**Filter command**

**Head command**

It is the complementary of [Tail](https://www.geeksforgeeks.org/tail-command-linux-examples/) command. The head command, as the name implies, print the top N number of data of the given input. By default, it prints the first 10 lines of the specified files. If more than one file name is provided then data from each file is preceded by its file name.

**Syntax:**

**head option file**

**. -n num:**Prints the first ‘num’ lines instead of first 10 lines. **num** is mandatory to be specified in command otherwise it displays an error.

**$ head -n 5 state.txt**

Andhra Pradesh

Arunachal Pradesh

Assam

Bihar

Chhattisgarh

**Applications of head Command**

1. **Print line between M and N lines:**  Let say from state.txt file we have to print lines between 10 and 20.
2. **$ head -n 20 state.txt | tail -10**

**How to use the head with pipeline(|):** The head command can be piped with other commands. In the following example, the output of the ls command is piped to head to show only the three most recently modified files or folders.

1. Display all recently modified or recently used files.
2. Cut three most recently used file.
3. **$ ls -t | head -n 3**
4. e.txt
5. d.txt
6. c.txt

It can also be piped with one or more filters for additional processing. For example, the sort filter could be used to sort the three most recently used files or folders in the alphabetic order.

**$ ls -t | head -n 3 | sort**

c.txt

d.txt

e.txt

There are number of other filters or commands along which we use head command. Mainly, it can be used for viewing huge log files in Unix.

Tail command

It is the complementary of [head](https://www.geeksforgeeks.org/head-command-linux-examples/) command.The tail command, as the name implies, print the last N number of data of the given input. By default it prints the last 10 lines of the specified files. If more than one file name is provided then data from each file is precedes by its file name.

**Syntax:**

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

**$ tail state.txt**

Odisha

Punjab

Rajasthan

Sikkim

Tamil Nadu

Telangana

Tripura

Uttar Pradesh

Uttarakhand

West Bengal

**Options:**

1. **-n num:** Prints the last ‘num’ lines instead of last 10 lines. **num** is mandatory to be specified in command otherwise it displays an error. This command can also be written as without symbolizing ‘n’ character but ‘-‘ sign is mandatory.

**$ tail -n 3 state.txt**

Uttar Pradesh

Uttarakhand

West Bengal

OR

**$ tail -3 state.txt**

Uttar Pradesh

Uttarakhand

West Bengal

Tail command also comes with an **‘+’** 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. For command: **tail +n file\_name**, data will start printing from line number ‘n’ till the end of the file specified.

$ tail +25 state.txt

Telangana

Tripura

Uttar Pradesh

Uttarakhand

West Bengal

**Applications of tail Command**

1. **How to use tail with pipes(|):** The tail command can be piped with many other commands of the unix. In the following example output of the tail command is given as input to the sort command with -r option to sort the last 7 state names coming from file state.txt in the reverse order.

**$ tail -n 7 state.txt**

Sikkim

Tamil Nadu

Telangana

Tripura

Uttar Pradesh

Uttarakhand

West Bengal

**$ tail -n 7 state.txt | sort -r**

West Bengal

Uttarakhand

Uttar Pradesh

Tripura

Telangana

Tamil Nadu

Sikkim

It can also be piped with one or more filters for additional processing. Like in the following example, we are using cat, head and tail command and whose output is stored in the file name list.txt using directive(>).

**$ cat state.txt | head -n 20 | tail -n 5 > list.txt**

**$ cat list.txt**

Manipur

Meghalaya

Mizoram

Nagaland

Odisha

# SORT command in Linux/Unix with examples

SORT command is used to sort a file, arranging the records in a particular order.

SORT command sorts the contents of a text file, line by line.

**$ sort filename.txt**

**Options with sort function**

1. **-o Option :** Unix also provides us with special facilities like if you want to write the **output to a new file**, output.txt, redirects the output like this or you can also use the built-in sort option -o, which allows you to specify an output file.

**Syntax :**

1. **$ sort inputfile.txt > filename.txt**

**$ sort -o filename.txt inputfile.txt**

1. **-r Option: Sorting In Reverse Order**: You can perform a reverse-order sort using the -r flag. the -r flag is an option of the sort command which sorts the input file in reverse order i.e. descending order by default.  
   Example: The input file is the same as mentioned above.  
   **Syntax :**

**$ sort -r inputfile.txt**

Command :

$ sort -r file.txt

Output :

satish

rajan

naveen

harsh

divyam

chitransh

abhishek

1. **-n Option**: To sort a file **numerically** used –n option. -n option is also predefined in unix as the above options are. This option is used to sort the file with numeric data present inside.  
   Example :  
   Let us consider a file with numbers:
2. Command :
3. $ cat > file1.txt
4. 50
5. 39
6. 15
7. 89
8. 200

**Syntax :**

**$ sort -n filename.txt**

Command :

$ sort -n file1.txt

Output :

15

39

50

89

200

1. **-nr option**: To sort a file with**numeric data in reverse order** we can use the combination of two options as stated below.  
   Example :The numeric file is the same as above.  
   **Syntax :**

**$ sort -nr filename.txt**

Command :

$ sort -nr file1.txt

Output :

200

89

50

39

15

1. **-k Option**: Unix provides the feature of sorting a table on the **basis of any column number by using -k option.**  
   Use the -k option to sort on a certain column. For example, use “-k 2” to sort on the second column.  
   Example :  
   Let us create a table with 2 columns

$ cat > employee.txt

manager 5000

clerk 4000

employee 6000

peon 4500

director 9000

guard 3000

**Syntax :**

**$ sort -k filename.txt**

Command :

$ sort -k 2n employee.txt

guard 3000

clerk 4000

peon 4500

manager 5000

employee 6000

director 9000

**-c option :** This option is used to check if the **file given is already sorted or not** & checks if a file is already sorted pass the -c option to sort. This will write to standard output if there are lines that are out of order.The sort tool can be used to understand if this file is sorted and which lines are out of order

**Syntax :**

**$ sort -c filename.txt**

Command :

$ sort -c cars.txt

Output :

sort: cars.txt:3: disorder: BMW

**Note : If there is no output then the file is considered to be already sorted**

1. **-u option :** To **sort and remove duplicates** pass the -u option to sort. This will write a sorted list to standard output and remove duplicates.  
   This option is helpful as the duplicates being removed gives us an redundant file.  
   Example : Suppose a file exists with a list of cars called cars.txt.
2. Audi
3. BMW
4. Cadillac
5. BMW
6. Dodge

**Syntax :**

**$ sort -u filename.txt**

Command :

$ sort -u cars.txt

$ cat cars.txt

Output :

Audi

BMW

Cadillac

Dodge