**操作系统课程设计实验报告**

——实验四：文件系统

负责人姓名：柴舜

学号：14061142

日期：2016.5.20

**小组成员**

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 姓名 | 学号 | 实验分工 |
| 1 | 何玥 | 13211114 | 创建、删除目录，实现绝对路径和相对路径，实现ls函数的功能再更新，实现显示时间 |
| 2 | 周啸辰 | 14061130 | 更新ls函数功能，实现显示当前目录的绝对路径 |
| 3 | 张诚 | 14061146 | 实现cf函数的功能更新，实现of函数，用来读取创建文件中的内容，攥写会议记录 |
| 4 | 柴舜 | 14061142 | 整合代码，实验报告 |

目录

[1.实验目的 4](#_Toc451677597)

[2.需求说明 4](#_Toc451677598)

[2.1基本要求 4](#_Toc451677599)

[2.2 提高要求 4](#_Toc451677600)

[2.3 完成情况 5](#_Toc451677601)

[3.设计说明 6](#_Toc451677602)

[3.1 基本要求实现说明 6](#_Toc451677603)

[3.2 提高要求实现说明 7](#_Toc451677604)

[3.3 新增功能实现说明 9](#_Toc451677605)

[4.遇到的问题及解决方案 10](#_Toc451677606)

[1、.h文件没有修改完整 10](#_Toc451677607)

[2、U盘的挂载问题 10](#_Toc451677608)

[3、of函数崩溃 11](#_Toc451677609)

[4、目录只占一个簇 11](#_Toc451677610)

[5.收获和感想 12](#_Toc451677611)

# 1.实验目的

1.了解文件系统的作用和工作方式。

2.了解FAT文件系统的结构

3.学习文件管理系统的一般开发方法

# 2.需求说明

## 2.1基本要求

1、设计并实现一个目录列表函数（无须支持选项，如ls -a、ls -l等），用来显示当前目录下包含的文件信息。

2、设计并实现一个改变目录函数（无须处理路径名，如../../directoryName等），用来把当前目录切换到上一层目录或当前目录的子目录。

3、设计并实现一个删除文件函数，使用当前目录中的要删除的文件名作为参数，用来删除指定的文件，要注意文件的隐藏、只读和其他系统属性。

4.设计并实现一个创建文件函数，使用要创建的文件名和文件大小作为参数，用来创建一个新的文件。

## 2.2 提高要求

1. 增加创建目录的功能。
2. 增加删除目录的功能：通常需要先判断目录是否为空目录，若目录不为空，则需给出提示，并删除其包含的所有子目录和文件；若是空目录则可以直接删除。
3. 增加绝对路径和多级目录的支持：这里需要对输入的目录路径字符串进行解析，然后逐级查找目录。
4. **对*ud\_cf()***函数进行改进，使其可以向文件中写入实际内容，并根据写入的内容计算文件的实际大小。
5. **对*ud\_ls()***函数进行改进，增加对全部非根目录信息的读取（基本要求中仅读取一个扇区的非根目录细信息）。

## 2.3 完成情况

完成了以下功能：

1.程序能够正常运行

2.能够执行cd，ls，rm，cf，of，mkdir，exit等指令

3.实现了创建（mkdir）、删除（rm）目录的命令

4.实现各处支持绝对路径和相对路径。

5．实现ls函数的功能再更新，支持ls函数后添加绝对路径显示改路径下大所有子文件或目录的信息

6.实现文件信息中显示时间

7. 更新ls函数功能，实现显示当前目录的绝对路径

8. 实现cf函数的功能更新，能够向创建的文件中写入信息

9. 实现of函数，用来读取创建文件中的内容

# 3.设计说明

## 3.1 基本要求实现说明

1、设计并实现一个目录列表函数（无须支持选项，如ls -a、ls -l等），用来显示当前目录下包含的文件信息。

源代码已经实现了该功能，但只能读取其中的一簇信息，而且显示时间的信息也有问题，需要再提高要求中进行改进。

之前现对一些信息进行打印，如果为根目录，则将ud定位在根目录的起始地址，然后从根目录起始位置开始遍历，每次读取一个根目录项，获取其中的信息，然后打印，直到到达数据区的起始地址。

2、设计并实现一个改变目录函数（无须处理路径名，如../../directoryName等），用来把当前目录切换到上一层目录或当前目录的子目录。

源代码已经实现了该功能，但没有实现绝对路径和相对路径的功能，需要再提高要求中进行改进。

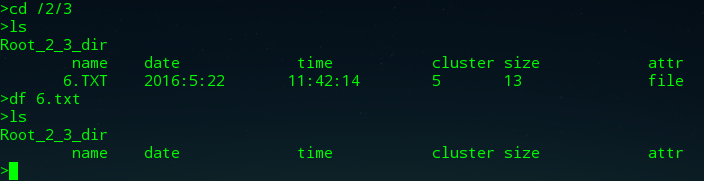
需要判断cd函数之后读取的信息是什么，如果为.或..且当前为根目录，则不切换，仍在当前目录。如果为..，且当前不为根目录，则切换到上一级子目录。

3、设计并实现一个删除文件函数，使用当前目录中的要删除的文件名作为参数，用来删除指定的文件，要注意文件的隐藏、只读和其他系统属性。

源代码已实现该功能。

首先找到需要删除的文件的第一个簇的簇号，然后读取簇的信息并记录，然后将该簇标记为0xe5，然后读取下一个簇的信息，直到为0xffff为止。

具体结果如下：



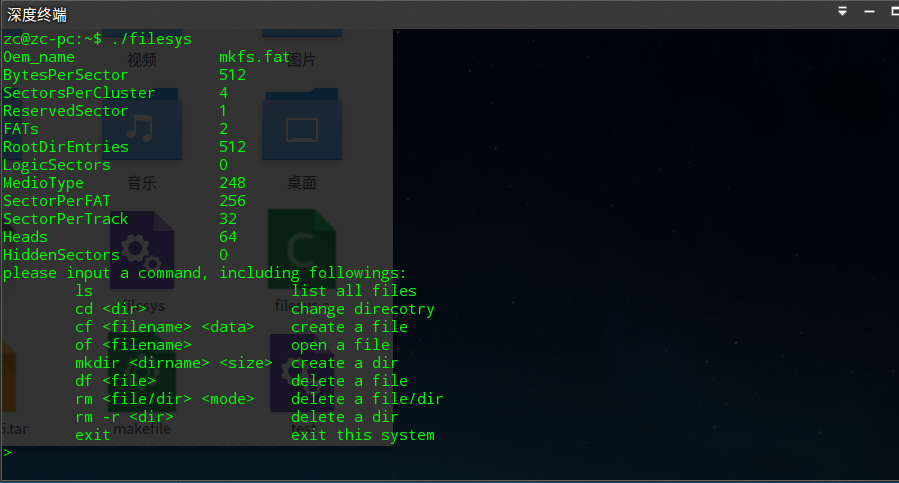
4.设计并实现一个创建文件函数，使用要创建的文件名和文件大小作为参数，用来创建一个新的文件。

源代码已经实现了这个功能，但是在创建文件的时候，没有读取当前创建文件时的系统时间，并将其存入文件信息中，此处需要改进。该函数现在只能创建有大小无内容的文件，这需要在提高要求中进行改进。

首先根据需要创建的大小判断文件需要占多少个簇，然后向后寻找空白簇，过程中需要将找到的下一个簇号存放在当前fat表中，直到文件存放完毕，然后将最后一个簇的fat表信息置为0xffff表示结束。之后将文件信息写入目录项中。若为根目录，则在根目录所在的簇中找到一个空白目录项或者已删除子目录项来存放；若为子目录则在子目录所在的簇中找到一个空白目录项或者已删除子目录项来存放。

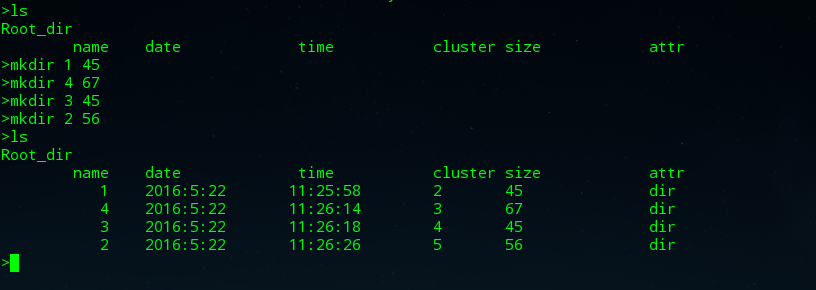
## 提高要求实现说明

首先打印启动项的内容



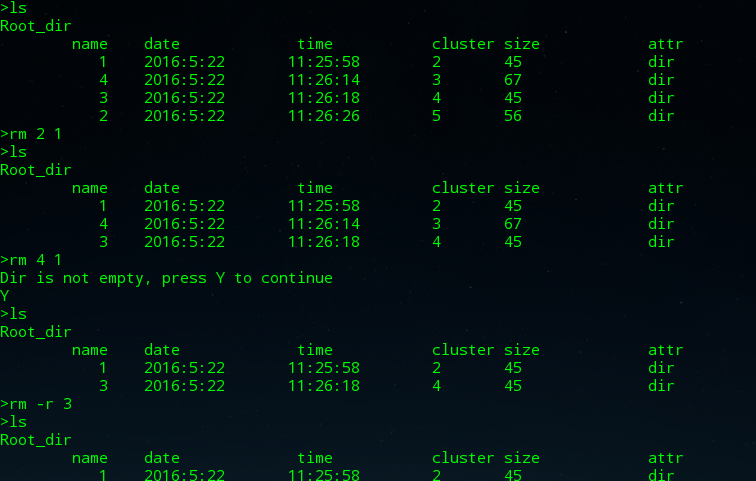
1. 增加创建目录的功能。

利用原先cf函数创建文件的功能进行扩展，创建目录时会限定目录的大小不能多于4个簇，然后寻找fat表的空区，记录目录所占簇的信息。如果当前创建的目录在根目录下，则将文件信息写入根目录项中；若位于子目录中，则将文件信息写入该位于数据区的子文件的子目录项中。



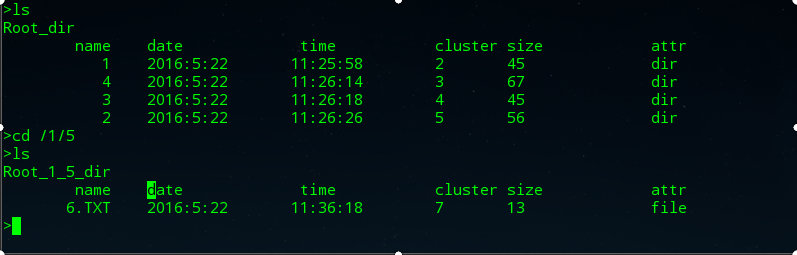
1. 增加删除目录的功能：通常需要先判断目录是否为空目录，若目录不为空，则需给出提示，并删除其包含的所有子目录和文件；若是空目录则可以直接删除。

删除目录的功能需要支持递归将目录中的所有信息删除，所以新增一个函数ls-r,能够将当前目录下所有的文件信息存放在一个二维数组中，便于删除文件及目录。现有两个删除函数df及df\_0，df支持删除文件、空目录和非空目录，df\_0支持删除目录，区别在于当利用df删除目录输入时不需要添加-r,删除的目录为非空目录时也会询问是否删除，而df\_0函数删除目录输入时需要加-r，删除非空目录时不会询问。df函数的先通过ls\_r函数将当前目录的所有子目录项信息存放，然后判断当前的路径是否为目录，若是，判断当前目录是否为非空目录，若是判断输入是否有大写Y若是，则递归将该目录中的所有子目录内容删除，使得该目录变成一个空目录，之后删除空目录。df\_0函数也同理，只不过中间不会判断是否有下一个字符大写Y输入，而是直接将非空目录转化为空目录，然后删除空目录。



1. 增加绝对路径和多级目录的支持：这里需要对输入的目录路径字符串进行解析，然后逐级查找目录。

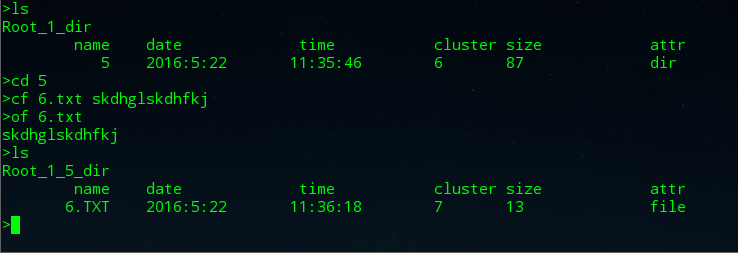
由于要求实现绝对路径和多级目录，在原先的cd函数中添加字符串解析的功能，将输入的路径标准化，将输入的路径全部转换为/……/……的规定标准格式，之后利用新添加的另一个函数cd\_f，将这段标准的路径按照/……来递归寻找路径，最终判断是否存在这样的路径，然后切换到该路径。



1. 对ud\_cf()函数进行改进，使其可以向文件中写入实际内容，并根据写入的内容计算文件的实际大小。

由于我们将创建文件和文件夹功能都写在了cf函数中，故保留了原来的函数。新添加了一个cfile函数专门用于创建文件，这个函数只传入两个参数，即文件名和要写入的内容，均为字符串类型。

新函数和cf内容类似，区别在于省去了文件和文件夹的判断语句，默认为文件，然后在创建文件成功后添加了向文件中写入字符串的语句。



5．对ud\_ls()函数进行改进，增加对全部非根目录信息的读取（基本要求中仅读取一个扇区的非根目录细信息）。

首先，ls函数的功能是实现将当前目录下的所有文件内容输出，这个功能在一开始的时候在读取子目录的时候只能读一个簇。实验的过程中我们写了一个do\_while()函数，用来读取当前簇并且调用函数获得下一个簇的簇号，如果簇号不是0XFFFF就继续进行循环，否则跳出。在实验过程中，为了测试是否可以读取多个簇的信息，在一个文件夹里面创建了2048个文件，然而在创建文件的时候出现了一些问题，当一个簇文件创建满了之后无法将文件创建到新的簇里面，

实际的情况是，我们当时刚刚写好的创建目录的函数，默认一个目录只会占有一个簇的大小，于是导致，即使创建了超过一个簇内容的目录项，也不可能创建成功，导致崩溃。

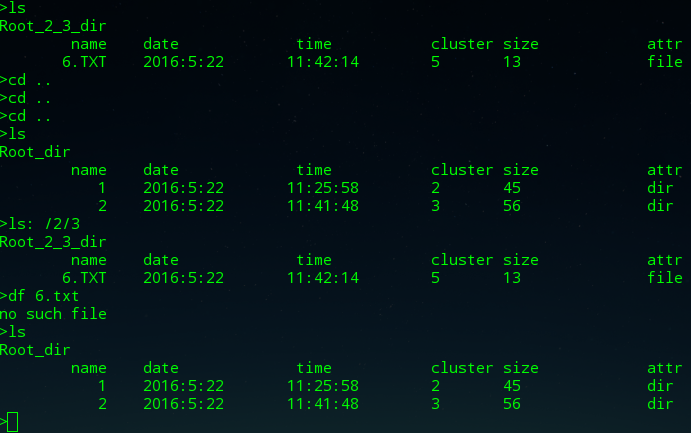
## 3.3 新增功能实现说明

1．实现显示时间

实验过程中发现利用ls打印文件信息是并没有显示创建文件时的日期和时间，所以我们在创建文件和创建文件夹的函数中增加了一段代码，来获取当前的系统时间，并将系统时间的内容分块存放在目录项的相应位置中，这样在ls显示的时候就可以打印文件的时间信息。

2. 实现ls函数后面接绝对路径，可以读该目录下的文件信息

当输入ls之后紧跟一个绝对路径时，利用cd 函数支持绝对路径和多级目录的功能，将当前目录切换至目标目录，然后利用ls函数打印当前目录的文件信息，之后返回原来的目录。



3. 实现of函数，读取创建文件中的字符内容

由于一开始无法挂载目录成功(实际上是挂载的姿势不对…)，所以导致我们无法在资源管理器下观察写入文件的内容是否正确，所以添加了of命令用来显示文件的写入内容。首先读取该文件的首簇号，计算在数据区存放该文件的的首地址，之后根据文件的大小利用read函数将文件的内容读出。

4. 实现在ls函数打印当前目录的绝对路径，方便显示

在调用函数过程中我们发现，如果在一个A文件夹中创建一个A文件夹，那么我在访问最里面的文件夹的时候显示的文件目录是A\_dir，这样我就不能判断出当前文件夹到底是哪一个。因此，我们在函数中添加了一些细节，显示当前的文件目录的绝对路径。

在得知当前文件的目录的时候可以获取该目录的父目录，于是可以递归得到之前所有文件的父目录的信息，直到路径为根目录。这样讲当前的绝对路径打出，方便显示。

# 4.遇到的问题及解决方案

## 1、.h文件没有修改完整

这个问题在我们发现的时候也是懊恼万分，首先第一次出现问题是在U盘的挂载上，每次挂载U盘之后执行程序都会显示打印启动项的信息有问题，程序强制退出，起初我们以为是我们挂在U盘的姿势不对，但经过多方询问之后发现是.h文件中的默认DEVNAME属性有误，该属性应该填写创建的data文件的路径，最终导致错误。

第二次出现问题比较晚，当我们得知正确的、可以从资源管理器看到文件的具体内容的方法之后，打开我们创建的文件，发现里面的内容有误，都是\00\等，又经过多方查证，发现是我们在处理data的大小上出现了问题，之前我们做写的程序是支持2G大小的虚拟文件，但现在我们做的只支持128M（由于创建的文件过大时，会导致真正意义上挂载不上的情况，所以我们选择把虚拟文件的大小改小），由于修改了U盘的大小，所以.h文件和.c文件中一些参数的大小都需要改变，但是我们之前都没有进行改动，所以导致终端没有问题，但是资源管理器中看到的内容错误。

## 2、U盘的挂载问题

这个问题是本次实验遇到的最大的问题，也是最难解决的问题，伴随着整个实验过程，甚至到最终我们也没有完全解决这个问题。

首先第一个大问题，已经创建好的data文件挂载好之后执行程序，在资源管理器中并看不到任何内容，由于这个问题非常的普遍，能够偶然看到内容的电脑非常少，以至于我们以为看不到内容才是正常情况，而这个错误一直持续到检查实验，隔壁告诉我们，**先执行程序内容再挂载U盘**，之后打开资源管理器，就能看到内容。

之前的问题的到来解决，但是下一个问题至今没有得到解决，就是在执行完程序之后，如果用资源管理器打开文件查看内容或使用终端查看内容之后，再启动程序会导致崩溃或不能执行操作，优势经过多方论证和考察，用U盘挂载之后，用资源管理器打开目录时会自动修改fat表的内容，导致信息错乱，目录或文件访问不能。而这个问题至今没有得到解决，原因也不明。

## 3、of函数崩溃

实现读取文件内容的函数of之后进行测试的时候，有时调用of函数就会显示段错误的崩溃，但是利用printf打印之后确定问题发生的地方，发生问题的原因是执行主函数时创建字符指针失败，导致不能将读入的内容写入。但是这个报错行为不能理解，而且崩溃时有时无，也不能查明原因。由于每一个文件和目录创建时都会规定所占空间不能多于四个簇的大小，所以直接将字符指针改为大小为四个簇+1的字符指针，让这个问题得以解决。

## 4、目录只占一个簇

之前我们为了测试ls函数是否能读取不止一个簇中的子目录项是，先创建了2000+个文件，但是发现ls只能读取一个簇中的内容，但是几经推敲发现，ls函数逻辑上没有问题，之后仔细研究创建目录的函数发现，默认目录只会占一个簇的大小，所以创建文件时，之后的超过一簇的文件都没有创建成功，所以导致ls函数读取也只能够读取一簇的子目录项。发现问题之后，将创建目录时限定为目录路的大小最多为4个簇的大小，这样就可以创建超过一簇的子目录项，最终测试ls函数是可以读取所有的子目录项。之后所写的创建文件的函数继承了这一点，要求文件的大小不能超过4个簇，解决of函数崩溃的问题。

# 5.收获和感想

这个实验的代码量虽然不是很大，但是要理解这些内容还是很不容易的，尤其是很多地方的功能相互存在连接，因此中间磨合的过程相对长了很多。不过在这过程中我们深刻的体会到几个人一起研究一个代码的乐趣和合作的艰难以及成就感，这种感觉是之前的几次程序没有的，最后一次实验给我们带来的感触也是最深刻的

实验各部分代码之间关联性很强，几个重要的变量如fatbuf、curdir等基本在每个方法都会出现，除非事先了解了整个设计框架，否则很多地方理解起来都比较头疼。

而上次作业的某些任务，只要弄懂某一个方法的覆盖范围，就可以完善该方法，其他的不用管太多但这次作业中几个方法的功能都是同级的，都与总体的框架密不可分。如果没有弄懂整个设计的体系，想要读懂或者写出其中任何一个方法都是不可能的。这也导致我刚开始写的时候进展非常缓慢，因为一直只关注当前方法的代码，不能对全局有个了解，对很多变量的理解都不够准确。因为不同方法涉及的功能领域可能完全不同。很难向上次作业那样，先读懂某个方法，再读懂某个方法，这次想读懂任何一个方法都要钻研很多代码，不过当读懂某个方法后，通常其他方法就大同小异了。这一问题在我们的代码交流中尤为突出，如果没有在会议过程中充分说明自己的理解和设计的话，其他人在自己修改后的基础上编程时，理解代码会很困难。不过通过阅读他人的修改，也可以提升自己理解程序和编写代码的熟练度。