Linux commands:

Linux is case sensitive for example: our directory name is Desktop, if we write the name in desktop like this , it showing an error like no such directory is available.

if we want to clear the terminal by using (clear command) or simply we can click (ctrl + n+ L)

1. ls – List directory contents(def: lists the content of any directory) instead of folder we use directory

Directory Def: a **directory** (often called a **folder**) is like a **container** on your computer or in a file system where you can **store** and **organize** your files.

Shows the files and directories in the current directory.

ls

Options:

- ls -l for detailed information. Of directory (like date name month)(in blue color are directories, white color is files)
- ls -a to show hidden files.
- Ls - la(it shows long listing and including hidden files) (hidden files: the files start with dot(.))

2. cd – Change directory(cd ..) (back to the directory by using cd)

Used to navigate between directories.

```
cd /path/to/directory
```

To go to the home directory:

```
cd ~
```

3. pwd – Print working directory(present working directory)(it tells in which directory we r working currently)

Displays the full path of the current directory.

```
pwd
```

4. cp – Copy files and directories

Used to copy files or directories.

```
cp source_file destination_file
cp -r source_dir destination_dir  # For directories
```

[—-- if we want to copy a file into another directory by using(cp filename / directory name)

—- if you are not a root user u can simply use (sudo cp filename / directory name)

— see the file is copied or not by using (cd folder name)

— move folder to another folder by using (mv f4 /f5)

– if u r not a root user u can simply use (sudo mv f4 / f5)]

5. mv – Move or rename files and directories

Moves or renames files and directories.

```
mv old_name new_name
mv file /path/to/destination/
```

6. rm – Remove files and directories

Deletes files or directories.

```
rm file
rm -r directory  # Remove directory and its contents
```

7. touch – Create an empty file

Creates an empty file or updates the timestamp of an existing file.

```
touch filename.txt
```

8. mkdir – Make a directory(creating a directory)(creates multiple folders at a time by using space between names, for example i have to create 5 folders at a time f1 space f2 space f3 space f4 space f5 space)(if we see our files or created or not by using (ls) command)

Creates a new directory.

```
mkdir new_directory
```

9. rmdir – Remove an empty directory

Deletes an empty directory.

```
rmdir directory_name
```

10. cat – Concatenate and display file content(creates some data in it by using cat , after writing some content click (ctrl + z) the file will be saved)

Displays the contents of a file.

```
cat file.txt
```

1.If we want to see what we have to write in file by using cat filename

2. If we want to edit that file(add something to that file) by using cat >> filename

11. more and less – View file content

Displays a file's content one screen at a time.

```
more file.txt
less file.txt  # Allows scrolling back
```

12. head – Display the first few lines of a file

Shows the first 10 lines of a file by default.

```
head file.txt
head -n 20 file.txt  # Display the first 20 lines
```

13. tail – Display the last few lines of a file

Shows the last 10 lines of a file by default.

```
tail file.txt
tail -n 20 file.txt  # Display the last 20 lines
tail -f file.txt     # Follow a log file in real-time
```

14. chmod – Change file permissions

Changes the permissions of a file or directory.

```
chmod 755 file.txt
chmod +x script.sh  # Make a script executable
```

15. chown – Change file owner and group

Changes the owner and/or group of a file or directory.

```
chown user:group file.txt
```

16. find – Search for files

Searches for files and directories based on various criteria.

```
find /path -name "file_name"
find . -name "*.txt"  # Find all .txt files in current directory and subdirectories
```

17. grep – Search for a pattern in files

Searches for specific patterns in files or outputs.

```
grep -r "pattern" /path  # Search recursively in directories
```

18. ps – Report process status

Shows the running processes.

```
ps aux
ps -ef  # Detailed process list
```

19. kill – Terminate a process

Terminates a running process by PID (Process ID).

```
kill <PID>
kill -9 <PID>  # Forcefully terminate a process
```

20. df – Disk space usage

Shows the disk space usage of file systems.

```
df -h  # Human-readable format (e.g., GB, MB)
```

21. find:

Purpose: find is used to search for files and directories based on certain conditions like name, size, type, and time.
Why we use it: It helps us locate specific files or directories quickly.
Where we use it: When you need to search for files in a directory or across the system.
Example: find /home/user -name "*.txt" (find all .txt files in /home/user).
Syntax:
find [path] [conditions]
find /home/user -name "*.txt"

22. ln 2 types: hard links, soft links
Purpose: ln creates links to files.
Hard Link: Another name for the same file content. It’s like creating an extra pointer to the same file data.
Soft Link (Symbolic Link): A shortcut or pointer to another file or directory.
Why we use it: Hard links keep data accessible through multiple filenames, while soft links can point to files or directories located elsewhere.
Where we use it: Use it to create shortcuts (soft link) or duplicate file references (hard link) without duplicating the actual content.
Example:
ln file1.txt link1.txt (hard link).
ln -s /path/to/file symlink.txt (soft link).
Syntax: Hard link
ln [source] [link_name]

ln file1.txt file2.txt
Syntax: Soft link
ln -s [source] [link_name]
Example:
ln -s /path/to/file symlink.txt

23. gzip
Purpose: gzip is used to compress files, making them smaller in size for storage or transfer.
Why we use it: To save disk space or reduce file size when sending over the network.
Where we use it: Compress files to send via email, store backups, or save space on a server.
Example: gzip file.txt (compress file.txt to file.txt.gz).
Syntax:
gzip [file]
Example:
gzip file.txt

24. gunzip
Purpose: gunzip is used to decompress files compressed using gzip.
Why we use it: To recover the original file from a compressed .gz file.
Where we use it: When you need to restore a compressed file.
Example: gunzip file.txt.gz (decompresses file.txt.gz back to file.txt).
Syntax:

gunzip [file.gz]
Example:
gunzip file.txt.gz

25.tar

Purpose: tar is used to archive multiple files into a single file and optionally compress them.
Why we use it: To create backups or combine multiple files into one file for easier storage or transfer.
Where we use it: When creating a compressed archive for backup or distribution.
Example: tar -czvf archive.tar.gz directory/ (create a compressed archive).
Syntax:
tar [options] [archive_name] [files or directories]
Example:
tar -czvf archive.tar.gz /path/to/dir

- - - - - -. alias - - - - - - - -

Purpose: alias allows you to create shortcuts for commands you use often.
Why we use it: To make long or complex commands easier to type.
Where we use it: When working in the terminal to save time and increase efficiency.
Example: alias ll='ls -l' (create a shortcut ll for ls -l).
Syntax:

alias [name]='[command]'
Example:
alias ll='ls -l'

- - - - - - WC - - - - - - - - -

Purpose: wc counts the lines, words, and characters in a file.
Why we use it: To get basic statistics about a file or input data.
Where we use it: When you need to check the size of a file or output from another command.
Example: wc file.txt (returns the number of lines, words, and characters in file.txt).
Syntax:
wc [file]
Example:
wc file.txt

- - - - - - sort - - - - - - - - - -

Purpose: sort arranges the content of a file or output in ascending or descending order.
Why we use it: To organize data (e.g., sorting a list alphabetically or numerically).
Where we use it: When working with lists of data, like names or numbers, that need to be sorted.
Example: sort file.txt (sort the contents of file.txt).
Syntax:
sort [file]
Example:

sort file.txt

- - - - - - uniq - - - - -

Purpose: uniq removes duplicate lines from a file.
Why we use it: To clean up data by removing repeated lines.
Where we use it: When processing files or command output that may contain duplicates.
Example: uniq file.txt (removes duplicates in file.txt).
Syntax:
uniq [file]
Example:
uniq file.txt

- - - - - diff - - - - -

Purpose: diff compares two files and shows the differences between them.
Why we use it: To see changes between two versions of a file.
Where we use it: In programming or document editing to compare different versions of a file.
Example: diff file1.txt file2.txt (shows the differences between file1.txt and file2.txt).
Syntax:
diff [file1] [file2]
Example:
diff file1.txt file2.txt

- - - - - - echo - - - - - -

Purpose: echo displays text or outputs the result of commands.
Why we use it: To display messages or command output in the terminal.
Where we use it: In scripts or directly in the terminal to display text or variables.
Example: echo "Hello, World!" (prints "Hello, World!" on the screen).

Syntax:

```
echo [text]
```

Example:

```
echo "Hello, World!"
```

- - - - - - umask - - - - -

Purpose: umask sets default file permissions for newly created files and directories.
Why we use it: To control how readable, writable, or executable files are by default when they are created.
Where we use it: In system administration to ensure the correct default permissions for new files.
Example: umask 022 (sets default permissions).

Syntax:

```
umask [permissions]
```

Example:

```
umask 022
```

- - - - - - du - - - - - -

Purpose: du shows the disk usage of files and directories.

Why we use it: To see how much space a file or directory takes up.
Where we use it: When you want to monitor disk space usage on your system.
Example: du -sh /path/to/dir (shows the size of a directory).
Syntax:
du [options] [path]
Example:
du -sh /path/to/dir

- - - - - -basename - - - - -

Purpose: basename strips the directory path from a file, leaving just the file name.
Why we use it: To extract the file name from a full path.
Where we use it: When you need just the name of a file for processing in a script or command.
Example: basename /home/user/file.txt (returns file.txt).
Syntax:
basename [path]
Example:
basename /home/user/file.txt

- - - - - -dirname - - - - - -

Purpose: dirname strips the file name from a path, leaving only the directory path.
Why we use it: To extract the directory path from a file path.
Where we use it: When you need the directory part of a file path

in a script.
Example: dirname /home/user/file.txt (returns /home/user).

Syntax:

```
dirname [path]
```

Example:

```
dirname /home/user/file.txt
```

- - - - - PS - - - - - -

Purpose: ps shows information about running processes on the system.
Why we use it: To monitor and manage processes.
Where we use it: When you need to see which processes are running.
Example: ps aux (shows all running processes).

Syntax:

```
ps [options]
```

Example:

```
ps aux
```

- - - - - - top - - - -

Purpose: top provides a real-time view of the system's resource usage and running processes.
Why we use it: To monitor system performance, like CPU and memory usage.
Where we use it: On servers or systems when you need to observe live system resource usage.
Example: top (displays live process information).

Syntax:

top

Example:

top

- - - - - - killall- - - - -

Purpose: kill or killall is used to terminate running processes.
Why we use it: To stop processes that are no longer needed or are causing problems.
Where we use it: When you need to stop a program or process on your system.
Example: killall firefox (kills all instances of Firefox).

Syntax:

killall [process_name]

Example:

killall firefox

- - - - - - bg- - - - - -

Purpose: bg resumes a paused job in the background.
Why we use it: To allow a process to continue running without blocking the terminal.
Where we use it: When you need to pause a process and continue working in the terminal.
Example: bg %1 (resume job 1 in the background).

Syntax:

bg [job_number]

Example:

bg %1

- - - - - - - -fg- - - - - -

Purpose: fg brings a background job back to the foreground.
Why we use it: To interact with a job that was previously running in the background.
Where we use it: When you need to continue working with a background process.
Example: fg %1 (bring job 1 to the foreground).
Syntax:
fg [job_number]
Example:
fg %1

- - - - - - - - - type- - - - - -

Purpose: type tells you whether a command is a builtin, function, or external command.
Why we use it: To understand how a command will be interpreted by the shell.
Where we use it: When you want to know if a command is internal or external.
Example: type ls (shows that ls is an external command).
Syntax:
type [command]
Example:
type ls

- - - - - -which - - - - - -

Purpose: which shows the path of an executable command.
Why we use it: To find where a command is located on your system.
Where we use it: When you need to know the location of an executable.
Example: which ls (shows where ls is located).
Syntax:
which [command]
Example:
which ls

- - - - - -vim - - - - -

Purpose: vim is a powerful text editor for editing files in the terminal.
Why we use it: To edit text files with advanced features like syntax highlighting and multi-level undo.
Where we use it: In programming, system administration, and editing configuration files.
Example: vim file.txt (opens file.txt in vim).
Syntax:
vim [file]
Example:
vim file.txt

- - - - - - emacs - - - - -

Purpose: A text editor for creating and editing files.

Why Use: Used to write code, edit configuration files, and more.
Where Used: In the terminal for advanced text editing.
Syntax:
emacs filename
Example:
emacs mydogs.txt
This opens mydogs.txt in Emacs.

- - - - - nano - - - - - - - -

Purpose: A simple text editor for the terminal.
Why Use: Ideal for quick editing without needing complex features.
Where Used: In the terminal to edit text or configuration files.
Syntax:
nano filename
Example:
nano myfile.txt
This opens myfile.txt in Nano.

- - - - - - - whoami - - - - - -

Purpose: Displays the current user's name.
Why Use: To quickly check which user is logged in.
Where Used: In the terminal to confirm your login identity.
Syntax:
whoami
Example:
whoami

This shows the current logged-in user, for example alice.

- - - - - - - who - - - - - -

Purpose: Shows who is currently logged into the system.

Why Use: To see who else is using the system.

Where Used: In the terminal to view logged-in users.

Syntax:

```
who
```

Example:

```
who
```

This shows a list of users logged in.

- - - - - - su - - - - - -

Purpose: Switch to another user (often used to switch to root).

Why Use: To perform administrative tasks by logging in as a different user.

Where Used: In the terminal to switch users.

Syntax:

```
su [username]
```

Example:

```
su root
```

This switches to the root user.

- - - - - - -sudo - - - - - - -

Purpose: Run commands with superuser (root) privileges.

Why Use: To perform actions that require administrative rights.

Where Used: In the terminal when performing tasks like installing software.

Syntax:

```
sudo command
```

Example:

```
sudo apt update
```

This updates the package list with administrative rights.

- - - - - - passwd - - - - -

Purpose: Change your password or another user's password.

Why Use: To update or reset your password.

Where Used: In the terminal to modify passwords.

Syntax:

```
passwd [username]
```

Example:

```
passwd
```

This changes the password for the current user.

- - - - - - ping - - - - - -

Purpose: Check if a network server or website is reachable.

Why Use: To test internet connectivity or if a server is online.

Where Used: In the terminal for network troubleshooting.

Syntax:

```
ping [hostname or IP address]
```

Example:

```
ping google.com
```

- - - - - open - - - - - -

Purpose: Opens files or URLs in their default application (mostly used on macOS).
Why Use: To open files or websites directly from the terminal.
Where Used: On macOS in the terminal for file opening.
Syntax:
open [file or URL]
Example:
open myfile.pdf
This opens myfile.pdf with the default PDF viewer.

- - - - - - jobs - - - - - -

Purpose: Lists background jobs (processes) in the current session.
Why Use: To view and manage background tasks.
Where Used: In the terminal when you have background tasks running.
Syntax:
jobs
Example:
jobs
This lists all the background jobs.

- - - - - - clear - - - - - - -

Purpose: Clears the terminal screen.
Why Use: To clean up the terminal and get rid of previous output.
Where Used: In the terminal to refresh the screen.

Syntax:

```
clear
```

Example:

```
clear
```

This clears the terminal window.