课程设计报告

**课题名称： 五子棋设计**

**姓 名： 李逸欣 黄曼玲**

**学 号： 1800502125 1800502143**

**专 业：计算机科学与技术（外包特色班）**

**班 级： 外包181**

**指导教师： 熊伟老师**

**目 录**

第一部分 课程设计报告…………………………………………………………3

第一章 课程设计目的…………………………………………………3

第二章 课程设计内容和要求…………………………………………4

2.1 问题描述………………………………………………4

2.2 设计要求………………………………………………4

第三章 课程设计总体方案及分析……………………………………4

3.1 问题分析………………………………………………4

3.2 概要设计………………………………………………7

3.3 详细设计………………………………………………8

3.4 调试分析………………………………………………12

3.5 测试结果………………………………………………13

3.6 参考文献………………………………………………17

第二部分 课程设计总结…………………………………………………………19

附录(源代码)……………………………………………………………………20

**第二部分 课程设计报告**

1. **课程设计目的**

本次实训通过用编程语言实现一个简单的五子棋程序，充分理解数据结构的数组思想，且在五子棋

实际操作中得以充分运用，得以巩固这种数据结构的构造方法。在实际系统中使用且实现初步的人工智能的相关算法，进一步加深对人工智能算法的理解。

**第二章  课程设计内容和要求**

2.1问题描述：

五子棋要求绘制15\*15的棋盘，通过鼠标点击能进行黑白棋子交替下，且棋子的落子点必须在十字交叉处上，若该点已有棋子，则弹窗提醒。判断输赢方面要求能够判断相同棋子是否能连成五子，同时能进行悔棋且不可多步悔棋，若多步悔棋，则有弹窗提醒。实现棋盘的数据存储与读取，同时重新新的一局时棋盘数据全都初始化。界面方面要实现可选择人人对战和人机对战的2种玩法模式，同时人机对战方面要体现出人工智能算法思想进行下棋。

2.2设计要求：

设计程序输出如下：

(1) 建立一个15\*15的棋盘并且在屏幕上显示出来。

（2）实现能在屏幕上通过鼠标点击进行落子。

（3）在屏幕上能够实现黑白棋子交替下棋。

（4）棋子能够准确的落在十字交叉处。

（5）若棋盘该点已有棋子，则弹窗提醒。

（6）判断输赢要求能够判断是否相同颜色连成五子。

（7）实现悔棋功能。

（8）若用户多步悔棋，则弹窗提醒。

（9）对棋盘的数据进行存储与读取。

（10）每次胜负则进行弹窗提示，点击确认清除本局的数据并返回主界面。

（11）人机对战方面体现初步的人工智能思想算法。

（12）在主界面实现人人对战和人机对战两种玩法的选择。

（13）主界面上添加游戏说明和音乐开关的功能。

**第三章      课程设计总体方案及分析**

3.1 问题分析：

1.棋盘的建立：通过for循环和调用graphics.h头文件的line函数分别画出横竖各为15的棋盘；

2.鼠标实现落子：先定义鼠标的消息MOUSEMUG M,定义一个变量获取一条鼠标消息，设置点击左键时，调用函数setifillcolor填棋子颜色，solidcircle画半径为10的棋子，从而实现落子。

3.实现黑白棋子交替落子：每次下一步棋就定义flag记录步数，如果步数%2==1，表示白棋下，如果步数%2==0,代表黑棋下，从而实现黑白棋子交替落子。

4.棋子能够准确落在十字交叉点处：定义2个for循环遍历棋盘，定义2个变量分别记录鼠标当前的横纵坐标，一个格子的长度为25，如果鼠标坐标减去附件的线的坐标的绝对值小于12，即鼠标当前的横纵坐标更靠近哪条线，则落在该十字交叉处。

5.提醒该点已有棋子：定义一个二维数组board，每次下棋进行记录，如果是黑棋，则记录为1，且记录当前棋子的横纵坐标；如果为白棋，则记录为2，切记录当前的棋子的横纵坐标；鼠标进行点击，如果该点的board不为空，则调用函数MessageBox进行弹窗提醒。

6.判断输赢（即相同颜色是否连成五子）：对棋子的8个方向进行遍历，如遍历当前该点的横方向，用for循环从该子的前4个点进行循环判断，如果前四个位置到第五个位置的棋子颜色相同则说明连成五子，返回值为1；如果不是，则进行后一个点的遍历，竖方向、左斜方向、右斜方向的遍历查询方法同理。

7.实现悔棋功能；分为人人悔棋和人机悔棋；通过设置右键进行悔棋。人人悔棋：定义一维数组rrhuiqi分别记录当前最近下的棋子的横纵坐标。如果鼠标右键该点不为空，则让该点的board==0，重绘棋盘，遍历整个棋盘，根据原先board记录的数值画出对应的棋子颜色，步数撤回一步；人机悔棋：定义二维数组rjhuiqi分别记录玩家和电脑当前最近下的棋子的位置，鼠标右键，board==0，电脑的上一步进行撤回，重绘棋盘，遍历整个棋盘，根据原先board记录的数值画出对应的棋子颜色，步数撤回两步。如果该点为空或不在棋盘内下棋，则弹窗显示悔棋失败。

8.不允许多步悔棋：在人人对战中定义一维数组rrhuiqi记录当前最近的棋子的横纵坐标，当要悔棋时，只会悔rrhuiqi记录的坐标的棋子位置；人机对战中定义二维数组rjshuzu分别记录当前最近的玩家和机器的棋子的横纵坐标，当要悔棋时，只会悔rjshuzu记录的坐标的棋子位置。若要多步悔棋，则会进行弹窗提醒。

9.对棋盘数据进行存储和读取：board数组记录棋盘棋子的状态，0为该点为空，1为该点为黑棋，2为该点为白棋；score数组，人机对战人工智能思想算法权值法计算棋盘上空子的权值后放入到score数组，然后通过遍历棋盘算出权值最大的score；rrhuiqi是人人对战记录当前最近的棋子的位置；rjhuiqi是人机对战记录当前最近的棋子的位置。

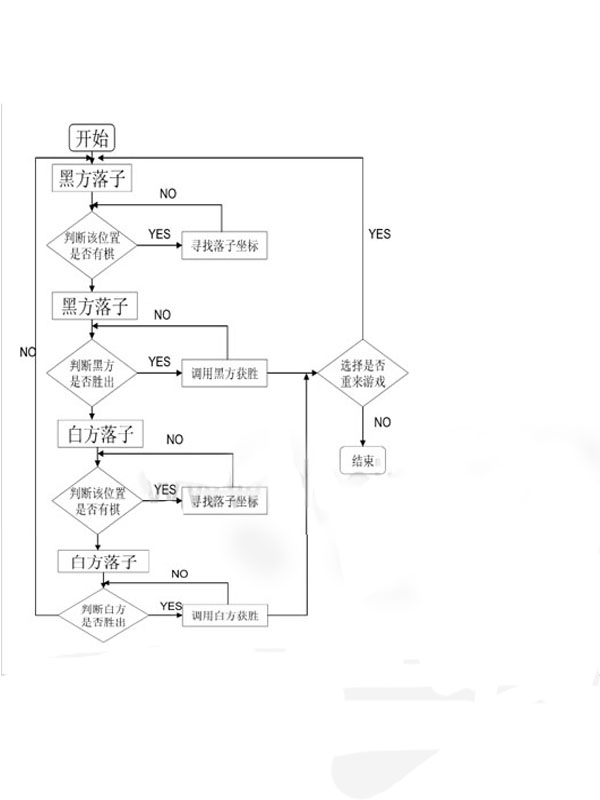
10.胜负提示并返回主界面：调用judge函数判断输赢，胜负已出弹窗，遍历整个棋盘使其数值初始化，调用主界面函数返回主界面。

11.人机对战算法思想：本次实验采用的方法为权值法。用2个for循环对棋盘的某个点的空子进行8个方向遍历，对人落子进行评分，要对人所下的子进行围堵，玩家在这儿落子后的估分越高，电脑就越有必要下在这儿，来打破玩家的意图。如果某个方向有玩家的棋子，则number1加1，如果遇到空子，empty加1，并跳出循环；一个方向的正反方向进行遍历，number1=1，权值为1；死三子：number1=2，empty=1，score为5；活三子：number1=2，empty=2，score=10；死四：number1=3，empty=1，score=20；活四：number1=3，empty=2，score=100；五子：number1=4，score=1000；

电脑再对自己落该空子进行评分，对该子的8个方向进行遍历，如果某个方向相同的棋子，则number2加1，如果遇到空子，empty加1，并跳出循环；一个方向的正反方向进行遍历，number2=1，权值为2；死三子：number2=2，empty=1，score为8；活三子：number2=2，empty=2，score=30；死四：number2=3，empty=1，score=50；活四：number2=3，empty=2，score=200；五子：number1=4，score=10000；然后遍历整个棋盘，比较出score最高的位置，电脑则下该点。

12.主界面实现人人对战和人机对战选择：先画出点击范围，分别在该范围内调用人人对战和人机对战的函数，点击鼠标，实现页面跳转。

13.主界面增添游戏说明和音乐开关功能；规定鼠标点击范围，在该范围内调用函数MCISendString实现音乐的开关，open和play表示开，pause表示暂停。在规定的范围内点击鼠标，调用youxishuoming函数，可以看到游戏说明。



3.2 概要设计

1. (1) 建立一个15\*15的棋盘并且在屏幕上显示出来。调用line函数进行棋盘绘制。

（2）设置鼠标监听实现能在屏幕上点击进行落子。

（3）定义flag每一步下棋就加1，通过余数实现黑白棋子交替下棋。

（4）棋子位置更靠近哪条线，则落在该点十字交叉处。

（5）若棋盘该点已有棋子，board不等于0，则弹窗提醒。

（6）每下一步棋进行8个方向遍历来判断是否相同颜色连成五子。

（7）右键实现悔棋，定义数组记录当前最近落子位置，实现悔棋。

（8）定义数组记录当前最近的落子位置，若用户多步悔棋，则弹窗提醒。

（9）定义二维数组board和score对棋盘的数据进行存储与读取。

（10）每次胜负则进行弹窗提示，点击确认清除本局的数据并返回主界面。

（11）人机对战方面用权值法体现初步的人工智能思想算法。

（12）在主界面实现人人对战和人机对战两种玩法的选择。

（13）主界面上添加游戏说明和音乐开关的功能。

2.本程序包含9个函数：

(1)主函数 main()

(2)主界面窗口函数 initGame()

(3)游戏说明函数void youxishuoming()

(4)初始化游戏棋盘界面void ChessBorad()

(5)人人对决函数void manvsmanplayChess()

(6)判断输赢int judge()

(7)每个空子的评估函数 权值法void evaluate\_naive()

(8)电脑获取最优解位置void Chess\_ai\_naive()

(9)人机下棋void manvsaiplayChess()

3.本程序调用5个头文件

#include<graphics.h>//图形库头文件

#include<math.h>//数学函数头文件

#include<mmsystem.h>//倒入声音头文件

#pragma comment(lib,"winmm.lib")//播放音乐库头文件

#include<windows.h>;//显示窗口头文件

3.3 详细设计

实现概要设计中定义的所有数据类型及操作的伪代码算法

1. 节点类型

五子棋棋盘状态矩阵类型：int board[15][15];为方便操作使其为全局变量

五子棋棋盘分数矩阵类型：int score[15][15];

人人悔棋一维数组：int rrhuiqi[2];

人机悔棋二维数组：int rjhuiqi[2][2];

2.人人对战的操作

void manvsmanplayChess(){

(1)生成空棋盘

void ChessBorad() {

定义i,j为循环变量

for(i<=m)

for(j<=n)

循环画线

}

(2)落子点的判断

while (1){

定义i,j为循环变量

for(i<=m)

for(j<=n)

通过数学库<math.h>中对鼠标点击处到棋盘交点处距离的绝对值小于每个棋盘格子长度的一半的判断来确保落子点在棋盘交界线处。

定义x，y变量记录落点距离；

定义a，b变量记录落子在棋盘上的位置；

}

(3)玩家交替下棋

如果 鼠标消息为鼠标左键在棋盘区域内按下时

定义一个变量flag，利用flag不断+1时奇偶性地不断变化来判断轮到哪个玩家下

如果 棋盘上有棋子board[a][b] != 0时，则提示不能下棋

如果 轮到黑棋玩家下棋，flag为偶数则黑棋下 在玩家落点处画个黑色实心圆。

如果 轮到白棋玩家下棋，flag为奇数则白棋下 在玩家落点处画个白色实心圆。

flag++；

(4)玩家悔棋

将下棋的横纵坐标保存在一维数组rrhuiqi中

如果右键鼠标落在棋盘区域，判断棋盘有子处且是最近的落子点，即只能是悔最近的棋子

即board[a][b] != 0 && (rrhuiqi[0] == a && rrhuiqi[1] == b)

悔棋后重置board数组该点处为空 board[a][b] = 0;

初始化棋盘 ChessBorad();

重绘棋盘 定义i，j变量

for(i<=m)

for(j<=n)

根据board数组绘制上一步局势

flag--；

如果鼠标落在无子或不在棋盘区域，则提示悔棋失败

(5)赢家显示消息

(flag -1)% 2 若为奇数则提示白棋落子后产生赢家，白棋获胜

(flag -1)% 2 若为偶数则提示黑棋落子后产生赢家，黑棋获胜

如果 没有赢家产生而flag=225，则提示平局，游戏结束

1. 返回大厅

如果 鼠标左键点击在返回大厅区域，清空所有数组数据memset(board, 0, sizeof(board))，flag=0；初始化游戏，跳至游戏初始化界面

3.人机下棋

void manvsaiplayChess() {

(1)人机交替下棋

flag偶数时玩家下，rjhuiqi数组记录玩家下棋x，y坐标，判断有无赢家产生，有无平局

flag奇数时电脑下，调用Chess\_ai\_naive(int& x, int& y)函数，rjhuiqi数组记录电脑下棋x，y坐标，判断有无赢家产生，有无平局

(2)人机悔棋

如果玩家悔棋棋子为玩家最后一步棋子的位置，即悔棋的x，y坐标与rjhuiqi数组里的玩家x，y坐标相同

撤回board数组中玩家该步的记录，及相应的电脑下棋该步后的记录，重新绘制棋局

否则

提示悔棋失败

}

4.电脑搜索最有解

void Chess\_ai\_naive(int& x, int& y){

调用evaluate\_naive();

定义变量i，j，maxpos

for(i<=m)

for(j<=n)

搜索score数组，找出分数最高的点，记录并返回

}

5.电脑权值法评估电脑空子分数

void evaluate\_naive(){

需要对人落子和电脑已落子进行评价。人落子电脑要堵，电脑自己落子要做成5子

清空score数组，即把sizeof(score)长度的score数组里的值全为0

定义变量number1为方向上人子数，number2为方向上电脑子数，empty为空子数

for(x<=m)

for(y<=n)

评估每个空子board[x][y] == 0时

利用<-1,0,1>与<-1,0,1>的笛卡尔积来判断每个空位的8个方向（0,0不动不用算）

1. 对人落子评分

每个方向上的正方向：查看空子在该方向上的五个相邻位置，玩家棋子有多少个记录在number1，遇到空子记录在empty则跳出循环，遇到边界或敌方棋子也跳出循环

每个方向上的反方向：查看空子在该方向上的五个相邻位置，玩家棋子有多少个记录在number1，遇到空子记录在empty则跳出循环，遇到边界或敌方棋子也跳出循环

根据棋子数number1及空子数empty,为score数组在该空子位置打上相应的分数

1. 对电脑落子评分

操作与对人落子评分同理

}

6.判断输赢机制

int judge(int a, int b){

判断横方向的5个棋子是否连成

判断竖方向的5个棋子是否连成

判断斜左上方的5个棋子是否连成

判断斜右上方的5个棋子是否连成

}

7.游戏说明

void youxishuoming() {

放置游戏说明背景图

如果 鼠标左键落在返回大厅界面区域，这跳至initGame();

}

8.菜单选择

while (true)

{

switch (m.uMsg)

{

case 当在“客户区”按下鼠标的消息：

如果 鼠标点击在玩家对战区域

{

跳至棋盘界面初始化函数

跳至人人对战函数

}

如果 鼠标点击在人机对战区域

{

跳至棋盘界面初始化函数

跳至人机对战函数

}

如果 鼠标点击在音乐开区域

{

设置背景音乐 多媒体控制接口

}

如果 鼠标点击在音乐关区域

{

关闭音乐函数

}

如果 鼠标点击在游戏说明区域

{

游戏说明函数

}

如果 鼠标点击在退出游戏区域

{

退出游戏代码

}

}}}

注：具体源代码见附录

3.4 调试分析

在调试过程中：

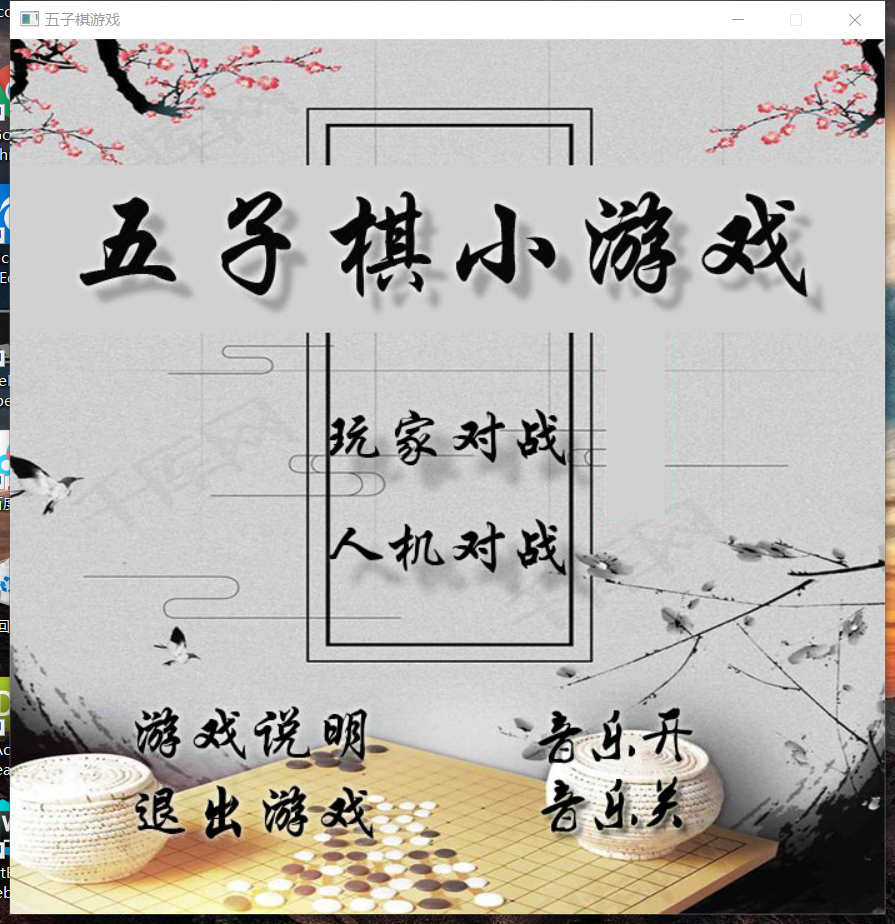
悔棋部分一开始是设计在玩家点击处若有棋，则可悔棋，此设计导致悔棋可以无限悔棋，没有步数限制，可以跨步悔棋，不按玩家下棋顺序，后改为人人下棋时用大小为2的一维数组储存玩家下的最后一步棋的横纵坐标，人机下棋则用大小为4的二维数组分别存放玩家，电脑最后一步棋子的横纵坐标，并且限制每次悔棋点是最后一步下棋点则可悔棋。

人机电脑下棋点判断部分一开始是设计为扫描每个空子的八个方向上五个相邻格子的同色棋子个数及空子数来得到该空子处相应的分数，总体分数最高处为电脑要下的地方，但没有设计要判断一个方向上的正反方向导致一个方向上的棋子局势被分割开来，而使得判断不够全面。后来设计了一个方向上的正反方向的棋子与空子数一同判断，才使电脑下棋点更为可靠。

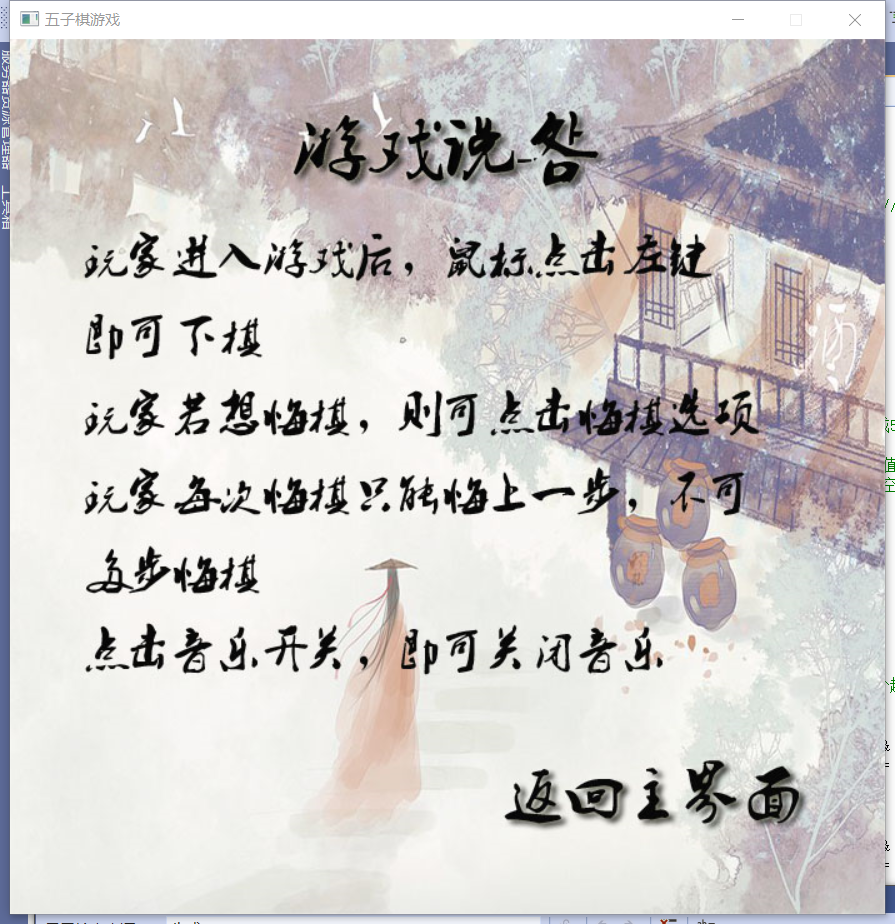
人机下棋电脑部分一开始是只对玩家的棋局势进行判断，没有对电脑自己局势进行判断，导致电脑永远在堵玩家的路，而不会自己创造优势，后来改为玩家，电脑局势都判断。

3.5 测试结果

1.游戏选择功能界面



2.游戏说明

****

3.玩家对战

****

****

****

4.人机对战



3.6 参考文献

【1】 易法令 《数据结构教程》 武汉大学出版社， 2016年8月

【2】 何钦铭 颜晖 《C语言程序设计》 高等教育出版社 2015年8月

**第二部分 课程设计总结**

在实际的上机操作过程中，不仅是让我们了解数据结构的理论知识，更重要的是培养解决实际问题的能力，所以相信通过此次实习可以提高我们分析设计能力和编程能力，为后续课程的学习及实践打下良好的基础。在学习完数据结构一年后，我们进行了第一次数据结构的实训，小组合作做一个小游戏，一开始，还没有实训之前，对于数据结构的理解还不是很深，对该课程的印象仅仅只是能懂，知道有什么数据储存类型，知道有什么算法。但仍然觉得算法思想抽象难以理解，不了解结构与算法应用在何处，不清楚算法思想与程序之间的转换，理解算法但不知道如何编程。之前每次做课堂上的练习仅仅只是停留在会用那些数据结构的功能，头文件。而经过短短一个月的实训，我们对数据结构与算法有了更深刻的理解，知道了什么情况下用什么类型的数据结构来储存比较好，也更了解各种数据结构最基础的设计思想，并且也对算法有了一定的认识。由于我们选择的是五子棋小游戏，在编写人机对战一程序中，深刻理解了电脑下棋的设计思想。并且也是对人工智能ai有了初步的认识，我们不仅在其中学习到了棋类人工智能不仅可以运用权值法的设计思想，更可以是博弈的设计思想，还可以是神经网络的算法。通过这我也认识到了人工智能ai的强大。以前老是认为自己所学的知识太基础，不能运动到很多实际的项目中，而通过这次实训，我们深有体会，复杂的项目的实训都是基础的知识，一步一步的实现一点一点的小功能，就能慢慢地迈向成功。

附录：

#include<graphics.h>//图形库头文件

#include<math.h>

#include<mmsystem.h>//倒入声音头文件

#pragma comment(lib,"winmm.lib")//播放音乐库头文件

#include<windows.h>;

int flag = 0;//表示下棋次数

int board[15][15] = { 0 };//初始化棋盘的状态

int score[15][15] = { 0 };//初始化棋盘的分数

int rrhuiqi[2];

int rjhuiqi[2][2];

int judge(int a, int b);

void Chess\_ai\_naive(int& x, int& y);//电脑获取最优解位置

void evaluate\_naive();

void youxishuoming();

void manvsmanplayChess();

void manvsaiplayChess();

void ChessBorad();

//游戏大厅设计

void initGame()

{

HWND hWnd = GetHWnd();//获取窗口句柄，即窗口的唯一标识符

SetWindowText(hWnd, "五子棋游戏");//改变窗口标题

IMAGE backgrand;//创建一个图像对象

loadimage(&backgrand, "背景1.jpg", 700, 700);//装载光标，图标，位图，此为装载游戏大厅背景图

putimage(0, 0, &backgrand);//在屏幕上输出一个位图

MOUSEMSG m; // 定义鼠标消息

while (true)

{

// 获取一条鼠标消息

m = GetMouseMsg();

//避免黑屏

//putimage(0, 0, &backgrand);

switch (m.uMsg)

{

case WM\_LBUTTONDOWN: //当在“客户区”按下鼠标的消息

if (m.x >= 250 && m.x <= 450 && m.y >= 270 && m.y <= 340) //玩家对战区域

{

ChessBorad(); //棋盘界面初始化

manvsmanplayChess();//人人对战

}

if (m.x >= 250 && m.x <= 450 && m.y >= 370 && m.y <= 440) //人机对战区域

{

ChessBorad();

manvsaiplayChess();//人机对战

}

if (m.x >= 430 && m.x <= 620 && m.y >= 520 && m.y <= 575) //音乐开区域

{

mciSendString("open 音乐.mp3", 0, 0, 0);//设置背景音乐 多媒体控制接口

mciSendString("play 音乐.mp3", 0, 0, 0);

}

if (m.x >= 430 && m.x <= 620 && m.y >= 590 && m.y <=650) //音乐关区域

{

mciSendString("pause 音乐.mp3", NULL, 0, NULL);

}

if (m.x >= 150 && m.x <= 350 && m.y >= 520 && m.y <= 580) {//游戏说明

youxishuoming();

}

if (m.x >= 150 && m.x <= 350 && m.y >= 600 && m.y <= 700) {//退出游戏

exit(0);

}

}

}

}

//游戏说明函数

void youxishuoming() {

IMAGE h2;

loadimage(&h2, "背景3.jpg", 700, 700);

putimage(0, 0, &h2);

MOUSEMSG mg;

while (true) {

mg = GetMouseMsg();

//避免黑屏

putimage(0, 0, &h2);

switch (mg.uMsg)

{

case WM\_LBUTTONDOWN:

if (mg.x >= 400 && mg.x <= 660 && mg.y >= 450 && mg.y <= 600)//返回大厅界面区域

{

initGame();

}

}

}

}

//初始化游戏棋盘界面

void ChessBorad() {

loadimage(NULL, "背景2.jpg", 700, 700);

setlinecolor(BLACK);//画线条为黑色

for (int i = 175; i <550; i += 25)//循环画线

{

line(i, 175, i, 525);//画线函数，画竖线：第一个点（i，175），第二个点（i，525）

line(175, i, 525, i);//画横线，第一个点（175，i），第二个点（525，i）

setfillcolor(BLACK);

solidcircle(350, 350, 3);

solidcircle(250, 250, 3);

solidcircle(450, 250, 3);

solidcircle(250, 450, 3);

solidcircle(450, 450, 3);

}

}

//人人对决函数

void manvsmanplayChess()

{

MOUSEMSG m;//保存鼠标消息

int x, y;

int a = 0, b = 0;

while (1)

{

m = GetMouseMsg();

for (int i = 0; i <=14; i++)//每行15个交点可以下，每格长宽25

{

for (int j = 0; j <=14; j++)

{

if (abs(m.x - (175 + (i \* 25))) < 12 && abs(m.y - (175 + (j \* 25))) < 12)//使得落子在交点上

{

x = 175 + i \* 25;//第一条线在150，边边线不下，从第二条线开始即175

y = 175 + j \* 25;

a = i;

b = j;//用a,b记录鼠标点击位置方便后续判断

}

}

}

//下棋

if (m.uMsg == WM\_LBUTTONDOWN && (m.x > 162 && m.x <537 && m.y >162 && m.y <537))//鼠标消息为鼠标左键在棋盘区域内按下时

{

if (board[a][b] != 0)//不能重复下棋

{

MessageBox(NULL, "这里有棋子了请重新选择", "提示", MB\_OK);//提示框，null为消息框没有拥有窗口

continue;

}

if (flag % 2 == 0)//利用flag不断+1时奇偶性地不断变化来判断轮到哪个玩家下

{

setfillcolor(BLACK);//当flag为偶数则黑棋下

solidcircle(x, y, 10);//获得鼠标x，y为画一个圆的圆心，半径10

board[a][b] = 1;//玩家1在棋盘的（a,b）位置下了棋，则设置board为1，表示该位置玩家1有棋

}

else

{

setfillcolor(WHITE);

solidcircle(x, y, 10);

board[a][b] = 2;//玩家2下棋记录为2

}

flag++;//每次下一个棋，flag就加1

rrhuiqi[0] = a;//利用一维数组来储存最近的一个落子点，rrhuiqi[0]记录落子点x值

rrhuiqi[1] = b;//rrhuiqi[1]记录落子点y值

}

//悔棋

else if (m.uMsg == WM\_RBUTTONDOWN && (m.x > 162 && m.x <537 && m.y > 162 && m.y <537)) {//右键鼠标落在棋盘区域

if (board[a][b] != 0 && (rrhuiqi[0] == a && rrhuiqi[1] == b)) {//棋盘有子处且是最近的落子点，即只能是悔最近的棋子，不能无限悔棋，不能跨步悔棋

board[a][b] = 0;//悔棋后重置board数组该点处为空

ChessBorad();//重绘棋盘上未落上一棋的局势

for (int i = 0; i <=14; i++) {

for (int j = 0; j <=14; j++) {

if (board[i][j] == 1) {

setfillcolor(BLACK);

solidcircle(175 + (i \* 25), 175 + (j \* 25), 10);

}

if (board[i][j] == 2) {

setfillcolor(WHITE);

solidcircle(175 + (i \* 25), 175 + (j \* 25), 10);

}

}

}

flag--;//悔棋玩家再下一次

}

//无子或不在棋盘区域

else {

MessageBox(NULL, "悔棋失败", "提示", MB\_OK);//提示框

continue;

}

}

if (judge(a, b))//每下一子判断是否产生赢家

{

if ((flag -1)% 2 == 1)//白棋落子后产生赢家

{

MessageBox(NULL, "白棋获胜", "游戏结束", MB\_OK);

memset(board, 0, sizeof(board));

flag = 0;

initGame();

}

else

{

MessageBox(NULL, "黑棋获胜", "游戏结束", MB\_OK);

memset(board, 0, sizeof(board));

flag = 0;

initGame();

}

}

if(!judge(a,b)&&flag==225) {

MessageBox(NULL, "平局", "游戏结束", MB\_OK);

memset(board, 0, sizeof(board));

flag = 0;

initGame();

}

if ((m.uMsg == WM\_LBUTTONDOWN)&& m.x >= 450 && m.x <= 625 && m.y >= 50 && m.y <= 100)//点击返回大厅

{

memset(board, 0, sizeof(board));

flag = 0;

initGame();

}

}

}

//判断输赢

int judge(int a, int b)

{

int i, j;

int t = 2 - flag % 2;//玩家1时，board记录也为1，玩家2则board也为2

for (i = a - 4, j = b; i <= a; i++)//判断横方向的5个棋子是否连成

{

if (i >= 0 && i <= 11 && t == board[i][j] && t == board[i + 1][j] && t == board[i + 2][j] && t == board[i + 3][j] && t == board[i + 4][j])

return 1;

}

for (i = a, j = b - 4; j <= b; j++)//判断竖方向的5个棋子是否连成

{

if (j >= 0 && j <= 11 && t == board[i][j] && t == board[i][j + 1] && t == board[i][j + 2] && t == board[i][j + 3] && t == board[i][j + 4])

return 1;

}

for (i = a - 4, j = b - 4; i<= a && j <= b; j++, i++)//判断斜左上方的5个棋子是否连成

{

if (j >= 0 && j <= 11 && i >= 0 && i <= 11 && t == board[i][j] && t == board[i + 1][j + 1] && t == board[i + 2][j + 2] && t == board[i + 3][j + 3] && t == board[i + 4][j + 4])

return 1;

}

for (i = a +4, j = b -4; i>= a && j <= b; i--, j++)//判断斜右上方的5个棋子是否连成

{

if (j >= 0 && j <= 11 && i >= 0 && i <= 11 && t == board[i][j] && t == board[i - 1][j +1] && t == board[i - 2][j + 2] && t == board[i - 3][j + 3] && t == board[i - 4][j + 4])

return 1;

}

return 0;

}

//每个空子的评估函数 权值法

void evaluate\_naive() //需要对人落子和电脑已落子进行评价。人落子电脑要堵，电脑自己落子要做成5子

{

memset(score, 0, sizeof(score));//清空score数组，即把sizeof(score)长度的score数组里的值全为0

int number1, number2, empty;//number1为方向上人子数，number2为方向上电脑子数，empty为空子数

for (int x = 0; x <=14; ++x)

for (int y = 0; y <=14; ++y)

if (board[x][y] == 0)//评估每个空子

for (int i = -1; i <= 1; ++i)

for (int j = -1; j <= 1; ++j)//每个空位的8个方向

if (!(i == 0 && j == 0))//0,0不动不用算

{

int k = 0;

number1 = 0, number2 = 0, empty = 0;

//对人落子评分（要对人所下的子进行围堵，玩家在这儿落子后的估分越高，电脑就越有必要下在这儿，来打破玩家的意图）

//每个方向上的正方向

for (k = 1; k <= 5; ++k)

if (x + i \* k >= 0 && x + i \* k <= 14 && y + j \* k >= 0 && y + j \* k <= 14 && board[x + i \* k][y + j \* k] == 1) ++number1;

else if (x + i \* k >= 0 && x + i \* k <= 14 && y + j \* k >= 0 && y + j \* k <= 14 && board[x + i \* k][y + j \* k] == 0) { ++empty; break; }

else break;

//每个方向上的反方向

for (k = -1; k >= -5; --k)

if (x + i \* k >= 0 && x + i \* k <= 14 && y + j \* k >= 0 && y + j \* k <= 14 && board[x + i \* k][y + j \* k] == 1) ++number1;

else if (x + i \* k >= 0 && x + i \* k <= 14 && y + j \* k >= 0 && y + j \* k <= 14 && board[x + i \* k][y + j \* k] == 0) { ++empty; break; }

else break;

if (number1 == 1) score[x][y] += 1; //人若下该空子，就两子了，那么不管死活都是+1

else if (number1 == 2)

{

if (empty == 1) score[x][y] += 5; //三子，死，+5

else if (empty == 2) score[x][y] += 10;//三子，活，+10

}

else if (number1 == 3)

{

if (empty == 1) score[x][y] += 20;//形成死四，+20

else if (empty == 2) score[x][y] += 100;//形成活四，+100

}

else if (number1 == 4) score[x][y] += 1000;//五子，+1000

//对电脑落子评分（我自己下了这个空子，会怎样）

empty = 0;

for (k = 1; k <= 5; ++k)

if (x + i \* k >= 0 && x + i \* k <= 14 && y + j \* k >= 0 && y + j \* k <= 14 && board[x + i \* k][y + j \* k] == 2) ++number2;

else if (x + i \* k >= 0 && x + i \* k <= 14 && y + j \* k >= 0 && y + j \* k <= 14 && board[x + i \* k][y + j \* k] == 0) { ++empty; break; }

else break;

for (k = -1; k >= -5; --k)

if (x + i \* k >= 0 && x + i \* k <= 14 && y + j \* k >= 0 && y + j \* k <= 14 && board[x + i \* k][y + j \* k] == 2) ++number2;

else if (x + i \* k >= 0 && x + i \* k <= 14 && y + j \* k >= 0 && y + j \* k <= 14 && board[x + i \* k][y + j \* k] == 0) { ++empty; break; }

else break;//碰到边边

if (number2 == 0) score[x][y] += 1; //电脑若下了该空子，就一子了，那么不管死活都是+1

else if (number2 == 1) score[x][y] += 2; //两子，+2

else if (number2 == 2)

{

if (empty == 1) score[x][y] += 8; //三子，死，+8

else if (empty == 2) score[x][y] += 30;//三子，活，+30

}

else if (number2 == 3)

{

if (empty == 1) score[x][y] += 50;//四子，死，+50

else if (empty == 2) score[x][y] += 200;//四子，活，+200

}

else if (number2 == 4) score[x][y] += 10000; //五子，+10000

}

}

//电脑获取最优解位置

void Chess\_ai\_naive(int& x, int& y)

{//搜的是当前的第一个最优解

evaluate\_naive();

int maxpos = 0;

for (int i = 0; i <=14; ++i)

for (int j = 0; j <=14; ++j)

if (score[i][j] > maxpos)

{

maxpos = score[i][j];

x = i;

y = j;//找出分数最高的点，记录并返回

}

}

//人机下棋

void manvsaiplayChess() {

MOUSEMSG m;//保存鼠标消息

int x, y;

int a = 0, b = 0;

while (1) {

if (flag % 2 == 0) {

m = GetMouseMsg();

outtextxy(300, 600, "玩家下");

for (int i = 0; i <=14; i++)

{

for (int j = 0; j <=14; j++)

{

if (abs(m.x - (175 + (i \* 25))) < 12 && abs(m.y - (175 + (j \* 25))) < 12)

{

x = 175 + i \* 25;

y = 175 + j \* 25;

a = i;

b = j;

}

}

}

//人下棋

if (m.uMsg == WM\_LBUTTONDOWN&& (m.x > 160 && m.x < 537 && m.y >162 && m.y < 537))

{

if (board[a][b] != 0)//不能重复下棋

{

MessageBox(NULL, "这里有棋子了请重新选择", "提示", MB\_OK);//提示框

continue;

}

setfillcolor(BLACK);

solidcircle(x, y, 10);//获得鼠标x，y为画一个圆的圆心，半径10

board[a][b] = 1;

flag++;//每次下一个棋，flag就加1

rjhuiqi[0][0] = a;

rjhuiqi[0][1] = b;

if (judge(a, b)) {

MessageBox(NULL, "黑棋获胜", "游戏结束", MB\_OK);

memset(board, 0, sizeof(board));

memset(score, 0, sizeof(score));

flag = 0;

initGame();

}

if (!judge(a, b) && flag == 225) {

MessageBox(NULL, "平局", "游戏结束", MB\_OK);

memset(board, 0, sizeof(board));

memset(score, 0, sizeof(score));

flag = 0;

initGame();

}

}

//人悔棋

if (m.uMsg == WM\_RBUTTONDOWN && (m.x > 160 && m.x < 537 && m.y >162 && m.y < 537)) {

if (board[a][b] != 0&&(rjhuiqi[0][0] == a&&rjhuiqi[0][1] == b)) {

board[a][b] = 0;

board[rjhuiqi[1][0]][rjhuiqi[1][1]] = 0;//电脑下的上一步也退回

ChessBorad();

for (int i = 0; i <=14; i++) {

for (int j = 0; j <=14; j++) {

if (board[i][j] == 1) {

setfillcolor(BLACK);

solidcircle(175 + (i \* 25), 175 + (j \* 25), 10);

}

if (board[i][j] == 2) {

setfillcolor(WHITE);

solidcircle(175 + (i \* 25), 175 + (j \* 25), 10);

}

}

}

flag-2;//人和电脑都退

}

else {

MessageBox(NULL, "不可悔棋", "提示", MB\_OK);//提示框

continue;

}

}

//人机返回大厅

if ((m.uMsg == WM\_LBUTTONDOWN) && m.x >= 450 && m.x <= 625 && m.y >= 50 && m.y <= 100)

{ //全部重置

memset(board, 0, sizeof(board));

memset(score, 0, sizeof(score));

flag = 0;

initGame();

}

}

if (flag % 2 == 1) {

outtextxy(300, 600, "电脑下");

Chess\_ai\_naive(a, b);//评估人下的每一个棋子

setfillcolor(WHITE);

solidcircle(175+ a \* 25, 175 + b \* 25, 10);

board[a][b] = 2;

flag++;

rjhuiqi[1][0] = a;//人机悔棋记录坐标，rjhuiqi[1][0]记录x坐标，rjhuiqi[1][1] = b记录y坐标

rjhuiqi[1][1] = b;

if (judge(a, b)){

MessageBox(NULL, "白棋获胜", "游戏结束", MB\_OK);

memset(board, 0, sizeof(board));

memset(score, 0, sizeof(score));

flag = 0;

initGame();

}

if (!judge(a, b) && flag == 225) {

MessageBox(NULL, "平局", "游戏结束", MB\_OK);

memset(board, 0, sizeof(board));

memset(score, 0, sizeof(score));

flag = 0;

initGame();

}

}

}

}

int main()

{

initgraph(700, 700);//初始化图形系统

initGame();

while (1);//防止闪退

return 0;

}