**CODE FOR TIC TOE GAME**

#include <iostream>

using namespace std;

// Function prototypes

void displayBoard(char board[3][3]);

bool checkWin(char board[3][3], char player);

bool checkDraw(char board[3][3]);

bool isValidMove(char board[3][3], int row, int col);

void switchPlayers(char &currentPlayer);

int main() {

char board[3][3] = {{' ', ' ', ' '}, {' ', ' ', ' '}, {' ', ' ', ' '}};

char currentPlayer = 'X';

bool gameWon = false;

bool gameDraw = false;

cout << "Tic-Tac-Toe Game" << endl;

do {

int row, col;

displayBoard(board);

// Player Input

cout << "Player " << currentPlayer << ", enter your move (row and column): ";

cin >> row >> col;

// Adjust row and col to match the array index (starting from 0)

row--;

col--;

if (isValidMove(board, row, col)) {

// Update Board

board[row][col] = currentPlayer;

// Check for Win

if (checkWin(board, currentPlayer)) {

displayBoard(board);

cout << "Player " << currentPlayer << " wins!" << endl;

gameWon = true;

}

// Check for Draw

if (checkDraw(board)) {

displayBoard(board);

cout << "It's a draw!" << endl;

gameDraw = true;

}

// Switch Players

switchPlayers(currentPlayer);

} else {

cout << "Invalid move. Please try again." << endl;

}

} while (!gameWon && !gameDraw);

cout << "Thank you for playing!" << endl;

return 0;

}

// Function to display the Tic-Tac-Toe board

void displayBoard(char board[3][3]) {

cout << endl;

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 3; j++) {

cout << " " << board[i][j] << " ";

if (j < 2) cout << "|";

}

cout << endl;

if (i < 2) cout << "---+---+---" << endl;

}

cout << endl;

}

// Function to check if a player has won

bool checkWin(char board[3][3], char player) {

// Check rows, columns, and diagonals

for (int i = 0; i < 3; i++) {

if (board[i][0] == player && board[i][1] == player && board[i][2] == player) return true; // Check rows

if (board[0][i] == player && board[1][i] == player && board[2][i] == player) return true; // Check columns

}

if (board[0][0] == player && board[1][1] == player && board[2][2] == player) return true; // Check main diagonal

if (board[0][2] == player && board[1][1] == player && board[2][0] == player) return true; // Check secondary diagonal

return false;

}

// Function to check if the game is a draw

bool checkDraw(char board[3][3]) {

// Check if there are no empty cells

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 3; j++) {

if (board[i][j] == ' ') return false;

}

}

return true;

}

// Function to check if a move is valid

bool isValidMove(char board[3][3], int row, int col) {

// Check if the cell is empty and within the valid range

return (row >= 0 && row < 3 && col >= 0 && col < 3 && board[row][col] == ' ');

}

// Function to switch players

void switchPlayers(char &currentPlayer) {

if (currentPlayer == 'X') {

currentPlayer = 'O';

} else {

currentPlayer = 'X';

}

}

OUTPUT:



