Vim

Modes, motions and operators: edit text without the mouse.

01 Install & open

```
brew install vim       # macOS
sudo apt install vim   # Ubuntu, Debian
winget install -e --id Vim.Vim
vim --version          # version, features
vim notes.txt          # open a file
vimtutor               # 30-minute lesson
```

To quit, press Esc, then type :q and Enter (:q! drops changes). :help opens the manual; :help dd explains dd.

02 Modes

`i` `a`	Insert before, after the cursor
`I` `A`	Insert at first non-blank, at end
`o` `O`	Open a new line below, above
`v` `V` `Ctrl-v`	Visual: characters, lines, block
`R`	Replace mode: type over text
`:`	Command-line mode

Vim starts in Normal mode, and Esc returns to it.

03 Basic motions

`h` `j` `k` `l`	Left, down, up, right
`w` `b` `e`	Next word, back, end of word
`0` `^` `$`	Line start, first non-blank, end
`fx` `Fx`	Next, previous x on the line
`tx` `Tx`	Just before, just after that x
`;` `,`	Repeat f or t, or reverse it

x is any character. W, B and E move by WORD, which runs between spaces. A count repeats a motion: 5j, 3w.

04 Jumps & scrolling

`gg` `G`	First line, last line
`42G` `:42`	Line 42
`%`	Matching (), [] or {}
`{` `}`	Previous, next paragraph
`Ctrl-d` `Ctrl-u`	Half a screen down, up
`zz` `zt` `zb`	This line to middle, top, bottom

Ctrl-f and Ctrl-b move a whole screen. % finds the next bracket on the line, then its match. (and) move by sentence.

05 Marks & jump list

`ma`	Set mark a at the cursor
`'a` `` `a ``	Go to its line, its exact spot
`''` ``` `` ```	Back to before the last jump
`Ctrl-o` `Ctrl-i`	Older, newer jump position
`g;` `g,`	Older, newer change position

Lowercase marks belong to one file; uppercase ones (mA) work across files. G, %, searches and marks count as jumps.

06 Delete, change & paste

`x` `X`	Delete character under, before
`dd` `D`	Delete line, to end of line
`cc` `C`	Change line, to end of line
`yy`	Yank (copy) the line
`p` `P`	Paste after, before the cursor

c deletes, then starts Insert mode; s and S change a character, the line. Deletes are cuts, so dd then p moves a line.

07 Operators & counts

`d` `c` `y`	Delete, change, yank
`>` `<` `=`	Indent, outdent, re-indent
`dw` `d$` `dG`	To next word, line end, file end
`3dd` `2yy`	Three lines, two lines
`c2w` `dt)`	Two words, up to the)

Operator, count, motion: d3w deletes three words. Doubled (dd, yy, >>) it acts on lines. gg=G re-indents the file.

08 Undo & repeat

`u`	Undo the last change
`Ctrl-r`	Redo
`.`	Repeat the last change
`:earlier 5m`	The text 5 minutes ago
`:later 5m`	Five minutes forward again

Dot repeats edits, not : commands (cwnew Esc, then n . at each match). :earlier 1f reverts to the last save.

09 Text objects

`iw` `aw`	Word; aw adds the space
`ip` `ap`	Paragraph
`i"` `a"`	Double-quoted string
`i(` `a(`	Parentheses, also ib ab
`i{` `a{`	Braces, also iB aB
`it` `at`	HTML or XML tag

Use after an operator or in Visual mode: ciw, di(, yi", vip. i is the inside, a adds the delimiters; start anywhere in it.

Vim

Search, files, windows, macros and your vimrc.

10 Visual mode

`v V Ctrl-v`	Characters, lines, block
`o`	Jump to the other end
`> < =`	Indent, outdent, re-indent
`~ u U`	Toggle, lower, upper case
`gv`	Reselect the last selection

d, c and y act on the selection; : runs a command on its lines. In block mode, I or A types on every line after Esc.

11 Search

`/text ?text`	Search forward, backward
`n N`	Next, previous match
`* #`	Word under cursor: next, previous
`/\cfoo /\Cfoo`	Ignore case, match case
`/\<word\>`	Whole word only

g* and g# also match inside words; :noh clears highlights. Patterns are regexes: escape . * [~ to match them as text.

12 Substitute

`:s/old/new/g`	Every match in the line
`:%s/old/new/g`	Every line in the file
`:%s/old/new/gc`	Confirm each change
`:5,10s/a/b/g`	Lines 5 to 10 only
`:'<,'>s/a/b/g`	The selected lines
`:%s/\s\+$//e`	Strip trailing spaces

Flags: g every match (not just the first), c confirm, i ignore case, I match case, e no error if nothing matches.

13 Save, quit & shell

`:w`	Write (save) the file
`:wq :x ZZ`	Save and quit
`:q! ZQ`	Quit, discard changes
`:qa :wqa`	Quit all, save all and quit
`:w copy.txt`	Write a copy, keep editing

:x and ZZ write only if there are changes; :sav new.txt is save as. :!ls runs a shell command, :r !date inserts its output.

14 Buffers & tabs

`:e file`	Open (edit) a file
`:ls`	List buffers
`:bn :bp`	Next, previous buffer
`:b name :b#`	By name, the alternate (last) one
`:tabnew file`	Open a file in a new tab
`gt gT`	Next, previous tab

Buffers are files in memory; windows and tabs view them. Use :b with a number or partial name; :bd closes one.

15 Windows

`Ctrl-w s v`	Split: stacked, side by side
`Ctrl-w h j k l`	Window left, down, up, right
`Ctrl-w H J K L`	Move the window to that edge
`Ctrl-w = _ |`	Equal sizes, max height, width
`Ctrl-w c o`	Close it, close the others

Ctrl-w w cycles windows. :sp file and :vsp file open a file in a new split. Ctrl-w + - < > resize by one.

16 Macros

`qa`	Start recording into a
`$a;<Esc>j`	Example: add a ; and go down
`q`	Stop recording
`10@a`	Run it on 10 more lines
`@@`	Replay the last macro

Any register a to z can hold a macro; qA appends to a. Ending on the next line lets a count walk down the file.

17 Registers

`"ayy "ap`	Yank a line into a, paste it
`"Ayy`	Append a line to register a
`"+y "+p`	System clipboard
`"_dd`	Delete without keeping it
`"0p`	Paste the last yank
`:reg`	Show all registers

"1 to "9 hold deletes of a line or more. ". "% ": "/ hold the last insert, file name, command and search. "+ needs +clipboard.

18 Your vimrc

```
source $VIMRUNTIME/defaults.vim
set number relativenumber
set ignorecase smartcase
set hlsearch incsearch
set expandtab tabstop=4 shiftwidth=4
set autoindent
colorscheme desert
```

Save as ~/.vimrc. Having a vimrc skips defaults.vim (syntax on and more), so line 1 loads it. smartcase needs ignorecase.