

操作系统课程设计文档



项目设计与功能说明文档

小组成员：欧阳桥梁 1753112

何立仁 1753127

周泓光 1753029

一. 项目描述

1. 项目目的

通过操作系统课程设计, 我们实现了一个简单而功能完善的操作系统, 通过实践来理解一个操作系统的诞生。从而更加深入具体地了解操作系统的进程, 内存, 文件, 以及外设管理的原理。

2. 开发环境

1. Ubuntu 16.04
2. Bochs 模拟器

3. 项目完成指标

本次操作系统课程设计小组成员为 3 人, 基于《Orange's 一个操作系统的实现》, 通过修改或重新实现源码的一个或多个模块来实现一个简单得操作系统。

本次课程设计主要实现了图形化, 对外设管理部分进行重新实现, 实现了 **B 级**项目难度; 同时对文件系统进行修改, 支持多级文件系统, 实现了 **B 级**难度; 并实现了显示图片和保存图片等系统级应用, 实现了 **C 级**难度; 最后在图形化的基础上实现了一个图形化的小游戏, 完成了 **D 级**难度。

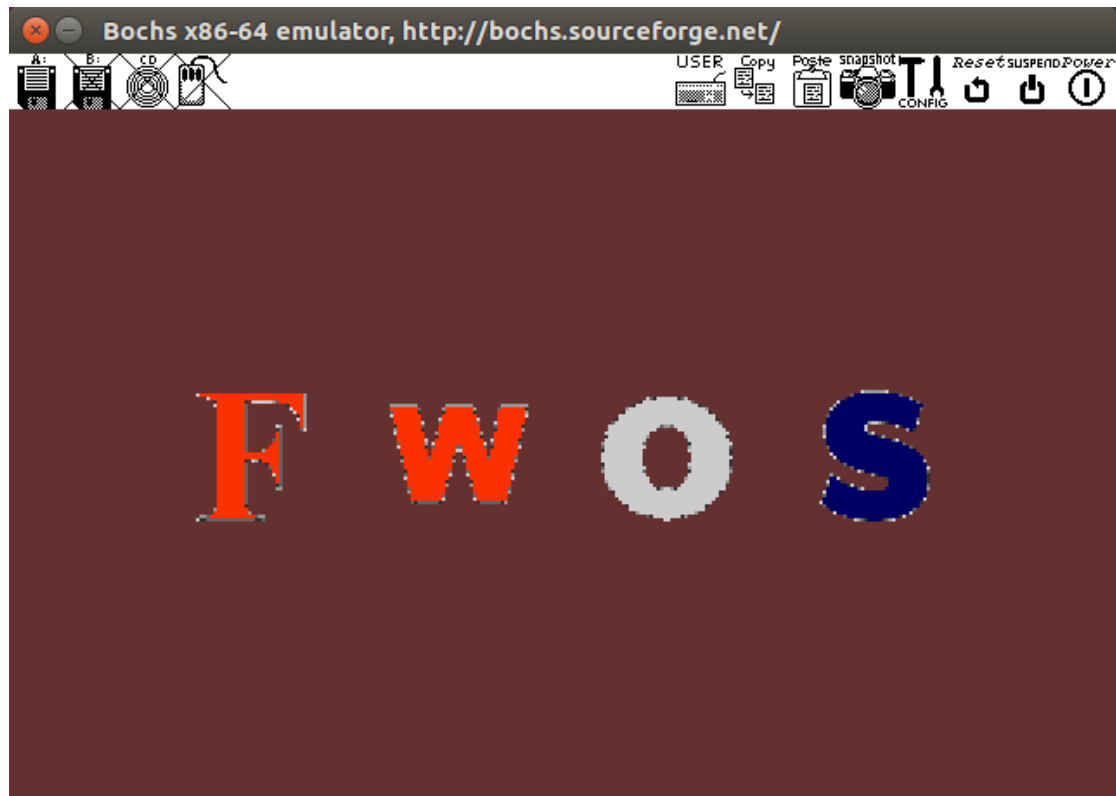
一. 功能说明

1. 图形化界面

本系统修改了外设显卡部分, 使系统支持图形化界面, 但由于时间精力有限, 我们的主要界面仍然是命令行的形式, 但是命令行中所有的文字都是使用像素点拼凑的形式展现出来的, 同时在图形化的基础上实现了开机动画, 图片显示, 以及一个小游戏。

2. 开机动画

进入系统后会有一个开机动画，不同颜色的字母 F W O S 依次出现在屏幕上，实现图形化的开机动画。



3. 图片显示

系统支持显示图片，用户输入 `showImage [filename]` 即可显示图片，显示的图片通过 `saveImage` 接口可以将 `bmp` 格式图片数组存入磁盘中，然后使用 `showImage` 即可将图片读取并显示出来。



4. 进程管理

本系统提供 `process` 命令查看系统全部进程，包括进程的 `pid`，名称，优先级以及是否在运行（YES 表示正在运行，NO 表示没有运行），具体运行结果如下图

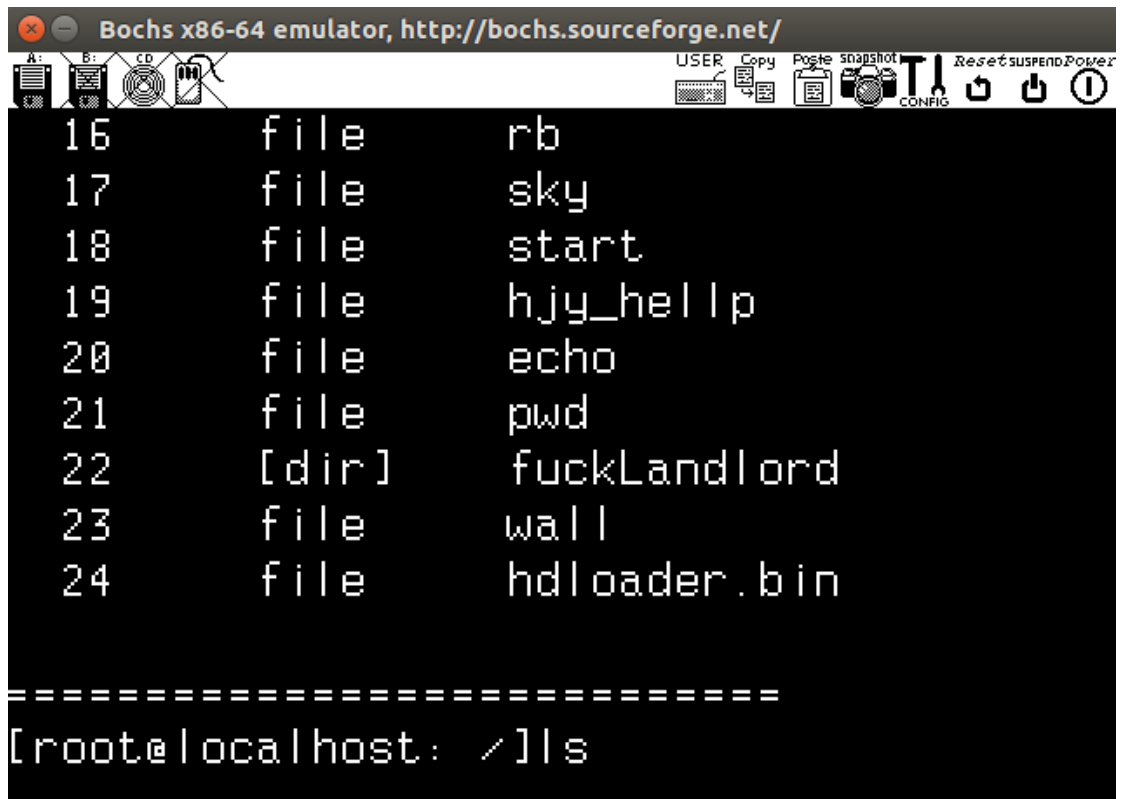
```
Bochs x86-64 emulator, http://bochs.sourceforge.net/
USER Copy Paste snapshot CONFIG Reset suspend Power
PID | name | spriority | running?
-----
0   TTY    15        YES
1   SYS    15        YES
2   HD     15        YES
3   FS     15        YES
4   MM     15        YES
5   TestA   5         YES
6   TestB   5         YES
7   TestC   5         YES
=====
[root@localhost: ~]process
```

5. 文件系统

5.1 显示文件列表

系统提供 ls 命令显示文件列表，用户输入 ls 命令可以查看当前目录下的所有文件和目录（多级目录系统）。

通过调用ls()方法向文件系统发送 LS 系统消息，当文件系统接收到消息之后会执行do_ls()方法，该方法会通过遍历当前节点下的子节点来打印当前目录下的文件信息。打印的内容包括子节点的 inode 编号，对应的文件类型和对应的文件名。

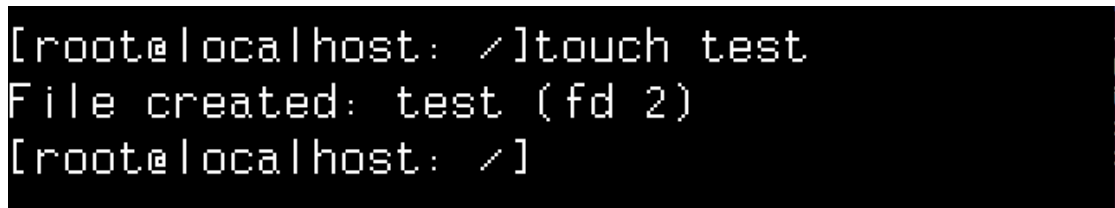
A screenshot of the Bochs x86-64 emulator window. The title bar reads "Bochs x86-64 emulator, http://bochs.sourceforge.net/". The window contains a terminal window with a list of files and their types. The files are: 16 file rb, 17 file sky, 18 file start, 19 file hjy_help, 20 file echo, 21 file pwd, 22 [dir] fuckLandlord, 23 file wall, and 24 file hdloader.bin. Below the list is a separator line of equals signs, followed by the prompt [root@localhost: ~]ls.

```
Bochs x86-64 emulator, http://bochs.sourceforge.net/
A: B: CD
16 file rb
17 file sky
18 file start
19 file hjy_help
20 file echo
21 file pwd
22 [dir] fuckLandlord
23 file wall
24 file hdloader.bin
=====
[root@localhost: ~]ls
```

5.2 创建文件

系统提供 touch 命令创建文件，用户输入 touch [filename]命令可以创建文件。

通过调用CreateFile()方法，该方法会调用open()方法来进行创建。根据方法返回值判断是否创建成功并给用户发出反馈消息。其中创建文件时会对文件进行重名检测。

A screenshot of a terminal window showing the execution of the touch command. The prompt is [root@localhost: ~], followed by the command touch test. The output is File created: test (fd 2). The prompt then returns to [root@localhost: ~].

```
[root@localhost: ~]touch test
File created: test (fd 2)
[root@localhost: ~]
```

5.3 删除文件/文件夹

系统提供 rm 命令删除文件/文件夹，用户输入 rm [filename]命令可以删除文件/文件夹。

调用DeleteFile()方法，向文件系统发送 UNLINK 消息，文件系统在接收到消息之后会调用do_unlink()方法进而实现文件或文件夹的删除。其中文件夹的删除实现是递归删除文件夹里的子节点。

```
[root@localhost: ~]# touch test
File created: test (fd 2)
[root@localhost: ~]# rm test
test deleted!
[root@localhost: ~]#
```

5.4 写文件

用户输入 “vi+filename” 指令，系统在定位到目标文件后显示文件路径，之后用户输入要写入文件的内容并回车，完成写文件操作。

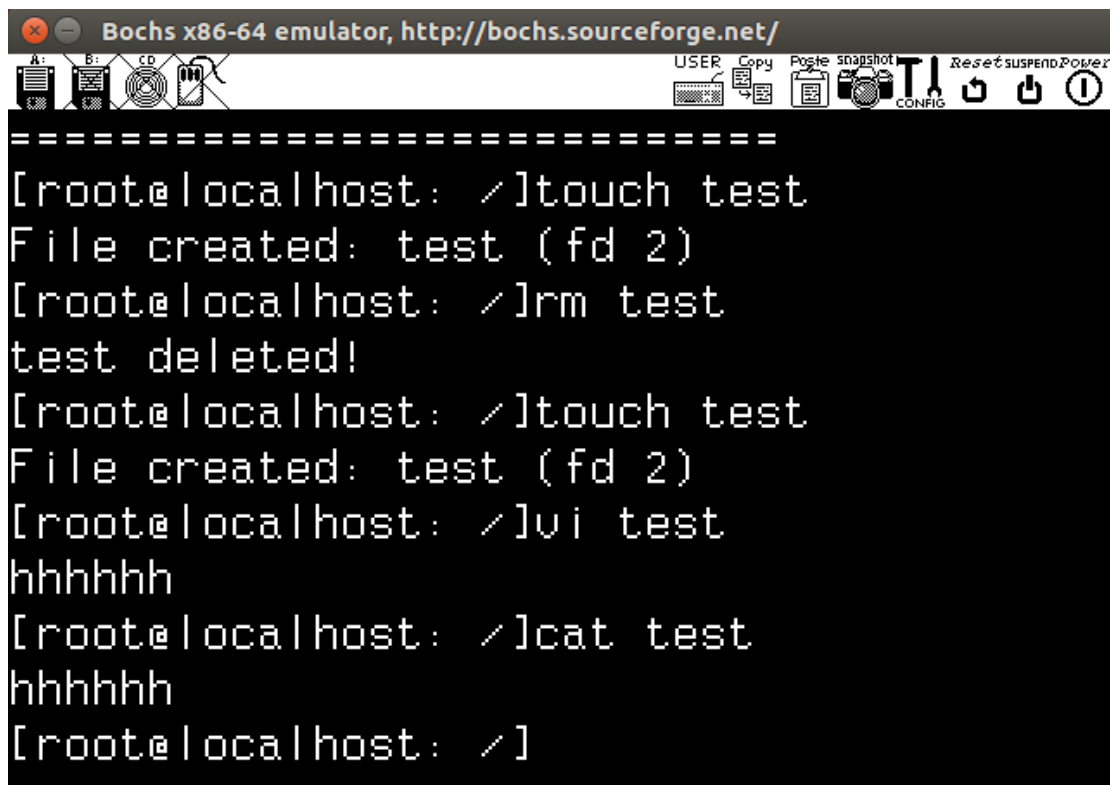
系统首先使用open()方法定位文件，然后调用write()方法对文件进行写操作。

```
[root@localhost: ~]# touch test
File created: test (fd 2)
[root@localhost: ~]# vi test
hhhhh
[root@localhost: ~]#
```

5.5 读文件

用户输入 “cat+filename” 指令，查看该文件内容，若文件中已写入内容，则如上图显示；若文件创建后未写入内容，则打印空行。

系统首先使用open()方法定位文件，然后调用read()方法对文件进行读操作。

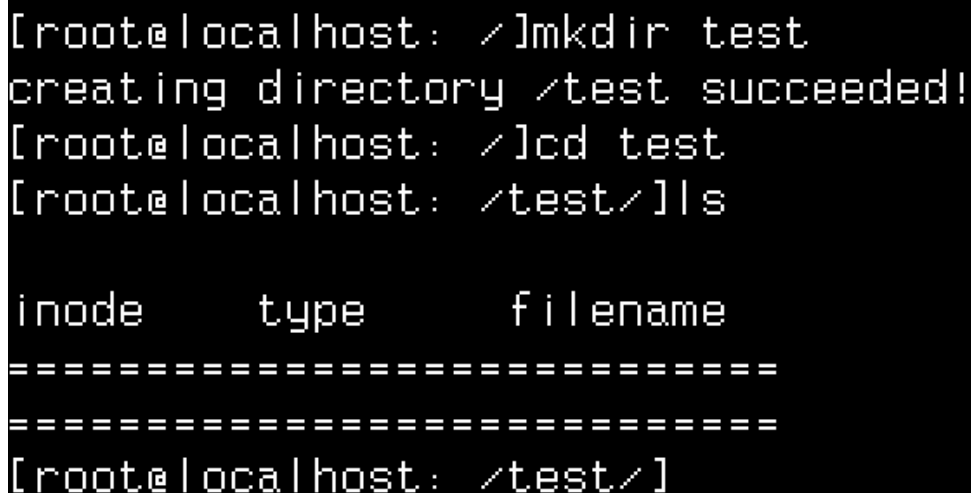
A screenshot of the Bochs x86-64 emulator window. The title bar reads "Bochs x86-64 emulator, http://bochs.sourceforge.net/". The window contains a terminal window with a black background and white text. The terminal shows a series of commands and their outputs. The commands are: touch test, rm test, touch test, vi test, and cat test. The outputs are: "File created: test (fd 2)", "test deleted!", "File created: test (fd 2)", and "hhhhh" (twice). The terminal prompt is [root@localhost: /].

```
=====
[root@localhost: /]touch test
File created: test (fd 2)
[root@localhost: /]rm test
test deleted!
[root@localhost: /]touch test
File created: test (fd 2)
[root@localhost: /]vi test
hhhhh
[root@localhost: /]cat test
hhhhh
[root@localhost: /]
```

5.6 创建目录

系统文件系统为多级文件系统，提供 mkdir 命令创建文件夹。

系统首先使用CreateDir()方法创建对应文件路径，然后调用open()方法来检测是否有同名文件，如果有的话则会报错。如果没有重名文件夹创建文件夹，成功或失败都会有相应提示信息。

A screenshot of a terminal window showing the execution of mkdir, cd, and ls commands. The commands are: mkdir test, cd test, and ls. The outputs are: "creating directory /test succeeded!", and a directory listing showing the current directory is empty. The terminal prompt is [root@localhost: /].

```
[root@localhost: /]mkdir test
creating directory /test succeeded!
[root@localhost: /]cd test
[root@localhost: /test/]ls

inode      type      filename
=====
=====
[root@localhost: /test/]
```


5.6 进入多级目录

系统文件系统为多级文件系统，提供 cd 命令进入多级目录。

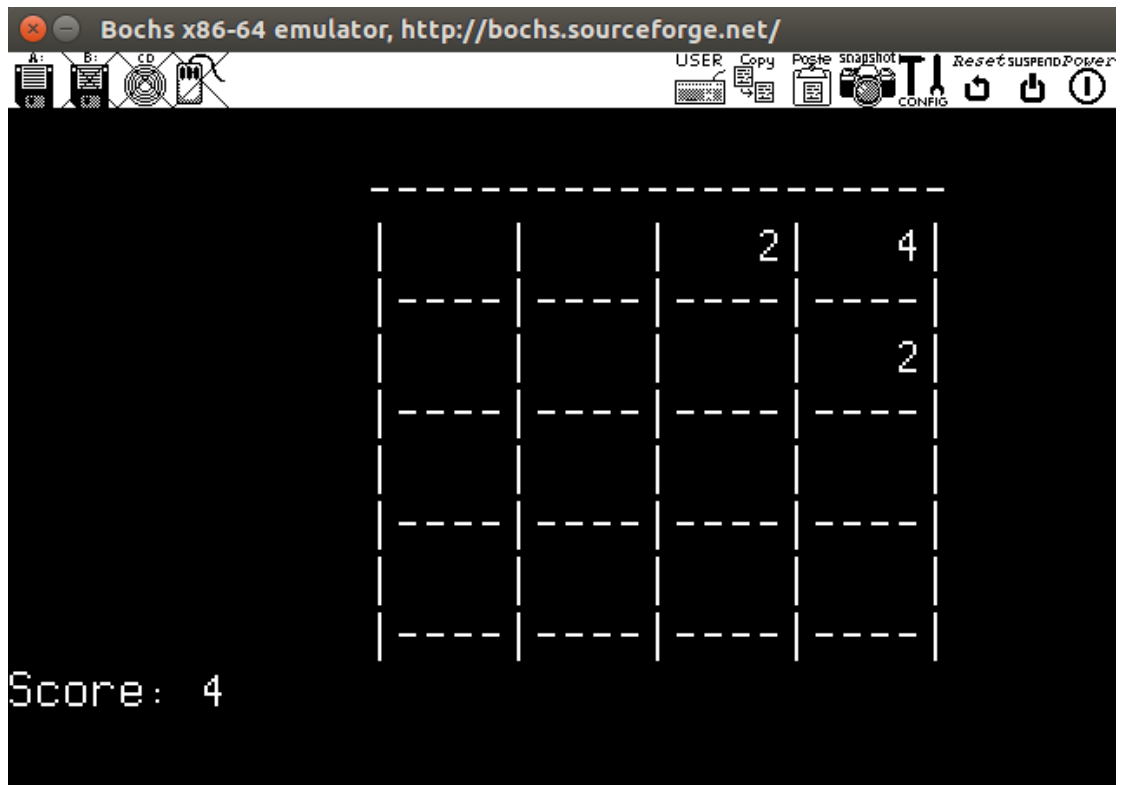
```
[root@localhost: ~]# mkdir test
creating directory /test succeeded!
[root@localhost: ~]# cd test
[root@localhost: /test/]# ls

inode      type      filename
=====
=====
[root@localhost: /test/]
```

6. 游戏

6.1 2048

用户输入 run2048 可以进入 2048 游戏。系统自动打印游戏控制方法(wsad 控制上下左右, enter 键退出游戏) 以及初始棋盘、当前分数 (初始为 0); 用户每次只能输入一个运动方向, 系统自动刷新棋盘及分数, 同时在棋盘内随机生成新数字格。



6.2 flappybird

用户输入 flappybird 可以进入 flappybird 游戏。空格可以控制小鸟上升躲避障碍物。



三. 核心代码

1. print_char()

```
void print_char(int x, int y, char c, char ch)
{
    x=x*8;
    y=y*16;
    char* font=hankaku + ch * 16;
    int i;
    int xsize=320;
    char *p, d /* data */;
    for (i = 0; i < 16; i++) {
        p = VGA_MEM_BASE + (y + i) * xsize + x;
        for(int j=0; j<8; j++)
            p[j]=0;
        d = font[i];
        if ((d & 0x80) != 0) { p[0] = c; }
        if ((d & 0x40) != 0) { p[1] = c; }
```

```

        if ((d & 0x20) != 0) { p[2] = c; }
        if ((d & 0x10) != 0) { p[3] = c; }
        if ((d & 0x08) != 0) { p[4] = c; }
        if ((d & 0x04) != 0) { p[5] = c; }
        if ((d & 0x02) != 0) { p[6] = c; }
        if ((d & 0x01) != 0) { p[7] = c; }
    }
    return;
}

```

2. _showImage()

```

void _showImage(char* path, char* filename)
{
    clear();
    char absoPath[512];
    convert_to_absolute(absoPath, path, filename);
    unsigned char size[3];
    int fd = open(absoPath, O_RDWR);
    assert(fd != -1);
    read(fd, size, 3);
    int w=size[0]+size[1];
    int h=size[2];
    int rd_bytes=w*h;
    unsigned char Img[rd_bytes];
    int n = read(fd, Img, rd_bytes);
    assert(n==rd_bytes);
    close(fd);
    char* p = (char *) VGA_MEM_BASE;
    for (int y = 0; y < h; y++) {
        for(int x = 0; x < w; x++)
        {
            p[y*320+x] = Img[y*w+x];
        }
    }
    console_table[current_console].cursor = ((h/16)+1)*8;
    waitkey();
}

```

3. GoDir()

```
void GoDir(char* path, char* file)
{
    int flag = 0; // 判断是进入下一级目录还是返回上一级目录
    char newPath[512] = {0};
    if (file[0] == '.' && file[1] == '.') // cd ..返回上一级目录
    {
        flag = 1;
        int pos_path = 0;
        int pos_new = 0;
        int i = 0;
        char temp[128] = {0}; // 用于存放某一级目录的名称
        while (path[pos_path] != 0)
        {
            if (path[pos_path] == '/')
            {
                pos_path++;
                if (path[pos_path] == 0) // 已到达结尾
                    break;
                else
                {
                    temp[i] = '/';
                    temp[i + 1] = 0;
                    i = 0;
                    while (temp[i] != 0)
                    {
                        newPath[pos_new] = temp[i];
                        temp[i] = 0; // 抹掉
                        pos_new++;
                        i++;
                    }
                    i = 0;
                }
            }
            else
            {
                temp[i] = path[pos_path];
                i++;
                pos_path++;
            }
        }
    }
}
```

```

    }
}
}
char absoPath[512];
char temp[512];
int pos = 0;
while (file[pos] != 0)
{
    temp[pos] = file[pos];
    pos++;
}
temp[pos] = '/';
temp[pos + 1] = 0;
if (flag == 1) // 返回上一级目录
{
    temp[0] = 0;
    convert_to_absolute(absoPath, newPath, temp);
}
else // 进入下一级目录
    convert_to_absolute(absoPath, path, temp);
int fd = open(absoPath, O_RDWR);
if (fd == -1)
    printf("%s is not a directory!\n", absoPath);
else
    memcpy(path, absoPath, 512);
}

```

4. flappybird 核心代码

```

void updateWithoutInput() // 与用户输入无关的更新
{
    if(bird_x+bird_width>bars[0].bar_x && bird_x <
bars[0].bar_x+bar_width)
    {
        if((bird_y>bars[0].bar_yTop)&&(bird_y+bird_hight<bars[
0].bar_yDown)){ // 小鸟未碰到挡板
        }
        else{

```

```

        gamestauts=0; // 小鸟没通过挡板开口小
    }
}

if(bird_y<high-dist_down)
{
    bird_y+=dist_down;
}
else{
    gamestauts=0; // 触地死亡
}

for(int i=0;i<bar_num;++i){ // 挡板移动
    bars[i].bar_x-=bar_speed;
}
if(bars[0].bar_x+bar_width<=0) // 挡板离开边界
{
    bar_num--;
    score++;
    if(bar_num==0){
        bars[0]=newbar();
        bar_num++;
    }
    else if(bar_num==1){
        bars[0]=bars[1];
    }else if(bar_num==2){
        bars[0]=bars[1];
        bars[1]=bars[2];
    }
}
if(bar_num<3 && bars[bar_num-1].bar_x<150 &&
rand2048()%2==0) // 产生新的挡板
{
    bars[bar_num]=newbar();
    bar_num++;
}
}

void updateWithInpute(int fd_stdin, int fd_stdout) //
与用户输入有关的更新
{
    if(mykey_pressed==1)

```

```

{
    mykey_pressed=0;
    if( (bird_y>0) && (bird_y<=high))    //小鸟向上跳
        bird_y=max(bird_y-dist_up,0);
    bird_status++;
}
else
{
    bird_status = 0;
}
}

```

四. 成员分工

姓名	学号	分工	占比
欧阳桥梁	1753112	图形化界面，显示图片，flappybird	33.3%
周泓光	1753029	图形化界面，多级文件系统，2048	33.3%
何立仁	1753127	图形化界面，开机动画，文件系统	33.3%