**实 验 报 告**

**课程名称：C语言及程序设计基础**

**专业及班级：2021级网安6班**

**指导教师：傅建明**

**小组成员：**

**赵伯俣 2021302181156**

**刘竞优 2021302181057**

**刘 宇 2021302181153**

**小组分工：**

**刘 宇： move，keydown，createfood函数的编写以及bug的修改。**

**赵伯俣：drawmap，gotoxy，failure函数以及主函数的编写，游戏bug的修改。**

**刘竞优：menu，die函数的编写，以及游戏bug的修改。**

**提交时间：2021年12月11日**

目 录

**一、实验目的**

**二、实验设备（环境）及要求**

**三、实验内容与步骤**

1.问题描述与分析

2.模块划分

3.流程设计

4.核心代码解读

**四、实验结果测试**

1.简单的用户使用说明。

2.主要操作界面和结果的演示和截图

**五、分析与讨论**

1.设计、编码过程中遇到的问题记录、解决方法

**一、实验目的**

1. 夯实C语言学习成果，掌握函数调用，循环等结构，递归，指针，数组，结构变量等知识，通过具体程序的编写熟悉C语言的使用。
2. 提高利用C语言解决实际问题，将自然语言转化为编程语言的能力。
3. 加深对程序设计思想的理解，致力于建设更加完备的程序设计思维模式。
4. 通过设计耳熟能详的小游戏的程序，增强学习的趣味性和自主性，培养自己从程序出发思考互联网中简单问题的能力。

**二、实验设备（环境）及要求**

1. Windows 10系统电脑
2. Dev-C 5.11

**三、实验内容与步骤**

1. 问题描述与分析：

（1）游戏设计：在经典模式之外增加了穿墙模式和困难模式，通过不同模式的选择，提高了游戏的趣味性。

（2）地图绘制：利用循环在四周生成作为“墙”的方块，以此框定贪吃蛇的移动范围。

（3）蛇的移动：通过删除蛇尾，改变蛇各部分的坐标，并重新打印蛇头，形成蛇向指定方向移动的效果。

（4）生成食物：通过随机函数生成坐标并打印，同时限制食物不会刷新在边框或蛇各部分所在位置。

（5）吃到食物：即当蛇头坐标与食物坐标重合，此时删除该“食物”，重新生成新的食物并增加蛇的长度。同时通过参数判断是否吃到食物，进行后续的程序运行。

（6）穿越墙壁：在穿墙模式中，不生成左右边框，当蛇头移动到左右边框时，将蛇头坐标移动至同行的另一侧，实现穿墙。

（7）蛇的加速：在困难模式中，当蛇长度增加时，通过sleep函数实现蛇本身的速度同步增加，提高游戏难度。

（8）游戏结束：当蛇头移动至四周边框或蛇本身时，游戏结束并结算游戏分数。在结束条件以前，蛇的运动将无限循环。

2.模块划分：

（1）gotoxy函数：将光标移动至地图中的指定位置。

（2）drawmap函数：根据不同模式绘制不同的游戏地图。

（3）menu函数：提供游戏开始和模式选择界面，设置字体、各个色块颜色。

（4）keydown函数：读取用户输入按键。

（5）move函数：改变蛇的位置和运动方向，形成移动的效果。

（6）createfood函数：随机生成“食物”。

（7）die函数：判断游戏是否结束。

（8）failure函数：输出游戏结束界面，结算游戏分数。

开始

3.流程设计：

玩家选择游戏模式

根据用户选择制作地图、食物

用户通过键盘控制蛇的移动

否

判断是否吃到食物

是

否

制作新食物，记录分数

判断是否死亡

是

游戏结束，显示分数

4.核心代码解读

（1）void drawmap()创造地图函数

{

经典模式和困难模式生成四周边框。

If（用户选择经典模式或困难模式）

{

for(循环MAPWIDTH-2次)

{

将光标移动至上方边框的指定位置

printf(■充当的墙体)

将光标移动至下方边框的指定位置

printf(■充当的墙体)

}

for(循环HEIGHT次)

{

将光标移动至左侧边框的指定位置

printf(■充当左侧墙体)

将光标移动至右侧边框的指定位置

printf(■充当的右侧墙体)

}

}

else if（用户选择穿墙模式）{

穿墙模式只生成上下边框。

{

利用for循环和printf函数打印游戏的上下边框

}

while 循环

{

srand随机生成“食物的横纵坐标”。

if (食物的横坐标为偶数)

跳出函数（因为食物图形占用两个字符所以食物坐标为奇数则会出现只吃一半食物的情况）

}

将光标移到食物的坐标处打印由㊣代表的食物

初始化蛇的长度为3，速度为200。

在屏幕中间打印蛇头

生成初始的蛇身。

（2）void keydown( )

{

记录前一个按键的方向

if (如果用户按下了键盘中的某个键）

{

fflush(stdin)清空缓冲区的字符

}

if(玩家按错键而导致游戏异常）{

key=pre\_key;

}

如果当前移动方向和前一次方向相反的话，把当前移动的方向改为前一次的方向

重新规定key和pre\_key的值改变蛇的方向

}

（3）void move()

{

if (蛇没有吃到食物)

{

在蛇尾的坐标位置printf“ ”擦去蛇尾

}

for ( 循环蛇的长度-1次) {

将蛇的前一项坐标赋值给后一项坐标

}

switch (用户输入的蛇的移动方向)

{

判断蛇的移动方向。

}

打印出蛇头

由于目前没有吃到食物，changFlag值为0}

（4）bool die()

{

if (蛇头碰到上下边界）

返回false游戏结束

if(模式为经典模式和困难模式)

{

if (蛇头碰到左右边界)

返回false游戏结束

}

if(模式为穿墙模式)

{

if(蛇头碰到左边界)

{

将蛇头坐标移动到右边界

删除原来的蛇头

}

if(蛇头碰到右边界){

将蛇头坐标移动到右边界

删除原来的蛇头

}

}

for ( 循环蛇的长度次)

{

if (蛇头坐标与蛇尾坐标重合)

返回false游戏结束

}

如果蛇头没有碰到墙壁且没有触碰到自身，返回ture，游戏继续

}

**四、实验结果测试**

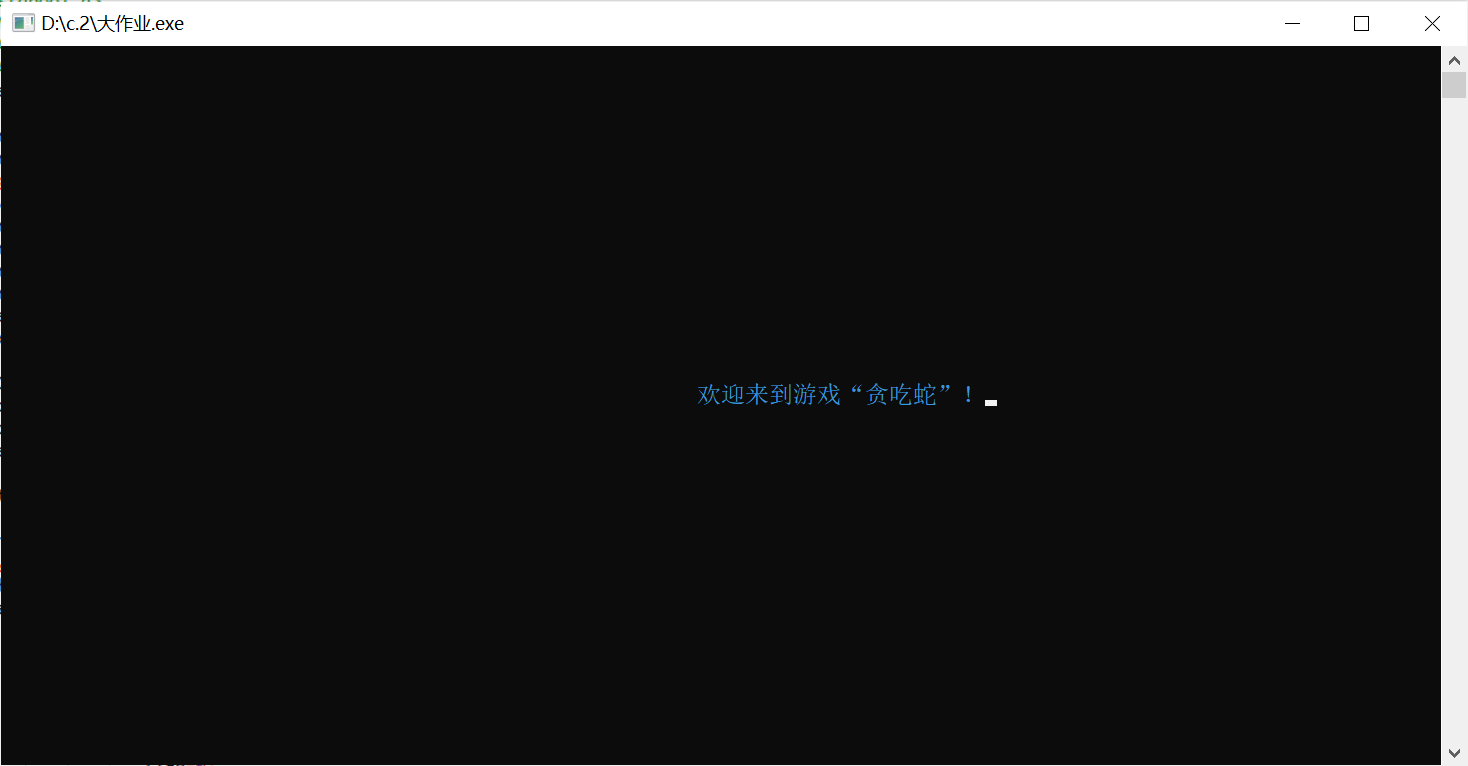
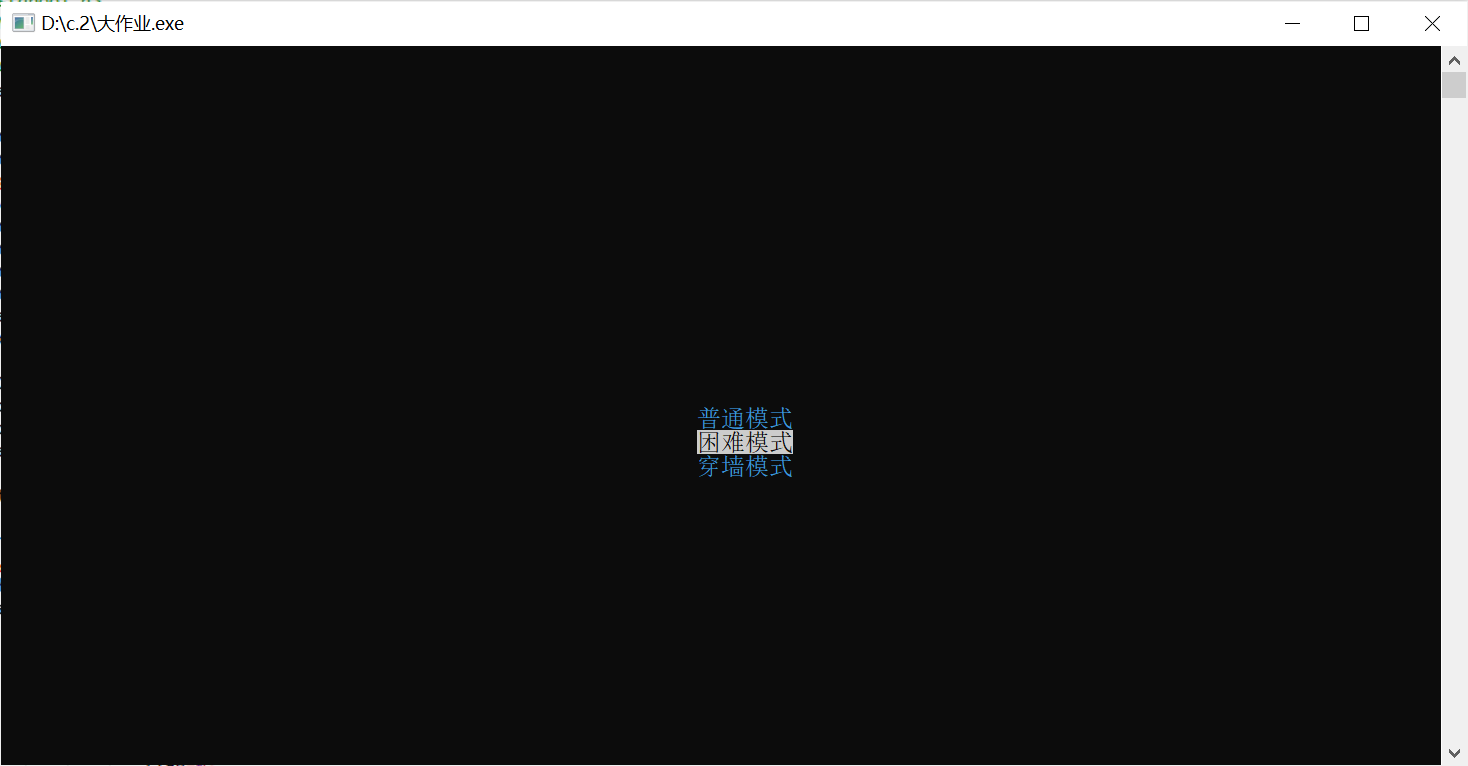


图1



**图2**

**用户使用说明：**打开程序，进入欢迎界面（图1），按任意键进入选择模式界面（图2），用户按PgUp/PgDn键选择自己喜欢的模式，按Enter键继续，显示该模式的说明（图3），按任意键继续，正式进入游戏（图4），游戏失败显示本次分数及是否重新开始游戏提示（图5）。图3、图4、图5提供困难模式的截图，若选择穿墙模式则为图6、图7，穿墙模式下用户可操作蛇穿过左右两侧墙体，以快速吃到食物。

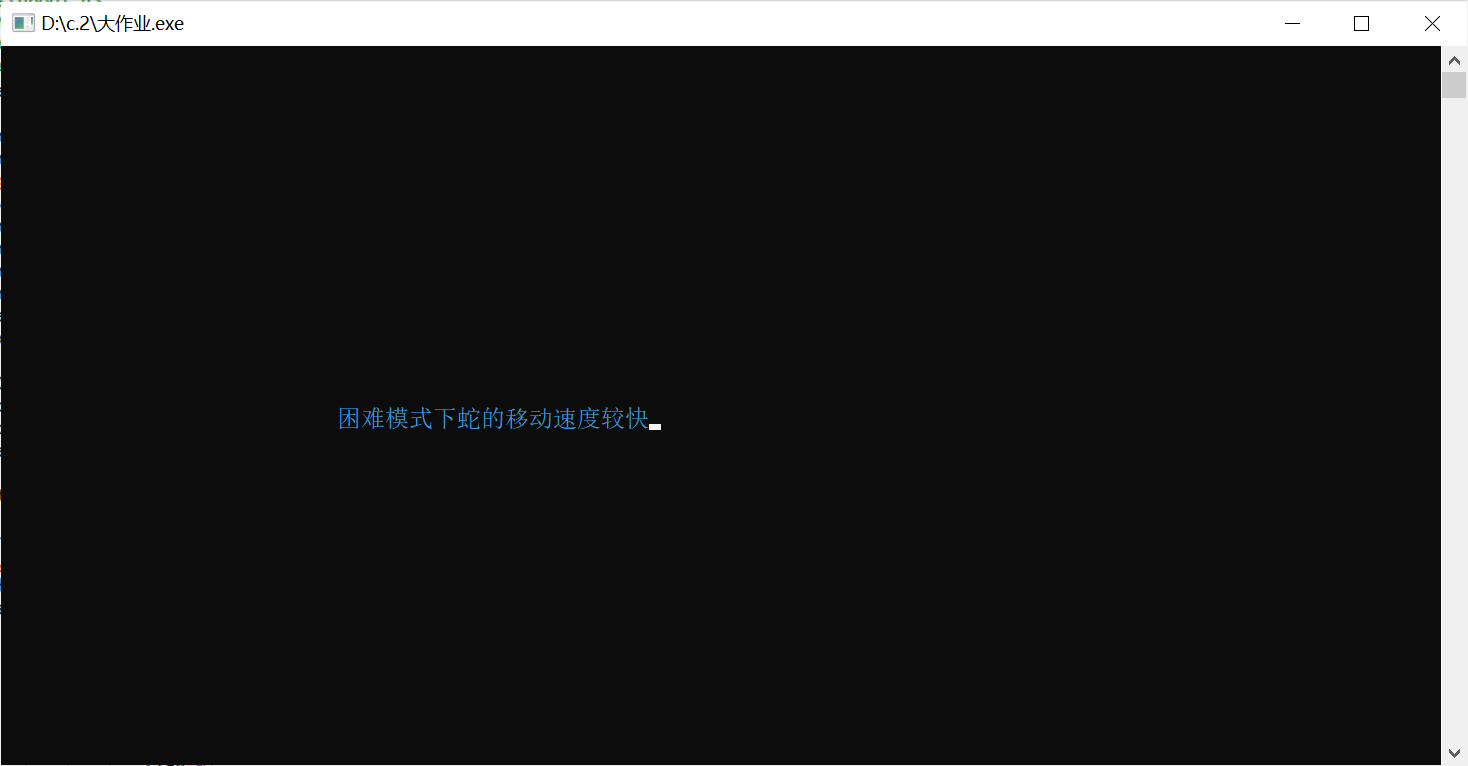


图3



**图4**

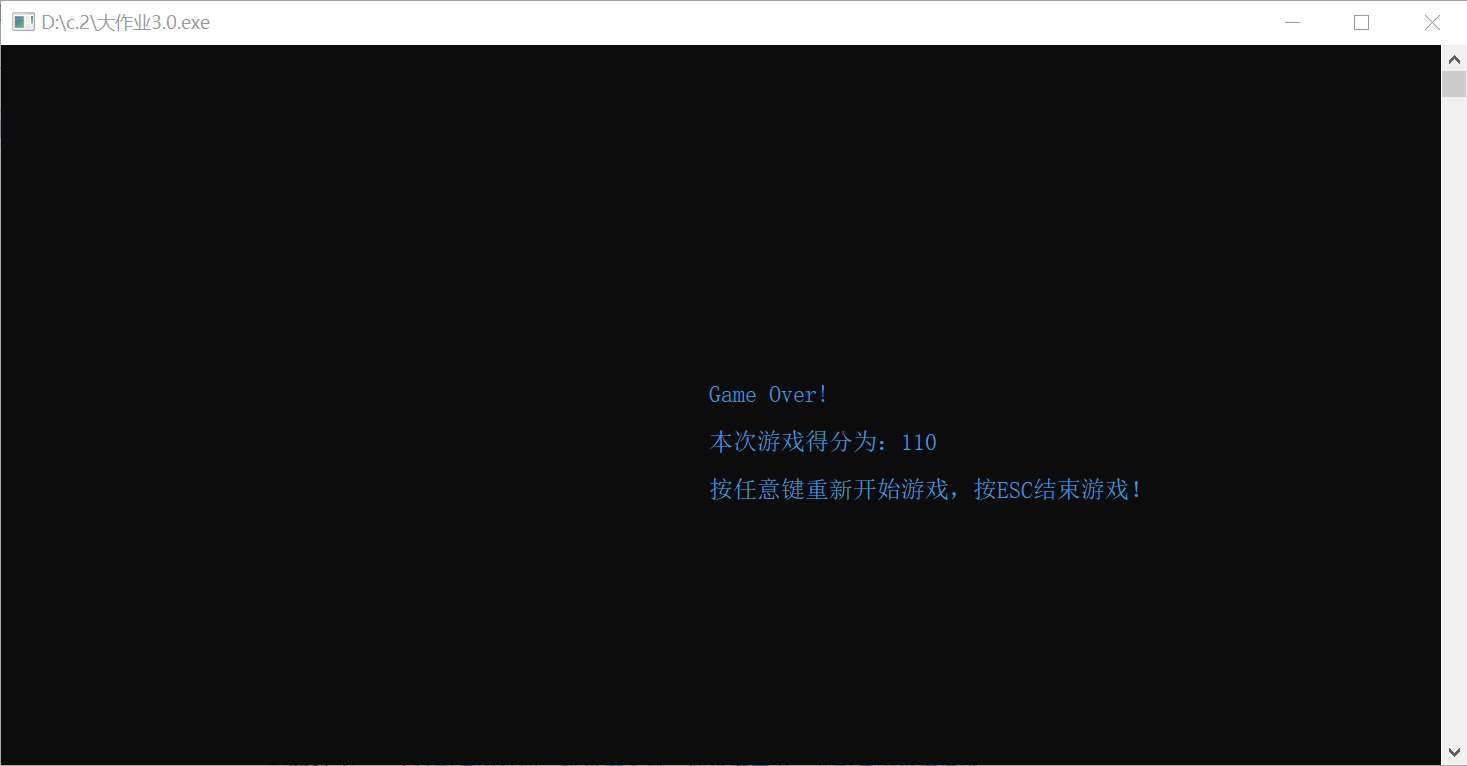


图5

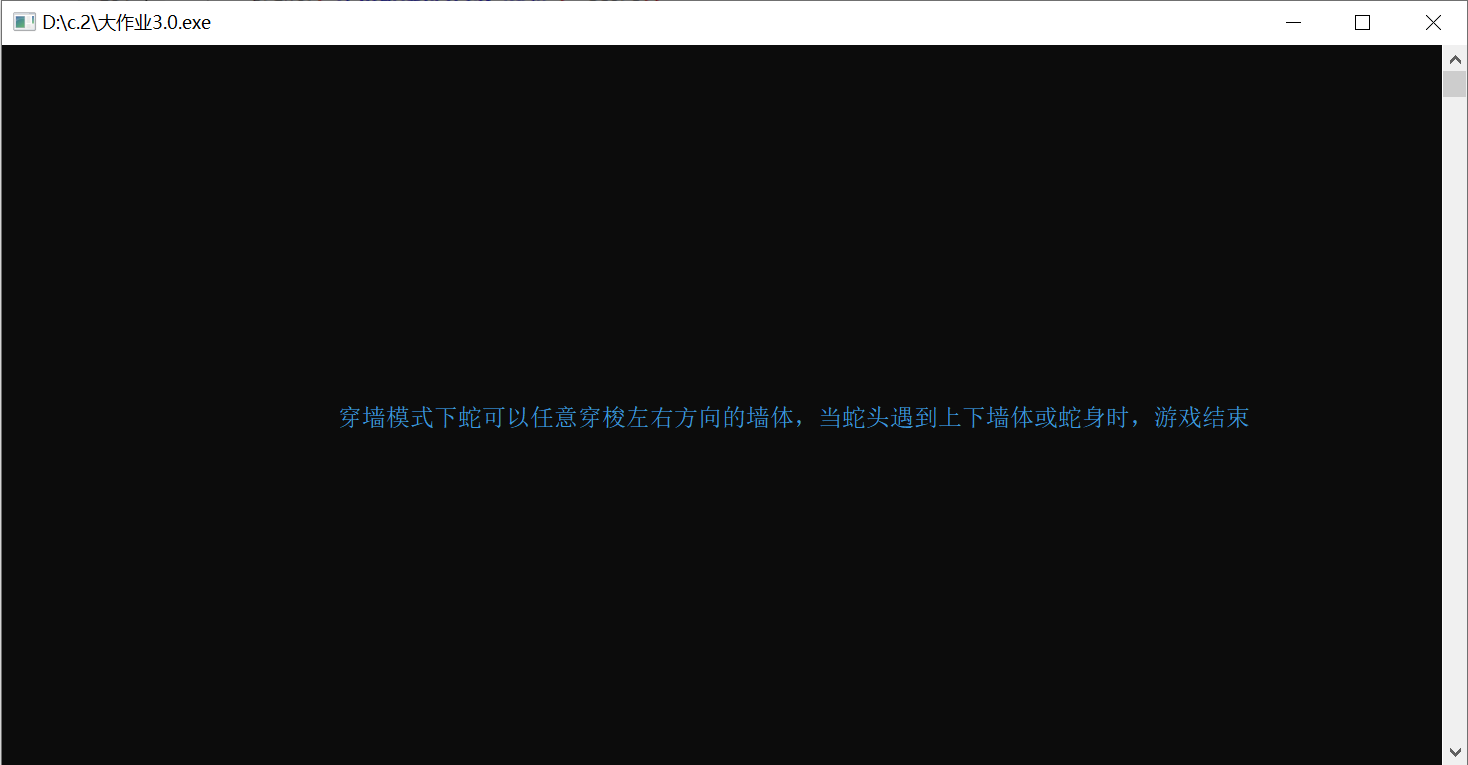


图6



**图7**

**五、分析与讨论**

通过该代码的编写，小组各成员都在应用中深化了所学知识的理解，同时有所拓展，学习了部分课外知识。期间遇到许多问题，并且解决了部分。

问题记录与解决方法

1. 未规定初始移动方向时，游戏会在生成蛇之后立即结束。具 体原因不得而知。

解决方法：规定蛇初始向上，即key=72.

1. 进入游戏界面后，如果放大界面会使地图打印出现错误。



在选择模式之前全屏并不会出现这种问题，而在生成地图时全屏生成的地图会混乱，尚未有有效的办法解决。

1. 在初期的穿墙模式中，穿过之后会留下一个方块在最后的位置，蛇头在碰到该方块时不会结束游戏。

解决方法：穿过之后将光标移到同列左右边缘处，打印两个空格覆盖掉留存的方块，形成穿过的效果。



4）在穿墙模式中穿过的位置始终有一格的空隙

因为在上一个问题的解决中，用空格覆盖了原本留在边缘位置的方块，所以在穿过时形成了间隔的效果，没有找到更好的解决办法。

5）地图大小无法改变，不能实时显示所得分数。

本想通过压缩地图留出位置显示分数，但是在改变地图数据后出现了打印混乱，即第二个及之后的食物无法成功显示，分析猜测是刷新在了地图之外或地图边缘，但没有取得有效的解决办法。

6）首次成功运行程序编译十次平均时间为0.89s随后对函数内循环进行优化，实现循环嵌套，使用全局变量减少函数调用参数，嵌套if语句，最终版本编译十次平均时间为0.78s

7）若使用递归函数生成地图则编译时间过长，编译十次平均时间0.92s，若使用循环则编译较快，编译十次平均时长0.78s。

六、教师评语

签名：

日期：

成绩：