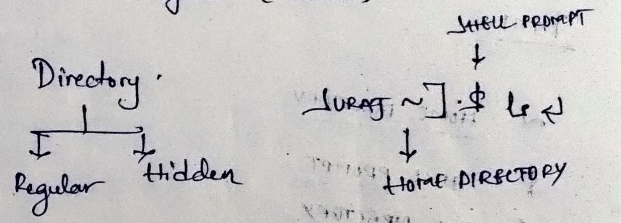
ELECTRONIC DESIGN AUTOMATION → EDA

FILES & DIRECTORIES

* Linux – independent of file extension
  (.tcl, .txt, .py, .png, .jpeg)

* files ← Hidden (Not visible directly) privacy
  Regular (visible)



* Hidden files and directories are represented with a dot in front of the name

* Hidden files are not normally displayed from the the user because to maintain the privacy of the file and to keep the file (in the contents of the file) from accidently or intentionally modified.

* Each and every EDA tools has the respective hidden files which are the configuration hidden file the helps the invoke properly by following certain procedures, and also to execute the tasks which are requested by the user

SYNOPSIS → . Synopsis_dc_setup

PD → . Synopsis_ice2_setup

LAYOUT → . Synopsis_cc_setup.

PATH

~ HOME DIRECTORY

$ SHELL PROMPT

/ ROOT DIRECTORY

~] $ PWD ↵

↳ present working directory

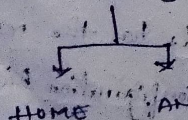
ABSOLUTE PATH

* Absolute path is the path that starts from the root directory

/ VE / PD / VISWAS / SURAJ

RELATIVE PATH

* present directory toward destination file/dir



~] $ ./VISWAS/

* . PRESENT DIRECTORY

* .. ONE DIRECTORY UPWARD

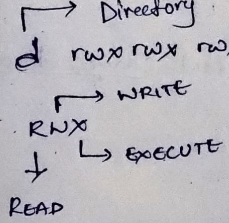
./../ PD

↧

PWD

<Directory_name>

* tree → tree structure of the directories available



⇒ Cal → Calender

⇒ date → date and time

⇒ echo → print

⇒ echo $ PATH

↓

Extracts the values of the variable.

cd → Change directory.

$ which VCS
which verdi
which dc_shell
} to search tools wheather installed or not

* To create a directory;

Syn: ~] $ MKDIR VE ↵

* To get in to the directory; (or) Change directory.

Syn: ~] $ CD VE ↵

↓

VE] $

@tgt_setup-1 ~]

↓

machine name;

COMMANDS;

* ls [LIST]

ls Command will list (or) display the files and directories belonging to the specified directory. It normally displays the contents of the directory in ascending alphabetical order

Examples;

Syntax; ls [-OPTION] [DIRECTORY_PATH]

↓

-l -r

-a

-d

-t

Additional flags and their Functionalities;

Short Flag	Long Flag	
* -l	No long flag	Show result in long format

-rw-rw-r--. 1 ve_balakumara ve_balakumara 16 Aug 7 09:48 cat1.tcl

-rw-rw---- 1 ve_balakumara ve_balakumara 4 Aug 7 10:16 Cat2.txt

* -S	No long flag	Sort results by file size

]$ ls -LS (or)]$ ls -L-S

-rw-rw-r--. 1 ve_balakumara ve_balakumara 16 37478 Aug 6 10:20 project_V1_31_08_202

* -r	--reverse	Show files & direction in reverse order
* -a	--all	Show all files includes hidden files

* to list hidden files in Linux

Syn: ~] $ ls -ld .*

(or)

Syn: ~] $ ls -d .*

(Wild card characters)

* to list only the directories.

~] $ ls -d */

~] $ Music/ Song/ image/

* to list only the directories hidden

~] $ ls -ld .*/

Column 1	2nd	3rd	4th
File permission	No. of links available to the file	Owner of the file	Group of the file

5th	6th	7th
Size of the file	Date of modification (or) time stamp	File name

* To print the number of files of a particular extension in a path;

~] $ ls *.log | wc -l

(pipe operator)

word count

* to print all the files.

[ve_balakumara@frg_sktpp ~] $ ls | wc

ls -F

The ls -F command is used to list files and directories in the current directory, with a special character appended to each entry to indicate its type. Here are the common indicators

* / for directories
* * for executable files
* @ for symbolic links
* | for FIFO's (named pipes)
* = for sockets.

* ~] $ chmod 777 fibonacci.log

It is used to change the permission of the particular files.

drwxrwxrwx

	R W X	O (rwx)	G (rwx)	U (rwx)
0 ⇒	0 0 0	0	0	0
1 ⇒	0 0 1	1	1	1
2 ⇒	0 1 0	2	2	2
3 ⇒	0 1 1	3	3	3
4 ⇒	1 0 0	4	4	4
5 ⇒	1 0 1	5 (R-X)	6 (RW-)	7 (RWX)
6 ⇒	1 1 0	6	6	6
7 ⇒	1 1 1	7 (RWX)	7 (RWX)	7 (RWX)

chmod 777 a.txt.

* CD [Change Directory]

CD is used to change directory

① Change the current working directory.

cd <specified-directory-path>

② To change current directory to home directory

syn! ~] $ cd ~ (or) cd

③ Change to the previous directory;

cd -

This will also echo the Absolute path of the previous directory.

* Adding a .. will allow you to move up from existing directory.

CAT COMMAND

* CAT stands for Concatenate

* The CAT command ~~is used to~~ allow us to Create single or multiple files, to view the content of a file or to concatenate files and redirect the output to the terminal or files

EXAMPLES:

① To display the content of a file in the terminal.

~] $ CAT <file name>

② To display the content of multiple files in terminal.

CAT file1 file2

③ To create a file with the cat command.

~] $ CAT > Suraj.txt

My name is Suraj
VLSI EXPERT !

Cntrl + D

④ To display contents of all files belonging to same filetype.

Cat *.<filetype>

⑤ To display the content of all the files in Current directory.

Cat *

⑥ To redirect the content of one file into another file.

old file new file

Cat <file1> > <file2>

(overwrite)

Cat bala.txt > Calz.txt

↳ Concatenate operator.

Cat Calz.txt

⑦. Use Cat command with more and less option.

Cat file name | more

Cat file name | less.

* It will display more no of lines over the screen
* press 'q' to quit from the command

⑧ Append the contents of file 1 to file 2.

Cat file 1 >> file 2
↳ redirection operator (for multiple)

Cat Cat 1.txt >> Cat 2.txt.

* It will adds the contents of file1 to file2.

⑨ To concatenate two files together in a new file.

Cat file1 name file2 name >> file3 name
↳ (Append or add)

⑩. Some implementations of cat, with option -n, its possible to show line number

Cat -n file1 file2 > new numbered file name

⑪. Cat >> file name

>> operator is used to add content in existing file.

PWD COMMAND

* Stands for Present working Directory
* It prints the paths from the Current ~~for~~ working directory.

Help command in linux:

(ls) -- help

mkdir --help

MKDIR COMMAND

① Make a directory named myfiles.

Syn: ~]$ mkdir myfiles.

② Create a directory named myfiles at the home directory.

Syn: ~] $ mkdir ~/myfiles.

③. Create the mydir directory, and set its file mode (-m) so that all users (a) may read (r), write (w), and execute (x) it.

$ mkdir -m a= rwx mydir
↓
user

ECHO COMMAND

* Echo command allows to display the line of text / string that is passed as an argument.

EXAMPLES:

① To show the line of text or string passed as an argument

Syn: echo Hello there

② To show all files / folders similar to the ls command

Syn: echo *

③ To save text to a file named foo.bar;

echo "Hello there" > foo.bar

④ To append text to file named foo.bar.

echo "Hello there" >> foo.bar.

MAN COMMAND

The man command is used to display the manual of any command that like, Descriptions, options;.

ALIAS

. Customizing command behaviour in linux is easy the alias command. To make these changes permanent, follow these steps:

* The .bashrc file is a script file that is executed when a user logs in
* The file itself contains a series of configurations for the terminal session
* This includes setting up or enabling;
  Colouring, Completion, shell history, command aliases.
* It is a hidden file

```
alias c='clear'
alias l='ll'
alias p='pwd'
alias d='ls -l -d */'
alias t='ll -l -t'
alias h='history'
alias i='ls -a | grep "^\."'
alias wmi='whoami'
alias s='ls -th'
```

TOUCH COMMAND

* to create empty files we use touch command
* This command can used when ever the user do not to store any values.

* Create an empty:
  to create an empty file. simply use;

  Syn: touch <file name>

* Create multiple files:

  Syn: touch file1. file2. file3.

  auto generate filenames (file type)

  Syn: touch newfile. -c{1..10}. txt

COPY COMMAND

* When you have multiple source files and want to copy them to a directory.

  Syn: cp source_file. source_file2 ... Dest_directory

* All source files are copied to dest_directory with the same names
* If dest_directory doesn't exist, it's created. Existing files are over-written

Syntax:

* Syn: cp file1.txt file2.txt file3.txt backup/
* Syn: cp -r dir1 dir2 cp - interactive

MOVE COMMAND

(1)
* Move command is used for moving and renaming files and directories
* move a file to different directory

  Syn: mv source file. target file/directory

(2)
* to move multiple files at once, provide all the file names followed by the destination dir.

  Syn: mv file1.txt file2.txt. file3.txt. target_dir

* Alternatively use a glob pattern to match files with a specific extension.

(3) Syn: mv *.txt target_dir

* Rename a file

  use the mv command to rename a file, Specify the new file as the destination.

  Syn: mv source_file. target_file/directory/target_file

  Ex: mv bala. /11TI_4FA-FF/bala1

* If target-file doesn't exist, it is created. If it does, it is overwritten without warning.

*

HOST NAME

* It will be used to return machine-name

Syn: hostname

ENV

* It displays all the linux env variable settings
* Includes vabables like PATH variable, PWD, HOSTNAME, USER, HIST SIZE, SHELL, HOME etc
* All environment variable settings are in CAPS.

CLEAR

* The clear command in linux is used to clear the terminal screen.
* Basic Usage
  - Simply type clear in your terminal and ↵ press Enter

⇒ RM :-

* It is used to remove files, directories, and symbolic links

* Remove single file

① Syn: rm filename

* Ex. to delete a file named file.txt, run:

Syn: rm file.txt.

② Remove multiple file (-i -r)

to remove multiples files at once, list their names seperate.

Syn: rm file1 file2 file3.

Remove Directories

① * rm -r . is to delete parent dir and also its associated sub dir

"-r" stands for recursive

② * rmdir is to delete empty directories

③ Interactive deletion:

-i (interactive) rm -i filename

used for confirming before deletion

Ex: rm -i important.docx.

Sort

* Sort command is used to arrange the contents of the text files.
* Sort file content in alphabetical order by checking the first letter / character of text files.
* Sort file content in numerical order by checking the first number
* Sort file content both in alpha and numberical order
* Sort file content in reverse order

Syn: sort <title> -r

Ex.1:
```
sort -n num.txt
3
4
12
13
55
100
```

Ex.2:
```
sort alpha_num.txt
2
23
3
aaa
bala
```

Ex.3:
```
sort -r num.txt
55
4
3
13
12
100
```

Ex.4:
```
sort -n -r num.txt
100
55
13
12
4
3
sort -r alpha_num.txt
```

Ex.5:
```
sort
Apple
Cat
fish
Gold
5
9
10
12
```

EXIT

* If you want to exit from the shell, you have to exit command.

Syn: exit

HEAD / TAIL

head GoKul.txt

* head command displays the first 10 lines of the file.

Syn: head <filename>

* tail command displays the last 10 lines of the file

Syn: tail <file_name>

⇒ WORD COUNT :

* We - prints new line, word, and byte count for each file

Syn: wc <file_name>

Ex: wc suraj.txt
explore
1
2
3
Banglore

5 (5) 17 suraj.txt
line count, word count, characters count

- -l : num of lines
- -w : num of words
- -c : num of bytes (characters)
- -m : number of characters [useful for multibyte encoding]
- -L : Displays the length of the longest line

FIND

* The find command can be used to search files in a directory hierarchy.
* to list the hidden files use -name option with a dot (.*) as the arguement

Syn: ~]$ find . -type * f -name ".*"

* the -type option tell to find command which type of files that you want to type.
* The most common type of files is regular file denoted with -f.

but there are other types of files, such as directories (d), symbolic links (l), and so on

hidden directories : find :/ -type d -name ".*"

hidden files : find ./ -type f -name ".*"

FOND : SEARCH

⇒ for searching files and directories in a path

* find / home -name → search "/home" directories for files name "index" recursively

find ./FULL_ADDER_LIB_DEMO/ -name lib.ndm

* find / home -size +100000k → find files over 100000k size in /home directory
10 MB 1 MB

* # find all files with .txt extension in the current directory and subdirectories

find . -name "*.txt"

find files that are larger than 1MB in home directory.

find ~ -size +1M

find that are greater than 1k in home directory

find ~ -size +1k

find all the files.

Limit the depth of the search

* lot of dir/sub dir. search in dir/subdir consumes time to reduces search time mention "max depth".

MINDEPTH;

* Min depth specifies the minimum depth at which the search should start.
* -min depth 2 means it should start from the level 2 directories and ignore the directories before level 2.

Syn; find ./Tool_based_flow/ -mindepth 2 -maxdepth 10 -name attach

Run

* Run is a case-insensitive search
* By default Run is case sensitive, we can use run a case insensitive search with the given name by using -iname instead of -name

Syn; find . -type f -iname SEARCH_NAME

Syn; find . -type d -iname SEARCH_NAME

To find no. of directories in a path;

Max_depth - represent the

Syn; find ./Tool_based_flow -maxdepth 2 -type d | wc -l

32

Syn; find ./tool_based_flow - maxdepth 2 -name "*.log" -type f | wc -l

8.

Syn; find ./tool_based_flow -maxdepth 2 -name "*.log" -type f

- Path -

GREP:

* Grep is filter that searches a file for a particular patterns of characters and displays the lines that contains in the files.
* The pattern that is searched in the file is reffered to as the regular expression (grep stands for global search for regular expressions)

~]$ grep [options] pattern [filename]

options		Description
(1)	-c	→ This prints only a count of the lines that match a patterns
(2)	-h	→ Displays the matched lines but do not displays the filenames
(3)	-i	→ Ignores, Case for matching.
(4)	-l	→ Displays list of a filenames only
(5)	-n	→ Displays the matched lines and their line number
(6)	-v	→ This prints out all the lines that not matches the pattern.
(7)	-e exp	→ Specifies expression with this option. Can use multiple times
(8)	-f file	→ Takes pattern from file, one per line
(9)	-E	→ Treats pattern as an extended regular expression (ERE)
(10)	-w	→ Match whole word.
(12)	-o	→ prints only the matched part of a matching line, with each such part on a seperate output line
(13)	-A n	→ prints searched line and n lines after the result
	-B n	→ Prints the searched line and n line before the result.
	-C n	→ Prints searched line and n lines after & befor the result

(1) Case-insensitive search : "UNIX", "Unix", "unix"

grep -i "Unix" greetfile.txt

-i, --ignore-case

Output: Highlights the search word irrespective Case all over the file.

grep

② Displaying the count of number of matches.

-c, --count : prints only a count of matching lines per FILE.

we can find no. of lines that matches the given string / pattern

Syn: grep -c "unix" greekfile.txt

③ Display the file names that matches the pattern

-l, --file-with-matches

we can just display that contains the given string / pattern.

Syn: grep -l

④ Checking for the whole words in a files

-w option to grep makes it match only the whole words.

Syn: grep -w "unix" greekfile.txt

⑤ Displaying only the matched pattern:

By default grep displays the entire line which has matched string. we can make the grep to display only the matched string by using the -o option

Syn: grep -o "unix" geekfile.txt

unix
unix
unix
unix
unix

[grep -o -i "apple" file1.txt.
grep -o "apple" file1.txt.]

⑤. Show line number while displaying the line matching

It print the line number where the search string / pattern matches.

grep -n "APPLE" file.txt

5: The fruit name is apple.

(7). Inverting the pattern match

You can display the lines that are not matched with the specified search string or pattern using -v.

Syn: grep -v "APPLE" file1.txt.

apple
Apple

Syn: grep -v -i "APPLE" file.txt

Syn: grep -v "unix" geekfile.txt.

learn operating system.

(8). Matching the lines that starts with the string:

* The ^ regular expression pattern specifies the start of a line
* This can be used in grep to match the lines which starts with the given string or pattern.

Syn: grep "^unix" geekfile.txt

(9) Matching the lines that ~~start~~ end with a string.

The $ regular expression pattern specifies the end of a line. This can be used in grep to match the line which end with given string (or pattern.

Syn: grep "pattern$" "filename.txt"

(10) Specifies the expression with -e option:

-e, -regexp = PATTERN use PATTERN for matching

Syn: grep -e "Ag.Pattern" -e "Pattern" -e "Pattern" "filename"

(11) -f file option takes pattern from file, one per line

Syn: grep -f pattern.txt geekfile.txt

(12). Print n specific lines from a file

-A prints the searched line and after the result,

-B prints the searched line and n lines before the result

-C prints the searched lines and n lines and after and before result

Syn: grep -A [No.of lines(n)] [search] [file]

grep -B [No.of lines(n)] [search] [file]

grep -C [No.of lines(n)] [search] [file]

(13). Search recursively for a pattern in the directory

* -R prints the searched pattern in the given directory recursively in all the files.

* Syn: grep -R [search] [directory]

Usage of ls and grep

* to find the number of directories in a path

-c for count

Syn: ls | grep -c ^d

Syn: ls | grep Adi

dc_demo
dc_full adder.v

⇒ user, group and other option in linux file permission

Reference	Class	Description
'u'	user	The user permission apply only to the owner of the file or directory, they will not impact the action of other users
'g'	group	The group permissions apply only to the group that has been assigned to the file or directory. they will not affect the action of other users.
'o'	others	The other permissions apply to all other users on the system, this is the permission group that you want to watch the most
'a'	all there	All three permissions (users, groups, others).

Change multiple permissions at once

Ex: if you want to take all (revoke) permissions away from everyone you would type

Chmod ugo-rwx xyz.txt

for revoking all permissions

Chmod a-rwx xyz.txt

⇒ READ & WRITE FOR USER/OWNER, GROUP REVOKE EXECUTE FOR OTHER USERS.

Syn: chmod ug+rw, o-x abc.mp4

The code is to add read (r) and write (w) permission to both user (u) and group (g) and revoke execute (x) permission.

⇒ ADD REMOVE OR REVOKE.

chmod ug=rw, o+r abc.mp4

⇒ OCTAL NOTATIONS FOR ASSIGN/REVOKE FILE PERMISSION.

Ex: chmod ugo+rwx. [filename]

chmod 777 [filename]

chmod u=r, g=rx, o=rw filename

29th OCT

VIM EDITOR IN LINUX

* Vim, short for Vi Improved
* Vim is a highly configurable text editor built to enable efficient text edition
* It's an enhanced version of the Vi (Visual) editor
* Vim is Case Sensitive.

MODES IN VIM EDITOR

* Normal mode
* Insert mode
* Visual mode
* Command line mode

NORMAL MODE

* Normal mode is a default mode when you open a file
* In this mode, you can navigate and perform various operations on text
* Commands like delete, copy, paste, undo, redo etc

INSERT MODE

* to enter insert mode and start typing (or) editing text do the following

(i) Press i (for insert before the cursor)

* Press a (for insert after the cursor)
* Press I (for insert at the beggining of the line)
* Press A (for insert at the end of the line)
* The Esc key will take you to the ~~command~~ Normal mode from insert mode

VISUAL MODE

* Visual mode allowes you to select text. You can enter Visual mode by pressing v.
* Press V ; to select entire lines and delete (visual mode)
* Ctrl + v ; to select particular selection object across multiple lines and deleting. (visual block)

'x' (or) 'd' for deletion

COMMAND LINE MODE

* This mode lets to enter Vim commands
* To enter command line mode from normal mode

Press or use : (colon)

BASIC COMMANDS

(1) COMMANDS TO CREATE OR EDIT A FILE

* Vim filename

Ex: Vim new.txt

(2) To save and quit the editor

* before writing saving or quit command you have to press colon (:)
* Colon allows you to give instructions to vi

Commands	Action
:wq	Save and quit
:w	Save
:q	quit
:w fname	Save as fname
ZZ	Save and quit
:q!	Quit discarding changes made
:w!	Save (and write to non-writable file)

⇒ To switch from NORMAL mode to insert mode:

Command	Action
o	Start typing on a new line after the current line
O	Start typing on a new line before the current line

⇒ To move / navigate around the file:

Commands	Action
j	To move down
k	To move up
h	To move left
l	To move Right

⇒ To jump lines (top / bottom of file)

gg — move the cursor to first line from anywhere

G — move the cursor to last line from anywhere.

⇒ To delete:

Command	Action
x	Delete the current character
X	Delete the character before the cursor
r	Replace the current character
xp	Switch two characters
dd	Delete the current line
D	Delete the current line from Current character to the end of the line
dG	Delete present line to the end of the line

⇒ To repeat and undo:

u — Undo the last command

. — Repeat the last command

Cntrl +r redo.

⇒ Command to cut, copy and paste!

* dd — Delete a line
* yy — (Yank Yank) copy a line
* p — Paste after the current line
* P — paste before the current line

⇒ Command to cut, copy and paste in blocks

<n> dd — Delete the specified n no. of lines

<n> yy — Copy the specified n no of lines

⇒ Joining lines!

J — Join two lines

yyp — Repeat the current lines

ddp — Swap two lines

⇒ Move forward or backward!

* w — move one word forward
* b — move one word backward
* <n> w — move specified number of words forward
* dw — Delete one word
* yw — copy one word
* <n> dw — delete specified number of words.

move cursor

^ - Start of the line — Shift ^ - move to start

$ - End of the line — Shift $ - move to end of the line

* to delete a particular line

: <line number> d

* to delete group of lines!

: <start line num, end line num> d

6th NOV

SYNTAX HIGHLIGHTING:

* In the **Vi** editor, the commands ':Syntax on' and ':Syntax off' are used to **enable and disable Syntax highlighting**,
* Syntax highlighting is a feature that displays text, especially source code, in different colours and fonts according to the category of terms, making it easier to read and understand.
* Syntax on and Syntax off can be used for current session, when

DISPLAY / HIDE LINE NUMBERS

* to display absolute line numbers use any of the following commands

Syntax:
```
:set number
or
:set nu
```

* To hide absolute line number use any of the following commands

Syntax:
```
:set nonumber
or
:set none
```

JUMPING TO LINE NUMBER

:500
↳ line number
(shift the cursor)

:f
↳ Gives the number of lines

* To display file name after opening the editor
the command to display the line name type ":f" in command mode

* To open another file without exiting file/editor:
:e filename.

SEARCH

* Search a pattern in a file there are two ways to search a pattern they are;
  * forward search
  * Backward search
* forward search will search the pattern from the top to the bottom of the file
* Backward search will search the pattern from the bottom to the top the file

forward search;

Syntax; /<word> to search

Backword search;

* to search a word from bottom to the top of the file.

Syntax; ?<word to search>

n = next search pattern for top to bottom search

N = Next search pattern for bottom to top search.

SEARCH & REPLACE;

:nohl - no highlight

* to search a pattern and replace with another pattern

Syntax; :% /get/got

dd search pattern

new pattern to be replaced

TO SEARCH IN EVERY LINE;

Syntax; :%s/get/got

Replaces first occurence

Syntax; :%s/get/got/g

replace every search pattern

* for line by line (or) word by word replacement

Syntax;

:s%/get/got/gc

y/n/q/l/a/^E/^N.

y - yes

n - no

q - quit at the instance

l - replace at the instance and quit

a - replace all

* for replacing at a particular line;

Syntax; :9s/get/got/gc

* To delete the text before the cursor

d^

* To delete the text from the cursor

d$

* to Comment the first 10 lines.

:1,10s/^/#

* to un comment

:1,10s/#/_

* to Comment all the lines in the file

:%s/^/#

:15,$s/VLSI EXPERT/$

* To add a word VLSI EXPERT for end of every line

:15,$s/VLSI EXPERT/.

* To replace the VLSI EXPERT with EXPERT VLSI

:15,$s/EXPERT VLSI$/EXPERT VLSI
VLSI EXPERT

⇒ VISUAL BLOCK FOR REPLACING A GROUP OF SIMILAR WORDS;

* Press Cntrl+v to select group of similar words
* Press x to delete selected group of words.
* Press Cntr+v where you want to add a new word
* Shift i and enter the pattern you want to insert
* Press Esc once or twice to see the changes

* DU

du → disk usage of files and directories

Syn: du -sh - Shows size of "Directory"
Shows size of whole directory which includes sub directories and files

du -h - Each and every directory

* DF

Syn: df -hi - Disk usage in human readable format

* Umask - Sets default permissions for new files

* Passwd
Change user password.

* Unalias - Removes the alias.

* rename - Rename multiple files

Syn: rename old_pattern new_pattern files

* at - Schedule one time tasks.

Syn: Echo "Command" | at time - Execute Commands at specified time

Syn: echo "backup.sh" | at 2:00 AM

* Uniq : filter out repeated lines

Removes duplicates lines

Syn : Sort file | uniq

* nl - Numbers lines in a file

Numbers each line

Syn : nl file

* free - Show memory usage

free -h - Memory stats in human-readable format

* list out the contents in Home dir from the VIM

:!ls

* to use shell command in VIM

:! <shell commands>

* :set hlsearch

highlights previous highlighted content.

* q:

* :tabnew file names

opens the file in another tab.

* :tabc

exits from the current tab

* :g/^$/d deletes the empty lines

g

* undo - g-
* redo - g+

* :earlier <time>