GETTING STARTED
WITH
BASIC FILTERS

WC:

- It is used to print newline, word, and byte counts for each file.

SYNTAX: **#wc [options] <file-name>**

→ Print lines, words, and byte counts of a file:

```
#wc aws
```

→ Print lines, words, and byte counts of multiple files:

```
#wc aws azure
```

→ Counting lines only in a file:

```
#wc -l aws
```

→ Counting words only in a file:

```
#wc -w aws
```

→ Counting bytes / characters only in a file:

```
#wc -c aws
```

NOTE: Create a file with name **“aws”** and add some content.

HEAD:

- Print the first 10 lines of each FILE to standard output.
- If more than one file name is provided then data from each file is preceded by its file name.

SYNTAX: **#head [options] <file-name>**

Print top 10 lines of a file:

```
#head aws
```

Print top 3 lines only:

```
#head -3 aws
```

TAIL:

- Print the last 10 lines of each FILE to standard output.
- If more than one file name is provided then data from each file is precedes by its file name.

SYNTAX: **#tail [options] <filename>**

→ To print last 10 lines of a file:

```
#tail aws
```

→ To print last 5 lines only:

```
#tail -5 aws
```

→ Print last ten lines of a file and will update when new lines are added. File open Continuously:

```
#tail -f logfilename
```

MORE:

- It is used to view file contents one screen at a time.

SYNTAX: **#more [options] <file>**

While viewing the text file use these controls:

Enter key	**:** to scroll down line by line.
Space bar	**:** To go to the next page.
b key	**:** To go to back one page.
q key	**:** To quit

→ Display content one screen at a time:

```
#more aws
```

LESS:

- It is used to view contents of a file one page (one screen) at a time.

```
#less aws
```

NOTE: Same as like more command.

ECHO:

- Echo is a built-in command that allows users to display lines of text or strings that are passed as arguments.
- It is commonly used in shell scripts and batch files to output status text to the screen or a file.

SYNTAX: **#echo [options] <string>**

→ Print a statement:

#echo "This is a RAJU…"

→ Using command substitute operator:

#echo "Today date is: `date`**"**

#echo "Current login user name is: `whoami`**"**

CUT:

- cutting out the sections from each line of files and writing the result to standard output.
- It can be used to cut parts of a line by byte position, character and field.
- Basically, the cut command slices a line and extracts the text.

SYNTAX: **#cut [options] <file-name>**

→ Cut a third field of a file:

#cut -f 3 filename

→ Cut a third and fifth field of a file:

#cut -f 3,5 filename

→ Cut a third field to fifth field in a file:

#cut -f 3-5 filename

→ Cut each field first character of a file:

#cut -c 1 filename

→ Cut each field third and fifth character:

#cut -c 3,5 filename

PASTE:

- It is used to join files horizontally (parallel merging) by outputting lines consisting of lines from each file specified, separated by tab as delimiter, to the standard output.
- Simply, merge lines of files.

SYNTAX: **#paste [options] <file>**

```
#cat states
Ts
Mah
Kar

#cat capitals
Hyd
Mum
Ban
```

→ Merge lines of files:

```
#paste states capitals
```

→ Merge lines with delimiter "@":

```
#paste -d "@" states capitals
```

→ Merge lines with delimiter "-":

```
#paste -d "-" states capitals
```

→ Merge lines with delimiter ":":

```
#paste -d ":" states capitals
```

→ Merge lines with delimiter "&" and output redirect to a file:

```
#paste -d "&" states capitals >file1

#cat file1
```

TR:

- It is used to translate or delete characters.

SYNTAX:

```
#tr [options] <file-name>

#cat devops

This is   a    sample    devops    text file.
```

→ Translating characters lower to upper in a file:

```
#tr “a-z” “A-Z” <devops
```

→ Translating characters form upper to lower:

```
#tr “A-Z” “a-z” <devops
```

→ To squeeze a sequence of repetitive characters:

```
#tr -s “ ” <devops
```

→ To save a output:

```
#tr -s “ ” <devops >devops1

#cat devops1
```

DIFF:

- It is used to compare files **line by line.**

SYNTAX:

```
#diff [options] file1 file2

#cat file1
Unix
Linux

#cat file2
Unix
Lanux

#Cat file3
Unix
Linux
```

→ Differentiate two files:

#diff file1 file2

→ There is no output without difference:

#diff file1 file3

CMP:

- Used to compare the two files byte by byte.

→ compare two files:

#cmp file1 file2

→ There is no output without difference:

#cmp file1 file3

SORT:

- Used to sort a file, arranging the records in a particular order. By default, the sort command sorts file assuming the contents are ASCII.

SYNTAX: **#sort [options] File**

#cat sample

```
Unix
Linux
Linux
Java
Linux
Linux
Php
Linux—Linux---LINUX
```

→ To sorting a file ascending order:

#sort sample

→ To descending / reverse order:

#sort -r sample

→ Eliminating duplicated / repeated lines:

```
#sort -u sample
```

→ Displaying with numeric order:

```
#sort -n sample
```

UNIQ:

- It reports or filters out the repeated lines in a file.
- It helps to detect the adjacent duplicate lines and also deletes the duplicate lines.

→ Print to eliminate adjacent line:

```
#uniq sample
```

→ To delete duplicated lines:

```
#uniq -d sample
```

→ To print only the uniq lines:

```
#uniq -u sample
```