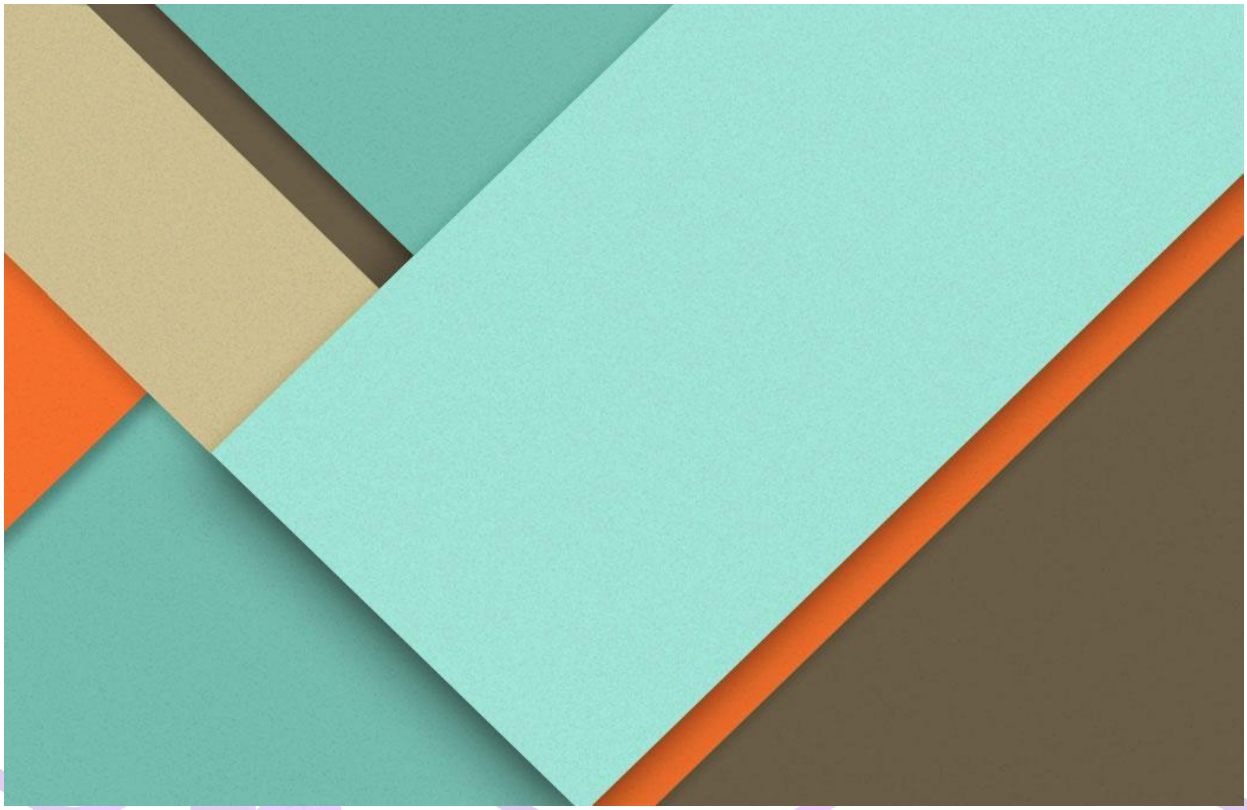
futureskills prime
A MeitY - NASSCOM Digital Skilling Initiative



Linux Basic Commands

CLOUD COMPUTING TEAM
CDAC
CHENNAI

Table of Contents

Linux Basic commands

File system commands

- **mkdir – make directory**

#mkdir <dirname> *-p > make parent directories as needed*	# mkdir -p path/test/test1

- **cd - change directory**

Type cd followed by the name of a directory to access that directory.	#cd /opt

- **mv - change the name of a directory**

Type mv followed by the current name of a directory and the new name of the directory.	#mv testdir newdirname

- **cp - copy files and directories**

cp source destination	#cp test1 test2
cp -r srcdir destdir *-r option - Copy all files from the directory "srcdir" to the directory "destdir" recursively.*	#cp -i myfile yourfile
cp -i myfile yourfile *With the "i" option, if the file "yourfile" exists, you will be prompted before it is overwritten.*	#cp -r srcdir destdir

- **rmdir - Remove an existing directory**

To remove a file	#rm filename
To remove directories and files within the directories recursively	#rm -r name

- **mount - Displays all mounted devices, their mount point, filesystem, and access.**

To display all mounted devices, their mount point, filesystem, and access.	#mount

Shell Metacharacters

These are special characters that are recognized by the shell

* - *matches 0 or more characters*	#ls *.c
? - *matches any single character*	#ls ab?.c
[] - *This will match any single character in the range This will find files such as tut0.m, tut9.m etc.,*	#ls tut[09].m
> - *Redirect standard output to a file*	#echo "hello world" > hello.txt
>> - *Appends standard output to a file.*	#echo "Hello Again" >> hello.txt
< - *Takes standard input from a file*	#cat < filename
| - *This is a pipe character. Sends the output of first command as input for the second command*	#who | grep sam

Basic Linux Commands

uname - *print system information*	#uname -a
diff - *find differences between two files eg)diff [options] fromfile tofile*	#diff -u testfile1 testfile2
sort –*reorders lines of text file.*	#sort testfile
sort -u - *To remove duplicates use u option with sort command*	#sort -u testfile
man - *displays the documentation for a command usage: man <command name>*	#man mkdir
pwd - *print working directory will show you the full path to the directory you are currently in.*	#pwd
link - *Creates a symbolic link named symlink that points to the file test*	#ln - s test symlink
free - *Displays the amount of used and free system memory.*	#free -m #free -g
df – *report file system disk space usage h > print sizes in human readable format*	#df -h

du - *summarize disk usage of each file, recursively for directories.*	#du -h
find - *Find locations of files/directories quickly across entire filesystem* *-type d -- search for the directory named appname* *-xdev -- Don't descend directories on other filesystems.* *-search -- against all directories below / for the appname found in directories but only on the existing filesystem.*	#find / -name appname -type d -xdev
find - *Command to find and remove files*	#find . -name "FILETOFIND" -exec rm -rf {} \;
lspci - *a utility for displaying information about PCI buses in the system and devices connected to them.* *-v – displays detailed information.*	#lspci -v
lsusb *– a utility for displaying information about USB buses in the system and the devices connected to them.* *v – displays detailed information.*	#lsusb -v
lshw - *list the hardware*	#lshw
cat /proc/cpuinfo *– gives information about cpu*	#cat /proc/cpuinfo
cat /proc/meminfo *- gives information about memory*	#cat /proc/meminfo
hwinfo *– probs for the hardware*	#hwinfo
ps *(i.e., process status) command is used to provide information about the currently running processes, including their process identification numbers (PIDs).* **ps** *– lists all the processes*	#ps -aux
kill *– to kill a process* *ps is most often used to obtain the PID of a malfunctioning process in order to terminate it with the kill command* *where pid – process id of the process to be killed*	#kill -9 pid

File Handling Commands

cat *-- used to display the contents of a small file on terminal*	#cat <file name>
more - *commands are used to view large files one page at a time*	#more <file name>
less - *commands are used to view large files one page at a time*	#less <file name>
wc - *command is used to count lines, words and characters, depending on the option used.*	#wc [options] [file name]

You can just print number of lines, number of words or number of characters by using following options: *-l : Number of lines* *-w : Number of words* *-c : Number of characters*	

Filters

Filters are commands which accept data from standard input, manipulate it and write the results to standard output.

head - *displays the lines at the top of the file* *when used without any option it will display first 10 lines of the file* *-n > print the first N lines instead of the first 10*	#head filename #head -n 10 filename
tail - *displays the lines at the end of the file. By default it will display last 10 lines of the file*	#tail filename
cut - *cut the columns/fields* **-c** *option to cut the columns from a file* **-f** *option you can cut the fields delimited by some character* **-d** *option is used to specify the delimiter and* **-f** *option used to specify the field number*	#cut -c 1,3-5 /etc/passwd #cut -d':' -f2 /etc/passwd
paste - *command will paste the contents of the file side by side*	#paste a.txt b.txt

Pattern Searching

grep - *scans its input for a pattern, displays the line containing that pattern*	#grep options pattern filename(s)
grep - *searching for a text string in one* *searches for the pattern boss in the /etc/passwd file*	#grep 'boss' /etc/passwd
grep - *searching for a text string in multiple files*	#grep 'root' *.txt
grep - *Case Insensitive file searching*	#grep -i 'hello' hello.txt
grep - *Reversing the meaning of a grep search.* *Displays all the lines that do not contain the specified pattern*	#grep -v 'boss' /etc/passwd
grep with pipeline	#ps -aux | grep firefox
egrep with pipeline - *Linux grep command to search for multiple patterns at one time*	#egrep 'boss|root' /etc/passwd

grep - *pattern matching and regular expressions (regex patterns)*	#grep '[FG]oo' * #grep '[0-9][0-9][0-9]' * #grep '^fred' /etc/passwd

Task Automation

Cron is the name of a program that enables linux users to execute commands or scripts (groups of commands) automatically at a specified time/date.
You can set up commands or scripts, which will repeatedly run at a set time.
- The cron service (daemon) runs in the background and constantly checks the /etc/crontab file, /etc/cron.*/ directories.
- It also checks the /var/spool/cron/ directory.

crontab - *To edit the crontab file, type the following command at the Linux shell prompt:* *Syntax of crontab (Field Description)* *where* **m**: *Minute (0 - 59)* **h**: *Hours (0 - 23)* **dom**: *Date (0 - 31)* **mon**: *Month (0 - 12 [12 == December])* **dow**: *week days(0- 7 [0 or 7 sunday])* **/path/to/command** - *Script or command name to schedule*	#crontab -e m h dom mon dow /path/to/command arg1 arg2
Every day at 3am, *If you wished to have a script named /root/backup.sh run everyday at 3 am, your crontab entry would look like as follows:*	0 3 * * * /root/backup.sh
Execute every minute *This script is being executed every minute.*	* * * * * /bin/script.sh
Execute every Friday 1AM *To schedule the script to run at 1AM every Friday, we would need the following cronjob:* *The script is now being executed when the system clock hits:* *1. minute: 0* *2. of hour: 1* *3. of day of month: * (every day of month)* *4. of month: * (every month)* *5. and weekday: 5 (=Friday)*	0 1 * * 5 /bin/execute/this/script.sh
Execute on workdays 1AM *To schedule the script to run from Monday to Friday at 1 AM, we would need* *the following cronjob:* *The script is now being executed when the system clock hits:* *1. minute: 0*	0 1 * * 1-5 /bin/script.sh

2. of hour: 1 *3. of day of month: * (every day of month)* *4. of month: * (every month)* *5. and weekday: 1-5 (=Monday till Friday)*	
Execute 10 past after every hour on the 1st of every month	**10 * 1 * * /bin/script.sh**
Run script every 10 minutes	**0,10,20,30,40,50 * * * * /bin/script.sh** **(or)** ***/10 * * * * /bin/script.sh**
Special Words ***If you use the first (minute) field, you can also put in a keyword instead of a number*** **@reboot Run once, at startup** **@yearly Run once a year "0 0 1 1 *"** **@annually (same as @yearly)** **@monthly Run once a month "0 0 1 * *"** **@weekly Run once a week "0 0 * * 0"** **@daily Run once a day "0 0 * * *"** **@midnight (same as @daily)** **@hourly Run once an hour "0 * * * ***	**@daily /bin/script.sh**
Storing the crontab output ***To store the output in a separate log file.***	***/10 * * * * /bin/script.sh 2>&1 >> /var/log/script_output.log**