目录

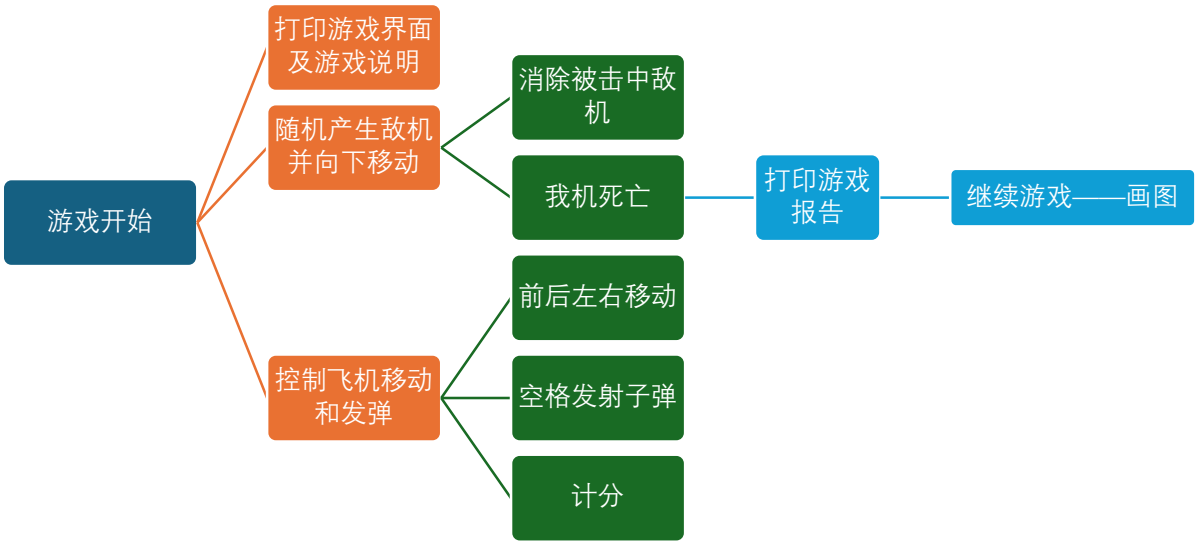
1 小程序设计概要

1.1 设计目标

使用 C 语言设计一个贪吃蛇规模的小程序，学习用 C 语言设计程序，为期末考试打下坚实基础。

1.2 程序功能逻辑设计

a) 游戏开始时，显示游戏说明和控制方式——上下左右控制飞机移动和空格键发射子弹，按 ESC 键可以退出游戏。
b) 隐藏光标，并绘制一个固定的边框作为游戏界面以及飞机的初始位置。
c) 不断检测玩家的键盘输入。kongzhi() 函数用于控制飞机的移动和操作。根据玩家输入的方向键，更新飞机的坐标，并在界面上移动飞机的位置。空格键发射子弹，子弹向上移动，并实时进行碰撞检测和计分。
d) dfeiji() 函数用于随机产生敌机并控制敌机的移动。敌机会向下移动，当敌机与玩家的飞机坐标重合时，游戏结束。玩家可以通过发射子弹消灭敌机，并得到相应的得分。游戏过程中，实时更新并显示玩家的得分，包括击落的敌机数量和总得分。
e) 当游戏结束时，显示游戏结束信息，包括总得分、敌机数、歼敌数和命中率。玩家可以按任意键复活以继续游戏。



1.3 程序模块逻辑框图

<table>
<tr><td colspan="7">main</td></tr>
<tr><td rowspan="2">hidden</td><td rowspan="2">gotoxy</td><td rowspan="2">huatu</td><td>shuoming</td><td colspan="2">kongzhi</td><td colspan="2">dfeiji</td></tr>
<tr><td>gotoxy</td><td>gotoxy</td><td>jifen</td><td>gotoxy</td><td colspan="2">byebye</td></tr>
<tr><td colspan="5"></td><td>gotoxy</td><td>huatu</td></tr>
</table>

2 技术原理

2.1 不堵塞地得到按键输入

对于非阻塞的键盘输入，标准 C 库并没有直接的支持。为了能够不堵塞地得到按键输入，我使用了一个非标准的 C 库 conio.h 。通过 kbhit()函数检查是否有按键输入，并使用 getch()获取按键信息，从而实现不堵塞地获取按键输入。

2.2 绘制矩形画面

使用函数 printf()打印一个 21×21 的全*界面，通过 printf()函数和 gotoxy()函数控制光标位置，将中间的 19×19 打印空格，形成矩形画面。

```
// 画图
void huatu()
{
    int i, n;

    for (i = 0; i <= 20; i++)
    {
        for (n = 0; n <= 20; n++)
        {
            printf("*");
        }
        printf("\n");
    }
    for (i = 1; i <= 19; i++)
    {
        for (n = 1; n <= 19; n++)
        {
            gotoxy(i, n);
            printf(" ");
        }
    }
}
```

2.3 基于像素点画面绘制及画面刷新

①通过 printf()函数和 gotoxy()函数控制光标位置并在对应位置打印。②使用 sleep()函数设置适当的延迟时间再进行下一操作，实现画面刷新。如以下代码，通过 printf()函数和 gotoxy()函数，在三个对应位置打印飞机，使用 sleep()函数设置适当的延迟时间，使得敌机每 900 毫秒下降 1 次。

```
        while (t)
        {
            r++;
            r1++;
            r2++;
```

```
            gotoxy(d, r);
            printf("Ψ");
            gotoxy(d1, r1);
            printf("Ж");
            gotoxy(d2, r2);
            printf("♀");
            Sleep(900);
            gotoxy(d, r);
            printf(" ");
            gotoxy(d1, r1);
            printf(" ");
            gotoxy(d2, r2);
            printf(" ");

            kongzhi(0, 0);
            byebye();
            if (r == 18)
                r = 0;
            if (r1 == 18)
                r1 = 0;
            if (r2 == 18)
                r2 = 0;
            if (r == 0 || r1 == 0 || r2 == 0)
                break;
        }
```

2.4 游戏存档

在游戏结束时，可以显示玩家的得分、敌机数量、歼敌数信息。按任意键可以回到函数继续游戏。

```
printf(" !!! 游戏结束 !!!\n"
               "********************\n"
               " 您的总得分: %d\n\n"
               " 敌机数: %d\n"
               " 歼敌数: %d\n"
               " 命中率: %.0f %%\n"
               "********************\n",
               f, m, j, ((float)j / (float)m) * 100);
```

3 程序详细解释

3.1 头文件

引入四个头文件。

stdio.h 头文件定义标准输入输出函数。

windows.h 头文件是 C 语言中调用 Windows 操作系统 API 的主要入口，它包含了大量 Windows API 函数的声明，允许程序员调用这些函数执行各种与操作系统交互的任务，在本程序主要用于 coord()函数和 sleep()函数。

conio.h 控制台输入输出, 在本程序主要用于 kbhit()函数和 getch()函数。

time.h 是日期和时间头文件，用于需要时间方面的函数。本程序利用 time(NULL)函数获得当前时间作为种子值，从而生成伪随机数序列。

```
#include <stdio.h>
#include <windows.h>
#include <conio.h>
#include <time.h>
```

3.2 宏定义

将从键盘接收到的特定按键的扫描码与变量相关联。

```
#define Esc 27 // 退出
#define Up 72  // 上，下，左，右
#define Down 80
#define Left 75
#define Right 77
#define Kong 32 // 发射子弹
```

3.3 定义变量

```
int x = 10; // 飞机坐标
int y = 18;

int d2 = 10; // 敌机坐标
int d1 = 10;
int d = 10;
int r = 1;
int r1 = 1;
int r2 = 1;

int t = 1; // 游戏结束
int f = 0; // 计分数
int m = 3; // 敌机数
int j = 0; // 歼敌数
char p;    // 接受按键
```

3.4 声明函数

```
void kongzhi(int bx, int by); // 声明函数
void huatu();
```

3.5 定义 gotoxy()函数

COORD 是<windows.h>中的一个结构体，通常用于表示控制台屏幕上的坐标。这个结构体有两个成员：X 和 Y，分别代表水平和垂直坐标。然后定义了一个 COORD 类型的变量 coord，用于存储要移动到的目标坐标。将传入的 x 和 y 参数值分别赋给 coord 结构体的 X 和 Y 成员，最后使用 Windows 提供的 API 函数 SetConsoleCursorPosition，将计算得到的线性位置传递给系统，

从而实现光标的移动到指定位置。

```
void gotoxy(int x, int y) // 移动坐标
{
    COORD coord;
    coord.X = x;
    coord.Y = y;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), coord);
}
```

3.6 定义 hidden()函数

通过网上查阅，该函数使用 GetStdHandle 函数获取与调用进程相关联的标准输出设备的句柄，并将其存储在 hOut 变量中。STD_OUTPUT_HANDLE 是一个预定义的常量，表示标准输出设备（通常是控制台）。CONSOLE_CURSOR_INFO 是一个结构体，用于表示控制台光标的信息。GetConsoleCursorInfo 函数用于获取与指定控制台屏幕缓冲区关联的光标信息，并将其存储在 cci 结构体中。将 cci 结构体中的 bVisible 成员设置为 0，表示将光标设置为不可见（隐藏）。如果要将光标设置为可见，可以将 bVisible 设置为 1。使用 SetConsoleCursorInfo 函数将修改后的光标信息（即隐藏的光标）设置回控制台屏幕缓冲区。通过该函数可以隐藏光标，使界面美观。

```
void hidden() // 隐藏光标
{
    HANDLE hOut = GetStdHandle(STD_OUTPUT_HANDLE);
    CONSOLE_CURSOR_INFO cci;
    GetConsoleCursorInfo(hOut, &cci);
    cci.bVisible = 0; // 赋 1 为显示，赋 0 为隐藏
    SetConsoleCursorInfo(hOut, &cci);
}
```

3.7 定义 shuoming()函数

在界面右侧打印说明。

```
void shuoming()
{
    printf("\t\t\t\n\n\n\n");

    printf("\t\t\t 方向控制\n\n"
           "\t\t\t 上 ↑\n\n"
           "\t\t\t 下 ↓\n\n"
           "\t\t\t 左 ←\n\n"
           "\t\t\t 右 →\n\n"
           "\t\t\t 子弹 空格\n\n\n"
           "\t\t\t 退出请按 ESC\n");
    gotoxy(0, 0);
}
```

3.8 定义 byebye()函数

判断我机坐标是否和敌机坐标相等，相等即我机与敌机相碰撞，死亡，函数会移动到控制台上的特定位置（(1, 3)），并显示游戏结束的信息，包括总得分 f、敌机数 m、歼敌数 j 以及命中率（歼敌数除以敌机数）。按任意键继续，我机复活，游戏继续。

```
// 判断我机死没死/游戏结束
void byebye()
{
    if ((x == d && y == r) || (x == d1 && y == r1) || (x == d2 && y == r2))
    {
        gotoxy(1, 3);
        printf(" !!! 游戏结束 !!!\n"
               "********************\n"
               " 您的总得分: %d\n\n"
               " 敌机数: %d\n"
               " 歼敌数: %d\n"
               " 命中率: %.0f %%\n"
               "********************\n",
               f, m, j, ((float)j / (float)m) * 100);
        while (!kbhit())
        {
            Sleep(500);
            gotoxy(1, 12);
            printf(" 继续请按任意键...\n\n\n");
            Sleep(900);
            gotoxy(1, 12);
            printf(" ");
        }
        gotoxy(0, 0);
        huatu();
        f = 0;
        m = 0;
        j = 0;
        if (x >= 18)
            x--;
        else
            x++;
        gotoxy(x, y);
        printf("Ж");
    }
}
```

3.9 定义 jifen()函数

函数首先检查我机的位置（x, y）是否与三个敌机的位置（d, r），（d1, r1），（d2, r2）相匹配。如果玩家的位置与任何一个敌机的位置相匹配，则积分 f 加上对应分数，击落敌机 j 加 1，并在该位置短暂地显示一个数字（3、1 或

2），表示玩家获得的分数。然后将光标移动到相应的位置，并使用 printf 打印分数。使用 Sleep(200); 暂停 200 毫秒，然后再次使用 gotoxy 和 printf 通过打印一个空格清除该位置。

```
// 计分/更新敌机
void jifen()
{
    if (x == d && y == r)
    {
        gotoxy(d, r);
        printf("3");
        Sleep(200);
        gotoxy(d, r);
        printf(" ");
        f += 2;
        r = 0;
        j++;
    }
    if (x == d1 && y == r1)
    {
        gotoxy(d1, r1);
        printf("1");
        Sleep(200);
        gotoxy(d1, r1);
        printf(" ");
        f += 3;
        r1 = 0;
        j++;
    }
    if (x == d2 && y == r2)
    {
        gotoxy(d2, r2);
        printf("0");
        Sleep(200);
        gotoxy(d2, r2);
        printf(" ");
        f += 1;
        r2 = 0;
        j++;
    }

    gotoxy(26, 2);
    printf(" %d \n", f);
}
```

3.10 定义 huatu()函数

使用函数 printf()打印一个 21×21 的全*界面，通过 printf()函数和 gotoxy()函数控制光标位置，将中间的 19×19 打印空格，形成矩形画面。

```
// 画图
void huatu()
{
    int i, n;

    for (i = 0; i <= 20; i++)
    {
        for (n = 0; n <= 20; n++)
        {
            printf("*");
        }
        printf("\n");
    }
    for (i = 1; i <= 19; i++)
    {
        for (n = 1; n <= 19; n++)
        {
            gotoxy(i, n);
            printf(" ");
        }
    }
}
```

3.11 定义 dfeiji()函数

敌机的 x 坐标被随机生成在 1 到 17 的范围内，控制在方框内（1≤d/d1/d2≤17）随机产生敌机并显示 900 毫秒，同时执行 kongzhi()和 baibai()函数

```
// 随机产生敌机
void dfeiji()
{
    while (t)
    {
        if (!r)
        {
            d = rand() % 17 + 1;
            m++;
        }
        if (!r1)
        {
            d1 = rand() % 17 + 1;
            m++;
        }
        if (!r2)
        {
            d2 = rand() % 17 + 1;
            m++;
        }
```

```
        while (t)
        {
            r++;
            r1++;
            r2++;
            gotoxy(d, r);
            printf("Ψ");
            gotoxy(d1, r1);
            printf("Ж");
            gotoxy(d2, r2);
            printf("♀");
            Sleep(900);
            gotoxy(d, r);
            printf(" ");
            gotoxy(d1, r1);
            printf(" ");
            gotoxy(d2, r2);
            printf(" ");

            kongzhi(0, 0);
            byebye();
            if (r == 18)
                r = 0;
            if (r1 == 18)
                r1 = 0;
            if (r2 == 18)
                r2 = 0;
            if (r == 0 || r1 == 0 || r2 == 0)
                break;
        }
    }
}
```

3.12 定义 kongzhi()函数

通过 kbhit()函数检查是否有按键输入，并使用 getch()获取按键信息，用 switch()分别执行按下规定按键后的操作。

```
// 操控飞机
void kongzhi(int bx, int by)
{
    int a;

    while (kbhit())
    {
        if ((p = getch()) == -32)
            p = getch();
        a = p;
```

```
        gotoxy(22, 5);

        switch (a)
        { // 控制方向
        case Up:
            if (y != 1)
            {
                gotoxy(x, y);
                printf(" ");
                y--;
                gotoxy(x, y);
                printf("Ж");
            }
            break;
        case Down:
            if (y != 18)
            {
                gotoxy(x, y);
                printf(" ");
                y++;
                gotoxy(x, y);
                printf("Ж");
            }
            break;
        case Left:
            if (x != 1)
            {
                gotoxy(x, y);
                printf(" ");
                x--;
                gotoxy(x, y);
                printf("Ж");
            }
            break;
        case Right:
            if (x != 18)
            {
                gotoxy(x, y);
                printf(" ");
                x++;
                gotoxy(x, y);
                printf("Ж");
            }
            break;
        case Kong:
        {
            bx = y;
            for (by = y; by > 1;) // 发射子弹
```

```
            {
                by--;
                gotoxy(x, by);
                printf("0");
                Sleep(10);
                gotoxy(x, by);
                printf(" ");
                y = by;
                jifen();
                if (r == 0 || r1 == 0 || r2 == 0)
                    break;
            }
            y = bx;
        }
        break;

        case Esc:
            t = 0;
            break; // 退出

        default:
            break;
        }
    }
}
```

3.13 主函数

①在 main()函数的开始，使用 srand(time(NULL));将时间作为种子，用于随机数生成。

②在 main()函数中调用 shuoming()函数，显示游戏说明。然后调用 hidden()函数以隐藏光标，并调用 huatu()函数绘制游戏画面。

③进入 while(t)循环，其中调用 kongzhi(0, 0)函数以接受玩家输入并操纵飞机，同时调用 dfeiji()函数生成敌机并控制敌机的移动。

④当游戏结束时，显示游戏结束信息，玩家可以按任意键继续游戏。

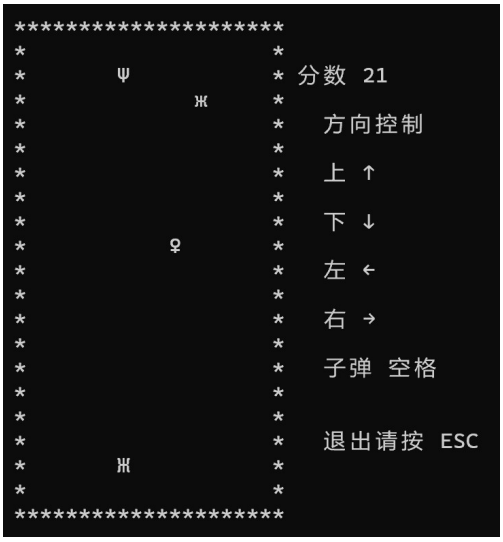
```
void main()
{
    srand(time(NULL));
    shuoming();
    hidden();
    huatu();
    gotoxy(x, y);
    printf("Ж");

    gotoxy(22, 2);
    printf("分数:");
    while (t)
    {
```

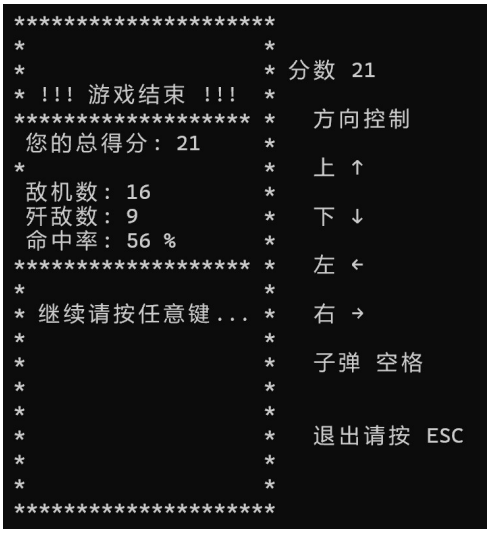
```
        kongzhi(0, 0);
        if (t)
            dfeiji();
    }
}
```

4 功能测试

4.1 游戏界面



4.2 我机死亡界面



5 问题及解决办法、心得体会

我个人认为，能够自己学以致用，做一个 C 语言小游戏是一件十分有趣的事情。因为网页上说了贪吃蛇规模的程序，我预计会有很多人会做贪吃蛇，而我想要创新，想要与众不同，于是我打开了一个充满童年回忆的网站——4399，也终于让我找到一款打飞机的游戏让我觉得简化一下，以我的能力是可以实现

的，于是，一个简化的打飞机游戏将要诞生。

可是，眼高手低的我，一开始就犯了难。打印游戏界面等一些简单问题我还能解决，但一些还没见过的问题比如如何实现多架敌机的随机生成和缓慢向下移动、如何在键盘获取上下左右的输入等的问题我可以说是一窍不通。所以我需要进一步的学习，因此我发现了一个对于程序员十分友好的网站——CSDN，在这个网站我搜索到了很多关于制作这个小程序新知识，例如 sleep()函数、rand()函数，为了方便打印飞机封装的 gotoxy()函数和隐藏光标的 hidden()函数， conio.h 库里的 kbhit()函数检查是否有按键输入以及 getch()获取按键信息，对于整个程序的设计有极大的帮助。

理解好整个游戏的流程，按功能分好各模块，就开始码代码。这一部分特别需要耐心，需要理清楚每一个语句的作用，尽管我列写了程序的框架，但也会常常搞乱。我经常遇到想起什么就加点什么的情况，使得我的思路较为混乱，另外我定义的全局变量也比较多，所以用的时候要经常去找对应哪个变量，关注它们需不需要初始化。在写报告时我顿悟了，可以加更多的注释，让代码一目了然。

当然，模块化的函数写好了，还需要主函数调用看它们之间协调的效果，这个阶段我进行了很多次调试，所以当程序真正能运行时，是相当高兴。这个小游戏还可以添加其他的一些功能，比如敌机可以发射子弹，飞机有血量限制，游戏结束时显示排行榜……还可以有很多很多功能，以后有机会可以继续完善。