\n	toggle relative number				
\d	dispatch and make current file				
\w	toggle word wrap				
F2	YcmCompleter GoTo			tmux	
F3	YcmCompleter GoToDeclaration	Multiple cursors		C - [enter text mode, hjkl to move, /? to search, C-ud to
F4	YcmCompleter GoToDefinition	C-n in Normal	highlights the current word under the cursor in Visual mode/finds next	C-a [page
F5	open nerd tree	C-n in Visual	virtual cursor at every line and leaves you in Normal mode.	C-a:	enter command mode
F8	toggle ctag bar		remove the current virtual cursor and go back to the previous virtual cursor	C-a c	create window
tcomment		C-p in Visual	location.	C-a .	rename window
gc{motion}	Toggle inline comments	C-x in Visual	remove the current virtual cursor and skip to the next virtual cursor	C-a -	split horizontal
gc[Count]c{motion}	Toggle comment with count and motion	C-X III VISUAI	location.	C-a	split vertical
gcc	Toggle comment for the current line	$: \{r\} \\ Multiple Cursors \\ Find$	regex	C-a 19	jump to window
g<{motion}	Uncomment region	Ctrlp		Buffer	
g	Uncomment the current line	C-p	CtrlP file	C-w ghjk	move to the left/bottom/top/right window
g	Uncomment the current region as block	\b	CtrlP buffer	C-w W	move to next window
g>{motion}	Comment region	\m	CtrlP mixed	:sp	horizontal split
g>c	Comment the current line	\r	CtrlP most recently used	:vsp	vertical split
g>b	Comment the current region as block	\t	CtrlP buffer tags	:b	open buffer with filename
gc	Toggle comments (visual mode)	F5	(in CtrlP) purge the cache	:b	open buffer with number
g>	Comment selected text (visual mode)	C-f/C-b	(in CtrlP) cycle between modes	:bd	close current buffer
CTags	· · · · · · · · · · · · · · · · · · ·	C-j/C-k	(in CtrlP) up/down	:bn	next buffer
C-]	jump to tag	C-d	(in CtrlP) switch to filename only search	:bp	prev buffer
C-t	ctags jump back from tag	C-r	(in CtrlP) switch to regexp mode	C-6	last buffer
gd	goto local declaration	Format		:enew	new buffer, with filename creates
gD	goto global declaration	\fx	format XML	:e	reload buffer, with filename creates
g]	list matching tags	\fj	format JSON	:bufdo e	reload buffer all buffers, runs foreach buffer
gC-]	jump to tag if defined once otherwise list matching tags	:%!xmllintformat -	format xml in whole file	A	
:tags	list tag stack	:{range}!xmllint	format xml in selection	\a	switch to the header/cpp file
\ut	update ctags	format -	format xini in selection	\as	splits and switches
Cscope		:%!python -m json.tool	format json in selection	\av	vertical splits and switches
C-\ s	show uses of symbol in quickfix			\at	new tab and switches
Find		v	enter Visual Mode	\an	cycles through next match
\gf (Normal)	(global find) start empty Ag for cpp	C-v	enter Visual Mode	Easy	
\gf (Visual)	(global find) Ag with selection for cpp	C-v	in insert mode adds tab char	Motion	
\f (Normal)	(find)start empty Ag in current file	:retab	fix tabs to spaces	\\ w	easy motion word forward
\f (Visual)	(find) Ag with selection in current file	:so ~/.vimrc	Reload vimrc	\\b	easy motion word back
Replace		:! wc %	run external command on current file	\\ k	easy motion up line
rx	replace all chars in selection with x	:noh	turn off search highlight	\\j	easy motion down line
:{range}sort u	sort and remove dups	vim -d file1 file2	diff 2 files	Errors	
:{range}sort	sort	%	in {range} will use whole file	[q	:cprevious prev error
:{range}sort!	sort reverse	:set pastetoggle]q	:cnext next error
:{range}sort n	numeric sort	:set paste	turn off paste formatting	[Q	:cfirst first line
:s/foo/bar/g	Change each 'foo' to 'bar' in the current line.	:set nopaste]Q	:clast last line
:%s/foo/bar/gc	Change each 'foo' to 'bar' in all the lines., but ask for confirmation first.			:copen	open Quickfix
:{range}s/foo/bar/gc	Change each 'foo' to 'bar' in all lines within a visual selection, but ask for confirmation first.				

 $:!ctags \text{--}R \text{--}c++-kinds=+p \text{--}fields=+iaSl --extra=+q \text{--}links=yes \ .$

remove every line matching pattern

:g/pattern/d

update tag file

:!find . -iname *.c -iname *.c- -iname *.hp -iname *.h -iname *.cp >.cscopelist.tmp; cscope -q -R -b -i .cscopelist.tmp; rm .cscopelist.tmp update csope db