# 模拟操作系统实验报告

## 一、新设计的命令

### 1.Help

> 通过命令菜单显示所有的命令，并对其进行编号，通过读入控制台输入的序号，用switch语句进行判别

```
// 测试用例
help
2
```

> 运行结果

```
C:\Users\caoji\Desktop\模拟操作系统\myCode\OSDemo.exe

现在你可以输入各种操作命令.
Help —— 简易帮助信息.
exit —— 退出本程序.

C:/>help

* * * * * * * * * * 本系统主要的文件操作命令简述如下 * * * * * * * * * *

请选择需要帮助的命令编号:
[1] create    [2] open      [3] write    [4] read
[5] close     [6] del       [7] dir      [8] cd
[9] md        [10] rd        [11] ren     [12] attrib
[13] copy      [14] type      [15] rewind  [16] fseek
[17] block     [18] closeall  [19] uof     [20] undel
[21] exit      [22] prompt    [23] fat      [24] check
[0] 返回上级菜单

2
open <文件名> ——打开文件，操作类型可为r、h或(与)s。
示例：open testfile.txt ——打开名为testfile.txt的文件进行操作。
0
```

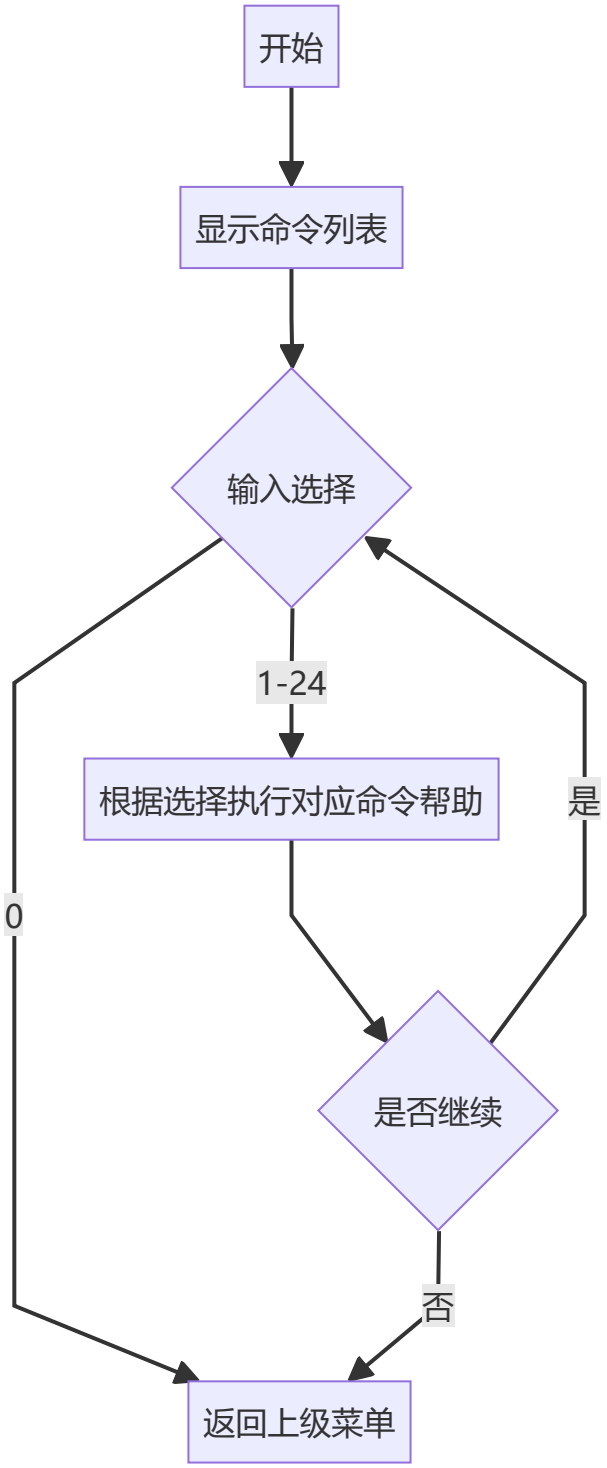
```
// 命令完整代码
void HelpComd()
{
    cout << "\n* * * * * * * * * 本系统主要的文件操作命令简述如下 * * * * * * * * *
*\n\n";
    cout << "请选择需要帮助的命令编号：\n";
    cout << "[1] create    [2] open      [3] write    [4] read\n";
    cout << "[5] close     [6] del       [7] dir      [8] cd\n";
```

```
    cout << "[9] md        [10] rd        [11] ren     [12] attrib\n";
    cout << "[13] copy      [14] type      [15] rewind  [16] fseek\n";
    cout << "[17] block     [18] closeall  [19] uof     [20] undel\n";
    cout << "[21] exit      [22] prompt    [23] fat      [24] check\n";
    cout << "[0] 返回上级菜单\n\n";

    int choice = 100;
    while(choice != 0)
    {
        cin >> choice;
        if(choice == 0)return;
        switch (choice)
        {
            case 1:
                cout << "create <文件名>[ <文件属性>] --创建新文件,文件属性是r、h或s。
\n";
                cout << "示例：create testfile.txt --创建一个名为testfile.txt的新文
件。\n";
                break;
            case 2:
                cout << "open <文件名> --打开文件，操作类型可为r、h或(与)s。\n";
                cout << "示例：open testfile.txt --打开名为testfile.txt的文件进行操
作。\n";
                break;
            case 3:
                cout << "write <文件名> [<位置/app>[ insert]] --在指定位置写文件(有插
入功能)。\n";
                cout << "示例：write testfile.txt append --在testfile.txt文件末尾追
加内容。\n";
                break;
            case 4:
                cout << "read <文件名> [<位置m> [<字节数n>]] --读文件，从第m字节处读n个
字节。\n";
                cout << "示例：read testfile.txt 10 20 --从testfile.txt的第10字节开
始读取20个字节。\n";
                break;
            case 5:
                cout << "close <文件名> --关闭文件。\n";
                cout << "示例：close testfile.txt --关闭已打开的名为testfile.txt的文
件。\n";
                break;
            case 6:
                cout << "del <文件名> --撤消(删除)文件。\n";
                cout << "示例：del testfile.txt --删除名为testfile.txt的文件。\n";
                break;
            case 7:
                cout << "dir [<路径名>] [|<属性>] --显示当前目录。\n";
                cout << "示例：dir / --显示根目录中的所有文件和子目录。\n";
                break;
            case 8:
                cout << "cd [<路径名>] --改变当前目录。\n";
                cout << "示例：cd /usr --切换到/usr目录。\n";
                break;
            case 9:
                cout << "md <路径名> [<属性>] --创建指定目录。\n";
                cout << "示例：md /newdir --在根目录下创建名为newdir的新目录。\n";
```

```
                break;
            case 10:
                cout << "rd [<路径名>] --删除指定目录。\n";
                cout << "示例：rd /newdir --删除名为newdir的目录。\n";
                break;
            case 11:
                cout << "ren <旧文件名> <新文件名> --文件更名。\n";
                cout << "示例：ren oldfile.txt newfile.txt --将oldfile.txt重命名为
newfile.txt。\n";
                break;
            case 12:
                cout << "attrib <文件名> [±<属性>] --修改文件属性(r、h、s)。\n";
                cout << "示例：attrib testfile.txt +r --将testfile.txt设置为只读文
件。\n";
                break;
            case 13:
                cout << "copy <源文件名> [<目标文件名>] --复制文件。\n";
                cout << "示例：copy testfile.txt newfile.txt --创建testfile.txt的副
本newfile.txt。\n";
                break;
            case 14:
                cout << "type <文件名> --显示文件内容。\n";
                cout << "示例：type testfile.txt --显示testfile.txt的内容。\n";
                break;
            case 15:
                cout << "rewind <文件名> --将读、写指针移到文件第一个字符处。\n";
                cout << "示例：rewind testfile.txt --将testfile.txt的读写指针重置到文
件开头。\n";
                break;
            case 16:
                cout << "fseek <文件名> <位置> --将读、写指针都移到指定位置。\n";
                cout << "示例：fseek testfile.txt 30 --将testfile.txt的读写指针移动到
第30个字节。\n";
                break;
            case 17:
                cout << "block <文件名> --显示文件占用的盘块号。\n";
                cout << "示例：block testfile.txt --显示testfile.txt占用的盘块号。
\n";
                break;
            case 18:
                cout << "closeall --关闭当前打开的所有文件。\n";
                cout << "示例：closeall --关闭所有已打开的文件。\n";
                break;
            case 19:
                cout << "uof --显示UOF(用户打开文件表)。\n";
                cout << "示例：uof --显示当前用户打开的所有文件的列表。\n";
                break;
            case 20:
                cout << "undel [<路径名>] --恢复指定目录中被删除的文件。\n";
                cout << "示例：undel /usr --恢复/usr目录中被删除的文件。\n";
                break;
            case 21:
                cout << "exit --退出本程序。\n";
                cout << "示例：exit --退出文件管理系统。\n";
                break;
            case 22:
```

```
                cout << "prompt --提示符是否显示当前目录(切换)。\n";
                cout << "示例：prompt --切换提示符是否显示当前目录的状态。\n";
                break;
            case 23:
                cout << "fat --显示FAT表中空闲盘块数(0的个数)。\n";
                cout << "示例：fat --显示当前磁盘上的空闲块数。\n";
                break;
            case 24:
                cout << "check --核对后显示FAT表中空闲盘块数。\n";
                cout << "示例：check --核对并显示空闲盘块数。\n";
                break;
            case 0:
                // 返回上级菜单的逻辑可以在这里实现
                break;
            default:
                cout << "无效的选择，请重新输入。\n";
                break;
        }
    }
}
```

> 流程图




### 2.Fc

> 该命令的实现先要通过FindFCB()函数找到两个文件各自对应的FCB结构体指针，
>
> 然后通过file_to_buffer()函数将文件内容读取到buffer数组中，
>
> 通过“双指针”对buffer数组里的每一个元素挨个对比，如果出现不一样的内容，就break掉循环，返回不一样的位置

```
// 测试用例
create t1
write t1
asd
create t2
write t2
ase
```

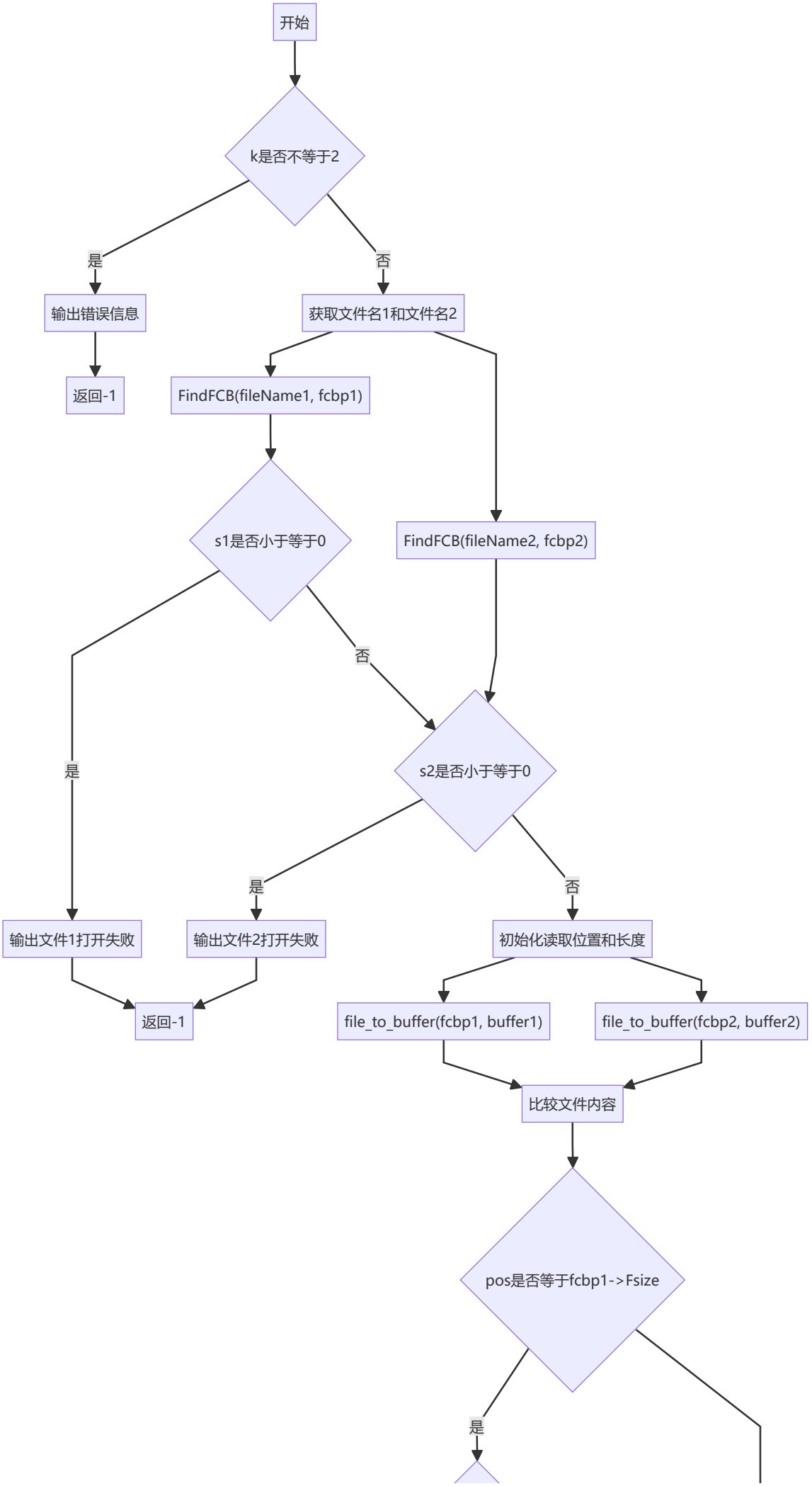
运行结果

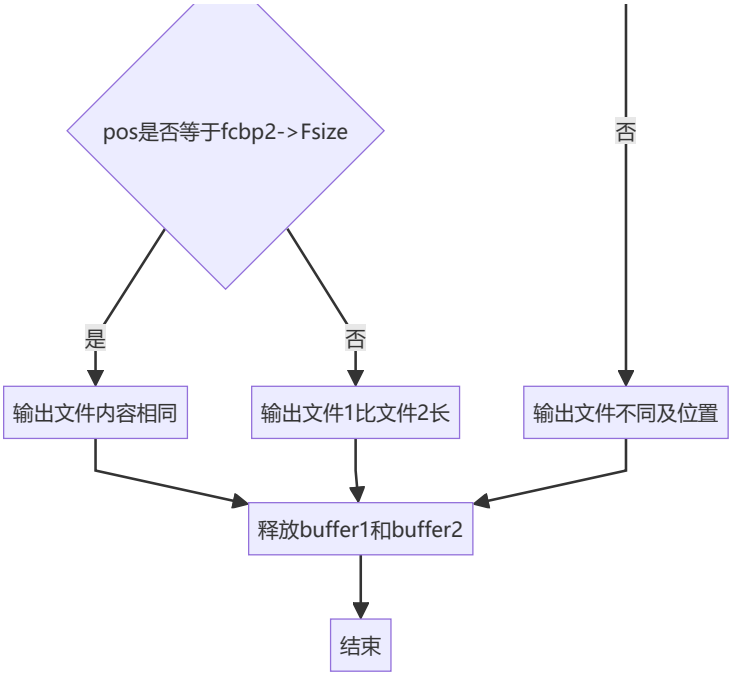
```
C:\Users\caoji\Desktop\模拟操作系统\myCode\OSDemo.exe

现在你可以输入各种操作命令.
Help —— 简易帮助信息.
exit —— 退出本程序.

C:/>create t1

文件/t1建立成功

C:/>write t1

请输入写入文件的内容(最多允许输入2560个字节):
asd

C:/>create t2

文件/t2建立成功

C:/>write t2

请输入写入文件的内容(最多允许输入2560个字节):
ase

C:/>fc t1 t2

文件不同，在第 3 个字符：文件1是 d，文件2是 e

C:/>
```

```
// 命令完整代码
int FcComd(int k) {
    if (k != 2) {
        cout << "\n命令参数错误：需要两个文件名作为参数。\n";
        return -1;
    }

    char *fileName1 = comd[1];
    char *fileName2 = comd[2];

    // 获取两个文件的FCB指针
    FCB *fcbp1, *fcbp2;
    int s1 = FindFCB(fileName1, curpath.fblock, '\040', fcbp1); // 假设文件在当前目
录
    int s2 = FindFCB(fileName2, curpath.fblock, '\040', fcbp2); // 假设文件在当前目
录

    if (s1 <= 0 || s2 <= 0) {
        if (s1 <= 0) cout << "\n文件 " << fileName1 << " 打开失败。\n";
        if (s2 <= 0) cout << "\n文件 " << fileName2 << " 打开失败。\n";
        return -1;
    }

    // 初始化文件内容读取位置
    int pos1 = 0, pos2 = 0;
    int len1 = fcbp1->Fsize;
    int len2 = fcbp2->Fsize;
    int isDifferent = 0;
    int diffPos = 0;

    // 读取文件内容
    char* buffer1 = new char[fcbp1->Fsize + 1];
    char* buffer2 = new char[fcbp2->Fsize + 1];
    file_to_buffer(fcbp1, buffer1);
    file_to_buffer(fcbp2, buffer2);

// 比较文件内容
    int pos = 0;
    while (pos < fcbp1->Fsize && pos < fcbp2->Fsize) {
        if (buffer1[pos] != buffer2[pos]) {
            break;
        }
        pos++;
    }

    if (pos == fcbp1->Fsize && pos == fcbp2->Fsize) {
        cout << "\n文件内容相同。\n";
    }
    else {
        if (pos < fcbp1->Fsize && pos < fcbp2->Fsize) {
            cout << "\n文件不同，在第 " << pos + 1 << " 个字符：";
            cout << "文件1是 " << buffer1[pos] << "，文件2是 " << buffer2[pos] <<
endl;
        }
        else if (pos < fcbp1->Fsize) {
            cout << "\n文件1比文件2长。\n";
```

```
        }
        else {
            cout << "\n文件2比文件1长。\n";
        }
    }

    delete[] buffer1;
    delete[] buffer2;
    return 1;


}
```

> 流程图


开始
k是否不等于2
是
否
输出错误信息
获取文件名1和文件名2
返回-1
FindFCB(fileName1, fcbp1)
s1是否小于等于0
FindFCB(fileName2, fcbp2)
是
否
s2是否小于等于0
是
否
输出文件1打开失败
输出文件2打开失败
初始化读取位置和长度
返回-1
file_to_buffer(fcbp1, buffer1)
file_to_buffer(fcbp2, buffer2)
比较文件内容
pos是否等于fcbp1->Fsize
是




### 3.Replace

> 这个命令要实现同名文件的替换

### 4.Batch

### 5.Lose Type