Linux Commands

1. Getting help

a. The **man** command in Linux displays the manual pages of the specified command or configuration file. Type man followed by a command (for which you want help) and start reading. Press q to quit the man page. Some man pages contain examples (near the end).
$ man cp
This command displays help about the **cp** command.

b. It can also provide help about some of the configuration files.
$ man yum.conf

c. Sometimes a particular information may be present in a specific section of the manual page. It may be shown as **passwd(5).** The number in the bracket gives the section number of the man page. To access this section issue the following command.
$ man 5 passwd

d. It may also provide help about some running processes or daemons.
$ man auditd

e. To see just the description of a manual page, use whatis followed by a string.
$ whatis ifconfig

f. Should you be convinced that a man page exists, but you can't access it, then try running **mandb**
$ mandb

g. The **whereis** command displays the location of a command and the location of its manual pages also.
$ whereis ls

2. Directory Related Commands

a. **pwd** command (Print Working Directory) displays your current directory.
$ pwd

b. You can change your current directory with the **cd** command (Change Directory).
$ **cd /etc** → This command will change your current directory to /etc

c. The **cd** is also a shortcut to get back into your home directory. Just typing **cd** without a target directory, will put you in your home directory. Typing **cd ~** has the same effect.
$ cd → just typing **cd** at the prompt changes user current directory to user home directory
$ cd ~ → has the same effect as above command.

d. To go to the parent directory (the one just above your current directory in the directory tree), type **cd ..** .
$ cd ..

e. Another useful shortcut with **cd** is to just type **cd -** to go to the previous directory.
$ cd - → will take you to the earlier directory you were working in.

f. You can list the contents of a directory with **ls.**
$ ls

g. When a file name on a Linux file system starts with a dot, it is considered a hidden file and it doesn't show up in regular file listings. To show all files including the hidden files use option -**a** with **ls.**
$ ls -a

h. Typing **ls -l** (that is a letter L, not the number 1) gives you a long listing.
$ ls -l

i. Another frequently used **ls** option is **-h**. It shows the numbers (file sizes) in a more human readable format.
$ ls -h

j. **mkdir** command allows you to create directories.
$ mkdir dirname

k. **mkdir -p** allows you to create entire directory structure.
$ mkdir -p /parentdir/child1dir/dirname → creates all the three directories.

l. When a directory is empty, you can use **rmdir** to remove the directory.
$ rmdir dirname

m. you can use **rmdir –p** to recursively remove empty directories.
$ rmdir -p /parentdir/child1dir/dirname → will remove all the three directories if they are empty.

n. To remove a directory with its sub directories and files (means not empty directory) use **rm -r** command.
$ rm -r /parentdir → will delete /parentdir and everything below it.

However the above command prompts to confirm deletion of each file. To delete without confirmation use **–f** option with **rm**.

$ rm -rf /parentdir

0. The **tab key** can help you in typing a path without errors**.** Typing **cd /et** followed by the **tab key** will expand the command line to **cd /etc/.**

3. Working with files.

a. Files on Linux (or any Unix) are case sensitive. This means that **FILE1** is different from **file1**, and **/home** is different from **/Home**

b. In Linux everything is represented as a file. A directory is a special kind of file. Each terminal window (for example /dev/pts/4), any hard disk or partition (for example /dev/sdb1) and any process are all represented somewhere in the file system as a file.

c. The **file** utility determines the file type. Linux does not use extensions to determine the file type. The command line does not care whether a file ends in .txt or .pdf. One should use the **file** command to determine the file type.
$ file filename

Use **file -s** for special files like those in **/dev** and **/proc.**

d. One easy way to create an empty file is with **touch**.
$ touch filename

e. When you no longer need a file, use **rm** to remove it. Unlike some graphical user interfaces, the command line in general does not have a **waste bin** or **trash can** to recover files. When you use **rm** to remove a file, the file is gone. Therefore, be careful when removing files!
$ rm filename

Or **$ rm -rf filename** → delete forcefully without confirmation.

f. To copy a file, use **cp** with a source and a target argument.
$ cp source file target

$ cp /data/file1 /backup → This command copies file1 in data directory to the /backup directory.

g. To copy entire directory with its sub directories and files use **cp -r** command.
$ cp -r /data /backup → this command copies the entire data directory to the /backup directory.

h. Copy multiple files with **cp** as shown below.
$ cp file1 file22 file 33 /backup → This will copy three files to the /backup directory.

$ cp file* /backup → This will copy all files with name starting with file will be copied to the /backup directory.

i. The **mv** command moves (cut + paste in windows) the files or directories from one location to another.
$ mv /data/file1 /backup → will move the file1 from /data directory to /backup directory.

$ mv /data /backup → will move the entire data directory with all its contents to the /backup directory.

j. The **mv** command can rename the files and directories.
$ mv /data/file1 /data/file1.bak → will rename the **file1** in data directory to **file1.bak** name.

k. The **rename** command also can be used to rename files.
$ rename .txt .doc *.txt → this command will change the extension from .txt to .doc of those files which are present in the current directory.