**高级语言程序设计**

**课设报告**

**学 号\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**姓 名\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**指导教师\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**提交日期\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**成绩评价表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **报告内容** | **报告结构** | **报告图表** | **报告与程序一致性** | **最终成绩** |
| **□丰富正确**  **□基本正确**  **□有一些问题**  **□问题很大** | **□完全符合要求**  **□基本符合要求**  **□有比较多的缺陷**  **□完全不符合要求** | **□符合规范**  **□基本符合规范**  **□有一些错误**  **□完全不正确** | **□完全一致**  **□基本一致**  **□基本不一致** |  |
| **程序功能实现** | **程序执行情况** | **问题回答情况** | **总体评价** | |
| **□完成基本功能和 扩展功能**  **□完成基本功能几乎无扩展功能**  **□基本完成基本功能**  **□未完成基本功能** | **□顺畅**  **□有问题，经过老师指出之后改正**  **□有问题，无法改正** | **□立即正确回答**  **□经思考后正确回答**  **□回答有部分错误**  **□回答完全错误**  **□不能回答问题** |  | |

**教师签字:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1 需求分析

游戏名称：连连看。这是一款区别于普通连连看的小游戏。玩家以某个速度顺时针连续旋转，在旋转过程中根据键盘或鼠标命令射出直线连接 界面上的小球，每连接一个小球，玩家则占领此地，连接下一个小球，直至连接所有小球成闭环。

1.1 功能需求

1.1.1基本功能：

①完成游戏登入界面的设计

②在登入界面按下空格，将弹出对话框用于保存当前玩家的用户名

③游戏开始后：

1）载入背景画面，主球和其余小球随机分布在相应位置上

2）主球以某个角速度顺时针旋转

3）玩家点击鼠标控制球发出子弹，子弹击中小球后将此球连接，表示该球被占领。则上一个小球停止旋转、炮针消失和失去发射子弹的功能；而当前被占领的球开始旋转、生成炮针和具有发射子弹的功能。以此类推，直至所有小球闭合成环或者形成“主球—其他球—主球”的连接路线。

4）游戏时间固定（本游戏限时60秒），剩余游戏时间显示在屏幕右上角，游戏时间实时更新

5）游戏时间到或玩家成功连接小球，游戏结束

6）游戏结束，记录总得分，并显示在界面上

7）保存用户的成绩：单击“保存成绩”可记录当前玩家的用户名和游戏得分，并将其存入文件中（同一个用户名可有多个成绩）

1.1.2拓展功能：

①增加暂停按钮：点击暂停按钮后，游戏和游戏时间暂停，可选择继续游戏或者回到主菜单，

②显示排行榜：主菜单中单击“显示排行榜”按钮，读取文件中的信息并进行排序，排行榜输出前10名的玩家信息显示在新界面中。

③增加音乐：在登入游戏时和进行游戏时增加两段背景音乐

④游戏分级：在主菜单界面增加“游戏难度”按钮，点击后可进入新界面选择游戏的等级，并在此界面左上角增加“返回”键，供玩家直接返回主菜单。

⑤主菜单增加“退出游戏”：可供玩家直接中断程序，增加了界面的交互性

⑥本游戏小球增加“魔性”运动功能，提高了游戏的难度和趣味性

⑦在每次游戏结束后自动弹出一个显示玩家得分和是否成功的界面，界面上增加有“保存成绩”按钮和“返回主菜单”按钮，提升了交互性。

⑧增加游戏帮助：帮助玩家更好地理解游戏

⑨优化界面成3 个区域：游戏区、控制区、信息显示区。增加游戏的控制和显示信息。

1.2 数据需求

输入数据：用户的ID

中间结果：排行榜中的用户名ID、排名和得分，游戏中不断更新的时间和当前得分

输出结果：用户的游戏得分

1.3 界面需求

①游戏登入界面：



图1 游戏登入界面

②游戏主菜单界面：



图2 游戏运行界面

③游戏运行界面：

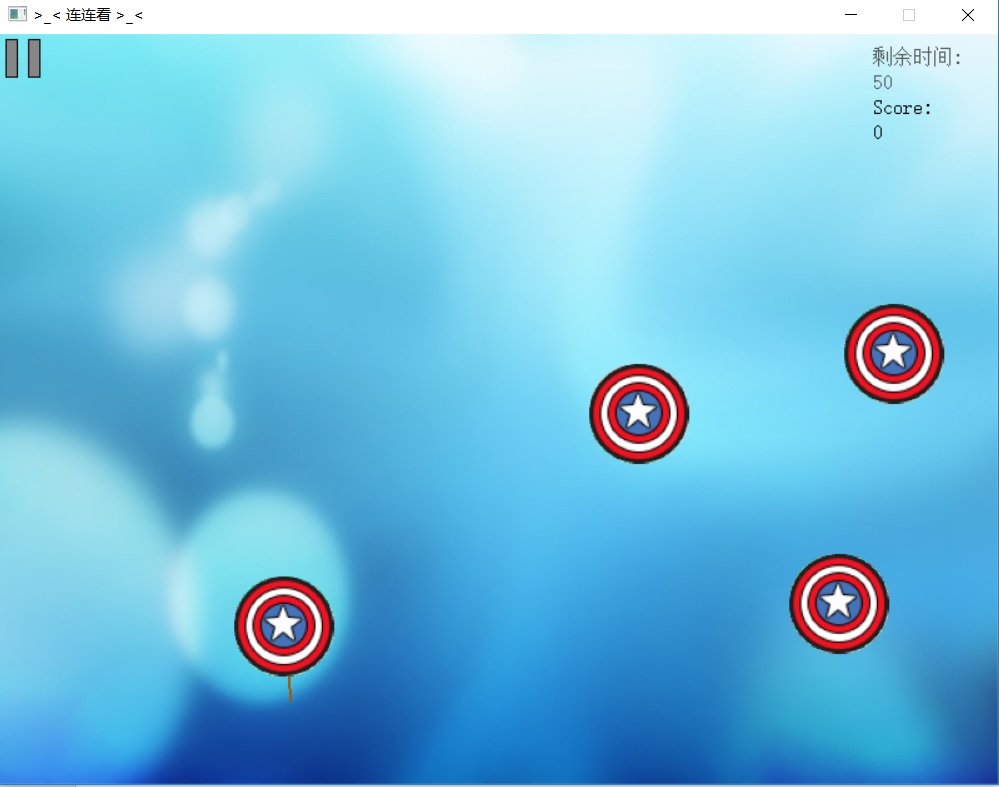


图3 游戏运行界面

④游戏成功与失败界面：

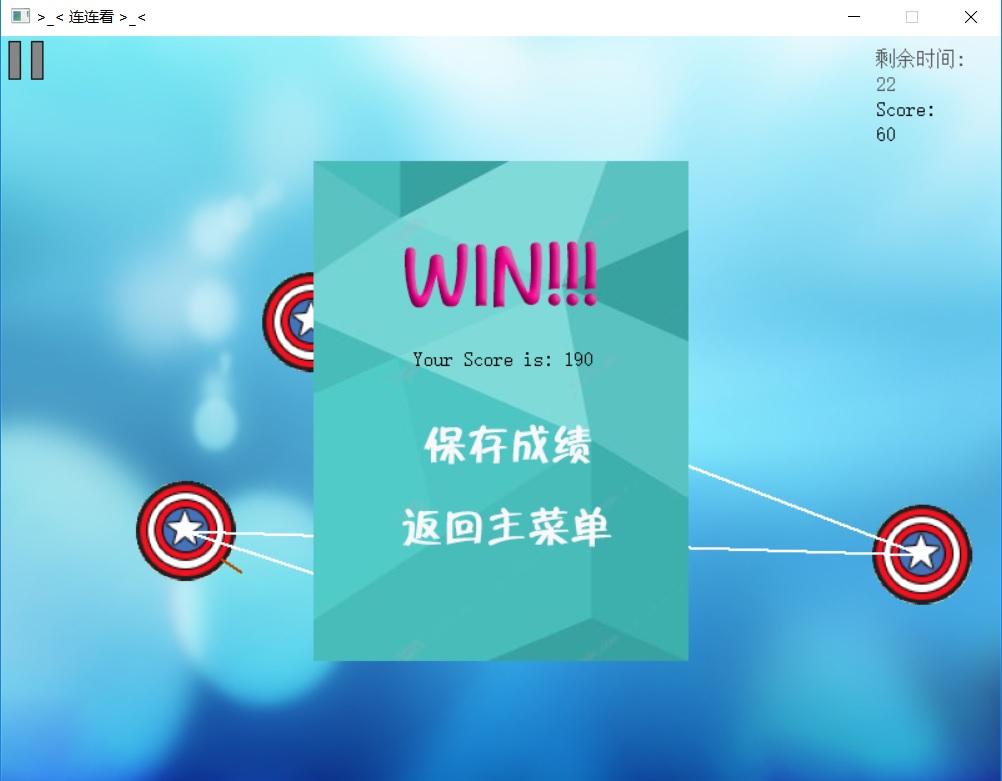


图4 游戏成功界面

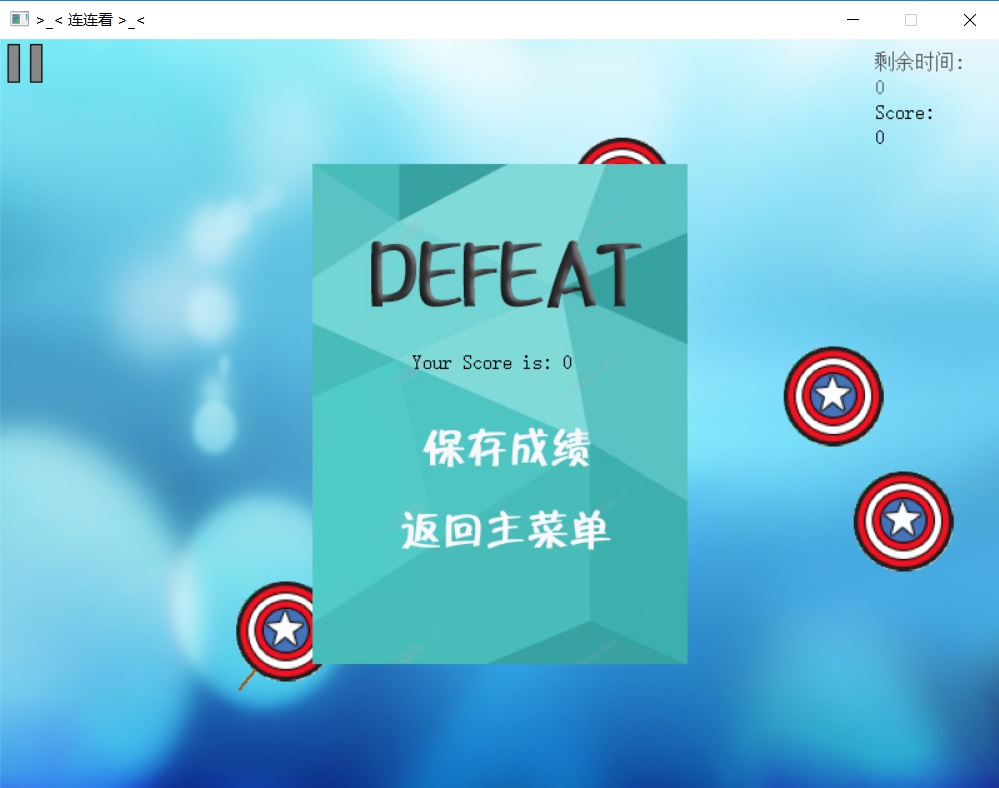


图5 游戏失败界面

⑤游戏暂停界面：

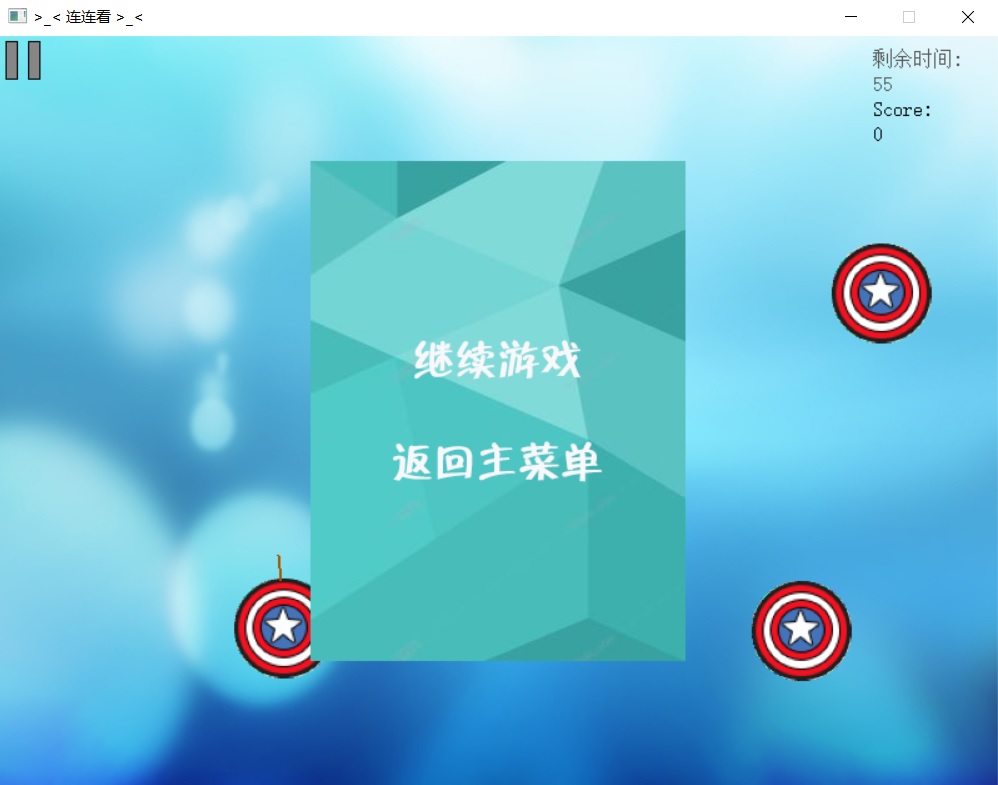


图6 游戏暂停界面

1.4 开发与运行环境需求

①开发环境：

Microsoft Visual Studio 2017

②运行环境：

Windows 10 Family Chinese Version 64-bit (10.0, version 17134)

1.5 其他方面需求

①为了获取对话框中的用户名，使用TcharToChar函数将Tchar类型的字符串转成char类型的字符串，从而将用户信息保存到文件中。

②运行环境在创建项目时选择了SDL安全检查，具有较好的安全性，如在outtextxy函数的中的字符串使用了\_T(“”)形式，fprintf改成更为安全的fprintf\_s形式，用\_getch()代替了不具有移植性的getch()等。

③在构建文件来保存成绩、销毁玩家成绩等功能中使用了链表。

2 概要设计

2.1 主要数据结构

2.1.1结构体：

struct BALL {

int x; //小球坐标

int y;

int needle\_x1, needle\_y1; //表示针距离球远的一端的坐标

int needle\_x2, needle\_y2; //表示针距离球远的一端的坐标

int radius; //半径

bool isbullet; //判断谁当下准备开火,有开火权，但被ruin后仍为true，表示有过炮针的痕迹，便于进行连线,另外一方面可以充当副球们是否遍历的情况

int bullet\_x1, bullet\_y1, bullet\_x2, bullet\_y2; //表示炮弹的坐标

int bullet\_angle; //炮弹的角度

bool ruin;

int vx, vy; //小球速度

};

struct file {

int score; //得分

char name[10]; //保存名字

struct file \*next; //下一个文件的地址

};

struct getrank {

char name[10]; //用户名

int score; //分数

char info[20]; //用于输出

TCHAR userinfo[20];

}:

2.1.2数组：

struct getrank rank[100]; //记录排名信息

struct BALL ball[10]; //记录小球的信息

int toline[20]; //存入连线时的小球序号

TCHAR username[10]; //获取对话框里的用户名

char name[10]; //保存转化后的用户名

2.1.3文件：

①图片：

储存格式：.jpg

数据：

img\_bk //背景图片 ： IMAGE

img\_pause1 //暂停图标遮罩图 ： IMAGE

img\_pause2 //暂停图标 ： IMAGE

img\_ball1 //球的遮罩图 ： IMAGE

img\_ball2 //旋转球 ： IMAGE

img\_needle1 //针的遮罩图 ： IMAGE

img\_needle2 //针 ： IMAGE

img\_enter //登入封面 ： IMAGE

img\_pausemenu //暂停界面 ： IMAGE

img\_mainmenu //主菜单 ： IMAGE

img\_mainmenu1 //主菜单的背景 ： IMAGE

img\_return1, img\_return2 //返回键 ： IMAGE

img\_win, img\_defeat //成功与失败 ： IMAGE

img\_difficulty //难度 ： IMAGE

img\_instruction //游戏说明 ： IMAGE

②音乐：

储存格式：.mp3

数据：

mainmenumusic.mp3

gamemusic.mp3

2.1.4常量：

①时间：

保存倒计时 int

开始时间 clock\_t

当前时间 clock\_t

暂停时的时间 clock\_t

暂停结束时的时间 clock\_t

时差 int

②主球：

主球的x, y坐标 int

主球的半径 int

转动的角度 double

针的长度（主副球的） int

炮针两端的x, y坐标 int

子弹的长度（主副球的） int

子弹两端的坐标 int

主球状态 bool

③其他：

连线下标 int

得分 int

难度信息 int  
小球个数 int

场景编号 int

2.2 程序总体结构

2.2.1 模块调用图

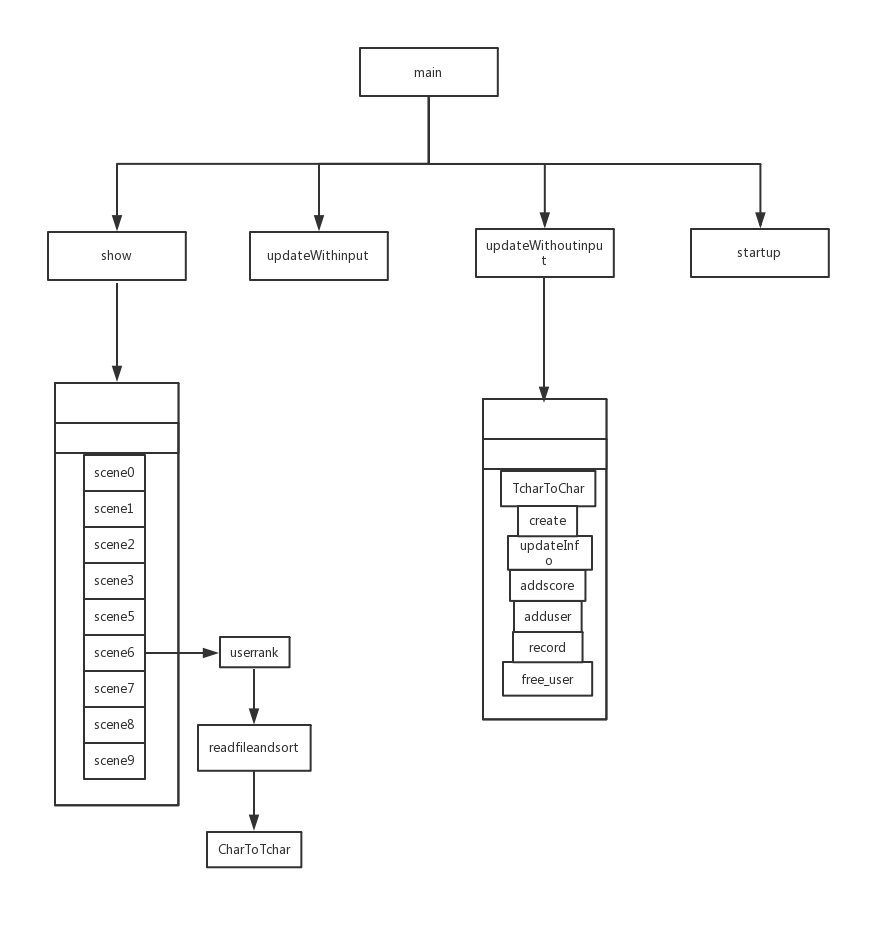


图7 模块调用图

2.2.2 主程序流程图

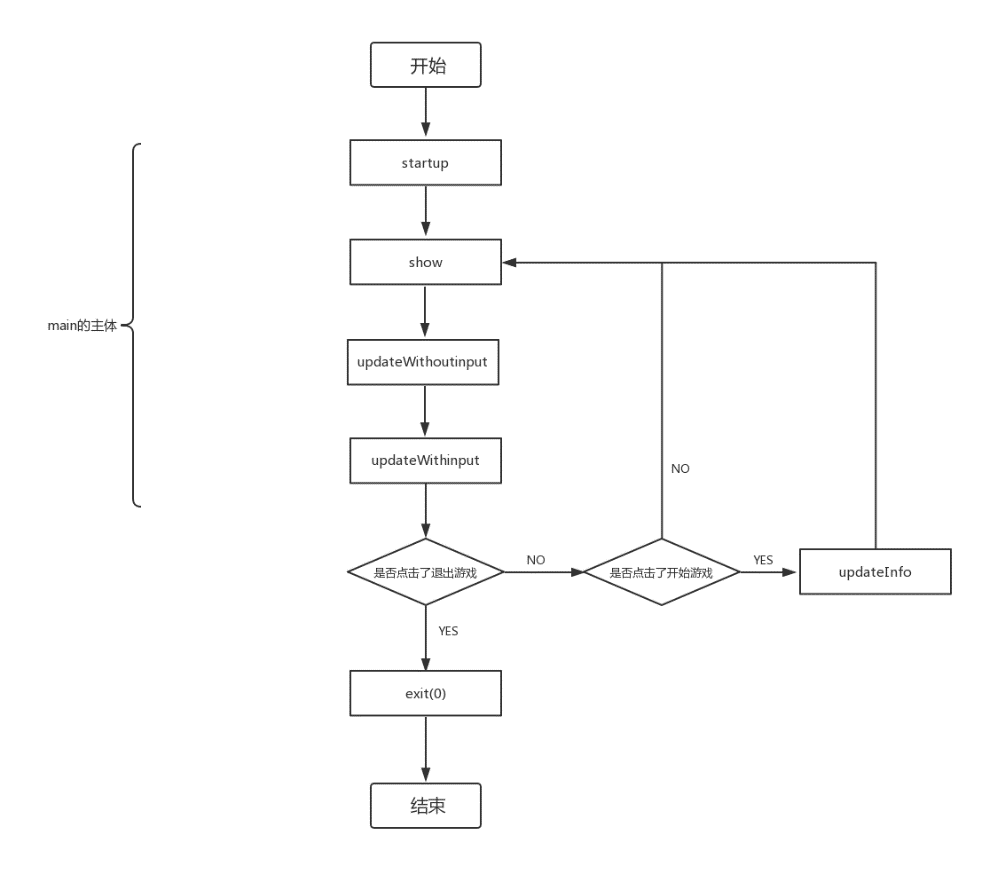


图8 主程序流程图

2.3 子模块设计

2.3.1显示界面、图片和图标的函数模块：

void show(int \*oclock, IMAGE \*img\_bk, IMAGE \*img\_pause1, IMAGE \*img\_pause2, IMAGE \*img\_ball1, IMAGE \*img\_ball2, IMAGE \*img\_needle1, IMAGE \*img\_needle2, IMAGE \*img\_enter, IMAGE \*img\_pausemenu, IMAGE \*img\_mainmenu, IMAGE \*img\_mainmenu1, IMAGE \*img\_return1, IMAGE \*img\_return2, IMAGE \*img\_win, IMAGE \*img\_defeat, IMAGE \*img\_difficulty, IMAGE \*img\_instruction ,int \*ball\_x, int \*ball\_y, int \*radius, int \*needle\_x1, int \*needle\_y1, int \*needle\_x2, int \*needle\_y2, int \*bullet\_x1, int \*bullet\_y1, int \*bullet\_x2, int \*bullet\_y2,int \*score,int \*n,int toline[],int \*scene)

①显示登入界面的函数模块：

void scene0(IMAGE \*img\_enter);

②显示主菜单的函数模块：

void scene1(IMAGE \*img\_mainmenu);

③显示游戏运行界面的函数模块：

void scene2(int \*oclock, IMAGE \*img\_bk, IMAGE \*img\_pause1, IMAGE \*img\_pause2, IMAGE \*img\_ball1, IMAGE \*img\_ball2, int \*ball\_x, int \*ball\_y, int \*radius, int \*needle\_x1, int \*needle\_y1, int \*needle\_x2, int \*needle\_y2, int \*bullet\_x1, int \*bullet\_y1, int \*bullet\_x2, int \*bullet\_y2, int \*score, int \*n, int toline[]);

④显示暂停界面的函数模块：

void scene3(IMAGE \*img\_pausemenu);

⑤显示选择游戏难度界面的函数模块：

void scene5(IMAGE \*img\_return1, IMAGE \*img\_return2, IMAGE \*img\_difficulty);

⑥显示游戏排行榜界面的函数模块：

void scene6(IMAGE \*img\_mainmenu1, IMAGE \*img\_return1, IMAGE \*img\_return2);

⑦显示游戏帮助界面的函数模块：

void scene7(IMAGE \*img\_instruction, IMAGE \*img\_return1, IMAGE \*img\_return2);

⑧显示游戏成功界面的函数模块：

void scene8(IMAGE \*img\_win, int \*score);

⑨显示游戏失败界面的函数模块：

void scene9(IMAGE \*img\_defeat, int \*score);

2.3.2无操作（即无键盘输入、鼠标消息）的更新变量的函数模块：

void updateWithinput(clock\_t \*time1, clock\_t \*time2, clock\_t \*timepause, clock\_t \*timeend, int \*jetlag, int \*oclock, double \*bullet\_angle, double \*angle, int \*bullet\_x1, int \*bullet\_y1, int \*bullet\_x2, int \*bullet\_y2, int \*needle\_x1, int \*needle\_y1, int \*bullet\_length, int \*ball\_x, int \*ball\_y, int \*needle\_x2, int \*needle\_y2, int \*radius, int \*needle\_length,int \*n,int \*state,int \*degree, int \*score,int \*lineNum,int \*scene,bool \*ruin, bool \*reborn, int toline[]);

2.3.3有操作（即有键盘输入、鼠标消息）的更新变量的函数模块：

void updateWithinput(clock\_t \*time1, clock\_t \*time2, clock\_t \*timepause, clock\_t \*timeend, int \*jetlag, int \*oclock, double \*bullet\_angle, double \*angle, int \*bullet\_x1, int \*bullet\_y1, int \*bullet\_x2, int \*bullet\_y2, int \*needle\_x1, int \*needle\_y1, int \*bullet\_length, int \*ball\_x, int \*ball\_y, int \*needle\_x2, int \*needle\_y2, int \*radius, int \*needle\_length, int \*n, int \*state, int \*degree, int \*score, int \*lineNum, int \*scene, bool \*ruin, bool \*reborn, int toline[]);

①创建保存玩家信息的链表的头结点函数模块：

F \*create()；

②增加保存玩家用户名的节点的函数模块：

void adduser(F \* h,char name[]);

③销毁节点的函数模块：

void free\_user(F \* h);

④将链表中储存的玩家信息保存到文件中的函数模块：

void record(F \* h);

⑤读取文件内容并进行排序的函数模块：

void readfileandsort();

⑥将排好序的玩家信息进行显示输出的函数模块

void userrank();

3 详细设计

3.1 总体设计

3.1.1需求概述

1.在登入界面按下空格，将弹出对话框用于保存当前玩家的用户名

2.游戏开始后：载入背景画面，主球和其余小球随机分布在相应位置上，玩家点击鼠标控制球发出子弹，子弹击中小球后将此球连接，此游戏时间固定（本游戏限时60秒），剩余游戏时间显示在屏幕右上角，游戏时间实时更新

3.增加暂停按钮，点击暂停按钮后，游戏和游戏时间暂停，可选择继续游戏或者回到主菜单

4.保存用户的成绩：游戏结束后单击“保存成绩”可记录当前玩家的用户名和游戏得分，并将其存入文件中

5.显示排行榜 主菜单中单击“显示排行榜”按钮，读取文件中的信息并进行排序，排行榜输出前10名的玩家信息显示在新界面中。

6.登入游戏时和进行游戏时带有两段背景音乐

7.游戏分级：在主菜单界面增加“游戏难度”按钮，点击后可进入新界面选择游戏的等级，并在此界面左上角增加“返回”键，供玩家直接返回主菜单。

8.主菜单增加“退出游戏”按钮，可供玩家直接中断程序，增加了界面的交互性

9.本游戏小球增加“魔性”运动功能，提高了游戏的难度和趣味性

3.1.2 系统结构

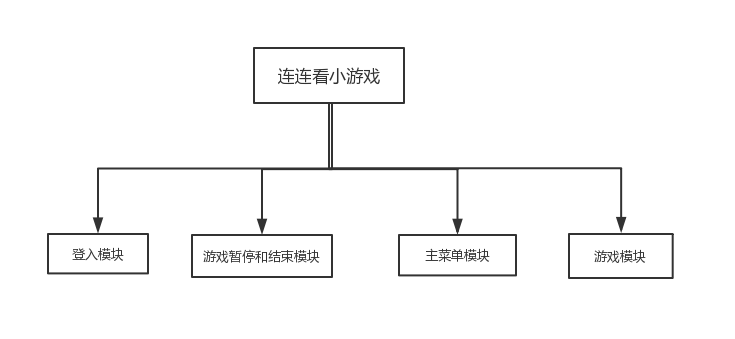


图9 系统结构

3.2程序描述

3.2.1登入模块：

目的与意义：完成用户的创建

功能：根据玩家输入的用户名创建新用户

输入：用户名

输出：文件和排行榜中的的用户成绩

程序逻辑：

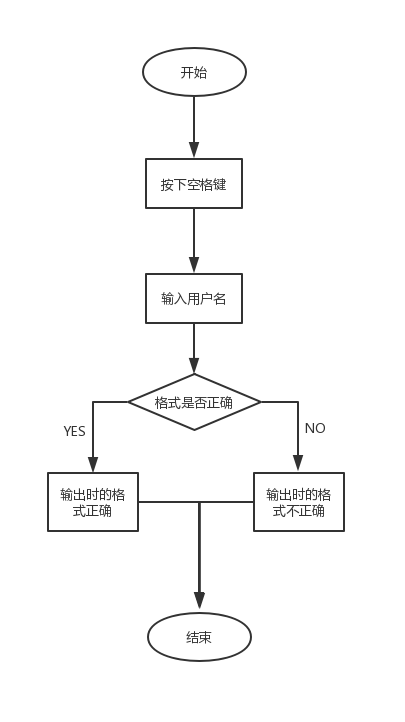


图10 登入模块图

//以下位于updateWithinput函数中

TCHAR username[10];

static char name[10];

static F \* head; //头结点

char input;

if (\_kbhit()) {

input = \_getch();

if (\*scene == 0 && input == 32) {

InputBox(username, 10, \_T("请输入用户名(非中文")); //解决名字问题

TcharToChar(username, name);

head = create();

mciSendString(\_T("close bkmusic"), NULL, 0, NULL);

\*scene = 1;

}

//登入图的显示

void scene0(IMAGE \*img\_enter) { //即登入图

cleardevice(); //把加载的图片全部清除

putimage(0, 0, img\_enter);

FlushBatchDraw(); //关键因素

}

}

3.2.2主菜单模块：

目的与意义：增加了界面的交互性，提高玩家的体验感

功能：玩家可从主菜单上根据自身需求选择不同的操作

输入：无

输出：无

程序逻辑：

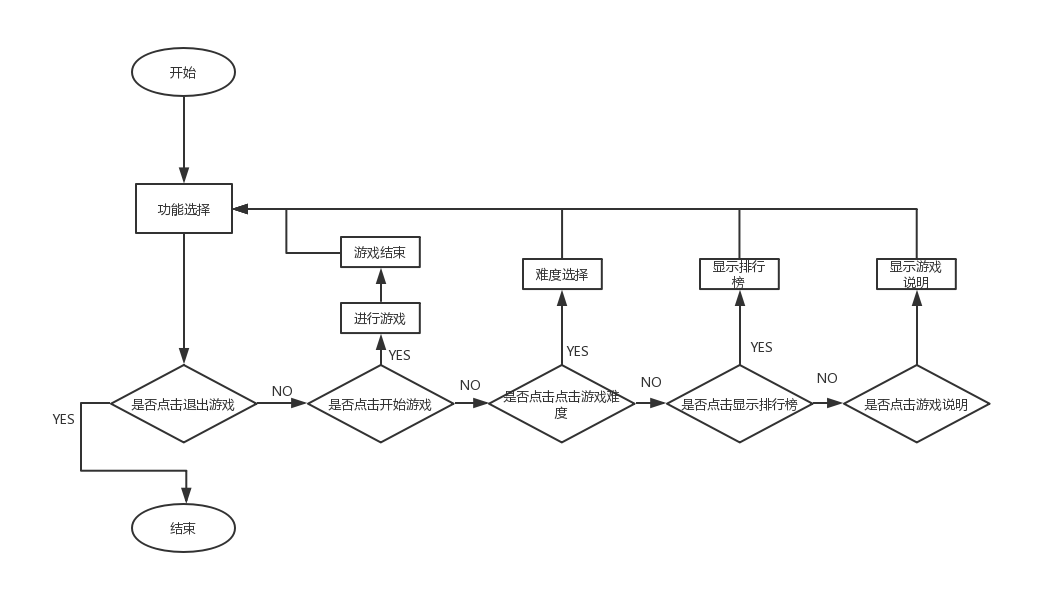


图11 主菜单模块图

//主菜单的显示

void scene1(IMAGE \*img\_mainmenu) {//即startmenu()

cleardevice();

putimage(0, 0, img\_mainmenu);

FlushBatchDraw();

}

//主菜单的鼠标点击选择

while (MouseHit()) { //如果检查到按键弹起的消息，发射子弹

m = GetMouseMsg();

if (m.uMsg == WM\_LBUTTONUP) {

F \*h = head; //将h座位头结点传入之后的函数中，防止head头结点可能遭到修改

if (\*scene == 1) {

if (m.x <= 480 && m.x >= 300 && m.y <= 150 && m.y >= 110) {

\*scene = 2;

updateInfo(time1, time2, timepause, timeend, jetlag, oclock, ball\_x, ball\_y, radius, needle\_x1, needle\_y1, needle\_x2, needle\_y2, bullet\_x1, bullet\_y1, bullet\_x2, bullet\_y2, needle\_length, bullet\_length, bullet\_angle, angle, n, state, score, lineNum, ruin, reborn, toline, degree);

adduser(h, name);

\*time1 = clock(); //计时的起点

\*time2 = clock();

mciSendString(\_T("close bkmusic"), NULL, 0, NULL);

}

else if (m.x <= 480 && m.x >= 300 && m.y <= 240 && m.y >= 200) {

\*scene = 5;

}

else if (m.x <= 460 && m.x >= 320 && m.y <= 320 && m.y >= 280) {

\*scene = 6;

}

else if (m.x <= 480 && m.x >= 300 && m.y <= 410 && m.y >= 365) {

\*scene = 7;

}

else if (m.x <= 480 && m.x >= 300 && m.y <= 500 && m.y >= 460) {

exit(0);

}

break;

}

}

}

3.2.3游戏模块：

目的与意义：玩家进行游戏

功能：玩家进行游戏

输入：无

输出：游戏结果（包括成功与否和游戏得分）

程序逻辑：

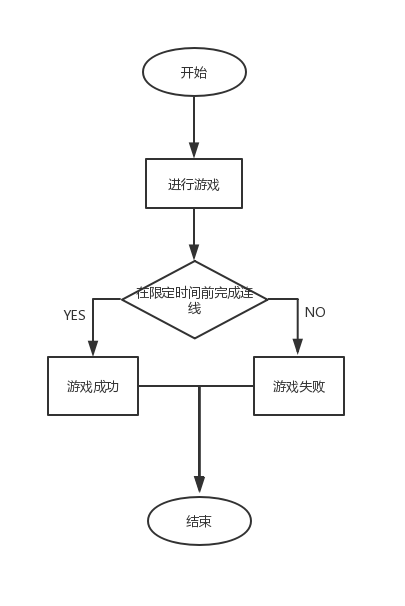


图12 游戏模块图

①小球的连线

setcolor(WHITE);

if (ball[toline[0]].isbullet == true) {

line(\*ball\_x, \*ball\_y, ball[toline[0]].x, ball[toline[0]].y);

}

for (int i = 1; i < \*n; i++) { if (toline[i] >= 0) {

if (ball[toline[i]].isbullet == true) line(ball[toline[i - 1]].x, ball[toline[i - 1]].y, ball[toline[i]].x, ball[toline[i]].y);

}

}

②小球的移动和控制小球不重叠、不出边界

//小球移动,每个球都控制在一个小区域内

for (int i = 0; i < \*n; i++) {

int k = rand() % 4;

switch (k) { //解决小球会碰撞问题

case 0: ball[i].vx = 5; ball[i].vy = 0; break; //处理小球重叠问题

case 1: ball[i].vx = 0; ball[i].vy = 5; break;

case 2: ball[i].vx = -5; ball[i].vy = 0; break;

case 3: ball[i].vx = 0; ball[i].vy = -5; break;

}

}

for (int i = 0; i < \*n; i++) {

int flag = 0;

for (int j = 0; j < \*n; j++) {

if (j == i) continue;

if (!((ball[i].x - 2 \* ball[i].radius + ball[i].vx >= ball[j].x || ball[i].x + 2 \* ball[i].radius + ball[i].vx <= ball[j].x || ball[i].y - 2 \* ball[i].radius + ball[i].vy >= ball[j].y || ball[i].y + 2 \* ball[i].radius + ball[i].vy <= ball[j].y) && (ball[i].x - 2 \* ball[i].radius + ball[i].vx >= \*ball\_x || ball[i].x + 2 \* ball[i].radius + ball[i].vx <= \*ball\_x || ball[i].y - 2 \* ball[i].radius + ball[i].vy >= \*ball\_y || ball[i].y + 2 \* ball[i].radius + ball[i].vy <= \*ball\_y))) {

flag = 1;

//倘若两球相碰撞时让他们反着走

ball[i].x -= ball[i].vx;

ball[i].y -= ball[i].vy;

}

}

if (flag == 0) {

if (ball[i].x >= ball[i].radius + 5 && ball[i].x <= Width - ball[i].radius - 5 && ball[i].y >= ball[i].radius + 5 && ball[i].y <= High - ball[i].radius - 5) {

ball[i].x += ball[i].vx;

ball[i].y += ball[i].vy;

}

else {

//使小球不走到控制区域

if (ball[i].x - ball[i].radius <= 40 && ball[i].y - ball[i].radius <= 40) {

ball[i].x += 5;

ball[i].y += 5;

}

else if (ball[i].x + ball[i].radius >= Width \* 7 / 8 && ball[i].y - ball[i].radius <= 100) {

ball[i].x -= 5;

ball[i].y += 5;

}

else {

if (ball[i].x <= ball[i].radius + 5) ball[i].x += 5;

if (ball[i].x >= Width - ball[i].radius - 5) ball[i].x -= 5;

if (ball[i].y <= ball[i].radius + 5) ball[i].y += 5;

if (ball[i].y >= High - ball[i].radius - 5) ball[i].y -= 5;

}

}

}

}

③时间的判断

if (\*oclock == -1) {

\*scene = 9;

mciSendString(\_T("close bkmusic"), NULL, 0, NULL); // 关闭音乐

/\*score = 0;\*/

}

④判断是否连成线

for (int i = 0; i < \*n; i++) { //返回去连接主球

if (ball[i].isbullet == true && ball[i].ruin == false) {

if ((ball[i].bullet\_x2 - \*ball\_x)\*(ball[i].bullet\_x2 - \*ball\_x) + (ball[i].bullet\_y2 - \*ball\_y)\*(ball[i].bullet\_y2 - \*ball\_y) <= (\*radius) \* (\*radius)) {

//设置只有最后连成环的时候才能成功，否则之前返回去连接属于失败

//如果全部遍历到表示成功，否则则跳出

int flag = 1;

for (int j = 0; j < \*n; j++) {

if (ball[j].isbullet == false) flag = 0; //代表着还没有遍历完成

}

if (flag) {

\*scene = 8;//表示游戏的成功

mciSendString(\_T("close bkmusic"), NULL, 0, NULL);

/\*score = oclock \* 10;\*/

ball[i].needle\_x1 = -1; ball[i].needle\_y1 = -1;//上一个球的针消失

ball[i].needle\_x2 = -1; ball[i].needle\_y2 = -1;

ball[i].ruin = true;

\*score += 20;//表示连到最后一个球加20分

\*score += \*oclock \* 5;//表示游戏结束时的时间奖励分

setcolor(WHITE);

if (\*reborn == true) line(ball[toline[(\*lineNum) - 1]].x, ball[toline[(\*lineNum) - 1]].y, \*ball\_x, \*ball\_y);

}

}

break;

}

}

⑤针位置的更新

if (!(\*ruin)) { //摧毁了变便再无小针

\*needle\_x1 = \*ball\_x + (int)(\*radius \* sin(\*angle));

\*needle\_y1 = \*ball\_y - (int)(\*radius \* cos(\*angle));

\*needle\_x2 = \*ball\_x + (int)((\*needle\_length + \*radius) \* sin(\*angle));

\*needle\_y2 = \*ball\_y - (int)((\*needle\_length + \*radius) \* cos(\*angle));

}

for (int i = 0; i < \*n; i++) {

if (ball[i].isbullet == true && ball[i].ruin == false) {

ball[i].needle\_x1 = ball[i].x + (int)(ball[i].radius \* sin(\*angle));

ball[i].needle\_y1 = ball[i].y - (int)(ball[i].radius \* cos(\*angle));

ball[i].needle\_x2 = ball[i].x + (int)((\*needle\_length + ball[i].radius) \* sin(\*angle));

ball[i].needle\_y2 = ball[i].y - (int)((\*needle\_length + ball[i].radius) \* cos(\*angle));

}

}

⑥子弹位置的更新

if (\*bullet\_x2 != -1 && \*bullet\_y2 != -1) { //子弹的上升

\*bullet\_x1 += (int)(40 \* sin(\*bullet\_angle));

\*bullet\_y1 -= (int)(40 \* cos(\*bullet\_angle));

\*bullet\_x2 += (int)(40 \* sin(\*bullet\_angle));

\*bullet\_y2 -= (int)(40 \* cos(\*bullet\_angle));

}

⑦时间的更新

\*jetlag += \*timeend - \*timepause;

\*timeend = 0; \*timepause = 0; //更新为0

\*time2 = clock();

\*oclock = (int)(60 - ((\*time2 - \*jetlag - \*time1) / 1000));

⑧子弹的消失

for (int i = 0; i < \*n; i++) {

if (ball[i].bullet\_x2 >= Width || ball[i].bullet\_y2 <= 0 || ball[i].bullet\_x2 <= 0 || ball[i].bullet\_y2 >= High) {

ball[i].bullet\_x1 = -1; ball[i].bullet\_y1 = -1; //使子弹不在画面内

ball[i].bullet\_x2 = -1; ball[i].bullet\_y2 = -1;

ball[i].bullet\_angle = 0;

}

}

3.2.4游戏暂停和结束模块：

目的与意义：弹出小界面增加游戏交互性

功能：使游戏暂停和显示游戏结果

输入：无

输出：无

程序逻辑：

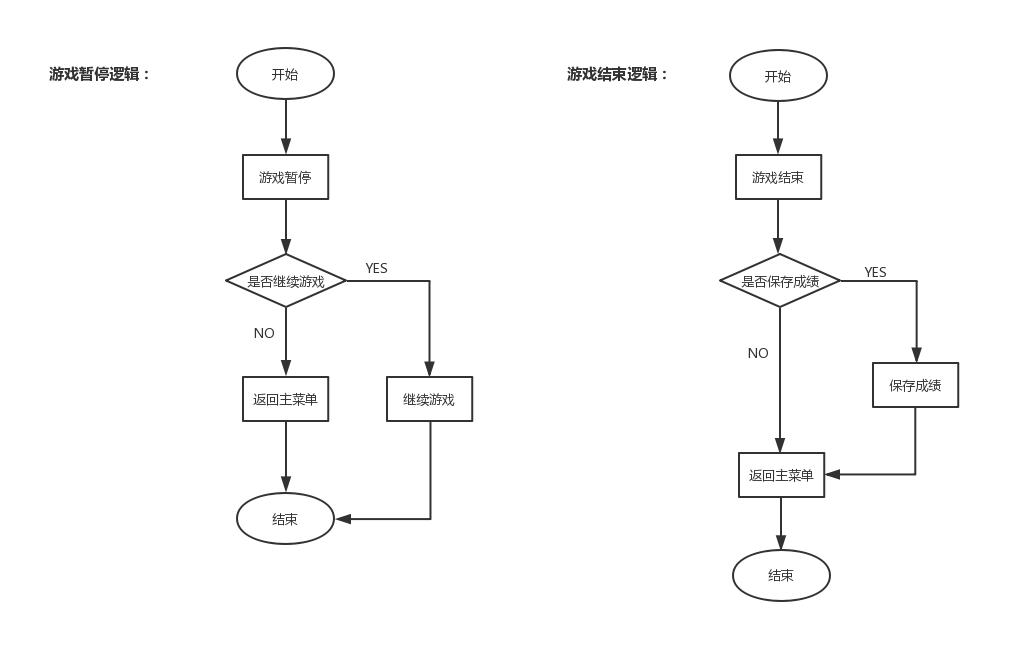


图13 游戏暂停和结束模块图

暂停：

//暂停界面显示

void scene3(IMAGE \*img\_pausemenu) {//即pausemenu()

putimage(Width / 2 - 150, High / 2 - 200, img\_pausemenu);

FlushBatchDraw();

}

//鼠标点击暂停的响应

while (MouseHit()) {

m = GetMouseMsg();

if (m.uMsg == WM\_LBUTTONUP) {

if (\*scene == 3) {

if (m.x <= Width / 2 - 150 + 220 && m.x >= Width / 2 - 150 + 80 && m.y <= High / 2 - 200 + 170 && m.y >= High / 2 - 200 + 140) {

\*scene = 2;

mciSendString(\_T("resume bkmusic"), NULL, 0, NULL); // 恢复播放

\*timeend = clock();

}

else if (m.x <= Width / 2 - 150 + 230 && m.x >= Width / 2 - 150 + 65 && m.y <= High / 2 - 200 + 255 && m.y >= High / 2 - 200 + 230) {

free\_user(h);

\*scene = 1;

}

break;

}

}}

游戏结束：

//成功界面（成功前提见上述的小球成功连成线）

void scene8(IMAGE \*img\_win, int \*score) { //成功界面

putimage(Width / 2 - 150, High / 2 - 200, img\_win);

TCHAR SCORE[5];

swprintf\_s(SCORE, \_T("%d"), \*score);

setcolor(BLACK); //将倒计时文字设为黑色

setbkmode(TRANSPARENT); //将文字设为透明的不带底色

settextstyle(16, 0, \_T("宋体"));

outtextxy(Width / 2 - 150 + 80, High / 2 - 200 + 150, \_T("Your Score is:"));

outtextxy(Width / 2 - 150 + 200, High / 2 - 200 + 150, SCORE);

FlushBatchDraw();

}

//失败界面（失败前提见上述的时间判断即时间截止）

void scene9(IMAGE \*img\_defeat, int \*score) { //失败界面

putimage(Width / 2 - 150, High / 2 - 200, img\_defeat);

char s[5];

TCHAR SCORE[5];

swprintf\_s(SCORE, \_T("%d"), \*score);

sprintf\_s(s, "%d", \*score);

setcolor(BLACK); //将倒计时文字设为黑色

setbkmode(TRANSPARENT); //将文字设为透明的不带底色

settextstyle(16, 0, \_T("宋体"));

outtextxy(Width / 2 - 150 + 80, High / 2 - 200 + 150, \_T("Your Score is:"));

outtextxy(Width / 2 - 150 + 200, High / 2 - 200 + 150, SCORE);

FlushBatchDraw();

}

4 测试

输入测试：

测试目的：判断用户名是否输入正确

测试方法：通过输入用户名并且在后台文件和排行榜进行查看

1.正确用例

正确的测试用例：输入长度不超过10的英文字母或者符号、数字

测试结果：游戏结束保存游戏成绩时，程序自动将成绩与该玩家用户名匹配并保存在文件中，游戏排行榜实时更新玩家的排名和用户名。

2.错误用例

错误的测试用例：在用户名中输入长度大于6的汉字

测试结果：在文件读取过程中会发生异常而迫使程序中断

3.边界用例

边界测试用例：不输入用户名（即用户名长度为0）

测试结果：文件能正常读取，但在保存成绩和显示排名时用户名一直是空。

边界测试用例：输入少量汉字

测试结果：成绩能正常保存，文件能正常读取，但显示排行榜时会有格式差异。

鼠标测试：

测试目的:鼠标是用户对游戏状态控制（暂停 开始 结束）的重要途径，鼠标的可操作性决定着游戏的流畅度和可玩性。

测试方法：操控鼠标对按钮和界面进行点击。

错误实例1：图标点击不反应。

原因：鼠标设定的点击坐标不精确。

改正：通过软件进行精准坐标测量，得到精确的鼠标点击坐标值。

5 用户手册

5.1应用程序功能的详细说明：

1.登入模块：本过程需要玩家输入用户名，完成用户信息的建立。

2.主菜单模块：用户通过主菜单来进行功能的选择，

①开始游戏——玩家进入游戏

②游戏难度——玩家选择游戏的难易程度

③显示排行榜——玩家可查看总的成绩排名

④游戏说明——玩家了解到游戏的具体规则和积分规则

⑤退出游戏——可供玩家点击后直接退出程序

3.游戏模块：玩家在此模块进行游戏

4.游戏暂停和结束模块：

玩家可点击游戏界面的左上角使游戏暂停，在游戏结束时自动弹出游戏失败或成功的界面，玩家可在此界面上选择保存成绩和返回主菜单。

5.2应用程序运行环境的要求：

Microsoft Visual Studio等支持easyX库的编译器。

5.3应用程序的安装与启动方法：

①安装方法：直接将该程序所在的文件夹一并复制到另一台机器即可。

②启动方法：可以通过VS等IDE打开源代码再运行程序，也可直接跑Release文件夹中的exe应用程序。

5.4程序的界面交互方式和操作方法：

①交互方式:该程序的交互方式通过键盘输入和鼠标点击进行的。

②操作方法:在程序运行时按下空格键和输入，其余情况只需要点击字体或者小图标即可完成交互操作。

5.5输入数据类型、格式和内容限制：

在用户登入时，用户名应该为字母、数字和符号组成（字母区分大小写），长度不超过10。

5.6 在应用程序运行中，用户需要使用的交互命令名称、功能和格式的详细说明：

1.点击空格继续

功能：弹出对话框

格式：键盘输入

2.输入用户名

功能：完成用户信息的建立

格式：键盘输入

3.暂停

功能：使游戏暂停

格式：鼠标点击

4.其他等交互

功能：完成按钮上所显示的功能

格式：鼠标点击

6 总结提高

6.1 课程设计总结

在课设开始时，本人将程序逐步地划分成一个小目标或者小任务去实现，并将其记录在清单上，只有便按照清单的内容逐步去实现这些功能。比如，最开始时第一步先用easyX库画出一个圆，然后让圆“长出”一根炮针，再去实现点击鼠标使炮针发射子弹。

随着程序复杂度的增高，清单上的小目标不断增加，然后一一将这些目标实现即可，如此一来便不会导致自己的自己的头脑过于混乱。

本人写课设的顺序大致是：先让小球成功连接成环，把游戏的主体先实现，之后再去进行UI的设计，最后进行一些性能上的优化和增加一些拓展功能。在开发中遇到的一些困难主要是一些细节方面的问题，由于个人的疏漏等因素使得在开发过程中的难度增加。另外，由于在创建项目时使用了SDL检查等属性，导致在进行在进行字符串显示时花了较多时间。

最后也感谢最开始蔡老师的点拨，让我对此次的开发有个大致的了解尤其是对游戏主题的掌握，从而能更清楚自己需要去完成的功能。

评价：在这次课设中基本达到了自己的目标，但游戏的UI设计和一些趣味性的细节方面和游戏的内容丰富性上还有待提高。

6.2 对本课程意见与建议

本人在课设开始前的寒假中看过一本叫做《C课程设计与游戏开发》的书，里面有一些初级简单的小游戏。该书内容不多但实用性强，可以较好地帮助学生理解游戏开发过程中的开发模式，有一个更清晰的游戏思维。本人在游戏中所用的startup()、show()、updateWithout()、updateWith()游戏框架便是从中所学。看完书后，本人在寒假里自主写了300多行代码的小游戏，虽然其中存在一些bug，但实际上非常好的巩固了所学知识。所以十分推荐在进行课程设计之前先看完这本书，之后再进行游戏的开发。

6.3 附件：程序源代码见同一目录下的 C课设连连看.sln