高级语言程序设计-实验报告

贪吃蛇

报告名称：实验报告-贪吃蛇

班级：国 03

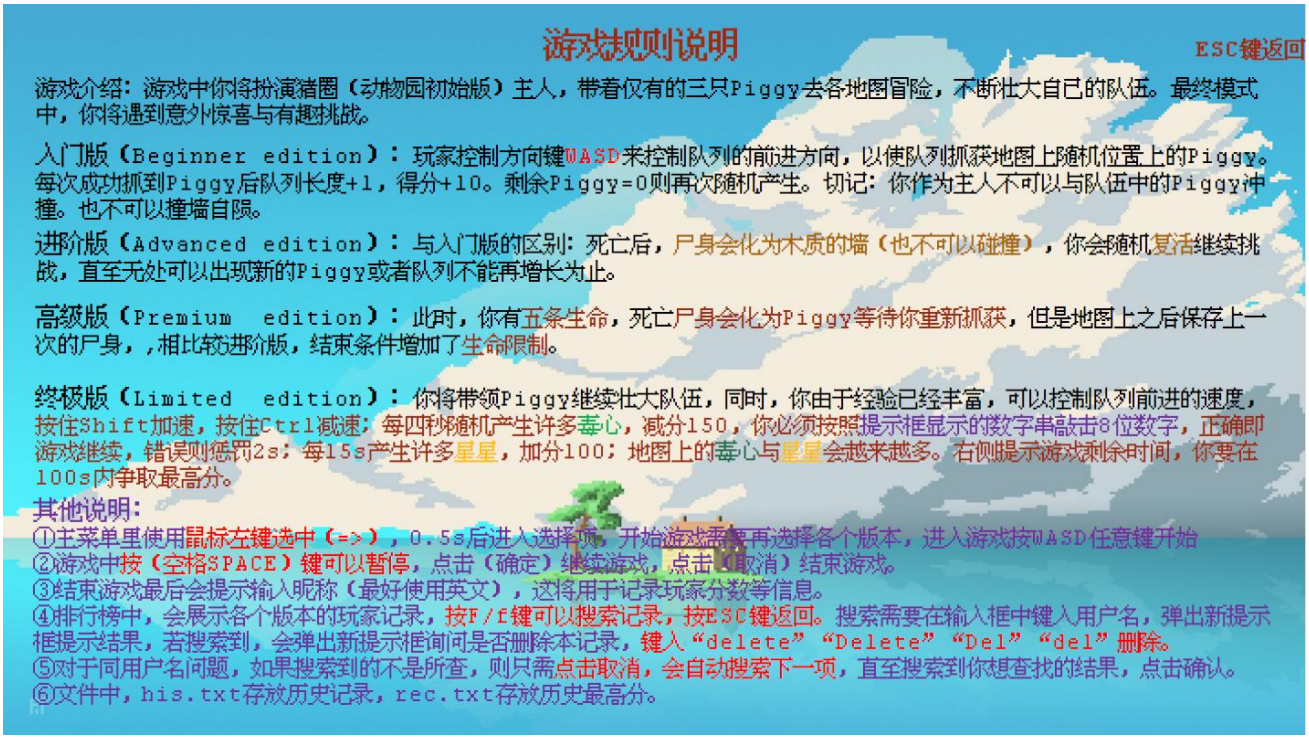
学号：2253156

姓名：闫浩扬

完成日期：2023 年 6 月 23 日

一．游戏介绍

入门版：要求玩家控制方向键来控制小蛇的前进方向，以使蛇吃掉面板上随机位置上的食物（位置随机，数量为 1-5 随机）。每次成功吃掉食物后小蛇体长将增加一点，得分增加。食物吃光则再次随机产生。当小蛇撞到边界或者蛇头与蛇身相撞时，蛇将挂掉，游戏结束。
进阶版：蛇挂掉后，此时蛇尸身改变显示颜色变成边界，再随机产生新的食物和蛇，游戏继续。直到剩余空间不足以生成新的蛇和食物为止。
高级版：蛇挂掉后，此时蛇尸身改变显示颜色变成食物，再随机产生新的食物和蛇，游戏继续。直到撞墙次数>5，或剩余空间不足以生成新的蛇和食物为止。
游戏应实现历史记录功能，将每次游戏的结果记录在文件中，文件编码的格式可以自行决定，但是要求能成功解析并输出到屏幕。应至少保存如下信息，且程序可以灵活增删改查（改查仅限针对用户名）。



二．设计思路

（1）面向对象的思考

由于作业要求使用面向对象，第一版未认真读题，没有使用面向对象的方法。第二版，上网搜索，借鉴他人想法，将蛇，墙，食物，菜单等封装为单独的类，在类内声明成员函数以及数据等，通过将相关功能和数据封装在类中，可以将代码分割为独立的模块。这提高了代码的可维护性和可重用性。不同的元素可以独立开发、测试和维护，使得代码更易于理解和扩展。

（2）各种类的数据结构构造

① 蛇 （duiLie 类）

成员变量：

len：表示贪吃蛇的长度。

life：表示贪吃蛇的生命值。

score：表示贪吃蛇的得分。

lastDir：表示上一个移动方向。

ifDie：表示贪吃蛇是否死亡。

sleepChange1、sleepChange2：用于调整贪吃蛇移动速度的变量。

head：表示贪吃蛇的头部节点。

成员函数：

构造函数 duiLie(Wall& tempWall, Animal& tempAnimal)：接受一个 Wall 对象和一个 Animal 对象作为参数，并初始化一些成员变量。

move(int x, int y)：根据给定的移动增量在指定方向上移动贪吃蛇。

initDuiLie()：初始化贪吃蛇。

randBorn()：在随机位置生成贪吃蛇。

ISDEAD()：检查贪吃蛇是否死亡。

getScore()：获取贪吃蛇的得分。

getLength()：获取贪吃蛇的长度。

getLife()：获取贪吃蛇的生命值。

getHx()：获取贪吃蛇头部的 x 坐标。

getHy()：获取贪吃蛇头部的 y 坐标。

moveDuiLie(char Dir)：根据指定的移动方向移动贪吃蛇。

sleep()：根据贪吃蛇的级别调整休眠时间。

deleteDuiLie()：删除贪吃蛇。

saveData()：保存游戏记录和历史记录。

duiLieToWall()：将贪吃蛇绘制到墙上。

duiLieToAnimal()：将贪吃蛇绘制到 Piggy 上。

duiLieToSpace()：将贪吃蛇绘制到空白区域上。

Die()：贪吃蛇死亡时的处理。

reduceLife(int n)：减少贪吃蛇的生命值。

showDuiLie()：显示贪吃蛇。

② 墙（Wall 类）

表示游戏中的墙体的类。管理游戏地图和墙体，以及显示右侧数据记录，如得分，长度，时间等

成员变量：

map[150][150]：存储游戏地图的二维字符数组。

addWallNum：记录添加的墙体数量。

addWall[100]：存储添加的墙体的坐标。

startTIME：记录游戏开始的时间。

构造函数：

Wall()：初始化 Wall 类的对象。

成员函数：

coutXY(int x, int y)：设置光标在指定的坐标位置输出字符。

initMAP()：初始化游戏地图。设置边界墙体和空白区域。

showWALL()：显示墙体。使用图像库加载墙体图片，并根据图片的尺寸和位置绘制墙体。

change(int x, int y, char ch)：将指定位置的字符改为指定的字符。

getWhat(int x, int y)：获取指定位置的字符。

showUI(int len, int score, int record, int lev, int life = 0)：显示游戏的 UI 界面。加载并显示头像图片，设置文本样式和颜色，并在指定位置显示游戏相关信息，如长度、得分、最高历史记录、游戏时间等。

addWALL(int x, int y)：在指定位置添加墙体。将墙体的坐标存储到 addWall 数组中

getTime()：获取游戏开始的时间，并计算已经经过的秒数。

③ 食物（Animal 类）

成员变量：

wall：存储 Wall 类的引用，用于访问和操作游戏地图。

cuteX[50]：存储 Piggy 的 X 坐标。

cuteY[50]：存储 Piggy 的 Y 坐标。

isEat[150][150]：记录 Piggy 是否被吃掉的标记数组。

sum：记录 Piggy 的总数量。

num：记录未被吃掉的 Piggy 数量。

构造函数：

Animal(Wall& tempWall)：初始化 Animal 类的对象。接受一个 Wall 类的引用作为参数，并将其存储在 wall 成员变量中。同时，将 cuteX、cuteY 和 isEat 数组清零。

成员函数：

randAnimal()：随机生成 Piggy。遍历游戏地图，找到可放置 Piggy 的空白位置，并在随机选择的位置放置 Piggy。Piggy 数量至少为 5 个。生成的 Piggy 的坐标存储在 cuteX 和 cuteY 数组中，并更新相关变量。

addAnimal(int x, int y)：在指定的位置添加 Piggy。将 Piggy 的坐标存储到 cuteX 和 cuteY 数组中，并将对应位置的地图字符设置为 Piggy。

clearAnimal(int lev)：清除 Piggy。遍历 Piggy 的坐标数组，将未被吃掉的 Piggy 从地图上清除，并将其标记为被吃掉。

writeEat(int x, int y)：标记指定位置的 Piggy 为已被吃掉。

showAnimal()：显示 Piggy。使用图像库加载 Piggy 的图片，并根据 Piggy 的坐标位置绘制 Piggy。

④ 菜单（Menu 类）

主要成员函数：

getScore(int Lv)：根据游戏难度等级获取分数。根据传入的游戏难度等级，从记录数组中查找对应难度的最高分数，并返回该分数。

search()：搜索历史记录。通过输入对话框获取要搜索的用户名，并在历史记录链表中查找对应用户名的记录。如果找到记录，显示确认对话框，确认是否为该玩家，是则返回查找次数，否则继续搜索。如果未找到记录，显示提示框。

Menu(Wall& tempwall)：构造函数，对 Menu 类的对象进行初始化。设置控制台光标的显示和隐藏、初始化菜单选项和难度等级数组。

主要成员变量：

choseNum：记录菜单选项的选择。

choseLevel：记录游戏难度等级的选择。

sumHis：记录历史记录的数量。

record[10]：存储历史记录的数组，包括记录的名称和分数。

h[10]：存储历史记录的结构体数组，包括记录的难度等级、名称和分数。

head：历史记录的链表头指针。

isChoose：记录是否进行了菜单选择。

isFinish：记录是否完成游戏。

实现了游戏菜单和历史记录的管理。它可以根据游戏难度等级获取最高分数，搜索并显示历史记录，以及对菜单进行初始化。

（3）游戏菜单实现

主菜单 homeChose（）函数：通过 Easyx 中的鼠标操作监视函数，对鼠标点击操作和点击位置判断，返回不同的值，从而达到点击菜单项的目的。

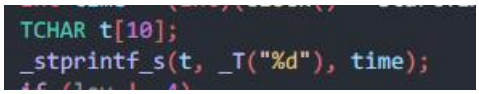
游戏模式选择同上。

（4）一些必要的交互操作

例如游戏内暂停，返回界面，输入提示，记录用户名等较为重要的部分，以及为了提高游戏美观度和玩家体验感所添加的音效播放，easyx 界面美化等。

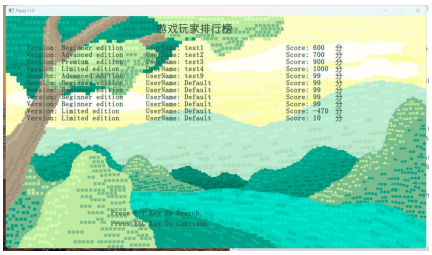
这里就显现出类的好处，逻辑条理清晰。

但是遇到问题，就是 easyx 库中的 outtextxy 函数仅支持特定类型的数据，如何将 int 型和 string 型变量也进行输出（不仅仅是屏幕上的还有提示框等），搜索发现，可以自定义类型转化函数，实现各个类型的转化，也可以



（5）文件的读取与记录

由于最高分实时更新，每次读取后，在队列里比较有些麻烦，因此，直接新建一个 rec.txt 文件存储每个模式的最高分。



历史记录函数 displayHistory（）

函数逻辑如下：

打开历史记录文件 his.txt，必要的错误处理。创建一个头结点 head。使用节点循环读取文件中的历史记录数据，并创建对应的链表节点。将历史记录存储在链表节点中。使用循环遍历链表，将排行榜的内容输出到屏幕上。

显示完成后增加了两个键读取判断，如果按下 F/f 键可以调用 search()函数进行搜索与删除操作。

弹出输入框，要求输入"delete"来确认删除操作。

如果输入的是"delete"，则从链表中删除对应的节点，更新链表和文件内容，并显示删除成功的提示信息。

调用 displayHistory()函数重新显示排行榜。

如果按下的是"ESC"键，退出函数。

displayHistory()函数，由于时间所剩不多，本游戏仅实现了查找以及删除操作，增加与修改按照删除的部分稍作修改即可。

三．在实验过程中遇到的问题及解决方法

1.图像大小

使用二维数组 map 进行地图信息的存储，地图大小 1220*660；使用 easyx 贴图的话，如果每个像素点贴图，那显然太小，所以需要成比例放大，起初使用 10 倍放大，及输出位置（i，j）均乘于 10，实现等比例放大，但是到后期，发现 10*10 图片仍是很小，显得地图又太大了，想要更改，但是发现有许多地方都写死了，没有养成使用宏定义的习惯，所以改起来 bug 很多。

2.贴图

透明贴图的问题,搜索发现easyx实现透明贴图有几种方式,例如基于 Windows API 函数 TransparentBlt；基于直接操作显示缓冲区；使用三元光栅操作实现透明贴图（掩码图）等。尝试了几个，由于程序内屏幕刷新频率过快，导致了透明贴图成功，但是频繁闪烁的情况。导致只有在新一次按键时才会出现贴图。经过搜索发现可以通过双缓存操作解决，短暂尝试未果，迫于时间紧迫，先放弃，待以后尝试。
还有之前同学询问过她在 cleardevice（）之后，屏幕不再绘画，好像卡住一样，使用 flushbatch();刷新之后成功，并搜索发现，可能是窗口分辨率的原因，窗口分辨率小时易出现绘图卡住。
总之，学到了很多绘图方面的知识，了解了很多。

3.音乐播放

最初播放音效只播放一次，通过搜索发现了循环播放，以及每次都要播放应该怎么做。

加入音效以后，例如吃到星星时，音效播放慢半拍，通过调节播放起始位置解决。

4.贪吃蛇中的链表

使用单链表储存蛇以及使用链表储存历史数据等，出现了许多问题，例如边界处理，不对应等，经过画图分析等方式成功解决。

5.多源文件，多头文件的冲突问题

为了使代码条理清晰，将每个部分有意分割成不同文件，但是也遇到了这个头文件中的工具函数仅能在部分源文件中使用，如果多个源文件/头文件都包含了这个头文件，会出现函数重复定义的问题，编译才会报错，而且 error 巨多，多次遇到，这次也意识到是冲突的问题，所以尝试使用公用头文件 Tool.h 来包含都要用得到的工具函数。还有头文件仅是函数的声明，函数的定义应当尽量放在同名的源文件中等方法应也可以解决。这些问题也可以通过以下方法解决：

①使用头文件保护宏（Header Guards）：在头文件的开头和结尾添加预处理指令，以防止头文件被重复包含。可以使用条件编译指令，如#ifndef、#define 和#endif 来创建头文件保护宏。

通过使用头文件保护宏，即使多个源文件中都包含了同一个头文件，编译器只会处理一次该头文件的内容，避免了重复定义和重复包含的问题。

②使用 extern 关键字声明变量：如果在多个源文件中使用了同名的全局变量，会导致重复定义的错误。可以使用 extern 关键字在声明变量时指示该变量是在其他源文件中定义的。

6. 面向对象

虽然有意识地使用面向对象的方法，将不同对象封装为为一个类，封装函数以及数据成员，但是本程序还是大多按照面向过程的方式进行编写，模块化较之前有些进步，学到了很多。例如 Animal 类，与其他食物类型，有共同点，可以继承自 Animal 类，或使用老师新讲的多态，虚函数等概念，虽然有想法，但是知识点掌握不够，实行起来 bug 过多。

四．心得体会

1. 极大地体会到了互联网的便利，以及人工智能的便捷高效。贪吃蛇由于比较经典，网上参考资料很多，通过学习效仿他人的成果经验，可以思路更开阔，避免在一些弯路上浪费时间。以及有许多小 bug，网上没有资料，可以通过 ChatGPT 来辅助纠正错误，以及编写一些小工具，例如本程序中的类型转化函数。

2. 对面向对象的理解更深了一步，也亲身感受到面向对象的便利。第二版的面向对象要比第一版的面向过程扩展性高很多，编写起来也条理清晰。

3. 数据结构的选择：虽然有意识地使用面向对象的方法，将不同对象封装为为一个类，封装函数以及数据成员，但是本程序还是大多按照面向过程的方式进行编写，模块化较之前有些进步，学到了很多。例如 Animal 类，与其他食物类型，有共同点，可以继承自 Animal 类，或使用老师新讲的多态，虚函数等概念，虽然有想法，但是知识点掌握不够，实行起来 bug 过多。

4. 交互性，大多数程序还是面向用户的，所以要合理地规划函数，给与必要的功能，异常处理，可视化操作等，方便用户使用。

5. 异常处理的重要性：在文件操作、内存分配和算法执行过程中，可能会出现各种异常情况。要有必要的异常处理机制，避免潜在的错误和问题，队列和记录是链表，动态申请而来，应当要释放，以及文件操作的方式，操作后要即使关闭。

6. 模块化设计，规范命名，提高代码的可读性和可维护性。在编写程序时，有意识地将各个功能分割，将相关功能的函数放在独立的 cpp 文件中，头文件放置函数及其他数据结构的声明，单独的 cpp 文件放置函数及数据结构的定义及实现，main 函数中调用这些函数。可维护性强，代码结构更加清晰和模块化。

五. 源代码

头文件

Animal.h

```
#pragma once
#include"Wall.h"
#include "enum.h"
#include<graphics.h>
#include"tools.h"
class Animal
{
public:
	Animal(Wall& tempWall) : wall(tempWall)
	{
		memset(cuteX, 0, sizeof(cuteX));
		memset(cuteY, 0, sizeof(cuteY));
		memset(isEat, 0, sizeof(isEat));
	};
	bool randAnimal();
	void addAnimal(int x, int y);
	void clearAnimal(int level);
	void writeEat(int x, int y);
	void showAnimal();
	COORD getAnimal();
	bool checkIfEat(COORD& a);
	int num;
protected:
	Wall& wall;
	int cuteX[50];
	int cuteY[50];
	int sum;
	int isEat[150][150];
};

COORD Animal::getAnimal()
{
	while (1)
	{
		int i = rand() % sum;
		if (wall.getWhat(cuteX[i], cuteY[i]) == ANIMAL && !isEat[cuteX[i]][cuteY[i]])
		{
			COORD a;
			a.X = cuteX[i];
			a.Y = cuteY[i];
			return a;
		}
	}
}

bool Animal::checkIfEat(COORD& a)
{
	if (isEat[a.X][a.Y])
		return 1;
	else
		return 0;
}



bool Animal::randAnimal()
{
	bool good = 0;
```

```
	bool break0 = 0;
	num = 0;
	for (int x = 1; x <= wall.COL; x++)
	{
		for (int y = 1; y <= wall.ROW; ++y)
		{
			if (wall.getWhat(x, y) == SPACE)
			{
				good = 1;
				num++;
			}
			if (num >= 5)
			{
				break0 = 1;
				break;
			}
		}
		if (break0)
			break;
	}
	if (!good)
		return 0;
	srand((unsigned int)time(NULL));
	int num2 = 0;
	int n = rand() % num + 1;
	num = n;
	for (int i = 0; i < n; ++i)
	{
		while (1)
		{
			int x = rand() % (wall.COL - 50) + 20;
			int y = rand() % (wall.ROW - 50) + 20;
			if (wall.getWhat(x, y) == SPACE)
			{
				wall.change(x, y, ANIMAL);
				isEat[x][y] = 0;
				cuteX[num2] = x;
				cuteY[num2] = y;
				num2++;
				break;
			}
		}
	}
	sum = num;
	return 1;

}


void Animal::addAnimal(int x, int y)
{
	cuteX[sum] = x;
	cuteY[sum] = y;
	wall.change(x, y, ANIMAL);
	isEat[x][y] = 0;
	num++;
	sum++;
}

void Animal::clearAnimal(int lev)
{

	for (int i = 0; i < sum; i++)
	{
		if (!isEat[cuteX[i]][cuteY[i]])
		{
			wall.change(cuteX[i], cuteY[i], SPACE);
			isEat[cuteX[i]][cuteY[i]] = 1;
		}
```

```
            }
            sum = 0;
            num = 0;
      }

      void Animal::writeEat(int x, int y)
      {
            num--;
            isEat[x][y] = 1;
      }

      void Animal::showAnimal()
      {
            IMAGE pig(10,10);
            loadimage(&pig, _T("pig.png"), 10, 10);
            for (int i = 0; i < sum; ++i)
            {
                  if (!isEat[cuteX[i]][cuteY[i]])
                  {
                        putimage(10*cuteX[i], 10*cuteY[i], &pig);
                  }
            }
}
```

DuiLie.h

```
      #pragma once
      #include"Wall.h"
      #include"Animal.h"
      #include "enum.h"
      #include<fstream>
      #include<iostream>
      using namespace std;

      class duiLie
      {
      private:
            int len;
            int life;
            int score;
            char lastDir;
            bool ifDie;
            int sleepChange1;
            int sleepChange2;
            //
            struct node //链表储存
            {
                  int x;
                  int y;
                  node* next;
            };
            node* head; //头

      public:
            duiLie(Wall& tempWall, Animal& tempAnimal): wall(tempWall),animal(tempAnimal) {
                  head = new node;
                  ifDie = false;
                  len = 4;
                  score = 0;
                  life = 5;
                  lastDir = UP;
```

```
        Lv = 1;
        ifGift = 0;
        ifPoison = 0;
        sleepChange1 = 0;
        sleepChange2 = 0;
    }
    Wall& wall;
    Animal& animal;
    void move(int x, int y);
    int Lv;
    bool ifGift;
    bool ifPoison;

    void initDuiLie();
    bool randBorn();
    bool ISDEAD();
    int getScore();
    int getLength();
    int getLife();
    int getHx();
    int getHy();
    void moveDuiLie(char Dir);
    void sleep();
    void deleteDuiLie();
    void saveData();
    void duiLieToWall();
    void duiLieToAnimal();
    void duiLieToSpace();
    void Die();
    void reduceLife(int n);
    void showDuiLie();
};

void duiLie::move(int a, int b)
{
    bool ifDrawHead = 1;
    node* p = head;
    node* last = NULL;
    p = new node;
    p->x = head->x + a;
    p->y = head->y + b;
    p->next = head;
    head = p;
    while (p->next != NULL)
    {
        last = p;
        p = p->next;
    }
    if (Lv == 4)
    {

        bool isAccelerating = false;  // 标记是否正在加速
        bool isDecelerating = false;  // 标记是否正在加速
            // 检测 Shift 键的状态
```

```
            if (GetAsyncKeyState(VK_SHIFT) & 0x8000)
                isAccelerating = true;
            else
                isAccelerating = false;
            if (isAccelerating)
                sleepChange1 = -100;
            else
                sleepChange1 = 0;


            if (GetAsyncKeyState(VK_CONTROL) & 0x8000)
                isDecelerating = true;
            else
                isDecelerating = false;
            if (isDecelerating)
                sleepChange2 = 100;
            else
                sleepChange2 = 0;
    }
    if (wall.getWhat(head->x, head->y) == SPACE || (head->x == p->x && head->y == p->y))
    {
        wall.change(p->x, p->y, SPACE);
        delete p;
        last->next = NULL;
    }
    if (wall.getWhat(head->x, head->y) == ANIMAL)
    {
        mciSendString(TEXT("play eatmusic from 0"), NULL, 0, NULL);
        animal.writeEat(head->x, head->y);
        len++;
        score += 10;
    }
    if (wall.getWhat(head->x, head->y) == GIFT)
    {
        mciSendString(TEXT("setaudio giftmusic volume to 1000"), NULL, 0, NULL);
        mciSendString(TEXT("play giftmusic from 800"), NULL, 0, NULL);
        ifGift = 0;
        len+=3;
        wall.change(head->x, head->y, SPACE);
        score += 100;
    }
    if (wall.getWhat(head->x, head->y) == POISON)
    {
        mciSendString(TEXT("play poisonmusic from 20"), NULL, 0, NULL);
        score -= 150;
        PoisonNote();//中毒
        wall.change(head->x, head->y, SPACE);
    }

    if (wall.getWhat(head->x, head->y) == WALL)
    {
        mciSendString(TEXT("play wallmusic from 20"), NULL, 0, NULL);
        ifDie = TRUE;
        life--;
```

```
            return;
        }

        //身体
        if (wall.getWhat(head->x, head->y) == BODY)
        {
            mciSendString(TEXT("play wallmusic from 20"), NULL, 0, NULL);
            ifDie = TRUE;
            life--;
            return;
        }

        //剩余身体的绘制
        p = head;
        while (p != NULL)
        {
            wall.change(p->x, p->y, BODY);
            p = p->next;
        }

    }

    void duiLie::initDuiLie()
    {
        ifDie = false;
        lastDir = UP;
    }
    bool duiLie::randBorn()
    {
        bool ifBorn = 0;
        int bornX[500] = { 0 };
        int sum = 0;

        for (int i = 0; i <= wall.COL; ++i)
        {
            for (int j = 1; j <= wall.ROW - 3; ++j)
            {
                if (wall.getWhat(i, j) == SPACE && wall.getWhat(i, j + 1) == SPACE && wall.getWhat(i, j
+ 2) == SPACE && wall.getWhat(i, j + 3) == SPACE)
                {
                    ifBorn = 1;
                    bornX[sum++] = i;
                    break;
                }
            }
        }
        if (ifBorn)
        {
            ifDie = false;
            lastDir = UP;

            srand((unsigned int)time(NULL));
            int x = bornX[rand() % sum];
            int y = rand() % (wall.ROW - 3) + 1;
```

```
            while (1)
            {
                if (wall.getWhat(x, y) == SPACE && wall.getWhat(x, y + 1) == SPACE && wall.getWhat(x, y + 2) == SPACE && wall.getWhat(x, y + 3) == SPACE)
                {
                    break;
                }
                y = rand() % (wall.ROW - 3) + 1;
            }
            len = 4;

            //
            head = new node;
            head->x = x;
            head->y = y;
            head->next = NULL;
            wall.change(head->x, head->y, BODY);
            node* now = NULL;
            node* last = head;
            for (int i = 1; i < 4; ++i)
            {
                wall.change(x, y + i, BODY);
                now = new node;
                now->x = x;
                now->y = y + i;
                last->next = now;
                last = now;

            }
            now->next = NULL;
            return 1;
        }
        else
            return 0;
    }
    bool duiLie::ISDEAD()
    {
        return ifDie;
    }
    int duiLie::getScore()
    {
        return score;
    }
    int duiLie::getLength()
    {
        return len;
    }

    int duiLie::getLife()
    {
        return life;
    }

    int duiLie::getHx()
```

```
    {
        return head->x;
    }

    int duiLie::getHy()
    {
        return head->y;
    }

    void duiLie::moveDuiLie(char Dir)
    {
        if ((lastDir == UP && Dir == DOWN) || (lastDir == DOWN && Dir == UP) || (lastDir == LEFT && Dir
== RIGHT) || (lastDir == RIGHT && Dir == LEFT))
        {
            Dir = lastDir;
        }
        switch (Dir)
        {
            case UP:
                move(0, -1);
                break;
            case DOWN:
                move(0, 1);
                break;
            case RIGHT:
                move(1, 0);
                break;
            case LEFT:
                move(-1, 0);
                break;
        }
        lastDir = Dir;
    }

    void duiLie::sleep()
    {
        switch (Lv)
        {
            case 1:
                Sleep(60);
                break;
            case 2:
                Sleep(50);
                break;
            case 3:
                Sleep(3);
                break;
            case 4:
            {
                int s = 100 + sleepChange1 + sleepChange2;
                if (s < 3)
                    s = 2;
                Sleep(s);
                break;
```

```
            }
        }
    }

    void duiLie::deleteDuiLie()
    {
        node* p = head;
        node* k = NULL;
        while (p != NULL)
        {
            k = p;
            p = p->next;
            delete k;
        }
    }

    void duiLie::saveData()
    {
        //open file
        fstream frec("rec.txt");
        if (!frec.is_open())
        {
            HWND hwnd = GetHWnd();//获取窗口句柄
            MessageBox(hwnd, L"Fail to open the Record File!", L"Error", MB_OK);//设置模态对话框
            return;
        }
        fstream fhis("his.txt");
        if (!fhis.is_open())
        {
            HWND hwnd = GetHWnd();
            MessageBox(hwnd, L"Fail to open the History File!", L"Error", MB_OK);
            return;
        }
        string name[6] = { " ","---","---","---" ,"---","---" };
        int userScore[6] = { 0 };
        for (int i = 0; i < 5; ++i)
        {
            frec >> name[i] >> userScore[i];
        }

        if (userScore[Lv] < score)
        {
            wchar_t USERNAME[10];
            InputBox(USERNAME, 10, L"恭喜你打破了本模式记录！\n请输入昵称", L"记录玩家昵称",
L"Default");
            name[Lv] = ConvertWCharToString(USERNAME);
            userScore[Lv] = score;
            frec.clear();
            frec.seekp(0, ios::beg);
            for (int i = 1; i <= 5; ++i)
            {
                frec << name[i] << " " << userScore[i] << " ";
            }
            fhis.seekp(0, ios::end);
```

```
            fhis << " " << Lv << " " << name[Lv] << " " << score;

        }
        else {
            wchar_t USERNAME[10];
            InputBox(USERNAME, 10, L"很遗憾没有打破纪录，但是你也很厉害了！\n请输入昵称", L"起个名字",
L"Default");
            //name[Lv] = USERNAME.str();
            name[Lv] = ConvertWCharToString(USERNAME);
            fhis.seekp(0, ios::end);
            fhis << " " << Lv << " " << name[Lv] << " " << score;
            //system("pause");
        }


        frec.close();
        fhis.close();

    }

    void duiLie::duiLieToWall()
    {
        node* p = head->next;
        int x, y;
        while (p != NULL) {
            x = p->x;
            y = p->y;
            wall.change(x, y, WALL);
            wall.addWALL(x, y);
            p = p->next;
        }
    }

    void duiLie::duiLieToAnimal()
    {
        node* p = head->next;
        int x, y;
        while (p != NULL)
        {
            x = p->x;
            y = p->y;
            animal.addAnimal(x, y);
            p = p->next;
        }
    }

    void duiLie::duiLieToSpace()
    {
        node* p = head->next;
        int x, y;
        while (p != NULL)
        {
            x = p->x;
            y = p->y;
```

```
            wall.change(x, y, SPACE);
            p = p->next;
        }
    }

    void duiLie::Die()
    {
        mciSendString(TEXT("play diemusic from 0"), NULL, 0, NULL);
        IMAGE over(WIDTH, HIGH);
        loadimage(&over, L"over.jpg", WIDTH, HIGH);//加载背景图片
        putimage(0, 0, &over);
        setbkmode(TRANSPARENT);
        settextstyle(30, 0, _T("幼圆"), 0, 0, 800, 0, 0, 0);
        settextcolor(HSLtoRGB(157, 0.20f, 0.26f));
        outtextxy(410, 500, _T("Press Any Key To Continue."));
    }

    void duiLie::reduceLife(int n)
    {
        life -= n;
    }

    void duiLie::showDuiLie()
    {
        node* p = head;
        while (p != NULL)
        {
            if (p == head)
            {
                IMAGE sn(13, 13);
                loadimage(&sn, _T("sn.png"), 13, 13);
                putimage(10 * p->x, 10 * p->y, &sn);
            }
            else {
                IMAGE pig(10, 10);
                loadimage(&pig, _T("pig.png"), 10, 10);
                putimage(10 * p->x, 10 * p->y, &pig);
            }
            p = p->next;
        }
    }
```

一些函数

```
    void Menu::displayHistory()
    {
        IMAGE img_bk3(WIDTH, HIGH);
        loadimage(&img_bk3, L"rank.png", WIDTH, HIGH);//加载背景图片
        putimage(0, 0, &img_bk3);

        setbkmode(TRANSPARENT);
        //settextstyle(80, 0, _T("幼圆"));
        settextstyle(20, 0, _T("幼圆"), 0, 0, 800, 0, 0, 0);
        settextcolor(HSLtoRGB(157, 0.20f, 0.26f));
```

```
        //
        LOGFONT f;//字体变量
        gettextstyle(&f);
        f.lfQuality = ANTIALIASED_QUALITY;
        settextstyle(&f);

        fstream fhistory("his.txt");
        if (!fhistory.is_open())
        {
            HWND hwnd = GetHWnd();
            MessageBox(hwnd, L"Fail to open the History File!", L"Error", MB_OK);
            return;
        }

        head = new hisDate;
        hisDate* p = head, * q = NULL;
        sumHis = 0;
        while (fhistory.peek() != EOF)
        {
            q = new hisDate;
            p->next = q;
            p = q;
            fhistory >> p->hisLv >> p->name >> p->hisScore;
            sumHis++;
        }
        p->next = NULL;
        fhistory.close();

        {
            hisDate* p = head->next, * q = NULL;
            int n = 1;
            settextstyle(30, 0, _T("幼圆"), 0, 0, 800, 0, 0, 0);
            outtextxy(WIDTH / 3+30, 20, L"游戏玩家排行榜");
            settextstyle(20, 0, _T("幼圆"), 0, 0, 800, 0, 0, 0);
            while (p != NULL)
            {
                if (n<=30)
                {
                    outtextxy(WIDTH / 20, 60+20*n, StringToLPCTSTR("Version: " + Level[p->hisLv]));
                    outtextxy(WIDTH / 3, 60+20*n, StringToLPCTSTR("UserName: " + p->name));
                    outtextxy(WIDTH / 2+200, 60+20*n, StringToLPCTSTR("Score: " +
to_string(p->hisScore)));
                    outtextxy(WIDTH / 2 + 340, 60 + 20 * n, L"分");
                }
                p = p->next;
                n++;
            }
            outtextxy(WIDTH / 2 - 300, HIGH / 2 + 220, L"Press F/f Key To Search.");
            outtextxy(WIDTH / 2 - 300, HIGH / 2 + 250, L"Press ESC Key To Continue.");

            while (true) {
                string nameTosearch;
                int ch = _getch();
                if (ch == 70 || ch == 102) {  // 检查"F"键
```

```
                    int times = search();
                    cout << times;
                    if (times) {
                        wchar_t inputs0[10];
                        InputBox(inputs0, 10, L"输入“delete/Delete/Del/del 删除本用户”\n", L"删除",
L"Del");
                        std::wstring inputs(inputs0);
                        if (inputs == L"delete"|| inputs == L"Delete" || inputs == L"del" || inputs ==
L"Del")
                        {
                            hisDate* p = head->next, * last = head;
                            for (int i = 0; i < times-1; ++i)
                            {
                                last = p;
                                p = p->next;
                            }
                            last->next = p->next;
                            delete p;
                            sumHis--;


                            // 清空文件内容
                            std::ofstream file2("his.txt", std::ios::out | std::ios::trunc);
                            file2.clear();
                            file2.close();

                            // 重新写入文件
                            std::ofstream file("his.txt", std::ios::out | std::ios::app);
                            hisDate* currentNode = head->next;
                            while (currentNode != nullptr) {
                                file << currentNode->hisLv << " " << currentNode->name << " " <<
currentNode->hisScore;

                                if (currentNode->next != nullptr) {
                                    file << " ";
                                }
                                currentNode = currentNode->next;
                            }
                            currentNode = NULL;
                            file.close();


                            HWND hwnd = GetHWnd();
                            MessageBox(hwnd, L"已删除", L"Delete", MB_OK);
                            displayHistory();
                        }
                    }
                }

                if (ch == 27) {  // 检查"ESC"
                    return;
                }
            }

        }
```

```
}

void homeChose()
{
    mciSendString(L"open click.mp3 alias clickmusic", NULL, 0, NULL);
    ExMessage m;
    int condition = 1;
    while (condition == 1)
    {
        // 获取一条鼠标或按键消息
        m = getmessage(EX_MOUSE | EX_KEY);

        switch (m.message)
        {
            case WM_MOUSEMOVE:
                break;
            case WM_LBUTTONDOWN:
                if (m.ctrl) {
                }
                else {
                    if (m.x >= WIDTH / 2 - 120 && m.x <= WIDTH / 2 + 60)
                    {
                        if (m.y >= HIGH / 2 - 60 && m.y <= HIGH / 2 - 30)
                        {

                            homeMenu();
                            outtextxy(WIDTH / 2 - 150, HIGH / 2 - 60, L"=>");
                            mciSendString(TEXT("play clickmusic from 0"), NULL, 0, NULL);
                            STATUS = 1;
                            Sleep(500);
                            return;
                        }
                        if (m.y >= HIGH / 2 - 15 && m.y <= HIGH / 2 + 15)
                        {
                            homeMenu();
                            outtextxy(WIDTH / 2 - 150, HIGH / 2 - 15, L"=>");
                            mciSendString(TEXT("play clickmusic from 0"), NULL, 0, NULL);
                            STATUS = 2;
                            Sleep(500);
                            return;
                        }
                        if (m.y >= HIGH / 2 + 30 && m.y <= HIGH / 2 + 60)
                        {
                            homeMenu();
                            outtextxy(WIDTH / 2 - 150, HIGH / 2 + 30, L"=>");
                            mciSendString(TEXT("play clickmusic from 0"), NULL, 0, NULL);
                            STATUS = 3;
                            Sleep(500);
                            return;
                        }
                        if (m.y >= HIGH / 2 + 75 && m.y <= HIGH / 2 + 175)
                        {
                            homeMenu();
```

```
                                outtextxy(WIDTH / 2 - 150, HIGH / 2 + 75, L"=>");
                                mciSendString(TEXT("play clickmusic from 0"), NULL, 0, NULL);
                                STATUS = 4;
                                Sleep(500);
                                return;
                            }
                        }
                    }
                    break;
                case WM_KEYDOWN:
                    if (m.vkcode == VK_ESCAPE)
                        return;  // 按 ESC 键退出程序
            }
        }
    }

    int startLV()
    {
        mciSendString(L"open click.mp3 alias clickmusic", NULL, 0, NULL);
        choseLV = 0;
        ExMessage m;      // 定义消息变量
        int condition = 1;
        showStartLv();
        while (condition == 1)
        {
            // 获取一条鼠标或按键消息
            m = getmessage(EX_MOUSE | EX_KEY);

            switch (m.message)
            {
                case WM_MOUSEMOVE:
                    break;
                case WM_LBUTTONDOWN:
                    if (m.ctrl) {
                    }
                    else {
                        if (m.x >= WIDTH / 2 - 120 && m.x <= WIDTH / 2 + 60)
                        {
                            if (m.y >= HIGH / 2 - 60 && m.y <= HIGH / 2 - 30)
                            {
                                showStartLv();
                                outtextxy(WIDTH / 2 - 150, HIGH / 2 - 60, L"=>");
                                mciSendString(TEXT("play clickmusic from 0"), NULL, 0, NULL);
                                Sleep(400);
                                choseLV = 1;
                                return choseLV;
                            }
                            if (m.y >= HIGH / 2 - 15 && m.y <= HIGH / 2 + 15)
                            {
                                showStartLv();
                                outtextxy(WIDTH / 2 - 150, HIGH / 2 - 15, L"=>");
                                mciSendString(TEXT("play clickmusic from 0"), NULL, 0, NULL);
                                Sleep(400);
                                choseLV = 2;
```

```
                    return choseLV;
                }
                if (m.y >= HIGH / 2 + 30 && m.y <= HIGH / 2 + 60)
                {
                    showStartLv();
                    outtextxy(WIDTH / 2 - 150, HIGH / 2 + 30, L"=>");
                    mciSendString(TEXT("play clickmusic from 0"), NULL, 0, NULL);
                    Sleep(400);
                    choseLV = 3;
                    return choseLV;
                }
                if (m.y >= HIGH / 2 + 75 && m.y <= HIGH / 2 + 175)
                {
                    showStartLv();
                    outtextxy(WIDTH / 2 - 150, HIGH / 2 + 75, L"=>");
                    mciSendString(TEXT("play clickmusic from 0"), NULL, 0, NULL);
                    Sleep(400);
                    choseLV = 4;
                    return choseLV;
                }
            }
        }
        break;
    case WM_KEYDOWN:
        if (m.vkcode == VK_ESCAPE)
            return choseLV;
    }
  }
  return choseLV;
}

long long RandomNumber() {
    std::random_device rd;
    std::mt19937 gen(rd());
    std::uniform_int_distribution<int> dist(0, 9);

    long long number = 0;
    for (int i = 0; i < 8; ++i) {
        number = number * 10 + dist(gen);
    }
    return number;
}

int checkInput(long long number) {
    long long input = 0;
    for (int i = 0; i < 8; ++i) {
        char ch = _getch();
        if (ch >= '0' && ch <= '9') {
            input = input * 10 + (ch - '0');
        }
        else {
            i--;
        }
    }
```

```
        if (input == number) {
            return 1;
        }
        else {
            return 0;
        }
    }

    void PoisonNote()
    {
        long long number = RandomNumber();
        HWND hwnd = GetHWnd();
        wchar_t message[100];
        swprintf_s(message, L"中毒了!\n 按照顺序敲击 8 位数字串解毒！\n%I64d", number);
        MessageBox(hwnd, message, L"中毒", MB_OK);

        if (checkInput(number)) {
            mciSendString(TEXT("play fuhuomusic from 20"), NULL, 0, NULL);
            return;
        }
        else {
            mciSendString(TEXT("play poison2music from 0"), NULL, 0, NULL);
            MessageBox(hwnd, L"解毒失败，暂停两秒！", L"中毒", MB_OK);
            Sleep(2000);
        }
        Sleep(1000);
}
```