E:\tic tac toe.cpp

```cpp
#include <iostream>
#include <vector>

using namespace std;

// Function prototypes
void printBoard(const vector<vector<char>>& board);
bool checkWin(const vector<vector<char>>& board, char player);
bool checkDraw(const vector<vector<char>>& board);
bool isValidMove(int row, int col, const vector<vector<char>>& board);
void makeMove(int row, int col, vector<vector<char>>& board, char player);

int main() {
    vector<vector<char>> board(3, vector<char>(3, ' '));
    char currentPlayer = 'X';
    int row, col;
    bool gameWon = false;

    while (true) {
        printBoard(board);
        cout << "Player " << currentPlayer << ", enter your move (row and column): ";
        cin >> row >> col;

        if (isValidMove(row, col, board)) {
            makeMove(row, col, board, currentPlayer);
            gameWon = checkWin(board, currentPlayer);
            if (gameWon) {
                printBoard(board);
                cout << "Player " << currentPlayer << " wins!" << endl;
                break;
            }
            if (checkDraw(board)) {
                printBoard(board);
                cout << "The game is a draw!" << endl;
                break;
            }
            currentPlayer = (currentPlayer == 'X') ? 'O' : 'X';
        } else {
            cout << "Invalid move, try again." << endl;
        }
    }

    return 0;
}

void printBoard(const vector<vector<char>>& board) {
    cout << "  0 1 2" << endl;
    for (int i = 0; i < 3; ++i) {
```

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

bool checkWin(const vector<vector<char>>& board, char player) {
    // Check rows and columns
    for (int i = 0; i < 3; ++i) {
        if ((board[i][0] == player && board[i][1] == player && board[i][2] == player) ||
            (board[0][i] == player && board[1][i] == player && board[2][i] == player)) {
            return true;
        }
    }
    // Check diagonals
    if ((board[0][0] == player && board[1][1] == player && board[2][2] == player) ||
        (board[0][2] == player && board[1][1] == player && board[2][0] == player)) {
        return true;
    }
    return false;
}

bool checkDraw(const vector<vector<char>>& board) {
    for (const auto& row : board) {
        for (char cell : row) {
            if (cell == ' ') {
                return false;
            }
        }
    }
    return true;
}

bool isValidMove(int row, int col, const vector<vector<char>>& board) {
    return row >= 0 && row < 3 && col >= 0 && col < 3 && board[row][col] == ' ';
}

void makeMove(int row, int col, vector<vector<char>>& board, char player) {
    board[row][col] = player;
}

```