WC-word count command:-

wc stands for word count. As the name implies, it is mainly used for counting purpose.

It is used to find out number of lines, word count, byte and characters count in the files specified in the file arguments.

By default it displays four-columnar output.

First column shows number of lines present in a file specified, second column shows number of words present in the file, third column shows number of characters present in file and fourth column itself is the file name which are given as argument.

Syntax: wc [OPTION]... [FILE]...

Let us consider two files having name state.txt and capital.txt containing 5 names of the Indian states and capitals respectively.

$ cat state.txt

Andhra Pradesh

Arunachal Pradesh

Assam

Bihar

Chhattisgarh

$ cat capital.txt

Hyderabad

Itanagar

Dispur

Patna

Raipur

Passing only one file name in the argument.

$ wc state.txt

5 7 58 state.txt

OR

$ wc capital.txt

5 5 39 capital.txt

Passing more than one file name in the argument.

$ wc state.txt capital.txt

5 7 58 state.txt

5 5 39 capital.txt

10 12 97 total

Note : When more than file name is specified in argument then command will display four-columnar output for all individual files plus one extra row displaying total number of lines, words and characters of all the files specified in argument, followed by keyword total. Options: 1. -l: This option prints the number of lines present in a file. With this option wc command displays two-columnar output, 1st column shows number of lines present in a file and 2nd itself represent the file name.

With one file name

$ wc -l state.txt

5 state.txt

With more than one file name

$ wc -l state.txt capital.txt

5 state.txt

5 capital.txt

10 total

2. -w: This option prints the number of words present in a file. With this option wc command displays two-columnar output, 1st column shows number of words present in a file and 2nd is the file name.

With one file name

$ wc -w state.txt

7 state.txt

With more than one file name

$ wc -w state.txt capital.txt

7 state.txt

5 capital.txt

12 total

3. -c: This option displays count of bytes present in a file. With this option it display two-columnar output, 1st column shows number of bytes present in a file and 2nd is the file name.

With one file name

$ wc -c state.txt

58 state.txt

With more than one file name

$ wc -c state.txt capital.txt

58 state.txt

39 capital.txt

97 total

4. -m: Using -m option ‘wc’ command displays count of characters from a file.

With one file name

$ wc -m state.txt

56 state.txt

With more than one file name

$ wc -m state.txt capital.txt

58 state.txt

39 capital.txt

97 total

5. -L: The ‘wc’ command allow an argument -L, it can be used to print out the length of longest (number of characters) line in a file. So, we have the longest character line Arunachal Pradesh in a file state.txt and Hyderabad in the file capital.txt. But with this option if more than one file name is specified then the last row i.e. the extra row, doesn’t display total but it display the maximum of all values displaying in the first column of individual files. Note: A character is the smallest unit of information that includes space, tab and newline.

With one file name

$ wc -L state.txt

17 state.txt

With more than one file name

$ wc -L state.txt capital.txt

17 state.txt

10 capital.txt

17 total

6. –version: This option is used to display the version of wc which is currently running on your system.

$ wc --version

wc (GNU coreutils) 8.26

Packaged by Cygwin (8.26-1)

Copyright (C) 2016 Free Software Foundation, Inc.

License GPLv3+: GNU GPL version 3 or later .

This is free software: you are free to change and redistribute it.

There is NO WARRANTY, to the extent permitted by law.

Applications of wc Command:-

1. To count all files and folders present in directory: As we all know ls command in unix is used to display all the files and folders present in the directory, when it is piped with wc command with -l option it display count of all files and folders present in current directory.

$ ls gfg

a.txt

b.txt

c.txt

d.txt

e.txt

geeksforgeeks

India

$ ls gfg | wc -l

7

2. Display number of word count only of a file: We all know that this can be done with wc command having -w option, wc -w file\_name, but this command shows two-columnar output one is count of words and other is file name.

$ wc -w state.txt

7 state.txt

So to display 1st column only, pipe(|) output of wc -w command to cut command with -c option. Or use input redirection(<).

$ wc -w state.txt | cut -c1

7

OR

$ wc -w < state.txt

7

**Sort Commands:-**

In Linux, the `sort` command is used to arrange lines of text files or command output in a specific order. It is a versatile command with various options and functionalities. Below are detailed notes on the `sort` command in Linux:

Basic Usage:

1. Sort a File:

```bash

sort filename

```

This command sorts the lines of the specified file in lexicographical order (ASCII order) and prints the result to the standard output.

2. Sort Command Output:

```bash

command | sort

```

You can use the `sort` command to sort the output of another command.

Sorting Options:

1. Sort in Reverse Order:

```bash

sort -r filename

```

This sorts the file in reverse order (descending).

2. Sort Numerically:

```bash

sort -n filename

```

Perform a numerical sort instead of a lexicographical one.

3. Sort by a Specific Field:

```bash

sort -k <field\_number> filename

```

Sorts based on a specific field (specified by a field number). Fields are typically separated by whitespace.

4. Sort based on a Delimiter:

```bash

sort -t"<delimiter>" -k<field\_number> filename

```

Sorts based on a specific field using a custom delimiter.

5. Sort with Unique Lines:

```bash

sort -u filename

```

Removes duplicate lines, displaying only unique lines.

Other Options:

1. Ignoring Case:

```bash

sort -f filename

```

Ignores case distinctions in both lower and upper case.

2. Ignoring Leading Blanks:

```bash

sort -b filename

```

Ignores leading blanks when determining the starting point of a key.

3. Merging Already Sorted Files:

```bash

sort -m file1 file2

```

Merges two or more sorted files.

4. Specify Sort Key Start and End:

```bash

sort -k <start>,<end> filename

```

Specifies a key that starts and ends at a specific position in each line.

Examples:

1. Sort a File and Save the Output:

```bash

sort input.txt > output.txt

```

2. Sort a File Numerically:

```bash

sort -n numbers.txt

```

3. Sort by Second Field Using a Colon as Delimiter:

```bash

sort -t":" -k2 data.txt

```

4. Sort Command Output and Remove Duplicates:

```bash

command | sort -u

```

5. Sort in Reverse Order and Display Line Numbers:

```bash

cat file.txt | sort -rn | cat -n

```

These are just some of the many options available with the `sort` command in Linux. You can refer to the manual page (`man sort`) for more detailed information on the various options and use cases.