ASSIGNMENT 4

CPSC 1280

DR. BAGHERI

JUNE 12TH, 2014

RYAN WOODWARD

100201137

**3.1**

A directory file contains an entry for every file and subdirectory file that it houses. Each entry has the filename and the inode number associated with that file. The directory does not actually contain the file but rather just filename and the inode number. So when you use the commands cp, mv, and rm; the kernel is updating the directory by adding or removing the entry (filename and inode number) of the file. The directory file is quite small due to the fact that it only contains a max of 256 bytes for the filename and bytes for the inode number which are normally between 128 and 256 bytes.

**3.2**

The device file governs the operation of the external device that is connected to the computer. The file is necessary as it acts as the driver (software) that allows the computer to interact with the external device.

**3.3**

1. mkdir a/b/c
   1. This command will work since the directory but a/b has to exist in the first place in order for the subdirectory c to be created under b
2. mkdir a a/b
   1. This command works since it first creates directory a, and then creates the subdirectory b within a
3. rmdir a/b/c
   1. This command will work given that the c directory has been created and is empty
4. rmdir a a/b
   1. This command will not work as it will attempt to remove the directory a first. Since a is not empty because it contains b, the command will fail
5. mkdir /bin/foo
   1. This command will not work as the /bin directory does not have the write permission enabled. Therefore we cannot add a file to that directory

**3.5**

We cannot use . and .. as a filename since they are reserved for directories. However we are able to create a file name using … This file will not show up with the regular ls command however since it has a leading dot. Therefore it is a hidden file.

**3.6**

The rmdir bar command failed listing the fact that the directory was not empty even though ls displayed no files when the command was executed. This is because there are hidden files which the simple command ls would not display.

**3.7**

1. mv bar1 bar2
   1. When bar2 exists, bar1 is moved into bar2 and is now a subdirectory
   2. When bar2 doesn’t exist this is simply a mechanism to rename a file. bar1 has been effectively renamed bar2

**3.8**

The command cd ~Charlie wants to change into the directory with the name ~Charlie. The command cd ~/Charlie wants to go into the home directory and then change into the subdirectory Charlie. Both commands are capable of working provided the directories exist.

**3.9**

Charlie could have avoided this problem by using a relative pathname in his scripts. Therefore he could’ve typed ~/charlie/html that way if the home directory changes from /usr to /home the pathname of the ~ would change as well. Therefore no errors would occur

**3.12**

If ls bar and ls –d bar both present the same output it could either be bar or /bar. This would indicate that bar is an ordinary file or a directory file.

**3.15**

Even if all the files exist a cp command can fail

1. if the directory /backup does not have the write permission then we aren’t able to add a file
2. although backup/hosts.bak exists you could be in a different directory when issuing this command, therefore it won’t see that directory
3. If you were to look for the directory you would have to provide /backup/hosts.bak, removing the slash confuses the command interpreter as it will think backup/host.bak is an ordinary file.