Use 'tldr' command to get quick info about any command.
https://www.freecodecamp.org/news/the-linux-commands-handbook/

1. **man**:
   Doc for any command (complete documentation)
   Use tldr <command> (to get short and quick info about any command),
   Install : sudo apt install tldr

2. **ls**: list files
   List content of any directory

3. **cd**: change directory
   Move from one directory to another
   Special indicators are –
   a. Period (.) = current working directory
   b. Double period (..) = parent directory of current working directory
   Cd ~<username> : cd ~aryantapre (moves to user directory)

4. **pwd** : print current working directory path (absolute)
   aryantapre@frontman: pwd
   /media/aryantapre/nvme0n1p6

5. **mkdir** : creating fresh directories
   Options
   A. -m | --mode = give permissions
   Execute: 1
   Read: 4
   Write: 2

6. **rmdir**: remove directories

7. **rm**: remove files
   Options
   Rm -rf (remove forcefully)

8 **mv** : move or rename files / directories

9. **cp:** copy files / directories

10. **open** : open files / URL

Use 'tldr' command to get quick info about any command.
https://www.freecodecamp.org/news/the-linux-commands-handbook/

open x.png
Open 'https://google.com'

11. touch : create an empty file

Rsync: advanced copy function.

Powerful & efficient copying and synchronozing file and directores
In remote locatio ns.
algorithm to copy only the differences between source and destination
Files,

Options:

`-r`
Recursively copy directories and their contents.
`-a`
Archive mode; preserves symbolic links, permissions, timestamps, etc.
`-v`
Verbose output; shows detailed progress during the transfer.
`-z`
Compresses file data during transfer (useful for remote sync).
`-P`
hwCombines `--progress` (shows progress) and `--partial` (keeps partial files).
`--delete`
Deletes files in the destination that are no longer in the source.
`-e`
Specifies the remote shell (e.g., `-e ssh` for SSH).
`--exclude`
Excludes specific files or patterns from syncing (e.g., `--exclude='*.log'`).

12. find : search for files / folders matching a particular searching pattern (regex)
tldr find

* = all chars matching
? = single char matching
[char]
[!char]
[[:class:]]

Use 'tldr' command to get quick info about any command.
https://www.freecodecamp.org/news/the-linux-commands-handbook/

13. **ln** : create hard and soft links (symbolic links)
Limitations of hard link
- can't connect outside file system
- can't connect to directory

Soft links : overcomes limitation of hard link
Syntax: ln -s <original-file> <linked-file>

14. **gzip :** compress file using LZ77 protocol.
By default it deletes original after compressing
gzip -k : keep original after *compressing*
gzip - <range> : from 1 to 9 (low-high) compression level
gzip -v : defines percentage level

The best file compression utility is 'xz' command

15. **tar** : used to archive files into single one
To archive
Syntax: tar -cf <archieve_file_name> file1 file2 (cf =create file)
To extract
Syntax: tar -xf <archive_file_name> –directory=path / . (xf = extract file)
E.g
tar -cf aryan.tar file1 file2
Tar -xf aryan.tar –directory=data/. (extracting onto data directory)

16. **Alias:** used to create aliases of commands
Like for *ls -la* we can create the following–
Syntax: alias <name> = ' expression/ command '
alias ll='ls -la'

17. **Cat:** creates a new file, concat multiple files into one
Cat file1 file2 >> file3 (append file1 & file2 to file3)
Creating new file
Cat > <file_name.extension>

18. **less:** shows content of file in interactive mode
Syntax: less <file_name>
To search : use forward slash

Use 'tldr' command to get quick info about any command.
https://www.freecodecamp.org/news/the-linux-commands-handbook/

19. **tail:** open file from last *(mostly used to see log files)*
Argument: -f (any time there is new content in file, printed in window)
E.g tail -f data.txt

20. **wc:** determines supportive information of file like no of lines, no of words, no of Bytes etc
Syntax: wc –lines –words –bytes <file_name>
Even use it with pipe operator with other command
Ls -la | wc

21. **grep :** used to search in files, combine it with pipe to filter the output of another Command.
Syntax: grep <regex> or <search_pattern> <file_name>
*grep *.pdf ./destinations*
-r -n (recursive, show line number)
~by default search in case insensitive, use -i for sensitive…

22. **sort:** used to sort data
By default sort is in ascending order
Arguments:
-r = for reverse sorting i.e descending order
Syntax: sort -r <file_name> (sort in reverse…)

23. **Diff:** determine difference in files , directories and etc
Syntax: diff <options> <files_name1> <file_name2> ..
Arguments: -y (compare files side-by-side)
-u (compare files as git does — ++++ and so on..)
-r (recursively compares files in directories)

24. **echo :** print to output the arguments passed to it.
echo " hello world "
echo "my path is $PATH" (interpolate variable)

Printing… files of directory
*echo * (print all files)*
echo ~aryantapre (print home directory of user)

25. **Chown:** used to define ownership to file / directory
UNIX system has Ownership for every file and directory
I.e root, group, others

Use 'tldr' command to get quick info about any command.
https://www.freecodecamp.org/news/the-linux-commands-handbook/

Syntax: chown <root_user>:<group_name> <file_name>
Arguments:
-R, -h (recursively, symbolic link) change ownership of directories
ls -la
Chown root:aryantapre sample.txt

26. Chmod: changes permission to existing file / directories in context of owner, group and others users

2 ways to use chmod:

A. symbolic
a = stands for all
u = stands for user
g = stands for group
o = stands for others

Use + to add , and - to remove permissions
r=read, w=write, x= execute

Syntax: chmod a+rwx <file_name> (read, write and execute permissions to all).

Arguments -r (for recursively apply permissions to directories and sub-directories)

B. Numeric:
Permissions: 1+4+2 = 7 (read, write and execute..)
Execute - 1
Read - 4
Write - 2

Syntax : chmod <owner,group,users> <file_name>
E.g chmod 777 sample.txt (read, write and execute)

27. Umask: it sets the default permissions for newly created file / directories.
Works only for current session, permission will not sustain for every OS boot.
Syntax: umask <numeric-permission> default; 002
Default permission

Use 'tldr' command to get quick info about any command.
https://www.freecodecamp.org/news/the-linux-commands-handbook/

~ file = 666
~ directory = 777

Working,
Generally umask subtracts some values from default to set Permissions based on file / directory.

1 = execute, 4 = read, 2 = write
Changing permission for file to rw
666 - 000(mask) = 666 (4+2= rw)
To get rw i.e 6 need to subtract 0(mask) from default

Umask 000

28. **du :** calculate the size of directory as a whole
*Syntax: du <options> <directory_name / *>*
Options:
-m = display values in MB
-g = display values in GB
-h = display human-readable notations

Du -m ./destination

29. **Df:** define current disk usages
Syntax: *df <options>*
Options:
-h = human-readable format
cd /
cd media/aryantapre
*df -h **

30. **basename:** returns last segment of path to file / directory.
Path = /media/nvme0n1p6/DSA/stack.cpp
Basename /media/nvme0n1p6/DSA/stack.cpp
It will return stack.cpp

31. **dirname:** returns directories segment of path to file / directory.
Path = /media/nvme0n1p6/DSA/stack.cpp
Basename /media/nvme0n1p6/DSA/stack.cpp
It will return **/media/nvme0n1p6/DSA**

Use 'tldr' command to get quick info about any command.
https://www.freecodecamp.org/news/the-linux-commands-handbook/

32. **ps:** list all running processes of computer
Default: list processes of current session only (Terminal)
Options
-ax = (also list other user process, show processes not linked to terminal)

Pid = process id
TT = terminal id used
STAT = state of the process

I = a process is idle (sleeping for longer than 20 seconds)
R = runnable process
S = sleeping less than 20 seconds
T = stopped process
U = uninterruptible
Z = dead process (zombie)

+ (process is in foreground in its terminal)
s (session leader)

TIME = how long process is being running.

33. **top:** used to display dynamic real time information of all running processes in the system

34. **htop:** an interactive top will all features, better graphics

35. **kill:** linux processes receive signal and react based on them
Syntax: kill <signal> <process_id>
Signals:
HUP= hangsup process automatically send when terminal window that Started process get closed before terminating the process.
INT = means interrupt, it sends signal when we press ctrl + c in terminal, Which usually terminate the process.
KILL = immediately kernel terminates the process.
TERM = process does self destruction (TERM = terminate)
CONT = continue to run process
STOP = pause / STOP the process kernel does it.

36. **killall:** similar to kill, sends signals to multiple processes all at once.
Syntax: killall <name>

Use 'tldr' command to get quick info about any command.
https://www.freecodecamp.org/news/the-linux-commands-handbook/

37. **jobs:** used to run a command in background using '&' symbol after command

E.g = *top &*

*If we run **top &** and **top -o mem &** we have two top instances running right ?*

Jobs (displays all running background jobs..)
*We can switch to job using **fg <job_id>***
Stop program = hit CTRL + Z

Jobs -l = print processID for each job

38. **bg:** resume suspended jobs (e.g using ctrl + Z), and keeps them running in the background.

Syntax: bg <job_id>
Before that get jobID using jobs -l
Bg = background

39. **fg :** runs command in foreground, that is already running in background using or 'bg' command.

Syntax: fg <job_id>

40. **type:** display type of command shell will execute

Command in MacOS / LInux OS are of 4 types

a. executable
b. shell built-in program
c. shell function
d. Alias

Syntax: type <command_name>
type ls

41. **Which:** locates program in users path

Only works with User associated programs not built-in
Syntax: which <command_name>
Which google-chrome

42. **nohup:** allows process to live when terminal get killed

To run long -living process on a remote machine, you won't want command get Halted due to network issue
You want process should continue to run even after logout

Use 'tldr' command to get quick info about any command.
https://www.freecodecamp.org/news/the-linux-commands-handbook/

Syntax: nohup <command>

43. **xargs:** output of one command is used as input of another command.
Used using pipe (|)
E.g we have three files file1.txt, file2.txt and file3.txt
In file todelete.txt we have names of these files as

Todelete.txt

file1.txt
file2.txt
file3.txt

cat todelete.txt | xargs -p rm -rf

Options

-p = print confirmation before executing action
-n = perform one iteration at a time , can individually confirm them

44. **Vim:** vim is one of the powerful editor of unix like system ubuntu / macOS
Has two main mode:
Command = where all commands are executed..
Insert = to insert text content to file

Arrow keys:

H-L = left and right
J-K = down and up

Commands:

:w = save the file
:wq = save and exit from file
U = undo
R = redo

X = deletes character currently highlighted
Capital A = goes to end of currently selected line
0 = goes to start of line
$ = goes to end of line.
Dw = deletes a single word.
D2w = deletes 2 words in forward (number can be any).
Dd = deletes entire line.

Use 'tldr' command to get quick info about any command.
https://www.freecodecamp.org/news/the-linux-commands-handbook/

Type vimtutor for full overview at terminal….

45. **whoami:** prints current user logged in to the terminal

46. **Who:** displays users logged to the system
Defines when did user logged in / when session was started.
Related data (process, boot time)

47. **su:** used to switch between user account
Syntax: su <username>
su aryantapre
su root

Setting new root password:
command = su passwd root

48. **sudo :**commonly used to run commands as root user

E.g sudo vim /etc/fstb

49. **passwd:** used to change passwords of user Account.|
Used in two way.
A. when you want to change your password (current user)
type 'passwd' command in terminal
B. you want to change password of other account , only possible if you Loggedin as 'root ' user
Syntax:
passwd <user_name> <new_password>

50. **ping:** used to check reachability of n/w host, on local n/w or the Internet
Sends ICMP ECHO_REQUEST and gets ICMP_REPLY back
Options
-c <number> = defines ping count
Ping commands sents ECHO_REQUEST every second until we STOP it
Using CTRL + C

51. **Traceroute:** gather all information of packet travelled from your local machine to Network host, prints onto the console.
Syntax:traceroute <options> <host-name> / <ip-address>
By default traceroute tries 3 times to get better indication of packets

Use 'tldr' command to get quick info about any command.
https://www.freecodecamp.org/news/the-linux-commands-handbook/

Options:

-q = count traceroute tries to reach host

traceroute -q 1 google.com

traceroute to google.com (142.250.192.110), 30 hops max, 60 byte packets
1 _gateway (192.168.38.203) 2.926 ms
2 *
3 10.0.242.109 (10.0.242.109) 44.478 ms
4 aes-static-253.85.22.125.airtel.in (125.22.85.253) 36.254 ms
5 182.79.141.205 (182.79.141.205) 50.252 ms
6 72.14.212.48 (72.14.212.48) 51.230 ms
7 *
8 142.250.214.100 (142.250.214.100) 50.203 ms
9 72.14.237.139 (72.14.237.139) 50.137 ms
10 192.178.110.107 (192.178.110.107) 77.824 ms
11 bom12s17-in-f14.1e100.net (142.250.192.110) 77.424 m

52. **clear:** clears the text written on screen, keep scrolling to get previous Content.

53. **History:** shows all the commands used before as 'history'

It memorized every command as 'history'

To clear History

Syntax: history -c

54. **export:** exports shell variables to child processes

Making shell variable available globally over any terminal session

Creating a variable

<variable_name> = "<value>"

E.g aryan="aryan"

export aryan

55. **Crontab:** Schedule cron jobs on a time interval for the current user.

Generally used on server to automate tasks, meaning running scripts automatically on specific time interval

Use 'tldr' command to get quick info about any command.
https://www.freecodecamp.org/news/the-linux-commands-handbook/

Syntax: crontab -l

56. Uname: prints information about current machine and OS running on it

uname without any arguments prints OS codename

Options:

A. -m = hardware name
B. -p = processor architecture
C. -s = OS name
D. -r = release
E. -v = version
F. -n = node network name (device name like 'frontman')
G. -a = prints all information

57. env: env command is used to pass environment variable without setting in external environment (current shell)

Suppose you want to run nodejs app and set some environment variable
You can run as

```
env USER=flavio node app.js
```

58. Printenv: used to print the values of all environment variables

Syntax: printenv

59. Id: displays current User and group identify

60. free: Display amount of free and used memory in the system…

Syntax: free <options>
Options:
Free - b | k | m | g (bytes, KB, MB, GB)

61. mount: used to mount a storage media to file system tree

e.g mount /dev/nvme0n1p7

62. umount: used to unmount a storage media from a file system tree

e.g umount /dev/nvme0n1p7

63. fdisk: Manage partition tables and partitions on a hard disk.

Use 'tldr' command to get quick info about any command.
https://www.freecodecamp.org/news/the-linux-commands-handbook/

========== CURSOR MOVEMENTS==========

1. **CTRL+a** = move cursor at start of line
2. **CTRL+e** = move cursor at end of line
3. **CTRL+f** = move cursor forward character-by-character
4. **CTRL+b** = move cursor backward character-by-character
5. **ALT+F** = move cursor forward word-by-word
6. **ALT+B** = move cursor backward word-by-word

ALTERING TEXT

1. **CTRL+d** = Deletes single character highlighted
2. **ALT+d** = Deletes/cuts single word highlighted
3. **CTRL+w** = deletes a word before cursor

4. **ALT+u** = convert text to uppercase.
5. **ALT+l** = convert text to lowercase.

6. **CTRL+k** = Cut everything after the cursor (stores it in a buffer).
7. **CTRL+y** = Copy/ yank what you just cut (yank it back).
8. **CTRL + p =** Paste whatever is cutted / copied.

EDITING MULTIPLE FILES IN SINGLE VIM SESSION

1. **Opening multiple files at once:**
   Syntax: vim file1 file2 file3 fileN
   (This opens specified files in buffers)

2. **Switching between files**

Use 'tldr' command to get quick info about any command.
https://www.freecodecamp.org/news/the-linux-commands-handbook/

Syntax: :bn = move to next file (Buffer Next)
:bp = move to previous file (Buffer Previous)

3. **List all open files**
Syntax: :buffers

4. **Switch to specific buffer**
Syntax: buffer N (where 'N' is buffer number)

SEARCH & REPLACE IN ENTIRE FILE

Syntax: :%s/<search_pattern>/<replace_text>/g

: = stands for exec command
% = defines range of lines for operation by-default set to 'Entire File'
s = operation name 'substitution' means search and replace
search_pattern = define the pattern/text to search
Replace_text = define the replacement text
G = global i.e entire file

TMUX

Tmux server
Session
Window
Pane

1. Attach and detach

A. tmux :
Start new tmux session (create default session number 0 with a single window)
; **b**You can even name a session as follows:
tmux new -s <session_name>

B. tmux a -t : <window_number>
Attach a specific window

Use 'tldr' command to get quick info about any command.
https://www.freecodecamp.org/news/the-linux-commands-handbook/

C. CTRL+ b d :
   Detach from tmux session, leaving it running in background
D. CTRL + b & :
   Exit and quit tmux
E. CTRL + b x :
   Kill / close a pane
F. CTRL + b ? :
   List all key bindings (press Q to exit help screen)

2. Window management

A. CTRL + b c :
   Create new window
B. CTRL + b n :
   Move to next window
C. CTRL + b p :
   Move to previous window
D. CTRL + b l :
   Move to last window
E. CTRL + b 0-9 :
   Move to window by index number
F. CTRL + b w :
   Open a panel to navigate across windows in multiple sessions

G. CTRL + b comma (,) :
   To rename a window

3. Session management

A. CTRL + b) :
   Move to next session
B. CTRL + b (:
   Move to previous session
C. CTRL + b :
   Suspend the session

4. Split windows into panes

A. CTRL + b % :
   Vertical split (panes side by side)

Use 'tldr' command to get quick info about any command.
https://www.freecodecamp.org/news/the-linux-commands-handbook/

B. CTRL + b " :
   Horizontal split (one pane below the other)
C. CTRL + b arrow_keys :
   To navigate b/w panes
D. CTRL + b o :
   Moves to other pane
E. CTRL + b CTRL-up/down:
   Resize current pane (due north/south)

F. CTRL + b CTRL-left/right :
   Resize current pane (due west/east)

5. Multilex

A. CTRL + b colon :
   Access tmux command prompt

6. Killing tmux (runs on Terminal only not inside tmux windows/panes**)**

A. kill-pane :
   Destroy a give pane
B. kill-server :
   Kill clients, sessions and server
C. kill-session :
   Destroy a given session
   tmux kill-session -t <session-name>
D. kill-window :
   Destroy a given window