Part-1

UNIX / LINUX Concepts & Commands

Agenda

Topic-1: Overview of UNIX/Linux Operating System

Topic-2: Linux File System

Topic-3: Linux Installation

Topic-4: ls, date and cal commands

Topic-5: Working with Directories

Topic-6: Working with Files

Topic-7: Comparing Files

Topic-8: Creation of Link Files

Topic-9: Word count command (wc command)

Topic-10: Sorting content of the file

Topic-11: Find unique content in the file by using uniq command

Topic-12: Input and Output of Commands and Redirection

Topic-13: Piping

Topic-14: How to use multiple commands in a single line

Topic-15: Regular Expressions and Wildcard Characters

Topic-16: Command Aliasing

Topic-17: Locate and Find Commands

Topic-18: Compression and Uncompression of files (tar, gzip, gunzip, bzip2, bunzip2)

Topic-19: grep command

Topic-20: cut command

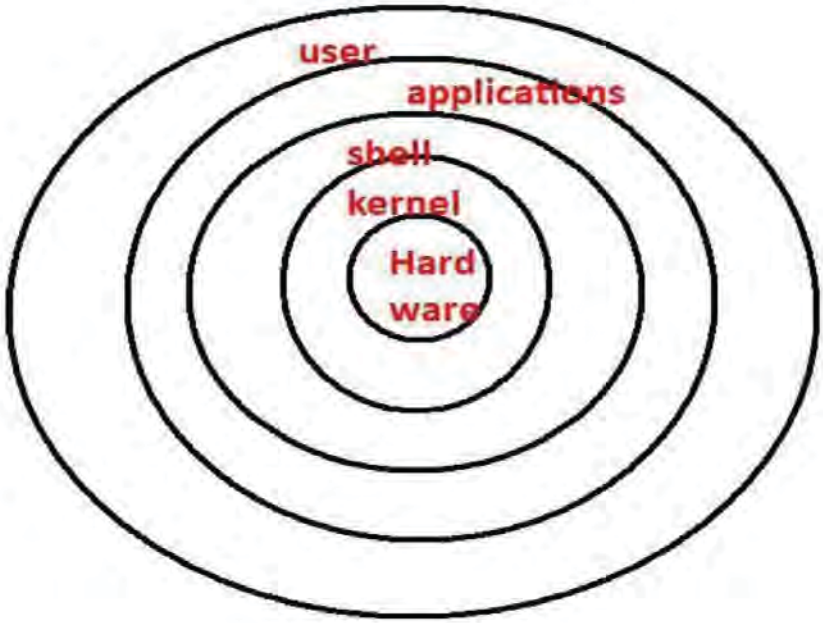
Topic-21: File Permissions

Topic-22: Working with editors

All these flavours have lot of similarity. Hence if we are perfect with one flavour, we can work on any other flavour very easily.

Note: We can view a detailed list of Linux flavours in www.distrowatch.com

Components of UNIX



Shell:

* It is the outer layer of UNIX operating System.
* It reads our command, verify syntax and check whether the corresponding command related application is available or not.
* If everything is proper, then shell interprets our command into kernal understandable form and handover to the kernal.
* Shell acts as interface between user and kernal.

Kernal:

* It is the core component of UNIX operating system.
* It is responsible to execute our commands.
* It is responsible to interact with hardware components.
* Memory allocation and processor allocation will takes care by kernal.

5) Once command execution completed, then shell returns unix prompt ($ OR # OR %).
6) $ OR # OR % represents it is ready for the next command.

Online UNIX Terminal:

http://www.masswerk.at/jsuix

* It is free terminal and written in JavaScript.
* We can access by using any browser.
* We can use this terminal to check very basic commands functionality.

The Most commonly used Basic Commands:

1) pwd → Print working directory
2) ls → List our all files and directories
3) mkdir → Create directory
4) cd → Change directory
5) touch → To create a file
6) rmdir → Remove directory
7) rm → To remove file
8) cal → Display Monthly Calander
9) date → Display current date and time.
10) help → To display list of commands.
11) hello → To display brief system information.
12) clear → To clear terminal.
13) exit → To logout session.

Topic-2: Linux File System

Types of Files in Linux:

In Linux everything is treated as File.
All files are divided into 3 types

1) Normal or Ordinary files:

These files contain data. It can be either text files (like abc.txt) OR binary files (like images, videos etc).

2) Directory Files:

- These files represent directories.
- In windows, we can use folder terminology where as in linux we can use directory terminology.
- Directory can contains files and sub directories.

3) Device Files:

In Linux, every device is represented as a file. By using this file we can communicate with that device.

Note: short-cut commands to open and close terminal

ctrl+alt+t ➔ To open terminal
ctrl+d ➔ To close terminal

How to check File Type:

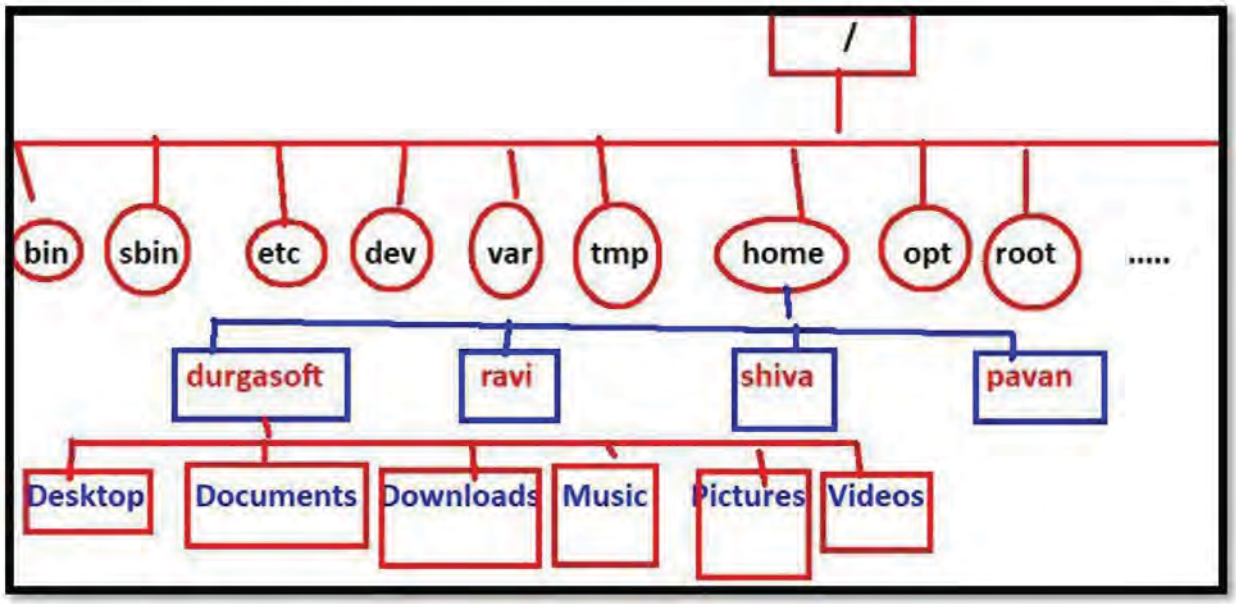
In Ubuntu, blue color files represents directories and all remaining are considered as normal files. This color conventions are varied from flavour to flavour. Hence it is not standard way to check file type.

We have to use 'ls -l' command.

```
/home/durgasoft/Downloads/coreutils-8.31/src
durgasoft@durgasoft:~/Downloads/coreutils-8.31/src$ cd ../../..
durgasoft@durgasoft:~$ pwd
/home/durgasoft
```

Linux File System Hierarchy:

Linux file system has Tree Like Structure.
It starts with root(/).
/ is the topmost directory
This root directory contains the following important sub directories.
bin,sbin,lib,etc,dev,opt,home,usr,tmp,media etc



1) bin Directory:

bin means binary. This directory contains all binary executables related to our linux commands.

2) sbin Directory:

sbin means systembin. It contains all binary executables related to high end admin (super user OR root) commands.
Eg: Disk partitioning, network management etc

Q1) What is the difference between bin and sbin?

bin contains binary executables related to commands used by normal user.
sbin contains binary executables related to commands used by superuser.

3) etc Directory:

- This directory contains all system configuration files. These configurations can be used to customize behaviour of linux os.
- All users information available in /etc/passwd file.
- All groups information available in /etc/group file.
- Hosts information (ip address and dns names) available in /etc/hosts file.

4) tmp Directory:

- tmp means temporary. It contains all temporary files created in the current session.
- If any file is required only for the current session, then create that file inside tmp directory. These files will be deleted automatically at the time of system shutdown.
- If any file which is required permanently, then it is not recommended to create inside tmp directory.

5) dev Directory:

- dev means device.
- In Linux, everything is treated as a file including devices also. i.e every device is represented as a file. By using these files, we can communicate with the devices.
- All device related files will be stored inside dev directory.

Eg:
tty → Terminal related File
fd → Floppy Drive related File
hd → Hard Disk related File
ram → RAM related File
stdin → standard Input Device File (keyboard)
stdout → Standard Output Device File (Terminal/Monitor)
stderr → Standard Error Device File (Terminal/Monitor)

6) mnt Directory:

- mnt means mounting.
- We have to attach external file system files from Pen drive, CD, external hard disk etc to the Linux File System. Then only we can use those external files. This attachment process is called mounting.
- In the old operating systems, we have to perform mounting manually. But in recent operating systems, mounting is performing automatically and we are not required to perform manually.
- The files of manual mounting will be placed inside mnt directory.

7) media Directory:

The files of automatic mounting will be placed inside media directory.

Q2) What is the difference between mnt and media?

mnt → Contains manual mounting files.
media → Contains automatic mounting files.

8) opt Directory:

- opt means optional.
- This directory contains all 3rd party software installation files.

  Eg:

  If we are installing any software explicitly like google chrome, then the corresponding installation files will be stored inside opt directory.

9) lib Directory:

lib means library. It contains Linux os libraries which are required by our commands and applications.

10) var Directory:

- var means variable data. If any data which is keep on changing, such type of data will be stored inside var directory.
- log files will be stored inside var.

11) home Directory:

- As linux is multi user operating system, for every user a separate directory will be created to hold his specific data like videos, images, documents etc. All these user directories will be stored inside home directory.
- $ ls /home
- demo demo1 demo2 durga durga1 durga2 durga5 durgasoft

Note:

/home/durgasoft → Is called durgasoft user home directory. It contains multiple sub directories like Desktop, Downloads, Movies, Pictures etc.

12) proc Directory:

- proc means processes.
- In Linux, multiple processes are running simultaneously. For every process a unique id will be there, which is also known as PID (Process ID).
- The data related to current running processes will be stored inside proc directory. For every process a separate directory will be created inside proc to maintain that data.The name od this directory is same as PID.

Note: We can find all running proceses information by using ps command.
ps means process status.

```
durgasoft@durgasoft:/$ ps -ef
UID          PID  PPID  C STIME TTY          TIME CMD
root           1     0  0 11:05 ?        00:00:03 /sbin/init
root           2     0  0 11:05 ?        00:00:00 [kthreadd]
root           3     2  0 11:05 ?        00:00:00 [rcu_gp]
root           4     2  0 11:05 ?        00:00:00 [rcu_par_g
....
```

13) root Directory:

It is the home directory of super user.

Note:

/home/durgasoft → Durgasoft User Home Directory
/root → Super User Home Directory

Q3) What is the difference between / and Root Directories?

/ acts as root for Linux file system. It is the topmost directory of linux file system.
root is subdirectory of /, which acts as home directory for the super user.

14) boot Directory:

This directory contains the files which are required to boot linux os.

15) usr Directory:

usr means user. This directory contains all user related softwares.

Note:

1) The main advantage of Linux File System is, operating system can locate required files very easily.
2) For every File System, a separate name will be assigned.
3) ext2,ext3,ext4,XFS are names of Linux File Systems.
4) NTFS, FAT are names of Windows File Systems.

Topic-3: Linux Installation

1) Oracle Virtual Box Installation
2) Virtual Machine Installation with Ubuntu OS

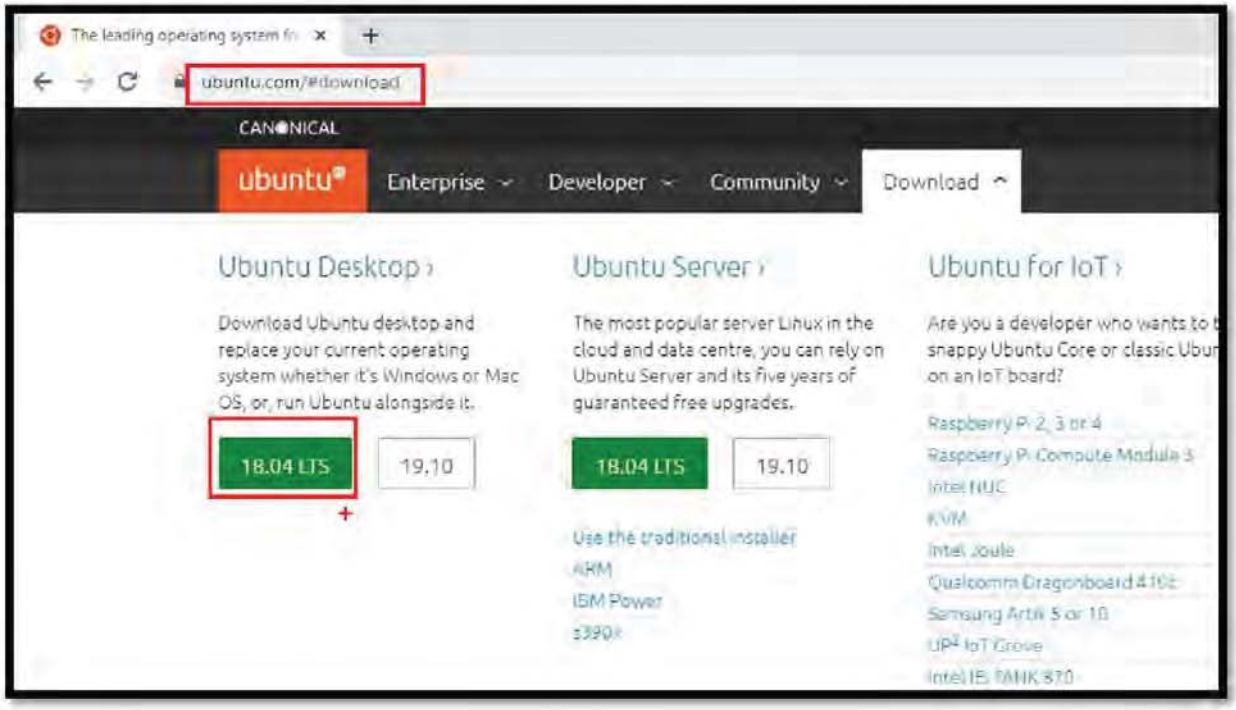
1) Oracle Virtual Box Installation:

To run virtual computers in our system without effecting original computer, we should go for virtual box.
It can extends the capability of our existing computer so that we can run multiple operating systems simultaneously.

Donwload virtual box from:
virtualbox.org → Downloads → Windows hosts
VirtualBox-6.0.14-133895-Win.exe

2) Creation of Virtual Machine with Ubuntu OS:

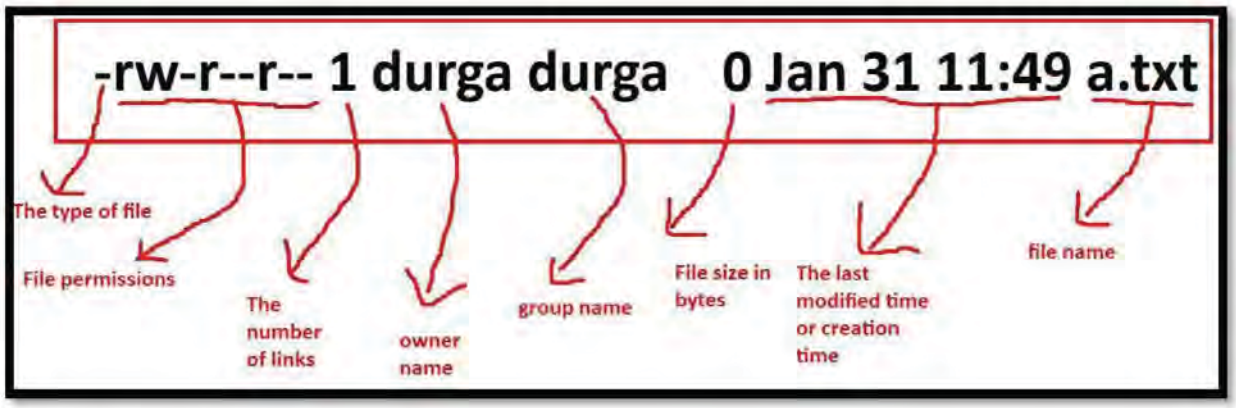
We have to download ubuntu software from
ubuntu.com → Download → Ubuntu Desktop → Ubuntu 18.04.3 LTS → Download
ubuntu-18.04.3-desktop-amd64.iso



Note: **Download softwares from: bit.ly/36evx8y**

Various Utilities:

1) **To make Full Screen: Devices → Insert Guest Additions CD Image**
2) **To open terminal: ctrl+alt+t**
3) **To close terminal: ctrl+d**
4) **To increase font in terminal: ctrl+shift+plus symbol**
5) **To decrese font in terminal: ctrl+ minus symbol**
6) **To copy and paste from windows to ubuntu and from ubuntu to windows:**
   Devices → Shared Clipboard → Bidirectional
7) **To drag and drop files from windows to ubuntu and from ubuntu to windows:**
   Devices → Drag and Drop → Bidirectional



6) ls -t

To display all files based on last modified date and time. Most recent is at top and old are at bottom.

7) ls -rt

To display all files based on reverse of last modified date and time. Old files are at top and recent files are at bottom.

8) ls -a

a means all

To display all files including hidden files. Here . and .. also will be displayed.

9) ls -A

A means almost all

To display all files including hidden files except . and ..

```
durga@durga-VirtualBox:~$ ls -a
.               msmtp.log
..              .msmtprc
.bash_history   Music
.bash_logout    .mysql_history
.bashrc         Pictures
.cache          .profile
.config         Public
.dbus           script1.sh
durga@durga-VirtualBox:~$ la -A
.bash_history   .msmtprc
.bash_logout    Music
.bashrc         .mysql_history
.cache          Pictures
.config         .profile
.dbus           Public
```

10) ls -F

To display all files by type.

directory → /
executable file → *
link file → @

Eg:
initctl@ → Link File
pts/ → Directory
ls* → Executable File

11) ls -f

To disable colors

12) ls -i

- To display all files including inode number.
- i-node is the address of location, where file attributes are stored.
- i-node is the address of the location, where file attributes are stored.

- The following are various file attributes
    1) The size of the file
    2) The number of links
    3) The owner
    4) The group
    5) The creation time
    6) The last modified time
    7) The last accessed time

    etc

13) ls -R

- R means Recursive.
- It will list all files and directories including sub directory contents also. By default ls will display only direct contents but not sub directory contents.

14) ls -s

The number of blocks used by file will be displayed.
1 Block = 1Kb
Note: In ubuntu each block is of 1KB but not 4KB.

15) ls -h

display in human readable format

Note: If the number of files are very huge, then we can use less and more commands with ls to display page by page.

$ ls /dev | less
$ ls /dev | more

If we want only fixed number of files either from top or from bottom we have to use head and tail commands with ls commands.

$ ls /dev | head -5 → disply only top 5 lines
$ ls /dev | tail -5 → display only bottom 5 lines

Note: We can use these options simultaneously. When ever using options simultaneosuly then the order is not important.

Eg: All the following commands are equal
$ ls -l -t -r
$ ls -t -r -l
$ ls -l -r -t
$ ls -ltr
$ ls -trl

Q1) Write the Command to display all Files including Hidden Files in Last Modifiction Time Order. Oldest should be First and recent should be Last. It should include Inode Number and the Number of Blocks used by that File. The Output should be in Long listing Form?
$ ls -atrisl
131279 4 -rw-r--r-- 1 durga durga 807 Jan 3 12:57 .profile
131277 4 -rw-r--r-- 1 durga durga 3771 Jan 3 12:57 .bashrc
162011 4 -rw-r--r-- 1 durga durga 220 Jan 3 12:57 .bash_logout
132496 4 drwx------ 3 durga durga 4096 Jan 3 13:03 .gnupg
132517 4 drwx------ 3 durga durga 4096 Jan 3 13:03 .local
404481 4 drwxr-xr-x 2 durga durga 4096 Jan 3 13:03 Templates

Q2) Which Command will Lists all Files including Hidden Files along with their Inode Numbers?
ls -ai

Q3) Which Command will make a Long listing of all the Files in our System including Hidden Files, sorted by Modification Date (Oldest First)?
ls -latr

Q4) Is -r will List the Files sorted by Modification Date (Oldest First)?

False.

It lists the files based on reverse of alphabetical order of names.

ls –rt → It will list the files sorted by modification date (Oldest first)

Q5) ls -la will not produce the Same Result as ls -al

False

2) date Command:

We can use date command to display date and time of system.

Various Options:

1) date +%D

To display only date in the form: mm/dd/yy

2) date +%T

To display only time in the form: hh:mm:ss

3) date +%d

To display only day value

4) date +%m

To display only month value

5) date +%y

To display only year value in yy form

6) date +%Y

To display only year value in yyyy form.

7) date +%H

To display only Hours value (in 24 hours scale format)

8) date +%M

To display only Minutes value

9) date +%S

To display only Seconds value

Eg 1: To display current system date in dd-mm-yyyy format.
default format: mm/dd/yy
date +%d-%m-%Y

Eg 2: Create an empty file where file name contains current system date.
touch "durgajobs$(date +%d%m%Y).log"

durgajobs31102019.log
durgajobs01112019.log
durgajobs02112019.log
durgajobs03112019.log
durgajobs04112019.log

Eg 3: Create an empty file where file name contains current system date and time
touch "durgajobs$(date +%d%m%Y%H%M%S).log"

durgajobs31102019205834.log

Note:
If the file name contains date and time then that file is said to be timestamped file (file with timestamp)

cal Command:

$ cal → To display current month calendar.
$ cal 2020 → To display total year calendar.
$ cal 1 → To display 1st year calendar.
$ cal 9999 → To display 9999th year calendar.
$ cal 10000 → cal: year '10000' not in range 1..9999
$ cal 08 2019 → To display august 2019th calendar

Note: cal command can provide support only for the years 1 to 9999.

Topic-5: Working with Directories

1) Creation of Directories:

We can create directories by using mkdir command.

1) **mkdir dir1**
   To create a directory

2) **mkdir dir1 dir2 dir3**
   To create multiple directories

3) **mkdir dir1/dir2/dir3**
   To create dir3. But make sure dir1 and in that dir2 should be available already.

4) **mkdir -p dir1/dir2/dir3**

- -p means path of directories.
- All directories in the specified path will be created.
- First dir1 will be created and in that dir2 will be created and within that dir3 will be created.

Case Study-1: Film Heroine's Manager

```
heroines
    sunny
      jan2020,feb2020,mar2020,.....dec2022
  katrina
      jan2020,feb2020,mar2020,.....dec2022
  kareena
      jan2020,feb2020,mar2020,.....dec2022
```

```
jan2020
  schedule_1.txt
  schedule_2.txt
  ...
  schedule_31.txt
```

```
$ mkdir heroines
$ cd heroines
```

$ mkdir sunny katrina kareena
$ mkdir
{sunny,katrina,kareena}/{jan,feb,mar,apr,may,jun,jul,aug,sep,oct,nov,dec}_{2020,2021,2022}

$ touch
{sunny,katrina,kareena}/{jan,feb,mar,apr,may,jun,jul,aug,sep,oct,nov,dec}_{2020,2021,2022}/schedule_{1..31}.txt

Case Study-2:

Create 5 directories named with dir6,dir7,dir8,dir9 and dir10. In these directories create empty files with a.txt,b.txt,c.txt and d.txt

$ mkdir dir{6..10}
$ touch dir{6..10}/{a..d}.txt

Note: *,[],{ } etc are called wild characters. We can use wild card characters in every command.

2) How to remove Directories:

We can remove directories by using rmdir command.

1) **$ rmdir dir1**

   To remove empty directory dir1

2) **2. $ rmdir dir1 dir2 dir3**

   To remove multiple empty directories

Note: rmdir command will work only for empty directories. If the directory is not empty then we will get error. We cannot use rmdir for files. Hence the most useless (waste) command in linux is rmdir.

durga@durga-VirtualBox:~$ rmdir heroines/
rmdir: failed to remove 'heroines/': Directory not empty

If the directory is not empty then to remove that directory we should use rm command. All internal content also will be removed. rm command can work for files also. Hence rm is recommended to use than rmdir.

durga@durga-VirtualBox:~$ rm heroines
rm: cannot remove 'heroines': Is a directory

Whenever we are using rm command for directories, we should use -r or -R option. Here case is not important.

$ rm -r heroines
$ rm -R heroines

Note: In Linux operating system, there is no way to perform undo operation. Once we delete a file or directory, it is impossible to retrieve that. Hence while using rm command we have to take special care.
The following command is the most dangerous command in linux, because it removes total file system.

rm -r /

Various options with rm Command:

1) interactive Option(-i)

While removing files and directories, if we want confirmation then we have to use -i option.

```
durga@durga-VirtualBox:~$ rm -ri dir7
rm: descend into directory 'dir7'? y
rm: remove regular empty file 'dir7/c.txt'? y
rm: remove regular empty file 'dir7/d.txt'? y
rm: remove regular empty file 'dir7/a.txt'? y
rm: remove regular empty file 'dir7/b.txt'? y
rm: remove directory 'dir7'? y
```

2) force removal(-f):

While removing files and directories, if we don't want any error messages, then we should use -f option. It is opposite to -i option.

```
durga@durga-VirtualBox:~$ rm -r dir99
rm: cannot remove 'dir99': No such file or directory
durga@durga-VirtualBox:~$ rm -rf dir99
durga@durga-VirtualBox:~$
```

Even dir99 is not available, we won't get any error message, because we used -f option.

3) verbose Option(-v):

If we want to know the sequence of removals on the screen we should go for -v option.

```
durga@durga-VirtualBox:~$ rm -r dir6
durga@durga-VirtualBox:~$ rm -rv dir8
removed 'dir8/c.txt'
removed 'dir8/d.txt'
removed 'dir8/a.txt'
removed 'dir8/b.txt'
```

removed directory 'dir8'
durga@durga-VirtualBox:~$

Q1) What is the difference between the following 2 Commands?

$ mkdir dir1/dir2/dir3
$ mkdir -p dir1/dir2/dir3

mkdir dir1/dir2/dir3

Only dir3 will be created and compulsory dir1 and in that dir2 should be avilable already. If dir1 or dir2 not available then this command won't work.

mkdir -p dir1/dir2/dir3

-p means complete path
All 3 directories will be created.

Q2) What is the Advanatage of using rm Command over rmdir Command while removing Directories?

rmdir command will work only for empty directories.
rm command will work for both empty and non-empty directories. Even we can rm command for files also.

Q3) Assume that dir1 is an Empty Directory. Which of the following Commands will remove dir1?

- rm dir1
- remove dir1
- rmdir dir1
- del dir1

- Ans: C

Q4) Assume that dir1 is non-empty Directory. Which of the following Commands will remove dir1?

- rmdir dir1
- rm -R dir1
- rm -i dir1
- rm -f dir1
- rm -v dir1

- Ans: B

Q5) Assume that dir1 is non empty Directory and text1 is just a Text File. Which of the following Command will remove both dir1 and text1 successfully?

- rm text1 dir1
- rm -v text1 dir1
- rm -R text1 dir1

Ans: C

Q6) How to Create a Directory called pythonclasses in the Videos Directory within User Home Directory?

$ mkdir ~/Videos/pythonclasses

Q7) How to Create a Directory named A and in that a Directory B and inside that a Directory C?

-$ mkdir -p A/B/C

Q8) How many Directories will be created after running the following Command?

$ mkdir {a..c}{1..3}

9 Directories named with a1, a2, a3, b1, b2, b3, c1, c2, c3

Q9) To Create a Directory named with Java Classes, is the following Command valid?

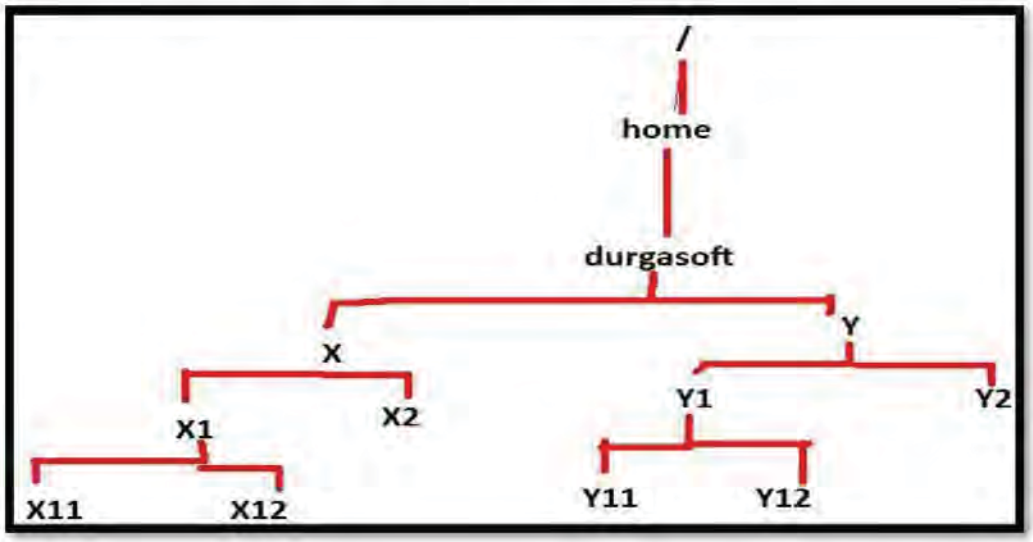
$ mkdir java classes

This command will create two directories java and classes.
To create a single directory we have to use:
$ mkdir "java classes"

Note: In file or directory names, it is not recommended to use space. Instead of that we have to use _ symbol like java_classes.

Case Study:

Write commands to create the following directory structure



1st way:

```
durga@durga-VirtualBox:~$ pwd
/home/durga
durga@durga-VirtualBox:~$ mkdir x y
durga@durga-VirtualBox:~$ cd x
durga@durga-VirtualBox:~/x$ mkdir x1 x2
durga@durga-VirtualBox:~/x$ cd x1
durga@durga-VirtualBox:~/x/x1$ mkdir x11 x12
durga@durga-VirtualBox:~/x/x1$ cd ..
durga@durga-VirtualBox:~/x$ cd ..
durga@durga-VirtualBox:~$ cd y
durga@durga-VirtualBox:~/y$ mkdir y1 y2
durga@durga-VirtualBox:~/y$ cd y1
durga@durga-VirtualBox:~/y/y1$ mkdir y11 y12
```

2nd way:

```
$ mkdir x x/x1 x/x2 x/x1/x11 x/x1/x12 y y/y1 y/y2 y/y1/y11 y/y1/y12
```

3rd way:

```
$ mkdir -p x/x1/x11  x/x1/x12 x/x2  y/y1/y11 y/y1/y12 y/y2
```

4th way:

```
$ mkdir -p x/x{1,2} x/x1/x1{1,2} y/y{1,2} y/y1/y1{1,2}
```

Q10) To Remove Directories dir1, dir2, dir3.... dir10

If Directories are Empty → $ rmdir dir{1..10}

If Directories are non Empty → $ rm -R dir{1..10}

Q11) To Remove 3 Empty Directories dir2, dir4, dir6

$ rmdir dir{2,4,6}

Q12) To Remove Directories where Name contains 2 OR 4 → $ rmdir *[24]*

Q13) To Remove Directories where Name Starts with 'd' → $ rmdir d*

Q14) To Remove Directories where Name Starts with 'd' and Ends with 'n'

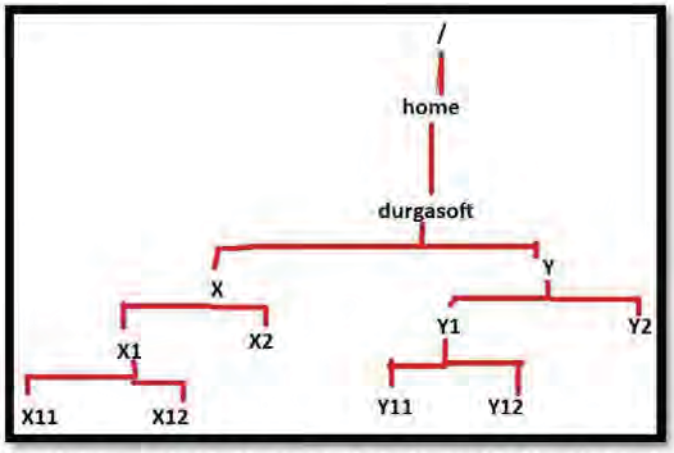
$ rmdir d*n

Q15) To Remove Directoreis where Name Starts with 'd' OR 'x' → $ rmdir [dx]*

Absolute Path vs Relative Path:

Absolute Path: It is the path from root(/) to destination. ie it is complete path.

Relative Path: It is the path from current working directory to destination directory.It is always wrt current location.



$ mkdir -p x/x{1,2} x/x1/x1{1,2} y/y{1,2} y/y1/y1{1,2}

Eg 1: Assume we are in x11 directory. To change to y2 directory

Absolute Path:

$ cd /home/durga/y/y2

$ cd ~/y/y2

Relative Path:

$ cd ../../../y/y2

Eg 2: Assume we are in y2 directory. To change to x11

Absolute Path:

```
$ cd  /home/durga/x/x1/x11
$ cd  ~/x/x1/x11
```

Relative Path:

```
$ cd  ../../x/x1/x11
```

Eg 3: Assume we are in x11 directory. To create y21 directory inside y2 without enter into y2 directory.

Absolute Path:

```
$ mkdir  /home/durga/y/y2/y21
$ mkdir  ~/y/y2/y21
```

Relative Path:

```
$ mkdir  ../../../y/y2/y21
```

4) Copy Command (cp)

1) To Copy from File1 to File2 (File to File)

- $ cp source_file destination_file
- $ cp file1 file2
- Total content fo file1 will be copied to file2.
- If file2 is not already available, then this command will create that file.
- If file2 is already available and contains some data, then this data will be over write with file1 content.

2) To Copy File to Directory:

- $ cp file1 file2 output
- file1 and file2 will be copied to output directory.
- Here we can specify any number of files, but last argument should be directory.
- output directory should be available already.

3) To Copy all Files of One Directory to another Directory:

- $ cp dir1/* dir2
- All files of dir1 will be copied to dir2
- But dir2 should be available already.

4) To Copy Total Directory to another Directory:

- $ cp dir1 dir2
- cp: -r not specified; omitting directory 'dir1'

- Whenever we are copying one directory to another directory, compulsory we should use -r option.

- $ cp -r dir1 dir2
- total dir1 will be copied to dir2

Note:

If the destination directory (dir2) already available then total dir1 will be copied to dir2.
If the destination directory (dir2) not already available, then destination directory will be created and all files of source directory will be copied to destination directory but source directory won't be copied.

5) To Copy Multiple Directories into a Directories:

- $ cp -r dir1 dir2 dir3 dir4 dir5
- dir1,dir2,dir3 and dir4 will be copied to dir5

Q16) Write Command to Copy Data from a.txt, b.txt, c.txt to d.txt?

$ cp a.txt b.txt c.txt d.txt → It won't work.
We will discuss solution in the next classes.

Moving and Renaming Directories:

Both moving and renaming activities can be performed by using single command: mv

1) Renaming of files:

$ mv oldname newname
Eg: $ file1.txt file2.txt
file1.txt will be renamed to file2.txt

2) Renaming of Directories:

$ mv dir1 dir2
dir1 will be renamed to dir2

3) Moving files to directory:

$ mv a.txt b.txt c.txt output
a.txt,b.txt and c.txt will be moved to output directory.

4) Moving of all files from one directory to another directory:

$ mv dir1/* dir2
All files of dir1 will be moved to dir2. After executing this command dir1 will become empty.

5) Moving total directory to another directory:

$ mv dir1 dir2

Note: If dir2 is already available then dir1 will be moved to dir2. If dir1 is not already available then dir1 will be renamed to dir2.

Summary of Directory related Commands:

```
mkdir dir1
mkdir dir1 dir2 dir3
mkdir dir1/dir2/dir3
mkdir -p dir1/dir2/dir3
mkdir dir{1..6}

rmdir  dir1
rmdir  dir1 dir2 dir3
rm -r  dir1
rm -ri  dir1
rm -rf  dir1
rm -rv  dir1
rm -r dir*
rm -r dir{2..6}
rm -r dir[2,4]
cp file1.txt  file2.txt
cp file1.txt file2.txt file3.txt output
cp dir1/* dir2
cp -r dir1 dir2
mv file1.txt file2.txt
mv dir1 dir2 (rename b'z dir2 not available)
mv  dir1/*  dir2
mv  dir1 dir2 (move dir1 to dir2 because dir2 available)
cd
cd ../../..
cd /
cd ~
cd -
```

Q1) What is the difference between Touch and Cat?

touch for creating empty file where as cat for creating a file with some content.

Q2) How we can perform overwriting and appending with cat Command?

> meant for overwriting

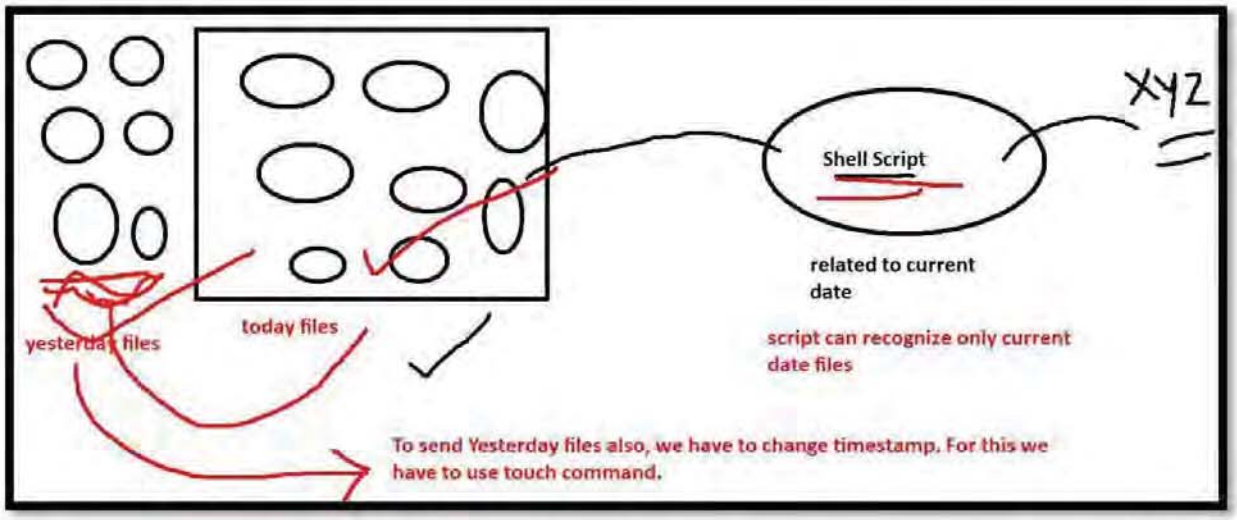
>> meant for appending/concatenation

Q3) If we are using Touch Comamnd, but the File is already available then what will happend?

The content of the file won't be changed. But last modified date and time (i.e., timestamp) will be updated.

```
durga@durga-VirtualBox:~$ ls -l file1.txt
-rw-r--r-- 1 durga durga 77 Jan  9 12:24 file1.txt
durga@durga-VirtualBox:~$ touch file1.txt
durga@durga-VirtualBox:~$ ls -l file1.txt
-rw-r--r-- 1 durga durga 77 Jan  9 12:33 file1.txt
```

Use Case:



Assume that we write one shell script. The job of this shell script is to send all current date files to remote server. Assume that this script won't be executed on sat and sun. But on Monday all 3 days files have to be send.

But the problem with this script is it can recognize only current date files. To change timestamp of sat and Sunday files, we have to use touch command.

Note: We can use touch command for the following two purposes:

1) To create an empty file.
2) To change timestamp of existing file.

2) View Content of the Files

We can view content of the file by using the following commands

1) cat
2) tac
3) rev
4) head
5) tail
6) less
7) more

1. View Content of the File by using cat Command:

$ cat < file1.txt OR $ cat file1.txt
< is optional

```
durga@durga-VirtualBox:~$ cat < file1.txt
This is first line
This is second line
This is third line
This is extra line
durga@durga-VirtualBox:~$ cat file1.txt
This is first line
This is second line
This is third line
This is extra line
```

While viewing file content we can include line numbers by using -n option.

```
durga@durga-VirtualBox:~$ cat -n file1.txt
     1  This is first line
     2  This is second line
     3  This is third line
     4  This is extra line
```

While display file content we can skip blank lines by using -b option.

```
durga@durga-VirtualBox:~$ cat -n file1.txt
     1  This is first line
     2
     3  This is second line
     4
     5  This is third line
```

```
6
7
8    This is extra line

durga@durga-VirtualBox:~$ cat -b file1.txt
1    This is first line

2    This is second line

3    This is third line

4    This is extra line
```

We can view multiple files content at a time by using cat command.

```
$ cat file1.txt  file2.txt   file3.txt
```

Note: The word cat derived from "con'cat'enation"

Various utilities of cat Command:

1) To create new file with some content

```
$ cat > filename
 data
 ctrl+d
```

2) To append some extra data to existing file

```
$ cat >> filename
 extra data
 ctrl+d
```

3) To view content of file

```
 $ cat < filename  or  $ cat filename
```

4) Copy content of one file to another file

```
$ cat input.txt > output.txt
```

5) To copy content of multiple files to a single file

```
$ cat file1.txt file2.txt file3.txt > file4.txt
```

6) Merging/appending of one file content to another file

```
$ cat file1.txt >> file2.txt
```

2. tac Command:

It is the reverse of cat.
It will display file content in reverse order of lines. i.e first line will become last line and last line will become first line.
This is vertical reversal.

```
durga@durga-VirtualBox:~$ cat abc.txt
CAT
RAT
MAT
durga@durga-VirtualBox:~$ tac abc.txt
MAT
RAT
CAT
```

3. rev Command:

rev means reverse.
Here each line content will be reversed.
It is horizontal reversal.

```
durga@durga-VirtualBox:~$ cat abc.txt
CAT
RAT
MAT
durga@durga-VirtualBox:~$ rev abc.txt
TAC
TAR
TAM
```

Note:
cat command will display total file content at a time. It is best suitable for small files. If the file contains huge lines then it is not recommended to use cat command. We should go for head, tail, less and more commands.

4. head Command:

We can use head command to view top few lines of content.

* head file1.txt
  - It will display top 10 lines of file1.txt.
  - 10 is the default value of number of lines.

* head -n 30 file1.txt OR head -30 file1.txt
  - To display top 30 lines of the file.
  - Instead of 30 we can specify any number.

* head -n -20 file1.txt
  To display all lines of file1.txt except last 20 lines.

* head -c 100 file1.txt
  To display first 100 bytes of file content.

5. tail Command:

- We can use tail command to view few lines from bottom of the file.
- It is opposite to head command.

* tail file1.txt
  Last 10 lines will be displayed.

* tail -n 30 file1.txt OR tail -30 file1.txt OR tail -n -30 file1.txt
  It will display last 30 lines.

* tail -n +4 file1.txt
  It will display from 4th line to last line

* tail -c 200 file1.txt
  It will display 200 bytes of content from bottom of the file.

6. more Command:

We can use more command to view file content page by page.

* more file1.txt
  - It will display first page.
  - Enter → To view next line
  - Space Bar → To view next page
  - q → To quit/exit

* more -d file1.txt
  -d option meant for providing details like
  --More--(5%)[Press space to continue, 'q' to quit.]

7. less Command:

- By using more command, we can view file content page by page only in forward direction.
- If we want to move either in forward direction or in backward direction then we should go for less command.

less file1.txt
It will display first page

d → To go to next page.(d means down)
b → To go to previous page. (b means backward)

Eg: Assume a file contains enough data. Write command to display from 3rd Line to 7th Line.

1) Katrina Kaif
2) Kareena Kapoor
3) Karishma Kapoor
4) Sunny Leone
5) Mallika Sharawath
6) Sonakshi Sinha
7) Alia Butt
8) Pooja
9) Anushka
10) Deepika

head -7 demo.txt

1. Katrina Kaif
2. Kareena Kapoor
3. Karishma Kapoor
4. Sunny Leone
5. Mallika Sharawath
6. Sonakshi Sinha
7. Alia Butt

tail -5 demo.txt

3. Karishma Kapoor
4. Sunny Leone
5. Mallika Sharawath
6. Sonakshi Sinha
7. Alia Butt

head -7 demo.txt | tail -5

3. Karishma Kapoor
4. Sunny Leone
5. Mallika Sharawath
6. Sonakshi Sinha
7. Alia Butt

Copying of Files:

cp file1.txt file2.txt

If file2.txt not available, then file2.txt will be created and the content will be copied.
If file2.txt is already available and contanis some data then that data will be overwritten with file1.txt data.
Before overwriting if we want confirmation, then we should go for -i option.
i means interactive.
cp -i file1.txt file2.txt

$ cp -i a.txt b.txt
cp: overwrite 'b.txt'?

If we want verbose output then we can use -v option.
cp -v file1.txt file2.txt

$ cp -v a.txt b.txt
'a.txt' -> 'b.txt'

cp file1.txt file2.txt file3.txt file4.txt output

Note: To copy multiple files content to the single file, we should not use cp command. we should use cat command.

cp a.txt b.txt c.txt d.txt → Invalid
cp: target 'd.txt' is not a Directory

cat a.txt b.txt c.txt > d.txt → Valid

Moving and Renaming Directories:

Both moving and renaming activities can be performed by using single command: mv

1) Renaming of Files:

- **mv oldname newname**
- **mv file1.txt file2.txt**
- **file1.txt will be renamed to file2.txt**

2) Renaming of Directories:

- **mv dir1 dir2**
- **dir1 will be renamed to dir2**

3) Moving of files from one directory to another directory:

- **mv dir1/* dir2**
- **All files of dir1 will be moved to dir2. After executing this command, dir1 will come empty.**

4) Move total directory to another directory:

- **mv dir1 dir2**
- **dir1 will be moved to dir2**
- **In the case of overwriting, if we want confirmation alert then we can use -i option with mv command.**
- **$ mv -i a.txt d.txt dir1**
- **mv: overwrite 'dir1/a.txt'?**

Topic-8: Comparing Files

We can compare data of two files by using the following commands:

1) cmp
2) diff
3) sdiff
4) vidiff
5) comm

1) cmp Comamnd:

It will compare byte by byte.

cmp file1.txt file2.txt
If content is same then we won't get any output.
If the content is different, then it provides information about only first difference. byte number and line number will be provided.

$ cmp a.txt c.txt
a.txt c.txt differ: byte 7, line 2

Note: cmp command won't show all differences and show only first difference.

2) diff Comamnd:

It will show all differences in the content.

diff file1.txt file2.txt
If the content is the same then no output.
If the content is different then it will show all differences.

$ diff a.txt b.txt
$ diff a.txt c.txt
2,3c2,3
< Bunny
< Chinny

> bunny

> chinny
b

For the diff command we can use the following options.
-q shows message when files are different.
-s shows message when files are same | identical
-y shows comparison line by line (parallel comparison)

```
$ diff -q a.txt c.txt
Files a.txt and c.txt differ
$ diff -s a.txt b.txt
Files a.txt and b.txt are identical
$ diff -y a.txt c.txt
Sunny                                   Sunny
Bunny                                 | bunny
Chinny                                | chinny
Vinny                                   Vinny
Pinny                                   Pinny
```

If we want to suppress common lines then we should use --suppress-common-lines option with -y option.

```
$ diff -y --suppress-common-lines a.txt c.txt
Bunny                                 | bunny
Chinny                                | chinny
```

3) sdiff Command:

We can use sdiff command for side by side comparison (parallel comparison)

```
$ sdiff a.txt b.txt
Sunny                                   Sunny
Bunny                                   Bunny
Chinny                                  Chinny
Vinny                                   Vinny
Pinny                                   Pinny
$ sdiff a.txt c.txt
Sunny                                   Sunny
Bunny                                 | bunny
Chinny                                | chinny
Vinny                                   Vinny
Pinny                                   Pinny
```

Note: sdiff comamnd and diff command with -y option are same.

4) vimdiff Command:

- It will highlight differences in vim.

- To support this command, we have to install vim by using the following command.
- sudo apt install vim

- vimdiff a.txt b.txt

- ctrl+w+w → To go to next window
- :q → Close current window
- :qa → Close all windows
- :qa! → Close all windows forcely.

5) comm Command:

By using this command we can compare data of two files.

comm file1.txt file2.txt

It display results in 3 columns

column-1: Data present only in file1.txt but not in file2.txt
column-2: Data present only in file2.txt but not in file1.txt
column-3: Common data of both files.

```
$ comm a.txt c.txt
                Sunny
        bunny
Bunny
        chinny
Chinny
                Vinny

                Pinny
```

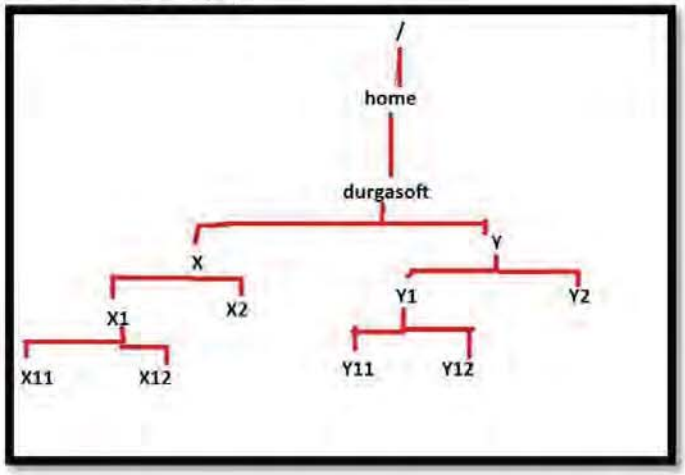
With comm command we can use the following options

-1→ If we don't want to display column-1
-2 → If we don't want to display column-2
-3 → If we don't want to display column-3
-12 → If we don't want to display columns 1 and 2

Note: We can compare files from various builds by using our comparison commands (cmp, diff, sdiff, vimdiff, comm).

RG → RA → D → C → Testing → Production Build-1 released Test.java
RG → RA → D → C → Testing → Production Build-2 released Test.java

Case Study:



mkdir -p x/x1/x1{1,2} x/x2 y/y1/y1{1,2} y/y2

durgasoft@durgasoft-VirtualBox:~$ cat > x/x1/x11/file1.txt
This is file1 content
This is file1 content

Requirement-1: assume file1.txt is available inside x11 directory and we are in user home directory. copy this file to y2 directory.

cp /home/durgasoft/x/x1/x11/file1.txt /home/durgasoft/y/y2
cp ~/x/x1/x11/file1.txt ~/y/y2
cp x/x1/x11/file1.txt y/y2

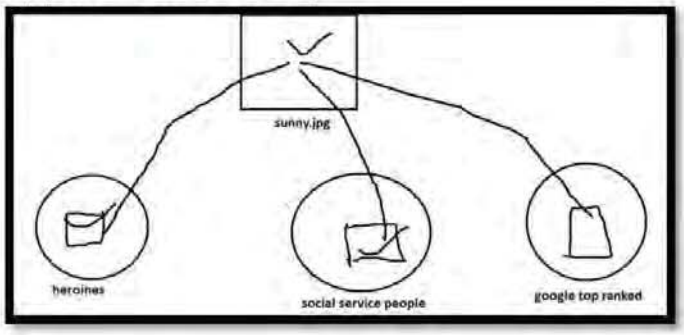
Requirement-2: assume file1.txt is available inside x11 directory and we are in user home directory. move this file to y11 directory.

mv /home/durgasoft/x/x1/x11/file1.txt /home/durgasoft/y/y1/y11
mv ~/x/x1/x11/file1.txt ~/y/y1/y11
mv x/x1/x11/file1.txt y/y1/y11

Topic-9: Creation of Link Files

There are 2 types of link files
1) Hard Link files
2) Soft Link files

1) Hard Link Files:



It is just another name of the same exact file.
We can create hard link file by using ln command.
ln originalfile hardlinkfile

Eg: ln file1.txt file2.txt
Here file1.txt is original file and file2.txt is hard link file.

Important conclusions about hard link file:

1) Both original file and hardlink file have same inode number, same size, same timestamp.
2) If we delete original file, then there is no effect on hardlink file.

2) Soft Link File:

- A softlink is a pointer to another file. It is just like windows shortcut.
- It is also known as symbolic link.
- We can create soft link file by using ln command but with -s option.
- ln -s originalfile softlinkfile

- Eg: ln -s file1.txt file2.txt
- Here file1.txt is original file and file2.txt is link file.

Important conclusions about softlink file:

1) Original file and softlink file have different inode numbers, different file sizes and different timestamps.
2) Usually softlink file has smaller file size than original file size.
3) If we delete original file then softlink files will become useless.

Link files for directories:

We cannot create hardlink for directories because it breaks Linux File System. Having two root directories is meaningless.

```
$ ln dir1 dir2
ln: dir1: hard link not allowed for directory
```

We can create softlink for directories

```
$ ln -s dir1 dir2
```

Note: For files we can create both hard and soft links. But for directories we can create only softlinks butnot hardlinks.

Case Study:

Assume dir1 contains dir2.
dir2 contains softlink dir3 pointing to dir1

dir1 → dir2 → dir3

```
$ mkdir -p dir1/dir2
$ cd dir1/dir2
$ ln -s ~/Desktop/dir1 dir3
```

It will form a loop.

Note: While creating link files there may be a chance of forming loops. Take a bit special care.

Q1) Which of the following is valid about Hard Link?

A) Inode number is different when compared to that of original file.
B) File Size is different when compared to that of original file.
C) It will become useless if we delete original file.
D) Inode number, file size and timestamp are same when compared to that of original file.

Ans: D

Q2) Which of the following is Valid Way to Create Soft Link for sunny.jpg Present in Pictures Directory?

A) ln ~/Pictures/sunny.jpg newimg.jpg
B) ln -s ~/Pictures/sunny.jpg newimg.jpg
C) ln newimg.jpg ~/Pictures/sunny.jpg
D) ln -s newimg.jpg ~/Pictures/sunny.jpg

Ans: B

Q3) We can Create both Hard and Soft Links to the Directories. Is it Valid?

A) True
B) False

Ans: B

Note: If we perform any change to the content of original file, then these changes will be reflected to the link file. Similarly, if we perform any change to the link file, then those changes will be reflected to the original file. This is true for both hard and soft links.

Topic-10: Word Count Command (wc Command)

We can use wc command to count number of lines, words and characters present in the given file.

wc filename
no_of_lines no_of_words no_of_characters filename

Eg:

```
$ wc a.txt
 4  26 166 a.txt
```

4 → Number of Lines
26 →Number of words
166 → Number of characters (File size in bytes)

We can use the following options with wc Command

-l → To print only number of lines
-w → To print only number of words
-c → To print only number of characters
-lw → To print only number of lines and words
-lc → To print only number of lines and characters
-wc → To print only number of words and characters
-L → To print number of characters present in Longest Line.

```
$ wc -l a.txt
4 a.txt
$
$ wc -w a.txt
26 a.txt
$ wc -c a.txt
166 a.txt
$ wc -lw a.txt
 4  26 a.txt
$ wc -lc a.txt
 4 166 a.txt
$ wc -wc a.txt
 26 166 a.txt
```

```
$ wc -L a.txt
57 a.txt
```

We can use wc command for multiple files simultaneously.

```
$ wc a.txt b.txt c.txt
 4  26 166 a.txt
 3   4  27 b.txt
 4   4 112 c.txt
11  34 305 total
```

Topic-11: Sorting Content of the File

We can sort data of the file by using sort command.

sort filename
Here sorting is based on alphabetical order.

```
$ cat a.txt
Sunny
Bunny
Chinny
Vinny
Pinny

$ sort a.txt
Bunny
Chinny
Pinny
Sunny
Vinny
```

If we want to sort based on reverse of alphabetical order, then we should use -r option.

```
$ sort -r a.txt
Vinny
Sunny
Pinny
Chinny
Bunny
```

If the file contains alphanumeric data, then first numbers will be considered and then alphabet symbols.

```
$ cat a.txt
7
Sunny
8
Bunny
1
```

If we want to sort based on numeric value then we have to use -n option.
-n means numeric value

```
$ sort -n a.txt
2
7
9
11
2222222
```

Bydefault sort command will display duplicate lines. If we want only unique lines then we have to use -u option.
-u meant for unique lines.

```
$ cat a.txt
1
1
2
2
Sunny
Sunny
Bunny
$ sort a.txt
1
1
2
2
Bunny
Sunny
Sunny
$ sort -u a.txt
1
2
Bunny
Sunny
```

Note: We can use -ur and -un options also.
To play with unique data, there is special command available: uniq

Q1) Assume a.txt contains the following Data

Sunny
Bunny
Chinny
Vinny

Without using -r option with sort command, sort the content based on reverse of alphabetical order and store the result inside sorted.txt?

sort a.txt | tac > sorted.txt
sort -r a.txt > sorted.txt

To remove duplicate lines also:

sort -u a.txt | tac > sorted.txt
sort -ru a.txt > sorted.txt

Sorting Tabular Data by using -k Option:

-k means KEYDEF (key definition). Based on which key (column) we have to sort.

```
$ ls -l /etc | head -10
total 1068
drwxr-xr-x  3 root root    4096 Aug  6 00:34 acpi
-rw-r--r--  1 root root    3028 Aug  6 00:28 adduser.conf
drwxr-xr-x  2 root root    4096 Nov  7 19:47 alternatives
-rw-r--r--  1 root root     401 May 29  2017 anacrontab
-rw-r--r--  1 root root     433 Oct  2  2017 apg.conf
drwxr-xr-x  6 root root    4096 Aug  6 00:30 apm
drwxr-xr-x  3 root root    4096 Aug  6 00:33 apparmor
drwxr-xr-x  8 root root    4096 Nov  7 06:32 apparmor.d
drwxr-xr-x  4 root root    4096 Nov  7 06:33 apport
```

Sort based on File Size in ascending Order:

```
$ ls -l /etc | head -10 | sort -k 5n
total 1068
-rw-r--r--  1 root root     401 May 29  2017 anacrontab
-rw-r--r--  1 root root     433 Oct  2  2017 apg.conf
-rw-r--r--  1 root root    3028 Aug  6 00:28 adduser.conf
drwxr-xr-x  2 root root    4096 Nov  7 19:47 alternatives
drwxr-xr-x  3 root root    4096 Aug  6 00:33 apparmor
drwxr-xr-x  3 root root    4096 Aug  6 00:34 acpi
drwxr-xr-x  4 root root    4096 Nov  7 06:33 apport
drwxr-xr-x  6 root root    4096 Aug  6 00:30 apm
drwxr-xr-x  8 root root    4096 Nov  7 06:32 apparmor.d
```

Sort based on Month:

6th column provides total date. If we want consider only month then we should use M.

```
$ ls -l /etc | head -10 |sort -k 6M
total 1068
-rw-r--r--  1 root root     401 May 29  2017 anacrontab
drwxr-xr-x  3 root root    4096 Aug  6 00:33 apparmor
drwxr-xr-x  3 root root    4096 Aug  6 00:34 acpi
drwxr-xr-x  6 root root    4096 Aug  6 00:30 apm
-rw-r--r--  1 root root    3028 Aug  6 00:28 adduser.conf
-rw-r--r--  1 root root     433 Oct  2  2017 apg.conf
drwxr-xr-x  2 root root    4096 Nov  7 19:47 alternatives
drwxr-xr-x  4 root root    4096 Nov  7 06:33 apport
drwxr-xr-x  8 root root    4096 Nov  7 06:32 apparmor.d
```

Sort based on Number of Links in descending Order:

```
$ ls -l /etc | head -10 |sort -k 2nr
total 1068
drwxr-xr-x  8 root root    4096 Nov  7 06:32 apparmor.d
drwxr-xr-x  6 root root    4096 Aug  6 00:30 apm
drwxr-xr-x  4 root root    4096 Nov  7 06:33 apport
drwxr-xr-x  3 root root    4096 Aug  6 00:33 apparmor
drwxr-xr-x  3 root root    4096 Aug  6 00:34 acpi
drwxr-xr-x  2 root root    4096 Nov  7 19:47 alternatives
-rw-r--r--  1 root root    3028 Aug  6 00:28 adduser.conf
-rw-r--r--  1 root root     401 May 29  2017 anacrontab
-rw-r--r--  1 root root     433 Oct  2  2017 apg.conf
```

Topic-12: Find Unique Content in the File by using uniq Command

We can use uniq command to display unique content in the file.
But to use uniq command, compulsory the file should be sorted, otherwise it won't work properly.

```
$ cat a.txt
Sunny
sunny
Bunny
Chinny
Sunny
Bunny
Chinny

$ uniq a.txt
Sunny
sunny
Bunny
Chinny
Sunny
Bunny
Chinny

$ sort a.txt | uniq
Bunny
Chinny
sunny
Sunny
```

With uniq command we can use multiple options:

- -d → To display only duplicate lines
- -c → To display number of occurrences of each line
- -i → Ignore case while comparing
- -u → To display only unique lines i.e the lines which are not duplicated.

1. To display only duplicate lines:

```
$ sort a.txt | uniq -d
Bunny
Chinny
Sunny
```

2. To display number of occurrences of each line:

```
$ sort a.txt | uniq -c
    2 Bunny
    2 Chinny
    1 sunny
    2 Sunny
```

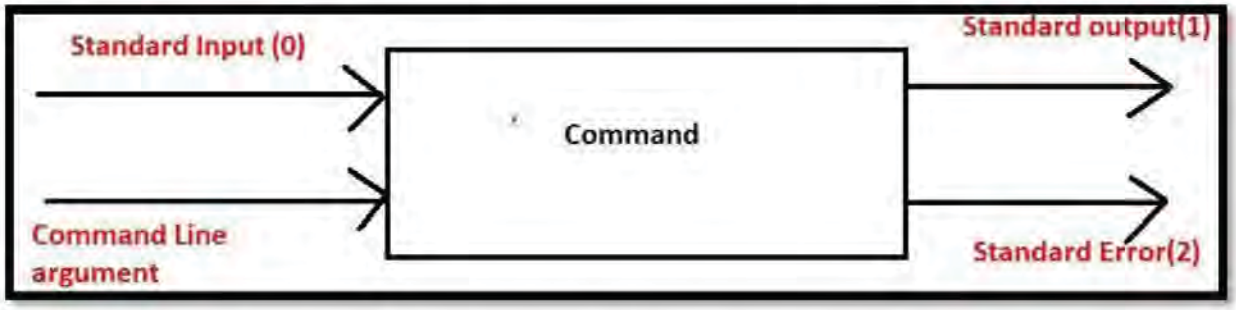
3. To ignore case while comparing:

```
$ sort a.txt | uniq -i
Bunny
Chinny
sunny
$ sort a.txt | uniq -ic
    2 Bunny
    2 Chinny
    3 sunny
```

4. To display only unique lines i.e the lines which are not duplicated:

```
$ sort a.txt | uniq -u
sunny
```

Topic-13: Input and Output of Commands and Redirection



Commands can take input, perform required operation and produces some output. While executing command if anything goes wrong then we will get error message.

Command can take input either from standard Input or from command line arguments. Command will produce results to either Standard output or Standard Error.

Standard Input, Standard Output and Standard Error are Data Streams and can flow from one place to another place. Hence redirection and piping are possible.

Command Line arguments are static and these are not streams. Hence redirection and piping concepts are not applicable to command line arguments.

These data streams are associated with some numbers.
Standard Input associated with 0.
Standard Output associated with 1.
Standard Error associated with 2.

Bydefault Standard input connected with keyboard, Standard output and Standard Error connected with Terminal. But we can redirect.

Standard Input from the keyboard and output to Standard Ouput Device:

```
$cat
read required input from the keyboard
this data will be displayed to the standard output.
ctrl+d
```

Note: For the cat command if we are not providing any arguments, then the input will be taken from standard input device (keyboard) and display the output to the standard output device (Terminal).

$ cat
This is data provided from Standard Input
This is data provided from Standard Input

Input from command line arguments and error messages to the Standard Error:

$ rm file100
rm: cannot remove 'file100': No such file or directory

We are providing filename as command line argument to the rm command.
Specified file not available and hence this command will produces error message to the Standard Error device (Terminal).

Note: Some commands may accept Standard Input and Some commands may accept command line arguments.

1) rm command will always accept command line arguments only.
   rm file1 file2
2) echo command will always aceept command line arguments only.
   echo "durgasoft"
3) cat command can accept input either from standard input or from comamnd line arguments.

Redirection

As Standard Input, Standard Ouput and Standard Error are Data streams, we can redirect these streams.

Redirecting Standard Ouput:

We can redirect standard output by using > and >> symbols.

> will perform overwriting of existing data
>> will perform appending to existing data

Eg 1: To redirect the standard output of cat command from terminal to output.txt

$cat 1> output.txt
sample data
ctrl+d

sample data won't be displayed to the terminal and will write to output.txt

Redirection symbol > is always associated with 1 by default. Hence we are not required to specify 1 explicilty.

$cat > output.txt
sample data
ctrl+d

Instead of overwriting if we want to perform appending then we should use >>.

Redirecting Standard Error:

We can redirect error messages from the terminal to our own file by using > and >> symbols.

$ cal 34 w3892384208342 2>> error.txt

Now error message won't be displayed to the console and will written to error.txt.
For error redirection 2 is mandatory.

Redirecting Standard Input:

We can redirect standard input from keyboard to our required file.
We can perform input redirection by using < symbol.

$cat 0< a.txt 1>>output.txt 2>>error.txt

< symbol is always associated with 0 by default. Hence we can remove.
$cat < a.txt >>output.txt 2>>error.txt

***Note: To redirect both standard output and standard error to the same destination we can use shortcut as follows

$ cat < a.txt &> output.txt
&> means both standard output and standard error.

Redirecting Standard output from one terminal to another terminal:

In unix every thing is treated as file even our terminal also.
We can find terminal related file by using tty command.

termainal - 2:
$ tty
/dev/pts/1

terminal - 1:
$ls -l 1> /dev/pts/1

terminal-1 long listing output will be displayed to terminal-2

Bits

Q1) In How Many Ways Command can get Input?

2 ways. Either from Standard Input or from command line arguments.

Q2) Which of the following contains Data Streams?

A) Standard Input
B) Standard Output
C) Standard Error
D) Command Line arguments

Ans: A,B,C

Q3) By Default Standard Input connected to

A) Terminal
B) Keyboard
C) A File

Ans: B

Q4) By Default Standard Ouput connected to Terminal

Q5) By Default Standard Error connected to **Terminal**

Q6) What Number represents Standard Input Stream? **0**

Q7) What Number represents Standard Output Stream? **1**

Q8) What Number represents Standard Error? **2**

Q9) How we can redirect Standard Output of the ls Command to a File called Output.txt?

A) ls 2> output.txt
B) ls 0< output.txt
C) ls 1< output.txt
D) ls 1> output.txt

Ans: D

ls 2> output.txt → redirecting standard error from terminal to output.txt
ls 0< output.txt → redirecting standard input from keyboard to output.txt
ls 1< output.txt → Meaningless

Q10) How we can redirect the Standard Output of the ls Command to Output.txt, but at the Same Time, redirect Standard Error to error.txt?

ls 1> output.txt 2> error.txt

ls > output.txt 2> error.txt

Q11) Explain the difference between <, >, >> in Redirection?

< symbol meant for input redirection

> symbol meant for output redirection where the existing data will be overwritten.

>> symbol meant for output redirection where the data will be appended instead of overwriting.

Topic-14: Piping

Sometimes we can use output of one command as input to another command. This concept is called piping.
By using piping, multiple commands will work together to fulfill our requirement.

We can implement piping by using vertical bar (|).

```
$ ls -l /etc | wc
   215   1940   11872
```

First ls got executed and the output of this command will become input to wc command.

Eg 2: $ ls -l /etc | more

Eg 3: $ ls -l /etc | wc |wc -l
The output is: 1

Eg 4: $ ls -l /etc | head -5

Note: instead of ls -l we can use ll command, most of linux flavours provides support.

tee Command:

Requirement:

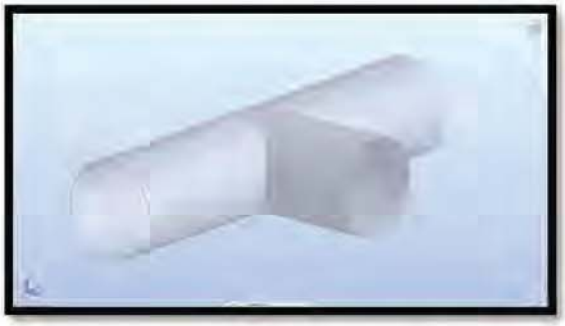
The output of the ls command should be saved to output.txt and should be provided as input to wc command:

```
ls -l 1>output.txt | wc
```

This command won't work because if we are using redirection in the middle of piping, it will break piping concept.

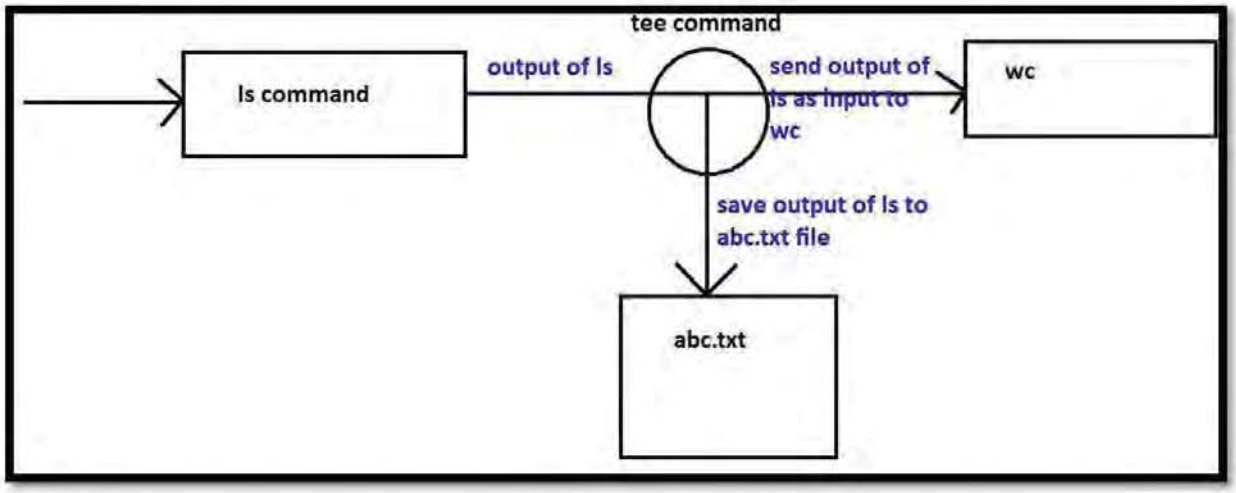
In piping, if we want to save the output of one command to a file and if we want to pass that output as input to next command simultaneously, then we should go for tee command.

tee command is just like T-Junction or T-Pipe. It will take one input but provides two outputs.



<u>Eg 1:</u> To save the output of ls command to a file and to display to the terminal simultaneously.

$ ls –l → It will display to the terminal
$ ls -l > abc.txt → It will save to the abc.txt but won't display to the terminal.
$ ls -l | tee abc.txt



<u>Eg 2:</u> To save the output of ls command to a file and send that output as input to wc command

$ ls -l | tee output.txt | wc -l

xargs Command:

Q1) Display the Output of Date Command by using echo Command with Piping Concept?

$ date | echo → It won't work because the output of date command is stream, but echo command will accept only command line arguments but not stream.

$ date | xargs echo → xargs command will convert the output stream of date command into command line arguments and these arguments will be passed as input to echo command.

Hence the job of xargs command is to convert output stream into command line arguments

Eg 1: Assume input.txt contains file names. Each file contains some data.
Read file names from the input.txt, write total content to output.txt and display the total number of lines present in output.txt.

$ cat input.txt | xargs cat | tee output.txt | wc -l

Eg 2: Assume input.txt contains file names. Read file names from the input.txt and remove all these files.

$ cat input.txt | xargs rm

Assignment:

list out all contents of /dev folder and save to file1.txt.
list out all contents of /bin folder and save to file2.txt.c

Write a single pipeline for the following requirement:
read content of file1 and file2, save to fil3.txt. By using sort command reverse contents of file3.txt and save to sorted.txt.

$ ls /dev > file1.txt
$ ls /bin > file2.txt
$ cat file1.txt file2.txt | tee file3.txt | sort -r > sorted.txt

Q2) What is Piping?

A) A way of connecting commands together.
B) A way of passing data from the standard output of one command to the standard input of another command.

Ans: A,B

Q3) How to Pipe Standard Output of X Command to the Standard Input of Y Command?

A) X > Y
B) X >> Y
C) X < Y
D) X | Y

Ans: D

Q4) How we can use tee Command when Piping together Commands A, B and C to save Output of B Command to results.txt.

A | B | tee results.txt | C

Q5) How to Pipe Data from Command A to Command B, but B won't accept Standard Input and accepts only Command Line Arguments?

A | xargs B

Topic-15: How to Use Multiple Commands in a Single Line

We can execute multiple independent commands in a single line by using the following two ways

1st Way: By using semicolon (;)

cmd1;cmd2;cmd3;.....;cmdn

First cmd1 will be executed and then cmd2 followed by rest of the commands.
If any command fails in the middle, still rest of the commands will be executed.

2nd Way: By using &&

cmd1 && cmd2 && cmd3 &&..... && cmdn

First cmd1 will be executed and then cmd2 followed by rest of the commands.
If any command fails in the middle, then rest of the commands won't be executed.

Eg:
create a directory dir1
create files a.txt,b.txt,c.txt in that dir1
write current system date and time to a.txt
write current month calendar to b.txt

```
mkdir dir1 ; touch dir1/{a,b,c}.txt ; date > dir1/a.txt ; cal > dir1/b.txt
mkdir dir1 && touch dir1/{a,b,c}.txt && date > dir1/a.txt && cal > dir1/b.txt

mkdir dir1 ; touch dir1/{a,b,c}.txt ; Date > dir1/a.txt ; cal > dir1/b.txt
```
Here 3rd command fails, but still 4th command will be executed.

```
mkdir dir1 && touch dir1/{a,b,c}.txt && Date > dir1/a.txt && cal > dir1/b.txt
```
Here 3rd command fails, and hence 4th command won't be executed.

Topic-16: Regular Expressions and Wildcard Characters

If we want to represent a group of strings according to a particular pattern, then we should go for regular expressions.

By using wildcard characters, we can build regular expressions.

A wildcard character can be used as a substitute for required sequence of characters in the regular expression.

1) → Represents zero or more characters
2) ? → Represents only one character
3) []→ Range of characters
4) [abc] → Either a or b or c
5) [!abc] →Any character except a,b and c
6) [a-z] → Any lower case alphabet symbol
7) [A-Z] → Any upper case alphabet symbol
8) [a-zA-Z] → Any alphabet symbol
9) [0-9] → Any digit from 0 to 9
10) [a-zA-Z0-9] → Any alphanumeric character
11) [!a-zA-Z0-9] → Except alpha numeric character (i.e special symbol)
12) [[:lower:]] → Any lower case alphabet symbol
13) [[:upper:]] → Any upper case alphabet symbol
14) [[:alpha:]] → Any alphabet symbol
15) [[:digit:]] → Any digit from 0 to 9
16) [[:alnum:]] → Any alpha numeric character
17) [![:digit:]] → Any character except digit
18) {} → List of files with comma separator

1) To list out all files present in current working directory ➔ $ ls *

2) To list out all files with some extension ➔ $ ls *.*

3) To list out all files starts with a ➔ $ ls a*

4) To list out all files starts with a and ends with t ➔ $ ls a*t

5) To list out all .java files ➔ $ ls *.java

6) To list out all files where file name contains only 2 characters and first character should be 'a' ➔ $ ls a?

7) To list out all files where file name contains only 3 characters ➔ $ ls ???

8) To list out all files where file name contains atleast 3 characters ➔ $ ls ???*

9) To list out all files where file name starts with a or b or c ➔ $ ls [abc]*

10) To list out all files where file name should not starts with a, b and c ➔ $ ls [!abc]*

11) To list out all files starts with lower case aphabet symbol
    $ ls [a-z]* OR $ls [[:lower:]]*

12) To list out all files starts with upper case aphabet symbol
    $ ls [A-Z]* OR $ls [[:upper:]]*

13) To list out all files starts with digit.
    $ ls [0-9]* OR $ls [[:digit:]]*

14) To list out all files where first letter should be upper case alphabet symbol, second letter should be digit and third letter should be lower case alphabet symbol.
    $ ls [[:upper:]][[:digit:]][[:lower:]]

15) To list out all files starts with special symbol
    $ls [![:alnum:]]*

16) To list out all files with .java and .py extension
    $ ls {*.java, *.py}

Note: We can use these wildcard characters with the following commands also.
cp, mv, rm

17) To copy all files starts with digit to dir1 directory.
    $cp [[:digit:]]* dir1
    $cp [0-9]* dir1

18) To move all files starts with alphabet symbol and with .txt extension to dir2 directory?
    $mv [[:alpha:]]*.txt dir2

19) Remove all files starts with a or b or c and ends with e or t.
    $rm [abc]*[et]

Q1) Which of the following Command will List all Files that has exactly 3 Characters Present in Current Working Directory?

A) ls ***
B) ls ???
C) ls !!!
D) ls &&&

Ans: B

Q2) Which of the following Commands will Copy all Files that Ends with .pdf to dir1?

A) cp ?.pdf dir1
B) cp .pdf* dir1
C) cp *.pdf dir1

Ans: C

Q3) Which of the following Command will move all the Files that begins with Letter a and Ends with Letter n to dir1?

$mv a*n dir1

Q4) Which of the following Commands will display Contents of all Files that begins with a Digit and Ends with Letter a OR e OR i OR o OR u?

$cat [[:digit:]]*[aeiou]
$cat [0-9]*[aeiou]

Q5) Which of the following Commands will List all Files that begins with a Lower Case Alphabet Symbol and has a Letter d in the 3rd Character Position, and Ends with an Upper Case Letter?

$ls [[:lower:]]?d*[[:upper:]]

Q6) Which of the following Command will List all .jpg Files Present in Pictures Directory?

$ls /home/durgasoft/Pictures/*.jpg
$ls ~/Pictures/*.jpg

Q7) Which of the following Regular Expressions will Match the File named with demoA.txt ?

A) *
B) demo?.txt
C) demo*
D) *.txt
E) demo[A-Z].txt
F) All of these

Q8) Which of the following Regular Expressions can Match the Files?

student_reportA.pdf
student_reportB.pdf
student_reportC.pdf

A) ??.pdf
B) report*.pdf
C) *[A-Z].pdf
D) student*.pdf

Ans: C, D

Topic-17: Command Aliasing

Alias means other alternative name or nickname.
We can give our own more convinient nicknames for unix commands. This concept is called command aliasing.

Note: we can use type command, to check whether the command is already available or not.

How to Create Alias Names?

$ alias nickname='original command'
$ alias nickname="original command"

After aliasname space is not allowed. Hence the following are invalid
alias nickname ='original command'
alias nickname= 'original command'
alias nickname = 'original command'

How to List all available Aliases?

By using alias command without any arguments
$alias

How to Remove Alias Names?

By using unalias command.
$unalias alias_name

Where we can use aliasing:

1. If any lengthy command repeatedly required, then we can create shortcut alias name and we can use that short alias name every time.

alias d20f='mkdir dir1;touch dir1/file{1..20}.txt'

$d20f

Eg: To list out all files present in current working directory, save this data to output.txt and display the number of lines to the terminal. Define alias name 'current' for this total activity.

```
alias current='ls -l | tee output.txt | wc -l'
```

2. To use other operating system (like windows) commands directly in linux

```
alias cls='clear'
alias rename='mv'
```

3. To handle typing mistakes

```
alias grpe='grep'
```

4. To handle language barriers:
In Germany datum means date.

```
alias datum='date'
```

How to persist aliases permanently?

Whatever aliases we created, are bydefault available only in the current session. Once we close the terminal, all aliases will be lost.

But we can make our created aliases permanently in our system by using the following 2 ways:

1st Way:

We have to define our aliases in .bashrc file present in our home directory.

```
gedit .bashrc
```

Add the following lines in that file.

```
# myown aliases
alias cls='clear'
alias ddd='date;date;date'
```

Note: To reflect these aliases, compulsory we have to close and open terminal.

2nd Way:

Instead of editing .bashrc file, we can create our own file to maintain our defined aliases. The name of the file should be .bash_aliases and should be present in home directory.

.bash_aliases

```
alias ccc='cal;cal;cal'
alias ct='cal;date'
```

Note: To reflect these aliases, compulsory we have to close and open terminal.

Q1) What is the Purpose of Alias Comamnd?

To list out all available aliases and to create new alias.

Q2) How to Use Unalias Command?

unalias alias_name → To remove a particular alias
unalias –a → To remove all aliases

Q3) To Persist Aliases, in which File we have to define Aliases?

.bashrc OR .bash_aliases

Q4) Which of the following is Valid Way of creating Alias?

A) alias rename ="mv"
B) alias rename= "mv"
C) alias rename = "mv"
D) alias rename="mv"

Ans: D

While creating aliases, we should not give space.

Topic-18: Locate and Find Commands

locate Command:

We can use locate command to locate files and directories in our system.

Internally locate command will search in the database for the required files and directories and returns the results.

As locate command is searching in the database instead of filesystem, performance will be improved.

1. To locate all .jpg files
   $ locate *.jpg

To ignore case:

By default locate command will consider case. If we want to ignore case, we have to use -i option.

$locate -i *.jpg

Note: In ubuntu -i option is not working

We can limit the number of lines in the result by using --limit option.
$ locate --limit 5 *.conf

/etc/adduser.conf
/etc/apg.conf
/etc/appstream.conf
/etc/brltty.conf
/etc/ca-certificates.conf

It display only 5 lines.

Before display results, to check whether the file exists or not , we have to use -e option or --existing option.

$locate --existing *jpg

Reason:
locate command using database to find results. This database will be updated only once per day bydefault. After updating database, some files may be deleted. Hence before printing results, to check whether files are existing or not, we have to use -e or --existing option.
Before displaying results, to check whether symbolic links pointing to original files or not, we have to use -L or --follow option. i.e if we use --follow option, broken symbolic links won't be displayed in the output.

```
$locate --follow *.txt
```

Note: We can use all these options together

```
$locate --existing --follow -i --limit 5 *.conf | wc -l
```

observe the difference in results:

```
$locate *.conf | wc -l
$locate -i  *.conf | wc -l
$locate --existing  -i  5 *.conf | wc -l
$locate --existing --follow -i  *.conf | wc -l
$locate --existing --follow -i --limit 10 *.conf | wc -l
```

How to Update Database?

We can see the database by using locate command with -S option.

```
$ locate -S
Database /var/lib/mlocate/mlocate.db:
        39,527 directories
        3,69,078 files
        2,42,42,520 bytes in file names
        93,40,974 bytes used to store database
```

This database will be updated only once per day. If we are creating or removing files and directories, to reflect these changes we have to update database explicitly by using updatedb command. But admin privileges must be required.

```
$ sudo updatedb
[sudo] password for durgasoft:
```

Q1) Which of the following are Valid Options to locate Command?

A) -i
B) --limit
C) -L
D) -e
E) --existing
F) --follow
G) All of these

And: G

Q2) Locate Command internally Uses Database to find Results?

True
False

Ans: True

Q3) We created a File called demo.txt. But it has been 2 Hours since locate Database has been updated. Is locate Command able to find this demo.txt?

A) Yes
B) No

Ans: No

Q4) How to Update Database which is used by locate Command?

A) update locatedb
B) updatedb
C) sudo updatedb

Ans: C

find Command:

We can use find command to find files and directories present in our system.
It provides more search options when compared with locate command like

1) Search only files
2) Search only directories
3) Search by name
4) Search by size
5) We can use search results automatically for some other commands
6) We can restrict maxdepth
   etc

1. $find

It will find all files and directories in current working directory and below in linux file system. This is the default behaviour.

2. We can find all files and directories in the specified directory and below.
$ find /dev
$ find /etc

3. maxdepth Option:

usually find command will search in all depth levels. But we can specify the required depth level by using maxdepth option.

```
Desktop
    |-file1.txt
    |-level_1_dir
         |-file2.txt
         |-levlel_2_dir
           |-file3.txt
           |-level_3_dir
                |-file4.txt
                |-level_4_dir
                   |-file5.txt
```

$mkdir -p level_1_dir/level_2_dir/level_3_dir/level_4_dir
$touch file1.txt level_1_dir/file2.txt level_1_dir/level_2_dir/file3.txt
level_1_dir/level_2_dir/level_3_dir/file4.txt
level_1_dir/level_2_dir/level_3_dir/level_4_dir/file5.txt

Observe the difference in results by executing the following commands:

1. $ find . -maxdepth 1
2. $ find . -maxdepth 2
3. $ find . -maxdepth 3
4. $ find . -maxdepth 4
5. $ find . -maxdepth 100

Note:

1. For maxdepth option we should use singl - but not double --
   -maxdepth → Valid
   --maxdepth → Invalid
2. find command will find hidden files and directories also.

Find by Type:

We can find only files or only directories by using type option.
-type f → means only files
-type d → means only directories

```
durgasoft@durgasoft-VirtualBox:~/Desktop$ find -type f
./level_1_dir/file2.txt
./level_1_dir/level_2_dir/file3.txt
./level_1_dir/level_2_dir/level_3_dir/level_4_dir/file5.txt
./level_1_dir/level_2_dir/level_3_dir/file4.txt
./file1.txt
./.securefile1.txt

durgasoft@durgasoft-VirtualBox:~/Desktop$ find -type d
.
./.db_info
./level_1_dir
./level_1_dir/level_2_dir
./level_1_dir/level_2_dir/level_3_dir
./level_1_dir/level_2_dir/level_3_dir/level_4_dir
```

Note: We can use these options simultaneously, but we should use first -maxdepth and then -type.

$find -type f -maxdepth 2 → Generates warning
$find -maxdepth 2 -type f → No warnings

Find by Name:

We can find files and directories by name by using -name option.

```
$ touch {A..D}.txt
$ touch {A,B}{A,B}.txt

$ find . -name 'A.txt'
$ find . -name '?.txt'
$ find . -name '??.txt'
$ find . -name '*.txt'
$ find . -maxdepth 2 -name '*.txt'
```

If we want to ignore case then we should use -iname option.

```
$ find -iname 'a.txt'
```

Find Files by Size:

We should use -size option.

+ symbol means greater than (over)

- symbol means less than

1. To list out all file names where size is over 200kb

$ find / -type f -size +200k This command required root privileges

$ sudo find / -type f -size +200k | wc -l

2. To list out all file names where size is over 200kb but less than 4MB.

$ find / -type f -size +200k -size -4M | wc -l

3. To list out all file names where file size is less 200kb or more than 4MB.

$ find / -type f -size -200k -o -size +4M | wc -l

-o means or

Q1) What is the Output of the following Command?

$ find / -type f -size -200k -size +4M | wc -l

The answer should be: 0

Note:

+n for greater than n

-n for less than n

n for exactly n

-empty File is empty and is either a regular file or a directory

Q2) How to Use Search Results of Find Command?

We can perform any operation (like cp, mv, rm etc) on the results of find command.

For this we have to use -exec option.

-exec means execution.

Q3) To Copy all Files Present in /etc Folder where File Size is < 2KB to dir1 Directory Present in the Desktop?

$ find /etc -type f -size -2k -exec cp {} dir1 \;

Before performing required copy operation if we want confirmation then we should use -ok option instead of -exec.

$ find /etc -type f -size -2k -ok cp {} dir1 \;

durgasoft@durgasoft-VirtualBox:~/Desktop$ find /etc -type f -size -2k -ok cp {} dir1 \;

< cp ... /etc/magic.mime > ? y
< cp ... /etc/fwupd/uefi.conf > ? y
< cp ... /etc/fwupd/daemon.conf > ? y

Magic Assignment:

```
$ mkdir magic
$ mkdir magic/dir{1..100}
$ touch magic/dir{1..100}/file{1..100}.txt

$ mkdir magic;mkdir magic/dir{1..100};touch magic/dir{1..100}/file{1..100}.txt

$ touch magic/dir$(shuf -i 1-100 -n 1)/sunny.txt

$ find magic -type f -name 'sunny.txt'
$ find magic -type f -name 'sunny.txt' -exec mv {} ~/Desktop \;
$ find magic -type f -name '*.txt' -exec rm {} \;
```

Q4) The Find Command Uses a Database to Search Files and Directories

A) True
B) False

Ans: B

Q5) To find Files and Directories inside /dev Folder and Limit its Search to only 2 Levels of Deep?

A) find -start /dev -depth 2
B) find /dev -depth 2
C) find /dev -maxdepth 2

Ans: C

Q6) To find only Directories inside /dev Folder and Limit its Search to only 2 Levels of Deep?

A) find /dev -maxdepth 2 -type f
B) find /dev -maxdepth 2 -type d
C) find /dev -type d -maxdepth 2

Ans: B

Q7) To find only Files Starts from Root Directory (/) where File Name Ends with .txt?

A) find / '*.txt'
B) find / -type d -name '*.txt'
C) find / -type f -name '*.txt'

Ans: C

Q8) To find all Files and Directories inside /dev Directory upto Maximum of 3 Levels Deep and Size is Greater than 200 Kilo Bytes?

A) find / -maxdepth 3 -size 200k
B) find /dev -maxdepth 3 -size 200k
C) find /dev -maxdepth 3 -size -200k
D) find /dev -maxdepth 3 -size +200k

Ans: D

Q9) Which of the following Command will find all Files below our Home Directory where File Size is Greater than 3 Mega Bytes and Remove all those Files?

A) find ~ -type f -size +3M -exec rm {} \;
B) find ~ -type f -size -3M -exec rm {}
C) find ~ -type f -size -3M -exec rm {} \;
D) find ~ -type f -size +3M rm {} \;

Ans: A

Difference between find and locate Commands

find	locate
1. It searches directly in the file system for the required files and directories.	1. It searches in the database for the required files and directories.
2. find command has number of options and easy to customize based on our requirement.	2. locate command has very few options.
3. We can find files based on name, type,size,depth,age, user,group ,permissions etc	3. We can find files only based on name and permissions.
4. We can reduce the depth of search.	3. We cannot reduce depth of search.
5. find command will produce accurate results as it searches directly in the file system.	5. locate command won't produce accurate results as it searches in the database which will be updated only once per day bydefault.
6. find command won't produce deleted files in search results as it searches directly in the file system.	6. locate command may produce deleted files in search results as it searches in the database which may not be updated.
7. There is a way to use search results by using -exec option.	7. There is no direct way to use search results.
8. find command operates slowly.	8. locate command operates fastly.