less and man	
q	quit
/	find
u	half page up
d	half page down
k	one line up
j	one line down

vi – Command Mode	
:q	quit
:q!	force quit (LOSE UNSAVED INFO)
:w	save
:wq	save and quit
:! <command>	execute a shell command
/	find
i	enter INSERT mode (ESC to exit)
v	enter VISUAL mode (ESC to exit)
Shift-v	enter VISUAL-LINE mode
Ctrl-Shift-v	enter VISUAL-BLOCK mode

vi – Command Mode – basic editing commands	
dd	delete a line
x	delete character or selection
r	replace a character or selection
p	paste deleted/yanked text after cursor
P	paste deleted/yanked text before cursor
u	undo
Ctrl-r	redo

vi – Command Mode - Navigation	
^	move to the beginning of a line
$	move to the end of a line
h	left
j	down
k	up
l	right
For more info: **vimtutor**	

Piping and redirection	
A | B	output of A is sent to B
A > f	output of A overwrites file f
A >> f	output of A appends file f

wc – word counter	
-l, --lines	line count
-w, --words	word count
-m, --chars	character count

grep – pattern searcher	
--help list of options and brief explanations	
-c, --count	-A, --after-context
-v, --invert-match	-B, --before-context
-i, --ignore-case	-C, --context
-w, --word-regexp	-h, --no-filename
-l, --files-with-match	-L, --files-without-match

sort	
-g, --general-numeric-sort	
-n, --numeric-sort	
-r, --reverse	
-u, --unique	
-k, --key=POS	
-t, --field-separator=SEP	
sort -k3g a.tab	#sort 3rd column

uniq	
-c, --count	count occurrences of each line
-d, --repeated	print only duplicated lines
-u, --unique	print only uniq lines

Regular Expressions	
.	matches any character except a newline
*	matches 0 or more of previous character
+	matches 1 or more of previous character
[xyz]	matches characters x, y, z
[^xyz]	matches characters OTHER than x, y, z
^	anchors match at BEGINNING of line
$	anchors match at the END of the line
\	escapes the following special character
	\t (tab) \n (new line) \r (win new line)
(...)	captures the enclosed sequence
\n	recalls nth captured sequence
+	matches 1 or more of previous character
|	OR

Sed syntax – by example

```
sed ‘2d’                          # delete line 2
sed ‘2,5d’                        # delete lines 2-5
sed ‘/pattern/,/pattern/’         # delete a range
sed  -n ‘/pattern/p’              # print if match
sed ‘[ADDRESS]d’
sed ‘[ADDRESS] s/pattern/replacement/’
```

Sed flags

g	global
I	ignore case
P	print

AWK syntax – by example

```
awk ‘$2>0 `                    # print if col 2 is “Fred”
awk ‘{print $2,$4}’             # prints columns 2 and 4
awk –F’,’ ‘{print $2,$4}’       #  change delimiter to ‘,’
awk ‘/pattern/’                 # print if match
awk ‘/pattern/,/pattern/’  # prints between patterns
```

AWK – logical operators

||	logical OR
&&	logical AND
!	logical NOT

AWK - builtin variables

NR	Current line number

Parameter Expansion - ${<varname>}

```
$ x=’toy’
$ echo ${toy}s
toys
```

Command substitution - $(<command>)

```
$ echo “date: $(date)”
```

Arithmetic substitution - $((<expression>))

```
$ echo $(( 3 + 5 ))
8
$ x=1; y=2
$ echo $(( x + y))
3
```

Bash for-loop syntax

```
for x in <list>
do
        <code>
done
```

```
for x in <list; do <code>; done
```

Bash if statement

```
if [[ -r file.txt ]]
then
        <code>
Fi
```

-r	file is readable
-f	file exists
-d	directory exists
-z	test variable is empty