

Systems Programming

Linux/Unix Commands

Linux/Unix Pipes , Grep & Sort Command

- The symbol '|' denotes a pipe
 - Used to run two or more commands consecutively at the same time
 - E.g. to show you only one scroll length of the content of a file at a time
 - ***cat filename | less***
 - ***cat Filename | pg***
- or
- ***cat Filename | more***

while using 'cat' command

```
home@VirtualBox:~$ cat sample
```

The screen zooms to the end of the file

```
First
Eat
Hide
home@VirtualBox:~$
```

But with piping and using 'less' command

```
home@VirtualBox:~$ cat sample | less
```

You can scroll file content using the arrow keys or PageUp/PageDown as you read

```
Bat
Boat
Apple
Dog
First
:
```

Once you reach end of file,

Press q to exit

```
Apple
Dog
First
Eat
Hide
(END)
```

The 'grep' command

- scans the document for the desired information and present the result in a format you want
- Options:
 - **-v** Shows all the lines that do not match the searched string
 - **-c** Displays only the count of matching lines
 - **-n** Shows the matching line and its number
 - **-i** Match both (upper and lower) case
 - **-l** Shows just the name of the file with the string

```
home@VirtualBox:~$ cat sample | grep -i a
Bat
Goat
Apple
Eat
```

The contents of the 'sample' file

```
home@VirtualBox:~$ cat sample
Bat
Goat
Apple
Dog
First
Eat
Hide
```

Using 'grep' for searching Apple

```
home@VirtualBox:~$ cat sample | grep Apple
Apple
```

Using 'grep' for searching Eat

```
home@VirtualBox:~$ cat sample | grep Eat
Eat
```

The 'sort' command

- *sort Filename*
- sorts out the contents of a file alphabetically
- Options:
- **-r** Reverses sorting
- **-n** Sorts numerically
- **-f** Case insensitive sorting

View the contents of file 'abc' :

```
guru99@VirtualBox:~$ cat abc  
a  
b  
c  
d  
e
```

Sort the contents of file 'abc' :

```
guru99@VirtualBox:~$ sort abc  
a  
b  
c  
d  
e
```

shows reverse sorting of the contents in file 'abc' :

```
guru99@VirtualBox:~$ sort -r abc  
e  
d  
c  
b  
a
```

Filter

- Filter is the output from first command which becomes the input for the next command

e.g.

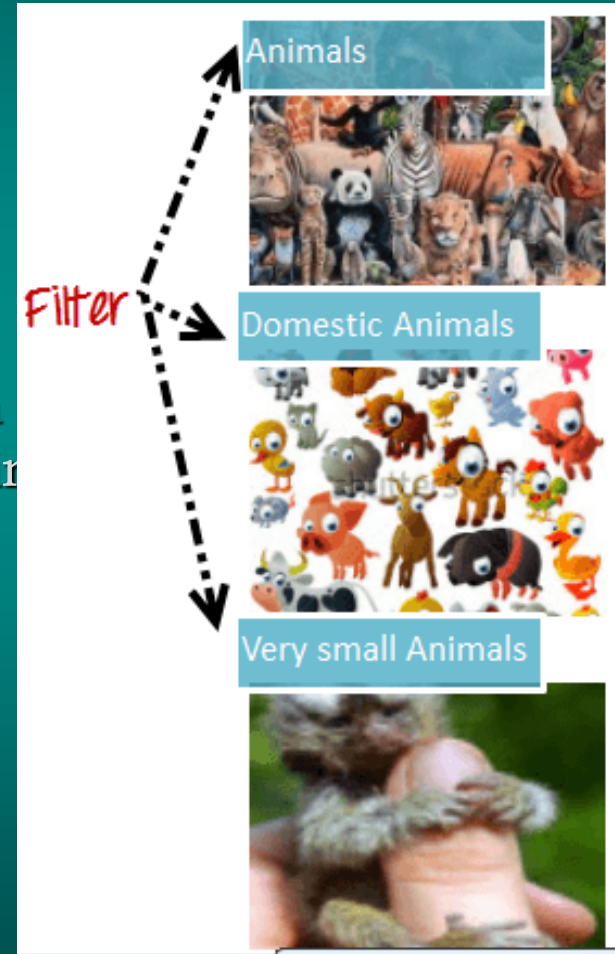
cat sample | grep -v a | sort -r

- highlight only the lines that do not contain the character 'a', but the result should be in reverse order

```
home@VirtualBox:~$ cat sample
Bat
Goat
Apple
Dog
First
Eat
Hide
```

Filtered Results given to the next command

```
home@VirtualBox:~$ cat sample | grep -v a | sort -r
Hide
First
Dog
Apple
```



Regular Expressions

- special characters which help search data, matching complex patterns. Regular expressions are shortened as 'regexp' or 'regex'
- Types:
 - Basic regex
 - Interval regex
 - Extended regex

Basic Regular Expression

- Some of the commonly used commands with Regular expressions are tr, sed, vi and grep . Listed below are some of the basic Regex.

| Symbols | Descriptions |
|---------|---|
| . | replaces any character |
| ^ | matches start of string |
| \$ | matches end of string |
| * | matches up zero or more times the preceding character |
| \ | Represent special characters |
| () | Groups regular expressions |
| ? | Matches up exactly one character |

Basic Regex: Example

Execute cat sample to see contents of an existing file

```
guru99@guru99-VirtualBox:~$ cat sample
apple
bat
ball
ant
eat
pant
people
taste
guru99@guru99-VirtualBox:~$
```

Select only those lines that end with t using \$

```
guru99@guru99-VirtualBox:~$ cat sample | grep t$
bat
ant
eat
pant
guru99@guru99-VirtualBox:~$
```

Search for content containing letter 'a'.

```
guru99@guru99-VirtualBox:~$ cat sample | grep a
apple
bat
ball
ant
eat
pant
taste
guru99@guru99-VirtualBox:~$
```

'^' matches the start of a string.

```
guru99@guru99-VirtualBox:~$ cat sample | grep ^a
apple
ant
guru99@guru99-VirtualBox:~$
```


Interval Regular expressions

- They tell us about the number of occurrences of a character in a string.

| Expression | Description |
|--------------|---|
| {n} | Matches the preceding character appearing 'n' times exactly |
| {n,m} | Matches the preceding character appearing 'n' times but not more than m |
| {n, } | Matches the preceding character only when it appears 'n' times or more |

To check that the character 'p' appears exactly 2 times in a string one after the other.:

```
cat sample | grep -E p\{2}
```

```
guru99@guru99-VirtualBox:~$ cat sample|grep -E p\{2}
apple
guru99@guru99-VirtualBox:~$
```

Extended regular expressions

- contain combinations of more than one expression

| Expression | Description |
|------------|--|
| \+ | Matches one or more occurrence of the previous character |
| \? | Matches zero or one occurrence of the previous character |

Searching for all characters 't'

```
guru99@guru99-VirtualBox:~$ cat sample|grep t
bat
ant
eat
pant
taste
```

to filter out lines where character 'a' preceeds character 't'
We can use command like
cat sample|grep "a\+t"

```
guru99@guru99-VirtualBox:~$ cat sample|grep "a\+t"
bat
eat
guru99@guru99-VirtualBox:~$
```

Brace expansion

- The syntax for brace expansion is either a sequence or a comma separated list of items inside curly braces "{}". The starting and ending items in a sequence are separated by two periods "..".
- The echo command creates strings using the brace expansion

```
guru99@VirtualBox:~$ echo {aa,bb,cc,dd}
aa bb cc dd
guru99@VirtualBox:~$ echo {0..11}
0 1 2 3 4 5 6 7 8 9 10 11
guru99@VirtualBox:~$ echo {a..z}
a b c d e f g h i j k l m n o p q r s t u v w x y z
guru99@VirtualBox:~$ echo a{0..9}b
a0b a1b a2b a3b a4b a5b a6b a7b a8b a9b
guru99@VirtualBox:~$
```

Environment Variables

- Dynamic values which affect the processes or programs on a computer
- Exist in every operating system; types may vary
- Can be created, edited, saved and deleted
- Gives information about the system behavior
- Example:
 - \$LANG environment variable stores the value of the language that the user understands. This value is read by an application such that a Chinese user is shown a Mandarin interface while an American user is shown an English interface

Environment Variables

Environment variables govern behavior of programs in your Operating System

| Variable | Description |
|---------------|---|
| PATH | <p>This variable contains a colon (:)-separated list of directories in which your system looks for executable files.</p> <pre>guru99@VirtualBox:~\$ echo \$PATH /usr/lib/lightdm/lightdm:/usr/local/sbin:/usr/local/bin:/usr :/bin:/usr/games</pre> <p>When you enter a command on terminal, the shell looks for the command in different directories mentioned in the \$PATH variable. If the command is found , it executes. Otherwise, it returns with an error 'command not found'.</p> |
| USER | The username |
| HOME | Default path to the user's home directory |
| EDITOR | Path to the program which edits the content of files |
| UID | User's unique ID |
| TERM | Default terminal emulator |
| SHELL | Shell being used by the user |

Creating New Variables

- **variables are case-sensitive** and usually they are created in upper case.
- E.g.
 - **VARIABLE_NAME= variable_value**

Define a new variable

```
ubuntu@ubuntu:~$ NEWVARIABLE=value1234
```

Check value of the newly created variable

```
ubuntu@ubuntu:~$ echo $NEWVARIABLE  
value1234
```

TIP: Do not leave space between variable name and "=" sign
you will get an error

```
ubuntu@ubuntu:~$ NEWVARIABLE = value1234  
NEWVARIABLE: command not found
```

TIP: Do not forget the "\$" sign when you check the value of variable

```
ubuntu@ubuntu:~$ echo NEWVARIABLE  
NEWVARIABLE
```

Deleting variables

- Unset VARIABLENAME
 - This **would remove the Variable** and its value permanently

Define a new variable

```
guru99@VirtualBox:~$ VARIABLE1=1234
```

Check value of the newly created variable

```
guru99@VirtualBox:~$ echo $VARIABLE1  
1234
```

Use the 'unset' command

```
guru99@VirtualBox:~$ unset VARIABLE1
```

The variable value is removed

```
guru99@VirtualBox:~$ echo $VARIABLE1
```

```
guru99@VirtualBox:~$
```

Environment Variable

| Command | Description |
|-------------------------------------|---|
| echo \$VARIABLE | To display value of a variable |
| env | Displays all environment variables |
| VARIABLE_NAME=variable_value | Create a new variable |
| unset | Remove a variable |
| export Variable=value | To set value of an environment variable |

Communication in Linux/Unix

- While working on a Linux operating system you may need to **communicate with other devices**. For this, there are some basic utilities that you can make use of.
- These utilities can help you communicate with:
 - networks,
 - other Linux systems
 - and remote users
 - So, let us learn them one by one

Ping

- This utility is commonly used to check whether your connection to the server is healthy or not. This command is also used in -

ping 172.16.170.1

```
home@VirtualBox:~$ ping 172.16.170.1
PING 172.16.170.1 (172.16.170.1) 56(84) bytes of data.
64 bytes from 172.16.170.1: icmp_req=1 ttl=64 time=0.423 ms
64 bytes from 172.16.170.1: icmp_req=2 ttl=64 time=0.434 ms
64 bytes from 172.16.170.1: icmp_req=3 ttl=64 time=0.432 ms
64 bytes from 172.16.170.1: icmp_req=4 ttl=64 time=0.405 ms
^C
--- 172.16.170.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms
rtt min/avg/max/mdev = 0.405/0.423/0.434/0.023 ms
```

ping www.google.com

```
guru99@VirtualBox:~$ ping www.google.com
PING www.google.com (173.194.36.19) 56(84) bytes of data.
64 bytes from bom04s01-in-f19.1e100.net (173.194.36.19): icmp_r
32.2 ms
64 bytes from bom04s01-in-f19.1e100.net (173.194.36.19): icmp_r
45.0 ms
^C
--- www.google.com ping statistics ---
3 packets transmitted, 2 received, 33% packet loss, time 2009ms
rtt min/avg/max/mdev = 32.257/38.655/45.054/6.401 ms
guru99@VirtualBox:~$
```

- Analyzing network and host connections
- Tracking network performance and managing it
- Testing hardware and software issues
- The system has sent 64 bytes data packets to the IP Address (172.16.170.1) or the Hostname(www.google.com). If even one of data packets does not return or is lost, it would suggest an error in the connection. Usually, internet connectivity is checked using this method
- You may Press Ctrl + c to exit from the ping loop.

FTP

- FTP is file transfer protocol. It's the most preferred protocol for data transfer amongst computers.
- You can use FTP to -
 - Logging in and establishing a connection with a remote host
 - Upload and download files
 - Navigating through directories
 - Browsing contents of the directories

FTP syntax

- ftp
- Once you enter this command, it will ask you for authentication via username and password

Login using ftp ip/hostname 

```
guru99@VirtualBox:~$ ftp ftp.javatutorialhub.com
Connected to ftp.javatutorialhub.com.
220----- Welcome to Pure-FTPd [privsep] [TLS] -----
220-You are user number 1 of 50 allowed.
220-Local time is now 06:19. Server port: 21.
220-This is a private system - No anonymous login
220-IPv6 connections are also welcome on this server.
220 You will be disconnected after 15 minutes of inactivity.
Name (ftp.javatutorialhub.com:guru99):
```

Enter Username & Password

```
Name (ftp.javatutorialhub.com:guru99): linux@javatutorialhub.co
331 User linux@javatutorialhub.com OK. Password required
Password:
```

FTP account is logged in and ready for use

```
Password:
230 OK. Current restricted directory is /
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

FTP actions

Once connection is established and you are logged in , you may use the following commands to perform different actions

| Command | Function |
|---------------------|---|
| dir | Display files in the current directory of remote computer |
| cd "dirname" | change directory to "dirname" on remote computer |
| put file | upload 'file' from local to remote computer |
| get file | Download 'file' from remote to local computer |
| quit | Logout |

FTP actions

Displaying files in the directory using the 'dir' command

```
ftp> dir
200 PORT command successful
150 Connecting to port 38487
drwxr-xr-x  2 java      java           4096 Sep 19 06:23 .
drwxr-xr-x  2 java      java           4096 Sep 19 06:23 ..
-rw-r--r--  1 java      java              4 Sep 18 04:18 .ftpquota
-rw-r--r--  1 java      java          67330 Sep 19 05:44 functions.php
-rw-r--r--  1 java      java           18 Sep 19 06:23 sample.txt
-rw-r--r--  1 java      java          2770 Sep 19 05:45 single-Portfolio.
hp
-rw-r--r--  1 java      java          3266 Sep 19 05:45 single.php
-rw-r--r--  1 java      java          2732 Sep 19 05:44 sitemap.php
-rw-r--r--  1 java      java           967 Sep 19 05:44 style.css
-rw-r--r--  1 java      java       438861 Sep 19 05:44 udesign_options_p
ge.php
226-Options: -a -l
226 10 matches total
ftp>
```

uploading a file using 'put file' command

```
ftp> put sample.txt
local: sample.txt remote: sample.txt
200 PORT command successful
150 Connecting to port 57968
226-File successfully transferred
226 0.262 seconds (measured here), 68.67 bytes per second
18 bytes sent in 0.00 secs (195.3 kB/s)
ftp>
```

downloading a file using 'get filename' command

```
ftp> get sitemap.php
local: sitemap.php remote: sitemap.php
200 PORT command successful
150 Connecting to port 44051
226-File successfully transferred
226 0.000 seconds (measured here), 37.81 Mbytes per second
2732 bytes received in 0.00 secs (1095.2 kB/s)
ftp>
```

logging out from FTP

```
ftp> quit
221-Goodbye. You uploaded 1 and downloaded 3 kbytes.
221 Logout.
n10@N100:~$
```

Telnet

- Telnet helps to -
 - connect to a remote Linux computer
 - run programs remotely and conduct administration
 - This utility is similar to the Remote Desktop feature found in Windows Machine.
- Telnet syntax:
 - telnet
- Example:
 - telnet localhost
- Once authenticated, you can execute commands just like you have doing so far, using the Terminal. The only difference is, if you are connected to a remote host, the commands will be executed on the remote machine ,and not your local machine.
- You may exit the telnet connection by entering the command 'logout'

SSH (Secure shell)

- used to securely connect to a remote computer.
- Compare to Telnet , SSH is secure wherein the client /server connection is authenticated using a digital certificate and passwords are encrypted.
- it's widely used by system administrators to control remote Linux servers.
- syntax :
 - SSH username@ip-address or hostname
- Once your are logged in , you can execute any commands that you do in your terminal

```
guru99@VirtualBox:~$ ssh guru99@68.233.250.32
guru99@68.233.250.32's password:
Last login: Mon Feb 11 04:51:24 2013 from 59.177.37.89
[guru99@cp ~]$
```

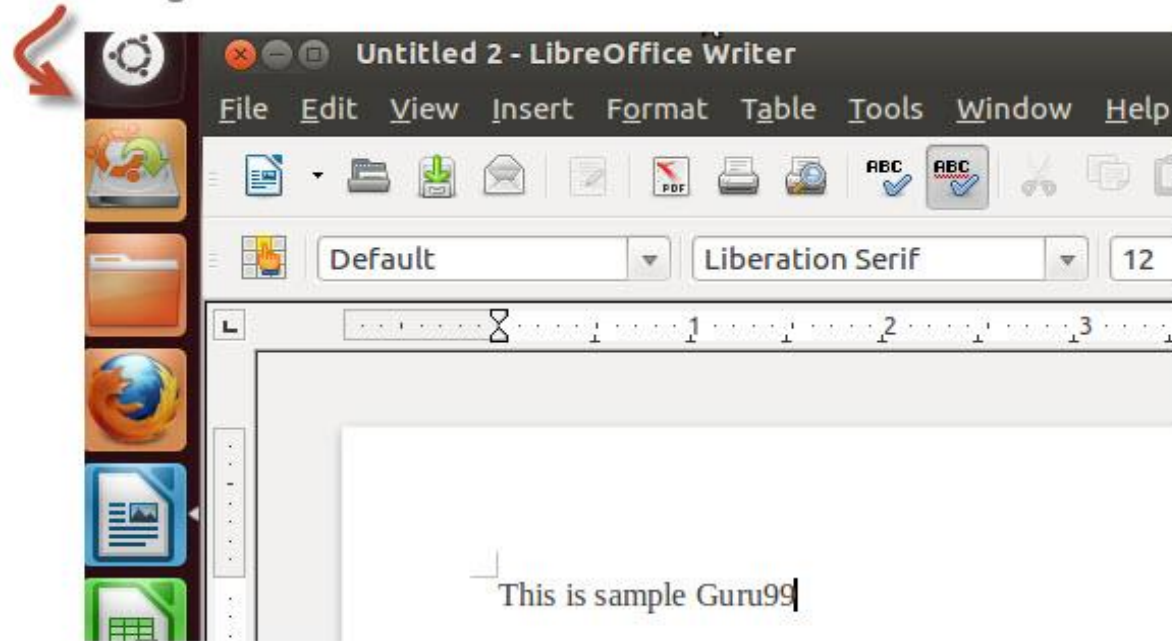
```
[guru99@cp ~]$ ls
access-logs      etc              mail            public_ftp      tmp
cpbackup-exclude.conf  local_file      perl5          public_html     www
[guru99@cp ~]$
```

```
[guru99@cp ~]$ pwd
/home/guru99
[guru99@cp ~]$
```


Managing Processes in Linux/Unix

- any command that you give to your Linux machine starts a new process.
- It's possible to have multiple processes for the same program

When you launch Office to write some article



Corresponding process is created



Process Types

- Foreground Processes: They run on the screen and need input from the user. For example : Office Programs
- Background Processes: They run in the background and usually do not need user input. For example Antivirus.

Running a Process

- Foreground process:
 - run it from the dash board or;
 - run it from the terminal.
- When using the Terminal, you will have to wait, until the foreground process runs.
- Background process
 - If you start a foreground program/process from the terminal, then you cannot work on the terminal, till the program is up and running.
 - Certain, data intensive tasks take lots of processing power and may even take hours to complete. You do not want your terminal to be held up for such a long time.
 - To avoid such a situation, you can run the program and send it to the background so that terminal remains available to you. Let's learn how to do this -

Managing processes: “fg” command

- You can use the command "fg" to continue a program which was stopped and bring it to the foreground.
- The simple syntax for this utility is:
 - **fg**
- Example
 - Launch 'banshee' music player
 - Stop it with the 'ctrl +z' command
 - Continue it with the 'fg' utility.

Start the program and press ctrl+z

```
guru99@VirtualBox:~$ banshee
[Info 16:08:36.688] Running Banshee 2.2.1: [Ubuntu 11.
11-12-19 14:51:26 UTC]
^Z
[1]+  Stopped                  banshee
```

Type 'bg' to send the process to the background

```
guru99@VirtualBox:~$ bg
```

```
home@VirtualBox:~$ banshee
^Z
[1]+  Stopped                  banshee
home@VirtualBox:~$ fg banshee
banshee
[Info 00:36:19.400] Running Banshee 2.2.0: [Ubuntu oneiric
(linux-gnu, i686) @ 2011-09-23 04:51:00 UTC]
```

Managing processes: “Top” command

- This utility tells the user about all the running processes on the Linux machine
- Press 'q' on the keyboard to move out of the process display.

```
home@VirtualBox:~$ top
top - 23:57:43 up 2:54, 1 user, load average: 0.00, 0.01, 0.05
Tasks: 189 total, 2 running, 187 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.7%us, 3.0%sy, 0.0%ni, 96.3%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Mem: 1026080k total, 924508k used, 101572k free, 37000k buffers
Swap: 1046524k total, 21472k used, 1025052k free, 367996k cached
```

| PID | USER | PR | NI | VIRT | RES | SHR | S | %CPU | %MEM | TIME+ | COMMAND |
|------|------|----|----|-------|------|------|---|------|------|---------|-----------------|
| 1525 | home | 20 | 0 | 1775m | 100m | 28m | S | 1.7 | 10.0 | 5:05.34 | Photoshop.exe |
| 961 | root | 20 | 0 | 75972 | 51m | 7952 | R | 1.0 | 5.1 | 2:23.42 | Xorg |
| 1507 | home | 20 | 0 | 7644 | 4652 | 696 | S | 1.0 | 0.5 | 2:42.66 | wineserver |
| 1564 | home | 20 | 0 | 75144 | 29m | 9840 | S | 0.3 | 3.0 | 0:25.96 | ubuntuone-syncd |
| 2999 | home | 20 | 0 | 127m | 13m | 10m | S | 0.3 | 1.4 | 0:01.36 | gnome-terminal |
| 3077 | home | 20 | 0 | 2820 | 1188 | 864 | R | 0.3 | 0.1 | 0:00.76 | top |
| 1 | root | 20 | 0 | 3200 | 1704 | 1260 | S | 0.0 | 0.2 | 0:00.98 | init |
| 2 | root | 20 | 0 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.00 | kthreadd |
| 3 | root | 20 | 0 | 0 | 0 | 0 | S | 0.0 | 0.0 | 0:00.95 | ksoftirqd/0 |

Running processes: Terminology

| Field | Description | Example 1 | Example 2 |
|----------------|--|-----------------------|-----------|
| PID | The process ID of each task | 1525 | 961 |
| User | The username of task owner | Home | Root |
| PR | Priority Can be 20(highest) or -20(lowest) | 20 | 20 |
| NI | The nice value of a task | 0 | 0 |
| VIRT | Virtual memory used (kb) | 1775 | 75972 |
| RES | Physical memory used (kb) | 100 | 51 |
| SHR | Shared memory used (kb) | 28 | 7952 |
| S | Status There are five types: 'D' = uninterruptible sleep 'R' = running 'S' = sleeping 'T' = traced or stopped 'Z' = zombie | S | R |
| %CPU | % of CPU time | 1.7 | 1.0 |
| %MEM | Physical memory used | 10 | 5.1 |
| TIME+ | Total CPU time | 5:05.34 | 2:23.42 |
| Command | Command name | <u>Photoshop</u> .exe | Xorg |

“PS” Command

- This command stands for 'Process Status'.
- similar to the Windows "Task Manager"
- similar to 'top' command but the information displayed is different.
- To check all the processes running under a user, use the command -

```
• ps ux home@VirtualBox:~$ ps ux
```

| USER | PID | %CPU | %MEM | VSZ | RSS | TTY | STAT | START | TIME | COMMAND |
|------|------|------|------|--------|-------|-----|------|-------|------|------------|
| home | 1114 | 0.0 | 0.8 | 46548 | 8512 | ? | Ssl | Sep03 | 0:00 | gnome-sess |
| home | 1151 | 0.0 | 0.0 | 3856 | 140 | ? | Ss | Sep03 | 0:00 | /usr/bin/s |
| home | 1154 | 0.0 | 0.0 | 3748 | 484 | ? | S | Sep03 | 0:00 | /usr/bin/d |
| home | 1155 | 0.1 | 0.2 | 6656 | 3036 | ? | Ss | Sep03 | 0:18 | //bin/dbus |
| home | 1157 | 0.0 | 0.2 | 9148 | 2368 | ? | S | Sep03 | 0:00 | /usr/lib/g |
| home | 1162 | 0.0 | 0.2 | 31588 | 2296 | ? | Ssl | Sep03 | 0:00 | /usr/lib/g |
| home | 1174 | 0.0 | 1.4 | 132472 | 14884 | ? | Sl | Sep03 | 0:03 | /usr/lib/g |

You can also check the process status of a single process , use the syntax -

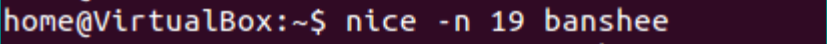
ps PID

“kill” Command

- This command terminates a running processes on a Linux machine.
- In order to use this utility you need to know the PID (process id) of the process you want to kill
- Syntax -
 - kill PID
- To find the PID of a process simply type
- pidof Processname

```
home@VirtualBox:~$ pidof Photoshop.exe  
1525  
home@VirtualBox:~$ kill 1525
```


NICE

- Linux can run a lot of processes at a time, which can slow down the speed of some high priority processes and result in poor performance.
- To avoid this, you can tell your machine to prioritize processes as per your requirements
- This priority is called Niceness in Linux and it has a value between -20 to 19. The lower the Niceness index the higher would be priority given to that task.
- The default value of all the processes is 0.
- To start a process with a niceness value other than the default value use the following syntax
- `nice -n 'Nice value' process name` 
- If there is some process already running on the system then you can 'Renice' its value using syntax.
- `renice 'nice value' -p 'PID'`
- To change Niceness you can use the 'top' command to determine the PID (process id) and its Nice value. Later use the renice command to change the value.

“Niceness” – Illustration

Checking the niceness value of the process 'banshee'

| PID | USER | PR | NI | VIRT | RES | SHR | S | %CPU | %MEM | TIME+ | COMMAND |
|------|------|----|----|------|-----|-----|---|------|------|---------|---------|
| 3293 | home | 20 | 0 | 277m | 64m | 35m | S | 96.4 | 6.4 | 9:56.72 | banshee |

Renicing the value to -20

```
home@VirtualBox:~$ sudo renice -20 -p 3293
[sudo] password for home:
3293 (process ID) old priority 0, new priority -20
```

The value changed to -20

| | | | | | | | | | | | |
|------|------|---|-----|------|-----|-----|---|------|-----|---------|---------|
| 3293 | home | 0 | -20 | 277m | 64m | 35m | S | 95.2 | 6.4 | 3:32.95 | banshee |
|------|------|---|-----|------|-----|-----|---|------|-----|---------|---------|

df

- reports the free disk space(Hard Disk) on all the file systems

```
guru99@guru99-VirtualBox:~$ df
Filesystem      1K-blocks    Used Available Use% Mounted on
/dev/sda1        7837756 2921376   4523216  40% /
udev             246488      4    246484    1% /dev
tmpfs            101512     752    100760    1% /run
none              5120        0      5120    0% /run/lock
none             253776     76    253700    1% /run/shm
```

- If you want the above information in a readable format , then use the command
 - 'df -h'**

```
guru99@guru99-VirtualBox:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1        7.5G  2.8G  4.4G  40% /
udev             241M  4.0K  241M   1% /dev
tmpfs            100M  752K   99M   1% /run
none             5.0M    0   5.0M   0% /run/lock
none             248M  76K   248M   1% /run/shm
```

free

- shows the free and used memory (RAM) on the Linux system

```
home@VirtualBox:~$ free
```

| | total | used | free | shared | buffers | cached |
|--------------------|---------|--------|---------|--------|---------|--------|
| Mem: | 1026080 | 803604 | 222476 | 0 | 36312 | 343376 |
| -/+ buffers/cache: | | 423916 | 602164 | | | |
| Swap: | 1046524 | 35832 | 1010692 | | | |

- You can use the arguments
 - free -m to display output in MB
 - free -g to display output in GB

The VI Editor

- the vi editor is the most popular and classic text editor in the Linux family
 - It is available in almost all Linux Distributions
 - It works the same across different platforms and Distributions
 - It is user friendly. Hence, millions of Linux users love it and use it for their editing needs
- Advanced versions of the vi editor
 - VIM (Vi Improved), Elvis, Nvi , Nano and Vile.
- vi is feature-rich and offers endless possibilities to edit a file.
- vi operation modes:
 - Command mode
 - Insert mode

vi operation modes

- Command mode:
 - The vi editor opens in this mode and only understands commands
 - In this mode , you can , move the cursor and cut, copy, paste the text
 - This mode, also saves the changes you have made to the file
 - Commands are case sensitive .You should use the right letter case
- Insert mode:
 - This mode is for inserting text in the file
 - You can switch to the Insert mode from the command mode by pressing 'i' on the keyboard
 - Once you are in Insert mode, any key would be taken as an input for the file on which you are currently working.
 - To return to the command mode and save the changes you have made you need to press the Esc key

Starting the vi editor

- Open the Terminal (CLI) and type
- ***vi <filename_NEW> or <filename_EXISTING>***
- If you specify an existing file then the editor would open it for you to edit. Else, you can create a new file

→ creating a new file

```
guru99@VirtualBox:~$ vi samplefile
```

- vi editor opens in the command mode

Command Mode

'~' shows unused lines

| | | |
|-------------------------|-------|-----|
| "samplefile" [New File] | 0,0-1 | All |
|-------------------------|-------|-----|

press 'i' to enter the insert mode

Insert Mode

```
-- INSERT --
```

0,1

All

Starting the vi editor

- Open the Terminal (CLI) and type
- ***vi <filename_NEW> or <filename_EXISTING>***
- If you specify an existing file then the editor would open it for you to edit. Else, you can create a new file

— Add content

Hello World!

~~~~~

1,12

ALL

Press `esc` to enter command mode. Press `:wq` to save and quit

Hello World!

100

:WQ

- check the content of file

```
guru99@VirtualBox:~$ cat samplefile
```

# Hello World!

guru99@VirtualBox:~\$



# vi Editing commands

**Note:** You should be in the "**command mode**" to execute these commands. VI editor is **case-sensitive** so make sure you type the commands in the right letter-case.

| Keystrokes | Action                                                                                        |
|------------|-----------------------------------------------------------------------------------------------|
| i          | Insert at cursor (goes into insert mode)                                                      |
| a          | Write after cursor (goes into insert mode)                                                    |
| A          | Write at the end of line (goes into insert mode)                                              |
| ESC        | Terminate insert mode                                                                         |
| u          | Undo last change                                                                              |
| U          | Undo all changes to the entire line                                                           |
| o          | Open a new line (goes into insert mode)                                                       |
| dd<br>3dd  | Delete line<br>Delete 3 lines.                                                                |
| D          | Delete contents of line after the cursor                                                      |
| C          | Delete contents of line after the cursor and insert new text. Press esc key to end insertion. |
| dw<br>4dw  | Delete word<br>Delete 4 words                                                                 |
| cw         | Change word                                                                                   |
| x          | Delete character at cursor                                                                    |
| r          | Replace character                                                                             |
| R          | Overwrite characters from cursor onward                                                       |
| s          | Substitute one character under cursor continue to insert                                      |
| S          | Substitute entire line and begin to insert at beginning of the line                           |
| ~          | Change case of individual character                                                           |

# Moving within a file

- You need to be in the command mode in order to move within a file. The default keys for navigation are mentioned below else, you can **also use the arrow keys on the keyboard.**

| Keystroke | Use               |
|-----------|-------------------|
| <b>k</b>  | Move cursor up    |
| <b>j</b>  | Move cursor down  |
| <b>h</b>  | Move cursor left  |
| <b>l</b>  | Move cursor right |

# Saving and Closing the file

- You should be in the **command mode** to **exit the editor and save changes** to the file

| Keystroke       | Use                            |
|-----------------|--------------------------------|
| <b>Shift+zz</b> | Save the file and quit         |
| <b>:w</b>       | Save the file but keep it open |
| <b>:q</b>       | Quit without saving            |
| <b>:wq</b>      | Save the file and quit         |

