UNIT 6: FILE ORGANIZATION

UNIT STRUCTURE

6.1 Learning Objectives

6.2 Introduction

6.3 Files and File System

6.4 Working with Files

6.5 File System Mounting and Unmounting,

6.6 Organization of the File Tree

6.7 Disk Related Commands

6.8 Let Us Sum Up

6.9 Further Readings

6.10 Answer to Check Your Progress

6.11 Model Questions

6.1 LEARNING OBJECTIVES

After going through this unit, you will be able to:

- to distinguish different files system,
- to cope up with different commands to work on file,
- understand basic file organization of Linux,
- to learn different disk related commands.

6.2 INTRODUCTION

In the previous unit, we have discussed the Linux File System and the related properties. The functions of different file systems in Linux are explained. We have also discussed the different types of files, I-node and the file attributes. In this unit, we will discuss about the file organisation. File is collection of information, and a file system is used to control how data is stored and retrieved. Without a file system, data placed in a storage medium would be a random assortment of some facts only, without having any specific beginning or ending for a particular data. Unix system treat everything inside it as files. Hence it has thousands of files. Unix file attributes

like name, size and access permission are stored separately rather than putting in the file like in Windows system.

6.3 FILES AND FILE SYSTEM

In general, we can categorize files into 3 categories:

- **Ordinary files:** These are regular files which contain stream of characters as data. This can be further subdivided it into Text file and Binary File
- **Directory files:** Unix system treat directories are as files. A directory file contains an entry for every file and subdirectory it has. Each entry consists of two entry the file and unique identification number (inode number) for the file or directory
- **Device files:** In Unix all devices and peripherals are also treat as files. To perform any operation on devices we must invoke associated with files. Device fil name are generally store under single directory structure called, /dev.

A filename in Unix can consist of up to 255 characters. Filename can be any ASCII character except the / and NULL character. But we should avoid using control characters in naming files.

All files in UNIX are related to one another and organized in hierarchical inverted tree structure. Every UNIX file system has a reference point at top of the tree structure which is called root and represented as / (forward slash). Under root we have a directory called home directory. Whenever we log on to UNIX system it automatically places us at home directory. for example, if you log into your system and open Unix Terminal (ctrl+Alt+T for Ubuntu users) and type pwd after the dollar symbol you will get a result like this:

/home/host_name

Here in the above, first slash "/" is synonym for root directory, inside the root directory we have home directory inside it we have our own directory here directory name is Ubuntu, Home directory and ubuntu directory is separated by the delimiter "/". The pwd (print working directory) command shows you the current working directory.



Fig 6.1: Snapshot of pwd command

We can use ls (listing command) to list out all the content present in the current directory. It will show a complete list of filenames in the current directory in ASCII collating sequence (number first, uppercase then lower case). It shows directories also.



Fig 6.2: Snapshot of ls command

At above picture it is showing contents of ubuntu (user directory) where all are directories except test0.txt which is text file

ls Options

ls command has different options which syntax is like ls -Options. We will present some important them.

Type ls -F in a terminal window to add "/" at the end of each directory. This command will help you to distinguish directories from files. See image below for further details:

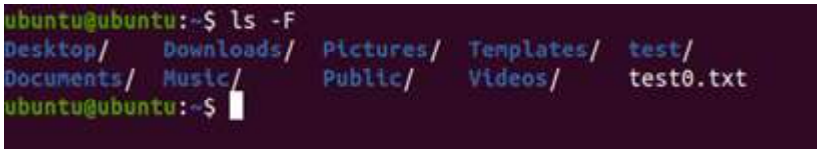


Fig 6.2: Snapshot of ls-F command

When using the basic ls command, you can't see hidden files and files starting with ".". But the ls -a command to display them as seen in the image below:



Fig 6.3: Snapshot of ls -a command

The get information about the user, size of the file, and date and time of modification. We can use ls -l command. In the picture bellow.

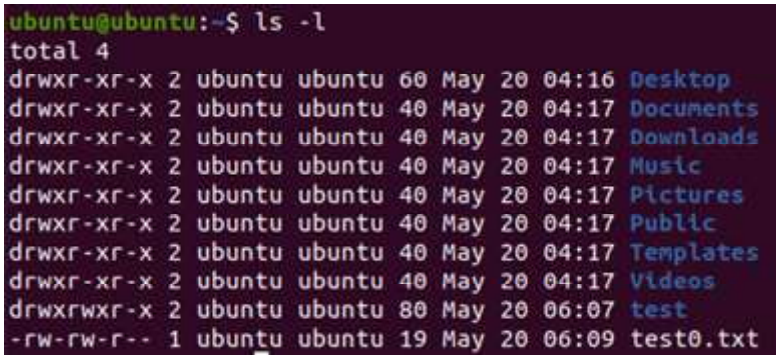


Fig 6.4: Snapshot of ls-l command

Let's us discuss this output little bit more elaborately. If we take the last line of the output, it looks like this:

-rw-rw-r-1 ubuntu ubuntu 19 May 20 06:09 test0.txt

1st Character - File Type: First character specifies the type of the file. In the example above the hyphen (-) in the 1st character indicates that this is a normal file. Following are the possible file type options in the 1st character of the ls -l output.

Field Explanation

- - normal file
- d : directory
- s : socket file
- l : link file
- **Field 1 - File Permissions:** Next 9 character specifies the files permission. The every 3 characters specifies read, write, execute permissions for user(root), group and others respectively in order. Taking above example, -rw-rw-r- indicates read-write permission for user(root) , read permission for group, and no permission for others respectively. If all three permissions are given to user(root), group and others, the format looks like -rwxrwxrwx
- **Field 2 - Number of links:** Second field specifies the number of links for that file. In this example, 1 indicates only one link to this file.
- **Field 3 - Owner:** Third field specifies owner of the file. In this example, this file is owned by username 'ubuntu'.
- **Field 4 - Group:** Fourth field specifies the group of the file. In this example, this file belongs to" ubuntu' group.

- **Field 5 - Size:** Fifth field specifies the size of file in bytes. In this example, '19' indicates the file size in bytes.
- **Field 6 - Last modified date and time:** Sixth field specifies the date and time of the last modification of the file. In this example, 'May 20 06:09' specifies the last modification time of the file.
- **Field 7 - File name:** The last field is the name of the file. In this example, the file name is test0.txt



CHECK YOUR PROGRESS

Q 1: What are the 3 types of file present in Linux?

.....

Q 2: Why ls command is used?

.....

6.4 WORKING WITH FILES

In this section we will try to explore different operations like creating, reading, searching, copying, moving, renaming, deleting, linking, and editing Files. First, we will go through operations of directory then we will go files as directory operations are little bit different.

Changing Directories

The pwd (print working directory) shows current directory we are in, and cd (change directory) is used switch your working directory from one to another. For more illustration look at the picture bellow. As per pwd command shows we are at first home directory "ubuntu", now we can use: cd pathname to move to different directory. Here we use cd test to move to directory test. We can use cd .. command to move out of the directory. After using this command, we can we have again come to "ubuntu" directory. Actually, the "." (dot) symbol refers to the working directory itself and ".." the (dot dot) symbol refers to the parent directory of the working directory

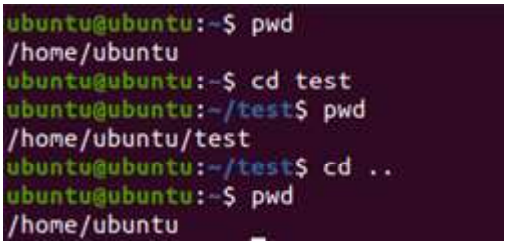


Fig 6.4: Snapshot of cd command

Creating and Deleting Directories

In order to create a directory, we use the mkdir (make directory) command, syntax for this command is mkdir directory_name. It will create a new directory in current directory. To remove a directory, we use the rmdir command. Remember that the rmdir command can only be used to remove empty directories. We can use rm (remove) command to delete directory, but it can also delete empty directories but with -rf option i.e.rm -rf directory_name We can force to delete the directory.



Fig 6.5: Snapshot of mkdir & rm -rf command

File Operations

we can create delete edit files using Linux GUI itself as like in windows. There is a general-purpose text editor for genome desktop called "gedit "with which we can create, edit and delete files. The general syntax is:

$gedit filename

For more details we can look into Linux manual i.e., $man gedit. Next, we have discussed some other means of file operations. All the operations have lots of different options, which can be used to do extra functionalities of the commands. But here we have only use straight forward utility of the commands only. We urged users to explore further possibilities with these commands

The touch command can be used to create a new file. It will create and open a new blank file if the file with a filename does not exist. And in

case the file already exists then the file will not be affected.

$touch filename

```
ubuntu@ubuntu:~$ ls
Desktop    Downloads  Pictures  Templates  test
Documents  Music      Public    Videos     test0.txt
ubuntu@ubuntu:~$ touch new_file
ubuntu@ubuntu:~$ ls
Desktop    Downloads  Pictures  Templates  new_file  test0.txt
Documents  Music      Public    Videos     test
```

Fig 6.6: Snapshot of touch command

In above picture with ls command, we can see there is no "new_file" in directory, but with touch command we have created the file.

cat command can be used to display the contents of a file. This command will display the contents of the 'filename' file. And if the output is very large then we could use more or less to fit the output on the terminal screen otherwise the content of the whole file is displayed at once.

Syntax for cat command is:

$cat [Option] filename

```
ubuntu@ubuntu:~$ cat new_file
this is the content of new_file .
```

Fig 6.7: Use of cat command

The cat is a very powerful tool user can look for its more uses. For example:

$cat > filename creates a new file

$cat filename1 filename2>filename3 joins two files (1 and 2) and stores the output of them in a new file (3)

Copying a File

cp command could be used to create the copy of a file. It will create the new file in destination with the same name and content as that of the file 'filename'. It remains in the same working directory. General syntax for cp is:

$cp [Option] source/filename destination/filename

First image is copy without path name and second one is copy with a pathname.

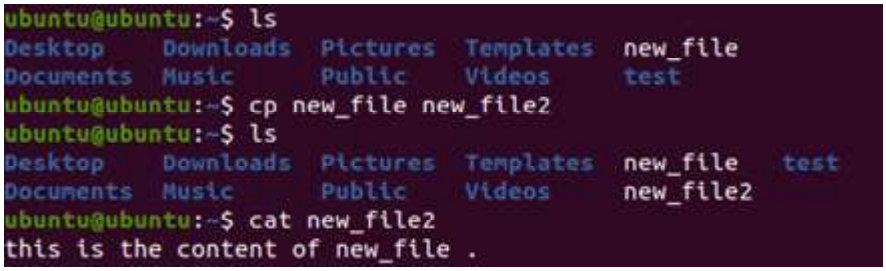


Fig 6.8: Snapshot of copy without path name

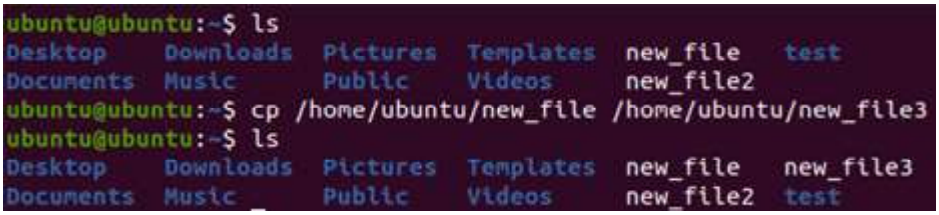


Fig 6.9: Snapshot of copy with path name

mv command could be used to move a file from source to destination. It will remove the file filename from the source folder and would be creating a file with the same name and content in the destination folder.

$mv [Option] source/filename destination/

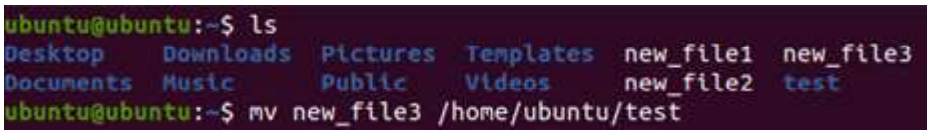


Fig 6.10: Snapshot of moving a file

In this screen shot we have move "new_file3" to "test" directory

rm command could be used to delete a file. It will remove the filename file from the directory. The general syntax for it is

$rm [Option] filename

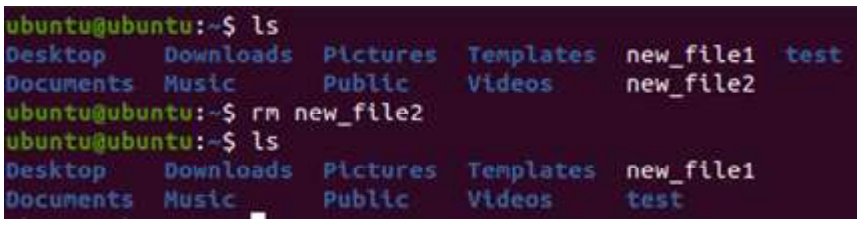


Fig 6.11: Snapshot of removing a file

The find command in UNIX is a command line utility for walking a file hierarchy. It can be used to find files and directories and perform subsequent operations on them. It supports searching by file, folder, name, creation date, modification date, owner and permissions. By using the '-

exec' other UNIX commands can be executed on files or folders found.

$ find /directory where to search [expression determines what to find] [-options] [what to find]



Fig 6.12: Snapshot of find command

Here we search new_file3 from the test directory using name option in find command.

Absolute Path vs Relative Path

A path is a unique location to a file or a folder in a file system of an OS. A path to a file is a combination of / and alpha-numeric characters. An absolute path is defined as the specifying the location of a file or directory from the root directory(/). In other words we can say absolute path is a complete path from start of actual filesystem from / directory. For example

/home/ubuntu/test

/home//boot/grub/grub.conf

all these paths started from / directory which is a root directory for every Linux/Unix machines

Relative path is defined as path related to the present working directory(pwd). Suppose at the moment we are at /home/ubuntu, we want to change it to /home/ubuntu/test, then we can directly change it t test using cd test command. If you observe there is no / before test which indicates it is a relative directory to present working directory. We can use an absolute path from any location where as if you want to use relative path we should be present in a directory where we are going to specify relative to that present working directory.

6.5 FILE SYSTEM MOUNTING AND UNMOUNTING

All files in a Linux filesystem are arranged in form of a big tree rooted at '/'.We will discussed this organization mor thoroughly later in this chapter .These files can be spread out on various devices based on partition table, initially parent directory is mounted(i.e attached) to this tree at '/',

others can be mounted manually using GUI interface(if available) or using mount command. The mount command is used to mount the filesystem found on a device to big tree structure (Linux filesystem) rooted at '/'. Conversely, another command umount can be used to detach these devices from the Tree.

Syntax:

mount -t type device dir

Other forms:

mount [-l|-h|-V]

mount -a [-fFnrsvw] [-t fstype] [-O optlist]

mount [-fnrsvw] [-o options] device|dir

mount [-fnrsvw] [-t fstype] [-o options] device dir

All these commands tell the Kernel to attach the filesystem found at device to the dir.

Note:

- If you leave the dir part of syntax it looks for a mount point in /etc/fstab.
- You can use -source or -target to avoid ambivalent interpretation. mount --target /mountpoint
- /etc/fstab usually contains information about which device needs to be mounted where.
- Most of the devices are indicated by files like /dev/sda4, etc. But it can be different for certain filesystems. For more information explore man mount

Some Important Options:

- l : Lists all the file systems mounted yet.
- h : Displays options for command.
- V : Displays the version information.
- a : Mounts all devices described at /etc/fstab.
- t : Type of filesystem device uses.
- T : Describes an alternative fstab file.
- r : Read-only mode mounted.

Example: **mount dev/sd4 /home/ubuntu/test**

Here we have mounted device/sd4 file system to test directory.

6.6 ORGANIZATION OF THE FILE TREE

The Linux File Hierarchy Structure or the Filesystem Hierarchy Standard (FHS) defines the directory structure and directory contents in Unix-like operating systems. It is maintained by the Linux Foundation. In the FHS, all files and directories appear under the root directory /, even if they are stored on different physical or virtual devices. So it looks like a tree organization where root '/'is the top directory it contain first level subdirectories and each first level sub directories contain second level subdirectories, hence the tree structure propagate. The picture below depicts how directories are organized in tree structure

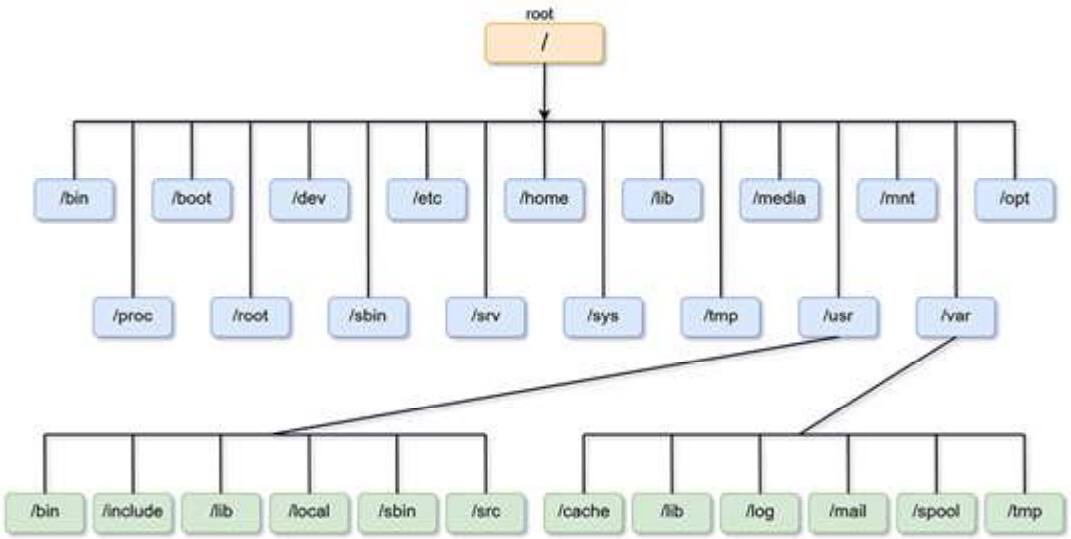


Fig 6.13: Linux File Structure Tree Organization

/ (Root): Primary hierarchy root and root directory of the entire file system hierarchy. Every single file and directory start from the root directory. The only root user has the right to write under this directory. the /root is the root user's home directory, which is not the same as /. Under root we have following subdirectories

- bin - This directory contains the binary or executable files that launch applications, all users have access to these applications. These represent the bare minimum set of programs required for a user to use the system.
- boot - This directory contains the files necessary to boot up your

computer including the boot loader and kernel activation files.

- dev - This directory contains the file pointers for character and block devices, which means the pointers to hard drives, flash drives, multimedia cards, external devices and system ports.
- etc - this directory contains the system configuration files and scripts.
- home - This directory contains the user profiles, desktops and user files, it functionality is similar to the "Documents and Settings" directory in MS-Windows.
- lib - This directory contains library files that are necessary for all programs to work.
- media - This directory is used by hal, the Linux auto-mounter, to load external devices for navigation such as floppy disks, cd-roms, dvd-roms, flash drives, etc…
- mnt - This directory is used to load external devices on system that do not use hal.
- opt - This directory contains optional software packages. Usually the contents of X11 and your window manger and loaded into opt, but you can add whatever programs you wish to this directory.
- proc - This directory is unique because it is not necessarily part of the file system, but a virtual file system that provides access to kernel or system information.
- root - This is the home directory for your root or system administrator. It is kept separate from the other users in case the partition that home is mounted on fails.
- sbin - This directory has a functionality similar to bin, which means it contains application executables, but rather than allowing all users access to these programs it restricts access only to the root or administrator.
- tmp - This is the temporary storage location. All users have read and write rights to the contents of this directory.
- usr - This is the largest directory on a Linux system. Pretty much everything that does not belong in the other directories is placed here. The contents usually include program, documentation and the kernel

source code.

- var - This directory contains most of the frequently changing files such as logging files, cache file and record locks.

If "tree" utility is installed is installed at your machine, we can check tree organization of file system inn UNIX/LINUX systems. It is a recursive directory listing program that produces a depth-indented listing of files. With no arguments, tree lists the files in the current directory. When directory arguments are given, tree lists all the files or directories found in the given directories each in turn. Upon completion of listing all files and directories found, tree returns the total number of files and directories listed. There are options to change the characters used in the output, and to use colour output. The most general way of using the tree command: $tree and the out will be like bellow.



Fig 6.14: Output of Tree Command



CHECK YOUR PROGRESS

Q 3: How to check tree structure of Linux?

.....

Q 4: What is the command for mounting file in Linux system

.....

6.7 DISK RELATED COMMANDS

There are several ways to list all the hard drives present in a system through Linux command lines. A hard drive could be physically connected, virtually connected, or even emulated. Here are some different commands which can list the hard drives, keep in mind there are others, but these are probably the most commonly used and easy to get the job done.

The du (Disk Usage) command shows the usage of disk. The du command will show you the disk usage for specific directories in Linux. The syntax for this command is :

$ du [optios] FILESYSTEM

When we use this command with no parameter, then this command will show you the information related to all the mounted file systems

$du



Fig 6.15: Snapshot of du command

Some other Functionality of du command

- **du -h:** - This will show you the disk usage in a human-readable format for each directory and subdirectory
- **du -a:** - This will show you the disk usage for all the files.
- **du -s:** - This will show you the total disk space used by a specific directory or file.

The df (Disk free) command will show you used and available disk spaces for in Linux system. The syntax for this command is:

$ df [options] FILESYSTEM

When we use this command with no parameter, then this command will show you the information related to all the mounted file systems:

$df

```
ubuntu@ubuntu:~$ df
Filesystem     1K-blocks    Used Available Use% Mounted on
udev              449040       0    449040   0% /dev
tmpfs             100232    1380     98852   2% /run
/dev/sr0         2999936 2999936         0 100% /cdrom
/dev/loop0       2109440 2109440         0 100% /rofs
/cow              501144  137112    364032  28% /
tmpfs             501144       0    501144   0% /dev/shm
tmpfs               5120       8      5112   1% /run/lock
tmpfs             501144       0    501144   0% /sys/fs/cgroup
tmpfs             501144       0    501144   0% /tmp
/dev/loop1         66688   66688         0 100% /snap/gtk-common-themes/1515
/dev/loop2         56832   56832         0 100% /snap/core18/2128
/dev/loop3         52224   52224         0 100% /snap/snap-store/547
/dev/loop4        224256  224256         0 100% /snap/gnome-3-34-1804/72
/dev/loop5         33152   33152         0 100% /snap/snapd/12704
tmpfs             100228      56    100172   1% /run/user/999
```

Fig 6.16: Snapshot output of df command

The following columns are contained in each line:

"Filesystem" - The filesystem's name.

"1K-blocks"- The filesystem's size in 1K blocks.

"Used"- Used means the used space in 1K blocks.

"Available" - Available means space available in 1K blocks.

"Use%"- The percentage of the used space.

"Mounted on"- The directory on which the filesystem is mounted/

Some options with df command

- 'df -h' option if we want to show information of all file system statistics in Gigabyte (GB).
- df-T option to tell df to display the file system types
- df -h If we want to show the details of disk drives in a human-readable format such as gigabytes, megabytes, kilobytes etc

fdisk can be used to manage disk partitions, you can create a new paritition, delete an existing partition…

the below let us see how to use fdisk command:

$ fdisk [options] FILESYSTEM

$fdisk -l

```
ubuntu@ubuntu:~$ sudo fdisk -l
Disk /dev/loop0: 2.1 GiB, 2160009216 bytes, 4218768 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes


Disk /dev/loop1: 65.1 MiB, 68259840 bytes, 133320 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

Fig 6.17: Snapshot of fdisk -l command



6.8 LET US SUM UP

- In Linux everything is treated as file and the file system is a hierarchical tree structure where root in the first node
- Command associated to work with file is ls, cd, rm, cp, mv
- Absolute path names are full path name to the file whereas relative pathnames are relative to the working directory
- For mounting a file system in Linux system, we use mount command and for unmounting we use unmount command
- df and du are two commands associated with Linux disk utility
- Leaner must go through to Linux manual to grasp the total functionality of these commands



6.9 FURTHER READINGS

1) Das, S. (2000).UNIX: Concepts and Applications. Tata McGraw-Hill.
2) Das, S. (2012). Your UNIX/Linux: The Ultimate Guide. (Third Edition). (Science/Engineering/Math). McGraw-Hill.
3) Christopher Negus (2012). Linux Bible. (8th Edition). Wiley.



6.10 ANSWERS TO CHECK YOUR PROGRESS

Ans to Q No 1: 3 types of file: Ordinary file, Directory File , Device File

Ans to Q No 2: For listing of files.

Ans to Q No 3: $tree command

Ans to Q No 4: $ mount command



6.11 MODEL QUESTIONS

Q 1: Describe tree organization of Linux file system.

Q 2: Create a test file in your home directory write something it display it on monitor then move the file to another directory.

Q 3: Explain how commands for directory is different than command used in case of file system.

Q 4: Explain the output of ls-l command in details.

Q 5: Why we need to mount a filesystem in Linux?

Q 6: Use df and du command to check disk utility.

*** ***** ***