Keyboard Shortcuts

Shortcut	What It Does	Notes
Esc + .	Insert last argument entered	Repeat to go back to previous arguments
Up Arrow	Auto-fill last command entered	Repeat to go back to previous commands
Ctrl+D	Logout / Exit	
Ctrl+L	Clear terminal output	
Ctrl+Shift+T	Open new terminal tab	
Ctrl+Arrow (Right/Left)	Move to beginning/end of a word	

Important Directories & Files

Directory/File	(Stands For) Purpose
/root	Root user's home directory
/home	Normal users' home directories
/var	(Variables) Database locations, spool files for mail, etc.
/usr	(User System Resources) System resources used by users
— /usr/bin	(Binaries) Executables usable by normal users
— /usr/sbin	(System Binaries) Executables only usable by root user
— /usr/lib	52-bit libraries
— /usr/lib64	64-bit libraries
/etc	(Extended Text Configurations) Config files
— /etc/passwd	User account config files
— /etc/group	Group membership info & config files
— /etc/shadow	User password info & config files
— /etc/sudoers	Main file for sudo config
— /etc/sudoers.d/	Destination for sudo config "dropping files"
— /etc/login.defs	Defines default properties used for new user accounts

File Redirections

`>`
`>>`
`1>`

Basic File/Directory Commands

Command		Description
`ls` `dir1`		List contents of dir1
`-l`	additional info (owner, perms, etc.)	
`-a`	lists all, including hidden files	
`-i`	adds file inode to info displayed	
`-d`	info for directory itself (rather than its contents)	
`mkdir` `dir1`		Create a new directory
`-p`	creates full path (multiple directories if needed)	
`-v`	verbose (output shows action taken)	
`cp` `dir1 dir2`		Copy dir1 into dir2
`-r`	recursively ("and its contents")	
`rm` `dir1`		Removes (deletes) an empty directory
`-r`	recursively ("and it's contents"), including special files	
`mv` `file1 dir1`		Move file1 into dir1
`-v`	verbose (output shows action taken)	
`ln` `target link`		Creates a hard link from file "link" to file "target"

Basic File/Directory Commands (cont)

`-s`	makes a symbolic link instead of a hard link

Users & Groups

Command	Options
`id` *user1*	Show UID, GID, & secondary groups (current user if not specified)
`getent passwd` *usr1*	Find out if usr1 is known to the system
`useradd` *user1*	Add a new user
`-u` *#*	set a specific UID
`-s /sbin/nologin`	
`usermod` *user1*	Change properties of existing user
`-c "`*text*`"`	adds text to comment field
`-g` *group1*	changes **primary group** to group1
`-G` *group1*	**replaces** supplementary group with group1
`-aG` *group1*	**appends** group1 to supplementary groups
`-L` *user1*	
`newgrp` *group1*	Change current user's primary group (**temporary**; current session only)
`userdel` *user1*	Deletes user1 but **not** their home directory

Users & Groups (cont)

`-r`	
`groups` *user1*	Displays simple list of groups user1 is a member of
`passwd` *user1*	Set password for user1
`chage` *user1*	Change password aging properties for user
`-m` *days*	minimum # of days between password changes
`-M` *days*	maximum # of days between password changes
`-W` *days*	warning period before password expires
`-I` *days*	inactivity period (password usable after expiration)
`-d 0`	require password change on next login
`-E` *date*	date when **account** expires
`getent group` *grp1*	Find out if grp1 is known to the system
`groupadd` *group1*	Create a new group
`-g` *#*	set a specific GID
`-r`	create a **system** group
`groupmod` *group1*	Change properties of existing group
`-g` *#*	change GID to specified number

To set the number of days from today when user's **account** expires:
`chage -E $(date +`*days* `%Y-%m-%d)`

To give full admin privileges to a user or group:
`echo "[`*user1*`|%`*group1*`] ALL=(ALL) ALL" >> /etc/s udo ers.d/`*name*

Process States & Signals

Running	**R**	currently using CPU
Runnable	**R**	
Sleeping	**S**	
I (idle)		
D		*(not yet addressed)*
K		*(not yet addressed)*
Stopped	**T**	defined (has PID) but prevented from getting CPU cycles

--- Signals ---		
1) SIGHUP	Hang Up	toss current config file in memory & reload it from the file system
9) SIGKILL	Kill it dead!	immediately end process that isn't responding to "terminate" signal
15) SIGTERM	Terminate	shut down process (default used by `kill`)
18) SIGCONT	Continue ("unstop")	
19) SIGSTOP	Stop	

Wildcards & Shortcuts

Wildcard	What It Represents	Example
~	Current user's home directory	
.	The directory you're currently in	
..	Parent directory of your current directory	
*****	All files in current directory	`ls i*` = all files starting with "i"
?	"Any character"	`f??e` matches anything with 4 characters that starts with "f" & ends with "e"
[aou**]**	Match to any **one** of the enclosed characters	

Wildcards & Shortcuts (cont)

!!	"Last command run" (inserts into current command)	**sudo !!** = rerun command as root
^x**^**y		
!3	Rerun 3rd command from command history	
!ls	Runs most recent command from history that had "ls" in it	

vim Editor Controls

--- Command Mode ---	
o	Starts new line **and** switches to Insert mode
cw	"Change word" - deletes word **and** switches to Insert mode
G	Go to end of file
dd	Delete **or cut** entire line
yy	Copy entire line
p	Paste copied text
x	Delete character cursor is on top of
dw	Delete word cursor is on top of
d$	Delete from cursor to end of line
ZZ	Save & quit (existing files)
--- Extended Command Mode ---	
:w *filename*	Save (new files)
:wq	Save & quit (existing files)

Misc. Commands

whatis *command*	
echo	Creates output (equivalent of `print` in Python)
date	Displaying, setting, calculating... basically all things date-related
+%F	display current date in international format (YYYY-mm-dd)

By **River L.** (Tamaranth)
cheatography.com/tamaranth/

Not published yet.
Last updated 17th June, 2025.
Page 3 of 6.

Misc. Commands (cont)

+%R	display current time (24-hr clock)
+%s	display # of seconds since epoch
ssh *user@ip.addr*	Start ssh connection
-v	verbose; shows in detail what's happening while establishing connection
-Y	enables graphical application support
-p *port#*	connect to ssh service not listening on default port 22
reads private key from identity file for public key authentication	
tmux	Splits terminal window into multiple separate panes *(I think)*
gnome- cha racters	Opens window of gnome icons/emotes
gnome- cal culator	Opens built-in calculator (like Windows)
sleep *seconds*	Waits & does nothing for specified time

Ownership & Permissions

chown *user:group*	Change owning user **and** owning group
chown *:group*	Change only owning group
chgrp *group file*	Alternate way to change owning group

Ownership & Permissions (cont)

chmod *[perms] file*	Change file/directory permissions
-R	recursive; applies to directory and all contents
g+r	grants owning **g**roup the **r**ead permission
a-x	
-R a=rwX	recursively set perms for **a**ll parties to **rw**, with **x** perms on directory **only**
770	octally set rwx perms for owning user & group, 0 for other
chmod *[perms] file*	(cont'd) Change special permissions
o+t *dir*	set S**t**icky bit (in **o**ther) ** For directories only**
1*xxx*	
g+s *dir/file*	
2*xxx*	
3*xxx*	

Ownership & Permissions (cont)

`u+s` *file*	
`4`*xxx*	
`umask`	Displays currently set umask (when used by itself)
`077`	removes default perms (0 from user, 7 from group/other)
`0077`	preserves any special perms when removing defaults

Octal Expression Values: **r=4** | **w=2** | **x=1**

Default permissions (on vanilla Linux):
directories=**777** | files=**666**

Managing Processes

`w`	Displays currently logged-in users, login method, time, & resource usage info
	`-u` *username*
command `&`	Starts a new job in the background
`jobs`	Displays jobs running in the background ("**+**" = default job)
`fg` *%1*	Brings job 1 to the foreground
`bg` *%1*	Sends job 1 to the background
`pidof` *command*	Returns the PID(s) of a currently running job

Managing Processes (cont)

`systemctl status` *cmd*	Shows main PID of a process, among other things (can use if `pidof` returns multiple)
--- CPU Load Handling ---	
`uptime`	Displays the load average for the last 1, 5, and 15 mins
`lscpu`	Displays number of CPUs in the system, among other things
`-p`	simplified output (each row of numbers = 1 CPU)
--- Viewing & Managing Processes ---	
`ps`	Shows info about processes; pipe to `head`/`grep`/etc for less output ("**[**output**]**" = kernel thread)
`aux`	
`lax`	
`-ef`	
`--forest`	shows output in visual "tree" format
`pstree`	Shows processes in visual tree format *(may not always be installed)*
`top`	Linux's in-terminal equivalent of Task Manager (Also shows load average) Keyboard Controls:
`k`	send a signal to a process from within `top`
Shift+M	sort by memory usage

Managing Processes (cont)	
Shift+P	return to sorting by CPU usage
l / t / m	toggle "load", "threads", & "memory" header lines
kill *PID*	Sends signal 15 to specified process ** Must use process ID**
-*signal*	
-1	displays list of signal options
killall *process*	Kills processes by name instead of PID
pkill	Look up & manage processes by name or other attributes (default is signal 15)
-*signal*	specify signal to send
-u *username*	sends signal to user's account (forces logout & shuts down their processes)
-t *pts/22*	kills specified terminal session (TTY)
nice -n *x command*	Starts a **new** process with an adjusted priority value (*x*)
renice -n *x* -p *PID*	Change the priority value (*x*) of an **existing** process
Process Priority Values: **-20** to **19** (default = **0**) (*negative* = **higher** priority | *positive* = **lower** priority)	