成绩	

淮南師範學院

课程设计报告

题　　目：C 语言编程实现贪食蛇游戏

学生姓名：赵伟

学生学号：0908040246

系　　别：电气信息工程学院

专　　业：通信工程

届　　别：2013 届

指导教师：王丽

电气信息工程学院制

2012 年 5 月

C 语言编程实现贪食蛇游戏

学生：赵伟
指导教师：王丽
电气信息工程学院

1 课程设计的任务与要求

1.1 课程设计的任务

通过对 C 语言编写程序实现贪食蛇游戏系统。掌握结构化，模块化程序设计的思想，培养 C 语言编程编写实战能力。

1.2 课程设计的要求

先在程序中设计好数组元素与蛇，食物的对应关系；

产生一个固定大小有边界的游戏区域，蛇从区域中随机一点出发，运动限制在游戏区域内；

蛇的运动方向为直线运动，只走横和竖的方向，不走斜线；

食物出现按随即分布原则，蛇吃掉一份后随即在游戏区域内出现一份新的食物；

蛇的运动速度由得分来控制，得分越高，速度越快；

得分按蛇每吃掉一个食物得 10 分计算；

蛇的身体长度从 3 开始，每吃掉一个事物增加一个长度；

游戏结束条件为：在控制过程中蛇头撞到墙壁或者与蛇身相撞。

1.3 课程设计的研究基础

课程设计基础为 C 语言程序的设计与编写。

C 语言是在国内外广泛使用的一种计算机语言，它具有高级语言的特点，有具有汇编语言的特点。

C 语言功能丰富、表达力强、使用灵活方便、应用面广、目标程序高、可移植性好，既具有高级语言的优点，有具有低级语言的许多特点，因此特别适合于编写系统软件，三维，二维图形和动画，具体应用比如单片机以及嵌入式系统开发，著名的 UNIX 操作系统就是用 C 语言编写的[1]。

C 语言是由玫瑰贝尔研究所的 D.Mritchie 于 1972 年推出，1978 年后，C 语言已经先后被移植到大、中、小及微型机上，它可以作为工作系统设计语言，编写系统应

用程序，也可以作为应用程序设计语言，编写不依赖计算机硬件的应用程序[2]。

我们利用电脑或者游戏机玩的游戏都可以利用 C 语言来进行编写，大部分也都是用 C 进行编写的，其中 2D 的贪食蛇游戏是利用编写的经典案例[3]。

2 贪食蛇系统方案制定

2.1 方案的提出

方案一：利用 C 语言编程实现贪食蛇游戏

流程图如下

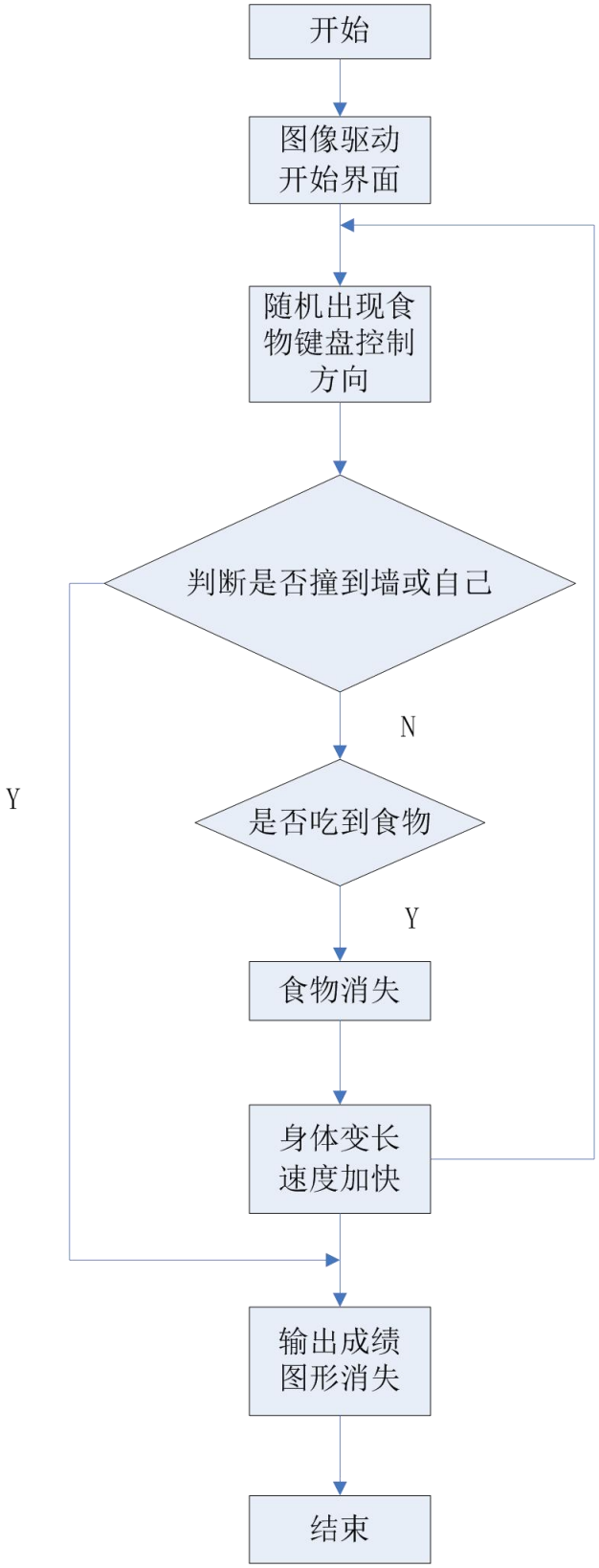


图 1 C 语言编写贪食蛇流程图

方案二：利用 JAVA 语言编程实现贪食蛇游戏[4][5]

流程图如下

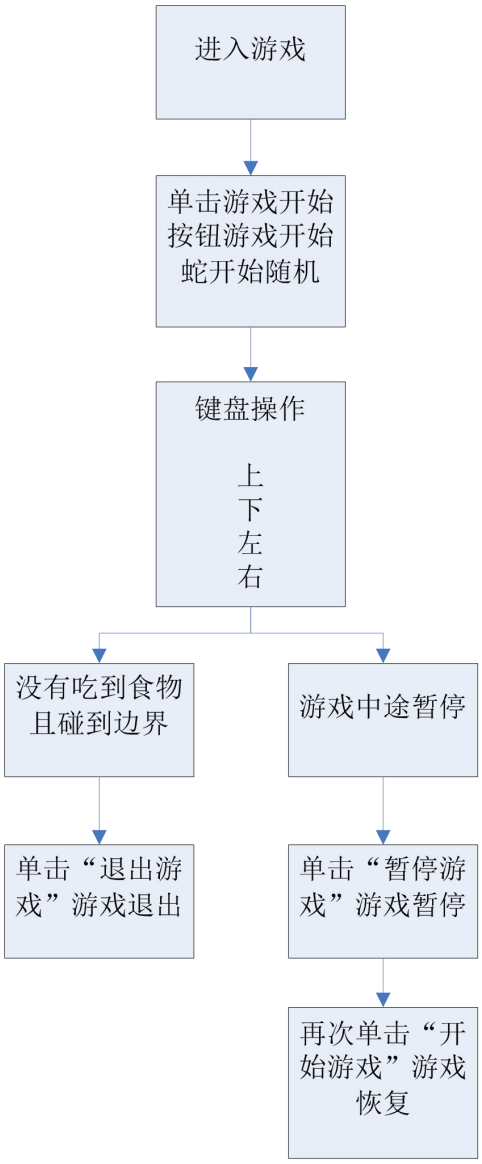


图 2 利用 Java 编写贪食蛇流程图

2.2 方案比较

通过流程图可以看出，方案一与方案二的区别在于游戏可否中途退出和暂停，这一模块牵扯到很多的内容。

2.3 方案的论证

利用 Java 设计比较复杂，它涉及面广，牵涉方面多，如果不好好考虑和设计，将难以成功开发出这个游戏。在这个游戏的设计中，牵涉到图形界面的显示与更新，数据的收集与更新。而且在这个游戏的开发中，还要应用到类的继承机制以及一些设计模式。在设计开发过程中，需要处理好各个类之间的继承关系。还要处理各个类相

应的封装，并且还要协调好各个模块之间的逻辑依赖关系和数据通信关系[6]。而利用 C 语言则比较方便，并且我们对 C 语言的熟悉度比较高，所以，我们选择了利用 C 来开发这个游戏。

2.4 方案的选择

通过比较，得出利用 JAVA 实习目标较复杂。且相比较而言。我们更熟悉 C 编程，从而选择方案一，利用 C 语言来实现设计的目标。

3 系统方案设计

3.1 各单元模块的划分与功能介绍

本程序采用结构化程序设计的方法，按照自顶向下，逐步细化的方法对要解决的问题进行逐层分解。首先画出顶层模块，即主控模块，之完成对下层模块的调用功能，即调用其他的功能模块；接着，按需求分析中的功能需求设计第一层模块，有图形驱动，画主界面，游戏过程，结束处理，退出等第 6 个主要功能；接着，画出第二层模块[7][8]。总体模块结构如图

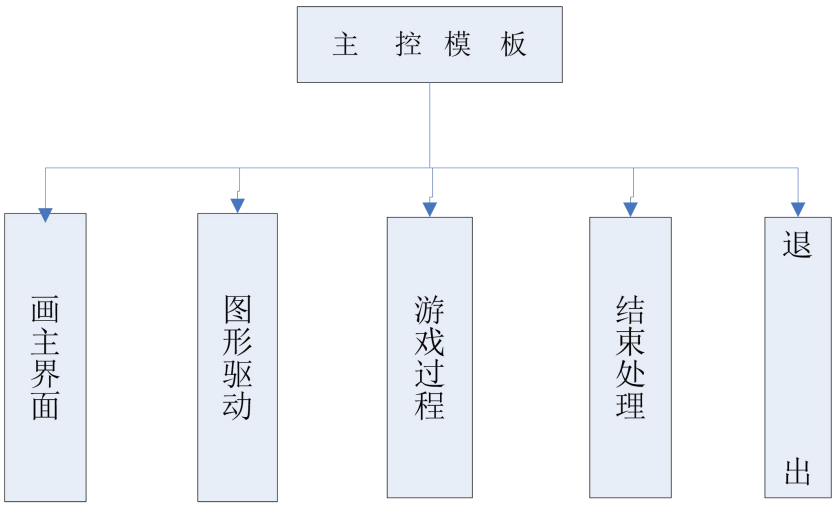


图 3 总体模块结构图

3.2 总体数据结构设计

设计思路：测序的关键在于表示蛇的图形及蛇的移动。用一个小局限性方块表示蛇的一节身体，身体每长一节，增加一个矩形块，蛇头业用同样的一节小矩形方块表示移动时必须从蛇可以上向前爬行，档案下有效方向键后，应先确定蛇头的位置，而后蛇的身体虽蛇头移动，图形的实现是从身体新位置开始画出蛇。这时，由于未清屏的原因，原来的蛇的位置和新蛇的位置差一节蛇身，即看起来蛇多一节身体，所以

将蛇的最后一节用背景色覆盖。食物的出现与消失意识画矩形块和覆盖矩形块。为了便于理解，定义两个结构体：食物与蛇。下面介绍贪吃蛇游戏程序的主要数据结构。

3.3 主要函数的介绍

① 主控模块 main 函数

主函数是程序的主控模块。首先初始化图形系统，然后使用 draw 函数播放动画，接着调用 init 函数[9]初始化图形系统，之后调用 drawk 函数[9]画出游戏开始画面，在调用 gameplay 函数[9]，即开始了游戏的具体过程，游戏结束后调用 endplay 函数[9]进行游戏结束处理；最后关闭图形系统，结束程序。主控模块的流程图如下图所示。其中带有两个竖线的矩形框表示对自定义函数的调用。

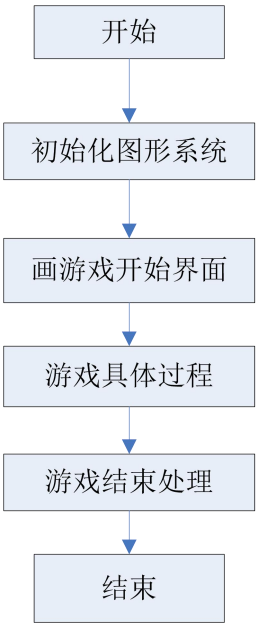


图 4 main 函数模块

② 绘制游戏开始界面 drawk 函数

主界面就是一个封闭的围墙，用两个循环语句分别在水平方向和垂直方向输出连续的矩形小方块，围成封闭图形表示围墙，在界面的左上部输出游戏程序的版本信息，在右上部输出游戏成绩（score）的表头。

3.4 系统整体程序设计

贪食蛇游戏具体实现过程 gameplay 函数，该函数是游戏的核心部分。游戏具体过程 gameplay 函数的大致算法流程图如下图所示。

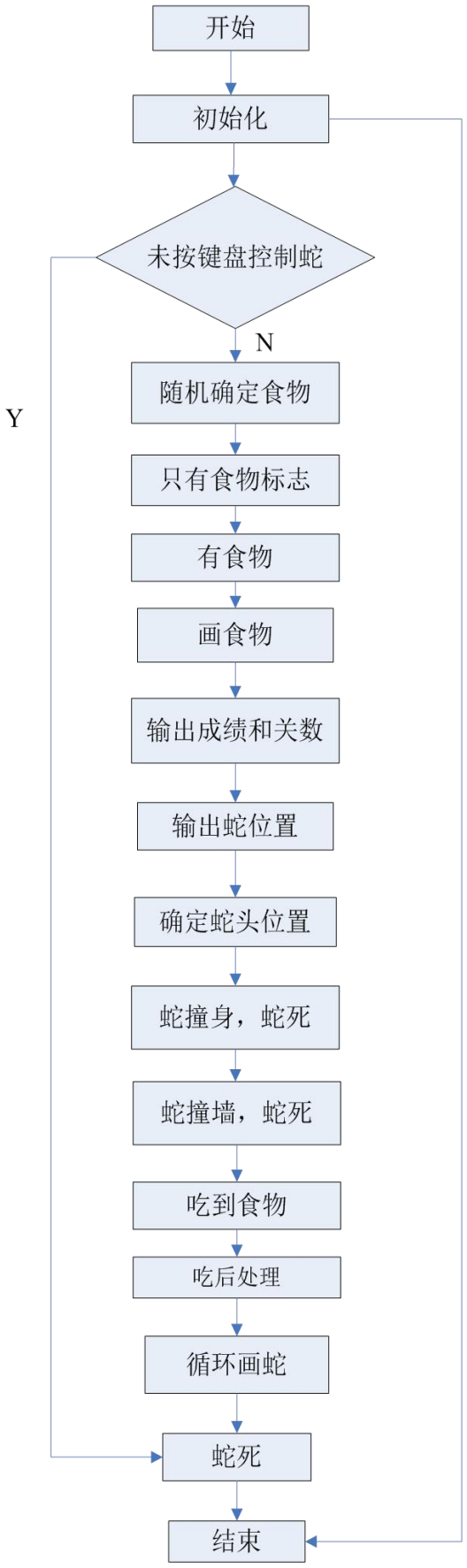


图 5 gameplay 函数流程图

① 初始化

为防止食物总是出现在一个位置上，要设置随机数发生器的种子数，产生真正的随机数。由于还没有画出食物，并设置需要食物，并设置蛇活着。初始时，蛇只有蛇头和 1 节蛇尾，设置这 2 节坐标。设定蛇开始的爬行方向左右。

② 随机确定食物位置

由于蛇吃到食物的判断是蛇头的坐标和食物的坐标相等，所以要确保食物出现的位置在 10 的倍数位置上。先用两个带随机函数的表达式产生一个位于围墙内的 x、y 坐标，然后用两个 while 循环[10]将两个坐标值调整到 10 的倍数上，这样就可以让蛇吃到。

③ 循环确定蛇身的新坐标

这里的难点是表示蛇的新位置并消除前一次的图形。采用的方法是每次移动的时候从最后一节开始到第二节，将前一节的坐标值赋给后一节的坐标，移动后只要把最后一节用背景色擦出即可，因为新位置 0 到 n-1 节还是要出现在画面上的。这里用一个 for 循环来确定蛇身的新坐标。

④ 吃到食物后的处理

蛇吃到食物后，首先将食物擦除，即用背景色画出该食物，然后给蛇的节数加 1，设置需要食物标志，是游戏成绩加 10 分，如果成绩达到 100 分的倍数，则给关数加 1，并加快游戏速度。

⑤ 有按键判断蛇的方向

这是一个内嵌的嵌套的条件选择结构，根据按动上下左右键来设定蛇的移动方向。判断还需考虑相反的方向键无效，比如蛇正在向上爬行，按下一键方向是无效的。

4 系统仿真与调试

4.1 仿真软件介绍

Win-TC 软件是一款用于编写 C 语言程序的软件，是 TC2 WINDOWS 平台开发的一个工具，是 Turbo C 2.0(简称 TC2.0)的一种扩展形式，是在 TC2.0 的基础上，增强了系统的兼容性和共享性，允许进行复制粘贴的多项可以用鼠标来操作的功能，比 TC2.0 使用起来方便。该软件使用 TC2 为内核，提供 WINDOWS 平台的开发界面，因此也就支持 WINDOWS 平台下的功能，例如剪切、复制、粘贴和查找替换等。而且在功能上也有它的独特特色例如语法加亮、C 内嵌汇编、自定义扩展库的支持等。它的主要特点如下：

1）在 WINDOWS 下编辑 TC 代码，可以充分利用 WINDOWS 的支持剪贴版和中文的特点；

2）Include 和 Lib 路径自动定位，告别 TC 设置路径的历史；

3）编译错误捕捉功能，编译时不会出现烦人的 DOS 窗口；

4）支持 C 内嵌汇编从而实现 C/ASM 混合编程；

5）支持 C 扩展库（自定义 LIB 库）；

6）允许自定义设置输入风格，能够实现 VC 类似的输入风格；

7）错误警告定位功能、出现编译错误时双击输出框里的出错行信息可以自动寻找到错误的行，就像 DOS 的 TC 那样；

4.2 系统仿真实现

在 window 环境下装好 win-TC 之后，打开 win-TC 并新建文件夹，把贪食蛇游戏的源程序输入进去，然后运行，并发现错误，修改之后，把文件保存在软件的源文件里面，也就是 project 文件夹里。之后就可以成功运行贪食蛇游戏了。

4.3 系统测试

预期结果：蛇运行时如果撞到墙壁，则游戏结束。

实际运效果：与预期结果一致，如下图

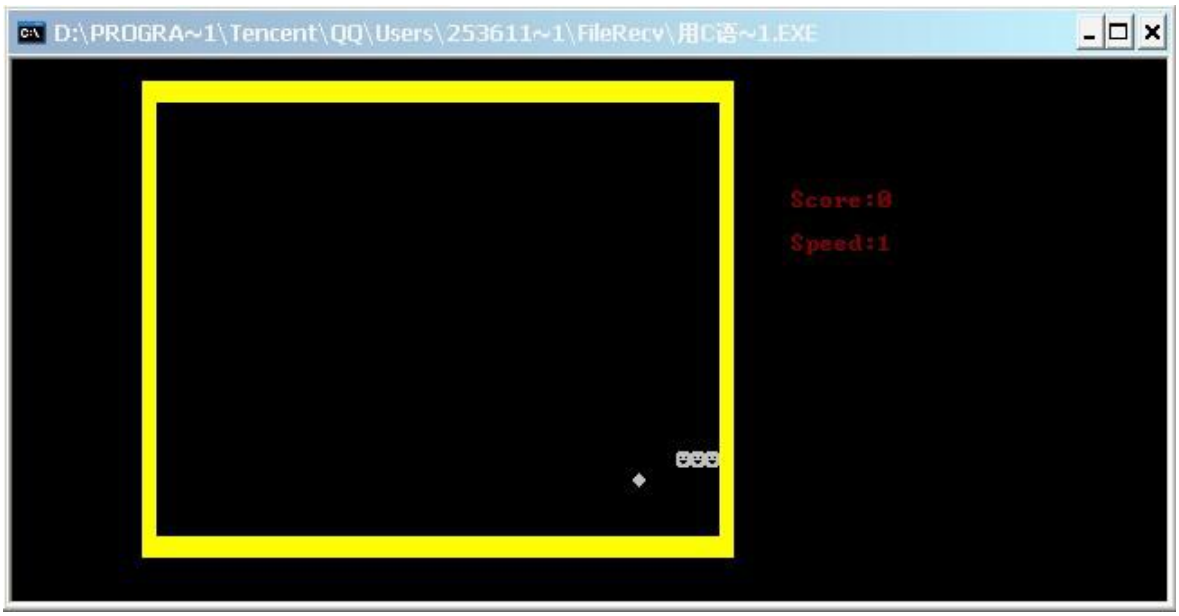


图 6 贪吃蛇撞到墙壁

图 7　游戏结束

预期的结果：当蛇得头部与蛇身相撞时，游戏应当结束。

实际运效果：与预期结果一致，如下图

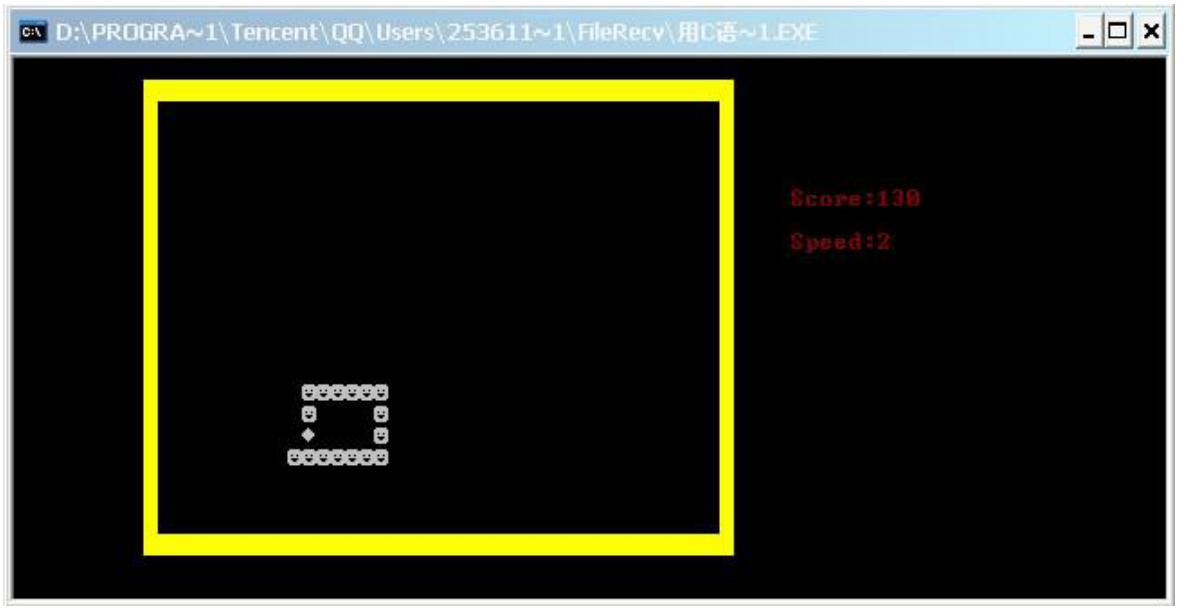


图 8 贪吃蛇头部与身体相撞

图 9 游戏结束

4.4 数据分析

经过对各个模块的调试和单元测试并修改了错误后，将各个模块组装成一个系统，并进行集成测试。在集成的过程中发现了一些错误，比如全局变量重复定义、函数重复声明等，分别进行了改正。最后连续运行了 5 次，以便测试该游戏程序的功能、性能是否达到了预期的目标，并根据所使用的具体机器对蛇的爬行速度进行适当的调整。测试内容包括蛇的运行、分数、关数、动画等功能，情况均正常。

5 总结

5.1 设计小结

通过 2 周的课程设计，在导师的指导和开发小组共同的努力下，终于完成了贪吃蛇游戏程序的设计与编写。该游戏程序实现了用方向键控制蛇在围墙内爬行、随机产生食物、控制蛇吃掉食物并加分、控制游戏中闯关的数目和蛇的运行速度、实时显示得分和关数、结束时处理和显示排行榜等功能。该游戏程序具有界面友好、操作方便、控制准确和容错能力强等特点。

然而这个贪吃蛇游戏还存在一些缺陷，应该在如下几个方面加以改进。

1）进入游戏后，到蛇死亡，只能玩一次就会退出程序。应加入多次游戏的控制能力。

2）蛇的美观不足，需将蛇头、蛇尾及蛇身做进一步美化，使其更像真实的蛇。

3）当分数超过 350 分后，蛇的速度变得非常快，一下子就撞到围墙上了，应适当控制蛇的速度和关数，以便使游戏更具有吸引力。

5.2 收获体会

贪吃蛇的设计与编写对我们有很多帮助，学习编写贪吃蛇的游戏对掌握 C 语言的知识有很大的帮助。通过编程实践，还能拓展思路，让我们去寻找需要调用那些函数，怎样提高程序的质量等。

要写出好的程序，需要我们有扎实的基础，这样遇到一些基本算法的时候就会游刃有余了。在编程时我们要有丰富的想象力，不要拘泥与固定的思维方式，遇到问题的时侯要多想几种解决问题的方案。丰富的想象力是建立在丰富的知识基础上，所以我们要通过多个途径来帮助自己建立较丰富的知识结构。

贪吃蛇游戏的编程练习思考数据结构:定义食物的坐标来控制它出现的位置，用一系列的函数时进行表示，比如用函数 rectangle 来画出矩形，用 life 变量的值表示蛇的生命，用 direction 变量的值表示蛇移动的方向等，还有用数组来存放蛇身各节的坐标，这些都让我们熟悉了对数组的操作，此外还熟悉了各种函数的应用。

在编程是我们碰倒了很多的困难，这就需要我们多与别人交流。在编程的过程中，我们也发现有良好的编程风格是十分重要的，至少在时间效率上就体现了这一点。养成良好的习惯，代码的缩进编排，变量的命名规则要始终保持一致，这些都是提高我们编程能力的要点。

在进行课程设计的过程中我们也学到了许多别的东西。首先，我们学会了合作，要以别人的眼光看看问题，也许这样得到的会比各自得到的都要多；其次，我们学会了分工，分工是为了更好地合作，分工才能提高合作的效率；最后，我们学会了奋斗，我们相信，通过四年的学习，我们一定能写出更精彩的程序，将来会描绘出更精彩的人生。

在这里，我们要感谢课程设计指导老师王丽老师给予我们悉心的指导。老师多次询问编写进程，并为我们指点迷津，帮助我们开拓研究思路，精心点拨、热心鼓励。老师一丝不苟的工作作风，严谨求实的态度，踏踏实实的精神，不仅受我以文，而且教我做人，给以终生受益无穷之道。还有我们设计小组之间的团结与努力，正是由于我们团结协作，才顺利的完成了课程设计任务。

5.3 展望

通过这次的课程设计，使得我们对 C 语言有了一个更深层面的了解与认识。从一开始的对待新事物的恐惧与无从下手到最后逐渐产生了兴趣。我觉得这次课程设计不单是考验了我们对 C 语言的掌握，更是对我们的团队协作能力的一个挑战。我们

希望以后还可以更加进步。

6 参考文献

[1] 谭浩强. 程序设计（第三版）[M]. 北京：清华大学出版社，2005.

[2] 王成瑞，魏先民. 语言程序设计实训[M]. 中国水利水电出版社，2005.

[3] 谭浩强. 程序设计题解与上级指导（第三版）[M]. 北京：清华大学出版社,2005.

[4] 王路群. Java 高级程序设计[M]. 北京：中国水利水电出版社，2006.

[5] 陈轶，姚晓昆编著. Java 程序设计实验指导[M]. 北京：清华大学出版社，2006.

[6] 施宏斌译. JavaScript 入门经典（第 3 版）[M]. 北京：清华大学出版社，2009.

[7] 谭浩强著. C 程序设计（第二版）[M]. 北京：清华大学出版社，1999.

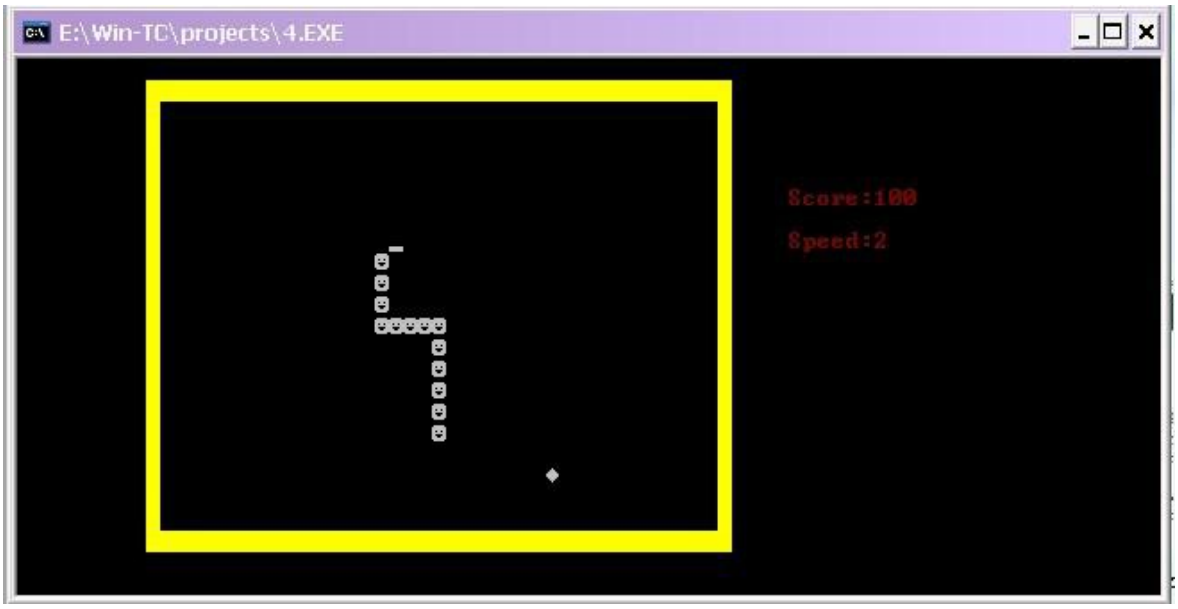
[8] 谭浩强，张基温，唐永炎编著. C 语言程序设计教程[M]. 北京：高等教育出版社，1992.

[9] 戴健鹏译. C 语言大全（第二版）[M]. 北京：电子工业出版社，1994.

[10] C 编写组编. 常用 C 语言用法速查手册[M]. 北京：龙门书局，1995.

7 附录

系统主要功能展示图



元器件清单

Win-TC 软件，PC 机

贪吃蛇程序源码：

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
#include <stdlib.h>
#include <time.h>
```

```
/*蛇的最大节数*/
#define MAX 200
#define LEFT 10
#define TOP 2
#define RIGHT 50
#define BOTTOM 23
#define TIME 0x1c /*时钟中断点*/

#define VK_UP 0x4800
#define VK_DOWN 0x5000
#define VK_LEFT 0x4b00
#define VK_RIGHT 0x4d00
#define ESC 0x11b

#define randx() (rand()%(RIGHT-LEFT-1))+LEFT+1
#define randy() (rand()%(BOTTOM-TOP-1))+TOP+1

int iScore=0;       /*分数*/
int iTimeCount=0;/*记录时钟中断*/

/*定义蛇节的结构*/
  struct snakenode
{
int x;
int y;
};

/*定义蛇的结构*/
typedef struct
{
```

```
struct snakenode s[MAX];
int iLength;       /*蛇的长度*/
int iDirection;/*蛇移动的方向*/
}SNAKE;

/*定义食物的结构*/
typedef struct
{
int x;
int y;
}FOOD;


/*定义蛇变量*/
SNAKE snake;
/*定义食物变量*/
FOOD food;


/*保存旧的时钟中断*/
void interrupt (*oldtime)();


/*新时钟函数*/
void interrupt newtime()
{
iTimeCount++;
oldtime();
}
```

```
/*设置时钟中断*/
void SetTime(void interrupt (*pInterrupt)())
{
oldtime=getvect(TIME);
disable();
setvect(TIME,pInterrupt);
enable();
}

/*恢复以前的时钟中断*/
void KillTime()
{
disable();
setvect(TIME,oldtime);
enable();
}


void DrawFrame()/*画边框*/
{
int x,y;
textcolor(YELLOW);
for(x=LEFT;x<RIGHT+1;x++)
{
gotoxy(x,TOP);
cprintf("%c",219);
gotoxy(x,BOTTOM);
cprintf("%c",219);
}
for(y=TOP;y<BOTTOM+1;y++)
```

```
{
gotoxy(LEFT,y);
cprintf("%c",219);
gotoxy(RIGHT,y);
cprintf("%c",219);
}
textcolor(WHITE);
}

/*显示分数与速度*/
void DrawInfo()
{
textcolor(RED);
gotoxy(RIGHT+5,TOP+5);
cprintf("Score:%-5d",iScore);
gotoxy(RIGHT+5,TOP+7);
cprintf("Speed:%-2d",iScore/10+1);
textcolor(WHITE);
}

/*产生贪吃蛇*/
void CreateSnake()
{
int i;
int x,y;
x=randx();
y=randy();
for(i=0;i<4;i++)
```

```
{
snake.s[snake.iLength].x=x+i;
snake.s[snake.iLength].y=y;
snake.iLength++;
}
snake.iDirection=2;/*蛇向左移*/
}

/*显示贪吃蛇*/
void ShowSnake()
{
int i;
int length=snake.iLength;
for(i=length-1;i>0;i--)
{
gotoxy(snake.s[i].x,snake.s[i].y);
printf("%c",2);
}
gotoxy(snake.s[0].x,snake.s[0].y);
printf("%c",' ');
}

/*检查产生食物是否在贪吃蛇身上*/
int IsInSnake()
{
int i;
int flag=0;/*食物不在贪吃蛇身上*/
int length=snake.iLength;
for(i=0;i<length;i++)
{
```

```
if((snake.s[i].x==food.x) && (snake.s[i].y==food.y))
{
flag=1;/*食物在贪吃蛇身上*/
break;
}
}
return flag;
}

/*产生食物*/
void CreateFood()
{
int x,y;
int flag=1;
while(1==flag)
{
x=randx();
y=randy();
food.x=x;
food.y=y;
if(0==IsInSnake())/*食物不在贪吃蛇身上*/
{
flag=0;/*退出循环*/
}
}
}

/*显示食物*/
void ShowFood()
{
```

```
gotoxy(food.x,food.y);
printf("%c",4);
}

/*游戏初始化*/
void InitGame()
{
/*初始化随机数组*/
srand((int)time(0));

/*初始化游戏时钟*/
SetTime(newtime);

/*绘制游戏边框*/
DrawFrame();

/*显示游戏分数与游戏速度*/
DrawInfo();

/*初始化蛇的位置与长度*/
CreateSnake();
/*显示贪吃蛇*/
ShowSnake();

/*初始化食物的位置*/
CreateFood();
/*显示食物*/
ShowFood();
}
/*按方向移动蛇*/
```

```
void MoveSnake()
{
int dir;
int length;
int i;
dir=snake.iDirection;
length=snake.iLength;
switch(dir)
{
case 0:/*左移*/
for(i=0;i<length-1;i++)
{
snake.s[i]=snake.s[i+1];
}
snake.s[length-1].x-=1;
break;
case 1:/*上移*/
for(i=0;i<length-1;i++)
{
snake.s[i]=snake.s[i+1];
}
snake.s[length-1].y-=1;
break;
case 2:/*右移*/
for(i=0;i<length-1;i++)
{
snake.s[i]=snake.s[i+1];
}
snake.s[length-1].x+=1;
break;
```

```
case 3:/*下移*/
for(i=0;i<length-1;i++)
{
snake.s[i]=snake.s[i+1];
}
snake.s[length-1].y+=1;
break;
default:
break;
}
}
/*蛇的长度增加*/
void IncreaseSnake()
{
int length;
int i;
snake.iLength++;
length=snake.iLength;
switch(snake.iDirection)
{
case 0:
for(i=length-1;i>0;i--)
{
snake.s[i]=snake.s[i-1];
}
snake.s[0].x-=1;
break;
case 1:
for(i=length-1;i>0;i--)
{
```

```
snake.s[i]=snake.s[i-1];
}
snake.s[0].y-=1;
break;
case 2:
for(i=length-1;i>0;i--)
{
snake.s[i]=snake.s[i-1];
}
snake.s[0].x+=1;
break;
case 3:
for(i=length-1;i>0;i--)
{
snake.s[i]=snake.s[i-1];
}
snake.s[0].y+=1;
break;
default:
break;
}
}
int IsGameOver()
{
int flag=0;
int length=snake.iLength;
int i;
int x=snake.s[length-1].x;
int y=snake.s[length-1].y;
```

```
/*是否出界*/
if(x<=LEFT || x>=RIGHT || y<=TOP || y>=BOTTOM)
{
flag=1;
}

/*判断是否吃到自己*/
for(i=0;i<length-1;i++)
{
if(snake.s[i].x==x && snake.s[i].y==y)
{
flag=1;
break;
}
}
return flag;
}
void PlayGame()
{
int i;
int key;/*保存按键值*/
int flag=0;/*游戏结束标志*/
while(!flag)/*游戏运行中*/
{
/*按下了一个键*/
if(bioskey(1))
{
/*获得按键的扫描码*/
key=bioskey(0);
switch(key)
```

```
{
case VK_LEFT:/*按下向左键*/
/*蛇向左走一步*/
if(2!=snake.iDirection)
{
snake.iDirection=0;
MoveSnake();
ShowSnake();
}
break;
case VK_RIGHT:/*按下向右键*/
/*蛇向右走一步*/
if(0!=snake.iDirection)
{
snake.iDirection=2;
MoveSnake();
ShowSnake();
}
break;
case VK_DOWN:/*按下了向下键*/
/*蛇向下走一步*/
if(1!=snake.iDirection)
{
snake.iDirection=3;
MoveSnake();
ShowSnake();
}
break;

case VK_UP:/*按下了向上键*/
```

```
/*蛇向上走一步*/
if(3!=snake.iDirection)
{
snake.iDirection=1;
MoveSnake();
ShowSnake();
}
break;
default:
break;
}
}
/*时钟中断*/
if(iTimeCount>18-2*iScore/100)
{
MoveSnake();
ShowSnake();
iTimeCount=0;
}
/*蛇吃到食物*/
if(1==IsInSnake())
{/*蛇的长度增加*/
IncreaseSnake();
ShowSnake();
/*分数增加*/
iScore+=10;
DrawInfo();
/*重新产生食物*/
CreateFood();
ShowFood();
```

```
}
/*游戏结束*/
if(1==IsGameOver())
{
flag=1;
KillTime();
clrscr();
printf("Game Over!\nThank you for the game!\n");
printf("Press any key to quit();\n");
getch();
}
}
}
int main()
{
InitGame();
PlayGame();
   return 0;
}
```