2019 年操作系统课程设计

E. D. I. T. H

项目设计与功能说明文档

小组成员： 朴 雪 1752124

郭玮佳 1751292

孙文颖 1750802

朱丽雯 1751169

指导教师： 王冬青

目录

1.操作系统项目简介

1.1 项目目的

通过阅读和实践，更深刻地理解进程、内存、文件系统等操作系统相关概念，在书上代码的基础上实现自己简单的操作系统。操作系统课程设计的目目的就是通过自自己己学习、实现 一一个简单而而功能完善的操作系统，来让我们真正理理解一一个操作系统是如何从无无到有、一一步步实现的。

1.2 开发环境

开发系统环境：UBUNTU 14.04.6(32BIT)

相关软件：BOCHS 2.6.9

开发语言：C 语言，汇编语言

工具：GCC, MAKE

1.3 项目成果

本项目参考《ORANGR'S 一个操作系统的实现》，对文件系统模块进行了重新实现，，通过修改或者重新实现参考源码的一一个或多个模块来实现一一个简单的操作系统。

1. 实现了多级文件系统，项目新建代码量达到文件模块的一半，达到 B 级项目难度。
2. 同时，通过调用较多 API 以实现系统级应用，实现对系统的检测和控制，如磁盘、工具台等，实现 C 级项目应用。
3. 调用较少 API 以实现用户级应用—五子棋和扫雷小游戏，实现 D 级项目应用。
4. 最后为操作系统添加了开机动画界面。

2.操作系统项目设计

2.1 设计方案

本系统参考《ORANGE'S:一个操作系统的实现》中的代码，对文件系统进行了重新实现，能够针对文件/文件目录进行增删查改操作，并且实现了多级目录。另外，通过调用系统 API，实现了应用级程序：五子棋小游戏。还为系统添加了开机动画效果。

2.2 功能结构及使用说明

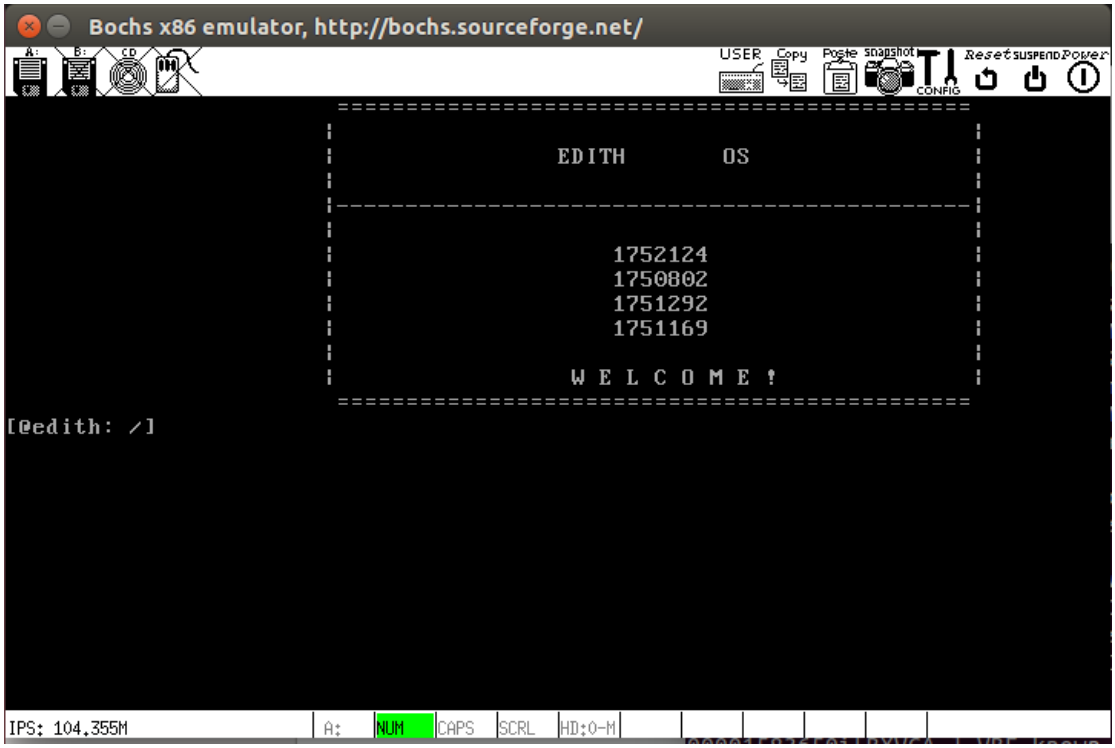
2.2.1 开机动画

◆描述：进入系统后，屏幕闪烁 UNITH 字样，随后闪出卡通图案。

◆实现方法：开机动画使用用逐帧 PRINTF 的方方式实现，通过设置较小小的延迟时间形成动画效果。

2.2.2 欢迎界面

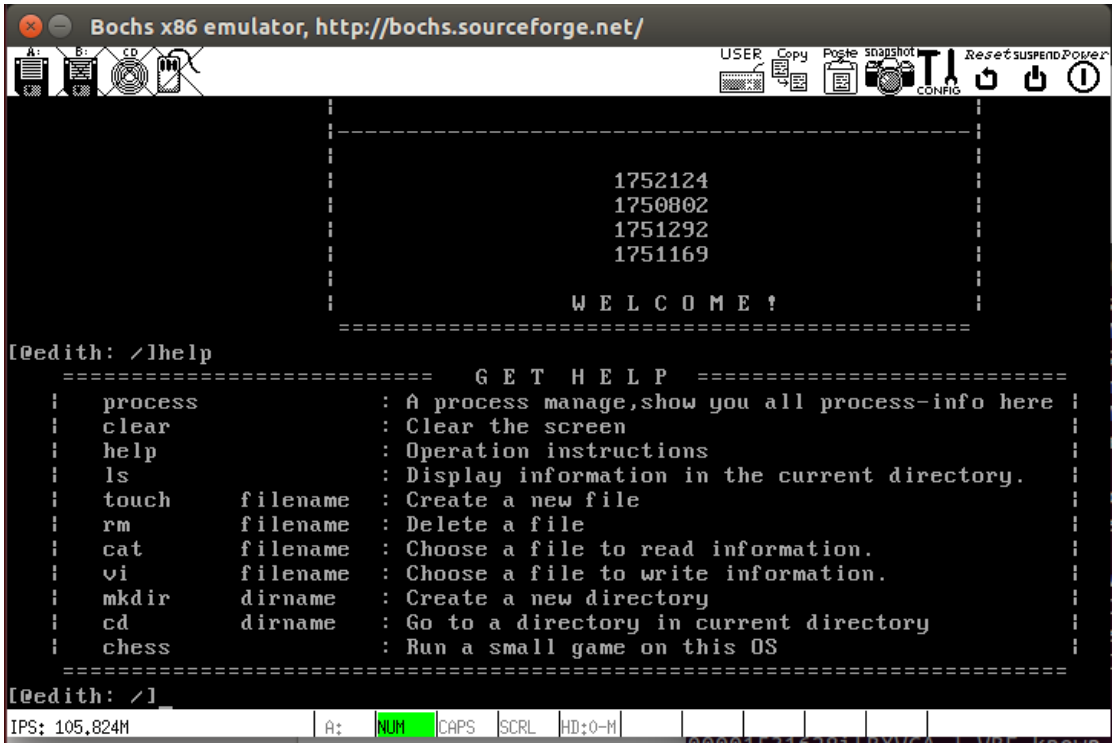
◆描述：开机动画之后打印欢迎界面。
◆实现方法：用字符串逐行输出。



2.2.3 帮助界面

◆描述：用户输入 HELP 指令后，屏幕会显示出系统所有命令列表。

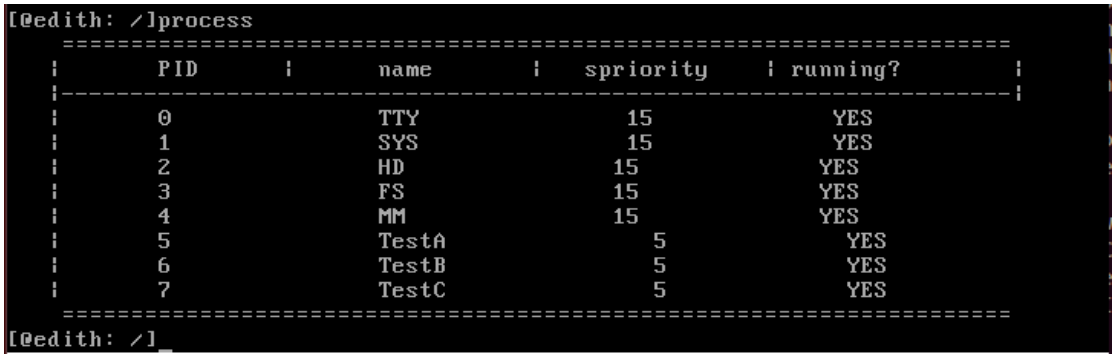
◆实现方法：用用户输入入指令后，系统通过判断字符串串调用用对应 API，进入入相应功能（详见见下文文）。



2.2.4 进程管理

◆描述：用用户输入入“PROCESS”指令，进入入进程管理理功能，系统打印出当前全部进程（包括系统进程和用用 户进程）的 PID，名称，优先级（15 表示系统级进程，5 表示用用户级进程）以及运行行行状态（YES 表示 正在运行行行，NO 表示没有运行行行）。

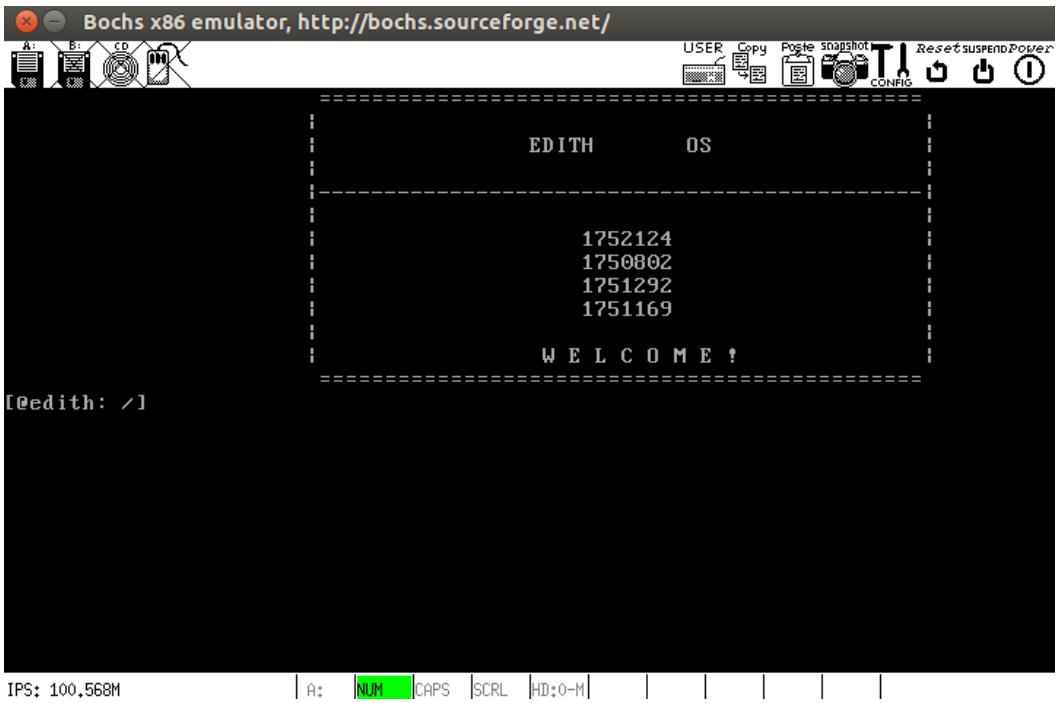
◆实现方法：通过调用用 PROCESSMANAGE() API，遍历整个进程列列表，逐个打印，同时根据其 P_flAGS 是否为 FREE_SLOT 来判断进程运行行行状态。



2.2.5 清屏功能

◆描述：用户输入 CLEAR 指令后，屏幕清屏，打印出欢迎界面。

◆实现方法：通过调用用 CLEAR() API 实现清屏功能，同时重新设置控制台相应初始值。



2.2.6 创建文件夹

◆描述：用户输入 MKDIR 指令和文件夹名后，系统创建一个新的文件夹并提示创建成功。

◆实现方法：通过调用 CREATEDIR(),查找空白盘块号，分配子目录盘块并且初始化，返回创建成功的提示。

```
The folder /111 have been created successfully.
```

2.2.7 创建文件

◆描述：用户输入 TOUCH 指令和文件名后，系统创建一个新的文件并提示创建成功。

◆实现方法：通过调用 CREATE()，在当前目录下创建文件，为文件分配磁盘块号，返回创建成功的提示。

```
File created: ppp (fd 3)
```

2.2.8 删除文件

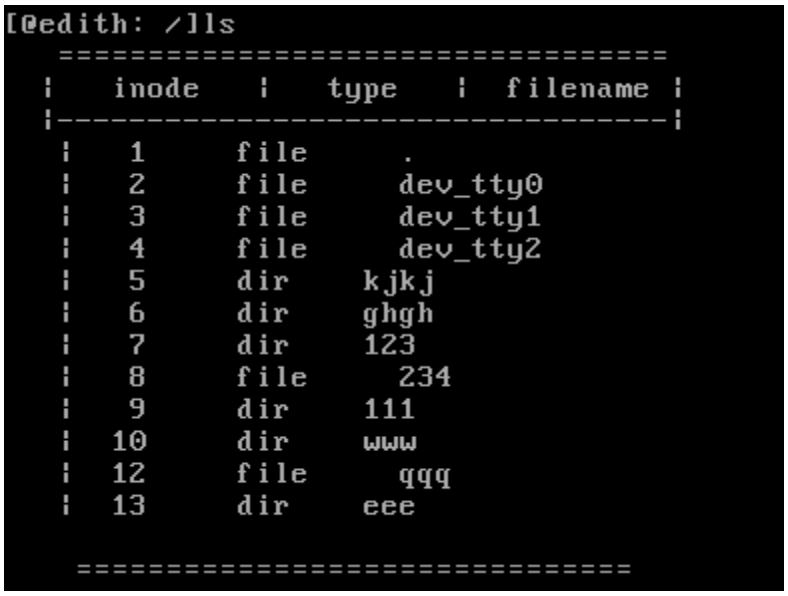
◆描述：用户输入 RM 指令和文件名后，系统删除此文件。

◆实现方法：通过调用 DELFILE()，查找该文件的盘块号，清除原本文件的内容，标记该目录项为空。

```
ppp deleted!
```

2.2.9 显示目录

◆描述：用户输入“IS”指令后，系统显示当前文件目录。
◆实现方法:通过调用 LISTSHOW(),查找普通文件和目录文件并把它们打印出来。



2.2.10 写文件

◆描述：用户输入“VI”指令和文件名后，系统进入写文件功能，用户可对此文件进行写入。

◆实现方法：通过调用 READ(),计算文件物理地址，打印文件内容。

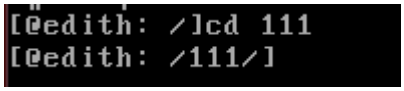
2.2.11 打印文件内容

◆描述：用户输入“CAT”指令和文件名后，系统进入读文件功能，在屏幕上打印出此文件的内容。
◆实现方法：通过调用 READ(),计算文件物理地址，打印文件内容。

2.2.12 进入多级目录

◆描述：用户输入“CD”指令和路径之后，系统自动检索目录路径，检索成功后进入当前目标目录。

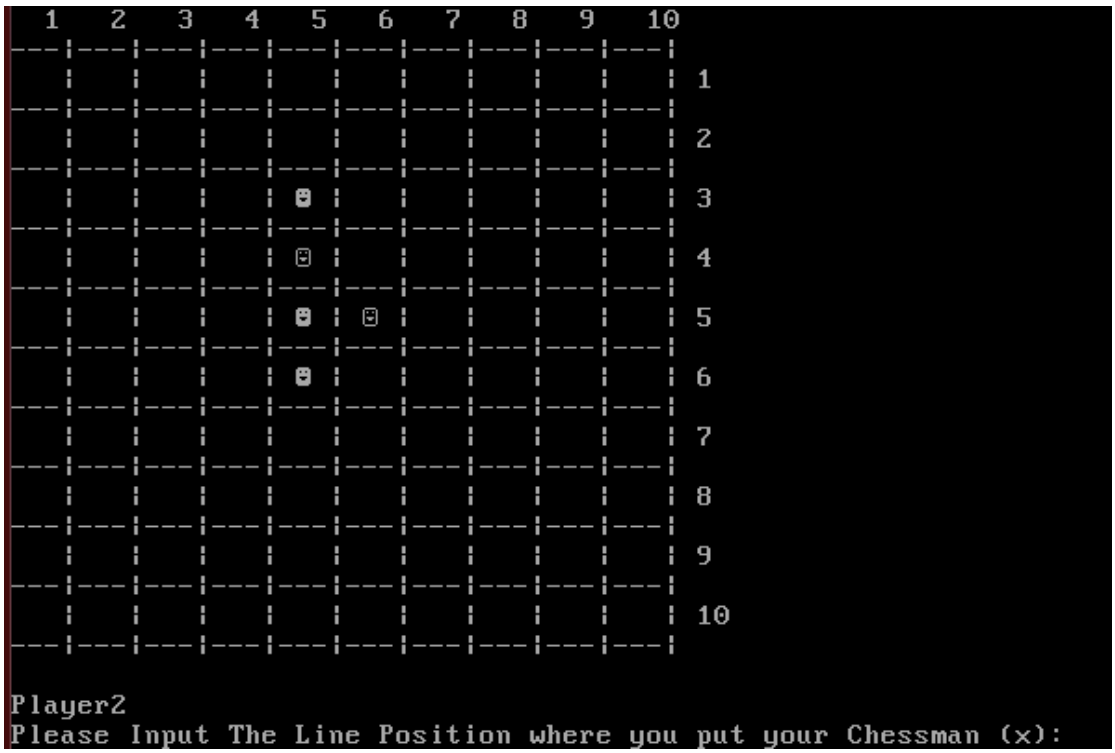
◆实现方法：通过调用 CHANGEPATH()，查找目标文件夹，修改当前路径信息。



2.2.13 五子棋游戏

◆描述：用户输入“CHESS”指令,进入五子棋游戏界面；首先系统打印空白棋盘（QP），然后用户 1 和用户 2 通过输入棋子行、列轮流下棋，直至出现死局或赢家。

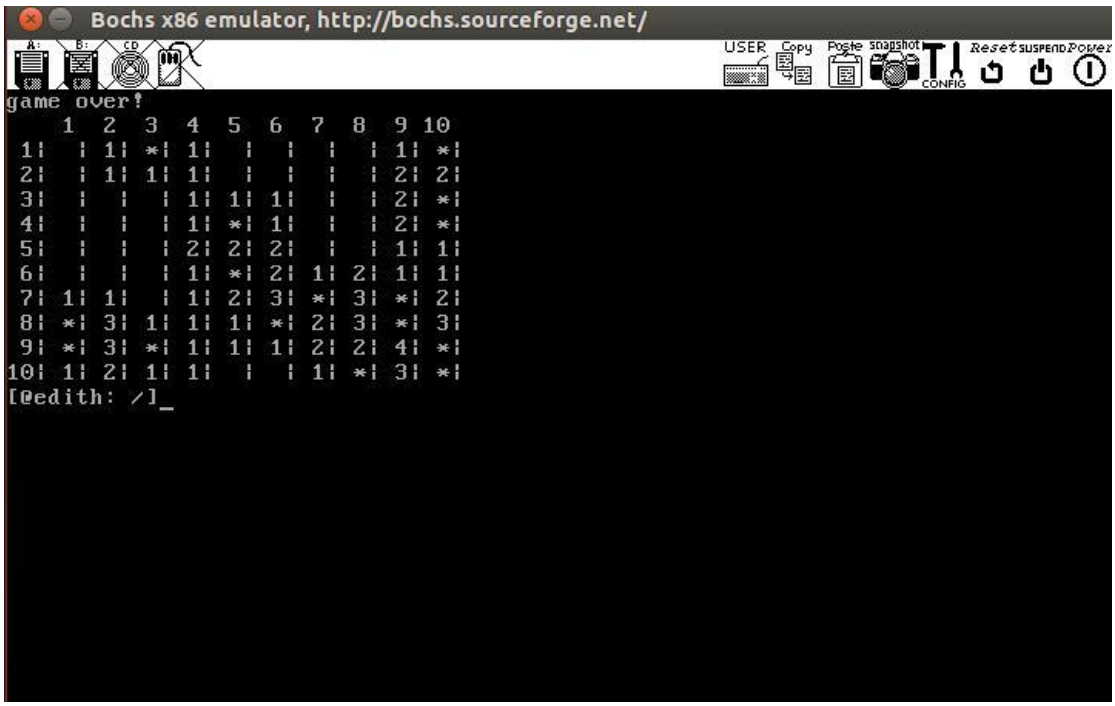
◆实现方法：调用五子棋函数 CHESS()方法，其中调用多个五子棋相关函数（如 USER1INPUT(),USER2INPUT(), WIN()等），实现五子棋游戏的相应操作。

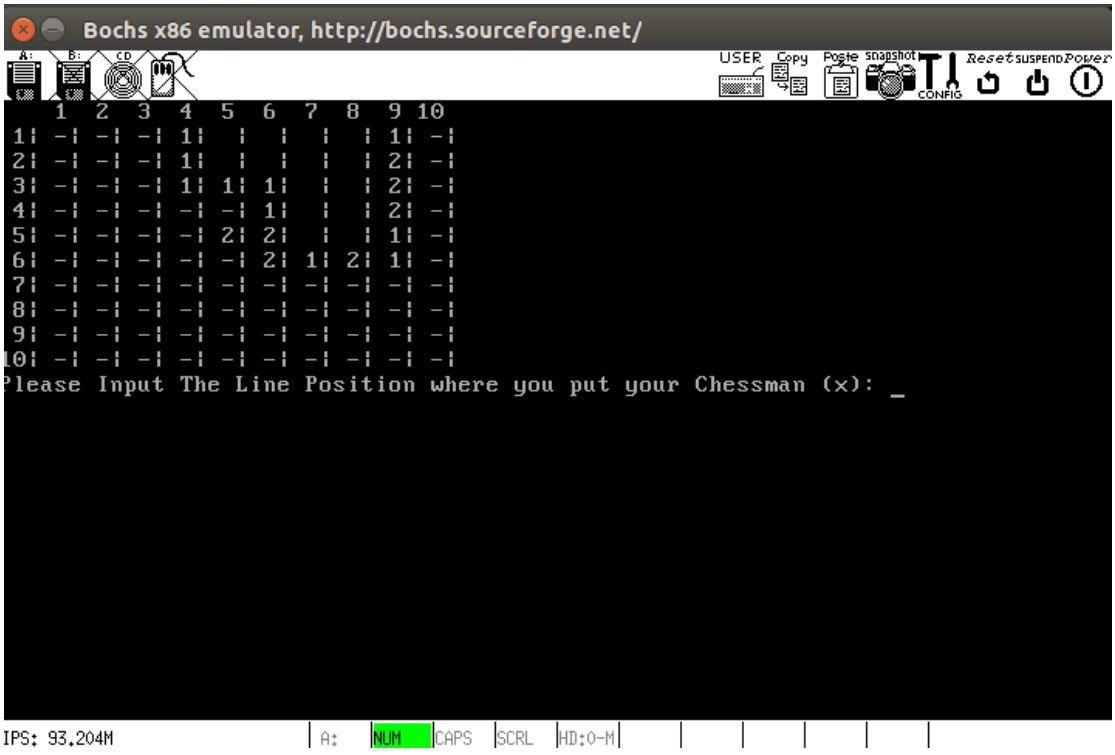


2.2.14 扫雷游戏

◆描述：用户输入“MINE”指令,进入扫雷游戏界面；首先系统打印扫雷地图，然后用户通过输入行、列坐标，直至踩到雷区或完成全部扫雷。

◆实现方法：调用扫雷函数 MINE()方法，其中调用多个扫雷相关函数（如MINE_COUNT(),OPEN_MIND()等），实现扫雷游戏的相应操作。





3.核心代码

3.1 创建文件夹

```
osPoint->FAT1[iFAT]=osPoint->FAT2[iFAT] = 2;   //2表示分配给下级目录文件
//填写该分派新的盘块的参数
strcpy(dir->fcb[temp].fname,sonfname);
dir->fcb[temp].type=DIRECTORY;
dir->fcb[temp].fatherBlockNum=current;
dir->fcb[temp].currentBlockNum=iFAT;
//初始化子目录文件盘块
dir=(struct dirFile*)(osPoint->data [iFAT-3]);   //定位到子目录盘块号
dir->init (current,iFAT,sonfname);//iFAT是要分配的块号，这里的current用来指要分配的块的父块号
printf("The folder have been created successfully.\n\n");
```

3.2 创建文件

```
//查找FAT表寻找空白区，用来分配磁盘块号j
for(i = 3;i<BlockCount;i++)
{
    if(osPoint->FAT1[i]==0)
        break;
}
if(i==BlockCount)
{
    printf("The disk is full!\n");
    return 0;
}
iFAT=i;

/*------进入分配阶段----------*/
//分配磁盘块
osPoint->FAT1[iFAT] = osPoint->FAT2[iFAT] = 1;
/*-----------接下来进行分配----------*/

//填写该分派新的盘块的参数
strcpy(dir->fcb[emptyNum].fname,name);
dir->fcb[emptyNum].type=GENERAL;
dir->fcb[emptyNum].fatherBlockNum=current;
dir->fcb[emptyNum].currentBlockNum=iFAT;
dir->fcb[emptyNum].size =0;
char* p = osPoint->data[iFAT -3];
memset(p,4,BlockSize);
printf("The file have been created successfully.\n");
```

3.3 删除文件

```
/*开始删除文件操作*/
j = dir->fcb [temp].currentBlockNum ;     //查找盘块号j
osPoint->FAT1[j]=osPoint->FAT2[j]=0;      //fat1,fat2表标记为空白
char *p=osPoint->data[j - 3];
memset(p,0,BlockSize); //清除原文本文件的内容
dir->fcb[temp].initialize();    //type=0;      //标记该目录项为空文件
printf("File has been deleted.\n");
return 1;
```

3.4 打印文件内容

```
//计算文件物理地址
fileStartNum = openlist->f[active].currentBlockNum - 3 ;
startPoint = osPoint->data[fileStartNum];
endPoint = osPoint->data[fileStartNum + 1];
//end_dir=(struct dirFile *)[BlockSize-1];

//q=(char *)end_dir;

while((*startPoint)!=4&& (startPoint < endPoint))
{
    putchar(*startPoint++);
}
```

3.5 进入多级目录

```
struct dirFile *dir;      //当前目录的指针
if(current==2)
    dir=&(osPoint->root);
else
    dir=(struct dirFile *)(osPoint->data [current-3]);
/*回到父目录*/
if(strcmp(sonfname,"..")==0)
{
    if(current==2)
    {
        printf("You are already in the root directory!\n");
        return 0;
    }
    current = dir->fcb[0].fatherBlockNum ;
    //currentPath = currentPath.substr(0,currentPath.size() - strlen(dir->fcb[0].fname )-1);
    return 1;
}
/*进入子目录*/
int i,temp;
//确保当前目录下有该目录,并且记录下它的FCB下标
for(i = 1; i < BlockFcbCount; i++)
{     //查找该文件
    if(dir->fcb[i].type==DIRECTORY && strcmp(dir->fcb[i].fname,sonfname)==0 )
    {
        temp=i;
        break;
    }
}
```

3.6 写文件

```
//计算文件物理地址
    int active=i;
    int fileStartNum = openlist->f[active].currentBlockNum - 3 ;
    startPoint = osPoint->data[fileStartNum];
    endPoint = osPoint->data[fileStartNum + 1];



    char input;
    while(((input=getchar())!=4))
    {
        if(startPoint < endPoint-1)
        {
            *startPoint++ = input;
        }
        else
        {

            *startPoint++ = 4;
            break;
        }
    }
```

3.7 五子棋输入落子

```
84  //用户1通过此函数来输入落子的位置，
85  //比如，用户输入3 1，则表示用户在第3行第1列落子。
86  void UserInput1(int fd_stdin,int fd_stdout)
87  {
88      printf("Player1:\n");
89      int n;
90      int pos = -1,x,y;
91      char szCmd[80]={0};
92  L1: printf("Please Input The Line Position where you put your Chessman (x): ");
93      n = read(fd_stdin,szCmd,80);
94      szCmd[1] = 0;
95      atoi(szCmd,&x);
96      printf("Please Input The Column Position where you put your Chessman (y): ");
97      n = read(fd_stdin,szCmd,80);
98      szCmd[1] = 0;
99      atoi(szCmd,&y);
100     if(x>0&&x<16&&y>0&&y<16&&QP[x-1][y-1]==0)
101         QP[x-1][y-1]=1;
102     else
103     {
104         printf("Input Error!\n");
105         goto L1;
106     }
107
108 }
109
```

3.8 五子棋判断输赢

```
156   //判断输赢
157   int Win(int QP[10][10], int row, int col)
158   {
159       int i = 0;
160       int j = 0;
161
162       // 横线上五子连成一线，赢家产生
163       for (i = 0; i < 10; i++)
164       {
165           for (j = 0; j < 10 - 4; j++)
166           {
167               if (QP[i][j] == QP[i][j + 1]
168                   && QP[i][j + 1] == QP[i][j + 2]
169                   && QP[i][j + 2] == QP[i][j + 3]
170                   && QP[i][j + 3] == QP[i][j + 4]
171                   && QP[i][j] != 0)
172               {
173                   return QP[i][j];
174               }
175           }
176       }
177
178       // 竖线上五子连成一线，赢家产生
179       for (j = 0; j < 10; j++)
```

3.9 计算周围雷数

```
//计算周围雷数的函数，返回数目
static int mine_count(char mine[][COL + 2], int x, int y)
{
    int count = 0;
    mine[x - 1][y - 1] == '*' ? count++ : count;
    mine[x - 1][y    ] == '*' ? count++ : count;
    mine[x - 1][y + 1] == '*' ? count++ : count;
    mine[x    ][y - 1] == '*' ? count++ : count;
    mine[x    ][y + 1] == '*' ? count++ : count;
    mine[x + 1][y - 1] == '*' ? count++ : count;
    mine[x + 1][y    ] == '*' ? count++ : count;
    mine[x + 1][y + 1] == '*' ? count++ : count;
    return count;
}
```

3.10 扫雷自动展开

```
//无雷区域自动展开的函数，核心函数，这里用的是递归，效率比较低下，后期有待改进
//该函数的出口条件是，扫描当前的格子已经不是‘0’无雷了，那么就停止递归，也就是碰到边界，或者碰到数字，或者碰到‘*’雷停止递归
//否则，以自己为中心向他的8个方向开始递归
static void open_mine(char mine[][COL + 2], char show[][COL + 2], int x, int y)
{
    while(mine[x][y] == '0')  //这个是循环条件也是出口条件，满足条件时继续递归，不满足条件时结束递归
    {
        mine[x][y] = '/';  //只要进来说明这个坐标的格子不是雷，立马把对应坐标的mine数组改为‘/’，表明这个地方扫描过了，防止，重新排查
        show[x][y] = ' ';  //没有雷的话，就向用户展示‘ ’，  而不是‘0’
        open_mine(mine, show, x, y - 1);
        open_mine(mine, show, x, y + 1);
        open_mine(mine, show, x - 1, y);
        open_mine(mine, show, x + 1, y);
        open_mine(mine, show, x - 1, y - 1);
        open_mine(mine, show, x - 1, y + 1);
        open_mine(mine, show, x + 1, y - 1);
        open_mine(mine, show, x + 1, y + 1);
    }
    if(mine[x][y] != '*' && mine[x][y] != '/')   //如果是数字，那么就显示数字，这个数字表示以这个数字为中心的格子周围有几个雷
    {
        show[x][y] = mine[x][y];
    }
}
```

4.成员分工

姓名	学号	分工	占分
朴雪	1752124	项目整合、命令行功能实现	25%
郭玮佳	1751292	用户级应用实现	25%
孙文颖	1750802	多级文件系统实现	25%
朱丽雯	1751169	动画设计、文档编写	25%