Random Notes About vi Editor.

1 Command mode operations:

1.1 Entering Input mode.

i	Insert at current cursor location.
I	Insert at the start of line.
a	Append right of the cursor.
A	Append at the end of line.
o	Open below the current line.
O	Open above the current line.

1.2 cursor movement .

h	move to left. If it is follows *n* then movement will be n times.
l	move to right.
k	move up.
j	move down.
b	move to begining of word.
e	move to end of word.
w	move to start of next word.
B	move to begining of blank space seperated word.
E	move to end of blank space seperated word.
W	move to start of blank space seperated next word.
0/^	move to start of line.
$	move to end of line.
n|	move to *n* th coloumn.
(/)	starting/end of sentence.
{ /[/ } /}	start/end of paragraph.
%	matching bracket.
f*ch*	find *ch* character in forward direction in the current line.

1.2 cursor movement .

F*ch*	find *ch* character in reverse direction in the current line.
t*ch*	find *ch* character in forward direction in the current line, place the cursor before *ch*.
T*ch*	find *ch* character in reverse direction in the current line, place the cursor before *ch*.
;	repeat the seach operation by f/F/t/T.
,	repeat the seach operation by f/F/t/Tin reverse direction.

1.3 Positioning cursor in a page.

***n*G**	move to nth line in the file. Onle G will move the cursor to last line of file.
:*n*	also moves to nth line.
<ctl>f	position the cursor in forward frame.
<ctl>b	position the cursor in backward frame.
<ctl>u	position the cursor in up in the frame.
<ctl>d	position the cursor in down in the frame.
H	Move cursor to first line being displayed.
M	Move cursor to midlle line being displayed.
L	Move cursor to last line being displayed.
***n*<ctl>e**	scrol forward n lines.
***n*<ctl>y**	scrol backward n lines.
z<enter>	puts current line on the top of screen.
z.	puts current line at middle of screen.

1.4 delete/change/yank etc.

d- for delete.
c- for change
y- for yank.

each of these operators can follow a numerical count and should be followed by a cursor movement (any from table 2). For example **cw** changes till end of word.

If these operators are used in succession then the operation is done on the complete line. The table below gives some examples.

***n*dd**	for delating the *n* lines starting from current line.
***n*dw**	for delating the *n* words starting from current word.
df*ch*	for deleting all characters till *ch* is found, including *ch*.
dt*ch*	for deleting all charecters before *ch*.(functions of **f** and **t** are in next tables.)
D	same as **d$**
Y	same as **yy** , not as **y$**.
C	same is **c$**.
s	same as **cl**
S	same as **cc**
x	same as **dl** . deletes the character at the cursor.
X	same as **dh** . deletes the character left of cursor.

1.5 Other command mode operations.

p	paste the content of buffer loaded by delete/yank command to the right of cursor .
P	paste the content of buffer loaded by delete/yank command at the cursor .
u	undo last change.
U	undo all the last unsaved changes in current line.
.	repeat the previous change.
J	Join next line with current line.
r*ch*	replace the character with ch.
R	replace characters till *esc* .
m*ch*	mark the current location with character *ch.*
'*ch*	move to mark *ch.*
"	move to previous location.
"*ch*y	yanked chars are stored in in a buffer *ch.*

1.5 Other command mode operations.

"*ch***d**	deleted chars are stored in in a buffer *ch.*
"*ch***p**	content of buffer *ch* are pasted.
>>	inserts a tab space.
<<	removes a tab space.

2 ex mode operations:

2.1 Search and replace operations:

search commands.

/<*pattern*>	forward search.
?<*pattern*>	reverse search.
/<*pattern*>/+***n***	go to *n*th line after *pattern* is found.
/<*pattern*>/-***n***	go to *n*th line before *pattern* is found.
n	repeat previous search.
N	repeat previous search in opposite direction.

Metacharacters for search.

^	start of line.
$	end of line.
\<	start of word.
\>	end of word.
[ ]	range/set of characters.
*****	zero or more characters.
.	one character.

Examples of substitute commands.

:*m***,***n* **s/str1/str2/g**	replace all occurances of str1 with str2 between lines *m* and *n.* **g** is for globally- all occurances in the line.

Examples of substitute commands.

:.,$ s/str1/str2/g	replace all occurances of str1 with str2 between current line and end of file.
:1,.s/str1/str2/g	replace all occurances of str1 with str2 between current line and start of file.
:%s/str1/str2/gc	replace all occurances of str1 with str2 in the file. **%** indicates entire file. **c**- for interactive substitution.
:%s/str1/&_str2/ g	replaces all occurances of str1 with str1_str2. Such a need arises when str1 is matched using metacharacters.
Note: All the metacharacters in the above table can be used for these substitution commands.	

2.2 other ex commands :

:*n*d	deletes nth line.
:*m*,*n*d	delates all lines from m to n.
:m **mo** p	Moves mth line after pth line.
:m **co** p	Copies mth line after pth line.
:m,n **mo** p	Moves m-n lines after pth line.
:m,n **co** p	Copies m-n lines after pth line.
:m,n **w** <*filename*>	writes m-n lines of this file to *filename* file.
:w	write the opened file.
:wq	write and quit.
:w <*filename*>	save as *filename*
:w! <*filename*>	save as *filename* although no write permission.
:w!	forced write. write even if permission is not there.
:q	quit.
:q!	quit without saving.
ZZ	same as **:wq.**
:e!	edit the file again discarding changes.

:e <*filename*>	edit file <*filename*> in the buffer.
:e#	edit alternate file.
:e!#	edit alternate file discarding the changes.
:n	open the next file already loaded.
:rew	rewind and open first file in the buffer.
:!*cmd*	unix *cmd* is executed. :!% runs the current file. % is for current file name. This will be very much useful while checking the shell scripts.
:r <*filename*>	read content of *filename* at the cursor position.
:r !*cmd*	read output of *cmd* at the cursor location.
:abr <*abr*> <*long-form*>	*abr* is abbrivated to *longform.* Example alw can be aliased for always @ (posedge clk) \n begin \n .
:abr	lists all abbrivations.
:una <*abr*>	unabbrivate.
:map *key keyset*	*key* is mapped to *keyset*. to use special characters (like enter /esc/function keys) press ctl-alt-v followed by the special key. to use f1-f12 we can use #1,#2 ... in the map command.
:map! *key keyset*	*key* is mapped to *keyset* in append mode.
:g/str1/d	Deletes all lines that contain the str1.
:g/str1/p	Prints all lines that contain the str1.

2.3 Environment variables of vi.

ai/autoindent	autoindent.
ic/ignorecase	ignore case.
showmode	displays the mode.
nu/number	displays line number.
sm/showmatch	shows matching bracket.
beautify	beautifies.

magic	If set "**.**", "**[**" and "*" have special meaning.
wm/wrapmargin	If this variable is set to a value *n* then while typing in input mosd a line feed is automatically inserted at the start of current word as soon as (width -n) characters are typed.
wa/writeany	can overwrite an existing file without warning. (never use it)
aw/autowrite	writes the file for every shell escape. Very much useful while testing the shell scripts using :!%.