[6] == "O")  
            {  
                return **true**;  
            }  
            **else** **if** (pos[7] == "O" && pos[8] == "O" && pos[9] == "O")  
            {  
                return **true**;  
            }  
            **else** **if** (pos[1] == "O" && pos[5] == "O" && pos[9] == "O") // Diagonal -----------------------------------------  
            {  
                return **true**;  
            }  
            **else** **if** (pos[7] == "O" && pos[5] == "O" && pos[3] == "O")  
            {  
                return **true**;  
            }  
            **else** **if** (pos[1] == "O" && pos[4] == "O" && pos[7] == "O")// Coloumns ------------------------------------------  
            {  
                return **true**;  
            }  
            **else** **if** (pos[2] == "O" && pos[5] == "O" && pos[8] == "O")  
            {  
                return **true**;  
            }  
            **else** **if** (pos[3] == "O" && pos[6] == "O" && pos[9] == "O")  
            {  
                return **true**;  
            }  
            **if** (pos[1] == "X" && pos[2] == "X" && pos[3] == "X") // Horizontal ----------------------------------------  
            {  
                return **true**;  
            }  
            **else** **if** (pos[4] == "X" && pos[5] == "X" && pos[6] == "X")  
            {  
                return **true**;  
            }  
            **else** **if** (pos[7] == "X" && pos[8] == "X" && pos[9] == "X")  
            {  
                return **true**;  
            }  
            **else** **if** (pos[1] == "X" && pos[5] == "X" && pos[9] == "X") // Diagonal -----------------------------------------  
            {  
                return **true**;  
            }  
            **else** **if** (pos[7] == "X" && pos[5] == "X" && pos[3] == "X")  
            {  
                return **true**;  
            }  
            **else** **if** (pos[1] == "X" && pos[4] == "X" && pos[7] == "X") // Coloumns ------------------------------------------  
            {  
                return **true**;  
            }  
            **else** **if** (pos[2] == "X" && pos[5] == "X" && pos[8] == "X")  
            {  
                return **true**;  
            }  
            **else** **if** (pos[3] == "X" && pos[6] == "X" && pos[9] == "X")  
            {  
                return **true**;  
            }  
            **else**  
            {  
                **for** (**int** i = 1; i < 10; i++)  
                {  
                    **if** (pos[i] != "O" && pos[i] != "X")  
                    {  
                        return **false**;  
                    }  
                }  
                Console.**WriteLine**("It's a draw! Press any key to continue");  
                **for** (**int** i = 1; i < 10; i++)  
                {  
                    pos[i] = i.**ToString**();  
                }  
                Console.**ReadKey**();  
                return **false**;  
            }  
        }  
    }

# **CONSOLE SCREENSHOTS:**

**Screenshot 1:** Method responsible for displaying the Tic Tac Toe board with positions marked by “*X*”and “*O*”.



**Screenshot 2:** This portion initializes the program. It declares variables for player’s names, scores, and game flags. It then prompts players to input their names and displays the initial instructions.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Screenshot 3**:

The main loop manages the gameplay flow. It repeatedly executes turn until a win or a draw occurs.

* The outer ‘**while**’ loop controls the overall game and continues if the ‘**playing**’ flag is ‘**true**’.
* The inner ‘**while**’ loop handles a single game session and continues until a win is detected or the games ends in a draw.
* Within the game loop, the code displays the board, prompts players for their moves, validate input, marks the board, check for the win condition, and switches players turn.

The game continues until a player wins or there a draw. Afterward, it prompts players to decide whether they want to play again or leave the game.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Screenshot 4**: Method to check if there’s a winner based on the current state of the board. It checks for all possible win conditions: horizontal, vertical, and diagonal lines for ‘X’ and ‘O’.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

# **FUNDAMENTALS OF PROGRAMMING APPLIED:**

**1.1** Variables**:**

**Player Names**

* ‘player1’ and ‘player2’: Both are strings (‘string’), representing the names of the players.

**Game State and Control Variables**

* ‘choice’: An integer (‘int’) used to store the player's chosen position on the board.
* ‘turn’: An integer (‘int’) indicating the current player's turn (1 for Player 1 and 2 for Player 2).
* ‘score1’ and score2: Integers (‘int’) keeping track of each player's score.
* ‘winFlag’: A boolean (‘bool’) used to determine if there's a winner or not.
* ‘playing’: A boolean (‘bool’) controlling the game's continuation.

**Game Board Representation**

* ‘pos’: An array of strings (‘string[]’) used to represent the positions on the game board. Elements are strings representing the board positions.

**1.2** Data Types**:**

* ‘string’: Used for player names, board position representations, and messages.
* ‘int’: Used for choices, turns, and scores.
* ‘bool’: Used to control game state.

**2**. Control Structures (if statements, loops):

**While Loop**: Used for the game loop and input validation.

* ‘while (playing == true)’ controls the overall game flow.
* ‘while (winFlag == false)’ manages the game until a win or draw occurs.
* ‘while (correctInput == false)’ ensures correct input for position selection and menu options.

**If Statements**: Control flow based on conditions.

* ‘if (turn == 1)’ and ‘if (turn == 2)’ determines whose turn it is.
* ‘if (pos[choice] == "X")’ and ‘if (pos[choice] == "O")’ checks whether a position is already taken.

**3**. Functions/Methods:

* **‘DrawBoard()’**: This method is responsible for drawing the Tic Tac Toe board on the console.
* **‘Main()’**: The main method that handles the game flow, including player inputs, game loop, checking for wins, updating scores, and determining the game continuation.
* **‘CheckWin()’**: This method checks for a win condition by examining the contents of the pos array, which represents the state of the Tic Tac Toe board.

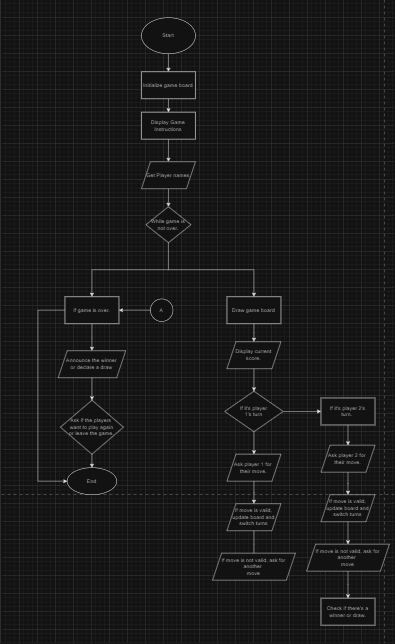
**4**. Arrays or Collections:

* **string[] pos = new string[10] { "0", "1", "2","3","4","5","6","7","8","9" };** This array pos stores the board positions. It represents the Tic Tac Toe game board with each element representing a position. The positions are initialized with strings representing their indices.

**Conclusion:**

**It was a journey of logic and strategy to create this Tic Tac Toe console app. As the game comes to a close, I hope this experience sparked your competitive spirit and challenged your tactical thinking. Remember that whether you win or lose, the thrill is in mastering this classic game. Allow this to be just the beginning of your coding and gaming adventures. Continue exploring, playing, and coding! Cheers to limitless games and limitless possibilities!**

Flowchart and Pseudocode:



Pseudocode:

1. Start
2. Initialize game board
3. Display game instructions
4. Get player names
5. While game is not over.
6. Draw game board:
7. Display current score
8. If it's player 1's turn:
9. Ask player 1 for their move
10. If move is valid, update board and switch turns
11. If move is not valid, ask for another move
12. If it's player 2's turn:
13. Ask player 2 for their move
14. If move is valid, update board and switch turns
15. If move is not valid, ask for another move
16. Check if there's a winner or a draw
17. End