高级语言程序设计大作业实验报告

1. 作业题目

基于Visual Studio和easyx图形库实现了“五子棋”小游戏

1. 开发环境

Visual Studio 2022

1. 设计目标及功能概述

项目的核心目标是实现一个支持双人对弈的图形化五子棋游戏，具备以下主要功能：

1. 游戏主菜单：包括“开始游戏”“退出游戏”“音乐开关”等按钮；
2. 图形界面棋盘绘制：标准 15×15 棋盘，含边框、坐标和星位；
3. 鼠标落子交互：黑白双方交替下棋，避免重复落子；
4. 胜负判断：支持四个方向（水平、垂直、两条对角线）的五子连珠判断；
5. 胜利提示
6. 程序结构与模块化设计

为提高程序的可维护性与可扩展性，采用模块化设计思路，将各功能封装成类或函数模块。主要模块如下：

1. GameUI：负责启动界面、主菜单和按钮绘制；
2. ChessBoard：负责棋盘绘制与棋子状态记录；
3. GameLogic：负责胜负判断与轮换逻辑；
4. MusicController：封装音乐控制逻辑；
5. main 函数作为程序入口，串联各模块执行流程。
6. 核心功能实现与关键代码解析
7. 棋盘绘制与初始化

游戏采用 600×600 像素的棋盘，分为 15 条线段，通过循环绘制横纵线，并绘制五个星位点，增强棋盘视觉效果：

void drawBoard() {

setlinecolor(BLACK);

for (int i = 0; i < 15; ++i) {

line(OFFSET + i \* GRID\_SIZE, OFFSET,

OFFSET + i \* GRID\_SIZE, OFFSET + (15 - 1) \* GRID\_SIZE);

line(OFFSET, OFFSET + i \* GRID\_SIZE,

OFFSET + (15 - 1) \* GRID\_SIZE, OFFSET + i \* GRID\_SIZE);

}

int starPoints[5][2] = {

{3, 3}, {11, 3}, {7, 7}, {3, 11}, {11, 11}

};

for (auto &pt : starPoints) {

fillcircle(OFFSET + pt[0] \* GRID\_SIZE, OFFSET + pt[1] \* GRID\_SIZE, 4);

}

}

1. 落子与坐标转换

根据鼠标点击位置，将像素坐标转换为棋盘索引坐标，避免用户误点非网格中心：

bool getMousePosition(int mouseX, int mouseY, int &row, int &col) {

col = (mouseX - OFFSET + GRID\_SIZE / 2) / GRID\_SIZE;

row = (mouseY - OFFSET + GRID\_SIZE / 2) / GRID\_SIZE;

if (row >= 0 && row < 15 && col >= 0 && col < 15)

return true;

return false;

}

玩家交替落子，黑白棋通过变量 currentPlayer 控制，并更新棋盘状态二维数组：

int board[15][15] = {0}; // 0 表示空，1 表示黑子，2 表示白子

int currentPlayer = 1;

void placeChess(int row, int col) {

if (board[row][col] == 0) {

board[row][col] = currentPlayer;

drawChess(row, col, currentPlayer);

currentPlayer = 3 - currentPlayer; // 切换玩家

}

}

1. 胜负判断算法

每次落子后，从当前位置向四个方向（横、竖、主对角、副对角）扩展判断是否存在五子连珠。以下为判断函数关键逻辑：

bool checkWin(int row, int col) {

int directions[4][2] = {{0,1}, {1,0}, {1,1}, {1,-1}};

int player = board[row][col];

for (auto& d : directions) {

int count = 1;

for (int i = 1; i < 5; ++i) {

int r = row + i \* d[0], c = col + i \* d[1];

if (r >= 0 && r < 15 && c >= 0 && c < 15 && board[r][c] == player)

count++;

else break;

}

for (int i = 1; i < 5; ++i) {

int r = row - i \* d[0], c = col - i \* d[1];

if (r >= 0 && r < 15 && c >= 0 && c < 15 && board[r][c] == player)

count++;

else break;

}

if (count >= 5) return true;

}

return false;

}

判断成功后弹出胜利提示框并播放胜利音效：

if (checkWin(row, col)) {

MessageBox(NULL, currentPlayer == 2 ? L"黑棋胜！" : L"白棋胜！", L"游戏结束", MB\_OK);

PlaySound(L"win.wav", NULL, SND\_FILENAME | SND\_ASYNC);

}

1. 音乐播放控制模块

使用 mciSendString 实现音乐播放、暂停与关闭控制，配合用户在主菜单点击操作：

void MusicController::play() {

mciSendString(L"open bgm.mp3 alias BGM", NULL, 0, NULL);

mciSendString(L"play BGM repeat", NULL, 0, NULL);

}

void MusicController::pause() {

mciSendString(L"pause BGM", NULL, 0, NULL);

}

void MusicController::close() {

mciSendString(L"close BGM", NULL, 0, NULL);

}

1. 悔棋功能实现

将每步落子记录入栈，用户点击“悔棋”按钮时弹出上一步操作，并恢复棋盘状态：

stack<pair<int, int>> moveHistory;

void undoMove() {

if (!moveHistory.empty()) {

auto [r, c] = moveHistory.top();

moveHistory.pop();

board[r][c] = 0;

currentPlayer = 3 - currentPlayer;

redrawBoard();

}

}

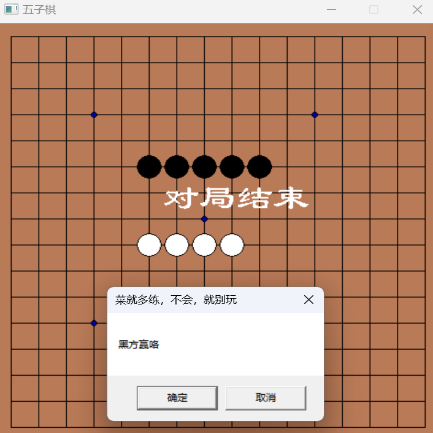
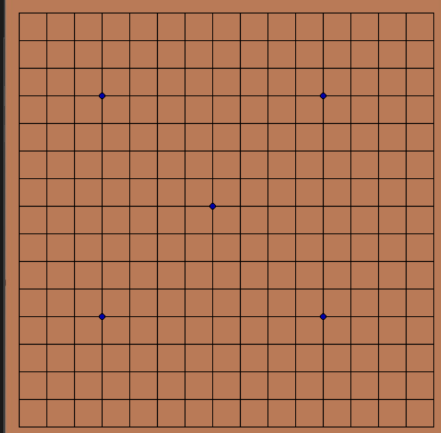
1. 实现效果

游戏运行顺利，实现了所有的预期效果。以下时一些效果展示：

左图为游戏加载页面，右图为游戏主菜单界面



左下图为棋盘画面，右下图为游戏结束后画面



1. 总结和展望

通过本次五子棋项目开发，我不仅加深了对 C++ 类与对象、事件处理、图形绘制的理解，也掌握了音效控制与状态管理的实用技巧。整个项目锻炼了我对模块划分、逻辑组织、用户交互等方面的综合能力。

后续可优化方向包括：引入 AI 算法支持人机对弈；采用更精美的美术资源提升视觉体验；加入网络对战功能扩展游戏范围；封装通用引擎便于后续游戏复用。