Name: Shalini Das

List out at least 20 basic Linux commands and identify its purpose in Linux.

1. **ls — List directory contents**

ls is probably the most common command. The ls command allows you to quickly view all files within the specified directory.

- Syntax: ls [option(s)] [file(s)]
- Common options: -a, -l

2. **echo — Prints text to the terminal window**

echo prints text to the terminal window and is typically used in shell scripts and batch files to output status text to the screen or a computer file. Echo is also particularly useful for showing the values of environmental variables, which tell the shell how to behave as a user works at the command line or in scripts.

- Syntax: echo [option(s)] [string(s)]
- Common options: -e, -n

3. **touch — Creates a file**

touch is used to create new files, but it can also be used to change timestamps on files and/or directories. We can create as many files as you want in a single command without worrying about overwriting files with the same name.

- Syntax: touch [option(s)] file_name(s)
- Common options: -a, -m, -r, -d

4. **mkdir — Create a directory**

mkdir is a useful command you can use to create directories. Any number of directories can be created simultaneously which can greatly speed up the process.

- Syntax: mkdir [option(s)] directory_name(s)

- Common options: -m, -p, -v

5. **pwd — Print working directory**

pwd is used to print the current directory we're in.

- Syntax: pwd [option(s)]
- Common options: options aren't typically used with pwd

6. **cd — Change directory**

cd will change the directory we're in so that you can get info, manipulate, read, etc. the different files and directories in your system.

- Syntax: cd [option(s)] directory
- Common options: options aren't typically used with cd

7. **rmdir — Remove directory**

rmdir will remove empty directories. This can help clean up space on your computer and keep files and folders organized. There are two ways to remove directories: rm and rmdir. The distinction between the two is that rmdir will only delete empty directories, whereas rm will remove directories and files regardless if they contain data or not.

- Syntax: rmdir [option(s)] directory_names
- Common options: -p

8. **locate — Locate a specific file or directory**

This is by far the simplest way to find a file or directory. We can keep our search broad if we don't know what exactly it is you're looking for, or you can narrow the scope by using wildcards or regular expressions.

- Syntax: locate [option(s)] file_name(s)
- Common options: -q, -n, -i

9. **cat — Read a file, create a file, and concatenate files**

cat is one of the more versatile commands and serves three main functions: displaying them, combining copies of them, and creating new ones.

- Syntax: cat [option(s)] [file_name(s)] [-] [file_name(s)]
- Common options: -n

10. **chmod — Sets the file permissions flag on a file or folder**

There are situations where someone will try to upload a file or modify a document but receive an error because they don't have access. The quick fix for this is to use chmod. Permissions can be set with either alphanumeric characters (u, g, o) and can be assigned their access with w, r, x. Conversely, you can also use octal numbers (0-7) to change the permissions. For example, chmod 777 my_file will give access to everyone.

- Syntax: chmod [option(s)] permissions file_name
- Common options: -f, -v

11. **exit — Exit out of a directory**

The exit command will close a terminal window, end the execution of a shell script, or log you out of an SSH remote access session.

- Syntax: exit
- Common options: n/a

12. **history — list your most recent commands**

An important command when you need to quickly identify past commands that you've used.

- Syntax: history
- Common options: -c, -d

13. clear — Clear your terminal window

This command is used to clear all previous commands and output from consoles and terminal windows. This keeps the terminal clean and removes the clutter so that we can focus on subsequent commands and their output.

- Syntax: clear
- Common options: n/a

14. cp — copy files and directories

Use this command when you need to back up your files.

- Syntax: cp [option(s)] current_name new_name
- Common options: -r, -i, -b

15. kill — terminate stalled processes

The kill command allows you to terminate a process from the command line. You do this by providing the process ID (PID) of the process to kill. To find the PID, you can use the ps command accompanied by options -aux.

- Syntax: kill [signal or option(s)] PID(s)
- Common options: -p

16. sleep — delay a process for a specified amount of time

sleep is a common command for controlling jobs and is mainly used in shell scripts. You'll notice in the syntax that there is a suffix; the suffix is used to specify the unit of time whether it be s (seconds), m (minutes), or d (days). The default unit of time is seconds unless specified.

- Syntax: sleep number [suffix]
- Common options: n/a

17. mv — Move or rename directory

mv is used to move or rename directories. Without this command, you would have to individually rename each file which is tedious. mv allows you to do batch file renaming which can save you loads of time.

- Syntax: mv [option(s)] argument(s)
- Common options: -i, -b

18. man — Print manual or get help for a command

The man command is your manual and is very useful when you need to figure out what a command does.

Syntax: man [option(s)] keyword(s)

- Common options: -w, -f, -b

19. less — view the contents of a text file

The less command allows you to view files without opening an editor.

- Syntax: less file_name
- Common options: -e, -f, -n

20. grep — search

grep is used to search text for patterns specified by the user. It is one of the most useful and powerful commands.

Syntax: grep [option(s)] pattern [file(s)]

- Common options: -i, -c, -n