Basics

man {cmd}	man page for cmd
type {cmd}	executable/shell/alias
help {cmd}	if there is no man page
wc	word count (`-l`ines `-w`ords)
sort {file}	alphabetically (`-r`everse `-n`umeric `-u`nique `-k` column)
ls | sort -nk4	sort by file size
mkdir [-p]	also make `-p`arents if not exists
rm [-r -d -i]	(`-r`ecursive `-d` empty directories `-i` prompt before every removal)
mv {source ...} {destination}	move file(s) or directories or rename (if single file or folder is specified as source & destination)
cp {source ...} destination	copy file(s)
history | less	open history in pager
history | grep {term}	search for term in history
cat {file1} {file2}	con*cat*enate & print contents of file(s)
less {file}	page file contents
head -{n} & tail	print firts / last n lines of file
tail -f	follow (eg. for logs)
tr {search} {replace}	replace search with replace in stdout (eg. `cat msg | tr s S` to uppercase all `s`) (`-d`elete {search} instead of replacing)

Grep

man grep | grep "count" -i -C5	grep in `man` for "count"
-i	insensitive case
-w	word search
-r	recursive
-c	count number of matches
-A {n}	end n lines after match
-B {n}	start n lines before match
-C {n}	start & end n lines before & after match
-n	add line number for each match
-m {n}	limit to n matches
-l	show filenames

Search for patterns in each file's contents (supports regex!)

`grep "{pattern}" {file}` print each line from `{file}` that contains `{pattern}`

`egrep` (or `grep -E`) - use regex meta-characters (eg. `? { }` etc.) for their special meaning and not as a normal character

Shortcuts

ctrl+l	clear screen
ctrl+a / ctrl+e	move to beginning/end of line
option+← / option+→	move left/right one word
ctrl+t	swap char with one before
option+t	swap word with one before
ctrl+u	kill from cursor to beginning of line
option+d	kill word forward
ctrl+w	kill word backward

Shortcuts (cont)

!!	last command
!-{n}	current command minus n
ctrl+r	incremental history search

Some options on some OSes / terminals need to be set with bindkey, eg. `bindkey "^[b" backward-word`

Streaming / redirection / piping

>	redirect output to specific file (overwrite!)
>>	append new data to specific file
<	pass contents of file to stdin
2>	redirect stderr to specific file
2>>	append stderr to file
2>&1	redirect both sderr and stdout to same file
command1 | command2	pipe (redirect) stdout of 1 to stdin of 2
tee	read stdin and copy both to stdout and a file.
cat a.txt b.txt | tee both.txt | wc -l	see above

If redirecting both stdout and stderr, out has to be first! Eg. cat a.txt b.txt > both.txt 2> error.txt

less

space or f	next page
b	previous page
enter or down arrow	scroll one line
/ {pattern}	search for pattern
q	quit

Pager to read manuals, docs, etc.

Permissions

r	file can be read, dir contents listed
w	file can be modified, dir contents modified
x	file can be treated as a program to be executed, dir can be `cd`d into
-	file/dir can't be read/modified/executed (depending on where the dash is)
chmod {permission} {file}	change permissions for given file
chmod g+w	add write permissions for group
chmod a-w	remove write permissions for all
chmod a=r	set permissions to read **ONLY** for all
chmod 020	--- -w- ---
chmod 700	rwx --- ---
chmod 644	rw - r-- r--

Modes: **u**ser, **g**roup, **o**thers, **a**ll
What: **-**: remove, **+**: add, **=**: only
Which: **r**ead, **w**rite, **x**ecute

Expansion

~	home dir
~{someuser}	home dir of someuser
*	zero or more characters

Expansion (cont)

?	any single character
[]	range of characters eg. [123] [0-9] [A-F]
[^]	negate range eg. [^0-9]* -> not starts with a digit
{}	generate arbitrary strings, eg. touch page{1,2,3}.txt will generate 3 files
{...}	generate between range eg. {1..31} {a..e} {2..10..2} (start,stop,step)
{x,y{1..5},z}	will expand to `x y1 y2 y3 y4 y5 z`
$((expression))	arithmetic expansion (`+`, `-`, , `/`, `*`, `%`)
$(command)	command substitution (eg. echo today is $(date))
"double quotes"	respect spacing & ignore special characters except $, \ and ` (backtick)
'single quotes'	suppress all forms of substitution

Before any command runs, some special commands expand to match given files eg.: `ls -l *.txt` will be expanded to contain all files ending with `.txt`
`mv app?.css Styles/` -> move **all** files matching pattern to Styles directory

Finding files

locate {searchterm}	find all files containing searchterm. (-ignore case `-l` limit to n options `-e` only files that exist now)
find {dir/}	list every file and directory inside dir/

Finding files (cont)

-type d	only find directories (or `f`iles)
-name "{pattern}"	find files with given pattern (**case sensitive!**) (`iname` for case insensitive)
-size +1G	larger than 1gb (or -50M, 20k -> exactly 20k)
-user {uname}	files belonging to uname
-empty	empty files/folders
[acm]min {n}	**a**ccessed / **c**hanged / **m**odified n minutes ago (+- for greater/less than)
[acm]time {n}	a/c/m n * 24 hours ago
`-and -or -not`	logical operators for more complex needs eg. find -type f -not -name "*.html"

`-exec command '{}' ';'`	execute cmd for each match, eg.: find ~/Downloads -type f -empty -exec ls -l '{}' ';'
find -type f -name "*.html -exec cp '{} '{}_COPY' ';'	find each matching file & create a copy with suffix
-ok command '{}' ';'	same as above but ask for confirmation each time (`y`/`n`)
find -name "*.txt" | xargs ls	equivalent to the -exec ls '{}' ';'