UNIX Commands

Command	Function	Common Options	Example/Notes
General Commands			
ctrl+a	go to beginning of line		
ctrl+e	go to end of line		
ctrl+u	delete text before cursor		
ctrl+k	delete text following cursor		
ctrl+w	delete to beginning of word		
alt+f	go to next word		On a Mac go to Terminal > Preferences > Keyboard and select use Option as Meta key
alt+b	go to previous word		On a Mac go to Terminal > Preferences > Keyboard and select use Option as Meta key
ctrl+c	stop current command		
ctrl+d	log out of current session, similar to exit		
clear	clear shell		
whatis	short description of a command		whatis *bc*
man	display manual for command		man *bc* (type q at the : prompt to exit)
date	display date and time		date
cal	display calander	month year	cal *01 2021*
pwd	display current working directory		pwd
cd	change directory	.. (up one level), otherwise home directory	cd ..
ls	list files in current directory	-l (list file permissions) *directoryname*	ls (for working directory) OR ls *directory* (for any directory on the computer)
mkdir	make new directory		mkdir *directory*
history	display command history		history
bc	basic calculator		bc (to end, type ctrl-D)
cat	display file contents		cat > *file* (place input into *file* , can be used to combine files, eg cat *.fasta >newfile)
more (or less)	display file screenful at time	/pattern (search), q (quit)	more *file* (use zmore for compressed files)
head	output the first 10 lines of a file	-n (custom number of lines to display)	head *file*
tail	output the last 10 lines of a file	-n (custom number of lines to display), -f (monitor file in real time)	tail *file*
wc	word, line, character count	-l (linecount only), -c (character count only)	wc *file* (line count in column1, word count in column 2, character count in column 3)
diff	compare file contents		dif *file1 file2*
uniq	return or omit unique lines	-d (see list of duplicate lines)	uniq *file*
touch	touch a file; create it if it doesn't exist		touch *file*
open	open a file	-a *program*	open -a *"Microsoft Word" file*
cp	copy file	-R (recursive), -X (ignore attributes)	cp -RX *dir1 dir2*
mv	move or rename file		mv *file1 file2*
rm	remove (delete) file	-r (remove directory), -f (force remove file)	rm *file*
paste	paste lines of separate files on the same line of one file		paste *file1 file2 file3*
echo	write arguments to standard output	-n (don't add new line after input)	echo Hello, world!
|	pipe - output from one command to another		for example, see ifconfig command below
tee	capture output at an intermediate step while piping		ls | tee list.txt | tail
df	disc usage	-g use 1 gbyte blocks insteak of 512 kbyte	df -g
top	display all running processes		top
ifconfig	configure a network interface		ifconfig | grep "inet " (use to get IP address)
find	search a directory tree for files that match a name		find */path* -name *filename*
vi	visual editor, shell-based text editor		vi *file* (to edit, type I for insert to exit insert mode hit esc; to save type :w, to quit, type :q)
screen	GNU Screen for running jobs on remote machines	ctrl+a+d (detach window) screen -DR (show detached windows) screen -r (resume a detached window)	screen
./	Execute program		*./program.pl*
Text Searching and Manipulation			
grep	search for pattern in file	-r (grep directory), -c (return only count) -o (output matches)	grep *pattern files* OR grep -r *directory* (use zgrep for compressed files)
sort	display sorted content	-r sort in reverse order	sort *inputfile >output file*
sed	stream editor for transforming text, line-based	-i (make changes to input file), -e (for multiple commands in one line)	sed 's/[[:<:]]*old_pattern* [[:>:]]*/new_pattern/g;' inputfile >outputfile*
awk	pattern scanning and processing language	-F (field seperator)	awk *'pattern { action }' inputfile1 inputfile2* (optional) *>outputfile*
tr	translate, replace simple text like single characters	-d (delete), -cd (delete everything but)	tr a-z A-Z *<inputfile >outputfile* (replace all lowercase letters with uppercase)
rev	reverse characters in every line		rev
tac	reverse order lines		
join	join lines of two files based on a common field	-t (field separator) -1 (field 1, etc)	join -t "\t" -1 *N* -2 *N file1 file2*
split	split up a file	-l (linecount), -b (size in bytes)	split -l *linecount inputfile outputfile*
csplit	split up a file by text match	-f (prefix for each new file)	csplit -f *prefix file* "/pattern1/" "/pattern2/".. "/patternN/"
File Compression			
unzip	uncompress a .zip file		unzip *file.zip*
tar cf	create a single tar file containing multiple files		tar cf *newname.tar file1 file2 fileN*
tar xf	extract files from a tar		tar xf *file.tar*
tar cvzf	create a single tar file and compress it		tar cvzf *newname.tar file1 file2 fileN*
tar xvzf	extract and uncompress files from a tar		tar xvzf *file.tar.gz*
gzip	gzip compress a file	-d (decompress a gzip)	gzip *file* OR gzip -d *file.gz*
Server Access and File Transfer			
ftp	file transfer protocol (non-secure)		ftp *username@server.com* (mget *file* will transfer file to current directory)
sftp	secure file transfer protocol (encrypted)		sftp *username@server.com*
ssh	secure shell - use to access remote computers and servers		ssh *username@server.com*
scp	copy something from one machine to another	-r (for copying directories), use . for path to working directory	Copy something from this machine to some other machine: scp */path/to/local/file username@hostname:/path/to/remote/file* Copy something from another machine to this machine: scp *username@hostname:/path/to/remote/file /path/to/local/file*
UNIX Symbols			
>	redirect standard output		
<	redirect standard input		
>>	redirect and append standard output		
;	separate commands on same line		
( )	group commands on same line		
/	separator in a pathname		
~	(tilde) your home directory		
.	present working directory		
..	parent of present working directory		
[ ]	Wildcard match of any one bracketed character in file name		
Regular Expressions (use in grep, sed, awk, and tr, with some exceptions, particularly with tr)			
.	a wildcard character (doesn't match newline).		
*	match the preceding item zero or more times.		
.*	match any item zero or more times.		
+	match the preceding item one or more times.		
?	match the preceding item once or not at all.		
\b	match at either end of word		
\n	newline		
\t	tab		
\	used to negate something else, such as \\t would be \t, not tab		
File Permissions			
chmod	change access permissions	Owner, position 1; Group, position 2; Anyone, position 3 Read = 4, write = 2, execute = 1, none=0 1 = execute only; 2 = write only; 3 = write and execute; 4 = read only 5 = read and execute; 6 = read and write; 7 = read, write and execute	chmod 755 (only owner can write, anyone can read and execute) ls -l (list file permissions)
Shell Scripting			
#!/bin/bash	directs shell to bash interpretor		Start shell script with #!/bin/bash. Execute script with *./scriptname*
#	note		Information following "#" is not executed, except for special #!/bin/bash
var=value	variable assignment	`*command* ` (use backticks to store output from commands as variable	*variable_name=value* (to call variable use *$variable_name*)
read	prompt for input from command line	-p (display prompt info)	read -p "Please enter value: " value (entry stored as variable value)
if	condidtional statements		if [*condition*]; then *commands* ; elif [*condition*]; then *commands* ; else *commands;* fi
while loops	repeat a task while some value is defined or true		while *condition* ; do *commands* ; done