vi/Vim practical quick reference

Everyday editing patterns and how commands compose

Opening, saving and exiting

Start an editor with a filename, use Escape to ensure normal mode, and use colon commands for file operations. :w writes changes; :q exits if no unsaved changes remain; :wq writes and exits; :q! exits without saving. :e filename opens another file, potentially requiring you to save or discard changes first.

Vim offers :x, which writes only if modified, and :help for extensive local documentation. Do not confuse the shell command vi with an ex command typed inside the editor.

```
vi /etc/hosts
:w
:q
:wq
:q!
:e otherfile
```

Insert and replace

i inserts before the cursor; a appends after it; I inserts at the first nonblank on the line; A appends at line end. o opens a line below; O opens one above. Escape returns to normal mode. c followed by a motion combines deletion and insertion: cw changes a word, c$ changes through line end, and c} changes to the next paragraph boundary.

The benefit of learning the operator grammar is reuse. Once you know w, e, $, } and f{char}, you can combine them with d, c and y rather than memorising a separate edit command for every region.

```
i  a  I  A  o  O
cw  ce  c$  c}  cfX
```

Find, move and repeat

w advances by word, b moves backward and e moves to word end. 0, ^ and $ navigate within a line. /text searches forward; n repeats the search. fX finds the next X on the current line; ; repeats the same find. % jumps between matching delimiters in Vim and many vi implementations.

Counts are powerful: 5j moves down five lines, 3w moves forward three words and 10G jumps to line ten. A count before an operator can change how much text is affected.

```
5j  3w  10G
/text
n
fX  ;
%
```

Copy, delete and undo

yy yanks a whole line; p puts it below. dd deletes a line; 3dd deletes three. dw deletes through a word motion; d$ deletes to line end. u undoes a change; in Vim, Ctrl-r redoes and . repeats the last change. Because deletion also fills registers, p can often restore recently deleted text, although explicit undo is usually safer.

In Vim, visual selections with v, V and Ctrl-v are convenient for characterwise, linewise and blockwise editing. They are extensions beyond the strictest classic vi subset.

```
yy  3yy  p  P
dd  3dd  dw  d$
u  <Ctrl-r>  .
```

Substitution and configuration

Ex substitution is efficient for repeated changes. :s/old/new/ changes the first match on the current line; :s/old/new/g changes all matches on the current line; :%s/old/new/g changes all matches in the file. Add c for interactive confirmation in implementations supporting it.

:set number shows line numbers; :set nonumber hides them. Vim-specific options and mappings should be kept distinct from portable vi behaviour. Use :help when an unfamiliar command behaves differently on a particular system.

```
:s/old/new/
:%s/old/new/gc
:set number
:set nonumber
```

Reference scope: commands and defaults vary by distribution and software version. Consult the locally installed manual pages for exact syntax and behaviour.