

中山大学数据科学与计算机学院

# 操作系统实验

实验题目： 文件系统

专业班级： 2014 级计科一班

学生姓名： 郭达雅

学 号： 14348024

指导老师： 凌应标

## 一、 实验内容

1.实验题目.....	3
2.实验目的.....	3
3.实验要求.....	3

## 二、 实验方案

1.相关知识原理.....	4
2.方案设计.....	4
3.函数说明.....	6

## 三、 实验过程

1.实验步骤.....	8
2.使用说明.....	8
3.结果测试.....	9

## 四、 总结与体会

## 1. 实验题目：文件管理系统

在实验五或更后的原型基础上，进化你的原型操作系统，原型保留原有特征的基础上，设计满足下列要求的新原型操作系统：

- (1) 在内核中实现文件系统管理
- (2) 在用户中实现调用，包括 `fopen()`, `fclose()`, `fputc()`, `fgetc()`
- (3) 编写 c 语言程序，测试所有功能

## 2. 实验目的

- (1) 学习如何创建进程
- (2) 学习如何组织 PCB
- (3) 学习如何创建子进程并撤销
- (4) 学习如何进程控制与通信

## 3. 实验要求

- (1) 一个项目可提交多个改进的版本，实现新功能和个性化特征都 有利于提高相应项目的成绩。
- (2) 实验项目提交内容用 `winrar` 工具整体压缩打包，统一格式命名为： <学号>+<姓名>+<实验项目号>+<版本号>.rar
- (3) 实验项目，迟交影响成绩评价
- (4) 一系列基础实验项目必须连续完成，不可抄袭

## 1. 相关知识原理

### (1) BIOS 调用读写功能

对于软盘读写可以调用 BIOS 中断，在 13h 中断，在第二个实验的实验报告中已经告诉了如何读软盘，只需要把 ah 也就是功能号改成 03h 即为写操作，其余的参数没改变

### (2) Fopen 函数的功能描述

fopen 函数有几种打开模式，在这里只有三种，分别是读打开，写打开，追加打开。其中如果文件不存在读打开是会发生错误的，而写打开和追加打开会创建一个新的文件。如果文件存在，则写打开会清空里面所有的内容。

### (3) Fclose 函数的功能描述

fclose 函数为了关闭已打开的文件，目的有两个：①是为了实现共享，如果一个文件在读，那么必须关闭才能使另一进程进行写，反之亦然。②为了保存文件，如果一个文件在写或者追加，只有关闭文件才能保存信息。

## 2. 方案设计

### (1) 设计思路

对于 fgetc 和 fputc 函数来说，是非常简单的，只需要根据用户提供的文件指针对文件进行读写就好了，老师可看我的代码，这两个函数只用了一行的代码。

而对于 fopen 函数来说，要分为三种模式，但效果其实是一样都是返回文件指针，所以只需要将对应的文件拷贝到内存中，而后返回一个文件指针即可。但要注意几点：对于读操作，要返回文件首地址和文件大小；对于写操作，由于会清空文件内容，所以只需要返回一个空文件的首地址；对于追加操作，需要找到文件返回文件的末尾地址和文件大小。（注：我将文件的保存全都放在 fclose 函数来操作）

对于 fclose 函数来说，主要分为两种，一种是读文件关闭，另一种是写文件或追加文件关闭。对于读文件关闭，只需要把读者数量减 1 即可。而对于写文件或追加文件关闭，需要更改根目录和 FAT 表，这是本实验中最大的难度。

## (2) Fopen 函数的实现

分为以下几个步骤：

- ①判断文件打开类型：分别为 r（读打开），w（写打开），a(追加打开)
- ②若打开类型为 r，根据根目录找到文件，并将文件拷贝一份进内存中，并返回该文件在内存中的首地址和文件大小。**同时需要把读者数量加 1，为了互斥使用。如果该文件正在写，那么需要返回错误**
- ③若打开类型为 w，在内存中重新创建一个空文件，然后返回内存的首地址，之所以不用查找根目录看文件是否存在，是因为我把文件的存储放在 fclose 函数中了。**如果该文件正在打开，则返回错误。**
- ④若打开类型为 a，根据根目录找到文件，并将文件拷贝一份进内存中，并返回该文件在内存中的尾地址和文件大小。**如果该文件正在打开，则返回错误。**

## (3) Fclose 函数的实现

- ①判断该关闭文件的打开类型：分别为 r,w/a
- ②若打开类型为 r，只需要把读者数量减 1 即可
- ③若打开类型为 w/a，则分为以下几个步骤：
  - (1) 在根目录中查找，如果找不到该文件的目录项，则自己要在空闲的地方中创建一个
  - (2) 如果自己创建了目录项，则需要根据 FAT 表找到一个空闲簇号作为空闲簇号，如果根目录中已经存在该文件信息，则可直接获得首簇号
  - (3) **修改根目录，在文件对应的目录项中修改文件名、文件大小、首簇号、和修改时间。（由于利用 CMOS RAM 提取时间太麻烦了，因为提取出来的是 BCD 码，所以就没修改文件时间了）**
  - (4) **判断文件大小需要多少个扇区，然后根据首簇号，一个 for 循环把文件一个一个添加到空闲簇号，如果当前簇号没有指向下个簇，则需要找到空闲簇号。**
  - (5) 最后调用 13h 中断将根目录和两个 FAT 表写回软盘中。**注意这些操作都是要原子操作**

## 3. 函数使用说明

首先文件的表示是一个文件结构体类型 FILE

### (1) struct FILE\* fopen(char\*, char) 打开文件

第一的参数文件名，第二参数是打开模式，分别有 r（读操作）、w（写操作）、a（追加操作）。如果打开成功则返回一个文件指针，如果打开失败则返回一个 NULL

代码示例：打开写操作模式的文件 guo.txt

```
struct FILE *infile, *outfile;
if((infile=fopen("guo.txt", 'w'))==NULL)
{
    printf("The file can not be open.\n");
    return 0;
}
```

### (2) char fgetc(struct FILE\*) 读取字符

参数是要读字符的文件指针，如果成功则返回相应的字符，否则返回一个 EOF。失败的原因可能是读到末尾，或者该文件的打开模式不是读操作。

代码示例：读取文件 guo.txt

```
if((infile=fopen("guo.txt", 'r'))==NULL)
{
    printf("The file can not be open.\n");
    return 0;
}
ch=fgetc(infile);
while(ch!=EOF)
{
    putchar(ch);
    ch=fgetc(infile);
}
```

### (3) int fputc(struct FILE\*, char) 写文件

第一参数是文件指针，表示要写的文件，第二个参数是表示要写的字符。如果成功返回 0，失败则返回 EOF（注：通过 fputc 写了个 fputs，写一个字符串到文件中）

代码示例：写一个“hello!”到 guo.txt 中

# 实验方案

```
struct FILE *infile,*outfile;
if((infile=fopen("guo.txt",'w'))==NULL)
{
    printf("The file can not be open.\n");
    return 0;
}
fputs(infile,"hello! ");
fclose(infile);
```

## (4) int fclose(struct FILE\*) 关闭文件

参数需要关闭的文件指针，如果成功返回 0，否则返回 EOF。只有关闭文件才能保存文件

代码示例：用 w 模式写一个“hello, ”,用 a 模式追加一个“world!”到 guo.txt 中，然后用 r 模式读取文件内容。

```
struct FILE *infile,*outfile;
if((infile=fopen("guo.txt",'w'))==NULL)
{
    printf("The file can not be open.\n");
    return 0;
}
fputs(infile,"hello! ");
fclose(infile);
```

```
if((infile=fopen("guo.txt",'a'))==NULL)
{
    printf("The file can not be open.\n");
    return 0;
}
fputs(infile,"world!");
fclose(infile);
```

```
if((infile=fopen("guo.txt",'r'))==NULL)
{
    printf("The file can not be open.\n");
    return 0;
}
ch=fgetc(infile);
while(ch!=EOF)
{
    putchar(ch);
    ch=fgetc(infile);
}
fclose(infile);
```

## 1. 实验步骤

用 VM 运行文件中的镜像，通过命令行调用用户程序

## 2. 使用说明

文件中包含四个文件夹，在一些文件夹中有 txt 文档说明。

这里有四个测试程序：

- ①r.com: 读取并显示 guo.txt
- ②w.com: 向 guo.txt 写 “I am guo da ya!”
- ③a.com: 向 guo.txt 追加一个 “I am handsome!”
- ④c.com: 向 new.txt 写 100 个 “This is my 0s”

一：相关命令：

查看 BPB 信息: bpb

查看镜像的文件信息: dir

查看相关文件的存放信息: fat

读 guo.txt 文件:r.com

写 guo.txt 文件:w.com

追加写 guo.txt 文件:a.com

创建一个新的文件，并且写大量内容:c.com

清屏: cls

调用程序:程序名 1 程序名 2 程序名 3 ... 程序名 n

(如运行 one.com, two.com, three.com, 则输入 one.com two.com three.com。

每个程序之间用空格隔开)

退出所有程序: quit

(指令不区分大小写)

二：已实现的键盘功能：

左箭头

右箭头

delete 键

回车键



## 3. 结果测试

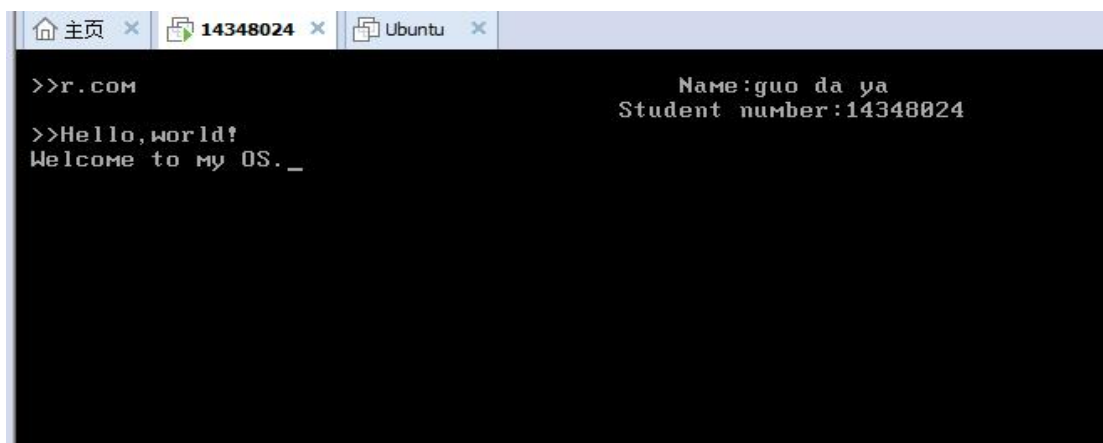
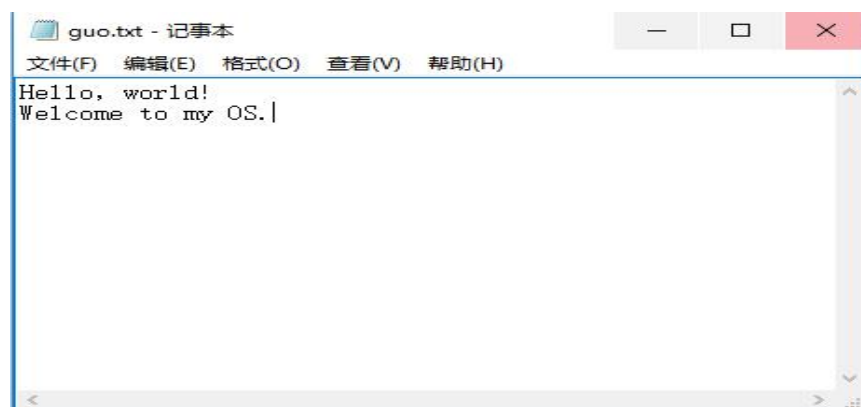
①读取并显示 guo.txt 文件：输入 r.com

代码如下：这是一个简单的读文件，读取 guo.txt 文件并且显示出来

```
int main()
{
    int is,a,i;
    char ch;
    struct FILE *infile,*outfile;
    if((infile=fopen("guo.txt",'r'))==NULL)
    {
        printf("The file can not be open.\n");
        return 0;
    }
    sleep();
    ch=fgetc(infile);
    while(ch!=EOF)
    {
        putchar(ch);
        ch=fgetc(infile);
    }
    fclose(infile);
}
```

运行结果：

第一幅图是用记事本打开的，第二幅图是程序运行的结果，可看出 r.com 能正确的读取并显示文件



## 实验过程

②测试写文件和通信机制：先输入 w.com, 然后再输入 r.com

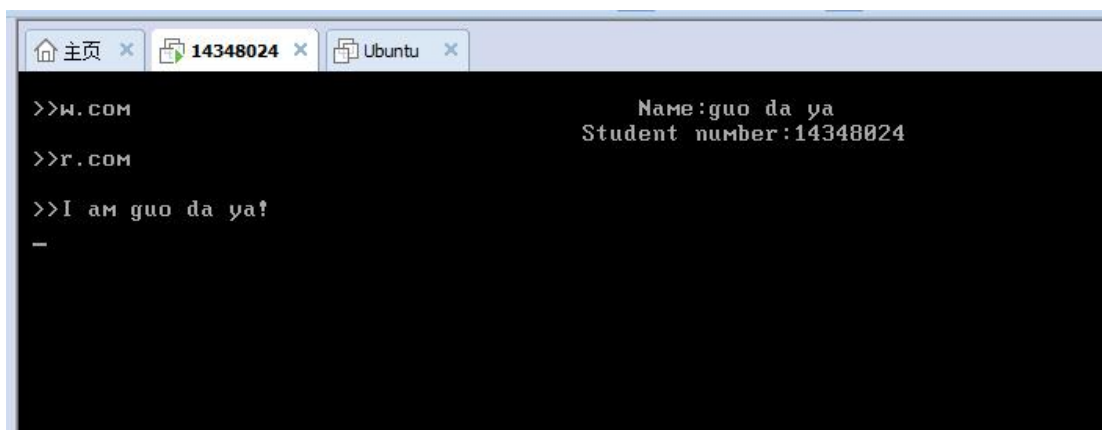
该测试时 w 进程修改 guo.txt 的内容，然后用 r 进程读取修改后的内容。

w.com 的代码如下：向 guo.txt 文件写一个字符串 “I am guo da ya!”

```
int main()
{
    int is,a,i;
    char ch;
    struct FILE *infile,*outfile;
    if((infile=fopen("guo.txt",'w'))==NULL)
    {
        printf("The file can not be open.\n");
        return 0;
    }
    fputs(infile,"I am guo da ya!\n");
    fclose(infile);
}
```

运行结果：

可以看到，写模式打开的文件把原文件的内容删掉，然后写入的新的内容进行保存，而且在这之中也可以看到进程之间的通信，w.com 进程向 r.com 发送一个消息 “I am guo da ya”，然后进程可以读取到相应的消息。



## 实验过程

③测试追加写文件和多个进程的通信:先输入 w. com, 再输入 a. com, 再输入 r. com

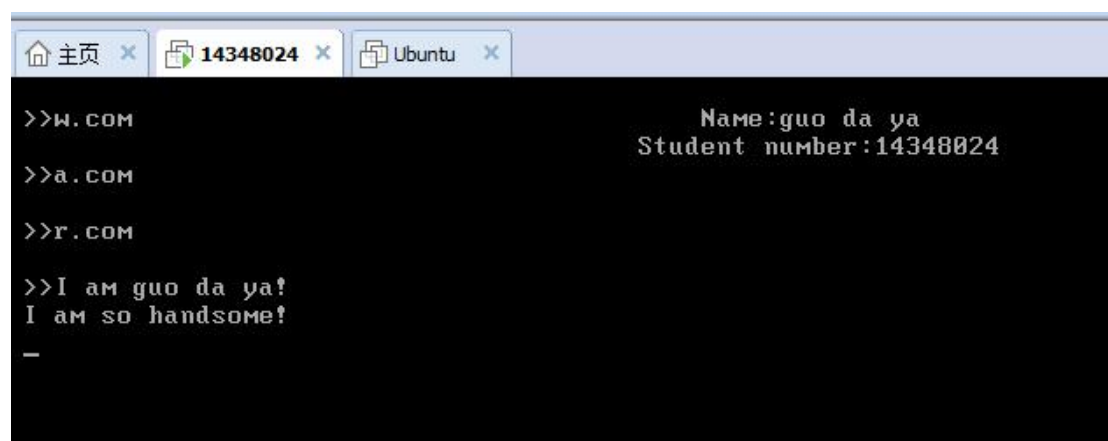
w 进程修改 guo. txt 的内容, 然后 a 进程对 guo. txt 的内容进行追加写入, 最后 r 进程读取 guo. txt 的内容

a. com 进程的代码如下:

```
int main()
{
    int is,a,i;
    char ch;
    struct FILE *infile,*outfile;
    if((infile=fopen("guo.txt",'a'))==NULL)
    {
        printf("The file can not be open.\n");
        return 0;
    }
    fputs(infile,"I am so handsome!\n");
    fclose(infile);
}
```

运行结果:

可以看到 w 进程向 guo. txt 写入了 “I am guo da ya!”, 然后 a 进程再向 guo. txt 中添加一行内容 “I am so handsome!”, 最后 r 进程读取了他们两个修改过后的消息, 可以看出符合结果.



```
>>w.COM                                     Name:guo da ya
                                              Student number:14348024
>>a.COM
>>r.COM
>>I am guo da ya!
I am so handsome!
-
```

# 实验过程

## ④创建新的文件：输入 c.com

这个程序打开一个不存在的文件 new.txt，由于是写模式，所以会创建一个新的文件，同时向该文件中写入大量数据。c.com 程序的代码如下：

```
int main()
{
    int is,a,i;
    char ch;
    struct FILE *infile,*outfile;
    if((infile=fopen("new.txt",'w'))==NULL)
    {
        printf("The file can not be open.\n");
        return 0;
    }

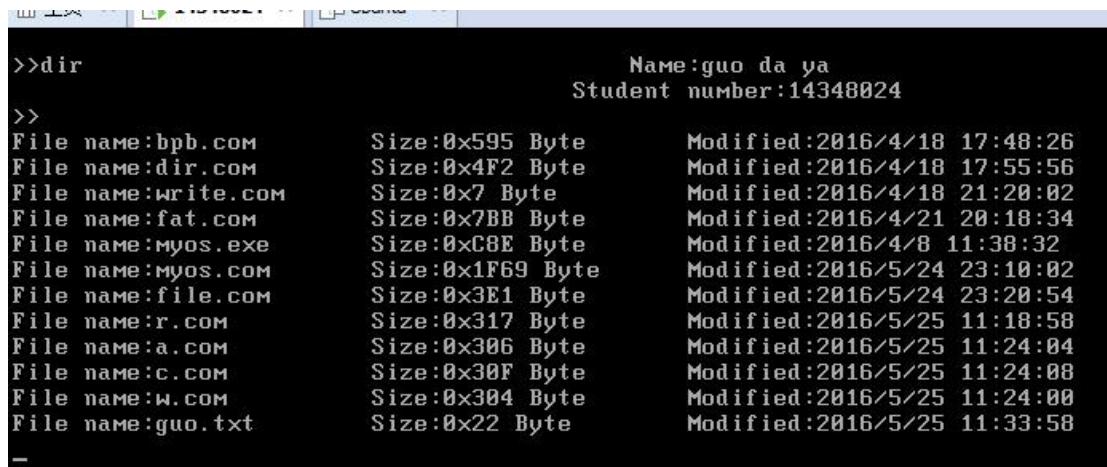
    for(i=0;i<100;++i)
    {
        fputs(infile,"This is my 0s!\n");
    }
    fclose(infile);
}
```

## 运行结果：

第一幅图是运行前的文件目录，可以看出并不存在 new.txt 文件。

第二幅图是运行后的文件目录，可以看到多了一个新的文件 new.txt，而且文件大小是 5DC=1500B，即每句话 15 个字节，一数 “This is my 0s!\n” 真的是 15 个字节，比较遗憾的是，我没有修改时间。

第三幅图是用记事本打开 new.txt 的内容，里面确实是 This is my 0s 重复 100 遍，但是为什么没有换行符呢？其实 txt 文件中换行符是占两个字节的，所以我输出的换行符并不符合 txt 文档的格式。但总的来说还是正确的。



```
>>dir
Name: guo da ya
Student number: 14348024
>>
File name: bpb.com      Size: 0x595 Byte      Modified: 2016/4/18 17:48:26
File name: dir.com      Size: 0x4F2 Byte      Modified: 2016/4/18 17:55:56
File name: write.com    Size: 0x7 Byte        Modified: 2016/4/18 21:20:02
File name: fat.com      Size: 0x7BB Byte      Modified: 2016/4/21 20:18:34
File name: myos.exe     Size: 0xC8E Byte      Modified: 2016/4/8 11:38:32
File name: myos.com     Size: 0x1F69 Byte     Modified: 2016/5/24 23:10:02
File name: file.com     Size: 0x3E1 Byte      Modified: 2016/5/24 23:20:54
File name: r.com        Size: 0x317 Byte      Modified: 2016/5/25 11:18:58
File name: a.com        Size: 0x306 Byte      Modified: 2016/5/25 11:24:04
File name: c.com        Size: 0x30F Byte      Modified: 2016/5/25 11:24:08
File name: w.com        Size: 0x304 Byte      Modified: 2016/5/25 11:24:00
File name: guo.txt      Size: 0x22 Byte       Modified: 2016/5/25 11:33:58
```

# 实验过程

```
>>c.com                                     Name:guo da ya
                                           Student number:14348024

>>dir

>>
File name:bpb.com           Size:0x595 Byte      Modified:2016/4/18 17:48:26
File name:dir.com           Size:0x4F2 Byte      Modified:2016/4/18 17:55:56
File name:write.com         Size:0x7 Byte        Modified:2016/4/18 21:20:02
File name:fat.com           Size:0x7BB Byte      Modified:2016/4/21 20:18:34
File name:myos.exe          Size:0xC8E Byte      Modified:2016/4/8 11:38:32
File name:myos.com          Size:0x1F69 Byte     Modified:2016/5/24 23:10:02
File name:file.com          Size:0x3E1 Byte      Modified:2016/5/24 23:20:54
File name:r.com             Size:0x317 Byte      Modified:2016/5/25 11:18:58
File name:a.com             Size:0x306 Byte      Modified:2016/5/25 11:24:04
File name:c.com             Size:0x30F Byte      Modified:2016/5/25 11:24:08
File name:w.com             Size:0x304 Byte      Modified:2016/5/25 11:24:00
File name:guo.txt           Size:0x22 Byte       Modified:2016/5/25 11:33:58
File name:new.txt           Size:0x5DC Byte      Modified:1980/0/0 00:00:00
```

NEW - 记事本

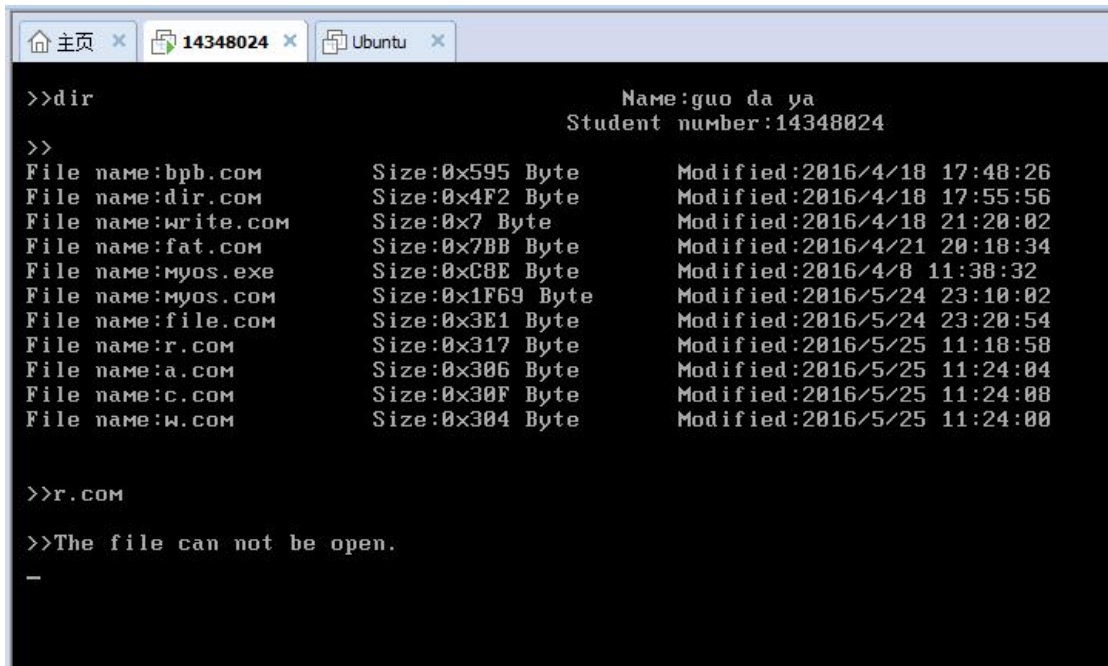
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

This is my Os!This is my Os!This is my Os!This is my Os!This i  
is my Os!This is my Os!This is my Os!This is my Os!This is my

# 实验过程

⑤打开不存在的文件：删除 guo.txt，然后输入 r.com

可以看到 guo.txt 不存在，然后 r.com 打开 guo.txt，会收到一个错误消息。



The screenshot shows a terminal window with three tabs: '主页', '14348024', and 'Ubuntu'. The terminal output is as follows:

```
>>dir
Name:guo da ya
Student number:14348024
>>
File name:bpb.com      Size:0x595 Byte      Modified:2016/4/18 17:48:26
File name:dir.com      Size:0x4F2 Byte      Modified:2016/4/18 17:55:56
File name:write.com    Size:0x7 Byte        Modified:2016/4/18 21:20:02
File name:fat.com      Size:0x7BB Byte      Modified:2016/4/21 20:18:34
File name:myos.exe     Size:0xC8E Byte      Modified:2016/4/8 11:38:32
File name:myos.com     Size:0x1F69 Byte     Modified:2016/5/24 23:10:02
File name:file.com     Size:0x3E1 Byte      Modified:2016/5/24 23:20:54
File name:r.com        Size:0x317 Byte      Modified:2016/5/25 11:18:58
File name:a.com        Size:0x306 Byte      Modified:2016/5/25 11:24:04
File name:c.com        Size:0x30F Byte      Modified:2016/5/25 11:24:08
File name:w.com        Size:0x304 Byte      Modified:2016/5/25 11:24:00

>>r.com

>>The file can not be open.
_
```

## 总结与体会：

好久没做操作系统实验，现在突然做，感觉有点怀念。之前做了终端的，但我发现显示经常换入换出非常慢，所以最后还是不做了。现在做了文件管理系统，不负老师的教诲，**用了 197 行写完了代码**，其实如果不算换行的重复代码的话，其实也就 100 行多一点，总的来说，这是一个简单的文件系统，因为没有写子目录什么的。

其实在这里面是实现了文件读写的共享和互斥的，只不过没有展示出来，本来是想同时运行多个 `r.com` 和 `w.com` 程序的，但是不知道为什么控制台突然只能一条指令运行一个程序，这使得我刚输入完下一条指令的时候，上一个进程就已经结束，所以没法看到互斥现象。当然我可以在一个程序中打开读文件的同时打开写文件，也能看到不可访问，由于时间关系，也没有写个程序去测试。

**接下里应该不会再写操作系统实验了吧，应该会开始写汇编游戏了。**