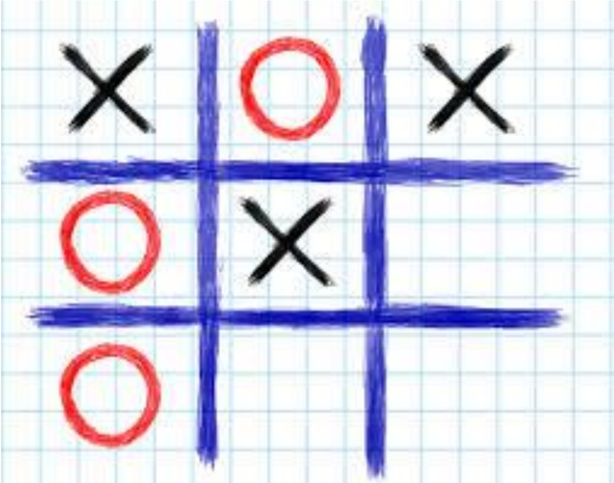
C++ Project on
Tic Tac Toe Game



NAME OF CODER: SYED ABDULLAH ASHRAF

Tic-Tac-Toe in C++

Tic-tac-toe is a straightforward two-player game that, if both players play their best, will always end in a tie. The game is also known as Xs and Os or zeros and crosses.

A computer or other device can be used to play the game of tic tac toe, which is typically played by drawing on paper. This timeless game provides the foundation for other ones, like Connect 4.

History of Tic-Tac-Toe

Around the first century B.C., a primitive version of the game was played in the Roman Empire. Three pebbles at a time is what the name "terni lapilli" denotes. Roman ruins have been discovered to be covered with chalk grid patterns from the game. Ruins in ancient Egypt have also yielded evidence of the game.

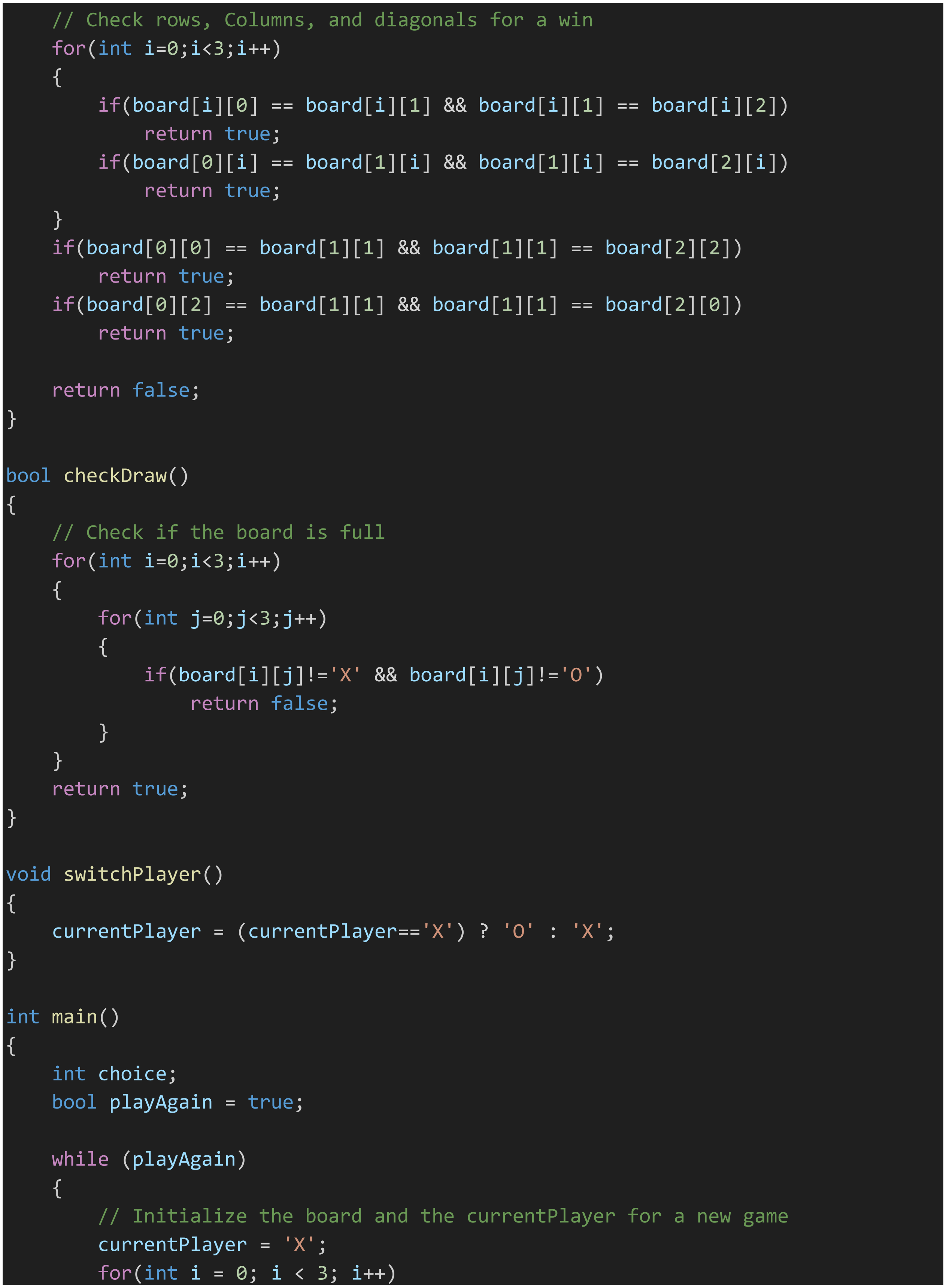
The game's British moniker, "noughts and crosses," saw its first print appearance in 1864. The term "tick-tack-toe" first appeared in literature in 1884, although it referred to a children's game played on a slate.

Rules of the Game

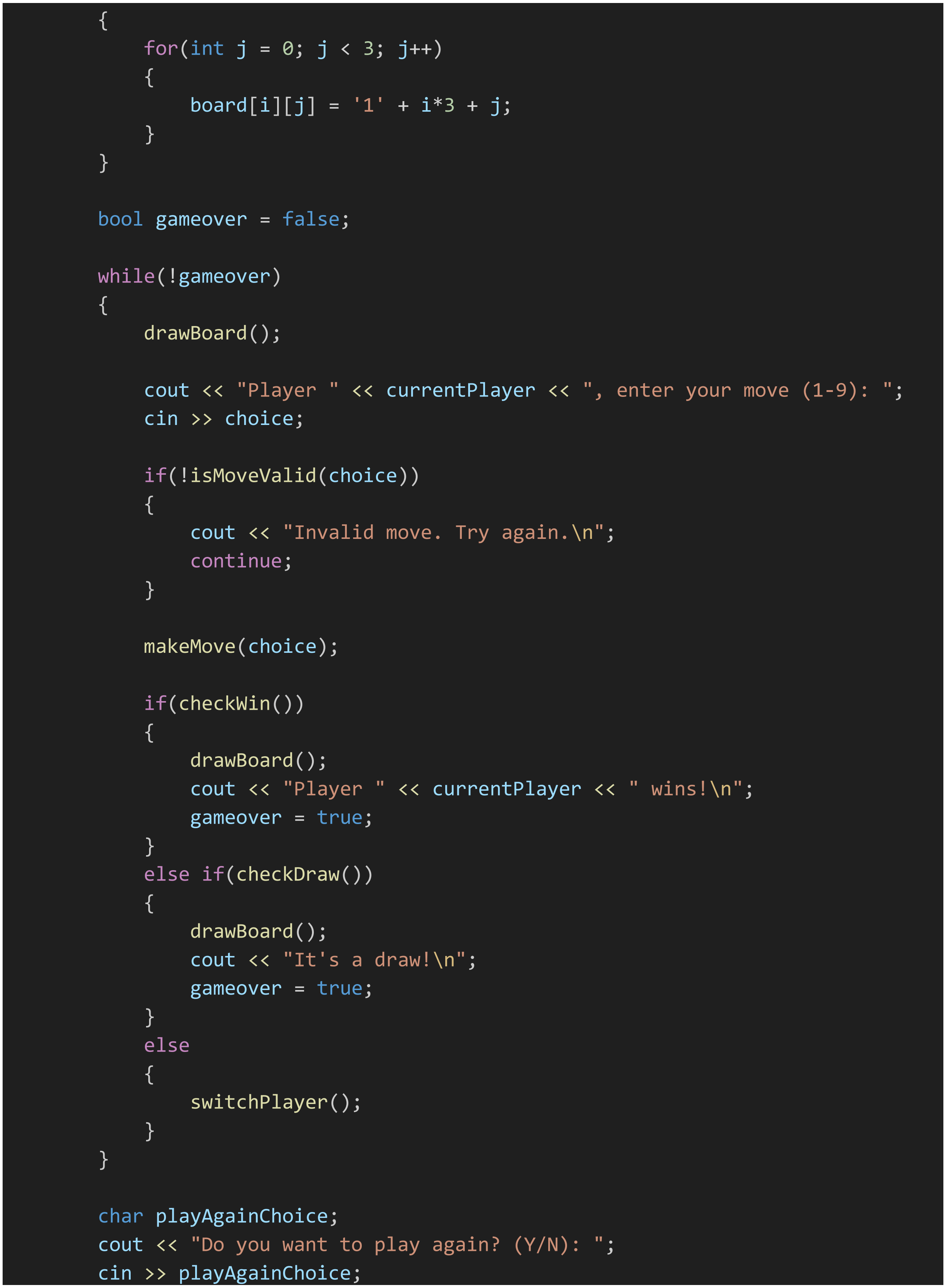
- The game must be played by two players (in this program between 2 HUMANS).
- Both players mark their cells with the letters "O" and "X".
- The game ends when one of the players fills an entire row, column or diagonal with either the character ('O' or 'X') of that player.
- If no one wins, the match is considered a draw.

Code:

```
#include <iostream>
using namespace std;

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

void drawBoard()
{
    cout<< "Tic Tac Toe\n";
    for(int i=0;i<3;i++)
    {
        for(int j=0;j<3;j++)
        {
            cout<<board[i][j]<< " ";
        }
        cout<<endl;
    }
}

bool isMoveValid(int choice)
{
    if(choice<1||choice>9)
        return false;

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

    if(board[row][col]=='X' || board[row][col]=='O')
        return false;

    return true;
}

void makeMove(int choice)
{
    int row = (choice-1)/3;
    int col = (choice-1)%3;
    board[row][col] = currentPlayer;
}

bool checkWin()
{
```



```
    // Check rows, Columns, and diagonals for a win
    for(int i=0;i<3;i++)
    {
        if(board[i][0] == board[i][1] && board[i][1] == board[i][2])
            return true;
        if(board[0][i] == board[1][i] && board[1][i] == board[2][i])
            return true;
    }
    if(board[0][0] == board[1][1] && board[1][1] == board[2][2])
        return true;
    if(board[0][2] == board[1][1] && board[1][1] == board[2][0])
        return true;

    return false;
}

bool checkDraw()
{
    // Check if the board is full
    for(int i=0;i<3;i++)
    {
        for(int j=0;j<3;j++)
        {
            if(board[i][j]!='X' && board[i][j]!='O')
                return false;
        }
    }
    return true;
}

void switchPlayer()
{
    currentPlayer = (currentPlayer=='X') ? 'O' : 'X';
}

int main()
{
    int choice;
    bool playAgain = true;

    while (playAgain)
    {
        // Initialize the board and the currentPlayer for a new game
        currentPlayer = 'X';
        for(int i = 0; i < 3; i++)
```



```
    {
        for(int j = 0; j < 3; j++)
        {
            board[i][j] = '1' + i*3 + j;
        }
    }

    bool gameover = false;

    while(!gameover)
    {
        drawBoard();

        cout << "Player " << currentPlayer << ", enter your move (1-9): ";
        cin >> choice;

        if(!isMoveValid(choice))
        {
            cout << "Invalid move. Try again.\n";
            continue;
        }

        makeMove(choice);

        if(checkWin())
        {
            drawBoard();
            cout << "Player " << currentPlayer << " wins!\n";
            gameover = true;
        }
        else if(checkDraw())
        {
            drawBoard();
            cout << "It's a draw!\n";
            gameover = true;
        }
        else
        {
            switchPlayer();
        }
    }

    char playAgainChoice;
    cout << "Do you want to play again? (Y/N): ";
    cin >> playAgainChoice;
```

```
        playAgain = (playAgainChoice == 'Y' || playAgainChoice == 'y');
    }

    cout << "Thanks for playing Tic Tac Toe!\n";
    return 0;
}
```

This C++ code represents a simple text-based implementation of the classic game "Tic Tac Toe" for two players. The game is played on a 3x3 grid, and the players take turns to place their marks ('X' and 'O') on the board until one of them wins or the game ends in a draw. The code includes functions for drawing the board, checking the validity of moves, making moves, checking for a win, checking for a draw, and switching players.

Let's break down the code step by step:

1. Global Variables:

- `board[3][3]`: A 2D array representing the game board.
- `currentPlayer`: A variable to keep track of the current player (initially set to 'X').

2. `drawBoard()` Function:

- This function is responsible for displaying the current state of the game board on the console.

3. `isMoveValid(int choice)` Function:

- This function takes an integer `choice` as input, representing the cell number (1 to 9) where the current player wants to place their mark.

- It checks whether the chosen cell is a valid move or not based on the following conditions:

  - The `choice` is within the valid range (1 to 9).

  - The cell corresponding to the `choice` is not already occupied by 'X' or 'O'.

- If the move is valid, the function returns `true`, otherwise `false`.

4. `makeMove(int choice)` Function:

- This function takes the `choice` of the current player and updates the game board with the player's mark ('X' or 'O') in the corresponding cell.

5. `checkWin()` Function:

- This function checks if any player has won the game.

- It examines the rows, columns, and diagonals to find three consecutive cells with the same mark ('X' or 'O').

- If it finds a winning pattern, it returns `true`; otherwise, it returns `false`.

6. `checkDraw()` Function:

- This function checks if the game has ended in a draw.

- It verifies whether all cells on the board are filled with 'X' or 'O'.

- If all cells are occupied, it returns `true`, indicating a draw; otherwise, it returns `false`.

7. `switchPlayer()` Function:

- This function is responsible for switching the current player from 'X' to 'O' and vice versa.

8. `main()` Function:

- The `main()` function is the entry point of the program.

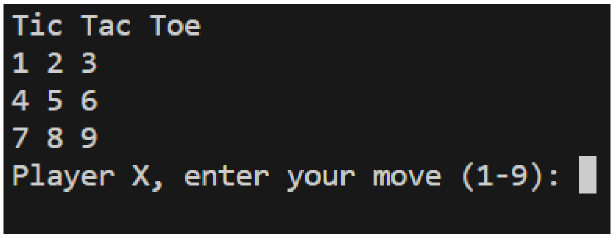
- It contains the main game loop where the players take turns making moves until there is a winner or the game ends in a draw.

- It displays the board, prompts the current player for their move, and handles the game's progression accordingly.

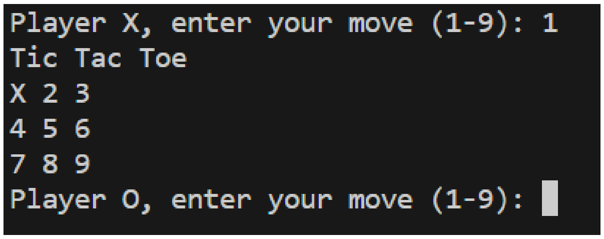
- After a game ends, it asks the players if they want to play again. If they choose to continue, a new game starts; otherwise, the program terminates.

Overall, this code creates a simple, console-based Tic Tac Toe game that allows players to play multiple rounds and keeps track of the game's outcome. It demonstrates the use of functions, loops, conditional statements, and 2D arrays in C++.

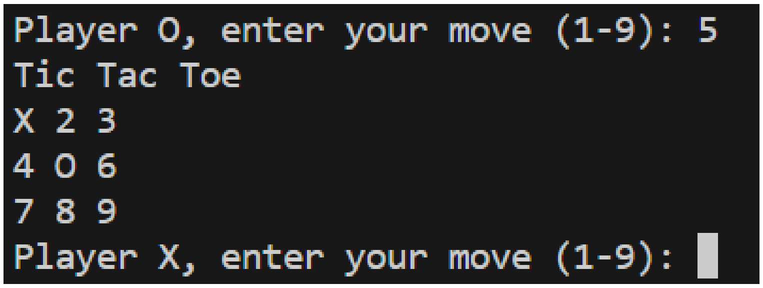
Output:



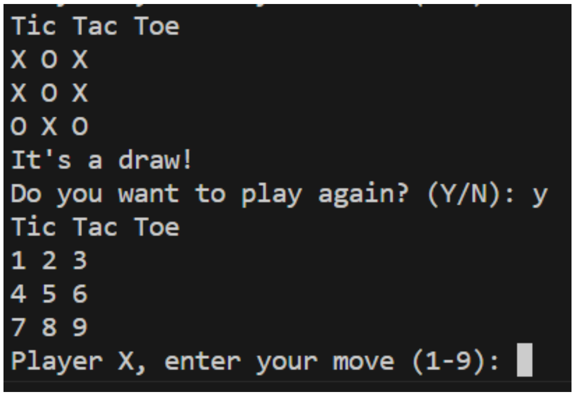
```
Tic Tac Toe
1 2 3
4 5 6
7 8 9
Player X, enter your move (1-9):
```



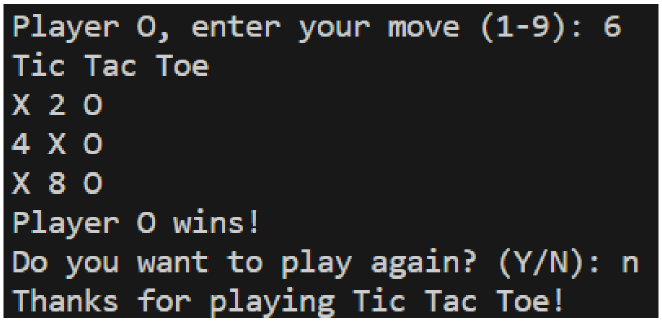
```
Player X, enter your move (1-9): 1
Tic Tac Toe
X 2 3
4 5 6
7 8 9
Player O, enter your move (1-9):
```



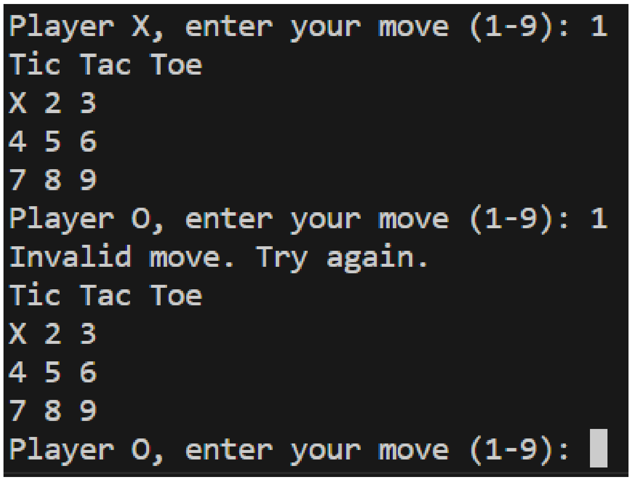
```
Player O, enter your move (1-9): 5
Tic Tac Toe
X 2 3
4 O 6
7 8 9
Player X, enter your move (1-9):
```



```
Tic Tac Toe
X O X
X O X
O X O
It's a draw!
Do you want to play again? (Y/N): y
Tic Tac Toe
1 2 3
4 5 6
7 8 9
Player X, enter your move (1-9):
```



```
Player O, enter your move (1-9): 6
Tic Tac Toe
X 2 O
4 X O
X 8 O
Player O wins!
Do you want to play again? (Y/N): n
Thanks for playing Tic Tac Toe!
```



```
Player X, enter your move (1-9): 1
Tic Tac Toe
X 2 3
4 5 6
7 8 9
Player O, enter your move (1-9): 1
Invalid move. Try again.
Tic Tac Toe
X 2 3
4 5 6
7 8 9
Player O, enter your move (1-9):
```