What is Operating System?

=> Operating System is a software which acts as mediator between Users and Computers

=> OS is mandatory to use any computer

=> There is no use of Computer which doesn't have Operating System

=> By using OS we can run applications in computer

What is Windows OS?

-> Windows OS developed by Microsoft (Bill Gates) Company

-> Windows OS is licensed s/w (we have to purchase)

-> Windows OS is single user Operating System

-> Security Features are less in Windows OS

-> Anti Virus S/w we have to install to proectect our System.

-> Windows OS is GUI (Graphical User Interface) based.

-> Windows OS is recommended for Personal Use (gaming/movies/calculations/files)

Linux OS

-> Linux is free OS

-> Linux is Multi User Operating System

-> Security Features are very good in Linux OS (Anti Virus not required)

-> To run Servers, Databases Linux OS is highly recommended

-> Linux is CLI (Command Line Interface) based

Note: We should learn Linux commands to use Linux Machines.

History of Linux

-> Linux OS developed by" Linus Torvalds"

-> Before Linux we had Unix Operating System. Linus Torvalds identified some issues in Unix OS and told to that company to fix those issues. But Unix company rejected the suggestions given by Linux Torvalds.

-> Linus Torvalds used 'Minux' OS and developed his Own OS that is called as Linux OS.

(Li) nus + Mi (nux) => Linux

-> Linus Torvalds developed Linux OS and released Linux OS source code into market for free of cost.

-> So many companies downloaded the Linux OS source code and modified according to their requirement and released into market with their brand names those are called as Linux Distributions/ Linux flavours.

-> Linux having 200+ Distributions.

-> RED HAT Linux
-> Cent OS Linux
-> Ubuntu Linux
-> Debian Linux
-> SUSE Linux
-> FEDORA Linux
-> **KALI** Linux
-> Amazon Linux

Linux Machine Setup

1) Install Linux OS directley in our Machine

2) Create Virtual Machine in Cloud Platform (AWS)

Lab Task for Today

1) Create Linux Virtual Machine in AWS Cloud (Amazon Linux)

2) Download MobaXterm s/w

3) Connect with Linux VM using MobaXterm.

Linux Commands

whoami : Logged in username

date : display current date

cal : display calendar

clear : to erase terminal content

pwd : To display present working directory

mkdir   to create directory

rmdir   to delete empty directory

Note: To delete non-empty directories we will use below command

```
$ rm -r <dirname>
```

cd   to change directory

ls   To display files & directories of present working directory

```
                ls -1 : display files and directories in alphabetical order

                ls -lr  reverse of alphabetical order

                ls -lt   Display files based creation date & time (latest on top)

                ls -ltr  Display files based creation date & time (latest on bottom)

touch   To create empty files

                $ touch f1.txt f2.txt f3.txt

cat    create files with data, append data to existing file, display file content

                        cat> filename  (To create file with content)

                        cat>> filename (append data to existing file)

                        cat filename (display file content)

Note: cat command will display file data from top to bottom.

tac : tac command will display file data from bottom to top

                $ tac filename

cp   It is used to copy data from one file to another file

                $ cp f1.txt f2.txt

Note: To copy data from multiple files we need to use 'cat' command.

                $ cat f1.txt f2.txt > f3.txt

rm   to delete file

                $ rm filename

mv   to move/ rename the file

                $ mv old-name  new-name

                $ mv filename <location>

wc   word count

head   To display first 10 lines of file

                $ head <filename>  (default first 10 lines)

                $ head -n 12 <filename>  (it   will give first 12 lines)

                $ head -n 50 <filename>  (it will give first 50 lines)

tail   To display last 10 lines of file

                $ tail <filename>  (default last 10 lines)

                $ tail -n 15 <filename>  (it   will give last 15 lines)
```

```
Note: To check latest logs of the application we will use 'tail' command.
```

Text Editor in Linux

vi : visual editor

=> Using 'vi' command we can work with text editor

=> To open file we can use below command

```
                $ vi ashokit.txt
```

insert mode   press 'i'

escape mode   press 'esc'

To save changes and close file=> esc + :wq

Close file without saving changes=> esc + :q!

SEO command

=> SEO stands for stream editor

=> SEO is used for substitution/replacement

=> We can replace one word with another word using SEO command without opening the file

```
        $ sed 's/aws/azure/' ashokit.txt
```

Note: By default, SEO command will not make changes in the original file. It will just print output on terminal.

```
        $ sed -i 's/aws/azure/' ashokit.txt
```

=> Delete last line of the file

```
        $ sed '$d' ashokit.txt
```

=> Delete 5th line of the file

```
        $ sed 'Sd' ashokit.txt
```

=> Delete 10th line to last line

```
        $ sed '10,$d' ashokit.txt
```

File Permissions

=> We have 3 types of permissions

```
        r    read
        w    write
        x    execute
```

=> File Permission contains 3 sections (total 9 characters)

1) User section (first 3 characters)
2) Group Section (middle 3 characters)
3) Others section (last 3 characters)

```
-rwxrwxrwx    f1.txt

-rw-rw-r--    f2.txt

-r--rw-rw-    f3.txt

dr-xr--rwx    linux
```

=> To change file permission we have 'chmod' command in linux

```
             +     means add permission

                   means remove permission
```

```
chmod u+x  f1.txt (adding execute permission for user)

chmod g+x  f2.txt  (adding execute permission for group)

chmod o-x  f3.txt  (removing execute permission for others)

chmod u+wx f4.txt  (adding write and execute for user)
```

=> File permissions we can manage with numeric numbers also

0	No Permission
1	Execute
2	Write
3	Execute + Write
4	Read
5	Read+ Execute
6	Read + Write
7	Read + Write + Execute

Q-1) What is the default permission for the file in numeric format? (664)

Q-2) What is the default permission for the directory in numeric format? (775)

Users & Groups in Linux

-> Linux is Multi User based OS

-> We can create Multiple User accounts in one Linux Machine

-> In Every Linux machine by default 'root' user account will be available

root user : super user

-> When we launch EC2 instance (Amazon Linux AMI) We got 'ec2-user' account also.

-> For every user account one home directory will be available under /home directory

```
ec2-user  /home/ec2-user
ashok     /home/ashok
john       /home/john
cathy     /home/cathy
raju      /home/raju
```

=> We can get user account information by using 'id' command

```
$ id <username>
```

# switch to root user

```
$ sudo su (switch to root account)
$ sudo su - (switch to root user and point to root user home directory)
$ exit  (exit from user account)
```

Working with Users and Groups in Linux

# Switch to root user

```
$ sudo su -
```

# Create New User account

```
$ useradd <uname>
```

# Set Password for User

```
$ passwd <uname>
```

# Display Users created

```
$ cat /etc/passwd
```

# Switch to user account

```
$ su <username>
```

# Delete user

```
$ userdel <uname>  (user will be deleted but user home directory will not be deleted)
$ userdel <uname> --remove  (user deletion+ user home directory deletion)
```

```
# Display all groups available in Linux

        $cat/etc/group

# Create New Group

        $ groupadd <group-name>

# Adding user to group

        $ usermod -aG <group-name> <user-name>

#  Remove user from the group

        $ gpasswd -d <user-name> <group-name>

# Display Users belongs to a group

        $ sudo lid -g <group-name>

# Delete Group

        $ groupdel <group-name>

# Change Group Name

        $ groupmod -n <new-name> <old-name>

# change username

        $ usermod -l <new-name> <old-name>

# Change User password

        $ passwd <uname>
```

chown command

=> This is used for ownership change

```
# change owner of the file

        $ sudo chown <uname> <file-name/directory-name>

# We can change owner of the file using user-id also

        $ sudo chown <UID> <file-name/directory-name>

# change owner-group

        $ sudo chown :group-name <file-name/directory-name>

# change owner-group using group-id

        $ sudo chown :group-id <file-name/directory-name>

# change owner and group at a time

        $ sudo chown <user>:<group> <file>
```

locate and find commands

=> 'locate' and 'find' commands are used for file search

=> In every linux machine locate db will be available which stores linux file system details.

=> When we execute locate command it will fetch details from locate db.

=> 'find' command will do search in entire linux file system

=> 'find' command is used for more advanced search

```
        # search for the files under home directory with given name
        $ sudo find /home -name f1.txt

        # find empty files under /home

        $ sudo find /home -type f -empty

        # find empty directories under /home

        $ sudo find /home -typed -empty

        # Find 30 days old files under /home directory

        $ sudo find /home -mtime 30 -print

        # Find & delete 30 days old files under /home directory

        $ sudo find/ -mtime 30 -delete
```

Working with Zip files in Linux

```
# Create few files using touch command
$ touch f1.txt f2.txt f3.txt

# Veriy zip command is installed or not
$zip-version

# Create zip with all text files
$ zip <zip-file-name> *.txt

# print content of zip file
$zip-sf  <filename>

# Add new file/directory to existing zip file
$ zip -r <zip-name> <file/directory>

# Delete a file from zip
$ zip -d <zip-file-name> <file-name-to-delete>

# Create zip file with encryption (setting password)
$ zip -e <zip-file-name> *.txt

# Unzip
$ unzip <zip-file-name>
```

'man' command is called as helper command ping : To check connectivity

```
        $ ping www.gmail.com
        $ ping 13.12.109.21
```

wget : To download files from internet using url

```
$ wget https://dlcdn.apache.org/tomcat/tomcat-10/v10.1.10/bin/apache-tomcat-10.1.10-windows-x64.zip
```

curl : It is used to send HTTP Request to URL

```
        $ curl 192.168.1.123
```

ifconfig : To get IP Address of our machine

```
        $ ifconfig
```

free : To display memory details

top : To display running processes

htop : To display running processes in Graphaical format

history : It is used to disply all commands executed (commands history)

Package Managers in Linux

-> Pkg managers are used to install softwares in linux machines.

apt : Advanced Package Tool (debian based distributions)

                Ex: Ubuntu

        Syntax : sudo apt install git

dpkg : It is underlying pkg manager used by 'apt'

yum : Yellowdog updater, modified (RED Hat based distributions)

                Ex: RED Hat Linux, Amazon Linux, Cent OS linux

dnf: DANDIFIED YUM (It is replacing yum pkg manager in latest versions of fedora and Cent OS)

How to install softwares in linux

Ubuntu linux	we should use 'apt' package manager
Amazon linux	we should use 'yum' package manager

```
# Install Java software
```

```
$ sudo yum install java

# Install java 1.8 version
$ sudo yum install java-1.8.0-openjdk

# Install Maven software
$ sudo yum install maven

# Install Git client
$ sudo yum install git
```

How to host static website in Amazon Linux

=> Website means collection of web pages

=> Websites are divided into 2 types

1) Static Website

2) Dynamic Website

=> Static websites will give same response for all users

=> Dynamic Websites will give response based on logged in user

Note: To run a website we need a web server

Ex: apache, httpd etc...

```
# update the packages
$ sudo yum update

# install httpd webserver
$ sudo yum install httpd

# Start the service
$ sudo service httpd start
```

Note: After starting httpd service, we can access our webserver using EC2-VM public IP address.

Note: Enable HTTP Protocol in Security Group Inbound Rules to allow incoming traffic.

=> After enabling HTTP port then access EC2-VM public IP in browser

```
        URL : http://65.1.132.215/

             http://65.1.132.215:80/
```

=> We can modify the content of our static website

```
        $ cd /var/www/html

        $ sudo vi index.html
```

Note: Write website content in index.html then save and close that file then access Ec2-vm public ip.

```
What is Grep Command
```

=> It stands for "Global Regular Expression Print"

=> The grep command is a powerful utility in Linux used for searching and filtering text

=> The grep command searches for patterns or regular expressions in files or input streams and displays lines that match the specified pattern.

Syntax: grep [options] pattern [file...]

```
# Pattern matching print
$ grep "keyword" file.txt

# Recursive print
$ grep -r "pattern" directory/

# Ignore case
$ grep -i "pattern" file.txt

# print line numbers
$ grep -n "pattern" file.txt

# Invert match (print files which are not matching given pattern)
$ grep -v "pattern" file.txt
```

What is AWK Command

=> The awk command is a versatile text processing tool available in Unix-like operating systems, including Linux.

=> It allows you to manipulate and extract data from structured text files, usually in a columnar format.

=> awk takes input, processes it line by line, and performs actions based on specified patterns and rules.

Sytax awk 'pattern {action}' file

```
$cat>  employee.txt

Ashok manager account 45000
John clerk account 25000
Smith manager sales 50000
Charles manager account 47000
Ganesh peon sales 15000
Mahesh clerk sales 23000
Ram peon sales 13000
Cathy director purchase 80000
```

```
$ awk '{print}' employee.txt

$ awk '/manager/ {print}' employee.txt

$ awk '{print $1,$4}' employee.txt

$ awk '{print NR,$0}' employee.txt
$ awk '{print NR "- "$1  }' employye.txt
```

1) whoami
2) pwd
3) date
4) cal
5) cd
6) ls **-ltr**
7) mkdir
8) rmdir
9) rm
10) touch
11) cat
12) tac
13) wc
14) cp
15) mv
16) head
17) tail
18) chmod
19) chown
20) useadd
21) passwd
21) usermod
22) gpasswd
22) userdel
23) id
24) groupadd
25) groupdel
26) grep
27) awk
28) man
29) ifconfig
30) ping
31) wget
32) curl
33) locate
34) find
35) free
36) top
37) htop
38) zip
39) unzip
40) vi
41) sed
42) sudo su -
43) cat /etc/as-release
44) uname -r
45) cat /etc/passwd
46) cat /etc/group
47) visudo

Q) What is sudoers file?

=> Sudoers file is used to configure user permissions.

Q) What is .bashrc file?

```
=> For every user, .bashrc file will be created under home directory.
```

=> .bashrc is a hidden file (it starts with .)

=> To display hidden files we will use 'ls -la'

=> Using .bashrc we can set Environment Variables for user account.

Linux Architecture

1) Applications

2) Shell

3) Kernal

4) Hardware

=> When we execute any command in linux then Shell will process our command execution.

=> Shell will verify command syntax is valid or not.

=> Shell will convert our command into 'kernel' understandable format

=> Kernel is responsible to communicate with Hardware components

=> Kernel is the mediator between Shell and Hardware.

=> Shell is mediator between 'user' and 'kernel'

```
# Print all shells of linux machine
$cat/etc/shells
```

```
# Print default shell of linux machine
$ echo $SHELL
```

Note: We have '/bin/bash' as default shell.

What is Shell Scripting?

=> Shell will act as mediator between Users and Kernel. Shell will convert commands into kernal understandable format.

=> Scripting means executing set of commands using a file.

##### The process of executing set of commands available in the file using SHELL is called as Shell Scripting#####

=> Shell Scripting is used to automate our daily routine works.

=> Shell Script file will have .sh extension

=> Below is the command to execute shell script file

```
        $ sh <filename>
```

Script-1 : work.sh

```
#! /bin/bash

whoami
pwd
date

#### Run : sh work.sh####
```

Script-2 : msg.sh

```
#! /bin/bash

echo 'Hello World'
echo 'Welcome to DevOps World'
echo 'DevOps is very intresting'
```

Script-3 : NameDemo.sh

```
#! /bin/bash

echo 'Enter Your Name'
read a
echo "Good Evening $a"
```

Script-4 : Sum.sh

```
#! /bin/bash

a=10
b=20

C=$(($a+$b))

echo "Sum is : $c"
```

Script-5 : DynamicValSum.sh

```
#! /bin/bash

echo "Enter First Number"
read a

echo "Enter Second Number"
read b

C=$(($a+$b))

echo "Result  $c"
```

Variables

```
=> Variables are used to store the data
```

=> Variables are key-value pairs

```
        id=  101

        name   ashok

        gender   male
```

=> Variables are divided into 2 types

        1) Environment Variables (System Variables - Already Defined)

        2) User Defined Variables (We will create these variables)

Note: To access value of variable we will use '$' symbol.

```
        Ex: $SHELL  $USER
```

Command Line Arguments

=> The arguments which we we will pass to script file at the time of execution.

```
        Ex:  sh demo.sh ashokit 30
```

=> We can access command-line args in script file like below

$# - No.of args

$0 - script file name

$1 - First Cmd arg

$2 - Second Cmd arg

$3 - Third Cmd Args

$* - All cmd args

Script-6 : DynamicValSum.sh

```
#! /bin/bash

result=$(($1+$2))

echo "Sum is : $(($1+$2))"
```

Run : sh DynamicValSum.sh 10 20

=> We have 2 options to pass dynamic values to shell script file

1) Using 'read' command

2) Using cmd args

1) Conditional Statements (if - elif - else)

2) Loops (for & while)

3) Functions

Conditional Statements

=> Conditional Statements are used to execute 'commands' one time based on condition

Ex : if - else / if - elif - else

Syntax:

```
if  [ condition ]
then
   statements
else
   statements
```

Script-7 : if-else.sh

```
#!  /bin/bash

echo "Enter Name"
read name

if  [ $name    'john' ]
then
   echo "Hi"
else
    echo "Bye"
fi
```

Script - 8 : if-elif-else.sh

```
#!  /bin/bash

echo "Enter Name"
read name

if [$name== 'john'
then
   echo "Hi $name"

elif [$name== 'smith'
then
   echo "Hello $name"

else
    echo "Bye $name"
fi
```

Script - 9 : if-elif-else.sh

```
#!  /bin/bash
```

```
echo "Enter first number"
```

```
read a

echo "Enter second number"
read b

if [$a-gt $b]

then
   echo "$a is greater than $b"

elif [ $b -gt $a]
then
   echo "$bis greater than $a"

else
   echo "Both are equal"

fi
```

Loops

=> To print "hi" 5 times we will write script like below

```
echo "hi"
echo "hi"
echo "hi"
echo "hi"
echo "hi"
```

=> If we want to print "hi'' for 1000 times, can we write echo for 1000 times ? Not recommended

=> To execute same command multiple times then we will use "Loops"

=> We have 2 types of loops

1) Range Based Loop (Ex: for loop)

2) Conditional Based Loop (Ex: while loop)

=> Print "hi" message 10 times (Here we know the range to print msg - we can use 'for' loop)

=> Print ''good night" message till 10 PM (Here we don't range but we know condition - we can use while loop)

Script-10 : for-loopl.sh

```
#! /bin/bash

for (( i=l; i<=10; i++ ))
do
  echo "hi"
done
```

Script-11 : for-loop2.sh

```
#! /bin/bash
```

```
for (( i=1; i<=10; i++ ))
```

```
do
  echo "$i"
done
```

Script-12 : while-loop.sh

```
#! /bin/bash

i=10
while [ $i -ge 1]
do
  echo "$i"

 let i--;
done
```

Functions/ Methods

=> Functions are used to perform some action/ task

=> The big task can be divided into smaller tasks using functions

=> Using functions we can divide our tasks logically

=> Functions are re-usable

Syntax:

```
# writing function
function functionName ( ) {

        // function body
}

# calling function
functionName
```

Script - 13 : funl.sh

```
#! /bin/bash

function welcome ( ) {

   echo "this is line 1"
   echo "this is line 2"
   echo "this is line 2"

}

# calling function
welcome
```

Script - 14 : Read File name then print content of that file

```
#! /bin/bash
```

```
function readFileData ( ) {
   echo "Enter file name to read"
   read filename
   cat $filename
}

# calling function
readFileData
```

Script - 15 : Read File name as cmd arg then print content of that file

```
#! /bin/bash

filename=$1

function readFileData ( ) {
     cat $filename
}

# calling function
readFileData
```

Script-16: Check file presence

```
#!  /bin/bash

echo "enter filename"
read filename

if [ -f "$filename" ] ; then
echo "File already exist"
else
echo "File not exist, hence creating..."
touch $filename
echo "file created ..."
fi
```

Script-17: Check directory presence

```
#!  /bin/bash

echo "Enter directory name"
read dirname
if [ -d "$dirname" ] ; then
        echo "Directory exist"
else
        echo "directory not available, hence creating.."
        mkdir $dirname
        echo "directory created"
fi
```

Shell Script Programs For Assignment

1) Write shell script to check given number is even number or odd number
   (take number from user)

```
2) Write shell script to check given number is prime number or not
```

(take number from user)

3) Write shell script to check given year is leap year or not
    (take year from user)

4) Write shell script to check given string is palindrome or not
   (take string from user)

5) Write shell script to print table of given number (take number from user)

```
        2 * 1   2
        2 * 2 = 4
        2 * 3   6

        2 * 10 = 20
```

What is CRON?

=> Cron is a utility in Linux which is used to schedule jobs execution

=> In Realtime we will have several jobs for execution on daily/ weekly/ monthly/ yearly basis

1) Take Backup of files
2) Delete temp files
3) System Maintanence

Usecase

=> Execute shell script file for every 5 minutes.

Note: Instead of human executing script file for every 5 mintues we can schedule script file execution using CRON.

What is CROND?

=> In Linux machine CROND is a daemon process (background) which will check scheduled CRON JOBS for every minute.

=> If any job is scheduled then CROND will execute that job based on given schedule.

CRON JOB Syntax

```
        * * * * * <command/ script-file>
```

Note: Read CRON expression from left side

=>first* will represent minutes (0 - 59)

=>second* will represent hour (0 - 23)

=>third* will represent day of month (1 - 31)

=>fourth* will represent month of year (1 - 12)

```
=>fifth* will represent day of week ( 0 - 6, mon - sun)
```

Sample Cron Expressions

```
Run for every 15 mins      */15 * * * *  task.sh

Run every day @5 am         0 5 * * *  task.sh

Run every day @5 pm         0  17 ***task.sh

Run every month first day @9 am : 0 9 1 **task.sh
```

What is crontab file

=> In linux for every user account one crontab file will be available

=> crontab file is used to configure cronjobs for execution

```
# Open crontab file
$ crontab -e

# Display cronjobs schedules
$ crontab -1

# Remove crontab file
$ crontab -r
```

Check Cron Status

```
$ sudo systemctl status cron
```

CRON JOB Practice

1) Launch Linux Machine with Ubuntu AMI

2) Connect with Ubuntu VM using MobaXterm

3) Create shell script file (task.sh) like below

```
$ vi task.sh
```

Note: keep below content in shell script file

```
touch /home/ubuntu/f1.txt
touch /home/ubuntu/f2.txt
```

4) Provide execute permission for script file

```
        $ chmod +x task.sh
```

5) Open crontab file and configure job schedule

```
        $ crontab -e
```

Note: Enter below cron job in crontab file

```
        */1 ****/bin/bash   /home/ubuntu/taskl.sh
```

6) Save and close crontab file (ctrl + **x)**

7) Check files creation in pwd

```
        $ ls -1
```

Note: We can observer fl.txt and f2.txt files got created.

What is inode number?

=> !node is one unique number that will be assigned for every file in linux

=> Linux will use inode number to map our files with its name in the linux db

Hard Links and Soft Links

=> In linux we can create link files (similar to shorcut files in windows)

=> We have 2 types of link files in linux

        1) Hard Link

        2) Soft Link

Syntax To create Hard Link:

```
$ ln <orginal-file> <link-file>
```

Ex:

```
$ touch ml.txt
$ ln ml.txt mll.txt
```

Note: mll.txt is hard link file for ml.txt

```
$ ls -li
```

Note: ml.txt and mll.txt files are having same inode number

Note: If we write some data to ml.txt that data will reflect in mll.txt file also

Note: If we delete ml.txt file still there is no effect on mll.txt

Syntax To create Soft Link :

```
$ ln -s <orginal-file> <soft-link-file>
```

Note: Soft Link is like shortcut in windows

Ex:

```
$ touch sl.txt
$ ln -s sl.txt sll.txt
$ ls -li
```

Note: Original file and soft link file having different inode numbers

```
$ cat »  sl.txt
```

Note: Data writing in original file will reflect in soft link file also

```
$ rm sl.txt
```

Note: When we remove original file then soft link file will become dangling file. We can't access that.

Process management

=> Process represents a running program/application in OS

=> We can see the running process using 'Task Manager' in Windows
(CTRL +SHIFT+ ESC)

=> We can see the running process in linux using below command

```
$ ps
```

Note: For every process one unique PID will be available (Process ID)

ps : display all runnig processes

top : Display dynamic view of system processes

=> For every process there will be a state (Process state)

Running (R) : Currently executing

Sleeping (S) : Waiting for event/ source

Stopped (T) : Stopped execution

Zombie (Z) : Finished execution but it waiting for exit status.

Note: we can use kill command to kill a process

```
$ kill -9 <pid>
```

1) What is Operating System

2) Windows OS

3) Linux OS

4) Linux History

5) Linux Distributions *I* Flavours

6) Linux VM Setup in AWS Cloud (EC2)

7) How to connect with Linux VM

(mobaxterm & putty)

8) Linux Commands (50+

9) Working with Directories

10) Working with Files

11) File Editors (vi & sed)

12) File Data Processing (grep & awk)

13) File Search (locate & find)

14) Users & Groups Management

15) File Permissions (chmod)

16) Ownership change (chown)

17) Working with Zip & Unzip

18) File Links (Hard & Soft)

19) Networking commands (ping, curl, wget, ifconfig)

20) Package Managers

21) Softwares Installation

22) Static Website Hosting in Linux (httpd)

23) Process Management (ps)

24) Linux Architecture

25) What is Shell Scripting

26) Shell Script Execution

27) Variables

28) Reading Data From User

29) Command Line Arguments

30) Conditional Statements (if-elif-else-fi)

31) Loops (for & while)

32) Functions

33) CRON JOBS

34) What is Sudoers file

35) What is .bashrc file