CS 130 Intro to C++**VIM Editor Quick Reference Guide** 1/22/2009

Professor Stan Schuyler

The following table presents a quick reference for looking up operations and commands available in the vim editor. The operations and commands presented are a subset of the available commands in vim. To learn more, use a search engine with the string "vim cheat sheet".

Vim Quick Reference Command Sheet[1]		
Text File Operations	**vim Command**	**Description**
Modes	The term "mode" refers to the state of vim's listening for commands.	
wem	Window edit mode (wem)	Vim listens for simple text edit commands: e.g. "a" means start appending text from the current cursor location.
lem	Line edit mode (lem)	This is the mode entered when the ":" character is entered in window edit mode; a line is opened on the bottom of the terminal window allowing more complex line edit commands to be entered. Examples shown below "Escape."
To escape a current mode.	<esc>	In general, the <esc> key exits whatever mode you are in and returns vim to window edit mode.
To specify a <range> for commands using the ":" command line prefix.	**Illustrated using examples:** • **<number><comma><number>[line editor cmd] e.g. :1,20P means: Print (list) lines 1 through 20 at the bottom of window.** • **<mark><comma><mark>[vim cmd] e.g. :'a,'bm. means: move the lines from the line number in mark "a" to the line number in "b" to the current location of the cursor (".") (where the <marks> "a" and "b" are assumed to be set [see marks below]).** • **:g/<pattern>/[line editor cmd] e.g. :g/data/p means: Find all lines in the file that contain the string "data" and print (list) them at the bottom of the terminal window.**	
To Exit vim (lem)	:q<enter>	Quit Vim. This fails when changes have been made.
	:q!<enter>	Quit without writing.
	:wq<enter>	Write the current file and exit
	:wq <filename><enter>	Write to file named <filename> and quit (exit). E.g. "wq temp.txt"
	:<range>wq! <filename> <enter>	Write only the lines in [range] to the file named <filename>. E.g. "'a,'bwq temp.txt" writes only the lines from 'a to 'b to the file temp.txt
Editing a File (lem)	:e!<enter>	Edit the current file always. Discard any changes to the current buffer. This is useful if you want to start all over again.
	:e <filename><enter>	Edit {file}.
	:e! <filename><enter>	Edit {file} always. Discard any changes to the current buffer.

[1] Adapted from: http://www.ssel.montana.edu/HowTo/

Vim Quick Reference Command Sheet		
Text File Operations	**vim Command**	**Description**
Inserting Text (wem)	a	Append text after the cursor times.
	A	Append text at the end of the line.
	i	Insert text before the cursor.
	I	Insert text before the first non-blank in the line.
	o	Begin a new line below the cursor and insert text.
	O	Begin a new line above the cursor and insert text.
Inserting (read in) a file (lem)	:r <filename><enter>	Insert the file [name] below the cursor. e.g. ":r temp.txt"
	:r!{cmd}<enter>	Execute {cmd} and insert its standard output below the cursor.
Copy and paste line(s) within the file ("t" for "type" the lines) (lem)	: <range>t<location><enter>	Copy the lines in the range given a paste them in the location specified (usually "." meaning the current cursor position. Usually you use "marks" to establish the <range> e.g. ":10,19t." or e.g. ":'a,'bt." Where 'a and 'b are set as indicated next.
		To mark a line as "a" :ma<enter> Marks a position as 'a To mark a line as "b" :mb<enter> Marks a position as 'b Move cursor to desired location for the lines to be copied to.
Deleting Text (wem)	<Del>[count] or x[count]	Delete [count] characters from the cursor to the right. [count is 1 by default]. **NOTE: "[" and "]" mean optional, you do not type them.**
	X[count]	Delete [count] characters before the cursor
	d{motion}	Delete text that {motion} moves over The current {motion} characters are: w – word as in "dw" for delete word
	dd[count]	Delete [count] lines
	D	Delete characters from the cursor to the end of the line; e.g. "<shift> d"
(lem)	:[range]d<enter>	Delete [range] lines (default: current line; e.g. ":1-10d" deletes lines 1–10.

Vim Quick Reference Command Sheet		
Text File Operations	**vim Command**	**Description**
Changing (or replacing) Text (wem)	r{char}	Replace 1 character under the cursor with {char}; e.g. “rc” replaces whatever the character is at the cursor with “c”.
	R	Enter Insert mode, replacing characters rather than inserting (like insert mode in MS Word[tm].
	~[count]	Switch case of the character under the cursor and move the cursor to the right. If a [count] is given, do that many characters.
Marks (wem)	m{a-zA-Z}	Set mark {a-zA-Z} at cursor position (does not move the cursor, this is not a motion command). E.g.: “ma” simply associates the current line number with the label “a”.
	'{a-zA-Z}	Refers to the first non-blank character on the line marked as an {a-zA-Z}. E.g. if “ma” was done on line 11, then ‘a refers to line 11.
(lem)	:marks	List all the currently defined marks (not a motion command).
To write (save) to a file at any time (lem)	:w<enter>	Write all lines to the current file.
	:w <filename><enter>	Write all lines to the file named <filename>. E.g. “w temp.txt”
	:<range>w <filename> <enter>	Write only the lines in [range] to the file named <filename>. E.g. “’a,’bwq temp.txt” writes only the lines from ‘a to ‘b to the file temp.txt

Substituting Text (lem)	: [range]s/pattern/string/[c][g][p][i]	For each line in [range] replace a match of “pattern” with “string”.
	Example : g/pattern/s//string/p<enter>	For each line in the file with “pattern”, substitute with “string” all matching “patterns” on the line and print the line (the trailing command “p”, optional).
	The optional trailing arguments that you can use for the substitute commands:	[c] Confirm each substitution. Vim positions the cursor on the matching string. You can type: 'y' to subst. this match 'n' to skip this match [g] Replace all occurrences in the line. Without this argument, replacement occurs only for the first occurrence in each line. [i] Ignore case for the pattern. [p] Print the line containing the last substitute.
Searching (lem)	/pattern/<enter>	Search forward for the next occurrence of “pattern” (a string).
	//<enter>	Search forward for the next occurrence of the last used “pattern”.
	?pattern? <enter>	Search backward for the previous occurrence of “pattern”.
	?? <enter>	Search backward for the next occurrence last used “pattern”.
Undo/Redo/ Repeat	u	Undo [count] changes.
	:u<enter>	Undo one change.
	CTRL-R[count]	Redo [count] changes which were undone.
	:red[o]<enter>	Redo one change which was undone.
	U	Undo all latest changes on one line.
	.	Repeat last change.

Moving Around (wem)	Basic motion commands: uses the h, j, k, l keys as illustrated, or the arrow keys) (up) k ← h l → j (down)	h or h[count] characters to the left (exclusive). l or l[count] characters to the right (exclusive). k or CTRL-P [count] lines upward j or CTRL-J [count] lines downward.
	0	To the first character of the line (exclusive).
	<Home>	To the first character of the line (exclusive).
	^	To the first non-blank character of the line
	$ or <End>	To the end of the line and [count - 1] lines downward
	- [count]	[count] lines upward, on the first non-blank character (linewise).
	+ [count]	Count lines downward, on the first non-blank character (linewise).
	G[count]	Goto line [count], default last line, on the first non-blank character.
	gg[count]	Goto line [count], default first line, on the first non-blank character.
	word	A word consists of a sequence of letters, digits and underscores, or a sequence of other non-blank characters, separated with white space (spaces, tabs,). This can be changed with the 'iskeyword' option.
	WORD	A WORD consists of a sequence of non-blank characters, separated with white space. An empty line is also considered to be a word and a WORD.