```
#include<stdio.h>
#include<time.h>
#include<windows.h>
#include<stdlib.h>
#include <string.h>
#include <errno.h>
#include<conio.h>
#define U 1
#define D 2
#define L 3
#define R 4       //蛇的状态，U：上 ；D：下；L:左 R：右

//定义宏常量
#define MAX_ID 11
#define MAX_PWD 20
#define MAX_NAME 15
#define MAX_SEX 5
#define MAX_PHONE 12
using namespace std;

//定义全局变量 name 用来显示正在游戏中的用户
char Username[MAX_NAME] ;


//创建用户的结构体
typedef struct Users
{
 char id[MAX_ID];
 char pwd[MAX_PWD];
 char name[MAX_NAME];
 char sex[MAX_SEX];
 char phone[MAX_PHONE];
}Users;

//日志结构体
 struct UserGameLog {
    char username[50];
    char password[50];
    time_t start_time;
    time_t end_time;
    int score;
};
 UserGameLog logs[10];
```

```
typedef struct SNAKE //蛇身的一个节点
{
    int x;
    int y;
    struct SNAKE* next;
}snake;

//全局变量//
int score = 0, add = 10;//总得分与每次吃食物得分。
int status, sleeptime = 200;//每次运行的时间间隔
snake* head, * food;//蛇头指针，食物指针
snake* q;//遍历蛇的时候用到的指针
int endgamestatus = 0; //游戏结束的情况，1：撞到墙；2：咬到自己；3：主动退出游戏。

//声明全部函数//
void Pos();
void creatMap();
void initsnake();
int biteself();
void createfood();
//void cantcrosswall();
//void snakemove();
void cantcrosswall(UserGameLog *s,int k);
void snakemove(UserGameLog *s,int k);
void pause();
void gamecircle(UserGameLog *s,int k);
//void gamecircle();
void welcometogame();
//void endgame();
void endgame(UserGameLog *s,int k);
void gamestart();
//声明函数

//打印菜单
void menu();

//用户注册
void Register();

//登录
void Login();

//找回密码
```

```
void Reback();

//定位光标
void gotoxy();

//获取 x 的位置
int posx();

//获取 y 的位置
int posy();

//密码输入（含掩盖功能）
void Getpwd(char* pwd);


void Pos(int x, int y)//设置光标位置
{
    COORD pos;
    HANDLE hOutput;
    pos.X = x;
    pos.Y = y;
    hOutput = GetStdHandle(STD_OUTPUT_HANDLE);
    SetConsoleCursorPosition(hOutput, pos);
}

void creatMap()//创建地图
{
    int i;
    for (i = 0; i < 58; i += 2)//打印上下边框
    {
        Pos(i, 0);
        printf("■");
        Pos(i, 26);
        printf("■");
    }
    for (i = 1; i < 26; i++)//打印左右边框
    {
        Pos(0, i);
        printf("■");
        Pos(56, i);
        printf("■");
    }
}
```

```
void initsnake()//初始化蛇身
{
    snake* tail;
    int i;
    tail = (snake*)malloc(sizeof(snake));//从蛇尾开始，头插法，以 x,y 设
定开始的位置//
    tail->x = 24;
    tail->y = 5;
    tail->next = NULL;
    for (i = 1; i <= 4; i++)
    {
        head = (snake*)malloc(sizeof(snake));
        head->next = tail;
        head->x = 24 + 2 * i;
        head->y = 5;
        tail = head;
    }
    while (tail != NULL)//从头到为，输出蛇身
    {
        Pos(tail->x, tail->y);
        printf("■");
        tail = tail->next;
    }
}

int biteself()//判断是否咬到了自己
{
    snake* self;
    self = head->next;
    while (self != NULL)
    {
        if (self->x == head->x && self->y == head->y)
        {
            return 1;
        }
        self = self->next;
    }
    return 0;
}

void createfood()
{
    snake* food_1;
    srand((unsigned)time(NULL));
```

```
    food_1 = (snake*)malloc(sizeof(snake));
    do
    {
        food_1->x = rand() % 52 + 2;
        food_1->y = rand() % 24 + 1;
        q = head;
        while (q != NULL)
        {
            if (q->x == food_1->x && q->y == food_1->y) // 判断蛇身是否
与食物重合
            {
                food_1->x = rand() % 52 + 2;
                food_1->y = rand() % 24 + 1;
                q = head; // 重新检查食物位置
            }
            else
            {
                q = q->next;
            }
        }
    } while ((food_1->x % 2) != 0); // 保证其为偶数，使得食物能与蛇头对
其

    Pos(food_1->x, food_1->y);
    food = food_1;
    printf("■");
}
void cantcrosswall( UserGameLog *s,int k)//不能穿墙
{
    if (head->x == 0 || head->x == 56 || head->y == 0 || head->y ==
26)
    {
        endgamestatus = 1;
        endgame(s,k);
    }
}

void saveUserGameLogsToFile(struct UserGameLog logs[], int size) {
    FILE *file = fopen("GameLog.txt", "a"); // 以追加模式打开文件，如果
文件不存在则创建

    if (file == NULL) {
        printf("Error opening file.\n");
        return;
```

```
    }

    for (int i = 0; i < size; i++) {
        fprintf(file, "Username: %s\n", logs[i].username);
        fprintf(file, "Password: %s\n", logs[i].password);
        fprintf(file, "Start Time: %s", ctime(&logs[i].start_time));
        fprintf(file, "End Time: %s", ctime(&logs[i].end_time));
        fprintf(file, "Score: %d\n\n", logs[i].score);
    }

    fclose(file);
}




void snakemove( UserGameLog *s,int k)//蛇前进,上 U,下 D,左 L,右 R
{
    snake* nexthead;
    cantcrosswall(s,k);

    nexthead = (snake*)malloc(sizeof(snake));
    if (status == U)
    {
        nexthead->x = head->x;
        nexthead->y = head->y - 1;
        if (nexthead->x == food->x && nexthead->y == food->y)//如果下
一个有食物//
        {
            nexthead->next = head;
            head = nexthead;
            q = head;
            while (q != NULL)
            {
                Pos(q->x, q->y);
                printf("■");
                q = q->next;
            }
            score = score + add;
            createfood();
        }
        else                                                    //如果没有食物
//
        {
```

```
            nexthead->next = head;
            head = nexthead;
            q = head;
            while (q->next->next != NULL)
            {
                Pos(q->x, q->y);
                printf("■");
                q = q->next;
            }
            Pos(q->next->x, q->next->y);
            printf("  ");
            free(q->next);
            q->next = NULL;
        }
    }
    if (status == D)
    {
        nexthead->x = head->x;
        nexthead->y = head->y + 1;
        if (nexthead->x == food->x && nexthead->y == food->y)  //有食
物
        {
            nexthead->next = head;
            head = nexthead;
            q = head;
            while (q != NULL)
            {
                Pos(q->x, q->y);
                printf("■");
                q = q->next;
            }
            score = score + add;
            createfood();
        }
        else                                //没有食物
        {
            nexthead->next = head;
            head = nexthead;
            q = head;
            while (q->next->next != NULL)
            {
                Pos(q->x, q->y);
                printf("■");
                q = q->next;
```

```
        }
        Pos(q->next->x, q->next->y);
        printf("  ");
        free(q->next);
        q->next = NULL;
    }
}
if (status == L)
{
    nexthead->x = head->x - 2;
    nexthead->y = head->y;
    if (nexthead->x == food->x && nexthead->y == food->y)//有食物
    {
        nexthead->next = head;
        head = nexthead;
        q = head;
        while (q != NULL)
        {
            Pos(q->x, q->y);
            printf("■");
            q = q->next;
        }
        score = score + add;
        createfood();
    }
    else                                //没有食物
    {
        nexthead->next = head;
        head = nexthead;
        q = head;
        while (q->next->next != NULL)
        {
            Pos(q->x, q->y);
            printf("■");
            q = q->next;
        }
        Pos(q->next->x, q->next->y);
        printf("  ");
        free(q->next);
        q->next = NULL;
    }
}
if (status == R)
{
```

```
        nexthead->x = head->x + 2;
        nexthead->y = head->y;
        if (nexthead->x == food->x && nexthead->y == food->y)//有食物
        {
            nexthead->next = head;
            head = nexthead;
            q = head;
            while (q != NULL)
            {
                Pos(q->x, q->y);
                printf("■");
                q = q->next;
            }
            score = score + add;
            createfood();
        }
        else                                        //没有食物
        {
            nexthead->next = head;
            head = nexthead;
            q = head;
            while (q->next->next != NULL)
            {
                Pos(q->x, q->y);
                printf("■");
                q = q->next;
            }
            Pos(q->next->x, q->next->y);
            printf("  ");
            free(q->next);
            q->next = NULL;
        }
    }
    if (biteself() == 1)       //判断是否会咬到自己
    {
        endgamestatus = 2;
        endgame(s,k);
    }
}


void pause()//暂停
{
    while (1)
```

```
    {
        Sleep(300);
        if (GetAsyncKeyState(VK_SPACE))
        {
            break;
        }

    }
}
void gamecircle(UserGameLog *s,int k)//控制游戏
{
    Pos(64, 5);
    printf("%s 正在游戏中",Username);

    Pos(64, 15);
    printf("不能穿墙，不能咬到自己\n");
    Pos(64, 16);
    printf("用↑.↓.←.→分别控制蛇的移动.");
    Pos(64, 17);
    printf("F1 为加速，F2 为减速\n");
    Pos(64, 18);
    printf("ESC ： 退出游戏.space：暂停游戏.");
    Pos(64, 20);
    status = R;
    while (1)
    {
        Pos(64, 10);
        printf("得分：%d  ", score);
        Pos(64, 11);
        printf("每个食物得分：%d 分", add);
        if (GetAsyncKeyState(VK_UP) && status != D)
        {
            status = U;
        }
        else if (GetAsyncKeyState(VK_DOWN) && status != U)
        {
            status = D;
        }
        else if (GetAsyncKeyState(VK_LEFT) && status != R)
        {
            status = L;
        }
        else if (GetAsyncKeyState(VK_RIGHT) && status != L)
        {
```

```
            status = R;
        }
        else if (GetAsyncKeyState(VK_SPACE))
        {
            pause();
        }
        else if (GetAsyncKeyState(VK_ESCAPE))
        {
            endgamestatus = 3;
            break;
        }
        else if (GetAsyncKeyState(VK_F1))
        {
            if (sleeptime >= 50)
            {
                sleeptime = sleeptime - 30;
                add = add + 2;
                if (sleeptime == 320)
                {
                    add = 2;//防止减到 1 之后再加回来有错
                }
            }
        }
        else if (GetAsyncKeyState(VK_F2))
        {
            if (sleeptime < 350)
            {
                sleeptime = sleeptime + 30;
                add = add - 2;
                if (sleeptime == 350)
                {
                    add = 1;  //保证最低分为 1
                }
            }
        }
        Sleep(sleeptime);
        snakemove( s,k);
    }
}

void welcometogame()//开始界面
{

    Pos(40, 12);
```

```
    printf("欢迎来到贪食蛇游戏！ ");
    Pos(40, 25);
    system("pause");
    system("cls");
    Pos(25, 12);
    printf("用↑.↓.←.→分别控制蛇的移动， F1 为加速，2 为减速\n");
    Pos(25, 13);
    printf("加速将能得到更高的分数。\n");
    system("pause");
    system("cls");
}

void endgame(UserGameLog *s,int k)//结束游戏
{

    system("cls");
    Pos(24, 12);
    if (endgamestatus == 1)
    {
        printf("对不起，您撞到墙了。游戏结束!");
    }
    else if (endgamestatus == 2)
    {
        printf("对不起，您咬到自己了。游戏结束!");
    }
    else if (endgamestatus == 3)
    {
        printf("您已经结束了游戏。");
    }
    Pos(24, 13);
    printf("您的得分是%d\n", score);
    s[k].score=score;
    s[k].end_time=time(NULL);
   saveUserGameLogsToFile(s,k);
  //  goto end; ///
   exit(0);
}

void gamestart()//游戏初始化
{
    system("mode con cols=100 lines=30");
    welcometogame();
    creatMap();
    initsnake();
```

```
    createfood();
}

//打印菜单
void menu()
{
 printf("*********************************************\n");
 printf("***********欢迎来到贪吃蛇小游戏************\n");
 printf("***********选择对应的序号进入程序***********\n");
 printf("*****1.注册新用户         2.登录************\n");
 printf("*****3.找回密码           0.退出程序 *******\n");
 printf("*********************************************\n");
 printf("*********************************************\n");
}

//注册系统
void Register() {
    Users a; // 用于接收用户输入的临时变量
    FILE* pf;

    printf("欢迎来到注册界面!\n");

    // 尝试以二进制读模式打开文件
    pf = fopen("users.dat", "rb");
    if (!pf) {
        // 如果文件不存在（ENOENT），则创建文件
        if (errno == ENOENT) {
            pf = fopen("users.dat", "wb");
            if (!pf) {
                printf("创建文件失败: %s\n", strerror(errno));
                return;
            }
            // 如果文件创建成功，则可以直接写入，无需检查重复用户
        } else {
            // 其他错误
            printf("打开文件失败: %s\n", strerror(errno));
            return;
        }
    }
        printf("请输入您的账号>>");
        scanf("%19s", a.id); // 使用%19s 防止缓冲区溢出

    // 读取用户的其他信息
    printf("请输入您的姓名>>");
```

```
    scanf("%49s", a.name); // 使用%49s 防止缓冲区溢出
    printf("请输入您的性别>>");
    scanf("%9s", a.sex); // 使用%9s 防止缓冲区溢出
    printf("请输入您的电话号码>>");
    scanf("%19s", a.phone); // 使用%19s 防止缓冲区溢出
    printf("请输入您的密码>>");
    Getpwd(a.pwd);
    printf("\n 请再次确认您的密码>>");
    char tmp[20];
    Getpwd(tmp);

    // 验证两次密码是否匹配
    do {
        if (strcmp(a.pwd, tmp) == 0) {
            // 打开文件以追加模式
            pf = fopen("users.dat", "ab");
            if (!pf) {
                printf("打开文件失败以写入信息: %s\n", strerror(errno));
                return;
            }
            fwrite(&a, sizeof(Users), 1, pf);
            printf("\n 账号注册成功,请登录!\n");
            fclose(pf);
            return;
        } else {
            printf("\n 两次密码不匹配!请重新输入>>");
            Getpwd(a.pwd);
            printf("\n 请再次确认>>");
            Getpwd(tmp);
        }
    } while (1);
}
//登录系统
void Login()
{
 Users  a, b;//同理，a 是用来给用户输入的，b 是从文件中读取到 b 中，用 b 和 a
匹配比较

 FILE* pf = fopen("users.dat", "rb");//以只读的方式打开文件
 if (!pf)//如果读取失败
 {
  printf("%s\n", strerror(errno));
  return;
 }
```

```
printf("欢迎来到登录界面!\n");
//Sleep(1000);

fread(&b, sizeof(Users), 1, pf);

printf("请输入账号>>");
scanf("%s", a.id);
//name=a.name;


while (1)
{
 if (!strcmp(a.id, b.id))//在文件中找到了与用户输入相同的 id
 {
  break;
 }
 else
 {
  if (!feof(pf))//没读到文件末尾，继续读取文件中的 id 到 b 中
  {
   fread(&b, sizeof(Users), 1, pf);//继续从文件中读取用户信息进 b，直到
在文件中找到一个和 a 的信息相同的
  }
  else//读到文件末尾了，没有找到与用户输入相同的账号
  {
   printf("此账号不存在!请重新输入!\n");
   Sleep(500);
   fclose(pf);
   pf = NULL;
   return ;
  }
 }
}
do
{
 printf("请输入密码>>");
 Getpwd(a.pwd);//获取密码，
 if (!strcmp(a.pwd, b.pwd))//输入的密码与文件中的相同
 {
  printf("\n 登录成功!欢迎使用!\n");

   strcpy(Username, b.name);
   strcpy(logs[0].username,b.name);
   strcpy( logs[0].password,b.pwd);
```

```
    //printf("测试语句测试名字%s\n",b.name);
   Sleep(500);
   fclose(pf);//用完当然要把文件关啦
   pf = NULL;//置空，避免野指针
   return;
  }
  else
  {
   printf("\n 密码输入错误,请重新输入\n");
  }
 } while (strcmp(a.pwd, b.pwd));


}

//找回密码
void Reback()
{
 char tmp[20] = "";//密码匹配用的
 Users a, b;

 FILE* pf = fopen("users.dat", "rb+");//"rb+"是为了读和写以二进制打开文件的意思

 if (!pf)//老样子，先判断能不能顺利打开
 {
  printf("%s", strerror(errno));
  return;
 }

 fread(&b, sizeof(Users), 1, pf);//照样，读一个试试水

 printf("请输入您的账号>>");
 Sleep(800);

 scanf("%s", a.id);

 while (1)//在文件中找到与用户输入相同的 id
 {
  if (!strcmp(a.id, b.id))//如果读取到了相同的 id（在文件中找到了和用户输入一样的）
  {
   break;
  }
```

```
 else
 {
  if (!feof(pf))//没读到文件尾，继续读
  {
   fread(&b, sizeof(Users), 1, pf);
  }
  else
  {
   printf("您输入的账号不存在!请重新输入!\n");
   Sleep(500);
   fclose(pf);
   pf = NULL;
   break;
  }
 }
}

//下面是信息匹配验证
do//匹配姓名
{
 printf("请输入您的姓名>>");
 scanf("%s", a.name);
 if (!strcmp(a.name, b.name))
 {
  break;
 }
 else
 {
  printf("输入错误,请重新输入!\n");
 }
} while (strcmp(a.name, b.name));

do//匹配性别
{
 printf("请输入您的性别>>");
 scanf("%s", a.sex);
 if (!strcmp(a.sex, b.sex))
 {
  break;
 }
 else
 {
  printf("输入错误,请重新输入!\n");
 }
```

```
} while (strcmp(a.sex, b.sex));
do//匹配电话号码
{
 printf("请输入您的电话号码>>");
 scanf("%s", a.phone);
 if (!strcmp(a.phone, b.phone))
 {
  break;
 }
 else
 {
  printf("输入错误,请重新输入!\n");
 }
} while (strcmp(a.phone, b.phone));

//更改密码
printf("验证成功!请修改您的密码!\n");
printf("请输入您的密码>>");
Getpwd(a.id);
printf("请再次确认您的密码>>");
Getpwd(tmp);
if (!pf)
{
 printf("%s", strerror(errno));
 return;
}
//将原来的密码覆盖掉
do
{
 if (!strcmp(a.pwd, tmp))//两次密码相等
 {
  fseek(pf,-((int)(sizeof(Users)-MAX_ID)), SEEK_CUR);//将文件流调回到
要修改的密码的位置
  fprintf(pf, "%s", a.pwd);//覆盖原来的密码
  printf("密码修改成功,请登录!\n");
  Sleep(500);
  fclose(pf);
  pf = NULL;
  return;
 }
 else
 {
  printf("两次密码不匹配!请重新输入>>");
  scanf("%s", a.pwd);
```

```
   printf("请再次确认>>");
   scanf("%s", tmp);
  }
 } while (1);

}

//定位光标
void gotoxy(int x, int y)
{
 //更新光标位置
 COORD pos;
 HANDLE hOutput = GetStdHandle(STD_OUTPUT_HANDLE);//GetStdHandle 是一个 Windows API 函数。
 pos.X = x;
 pos.Y = y;
 SetConsoleCursorPosition(hOutput, pos);
}
//获取光标 x 坐标
int posx()
{
 CONSOLE_SCREEN_BUFFER_INFO ptr;
 GetConsoleScreenBufferInfo(GetStdHandle(STD_OUTPUT_HANDLE), &ptr);
 return (ptr.dwCursorPosition.X);
}
//获取光标 y 坐标
int posy()
{
 CONSOLE_SCREEN_BUFFER_INFO ptr;
 GetConsoleScreenBufferInfo(GetStdHandle(STD_OUTPUT_HANDLE), &ptr);
 return (ptr.dwCursorPosition.Y);
}

//输入密码
void Getpwd(char* pwd)
{
 int i = 0;
 int x, y;
 while (1)
 {
  pwd[i] = getch();//获取单个密码（字符）
  if (pwd[i] == VK_BACK && i >= 0)//如果输入的是回退键，VK_BACK 是键盘的键值，ASCII 码值为 8
  {
```

```
   if (i > 0)//i>0 说明有输入东西了，则回退前一格
   {
    i--;
    x = posx() - 1;//定位 x 并回退一格
    y = posy();//定位 y
    gotoxy(x, y);//定位光标
    printf(" ");//将*用空格掩盖
    x = posx() - 1;//再次回退，下次输入时光标才会显示在正确的位置
    y = posy();//定位 y
    gotoxy(x, y);
    continue;//然后跳过此次循环
   }
   else if (i == 0)//i==0 说明没输入东西，直接跳过此次循环即可
   {
    continue;
   }
  }
  if (i >= 0 && pwd[i] != VK_RETURN && pwd[i] != VK_BACK)//输入东西了
  {
   x = posx();
   y = posy();
   gotoxy(x, y);
   printf("*");
  }
  if (i == 0 && pwd[i] == VK_RETURN)//如果没输入东西直接按回车，直接跳过
此次循环，避免程序把回车当密码了
  {
   continue;
  }
  if (pwd[i] == VK_RETURN||i==MAX_PWD-2)//输入回车了或者到上限了
  {
   i++;
   pwd[i] = '\0';//结尾放'\0'
   break;
  }
  i++;
 }
}

void printFunction()
{
    FILE *file;
    char ch;
```

```
    // 打开文件
    file = fopen("GameLog.txt", "r");

    // 检查文件是否成功打开
    if (file == NULL) {
        printf("无法打开文件\n");
        return;
    }

    // 逐个字符读取并打印文件内容
    while ((ch = fgetc(file)) != EOF) {
        printf("%c", ch);
    }

    // 关闭文件
    fclose(file);
}

int main()
{
    char input;
    int k=0;
    logs[k].start_time = time(NULL);
     do {
        menu();
        input = getch(); // 使用 getch() 捕获键盘输入

        if (input == 0) {
            input = getch(); // 捕获功能键
            if (input == 63) { // F5 的键值为 63
                printFunction();
                printf("游戏日志打印成功！\n");
            }
        } else {
            switch (input) {
                case '1':
                    Register();
                    break;
                case '2':
                    Login();
                    gamestart();
                    gamecircle(logs, k);
                    endgame(logs, k);
                    break;
```

```
                case '3':
                    Reback();
                    break;
                case '0':
                    printf("退出成功!\n");
                    break;
                default:
                    printf("选择错误,请重新选择!\n");
                    break;
            }
        }
    } while (input != '0');
    return 0;
}
```