Linux Basics Cheat Sheet

Shell Anatomy

```
jmorgan@lab01:~$  ls  -lah  /var/log
       1          2    3       4
```

1 prompt (user@host:dir) · 2 command · 3 options · 4 argument

`$ prompt`	You are a regular user. ~ means your home directory.
`# prompt`	You are root (full admin). Be careful: no undo, no confirmation.
`-l --long`	Short options use one dash and can stack (`-lah`). Long options use two dashes.
`UPPER ≠ upper`	Linux is case-sensitive: `File.txt` and `file.txt` are different files.

Getting Help

`man ls`	Full manual page. `/word` searches, `n` next match, `q` quits.
`ls --help`	Quick summary of options for most commands.
`info coreutils`	Longer GNU documentation, organized as linked nodes.
`whatis passwd`	One-line description of a command.
`apropos network`	Search manual descriptions by keyword (same as `man -k`).
`tldr tar †`	Community cheat sheet with practical examples.

Navigation

`pwd`	Print the working (current) directory.
`cd /etc`	Absolute path: starts at `/`, works from anywhere.
`cd docs/notes`	Relative path: starts from where you are now.
`cd or cd ~`	Go to your home directory.
`cd ..`	Up one level. `../..` goes up two.
`cd -`	Jump back to the previous directory.
`ls`	List files in the current directory.
`ls -l`	Long listing: permissions, owner, size, date.
`ls -a`	Include hidden dotfiles (`.bashrc`, `.ssh`).
`ls -h`	Human-readable sizes; everyday combo: `ls -lah`.
`ls -t`	Sort by modification time, newest first.
`ls -R`	List subdirectories recursively.
`tree -L 2 †`	Directory tree view, limited to 2 levels deep.

Filesystem Hierarchy (FHS)

Path	What lives there
`/`	Root of the entire filesystem tree.
`/bin`	Essential user programs (`ls`, `cp`). Linked to `/usr/bin` on modern distros.
`/sbin`	System admin programs (`fdisk`, `reboot`). Linked to `/usr/sbin`.
`/etc`	System-wide configuration files (text).
`/home`	Users' personal directories (`/home/jmorgan`).
`/root`	The root user's home directory.
`/var`	Variable data: logs, caches, mail, spool files.
`/var/log`	System and application log files.
`/tmp`	Temporary files; often cleared on reboot.
`/usr`	Installed software, libraries, and docs.
`/opt`	Optional third-party application packages.
`/dev`	Device files (`/dev/sda`, `/dev/null`).
`/proc`	Virtual files exposing kernel and process info.
`/boot`	Kernel and bootloader files. Don't touch casually.
`/mnt`	Temporary manual mount point.
`/media`	Auto-mounted removable media (USB drives).

Keyboard Shortcuts (Bash)

`Tab`	Auto-complete commands and paths. Press twice to see options.
`Ctrl+C`	Cancel (interrupt) the running command.
`Ctrl+D`	End of input; at an empty prompt it logs out.
`Ctrl+L`	Clear the screen (same as `clear`).
`Ctrl+R`	Reverse-search history. Keep pressing for older matches.
`Ctrl+A / Ctrl+E`	Jump to start / end of the line.
`Ctrl+U / Ctrl+K`	Cut from cursor to start / to end of the line.
`↑ / ↓`	Scroll through previous commands.
`!!`	Repeat the last command (e.g., `sudo !!`).
`!$`	Last argument of the previous command.
`history`	Numbered command history; `!42` reruns entry 42.

> **Tip: Read before you run**
>
> Paste commands into a text editor first, check every path, and run as a regular user unless you truly need root.

Create, Copy, Move, Delete

`touch notes.txt`	Create an empty file (or update its timestamp).
`mkdir -p a/b/c`	Make directories, including missing parents.
`cp file.txt bak/`	Copy a file into a directory.
`cp -r src/ dst/`	Copy a directory and everything in it.
`mv old.txt new.txt`	Rename; or move: `mv f.txt ~/docs/`.
`rm file.txt`	Delete a file. No trash can, no undo.
`rm -r dir/`	Delete a directory and its contents. Add `-i` to confirm.
`rmdir dir/`	Remove an empty directory only (safe).
`ln -s target link`	Create a symbolic link (shortcut) named `link`.

Viewing Files

`cat file`	Print a whole file. Best for short files.
`less file`	Scroll page by page. `/` search, `q` quit.
`more file`	Older pager; forward-only scrolling.
`head -n 20 file`	First 20 lines (default is 10).
`tail -n 20 file`	Last 20 lines.
`tail -f app.log`	Follow a log live as lines are added. Ctrl+C stops.
`wc -l file`	Count lines. `-w` words, `-c` bytes.

Finding Files

`find . -name "*.log"`	By name (quote the pattern). `-iname` ignores case.
`find /etc -type f`	Files only. `-type d` for directories.
`find / -size +100M`	Files larger than 100 MB.
`find ~ -mtime -7`	Modified within the last 7 days.
`find . -name "*.sh" -exec ls -l {} \;`	Run a command on each match; `{}` = file.
`locate sshd_config †`	Fast search of a prebuilt index; refresh with `sudo updatedb`.
`which python3`	Path of the program your shell will run.
`whereis ssh`	Binary, source, and man page locations.

Wildcards (globbing)

`*` any characters (`*.txt`) · `?` exactly one (`file?.txt`)
`[0-9]` one from a set/range · `*.{jpg,png}` brace expansion

Text Processing

grep — search inside files

`grep "error" app.log`	Lines containing `error`.
`grep -i error app.log`	Ignore upper/lower case.
`grep -r "TODO" src/`	Search recursively through a directory.
`grep -v "^#" file`	Invert: lines that do NOT match (skip comments).
`grep -n root /etc/passwd`	Show matching line numbers.
`grep -E "fail|deny" log`	Extended regex: match either word.

Slice, sort, count, transform

`cut -d: -f1 /etc/passwd`	Field 1 using `:` as delimiter (usernames).
`sort names.txt`	Sort lines. `-n` numeric, `-r` reverse.
`sort file | uniq -c`	Count duplicate lines (sort first).
`tr 'a-z' 'A-Z' < file`	Translate characters (here, to uppercase).
`sed 's/old/new/g' file`	Replace text in output. `-i` edits the file in place.
`awk '{print $1}' file`	Print the first whitespace-separated column.

```
# Top 5 IPs in a web log
awk '{print $1}' access.log | sort | uniq -c \
  | sort -rn | head -5
```

Editors Quick-Start

nano — beginner-friendly

`nano file`	Open (or create) a file. Shortcuts listed at the bottom.
`Ctrl+O / Ctrl+X`	Save (Write Out) / Exit.
`Ctrl+W`	Search for text.
`Ctrl+K / Ctrl+U`	Cut the current line / paste it.

vim — survival kit (it starts in Normal mode)

`i`	Insert mode: start typing text.
`Esc`	Back to Normal mode (do this before any command).
`:w :q :wq`	Save · quit · save and quit.
`:q!`	Quit and discard all changes.
`/word`	Search forward; `n` jumps to the next match.
`dd / yy / p`	Delete (cut) line / yank (copy) line / paste.

Reading ls -l Output

```
-rwxr-xr-- 1 jm staff 4096 Oct 12 09:30 run.sh
    1      2 3    4    5        6         7
```

1 permissions · 2 hard links · 3 owner · 4 group · 5 size (bytes) · 6 last modified · 7 name

-	rwx	r-x	r--
type - file d dir l link	owner (u) read, write, execute	group (g) read, execute	others (o) read only

r = read (4) w = write (2) x = execute (1). On a directory, x means you can enter it and r means you can list it.

chmod — Change Permissions

Octal: add the values for each group (owner, group, others)

0	1	2	3	4	5	6	7
---	--x	-w-	-wx	r--	r-x	rw-	rwx

`chmod 644 file`	`rw-r--r--` Standard file: owner edits, everyone reads.
`chmod 755 script.sh`	`rwxr-xr-x` Programs and directories.
`chmod 600 id_ed25519`	`rw-------` Private files such as SSH keys.
`chmod 700 ~/private`	`rwx------` Owner-only directory.

Symbolic: who (u g o a) + op (+ - =) + perm (r w x)

`chmod u+x run.sh`	Add execute for the owner.
`chmod g-w file`	Remove write from the group.
`chmod o=r file`	Set others to exactly read-only.
`chmod a+r file`	Give everyone (all) read.
`chmod -R g+rX dir/`	Recursive; capital X adds execute only to directories.

Ownership & Defaults

`chown alice file`	Change the owner (needs sudo).
`chown alice:devs file`	Change owner and group together.
`chown -R alice dir/`	Apply to a directory recursively.
`chgrp devs file`	Change the group only.
`umask`	Show default permission mask. `022` → new files 644, dirs 755.

Account Files

File	Contains
`/etc/passwd`	Accounts: name, UID, GID, home, shell. World-readable.
`/etc/shadow`	Password hashes and aging. Root-only.
`/etc/group`	Groups and their members.

Special Permission Bits

Bit	Octal	Shows as	Effect
SUID	4000	`rwsr-xr-x`	Program runs with the file OWNER's rights (e.g., `passwd` runs as root).
SGID	2000	`rwxr-sr-x`	File: runs with group's rights. Dir: new files inherit the dir's group.
Sticky	1000	`rwxrwxrwt`	In a shared dir (`/tmp`), only a file's owner can delete it.

Security note: SUID is a classic privilege-escalation path

A SUID-root binary with a flaw (or one that can spawn a shell) can hand an attacker root. Audit them regularly: `find / -perm -4000 -type f 2>/dev/null`

Users & Groups

`whoami`	Your current username.
`id`	Your UID, GID, and every group you belong to.
`who / w`	Who is logged in / plus what they're running.
`sudo useradd -m -s /bin/bash bob`	Create user with a home dir and Bash shell.
`sudo passwd bob`	Set or change a user's password.
`sudo usermod -aG devs bob`	Append to a group. Without `-a`, other groups are removed!
`sudo groupadd devs`	Create a new group.
`sudo userdel -r bob`	Delete a user and their home directory.

Debian/Ubuntu also provides `adduser`, an interactive helper.

Privilege: sudo & su

`sudo command`	Run one command as root; logged and accountable.
`sudo -l`	List what you're allowed to run with sudo.
`su -`	Full login shell as root (needs root's password).
`su - alice`	Switch to another user with their environment.
`sudo visudo`	Safely edit `/etc/sudoers` with a syntax check.

Least privilege

Work as a normal user and elevate one command at a time. Admin group: `sudo` (Debian/Ubuntu) or `wheel` (RHEL/Fedora).

Processes & Jobs

`ps aux`	Every process: user, PID, %CPU, %MEM, command.
`ps aux | grep nginx`	Find a specific process.
`pgrep -a sshd`	PIDs (and command lines) matching a name.
`top`	Live view. q quit, k kill, P/M sort CPU/memory.
`htop` †	Friendlier, colorized `top` (RHEL: from EPEL).
`kill 1234`	Ask PID 1234 to stop (SIGTERM, signal 15).
`kill -9 1234`	Force-kill (SIGKILL). Last resort: no cleanup.
`killall firefox`	Signal every process with that name.
`command &`	Start a command in the background.
`jobs`	List this shell's background/stopped jobs.
`Ctrl+Z`	Pause (stop) the foreground job.
`bg %1 / fg %1`	Resume job 1 in background / bring to front.
`nohup ./job.sh &`	Keep running after you log out.
`nice -n 10 ./job.sh`	Start with lower priority (−20 high … 19 low).
`renice -n 5 -p 1234`	Change priority of a running process.

Services (systemd)

`systemctl status ssh`	Is it running? Recent log lines included.
`sudo systemctl start ssh`	Start now.
`sudo systemctl stop ssh`	Stop now.
`sudo systemctl restart ssh`	Stop, then start (apply config changes).
`sudo systemctl enable ssh`	Start automatically at boot. `--now` also starts it.
`sudo systemctl disable ssh`	Don't start at boot.
`journalctl -u ssh`	Logs for one service (unit).
`journalctl -f`	Follow new log entries live.
`journalctl -xe`	Jump to the end, with explanations.

SSH service name: `ssh` on Debian/Ubuntu, `sshd` on RHEL/Fedora.

Package Management

Task	Debian / Ubuntu	RHEL / Fedora
Refresh package list	`sudo apt update`	`sudo dnf check-update`
Upgrade everything	`sudo apt upgrade`	`sudo dnf upgrade`
Install	`sudo apt install nmap`	`sudo dnf install nmap`
Remove	`sudo apt remove nmap`	`sudo dnf remove nmap`
Search	`apt search nmap`	`dnf search nmap`
Show details	`apt show nmap`	`dnf info nmap`
List installed	`apt list --installed`	`dnf list installed`

Low-level tools: `dpkg` (Debian) and `rpm` (RHEL). Always run updates before installing new software.

System Information

`uname -a`	Kernel version, architecture, hostname.
`cat /etc/os-release`	Which distribution and version.
`hostnamectl`	Hostname, OS, kernel, hardware summary.
`uptime`	Time since boot, users, load averages.
`df -h`	Free/used space per mounted filesystem.
`du -sh dir/`	Total size of a directory.
`free -h`	RAM and swap usage.
`lsblk`	Disks and partitions as a tree.
`lscpu`	CPU model, cores, architecture.
`mount`	Mounted filesystems (`findmnt` is clearer).

Archives & Compression

`tar -czvf backup.tar.gz dir/`	Create a gzip-compressed archive.
`tar -xzvf backup.tar.gz`	Extract it here. `-C /path` sets the target.
`tar -tvf backup.tar.gz`	List contents without extracting.
`gzip file / gunzip file.gz`	Compress / decompress a single file (replaces it).
`zip -r site.zip site/` †	Create a .zip (cross-platform).
`unzip site.zip` †	Extract a .zip archive.

tar flags decoded

c create · x extract · t list · z gzip · v verbose · f file name follows (keep f last).

Networking

`ip a`	Interfaces and IP addresses (legacy: `ifconfig`).
`ip r`	Routing table and default gateway (legacy: `route`).
`hostname -I`	Just this host's IP addresses.
`ping -c 4 8.8.8.8`	Test reachability with 4 packets.
`ss -tulpn`	Listening ports + process (legacy: `netstat`).
`curl -I https://site`	Fetch a URL; `-I` = headers only.
`wget URL`	Download a file to the current directory.
`dig example.com` †	Detailed DNS lookup (`dnsutils` / `bind-utils`).
`nslookup example.com` †	Simple DNS lookup (same packages).
`traceroute host` †	Hops to a destination (or `tracepath`).
`/etc/hosts`	Static name → IP overrides, checked before DNS.
`/etc/resolv.conf`	DNS servers in use (often managed automatically).

Remote Access

`ssh user@host`	Secure remote shell. `-p 2222` for another port.
`ssh-keygen -t ed25519`	Create a modern key pair in `~/.ssh/`.
`ssh-copy-id user@host`	Install your public key on the server.
`scp file user@host:/tmp/`	Copy a file over SSH.
`rsync -avz src/ user@host:dst/`	Sync directories; only sends changes.

Redirection & Pipes

`cmd > out.txt`	Send output to a file (overwrite).
`cmd >> out.txt`	Append output to a file.
`cmd < in.txt`	Read input from a file.
`cmd 2> err.txt`	Send errors (stderr, stream 2) to a file.
`cmd > all.txt 2>&1`	Output and errors to the same file.
`cmd &> all.txt`	Bash shorthand for the line above.
`cmd1 | cmd2`	Pipe: output of cmd1 becomes input of cmd2.
`cmd | tee out.txt`	Show on screen AND save to a file.
`find . -name "*.bak" | xargs ls -l`	Turn input lines into arguments.

Environment & Scripting Starter

`echo $PATH`	Directories searched for commands.
`export EDITOR=nano`	Set a variable for this shell and its children.
`env`	List all environment variables.
`alias ll='ls -lah'`	Create a shortcut command.
`~/.bashrc`	Startup file for aliases/exports; reload: `source ~/.bashrc`.
`chmod +x hello.sh`	Make a script executable, then run `./hello.sh`.

```
#!/bin/bash            # shebang: use Bash
name="student"         # no spaces around =
echo "Hello, $name"
echo "Today: $(date +%F)"
for f in *.txt; do echo "Found: $f"; done
```

Security Essentials

`sudo ss -tulpn`	What is listening? Close anything you don't recognize.
`sudo tail -f /var/log/auth.log`	Logins and sudo use (Debian/Ubuntu).
`sudo tail -f /var/log/secure`	Same, on RHEL. Fedora: `journalctl -f`.
`last`	Recent successful logins and reboots.
`sudo lastb`	Recent FAILED login attempts.
`find / -perm -4000 -type f 2>/dev/null`	List SUID binaries to audit.
`sha256sum file.iso`	Hash a file; compare with the published value.

Firewall task	ufw (Ubuntu)	firewalld (RHEL/Fedora)
Status	`sudo ufw status`	`sudo firewall-cmd --list-all`
Allow SSH	`sudo ufw allow 22/tcp`	`sudo firewall-cmd --permanent --add-service=ssh`
Turn on / apply	`sudo ufw enable`	`sudo firewall-cmd --reload`

⚠ Dangerous commands: understand before running

`rm -rf /*`: erases the whole system. No undo.
`chmod -R 777`: anyone, incl. attackers, can alter files.
`dd of=/dev/sdX`: one wrong letter wipes a whole disk.
`curl URL | bash`: runs unseen code. Download, read, run.