山东大学 计算机科学与技术 学院

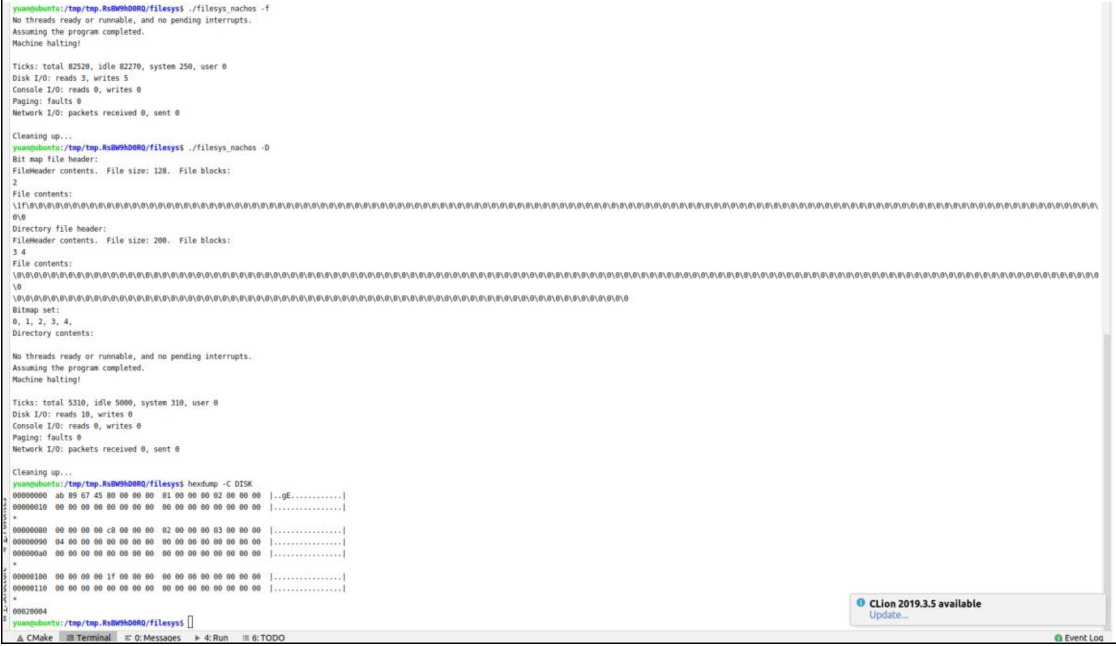
操作系统 课程实验报告

学号：201705130120	姓名：苑宗鹤	班级：17 1 班
实验题目：实验四		
实验学时：2	实验日期：2020/4/3	
实验目的：		
实验环境：ubuntu18 x64 windows10 clion		
实验步骤：		

实验结果：

仓库地址:

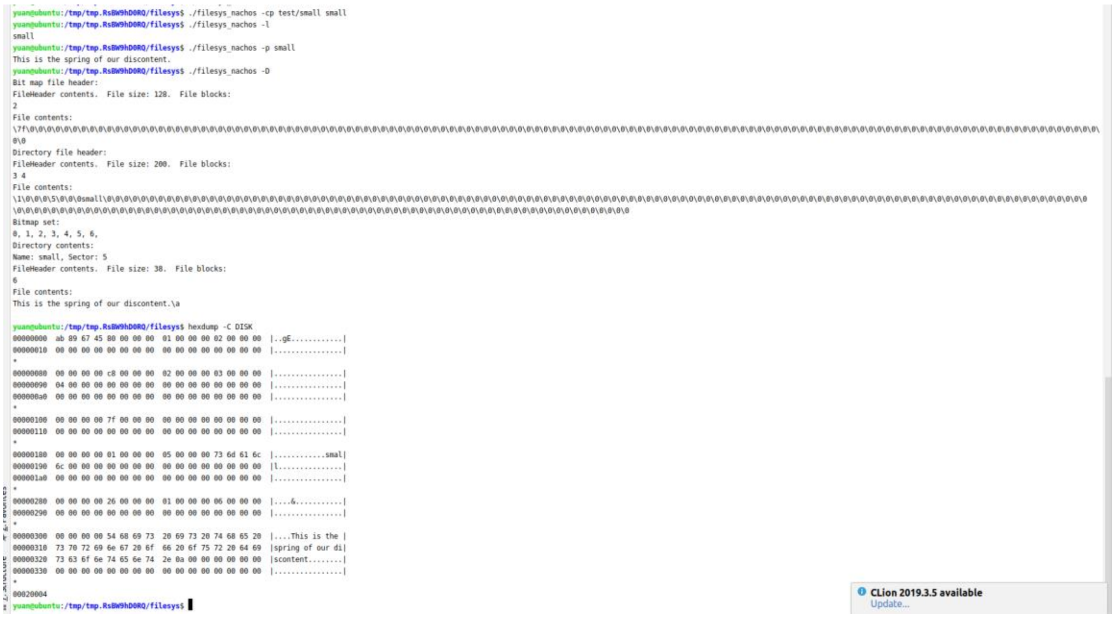
https://github.com/Yuandiaodiaodiao/nachos-cmake-x64



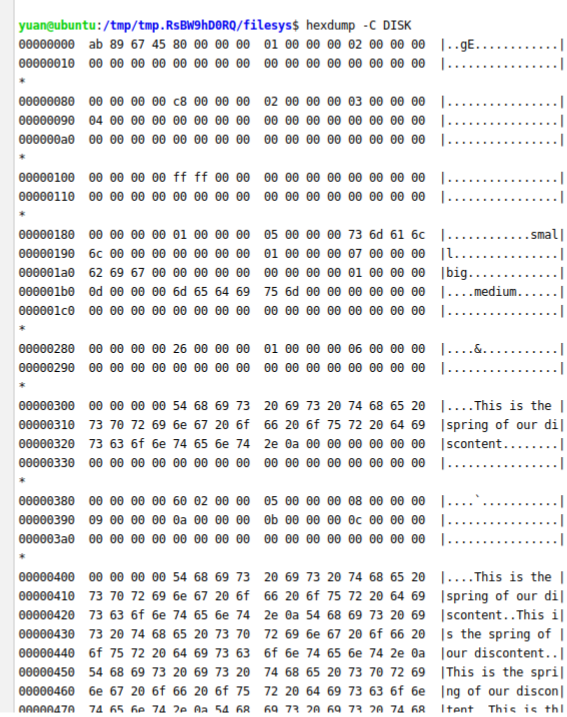
```
yuan@ubuntu:/tmp/tmp.RsBW9hD0RQ/filesys$ ./filesys_nachos -f
No threads ready or runnable, and no pending interrupts.
Assuming the program completed.
Machine halting!

Ticks: total 82520, idle 82270, system 250, user 0
Disk I/O: reads 3, writes 5
Console I/O: reads 0, writes 0
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
yuan@ubuntu:/tmp/tmp.RsBW9hD0RQ/filesys$ ./filesys_nachos -D
Bit map file header:
FileHeader contents.  File size: 128.  File blocks:
2
File contents:
\1f\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0
Directory file header:
FileHeader contents.  File size: 200.  File blocks:
3 4
File contents:
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Bitmap set:
0, 1, 2, 3, 4,
Directory contents:

No threads ready or runnable, and no pending interrupts.
Assuming the program completed.
Machine halting!

Ticks: total 5310, idle 5000, system 310, user 0
Disk I/O: reads 10, writes 0
Console I/O: reads 0, writes 0
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
yuan@ubuntu:/tmp/tmp.RsBW9hD0RQ/filesys$ hexdump -C DISK
00000000  ab 89 67 45 80 00 00 00  01 00 00 00 02 00 00 00  |..gE............|
00000010  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000080  00 00 00 00 c8 00 00 00  02 00 00 00 03 00 00 00  |................|
00000090  04 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000100  00 00 00 00 1f 00 00 00  00 00 00 00 00 00 00 00  |................|
00000110  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00020004
yuan@ubuntu:/tmp/tmp.RsBW9hD0RQ/filesys$ 
```

创建文件 显示文件系统结构 显示 hex

粘贴文件后查看文件系统结构



```
yuan@ubuntu:/tmp/tmp.RsBW9hD0RQ/filesys$ ./filesys_nachos -cp test/small small
yuan@ubuntu:/tmp/tmp.RsBW9hD0RQ/filesys$ ./filesys_nachos -l
small
yuan@ubuntu:/tmp/tmp.RsBW9hD0RQ/filesys$ ./filesys_nachos -p small
This is the spring of our discontent.
yuan@ubuntu:/tmp/tmp.RsBW9hD0RQ/filesys$ ./filesys_nachos -D
Bit map file header:
FileHeader contents.  File size: 128.  File blocks:
2
File contents:
\7f\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0
Directory file header:
FileHeader contents.  File size: 200.  File blocks:
3 4
File contents:
\1\0\0\0\5\0\0\0small\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Bitmap set:
0, 1, 2, 3, 4, 5, 6,
Directory contents:
Name: small, Sector: 5
FileHeader contents.  File size: 38.  File blocks:
6
File contents:
This is the spring of our discontent.\a

yuan@ubuntu:/tmp/tmp.RsBW9hD0RQ/filesys$ hexdump -C DISK
00000000  ab 89 67 45 80 00 00 00  01 00 00 00 02 00 00 00  |..gE............|
00000010  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000080  00 00 00 00 c8 00 00 00  02 00 00 00 03 00 00 00  |................|
00000090  04 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000100  00 00 00 00 7f 00 00 00  00 00 00 00 00 00 00 00  |................|
00000110  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000180  00 00 00 00 01 00 00 00  05 00 00 00 73 6d 61 6c  |............smal|
00000190  6c 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |l...............|
000001a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000280  00 00 00 00 26 00 00 00  01 00 00 00 06 00 00 00  |....&...........|
00000290  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000300  00 00 00 00 54 68 69 73  20 69 73 20 74 68 65 20  |....This is the |
00000310  73 70 72 69 6e 67 20 6f  66 20 6f 75 72 20 64 69  |spring of our di|
00000320  73 63 6f 6e 74 65 6e 74  2e 0a 00 00 00 00 00 00  |scontent........|
00000330  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00020004
yuan@ubuntu:/tmp/tmp.RsBW9hD0RQ/filesys$
```



```
yuan@ubuntu:/tmp/tmp.RsBW9hD0RQ/filesys$ hexdump -C DISK
00000000  ab 89 67 45 80 00 00 00  01 00 00 00 02 00 00 00  |..gE............|
00000010  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000080  00 00 00 00 c8 00 00 00  02 00 00 00 03 00 00 00  |................|
00000090  04 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000100  00 00 00 00 ff ff 00 00  00 00 00 00 00 00 00 00  |................|
00000110  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000180  00 00 00 00 01 00 00 00  05 00 00 00 73 6d 61 6c  |............smal|
00000190  6c 00 00 00 00 00 00 00  01 00 00 00 07 00 00 00  |l...............|
000001a0  62 69 67 00 00 00 00 00  00 00 00 00 01 00 00 00  |big.............|
000001b0  0d 00 00 00 6d 65 64 69  75 6d 00 00 00 00 00 00  |....medium......|
000001c0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000280  00 00 00 00 26 00 00 00  01 00 00 00 06 00 00 00  |....&...........|
00000290  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000300  00 00 00 00 54 68 69 73  20 69 73 20 74 68 65 20  |....This is the |
00000310  73 70 72 69 6e 67 20 6f  66 20 6f 75 72 20 64 69  |spring of our di|
00000320  73 63 6f 6e 74 65 6e 74  2e 0a 00 00 00 00 00 00  |scontent........|
00000330  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000380  00 00 00 00 60 02 00 00  05 00 00 00 08 00 00 00  |....`...........|
00000390  09 00 00 00 0a 00 00 00  0b 00 00 00 0c 00 00 00  |................|
000003a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000400  00 00 00 00 54 68 69 73  20 69 73 20 74 68 65 20  |....This is the |
00000410  73 70 72 69 6e 67 20 6f  66 20 6f 75 72 20 64 69  |spring of our di|
00000420  73 63 6f 6e 74 65 6e 74  2e 0a 54 68 69 73 20 69  |scontent..This i|
00000430  73 20 74 68 65 20 73 70  72 69 6e 67 20 6f 66 20  |s the spring of |
00000440  6f 75 72 20 64 69 73 63  6f 6e 74 65 6e 74 2e 0a  |our discontent..|
00000450  54 68 69 73 20 69 73 20  74 68 65 20 73 70 72 69  |This is the spri|
00000460  6e 67 20 6f 66 20 6f 75  72 20 64 69 73 63 6f 6e  |ng of our discon|
00000470  74 65 6e 74 2e 0a 54 68  69 73 20 69 73 20 74 68  |tent..This is th|
```

4 字节标识 +0 扇区 存放位示图文件位置大小

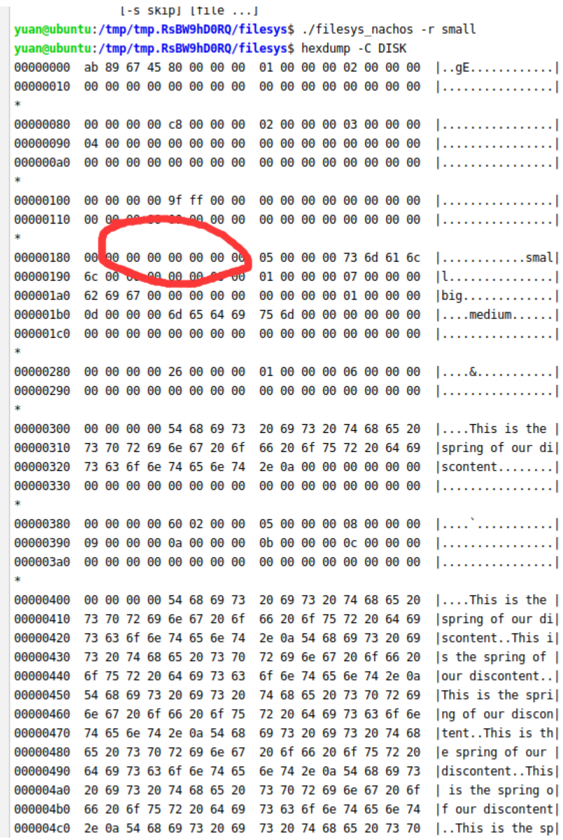
1 扇区 存放目录表大小 扇区数 扇区位置

2 扇区 位示图数据

3 扇区 存放目录表数据

5 扇区 small 文件头 声明文件大小 分配扇区数 扇区列表

删除 small 文件



```
              [-s skip] [file ...]
yuan@ubuntu:/tmp/tmp.RsBW9hD0RQ/filesys$ ./filesys_nachos -r small
yuan@ubuntu:/tmp/tmp.RsBW9hD0RQ/filesys$ hexdump -C DISK
00000000  ab 89 67 45 80 00 00 00  01 00 00 00 02 00 00 00  |..gE............|
00000010  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000080  00 00 00 00 c8 00 00 00  02 00 00 00 03 00 00 00  |................|
00000090  04 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000000a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000100  00 00 00 00 9f ff 00 00  00 00 00 00 00 00 00 00  |................|
00000110  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000180  00 00 00 00 00 00 00 00  05 00 00 00 73 6d 61 6c  |............smal|
00000190  6c 00 00 00 00 00 00 00  01 00 00 00 07 00 00 00  |l...............|
000001a0  62 69 67 00 00 00 00 00  00 00 00 00 01 00 00 00  |big.............|
000001b0  0d 00 00 00 6d 65 64 69  75 6d 00 00 00 00 00 00  |....medium......|
000001c0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000280  00 00 00 00 26 00 00 00  01 00 00 00 06 00 00 00  |....&...........|
00000290  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000300  00 00 00 00 54 68 69 73  20 69 73 20 74 68 65 20  |....This is the |
00000310  73 70 72 69 6e 67 20 6f  66 20 6f 75 72 20 64 69  |spring of our di|
00000320  73 63 6f 6e 74 65 6e 74  2e 0a 00 00 00 00 00 00  |scontent........|
00000330  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000380  00 00 00 00 60 02 00 00  05 00 00 00 08 00 00 00  |....`...........|
00000390  09 00 00 00 0a 00 00 00  0b 00 00 00 0c 00 00 00  |................|
000003a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000400  00 00 00 00 54 68 69 73  20 69 73 20 74 68 65 20  |....This is the |
00000410  73 70 72 69 6e 67 20 6f  66 20 6f 75 72 20 64 69  |spring of our di|
00000420  73 63 6f 6e 74 65 6e 74  2e 0a 54 68 69 73 20 69  |scontent..This i|
00000430  73 20 74 68 65 20 73 70  72 69 6e 67 20 6f 66 20  |s the spring of |
00000440  6f 75 72 20 64 69 73 63  6f 6e 74 65 6e 74 2e 0a  |our discontent..|
00000450  54 68 69 73 20 69 73 20  74 68 65 20 73 70 72 69  |This is the spri|
00000460  6e 67 20 6f 66 20 6f 75  72 20 64 69 73 63 6f 6e  |ng of our discon|
00000470  74 65 6e 74 2e 0a 54 68  69 73 20 69 73 20 74 68  |tent..This is th|
00000480  65 20 73 70 72 69 6e 67  20 6f 66 20 6f 75 72 20  |e spring of our |
00000490  64 69 73 63 6f 6e 74 65  6e 74 2e 0a 54 68 69 73  |discontent..This|
000004a0  20 69 73 20 74 68 65 20  73 70 72 69 6e 67 20 6f  | is the spring o|
000004b0  66 20 6f 75 72 20 64 69  73 63 6f 6e 74 65 6e 74  |f our discontent|
000004c0  2e 0a 54 68 69 73 20 69  73 20 74 68 65 20 73 70  |..This is the sp|
```

可以看到在目录表数据中 扇区 5 的占用已经被解除

但是其他的信息均未变化，实现删除的方法就是将文件打上未占用的标记

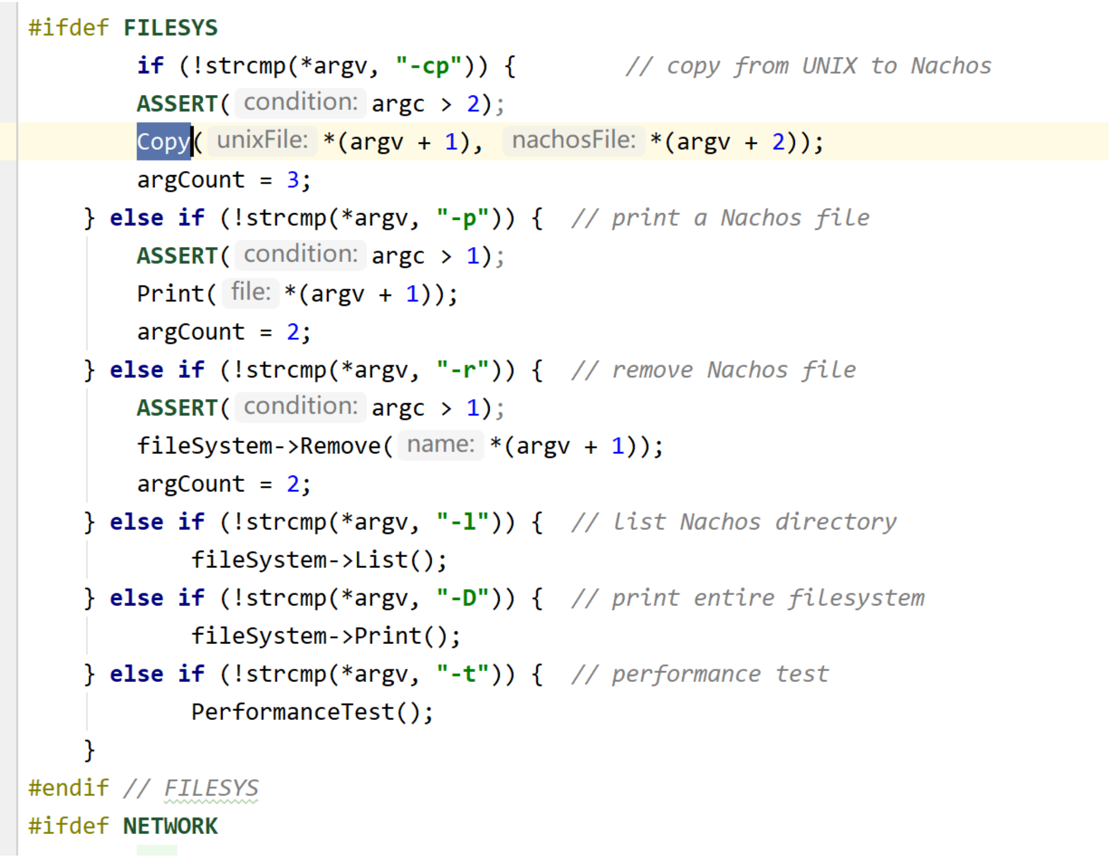
问题:

1. 复制三个文件后 DISK 上有三个文件
2. Big 的数据块扇区号是多少

   首先去扇区 1 找到目录表头文件
   发现目录表起始地址位于 3 扇区
   然后从 3 扇区开始寻找名为 big 的文件
   在 big 所在的 20 个字节 其中
   第 5 个字节开始 记录文件头的位置
   这里为 7 扇区(我粘贴了 small 之后就粘贴了 big)
   则 7 扇区就是 big 的文件头位置
   在 big 的文件头中找到第 9 个字节开始 是扇区号列表
   可以看到 第一个扇区号是在 8 号扇区
   则 big 的数据块位于 8 号扇区
3. big 的头文件位置在哪

   由上面知 头文件在 7 扇区
4. 确认位置 按照上面的分析对应 Hex 即可确定位置

命令行



```
#ifdef FILESYS
        if (!strcmp(*argv, "-cp")) {        // copy from UNIX to Nachos
        ASSERT( condition: argc > 2);
        Copy( unixFile: *(argv + 1),  nachosFile: *(argv + 2));
        argCount = 3;
    } else if (!strcmp(*argv, "-p")) {  // print a Nachos file
        ASSERT( condition: argc > 1);
        Print( file: *(argv + 1));
        argCount = 2;
    } else if (!strcmp(*argv, "-r")) {  // remove Nachos file
        ASSERT( condition: argc > 1);
        fileSystem->Remove( name: *(argv + 1));
        argCount = 2;
    } else if (!strcmp(*argv, "-l")) {  // list Nachos directory
            fileSystem->List();
    } else if (!strcmp(*argv, "-D")) {  // print entire filesystem
            fileSystem->Print();
    } else if (!strcmp(*argv, "-t")) {  // performance test
            PerformanceTest();
    }
#endif // FILESYS
#ifdef NETWORK
```

-cp -p -r -l -d -t 分别启动对应的函数来执行对应的功能

打开文件系统中的文件:

`FileSystem::Open =>`

如果文件存在 就使用 `OpenFile` 打开文件

下面我 word 崩溃了 临死之前截了个图

如果文件存在 就使用 OpenFile 打开文件

否则返回 NULL

FileSystem::Create

如果文件不存在

则尝试找到一个空白的扇区装入头文件

情况正常后将头文件写入磁盘

并更新目录表

OpenFile::Write

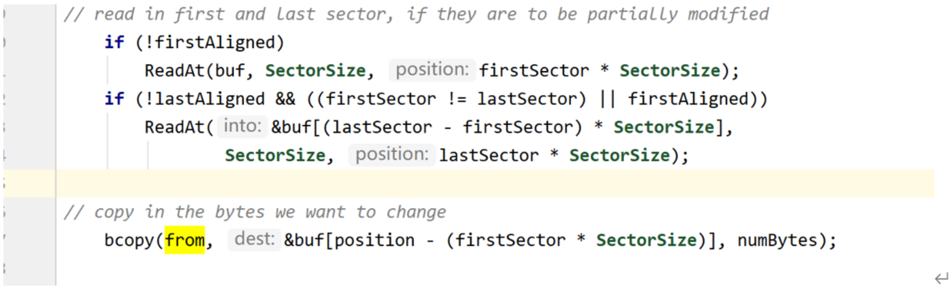
每个文件对象维护自身的写入指针

Write 用来维护写入指针的同时调用 OpenFile::WriteAt 进行实际上的写入

OpenFile::WriteAt

按照文件长度定位可写入的扇区范围

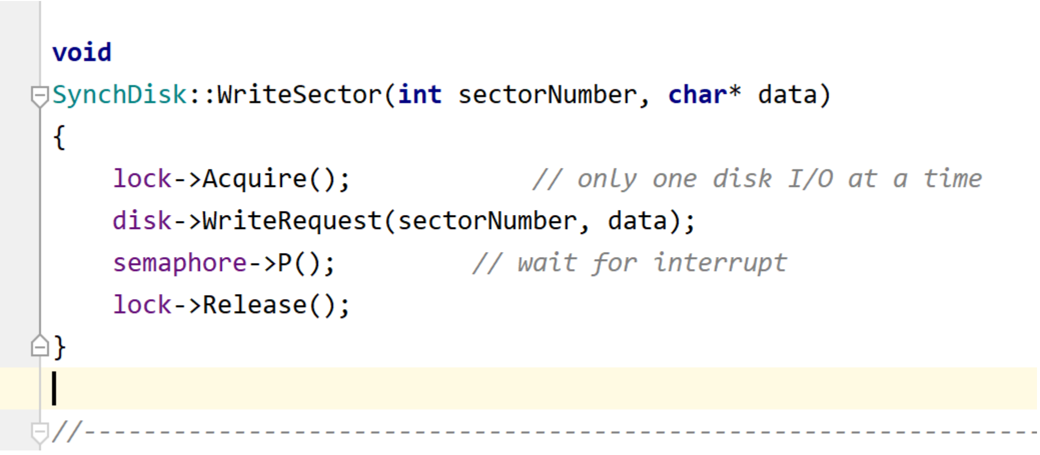
读入起止的扇区到 buff 里



```
// read in first and last sector, if they are to be partially modified
    if (!firstAligned)
        ReadAt(buf, SectorSize,  position: firstSector * SectorSize);
    if (!lastAligned && ((firstSector != lastSector) || firstAligned))
        ReadAt( into: &buf[(lastSector - firstSector) * SectorSize],
                SectorSize,  position: lastSector * SectorSize);

// copy in the bytes we want to change
    bcopy(from,  dest: &buf[position - (firstSector * SectorSize)], numBytes);
```

然后按照传入的指针进行文件数据覆盖

之后从 buff 里执行 synchDisk->WriteSector 来将文件从缓存中写入到真实文件系统中



```
void
SynchDisk::WriteSector(int sectorNumber, char* data)
{
    lock->Acquire();            // only one disk I/O at a time
    disk->WriteRequest(sectorNumber, data);
    semaphore->P();         // wait for interrupt
    lock->Release();
}

//----------------------------------------------------------------------
```

先加 io 锁 然后执行文件写入 在中断后归还 io 锁的权限

问题及收获：