哈爾濱工業大學（深圳）
Harbin Institute of Technology Shenzhen

《创新训练课 B》课程设计报告

学院：机电工程与自动化学院

题目：贪吃蛇的设计

班级：自动化五班

姓名：葛旭 黄弋斌

学号：190320517 190320514

教师：吴晓军

上交日期：2020 年 10 月 31 日

修订历史记录

日期	版本	说明	作者
	v1.0	第一版仅运用不断 printf 在控制台进行操作，较卡顿。	黄弋斌
	V2.0	第二版用了 easyx 库，相对界面友好，流畅	葛旭

目 录

1 引言

1.1 编写目的

市面上有很多种贪吃蛇游戏，但是最原始的版本仍然是吃食物变长的版本，既然学了 c 语言，就充分发挥它的用处，不再仅仅局限于控制台应用，而是创造一个 GUI 界面，让自己发挥创造力，编写自己想要的贪吃蛇游戏，创造独有的模式，尽量做到界面友好，运行流畅，规则简明。

1.2 背景

学习了 C 语言一个学期的情况下，重新拾起来编写一个贪吃蛇程序。以前没有接触过图形界面，要自己到网上查找一些库，了解相关的使用操作。在 python 这种简洁的语言盛行的时代，用 c 语言写小游戏虽然略显臃肿但是运行效率依然很高。

1.3 定义

一个依靠键盘选择模式并控制小蛇移动的游戏程序。

1.4 参考资料

《C 高级语言程序设计》《C pimer》

2 任务概述

设计一个贪吃蛇游戏，用连续的格子代替蛇，独立的格子代替食物或者障碍物。碰到食物蛇长加一，碰到墙壁或者障碍物即死亡。障碍物和蛇的颜色可以自己设定。

3 需求分析

3.1 用户需求分析

需要一些代表物：蛇，食物，砖块等。

需要设定模式：改变速度。

需要键盘操控。

需要满足贪吃蛇基本规则。

3.2 运行环境

Windows10+安装有 easyx 库的 visual studio。

4 功能及操作介绍

4.1 操作

按住 WASD 即可实现转向。

4.2 功能

开始界面按 1，2 或者 3 可以调整蛇的移动速度。

吃到食物蛇可以变长，蛇不能朝与头相反的方向移动。

蛇撞到墙会死，撞到自己会死。

5 系统设计

5.1 总体架构设计

1，我们需要一张地图，图形库初始化的窗口是自带坐标的（左上角为 0，0）。

2，我们需要一条蛇，这条蛇由蛇头和蛇身组成（这里我们初始化三节蛇）。

3，我们需要食物，并且在蛇吃掉食物之后将蛇的身体变长，而且重新生成一个食物。

4，蛇需要移动，这应该是最难实现的（实现蛇身坐标的传值，除蛇头以外，每一节蛇的坐标都是上节蛇的上一次运动之前的坐标）。

5，蛇撞到墙或者撞到自己的身体就会死亡（判断坐标是否重合就可以）。

6，我们需要能用键盘控制蛇的运动方向，这个会和蛇的移动有一些联系。

5.2 模块分析与设计

1.初始化游戏数据：init()，用枚举变量定义空地，蛇，墙，食物等

```
void init()
{
    srand((unsigned)time(NULL));    //随机种子
    setbkcolor(WHITE);              //设置背景颜色

    memset(map, SPACE, sizeof(map));//初始化 map 数组为 0 即 SPACE
    //每一行的 第一个 和 最后一个 是墙
    for (int i = 0; i < ROW; i++)
    {
        map[i][0] = map[i][COL - 1] = WALL;
    }
    //每一列的 第二个 和 倒数第二 个是墙
    for (int j = 1; j < COL - 1; j++)
    {
        map[0][j] = map[ROW - 1][j] = WALL;
    }
    //定义蛇头和蛇的身体
    map[3][5] = HEAD;
    map[3][4] = map[3][3] = SNAKE;
    //初始化蛇
```

```
    SnakeSize = 3;  //蛇 长
    SnakeDir = 'D'; //蛇方向向右
    snake[0].X = 3;
    snake[0].Y = 5;
    snake[1].X = 3;
    snake[1].Y = 4;
    snake[2].X = 3;
    snake[2].Y = 3;
    addwall();
    addfood();
}
```

2. 开始界面：start()，打印三个速度模式

```
void start()
{
    setbkcolor(WHITE);      //设置窗口背景色为白色
    cleardevice();          //刷新屏幕

    setbkmode(TRANSPARENT); //设置字体背景色为透明
    settextcolor(BLACK);        //设置字体颜色为黑色
    /*****************游戏规则*****************/
    outtextxy(290, 80, L"功能选择:");
    outtextxy(280, 150, L"1.低速模式");
    outtextxy(280, 200, L"2.中速模式");
    outtextxy(280, 250, L"3.高速模式");
    outtextxy(200, 280, L"数字键 1,2,3 选择模式，Enter 键进入游戏");
    outtextxy(200, 300, L"字母键 W,S,A,D 方向键 上下左右 控制方向");
    outtextxy(130, 350, L"相关内容： ");
    outtextxy(160, 380, L"作者：葛旭，黄弋斌");
}
```

3. 选择：chose(),选择其中一个速度模式

```
void chose()
{
    switch (_getch())
    {
    case '1':
        start();
        outtextxy(260, 150, L"->");
        level = 1;
        break;
    case '2':
        start();
        outtextxy(260, 200, L"->");
        level = 2;
        break;
```

```
        case '3':
            start();
            outtextxy(260, 250, L"->");
            level = 3;
            break;
        case 13:
            return;
            break;
        }
    }
```

4. 循环检索并填充:DrawMap()，根据每一个位置的值进行绘图，给不同的模块填充不同的颜色。

```
    void DrawMap()
    {
        BeginBatchDraw();   //开始绘图
        setbkcolor(RGB(247, 238, 214)); //设置背景颜色为白色
        settextcolor(RGB(238, 0, 0));
        cleardevice();      //清屏
        WCHAR arr[10];      //保存成绩
        //WCHAR level[10];
        wsprintf(arr, L"总分：%d", score); //将成绩格式化输出到字符串 arr 中
        //wsprintf(level, L"速度等级：%d", level);
        outtextxy(0, 0, arr);               //显示成绩
        //outtextxy(150, 0, level);
        for (int y = 0; y < ROW; y++)        //y 轴方向向下
        {
            for (int x = 0; x < COL; x++)    //x 轴方向向右
            {
                switch (map[y][x])
                {
                case SPACE:
                    break;
                case WALL:
                    setlinecolor(BLACK);
                    setfillcolor(RGB(238, 233, 233));   //灰色
                    fillrectangle(x * 10, y * 10 + 20, x * 10 + 10, y * 10 + 30);
                    break;
                case SNAKE:
                    setlinecolor(RGB(255, 165, 0));     //绿色
                    setfillcolor(RGB(255, 165, 0));
                    fillrectangle(x * 10, y * 10 + 20, x * 10 + 10, y * 10 + 30);
                    break;
                case HEAD:
                    //画蛇头
```

```
                setfillcolor(RGB(0, 0, 255));       //蓝  0 0 255
                solidrectangle(x * 10, y * 10 + 20, x * 10 + 10, y * 10 + 30);
                break;
            case FOOD:
                setfillcolor(RGB(255, 0, 0));           //红色
                solidrectangle(x * 10, y * 10 + 20, x * 10 + 10, y * 10 + 30);
                break;
            default:
                break;
            }
        }
    }
    EndBatchDraw();
}
void addfood()
{
    int row, col;
    do
    {
        row = rand() % (ROW - 1) + 1;
        col = rand() % (COL - 1) + 1;
    } while (map[row][col] != SPACE && map[row][col] != WALL);
    map[row][col] = FOOD;
}
void addwall()
{
    int row, col;
    int i = 0;
    for (i = 0; i < BLOCK; i++)
    {
        do
        {
            row = rand() % (ROW - 1) + 1;
            col = rand() % (COL - 1) + 1;
        } while (map[row][col] != SPACE && map[row][col] != WALL);
        map[row][col] = WALL;
    }
}
```

5. 添加食物：addfood()

```
void addfood()
{
    int row, col;
    do
    {
```

```
            row = rand() % (ROW - 1) + 1;
            col = rand() % (COL - 1) + 1;
        } while (map[row][col] != SPACE && map[row][col] != WALL);
        map[row][col] = FOOD;
    }
```

6. 添加墙体：addwall()

```
    void addwall()
    {
        int row, col;
        int i = 0;
        for (i = 0; i < BLOCK; i++)
        {
            do
            {
                row = rand() % (ROW - 1) + 1;
                col = rand() % (COL - 1) + 1;
            } while (map[row][col] != SPACE && map[row][col] != WALL);
            map[row][col] = WALL;
        }
    }
```

7.改变方向：ChangeDir(),根据按键字母改变方向定义

```
    void ChangeDir()
    {
        switch (_getch())
        {
        case'A':
        case'a':
        case 75:
            if (SnakeDir != 'D') SnakeDir = 'A';    //蛇不能后退
            break;
        case'D':
        case'd':
        case 77:
            if (SnakeDir != 'A') SnakeDir = 'D';
            break;
        case'W':
        case'w':
        case 72:
            if (SnakeDir != 'S') SnakeDir = 'W';
            break;
        case'S':
        case's':
        case 80:
```

```
            if (SnakeDir != 'W') SnakeDir = 'S';
            break;
        case 32:
            _getch();
            break;
        default:
            break;
        }
    }
```

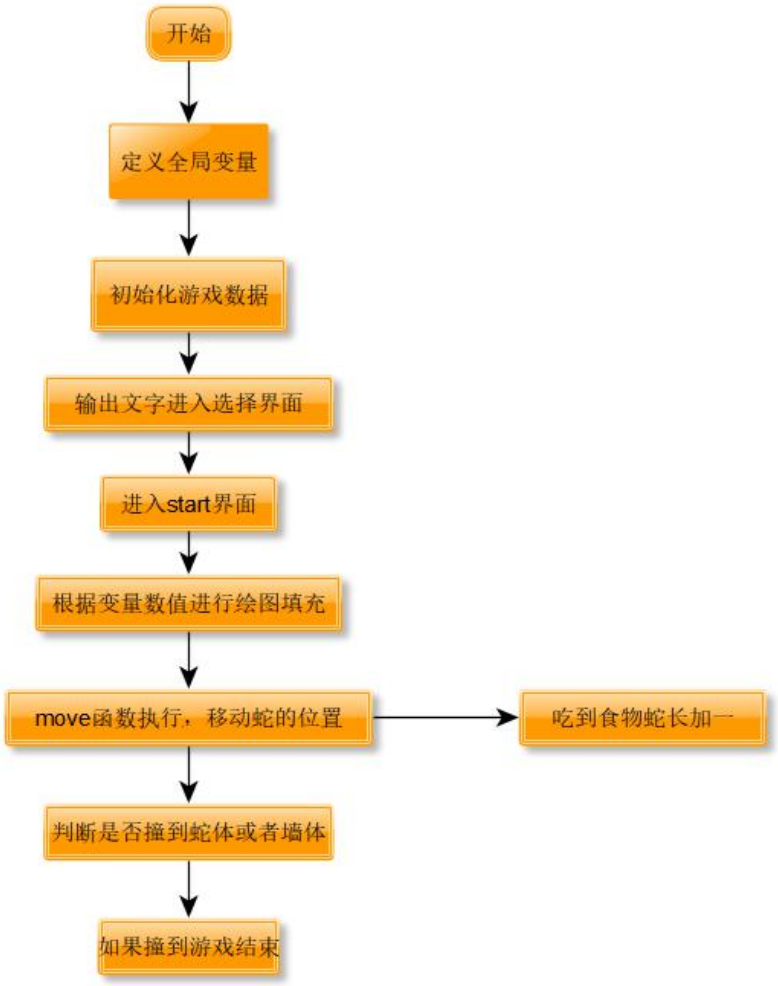
8.根据方向移动蛇体并判断下一个是否撞到蛇体或墙体：move(),

```
void move()
{
    COORD next;     //蛇头的下一个位置
    switch (SnakeDir)
    {
    case'A':
        next.X = snake[0].X;
        next.Y = snake[0].Y - 1;
        break;
    case'W':
        next.X = snake[0].X - 1;
        next.Y = snake[0].Y;
        break;
    case'D':
        next.X = snake[0].X;
        next.Y = snake[0].Y + 1;
        break;
    case'S':
        next.X = snake[0].X + 1;
        next.Y = snake[0].Y;
        break;
    default:
        break;
    }

    switch (map[next.X][next.Y])
    {
    case SPACE://直接移动
        map[snake[SnakeSize - 1].X][snake[SnakeSize - 1].Y] = SPACE;//地图蛇尾所在地置空
        for (int i = SnakeSize - 1; i > 0; i--)                    //蛇尾到蛇头整体移动一位
        {
            snake[i] = snake[i - 1];
        }
        map[snake[0].X][snake[0].Y] = SNAKE;                       //蛇头置 蛇
```

```
35.             snake[0] = next;                                        //将下一个位置赋值给蛇头
36.             map[snake[0].X][snake[0].Y] = HEAD;                     //设置头
37.             break;
38.         case WALL:
39.             MessageBox(GetHWnd(), L"游戏结束", L"SORRY", MB_OK);
40.             outtextxy(320, 200, L"是否要结束？");
41.             outtextxy(320, 280, L"结束按 N/n");
42.             if (_getch() == 'N' || _getch() == 'n')
43.             {
44.                 exit(0);
45.             }
46.             break;
47.         case SNAKE:
48.             MessageBox(GetHWnd(), L"游戏结束", L"SORRY", MB_OK);
49.             outtextxy(320, 200, L"是否要结束？");
50.             outtextxy(320, 280, L"结束按 N/n");
51.             if (_getch() == 'N' || _getch() == 'n')
52.             {
53.                 exit(0);
54.             }
55.             break;
56.         case FOOD://食物              蛇尾不变
57.             for (int i = SnakeSize; i > 0; i--)                     //蛇尾到蛇头整体移动一位
58.             {
59.                 snake[i] = snake[i - 1];
60.             }
61.             map[snake[0].X][snake[0].Y] = SNAKE;                    //蛇头 置 蛇
62.             snake[0] = next;                                        //将下一个位置赋值给蛇头
63.             score++;            //分数加一
64.             (SnakeSize)++;      //蛇尺度加一
65.             map[snake[0].X][snake[0].Y] = HEAD;                     //地图上重置蛇头
66.             addfood();
67.             break;
68.         default:break;
69.         }
70. }
```

5.3 软件结构（流程图）



6 调试与测试

6.1 调试过程

将主框架与各部分模块单独分写出来，例如生成食物，生成墙体，生成蛇，填充方块等，各个模块确认无误后再组合到一起，用全局变量进行交互，逻辑更清晰，减少了工作量。

6.2 测试结果

各代码块独立运行正常，组合到一起联合运行正常。

7 编程中遇到的问题

7.1 问题 1

界面显示不流畅，经常卡顿，并且不美观

解决办法：换用 easyx 库，用库函数以及 RGB 调色，使得界面尽量简洁美观。

7.2 问题 2

定义墙体，蛇体，食物等元素繁琐而不容易记忆。

解决办法：用枚举变量依次定义，并用字母代替，省去了宏定义的繁琐。

7.3 问题 3

蛇体的移动逻辑错误。

解决办法：将移动分为两个模块，一个用来后端定义方向，一个用于前端根据定义依次移动蛇体(蛇体的索引为二维数组)。

7.4 问题 4

使用链表经常出现各种 bug

解决办法：放弃单向链表改用二维数组，大大减少了出现错误的可能性。

8 分析总结与心得体会

在设计这个程序中我主要学会了如何运用以下有关 C 语言的知识:

1)函数定义是要做到顾名思义是很重要的，它对读程序的人正确认识程序十分重要，在修改这个程序的过程中也能很快找到程序各模块的作用，大大增加了程序的可读性。

3)在做程序的时候先列框架，将这个程序所要达到的目的(功能)分析出来，选择正确的数据结构然后在将程序模块化，按照模块编写函数更加简单合理。

4)我还了解了很多的库函数的作用，如字符串函数中有很多对字符串进行处理的函数，起功能我都有所了解。

5)学会了有关头文件的使用方法，及系统调配问题的解决方法等。

6)在调试过程中，遇到了很多的问题，比如，因为使用的刷屏函数，所以屏幕经常出现闪烁的情况,但是通过我和我的伙伴们的共同努力和学习，通过网上和书籍的查阅，我们也得到了很多的收获，我们的小组也在中真正的学会了很多的东西，这次实训给我们带来了很大的收获。可以按照程序运行的错误提示对原程序进行修改,在调试过程中有时也会遇到不懂的问题,我去图书馆或上网查阅- -些资料或者是向老师请教也解决了对源程序一一修改直到运行成功。