1.Illustrarte these commands

1. **More**

* more command is used to view the text files in the command prompt
* displaying one screen at a time in case the file is large (For example log files).
* The more command also allows the user do scroll up and down through the page.
* The syntax along with options and command is as follows.

more [-options] [-num] [+/pattern] [+linenum] [file\_name]

* ***[-options]****: change the way the file is displayed. (-d, -l, -f, -p, -c, -s, -u)*
* ***[-num]****: type the number of lines that you want to display per screen.*
* ***[+/pattern]****: replace the pattern with any string that you want to find in the text file.*
* ***[+linenum]****: use the line number from where you want to start displaying the text content.*
* ***[file\_name]****: name of the file containing the text that you want to display on the screen.*

1. **Less**

* used to read contents of text file one page(one screen) per time.
* It has faster access because if file is large, it don’t access complete file, but access it page by page.
* Syntax :

less filename

* For Example : If you want to read contents of dmesg command, it’s better to use it with less command: dmesg | less
* less -N /var/log/auth.log

1. **Head**

* print the top N number of data of the given input
* By default, it prints the first 10 lines of the specified files.
* Syntax :

head [OPTION]... [FILE]...

Options : -n : lines (mandatory )

-b: bytes

-q: quiet ( used when more than one file is given )

-v :verbose(data from the specified file is always preceded by its file name.)

1. **Tail**

* print the last N number of data of the given input.
* By default it prints the last 10 lines of the specified files.
* Syntax :

tail [OPTION]... [FILE]...

Options : -n : lines (mandatory ) without symbolizing ‘n’ character but ‘-‘ sign is mandatory.

Tail**+’** option which is not present in the head command. With this option tail command prints the data starting from specified line number of the file instead of end.

-b: bytes

-q: quiet ( used when more than one file is given )

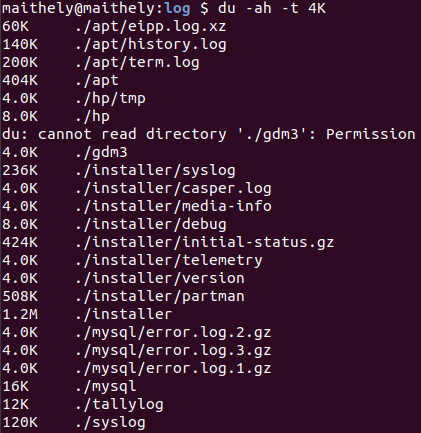
-v :verbose(data from the specified file is always preceded by its file name.)

-f:follow ( This option shows the last ten lines of a file and will update when new lines are added)

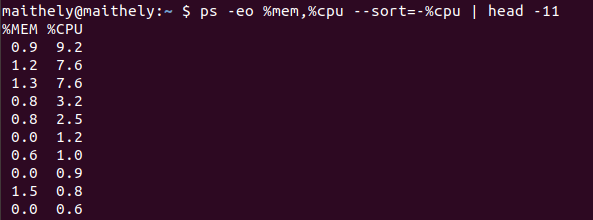
2. Command to check top 10 file / process which using

1. Disk

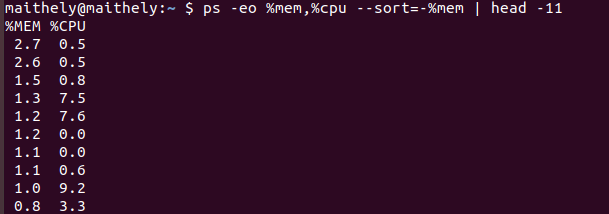
Sudo du -ah | sort -n -r | head -10



1. Cpu



1. Memory

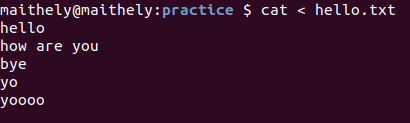


3.Illustrate and Difference in Stdin,Stdout,Stderr with examples to differentiate using commands.

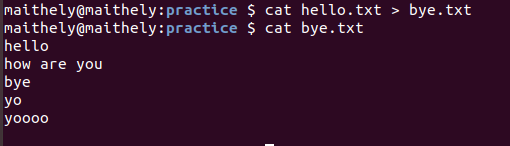
* stdin, stdout, and stderr are three data streams created when you launch a Linux command.
* You can use them to tell if your scripts are being piped or redirected.
* **stdin** (0<)is the standard input stream. This accepts text as its input

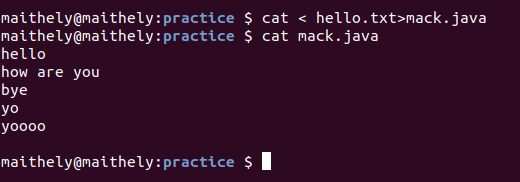
A standard input device, which is usually the keyboard, but Linux also allows you take standard input from a file.

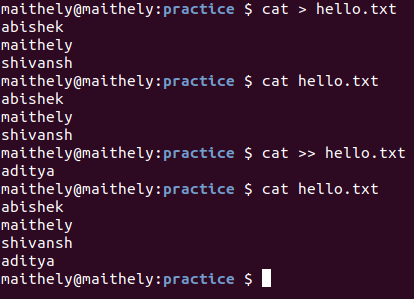
Since, at the time of creating a process, one standard input is associated with the process.



* Text output from the command to the shell is delivered via the **stdout(1>)** (standard out) stream.



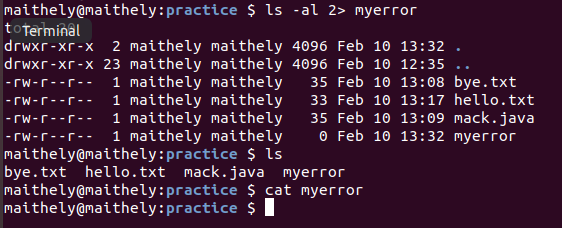




In the above example in first case it replaces and we enter data in it and comes out of the prompt when pressind ctrl=d

Where as in second case we will get the output as we are appending in the file

* Error messages from the command are sent through the **stderr** (standard error) stream.



find / -name "\*" -print 2> /dev/null to put any errors that the find command generates into **/dev/null**, because I'm not interested in them.

Every command could send it's output to one of two places:

a) it could be valid output or

b) it could be an error message.

4.Differnece in locate , find , xargs

* **Locate** looks for files from the internal linux database.store indexes in local db. Db is file system and update it . it searches in db and not in whole file system
* **find** looks for files in realtime.
* **Xargs** builds an execution pipeline for standard input. It is mostly used with find command. It is faster than --exec.

5.Diffrence in which, where is

* ***which:*** “which” search for executables in the directories specified by the environment variable PATH. And if found out, the full pathname of this executable will be printed.
* ***whereis: “****whereis****”*** search for executables, source files, and manual pages using a database built by system automatically.

6.Delete files from a range of time

find . -newerct "Feb 1 2020" ! -newerct "Feb 10 2020" | xargs rm

7.List all files with Extension .sh

Find / -name “\*.sh”

Top , aux ,