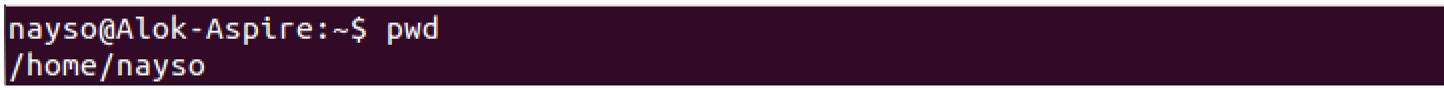
Linux Commands

Basic Commands

1. pwd

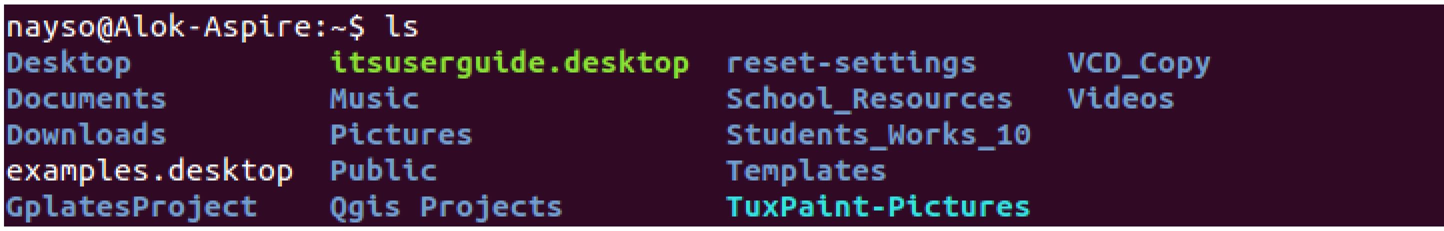
When you first open the terminal, you are in the home directory of your user. To know which directory you are in, you can use the "**pwd**" command. It gives us the Absolute Path, which means the path that starts from the root. The root is the base of the Linux filesystem. It is denoted by a forward slash(/). The user directory is usually something like /home/username.



```
nayso@Alok-Aspire:~$ pwd
/home/nayso
```

2. ls

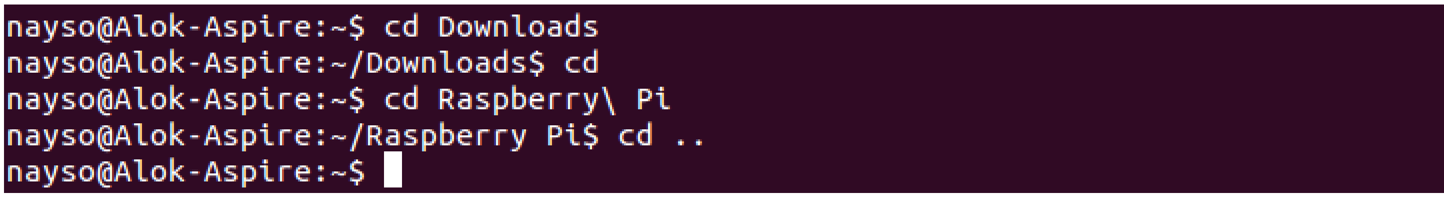
The **"Is"** command is used to know what files are there in the directory you are in. You can see all the hidden files by using the command "**ls -a**".



```
nayso@Alok-Aspire:~$ ls
Desktop            itsuserguide.desktop  reset-settings     VCD_Copy
Documents          Music                 School_Resources   Videos
Downloads          Pictures              Students_Works_10
examples.desktop   Public                Templates
GplatesProject     Qgis Projects         TuxPaint-Pictures
```

3. cd

"cd" is the command used to go to a directory. For example, if you are in the home folder, and you want to go to the Downloads folder, then you can type in "**cd Downloads**". Remember, this command is case sensitive and you have to type in the name of the folder exactly as it is. But there is a problem with these commands. Imagine you have a folder named "Raspberry Pi". In this case, when you type in **"cd Raspberry Pi"**, the shell will take the second argument of the command as a different one, so you will get an error saying that the directory does not exist. Here, you can use a backward slash. That is, you can use **"cd Raspberry\ Pi"** in this case. Spaces are denoted like this: If you just type **"cd"** and press Enter, it takes you to the home directory. To go back from a folder to the folder before that, you can type **"cd .."** . The two dots represent back.

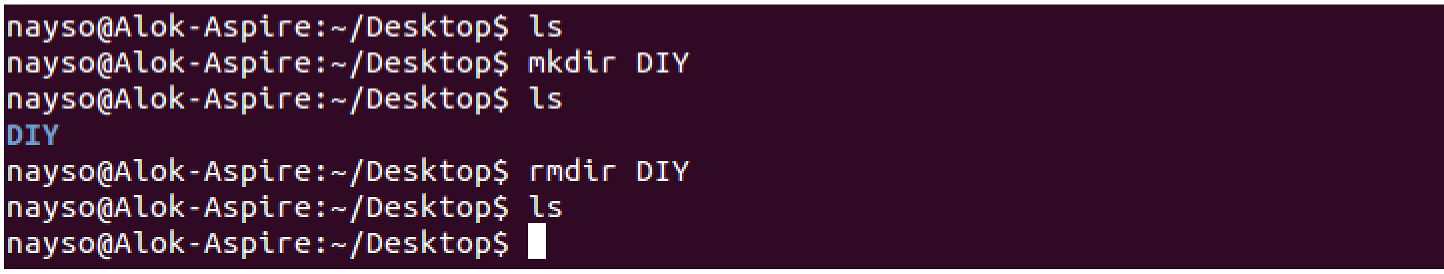


```
nayso@Alok-Aspire:~$ cd Downloads
nayso@Alok-Aspire:~/Downloads$ cd
nayso@Alok-Aspire:~$ cd Raspberry\ Pi
nayso@Alok-Aspire:~/Raspberry Pi$ cd ..
nayso@Alok-Aspire:~$ 
```

4. mkdir & rmdir

The **mkdir** command is used when you need to create a folder or a directory. For Example, if you want to make a directory called "DIY", then you can type "**mkdir DIY**". Remember, as told before, if you want to create a directory named "DIY Hacking", then you can type **"**mkdir **DIY\ Hacking"**.

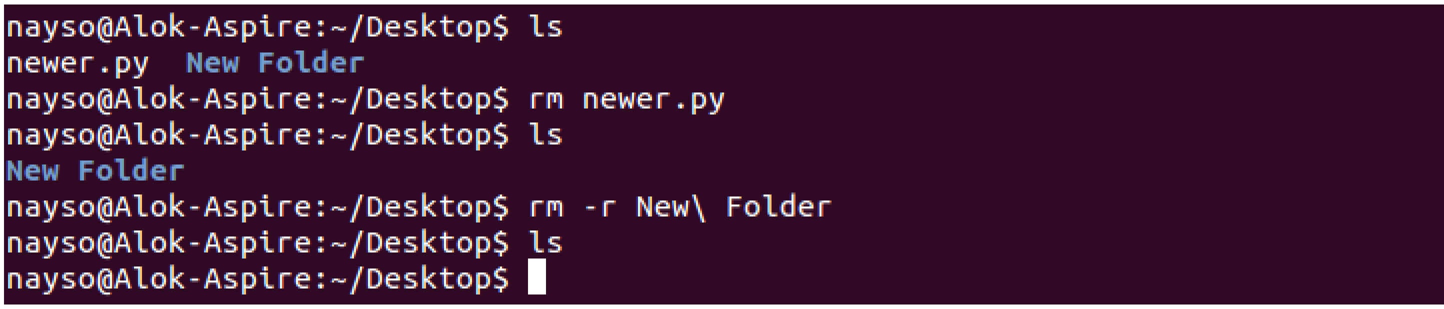
rmdir is the command used for deleting a directory. But, **rmdir** can only be used to delete an empty directory. To delete a directory containing files, **rm** is used.



```
nayso@Alok-Aspire:~/Desktop$ ls
nayso@Alok-Aspire:~/Desktop$ mkdir DIY
nayso@Alok-Aspire:~/Desktop$ ls
DIY
nayso@Alok-Aspire:~/Desktop$ rmdir DIY
nayso@Alok-Aspire:~/Desktop$ ls
nayso@Alok-Aspire:~/Desktop$ 
```

5. rm

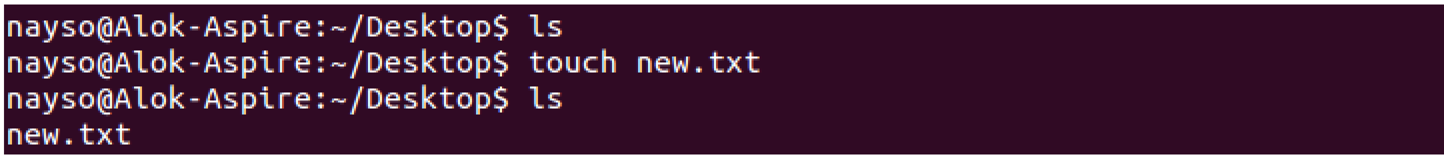
The **rm** command is used to delete files and directories. **rm** cannot simply delete a directory. "**rm -r**" is used to delete a directory. In this case, it deletes both the folder and the files in it.



```
nayso@Alok-Aspire:~/Desktop$ ls
newer.py  New Folder
nayso@Alok-Aspire:~/Desktop$ rm newer.py
nayso@Alok-Aspire:~/Desktop$ ls
New Folder
nayso@Alok-Aspire:~/Desktop$ rm -r New\ Folder
nayso@Alok-Aspire:~/Desktop$ ls
nayso@Alok-Aspire:~/Desktop$ 
```

6. touch

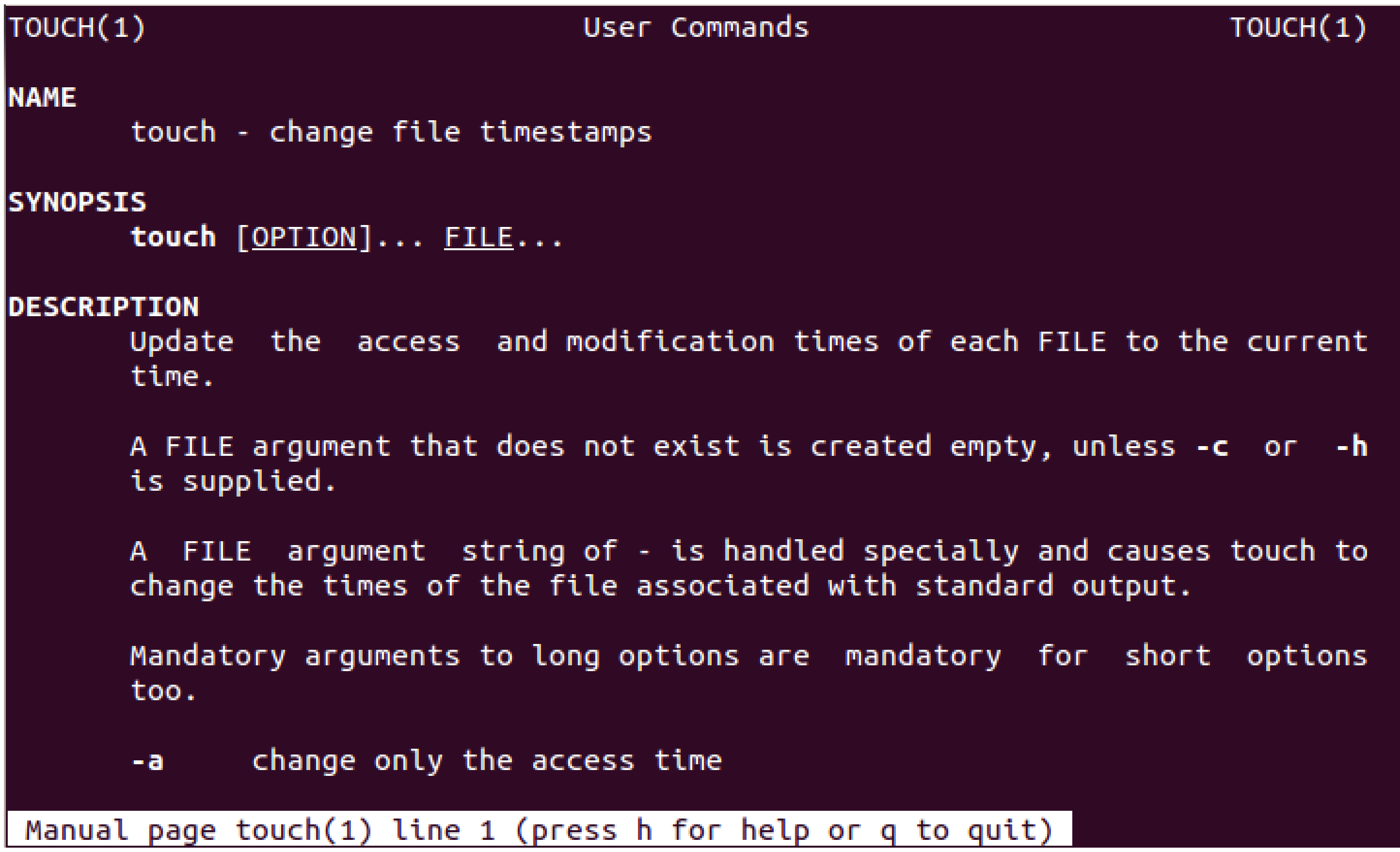
The **touch** command is used to create a file. It can be anything, from an empty txt file to an empty zip file. For example – "**touch new.txt**"



```
nayso@Alok-Aspire:~/Desktop$ ls
nayso@Alok-Aspire:~/Desktop$ touch new.txt
nayso@Alok-Aspire:~/Desktop$ ls
new.txt
```

7. man & –help

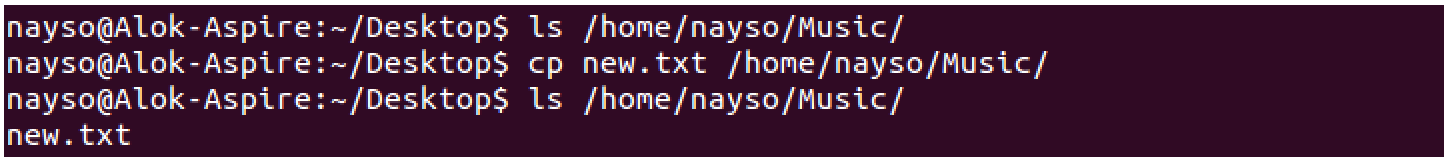
To know more about a command and on how to use it, the **man** command is used. It shows the manual pages of the command. For Example, "**man cd**" shows the manual pages of the **cd** command. Typing in the command name and the argument helps it show which ways the command can be used (Example – **cd –help**).



```
TOUCH(1)                         User Commands                        TOUCH(1)

NAME
       touch - change file timestamps

SYNOPSIS
       touch [OPTION]... FILE...

DESCRIPTION
       Update  the  access  and modification times of each FILE to the current
       time.

       A FILE argument that does not exist is created empty, unless -c  or  -h
       is supplied.

       A  FILE  argument  string of - is handled specially and causes touch to
       change the times of the file associated with standard output.

       Mandatory arguments to long options are  mandatory  for  short  options
       too.

       -a     change only the access time

 Manual page touch(1) line 1 (press h for help or q to quit)
```

8. cp

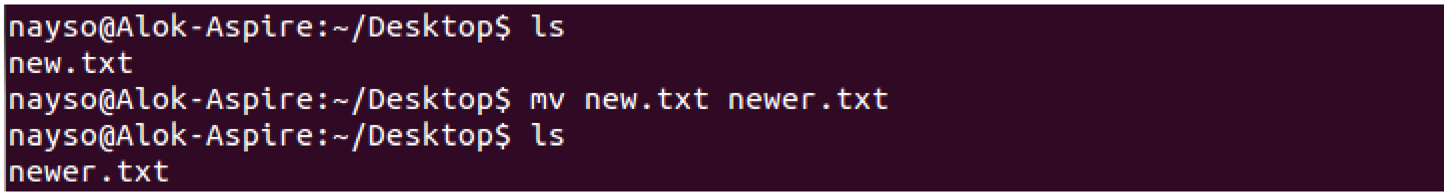
The **cp** command is used to copy files through the command line. It takes two arguments, the first one is location of the file to be copied, the second iswhere to copy.



```
nayso@Alok-Aspire:~/Desktop$ ls /home/nayso/Music/
nayso@Alok-Aspire:~/Desktop$ cp new.txt /home/nayso/Music/
nayso@Alok-Aspire:~/Desktop$ ls /home/nayso/Music/
new.txt
```

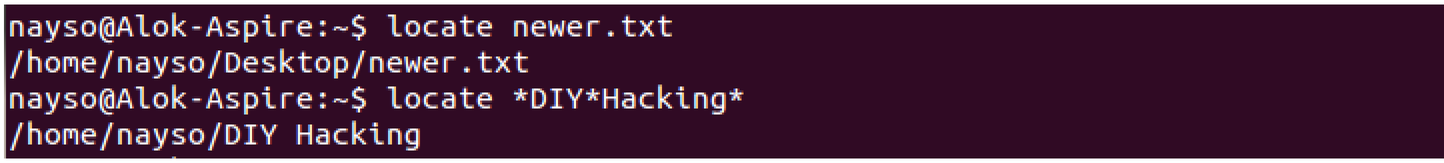
9. mv

The **mv** command is used to move files through the command line. We can also use the **mv** command to rename a file. For example, if we want to rename the file **“text”** to **“new”**, we can use “**mv text new”**. It takes the two arguments just like the **cp** command.


```
nayso@Alok-Aspire:~/Desktop$ ls
new.txt
nayso@Alok-Aspire:~/Desktop$ mv new.txt newer.txt
nayso@Alok-Aspire:~/Desktop$ ls
newer.txt
```

10. locate

The **locate** command is used to locate a file in a Linux System, just like the search command in Windows. This command is useful when you don't know where a file is saved or the actual name of the file. Using the **–**i argument with the command, helps to ignore the case (it doesn't matter if it is Capital or Small). So, if you want a file that has the word "hello", it gives the list of all the files in your Linux System containing the word "hello" when you type in "**locate -i hello**". If you remember two words, you can separate it using asterisk (*). For example, to locate a file containing the words "hello" and "this", you can use the command **"locate –**i ***hello*this "**


```
nayso@Alok-Aspire:~$ locate newer.txt
/home/nayso/Desktop/newer.txt
nayso@Alok-Aspire:~$ locate *DIY*Hacking*
/home/nayso/DIY Hacking
```

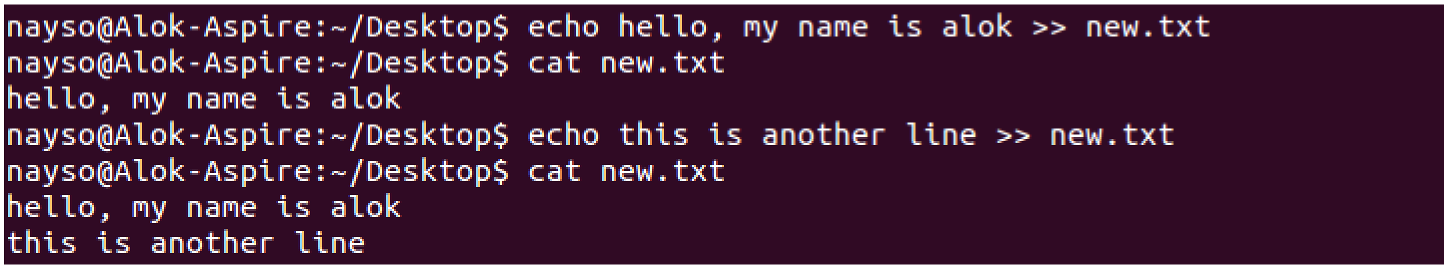
Intermediate Commands

1. echo

"echo" is a command that helps us move some data, usually text into a file. For example, if you want to create a new text file or add into an already made text file, then you just need to type in "**echo hello, my name is alok >> new.txt**". You do not need to separate the spaces by using the backward slash here because we put in two triangular brackets when we finish what we need to write.

2. cat

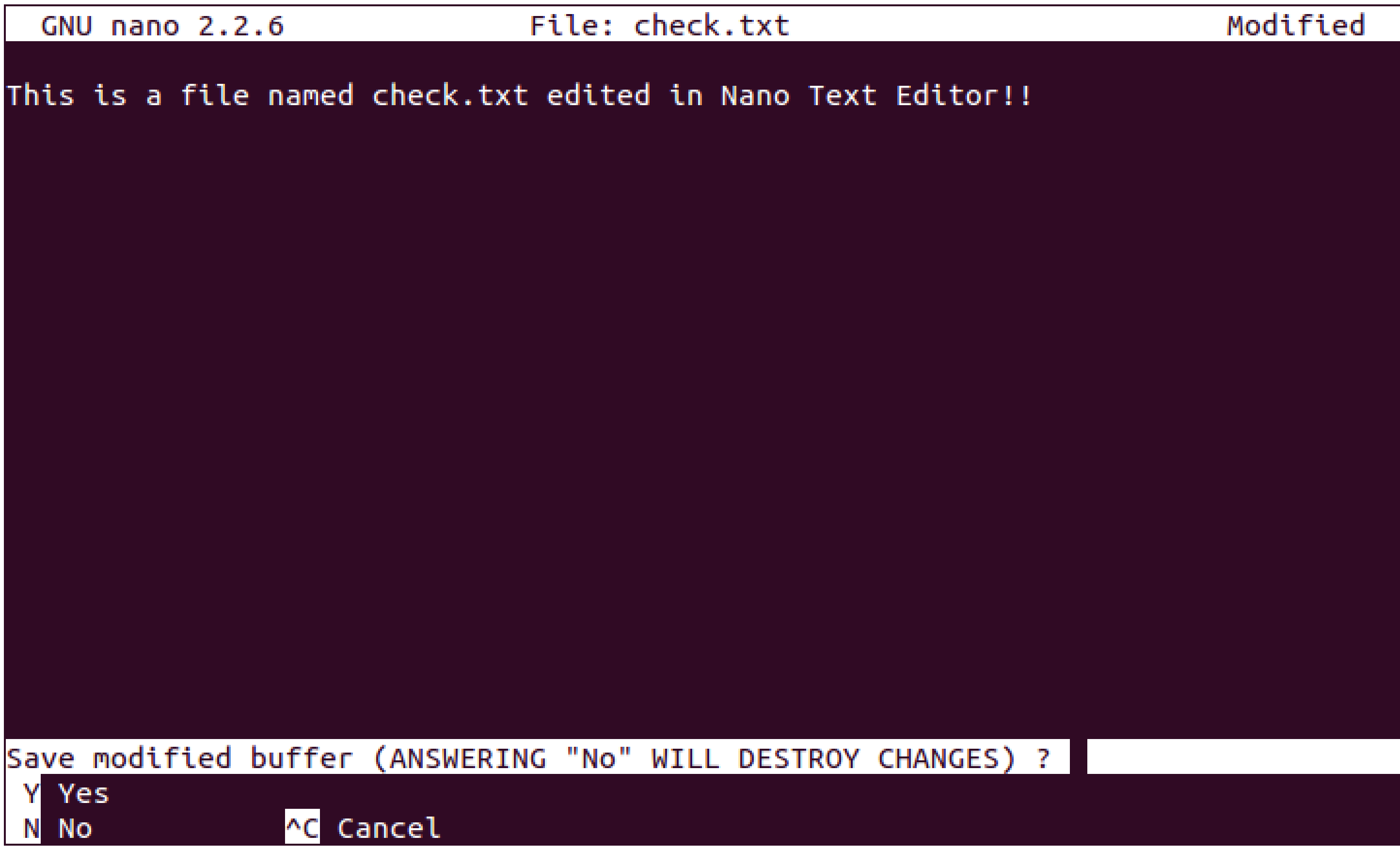
The cat command is used to display the contents of a file, usually used to easily view programs.



```
nayso@Alok-Aspire:~/Desktop$ echo hello, my name is alok >> new.txt
nayso@Alok-Aspire:~/Desktop$ cat new.txt
hello, my name is alok
nayso@Alok-Aspire:~/Desktop$ echo this is another line >> new.txt
nayso@Alok-Aspire:~/Desktop$ cat new.txt
hello, my name is alok
this is another line
```

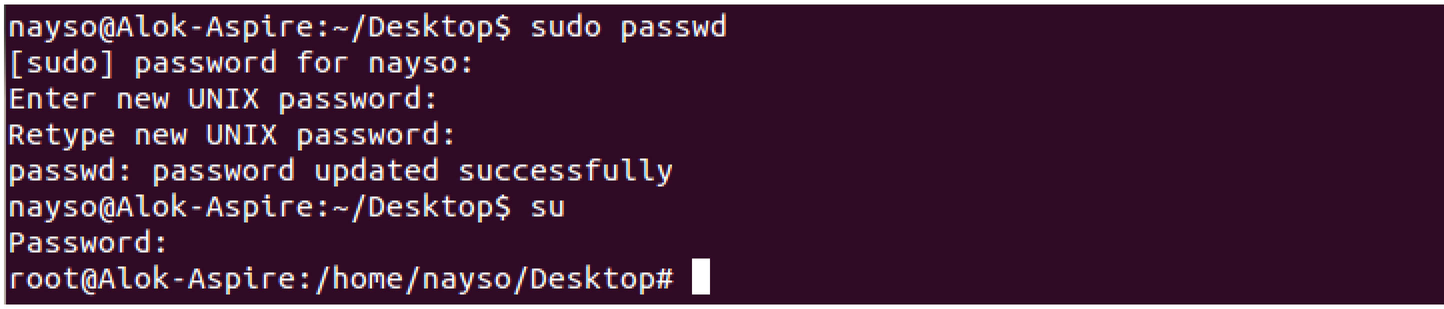
3. nano, vi, jed

nano and **vi** are already installed text editors in the Linux command line. **nano** is a good text editor which denotes keywords with color and can recognize most of the languages. **Vi** is simpler than **nano**. You can create a new file or modify one using this editor. For example, if you need to make a new file named "**check.txt"**, you can create it by using the command **"nano check.txt"**. You can save your files after editing by using the sequence, Ctrl+X, then Y (or N for no). In my experience, using **nano** for HTML editing doesn't seem so good, because of its color, so I recommend jed text editor. We will come to installing packages soon.


```
  GNU nano 2.2.6               File: check.txt                         Modified

This is a file named check.txt edited in Nano Text Editor!!













Save modified buffer (ANSWERING "No" WILL DESTROY CHANGES) ?
 Y Yes
 N No             ^C Cancel
```

4. sudo

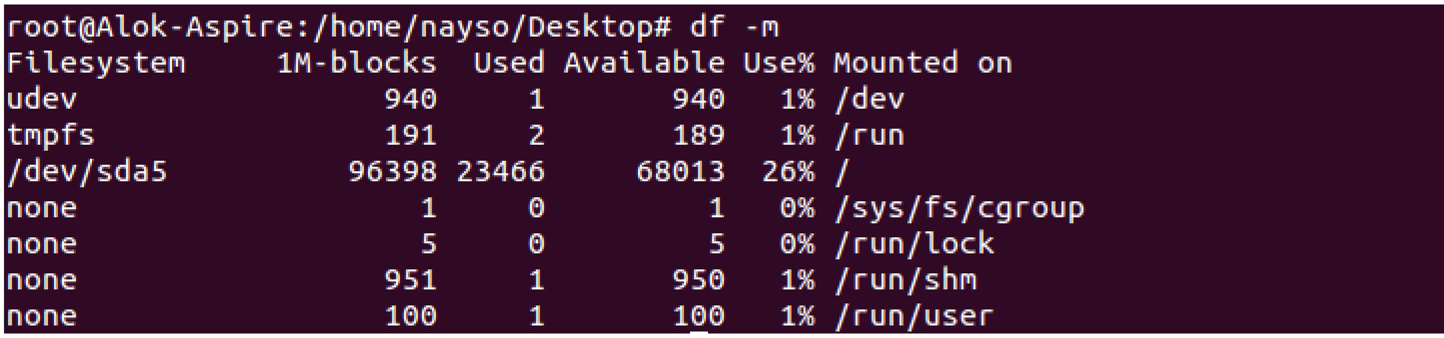
sudo is a widely used command in the Linux command line. **sudo** stands for "SuperUser Do". So, if we want any command to be done with administrative or root privileges, then you can use the **sudo** command. For Example, if you want to edit a file like **viz. alsa-base.conf** which needs root permissions, you can use the command – **sudo nano alsa-base.conf** . You can enter the root command line using the command **"sudo bash"**, then type in your user password. You can also use the command **"su"** to do this, but you need to set a root password before that. For that, you can use the command "**sudo passwd**"(it wasn't misspelled, it is **passwd**). Then type in the new root password.



```
nayso@Alok-Aspire:~/Desktop$ sudo passwd
[sudo] password for nayso:
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
nayso@Alok-Aspire:~/Desktop$ su
Password:
root@Alok-Aspire:/home/nayso/Desktop# 
```

5. df

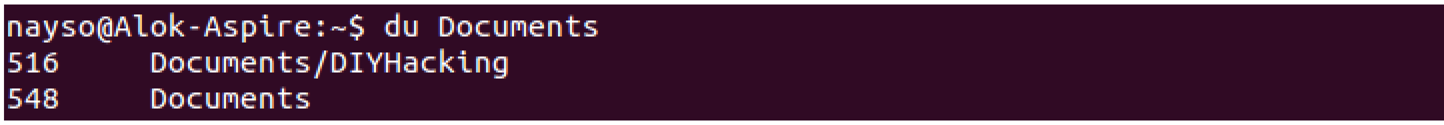
The **df** command is used to see the available disk space in each of the partitions in your system. You can just type in **df** in the command line and you can see each mounted partition and their used/available space in % and in KBs. If you want it shown in megabytes, you can use the command "**df -m**"



```
root@Alok-Aspire:/home/nayso/Desktop# df -m
Filesystem     1M-blocks  Used Available Use% Mounted on
udev                 940     1       940   1% /dev
tmpfs                191     2       189   1% /run
/dev/sda5          96398 23466     68013  26% /
none                   1     0         1   0% /sys/fs/cgroup
none                   5     0         5   0% /run/lock
none                 951     1       950   1% /run/shm
none                 100     1       100   1% /run/user
```

6. du

du is a command to know the disk usage of a file in your System. If you want to know the disk usage for a particular folder or file in Linux, then you can type in the command **df** and the name of the folder or file. For example, if you want to know the disk space used by the folder Documents in Linux, you can use the command "**du Documents**". You can also use the command **"ls -lah"** to view the file sizes of all the files in a folder.



```
nayso@Alok-Aspire:~$ du Documents
516     Documents/DIYHacking
548     Documents
```

7. tar

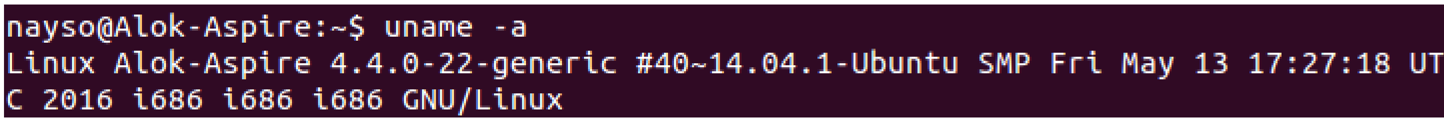
tar is a command used to work with tarballs (or files compressed in a tarball archive) in the Linux Command Line. It has a long list of uses. It can be used to compress and uncompress different types of **tar** archives like **.tar**, **.tar.gz**, .**tar.bz2**.etc. It works on the basis of the arguments given to it. For example, **tar -cvf** for creating a **.tar** archive, **-xvf** to untar a tar archive, **-tvf** to list the contents of the archive.etc. As it is a wide topic, here are some examples of tar commands.

8. zip, unzip

zip is a command used to compress files into a zip archive, **unzip** is used to extract files from a zip archive.

9. uname

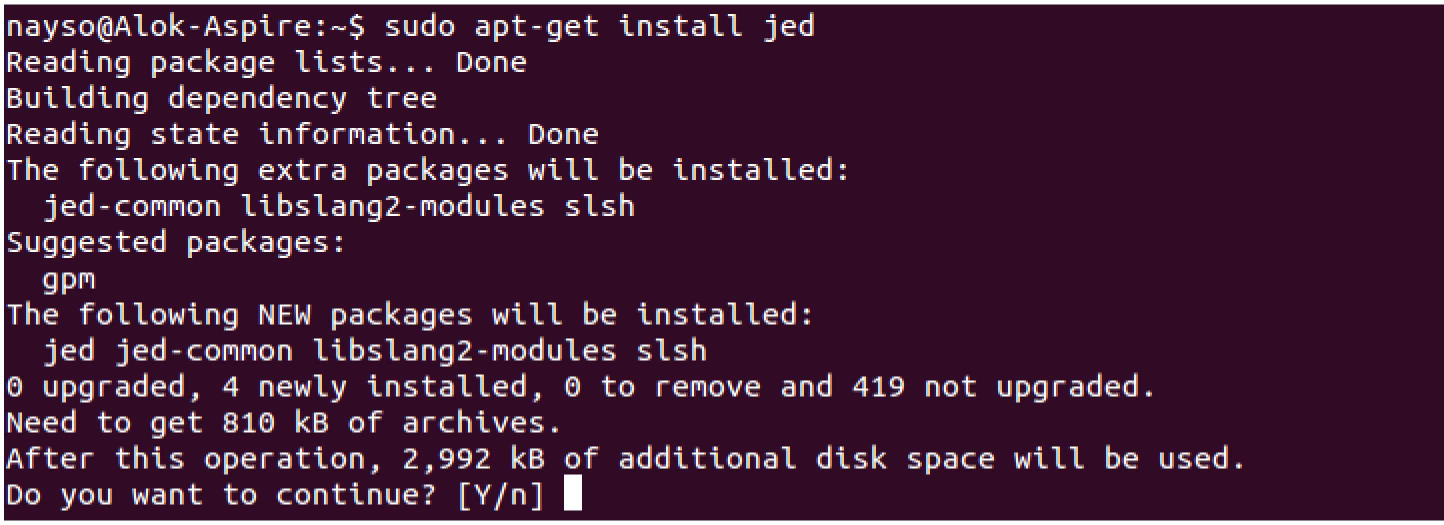
uname is a command used to show the Information about the system your Linux distro is running. Using the command "**uname -a**" prints most of the information about the system. This prints the Kernel release date, version, processor type. etc.



```
nayso@Alok-Aspire:~$ uname -a
Linux Alok-Aspire 4.4.0-22-generic #40~14.04.1-Ubuntu SMP Fri May 13 17:27:18 UT
C 2016 i686 i686 i686 GNU/Linux
```

10. apt-get

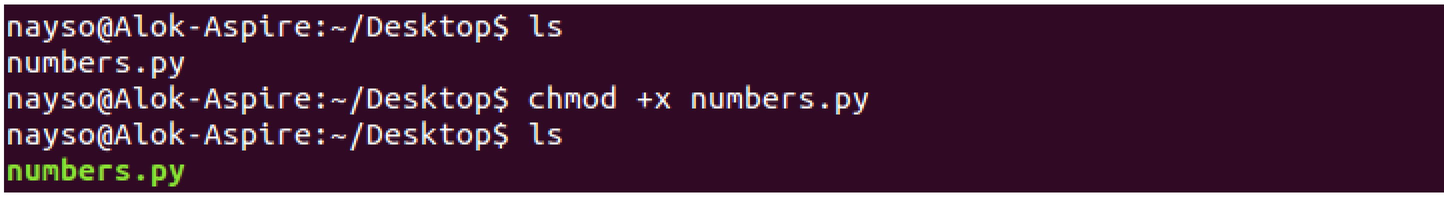
apt is a command used to work with packages in the Linux command line. **apt-get** is a command used to install packages. This requires root privileges, so we use the **sudo** command with it. For example, if we want to install the text editor jed (as I mentioned earlier), we can type in the command "**sudo apt-get install jed**". Similarly, any packages can be installed like this. It is good to update your repository each time you try to install a new package. You can do that by typing **"sudo apt-get update"**. You can upgrade the system by typing **"sudo apt-get upgrade"**. We can also upgrade the distro by typing **"sudo apt-get dist-upgrade"**. The command **"apt-cache search"** is used to search for a package. If you want to search for one, you can type in **"apt-cache search jed"**(This doesn't require root).



```
nayso@Alok-Aspire:~$ sudo apt-get install jed
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  jed-common libslang2-modules slsh
Suggested packages:
  gpm
The following NEW packages will be installed:
  jed jed-common libslang2-modules slsh
0 upgraded, 4 newly installed, 0 to remove and 419 not upgraded.
Need to get 810 kB of archives.
After this operation, 2,992 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

11. chmod

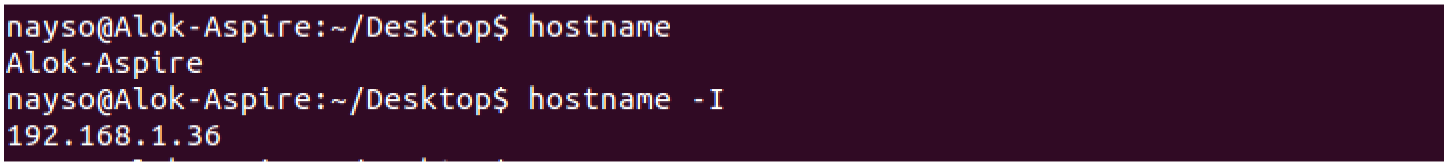
chmod is the command used to make a file executable and to change the permissions granted to it in Linux. Imagine you have a python code named **numbers.py** in your computer, you'll need to run **"python numbers.py"** every time you need to run it. Instead of that, when you make it executable, you'll just need to run **"numbers.py"** in the terminal to run the file. To make a file executable, you can use the command **"chmod +x numbers.py"** in this case. You can use **"chmod 755 numbers.py"** to give it root permissions or **"sudo chmod +x numbers.py"** for root executable. Here is some more information about the **chmod** command.



```
nayso@Alok-Aspire:~/Desktop$ ls
numbers.py
nayso@Alok-Aspire:~/Desktop$ chmod +x numbers.py
nayso@Alok-Aspire:~/Desktop$ ls
numbers.py
```

12. hostname

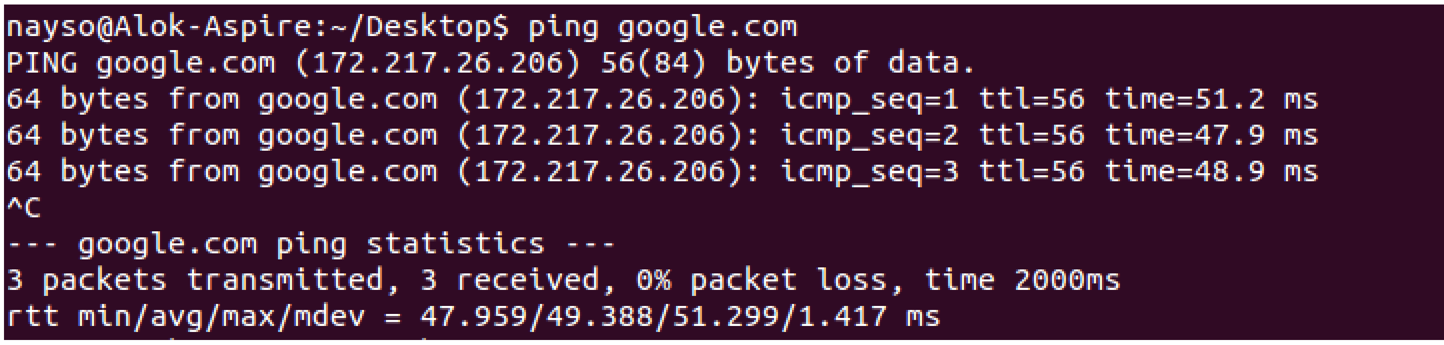
hostname is a command used to know your name in your host or your network. Basically, it displays your hostname and IP address. Typing just **"hostname"** gives the output, your hostname. Typing in "**hostname -I**" gives you your IP address in your network.



```
nayso@Alok-Aspire:~/Desktop$ hostname
Alok-Aspire
nayso@Alok-Aspire:~/Desktop$ hostname -I
192.168.1.36
```

13. ping

ping is a command used to check your connection to a server. Wikipedia says that "**Ping** is a computer network administration software utility used to test the reachability of a host on an Internet Protocol (IP) network". Simply, when you type in, for example, "**ping google.com**", it checks if it can connect to the server and come back. It measures this round-trip time and gives you the details about it. The use of this command for simple users like us is to check your internet connection. If it pings the Google server (in this case), you can confirm that your internet connection is active!



```
nayso@Alok-Aspire:~/Desktop$ ping google.com
PING google.com (172.217.26.206) 56(84) bytes of data.
64 bytes from google.com (172.217.26.206): icmp_seq=1 ttl=56 time=51.2 ms
64 bytes from google.com (172.217.26.206): icmp_seq=2 ttl=56 time=47.9 ms
64 bytes from google.com (172.217.26.206): icmp_seq=3 ttl=56 time=48.9 ms
^C
--- google.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 47.959/49.388/51.299/1.417 ms
```

Tips and Tricks in Using Linux Command Line

- You can use **clear** command to clear the terminal if it gets filled up with too many commands!
- **TAB** can be used to fill up in Terminal. For example, You just need to type **“cd Doc”** and then **TAB** and the terminal fills the rest up and makes it **“cd Documents”**.
- **Ctrl+C** can be used to stop any command in terminal safely. If it doesn’t stop with that, then **Ctrl+Z** can be used to force stop it.
- You can exit from the terminal by using the **exit** command.
- You can power off or reboot the computer by using the command **sudo halt** and **sudo reboot.**

The Modes

Some people disagree on how many modes Vim actually has. I'm going to define three: insert mode, command mode, and last-line mode. Let's start with the default mode you'll see when you start up Vim--command mode.

When you run vim *filename* to edit a file, Vim starts out in command mode. This means that all the alphanumeric keys are bound to commands, rather than inserting those characters. Typing j won't insert the character "j"--it will move the cursor down one line. Typing dd will delete an entire line, rather than inserting "dd."

To enter the *insert* mode, type i (for "insert") and now the keys will behave as you'd expect. You can type normally until you want to make a correction, save the file, or perform another operation that's reserved for command mode or last-line mode. To get out of insert mode, hit the Escape key.

Once you press Escape, you're in command mode again. What if you'd like to save your file or search through your document? No problem, press : and Vim will switch to *last-line* mode. Vim is now waiting for you to enter a command like :w to write the file or :q to exit the editor.

If that all sounds complicated, it's really not. It does take a few days to start training your brain to move between the modes and memorizing the most important keys for movement, commands, and so on. But once you start getting the hang of it, you'll be surprised by how fluid it is editing a file in Vim. Let's walk through some of the most common commands that you need to know.

The Basics of Moving in Vim

The first thing you'll want to learn is how to move around a file. When you're in command mode, you'll want to remember the following keys and what they do:

- h moves the cursor one character to the left.
- j moves the cursor down one line.
- k moves the cursor up one line.
- l moves the cursor one character to the right.
- 0 moves the cursor to the beginning of the line.
- $ moves the cursor to the end of the line.
- w move forward one word.
- b move backward one word.
- G move to the end of the file.
- gg move to the beginning of the file.
- `. move to the last edit.

Here's a handy tip: prefacing a movement command with a number will execute that movement multiple times. So, if you want to move up six lines, enter 6k and Vim will move the cursor up six lines. If you want to move over five words, enter 5w. To move 10 words back, use 10b.

Keep that tip in mind--you'll find that prefacing other commands with a number can come in handy as well.