Objective:- Assignments will help trainees to understand the basics of unix command and how to explore and use it.

\*\*\*DO NOT copy paste commands from internet, though you can take help of teh internet for the assignments below.

\*\*\*\*Be Honest to yourself!

important command and option to learn unix commands:-

1. man <-- Command

Example:-

man ls

2. --help <-- Option

Example:-

ls --help

NOTE:- Always use "man" command before using any unix commands and read about it and its options and how to use it.

Assignments:-

1. List the contents of a directory and their attributes

Ls -d \*

2. Long list the content with file, directory ownership, permissions,sizes, etc...

Ls –l

3. Display the size of the file in human readable format

Ls –lh /usr/bin/vim.py

du –h /usr/bin/vim

4. Show all files and folders including hidden one

Ls –a

5. list directories recursively

Ls –R

6. Sort the files by size with largest at the top

Ls -S

7. Sort the files by last time modified displaying the newest first.

Ls –t

8. Diplay the location of a program/command, where it is installed.

dpkg –L Filename | grep etc

9. Which command is used to switch directory from one to another

Cd

10. List all the environment variables set for the current shell environment

Env ,

11. Did you notice something in the output of "env" command?

printenv

12. Which command is used to print the text or any variables value in the Console/Terminal?

echo variable

13. Print the value of the env variable "PATH" on the console

echo $ PATH

14. Is linux a case-sensitive operating system?

Yes

15. Is, "Ls" same as "ls"?

NO

16. Display your currently logged in user

Who, who am I

17. how do you change the currrently logged in user to another user?

Su username

Password:

18. Which command is used to leave a shell environment that you are currently logged in to?

Exit N

Exit 0

19. How do you reboot the system?

# /sbin/reboot

$ sudo reboot

20. How do you shutdown the system?

# /sbin/shoutdown –r now

21. Display all the major running processes in the system

Top, ps

22. Understand the output of command used in above question of displaying processes, explain the meaning of each column and what data it displays?

23. Display the name of the system kernel

$uname –a

24. display the kernel release number

$uname –r

25. display teh machine type of the current kernel

$uname –m

26. Display the name of the operating system that the kernel is running on

$uname –0

27. Display all info that uname command can show.

$uname –a

28. Display the name of directory that you are currently pointing to

Pwd

29. change the current directory to another directory that you have in your system.

Cd

30. Go up one directory

Cd ../..

31. Return to last directory

$oldpwd

32. change the current directory to home(logged in user's) directory

Cd ~

33. How to check all the command used from the prompt (Command History)

#history

34. In which file the history of commands are stored in?

/home/xyz/.bash\_history

35. How many lines of history does the system keep and from where you can change it?

500

HISTFILESIZE

36. How can you modify bash's history behaviour

HISTFILESIZE=1000

37. Display all the commands entered so far, now, try to run a particular command from the history list without typing that command.

History | grep cmd

NOTE:- TAB key is your friend when it comes to command completion and having long file and directory names autocompleted at the bash prompt for you. JUST BE LAZY AND USE TAB KEY FOR AUTO COMPLETION ;-)

38. What are the different types of shell and where are they used and how do we use them?

Csh, bsh,ksh,bash

39. What is the difference between login shell and non-login shell?

login shell :- Works after login and its read credentials from /bin/login and password from /etc/password file

non-login shell:- calls program without login it calls directly ~/bashhrc

40. How do we start login shell and non-login shell?

If output of echo $0 starts with -, it's login shell (echo $0 output example: -bash)

non-login shell (echo $0 output example: bash).

41. What happens when you start a login shell (which files are read and used and Why)?

/etc/password file

`/.bash\_profile – all user profile stored within this

42. What happens when you start a non-login shell (Which files are read and used and Why)?

/etc/bashhrc

/etc/profile.d

39. What are Shell Configuration Files, why do we need it?

Shell configuration files are executed automatically when you log in

perform cleanup operations post logout

.bash\_profile, ---- set environment variables and profile

.bashrc --- contains commands that configure the shell

.bash\_logout --- executes when user log out

40. Explain the Order of file usage from the system/user's home directory when user logs in to the System.

/home/user1/file

41. What are Shell Variables, list major shell variables and what do they represent?

A shell variable is a special variable that is set by the shell and is required by the shell in order to function correctly. Some of these variables are environment variables whereas others are local variables

"variable\_name=variable\_value"

42. How we see all our env variables?

printenv > env.txt

cat env.txt

43. How we see all env variables in alphabical order?

44. What Format does the env var and its values are stored?

.env

45. How do you create your own varible?

Var = “xyz”

46. How do you start a new bash shell?

47. Difference between Local/Shell variables to Global Variable

Global variables are declared outside any function, and they can be accessed (used) on any function in the program. Local variables are declared inside a function, and can be used only inside that function. It is possible to have local variables with the same name in different functions.

48. Making a variable accessible from other the shell in the system.

There is no "global environment variable space". If you want to make your data *persistent*, then you need to store it in some file. Example:

!/bin/sh

# Path to the persistent storage file

file=~/name.txt

# If the persistent file exists, use its contents as name

if [ -e "$file" ]; then

name=$(cat "$file")

else

# Use a default value otherwise

name="Bob"

fi

echo "Your name was: $name"

echo "What is your name?"

read new\_name

# If a new (non-empty) name is given, save it to the file

if [ -n "$new\_name" ]; then

echo "$new\_name" > "$file"

fi

49. Show the real life use case of exporting a variable

There is no Global Variable so we are using local as Global and below ie the exporting command

[paul@RHEL4b ~]$ var3=three

[paul@RHEL4b ~]$ var4=four

[paul@RHEL4b ~]$ export var4

[paul@RHEL4b ~]$ echo $var3 $var4

three four

[paul@RHEL4b ~]$ bash

[paul@RHEL4b ~]$ echo $var3 $var4

four

50. Convert the above script file into a command, The file should run with "myappp" instead of "myapp.sh"

51. What is Globbing? Explain in depth with examples?

the process of expanding a non-specific file name containing a wildcard character into a set of specific file names that exist in storage on a computer, server, or network.

**[root@linuxnix tmp]# cp -Rv /tmp/abc/\* /tmp/lmn/**

**`/tmp/abc/def' -> `/tmp/lmn/def'**

**`/tmp/abc/def/nop' -> `/tmp/lmn/def/nop'**

**`/tmp/abc/ghi' -> `/tmp/lmn/ghi'**

**[root@linuxnix tmp]# df -hTP /?\*[0-9]**

**Filesystem Type Size Used Avail Use% Mounted on**

**/dev/mapper/vg\_cent68-lv\_root ext4 13G 2.3G 9.9G 19% /**

**/dev/mapper/test-lv06 ext4 93M 1.6M 87M 2% /test1**

**/dev/mapper/vg\_cent68-lv\_root ext4 13G 2.3G 9.9G 19% /**

**/dev/mapper/test-lv07 ext4 93M 1.6M 87M 2% /test24**

**/dev/mapper/test-lv01 ext4 93M 1.6M 87M 2% /u01**

**/dev/mapper/test-lv02 ext4 93M 1.6M 87M 2% /u02**

**/dev/mapper/test-lv03 ext4 93M 1.6M 87M 2% /u03**

**/dev/mapper/test-lv04 ext4 93M 1.6M 87M 2% /u04**

**/dev/mapper/test-lv05 ext4 93M 1.6M 87M 2% /u05**

52. List all entries with extension ".sh"

Ls -l \*.sh

53. List all entries with numbers in it.

Ls –l \*[0-9]

54. List all entries that starts with a character and ends with a number

Ls – l [a-z]\*[0-9]

55. List all entries that name length more than 5 characters

Ls –d ./????

56. What is Quoting? and Why do we need it?

The double quotes allowes to print the value of $SHELL variable, disables the meaning of [wildcards](https://bash.cyberciti.biz/guide/Wildcards), and finally allows command substitution.  
echo "$SHELL"  
echo "/etc/\*.conf"  
echo "Today is $(date)"

The single quotes prevents displaying variable $SHELL value, disabled the meaning of [wildcards](https://bash.cyberciti.biz/guide/Wildcards) /etc/\*.conf, and finally command substitution ($date) itself.   
echo '$SHELL'  
echo '/etc/\*.conf'  
echo 'Today is $(date)'

You can use \ before dollar sign to tell the shell to have no special meaning. Disable the meaning of the next character in $PATH (i.e. do not display value of $PATH variable):  
echo "Path is \$PATH"  
echo "Path is $PATH"

57. Write few(minimum 3) unique examples that shows, how a particular problem is solved using Quoting.

echo '$SHELL'  
echo '/etc/\*.conf'  
echo 'Today is $(date)'

echo "Path is \$PATH"  
echo "Path is $PATH"

echo "$SHELL"  
echo "/etc/\*.conf"  
echo "Today is $(date)"

58. How do you find a particular files/directories based on a particular search criteria?

HINT:- look for commands -> locate, find and whereis

$ locate “\*.html” –n 200

$find . –type f –name “\*.php”

59. Write major difference between locate, find and whereis?

The find program searches a directory tree to find a file or group of files. It traverses the directory tree and reports all occurrences of a file matching the user's specifications. The find program includes very powerful searching capability.

The locate program scans one or more databases of filenames and displays any matches. This can be used as a very fast find command if the file was present during the last file name database update.

So find is what you use when you want to search by particular criteria and also manipulate files. It has many more options than locate so allows far more fine-grained control of results. It is slow because it performs the requested test(s) on every file to see if it matches.

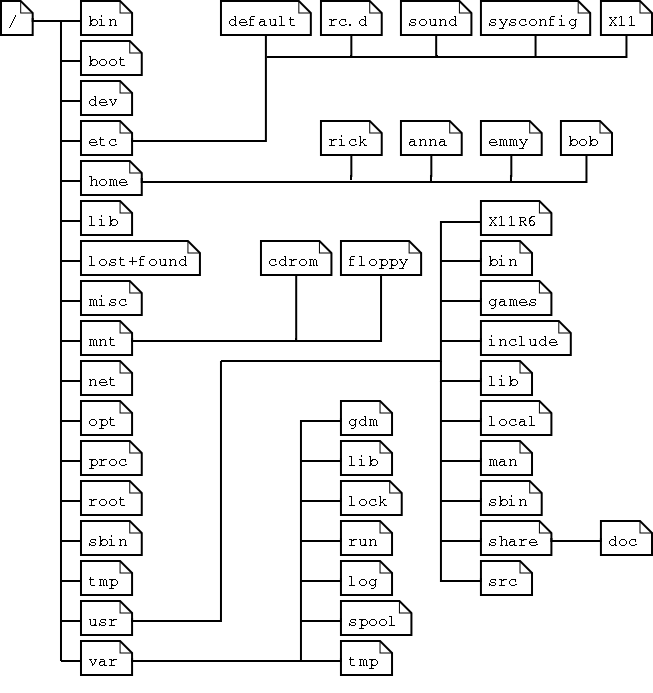
locate is used to scan the whole system quickly for something - you might use this when you have no idea where something is, or when you want to find all related files scattered across various unknown places. It's fast because it uses a binary database to index the system. To get new files to show up, first run sudo updatedb (the database it updated once per day by cron

the whereis command simply returns the location of the executables, the man pages and the sources of a program (see man whereis)

60. How Globbing is different from locate, find and whereis?

61. Explain the Linux File System.

| **Symbol** | **Meaning** |
| --- | --- |
| - | Regular file |
| d | Directory |
| l | Link |
| c | Special file |
| s | Socket |
| p | Named pipe |
| b | Block device |



| **Directory** | **Content** |
| --- | --- |
| /bin | Common programs, shared by the system, the system administrator and the users. |
| /boot | The startup files and the kernel, vmlinuz. In some recent distributions also grub data. Grub is the GRand Unified Boot loader and is an attempt to get rid of the many different boot-loaders we know today. |
| /dev | Contains references to all the CPU peripheral hardware, which are represented as files with special properties. |
| /etc | Most important system configuration files are in /etc, this directory contains data similar to those in the Control Panel in Windows |
| /home | Home directories of the common users. |
| /initrd | (on some distributions) Information for booting. Do not remove! |
| /lib | Library files, includes files for all kinds of programs needed by the system and the users. |
| /lost+found | Every partition has a lost+found in its upper directory. Files that were saved during failures are here. |
| /misc | For miscellaneous purposes. |
| /mnt | Standard mount point for external file systems, e.g. a CD-ROM or a digital camera. |
| /net | Standard mount point for entire remote file systems |
| /opt | Typically contains extra and third party software. |
| /proc | A virtual file system containing information about system resources. More information about the meaning of the files in proc is obtained by entering the command **man *proc*** in a terminal window. The file proc.txt discusses the virtual file system in detail. |
| /root | The administrative user's home directory. Mind the difference between /, the root directory and /root, the home directory of the *root* user. |
| /sbin | Programs for use by the system and the system administrator. |
| /tmp | Temporary space for use by the system, cleaned upon reboot, so don't use this for saving any work! |
| /usr | Programs, libraries, documentation etc. for all user-related programs. |
| /var | Storage for all variable files and temporary files created by users, such as log files, the mail queue, the print spooler area, space for temporary storage of files downloaded from the Internet, or to keep an image of a CD before burning it. |

62. Explain absolute and Relative Paths

An absolute path is defined as the specifying the location of a file or directory from the root directory(/).

cat /home/kt/abc.sql

Relative path is defined as the path related to the present working directly(pwd). It starts at your current directory and **never starts with a /**.

**.(a single dot)** - this represents the current directory.

**..(two dots)** - this represents the parent directory.

**Changing directory with relative path concept :**

$pwd

/home/kt

$cd abc

$pwd

/home/kt/abc

**Changing directory with absolute path concept:**

$pwd

/home/kt

$cd /home/kt/abc

$pwd

/home/kt/abc

63. What are the different ways of creating a File in linux System? Write an example of each and the difference between them.

64. In how many ways we can delete the files from linux system? write an example of each and teh difference between them.

65. Archiving files using linux command, write a command to archive set of files from linux commands.

66. Extract the archived files from the above step.