井字棋

实验内容

井字棋

实验代码

#include<stdio.h>

#include<stdlib.h>

#include<stdbool.h>

unsigned short chessboard[3][3]; //棋盘,1代表空格，2代表玩家，5代表AI

unsigned short scores[3][3]; //AI评分数组

void playerSet(); //玩家下棋

void AISet(); //电脑下棋

void display(); //显示棋盘

bool isFull(); //棋盘是否已满

bool isWin(); //是否有人获胜

void initChessboard(); //初始化棋盘

int main(void)

{

initChessboard();

display();

while (!isWin())

{

if (isFull())

{

puts("平局！");

exit(0);

}

playerSet();

AISet();

display();

}

system("pause");

return 0;

}

void initChessboard() //初始化棋盘

{

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

{

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

{

chessboard[i][j] = 1;

}

}

}

bool isFull() //棋盘是否已满

{

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

{

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

{

if (chessboard[i][j] == 1)

{

return false;

}

}

}

return true;

}

bool isWin() //是否有人获胜

{

int sum = 0;

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

{

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

{

sum += chessboard[i][j];

}

if (sum == 6)

{

puts("人类获胜");

return true;

}

if (sum == 15)

{

puts("电脑获胜");

return true;

}

sum = 0;

}

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

{

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

{

sum += chessboard[j][i];

}

if (sum == 6)

{

puts("人类获胜");

return true;

}

if (sum == 15)

{

puts("电脑获胜");

return true;

}

sum = 0;

}

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

{

sum += chessboard[i][i];

if (sum == 6)

{

puts("人类获胜");

return true;

}

if (sum == 15)

{

puts("电脑获胜");

return true;

}

sum = 0;

}

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

{

sum += chessboard[i][2 - i];

if (sum == 6)

{

puts("人类获胜");

return true;

}

if (sum == 15)

{

puts("电脑获胜");

return true;

}

sum = 0;

}

return false;

}

void display() //显示棋盘

{

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

{

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

{

switch (chessboard[i][j])

{

case 1:

printf(" ");

break;

case 2:

printf("X");

break;

case 5:

printf("O");

break;

}

}

printf("\n");

}

}

void playerSet() //玩家下棋

{

int x, y;

puts("请输入坐标X和Y:");

scanf\_s("%d%d", &x, &y);

while (chessboard[3 - y][x - 1] != 1)

{

fprintf(stderr, "THIS CELL HAS BEEN SETED!\n");

scanf\_s("%d%d", &x, &y);

}

chessboard[3 - y][x - 1] = 2;

}

void AISet() //电脑下棋

{

int sum = 0;

for (int i = 0; i < 3; i++) //初始化评分数组

{

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

{

scores[i][j] = 0;

}

}

for (int i = 0; i < 3; i++) //对每一行进行处理

{

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

{

sum += chessboard[i][j];

}

switch (sum)

{

case 3: //1+1+1 优先级2 权重5

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

{

if (chessboard[i][j] == 1)

{

scores[i][j] += 5;

}

}

break;

case 4: //1+1+2 优先级1 权重25

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

{

if (chessboard[i][j] == 1)

{

scores[i][j] += 25;

}

}

break;

case 7: //1+1+5 优先级3 权重30

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

{

if (chessboard[i][j] == 1)

{

scores[i][j] += 30;

}

}

break;

case 5: //1+2+2 电脑要输，优先级次高 4 权重300

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

{

if (chessboard[i][j] == 1)

{

scores[i][j] += 300;

}

}

break;

case 11: //1+5+5 电脑要赢，优先级最高 5 权重1000

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

{

if (chessboard[i][j] == 1)

{

scores[i][j] += 1000;

}

}

break;

}

sum = 0;

}

for (int j = 0; j < 3; j++) //对每列进行处理

{

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

{

sum += chessboard[i][j];

}

switch (sum)

{

case 3: //1+1+1 优先级2 权重5

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

{

if (chessboard[i][j] == 1)

{

scores[i][j] += 5;

}

}

break;

case 4: //1+1+2 优先级1 权重25

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

{

if (chessboard[i][j] == 1)

{

scores[i][j] += 25;

}

}

break;

case 7: //1+1+5 优先级3 权重30

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

{

if (chessboard[i][j] == 1)

{

scores[i][j] += 30;

}

}

break;

case 5: //1+2+2 电脑要输，优先级次高 4 权重300

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

{

if (chessboard[i][j] == 1)

{

scores[i][j] += 300;

}

}

break;

case 11: //1+5+5 电脑要赢，优先级最高 5 权重1000

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

{

if (chessboard[i][j] == 1)

{

scores[i][j] += 1000;

}

}

break;

}

sum = 0;

}

for (int i = 0; i < 3; i++) //斜线

{

sum += chessboard[i][i];

}

switch (sum)

{

case 3: //1+1+1 优先级2 权重5

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

{

if (chessboard[i][i] == 1)

{

scores[i][i] += 5;

}

}

break;

case 4: //1+1+2 优先级1 权重25

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

{

if (chessboard[i][i] == 1)

{

scores[i][i] += 25;

}

}

break;

case 7: //1+1+5 优先级3 权重30

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

{

if (chessboard[i][i] == 1)

{

scores[i][i] += 30;

}

}

break;

case 5: //1+2+2 电脑要输，优先级次高 4 权重300

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

{

if (chessboard[i][i] == 1)

{

scores[i][i] += 300;

}

}

break;

case 11: //1+5+5 电脑要赢，优先级最高 5 权重1000

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

{

if (chessboard[i][i] == 1)

{

scores[i][i] += 1000;

}

}

break;

}

sum = 0;

for (int i = 0; i < 3; i++) //反斜线

{

sum += chessboard[i][2 - i];

}

switch (sum)

{

case 3: //1+1+1 优先级2 权重5

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

{

if (chessboard[i][2 - i] == 1)

{

scores[i][2 - i] += 5;

}

}

break;

case 4: //1+1+2 优先级1 权重25

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

{

if (chessboard[i][2 - i] == 1)

{

scores[i][2 - i] += 25;

}

}

break;

case 7: //1+1+5 优先级3 权重30

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

{

if (chessboard[i][2 - i] == 1)

{

scores[i][2 - i] += 30;

}

}

break;

case 5: //1+2+2 电脑要输，优先级次高 4 权重300

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

{

if (chessboard[i][2 - i] == 1)

{

scores[i][2 - i] += 300;

}

}

break;

case 11: //1+5+5 电脑要赢，优先级最高 5 权重1000

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

{

if (chessboard[i][2 - i] == 1)

{

scores[i][2 - i] += 1000;

}

}

break;

}

sum = 0;

int maxI = 0, maxJ = 0;

for (int i = 0; i < 3; i++) //找出权重最大的格子

{

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

{

if (scores[maxI][maxJ] < scores[i][j])

{

maxI = i;

maxJ = j;

}

}

}

chessboard[maxI][maxJ] = 5;

}

实验结果：

