目录

**[1. 课题分析与方案选择 3](#_Toc32029)**

**2. [总体设计 4](#_Toc30232)**

[2.1 理论分析 5](#_Toc18537)

2.1.1 算法设计分析............................................................................................................5

2.1.2 数据结构分析............................................................................................................8

2.1.1 流程框图 ..........................................................................................................9

[2.2 代码编写 9](#_Toc16777)

2.2.1 工具类........................................................................................................................9

2.2.2 载入类......................................................................................................................11

2.2.3 蛇类..........................................................................................................................12

2.2.4 食物类......................................................................................................................13

2.2.5 地图类......................................................................................................................14

2.2.6 运行类......................................................................................................................15

**3. [游戏测试 15](#_Toc16610)**

[3.1 载入界面测试 16](#_Toc4570)

[3.2 简单模式测试 16](#_Toc17501)

[3.3 普通模式测试 1](#_Toc17731)6

[3.4 困难模式测试 1](#_Toc7983)7

**[4. 问题分析与解决 1](#_Toc10958)9**

**[5. 总结 2](#_Toc15182)0**

摘要

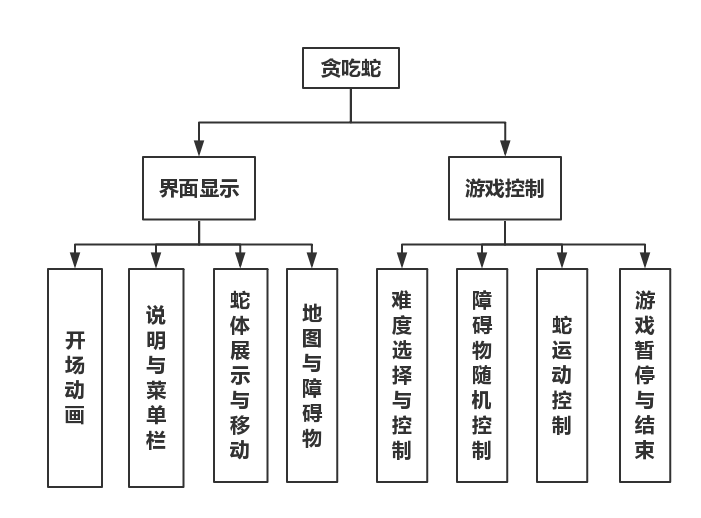
贪吃蛇游戏作为一款简单游戏，是手机游戏的代表，在十多年前风靡全世界，时至今日，贪吃蛇游戏任然活跃的网络的各个角落。本次编写的便是一款经典的贪吃蛇游戏，任务确定为实现贪吃蛇游戏过程。作为游戏的组成，通过必要的图形，文字界面来引导游戏者参与到这款游戏中，更有效的吸引游戏者的兴趣，为了拓展游戏者对高分和刺激的追求，设定各种难度的关卡也成为了游戏的必需品，贪吃蛇游戏的编写共有八个类，其中，主要包括游戏的控制与游戏界面的建立两个方面，对于前者，需要通过控制台来获取输入，而对于后者，要求应用合适的数据结构和算法，通过编写本次课程设计，对理论知识与实际作品之间的关系有了更加深刻影响。

关键字：贪吃蛇，界面，难度控制，数据结构

# 1 课题析与方案选择

## 1.1课题分析

本次设计为贪吃蛇游戏，一款游戏就必须要与使用者拥有良好的交互界面，使用C++对DOS界面显示的处理需要细致和精细，在游戏过程中，那么就会有游戏载入界面，游戏进行界面，和结束界面，在输入方面，这款游戏采用上下左右四个按键进行控制，需要用到回车键进行选择确定，而游戏的目标，则是获取更高分数，挑战更高难度。



功能设计框图

## 1.2方案选择

* + 1. 软件选择

本次编写软件为CODEBLOCKS 16.01

* + 1. 设计选择

通过分析，将游戏分为输入，游戏运行，输出三大部分，对于每个部分需要用不同的类进行实现和联系，大概将其分为五个类，分别为工具类，载入类，蛇体类，游戏界面类和运行类。

系统模块-功能表

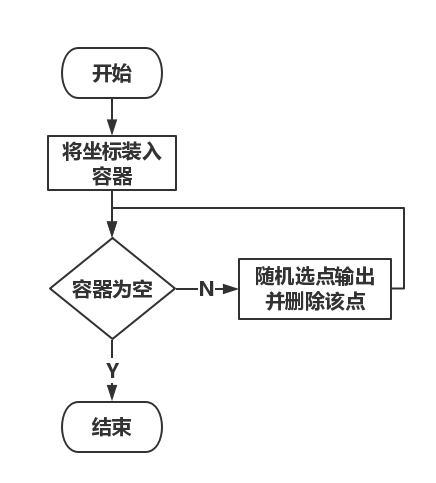
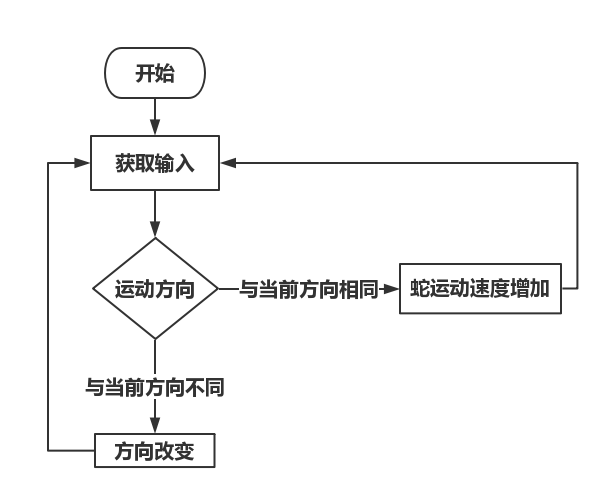
|  |  |
| --- | --- |
| 模块名称 | 功能 |
| 工具模块 | 在游戏中需要对界面进行不同的显示，比如输出位置和形状和颜色等，为了界面和谐美观，同时也需要控制DOS界面大小等，为了方便使用，需要一个工具模块，用于在编写过程中合理的运行和调试 |
| 蛇体模块 | 做为游戏的主要承载者，蛇本身也是游戏的重点，蛇的移动，转动方向，蛇体吃到食物变长，蛇撞到边界障碍物或者自己死亡等，需要用一个类对其进行详细的描述 |
| 界面模块 | 用于和使用者进行交互，譬如在游戏载入时，有难度选择，同时也会显示出选择的难度是什么，合理的使用字符让程序更美观。 |
| 运行模块 | 这个模块用于协调各个类之间的运行， 做为游戏的主要控制，根据输入和现有数据，对游戏运行做出正确的判断等，是游戏的处理和控制中心。 |

# 2 总体设计

2.1 理论分析

2.1.1 算法设计分析

1. 在载入界面时，在界面上设计显示出一个英文单词SNAKE，为了美观，便将其出现的方式进行改变，不再是按照行列进行单调的输出，采用随机在屏幕上打点的方式进行输出，为了实现这一目标，需要将SNAKE点阵的所有的坐标点保存到代码中，在输出时，随机取点，随机取颜色进行输出，通过控制延时，使整个单词显得游戏而且漂亮的输出，使游戏显得更有独特性。

载入动画流程图 控制方向流程图

（2）在游戏选择难度和控制方面，选用了通过键盘读取输入字符的方式来获得使用者的输入，如何显示出被选择的项目，通过用->标记以外，还使用了改变其背景底色的方式，明显且符号使用者的习惯。通过上下左右四个按键来控制菜单栏，是在各个程序都常用的方式。

|  |  |  |
| --- | --- | --- |
| 难度 | 属性控制 | 得分权重 |
| 简单难度 | 蛇运动反应时间为135 | 1 |
| 普通难度 | 蛇运动反应时间为110 | 2 |
| 困难难度 | 蛇运动反应时间为80，随机增加障碍物 | 3 |

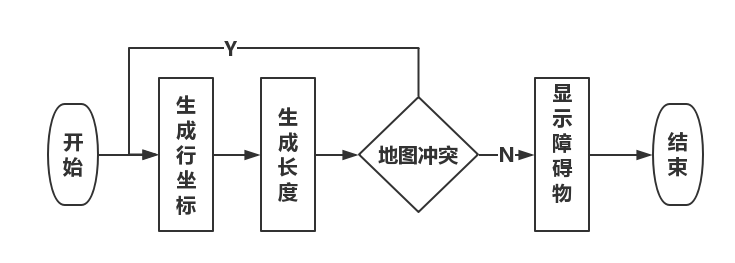
（3）如何控制蛇的移动，仔细分析蛇的移动可以看出，在蛇的每一次移动中，蛇头向前拓展一个单位，而后所有的蛇体都向前拓展一个单位，再把蛇尾去掉，倘若吃掉一个食物，那么就不用再把蛇尾去掉，蛇体的长度就又长了一个单位，为了描述当前蛇头的运动方向，还要用一个变量来保存，用其值表示当前蛇头的运动方向。

（4）用循环来实现蛇的移动，假设当前不对蛇体进行转向，那么在界面上蛇就会按照当前运动方向进行移动，如果通过键盘输入了方向，也将其指令分为三种类别，第一种是与当前蛇的运动相同，那么就加速，使蛇更快的朝着当前方向运动，第二种就是与当前蛇的运动相反，此时应该忽略掉这种指令，不能进行实现，第三种就是剩下的方向，那么就按照输入的方向，对现在蛇的方向进行修改。

（5）游戏结束的判断，贪吃蛇游戏结束存在两种判断，第一种是蛇头撞到障碍物比如地图边界，那么游戏结束，第二种是蛇头撞到自己的蛇体，游戏也会结束，那么就在蛇移动后判断，蛇头的坐标与障碍物和自己蛇体的所有点是否相同，如果存在一个相同的点，那么游戏结束，反之，游戏继续。

（6）游戏难度控制，如何控制游戏的难度，是使用者有不同的刺激体验，为此，结合了许多贪吃蛇游戏，我将难度控制放在了蛇移动速度和地图障碍物上面，蛇体的移动速度越快，那么留给游戏者的反应时间也就越短，而添加地图障碍物，则使得难度加大，地图不在是普通的长方形地图，存在与地图的障碍物也是游戏者需要用心注意，于是将简单难度控制为普通速度，在普通难度时，会增加速度，在困难难度时会在增加速度的同事，增加障碍物。

1. 食物和障碍物的控制，食物的出现必须在游戏开始界面，或者当蛇吃掉前一个食物的时候，才会生成下一个食物，也就是说整个地图只会有一个食物，这里引入随机，让下一个食物随机出现在地图的每一个地方，而障碍物的出现，便是在难度选择之后进行出现，且在这一局游戏中改变。但是食物和障碍物都会出现冲突，比如下一个食物的出现，用随机生成了一个坐标，但是这个坐标却是蛇体的一部分或者是某个障碍物，那么这就不合理，于是在随机的前提下，要确保生成数据的合理性，同时为了使障碍物更加规范，障碍物也不一定是一个点，可能是一堵墙，或者两堵墙挨在一起，这些都需要通过细节控制。



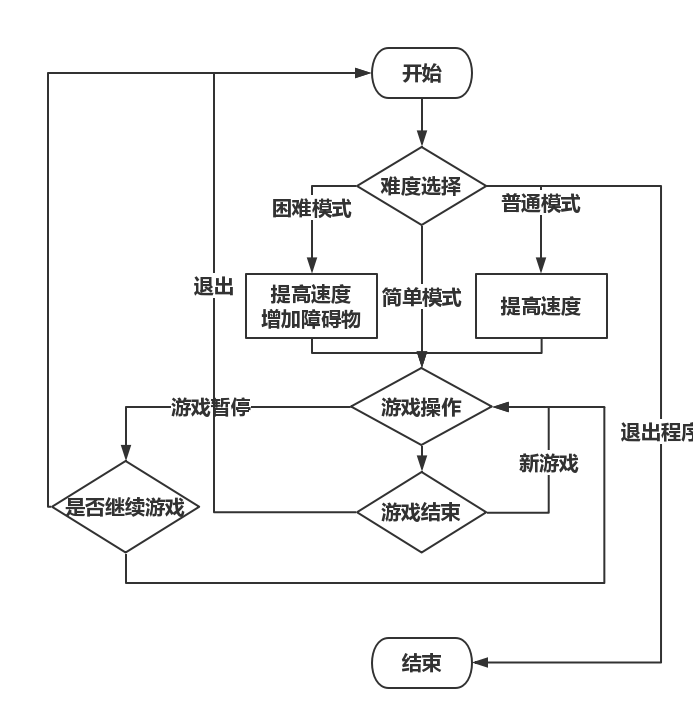
障碍物生成流程图

2.1.2 数据结构分析

在本次设计中，蛇需要有合适的数据结构进行保存和使用，为了方便对蛇的移动进行处理，蛇的移动会对蛇头和蛇尾频繁操作，而且为了更形象的描述蛇，需要数据结构有着线性结构，而为了保存地图，障碍物载入动画，需要对该数据结构有着较高读取方便的要求。

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 简介 | 说明 | 功能 |
| 双端队列deque | 双端队列是指两段都可以进行入队和出队操作的队列，其元素的逻辑结构仍是线性结构。将队列的两段分别称为前端和后端，两段都可以入队和出对 | 双端队列是适用于保存蛇坐标的数据结构，从其逻辑结构来看，与本次程序对蛇体的操作非常类似，蛇头可以看做是队列的前端，蛇尾则是后端，能够对两端进行操作 | Push\_back()  Push\_front()  Pop\_back()  Pop\_front() |
| 容器vector | vector是可以实自动增长的对象数组，用于多种功能，能够操作多种数据结构和算法的模板类和函数库 | 在不知道地图等数据的多少时，采用容器进行储存有许多的便利，能过快速的测定其大小和遍历容器内所有内容 | Push\_back()  Pop\_back()  Size() |

2.1.3 流程框图



2.2 代码编写

2.2.1 工具类

在工具类中，为了编写适用整个程序的工具，将需要的函数分为控制光标等七个子函数，其中使用到的大多数是对DOS界面的操作,主要函数为控制光标位置和修改输出格式。

void SetCursorPosition(int x,int y); //设置鼠标位置

void Setcmdsize(int wide,int high); //设置CMD大小

void Setcolour(int colour); //设置字符颜色

void Setbackcolour(); //设置黑底

void Clear\_colour(); //清空颜色

void HideCursor(); //光标隐藏

void Clearpoint(int x,int y); //清空字符

1. 控制光标位置

调用计算机函数SetConsoleCursorPosition，通过定义COORD pos，修改pos.x和pos.y的值就可以实现光标的位置控制

1. 修改字符颜色

使用SetConsoleTextAttribute，这是一个可以在应用程序编程接口中设置控制窗口字体演示和背景色的计算机函数。

原型：BOOL SetConsoleTextAttribute(HANDKE hConsoleOutput,WORD wAttributes);

功能表

|  |  |  |
| --- | --- | --- |
| wAttributes | 颜色 | 对应的值 |
| FOREGROUND\_BLUE | 字体颜色：蓝 | 1 |
| FOREGROUND\_GREEN | 字体颜色：绿 | 2 |
| FOREGROUND\_RED | 字体颜色：红 | 4 |
| FOREGROUND\_INTENSITY | 前景色高亮显示 | 8 |
| BACKGROUND\_BLUE | 背景颜色：蓝 | 16 |
| BACKGROUND\_GREEN | 背景颜色：绿 | 32 |
| BACKGROUND\_RED | 背景颜色：红 | 64 |
| BACKGROUND\_INTENSITY | 背景色高亮显示 | 128 |

2.2.2 载入类

void Select(); //选择难度

void show\_first(); //显示第一行被选中

void show\_second(); //显示第一行被选中

void show\_third(); //显示第一行被选中

void show\_fourth(); //显示第一行被选中

void add\_score(); //增加分数

int chooseend(int score); //游戏结束

void show\_YES(); //选择YES

void show\_NO(); //选择NO

void show\_load(); //载入界面显示

int show\_pause(); //游戏暂停

void show\_returnmeniu(); //返回菜单

void show\_continue(); //游戏继续

在载入类中，主要解决的是显示载入界面和选择难度两大部分，载入界面为闪烁显示字符点阵SNAKE，选择难度则是通过上下移动显示被选中状态。

1. 显示点阵：
2. 显示选择难度

int cnt = 1;

while(v.size()) //显示所有位置

{

int tmp = 0;

int now = rand() % v.size(); //随机选定

if(v[now].first > 40) //控制点阵字符距离

tmp += 2;

SetCursorPosition(v[now].first + 12 + tmp, v[now].second + 2);

cnt++;

cnt = cnt % 15 + 1; //循环字符演示

Setcolour(cnt);

cout << "\*"; //输出

Sleep(5); //休眠，达到闪烁效果

v.erase(v.begin() + now); //删除该点

}

在子函数中，为了达到显示某一行被选中的状态，通过先标记当前在第几行，然后以输入来控制上移动或者下移动，最后通过函数调用显示该行被选中状态。

while(ch = getch()) //获取键盘输入

{

if(ch == 224) ch = getch(); //处理出上下左右

if(ch == 72) //通过输入处理出移动

key--;

else if(ch == 80)

key++;

if(key == 0)

show\_fourth(), key = 4;

else if(key == 1)

show\_first();

else if(key == 2)

show\_second();

else if(key == 3)

show\_third();

else if(key == 4)

show\_fourth();

else if(key == 5)

show\_first(), key = 1;

if(ch == 13) break; //ENTER键确认

}

2.2.3 蛇类

蛇类的主要功能分为展示蛇体，控制蛇的移动与判断蛇是否与障碍物边界重合三大功能，而蛇的移动又分为，按照方向移动和改变运动方向，蛇体增加三个部分，需要分别实现。

1. 蛇移动

void show\_snake(); //输出蛇在界面

void add\_snake(); //向前移动一个单位

void move\_snake(); //尾部去掉一个单位

int check\_snake(vector<Cursor> v); //检测是否相撞

int change\_snake(); //改变方向

int eat\_food(int tmpx, int tmpy); //能否吃掉地图上的食物

根据蛇的运动方向，将头增加一个点，然后再将尾部去掉，就完成了蛇的一次移动

int dir[4][2] = { 0, -1, 0, 1, -1, 0, 1, 0}; //方向数组

int dx = snake.back().Getx() + dir[flag][0]; //增加新点

int dy = snake.back().Gety() + dir[flag][1];

snake.push\_back({dx, dy});

SetCursorPosition(dx, dy);

SetCursorPosition(snake.front().Getx(), snake.front().Gety());

snake.front().Clear\_point(); //去掉尾巴坐标

snake.pop\_front();

Setcolour(14);

snake.back().Print\_round();

1. 检查是否相撞

为了检查蛇是否与边界和障碍物还有自身相撞，需要传入障碍物参数，遍历所有障碍物的点坐标和蛇头坐标，如果存在一个点重合，那么此时游戏已经结束。

for(int i = 0; i < v.size(); i++)

if(v[i].Getx() == snake.back().Getx() && v[i].Gety() == snake.back().Gety())

return 0; //判断蛇头与自身是否相撞

for(int i = 0; i < snake.size(); i++) //遍历所有障碍物

{

int x = snake[i].Getx();

int y = snake[i].Gety();

if(x <= 0 || x >= 33 || y <= 0 || y >= 24) //判断边界

return 0;

if(i != snake.size() - 1)

{

if(x == snake.back().Getx() && y == snake.back().Gety())

return 0;

}

}

return 1;

2.2.4 食物类

食物类的功能主要是在地图上随机生成一个可以给蛇吃的食物，但是随机过种仍然有需要多限制，生成的食物不能是非法的坐标，比如边界外，或者是障碍物，或者是蛇体，这些都是不合法的位置，所以需要通过循环生成的方式，不断的判断，直到生成正确的食物的坐标。

2.2.5 地图类

while(flag != -1)

{

int tmpx = rand() % 32 + 1; //在地图范围能随机生成一个坐标

int tmpy = rand() % 23 + 1;

int cnt = 0;

for(int i = 0; i < qu.size(); i++) //确定这个不与所有蛇坐标重合

{

int dx = qu[i].Getx();

int dy = qu[i].Gety();

if(tmpx == dx || tmpy == dy)

cnt++;

}

for(int i = 0; i < v.size(); i++) //确定这个坐标不与所有障碍物重合

{

int dx = v[i].Getx();

int dy = v[i].Gety();

if(tmpx == dx || tmpy == dy)

cnt++;

}

if(cnt == 0) //满足条件，生成食物坐标成功

{

this->x = tmpx;

this->y = tmpy;

break ;

}

}

地图类的功能主要为绘制地图障碍物，更新分数，显示时间闪烁动画三大部分。

1. 绘制地图和障碍物

根据障碍物参数，将障碍物显示在地图上，绘制地图，根据所确定的地图范围，通过循环将所有坐标用容器保存下来，再进行输出。

1. 更新分数

当蛇吃掉一个食物，会得到一个得分，该得分为：当前得分+权重\*10，其中的权重便是所选中的难度，每吃掉一个食物就会通过地图类更新得分。

1. 显示时间和闪烁动画

为了方便使用者，在地图的右下角增加了显示系统时间，并且在地图的右上角增加闪烁小动画，使得整个界面更有动感。

int ret=rand()%15+1; //随机生成字符颜色

Setcolour(ret);

SetCursorPosition(2 \* 37, 2); //设定颜色和位置

cout << "贪 吃 蛇"; //输出字符

time\_t tt = time(NULL);

tm\* t = localtime(&tt); //获取系统时间

Setcolour(10);

SetCursorPosition(2 \* 42, 22); //设定位置

printf("%02d:%02d:%02d\n", t->tm\_hour, t->tm\_min, t->tm\_sec);

SetCursorPosition(2 \* 41, 23);

printf("%04d/%02d/%02d%\n", t->tm\_year + 1900, t->tm\_mon + 1, t->tm\_mday); //通过数学处理输出当前时间

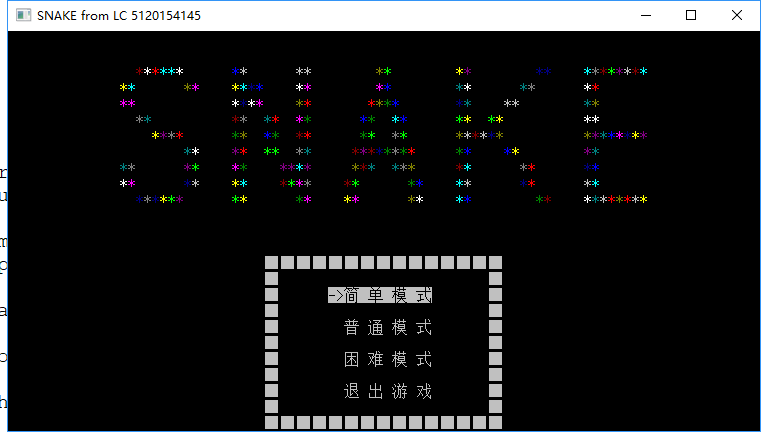
Clear\_colour();

2.2.6 运行类

运行类的功能是程序的控制中心，游戏从这里开始和结束，运行类通过调用其余所有类，依次显示载入动画，选择难度，进行游戏，游戏控制和判断结束游戏等。

3 游戏测试

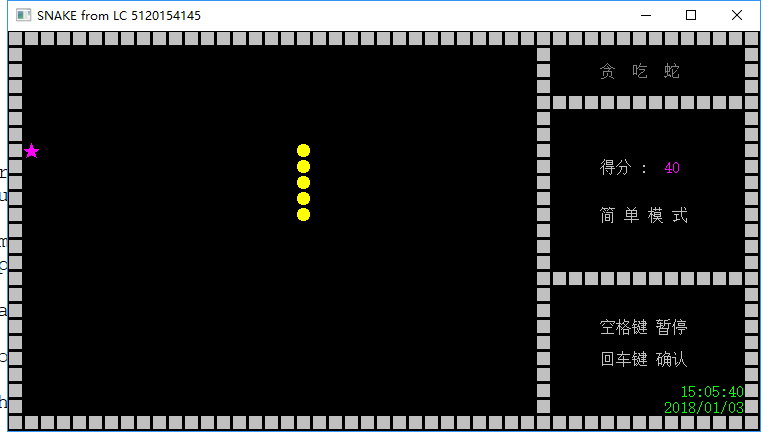
3.1 载入界面测试



载入界面图

在载入界面，闪烁出现点阵动画，控制延时时间，整体图形在2秒内显示完全，且点阵颜色随机，载入界面下放选择栏，通过获取键盘输入上下左右，模拟选择效果，回车键确认，载入界面功能基本实现。

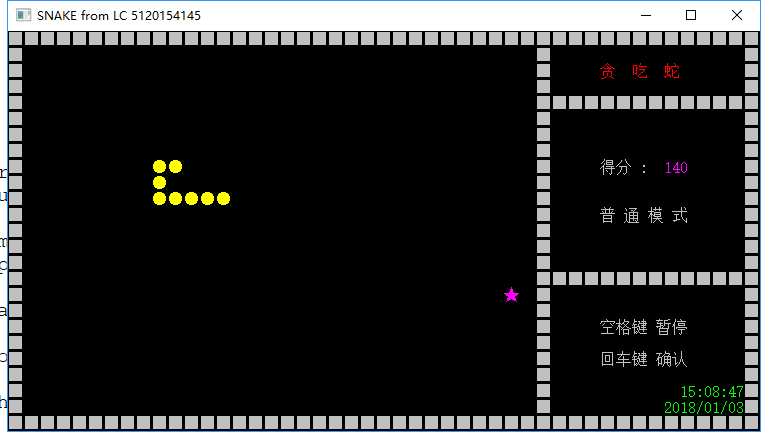
3.2 简单模式测试



简单模式界面图

在简单模式中，蛇运动速度较慢，难度控制合理，在地图上随机生成食物，食物生成未出现不合法情况，得分和时间更新正确，简单模式游戏功能完成，即使是新手玩家也能快速上手。

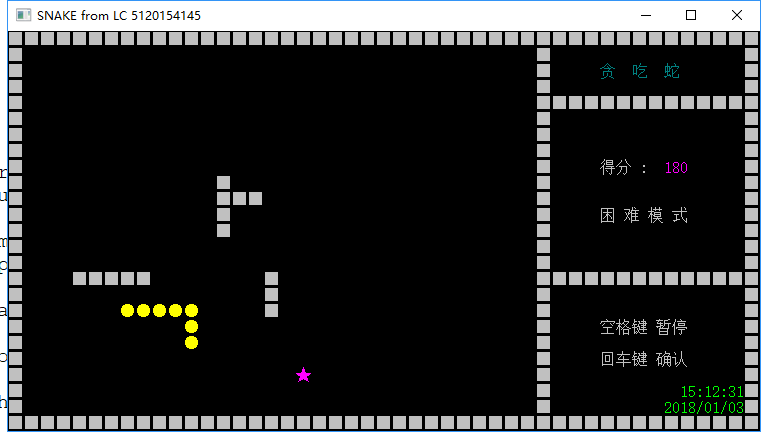
3.3 普通模式测试



普通模式界面图

在简单模式中，蛇运动速度适中，难度控制合理，在地图上随机生成食物，食物生成未出现不合法情况，得分和时间更新正确，相比简单模式，速度提高明显，适合有一定游戏基础的人进行游戏。

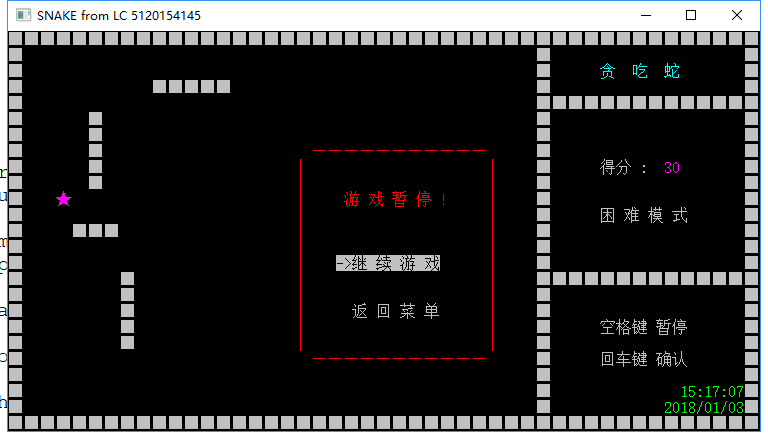
3.4 困难模式测试



苦难模式界面图

在苦难模式中，蛇的移动速度加快，并且在地图中随机出现障碍物，给游戏的操作带来了难度，需要有快速的反应力，困难模式功能基本实现。

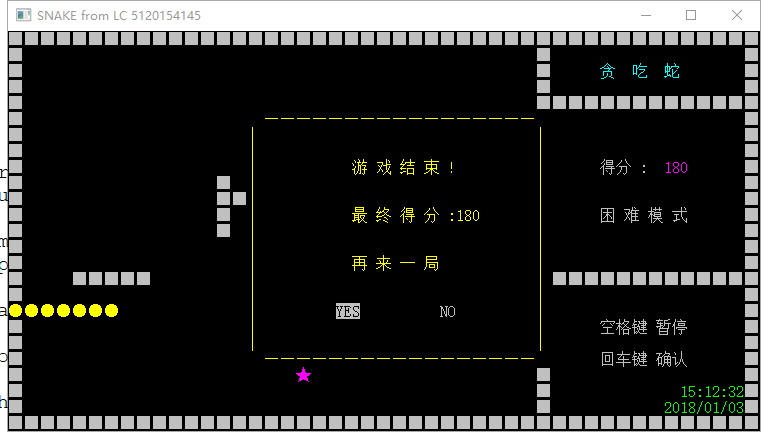
3.5 游戏暂停与继续游戏测试



游戏暂停界面图

当在游戏中按下空格键，则表示暂停游戏，在暂停游戏后，在界面中间弹出提示框，表示当前游戏暂停，需要进行选择是否继续游戏，该功能成功实现，在三种模式下测试均为发现异常。

3.6 游戏结束界面测试



游戏结束界面图

当游戏结束后，会在界面的中央弹出提示框，表面游戏结束和最终得分，并且能够选择是否继续游戏，在简单，普通，困难三种不同难度的模式下进行测试，均未发现异常，游戏结束界面功能基本实现。

4 问题分析与解决

1. 载入动画点阵

问题描述：如何将点阵显示在DOS界面上

解决办法：根据设想，需要将SNAKE以\*号点阵的样式输出到界面上，首先就要出处理好点阵坐标问题，为了更好的显示，需要将所有坐标保存下来，因此通过文件读入的方式，将点阵在文本中描绘好，在程序中预处理读入，保存在一个二维数组里面，通过遍历该数组中所有是\*号的点，然后输出该坐标，就能得到所有点的坐标，再将所有坐标存入容器中，便可以输出到DOS界面上。

（2）载入界面设计

问题描述：在设计载入动画时，将坐标存入容器后，随机选点然后输出在界面上，因为随机性的问题，选的点可能会不断选到重复的点，也就是会有一些点选中的时间会变长，若想要将所有点都完整的展现在界面上，所需要的时间大大增加，并且不能确定具体时间，与最初的设想相违背。

解决办法：思考容器的功能，为了保证不选取已经输出的点，当选中一个点后，就将该点从容器中删除，这样就能保证每次选中的点都是为出现的点，从整体上控制了载入动画的时间，使该功能能够完美实现。

（3）地图绘制

问题描述：为了描绘出整个地图，需要将所有边界坐标都显示出来，如何快速的处理出所有坐标

解决办法：通过for循环方式，两重for循环，第一重遍历横坐标，第二重遍历纵坐标，然后加入判断，就可以很方便快速的描绘出整个地图边界。

（4）蛇运动方向控制

问题描述：在蛇类中，一个重要属性就是蛇的运动方向，但是如何保存蛇的方向，便于程序使用。

解决办法：用一个整型变量，其数值为0，1，2，3分别代表当前蛇的方向是上，下，左，右，在使用上面，可以配合建立一个方向数组为{ 0, -1, 0, 1, -1, 0, 1, 0}，通过调用方向变量，对当前坐标进行运算，就可以便利的获得当前方向的下一个坐标。

5 总结

通过本次设计，首先对自己的编程能力有了一次较大的提高，将课堂上学到的东西用于实际也是一次巨大的进步，通过不断地使用和学习C++，更加熟悉了解了C++这门语言，同时，在设计中遇到了许多的问题，这些问题使得自己的思考方式更加贴近机器语言，有时候能够通过自己的思考能过解决的问题，提高了自己的逻辑思维能力，当遇到不能解决的问题，例如如何控制光标的位置和字符颜色，可以通过网上查找相关资料，然后加以调试实验，主动学习的效果使得自己提高了很多，在编程中也遇到了许多代码上的问题，学会了如何使用codeblock单步调试，更加熟练的使用这款软件，由于本次设计是一款小游戏，也让自己对游戏的概念深刻了许多，在玩游戏的同时，也会不停思考，如何实现这个功能，为什么要怎么写，收益巨大。在编写程序前，在脑海里建立整个游戏的大概框架，将游戏抽象为不同的类，深刻的理解到了C++面对对象编程的思想，当抽象为类后再对事物的属性功能依次实现，最后再通过函数统一调用，使程序不再仅仅只是冷漠的代码，更像是一个有着生命的实体。当代码量不再是课题上的几十排后，数百近千排的代码需要有着严谨的逻辑性，如何通过代码，将头脑中思考的游戏过程实现，是需要花很多时间去学习，通过不同的方法实现，代码量加大后，考虑得更多的就是自己对代码的注释说明，一步步的设计可以通过参考文档进行记录参数，实现方法和过程结果，提高了自己对解决问题的开拓性，能够积极的去解决问题，而不再是一味的询问他人。同时，也对程序等于算法加数据结构这句话有了更深刻的理解，实现过程有多种方法，但是好的算法和数据结构能够让头脑更清晰，使代码更加简洁易懂。

参考文献：

[1]姚庭宝. C++语言及编程技巧, 长沙：国防科技大学出版社，2003.10：15-21

[2]李春葆,苏光奎. 数据结构与算法教程, 北京：清华大学出版社，2005.06：47-64

[3]刘汝佳.算法竞赛入门经典第二版，北京：清华大学出版社，2014.06：13-17

代码清单：

#include "Play.h"

#include"Food.h"

#include"Cursor.h"

#include"Load.h"

#include"Tool.h"

#include"map.h"

#include"Snake.h"

#include"windows.h"

#include"bits/stdc++.h"

using namespace std;

void Playgame()

{

int flag = 0;

while(1)

{

system("cls");

HideCursor();

srand(0);

Setcmdsize(47, 25);

Load ld;

ld.Select();

if(ld.getkey() == 4)

break;

while(1)

{

ld.intiscore();

Clear\_colour();

Map mp;

mp.show\_time();

mp.show\_map(ld.getkey(), 1, ld.getscore());

Snake sk;

sk.show\_snake();

Food fd;

fd.show\_food(sk.snake, mp.stone, 1);

flag = 0;

while(sk.check\_snake(mp.stone))

{

mp.show\_time();

int change = sk.change\_snake();

if(change == -1)

{

if(ld.show\_pause() == 2)

{

flag = 2;

break;

}

else

{

mp.show\_map(ld.getkey(), -1, ld.getscore());

sk.show\_snake();

fd.show\_food(sk.snake, mp.stone, -1);

}

}

if(sk.eat\_food(fd.Getx(), fd.Gety()))

{

sk.add\_snake();

fd.show\_food(sk.snake, mp.stone, 1);

ld.add\_score();

mp.updatescore(ld.getscore());

}

else

sk.move\_snake();

int speed;

if(ld.getkey() == 1)

speed = 135;

else if(ld.getkey() == 2)

speed = 110;

else if(ld.getkey() == 3)

speed = 80;

if(change == -2)

speed = 5;

Sleep(speed);

}

if(flag == 2)

break;

if(ld.chooseend(ld.getscore()) == 2)

break;

}

}

}

#include "Snake.h"

#include"Cursor.h"

#include"Tool.h"

#include<bits/stdc++.h>

#include<conio.h>

#include <thread>

using namespace std;

void Snake::show\_snake()

{

for(int i = 0; i < snake.size(); i++)

{

Setcolour(14);

snake[i].Print\_round();

}

}

void Snake::add\_snake()

{

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

int dx = snake.back().Getx() + dir[flag][0];

int dy = snake.back().Gety() + dir[flag][1];

snake.push\_back({dx, dy});

SetCursorPosition(dx, dy);

Setcolour(14);

snake.back().Print\_round();

}

void Snake::move\_snake()

{

add\_snake();

SetCursorPosition(snake.front().Getx(), snake.front().Gety());

snake.front().Clear\_point();

snake.pop\_front();

}

int Snake::check\_snake(vector<Cursor> v)

{

for(int i = 0; i < v.size(); i++)

if(v[i].Getx() == snake.back().Getx() && v[i].Gety() == snake.back().Gety())

return 0;

for(int i = 0; i < snake.size(); i++)

{

int x = snake[i].Getx();

int y = snake[i].Gety();

if(x <= 0 || x >= 33 || y <= 0 || y >= 24)

return 0;

if(i != snake.size() - 1)

{

if(x == snake.back().Getx() && y == snake.back().Gety())

return 0;

}

}

return 1;

}

int Snake::change\_snake()

{

int ch, key = 0;

if(kbhit())

{

ch = getch();

if(ch == 72)

{

if(flag == 1)

return 0;

if(flag == 0)

return -2;

else

flag = 0, key = 1;

}

else if(ch == 80)

{

if(flag == 0)

return 0;

if(flag == 1)

return -2;

else

flag = 1, key = 1;

}

else if(ch == 75)

{

if(flag == 3)

return 0;

if(flag == 2)

return -2;

else

flag = 2, key = 1;

}

else if(ch == 77)

{

if(flag == 2)

return 0;

if(flag == 3)

return -2;

else

flag = 3, key = 1;

}

else if(ch == 32)

return -1;

}

return key;

}

int Snake::eat\_food(int tmpx, int tmpy)

{

if(snake.back().Getx() == tmpx && snake.back().Gety() == tmpy)

return 1;

return 0;

}

#include"Play.h"

int main()

{

Playgame();

return 0;

}

#include "Cursor.h"

#include "Tool.h"

#include<iostream>

void Cursor::Print\_square()

{

SetCursorPosition(2 \* x, y);

std::cout << "■" ;

}

void Cursor::Print\_round()

{

SetCursorPosition(2\*x, y);

std::cout << "●" ;

}

void Cursor::Clear\_point()

{

SetCursorPosition(2\*x, y);

std::cout << " " ;

}

void Cursor::Change\_point(int x, int y)

{

this->x = x;

this->y = y;

}

#include "Map.h"

#include<windows.h>

#include"Tool.h"

#include <time.h>

void Map::show\_map(int flag, int pause\_flag, int score)

{

system("cls");

for(int i = 0; i < newmap.size(); i++)

{

newmap[i].Print\_square();

if(pause\_flag != -1)

Sleep(5);

}

if(flag == 3 && pause\_flag != -1)

add\_stone();

for(int i = 0; i < stone.size(); i++)

stone[i].Print\_square();

SetCursorPosition(2 \* 37, 2);

cout << "贪 吃 蛇";

SetCursorPosition(2 \* 37, 8);

cout << "得分 : ";

Setcolour(13);

SetCursorPosition(2 \* 41, 8);

cout << score;

Clear\_colour();

SetCursorPosition(2 \* 37, 11);

if(flag == 1)

cout << "简 单 模 式";

else if(flag == 2)

cout << "普 通 模 式";

else if(flag == 3)

cout << "困 难 模 式";

SetCursorPosition(74, 18);

cout << "空格键 暂停";

SetCursorPosition(74, 20);

cout << "回车键 确认";

}

void Map::add\_stone()

{

time\_t tt = time(NULL);

tm\* t = localtime(&tt);

srand(t->tm\_min \* t->tm\_sec);

for(int k = 0; k < 2; k++)

{

int x = rand() % 30 + 1;

if(x == 1)

x++;

int ly, ry;

while(1)

{

ly = rand() % 20 + 1;

ry = rand() % 20 + 1;

if(ly == 1)

ly++;

if(ry == 1)

ry++;

if(ly > ry)

swap(ly, ry);

if(ry - ly + 1 <= 5 && ry - ly + 1 >= 2)

break;

}

for(int i = ly; i <= ry; i++)

stone.push\_back({x, i});

}

for(int k = 0; k < 2; k++)

{

int x = rand() % 20 + 1;

if(x == 1)

x++;

int ly, ry;

while(1)

{

ly = rand() % 30 + 1;

ry = rand() % 30 + 1;

if(ly == 1)

ly++;

if(ry == 1)

ry++;

if(ly > ry)

swap(ly, ry);

if(ry - ly + 1 <= 5 && ry - ly + 1 >= 3)

break;

}

for(int i = ly; i <= ry; i++)

stone.push\_back({i, x});

}

}

void Map::updatescore(int score)

{

SetCursorPosition(2 \* 41, 8);

cout << score;

}

void Map::show\_time()

{

int ret=rand()%15+1;

Setcolour(ret);

SetCursorPosition(2 \* 37, 2);

cout << "贪 吃 蛇";

time\_t tt = time(NULL);

tm\* t = localtime(&tt);

Setcolour(10);

SetCursorPosition(2 \* 42, 22);

printf("%02d:%02d:%02d\n", t->tm\_hour, t->tm\_min, t->tm\_sec);

SetCursorPosition(2 \* 41, 23);

printf("%04d/%02d/%02d%\n", t->tm\_year + 1900, t->tm\_mon + 1, t->tm\_mday);

Clear\_colour();

}

#include "Food.h"

#include"Tool.h"

#include<bits/stdc++.h>

#include<windows.h>

#include"Snake.h"

#include"Cursor.h"

using namespace std;

Food::Food()

{

x = 5, y = 5;

}

void Food::show\_food(deque<Cursor> qu, vector<Cursor> v, int flag)

{

while(flag != -1)

{

int tmpx = rand() % 32 + 1;

int tmpy = rand() % 23 + 1;

int cnt = 0;

for(int i = 0; i < qu.size(); i++)

{

int dx = qu[i].Getx();

int dy = qu[i].Gety();

if(tmpx == dx || tmpy == dy)

cnt++;

}

for(int i = 0; i < v.size(); i++)

{

int dx = v[i].Getx();

int dy = v[i].Gety();

if(tmpx == dx || tmpy == dy)

cnt++;

}

if(cnt == 0)

{

this->x = tmpx;

this->y = tmpy;

break ;

}

}

SetCursorPosition(x \* 2, y);

Setcolour(13);

cout << "★";

}

#include "Load.h"

#include "Tool.h"

#include"Cursor.h"

#include<iostream>

#include<conio.h>

#include<windows.h>

#include<bits/stdc++.h>

using namespace std;

void Load::show\_first()

{

Setbackcolour();

SetCursorPosition(40, 16);

cout << "->简 单 模 式" << endl;

Clear\_colour();

SetCursorPosition(40, 18);

cout << " 普 通 模 式" << endl;

SetCursorPosition(40, 20);

cout << " 困 难 模 式" << endl;

SetCursorPosition(40, 22);

cout << " 退 出 游 戏" << endl;

}

void Load::show\_second()

{

SetCursorPosition(40, 16);

cout << " 简 单 模 式" << endl;

Setbackcolour();

SetCursorPosition(40, 18);

cout << "->普 通 模 式" << endl;

Clear\_colour();

SetCursorPosition(40, 20);

cout << " 困 难 模 式" << endl;

SetCursorPosition(40, 22);

cout << " 退 出 游 戏" << endl;

}

void Load::show\_third()

{

SetCursorPosition(40, 16);

cout << " 简 单 模 式" << endl;

SetCursorPosition(40, 18);

cout << " 普 通 模 式" << endl;

Setbackcolour();

SetCursorPosition(40, 20);

cout << "->困 难 模 式" << endl;

Clear\_colour();

SetCursorPosition(40, 22);

cout << " 退 出 游 戏" << endl;

}

void Load::show\_fourth()

{

SetCursorPosition(40, 16);

cout << " 简 单 模 式" << endl;

SetCursorPosition(40, 18);

cout << " 普 通 模 式" << endl;

SetCursorPosition(40, 20);

cout << " 困 难 模 式" << endl;

SetCursorPosition(40, 22);

Setbackcolour();

cout << "->退 出 游 戏" << endl;

Clear\_colour();

}

void Load::show\_load()

{

vector<pair<int, int>> v;

v.push\_back({4, 0});

v.push\_back({5, 0});

v.push\_back({6, 0});

v.push\_back({7, 0});

v.push\_back({8, 0});

v.push\_back({9, 0});

v.push\_back({16, 0});

v.push\_back({17, 0});

v.push\_back({24, 0});

v.push\_back({25, 0});

v.push\_back({34, 0});

v.push\_back({35, 0});

v.push\_back({42, 0});

v.push\_back({43, 0});

v.push\_back({52, 0});

v.push\_back({53, 0});

v.push\_back({58, 0});

v.push\_back({59, 0});

v.push\_back({60, 0});

v.push\_back({61, 0});

v.push\_back({62, 0});

v.push\_back({63, 0});

v.push\_back({64, 0});

v.push\_back({65, 0});

v.push\_back({2, 1});

v.push\_back({3, 1});

v.push\_back({10, 1});

v.push\_back({11, 1});

v.push\_back({16, 1});

v.push\_back({17, 1});

v.push\_back({18, 1});

v.push\_back({19, 1});

v.push\_back({24, 1});

v.push\_back({25, 1});

v.push\_back({34, 1});

v.push\_back({35, 1});

v.push\_back({42, 1});

v.push\_back({43, 1});

v.push\_back({50, 1});

v.push\_back({51, 1});

v.push\_back({58, 1});

v.push\_back({59, 1});

v.push\_back({2, 2});

v.push\_back({3, 2});

v.push\_back({16, 2});

v.push\_back({17, 2});

v.push\_back({18, 2});

v.push\_back({19, 2});

v.push\_back({24, 2});

v.push\_back({25, 2});

v.push\_back({33, 2});

v.push\_back({34, 2});

v.push\_back({35, 2});

v.push\_back({36, 2});

v.push\_back({42, 2});

v.push\_back({43, 2});

v.push\_back({48, 2});

v.push\_back({49, 2});

v.push\_back({58, 2});

v.push\_back({59, 2});

v.push\_back({4, 3});

v.push\_back({5, 3});

v.push\_back({16, 3});

v.push\_back({17, 3});

v.push\_back({20, 3});

v.push\_back({21, 3});

v.push\_back({24, 3});

v.push\_back({25, 3});

v.push\_back({32, 3});

v.push\_back({33, 3});

v.push\_back({36, 3});

v.push\_back({37, 3});

v.push\_back({42, 3});

v.push\_back({43, 3});

v.push\_back({46, 3});

v.push\_back({47, 3});

v.push\_back({58, 3});

v.push\_back({59, 3});

v.push\_back({6, 4});

v.push\_back({7, 4});

v.push\_back({8, 4});

v.push\_back({9, 4});

v.push\_back({16, 4});

v.push\_back({17, 4});

v.push\_back({20, 4});

v.push\_back({21, 4});

v.push\_back({24, 4});

v.push\_back({25, 4});

v.push\_back({32, 4});

v.push\_back({33, 4});

v.push\_back({36, 4});

v.push\_back({37, 4});

v.push\_back({42, 4});

v.push\_back({43, 4});

v.push\_back({44, 4});

v.push\_back({45, 4});

v.push\_back({46, 4});

v.push\_back({47, 4});

v.push\_back({58, 4});

v.push\_back({59, 4});

v.push\_back({60, 4});

v.push\_back({61, 4});

v.push\_back({62, 4});

v.push\_back({63, 4});

v.push\_back({64, 4});

v.push\_back({65, 4});

v.push\_back({10, 5});

v.push\_back({11, 5});

v.push\_back({16, 5});

v.push\_back({17, 5});

v.push\_back({20, 5});

v.push\_back({21, 5});

v.push\_back({24, 5});

v.push\_back({25, 5});

v.push\_back({31, 5});

v.push\_back({32, 5});

v.push\_back({33, 5});

v.push\_back({34, 5});

v.push\_back({35, 5});

v.push\_back({36, 5});

v.push\_back({37, 5});

v.push\_back({38, 5});

v.push\_back({42, 5});

v.push\_back({43, 5});

v.push\_back({48, 5});

v.push\_back({49, 5});

v.push\_back({58, 5});

v.push\_back({59, 5});

v.push\_back({2, 6});

v.push\_back({3, 6});

v.push\_back({10, 6});

v.push\_back({11, 6});

v.push\_back({16, 6});

v.push\_back({17, 6});

v.push\_back({22, 6});

v.push\_back({23, 6});

v.push\_back({24, 6});

v.push\_back({25, 6});

v.push\_back({31, 6});

v.push\_back({32, 6});

v.push\_back({33, 6});

v.push\_back({36, 6});

v.push\_back({37, 6});

v.push\_back({38, 6});

v.push\_back({42, 6});

v.push\_back({43, 6});

v.push\_back({50, 6});

v.push\_back({51, 6});

v.push\_back({58, 6});

v.push\_back({59, 6});

v.push\_back({2, 7});

v.push\_back({3, 7});

v.push\_back({10, 7});

v.push\_back({11, 7});

v.push\_back({16, 7});

v.push\_back({17, 7});

v.push\_back({22, 7});

v.push\_back({23, 7});

v.push\_back({24, 7});

v.push\_back({25, 7});

v.push\_back({30, 7});

v.push\_back({31, 7});

v.push\_back({38, 7});

v.push\_back({39, 7});

v.push\_back({42, 7});

v.push\_back({43, 7});

v.push\_back({50, 7});

v.push\_back({51, 7});

v.push\_back({58, 7});

v.push\_back({59, 7});

v.push\_back({4, 8});

v.push\_back({5, 8});

v.push\_back({6, 8});

v.push\_back({7, 8});

v.push\_back({8, 8});

v.push\_back({9, 8});

v.push\_back({16, 8});

v.push\_back({17, 8});

v.push\_back({24, 8});

v.push\_back({25, 8});

v.push\_back({30, 8});

v.push\_back({31, 8});

v.push\_back({38, 8});

v.push\_back({39, 8});

v.push\_back({42, 8});

v.push\_back({43, 8});

v.push\_back({52, 8});

v.push\_back({53, 8});

v.push\_back({58, 8});

v.push\_back({59, 8});

v.push\_back({60, 8});

v.push\_back({61, 8});

v.push\_back({62, 8});

v.push\_back({63, 8});

v.push\_back({64, 8});

v.push\_back({65, 8});

int cnt = 1;

while(v.size())

{

int tmp = 0;

int now = rand() % v.size();

if(v[now].first > 40)

tmp += 2;

SetCursorPosition(v[now].first + 12 + tmp, v[now].second + 2);

cnt++;

cnt = cnt % 15 + 1;

Setcolour(cnt);

cout << "\*";

Sleep(5);

v.erase(v.begin() + now);

}

Clear\_colour();

for(int i=16;i<=30;i++)

{

Cursor now=Cursor(i,14);

now.Print\_square();

}

for(int i=16;i<=30;i++)

{

Cursor now=Cursor(i,24);

now.Print\_square();

}

for(int i=14;i<=24;i++)

{

Cursor now=Cursor(16,i);

now.Print\_square();

}

for(int i=14;i<=24;i++)

{

Cursor now=Cursor(30,i);

now.Print\_square();

}

}

void Load::Select()

{

HideCursor();

show\_load();

show\_first();

int ch, key = 1;

while(ch = getch())

{

if(ch == 224) ch = getch();

if(ch == 72)

key--;

else if(ch == 80)

key++;

if(key == 0)

show\_fourth(), key = 4;

else if(key == 1)

show\_first();

else if(key == 2)

show\_second();

else if(key == 3)

show\_third();

else if(key == 4)

show\_fourth();

else if(key == 5)

show\_first(), key = 1;

if(ch == 13) break;

}

this->key = key;

}

void Load::add\_score()

{

score += key \* 10;

}

void Load::show\_YES()

{

SetCursorPosition(41, 17);

Setbackcolour();

cout << "YES" ;

Clear\_colour();

SetCursorPosition(54, 17);

cout << "NO";

}

void Load::show\_NO()

{

SetCursorPosition(41, 17);

cout << "YES" ;

Setbackcolour();

SetCursorPosition(54, 17);

cout << "NO";

Clear\_colour();

}

int Load::chooseend(int score)

{

Setcolour(14);

for(int i = 15; i <= 33; i++)

for(int j = 5; j <= 20; j++)

{

SetCursorPosition(2 \* i, j);

if((i == 15 || i == 33) && (j != 5 && j != 20))

cout << "|";

else if((j == 5 || j == 20) && (i != 15 && i != 33))

cout << "——";

else

cout << " ";

}

SetCursorPosition(43, 8);

cout << "游 戏 结 束 !";

SetCursorPosition(43, 11);

cout << "最 终 得 分 :" << score;

SetCursorPosition(43, 14);

cout << "再 来 一 局";

show\_YES();

int ch, key = 1;

while(ch = getch())

{

if(ch == 224) ch = getch();

if(ch == 75)

key--;

else if(ch == 77)

key++;

if(key == 0)

show\_NO(), key = 2;

else if(key == 1)

show\_YES();

else if(key == 2)

show\_NO();

else if(key == 3)

show\_YES(), key = 1;

if(ch == 13) break;

}

return key;

}

void Load::show\_continue()

{

Setcolour(12);

SetCursorPosition(41, 14);

Setbackcolour();

cout << "->继 续 游 戏" ;

Clear\_colour();

SetCursorPosition(41, 17);

cout << " 返 回 菜 单";

}

void Load::show\_returnmeniu()

{

Setcolour(12);

SetCursorPosition(41, 14);

cout << " 继 续 游 戏" ;

Setbackcolour();

SetCursorPosition(41, 17);

cout << "->返 回 菜 单";

Clear\_colour();

}

int Load::show\_pause()

{

Setcolour(12);

for(int i = 18; i <= 30; i++)

for(int j = 7; j <= 20; j++)

{

SetCursorPosition(2 \* i, j);

if((i == 18 || i == 30) && (j != 7 && j != 20))

cout << "|";

else if((j == 7 || j == 20) && (i != 18 && i != 30))

cout << "——";

else

cout << " ";

}

SetCursorPosition(41, 10);

cout << " 游 戏 暂 停 !" ;

show\_continue();

int ch, key = 1;

while(ch = getch())

{

if(ch == 224) ch = getch();

if(ch == 72)

key--;

else if(ch == 80)

key++;

if(key == 0)

show\_returnmeniu(), key = 2;

else if(key == 1)

show\_continue();

else if(key == 2)

show\_returnmeniu();

else if(key == 3)

show\_continue(), key = 1;

if(ch == 13) break;

}

return key;

}

#ifndef MAP\_H

#define MAP\_H

#include<bits/stdc++.h>

#include"Cursor.h"

using namespace std;

class Map

{

public:

Map()

{

for(int i=0;i<=46;i++)

for(int j=0;j<=24;j++)

{

if((i==0||j==0||i==33||j==24)&&i<=33&&j<=24)

newmap.push\_back(Cursor(i,j));

else if((j==0||j==24)&&i<=46&&i>=34)

newmap.push\_back(Cursor(i,j));

else if(i==46)

newmap.push\_back(Cursor(i,j));

}

for(int i=33;i<=46;i++)

newmap.push\_back(Cursor(i,15)),newmap.push\_back(Cursor(i,4));

}

void add\_stone();

void show\_map(int flag,int pause\_flag,int score);

void updatescore(int score);

void show\_time();

vector<Cursor> stone;

private:

vector<Cursor> newmap;

};

#endif // MAP\_H

#ifndef PLAY\_H

#define PLAY\_H

void Playgame();

#endif // PLAY\_H

#ifndef FOOD\_H

#define FOOD\_H

#include"Snake.h"

#include<bits/stdc++.h>

#include"Cursor.h"

using namespace std;

class Food

{

public:

Food();

void show\_food(deque<Cursor> qu,vector<Cursor> v,int flag);

int Getx()

{

return this->x;

}

int Gety()

{

return this->y;

}

private:

int x, y;

friend class Snake;

};

#endif // FOOD\_H

#ifndef LOAD\_H

#define LOAD\_H

class Load

{

public:

Load(int key=1,int score=0):key(key),score(score){}

void Select();

void show\_first();

void show\_second();

void show\_third();

void show\_fourth();

void add\_score();

int chooseend(int score);

void show\_YES();

void show\_NO();

void show\_load();

int show\_pause();

void show\_returnmeniu();

void show\_continue();

void intiscore()

{

this->score=0;

}

int getkey()

{

return key;

}

int getscore()

{

return score;

}

private:

int key;

int score;

};

#endif // LOAD\_H

#ifndef SNAKE\_H

#define SNAKE\_H

#include<bits/stdc++.h>

#include"Food.h"

#include"Cursor.h"

using namespace std;

class Snake

{

public:

Snake()

{

snake.push\_back({23, 23});

flag = 0;

}

void show\_snake();

void add\_snake();

void move\_snake();

int check\_snake(vector<Cursor> v);

int change\_snake();

int eat\_food(int tmpx, int tmpy);

static void change()

{

cout<<"GG";

}

deque<Cursor> snake;

int flag;

friend class Food;

};

#endif // SNAKE\_H

#ifndef CURSOR\_H

#define CURSOR\_H

class Cursor

{

public:

Cursor() {}

Cursor(int x, int y): x(x), y(y) {};

void Print\_square();

void Print\_round();

void Clear\_point();

void Change\_point(int x, int y);

bool operator== (const Cursor& cur)

{

return (cur.x == this->x) && (cur.y == this->y);

}

int Getx()

{

return this->x ;

}

int Gety()

{

return this->y ;

}

private:

int x, y;

friend class Cursor;

};

#endif // CURSOR\_H

#ifndef TOOL\_H

#define TOOL\_H

void SetCursorPosition(int x,int y);

void Setcmdsize(int wide,int high);

void Setcolour(int colour);

void Setbackcolour();

void Clear\_colour();

void HideCursor();

void Clearpoint(int x,int y);

#endif // TOOL\_H

#include "Tool.h"

#include<windows.h>

#include<conio.h>

#include<bits/stdc++.h>

/\*

Setcmdsize(50,30);

\*/

void SetCursorPosition(int x, int y)

{

COORD pos = { x, y };

SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), pos);

}

void Setcmdsize(int wide, int high)

{

system("title SNAKE from LC 5120154145");

char cmd[30];

sprintf(cmd, "mode con cols=%d lines=%d", wide \* 2, high);

system(cmd);

}

void Setcolour(int colour)

{

HANDLE hOut;

hOut = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hOut, colour);

}

void Clear\_colour()

{

HANDLE hOut;

hOut = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hOut, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);

}

void Setbackcolour()

{

HANDLE hOut;

hOut = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hOut, BACKGROUND\_RED | BACKGROUND\_GREEN | BACKGROUND\_BLUE);

}

void HideCursor()

{

CONSOLE\_CURSOR\_INFO cursor\_info = {1, 0}; //后边的0代表光标不可见

SetConsoleCursorInfo(GetStdHandle(STD\_OUTPUT\_HANDLE), &cursor\_info);

}

void Clearpoint(int x, int y)

{

SetCursorPosition(x, y);

std::cout << " " ;

}