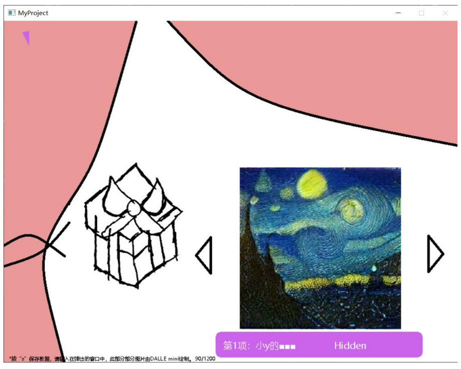
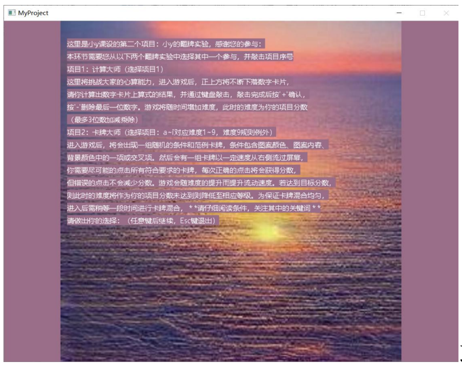
附件 1：游戏界面

初始界面：



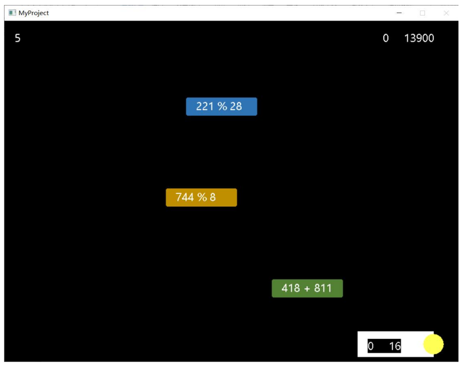
*按“s”保存数据，请输入在弹出的窗口中，此部分部分图片由DALLE mini绘制。90/1200

项目 2-初始界面：

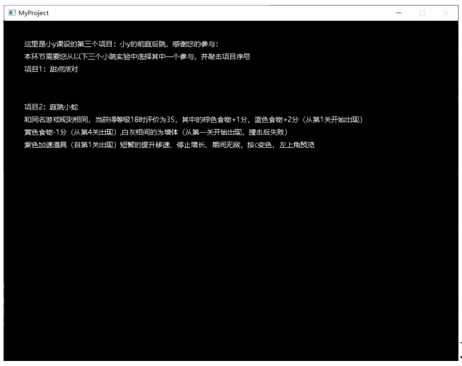


这里是小y课设的第二个项目：小y的翻牌实验，感谢您的参与：
本环节需要您从以下两个翻牌实验中选择其中一个参与，并敲击项目序号
项目1：计算大师（选择项目1）
这里将挑战大家的心算能力，进入游戏后，正上方将不断下落数字卡片，
请你计算出数字卡片上算式的结果，并通过键盘敲击，敲击完成后按'+'确认，
按'-'删除最后一位数字。游戏将随时间增加难度，此时的难度为你的项目分数
（最多3位数加减乘除）
项目2：卡牌大师（选择项目：a~i对应难度1~9，难度9规则例外）
进入游戏后，将会出现一组随机的条件和范例卡牌，条件包含图案颜色、图案内容、
背景颜色中的一项或交叉项。然后会有一组卡牌以一定速度从右侧流过屏幕，
你需要尽可能的点击所有符合要求的卡牌，每次正确的点击将会获得分数，
但错误的点击不会减少分数。游戏会随难度的提升而提升流动速度。若达到目标分数，
则此时的难度将作为你的项目分数未达到则降低至相应等级。为保证卡牌混合均匀，
进入后需稍等一段时间进行卡牌混合，**请仔细阅读条件，关注其中的关键词**
请做出你的选择：（任意键后继续，Esc键退出）

项目 2-1 界面示例：



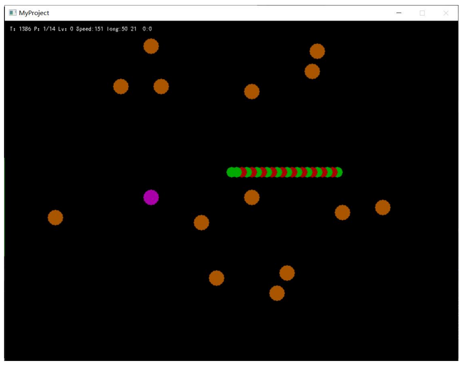
项目 3-初始界面：



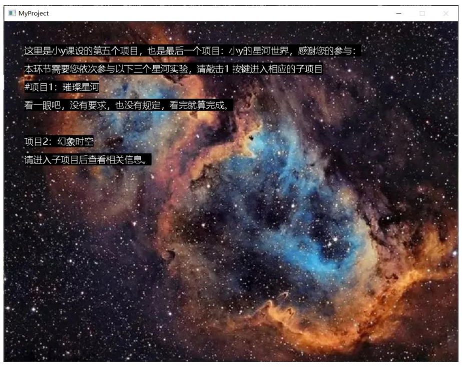
这里是小y课设的第三个项目：小y的前庭后院，感谢您的参与：
本环节需要您从以下三个小院实验中选择其中一个参与，并敲击项目序号
项目1：甜点派对

项目2：庭院小蛇
和同名游戏规则相同，当获得等级18时评价为3S，其中的棕色食物+1分，蓝色食物+2分（从第1关开始出现）
黄色食物-1分（从第4关出现）,白灰相间的为墙体（从第一关开始出现，撞击后失败）
紫色加速道具（自第1关出现）短暂的提升移速、停止增长、期间无敌，按c变色，左上角预览

项目 3-2 界面示例：

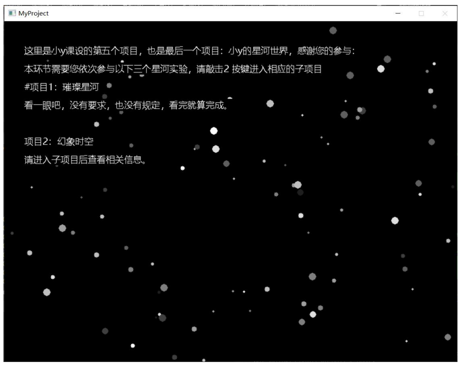


项目 5-初始界面：



这里是小y课设的第五个项目，也是最后一个项目：小y的星河世界，感谢您的参与：
本环节需要您依次参与以下三个星河实验，请敲击1 按键进入相应的子项目
#项目1：璀璨星河
看一眼吧，没有要求，也没有规定，看完就算完成。

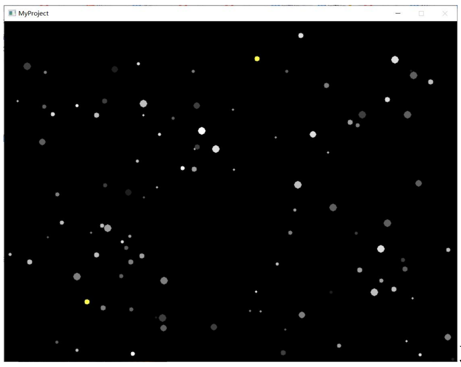
项目2：幻象时空
请进入子项目后查看相关信息。



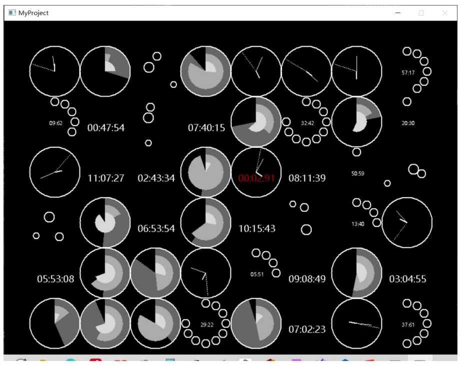
这里是小y课设的第五个项目，也是最后一个项目：小y的星河世界，感谢您的参与：
本环节需要您依次参与以下三个星河实验，请敲击2 按键进入相应的子项目
#项目1：璀璨星河
看一眼吧，没有要求，也没有规定，看完就算完成。

项目2：幻象时空
请进入子项目后查看相关信息。

项目 5-1 界面示例：



项目 5-2 界面示例：



附件 4：程序源代码

完整的代码如下：

源.cpp:

```
/***********************************************************************
    程序名:小 y 的课设作业
    版权: Copyright littley & LyIc
    作者:小 y（欧博远）
    北京工业大学  210241 班  21024112  欧博远  2022.春  高级语言程序设计课设课程作品
    打包日期: 2022.5.2  若文件更改时间晚于此时间则文件无效。
    项目组成：（在项目 1 至 5 中，仅需参与任意一个*计*分*项目，非计分项目不算（下面标记为
#），项目 4 完成项目任务即可，1-4 全部参与后解锁项目 5，项目 5 参与完成后结束）
    项目 1：小 y 的音游板（提交的作业中不包含该项目）
    项目 2：小 y 的翻牌实验
    项目 2-1：小 y 的计算大师        项目 2-2：小 y 的卡牌大师
    项目 3：小 y 的前庭后院
    项目 3-1：小 y 的甜点派对        项目 3-2：小 y 的庭院小蛇
    项目 4：小 y 的概率论（提交的作业中不包含该项目）
    项目 5：小 y 的星河世界
    项目 5-1：小 y 的幻想时空        #项目 5-2：小 y 的星河之旅
    需要注意的问题：
    其中参与到课设作业的项目有：2-1、3-2、3-3、5-1、5-2（包含有动态链表和/或文件记录的子
项目）.
***********************************************************************/
#include <stdio.h>
#include <graphics.h>
#include<conio.h>
#include<stdio.h>
#include<time.h>
#include<math.h>
#include <stdlib.h>
#include <windows.h>/
#include<iostream>


#pragma comment(lib,"Winmm.lib")
#define WIDTH    900
constexpr auto HEIGHT = 675;
#define PI 3.141592653589793238
#pragma warning(suppress : 4996)
#pragma warning(suppress : 4244)

//5-1
//星星的各个属性宏定义
#define MAX_STAR 100 //:数量
#define MAX_RADIUS 6
#define MAX_STEP 8
//星星的移动状态
#define STOP 0
#define UP 1
#define DOWN 2
#define LEFT 3
#define RIGTH 4
#define ALL_STATUS 5


//全局变量
ExMessage mouse;//鼠标信息
char s[128],key;//

void savep(int le,int fle, int n ) {//将游戏存档入文件里
        char name[24];
        srand(time(NULL));
        time_t timer = time(NULL);
        FILE* fp = NULL;
        int error;
        error = fopen_s(&fp, "./save.txt", "a"); //这里的返回值是，如果成功返回 0，如果不成功返回非 0
        if (fp)
        {
                if (n!=0)           fprintf_s(fp, "完成项目%d-%d，完成信息：Time：%s; Level:%d\n", le, fle, ctime(&timer), n);
                else fprintf_s(fp, "完成项目%d-%d，完成信息：Time：%s\n", le, fle, ctime(&timer));
        }
        _fcloseall();
}
void savep(int le, int fle, int n, int k) {//将游戏存档入文件里
        char name[24];
        srand(time(NULL));
        time_t timer = time(NULL);
        FILE* fp = NULL;
        int error;
        error = fopen_s(&fp, "./save.txt", "a"); //这里的返回值是，如果成功返回 0，如果不成功返回非 0
        if (fp)
        {
                if (n != 0)         fprintf_s(fp, "完成项目%d-%d，完成信息：Time：%s; Level:%d/// %d\n", le, fle, ctime(&timer), n, k);
                else fprintf_s(fp, "完成项目%d-%d，完成信息：Time：%s/// %d\n", le, fle, ctime(&timer), k);
        }
        _fcloseall();
}
/****************************项目文件存入****************************/
typedef struct Computing//用于算式信息
{
        int numa;
        int opera;
        int numb;
        int operb;
        int numc;
        int result;
        int     bracket;
        int x;
        int y;
        int co;
        struct Computing* next;
}Computing;

Computing* CreaterComputing() {
;
        Computing* head = NULL, * end = NULL;
        head = (Computing*)malloc(sizeof(Computing));
        srand(time(NULL));
        head->next = NULL;
        return head;
}
Computing* AddComputing(Computing* pt, int m) {
        int i = 0, a = 0, b = 0, c = 0, r = 0;
        Computing* p = pt, * t;
        srand(time(NULL));
        while (p->next != NULL) {
                p = p->next;
        }
        t = (Computing*)malloc(sizeof(Computing));
        if (t) {
                p->next = t;
                switch (m)
                {
                case(0):p->numa = rand() % 10 + 1;     p->numb = rand() % 10 + 1;        p->bracket = 0;  p->result = p->numa + p->numb; p->opera = 1; p->operb
= -1; p->numc = 0;     p->x = rand() % 70 * 10 + 50;     p->y = 0; p->co = rand() % 7; break;
                case(1):p->opera = rand() % 2;
                        if (p->opera == 0) { p->numa = rand() % 10 + 2;           p->numb = rand() % (p->numa - 1) + 1;              p->result = p->numa - p->numb; }
                        else if (p->opera == 1) { p->numa = rand() % 10 + 1;           p->numb = rand() % 10 + 1;                         p->result    =    p->numa    +
p->numb; }      p->bracket = 0;
                        p->operb = -1; p->numc = 0;      p->x = rand() % 70 * 10 + 50;     p->y = 0;   p->co = rand() % 7;                  break;
                case(2):p->opera = rand() % 2;
                        if (p->opera == 0) { p->numa = rand() % 40 + 10;         p->numb = rand() % (p->numa - 9) + 5;              p->result = p->numa - p->numb; }
                        else if (p->opera == 1) { p->numa = rand() % 40 + 10;          p->numb = rand() % 40 + 10;                        p->result    =    p->numa    +
p->numb; }      p->bracket = 0;
                        p->operb = -1; p->numc = 0;      p->x = rand() % 70 * 10 + 50;     p->y = 0;   p->co = rand() % 7;                  break;
                case(3):p->opera = rand() % 2;
                        if (p->opera == 0) { p->numa = rand() % 80 + 21;         p->numb = rand() % (p->numa - 20) + 10;           p->result = p->numa - p->numb; }
                        else if (p->opera == 1) { p->numa = rand() % 90 + 10;          p->numb = rand() % 90 + 10;                        p->result    =    p->numa    +
p->numb; }      p->bracket = 0;
                        p->operb = -1; p->numc = 0;      p->x = rand() % 70 * 10 + 50;     p->y = 0;   p->co = rand() % 7;                  break;
                case(4):p->opera = rand() % 5;
                        if (p->opera == 0) { p->numa = rand() % 10 + 1;          p->numb = rand() % (p->numa) + 1;           p->result = p->numa - p->numb; }
                        else if (p->opera == 1) { p->numa = rand() % 9 + 1;      p->numb = rand() % 9 + 1;                          p->result = p->numa + p->numb; }
                        else if (p->opera == 2) { p->numa = rand() % 10 + 1;     p->numb = rand() % 10 + 1;                         p->result    =    p->numa    *
p->numb; }
                        else if (p->opera == 3) { p->numb = rand() % 10 + 1;     p->result = rand() % 10 + 1;                       p->numa = p->result * p->numb; }
                        else if (p->opera == 4) { p->numa = rand() % 99 + 1;     p->numb = rand() % 8 + 2;                          p->result = p->numa % p->numb; }
                        p->bracket = 0;    p->operb = -1; p->numc = 0;      p->x = rand() % 70 * 10 + 50;     p->y = 0;   p->co = rand() % 7;                  break;
                case(5):p->opera = rand() % 5;
                        if (p->opera == 0) { p->numa = rand() % 980 + 20;                p->numb = rand() % (p->numa - 20) + 10;           p->result    =    p->numa    -
p->numb; }
                        else if (p->opera == 1) { p->numa = rand() % 990 + 10; p->numb = rand() % 990 + 10;                        p->result = p->numa + p->numb; }
                        else if (p->opera == 2) { p->numa = rand() % 100 + 1;  p->numb = rand() % 100 + 1;                         p->result    =    p->numa    *
p->numb; }
                        else if (p->opera == 3) { p->numb = rand() % 90 + 10;  p->result = rand() % 9 + 1;                   p->numa = p->result * p->numb; }
                        else if (p->opera == 4) { p->numa = rand() % 990 + 10; p->numb = rand() % 28 + 2;                          p->result    =    p->numa    %
p->numb; }
                        p->bracket = 0; p->operb = -1; p->numc = 0;p->x = rand() % 70 * 10 + 50;     p->y = 0;           p->co = rand() % 7;                        break;
                case(6):p->opera = rand() % 2;
                        if (p->opera == 0) { p->numa = rand() % 80 + 20;         p->numb = rand() % (p->numa - 20) + 10;           p->result = p->numa - p->numb; }
                        else if (p->opera == 1) { p->numa = rand() % 99 + 1;     p->numb = rand() % 99 + 1;                         p->result = p->numa + p->numb; }
                        p->operb = rand() % 2;
                        if (p->operb == 0) { p->numc = rand() % (p->result - 5) + 1;          p->result = p->result - p->numc; }
                        else if (p->operb == 1) { p->numc = rand() % 99 + 1;     p->result = p->result + p->numc; }
                        p->bracket = 0;  p->x = rand() % 70 * 10 + 50;     p->y = 0;           p->co = rand() % 7;                        break;
                case(7):p->opera = rand() % 4;      p->operb = rand() % 2; while (p->opera * p->operb == 6) { p->opera = rand() % 4;      p->operb = rand() % 2; }
                                if (p->operb == 1 || p->operb == 0) {
                                        if (p->opera == 0) { p->numa = rand() % 17 + 4;             p->numb = rand() % (p->numa - 3) + 2;              p->result  =  p->numa  -
p->numb; }
                                        else if (p->opera == 1) { p->numa = rand() % 19 + 1;       p->numb = rand() % 19 + 1;                         p->result  =  p->numa  +
p->numb; }
                                        else if (p->opera == 2) { p->numa = rand() % 9 + 1; p->numb = rand() % 9 + 1;                          p->result    =    p->numa    *
p->numb; }
                                        else if (p->opera == 3) { p->numb = rand() % 9 + 1; p->result = rand() % 8 + 2;                   p->numa = p->result * p->numb; }
                                        if (p->operb == 0) { p->numc = rand() % min((p->result - 2) + 1, 19) + 1;          p->result = p->result - p->numc; }
                                        else if (p->operb == 1) { p->numc = rand() % 19 + 1;       p->result = p->result + p->numc; }
                                }
                                else if (p->operb == 2 || p->operb == 3) {
                                        if (p->opera == 0 || p->opera == 1) {
                                                if (p->operb == 2) { p->numb = rand() % 9 + 1; p->numc = rand() % 9 + 1; p->result = p->numb * p->numc; }
                                                else if (p->operb == 3) { p->result = rand() % 9 + 1; p->numc = rand() % 9 + 1; p->numb = p->result * p->numc; }
                                                if (p->opera == 0) { p->numa = p->result + rand() % 9 + 1; p->result = p->numa - p->result; }
                                                else if (p->opera == 1) { p->numa = rand() % 19 + 1; p->result = p->numa + p->result; }
                                        }
                                        else if (p->opera == 2 || p->opera == 3) {
                                                 if (p->opera == 2 && p->operb == 2) { p->numa = rand() % 9 + 1; p->numb = rand() % 9 + 1; p->numc = rand() % 9 + 1;
p->result = p->numa * p->numb * p->numc; }
                                                else if (p->opera == 3 && p->operb == 3) { p->numc = rand() % 9 + 1; p->numb = rand() % 9 + 1; p->result = rand() % 9 + 1;
p->numa = p->result * p->numb * p->numc; }
                                        }
                                }
                                p->bracket = 0;     p->x = rand() % 70 * 10 + 50;     p->y = 0;           p->co = rand() % 7;                        break;
                case(8): p->opera = rand() % 4;      p->operb = rand() % 2; while (p->opera * p->operb == 6) { p->opera = rand() % 4;      p->operb = rand() % 2; }
                                if (p->operb == 1 || p->operb == 0) {
                                        if (p->opera == 0) { p->numa = rand() % 950 + 50;          p->numb = rand() % (p->numa - 30) + 20;           p->result  =  p->numa  -
p->numb; }
                                        else if (p->opera == 1) { p->numa = rand() % 990 + 10;   p->numb = rand() % 990 + 10;                       p->result  =  p->numa  +
p->numb; }
                                        else if (p->opera == 2) { p->numa = rand() % 90 + 10;     p->numb = rand() % 90 + 10;                              p->result           =
p->numa * p->numb; }
                                        else if (p->opera == 3) { p->numb = rand() % 90 + 10;     p->result = rand() % 70 + 30;                      p->numa  =  p->result  *
p->numb; }
                                        if (p->operb == 0) { p->numc = rand() % min((p->result - 20) + 15, int((rand() % 30 * 0.01 + 0.2) * p->result));           p->result     =
p->result - p->numc; }
                                        else if (p->operb == 1) { p->numc = rand() % 999 + 1;     p->result = p->result + p->numc; }
                                }
                                else if (p->operb == 2 || p->operb == 3) {
                                        if (p->opera == 0 || p->opera == 1) {
                                                if (p->operb == 2) { p->numb = rand() % 90 + 10; p->numc = rand() % 90 + 10; p->result = p->numb * p->numc; }
                                                else if (p->operb == 3) { p->result = rand() % 90 + 10; p->numc = rand() % 90 + 10; p->numb = p->result * p->numc; }
                                                if (p->opera == 0) { p->numa = p->result + rand() % 990 + 10; p->result = p->numa - p->result; }
                                                else if (p->opera == 1) { p->numa = rand() % 990 + 10; p->result = p->numa + p->result; }
                                        }
                                        else if (p->opera == 2 || p->opera == 3) {
                                                if (p->opera == 2 && p->operb == 2) { p->numa = rand() % 90 + 1; p->numb = rand() % 99 + 1; p->numc = rand() % 99 + 1;
p->result = p->numa * p->numb * p->numc; }
                                                 else if (p->opera == 3 && p->operb == 3) { p->numc = rand() % 90 + 10; p->numb = rand() % 90 + 10; p->result = rand() %
90 + 10; p->numa = p->result * p->numb * p->numc; }
                                        }
                                }
                                p->bracket = 0;     p->x = rand() % 70 * 10 + 50;     p->y = 0;           p->co = rand() % 7;                        break;
                }
                t->x = p->x;
                t->y = p->y;
                t->next = NULL;
                return pt;
        }
}
int _out(Computing *pt){
        int i = 0;
        settextstyle(28, 0, "微软雅黑");
    int wi = 0, co = 0;
    setbkmode(TRANSPARENT);
    while (pt->next != NULL)
    {
        pt = pt->next;
        wi = 0;
        if (pt->operb == -1 && pt->numa < 10)wi = 75;
        else if (pt->operb == -1 && pt->numa < 100)wi = 90;
        else if (pt->operb == -1)wi = 120;
        else if ((pt->operb == 1 || pt->operb == 2 || pt->operb == 3 || pt->operb == 0) && pt->numa < 100) wi = 150;
        else if ((pt->operb == 1 || pt->operb == 2 || pt->operb == 3 || pt->operb == 0) && pt->numa >= 100) wi = 180;
        if (wi > 0) {
            switch (pt->co)
            {
            case(0):setfillcolor(RGB(192, 0, 0)); break;            case(1):setfillcolor(RGB(198, 89, 17)); break;
    case(2):setfillcolor(RGB(191, 143, 0)); break;
            case(3):setfillcolor(RGB(84, 130, 53)); break;              case(4):setfillcolor(RGB(51, 63, 79)); break;
    case(5):setfillcolor(RGB(47, 117, 181)); break;
            case(6):setfillcolor(RGB(112, 48, 160)); break;
            }
            solidroundrect(pt->x - 20, pt->y - 3, pt->x + wi, pt->y + 32, 5, 5);

        }//就是背景了
        setlinestyle(PS_SOLID, 2);      setlinecolor(BLACK);      line(rand() % 675, rand() % 900, rand() % 675, rand() % 900);
        setlinestyle(PS_DOT, 2);        line(rand() % 675, rand() % 900, rand() % 675, rand() % 900);
        i++;
        if (pt->opera == 1 && pt->operb == -1) { sprintf(s, _T("%d + %d"), pt->numa, pt->numb);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 0 && pt->operb == -1) { sprintf(s, _T("%d - %d"), pt->numa, pt->numb);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 2 && pt->operb == -1) { sprintf(s, _T("%d * %d"), pt->numa, pt->numb);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 3 && pt->operb == -1) { sprintf(s, _T("%d / %d"), pt->numa, pt->numb);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 4 && pt->operb == -1) { sprintf(s, _T("%d %% %d"), pt->numa, pt->numb);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 0 && pt->operb == 0) { sprintf(s, _T("%d - %d - %d"), pt->numa, pt->numb, pt->numc);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 0 && pt->operb == 1) { sprintf(s, _T("%d - %d + %d"), pt->numa, pt->numb, pt->numc);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 0 && pt->operb == 2) { sprintf(s, _T("%d - %d * %d"), pt->numa, pt->numb, pt->numc);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 0 && pt->operb == 3) { sprintf(s, _T("%d - %d / %d"), pt->numa, pt->numb, pt->numc);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 1 && pt->operb == 0) { sprintf(s, _T("%d + %d - %d"), pt->numa, pt->numb, pt->numc);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 1 && pt->operb == 1) { sprintf(s, _T("%d + %d + %d"), pt->numa, pt->numb, pt->numc);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 1 && pt->operb == 2) { sprintf(s, _T("%d + %d * %d"), pt->numa, pt->numb, pt->numc);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 1 && pt->operb == 3) { sprintf(s, _T("%d + %d / %d"), pt->numa, pt->numb, pt->numc);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 2 && pt->operb == 0) { sprintf(s, _T("%d * %d - %d"), pt->numa, pt->numb, pt->numc);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 2 && pt->operb == 1) { sprintf(s, _T("%d * %d + %d"), pt->numa, pt->numb, pt->numc);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 2 && pt->operb == 2) { sprintf(s, _T("%d * %d * %d"), pt->numa, pt->numb, pt->numc);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 2 && pt->operb == 3) { sprintf(s, _T("Wrong")); outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 3 && pt->operb == 0) { sprintf(s, _T("%d / %d - %d"), pt->numa, pt->numb, pt->numc);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 3 && pt->operb == 1) { sprintf(s, _T("%d / %d + %d"), pt->numa, pt->numb, pt->numc);    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 3 && pt->operb == 2) { sprintf(s, _T("Wrong"));    outtextxy(pt->x, pt->y, s); }
        else if (pt->opera == 3 && pt->operb == 3) { sprintf(s, _T("%d / %d / %d"), pt->numa, pt->numb, pt->numc);    outtextxy(pt->x, pt->y, s); }
    }//算式
    setbkmode(OPAQUE);
    return 0;
}
void remove(Computing* pt) {
    if (pt->next) {
        Computing* p = pt->next, * t = NULL;
        pt->next = p->next;
        free(p);
    }
}
int correct(int* num, Computing* pt, int* Error, int* p) {
    if (pt->next) {
        if (pt->next->result == *num) {
            *p = *p + 1; return GREEN;
        }
        else {
            *Error = *Error + 1;    return RED;
        }
    }
}
int _out(int *num,Computing *pt,int *Error,int *p,int *t,int le) {
    setfillcolor(WHITE);
    solidrectangle(700, 615, 850, 665);
    int x = WHITE;
    if (_kbhit())
    {
        key = _getch();
        switch (key)
        {
        case('1'):if (*num < 10000)*num = *num * 10 + 1; break;
        case('2'):if (*num < 10000)*num = *num * 10 + 2; break;
        case('3'):if (*num < 10000)*num = *num * 10 + 3; break;
        case('4'):if (*num < 10000)*num = *num * 10 + 4; break;
        case('5'):if (*num < 10000)*num = *num * 10 + 5; break;
        case('6'):if (*num < 10000)*num = *num * 10 + 6; break;
        case('7'):if (*num < 10000)*num = *num * 10 + 7; break;
        case('8'):if (*num < 10000)*num = *num * 10 + 8; break;
        case('9'):if (*num < 10000)*num = *num * 10 + 9; break;
        case('0'):if (*num < 10000)*num = *num * 10 + 0; break;
        case('/'):*num *= -1; break;
        case('-'):*num /= 10; break;
        case(8):*num /= 10; break;
        case('+'):x=correct(num,pt,Error,p); *num = 0; remove(pt); return x; break;
        case(13):x=correct(num, pt, Error, p); *num = 0; remove(pt); return x; break;
        case('s'):*p = *p + 2; break;
        case('x'):*Error = -9; break;
        }
    }
    sprintf(s, _T("%d        %d"), *num, *p);            outtextxy(720, 630, s);
    sprintf(s, _T("%d        %d"), *Error, *t);     outtextxy(750, 20, s);
    sprintf(s, _T("%d"), le);             outtextxy(20, 20, s);

}
void moveC(Computing *pt) {
    while (pt->next != NULL)
    {
        pt = pt->next;
        pt->y += 5;
    }
}
int Computingmaster() {
    int level = 0, Error = 0, t = 0, number = 0, point = 0, interval = 1000, sleep = 25, co = YELLOW, m = YELLOW;
    int* num = &number, * Er = &Error, * p = &point, * ti = &t;
    srand(time(NULL));
    Computing* Cd;
    Cd = CreaterComputing();
    mciSendString("open Travelinthenaturalage.mp3 alias bkmusic0", NULL, 0, NULL);
    mciSendString("play bkmusic0", NULL, 0, NULL);
    BeginBatchDraw();
    while (key != 27 && Error <= 3) {
        cleardevice();
        _out(Cd);
        m = _out(num, Cd, Er, p, ti, level);
        if (m == GREEN || m == RED) { co = m; }   setfillcolor(co);
        solidcircle(850, 640, 20);           moveC(Cd);
        if (t > 2500 && Cd->next) { if (Cd->next->y >= 600) { remove(Cd); Error++; } }
        FlushBatchDraw();
        Sleep(sleep);
        t += sleep;
        if (t % interval == 0) { AddComputing(Cd, level); }
```

```
        if (point == 4 && level == 0) { level = 1; t = 0;    Error = 0; point = 0; interval = 1000; sleep = 25; }
        else if (point == 6 && level == 1) { level = 2; t = 0;    Error = 0; point = 0; interval = 2000; sleep = 50; }
        else if (point == 8 && level == 2) { level = 3; t = 0;    Error = 0; point = 0; interval = 2000; sleep = 50; }
        else if (point == 10 && level == 3) { level = 4; t = 0;    Error = 0; point = 0;    interval = 2000; sleep = 40; }
        else if (point == 14 && level == 4) { level = 5; t = 0;    Error = -5; point = 0; interval = 4500; sleep = 100; }
        else if (point == 20 && level == 5) { level = 6; t = 0;    Error = 0; point = 0; interval = 3000; sleep = 100; }
        else if (point == 14 && level == 6) { level = 7; t = 0;    Error = -2; point = 0; interval = 1800; sleep = 60; }
        else if (point == 16 && level == 7) { level = 8; t = 0;    Error = -5; point = 0; interval = 6000; sleep = 150; }/**/
        else if (point == 20 && level == 8) { Error = -5; }
    }
    EndBatchDraw();
    mciSendString("close bkmusic0", NULL, 0, NULL);
    if (level <= 7)    return level;
    else if (point <= 20)return 8;
    else return 9;

}
/****************************小 y 的计算大师绘制结束****************************/


typedef struct node0//用于 CardMaster 信息
{
    int backcolour;
    int coin;
    int coincolour;
    char letter;
}Card;
int Cardmaster(int level) {
    srand(time(NULL));
    int i, j, info[6] = { 9 }, t = 0;
    Card card[800] = { 0 };
    int cardcount = 0;//卡牌数量
    float flashtime = 0, alflashtime = 0;//流动时长、总时长
    int cardshow = 0, f = 0;//展示时长
    int project = rand() % 3, projectcount = 0, projectfinish = 0;//目标要求及数量,完项数量
    int point = 0;//分数
    IMAGE Gax1, Gax2, Gax3, Gax4, Gax5;
    loadimage(&Gax1,   _T("Galaxy1.jpg"));   loadimage(&Gax2,   _T("Galaxy2.jpg"));   loadimage(&Gax3,   _T("Galaxy3.jpg"));   loadimage(&Gax4,
_T("Galaxy4.jpg")); loadimage(&Gax5, _T("Galaxy5.jpg"));
    setlinestyle(PS_DOT, 5);
    switch (level)
    {
    case(1):cardcount = 120; flashtime = 10.0; cardshow = 5100; alflashtime = 3600; projectfinish = 4; break;
    case(2):cardcount = 155; flashtime = 9.0; cardshow = 4800; alflashtime = 4500; projectfinish = 5; break;
    case(3):cardcount = 200; flashtime = 8.0; cardshow = 4500; alflashtime = 5000; projectfinish = 6; break;
    case(4):cardcount = 255; flashtime = 7.1; cardshow = 4200; alflashtime = 5500; projectfinish = 7; break;
    case(5):cardcount = 320; flashtime = 6.1; cardshow = 3900; alflashtime = 6000; projectfinish = 9; break;
    case(6):cardcount = 395; flashtime = 5.2; cardshow = 3600; alflashtime = 6000; projectfinish = 11; break;
    case(7):cardcount = 480; flashtime = 4.2; cardshow = 3300; alflashtime = 6000; projectfinish = 12; break;
    case(8):cardcount = 575; flashtime = 3.2; cardshow = 3000; alflashtime = 5000; projectfinish = 13; break;
    //case(9):point = HardCardMaster(); return point; break;
    default:        break;
    }
    cleardevice();
    for (i = 0; i < cardcount + 2; i++)
    {
        if (i == 0) info[3] = rand() % 5;
        else while (info[3] == info[0]) info[3] = rand() % 7;
        while (info[4] == info[1])    info[4] = rand() % 5;
        while (info[5] == info[2] || info[5] == info[3])info[5] = rand() % 5;
        if (info[3] >= 5)for (j = 0; j < 6; j++)info[j] = 5;
        info[0] = info[3];
        info[1] = info[4];
        info[2] = info[5];

        switch (info[0])
        {
        case(0):card[i].backcolour = RED; break;
        case(1):card[i].backcolour = BLUE; break;
        case(2):card[i].backcolour = GREEN; break;
        case(3):card[i].backcolour = CYAN; break;
        case(4):card[i].backcolour = MAGENTA; break;
        default:card[i].backcolour = BLACK;   break;
        }
        card[i].coin = info[1];
        switch (info[2])
        {
        case(0):card[i].coincolour = RED; break;
        case(1):card[i].coincolour = BLUE; break;
        case(2):card[i].coincolour = GREEN; break;
        case(3):card[i].coincolour = CYAN; break;
        case(4):card[i].coincolour = MAGENTA; break;
        default:card[i].coincolour = BLACK;    break;
        }
    }
    for (j = 1; j < cardcount; j++) {
        switch (project) {
        case(0):if (card[j].backcolour == card[0].backcolour) projectcount++;    break;
        case(1):if (card[j].coincolour == card[0].coincolour) projectcount++;    break;
        case(2):if (card[j].coin == card[0].coin) projectcount++;    break;
        }
    }
    settextstyle(20, 0, _T("微软雅黑"));
    for (i = 1; i < 5; i++) {
        switch (i)
        {
        case(1):sprintf(s, _T("这里是你的第%d 项：你的任务目标：*尽可能*选出所有（注意图案还是图案颜色）"),level); outtextxy(40, 10 + 25 * i, s);
break;
        case(2):
            switch (project) {
            case(0):sprintf(s, _T("与下面图形背景颜色相同的图形,3 秒后展示图形，图形只展示%.2f 秒。"), float(cardshow)/1000); break;
            case(1):sprintf(s, _T("与下面图形图案颜色相同的图形,3 秒后展示图形，图形只展示%.2f 秒。"), float(cardshow) / 1000); break;
            case(2):sprintf(s, _T("与下面图形中图案相同的图形,3 秒后展示图形，图形只展示%.2f 秒。"), float(cardshow) / 1000); break;
            case(3):sprintf(s, _T("与下面图形背景与图案颜色均相同的图形，3 秒后展示图形，图形只展示%.2f 秒。"), float(cardshow) / 1000); break;
            case(4):sprintf(s, _T("与下面图形背景颜色与图案（非颜色）均相同的图形，3 秒后展示图形，图形只展示%.2f 秒。"), float(cardshow)
/ 1000); break;
            case(5):sprintf(s, _T("与下面图形图案及其颜色均相同的图形，3 秒后展示图形，图形只展示%.2f 秒。"), float(cardshow) / 1000); break;
            }outtextxy(40, 10 + 25 * i, s); Sleep(2000);    break;
        case(3):
            setfillcolor(card[0].backcolour);
            solidroundrect(100, 140, 300, 340, 5, 5);
            switch (card[0].coin)
            {
            case(0):setfillcolor(card[0].coincolour);    solidcircle(200, 240, 50); break;
            case(1):setfillcolor(card[0].coincolour);    solidellipse(140, 200, 260, 280); break;
            case(2):setfillcolor(card[0].coincolour);    solidrectangle(140, 200, 260, 280); break;
            case(3):setfillcolor(card[0].coincolour);    solidpie(140, 200, 260, 320, 0, 3.14); break;
            case(4):setfillcolor(card[0].coincolour); { POINT pts[] = { {200,190}, {200 + 100 * sqrt(3) / 3 , 290}, {200 - 100 * sqrt(3) / 3 , 290} };
solidpolygon(pts, 3); }break;
            default:              break;
            }break;
        case(4):sprintf(s, _T("图案总数为%d，符合要求的图案数量：%d，选出其中的%d 个即算完项，达到%d 个降档结算。（退出/Esc）"), cardcount,
projectcount, projectcount - projectfinish, projectcount - 2 * projectfinish + 1);outtextxy(40, 500, s);    break;
        }
    }
    f = rand() % 5;
    Sleep(cardshow);
    cleardevice();
```

```c
        sprintf(s, _T("请尝试回忆刚才闪过的图形，2 秒后图案将会从屏幕右侧流出（退出/Esc）")); outtextxy(40, 10 + 25 * i, s);
         Sleep(2000);
        float speed = 9 / flashtime;
        BeginBatchDraw();
        while (t<alflashtime) {
                switch (f)
                {
                case(1):putimage(225, 112, &Gax1); break;
                case(2):putimage(225, 112, &Gax2); break;
                case(3):putimage(225, 112, &Gax3); break;
                case(4):putimage(225, 112, &Gax4); break;
                case(0):putimage(112, 0, &Gax5); break;
                }
                for (i = 1; i < cardcount+1; i++) {
                        setfillcolor(card[i].backcolour);
                        solidroundrect(800 + 120 * ((i - 1) / 5) - speed * t, 50 + 120 * ((i - 1) % 5), 900 + 120 * ((i - 1) / 5) - speed * t, 150 + 120 * ((i - 1) % 5), 5, 5);
                        switch (card[i].coin)
                        {
                        case(0):setfillcolor(card[i].coincolour);    solidcircle(850 + 120 * ((i - 1) / 5) - speed * t, 100 + 120 * ((i - 1) % 5), 25); break;
                        case(1):setfillcolor(card[i].coincolour);    solidellipse(820 + 120 * ((i - 1) / 5) - speed * t, 80 + 120 * ((i - 1) % 5), 880 + 120 * ((i - 1) / 5) - speed * t, 120 + 120 * ((i - 1) % 5)); break;
                        case(2):setfillcolor(card[i].coincolour);    solidrectangle(820 + 120 * ((i - 1) / 5) - speed * t, 80 + 120 * ((i - 1) % 5), 880 + 120 * ((i - 1) / 5) - speed * t, 120 + 120 * ((i - 1) % 5)); break;
                        case(3):setfillcolor(card[i].coincolour);    solidpie(820 + 120 * ((i - 1) / 5) - speed * t,80 + 120 * ((i - 1) % 5), 880 + 120 * ((i - 1) / 5) - speed * t, 140 + 120 * ((i - 1) % 5), 0, 3.14); break;
                        case(4):setfillcolor(card[i].coincolour); { POINT pas[] = { {850 + 120 * ((i - 1) / 5) - speed * t,75 + 120 * ((i - 1) % 5)}, {850 + 50 * sqrt(3) / 3 + 120 * ((i - 1) / 5) - speed * t, 125 + 120 * ((i - 1) % 5)}, {850 - 50 * sqrt(3) / 3 + 120 * ((i - 1) / 5) - speed * t, 125 + 120 * ((i - 1) % 5)} }; solidpolygon(pas, 3); } break;
                        default:setlinecolor(card[i].coincolour);    line(820 + 120 * ((i - 1) / 5) - speed * t, 70 + 120 * ((i - 1) % 5), 880 + 120 * ((i - 1) / 5) - speed * t, 130 + 120 * ((i - 1) % 5)); break;
                        }
                }
                sprintf(s, _T("%d"),t); outtextxy(10, 2, s);
                peekmessage(&mouse, EM_MOUSE, 1);
                int cardclick = ((mouse.y - 50) / 120 + 1) + int((mouse.x + speed * t - 800) / 120) * 5;
                if (mouse.message == WM_LBUTTONUP) {
                        switch (project) {
                        case(0):if (card[cardclick].backcolour == card[0].backcolour) card[cardclick].coin = 9;    break;
                        case(1):if (card[cardclick].coincolour == card[0].coincolour) card[cardclick].coin = 9;    break;
                        case(2):if (card[cardclick].coin == card[0].coin)card[cardclick].coin = 9;    break;
                        case(3):if (card[cardclick].backcolour == card[0].backcolour && card[cardclick].coincolour == card[0].coincolour) card[cardclick].coin = 9; break;
                        case(4):if (card[cardclick].backcolour == card[0].backcolour && card[cardclick].coin == card[0].coin) card[cardclick].coin = 9; break;
                        case(5):if (card[cardclick].coin == card[0].coin && card[cardclick].coincolour == card[0].coincolour) card[cardclick].coin = 9; break;
                        }
                }
                if (_kbhit()) {
                        key = _getch();
                        if (key == 27)t = 9999;
                }

                FlushBatchDraw();
                if (level < 4)Sleep(10);
                else Sleep(6);//尝试对抗系统卡顿问题，可能会比设计快不少
                cleardevice();
                t += 1;
        }
        for (i = 0; i < cardcount; i++) {
                if (card[i].coin == 9)
                        point++;
        }
        sprintf(s, _T("%d/%d      %d/%d"), point, projectcount,projectcount - projectfinish, projectcount - 2 * projectfinish + 1); outtextxy(10, 22, s);
        if (point >= projectcount - projectfinish)     i = level;
        else if (point >= projectcount - 2 * projectfinish + 1) i = level - 1;
        else i = max(level - 5, 0);
        EndBatchDraw();
        if (t < 8000)Sleep(5000);
        return i;
}
/****************************小 y 的卡牌大师绘制结束****************************/
int centregame() {
        setbkcolor(RGB(154, 110, 137));
        cleardevice();
        srand(time(NULL));
        int i;
        IMAGE Gax1, Gax2, Gax3, Gax4, Gax5;
        loadimage(&Gax1,    _T("Galaxy1.jpg"));    loadimage(&Gax2,    _T("Galaxy2.jpg"));    loadimage(&Gax3,    _T("Galaxy3.jpg"));    loadimage(&Gax4, _T("Galaxy4.jpg")); loadimage(&Gax5, _T("Galaxy5.jpg"));
        i = 0;
        switch (i)
        {
        case(1):putimage(225, 112, &Gax1); break;
        case(2):putimage(225, 112, &Gax2); break;
        case(3):putimage(225, 112, &Gax3); break;
        case(4):putimage(225, 112, &Gax4); break;
        case(0):putimage(112, 0, &Gax5); break;
        }
        settextstyle(20, 0, _T("微软雅黑"));
        for (i = 1; i < 16; i++) {
                switch (i)
                {
                case(1):sprintf(s, _T("这里是小 y 课设的第二个项目：小 y 的翻牌实验，感谢您的参与：")); break;
                case(2):sprintf(s, _T("本环节需要您从以下两个翻牌实验中选择其中一个参与，并敲击项目序号")); break;
                case(3):sprintf(s, __T("项目 1：计算大师（选择项目 1）")); break;
                case(4):sprintf(s, _T("这里将挑战大家的心算能力，进入游戏后，正上方将不断下落数字卡片，")); break;
                case(5):sprintf(s, _T("请你计算出数字卡片上算式的结果，并通过键盘敲击，敲击完成后按'+'确认，")); break;
                case(6):sprintf(s, _T("按'-'删除最后一位数字。游戏将随时间增加难度，此时的难度为你的项目分数")); break;
                case(7):sprintf(s, _T("（最多 3 位数加减乘除）")); break;
                case(8):sprintf(s, _T("项目 2：卡牌大师（选择项目：a~i 对应难度 1~9，难度 9 规则例外）")); break;
                case(9):sprintf(s, _T("进入游戏后，将会出现一组随机的条件和范例卡牌，条件包含图案颜色、图案内容、")); break;
                case(10):sprintf(s, _T("背景颜色中的一项或交叉项。然后会有一组卡牌以一定速度从右侧流过屏幕，")); break;
                case(11):sprintf(s, __T("你需要尽可能的点击所有符合要求的卡牌，每次正确的点击将会获得分数，")); break;
                case(12):sprintf(s, _T("但错误的点击不会减少分数。游戏会随难度的提升而提升流动速度。若达到目标分数，")); break;
                case(13):sprintf(s, _T("则此时的难度将作为你的项目分数未达到则降低至相应等级。为保证卡牌混合均匀，")); break;
                case(14):sprintf(s, _T("进入后需稍等一段时间进行卡牌混合，  * *请仔细阅读条件，关注其中的关键词  * *")); break;
                case(15):sprintf(s, _T("请做出你的选择：（任意键后继续，Esc 键退出）")); break;
                }
                outtextxy(125, 10 + 25 * i, s);
        }
        setbkcolor(BLACK);
        key = _getch();
        if (key == '1') { i = Computingmaster(); savep(2, 1, i); }
        else if (key >= 'a' && key <= 'h') { i = Cardmaster(key - 'a' + 1); savep(2, 2, i); }
        else i = 0;
        return i;
}

/********************************************************    小 y 的翻牌实验绘制结束
********************************************************/
/********************************************************    小 y 的翻牌实验绘制结束
********************************************************/
typedef struct cake//用于 Cake 信息
{
        int type;
        int line;
        int last;
        int x;
        struct cake* next;
}Cake;
Cake* Create() {
        srand(time(NULL));
        int i = 0;
        Cake* head = NULL, * end = NULL, * p;
        head = (Cake*)malloc(sizeof(Cake));
        srand(time(NULL));
        end = head;
        for (i = 0; i < 10; i++) {
                p = (Cake*)malloc(sizeof(Cake));
                if (p)
                {
                        p->type = rand() % 3;
                        p->line = rand() % 3;
                        p->last = rand() % 500 + 500;
                        p->next = NULL;
                }

        }
        return head;
}
int out(Cake* pt)
{
        IMAGE img;
        loadimage(&img, _T("cake1.jpg"));
        int y = 0;
        putimage(100, 100, &img);
        while (pt->next != NULL)
        {
                pt = pt->next;
                y = 125 + 150 * pt->line;
                //if(pt->type==1)
        }
        return 0;
}
int Dessertparty() {
        cake* pt;
        pt = Create();
        out(pt);
        //load();
        return 0;
}

/****************************小 y 的甜点派对绘制结束****************************/
typedef struct snake//用于 Snack
{
        int x;
        int y;
        int dir;
        int colour;
        struct snake* next;
}Snake;
typedef struct food//用于 Food
{
        int x;
        int y;
        int p;
        struct food* next;
}Food;
Snake* CreateSnake(int lo) {
        int i = 0;
        srand(time(NULL));
        Snake* head = NULL, * end = NULL, * p;
        head = (Snake*)malloc(sizeof(Snake));
        end = head;
        for (i = 0; i < lo; i++) {

                p = (Snake*)malloc(sizeof(Snake));
                if (p)
                {
                        p->x = 450;
                        p->y = 300;
                        p->dir = rand() % 4 + 1;
                        p->next = NULL;
                        switch (i % 2) {
                        case(0):p->colour = RED; break;
                        case(1):p->colour = GREEN; break;
                        }
                        if (i == 0) p->colour = BLUE;
                        end->next = p;
                        end = p;
                }
        }
        return head;
}
Snake* CreateWall(int n,int *w) {
        int i = 0, ix = 0, iy = 0, length = 0, dir = 0, odir = 0;
        srand(time(NULL));
        Snake* head = NULL, * end = NULL, * p;
        head = (Snake*)malloc(sizeof(Snake));
        end = head;
        ix = rand() % 55 * 10 + 150;
        iy = rand() % 40 * 10 + 150;
        *w = ix; *(w + 1) = iy;
        for (i = 0; i < n; i++) {
                p = (Snake*)malloc(sizeof(Snake));
                if (p)
                {
                        if (i % 16 == 0) { dir = rand() % 4 + 1; }
                        if (i % 32 == 0) { ix += (rand() % 12 - 6) * 10; iy += (rand() % 12 - 6) * 10; }
                        if (dir == 1) {          ix += 10;        p->x = ix;            p->y = iy; }
                        else if (dir == 2) { ix -= 10;        p->x = ix;            p->y = iy; }
                        else if (dir == 3) { iy += 10;        p->x = ix;            p->y = iy; }
                        else if (dir == 4) { iy -= 10;        p->x = ix;            p->y = iy; }
                        p->dir = dir;
                        odir = dir;
                        p->next = NULL;
                        switch (i % 2) {
                        case(0):p->colour = WHITE; break;
                        case(1):p->colour = RGB(117,117,117); break;
                        }
                        if (i == 0) p->colour = BLUE;
                        end->next = p;
                        end = p;
                }

        }
        return head;
}
Food* Createfood(int n) {
        int i = 0;
        Food* head = NULL, * end = NULL, * p;
        srand(time(NULL));
        head = (Food*)malloc(sizeof(Food));
        end = head;
        if (n == 0) {
                for (i = 0; i < 14; i++) {
                        p = (Food*)malloc(sizeof(Food));
                if (p)
                {
                    if (i < 3) {
                        p->x = rand() % 40 * 10 + 50;
                        p->y = rand() % 30 * 10 + 50;
                    }
                    else if (i < 6) {
                        p->x = rand() % 40 * 10 + 450;
                        p->y = rand() % 30 * 10 + 50;
                    }
                    else if (i < 9) {
                        p->x = rand() % 40 * 10 + 50;
                        p->y = rand() % 30 * 10 + 350;
                    }
                    else {
                        p->x = rand() % 40 * 10 + 450;
                        p->y = rand() % 30 * 10 + 350;
                    }
                    p->p = 1;
                    p->next = NULL;
                    end->next = p;
                    end = p;
                }
            }
    }
    else {
        p = (Food*)malloc(sizeof(Food));
        if (p)
        {
            p->x = rand() % 80 * 10 + 50;
            p->y = rand() % 60 * 10 + 50;
            p->p = 10;
        }
        p->next = NULL;
        end->next = p;
        end = p;
    }
    return head;
}
int _out(Snake* pt)
{
    while (pt->next != NULL)
    {
        pt = pt->next;
        setfillcolor(pt->colour);
        solidcircle(pt->x, pt->y, 10);
    }
    return 0;
}
int _out(Food* fd)
{
    while (fd->next != NULL)
    {
        fd = fd->next;
        if (fd->p == 1)setfillcolor(BROWN);
        else if (fd->p == -1) setfillcolor(YELLOW);
        else if (fd->p == 2)    setfillcolor(BLUE);
        else setfillcolor(MAGENTA);
        solidcircle(fd->x, fd->y, 15);
    }
    return 0;
}
void Change(Snake* pt, int n, int m) {
    Snake* k = pt->next;
    int i = 0;
    if (n == -1) {
        setfillcolor(RGB(22, 97, 65)); solidcircle(20, 20, 10);         setfillcolor(RGB(111, 152, 40)); solidcircle(20, 35, 10); setfillcolor(RGB(22, 97, 65));
solidcircle(20, 50, 10);     setfillcolor(RGB(111, 152, 40)); solidcircle(20, 65, 10);
        setfillcolor(RGB(193, 176, 6)); solidcircle(45, 20, 10); setfillcolor(RGB(236, 71, 140)); solidcircle(45, 35, 10); setfillcolor(RGB(193, 176, 6));
solidcircle(45, 50, 10); setfillcolor(RGB(236, 71, 140)); solidcircle(45, 65, 10);
        setfillcolor(RGB(173, 224, 227)); solidcircle(70, 20, 10); setfillcolor(RGB(122, 67, 1)); solidcircle(70, 35, 10); setfillcolor(RGB(173, 224, 227));
solidcircle(70, 50, 10); setfillcolor(RGB(122, 67, 1)); solidcircle(70, 65, 10);
        setfillcolor(RGB(234, 224, 145)); solidcircle(95, 20, 10); setfillcolor(RGB(152, 134, 138)); solidcircle(95, 35, 10); setfillcolor(RGB(234, 224, 145));
solidcircle(95, 50, 10); setfillcolor(RGB(152, 134, 138)); solidcircle(95, 65, 10);
        setfillcolor(RGB(29, 56, 83)); solidcircle(120, 20, 10); setfillcolor(RGB(211, 169, 35)); solidcircle(120, 35, 10); setfillcolor(RGB(29, 56, 83));
solidcircle(120, 50, 10); setfillcolor(RGB(211, 169, 35)); solidcircle(120, 65, 10);
        setfillcolor(RGB(233, 212, 176)); solidcircle(145, 20, 10); setfillcolor(RGB(255, 250, 211)); solidcircle(145, 35, 10); setfillcolor(RGB(233, 212,
176)); solidcircle(145, 50, 10); setfillcolor(RGB(255, 250, 211)); solidcircle(145, 65, 10);
        setfillcolor(RGB(175, 219, 221)); solidcircle(170, 20, 10); setfillcolor(RGB(104, 116, 171)); solidcircle(170, 35, 10); setfillcolor(RGB(175, 219,
221)); solidcircle(170, 50, 10); setfillcolor(RGB(104, 116, 171)); solidcircle(170, 65, 10);
        setfillcolor(RGB(230, 128, 178)); solidcircle(195, 20, 10); setfillcolor(RGB(226, 22, 87)); solidcircle(195, 35, 10); setfillcolor(RGB(230, 128, 178));
solidcircle(195, 50, 10); setfillcolor(RGB(226, 22, 87)); solidcircle(195, 65, 10);
        setfillcolor(RGB(217, 220, 215)); solidcircle(220, 20, 10); setfillcolor(RGB(181, 188, 174)); solidcircle(220, 35, 10); setfillcolor(RGB(217, 220,
215)); solidcircle(220, 50, 10); setfillcolor(RGB(181, 188, 174)); solidcircle(220, 65, 10);
        setfillcolor(RGB(228, 65, 49)); solidcircle(245, 20, 10); setfillcolor(RGB(72, 176, 236)); solidcircle(245, 35, 10); setfillcolor(RGB(228, 65, 49));
solidcircle(245, 50, 10); setfillcolor(RGB(72, 176, 236)); solidcircle(245, 65, 10);
        FlushBatchDraw();
    }
    while (n != -1 && i < 50 && k->next != NULL) {
        k = k->next;
        i++;
        switch (n) {
        case(1):if (i % 2 == 0)k->colour = RGB(22, 97, 65);         else k->colour = RGB(111, 152, 40); break;
        case(2):if (i % 2 == 0)k->colour = RGB(193, 176, 6);    else k->colour = RGB(236, 71, 140); break;
        case(3):if (i % 2 == 0)k->colour = RGB(173, 224, 227);else k->colour = RGB(122, 67, 1); break;
        case(4):if (i % 2 == 0)k->colour = RGB(234, 224, 145);else k->colour = RGB(152, 134, 138); break;
        case(5):if (i % 2 == 0)k->colour = RGB(29, 56, 83);         else k->colour = RGB(211, 169, 35); break;
        case(6):if (i % 2 == 0)k->colour = RGB(233, 212, 176);else k->colour = RGB(255, 250, 211); break;
        case(7):if (i % 2 == 0)k->colour = RGB(175, 219, 221);else k->colour = RGB(104, 116, 171); break;
        case(8):if (i % 2 == 0)k->colour = RGB(230, 128, 178);else k->colour = RGB(226, 22, 87); break;
        case(9):if (i % 2 == 0)k->colour = RGB(217, 220, 215);else k->colour = RGB(181, 188, 174); break;
        case(10):if (i % 2 == 0)k->colour = RGB(228, 65, 49);   else k->colour = RGB(72, 176, 236); break;
        }
    }
    while (n != 1 && k->next != NULL) {
        k = k->next;
        i++;
        switch (m) {
        case(1):if (i % 2 == 0)k->colour = RGB(22, 97, 65);         else k->colour = RGB(111, 152, 40); break;
        case(2):if (i % 2 == 0)k->colour = RGB(193, 176, 6);    else k->colour = RGB(236, 71, 140); break;
        case(3):if (i % 2 == 0)k->colour = RGB(173, 224, 227);else k->colour = RGB(122, 67, 1); break;
        case(4):if (i % 2 == 0)k->colour = RGB(234, 224, 145);else k->colour = RGB(152, 134, 138); break;
        case(5):if (i % 2 == 0)k->colour = RGB(29, 56, 83);         else k->colour = RGB(211, 169, 35); break;
        case(6):if (i % 2 == 0)k->colour = RGB(233, 212, 176);else k->colour = RGB(255, 250, 211); break;
        case(7):if (i % 2 == 0)k->colour = RGB(175, 219, 221);else k->colour = RGB(104, 116, 171); break;
        case(8):if (i % 2 == 0)k->colour = RGB(230, 128, 178);else k->colour = RGB(226, 22, 87); break;
        case(9):if (i % 2 == 0)k->colour = RGB(217, 220, 215);else k->colour = RGB(181, 188, 174); break;
        case(10):if (i % 2 == 0)k->colour = RGB(228, 65, 49);   else k->colour = RGB(72, 176, 236); break;
        }
    }
}


void Changecolour(Snake* pt) {
    int i;
    int m = 0, n = 0;
    Change(pt, -1, 0);
    key = _getch();
    switch (key)
    {
    case('q'):m = 1; break; case('w'):m = 2; break; case('e'):m = 3; break; case('r'):m = 4; break; case('t'):m = 5; break;
    case('y'):m = 6; break; case('u'):m = 7; break; case('i'):m = 8; break; case('o'):m = 9; break; case('p'):m = 10; break;
    case('a'):m = 11; break; case('s'):m = 12; break; case('d'):m = 13; break; case('f'):m = 14; break; case('g'):m = 15; break;
    }
    key = _getch();
    switch(key)
    {
    case('q'):n = 1; break; case('w'):n = 2; break; case('e'):n = 3; break; case('r'):n = 4; break; case('t'):n = 5; break;
    case('y'):n = 6; break; case('u'):n = 7; break; case('i'):n = 8; break; case('o'):n = 9; break; case('p'):n = 10; break;
    case('a'):n = 11; break; case('s'):n = 12; break; case('d'):n = 13; break; case('f'):n = 14; break; case('g'):n = 15; break;
    }
    Change(pt, m, n);
    FlushBatchDraw();
}
void move(Snake* pt) {
    srand(time(NULL));
    int x = 0, y = 0, a = 0, b = 0;
    pt = pt->next;
    if (_kbhit()) {
        key = _getch();
        if (key == 'd' && pt->dir != 2)pt->dir = 1;
        else if (key == 'a' && pt->dir != 1)pt->dir = 2;
        else if (key == 's' && pt->dir != 4)pt->dir = 3;
        else if (key == 'w' && pt->dir != 3)pt->dir = 4;
        else if (key == 'c')Changecolour(pt);
    }
    switch (pt->dir) {
    case(1):pt->x += 10; break;
    case(2):pt->x -= 10; break;
    case(3):pt->y += 10; break;
    case(4):pt->y -= 10; break;
    }
    x = pt->x;
    y = pt->y;
    while (pt->next != NULL)
    {
        pt = pt->next;
        a = pt->x;
        b = pt->y;
        pt->x = x;
        pt->y = y;
        x = a;
        y = b;
    }

}
int Knock(Snake* pt,Snake *k) {//pt 为蛇头地址，k 为禁止撞击点地址（墙头地址/蛇头地址）
    Snake* body, * head = pt->next;//
    int i = 0;
    body = k->next;
    while (body->next != NULL)
    {
        body = body->next;
        if (body->x == head->x && body->y == head->y) i = 1;
    }
    if (body->x == head->x && body->y == head->y) i = 1;
    if (head->y > 1075 || head->y < -300 || head->x>1200 || head->x < -300) i = 9;
    if (_kbhit()) {
        key = _getch();
        if (key == 27)i = 1;

    }
    return i;
}
Snake* Createsnake(Snake* pt) {
    int i = 0;
    Snake* p = pt, * t;
    while (p->next != NULL) {
        i++;
        p = p->next;
    }
    t = (Snake*)malloc(sizeof(Snake));
    if (t) {
        p->next = t;
        if (p->colour == BLUE)          t->colour = YELLOW;
        else        t->colour = BLUE;
        t->x = p->x;
        t->y = p->y;
        t->next = NULL;
    }
    return pt;
}
void remove(Food* re, Food* fd) {
    Food* p = fd;
    while (p->next != re) {
        p = p->next;
    }
    p->next = re->next;
    free(re);
}
int eat(Snake* pt, Food* fd) {
    Snake* head = pt->next;
    Food* f = fd;
    int p = 0;
    while (f->next != NULL) {
        f = f->next;
        if (abs(head->x - f->x) < 20 && abs(head->y - f->y) < 20) {
            p += f->p; remove(f, fd); break;
        }
    }
    return p;
}
void Addfood(Food* fd, int k, int m,int *w1,int *w2) {
    int i = 0;
    Food* p = fd, * t;
    srand(time(NULL));
    while (p->next != NULL) {
        p = p->next;
    }
    if (m == 0) {
        for (i = 0; i < k; i++) {
            t = (Food*)malloc(sizeof(Food));
            if (t) {
                t->p = 1;
                if (i < k / 4 - 1) {
                    t->x = rand() % 40 * 10 + 50;                       t->y = rand() % 30 * 10 + 50;
                }
                else if (i < k / 2 - 1) {
                    t->x = rand() % 40 * 10 + 450;                      t->y = rand() % 30 * 10 + 50;
                }
                else if (i < k * 3 / 4 - 1) {
                    t->x = rand() % 40 * 10 + 50;                       t->y = rand() % 30 * 10 + 350;
                }
                else if (i < k - 1) {
                    t->x = rand() % 40 * 10 + 450;                      t->y = rand() % 30 * 10 + 350;
                }
                else {
                    t->x = rand() % 80 * 10 + 50;                    t->y = rand() % 60 * 10 + 50;                    t->p = 2;
                }
                while (abs(t->x - *w1) < 10 || abs(t->y - *(w1 + 1)) < 10 || abs(t->x - *w2) < 10 || abs(t->y - *(w2 + 1)) < 10) {
                    if (i < k / 4 - 1) {
                        t->x = rand() % 40 * 10 + 50;                    t->y = rand() % 30 * 10 + 50;
                    }
                    else if (i < k / 2 - 1) {
                        t->x = rand() % 40 * 10 + 450;                    t->y = rand() % 30 * 10 + 50;
                    }
                    else if (i < k * 3 / 4 - 1) {
                        t->x = rand() % 40 * 10 + 50;                    t->y = rand() % 30 * 10 + 350;
                    }
                    else if (i < k - 1) {
                        t->x = rand() % 40 * 10 + 450;                    t->y = rand() % 30 * 10 + 350;
                    }
                    else {
                        t->x = rand() % 80 * 10 + 50;                    t->y = rand() % 60 * 10 + 50;
t->p = 2;
                    }
                }
            }
            t->next = NULL;
            p->next = t;
            p = t;
        }
    }
    else if (m == 1) {
        t = (Food*)malloc(sizeof(Food));
        if (t) {
            for (i = 0; i < k; i++) {
                t->x = rand() % 80 * 10 + 50;
                t->y = rand() % 60 * 10 + 50;
                t->p = -1;
            }
        }
        t->next = NULL;
        p->next = t;
        p = t;
    }
    else {
        for (i = 0; i < k; i++) {
            t = (Food*)malloc(sizeof(Food));
            if (t) {
                t->x = rand() % 80 * 10 + 50;
                t->y = rand() % 60 * 10 + 50;
                t->p = 10;
            }
            t->next = NULL;
            p->next = t;
            p = t;
        }
    }
}

int Greedysnake() {
    Snake* pt = NULL, * walla = NULL, * wallb = NULL;
    Food* fd = NULL, * upspeed = NULL;
    int i = 1, m = 0, t = 0, point = 1, level = 0, lo = 50, inter = 0, k = 13, wa1[2] = { 0 }, wa2[2] = { 0 }, usp = 0, ust = 0;
    //i m 计数器，t 时间，point 当前关卡分数，level 等级，lo 蛇体长度，inter 间隔数目（非时间），k 分数目标，wal* 墙头坐标，usp 提升速
度大小，ust 提升速度效果时间。
    int* w1 = &wa1[0], * w2 = &wa2[0];
    pt = CreateSnake(lo);
    fd = Createfood(0);
    upspeed = Createfood(1);
    BeginBatchDraw();
    settextstyle(12, 0, "黑体");
        while (i) {
        cleardevice();//清屏
        _out(pt);//蛇绘制
        _out(fd);//食物绘制
        if(upspeed) _out(upspeed);//临时速度增益绘制
        if(walla) _out(walla);          if (wallb) _out(wallb);//墙体绘制
        move(pt);//蛇移动
        point += eat(pt, fd);//食物拾取
        if (eat(pt, upspeed)) {
            usp += min(50 + 10 * level + 15 * int(level / 4), 200); ust += min(3000+300 * level, 6000); point += 1;
            if (usp > min(50 + 10 * level + 15 * int(3 / 4), 200) * int(sqrt(2 * level + 1)))          usp  =  min(50  +  10  *  level  +  15  *  int(3  /  4),  200)  *
int(sqrt(2 * level + 1));
            if (ust > 10000 * level) { ust = 10000 * level; }
        }//临时速度增益拾取
        if (usp != 0 && ust > 0) { ust -= 10000 / (150 + 20 * level + point + usp); }//临时速度增益计时
        if (usp != 0 && ust <= 0) { usp = 0; ust = 0; }//临时速度增益结束
        if (t > 3200)i -= Knock(pt, pt->next);//自身撞击
        if (walla) i -= Knock(pt, walla);       if (wallb) i -= Knock(pt, wallb);       //ab 墙体撞击
        sprintf(s, _T("T：%d P：%d/%d Lv：%d Speed:%d long:%d %d    %d:%d"), t, point, k + 1, level, 150 + 20 * level + point + usp, lo, inter, usp, ust);
    outtextxy(10, 10, s);//监控面板
        FlushBatchDraw();
        Sleep(10000 / (150 + 20 * level + point + usp));//游戏时长
        t += 10000 / (150 + 20 * level + point + usp); //游戏时长
        inter++;//时间间隔
        if (inter % 240 == 0) { pt = Createsnake(pt); lo++; }//蛇长增加（按移动距离）
        if (point > 0 && point >= (k + 1) && point != 0) //升级
        {
            if (inter < 240 * level + 240) { level += 2; Addfood(fd, 2, 1, w1, w2); }
            else    level += 1;
            if (level < 6) k = 13;          else if (level < 18) k = 10 + level; else k = level * int(level / 8);//k 值计算
        if (level % 2 == 0)      walla = CreateWall(12 + 6 * level, w1); else { wallb = CreateWall(8 + 4 * level, w2); }//墙体重绘
        Addfood(fd, k, 0, w1, w2); Addfood(fd, 2, 1, w1, w2); Addfood(upspeed, level / 3 + 1, 2, w1, w2); point = 0;//添加食物、分数清零
        for (m = 0; m < 2 * level + 2; m++) { pt = Createsnake(pt); lo++; }//蛇体长度
        }
        if (usp > 0 && i > -2) { i = 1; inter--; }//临时速度增益期间半无敌（不免疫边界溢出）且不增加长度
        if (i < 0)          i = 0;//退出（防溢出）
    }
    EndBatchDraw();
    Sleep(2000);
    if (level > 18)level = 18;
    if (level < 0)level = 0;
    if (level <= 1)return level;
    else if(level <= 13)return int(level / 2 + 1);
    switch (level) { case(14):return 7; break;     case(15):return 8; break;     case(16):return 8; break;     case(17):return 8; break;     case(18):return          9;
break; }
}
/****************************小 y 的后院小蛇（？ ）绘制结束****************************/
int Vestibularbackyard() {
    int point = 0;
    IMAGE bk;
    loadimage(&bk, _T("bk3.jpg"));
    putimage(0, 0, &bk);
    int i;
    settextstyle(20, 0, "微软雅黑");
    cleardevice();
    for (i = 1; i < 10; i++) {
        switch (i)
        {
        case(1):sprintf(s, _T("这里是小 y 课设的第三个项目：小 y 的前庭后院，感谢您的参与：")); break;
        case(2):sprintf(s, _T("本环节需要您从以下三个小院实验中选择其中一个参与，并敲击项目序号")); break;
        case(3):sprintf(s, __T("项目 1：甜点派对")); break;
        case(4):sprintf(s, _T("")); break;
        case(5):sprintf(s, _T("")); break;
        case(6):sprintf(s, _T("项目 2：庭院小蛇")); break;
        case(7):sprintf(s, _T("和同名游戏规则相同，当获得等级 18 时评价为 3S，其中的棕色食物+1 分，蓝色食物+2 分（从第 1 关开始出现）"));
break;
        case(8):sprintf(s, __T("黄色食物-1 分（从第 4 关出现）,白灰相间的为墙体（从第一关开始出现，撞击后失败）")); break;
        case(9):sprintf(s, _T("紫色加速道具（自第 1 关出现）短暂的提升移速、停止增长、期间无敌，按 c 变色，左上角预览")); break;

        }
        outtextxy(40, 10 + 25 * i, s);
    }
    key = _getch();
    switch (key)
    {
    //case('1'):point = Dessertparty(); break;
    case('2'):point = Greedysnake(); break;
    default:break;
    }
    return point;
}
/**********************************************************     小     y     的     前     庭     后     院     绘     制     结     束
**********************************************************/
/**********************************************************     小     y     的     前     庭     后     院     绘     制     结     束
**********************************************************/
//星星结构体内部的链表  存的是其他星星结构体的地址
typedef struct _LinkNode {

    struct _LinkNode* prev; //上一个星星里 node 的地址
    struct _LinkNode* next; //下一个星星里 node 的地址
}_LinkNode;
//星星结构体
typedef struct {

    int x;
    int y;
    unsigned int radius;
    int status;//上面利用宏定义了 6 种状态
    int step;
    int color;
    _LinkNode node;
}_STAR;
//初始化星星首节点
bool starInit(_STAR*& L_star) {
    L_star = (_STAR*)malloc(sizeof(_STAR));    //开空间
    if (!L_star) {
        return false;
    }
    //不初始化星星属性//这里是首节点
    L_star->node.next = NULL;
    L_star->node.prev = NULL;
    return true;
}
//初始化星星里的属性
void initStar(_STAR*& p) {
    if (!p) {
        return;
    }
    p->x = rand() % 850 + 50;                       //50 - 850
    p->y = rand() % 610 + 30;                       //30-640
    p->radius = rand() % MAX_RADIUS + 2;                 // 2 - 7
    p->status = int((rand() % 25 + 1) / 3);                                              //状态 0~8(rand%25+1[→1-26])/3[→0-8]
    p->step = rand() % MAX_STEP + 1;                    //步长  1- 9
    int rgb = 255 * p->step / MAX_STEP;
    p->color = RGB(rgb, rgb, rgb);                      //颜色
    //if (p->status == 0) p->color = RGB(253, 231, 4);
}

//添加链表(头插法)
bool linkInsert_front(_LinkNode* L, _LinkNode* node) {

    //L:首个星星里 node 地址
    //node:添加星星里 node 地址
    if (!L || !node) {

        return false;
    }
    if (L->next) {
        L->next->prev = node;
    }
    node->next = L->next;
    node->prev = L;
    L->next = node;
    return true;
}
//尾插法
bool linkInsert_back(_LinkNode* L, _LinkNode* node) {
    if (!L || !node) {
        return false;
    }
    _LinkNode* last = L;
    //找到最后一个的地址
    while (last->next) {
        last = last->next;
    }
    node->next = NULL;
    node->prev = last;
    last->next = node;
    return true;
}
//显示星空
void starDisplay(_STAR*& L_star) {
    if (!L_star || !L_star->node.next) {
        return;
    }
    //指向第一个链表节点
    _LinkNode* p = L_star->node.next;
    //查看距离的_STAR 头的位置
    int offset = offsetof(_STAR, node);
    while (p) {
        _STAR* tmp = (_STAR*)((size_t)p - offset);
        //绘制图像
        setfillcolor(tmp->color);
        solidcircle(tmp->x, tmp->y, tmp->radius);
        //指向下一个星星结构体里的 node 地址
        p = p->next;
    }

}
//星星移动
void moveStar(_STAR*& L_star) {
    if (!L_star || !L_star->node.next) {
        return;
    }
    //指向第一个星星
    _LinkNode* p = L_star->node.next;
    //查看距离的_STAR 头的位置
    int offset = offsetof(_STAR, node);
    while (p) {
```

```c
        _STAR* tmp = (_STAR*)((size_t)p - offset);
        //擦除原来位置的星星
        setfillcolor(BLACK);
        solidcircle(tmp->x, tmp->y, tmp->radius);
        //判断移动状态,通过步长移动坐标
    //默认都是 UP
        switch (tmp->status) {
        case UP:
            tmp->y -= tmp->step;
            if (tmp->y <= 0)    tmp->y = HEIGHT;
            break;
        case DOWN:
            tmp->y += tmp->step;
            if (tmp->y >= HEIGHT)    tmp->y = 0;
            break;
        case LEFT:
            tmp->x -= tmp->step;
            if (tmp->x <= 0)    tmp->x = WIDTH;
            break;
        case RIGTH:
            tmp->x += tmp->step;
            if (tmp->x >= WIDTH)    tmp->x = 0;
            break;
        case    5:
            tmp->x += tmp->step;                    tmp->y += tmp->step;
            if (tmp->x >= WIDTH)    tmp->x = 0;     if (tmp->y >= HEIGHT)    tmp->y = 0; break;
        case    6:
            tmp->x += tmp->step;                    tmp->y -= tmp->step;
            if (tmp->x >= WIDTH)    tmp->x = 0;     if (tmp->y <= 0)    tmp->y = HEIGHT; break;
        case    7:
            tmp->x -= tmp->step;                    tmp->y += tmp->step;
            if (tmp->x <= 0)    tmp->x = WIDTH;     if (tmp->y >= HEIGHT)    tmp->y = 0; break;
        case    8:
            tmp->x -= tmp->step;                    tmp->y -= tmp->step;
            if (tmp->x <= 0)    tmp->x = WIDTH;     if (tmp->y <= 0)    tmp->y = HEIGHT; break;
        default:
            break;
        }
        //绘制新位置图像
        setfillcolor(tmp->color);
        solidcircle(tmp->x, tmp->y, tmp->radius);
        //遍历到下一个星星的 node 地址
        p = p->next;
    }
}

int ending(int n) {
    srand(time(NULL));
    _STAR star;
    if (n <= 1) mciSendString("open Allthoughtsarestars.mp3 alias bkmusic", NULL, 0, NULL);
    else if (n == 2)mciSendString("open Lifeline.mp3 alias bkmusic", NULL, 0, NULL);
    mciSendString("play bkmusic", NULL, 0, NULL); //  仅播放一次
    int s_time = 0, t = 0;
    star.color = YELLOW;        star.node.next = NULL; star.node.prev = NULL; star.radius = 5; star.status = 12; star.step = 3; star.x = 300; star.y = 300;
    cleardevice();
    //首星星的结构体指针
    _STAR* L_star = NULL;
    //存其他的添加的星星结构体
    _STAR* s_star = NULL;
    //初始化星星首节点
    starInit(L_star);
    //添加 MAX_STAR 个星星
    for (int i = 0; i < MAX_STAR; i++) {
        s_star = (_STAR*)malloc(sizeof(_STAR));
        //初始化分配空间里的星星各个属性
        initStar(s_star);
        //分配星星里 node 的链表,形成双向链表[后插法]
        linkInsert_back(&(L_star->node), &(s_star->node));
    }

    HWND hwnd = GetHWnd();
    //显示星空
    starDisplay(L_star);
    //星星不断移动
    while (t <90000) {
        //（尾部致谢词）
        moveStar(L_star);
        Sleep(50);
        setfillcolor(BLACK);
        if (rand() % 100 == 0) { s_time += 500; star.x = rand() % 800 + 50; star.y = rand() % 575 + 50; }
        if (s_time > 0) { setfillcolor(star.color);    solidcircle(star.x, star.y, star.radius); s_time -= 50; }
        t += 50;
        if (t < 20000 * n)sprintf(s, _T("%d/%d"), t, 20000 * n);
        else sprintf(s, _T("%d/%d, Press ESC to exit"), t, 210000 * n - 120000);
        if (t == 60000)cleardevice();
        outtextxy(0, 0, s);
        if (_kbhit()) { key = _getch(); if (key == 27 && t >= 20000 * n) { mciSendString("close bkmusic", NULL, 0, NULL); return t; } }

    }
    mciSendString("close bkmusic", NULL, 0, NULL); //  先把前面一次的音乐关闭
    return t;
}
/****************************小 y 的斗转星移绘制结束（*该子项目取材于课程示例项目）****************************/
typedef struct clocktime
{
    int hourc;
    int minutec;
    int secondc;
    int hour;
    int minute;
    int second;
    int number;
    int inter;
    int type;
    int jump;
    struct clocktime* next;
}Clocktime;
Clocktime* CreatingClocktime() {
    int i = 0;
    Clocktime* head = NULL, * end = NULL, * p;
    head = (Clocktime*)malloc(sizeof(Clocktime));
    srand(time(NULL));
    end = head;
    for (i = 0; i < 48; i++) {

        p = (Clocktime*)malloc(sizeof(Clocktime));
        if (p)
        {
            if (i != 20) {
                p->hourc = rand() % 12;
                p->minutec = rand() % 60;
                p->secondc = rand() % 60;
                p->hour = p->hourc; p->minute = p->minutec; p->second = p->secondc;
                p->number = i;
                p->inter = rand() % 7 * 100 + 700;
                p->type = rand() % 5 + 1;
                p->jump = rand() % 100 * 10;
                p->next = NULL;
            }
            else {
                p->hour = 0; p->minute = 0; p->second = 0; p->number = i; p->type = 1; p->inter = 1000; p->jump == 0; p->next = NULL;
            }
        }
        end->next = p;
        end = p;
    }
    return head;
}
int _out(Clocktime* pt,int *t)
{
    //settextstyle(18, 0, "微软雅黑");
    float i = 0;
    while (pt->next != NULL)
    {
        pt = pt->next;
        if (pt->type == 1) {
            circle(pt->number % 8 * 100 + 100, int(pt->number / 8) * 100 + 100, 50);
            setlinestyle(PS_SOLID, 2);          line(pt->number % 8 * 100 + 100, pt->number / 8 * 100 + 100, pt->number % 8 * 100 + 100 + 15 *
cos(pt->hour * PI / 6 + pt->minute * PI / 360 - PI / 2), pt->number / 8 * 100 + 100 + 15 * sin(pt->hour * PI / 6 + pt->minute * PI / 360 - PI / 2));
            setlinestyle(PS_SOLID, 1);          line(pt->number % 8 * 100 + 100, pt->number / 8 * 100 + 100, pt->number % 8 * 100 + 100 + 30 *
cos(pt->minute * PI / 30 - PI / 2), pt->number / 8 * 100 + 100 + 30 * sin(pt->minute * PI / 30 - PI / 2));
            setlinestyle(PS_DASH, 1);           line(pt->number % 8 * 100 + 100, pt->number / 8 * 100 + 100, pt->number % 8 * 100 + 100 + 45 *
cos(pt->second * PI / 30 - PI / 2), pt->number / 8 * 100 + 100 + 45 * sin(pt->second * PI / 30 - PI / 2));
            setlinestyle(PS_SOLID, 2);
        }
        else if (pt->type == 2) {
            settextstyle(24, 0, "微软雅黑");
            sprintf(s, _T("%02d:%02d:%02d"), pt->hour, pt->minute, pt->second);
            outtextxy(pt->number % 8 * 100 + 65, int(pt->number / 8) * 100 + 100, s);
        }
        else if (pt->type == 3) {
            circle(pt->number % 8 * 100 + 100 + 15 * cos(pt->hour * PI / 6 + pt->minute * PI / 360 - PI / 2), pt->number / 8 * 100 + 100 + 15 *
sin(pt->hour * PI / 6 + pt->minute * PI / 360 - PI / 2), 10);
            circle(pt->number % 8 * 100 + 100 + 30 * cos(pt->minute * PI / 30 - PI / 2), pt->number / 8 * 100 + 100 + 30 * sin(pt->minute * PI / 30 - PI /
2), 8);
            circle(pt->number % 8 * 100 + 100 + 45 * cos(pt->second * PI / 30 - PI / 2), pt->number / 8 * 100 + 100 + 45 * sin(pt->second * PI / 30 - PI /
2), 6);
        }
        else if (pt->type == 4) {
            setfillcolor(RGB(102, 102, 102)); solidpie(pt->number % 8 * 100 + 50, int(pt->number / 8) * 100 + 50, pt->number % 8 * 100 + 150,
int(pt->number / 8) * 100 + 150, -pt->second * PI / 30 + PI / 2, PI / 2);
            setfillcolor(RGB(172, 172, 172)); solidpie(pt->number % 8 * 100 + 65, int(pt->number / 8) * 100 + 65, pt->number % 8 * 100 + 135,
int(pt->number / 8) * 100 + 135, -pt->minute * PI / 30 + PI / 2, PI / 2);
            setfillcolor(RGB(217, 217, 217)); solidpie(pt->number % 8 * 100 + 80, int(pt->number / 8) * 100 + 80, pt->number % 8 * 100 + 120,
int(pt->number / 8) * 100 + 120, -pt->hour * PI / 6 + pt->minute * PI / 360 + PI / 2, PI / 2);
            circle(pt->number % 8 * 100 + 100, int(pt->number / 8) * 100 + 100, 50);
        }
        else if (pt->type == 5) {
            settextstyle(16, 0, "微软雅黑");          sprintf(s, _T("%02d:%02d"), pt->minute, pt->second);
            outtextxy(pt->number % 8 * 100 + 90, int(pt->number / 8) * 100 + 95, s);
            for (i = 0; i < pt->hour; i++) { circle(pt->number % 8 * 100 + 100 + 40 * cos(i * PI / 6 - PI / 2), pt->number / 8 * 100 + 100 + 40 * sin(i * PI /
6 - PI / 2), 8); }
        }
        //sprintf(s, _T("%02d:%02d:%02d/%d"),pt->hour,pt->minute,pt->second,pt->inter/100);
        //outtextxy(pt->number % 8 * 100 + 50, int(pt->number / 8) * 100 + 50, s);
        settextstyle(24, 0, "微软雅黑");
    }
    return 0;
}
void reckon(Clocktime *pt,int *ti) {
    while (pt->next != NULL)
    {
        pt = pt->next;
        pt->second = pt->secondc + ( * ti-pt->jump) / pt->inter;
        pt->minute = pt->minutec + (*ti - pt->jump) / 60.0 / pt->inter;
        pt->hour = pt->hourc + (*ti - pt->jump) / 60.0 / 60.0 / pt->inter;
        while (pt->second >= 60) { pt->second -= 60; pt->minute++; }
        while (pt->minute >= 60) { pt->minute -= 60; pt->hour++; }
        if (pt->hour == 12) { pt->hour = 0; }
        if (pt->number == 20 && *ti == 5000) {
            pt->hour = rand() % 12;
            pt->minute = rand() % 60;
            pt->second = rand() % 60;
            pt->inter = rand() % 7 * 100 + 700;
            pt->type = rand() % 4 + 1;
        }
    }
}
int Phantomspacetime() {
    int i, point = 0, t = 0, ta = 0;
    int* ti = &t;
    SYSTEMTIME time;
    IMAGE bk;
    loadimage(&bk, _T("bk51.jpg"));
    Clocktime* cl;
    cleardevice();
    putimage(0, 0, &bk);
    settextstyle(30, 0, "微软雅黑"); settextcolor(BLACK);
    setbkmode(TRANSPARENT);
    for (i = 1; i < 7; i++) {
        switch (i)
        {
        case(1):sprintf(s, _T("感谢您参与到项目 5-2：幻象时空（规则与“心理时钟密室”相同）")); break;
        case(2):sprintf(s, _T("在屏幕的右下角有一个红色按钮，按下回车，中央的时钟将开始倒计时，")); break;
        case(3):sprintf(s, __T("当你认为计时达到 30 秒，再次单击回车，倒计时停止。")); break;
        case(4):sprintf(s, _T("分数评定如下所示，此项目至多 10 分如果你的分数小于等于 0 分，须重新开始。")); break;
        case(5):sprintf(s, _T("*游戏内的钟表基于系统时间运行，由于算法因素，请*不要*在接近整点时运行")); break;
        case(6):settextstyle(16, 0, "微软雅黑"); sprintf(s, _T("os:这种小游戏就没有必要用秒表了吧。")); break;
        }
        outtextxy(40, 10 + 40 * i, s);
    }
    setbkmode(OPAQUE); settextcolor(WHITE);
    key = _getch();
    key = 0;
    cleardevice();
    cl = CreatingClocktime();
    GetLocalTime(&time);
    ta = time.wMinute * 60000 + time.wSecond * 1000 + time.wMilliseconds;
    BeginBatchDraw();
    while (key != 13) {
        cleardevice();
        GetLocalTime(&time);                t = time.wMinute * 60000 + time.wSecond * 1000 + time.wMilliseconds - ta;
        if (t < 5000) {
            settextcolor(RED);              settextstyle(24, 0, "微软雅黑");
            sprintf(s, _T("00:%02d.%02d"), t / 1000, t % 1000 / 10);            outtextxy(20 % 8 * 100 + 65, int(20 / 8) * 100 + 100, s);
            settextcolor(WHITE);
        }
        _out(cl, ti);
        FlushBatchDraw();
        sprintf(s, _T("%d"), ta);           outtextxy(0, 0, s);
        reckon(cl, ti);
        Sleep(10);
        if (_kbhit()) { key = _getch(); }
    }
    GetLocalTime(&time);
    t = time.wMinute * 60000 + time.wSecond * 1000 + time.wMilliseconds - ta;
    point = int(10 - sqrt(abs(t - 30000) / 60.0));
    if (point < 0)point = 0;
    sprintf(s, _T("time:%d ms/30000 ,point:%d"), t, point);        outtextxy(0, 0, s);
    EndBatchDraw();
    Sleep(2000);
    return point;
}
/****************************小 y 的幻象时空绘制结束（参照于“心理时钟密室”）****************************/

int Starworld() {
    int point = 0, project = 0;
    int i,t;
    settextstyle(24, 0, "微软雅黑");
    cleardevice();
    IMAGE bk;
    loadimage(&bk, _T("bk50.jpg"));
    putimage(0, 0, &bk);
    while (project != 2) {
        for (i = 1; i < 8; i++) {
            switch (i)
            {
            case(1):sprintf(s, _T("这里是小 y 课设的第五个项目，也是最后一个项目：小 y 的星河世界，感谢您的参与：")); break;
            case(2):sprintf(s, _T("本环节需要您依次参与以下三个星河实验，请敲击%d  按键进入相应的子项目"), project + 1); break;
            case(3):sprintf(s, __T("#项目 1：璀璨星河")); break;
            case(4):sprintf(s, _T("看一眼吧，没有要求，也没有规定，看完就算完成。")); break;
            case(5):sprintf(s, _T("")); break;
            case(6):sprintf(s, _T("项目 2：幻象时空")); break;
            case(7):sprintf(s, _T("请进入子项目后查看相关信息。")); break;
            }
            outtextxy(40, 10 + 36 * i, s);
        }
        key = _getch();
        switch (key)
        {
        case('1'):if (project == 0) { t = ending(1); project++; savep(5, 1, 0, t); }break;
        case('2'):if (project == 1) { point = Phantomspacetime(); savep(5, 2, point); if (point > 0)project++; else project--; } break;
        default:break;
        }
    }
    return point;
}
/*********************************************************    小    y    的    星    河    世    界    绘    制    结    束
*********************************************************/
/*********************************************************    小    y    的    星    河    世    界    绘    制    结    束
*********************************************************/
void save(int *a,int m) {//将游戏存档入文件里
    char name[24];
    int code[6] = { 0 };
    srand(time(NULL));
    code[0] = rand() % 9 + 1; code[1] = rand() % 9 + 1; code[2] = rand() % 9 + 1; code[3] = rand() % 9 + 1; code[4] = rand() % 9 + 1; code[5] = rand() % 9 +
1;
    if (m==0)  InputBox(s, 10, _T("项目开始，请输入用户名："));
    else if (m==1) InputBox(s, 10, _T("项目结束，请再次输入用户名：\n 如果您对本次课设提存在一些建议或意见，可以以文字或图片的形式填写
在弹出的链接当中，谢谢！同时希望能够获得文件中的 save.txt。"));
    FILE* fp = NULL;
    int error;
    name[20] ='\0';
    strcpy_s(name, s);
    if(m==0)   error = fopen_s(&fp, "./save.txt", "w");
    else   error = fopen_s(&fp, "./save.txt", "a");//这里的返回值是，如果成功返回 0，如果不成功返回非 0
    if (error != 0)
    {
        printf("打开失败");
    }
    if (fp)
    {
        if (m == 1 && *(a + 4) > 5)            fprintf_s(fp, "==========User:%s Complete Code:%d%d%d%d%d%d========== =====End=====\n",
name, code[0], code[1], code[2], code[3], code[4], code[5]);
        else if (m == 1)        fprintf_s(fp, "==========User:%s========== =====End=====\n", name);
        else if (m == 0)        fprintf_s(fp, "==========User:%s========== =====Begin=====\n", name);
    }
    _fcloseall();
}
/****************************关于文件的应用****************************/
void initialization(int* p, int n,int *t)//初始界面绘制
{
    key = 0;
    clearcircle(50, 50, 40);
    setbkmode(TRANSPARENT);
    IMAGE   img1, img01, img02, img03, img04, img05, img06;
    loadimage(&img1, _T("bk1.jpg")); loadimage(&img01, _T("bk01.jpeg")); loadimage(&img02, _T("bk02.jpg")); loadimage(&img03, _T("bk03.jpeg"));
loadimage(&img04, _T("bk04.jpg")); loadimage(&img05, _T("bk05.jpeg")); loadimage(&img06, _T("bk06.jpeg"));
    putimage(0, 0, &img1);
    putimage(0, 0, &img1);
    settextstyle(24, 0, "微软雅黑");
    setfillcolor(RGB(202, 100, 234));
    solidroundrect(420, 615, 830, 665, 20, 20);
    switch (n + 1)
    {
    case(1):sprintf(s, _T("第%d 项：小 y 的■■■"), n + 1);          outtextxy(435, 630, s); putimage(468, 290, &img01); break;
    case(2):sprintf(s, _T("第%d 项：小 y 的翻牌实验"), n + 1);       outtextxy(435, 630, s); putimage(468, 290, &img02); break;
    case(3):sprintf(s, _T("第%d 项：小 y 的前庭后院"), n + 1);       outtextxy(435, 630, s); putimage(468, 290, &img03); break;
    case(4):sprintf(s, _T("第%d 项：小 y 的■■■"), n + 1);          outtextxy(435, 630, s); putimage(468, 290, &img04); break;
    case(5):sprintf(s, _T("第%d 项：小 y 的星河世界"),n + 1);        outtextxy(435, 630, s); putimage(468, 290, &img05); break;
    case(6):sprintf(s, _T("End"));                                      outtextxy(435, 630, s); putimage(468, 290, &img06); break;

    default:                break;
    }
    switch (*(p + n))
    {
    case(-2):sprintf(s, _T("Hidden")); break;
    case(-1):sprintf(s, _T("Locked")); break;
    case(0):sprintf(s, _T("Incomplete")); break;
    case(1):sprintf(s, _T("Grade：F")); break;
    case(2):sprintf(s, _T("Grade：E")); break;
    case(3):sprintf(s, _T("Grade：D")); break;
    case(4):sprintf(s, _T("Grade：C")); break;
    case(5):sprintf(s, _T("Grade：B")); break;
    case(6):sprintf(s, _T("Grade：A")); break;
    case(7):sprintf(s, _T("Grade：S")); break;
    case(8):sprintf(s, _T("Grade：SS")); break;
    case(9):sprintf(s, _T("Grade：SSS")); break;
    default:sprintf(s, _T("Wrong Status"));       break;
    }
    outtextxy(655, 630, s);
    setbkcolor(BLACK);
    settextstyle(16, 0, "微软雅黑");
    settextcolor(BLACK);
    sprintf(s, _T("*按“s”保存数据，请输入在弹出的窗口中，此部分部分图片由 DALL.E mini 绘制。 %d/1200"),*t);
    outtextxy(10, 660, s);
    setbkmode(OPAQUE);
    settextcolor(WHITE);
    Sleep(300);
    *t = *t + 3;
    solidpie(20, 20, 80, 80, PI / 2, *t * 2 * PI / 1200 + PI / 2);
    FlushBatchDraw();
}
/*********************************************************    初    始    界    面    绘    制    结    束
*********************************************************/
/*********************************************************    初    始    界    面    绘    制    结    束
*********************************************************/
using namespace std;

int main()
{
    initgraph(WIDTH, HEIGHT, EW_SHOWCONSOLE | EW_NOCLOSE);
    int finish[6] = { -2,0,0,-2,-1,-1 };
    int now = 0, t = 0;
    int* a = &finish[0], * ti = &t;
    save(a, 0);
    while (finish[4] > -2 && finish[4] < 8 && finish[5] == -1) {
        if (*a * *(a + 1) * *(a + 2) * *(a + 3) == 0) { mciSendString("open HOME.mp3 alias initmusic", NULL, 0, NULL); }
        else { mciSendString("open TheRightPath.mp3 alias initmusic", NULL, 0, NULL); }
        mciSendString("play initmusic", NULL, 0, NULL);
        initialization(finish, now, ti);
        BeginBatchDraw();
        if (finish[0] * finish[1] * finish[2] * finish[3] != 0) finish[4] = 0;
        if (t > 1200)finish[4] = -2;
        if (_kbhit()) {
            key = _getch();
            switch (key)
            {
            case('a'):if (now > 0)now = now - 1; break;
            case(75):if (now > 0)now = now - 1; break;
            case('d'):if (now < 5)now = now + 1; break;
            case(77):if (now < 5)now = now + 1; break;
            case('9'):if (finish[4] <= 0)finish[4] = -2; else { mciSendString("close initmusic", NULL, 0, NULL); EndBatchDraw(); finish[5] = 0; } break;
            case(27):if (finish[4] <= 0)finish[4] = -2; else { mciSendString("close initmusic", NULL, 0, NULL); EndBatchDraw(); finish[5] = 0; } break;
            case('1'):now = 0; break;
            case('2'):now = 1; break;
            case('3'):now = 2; break;
            case('4'):now = 3; break;
            case('5'):now = 4; break;
            case('6'):now = 5; break;
            case(13):switch (now + 1)
            {
            case(2):if  (finish[1]  !=  -1  &&  finish[1]  !=  -2)  {  mciSendString("close  initmusic",  NULL,  0,  NULL);  EndBatchDraw();  finish[1]  =
centregame();    t = 0; } break;
            case(3):if  (finish[2]  !=  -1  &&  finish[2]  !=  -2)  {  mciSendString("close  initmusic",  NULL,  0,  NULL);  EndBatchDraw();  finish[2]  =
Vestibularbackyard();    t = 0; }    break;
            case(5):if (finish[4] != -1) { mciSendString("close initmusic", NULL, 0, NULL); EndBatchDraw(); finish[4] = Starworld();     t = 0; } break;
            }
            }
            cleardevice();
        }
    }
    if (finish[4] == -2) {
        return 0;
    }
    else {
        initialization(a, now,ti);
        save(a, 1);
        //（信息收集表单网址）
        ending(2);
        return 0;
    }
}
```