```
/**************************************************************
* Class:  CSC-415-02 Summer 2020
* Name: Team Penta - Duy Nguyen, Taylor Artunian, Wameedh Mohammed Ali
* Student ID: 917446249 (Duy), 920351715 (Taylor), 920678405 (Wameedh)
* Project: Basic File System - PentaFS
*
* Description: This is Team Penta's file system write up.
*
**************************************************************/
```

# Group Project - File System

Github Repository Link:
https://github.com/CSC415-02-Summer2020/group-term-assignment-file-system-tartunian

*Things to Note*: We had a fourth member but they decided to drop out of our group, making it difficult to distribute work and set a clear plan as to how to proceed. There are also three branches because of some issues with pushing and merging to github, using any of them is fine.

## ***Design:***

The way we designed our file system was:

- Penta File system does not support recursive folder/file creation.
- MAX children per folder (parent) is 64
- Our data allocation is indexed
- b_write by its nature overwrites files
- Penta File system does not remove directories that contain files or folders. It only remove empty folders/Dir
- Penta File system has name limit of 20 chars for each file/folder

## ***Summary of Work Done/Explanation*:**

This is Team Penta's file system, where our design uses indexed allocation as our allocation method and a bit vector to manage the free-space. It is divided into several files, with the main ones being titled with b_io, fsLow, fileExplorer, bitMap, fsMakeVol, mfs, and the fsshell. The fsshell.c is our driver program. It is a modified version of the shell file given to us.

The b_io files contain the methods of how we read from and write to our file system from Linux. They contain the methods b_init, b_getFCB, b_open, b_write, b_read and b_close. It contains the methods from previous assignments, modified to work with our file system. The b_init initializes our file control block, a structure that contains all the information pertaining to files being opened, read or written to in our file system. The b_getFCB looks for the next available slot in our file control block array. The b_open is a modified version of the ones from the assignments that will make sure the FCB array is initialized and take a file name to open in our file system with a flag that decides if they want to open, create or truncate the file. The b_write writes a given amount to a file in our file system using LBAwrite from fsLow. The b_read is modified to read a given amount from a file in our file system's disk. It will always read the same amount of bytes equal to the block size, in this case 512, each time which leads to additional characters being put into the buffer. The b_close will close an opened file in our file system and write any remaining bits if a file was being written to.

The bitMap files hold the methods to assign the bits in our bit vector for free-space. The methods in it are setBit, clearBit and findBit, which all take an integer array and an index. The setBit will take a bit vector and at that index, set it to being occupied which is 1. The clearBit will take a bit vector and free the bit at that index, in this case, free is equal to 0. The findBit will take a bit vector and return whether or not the bit at the given index is free or occupied.

The fsMakeVol files are how the volume control block is managed and contains the data for each block in the volume. It stores and manages the free-space bit vector

The fsFormat.c is how we format each block, given a name and specified sizes for the volume and each block.

The mfs files contain the data and methods that access and change the inodes in our file system. To help denote what type of inode it is, we created an enumeration of InodeType to distinguish whether it is a file or a directory. In addition to the methods that were already there, some of the ones we added are mfs_init, writeInodes, mfs_close, getInode, etc. Each method is used to help with readability and keep good structure within our code. This file holds the methods that create, edit, remove, open and close files and directories in our file system with several helper methods to get the information on the inodes needed.

***What Works***:

Almost everything in our file system works as long as the instructions to compile and run it are followed. Some things work to a certain extent and not as intended. It can make files when calling the b_io functions and somewhat copies from Linux to our file system and vice versa.

***What Doesn't Work and Why***:
The commands that do not work quite as intended are the cp2l and cp2fs. They copy but when doing it from Linux to ours, the file is gibberish. When copying from a file in our file system to Linux, it does copy the contents but does copy additional elements due to the way it reads.

These are the following commands that can be inputted into the shell with a small description as to whether it works and if so, how.

| **Command** | **Status** |
|---|---|
| help | Works, displays every command |
| cd | Works with relative and absolute paths |
| pwd | Works, displays current working directory |
| md | Works with absolute paths, creates directories |
| ls | Works but not with parameters/flags, is able to display files and directories in cwd |
| rm | Works for folders (absolute paths), can successfully remove files and directories |
| cp2fs | File gets created but it's data blocks appear to be gibberish. |
| cp | Errors during b_write. |
| cp2l | Creates a file in Linux just fine, but never writes any data. Permissions are not set properly. Can write more than needed, resulting in additional characters at the end |

***Difficulties We Had*:**

Some of us have another class and so those members would be unavailable during some days or certain times, but they were willing to contribute in any way they could. We also originally had 4 members but one person decided to drop out midway and it made it more difficult to decide how to split up work and slowed down our progress quite a bit.

Github was also difficult to work with sometimes when it came to pushing and merging. Sometimes, when we had multiple people working on the repository at the same time, merging became an issue and would sometimes not let us push or pull or even overwrote some of the work. We did not lose anything important luckily, but it did make us spend extra time to figure it out.

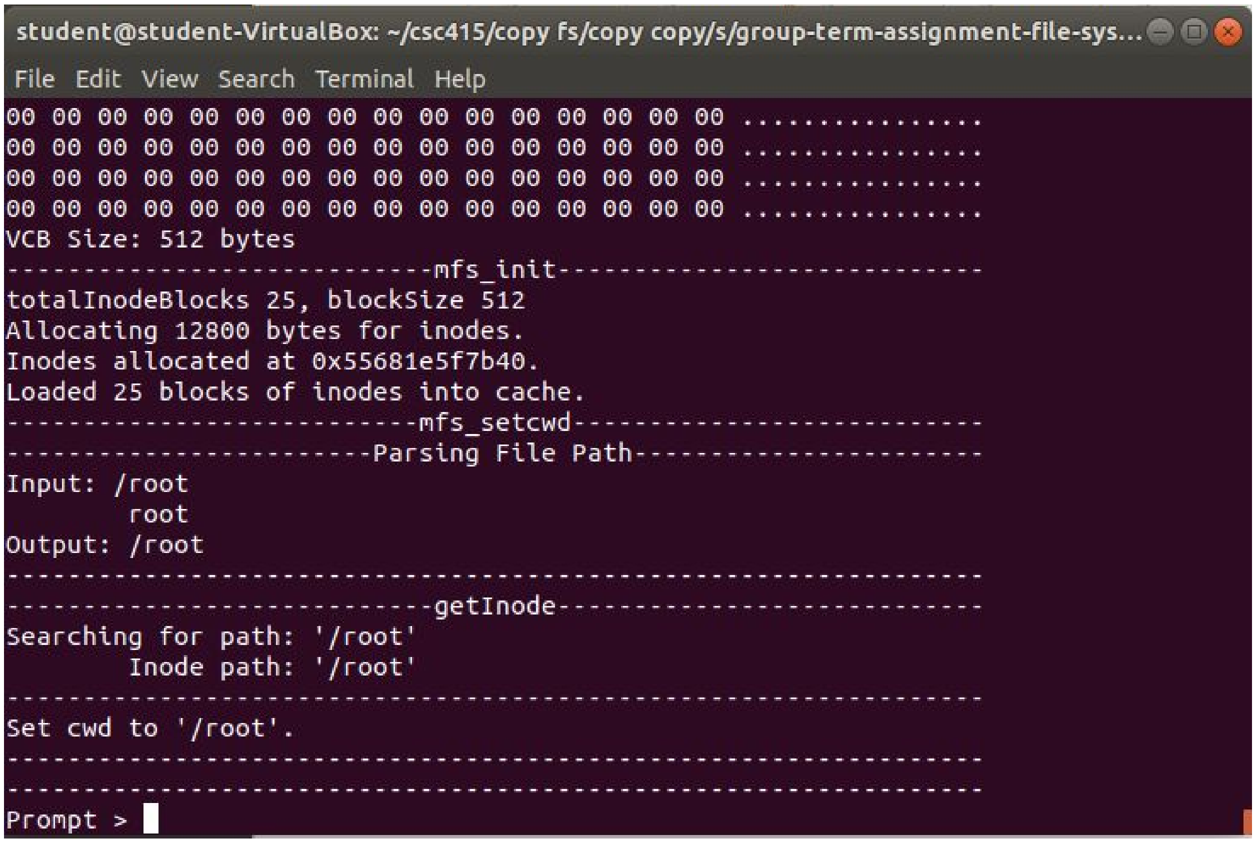
Some of the methods that cause(d)(s) us trouble are some of the b_io methods because of the adjustments of doing it from our file system. We had to stop and think about how we had to write to our file system and how to read from it, leading us to try different things that cost us time to debug.

***How The Driver Works*:**

For safety purposes, doing a make wipe, followed by make format. This will create a proper volume for the file system to use. After this is done, doing make run will run the shell program that interacts with our file system.

***Compile and Run Instructions***:

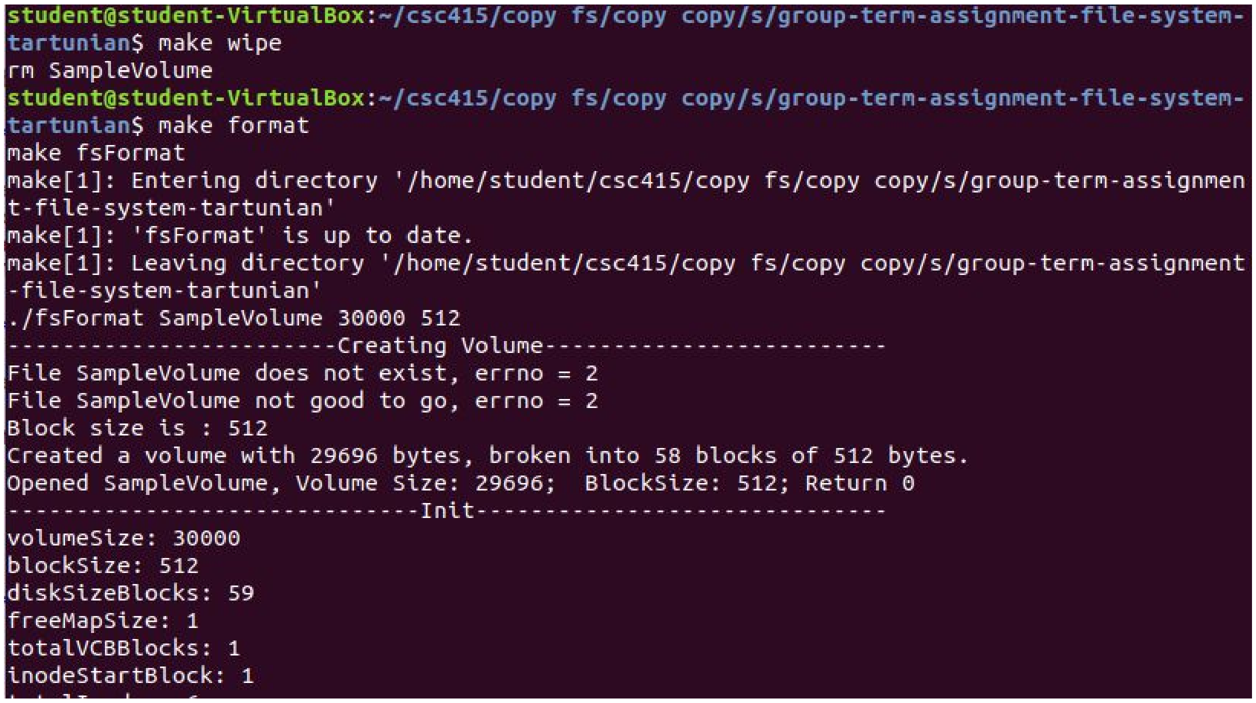
1. Open the terminal and clone the repository.
2. Change directories into the repository.
3. To actually run the code, first do “make wipe” and then a “make format”. This will make sure the volume is created and formatted to run with our file system.
4. Now, you can do a “make run” and the following should show up:


student@student-VirtualBox: ~/csc415/copy fs/copy copy/s/group-term-assignment-file-sys...
File Edit View Search Terminal Help

```
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
VCB Size: 512 bytes
----------------------------mfs_init----------------------------
totalInodeBlocks 25, blockSize 512
Allocating 12800 bytes for inodes.
Inodes allocated at 0x55681e5f7b40.
Loaded 25 blocks of inodes into cache.
---------------------------mfs_setcwd---------------------------
------------------------Parsing File Path-----------------------
Input: /root
        root
Output: /root
----------------------------------------------------------------
----------------------------getInode----------------------------
Searching for path: '/root'
        Inode path: '/root'
----------------------------------------------------------------
Set cwd to '/root'.
----------------------------------------------------------------
----------------------------------------------------------------
Prompt > 
```

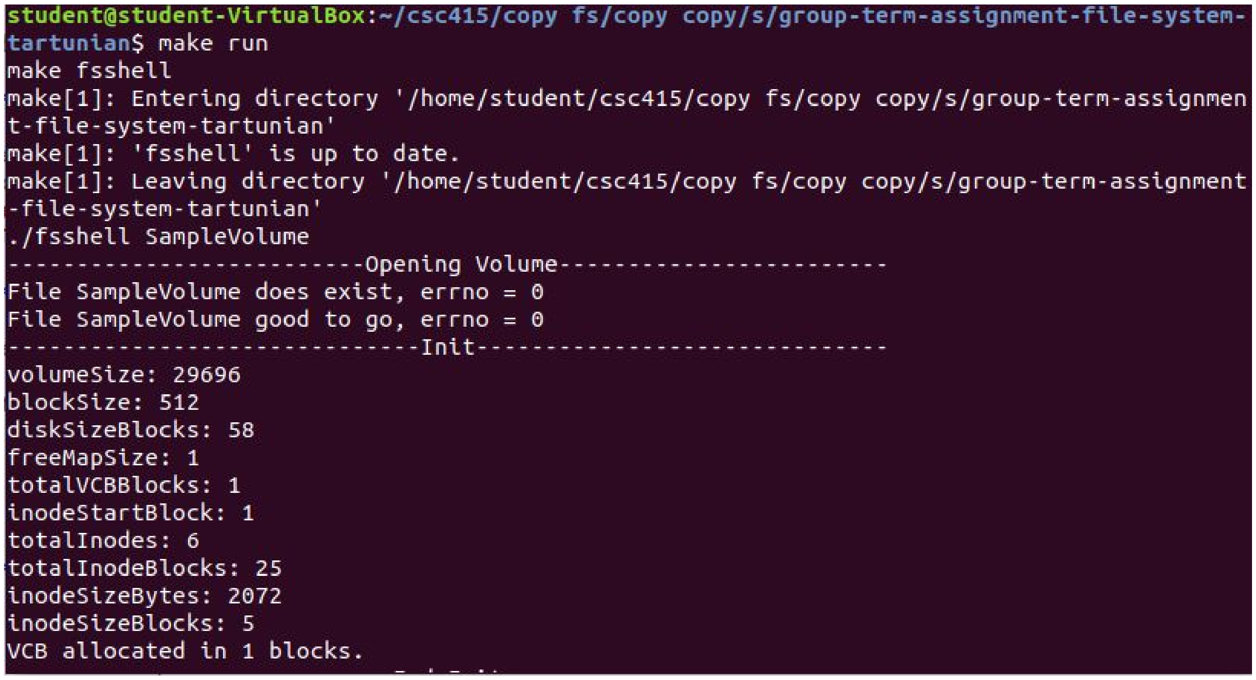
5. To display every available command, type help. Other than that, type in any commands available according to the table.
6. When you no longer want to use it, type exit.

*After make wipe and make format*



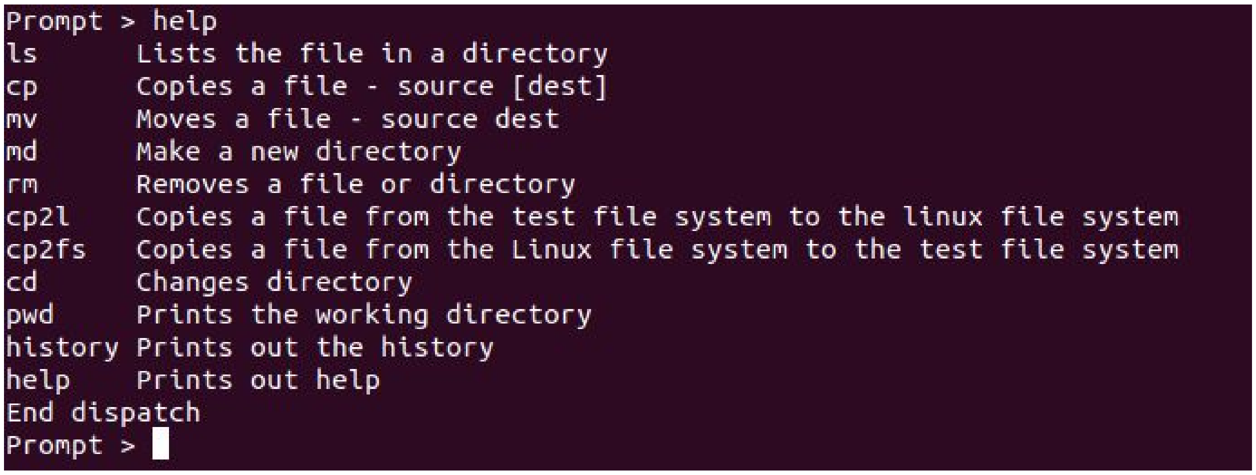
```
student@student-VirtualBox:~/csc415/copy fs/copy copy/s/group-term-assignment-file-system-
tartunian$ make wipe
rm SampleVolume
student@student-VirtualBox:~/csc415/copy fs/copy copy/s/group-term-assignment-file-system-
tartunian$ make format
make fsFormat
make[1]: Entering directory '/home/student/csc415/copy fs/copy copy/s/group-term-assignmen
t-file-system-tartunian'
make[1]: 'fsFormat' is up to date.
make[1]: Leaving directory '/home/student/csc415/copy fs/copy copy/s/group-term-assignment
-file-system-tartunian'
./fsFormat SampleVolume 30000 512
------------------------Creating Volume-------------------------
File SampleVolume does not exist, errno = 2
File SampleVolume not good to go, errno = 2
Block size is : 512
Created a volume with 29696 bytes, broken into 58 blocks of 512 bytes.
Opened SampleVolume, Volume Size: 29696;  BlockSize: 512; Return 0
------------------------------Init------------------------------
volumeSize: 30000
blockSize: 512
diskSizeBlocks: 59
freeMapSize: 1
totalVCBBlocks: 1
inodeStartBlock: 1
```

*After make run*



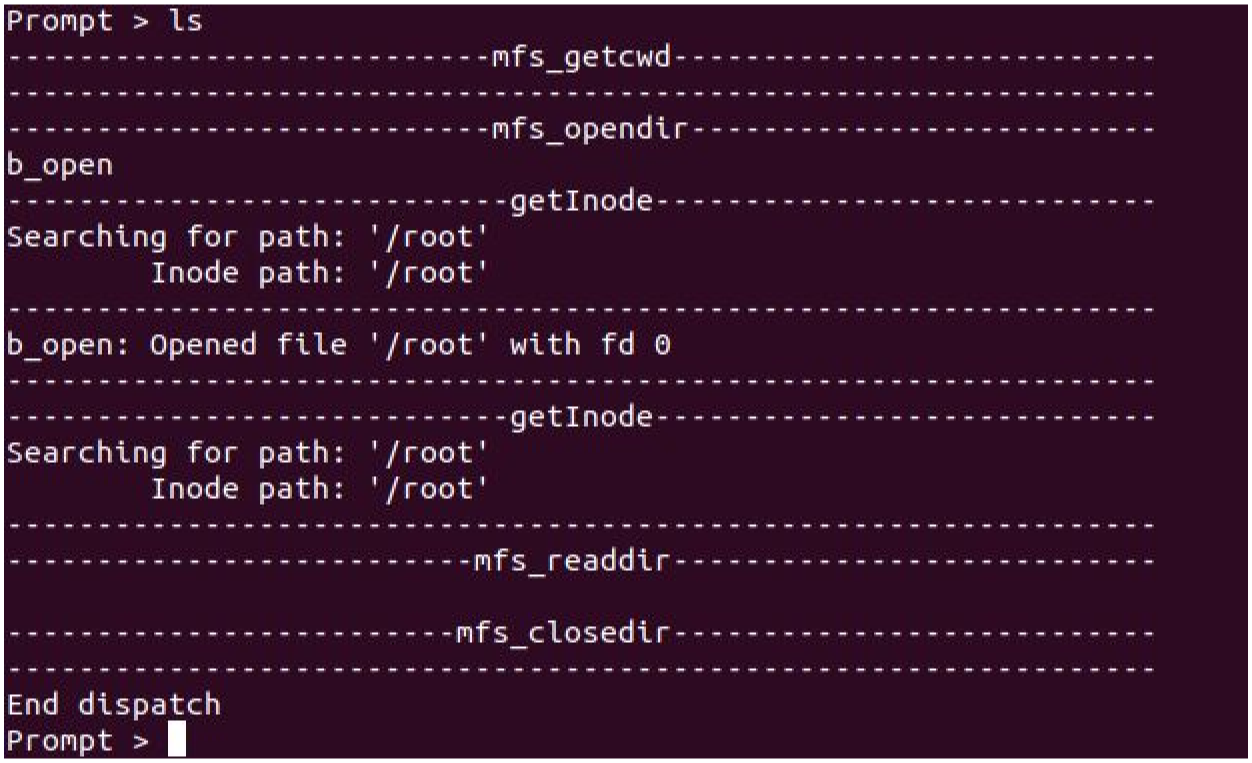
```
student@student-VirtualBox:~/csc415/copy fs/copy copy/s/group-term-assignment-file-system-
tartunian$ make run
make fsshell
make[1]: Entering directory '/home/student/csc415/copy fs/copy copy/s/group-term-assignmen
t-file-system-tartunian'
make[1]: 'fsshell' is up to date.
make[1]: Leaving directory '/home/student/csc415/copy fs/copy copy/s/group-term-assignment
-file-system-tartunian'
./fsshell SampleVolume
--------------------------Opening Volume------------------------
File SampleVolume does exist, errno = 0
File SampleVolume good to go, errno = 0
------------------------------Init------------------------------
volumeSize: 29696
blockSize: 512
diskSizeBlocks: 58
freeMapSize: 1
totalVCBBlocks: 1
inodeStartBlock: 1
totalInodes: 6
totalInodeBlocks: 25
inodeSizeBytes: 2072
inodeSizeBlocks: 5
VCB allocated in 1 blocks.
```

*Testing help command*



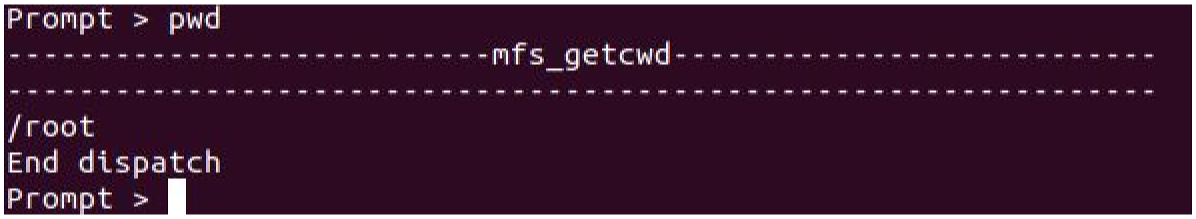
```
Prompt > help
ls      Lists the file in a directory
cp      Copies a file - source [dest]
mv      Moves a file - source dest
md      Make a new directory
rm      Removes a file or directory
cp2l    Copies a file from the test file system to the linux file system
cp2fs   Copies a file from the Linux file system to the test file system
cd      Changes directory
pwd     Prints the working directory
history Prints out the history
help    Prints out help
End dispatch
Prompt >
```

*Testing ls command*



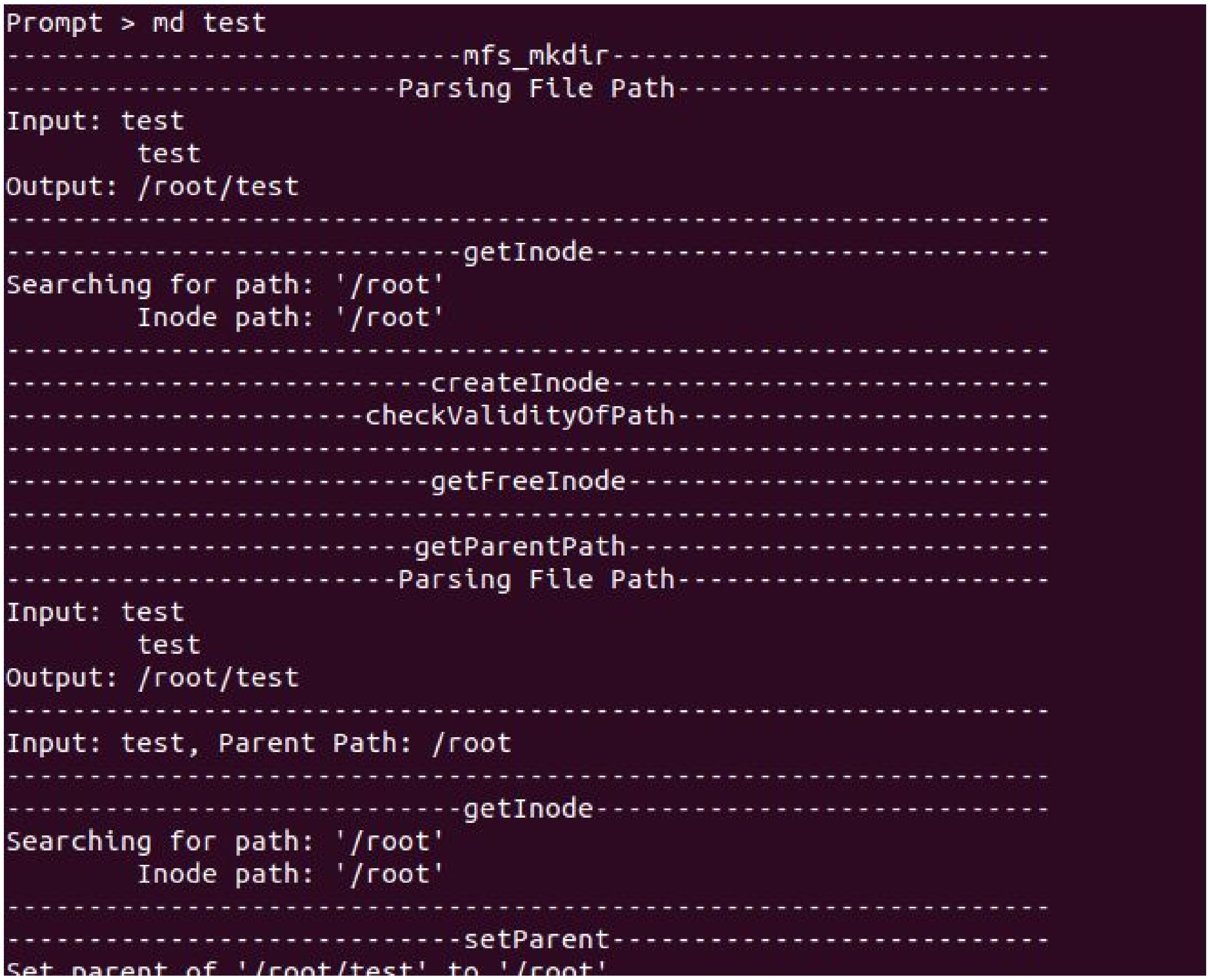
```
Prompt > ls
---------------------------mfs_getcwd---------------------------
-----------------------------------------------------------------
---------------------------mfs_opendir--------------------------
b_open
----------------------------getInode----------------------------
Searching for path: '/root'
        Inode path: '/root'
-----------------------------------------------------------------
b_open: Opened file '/root' with fd 0
-----------------------------------------------------------------
----------------------------getInode----------------------------
Searching for path: '/root'
        Inode path: '/root'
-----------------------------------------------------------------
--------------------------mfs_readdir---------------------------

-------------------------mfs_closedir---------------------------
-----------------------------------------------------------------
End dispatch
Prompt >
```

*Testing pwd command*



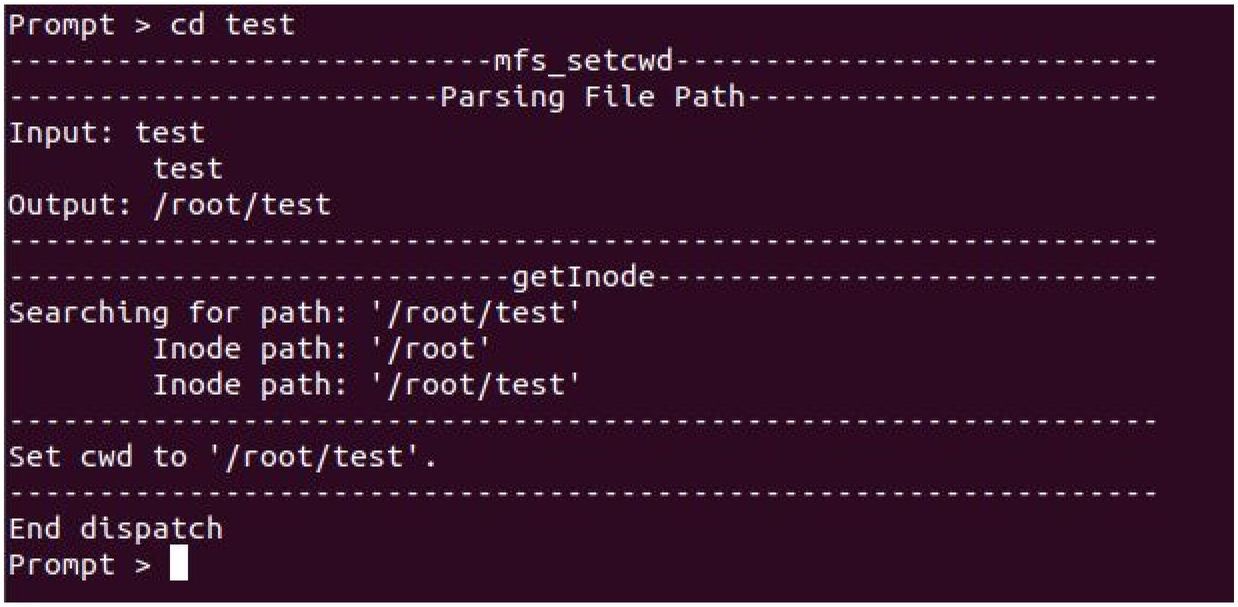
```
Prompt > pwd
---------------------------mfs_getcwd---------------------------
-----------------------------------------------------------------
/root
End dispatch
Prompt >
```

*Testing md command with name "test"*



```
Prompt > md test
----------------------------mfs_mkdir---------------------------
------------------------Parsing File Path-----------------------
Input: test
        test
Output: /root/test
-----------------------------------------------------------------
----------------------------getInode----------------------------
Searching for path: '/root'
        Inode path: '/root'
-----------------------------------------------------------------
--------------------------createInode---------------------------
----------------------checkValidityOfPath-----------------------
-----------------------------------------------------------------
--------------------------getFreeInode--------------------------
-----------------------------------------------------------------
-------------------------getParentPath--------------------------
------------------------Parsing File Path-----------------------
Input: test
        test
Output: /root/test
-----------------------------------------------------------------
Input: test, Parent Path: /root
-----------------------------------------------------------------
----------------------------getInode----------------------------
Searching for path: '/root'
        Inode path: '/root'
-----------------------------------------------------------------
----------------------------setParent---------------------------
Set parent of '/root/test' to '/root'
```

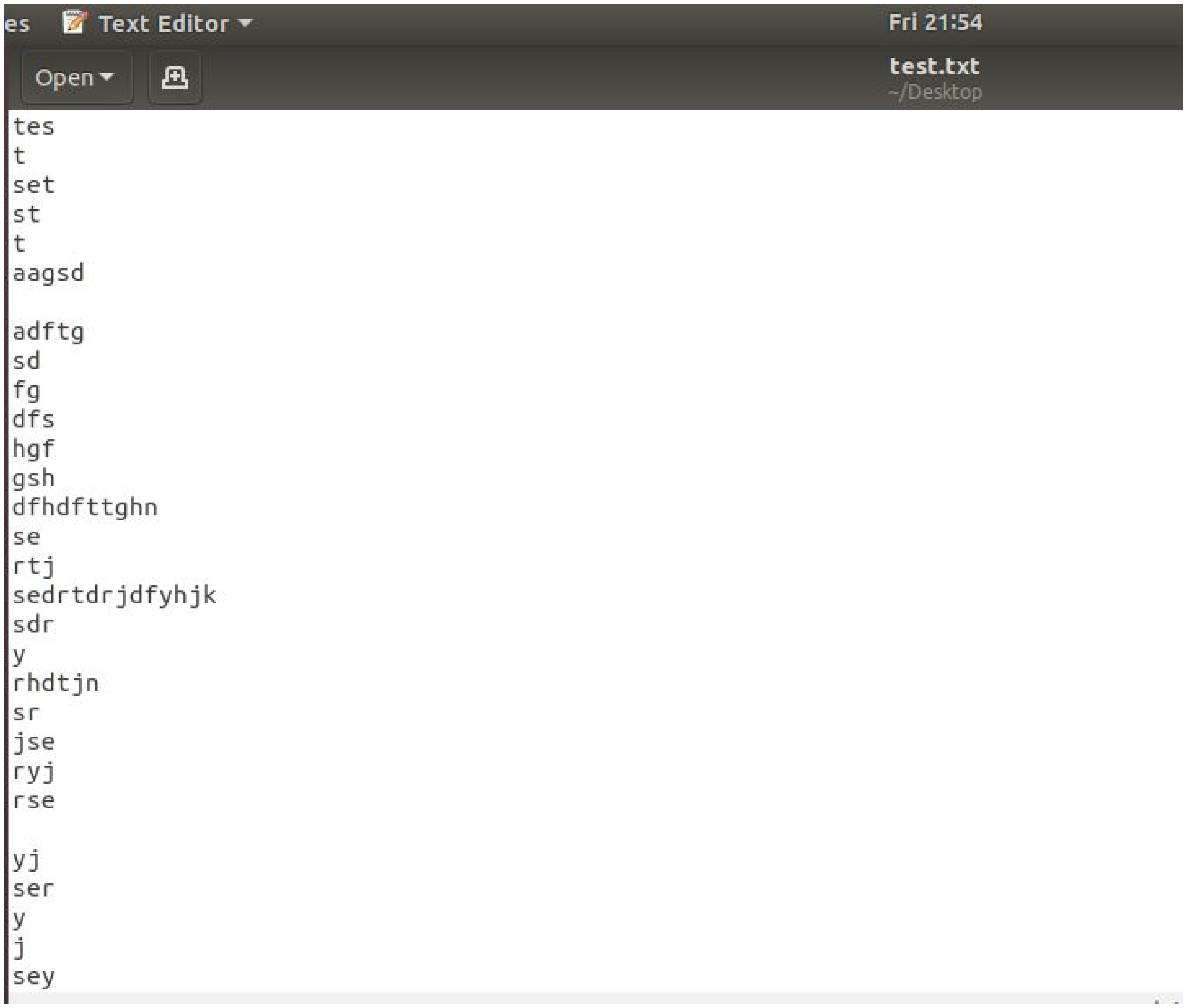
*Testing cd command to "test" directory*



```
Prompt > cd test
----------------------------mfs_setcwd--------------------------
------------------------Parsing File Path-----------------------
Input: test
        test
Output: /root/test
-----------------------------------------------------------------
----------------------------getInode----------------------------
Searching for path: '/root/test'
        Inode path: '/root'
        Inode path: '/root/test'
-----------------------------------------------------------------
Set cwd to '/root/test'.
-----------------------------------------------------------------
End dispatch
Prompt >
```

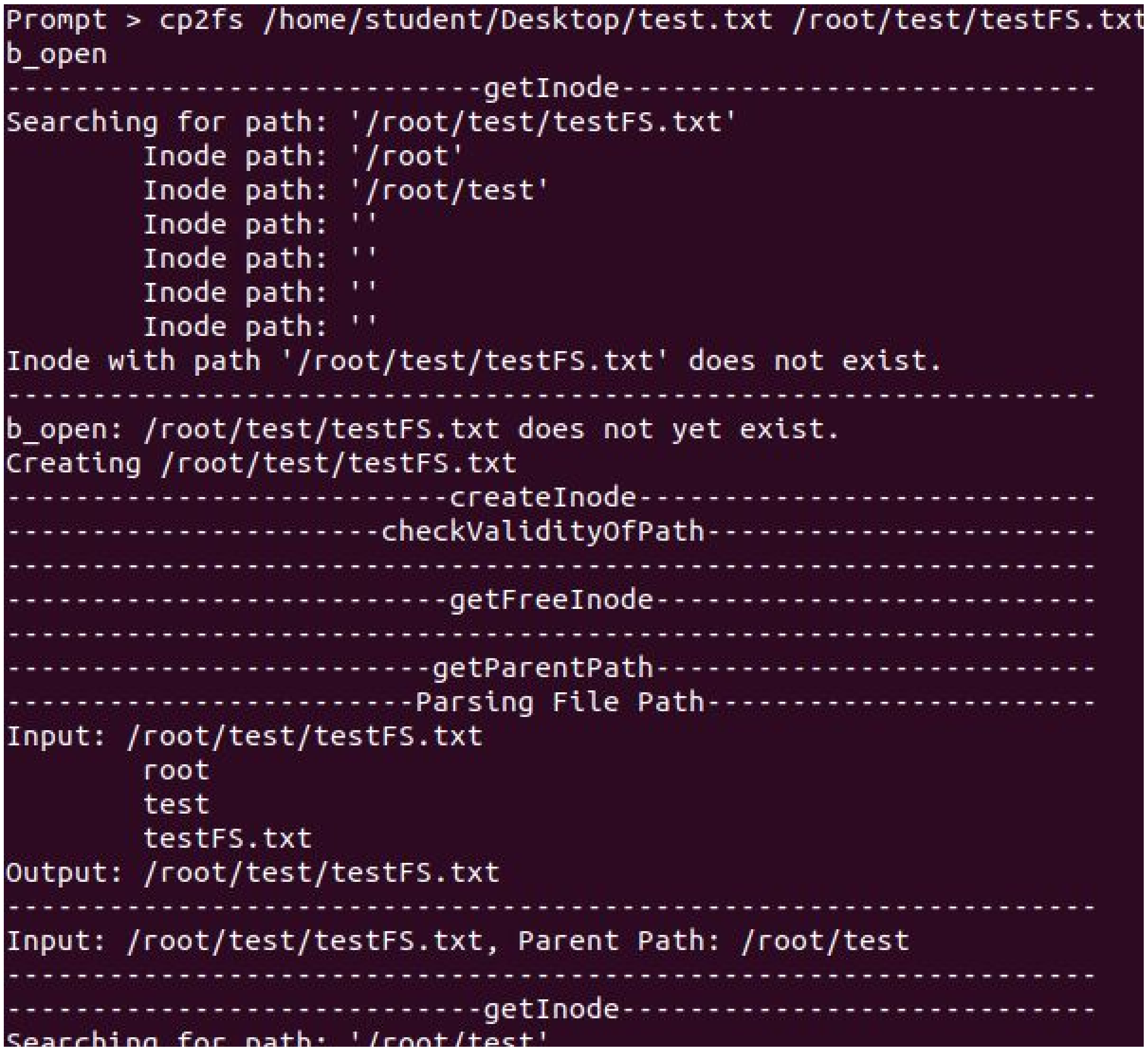
For testing the cp2l and cp2fs, using test.txt to copy from Linux to our FS, and copying from our FS to Linux.

*Contents of test.txt created in Linux used for output*



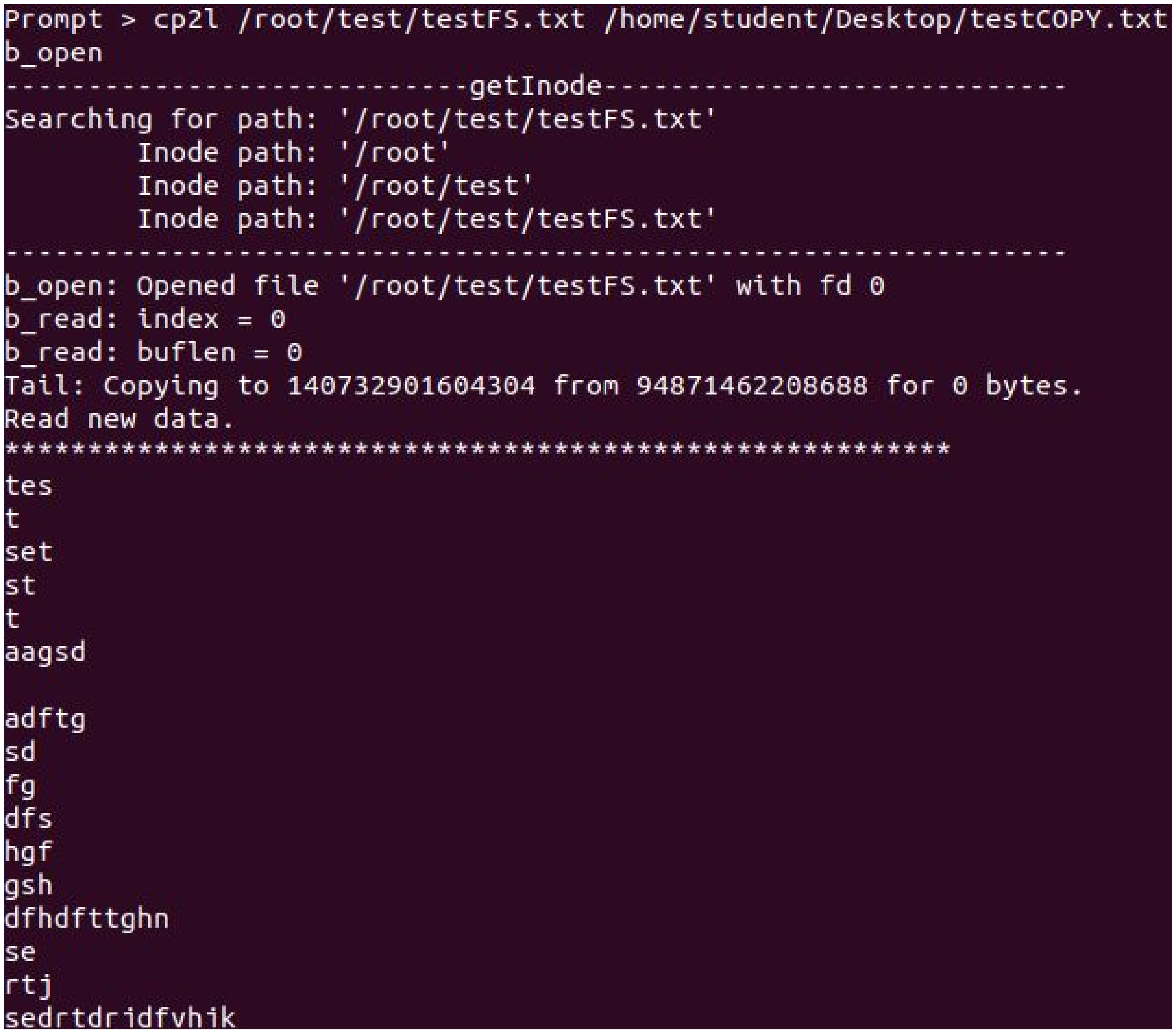
```
tes
t
set
st
t
aagsd

adftg
sd
fg
dfs
hgf
gsh
dfhdfttghn
se
rtj
sedrtdrjdfyhjk
sdr
y
rhdtjn
sr
jse
ryj
rse

yj
ser
y
j
sey
```

*Output of cp2fs using test.txt from Linux*



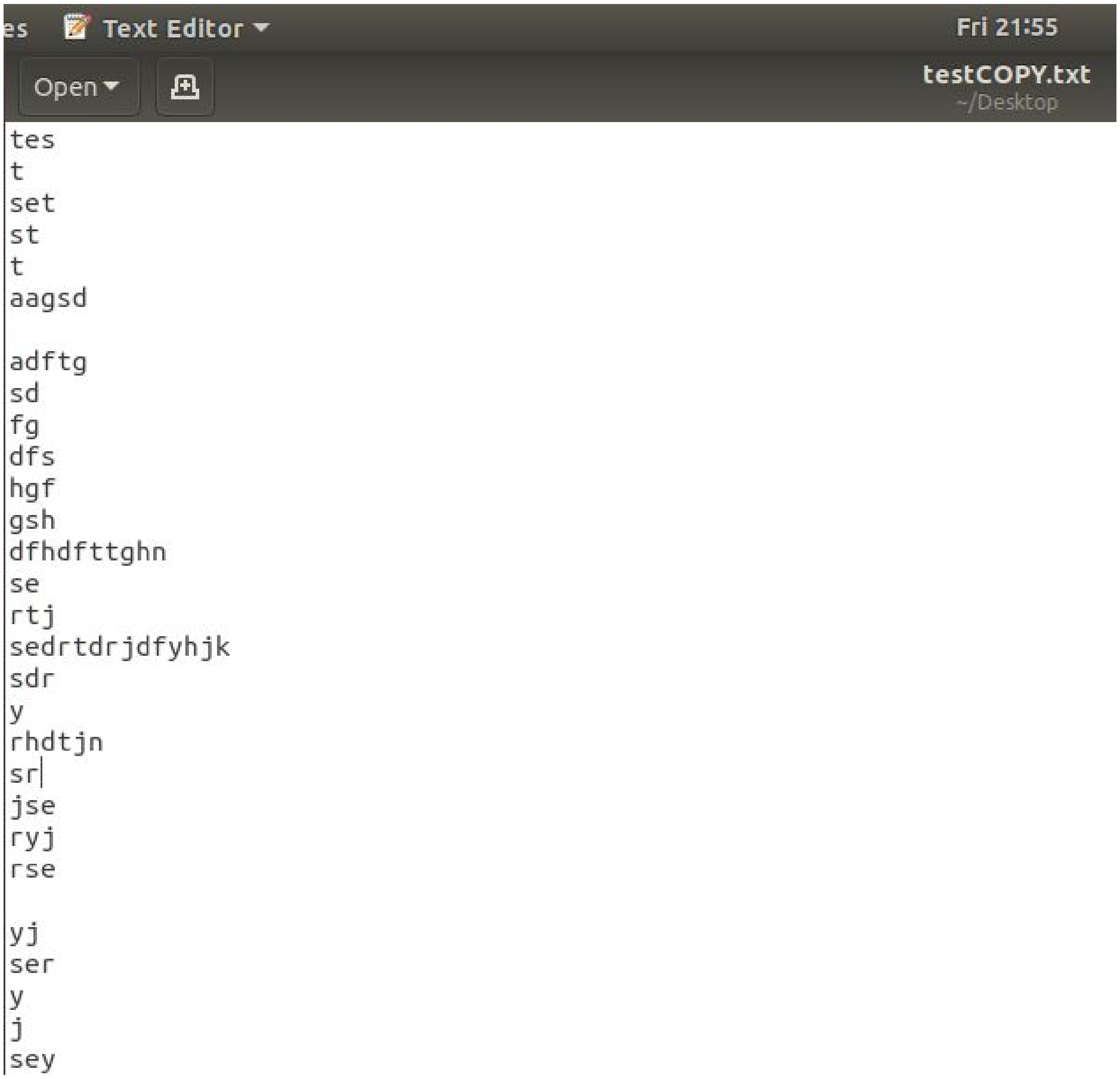
```
Prompt > cp2fs /home/student/Desktop/test.txt /root/test/testFS.txt
b_open
----------------------------getInode----------------------------
Searching for path: '/root/test/testFS.txt'
        Inode path: '/root'
        Inode path: '/root/test'
        Inode path: ''
        Inode path: ''
        Inode path: ''
        Inode path: ''
Inode with path '/root/test/testFS.txt' does not exist.
----------------------------------------------------------------
b_open: /root/test/testFS.txt does not yet exist.
Creating /root/test/testFS.txt
--------------------------createInode---------------------------
----------------------checkValidityOfPath----------------------
----------------------------------------------------------------
--------------------------getFreeInode--------------------------
----------------------------------------------------------------
-------------------------getParentPath--------------------------
------------------------Parsing File Path-----------------------
Input: /root/test/testFS.txt
        root
        test
        testFS.txt
Output: /root/test/testFS.txt
----------------------------------------------------------------
Input: /root/test/testFS.txt, Parent Path: /root/test
----------------------------------------------------------------
----------------------------getInode----------------------------
Searching for path: '/root/test'
```

*Output of cp2l using testFS.txt in our file system*



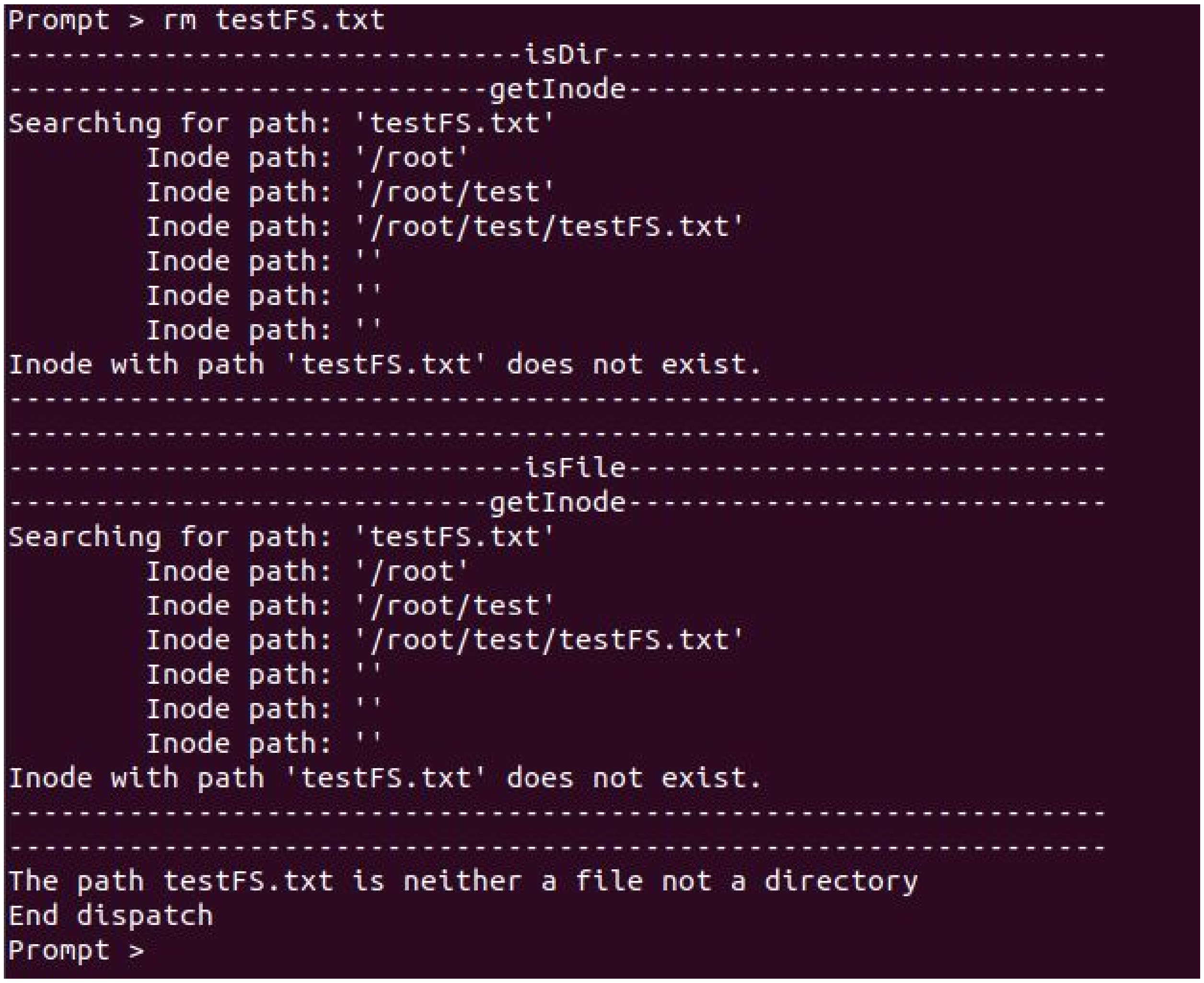
```
Prompt > cp2l /root/test/testFS.txt /home/student/Desktop/testCOPY.txt
b_open
----------------------------getInode----------------------------
Searching for path: '/root/test/testFS.txt'
        Inode path: '/root'
        Inode path: '/root/test'
        Inode path: '/root/test/testFS.txt'
----------------------------------------------------------------
b_open: Opened file '/root/test/testFS.txt' with fd 0
b_read: index = 0
b_read: buflen = 0
Tail: Copying to 140732901604304 from 94871462208688 for 0 bytes.
Read new data.
*********************************************************
tes
t
set
st
t
aagsd

adftg
sd
fg
dfs
hgf
gsh
dfhdfttghn
se
rtj
sedrtdridfvhik
```

*Contents of testCOPY.txt in Linux after using command to copy*



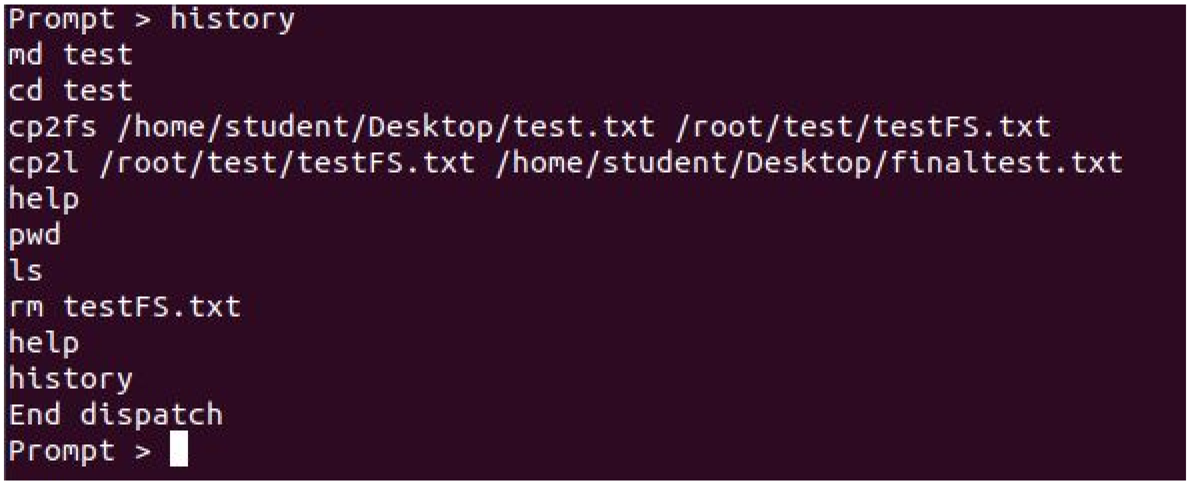
```
tes
t
set
st
t
aagsd

adftg
sd
fg
dfs
hgf
gsh
dfhdfttghn
se
rtj
sedrtdrjdfyhjk
sdr
y
rhdtjn
sr
jse
ryj
rse

yj
ser
y
j
sey
```

*Removing file testFS.txt with rm*



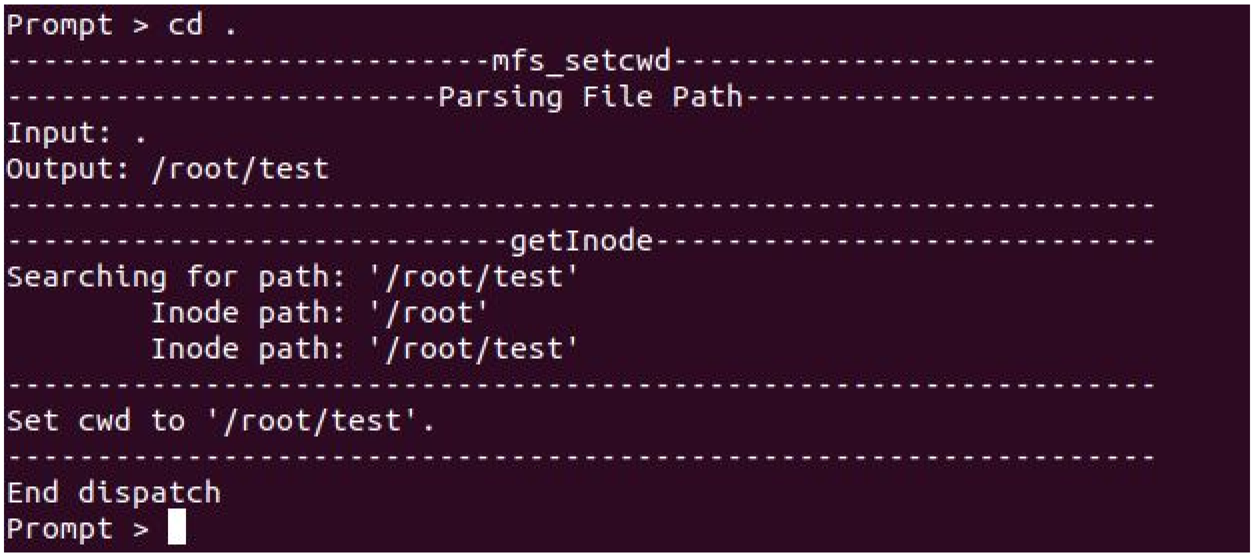
```
Prompt > rm testFS.txt
-----------------------------isDir-----------------------------
---------------------------getInode---------------------------
Searching for path: 'testFS.txt'
        Inode path: '/root'
        Inode path: '/root/test'
        Inode path: '/root/test/testFS.txt'
        Inode path: ''
        Inode path: ''
        Inode path: ''
Inode with path 'testFS.txt' does not exist.
--------------------------------------------------------------
--------------------------------------------------------------
-----------------------------isFile----------------------------
---------------------------getInode---------------------------
Searching for path: 'testFS.txt'
        Inode path: '/root'
        Inode path: '/root/test'
        Inode path: '/root/test/testFS.txt'
        Inode path: ''
        Inode path: ''
        Inode path: ''
Inode with path 'testFS.txt' does not exist.
--------------------------------------------------------------
--------------------------------------------------------------
The path testFS.txt is neither a file not a directory
End dispatch
Prompt >
```

*Testing history command*



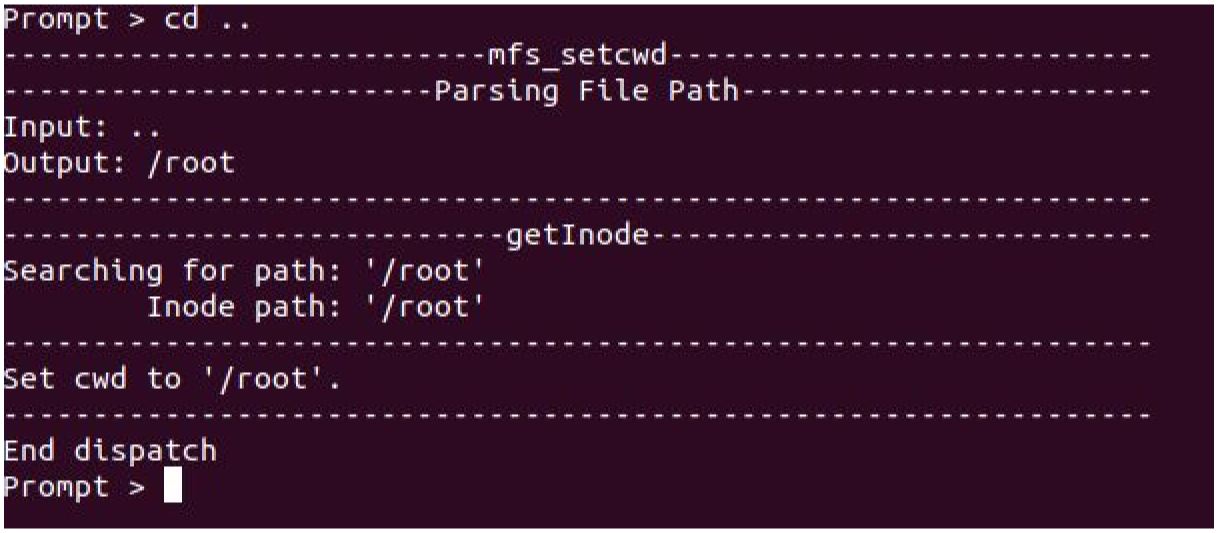
```
Prompt > history
md test
cd test
cp2fs /home/student/Desktop/test.txt /root/test/testFS.txt
cp2l /root/test/testFS.txt /home/student/Desktop/finaltest.txt
help
pwd
ls
rm testFS.txt
help
history
End dispatch
Prompt >
```

*Testing cd with . as input from test directory*



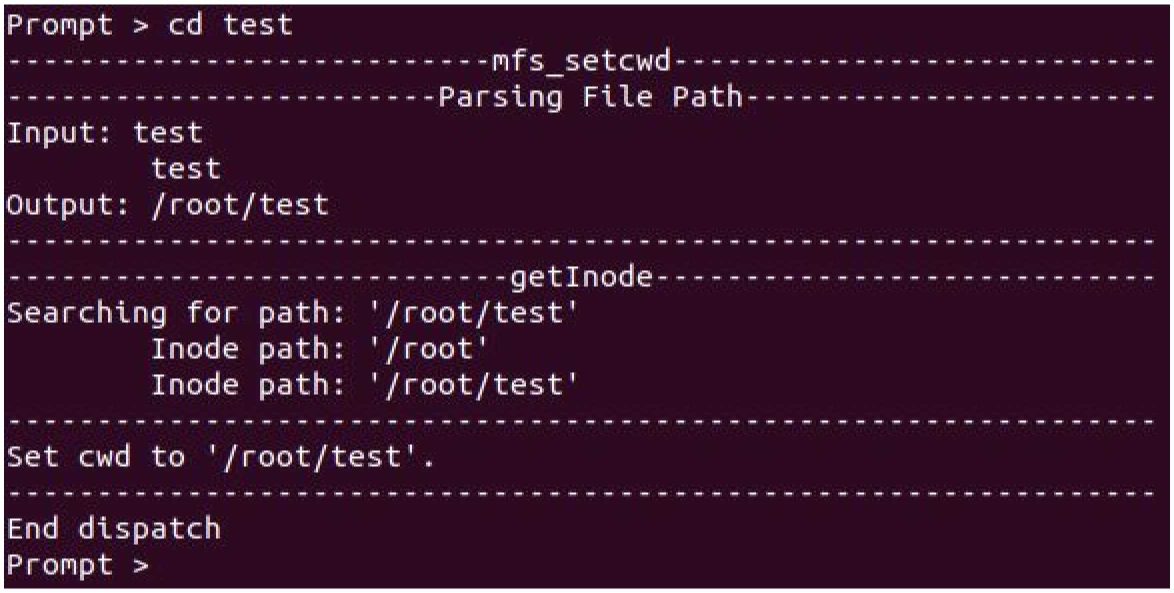
```
Prompt > cd .
---------------------------mfs_setcwd---------------------------
------------------------Parsing File Path-----------------------
Input: .
Output: /root/test
----------------------------------------------------------------
----------------------------getInode----------------------------
Searching for path: '/root/test'
        Inode path: '/root'
        Inode path: '/root/test'
----------------------------------------------------------------
Set cwd to '/root/test'.
----------------------------------------------------------------
End dispatch
Prompt >
```

*Testing cd with .. as input from test directory*



```
Prompt > cd ..
---------------------------mfs_setcwd---------------------------
------------------------Parsing File Path-----------------------
Input: ..
Output: /root
----------------------------------------------------------------
----------------------------getInode----------------------------
Searching for path: '/root'
        Inode path: '/root'
----------------------------------------------------------------
Set cwd to '/root'.
----------------------------------------------------------------
End dispatch
Prompt >
```

*Testing cd to test directory*



```
Prompt > cd test
---------------------------mfs_setcwd---------------------------
------------------------Parsing File Path-----------------------
Input: test
        test
Output: /root/test
----------------------------------------------------------------
----------------------------getInode----------------------------
Searching for path: '/root/test'
        Inode path: '/root'
        Inode path: '/root/test'
----------------------------------------------------------------
Set cwd to '/root/test'.
----------------------------------------------------------------
End dispatch
Prompt >
```