简单文件系统

实验报告

金科

08111101

1120112762

实验内容

通过对具体的文件存储空间的管理、文件的物理结构、目录结构和文件操作的实现，加深对文件系统内部功能和实现过程的理解。

实验要求

1. 在内存中开辟一个虚拟磁盘空间作为文件存储器，在其上实现一个简单的单用户文件系统。在退出这个简单的文件系统时，将该虚拟文件系统保存到磁盘上，以便下次再将它恢复到内存的虚拟磁盘空间中。
2. 提供以下操作：
    - new：建立一个新的简单文件系统；
    - sfs：打开一个简单文件系统；
    - exit：退出打开的简单文件系统；
    - mkdir：创建子目录；
    - rmdir：删除子目录；
    - ls：显示目录；
    - cd：更改当前目录；
    - create：创建文件；
    - open：打开文件；
    - close：关闭文件；
    - read：读文件；
    - write：写文件；
    - delete：删除文件。

代码编辑过程

- **语言的确定**

  在开始写 Lab4 前，一直在犹豫是用 `C` 还是 `Python` 去写，为了提高自己对 `Python` 的熟练度，最后决定选择了 `Python 2.7` 。

- **文件系统的保存**

  考虑到系统具有恢复文件系统的功能，要把编辑好的文件系统保存在硬盘里，所以，我把创建的文件系统命名为`.jk`格式的文件保存在执行目录下。

- **文件系统框架**

  我的实现参考了 `Linux` 的文件系统，即 `inode` 的处理方式，具体而言，文件系统会保存三部分数据，如下：

    - 目录树信息：在文件系统开头保存了一颗目录树，树的节点是文件夹类或者文件类。
    - `inode` 字典：紧接而来的是模仿 `Linux` 文件系统的 `inode` 字典，字典的左边是 `inode` 号码，

右边是对应文件的相关信息，这里只有文件总共的 `byte` 数和保存位置。

- 文件内容：最后一部分就是文件内容，可以由 `inode` 字典得到文件内容在文件系统中的位置，进而进行操作。

代码注释

- **类的代码**

```
  #文件夹类
  class Folder :
      def __init__( self, folderName = None, folderParent = None,folderContent =
list() ):
          self.folderName = folderName
              self.folderParent = folderParent
              self.folderContent = folderContent

  #文件类
  class File :
          def __init__( self, fileName = None, fileParent = None, fileInodeNo = None
):
              self.fileName = fileName
              self.fileParent = fileParent
              self.fileInodeNo = fileInodeNo

  #i节点信息类
  class inodeInfo :
          def __init__ (self, totalByte = None, block = None ):
              self.totalByte = totalByte
              self.block = block
```

这三段代码就是文件系统实现中使用到的三个类，分别是文件夹类、文件类和 i 节点信息类。

- **命令菜单**

```
  dispatchAction = {
      'new' : newFileSys,
      'sfs' : openFileSys,
      'exit' : exitFileSys,
      'mkdir' : mkdir,
      'rmdir' : rmdir,
      'ls' : showDirectory,
      'cd' : changeDirectory,
      'create' : createFile,
      'open' : openFile,
      'close' : closeFile,
      'read' : readFile,
      'write' : writeFile,
      'delete' : deleteFile,
      'help' : showHelp,
   }
```

上面的代码就是整个文件系统支持的所有的操作。

实验结果

实验完成！支持所有操作，操作过程配以大量提示，具有很强的鲁棒性。

具体实验流程如下：

- 打开程序

```
>>>
--------------------------------------------------------------------------------
Welcome to use the Simple File System created by JinKe
            Entry help to find some infomation
               Visit jinke.me to reach me
--------------------------------------------------------------------------------
```

- 输入 `help` 查看基本帮助

```
--------------------------------------------------------------------------------
Welcome to use the Simple File System created by JinKe
            Entry help to find some infomation
               Visit jinke.me to reach me
--------------------------------------------------------------------------------
help
You can use the following command :
            new         New a File System
            sfs         Open a File System
            exit        Exit a File System
            mkdir       Make a New Folder
            rmdir       Remove a Folder
            ls          Show Directory
            cd          Change Directory
            create      Create a New File
            open        Open a File
            close       Close a File
            read        Show the Content of the File
            write       Write Something Into the File
            delete      Delete a File
            quit        Quit the System
```

- 新建文件系统，默认新建后文件系统即打开

```
new jinke
Success!
You have created a new file system named jinke.jk
It is located in /Users/Hunter/Desktop !
```

- 若欲新建一个名字重复的文件系统，会报错

```
new jinke
Success!
You have created a new file system named jinke.jk
It is located in /Users/Hunter/Desktop !
new jinke
Error !
jinke.jk Exists !
```

- 新建文件夹，若建立名字重复的文件夹，会报错

```
mkdir Picture
Success !
You have created a new folder named Picture
mkdir Document
Success !
You have created a new folder named Document
mkdir Download
Success !
You have created a new folder named Download
mkdir Music
Success !
You have created a new folder named Music
mkdir Music
Error !
Music Exist !
```

- 新建文件，若建立名字重复的文件，会报错

```
create c
Success !
You have created a new file named c !
create python.py
Success !
You have created a new file named python.py !
create java.java
Success !
You have created a new file named java.java !
create java.java
Error !
java.java Exist !
```

- 显示目录，后面带`/`的表示是文件夹

```
ls
Picture/
Document/
Download/
Music/
c
python.py
java.java
```

- 删除文件夹，若删除不存在的文件夹，会报错，可用 ls 命令查看结果

```
rmdir Download
Success !
You have removed the folder named Download !
rmdir python.py
No such folder named python.py !
ls
Picture/
Document/
Music/
c
python.py
java.java
```

- 更改当前目录

  有三种方式，如下：

  - `cd` 后接文件夹名,若不存在该文件夹，则提示错误

    ```
    cd Download
    Sorry, No such folder named Download !
    cd Document
    ls
    mkdir Math
    Success !
    You have created a new folder named Math
    mkdir English
    Success !
    You have created a new folder named English
    ls
    Math/
    English/
    ```

  - `cd` 后接 `..`，代表返回上一级目录，若已是根目录，则不变

```
ls
Math/
English/
cd ..
ls
Picture/
Document/
Music/
c
python.py
java.java
```

- `cd` 后不接参数，直接返回根目录，若已是根目录，则不变

```
cd Document
ls
Math/
English/
cd
ls
Picture/
Document/
Music/
c
python.py
java.java
```

- 打开文件，若打开不存在的文件，会报错

```
open python.py
Success !
You have Opened the file named python.py !
```

```
open php.php
Error !
No such file named php.php !
```

- 关闭文件，若没有已经打开的文件，会报错

```
close
File closed !
close
Error !
No File Opened !
```

- 读文件，若文件未打开，会报错

```
close
File closed !
read
Error !
No File Opened !
open python.py
Success !
You have Opened the file named python.py !
read
import os
```

- 写文件，若文件未打开或已关闭，会报错

```
close
Error !
No File Opened !
write begin
Error !
No File Opened !
```

写文件操作支持3种方式，具体如下：

- `write`后接`begin`，表示将内容写入到文件起始处

```
read
import os

write begin
Please Enter the Content :
import pickle
Success !
read
import pickle
import os
```

- `write`后接`end`，表示将内容写入到文件末尾处

```
read
import pickle
import os

write end
Please Enter the Content :
orderList = ()
Success !
read
import pickle
import os
orderList = ()
```

- `write`后接数字，表示讲内容写入到文件中的某个地方去

```
read
import pickle
import os
orderList = ()

write 1000
Please Enter the Content :
+++
Error !
Mode Number Out of Range !
write 2
Please Enter the Content :
+++
Success !
read
im+++
port pickle
import os
orderList = ()
```

- 删除文件，若该文件不存在，提示错误

```
delete python.py
Success !
You have removed the file named python.py !
ls
Picture/
Document/
Music/
c
java.java
delete python.py
No such file named python.py !
```

- 退出当前文件系统

```
exit
Success!
You have exited the file system  named jinke.jk!
ls
Sorry, You should open or new a file system first !
cd
Sorry, You should open or new a file system first !
mkdir 3
Sorry, You should open or new a file system first !
```

- 打开一个文件系统

```
sfs jinke
Success!
You have opened the file system named jinke.jk
It is located in /Users/Hunter/Desktop!
ls
Picture/
Document/
Music/
c
java.java
```

- 退出程序

```
quit
--------------------------------------------------------------------------------
Goodbye !
Visit jinke.me to Find More Infomation
--------------------------------------------------------------------------------
>>>
```

个人总结

写这个程序花了差不多一天的时间，还是有很多收获的，具体如下：

- 加深了对文件系统内部功能和实现过程的理解
- 提高了 Python 的使用熟练度