grep

grep stands for
globally search a regular expression and print it
global regular expression print.
global regular expression parser

We can use grep command to search the given pattern in a single or multiple files.

Syntax: grep [options] pattern [files]
It prints all matched lines.

Option:
-c: This prints only a count of the lines that match a pattern
-h: Display the matched lines, but do not display the filenames.
-i: Ignores, case for matching (uppercase lowercase)
-l: Displays list of matched filenames only.
-L: Display the list of non-matched filenames only.
-n: Display the matched lines and their line numbers.
-v: This prints out all the lines that do not matches the pattern
-e exp: Specifies expression with this option. Can be used multiple times.
-f file: Takes patterns from file, one per line.
-w: Match whole word
-o: Print only matched parts of the line.
-q: Do not print anything to standard output.
-r: Recursive for a Directory
-A n: Prints searched line and n lines after the result.
-B n: Prints searched line and n lines before the result.
-C n: Prints searched line and n lines after and before the result.

Q1: Line have apple or orange but not banana
Q2: Mobile Number
Q3: Number in a line

Egrep

grep with -E option. It means we have to write extended regular expressions.

sort

SORT command is used to sort a file, arranging the records in a particular order. By default, the sort command sorts file assuming the contents are ASCII. Using options in the sort command can also be used to sort numerically.

SORT command sorts the contents of a text file, line by line.
sort is a standard command-line program that prints the lines of its input or concatenation of all files listed in its argument list in sorted order.
The sort command is a command-line utility for sorting lines of text files. It supports sorting alphabetically, in reverse order, by number, by month, and can also remove duplicates.
The sort command can also sort by items not at the beginning of the line, ignore case sensitivity, and return whether a file is sorted or not. Sorting is done based on one or more sort keys extracted from each line of input.
By default, the entire input is taken as the sort key. Blank space is the default field separator.
The sort command follows these features as stated below:

Lines starting with a number will appear before lines starting with a letter.
Lines starting with an uppercase letter will appear after lines starting with the same letter in lowercase.

Syntax: sort [options] filename

Options:

1) sort a.txt → Sort in an increasing order.
2) sort -r a.txt → Sort in reverse order.
3) sort -o filename.txt inputfile.txt → Write output to a new File.
4) sort -n a.txt → Sort on the basis of numbers(Counting).
5) sort -u a.txt → Remove duplicate Lines.
6) sort -k2 a.txt →Sort on the basis of mentioned column Number.Feild Separator is ‘ ‘.
7) -t → Change Field Separator.
8) -c →Checks whether the file is in sorted order or not.
9) -M → Sort by Month.
10) -h → Sorts human readable Numbers i.e. 5K,2G….

Note:

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

```
11
2222222
```

uniq

We can use the uniq command to display unique content in the file.
The uniq command in Linux is a command-line utility that reports or filters out the repeated lines in a file.
In simple words, uniq is the tool that helps to detect the adjacent duplicate lines and also deletes the duplicate lines. uniq filters out the adjacent matching lines from the input file(that is required as an argument) and writes the filtered data to the output file.
But to use the uniq command the file should be sorted, otherwise it won't work properly.
Syntax: uniq [options] file

Options:
1) -d → To display only duplicate lines
2) -c → To display number of occurrences of each line
3) -i → Ignore case while comparing
4) -u → To display only unique lines i.e the lines which are not duplicated
5) -f N / -skip-fields(N) → It allows you to skip N fields.
6) -s N / -skip-chars(N) → It doesn't compare the first N characters of each line while determining uniqueness.
7) -w N / -check-chars(N) → It only compares N characters in a line.
8) -z → If we don't want a newline after output.

Word Count (WC)

We can use wc command to count the number of lines, words and characters present in the given file.
Syntax: wc [Option] filenames
Output: no_of_lines no_of_words no_of_characters filename

We can use the following options with wc Command

1) -l → To print only number of lines
2) -w → To print only number of words
3) -c → To print only number of bytes
4) -m → To print only the number of characters.
5) -lw → To print only number of lines and words
6) -lc → To print only number of lines and characters
7) -wc → To print only number of words and characters
8) -L → To print the number of characters present in the Longest Line.

cmp

cmp command in Linux/UNIX is used to compare the two files byte by byte and helps you to find out whether the two files are identical or not.

When cmp is used for comparison between two files, it reports the location of the first mismatch to the screen if difference is found and if no difference is found i.e the files compared are identical.

cmp displays no message and simply returns the prompt if the files compared are identical.

Syntax: cmp [option] file1 file2

```
$ 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.

Options:

1) -b → If you want, cmp displays the differing bytes in the output when used with -b option.
2) -i [Skip / Skip1:Skip2] → If you want to skip starting bytes of both files.
3) -l → verbose i.e. print the byte difference and value for all different bytes.
4) -s → suppresses normal output. Print only identical or not using an exit code. 0 for identical, 1 for different, 2 for error.

5) -n N → limit number of bytes to compare to first N.

diff

diff stands for difference. This command is used to display the differences in the files by comparing the files line by line. Unlike its fellow members, cmp and comm, it tells us which lines in one file have is to be changed to make the two files identical.

The important thing to remember is that diff uses certain special symbols and instructions that are required to make two files identical. It tells you the instructions on how to change the first file to make it match the second file.

Special Symbols:
c = change
d = delete
a = add
< file1
> file2

Option
1) -c → Context Mode
2) -u → unified Mode
3) -q → shows messages when files are different.
4) -s → shows message when files are same identical
5) -y → shows comparison line by line (parallel comparison)
6) -i → ignore the uppercase and lowercase. Make command Case insensitive
7) -r → recursively compare any Subdirectories found.
8) Man Diff for more

sdiff :-

sdiff command in linux is used to compare two files and then writes the results to standard output in a side-by-side format. It displays each line of the two files with a series of spaces between them if the lines are identical. It displays greater than sign if the line only exists in the file specified by the File2 parameter, and a | (vertical bar) for lines that are different.
Syntax: sdiff [-l | -s] [-o OutFile] [-w Number] File1 File2
Options:
1) -l → it displays only left side when they are identical
2) -s → It does not display identical lines
3) -w N → It sets the width of the output file. Default is 130. Max: 2048, Min: 20
4) -o OutputFile → Stores the output in the output file.
5) -i → Ignore case

comm

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

Syntax: 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.

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

--File Archiving and Compression commands--

It is a very common requirement to pack and compress a group of files. The main advantages are:

1) It improves memory utilization
2) Transportation will become very easy
3) It reduces download times
4) etc.

This process involves the following 2 activities:
1) Creation of Archive file
2) Apply compression algorithms on that archive file

Creation of Archive File

tar

We can group multiple files and directories into a single archive file by using tar command.

Syntax: tar [options] [archive-file] [files or directory to be archived]
Options:

1) -c : Creates Archive
2) -x : Extract the archive
3) -f : creates archive with given filename
4) -t : displays or lists files in archive file
5) -v : Displays Verbose Information

6) -z : zip, tells tar command that creates tar file using gzip
7) -j : filter archive tar file using bzip2
8) -r : update or add file or directory in already existed .tar file

A) To create tar file
tar -cvf demo.tar file1.txt file2.txt file3.txt
tar -cvf demo.tar *
B) To display table of contents of tar file
tar -tvf demo.tar
C) To Extract contents of tar file
tar -xvf demo.tar
here v means verbose it will just display files which are compressed on terminal
we can ignore it

compressing any file
There are multiple compression and decompression algorithms.

gzip

It is very fast but less compression power
1) To Compress a file
$ gzip demo.tar

demo.tar.gz This file got created
we can compress any file not only tar file
2) To uncompress gz file:
$ gzip -d demo.tar.gz OR $ gunzip demo.tar.gz
This command will provide our original tar file