Shell pipelines:

- a **pipeline** is a set of processes chained by their standard streams, so that the output of each process (*stdout*) feeds directly as input (*stdin*) to the next one.
-
- `ps -ax | grep Finder`
-
- Use the `ps` command to get a list of processes running on the system, and pass the list to `grep` to search for lines containing "Finder". (Usually, it'll find two: the Finder, and the processes executing `grep Finder`.)

```
$ cat tecmint.txt
```

- C at command shows the contents of the file and the way it look.

```
Ex: grep root /etc/passwd
```

- Prints lines matching a pattern
- Displayes the lines from /etc/passwrd that contain the string root

```
awk '{print $3 "\t" $4}' marks.txt
```

- Prints culumns 3 and 4 from a text file marks.txt

```
awk '{print $4 "\t" $3}' marks.txt
```

-Prints culumns 3 and 4 from a text file marks.txt

```
$ sed -e '1,10d' /etc/services | more
```

- sed reads the input file, does the instructions (in this case ‘1,10d’) and outputs results
- sed never alters the original file
- in this example lines 1 to 10 are deleted from file “services” and the rest printed on stdout

```
$ tr [OPTION] SET1 [SET2]
```

- tr stands for translate.
- If both the SET1 and SET2 are specified and ‘-d’ OPTION is not specified, then tr command will replace each characters in SET1 with each character in same position in SET2.

```
tr abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ

thegeekstuff

THEGEEKSTUFF
```

- The following tr command is used to convert the lower case to upper case

```
$ sort tecmint.txt
```

- Sort commend sorts the contents of a text file and prints them out on stdout.
- The above command don’t actually sort the contents of text file but only show the sorted output on terminal.

```
$ sort tecmint.txt > sorted.txt

$ cat sorted.txt

sort -r tecmint.txt > reversesorted.txt
```

- The first line sorts the contents of the file and saves them in the file called sorted.txt
- Second line just prints the content of sorted.txt
- Third line reversally sorts contents from tecmint.txt and saves them in reversesorted.txt

```
$ cat test
aa
aa
bb
bb
bb
xx
$ uniq test
aa
bb
xx
```

- For example, when uniq command is run without any option, it removes duplicate lines and displays unique lines as shown below.

```
- $ uniq -c test

-       2 aa

-       3 bb

-       1 xx
```

- This option is to count occurrence of lines in file.

```
$ uniq -d test

aa

bb
```

- This option is to print only duplicate repeated lines in file. As you see below, this didn’t display the line “xx”, as it is not duplicate in the test file.

```
$ uniq -u test

Xx
```

- This option is to print only unique lines in file.

```
$ x=10
```

- echo its value:

```
$ echo The value of variable x = $x


The value of variable x = 10
```