## TIC TAC TOE

#include <stdio.h>

#include <stdlib.h>

#define SIZE 3

void initializematrix(char matrix[SIZE][SIZE]);

void displaymatrix(char matrix[SIZE][SIZE]);

int isValidMove(char matrix[SIZE][SIZE], int row, int col);

int makeMove(char matrix[SIZE][SIZE], int row, int col, char player);

int checkWin(char matrix[SIZE][SIZE], char player);

int checkDraw(char matrix[SIZE][SIZE]);

void gameLoop();

int main()

{

    gameLoop();

    return 0;

}

void initializematrix(char matrix[SIZE][SIZE])

{

    for (int i = 0; i < SIZE; i++)

    {

        for (int j = 0; j < SIZE; j++)

        {

            matrix[i][j] = ' ';

        }

    }

}

void displaymatrix(char matrix[SIZE][SIZE])

{

    printf("\n");

    for (int i = 0; i < SIZE; i++)

    {

        for (int j = 0; j < SIZE; j++)

        {

            printf(" %c ", matrix[i][j]);

            if (j < SIZE - 1)

            {

                printf("|");

            }

        }

        printf("\n");

        if (i < SIZE - 1)

        {

            printf("---|---|---\n");

        }

    }

}

int isValidMove(char matrix[SIZE][SIZE], int row, int col)

{

    return row >= 0 && row < SIZE && col >= 0 && col < SIZE && matrix[row][col] == ' ';

}

int makeMove(char matrix[SIZE][SIZE], int row, int col, char player)

{

    if (isValidMove(matrix, row, col))

    {

        matrix[row][col] = player;

    }

}

int checkWin(char matrix[SIZE][SIZE], char player)

{

    for (int i = 0; i < SIZE; i++)

    {

        if ((matrix[i][0] == player && matrix[i][1] == player && matrix[i][2] == player) ||

            (matrix[0][i] == player && matrix[1][i] == player && matrix[2][i] == player))

        {

            return 1;

        }

    }

    if ((matrix[0][0] == player && matrix[1][1] == player && matrix[2][2] == player) ||

        (matrix[0][2] == player && matrix[1][1] == player && matrix[2][0] == player))

    {

        return 1;

    }

    return 0;

}

int checkDraw(char matrix[SIZE][SIZE])

{

    for (int i = 0; i < SIZE; i++)

    {

        for (int j = 0; j < SIZE; j++)

        {

            if (matrix[i][j] == ' ')

            {

                return 0;

            }

        }

    }

    return 1;

}

void gameLoop()

{

    char matrix[SIZE][SIZE];

    initializematrix(matrix);

    char players[2] = {'X', 'O'};

    int currentPlayer = 0;

    int row, col;

    int gameWon = 0, gameDraw = 0;

    while (!gameWon && !gameDraw)

    {

        displaymatrix(matrix);

        printf("Player %c, enter your move (row and column): ", players[currentPlayer]);

        scanf("%d %d", &row, &col);

        fflush(stdin);

        if (isValidMove(matrix, row, col))

        {

            makeMove(matrix, row, col, players[currentPlayer]);

            gameWon = checkWin(matrix, players[currentPlayer]);

            if (gameWon)

            {

                displaymatrix(matrix);

                printf("Player %c wins!\n", players[currentPlayer]);

            }

            else

            {

                gameDraw = checkDraw(matrix);

                if (gameDraw)

                {

                    displaymatrix(matrix);

                    printf("The game is a draw!\n");

                }

                else

                {

                    currentPlayer = (currentPlayer + 1) % 2;

                }

            }

        }

        else

        {

            printf("Invalid move. Try again.\n");

        }

    }

}

## SPIRAL

#include <stdio.h>

#include <stdlib.h>

#define ROWS 4

#define COLS 4

void printMatrix(int matrix[ROWS][COLS]);

int main()

{

    int matrix[ROWS][COLS] = {

        {1, 2, 3, 4},

        {5, 6, 7, 8},

        {9, 10, 11, 12},

        {13, 14, 15, 16}};

    printMatrix(matrix);

    return 0;

}

void printMatrix(int matrix[ROWS][COLS])

{

    int top = 0, bottom = ROWS - 1, left = 0, right = COLS - 1;

    while (top <= bottom && left <= right)

    {

        // Print top row

        for (int i = left; i <= right; i++)

        {

            printf("%d ", matrix[top][i]);

        }

        top++;

        // Print right column

        for (int i = top; i <= bottom; i++)

        {

            printf("%d ", matrix[i][right]);

        }

        right--;

        // Print bottom row

        if (top <= bottom)

        {

            for (int i = right; i >= left; i--)

            {

                printf("%d ", matrix[bottom][i]);

            }

            bottom--;

        }

        // Print left column

        if (left <= right)

        {

            for (int i = bottom; i >= top; i--)

            {

                printf("%d ", matrix[i][left]);

            }

            left++;

        }

    }

}