#include <stdio.h>

#include <stdlib.h>

#include <time.h>

// 定义棋盘大小

#define BOARD\_SIZE 3

// 井字棋棋盘结构体

typedef struct {

char board[BOARD\_SIZE][BOARD\_SIZE];

} TicTacToeBoard;

// 初始化棋盘

void initBoard(TicTacToeBoard \*board) {

for (int i = 0; i < BOARD\_SIZE; i++) {

for (int j = 0; j < BOARD\_SIZE; j++) {

board->board[i][j] = ' '; // 使用空格表示空位置

}

}

}

// 打印棋盘

void printBoard(TicTacToeBoard board) {

printf(" 0 1 2\n");

for (int i = 0; i < BOARD\_SIZE; i++) {

printf("%d ", i);

for (int j = 0; j < BOARD\_SIZE; j++) {

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

if (j < BOARD\_SIZE - 1) {

printf("| ");

}

}

printf("\n");

if (i < BOARD\_SIZE - 1) {

printf(" -----\n");

}

}

}

// 判断是否有玩家获胜

int checkWin(TicTacToeBoard board, char player) {

// 检查行

for (int i = 0; i < BOARD\_SIZE; i++) {

for (int j = 0; j < BOARD\_SIZE; j++) {

if (board.board[i][j] != player) break;

if (j == BOARD\_SIZE - 1) return 1; // 一行全部为当前玩家，获胜

}

}

// 检查列

for (int j = 0; j < BOARD\_SIZE; j++) {

for (int i = 0; i < BOARD\_SIZE; i++) {

if (board.board[i][j] != player) break;

if (i == BOARD\_SIZE - 1) return 1; // 一列全部为当前玩家，获胜

}

}

// 检查对角线

if (board.board[0][0] == player && board.board[1][1] == player && board.board[2][2] == player) return 1; // 主对角线

if (board.board[0][2] == player && board.board[1][1] == player && board.board[2][0] == player) return 1; // 副对角线

return 0;

}