Vi / Vim — Command Mode Crib Sheet

A progressive teaching sequence: motions → operators → combinations → text objects → practical editing. Commands assume Normal mode unless stated otherwise.

1. First: moving the cursor

h j k l	Left, down, up, right
w / W	Start of next word / WORD (whitespace-delimited)
b / B	Start of previous word / WORD
e / E	End of current/next word / WORD
ge / gE	End of previous word / WORD (Vim)
0 / ^ / $	Column 1 / first non-blank / end of line
g_	Last non-blank character on line (Vim)
gg / G	First line (Vim) / last line
10G / 10gg	Go to line 10
{ / }	Previous / next paragraph
(/)	Previous / next sentence
%	Matching bracket/parenthesis (on supported pair)
fX / FX	Find next / previous X on current line
tX / TX	Move before next X / after previous X
; / ,	Repeat / reverse last f, F, t, T search
Ctrl-f / Ctrl-b	Page forward / backward
Ctrl-d / Ctrl-u	Half-page down / up
H / M / L	Top / middle / bottom of screen
/text / ?text	Search forward / backward; n / N repeat / reverse

2. The essential editing grammar

operator + motion	Apply action over the text traversed by motion
d / y / c	Delete / yank (copy) / change (then Insert mode)
p / P	Put after / before cursor (linewise puts below / above)
dd / yy / cc	Delete / yank / change whole line
D / C	Delete / change from cursor to end of line
Y	Yank whole line in Vim (equivalent to yy)
x / X	Delete character under / before cursor
rX / R	Replace one character with X / overwrite mode
i / a / I / A	Insert before / after cursor / first nonblank / end of line
o / O	Open new line below / above
u / Ctrl-r	Undo / redo (Ctrl-r is Vim)
.	Repeat last change
Esc	Return to Normal mode

3. Combine operators with word motions

dw / dW	Delete to next word / WORD boundary
de / dE	Delete through end of word / WORD
db / dB	Delete backwards to word / WORD start
yw / yW	Yank to next word / WORD boundary
ye / yE	Yank through end of word / WORD
yb / yB	Yank backwards to word / WORD start
cw / cW	Change word text to next word / WORD boundary
ce / cE	Change through end of word / WORD
cb / cB	Change backwards to word / WORD start
3w / 5j	Move 3 words / down 5 lines

`d3w / 3dw`	Delete across 3 word motions
`y2e / c2b`	Yank through 2 word ends / change backwards 2 words

4. Combine operators with line and file motions

`d$ / D`	**Delete to end of line**
`c$ / C`	Change to end of line
`y$`	Yank to end of line
`d0 / d^`	Delete back to column 1 / first nonblank
`c0 / y0`	Change / yank back to start of line
`dG / yG`	Delete / yank through last line
`dgg / ygg`	Delete / yank through first line (Vim)
`d10G / y10G`	Delete / yank through line 10
`3dd / 3yy / 3cc`	Delete / yank / change 3 whole lines
`3D / 3C`	Delete / change to end of line across 3 lines
`d} / y} / c}`	Delete / yank / change through next paragraph boundary
`d% / y%`	Delete / yank to matching bracket
`d/text ↵`	Delete up to next occurrence of text

5. Yank, delete, then put: live demonstrations

`yy → p`	**Copy current line; paste below**
`yy → P`	Copy current line; paste above
`3yy → 5j → p`	Copy 3 lines; move down; paste below
`dd → p`	Cut line; paste below current line
`dd → P`	Cut line; paste above current line
`dw → p`	Cut word-range; paste after cursor
`yw → p`	Copy word-range; paste after cursor
`"a3yy → "ap`	Yank 3 lines into register a; put from register a
`"Ayy`	Append current line to register a
`"0p`	Put latest yank from yank register 0 (Vim)
`"_dd`	Delete line without changing registers (Vim)

6. Character finds become edit targets

`fx / Fx`	**Find next / previous x on line**
`tx / Tx`	Stop just before / just after x
`df, / dt,`	Delete through next comma / up to (not including) comma
`cf) / ct)`	Change through next) / up to next)
`yf; / yt;`	Yank through next semicolon / up to it
`d; / c,`	Repeat last character-find motion as delete / reverse it as change
`2df,`	Delete through second following comma

7. Vim text objects: an especially useful next step

`diw / daw`	**Delete inner word / a word (includes surrounding space as appropriate)**
`ciw / caw`	Change inner word / a word
`yiw / yaw`	Yank inner word / a word
`di" / da"`	Delete inside quotes / including quotes
`ci' / ca'`	Change inside single quotes / including quotes
`di( / da(`	Delete inside parentheses / including parentheses
`ci{ / ca{`	Change inside braces / including braces
`yi[ / ya[`	Yank inside square brackets / including brackets
`dip / dap`	Delete inner paragraph / a paragraph

`cit / dat`	Change inside HTML/XML tag / delete a tag block (Vim)

8. Visual selections (Vim)

`v / V / Ctrl-v`	**Characterwise / linewise / blockwise Visual mode**
`v → motion → y`	Select by motion; yank selection
`V → 3j → d`	Select four lines; delete
`v → e → c`	Select through end of word; change
`Ctrl-v → jjj → I# → Esc`	Insert # at start of four selected block lines
`gv`	Reselect previous Visual selection

9. Search, substitutions and useful finishing moves

`*/#`	**Search word under cursor forward / backward**
`n / N`	Next / previous search match
`:%s/old/new/g`	Replace all occurrences in file
`:%s/old/new/gc`	Replace all with confirmation
`:10,20s/old/new/g`	Replace in lines 10–20
`>> / <<`	Indent / unindent line
`3>> / 3<<`	Indent / unindent 3 lines
`>ap / =ap`	Indent / auto-indent a paragraph (Vim)
`J / gJ`	Join lines with / without spacing adjustment (gJ Vim)
`~ / gUw / guw`	Toggle case of character / uppercase / lowercase word (gU, gu Vim)
`zz / zt / zb`	Center / top / bottom current line (Vim)
`:w / :q / :wq / :q!`	Write / quit / write-and-quit / quit without saving

Teaching notes / precision points

- Vim is not identical to historic vi: text objects, Visual mode, gg, Ctrl-r, registers such as "0, and several g-prefixed motions are Vim features.
- Operators act on motion ranges; a motion can be inclusive or exclusive. This explains why dw and de differ.
- cw has a useful special behavior: on a nonblank character it typically changes to the end of the word (like ce), not into following whitespace.
- p/P are not synonymous with “paste at the cursor”: characterwise text goes after/before the cursor; linewise text goes below/above the line.
- d and c normally change the unnamed register; y yanks without deleting. In Vim, Y yanks the whole line (unlike D/C).
- For demonstrations, use :set number and optionally :set showcmd; use :set nocompatible for modern Vim behavior.

Suggested video arc: navigate → delete/change/yank → put → counts → line/file targets → character finds → text objects → Visual mode → search and finishing tricks.