COP4600 Programming Assignment 3

due Thursday, November 7th by midnight

1 Problem

In this assignment you are going to implement hard and symbolic links for a file system simulator that uses inodes.

The file system simulator already supports the following commands:

```
$ simulator
Type help to find out the available commands
> help
Command                   Description
=======================   ================================================
help                      Displays commands and their description
pwd                       Print working directory
touch filename            Create empty file filename in the current working directory
cp filename1 filename2    Copy filename1 in the real current working directory to filename2
more filename             Display contents of file filename
rm filename               Delete file filename
stat filename             Shows number of blocks and number of links for file filename
mkdir dirname             Create directory dirname
ls                        List files in the current working directory
cd dirname                Changes to subdirectory dirname
up                        Changes to parent directory
exit                      Terminates the program

>
...
```

Specifically, your task is to implement hard links and symbolic links and provide 3 more commands:

- **ln source_file_name hardlink_name:** Create a hard link (hardlink_name) for source file (source_file_name). When a hard link is created only an entry is added to the current working directory's directory block: (hardlink_name, inode no of source_file_name) and the link count in the inode of source_file_name is incremented.
- **ln -s source_file_name symbolic_link_name:** Create a symbolic link. When a symbolic link is created a new inode is created and a new entry, (symbolic_link_name,no of the new inode) is added to the current working directory's directory block. Also, new data block(s) are created to hold

name of the source file (source_file_name) and these data block addresses are added to the new inode for the symbolic link.

- **stat -L file_name:** If file_name is a symbolic link then follow the link and print the source's status. Otherwise print file_name's status.

2 Guideline

2.1 Commands That Need To Be Reimplemented

You should change implementation of the following commands by taking links into consideration:

- **rm filename:** decrements the link count in the i-node of regular file filename. As before no effect on directories. Deletes the file if the link count becomes 0.
- **more filename:** If filename is a link displays the source file. If the source file does not exist (possible with symbolic links), displays an appropriate message and waits for the next command.
- **cp filename1 filename2:** If filename2 is a link then filename1 should be copied onto the source file by overwriting the original content of the source file.
- **cd dirname:** If dirname is a symbolic link that has been linked to a directory then change directory to the source directory.

Note that the behavior of the above commands should not change if the parameter does not represent a hard link or a symbolic link. Also, to keep things simple, file name parameters for the commands are assumed to be immediate file names (no full path names), which refer to a file in the current working directory.

2.2 A Test Case

A sample session using the File System Simulator is given below. Figure 1 shows the directory hierarchy after all the directories, files, and the links are created. The same figure also shows you how the directories are implemented in the simulator.

```
lin442-01:74% java FileSystem
Type help to find out the available commands
> help
Command                     Description
=====================       ================================================
help                        Displays commands and their description
pwd                         Print working directory
touch filename              Create empty file filename in the current working directory
```

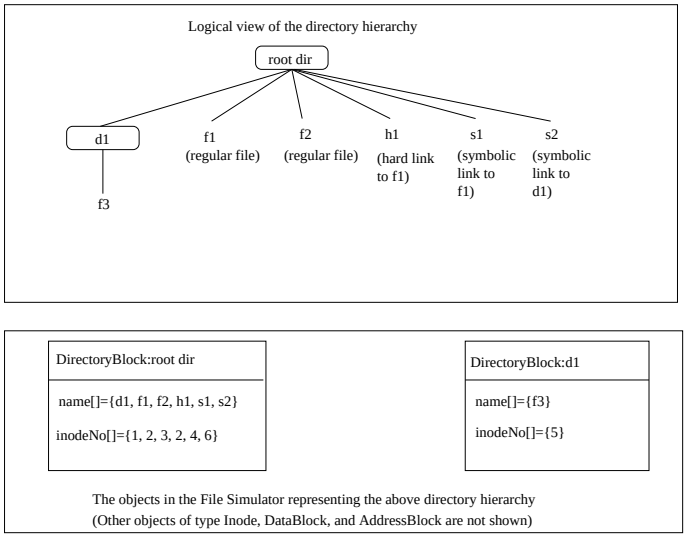

Figure 1: A snapshot of the directory hierarchy and some of the objects modeling the hierarchy in the simulator.

```
cp filename1 filename2      Copy filename1 in the real current working directory to filename2
more filename               Display contents of file filename
rm filename                 Delete file filename
stat (-L) filename          Shows number of blocks and number of links for file filename
mkdir dirname               Create firectory dirname
ls                          List files in the current working directory
cd dirname                  Changes to subdirectory dirname
up                          Changes to parent directory
ln (-s) filename1 filename2 Create a link to filename1
exit                        Terminates the program

> mkdir d1

> ls
d1

> touch f1

> ls
d1
f1

> cp temp1 f1

> ls
d1
f1

> more f1
hello
```

```
how are you?
such a cool project, etc.

> touch f2

> ls
d1
f1
f2

> more f2

> cp temp2 f2

> ls
d1
f1
f2

> more f2

file systems are cool

> ls
d1
f1
f2

> stat d1
Directory InodeNo: 1 # of data blocks 1  # of links: 1
> stat f1
Regular file InodeNo: 2 # of data blocks 3  # of links: 1
> stat f2
Regular file InodeNo: 3 # of data blocks 2  # of links: 1
> ln f1 h1

> ls
d1
f1
f2
h1

> stat h1
HardLink InodeNo: 2 # of data blocks 3  # of links: 2
> stat f1
Regular file InodeNo: 2 # of data blocks 3  # of links: 2
> more h1
hello
how are you?
such a cool project, etc.

> ln -s f1 s1

> ls
d1
f1
f2
```

```
h1
s1

> stat s1
SymLink InodeNo: 4 # of data blocks 1  # of links: 1
> stat -L s1
Regular file InodeNo: 2 # of data blocks 3  # of links: 2
> stat d1
Directory InodeNo: 1 # of data blocks 1  # of links: 1
> more s1
hello
how are you?
such a cool project, etc.

> cd d1

/d1> ls

/d1> touch f3

/d1> up

> ln -s d1 s2

> ls
d1
f1
f2
h1
s1
s2

> stat s2
SymLink InodeNo: 6 # of data blocks 1  # of links: 1
> stat -L s2
Directory InodeNo: 1 # of data blocks 1  # of links: 1
> cd s2

/s2> ls
f3

/s2> up

> more s2
s2 is a directory!

> ls
d1
f1
f2
h1
s1
s2

> rm f1

> ls
```

```
d1
f2
h1
s1
s2

> stat h1
HardLink InodeNo: 2 # of data blocks 3  # of links: 1
> more h1
hello
how are you?
such a cool project, etc.

> stat s1
SymLink InodeNo: 4 # of data blocks 1  # of links: 1
> more s1
File Not Found!
> exit
```

3 Resources

- The File System Simulator code (FileSystem.zip) with basic functionality is provided under Programming Assignment 3 on E-learning.
- Chapter 13 of the text book.

4 Submission Instructions

Please submit all the .java files in FileSystem.zip.