Linux

- First released October 5, 1991 by Linus Torvalds.
- Linux runs on a wide variety of computer hardware, including mobile phones, tablet computers, network routers, televisions, video game consoles, desktop computers, mainframes and supercomputers.
- Linux is a leading server operating system, and runs the 10 fastest supercomputers in the world.
- More than 90% of today's supercomputers run some variant of Linux.
- Linux is an operating system kernel, and UNIX is a certification for operating systems

Basic Features

Following are some of the important features of Linux Operating System.

- **Portable** − Portability means software can works on different types of hardware in same way. Linux kernel and application programs supports their installation on any kind of hardware platform.
- **Open Source** − Linux source code is freely available and it is community based development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
- **Multi-User** − Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
- **Multiprogramming** − Linux is a multiprogramming system means multiple applications can run at same time.
- **Hierarchical File System** − Linux provides a standard file structure in which system files/ user files are arranged.
- **Shell** − Linux provides a special interpreter program which can be used to execute commands of the operating system. It can be used to do various types of operations, call application programs. etc.
- **Security** − Linux provides user security using authentication features like password protection/ controlled access to specific files/ encryption of data.

Basic Commands

-

- Type ***uname*** *and Linux will tell his name to you*

```
$ uname
    Linux
```

- If you want to know your name type ***whoami***

```
$ whoami
    user1
$ who am i
    user1       tt3a    Jan 12 07:14
$ who
    user1       tty3a   Jun 10 09:15
    user2       tty3c   Jun 10 09:25
```

- To view a line of text in the shell: ***echo***

echo 'welcome to linux'

- To clear the shell : ***clear***
- **touch:**

  - There are two commands to create file.
    - *touch*
    - *cat*

```
$ touch sample
$ touch sample1 sample2 sample3
```

  - The size of the file would be zero bytes since *touch* doesn't allow to store anything in a file.
  - Useful to create several empty files quickly.

- **cat:**

  - Store lines in a file

```
$ cat > test
Hello world
I love my country
```

  - After completion of writing press the keys ***ctrl+d***.
  - In Unix the keys ***ctrl+d*** indicate the EOF or end of file character.

- **cat:**
  - To see the contents of the file test

```
$ cat test
Hello world
I love my country
```

  - It can concatenate the contents of two files and store them in the third file.

```
$ cat sample1 sample2 > newsample
```

  - This would create newsample which contains contents of sample1 followed by that of sample2.

```
$ cat sample1 sample2 >> newsample
```

- **cp** — Use the **cp** command to copy files through the command line. It takes two arguments: The first is the location of the file to be copied, the second is where to copy.

```
$ cp  /usr/home/chapter1  /usr/newhome/ch1
```

- **mv** — Use the **mv** command 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.

```
$ mv  olddir  newdir
```

- **rm** - Use the **rm** command to delete files and directories. Use "**rm -r**" to delete just the directory. It deletes both the folder and the files it contains when using only the **rm** command.

```
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$
```

- **ls -** To list information about directory or files : **ls**
  This command contains some options.

✓ -h [print sizes in human readable format]
✓ -l [use a long listing format]
✓ -S [sort by file size]

```
$ ls p*
pakde
pommies

$ ls ?ain
gain
main
Pain

$ ls /mydir/*x
cc_fax
i_tax
myUnix
```

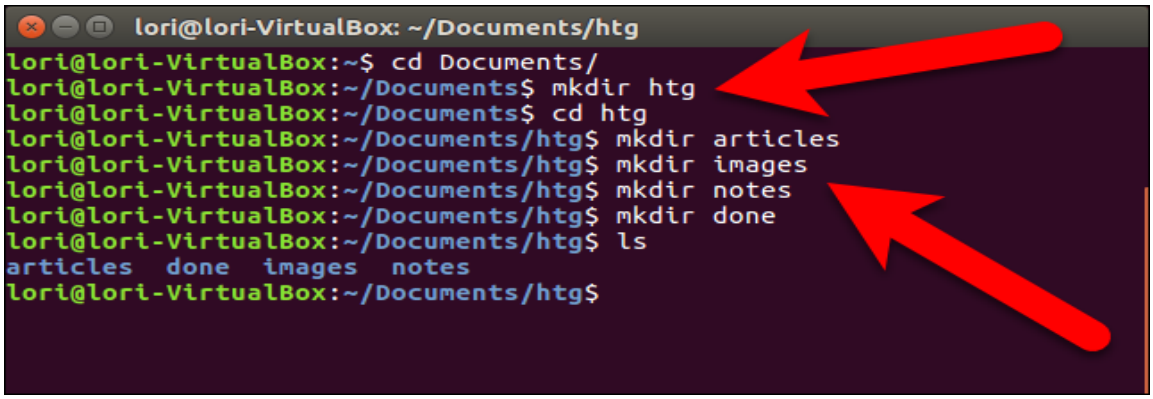
```
$ ls [aciou]*
area
even
ion

$ ls [!aciou]*
dell
sam

$ ls [a-m][c-z][4-9]??
az6ps
```

```
$ ls  -l
total 22
-rwxr-x--x   1   user1   group   24 Jun 06 10:12  carribeans
-rwxr-x-wx   1   user1   group   23 Jun 06 00:05  kangaroos
-rwxr-xr-x   1   user1   group   12 Jun 06 12:54  kiwis
drwxr-xr-x   1   user1   group   10 Jun 06 11:09  mydir
-rwxr-xrwx   2   user1   group   22 Jun 06 14:04  pakde
-rwxrwxr-x   2   user1   group   16 Jun 06 22:25  pommies
-rwxr-xr-x   1   user1   group   04 Jun 06 23:16  springboks
-rwxr-xr-x   1   user1   group   04 Jun 06 10:17  zulus
```

- **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 file system. It is denoted by a forward slash(/). The user directory is usually something like "/home/username".

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

- **cd** — Use the **"cd"** command 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.



- **mkdir & rmdir** — Use the **mkdir** command 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**". Use **rmdir** to delete a directory. But **rmdir** can only be used to delete an empty directory. To delete a directory containing files, use **rm**.

```
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$
```

- **df** — Use the **df** command 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
```

- **du** — Use **du** 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, 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 documents folder 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
```

- **hostname** — Use **hostname** to know your name in your host or network. Basically, it displays your hostname and IP address. Just typing "**hostname**" gives the output. 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
```

- Each file or directory has 3 security groups.

    a. Owner
    b. Group
    c. All Others

Each security group has 3 flags that control the access status : read, write, execute. They are listed as 'rwx' or a "-" if the access is turned off.

- ✓ rwxrwxrwx[read, write and executable for owner, group and all others]
- ✓ rw-r--r--[read and write by owner, read only for group and all others]

To change the permissions type chmod.

- ✓ u, g, o or all [whose permission you are changing]
- ✓ + or - [type of change: add or subtract permission]

To change Permission

- ✓ chmod [who] [+/-/=] [permissions] file
- ✓ file or directory [name of file or directory to change]
- ✓ chmod go+rw file1 file2 add read and write access for group and others for files 'file1' and 'file2'
- ✓ chmod a+rwx file1 add read, write and execute for everyone for 'file1'.

The sums of these numbers give combinations of these permissions:

i. 0 = no permissions whatsoever; this person cannot read, write, or execute the file
ii. 1 = execute only
iii. 2 = write only
iv. 3 = write and execute (1+2)

v. 4 = read only
vi. 5 = read and execute (4+1)
vii. 6 = read and write (4+2)
viii. 7 = read and write and execute (4+2+1)

Chmod commands on file apple.txt (use wildcards to include more files)

Command	Purpose
chmod 700 apple.txt	Only you can read, write to, or execute apple.txt
chmod 777 apple.txt	Everybody can read, write to, or execute apple.txt
chmod 744 apple.txt	Only you can read, write to, or execute apple.txt Everybody can read apple.txt;
chmod 444 apple.txt	You can only read apple.txt, as everyone else.

You can use the ls command with the -l option to show the file permissions set. For example, for apple.txt, I can do this:

```
$ ls -l apple.txt
-rwxr--r--   1 december december       81 Feb 12 12:45 apple.txt
$
```

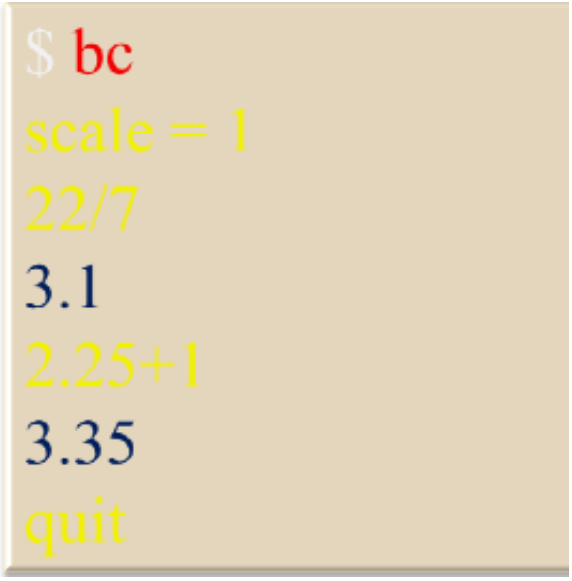
The sequence -rwxr--r-- tells the permissions set for the file apple.txt. The first - tells that apple.txt is a file. The next three letters, rwx, show that the owner has read, write, and execute permissions. Then the next three symbols, r--, show that the group permissions are read only. The final three symbols, r--, show that the world permissions are read only.

A Bit of Mathematics

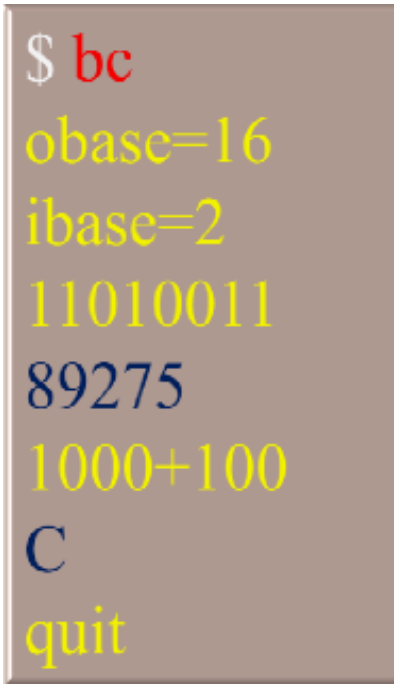
- Calculator is invoked at shell prompt by typing ***bc*** (basic calculator).
- The input to the calculator is taken line by line.
- By typing ***bc*** at prompt the calculator mode starts and the $ the prompt disappears.
- Typing **quit** ends with ***bc***.

```
$ bc
10/2*2
10
2.5*2.5+2
8.25
quit
```

- Working with floats



- After Setting the scale variable if the answer of an expression turns out more than what scale can provide then the value in scale is ignored and the correct answer is displayed.
- Working with different base
- By setting the variable ibase to 2 and obase to 16 all input that is supplied is taken as binary whereas all output is displayed in hexadecimal



- Working with function

```
$ bc
Sqrt(196)
14
```

- Working with variables

```
$ bc
a=4
b=5
c=b-a
c
1
quit
```

```
$ bc
for(i=1 ; i<=5 ; i++) i
1
2
3
4
5
quit
```

```
$ bc
4+2
6
 .+1
7
quit
```

```
Input: $ echo "i=1;while(i<=10) {i; i+=1}" | bc
Output:
1
2
3
4
5
6
7
8
9
10
```

```
for(i=1; i<=5; i++)
{
for(j=1; j<=i; j++)
{ "*"}
Print "\n"
}
output:
*
**
***
****
*****
```

- **Increment/ Decrement:**

Input: $ echo "var=10;++var" | bc
Output: 11
Input: $ echo "var=10;var++" | bc
Output: 10
Input: $ echo "var=10;--var" | bc
Output: 9
Input: $ echo "var=10;var--" | bc
Output: 10

- **Comparison or Relational Operators**

Input: $ echo "10>5" | bc
Output: 1

Input: $ echo "1==2" | bc
Output: 0

- Another math related command in Unix: ***factor***

```
$ factor 15
15: 3 5
$ factor
15
15: 3 5
28
28: 2 2 7
to quit: ctrl+d or q
```