《贪吃蛇》大作业报告

班级：智能交通与车辆 14 班

学号：2152402

姓名：段婷婷

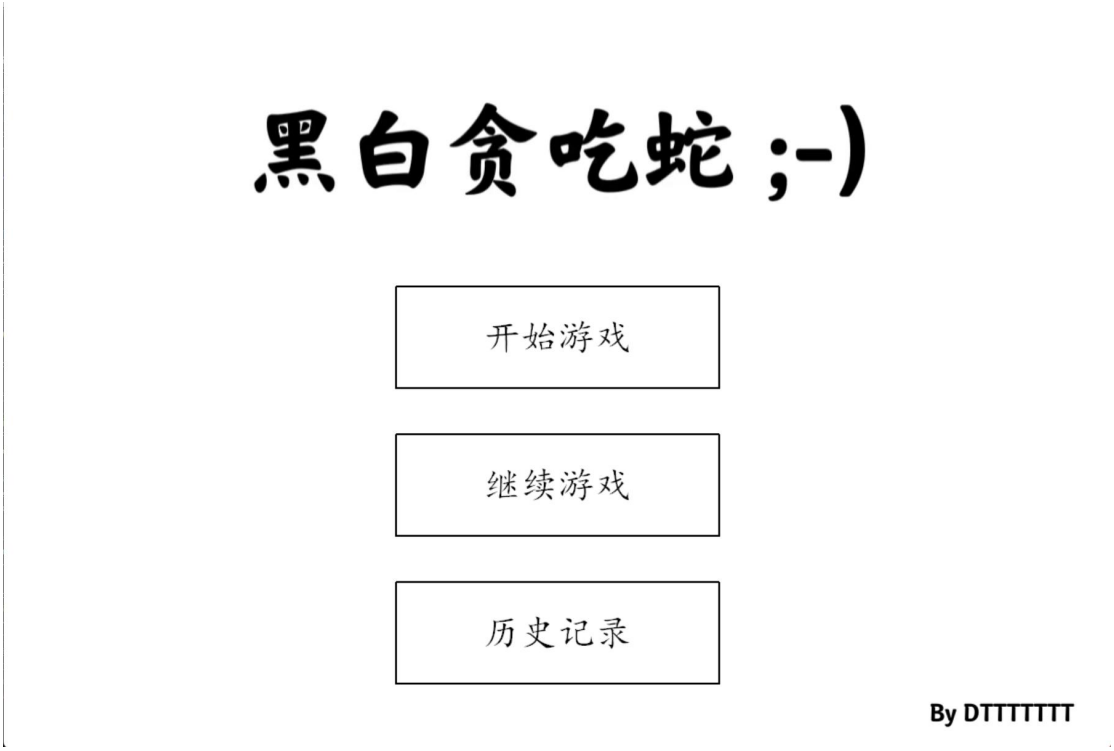
完成日期：2022.5.25

目录

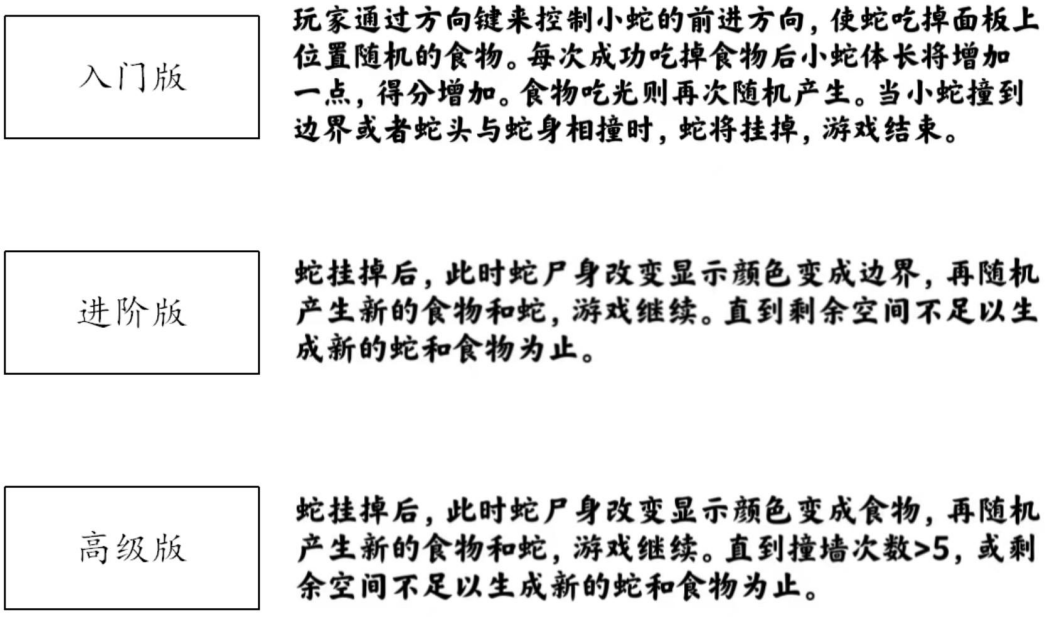
一. 功能描述与设计思路

1. 功能描述:

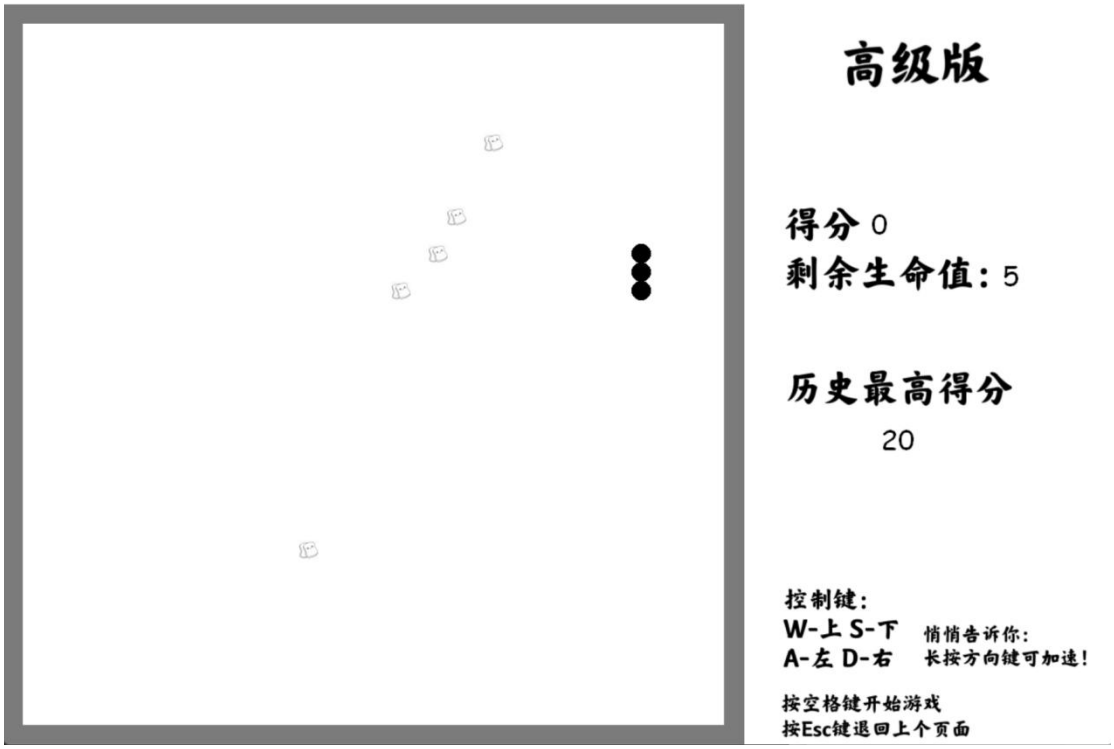
主菜单如下:



(1) 【开始游戏】



游戏分为三个模式：【入门版】【进阶版】【高级版】，游戏规则见上图。以【高级版】为例：



页面左侧是游戏方块，右侧是实时 UI。实时 UI 会显示当前得分、剩余生命值、历史最高分（若当前分数高于之前的历史最高分，则更新历史最高分为当前分数）。

游戏的控制键为 WSAD，长按方向键可以**加速**。进入游戏页面后，按空格键开始游戏。开始游戏后，可随时按 Esc 键退出至主菜单页面。游戏结束后（不同模式有不同的结束规则），同样按 Esc 键退出至主菜单页面。（此处需要注意两点 ~~两个 BUG~~，第一个是只有开始游戏后才能退出页面，第二个是按 Esc 键会退出至主菜单页面，而非上个页面）

(2) 【继续游戏】

可以继续**上一次未完成**的游戏，详细阐述如下：

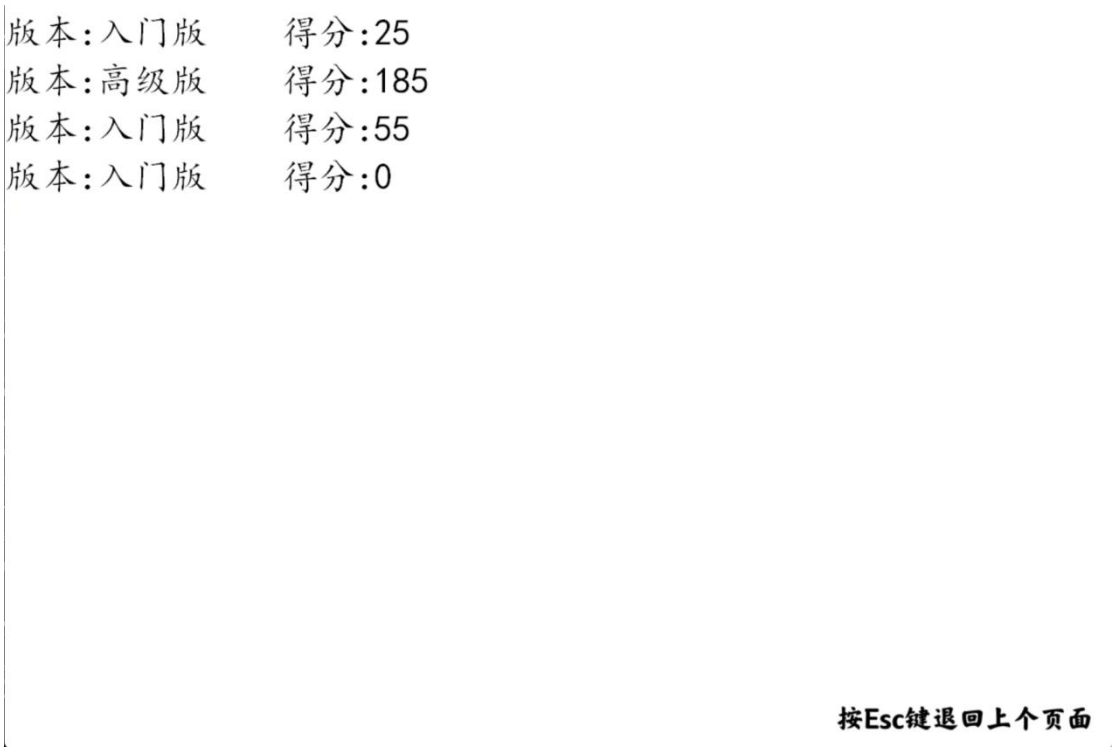
(1) 若**上一次**开启新一局游戏后，游戏尚未结束时玩家按 Esc 退出了游戏页面，可以通过【继续游戏】来继续。

(2) 若**上一次**在【继续游戏】中，游戏尚未结束时玩家按 Esc 退出了游戏页面，可以通过【继续游戏】再次继续。

(3) 若上一次的游戏已经结束，则没有可继续的游戏，会出现以下提示页面：

(3) 【历史记录】

按照以下形式按照游戏时间由早到晚的顺序显示游戏记录。



2. 设计思路:

一共包括了 7 个源文件：classes.h，Game.cpp，Snake.cpp，Wall.cpp，Food.cpp , menu.cpp , main.cpp。

(1) classes.h

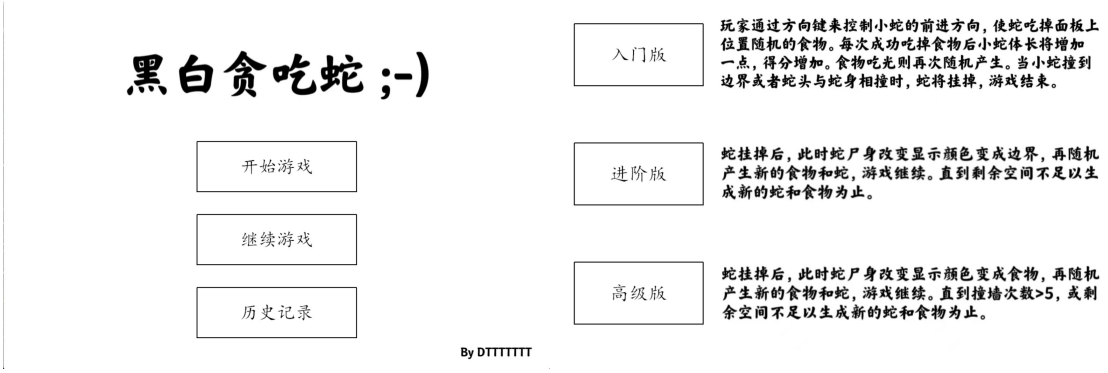
包括了所有的类（Wall, Food, Game, Snake）的声明、一些需要被多个 cpp

函数调用的函数的声明

（2）menu.cpp

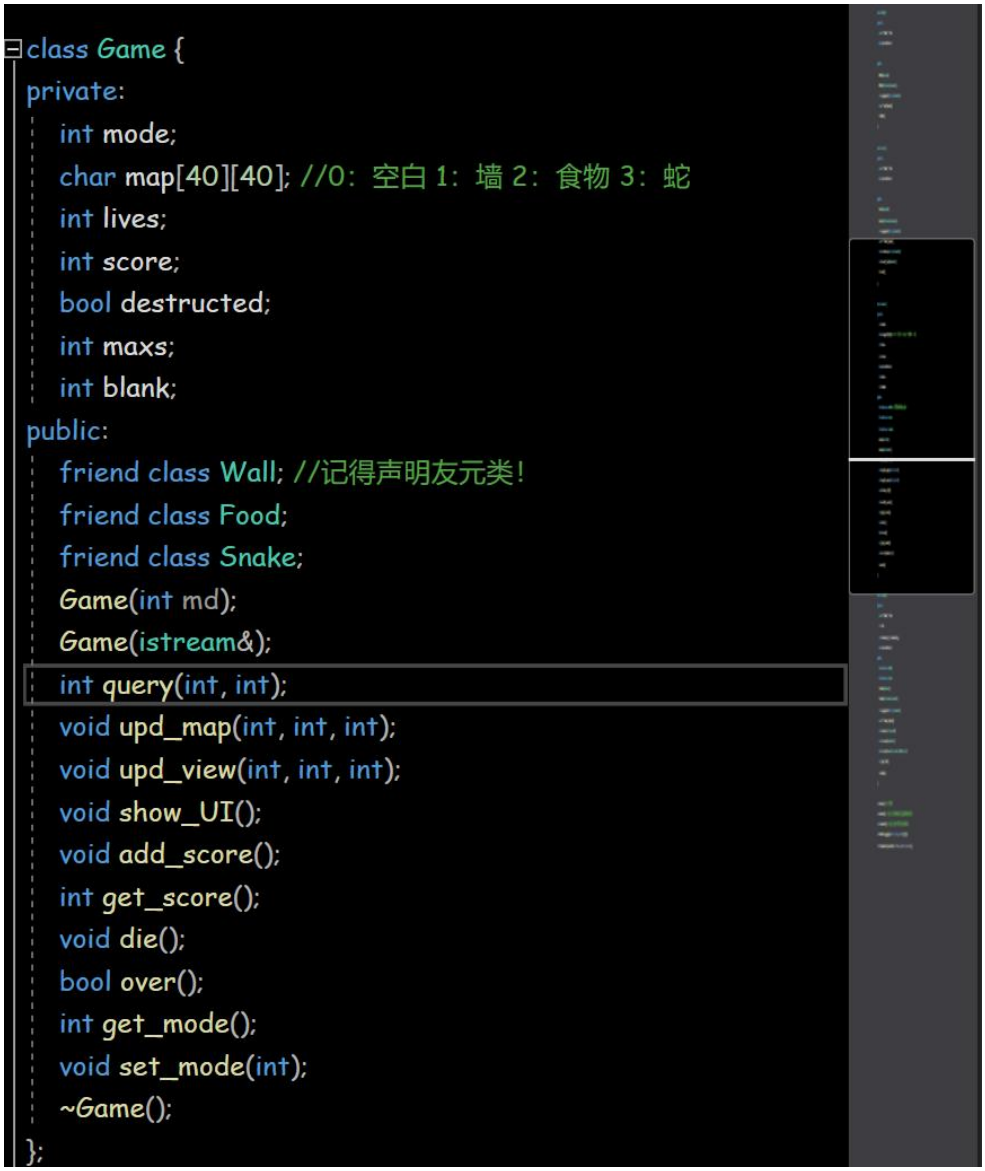
此 cpp 用于打印菜单页面，包括如下函数：

1) int button_judge(int,int,int[3][4]);
判断光标当前指在三个按钮的哪一个按钮，不同的按钮对应不同的返回值。
2) int menu();
显示主菜单页面，返回玩家选择的游戏类型。



3) int menu1();
显示游戏模式选择页面，并返回玩家选择的游戏类型。
4) void menu2();
显示【历史记录】页面，读取相应文档并且显示游戏历史记录。

（3）Game 类



```
class Game {
private:
    int mode;
    char map[40][40]; //0：空白 1：墙 2：食物 3：蛇
    int lives;
    int score;
    bool destructed;
    int maxs;
    int blank;
public:
    friend class Wall; //记得声明友元类！
    friend class Food;
    friend class Snake;
    Game(int md);
    Game(istream&);
    int query(int, int);
    void upd_map(int, int, int);
    void upd_view(int, int, int);
    void show_UI();
    void add_score();
    int get_score();
    void die();
    bool over();
    int get_mode();
    void set_mode(int);
    ~Game();
};
```

A. 包含的变量：

1）int mode：游戏模式

2）char map[40][40]：地图信息（0 代表空白，1 代表墙，2 代表食物，3 代表蛇）

3）int lives：剩余声明数

4）int score：分数

5）bool destructed：判断对象是否调用过析构函数（储存游戏记录时需要用到）

6）int maxs：当前游戏模式的最高分

7）int blank：当前地图剩余的空白格子数目，用于辅助判断是否由足够的格子来生成新的小蛇和食物。

B. 包含的函数：

1）Game(int md);

玩家选择【开始游戏】并且选择一个模式之后，初始化 Game 的对象。

2）Game(istream&);

玩家选择【继续游戏】后，读取相应文档来初始化 Game 的对象。

3）int query(int, int);
返回地图信息。

4）void upd_map(int x, int y, int tp);

更新地图信息，将坐标为(x,y)的格子更新为 tp 类型。

5）void upd_view(int, int, int);

更新视图信息，upd_view 在 upd_map 中的末尾调用。

6）void show_UI();

显示 UI 界面。

7）void add_score();
增加当前分数，若当前分数超过历史最高分数则相应地更新历史最高分数。每当小蛇吃到食物后调用此函数使分数增加 5。

8）int get_score();
获取当前分数。

9）void die();

在【高级版】中，小蛇死亡后调用此函数来更新 lives 的数目，并且更新 UI 上“剩余生命数”的显示。

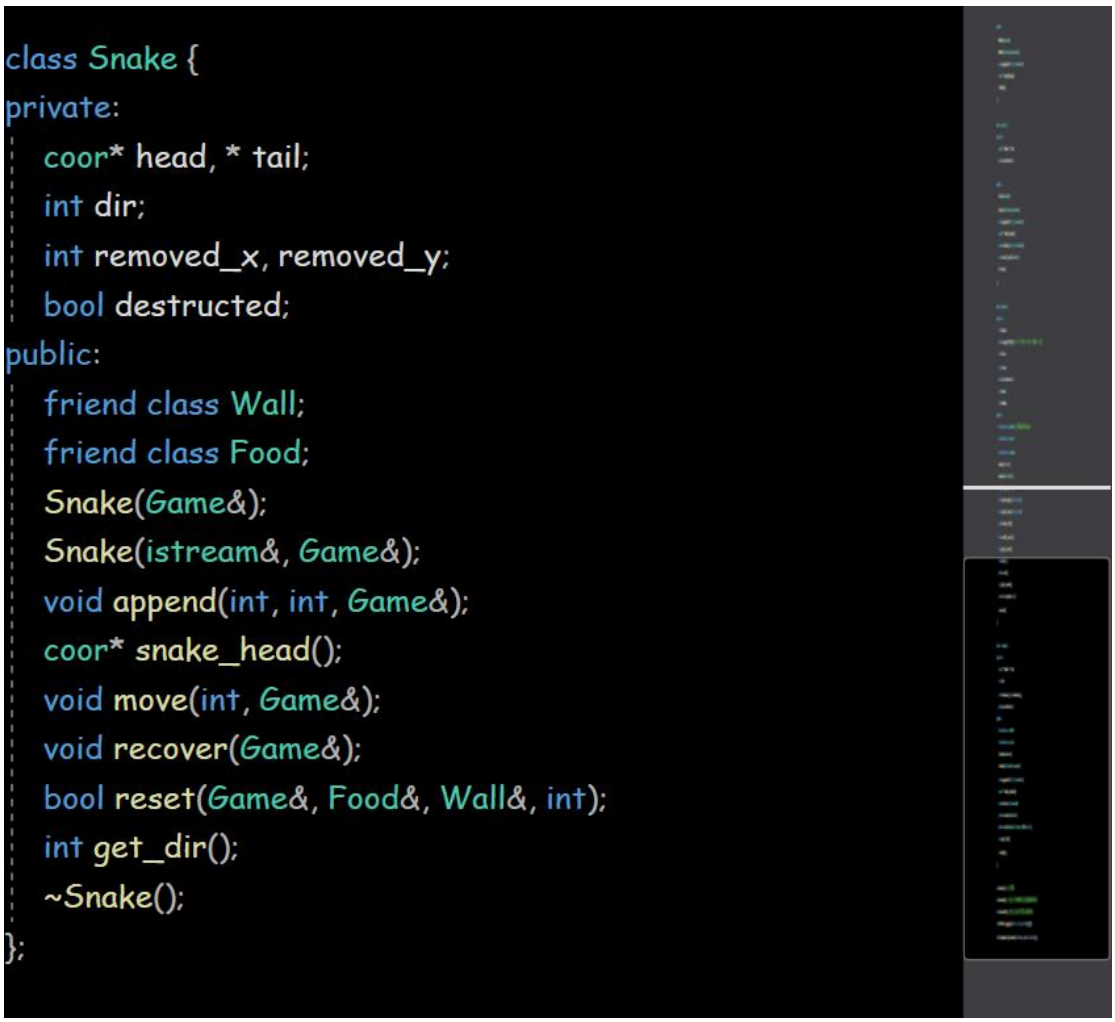
10）bool over();
辅助判断游戏是否结束（此函数中只判断剩余生命数是否大于零和剩余空格数是否足够生成新的食物）。

11）void set_mode(int);
设置游戏模式。

12）int get_mode();
获取游戏模式。

13）~Game();
析构函数，将游戏记录输出至相应的文档中。

（4）Snake 类



```
class Snake {
private:
    coor* head, * tail;
    int dir;
    int removed_x, removed_y;
    bool destructed;
public:
    friend class Wall;
    friend class Food;
    Snake(Game&);
    Snake(istream&, Game&);
    void append(int, int, Game&);
    coor* snake_head();
    void move(int, Game&);
    void recover(Game&);
    bool reset(Game&, Food&, Wall&, int);
    int get_dir();
    ~Snake();
};
```

A. 包含的变量

1）coor* head, * tail;
小蛇头尾的指针（小蛇用链表存储）。

2）int dir;

小蛇的头指向的方向（0 表示右，1 表示左，2 表示上，3 表示下）

3）int removed_x, removed_y;

在小蛇走了一步后，先去掉尾部，在增加头部。removed_x, removed_y 便在去掉尾部之后储存刚才去掉的尾部的坐标, 以便于吃到食物的情况下再加上这个尾部（recover）。

4）bool destructed;
判断对象是否调用过析构函数（储存游戏记录时需要用到）

B. 包含的函数

1）Snake(Game&);

玩家选择【开始游戏】并且选择一个模式、初始化 Game 的对象之后，初始化 Snake 的对象。参数需要带上 Game 是因为形成新的小蛇后需要更新 Game 存储的地图信息。

2）Snake(istream&, Game&);

玩家选择【继续游戏】并且初始化 Game 的对象后，读取相应文档来初始化 Game 的对象。

3）void append(int, int, Game&);

在小蛇的尾部增加 1 的长度，参数包括了增加处的坐标。

4）coor* snake_head();

返回蛇头的指针。

5）void move(int t, Game&);

小蛇朝 t 方向动一个单位。方法是：先去掉尾部，再增加头部（由于小蛇最短长度为 3，所以可以先去尾再加头，中间不会出现蛇链表为空的情况）。具体实现如下：

```
void Snake::move(int t, Game& G) {
    if (t == -1) t = dir;
    else dir = t;

    removed_x = tail->x, removed_y = tail->y;
    G.upd_map(tail->x, tail->y, 0);
    coor* tmp = tail;
    tail = tail->last;
    tail->next = NULL;
    delete tmp;  // 去掉尾部，下面开始增加头部

    int hx = head->x, hy = head->y;
    int dx = hx + tx[t], dy = hy + ty[t];
    coor* cur = new coor;
    cur->x = dx, cur->y = dy;
    cur->next = head, head->last = cur;
    head = cur;
}
```

6）void recover(Game&);

由于移动（增加头部）之后发现小蛇吃到了食物，此时小蛇需要增加长度，即重新加上移动时去掉了的尾部。

```
void Snake::recover(Game& G) {
    append(removed_x, removed_y, G);
}
```

7）bool reset(Game&, Food&, Wall&, int);

生成新的小蛇。在【进阶版】和【高级版】中，小蛇撞死后，原来的蛇身变成墙壁\食物，并且在空地生成新的小蛇。若剩余空间不足以生成新的小蛇，返回值为 0，反之为 1.

8）int get_dir();

返回小蛇头的朝向。

9）~Snake();

析构函数。将当前蛇的信息输出到记录文档中，并且释放动态申请的空间（储存蛇的链表）。

（5）Food 类

```
class Food {
private:
    coor* head, * tail;
    bool destructed;

public:
    Food(Game&);
    Food(istream&, Game&);
    void append(int, int, Game&);
    coor* food_head();
    void remove(int, int, Game&);
    void new_food(Game&);
    ~Food();
};
```

A. 包含的变量

1） coor* head, * tail;

食物头尾的指针（食物用链表存储）。

2） bool destructed;
判断对象是否调用过析构函数（储存游戏记录时需要用到）

B. 包含的函数

1） Food(Game&);

玩家选择【开始游戏】并且选择一个模式、初始化 Game 的对象之后，初始化 Food 的对象。参数需要带上 Game 是因为形成新的食物后需要更新 Game 存储的地图信息。

2） Food(istream&, Game&);

玩家选择【继续游戏】并且初始化 Game 的对象后，读取相应文档来初始化 Food 的对象。

3） void append(int x, int y, Game&);

在坐标为(x,y)处增加一个小面包。

4） coor* food_head();
返回储存面包链表的头指针。

5） void remove(int x, int y, Game&);

去掉(x,y)处的小面包。并且判断面包是否被吃完，若当前已无面包，需要调用 new_food(Game&)来生成新的小面包。

6） void new_food(Game&);
随机生成新的小面包。

7） ~Food();
析构函数。将当前蛇的信息输出到记录文档中，并且释放动态申请的空间（储存蛇的链表）。

（6）Wall 类

```
class Wall {
private:
    coor* head, * tail;
    bool destructed;

public:
    Wall(Game&);
    Wall(istream&, Game&);
    void append(int, int, Game&);
    coor* wall_head();
    ~Wall();
};
```

A. 包含的变量

1） coor* head, * tail;
墙壁头尾的指针（墙壁信息用链表存储）。

2） bool destructed;
判断对象是否调用过析构函数（储存游戏记录时需要用到）

B. 包含的函数

1） Wall(Game&);

玩家选择【开始游戏】并且选择一个模式、初始化 Game 的对象之后，初始化 Wall 的对象。参数需要带上 Game 是因为形成新的墙壁后需要更新 Game 存储的地图信息。

2） Wall(istream&, Game&);

玩家选择【继续游戏】并且初始化 Game 的对象后，读取相应文档来初始化 Wall 的对象。

3） void append(int x, int y, Game&);

在坐标为(x,y)处增加一处墙壁。

4） coor* wall_head();
返回储存墙壁链表的头指针。

5） ~Wall();
析构函数。将当前墙壁的信息输出到记录文档中，并且释放动态申请的空间

（储存墙壁的链表）。

（7）main.cpp

包含了以下三个函数：

1） int Keyboard_monitor(int time_limit, bool flag);

监听键盘活动并返回按下的键对应的 UnicodeChar。

增加 time 和 flag 两个参数是为了：在游戏过程中（flag 设为 1），监听键盘是为了确定小蛇前行的方向（或者按下 Esc 退出游戏），若在 time 的时间内未按下方向键或 Esc 键，则返回-1，小蛇按照自己的朝向走一步。

2） 主函数 int main();

调用 menu()并获取游戏模式。若为【继续游戏】且有为完成的游戏可以继续，则从相应文档读取来初始化 Game, Food, Wall, Snake 的对象；否则直接初始化。初始化完成后调用 play 函数开始游戏。

3） int play(Game& G, Wall& W, Food& F, Snake& S, int mode);

按下空格开始游戏。监听键盘来确定蛇的移动方向，每移动一步判断蛇头走到的位置：空白/食物/墙壁。若为食物，则增加蛇的长度；若为墙壁，则结束游戏或损失一条生命、生成新的小蛇（视游戏模式而定情况）。

返回值反映了游戏是否已经结束，若游戏结束是何原因，以便 main()函数中显示相应的结束提示、输出至记录文档。

二．在实验过程中遇到的困难及解决方法

1．**困难**：第一次写多个源文件的项目，如何组织？

走过的弯路：将每个类单独写在 XXX(名).h 中，互相引用头文件。问题多多 qaq。

解决方法：只写一个.h 的头文件，将所有的类、需要的函数声明在里面。具体的类内函数内容单独写在 XXX(名).cpp 中，这样一来，所有的 cpp 都引用.h 的头文件，main.cpp 包含.h 和所有实现函数的.cpp 头文件即可，非常有序直观。

2. **困难**：小蛇走了一步之后发现吃到了食物需要增加尾部长度，如何知道增加蛇尾处的坐标?

解决方法：多记录两个变量 removed_x, removed_y，为走一步去掉的尾部的坐标，若因吃到食物需要增加长度（恢复原来的尾部），直接在(removed_x, removed_y)处 append 即可。

3. **困难**：int Keyboard_monitor()返回的是按下键的 Unicode，但是在游戏过程中，若玩家不按键，需要在一定时间内结束此函数进程，回到 play 函数，小蛇继续走一步，如何达到这一需求呢?

解决方法：增加两个参数：int time_limit, bool flag。flag 表示当前是否在游戏过程中监听键盘，若是，则 time_limit 时间内如果按下方向键 or 退出键会返回相应的 Unicode，如果限制时间内没有按下相应键，会返回-1。

这样做的意外之喜是，达到了长按加速键可以加速的效果。

4. **困难**：如何实现按下 Esc 后回到主菜单而非退出程序?

解决方法: 在主函数中 while(1){} 其他内容写在循环内, 若监测到按下 Esc 键, 直接 continue 进入新的一次循环即可。

5. **困难**：实现【继续游戏】的功能时，需要在析构函数中将游戏信息写入记录文档，继续游戏时需读取该文档。此过程中需要确定的对象的析构顺序，如何固定不同类的对象的析构顺序呢?

解决方法：显示调用析构函数可以固定顺序。但是系统仍然会自动调用析构函数，为了避免像文档中重复写入信息，对每个类增加一个 bool destructed;的变量，来记录是否已经调用过析构函数。若是第一次调用（即显式调用），则写文档；否则不做处理。

三.—游戏中的小设计与需要注意的~~BUGS~~

1. 小设计：

1）进入主菜单时，页面会由黑到白慢慢变亮。

2）将光标放到按钮上时，按钮内部会黑白反色（变成黑底白字），移开时恢复。

3）结束游戏或中途退出游戏时**程序不会结束，会返回主菜单**（个人认为是值得强调的对玩家非常友好的设计 hhh！）

2. BUGS

1）进入到游戏页面后，要按空格开始游戏后，按 Esc 键才能返回主菜单。不能没开始游戏就返回主菜单。

2）如果选择【继续游戏】或【历史记录】，页面自动迅速返回主菜单，再按一次按钮进入就好了。

3）如果进入游戏，按空格小蛇还不动，关掉 exe 文件再重新运行即可（偶尔出现的不知名 bug QwQ）。

四．心得体会

1. 还是还是注意细节（每次都要强调的!

（1）打开文件后，检查文件是否成功打开

（2）打开的文件都要及时关闭

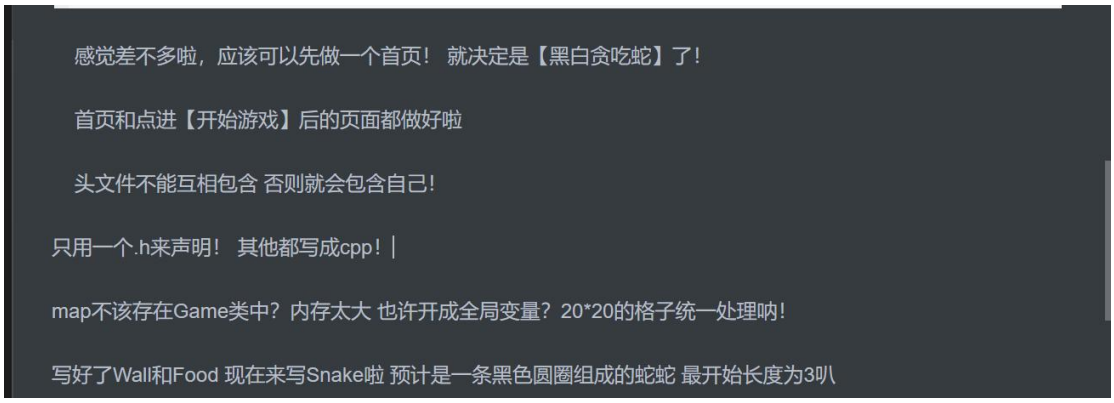
（3）申请的动态内存都要及时处理：

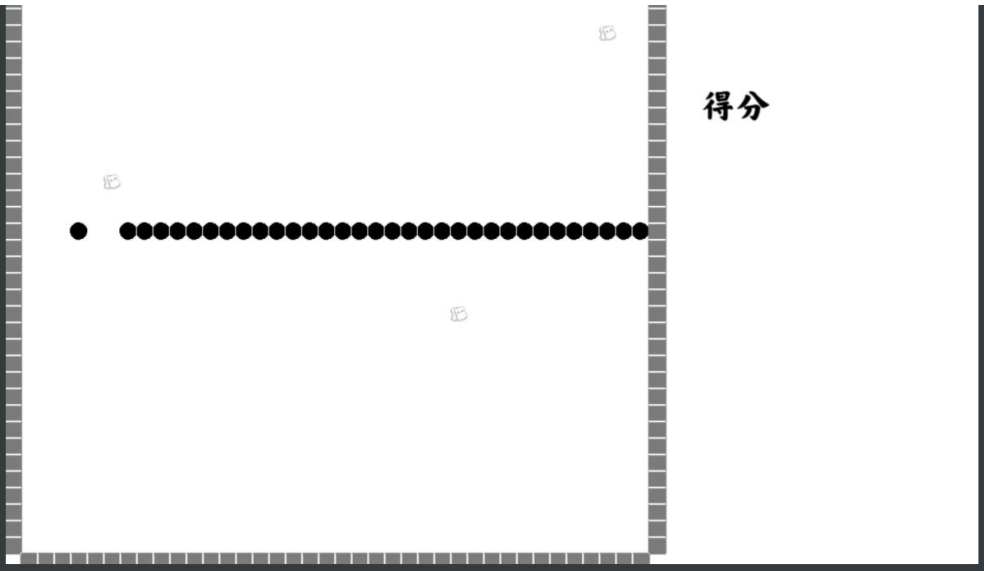
在析构函数中，释放链表。

2. 乍一看觉得很难无从下手时，分解步骤、理清思路。

也许一开始并不能完全想清楚整个框架，可以一步一步来。我最开始做的是 menu()，然后写好每个类的大概框架，形成游戏页面，再让小蛇动起来。可以写

一个单独的作业记录文档，写下当下要干啥，进度到哪了，对理清思路、减少焦虑、脚踏实地非常有帮助！！！（重要的事情打三个感叹号）放一些写贪吃蛇时记录文档的截图~（再读还是很~~有意思~~傻）





似乎尾部的清除不太行呐！

果然是menu里设置的二元光栅的问题！已修复！现在是保持正常长度的小蛇了

已经设置好了键盘控制哇！不过现在就是说 一方面我没按下键盘的时候小蛇不会自己向前走 另一方面是遇到小面包的时候小蛇会被卡住 既吃不掉又走不了

混淆了蛇的方向！

初始化的是蛇的尾巴方向，但是后面又认为dir是蛇的前进方向！

现在已经可以丝滑地行走啦！但是还是不能吃掉小面包呜呜！

现在是！可以吃掉面包！但是吃了面包之后就不动了呜呜qaq

现在吃了面包也可以继续走啦！但是吃完面包之后没有生成新面包了嗷呜

现在是 可以生成新面包了 但是吃掉第二轮的面包之后 程序会崩溃 可能是链表写崩了8

发现是初始化的时候没有初始化head=tail=NULL

还有个不太合理的问题就是蛇不应该能后退的 这样太容易死了8 不过先不管这个问题了

现在来写吃掉一个食物蛇会增长的8！

也可以保持这个先去掉尾部再增加头部的 但是可能需要多一个变量来存已经去掉的尾巴的坐标 以便于 如果吃掉的是食物的话 再重新补上

完成了！以及又发现了一个链表了bug 赶快去改下其他类链表的bug

(emmmm因为错误出在remove函数，而其他类没有remove函数（不对！蛇应该要有remove函数呢！

不对 在Snake::move里面都处理了 不需要单独的remove函数了

那么现在来最后（除了析构以外的）来完善一下入门版！计算出分数 以及死亡后显示相应的人性化提示

显示这个分数搞了好久诶，但是最后都搞好啦！包括显示控制键、”按空格键开始游戏“、游戏结束及其原因的提示、游戏结束后返回首页。

那么现在把【进阶版】和【高级版】来做一下8！

3. 感觉自己只是完成了，但是还有好多不足呜

对于这种代码量比较大、分好几个源文件的我还是写得蛮凌乱的，不像之前单个 cpp 文件的那么有条理了。以及感觉对类的部分还是没有设计好，似乎类与类的相互访问太多了些，破坏封装性太多（哭。一开始比较懵，对类与类之间的交互属于走一步算一步，没有规划好，这种核心的部分还是要多多想清楚嗷!

这个作业就作为一个新的开始吧！后面遇到这样类型的应该会更加熟悉一些、写得更好一些...加油加油！！！

五．源代码

1. classes.h

```
#pragma once
#include<iostream>
#include<fstream>
#include<time.h>

using namespace std;

class Food;
class Game;
class Wall;
class Snake;

struct coor {
	int x, y;
	coor* next, * last;
};

class Wall {
private:
	coor* head, * tail;
	bool destructed;

public:
	Wall(Game&);
	Wall(istream&, Game&);
	void append(int, int, Game&);
```

```
        coor* wall_head();
        ~Wall();
};

class Food {
private:
        coor* head, * tail;
        bool destructed;

public:
        Food(Game&);
        Food(istream&, Game&);
        void append(int, int, Game&);
        coor* food_head();
        void remove(int, int, Game&);
        void new_food(Game&);
        ~Food();
};

class Game {
private:
        int mode;
        char map[40][40]; //0：空白 1：墙 2：食物 3：蛇
        int lives;
        int score;
        bool destructed;
        int maxs;
        int blank;
public:
        friend class Wall; //记得声明友元类!
        friend class Food;
        friend class Snake;
        Game(int md);
        Game(istream&);
        int query(int, int);
        void upd_map(int, int, int);
        void upd_view(int, int, int);
        void show_UI();
        void add_score();
        int get_score();
        void die();
        bool over();
        int get_mode();
        void set_mode(int);
```

```
    ~Game();
};

class Snake {
private:
    coor* head, * tail;
    int dir;
    int removed_x, removed_y;
    bool destructed;
public:
    friend class Wall;
    friend class Food;
    Snake(Game&);
    Snake(istream&, Game&);
    void append(int, int, Game&);
    coor* snake_head();
    void move(int, Game&);
    void recover(Game&);
    bool reset(Game&, Food&, Wall&, int);
    int get_dir();
    ~Snake();
};

int menu(); //主页面
int menu1(); //进入【开始游戏】后选择游戏模式
void menu2(); //进入【历史记录】后的显示
int button_judge(int x, int y, int r[3][4]);
int Keyboard_monitor(int time_limit, bool flag);
```

2. menu.cpp

```
#define _CRT_SECURE_NO_WARNINGS
#include"classes.h"
#include<iostream>
#include<graphics.h>
#include<conio.h>
#include<fstream>

using namespace std;

int menu(); //主页面
int menu1(); //进入【开始游戏】后选择游戏模式
void menu2(); //进入【历史记录】后的显示
int button_judge(int x, int y, int r[3][4]);
```

```
int button_judge(int x, int y, int r[3][4])
{
	if (x > r[0][0] && x<r[0][2] && y>r[0][1] && y < r[0][3]) return 1;
	if (x > r[1][0] && x<r[1][2] && y>r[1][1] && y < r[1][3]) return 2;
	if (x > r[2][0] && x<r[2][2] && y>r[2][1] && y < r[2][3]) return 3;
	return 0;
}

int menu() {
	initgraph(1200, 800);
	for (int i = 0;i < 256;i += 5)
	{
		setbkcolor(RGB(i, i, i));//设置背景色，原来默认黑色
		cleardevice();//清屏（取决于背景色）
		Sleep(15);//延时 15ms
	}
	IMAGE title;
	loadimage(&title, _T("./image./1.jpg"), 700, 300);
	putimage(600 - 350, 0, &title);

	IMAGE pic;
	loadimage(&pic, _T("./image./dtt.jpg"), 140*1.2, 40*1.2);
	putimage(1000, 745, &pic);

	int butt[3][4] = { {380,280,820,430},{380,440,820,590},{380,600,820,750} };
	int r[3][4];
	for (int i = 0;i < 3;i++) {
		r[i][0] = butt[i][0] + 45;
		r[i][1] = butt[i][1] + 20;
		r[i][2] = butt[i][2] - 45;
		r[i][3] = butt[i][3] - 20;
	}
	RECT R1 = { butt[0][0],butt[0][1],butt[0][2],butt[0][3] }; //矩形指针
	RECT R2 = { butt[1][0],butt[1][1],butt[1][2],butt[1][3] };
	RECT R3 = { butt[2][0],butt[2][1],butt[2][2],butt[2][3] };
	LOGFONT f; //字体样式指针
	gettextstyle(&f); //获取字体样式 （？没太懂
	_tcscpy(f.lfFaceName, _T("楷体")); //设置字体为宋体
	//_tcscpy(f.lfFaceName, _T("bold"));
	f.lfHeight = 40, f.lfWeight = 133; //设置字体大小
	f.lfQuality = ANTIALIASED_QUALITY; //设置输出效果为抗锯齿
	settextstyle(&f); //设置字体样式
	settextcolor(BLACK);
```

```
    drawtext(_T("开始游戏"), &R1, DT_CENTER | DT_VCENTER | DT_SINGLELINE);//在矩形区域 R1 内输入文字，水平居中，垂直居中，单行显示
    drawtext(_T("继续游戏"), &R2, DT_CENTER | DT_VCENTER | DT_SINGLELINE);//在矩形区域 R2 内输入文字，水平居中，垂直居中，单行显示
    drawtext(_T("历史记录"), &R3, DT_CENTER | DT_VCENTER | DT_SINGLELINE);//在矩形区域 R3 内输入文字，水平居中，垂直居中，单行显示
    setlinecolor(BLACK);
    setlinestyle(PS_SOLID, 2);
    rectangle(r[0][0], r[0][1], r[0][2], r[0][3]);
    rectangle(r[1][0], r[1][1], r[1][2], r[1][3]);
    rectangle(r[2][0], r[2][1], r[2][2], r[2][3]);
    MOUSEMSG m; //鼠标指针
    int event = 0, jud = 0;
    while (1) {
        m = GetMouseMsg(); //获取一条鼠标消息
        switch (m.uMsg) {
        case WM_MOUSEMOVE:
            setrop2(R2_XORPEN);
            setlinecolor(LIGHTCYAN);//亮青色
            setlinestyle(PS_SOLID, 3);
            setfillcolor(WHITE);
            jud = button_judge(m.x, m.y, r);
            if (jud != 0) {
                if (event != jud) {
                    event = jud;
                    fillrectangle(r[event - 1][0], r[event - 1][1], r[event - 1][2], r[event - 1][3]);
                }
            }
            else if (event != 0) {//上次触发的按钮未被修正为原来的颜色
                fillrectangle(r[event - 1][0], r[event - 1][1], r[event - 1][2], r[event - 1][3]);//两次同或为原来颜色
                event = 0;
            }
            break;
        case WM_LBUTTONDOWN:
            setrop2(R2_NOTXORPEN);//二元光栅——NOT(屏幕颜色 XOR 当前颜色)
            for (int i = 0;i <= 10;i++)
            {
                setlinecolor(RGB(25 * i, 25 * i, 25 * i));//设置圆颜色
                circle(m.x, m.y, 2 * i);
                Sleep(30);//停顿 30ms
                circle(m.x, m.y, 2 * i);//抹去刚刚画的圆
            }
```

```
                jud = button_judge(m.x, m.y, r);
                if (jud == 1) {
                    cleardevice();
                    return menu1();
                }
                else if (jud == 2) return 4;
                else if (jud == 3) {
                    cleardevice();
                    menu2();
                    return 0;
                }
                break;
                FlushMouseMsgBuffer();//清空鼠标消息缓存区
            }
        }
        setrop2(R2_COPYPEN);
}
int menu1() {
        int butt[3][4] = { {0,0,400,250},{0,250,400,500},{0,500,400,750} };
        int r[3][4];
        for (int i = 0;i < 3;i++) {
            r[i][0] = butt[i][0] + 65;
            r[i][1] = butt[i][1] + 60;
            r[i][2] = butt[i][2] - 65;
            r[i][3] = butt[i][3] - 60;
        }
        RECT R1 = { butt[0][0],butt[0][1],butt[0][2],butt[0][3] }; //矩形指针
        RECT R2 = { butt[1][0],butt[1][1],butt[1][2],butt[1][3] };
        RECT R3 = { butt[2][0],butt[2][1],butt[2][2],butt[2][3] };
        LOGFONT f; //字体样式指针
        gettextstyle(&f); //获取字体样式 （? 没太懂
        //wcscpy_s(f.lfFaceName, _T("楷体"));
        _tcscpy(f.lfFaceName, _T("楷体")); //设置字体为楷体
        //_tcscpy(f.lfFaceName, _T("bold"));
        f.lfHeight = 40, f.lfWeight = 133; //设置字体大小
        f.lfQuality = ANTIALIASED_QUALITY; //设置输出效果为抗锯齿
        settextstyle(&f); //设置字体样式
        settextcolor(BLACK);
        drawtext(_T("入门版"), &R1, DT_CENTER | DT_VCENTER | DT_SINGLELINE);//在矩形区域 R1 内输入文字，水平居中，垂直居中，单行显示
        drawtext(_T("进阶版"), &R2, DT_CENTER | DT_VCENTER | DT_SINGLELINE);//在矩形区域 R2 内输入文字，水平居中，垂直居中，单行显示
        drawtext(_T("高级版"), &R3, DT_CENTER | DT_VCENTER | DT_SINGLELINE);//在矩形区域 R3 内输入文字，水平居中，垂直居中，单行显示
```

```
    setlinecolor(BLACK);
    setlinestyle(PS_SOLID, 2);
    rectangle(r[0][0], r[0][1], r[0][2], r[0][3]);
    rectangle(r[1][0], r[1][1], r[1][2], r[1][3]);
    rectangle(r[2][0], r[2][1], r[2][2], r[2][3]);
    IMAGE pic, pic1;
    loadimage(&pic, _T("./image./2.jpg"), 800, 250);
    putimage(400 - 32, 0, &pic);

    loadimage(&pic, _T("./image./3.jpg"), 800, 250);
    putimage(400 - 32, 245, &pic);

    loadimage(&pic, _T("./image./4.jpg"), 800, 250);
    putimage(400 - 32, 495, &pic);

    //loadimage(&pic1, _T("./image./Esc.jpg"), 700*0.4, 200*0.4);
    //putimage(900, 750, &pic1);



    MOUSEMSG m; //鼠标指针
    int event = 0, jud = 0;
    while (1) {
        m = GetMouseMsg(); //获取一条鼠标消息
        switch (m.uMsg) {
        case WM_MOUSEMOVE:
            setrop2(R2_XORPEN);
            setlinecolor(LIGHTCYAN);//亮青色
            setlinestyle(PS_SOLID, 3);
            setfillcolor(WHITE);
            jud = button_judge(m.x, m.y, r);
            if (jud != 0) {
                if (event != jud) {
                    event = jud;
                    fillrectangle(r[event - 1][0], r[event - 1][1], r[event - 1][2], r[event -
1][3]);
                }
            }
            else if (event != 0) {//上次触发的按钮未被修正为原来的颜色
                fillrectangle(r[event - 1][0], r[event - 1][1], r[event - 1][2], r[event - 1][3]);//
两次同或为原来颜色
                event = 0;
            }
            break;
```

```
            case WM_LBUTTONDOWN:
                setrop2(R2_NOTXORPEN);//二元光栅——NOT(屏幕颜色 XOR 当前颜色)
                for (int i = 0;i <= 10;i++)
                {
                    setlinecolor(RGB(25 * i, 25 * i, 25 * i));//设置圆颜色
                    circle(m.x, m.y, 2 * i);
                    Sleep(30);//停顿 30ms
                    circle(m.x, m.y, 2 * i);//抹去刚刚画的圆
                }
                jud = button_judge(m.x, m.y, r);
                if (jud != 0) return jud;
                break;
                FlushMouseMsgBuffer();//清空鼠标消息缓存区
            }
        }
    }
    void menu2() {
        IMAGE pic1;
        loadimage(&pic1, _T("./image./Esc.jpg"), 700*0.4, 200*0.4);
        putimage(900, 750, &pic1);

        ifstream fl("./file./game_record.txt", ios::app);
        int ry[2] = { 0,50 }, rx[2] = { 0,100 };
        while (!fl.eof()) {
            int type, score;
            if (!(fl >> type >> score))break;
            LOGFONT f; //字体样式指针
            gettextstyle(&f); //获取字体样式 （？没太懂
            _tcscpy(f.lfFaceName, _T("楷体")); //设置字体为宋体
            //_tcscpy(f.lfFaceName, _T("bold"));
            //f.lfHeight = 40, f.lfWeight = 133; //设置字体大小
            //f.lfQuality = ANTIALIASED_QUALITY; //设置输出效果为抗锯齿
            settextstyle(&f); //设置字体样式
            settextcolor(BLACK);
            char str[15] = "", number[15] = { '0','1','2','3','4','5','6','7','8','9','\0' };

            int cur = 9, tmp = score;
            while (cur != 0 && (int)(score / pow(10, cur)) == 0)cur--;
            for (int i = cur;i >= 0;i--) {
                strncat(str, number + (int)(tmp / (int)pow(10, i)), 1);
                tmp %= (int)pow(10, i);
            }

            int num = MultiByteToWideChar(0, 0, str, -1, NULL, 0);
```

```
        wchar_t* wide = new wchar_t[num];
        MultiByteToWideChar(0, 0, str, -1, wide, num);
        /*num 获得长字节所需的空间
        MultiByteToWideChar()表示将 s 中的字符传递到 ps 指向的内存中。-1 表示传输至 s 中的'\0'处，num 表示传递的字节个数。*/

        RECT R1 = { rx[0], ry[0], rx[1], ry[1] };
        RECT R2 = { rx[0] + 100, ry[0], rx[1] + 200, ry[1] };
        RECT R3 = { rx[0] + 300, ry[0], rx[1] + 300, ry[1] };
        RECT R4 = { rx[0] + 400, ry[0], rx[1] + 400, ry[1] };
        drawtext(_T("版本："), &R1, DT_LEFT | DT_VCENTER | DT_SINGLELINE);
        if (type == 1)drawtext(_T("入门版"), &R2, DT_LEFT | DT_VCENTER | DT_SINGLELINE);
        if (type == 2)drawtext(_T("进阶版"), &R2, DT_LEFT | DT_VCENTER | DT_SINGLELINE);
        if (type == 3)drawtext(_T("高级版"), &R2, DT_LEFT | DT_VCENTER | DT_SINGLELINE);
        drawtext(_T("得分："), &R3, DT_LEFT | DT_VCENTER | DT_SINGLELINE);
        drawtext((LPCTSTR)wide, &R4, DT_LEFT | DT_VCENTER | DT_SINGLELINE);
        ry[0] += 52   , ry[1] += 52;
    }

    while (Keyboard_monitor(0, 0) != 27); //按 Esc 退出页面
}
```

3. main.cpp

```
#define _CRT_SECURE_NO_WARNINGS
#include"classes.h"
#include<iostream>
#include<graphics.h>
#include<conio.h>
#include<windows.h>

using namespace std;

int Keyboard_monitor(int time_limit, bool flag) {
    time_t beg = clock(), cur;
    char black[] = "                    ";      // 为输出填加的尾部空白，用于覆盖输出
    INPUT_RECORD inRec = { 0 };                 // 输入信息结构体，记录输入信息
    DWORD res;
    HANDLE hInput = GetStdHandle(STD_INPUT_HANDLE);          // 获取控制台标准输入句柄
```

```
    HANDLE hOutput = GetStdHandle(STD_OUTPUT_HANDLE);        // 获取控制台标准输出句柄

    // 开启控制台窗口鼠标输入
    SetConsoleMode(hInput, ENABLE_EXTENDED_FLAGS | ENABLE_WINDOW_INPUT | ENABLE_MOUSE_INPUT);
    while (1) {
        cur = clock();
        //cout << (cur - beg) * 1.0 / CLOCKS_PER_SEC * 1000 << endl;
        if (flag && (cur - beg) * 1.0 / CLOCKS_PER_SEC * 1000 >= time_limit) break;

        // 获取当前输入信息，采用 WaitForSingleObject 来防止函数阻塞（异步监听）
        if (WaitForSingleObject(hInput, 0) == WAIT_OBJECT_0)
            ReadConsoleInputA(hInput, &inRec, 1, &res);

        // 将光标移动到左上角，覆盖输出
        COORD zeroPos = { 0,0 };
        SetConsoleCursorPosition(hOutput, zeroPos);

        if (flag && inRec.EventType == KEY_EVENT) {
            int num = inRec.Event.KeyEvent.uChar.UnicodeChar;
            if (num == 97 || num == 100 || num == 115 || num == 119 || num==27) return num;
        }
        if (flag) continue;
        // 监听事件状态
        switch (inRec.EventType) {
        case KEY_EVENT:
            // 键盘事件
            // dwControlKeyState 会记录键盘控制键状态，例如大小写是否开启等
            // wVirtualKeyCode wVirtualScanCode 是最终键盘按键的可以使用的组合变量
            return inRec.Event.KeyEvent.uChar.UnicodeChar;
            break;
        }
    }
    return -1;
}
int play(Game& G, Wall& W, Food& F, Snake& S, int mode) {

    int keyy = Keyboard_monitor(0, 0);
    while (keyy != 32) keyy = Keyboard_monitor(0, 0); //按空格开始游戏

    int is_over = 0;
```

```
if (mode == 1) {
    while (is_over == 0) {
        int keyb = Keyboard_monitor(150, 1);
        int dir = S.get_dir();
        if (dir == 0 && keyb == 97)continue;
        if (dir == 1 && keyb == 100)continue;
        if (dir == 2 && keyb == 119)continue;
        if (dir == 3 && keyb == 115)continue;
        switch (keyb) {
        case -1:
            S.move(-1, G);
            break;
        case 97:
            S.move(1, G);
            break;
        case 100:
            S.move(0, G);
            break;
        case 115:
            S.move(2, G);
            break;
        case 119:
            S.move(3, G);
            break;
        case 27:
            return is_over;
            break;
        }
        coor* sh = S.snake_head();
        int type = G.query(sh->x, sh->y);

        if (type == 0) { //走到空白处，无事发生
            G.upd_map(sh->x, sh->y, 3);
        }
        else if (type == 1) { //撞到墙，游戏结束
            is_over = 1;
            //break;
        }
        else if (type == 2) { //吃到食物，长度加 1，分数加 5
            //cout << "bread！ " << endl;
            F.remove(sh->x, sh->y, G); //在 remove 里擦白
            S.recover(G);
            G.upd_map(sh->x, sh->y, 3);
            G.add_score();
```

```
            }
            else if (type == 3) { //撞到自己，游戏结束
                is_over = 2;
            }
            Sleep(50);
        }
        if (is_over == 1) {
            IMAGE pic;
            loadimage(&pic, _T("./image/over1.jpg"), 300, 150);
            putimage(800 + 50, 350, &pic);
            return 1;
        }
        else if (is_over == 2) {
            IMAGE pic;
            loadimage(&pic, _T("./image/over2.jpg"), 300, 150);
            putimage(800 + 50, 350, &pic);
            return 2;
        }
    }
    else if (mode == 2) {
        bool ade = 1;
        while (!G.over() && ade) {
            int keyb = Keyboard_monitor(150, 1);
            int dir = S.get_dir();
            if (dir == 0 && keyb == 97)continue;
            if (dir == 1 && keyb == 100)continue;
            if (dir == 2 && keyb == 119)continue;
            if (dir == 3 && keyb == 115)continue;
            switch (keyb) {
            case -1:
                S.move(-1, G);
                break;
            case 97:
                S.move(1, G);
                break;
            case 100:
                S.move(0, G);
                break;
            case 115:
                S.move(2, G);
                break;
            case 119:
                S.move(3, G);
                break;
```

```
            case 27:
                return is_over;
                break;
            }
            coor* sh = S.snake_head();
            int type = G.query(sh->x, sh->y);

            if (type == 0) { //走到空白处，无事发生
                G.upd_map(sh->x, sh->y, 3);
            }
            else if (type == 1) { //撞到墙，游戏结束
                ade = S.reset(G, F, W, 1);//将原来的蛇变成新的墙壁，再产生新的蛇
                //is_over = 1;
                //break;
            }
            else if (type == 2) { //吃到食物，长度加 1，分数加 5
                //cout << "bread！ " << endl;
                F.remove(sh->x, sh->y, G); //在 remove 里擦白
                S.recover(G);
                G.upd_map(sh->x, sh->y, 3);
                G.add_score();
            }
            else if (type == 3) { //撞到自己，游戏结束
                ade = S.reset(G, F, W, 1);
                //is_over = 2;
            }
            Sleep(50);
        }
        if (is_over == 1) {
            IMAGE pic;
            loadimage(&pic, _T("./image/over1.jpg"), 300, 150);
            putimage(800 + 50, 350, &pic);
            //return 1;
        }
        else if (is_over == 2) {
            IMAGE pic;
            loadimage(&pic, _T("./image/over2.jpg"), 300, 150);
            putimage(800 + 50, 350, &pic);
           // return 2;
        }
    }
    else if (mode == 3) {
        bool ade = 1;
        while (!G.over() && ade) {
```

```
int keyb = Keyboard_monitor(150, 1);
int dir = S.get_dir();
if (dir == 0 && keyb == 97)continue;
if (dir == 1 && keyb == 100)continue;
if (dir == 2 && keyb == 119)continue;
if (dir == 3 && keyb == 115)continue;
switch (keyb) {
case -1:
	S.move(-1, G);
	break;
case 97:
	S.move(1, G);
	break;
case 100:
	S.move(0, G);
	break;
case 115:
	S.move(2, G);
	break;
case 119:
	S.move(3, G);
	break;
case 27:
	return is_over;
	break;
}
coor* sh = S.snake_head();
int type = G.query(sh->x, sh->y);

if (type == 0) { //走到空白处，无事发生
	G.upd_map(sh->x, sh->y, 3);
}
else if (type == 1) { //撞到墙，生命 j--
	ade = S.reset(G, F, W, 2);//将原来的蛇变成新的食物，再产生新的蛇
	G.die();
	//is_over = 1;
	//break;
}
else if (type == 2) { //吃到食物，长度加 1，分数加 5
	//cout << "bread！ " << endl;
	F.remove(sh->x, sh->y, G); //在 remove 里擦白
	S.recover(G);
	G.upd_map(sh->x, sh->y, 3);
	G.add_score();
```

```
                }
                else if (type == 3) { //撞到自己，生命--
                    ade = S.reset(G, F, W, 2);
                    G.die();
                    //is_over = 2;
                }
                Sleep(50);
            }
            is_over = 3;
        }
        return is_over;
}
int main() {
        while (1) {
            srand(time(0));
            int mode = menu();
            while (mode == 0) {
                mode = menu();
            }
            cleardevice();
            if (mode == 4) {

                ifstream fl("./file./record.txt");
                if (!fl.is_open()) {
                    cout << "文件打开失败" << endl;
                    exit(1);
                }
                int is_over;
                fl >> is_over;
                fl.close();
                if (is_over) {
                    cleardevice();
                    IMAGE pic1, pic2, pic3;
                    loadimage(&pic1, _T("./image/error.jpg"), 700, 200);
                    putimage(250, 120, &pic1);
                    loadimage(&pic2, _T("./image/nomore.jpg"), 700, 200);
                    putimage(250, 375, &pic2);
                    loadimage(&pic3, _T("./image/Esc.jpg"), 700 * 0.86, 200 * 0.86);
                    putimage(300, 580, &pic3);
                    while (Keyboard_monitor(0, 0) != 27); //按 Esc 退出

                }
                else {
                    Game G(fl);
```

```
                Food F(fl, G);
                Wall W(fl, G);
                Snake S(fl, G);
                G.show_UI();
                int is_ov = play(G, W, F, S, G.get_mode());
                ofstream fl("./file./record.txt");
                if (!fl.is_open()) {
                    cout << "文件打开失败" << endl;
                    exit(1);
                }
                fl << is_ov << endl;
                fl.close();
                G.~Game();
                F.~Food();
                W.~Wall();
                S.~Snake();
                while (Keyboard_monitor(0, 0) != 27);
            }
        }
        else {
            Game G(mode);
            Food F(G);
            Wall W(G);
            Snake S(G);
            G.show_UI();
            int is_ov = play(G, W, F, S, mode);
            ofstream fl("./file./record.txt");
            if (!fl.is_open()) {
                cout << "文件打开失败" << endl;
                exit(1);
            }
            fl << is_ov << endl;
            fl.close();
            G.~Game();
            F.~Food();
            W.~Wall();
            //IMAGE pic1;
            //loadimage(&pic1, _T("./image/error.jpg"), 700, 200);
            //putimage(250, 120, &pic1);
            S.~Snake();
            //Sleep(2000);
            while (Keyboard_monitor(0, 0) != 27);
        }
    }
```

```
	return 0;
}
```

4. Game.cpp

```
#define _CRT_SECURE_NO_WARNINGS
#include"classes.h"
#include<iostream>
#include<graphics.h>
#include<fstream>

using namespace std;

Game::Game(int md) {
	mode = md;
	lives = 6;
	score = -5;
	destructed = 0;
	blank = 1600;
	for (int i = 0;i < 40;i++)
		for (int j = 0;j < 40;j++)
			map[i][j] = 0;

	ifstream ifl;
	if (mode == 1) ifl.open("./file./max1.txt");
	if (mode == 2) ifl.open("./file./max2.txt");
	if (mode == 3) ifl.open("./file./max3.txt");
	ifl >> maxs;
	ifl.close();
}
int Game::query(int x, int y) {
	return map[x][y];
}
void Game::upd_map(int x, int y, int type) {
	if (map[x][y] == 0 && type != 0) --blank;
	if (map[x][y] != 0 && type == 0) ++blank;
	map[x][y] = type;
	upd_view(x, y, type);
}
void Game::upd_view(int x, int y, int type) {
	setrop2(R2_COPYPEN);
	//if(type==0)cout << "x=" << x << "    y=" << y << " type=" << type << endl;
	IMAGE pic;
	switch(type) {
```

```
        case 0://空白
            setlinecolor(WHITE);
            setfillcolor(WHITE);
            fillrectangle(x * 20, y * 20, x * 20 + 19, y * 20 + 19);
                break;
        case 1: //墙
            setlinecolor(RGB(123, 123, 123));
            setfillcolor(RGB(123, 123, 123));
            fillrectangle(x * 20, y * 20, x * 20 + 19, y * 20 + 19);
                break;
        case 2: //食物
            loadimage(&pic, _T("./image/bread.jpg"), 20, 20);
            putimage(x * 20, y * 20, &pic);
            break;
        case 3: //蛇
            setlinecolor(BLACK);
            setfillcolor(BLACK);
            fillcircle(x * 20 + 9.5, y * 20 + 9.5, 9.5);
            break;



        }
}
void Game::show_UI() {
        if (mode == 1) {
            IMAGE pic1, pic2, pic3, pic4, pic5, pic6, pic7;
            loadimage(&pic1, _T("./image/UI1.jpg"), 250, 125);
            putimage(800 + 60, 0, &pic1);
            loadimage(&pic2, _T("./image/UI4.jpg"), 150, 75); //如果这里还用 pic 会有问题
            putimage(800 + 10, 200, &pic2);
            loadimage(&pic3, _T("./image/ctrl.jpg"), 150, 150);
            putimage(800 + 30, 600, &pic3);
            loadimage(&pic4, _T("./image/begin.jpg"), 196, 56);
            putimage(800 + 30, 737, &pic4);
            loadimage(&pic5, _T("./image/Esc.jpg"), 196*1.06, 56*1.06);
            putimage(800 + 40, 770, &pic5);
            loadimage(&pic6, _T("./image/speed.jpg"), 196 * 1.2, 56 * 1.2);
            putimage(800 + 195, 668, &pic6);
            //loadimage(&pic7, _T("./image/time.jpg"), 196 * 0.8, 56 * 0.8);
            //putimage(800 + 48, 265, &pic7);
            add_score();
        }
        else if (mode == 2) {
            IMAGE pic1, pic2, pic3, pic4, pic5, pic6, pic7;
```

```
            loadimage(&pic1, _T("./image/UI2.jpg"), 250, 125);
            putimage(800 + 60, 0, &pic1);
            loadimage(&pic2, _T("./image/UI4.jpg"), 150, 75); //如果这里还用 pic 会有问题
            putimage(800 + 10, 200, &pic2);
            loadimage(&pic3, _T("./image/ctrl.jpg"), 150, 150);
            putimage(800 + 30, 600, &pic3);
            loadimage(&pic4, _T("./image/begin.jpg"), 196, 56);
            putimage(800 + 30, 737, &pic4);
            loadimage(&pic5, _T("./image/Esc.jpg"), 196 * 1.06, 56 * 1.06);
            putimage(800 + 40, 770, &pic5);
            loadimage(&pic6, _T("./image/speed.jpg"), 196 * 1.2, 56 * 1.2);
            putimage(800 + 195, 668, &pic6);
            //loadimage(&pic7, _T("./image/time.jpg"), 196 * 0.8, 56 * 0.8);
            //putimage(800 + 48, 265, &pic7);
            add_score();
        }
        else if (mode = 3) { //加上生命值的显示喔!
            IMAGE pic1, pic2, pic3, pic4, pic5, pic6, pic7, pic8;
            loadimage(&pic1, _T("./image/UI3.jpg"), 250, 125);
            putimage(800 + 60, 0, &pic1);
            loadimage(&pic2, _T("./image/UI4.jpg"), 150, 75); //如果这里还用 pic 会有问题
            putimage(800 + 10, 200, &pic2);
            loadimage(&pic3, _T("./image/ctrl.jpg"), 150, 150);
            putimage(800 + 30, 600, &pic3);
            loadimage(&pic4, _T("./image/begin.jpg"), 196, 56);
            putimage(800 + 30, 737, &pic4);
            loadimage(&pic5, _T("./image/left.jpg"), 196 * 1.17, 56 * 1.17);
            putimage(800 + 40, 265, &pic5);
            loadimage(&pic6, _T("./image/Esc.jpg"), 196 * 1.06, 56 * 1.06);
            putimage(800 + 40, 770, &pic6);
            loadimage(&pic7, _T("./image/speed.jpg"), 196 * 1.2, 56 * 1.2);
            putimage(800 + 195, 668, &pic7);
            //loadimage(&pic8, _T("./image/time.jpg"), 196 * 0.8, 56 * 0.8);
            //putimage(800 + 48, 322, &pic8);
            add_score();
            die();
        }
    }
    void Game::add_score() {
        score += 5;
        LOGFONT f; //字体样式指针
        gettextstyle(&f); //获取字体样式 （? 没太懂
        _tcscpy(f.lfFaceName, _T("Comic Sans MS")); //设置字体
        settextstyle(&f); //设置字体样式
```

```
    settextcolor(BLACK);
    f.lfHeight = 10, f.lfWeight = 10; //设置字体大小

    char str[15] = "", number[15] = { '0','1','2','3','4','5','6','7','8','9','\0' };
    int cur = 9, tmp=score;
    while (cur != 0 && (int)(score / pow(10, cur)) == 0)cur--;
    for (int i = cur;i >= 0;i--) {
        strncat(str, number + (int)(tmp / (int)pow(10, i)), 1);
        tmp %= (int)pow(10, i);
    }

    int num = MultiByteToWideChar(0, 0, str, -1, NULL, 0);
    wchar_t* wide = new wchar_t[num];
    MultiByteToWideChar(0, 0, str, -1, wide, num);
    /*num 获得长字节所需的空间
    MultiByteToWideChar()表示将 s 中的字符传递到 ps 指向的内存中。-1 表示传输至 s
中的'\0'处，num 表示传递的字节个数。*/
    RECT R = { 938,200,1101,275 };
    drawtext((LPCTSTR)wide, &R, DT_LEFT | DT_VCENTER | DT_SINGLELINE);

    if (maxs < score) maxs = score;
    IMAGE pic;
    loadimage(&pic, _T("./image/max.jpg"), 196 * 1.25, 56 * 1.25);
    putimage(800 + 45, 380, &pic);

    strcpy(str, "");
    cur = 9, tmp = maxs;
    while (cur != 0 && (int)(tmp / pow(10, cur)) == 0)cur--;
    for (int i = cur;i >= 0;i--) {
        strncat(str, number + (int)(tmp / (int)pow(10, i)), 1);
        tmp %= (int)pow(10, i);
    }

    num = MultiByteToWideChar(0, 0, str, -1, NULL, 0);
    wide = new wchar_t[num];
    MultiByteToWideChar(0, 0, str, -1, wide, num);
    /*num 获得长字节所需的空间
    MultiByteToWideChar()表示将 s 中的字符传递到 ps 指向的内存中。-1 表示传输至 s
中的'\0'处，num 表示传递的字节个数。*/

    RECT R1 = { 835,450,1100,490 };
    drawtext((LPCTSTR)wide, &R1, DT_CENTER | DT_VCENTER | DT_SINGLELINE);

}
```

```
int Game::get_score() {
	return score;
}
void Game::die() {
	lives--;
	char number[10] = { '0','1','2','3','4','5','\0' };
	char ch = number[lives];

	RECT R = { 1080,260.5,1080+19*1.17,260.5 + 56 * 1.17 };
	LOGFONT f; //字体样式指针
	gettextstyle(&f); //获取字体样式 （？没太懂
	_tcscpy(f.lfFaceName, _T("Comic Sans MS")); //设置字体
	settextstyle(&f); //设置字体样式
	settextcolor(BLACK);
	f.lfHeight = 10, f.lfWeight = 10; //设置字体大小

	drawtext(ch, &R, DT_LEFT | DT_VCENTER | DT_SINGLELINE);

}
bool Game::over() {
	if (lives == 0 || blank < 5) return 1;
	return 0;
}
int Game::get_mode() {
	return mode;
}
void Game::set_mode(int x) {
	mode = x;
}
Game::~Game() {
	if (!destructed) {
		ofstream fl("./file./record.txt", ios::app);
		if (!fl.is_open()) {
			cout << "文件打开失败" << endl;
			exit(1);
		}
		fl << mode << " " << lives << " " << score << endl;
		fl.close();
		fl.open("./file./game_record.txt", ios::app);
		if (!fl.is_open()) {
			cout << "文件打开失败" << endl;
			exit(1);
		}
		fl << mode << " " << score << endl;
```

```
            fl.close();
            ifstream ifl;
            if (mode == 1) ifl.open("./file./max1.txt");
            if (mode == 2) ifl.open("./file./max2.txt");
            if (mode == 3) ifl.open("./file./max3.txt");
            if (!ifl.is_open()) {
                cout << "文件打开失败" << endl;
                exit(1);
            }
            int cur;
            ifl >> cur;
            ifl.close();
            if (cur < score){
                if (mode == 1) fl.open("./file./max1.txt");
                if (mode == 2) fl.open("./file./max2.txt");
                if (mode == 3) fl.open("./file./max3.txt");
                if (!fl.is_open()) {
                    cout << "文件打开失败" << endl;
                    exit(1);
                }
                fl << score;
            }
            fl.close();
            //cout << "maxs" << maxs << endl << "Game over" << endl;

        }
        destructed = 1;
    }
    Game::Game(istream& in) {
        in >> mode >> lives >> score;
        lives += 1, score -= 5;
        destructed = 0;
        blank = 1600;
        for (int i = 0;i < 40;i++)
            for (int j = 0;j < 40;j++)
                map[i][j] = 0;

        ifstream ifl;
        if (mode == 1) ifl.open("./file./max1.txt");
        if (mode == 2) ifl.open("./file./max2.txt");
        if (mode == 3) ifl.open("./file./max3.txt");
        ifl >> maxs;
        ifl.close();
    }
```

5. Snake.cpp

```
#include"classes.h"
#include<iostream>
#include<fstream>

using namespace std;

int tx[4] = { 1,-1,0,0 }, ty[4] = { 0,0,1,-1 };
Snake::Snake(Game& G) {
	head = NULL, tail = NULL;
	destructed = 0;
	removed_x = -1, removed_y = -1;
	int x, y;
	while (1) {
		x = rand() % 40, y = rand() % 40, dir = rand() % 4;
		if (G.query(x, y) != 0) continue;
		if (G.query(x + tx[dir], y + ty[dir]) != 0) continue;
		if (G.query(x + tx[dir] * 2, y + ty[dir] * 2) != 0) continue;
		append(x, y, G);
		append(x + tx[dir], y + ty[dir], G);
		append(x + tx[dir] * 2, y + ty[dir] * 2, G);
		break;
	}
	if (dir == 0)dir = 1;
	else if (dir == 1)dir = 0;
	else if (dir == 2)dir = 3;
	else if (dir == 3)dir = 2;
}
void Snake::append(int x, int y, Game& G) {
	coor* p = new coor;
	p->x = x, p->y = y;
	if (head == NULL)
		head = p, tail = p;
	else
		tail->next = p, p->last=tail, tail = p;
	tail->next = NULL;
	G.upd_map(x, y, 3);
}
coor* Snake::snake_head() {
	return head;
}
void Snake::move(int t, Game& G) {
```

```
    if (t == -1) t = dir;
    else dir = t;

    removed_x = tail->x, removed_y = tail->y;
    G.upd_map(tail->x, tail->y, 0);
    coor* tmp = tail;
    tail = tail->last;
    tail->next = NULL;
    delete tmp;   //   去掉尾部，下面开始增加头部

    int hx = head->x, hy = head->y;
    int dx = hx + tx[t], dy = hy + ty[t];
    coor* cur = new coor;
    cur->x = dx, cur->y = dy;
    cur->next = head, head->last = cur;
    head = cur;
}
void Snake::recover(Game& G) {
    append(removed_x, removed_y, G);
}
bool Snake::reset(Game& G, Food& F,Wall& W,int type) {
    coor* p = head, * q;
    while (p != NULL) {
        q = p->next;
        if (p != head) {
            G.upd_map(p->x, p->y, 0); //进行一些擦白
            if (type == 1) W.append(p->x, p->y, G);
            else if (type == 2) F.append(p->x, p->y, G);
        }
        delete p;
        p = q;
    }
    //先判断剩余空间能不能生成新的小蛇!
    bool ade = 0; //是否有足够的空间
    for (int i = 0;i < 40;i++) {
        for (int j = 0;j < 40 - 2;j++) {
            if (G.query(i, j) == 0 && G.query(i, j + 1) == 0 && G.query(i, j + 2) == 0) {
                ade = 1;
                break;
            }
        }
        if (ade == 1) break;
    }
    if (!ade) {
```

```
            for (int j = 0;j < 40;j++) {
                for (int i = 0;i < 40 - 2;i++) {
                    if (G.query(i, j) == 0 && G.query(i + 1, j) == 0 && G.query(i + 2, j) == 0) {
                        ade = 1;
                        break;
                    }
                }
                if (ade == 1)break;
            }
        }
        if (!ade)return 0;
        //-----------进行一些构造函数的复制粘贴来生成新的小蛇------------
        head = NULL, tail = NULL;
        int x, y;
        while (1) {
            x = rand() % 40, y = rand() % 40, dir = rand() % 4;
            if (G.query(x, y) != 0) continue;
            if (G.query(x + tx[dir], y + ty[dir]) != 0) continue;
            if (G.query(x + tx[dir] * 2, y + ty[dir] * 2) != 0) continue;
            append(x, y, G);
            append(x + tx[dir], y + ty[dir], G);
            append(x + tx[dir] * 2, y + ty[dir] * 2, G);
            break;
        }
        if (dir == 0)dir = 1;
        else if (dir == 1)dir = 0;
        else if (dir == 2)dir = 3;
        else if (dir == 3)dir = 2;
        return 1;
    }
    int Snake::get_dir() {
        return dir;
    }
    Snake::~Snake() {
        if (!destructed) {
            ofstream fl("./file./record.txt", ios::app);
            if (!fl.is_open()) {
                cout << "文件打开失败" << endl;
                exit(1);
            }
            fl << dir << " " << removed_x << " " << removed_y << endl;
            coor* p = head, * q;
            while (p != NULL) {
                fl << p->x << " " << p->y << " ";
```

```
            q = p;
            p = p->next;
            delete q;
        }
        fl << "-1" << endl;
        fl.close();
        cout << "Snake over" << endl;
    }
    destructed = 1;
}
Snake::Snake(istream& in, Game& G) {
    head = NULL, tail = NULL;
    destructed = 0;
    in >> dir >> removed_x >> removed_y;
    int x, y;
    while (1) {
        in >> x;
        if (x == -1) break;
        in >> y;
        append(x, y, G);
    }
}
```

6. Food.cpp

```
#include"classes.h"
#include<iostream>
#include<fstream>

using namespace std;

Food::Food(Game& G) {
    head = NULL, tail = NULL;
    destructed = 0;
    int num = rand() % 5 + 1;
    while (num--) {
        int x = rand() % 38 + 1, y = rand() % 38 + 1;
        while (G.query(x, y) != 0)
            x = rand() % 38 + 1, y = rand() % 38 + 1;
        append(x, y, G);
    }
}
void Food::append(int x, int y, Game& G) {
    coor* p = new coor;
```

```
    p->x = x, p->y = y;
    if (head == NULL)
        head = p, tail = p;
    else
        tail->next = p, p->last = tail, tail = p;
    tail->next = NULL;
    G.upd_map(x, y, 2);
}
coor* Food::food_head() {
    return head;
}
void Food::remove(int x, int y, Game& G) {
    coor* p = head;
    while (!(p->x == x && p->y == y)) p=p->next;
    G.upd_map(x, y, 0); //将食物的部分擦白
    if (p == head && p == tail) {
        head = NULL, tail = NULL;
        delete p;
        //cout << "吃完食物啦" << endl;
        new_food(G);
    }
    else if (p == head) {
        head = p->next;
        p->next->last = NULL;
        delete p;
    }
    else if (p == tail) {
        tail = p->last;
        p->last->next = NULL;
        delete p;
    }
    else {
        p->last->next = p->next;
        p->next->last = p->last;
        delete p;
    }
}
void Food::new_food(Game& G) {
    int num = rand() % 5 + 1;
    while (num--) {
        int x = rand() % 40, y = rand() % 40;
        while (G.query(x, y) != 0)
            x = rand() % 40, y = rand() % 40;
        append(x, y, G);
```

```
    }
}
Food::~Food() {
    if (!destructed) {
        ofstream fl("./file./record.txt", ios::app);
        if (!fl.is_open()) {
            cout << "文件打开失败" << endl;
            exit(1);
        }
        coor* p = head, * q;
        while (p != NULL) {
            fl << p->x << " " << p->y << " ";
            q = p;
            p = p->next;
            delete q;
        }
        fl << "-1" << endl;
        fl.close();
        cout << "Food over" << endl;
    }
    destructed = 1;
}
Food::Food(istream& in, Game& G) {
    head = NULL, tail = NULL;
    destructed = 0;
    int x, y;
    while (1) {
        in >> x;
        if (x == -1) break;
        in >> y;
        append(x, y, G);
    }
}
```

7. Wall.cpp

```
#include"classes.h"
#include<iostream>
#include<fstream>

using namespace std;

Wall::Wall(Game& G) {
    head = NULL, tail = NULL;
```

```
	destructed = 0;
	for (int i = 0;i < 40;i++) {
		append(0, i, G);
		append(39, i, G);
		append(i, 0, G);
		append(i, 39, G);
	}
}
void Wall::append(int x, int y, Game& G) {
	coor* p = new coor;
	p->x = x, p->y = y;
	if (head == NULL)
		head = p, tail = p;
	else
		tail->next = p, p->last=tail, tail = p;
	tail->next = NULL;
	G.upd_map(x, y, 1);
}
coor* Wall::wall_head() {
	return head;
}
Wall::~Wall() {
	if (!destructed) {
		ofstream fl("./file./record.txt", ios::app);
		if (!fl.is_open()) {
			cout << "文件打开失败" << endl;
			exit(1);
		}
		coor* p = head, * q;
		while (p != NULL) {
			fl << p->x << " " << p->y << " ";
			q = p;
			p = p->next;
			delete q;
		}
		fl << "-1" << endl;
		fl.close();
		cout << "Wall over" << endl;
	}
	destructed = 1;
}
Wall::Wall(istream& in, Game& G) {
	head = NULL, tail = NULL;
	destructed = 0;
```

```
        int x, y;
        while (1) {
                in >> x;
                if (x == -1) break;
                in >> y;
                append(x, y, G);
        }
}
```