

# Linux Basic Commands

## pwd command

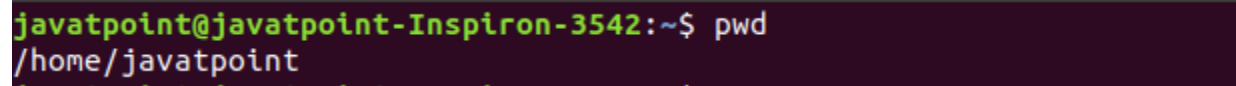
Use the **pwd** command to find out the path of the current working directory (folder) you're in. The command will return an absolute (full) path, which is basically a path of all the directories that starts with a forward slash (/). An example of an absolute path is **/home/username**.

The **pwd** command is used to display the location of the current working directory.

### Syntax:

```
pwd
```

### Output:



```
javatpoint@javatpoint-Inspiron-3542:~$ pwd
/home/javatpoint
```

## cd command

To navigate through the Linux files and directories, use the **cd** command. It requires either the full path or the name of the directory, depending on the current working directory that you're in.

There are some shortcuts to help you navigate quickly:

- **cd ..** (with two dots) to move one directory up
- **cd** to go straight to the home folder
- **cd-** (with a hyphen) to move to your previous directory

Note : Linux's shell is case sensitive.

## ls command

The **ls** command is used to view the contents of a directory. By default, this command will display the contents of your current working directory.

If you want to see the content of other directories, type **ls** and then the directory's path. For example, enter **ls /home/username/Documents** to view the content of **Documents**.

There are variations you can use with the **ls** command:

- **ls -R** will list all the files in the sub-directories as well
- **ls -a** will show the hidden files

- **ls -al** will list the files and directories with detailed information like the permissions, size, owner, etc.

## head Command

The [head](#) command is used to display the content of a file. It displays the first 10 lines of a file.

### Syntax:

1. head **<file name>**

### Output:

```
javatpoint@javatpoint-Inspiron-3542:~$ head Demo.txt
1
2
3
4
5
6
7
8
9
10
```

## 13. tail Command

The [tail](#) command is similar to the head command. The difference between both commands is that it displays the last ten lines of the file content. It is useful for reading the error message.

### Syntax:

1. tail **<file name>**

### Output:

```
javatpoint@javatpoint-Inspiron-3542:~$ tail Demo.txt
2
3
4
5
6
7
8
9
10
11
```

## more command

The [more](#) command is quite similar to the cat command, as it is used to display the file content in the same way that the cat command does. The only difference between both commands is that, in case of larger files, the more command displays screenful output at a time.

In more command, the following keys are used to scroll the page:

**ENTER key:** To scroll down page by line.

**Space bar:** To move to the next page.

**b key:** To move to the previous page.

**/ key:** To search the string.

### Syntax:

1. more **<file name>**

### Output:

```

;;; gyp.el - font-lock-mode support for gyp files.

;; Copyright (c) 2012 Google Inc. All rights reserved.
;; Use of this source code is governed by a BSD-style license that can be
;; found in the LICENSE file.

;; Put this somewhere in your load-path and
;; (require 'gyp)

(require 'python)
(require 'cl)

(when (string-match "python-mode.el" (symbol-file 'python-mode 'defun))
  (error (concat "python-mode must be loaded from python.el (bundled with "
                  "recent emacs), not from the older and less maintained "
                  "python-mode.el")))

(defadvice python-indent-calculate-levels (after gyp-outdent-closing-parens
                                              activate)
  "De-indent closing parens, braces, and brackets in gyp-mode."
  (when (and (eq major-mode 'gyp-mode)
              (string-match "^ *[])}][,)]* *$"
                          (buffer-substring-no-properties
                           (point)
                           (point-max))))
  - -More- - (7%)

```

## 16. less Command

The [less](#) command is similar to the more command. It also includes some extra features such as 'adjustment in width and height of the terminal.' Comparatively, the more command cuts the output in the width of the terminal.

### Syntax:

1. less <file name>

### Output:

```

;;; gyp.el - font-lock-mode support for gyp files.

;; Copyright (c) 2012 Google Inc. All rights reserved.
;; Use of this source code is governed by a BSD-style license that can be
;; found in the LICENSE file.

;; Put this somewhere in your load-path and
;; (require 'gyp)

(require 'python)
(require 'cl)

(when (string-match "python-mode.el" (symbol-file 'python-mode 'defun))
  (error (concat "python-mode must be loaded from python.el (bundled with "
                  "recent emacs), not from the older and less maintained "
                  "python-mode.el")))

(defadvice python-indent-calculate-levels (after gyp-outdent-closing-parens
                                             activate)

```

## sort Command

The [sort](#) command is used to sort files in alphabetical order.

### Syntax:

1. sort <file name>

### Output:

```

javatpoint@javatpoint-Inspiron-3542:~$ sort marks.txt
alen-70
alex-50
carry-85
celena-90
jon-75
justin-80

```

## date Command

The [date](#) command is used to display date, time, time zone, and more.

### Syntax:

1. date

### Output:

```
javatpoint@javatpoint-Inspiron-3542:~$ date
Fri May 22 21:51:05 IST 2020
```

### 38. cal Command

The `cal` command is used to display the current month's calendar with the current date highlighted.

#### Syntax:

1. `cal`

#### Output:

```
javatpoint@javatpoint-Inspiron-3542:~$ cal
      May 2020
Su Mo Tu We Th Fr Sa
                1  2
 3  4  5  6  7  8  9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
31
```

### time Command

The `time` command is used to display the time to execute a command.

#### Syntax:

1. `time`

#### Output:

```
javatpoint@javatpoint-Inspiron-3542:~$ time

real    0m0.000s
user    0m0.000s
sys     0m0.000s
```

### exit Command

Linux `exit` command is used to exit from the current shell. It takes a parameter as a number and exits the shell with a return of status number.

#### Syntax:

1. `exit`

### Output:

```
javatpoint@javatpoint-Inspiron-3542:~$ exit
```

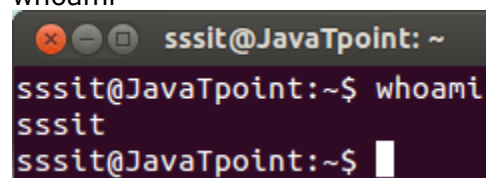
After pressing the ENTER key, it will exit the terminal.

## whoami

It tells you about the system's username.

### Syntax:

1. whoami



```
sssit@JavaTpoint: ~  
sssit@JavaTpoint:~$ whoami  
sssit  
sssit@JavaTpoint:~$
```

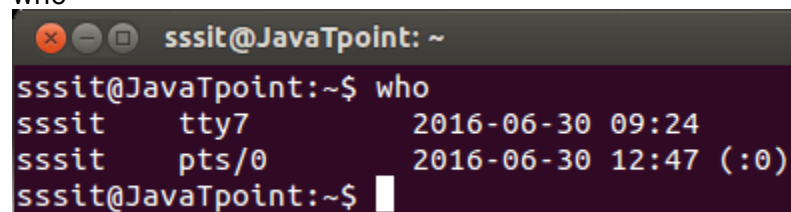
Look at the above snapshot, '**sssit**' is our system's username.

## who

The who command gives the information about the users logged on to the system.

### Syntax:

1. who



```
sssit@JavaTpoint: ~  
sssit@JavaTpoint:~$ who  
sssit    tty7          2016-06-30 09:24  
sssit    pts/0        2016-06-30 12:47 (:0)  
sssit@JavaTpoint:~$
```

## who am i

This command displays the information about the current user only.

### Syntax:

1. who am i

```

sssit@JavaTpoint: ~
sssit@JavaTpoint:~$ who am i
sssit pts/0 2016-06-30 12:47 (:0)
sssit@JavaTpoint:~$

```

Look at the above snapshot, in our system current logged in user is **sssit**.

## W

This command tells about the users who are logged in and what are they doing.

### Syntax:

1. w

```

sssit@JavaTpoint: ~
sssit@JavaTpoint:~$ w
16:02:31 up 6:37, 2 users, load average: 0.05, 0.12, 0.20
USER      TTY      FROM          LOGIN@      IDLE        JCPU       PCPU  WHAT
sssit     tty7          :0            09:24       6:37m      9:56       0.23s  gnome-session -
sssit     pts/0        :0            12:47       0.00s      0.17s      0.00s  w
sssit@JavaTpoint:~$

```

## Creating & Viewing Files :

### cat command

**cat** (short for concatenate) is one of the most frequently used commands in Linux. It is used to list the contents of a file on the standard output (sdout).

To run this command, type **cat** followed by the file's name and its extension. For instance: **cat file.txt**.

Other ways to use the **cat** command:

- **cat > filename** creates a new file
- Add content
- Press 'ctrl + d' to return to command prompt.
- **cat filename1 filename2>filename3** joins two files (1 and 2) and stores the output of them in a new file (3)
- to convert a file to upper or lower case use, **cat filename | tr a-z A-Z >output.txt**



## Example :

To view a file, use the command -

```
cat filename
```

```
guru99@VirtualBox:~$ cat sample1  
This is sample1
```

Let's see another file sample2

```
guru99@VirtualBox:~$ cat > sample2  
This is sample2
```

It can also be used for copying, combining and creating new text files.

The syntax to combine 2 files is -

```
cat file1 file2 > newfilename
```

Let's combine sample 1 and sample 2.

```
guru99@VirtualBox:~$ cat sample1 sample2 > sample
```

As soon as you insert this command and hit enter, the files are concatenated, but you do not see a result. This is because **Bash Shell (Terminal) is silent type**. Shell Commands will never give you a confirmation message like "OK" or "Command Successfully Executed". It will only show a message when something goes wrong or when an error has occurred.

To view the new combo file "sample" use the command

```
cat sample
```

```
guru99@VirtualBox:~$ cat sample  
This is sample1  
This is sample2
```

**Note:** Only text files can be displayed and combined using this command.

## cp command

Use the **cp** command to copy files from the current directory to a different directory. For instance, the command **cp scenery.jpg /home/username/Pictures** would create a copy of **scenery.jpg** (from your current directory) into the **Pictures** directory.

## Deleting Files

The 'rm' command removes files from the system without confirmation.

To remove a file use syntax -

```
rm filename
```

## Moving and Re-naming files

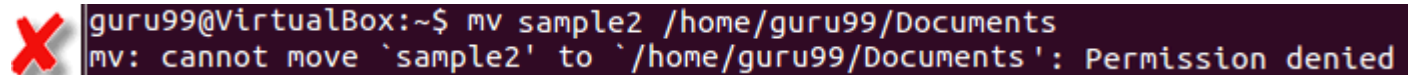
To **move a file**, use the command.

```
mv filename new_file_location
```

### Example :

Suppose we want to move the file "sample2" to location /home/guru99/Documents. Executing the command

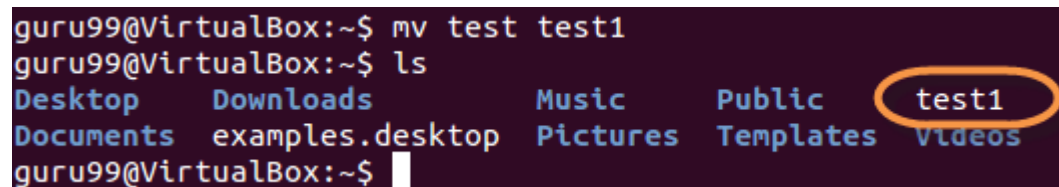
```
mv sample2 /home/guru99/Documents
```

A terminal window with a dark background. A red 'X' icon is on the left. The text shows a command prompt where the user attempts to move 'sample2' to '/home/guru99/Documents', but receives a 'Permission denied' error.

```
guru99@VirtualBox:~$ mv sample2 /home/guru99/Documents
mv: cannot move `sample2' to `/home/guru99/Documents': Permission denied
```

### For renaming file:(with super user privileges)

```
mv filename newfilename
```

A terminal window with a dark background. It shows a successful command to move 'test' to 'test1', followed by a 'ls' command. The output lists various directories, with 'test1' circled in orange.

```
guru99@VirtualBox:~$ mv test test1
guru99@VirtualBox:~$ ls
Desktop    Downloads    Music        Public    test1
Documents  examples.desktop  Pictures    Templates Videos
guru99@VirtualBox:~$
```

**NOTE:** By default, the password you entered for sudo is retained for 15 minutes per terminal. This eliminates the need of entering the password time and again.

You only need root/sudo privileges, only if the command involves files or directories not owned by the user or group running the commands

## Directory Manipulations

### Creating Directories

Directories can be created on a Linux operating system using the following command

```
mkdir directoryname
```

This command will create a subdirectory in your present working directory, which is usually your "Home Directory".

For example,

```
mkdir mydirectory
```

```
home@VirtualBox:~$ mkdir mydirectory
home@VirtualBox:~$ ls
Desktop    Downloads  Music      Pictures   Templates
Documents  examples.desktop mydirectory Public     Videos
home@VirtualBox:~$
```

To create a directory in a different location other than 'Home directory', use the following command -

```
mkdir
```

For example:

```
mkdir /tmp/MUSIC
```

will create a directory 'Music' under '/tmp' directory

```
home@VirtualBox:~$ mkdir /tmp/MUSIC
home@VirtualBox:~$ ls /tmp
keyring-yCD2no  pulse-0b9vyJcXyHZz  ssh-SSSsjczv1036  virtual-home.HaC7Mw
MUSIC          pulse-PKdhtXMmr18n  unity_support_test.1
home@VirtualBox:~$
```

You can also create more than one directory at a time.

```
home@VirtualBox:~$ mkdir dir1 dir2 dir3
home@VirtualBox:~$ ls
Desktop  dir2  Documents  examples.desktop  Pictures  Templates
dir1     dir3  Downloads  Music             Public    Videos
home@VirtualBox:~$
```

## Removing Directories

To remove a directory, use the command -

```
rmdir directoryname
```

Example

```
rmdir mydirectory
```

will delete the directory mydirectory

```
home@VirtualBox:~$ rmdir mydirectory
home@VirtualBox:~$ ls
Desktop  dir2  Documents  examples.desktop  Pictures  Templates
dir1     dir3  Downloads  Music             Public    Videos
home@VirtualBox:~$
```

**Tip:** Ensure that there is no file / sub-directory under the directory that you want to delete. Delete the files/sub-directory first before deleting the parent directory.

```
home@VirtualBox:~$ rmdir Documents
rmdir: failed to remove `Documents': Directory not empty
home@VirtualBox:~$
```

## Renaming Directory

The 'mv' (move) command (covered earlier) can also be used for renaming directories. Use the below-given format:

```
mv directoryname newdirectoryname
```

```
home@VirtualBox:~$ mv mydirectory newdirectory
home@VirtualBox:~$ ls
Desktop    Downloads    Music        Pictures    Templates
Documents  examples.desktop  newdirectory  Public      Videos
home@VirtualBox:~$
```

## The History Command

History command shows all the commands that you have used in the past for the current terminal session. This can help you refer to the old commands you have entered and re-used them in your operations again.

```
guru99@VirtualBox:~$ history
 1  cat > sample
 2  cat sample
 3  cat sample ^a
 4  cat sample a
 5  cat sample | grep a
 6  cat sample | grep ^a
 7  useradd home
 8  useradd mycomputer
 9  sudo useradd mycomputer
10  sudo adduser MyLinux
11  sudo adduser mylinux
12  vi scriptsample.sh
```

## The clear command

This command clears all the clutter on the terminal and gives you a clean window to work on, just like when you launch the terminal.

```
141 man
142 3a
143 man intro
144 man ls
145 man cat
146 man man
147 history
148 146
149 history 146
150 history
151 clear
152 history
guru99@VirtualBox:~$ clear
The window gets cleared
guru99@VirtualBox:~$
```

## Pasting commands into the terminal

Many times you would have to type in long commands on the Terminal. Well, it can be annoying at times, and if you want to avoid such a situation then copy, pasting the commands can come to rescue.

For copying, the text from a source, you would use **Ctrl + c**, but for pasting it on the Terminal, you need to use **Ctrl + Shift + p**. You can also try **Shift + Insert** or select **Edit>Paste** on the menu

NOTE: With Linux upgrades, these shortcuts keep changing. You can set your preferred shortcuts via **Terminal> Edit> Keyboard Shortcuts**.

## Printing in Unix/Linux

### 'pr' command

This command helps in formatting the file for printing on the terminal. The most used '**pr**' options are listed below.

| Option | Function |
|--------|----------|
|--------|----------|

|                |  |
|----------------|--|
| -x             | Divides the data into 'x' columns                        |
| -h "header"    | Assigns "header" value as the report header              |
| -t             | Does not print the header and top/bottom margins         |
| -d             | Double spaces the output file                            |
| -n             | Denotes all line with numbers                            |
| -l page length | Defines the lines (page length) in a page. Default is 56 |
| -o margin      | Formats the page by the margin number                    |

## Dividing data into columns

'Tools' is a file (shown below).

```
home@VirtualBox:~$ cat Tools
5/16" - 3/4" Standard Depth (6 Point)
3/8" - 3/4" Deep (6 Point)
9mm - 19mm Standard Depth (6 Point)
9mm - 19mm Deep (6 Point)
Ratchet
Extension - 3",6",12",18"
Universal Joint
Fractional Universal Impact Socket Set 3/8" - 3/4"
Metric Universal Impact Socket Set 9mm - 19mm
Slip Joint 6"
Needle Nose 6"
Diagonal Cutter 7"
Channel Locks 12" (water pump)
Long Reach End Cutter (Channel Lock #748)
Vise Grip Pliers 10" (10WR)
home@VirtualBox:~$
```

We want its content to be arranged in three columns. The syntax for the same would be:

```
pr -x Filename
```

The '-x' option with the 'pr' command divides the data into x columns.

```
home@VirtualBox:~$ pr -3 Tools
```

|                     |                     |                     |
|---------------------|---------------------|---------------------|
| 2012-09-02 19:27    | Tools               | Page 1              |
| 5/16" - 3/4" Standa | Extension - 3",6",1 | Needle Nose 6"      |
| 3/8" - 3/4" Deep (6 | Universal Joint     | Diagonal Cutter 7"  |
| 9mm - 19mm Standard | Fractional Universa | Channel Locks 12" ( |
| 9mm - 19mm Deep (6  | Metric Universal Im | Long Reach End Cutt |
| Ratchet             | Slip Joint 6"       | Vise Grip Pliers 10 |

## Assigning a header

The syntax is:

```
pr -h "Header" Filename
```

The '-h' options assigns "header" value as the report header.

```
home@VirtualBox:~$ pr -3 -h "Important Tools" Tools
```

|                     |                        |                     |
|---------------------|------------------------|---------------------|
| 2012-09-02 19:27    | <u>Important Tools</u> | Page 1              |
| 5/16" - 3/4" Standa | Extension - 3",6",1    | Needle Nose 6"      |
| 3/8" - 3/4" Deep (6 | Universal Joint        | Diagonal Cutter 7"  |
| 9mm - 19mm Standard | Fractional Universa    | Channel Locks 12" ( |
| 9mm - 19mm Deep (6  | Metric Universal Im    | Long Reach End Cutt |
| Ratchet             | Slip Joint 6"          | Vise Grip Pliers 10 |

As shown above, we have arranged the file in 3 columns and assigned a header

## Denoting all lines with numbers

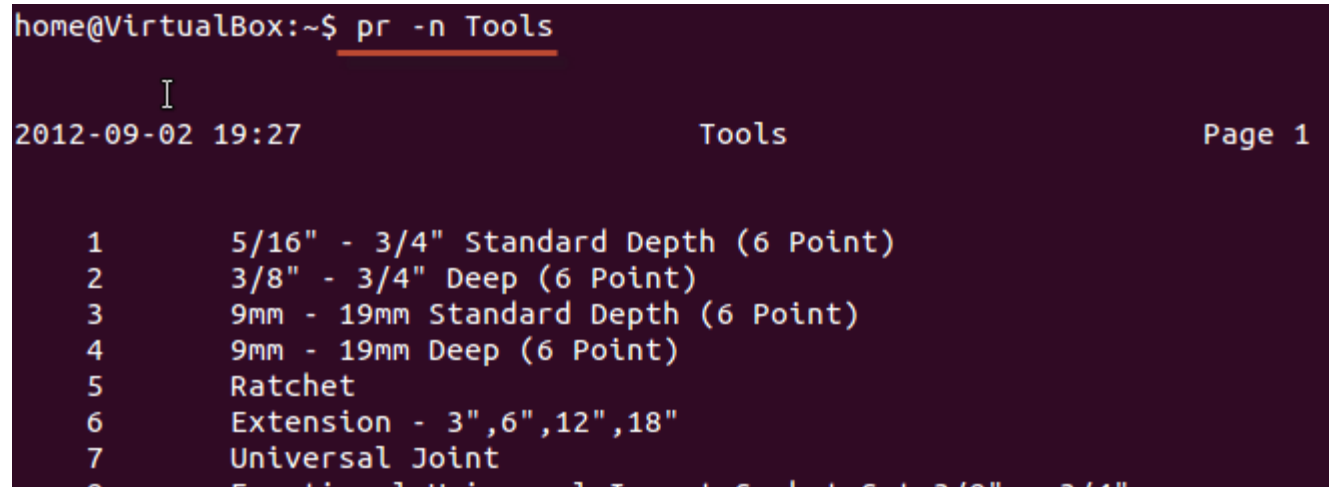


The syntax is:

```
pr -n Filename
```

This command denotes all the lines in the file with numbers.

```
home@VirtualBox:~$ pr -n Tools
```



These are some of the 'pr' command options that you can use to modify the file format.

## Printing a file

Once you are **done with the formatting**, and it is time for you to get a **hard copy** of the file, you need to use the following command:

```
lp Filename
```

or

```
lpr Filename
```

In case you want to print multiple copies of the file, you can use the number modifier.

## Print 10 Copies of a File

Specify -nNumber with 'lp' command

```
guru99@VirtualBox:~$ lp -n10 testfile
```

Specify Number with lpr command

```
guru99@VirtualBox:~$ lpr 10 testfile
```

In case you have multiple printers configured, you can specify a particular printer using the Printer modifier

In case of multiple printers , specify a particular printer

-dPrinter with lp command

```
guru99@VirtualBox:~$ lp -dHPofficejet testfile
```

-Pprinter with lpr command

```
guru99@VirtualBox:~$ lpr -PHPofficejet testfile
```

- form of packages. A package contains the program itself. Any dependent component needs to be downloaded separately.
- You can also send e-mails from terminal using the '**mail**' network commands. It is very useful linux commands.

## Linux Filters

Linux Filter commands accept input data from **stdin** (standard input) and produce output on **stdout** (standard output). It transforms plain-text data into a meaningful way and can be used with pipes to perform higher operations.

These filters are very small programs that are designed for a specific function which can be used as building blocks.

## Linux Cat Filters

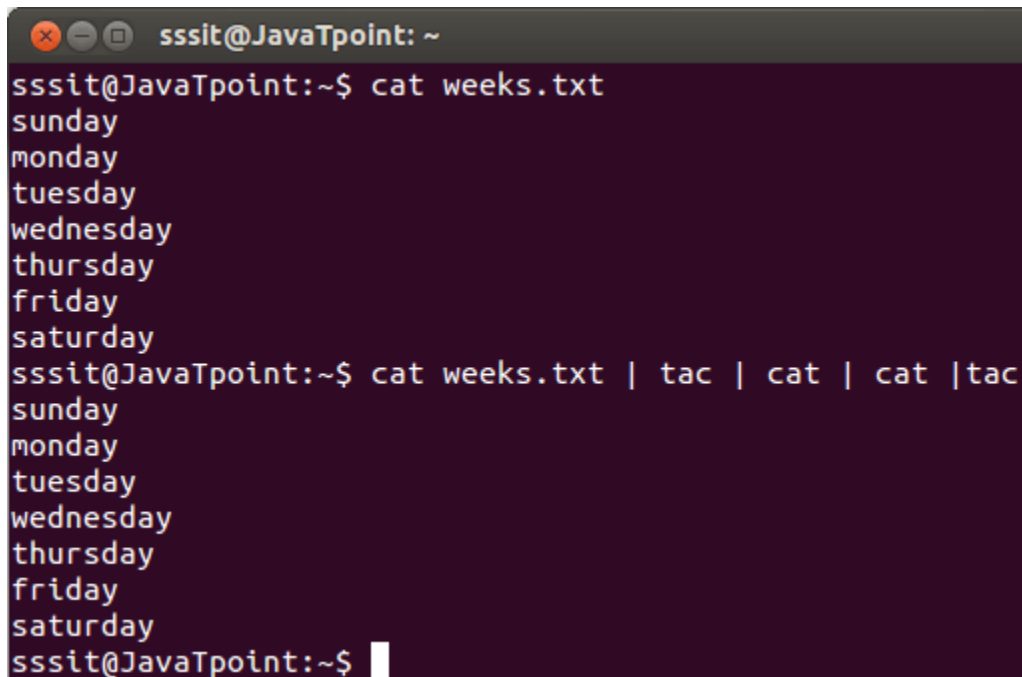
When cat command is used inside pipes, it does nothing except moving stdin to stout.

### Syntax:

cat <fileName> | cat or tac | cat or tac | . . .

### Example:

cat weeks.txt | tac | cat | cat | tac



```
sssit@JavaTpoint: ~  
sssit@JavaTpoint:~$ cat weeks.txt  
sunday  
monday  
tuesday  
wednesday  
thursday  
friday  
saturday  
sssit@JavaTpoint:~$ cat weeks.txt | tac | cat | cat | tac  
sunday  
monday  
tuesday  
wednesday  
thursday  
friday  
saturday  
sssit@JavaTpoint:~$
```

## Grep Command

The 'grep' command stands for "**global regular expression print**". grep command filters the content of a file which makes our search easy.

### grep with pipe

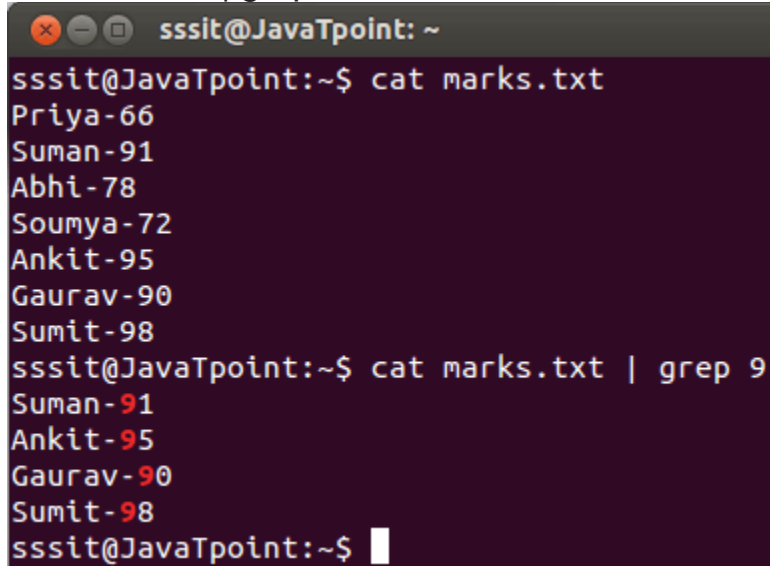
The 'grep' command is generally used with pipe (|).

### Syntax:

1. command | grep <searchWord>

### Example:

1. cat marks.txt | grep 9



```
sssit@JavaTpoint: ~  
sssit@JavaTpoint:~$ cat marks.txt  
Priya-66  
Suman-91  
Abhi-78  
Soumya-72  
Ankit-95  
Gaurav-90  
Sumit-98  
sssit@JavaTpoint:~$ cat marks.txt | grep 9  
Suman-91  
Ankit-95  
Gaurav-90  
Sumit-98  
sssit@JavaTpoint:~$
```

A terminal window with a dark purple background. The title bar shows 'sssit@JavaTpoint: ~'. The user enters 'cat marks.txt' and the output lists names and scores. Then, the user enters 'cat marks.txt | grep 9' and the output shows only the lines containing the number 9, with the numbers highlighted in red.

---

## grep without pipe

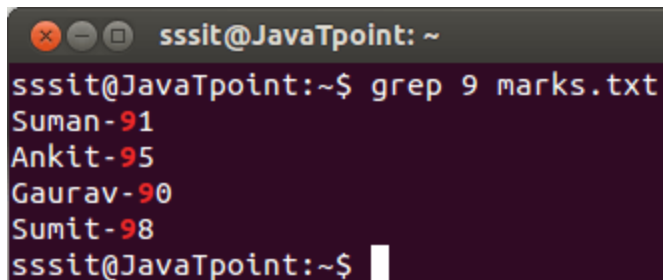
It can be used without pipe also.

### Syntax:

grep <searchWord> <file name>

### Example:

grep 9 marks.txt



```
sssit@JavaTpoint: ~  
sssit@JavaTpoint:~$ grep 9 marks.txt  
Suman-91  
Ankit-95  
Gaurav-90  
Sumit-98  
sssit@JavaTpoint:~$
```

A terminal window with a dark purple background. The title bar shows 'sssit@JavaTpoint: ~'. The user enters 'grep 9 marks.txt' and the output shows only the lines containing the number 9, with the numbers highlighted in red.

Look at the above snapshot, grep command do the same work as earlier example but without pipe.

---

grep options

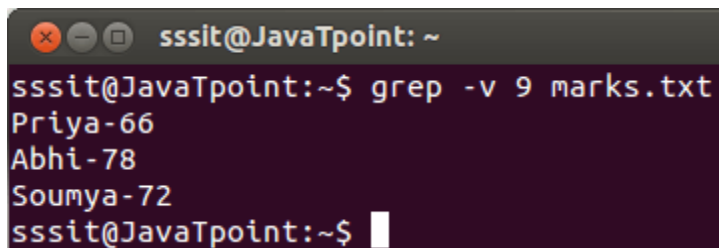
**grep -vM:** The 'grep -v' command displays lines not matching to the specified word.

**Syntax:**

grep -v <searchWord> <fileName>

**Example:**

grep -v 9 marks.txt

A terminal window with a dark purple background. The title bar shows 'sssit@JavaTpoint: ~'. The prompt is 'sssit@JavaTpoint:~\$'. The command 'grep -v 9 marks.txt' has been entered. The output consists of three lines: 'Priya-66', 'Abhi-78', and 'Soumya-72'. The prompt 'sssit@JavaTpoint:~\$' is followed by a white cursor bar.

```
sssit@JavaTpoint:~$ grep -v 9 marks.txt
Priya-66
Abhi-78
Soumya-72
sssit@JavaTpoint:~$
```

Look at the above snapshot, command "**grep -v 9 marks.txt**" displays lines which don't contain our search word '9'.

**grep -i:** The 'grep -i' command filters output in a case-insensitive way.

**Syntax:**

grep -i <searchWord> <fileName>

**Example:**

grep -i red exm.txt

```
sssit@JavaTpoint: ~  
sssit@JavaTpoint:~$ cat exm.txt  
Apple is red.  
Mango is yellow.  
your dress colour is Red.  
red colour suits on all.  
sssit@JavaTpoint:~$  
sssit@JavaTpoint:~$ grep -i red exm.txt  
Apple is red.  
your dress colour is Red.  
red colour suits on all.  
sssit@JavaTpoint:~$
```

Look at the above snapshot, command "**grep -i red exm.txt**" displays all lines containing 'red' whether in upper case or lower case.

### **grep -A/ grep -B/ grep -C**

grep -A command is used to display the **line after the result**.

grep -B command is used to display the **line before the result**.

grep -C command is used to display the **line after and line before** the result.

You can use (A1, A2, A3.....)(B1, B2, B3.....)(C1, C2, C3.....) to display any number of lines.

### **Syntax:**

grep -A<lineNumber> <searchWord> <fileName>

grep -B<lineNumber> <searchWord> <fileName>

grep -C<lineNumber> <searchWord> <fileName>

### **Example:**

grep -A1 yellow exm.txt

grep -B1 yellow exm.txt

grep -C1 yellow exm.txt

```
sssit@JavaTpoint: ~  
sssit@JavaTpoint:~$ grep -A1 yellow exm.txt  
Mango is yellow.  
your dress colour is Red.  
sssit@JavaTpoint:~$  
sssit@JavaTpoint:~$ grep -B1 yellow exm.txt  
Apple is red.  
Mango is yellow.  
sssit@JavaTpoint:~$  
sssit@JavaTpoint:~$ grep -C1 yellow exm.txt  
Apple is red.  
Mango is yellow.  
your dress colour is Red.  
sssit@JavaTpoint:~$
```

Look at the above snapshot, command "**grep -A1 yellow exm.txt**" displays searched line with next succeeding line, command "**grep -B1 yellow exm.txt**" displays searched line with one preceding line and command "**grep -C1 yellow exm.txt**" displays searched line with one preceding and succeeding line.

## Linux cut Command

Linux cut command is useful for selecting a specific column of a file. It is used to cut a specific sections by byte position, character, and field and writes them to standard output. It cuts a line and extracts the text data. It is necessary to pass an argument with it; otherwise, it will throw an error message.

To cut a specific section, it is necessary to specify the delimiter. A delimiter will decide how the sections are separated in a text file. Delimiters can be a space (' '), a hyphen (-), a slash (/), or anything else. After '-f' option, the column number is mentioned.

### Syntax:

1. cut OPTION... [FILE]...

### Options:

The following command line options are used by the cut command to make it more specific:

-b, --bytes=LIST: It is used to cut a specific section by bytes.

- c, --characters=LIST: It is used to select the specified characters.
- d, --delimiter=DELIM: It is used to cut a specific section by a delimiter.
- f, --fields=LIST: It is used to select the specific fields. It also prints any line that does not contain any delimiter character, unless the -s option is specified.
- n: It is used to ignore any option.
- complement: It is used to complement the set of selected bytes, characters or fields
- s, --only-delimited: It is used to not print lines that do not have delimiters.
- output-delimiter=STRING: This option is specified to use a STRING as an output delimiter; The default is to use "input delimiter".
- z, --zero-terminated: It is used if line delimiter is NUL, not newline.
- help: It is used to display the help manual.
- version: It is used to display the version information.

## **Examples of the cut command**

Let's see the following examples of the cut command:

- Cut by using Hyphen as delimiter
- Cut by using Space as delimiter
- Cut by byte position
- Cut by character
- Cut by complement pattern

### **Using Hyphen (-) As Delimiter**

To cut by using the hyphen (-) as the delimiter, execute the below command:

1. `cut -d- -f(columnNumber) <fileName>`

Consider the following commands:

1. `cut -d- -f2 marks.txt`



## 2. cut -d- -f1 marks.txt

from the above commands, the output will be trimmed from hyphen (-). Consider the below output:

```
javatpoint@javatpoint-Inspiron-3542:~$ cat marks.txt
alex-50
alen-70
jon-75
carry-85
celena-90
justin-80
javatpoint@javatpoint-Inspiron-3542:~$ cut -d- -f2 marks.txt
50
70
75
85
90
80
javatpoint@javatpoint-Inspiron-3542:~$ cut -d- -f1 marks.txt
alex
alen
jon
carry
celena
justin
```

As we can see from the above output, our delimiter is the hyphen (-); hence we have used (-) after (-d). Command "cut -d- -f1 marks.txt" displays column 1 and command "cut -d- -f2 marks.txt" displays column 2.

## Using Space As Delimiter

If we want to use space as a delimiter, then we have to quote the space (' ') with the cut command. To cut the output by using space as delimiter, execute the command as follows:

### 1. cut -d ' ' -f(columnNumber) <fileName>

Consider the following commands:

1. cut -d ' ' -f2 exm.txt
2. cut -d ' ' -f5 exm.txt

From the above commands, the output will be trimmed after space for the specified column. The above commands will produce the output as follows:

```
javatpoint@javatpoint-Inspiron-3542:~$ cat exm.txt
Apple is red.
mango is yellow.
your dress color is Red,
Red color suits on all.
javatpoint@javatpoint-Inspiron-3542:~$ cut -d ' ' -f2 exm.txt
is
is
dress
color
javatpoint@javatpoint-Inspiron-3542:~$ cut -d ' ' -f5 exm.txt

Red,
all.
javatpoint@javatpoint-Inspiron-3542:~$
```

From the above output, our delimiter is space; hence we have used (' ') after (-d). Command "cut -d ' ' -f2 exm.txt" displays column 2, command "cut -d ' ' -f5 exm.txt" displays column 5.

### Cut by byte

The '-b' option is used to cut a section of line by byte. To cut a file by its byte position, execute the command as follows:

1. cut -b <byte number> <file name>

Consider the below command:

1. cut -b 2 exm.txt

The above command will cut the line by a specified byte position. Consider the below output:

```
javatpoint@javatpoint-Inspiron-3542:~$ cut -b 2 exm.txt
p
a
o
e
```

### Cut by Character

The '-c' option is used to cut a specific section by character. However, these character arguments can be a number or a range of numbers, a list of comma-separated numbers, or any other character.

To cut by specified character, execute the command as follows:

1. `cut -c < characters> <file name>`

Consider the below commands:

1. `cut -c 1,6 exm.txt`
2. `cut -c 1-3 exm.txt`

The above commands will cut the line by the specified characters. Consider the below output:

```
javatpoint@javatpoint-Inspiron-3542:~$ cat exm.txt
Apple is red.
mango is yellow.
your dress color is Red,
Red color suits on all.
javatpoint@javatpoint-Inspiron-3542:~$ cut -c 1,6 exm.txt
A
m
yd
Ro
javatpoint@javatpoint-Inspiron-3542:~$ cut -c 1-3 exm.txt
App
man
you
Red
```

From the above output, we can see the first command is cutting the first and sixth character from each line, and the second command is cutting the first to the third character from each line.

## Linux wc Command

Linux wc command helps in counting the lines, words, and characters in a file. It displays the number of lines, number of characters, and the number of words in a file. Mostly, it is used with pipes for counting operation.

Syntax:

`wc [OPTION]... [FILE]...`

Options:

Some useful command line options supported by the `wc` command are as following:

**-c, --bytes:** It is used to print the byte counts.

**-m, --chars:** It is used to print the character counts.

**-l, --lines:** It is used to print the newline counts.

**--files0-from=F:** It is used to read input from specified files.

**-L, --max-line-length:** It is used to print the maximum display width.

**-w, --words:** It is used to print the word counts.

**--help:** It is used to display the help manual.

**--version:** It is used to display the version information.

## Display count information of multiple files

To display the complete count information of multiple files at once, specify the file names after space (' '). It is executed as follows:

```
wc <file1> <file2>
```

Consider the below example:

```
wc exm.txt marks.txt
```

The above command will display the number of words, the number of characters, and the number of the bytes from the files 'exm.txt' and 'marks.txt'. Consider the below output:

```
javatpoint@javatpoint-Inspiron-3542:~$ wc exm.txt marks.txt
 4  16  80 exm.txt
 6   6  52 marks.txt
10  22 132 total
```

## Display the number of lines in a file

The '-l' option is used to display the number of lines in a file. It is executed as follows:

`wc -l <file name>`

Consider the below example:

`wc -l exm.txt`

The above command will display the number of lines from 'exm.txt'. Consider the below output:

```
javatpoint@javatpoint-Inspiron-3542:~$ wc -l exm.txt
4 exm.txt
```

## Display the number of characters in a file

The '-m' option is used to display the number of characters in a file. It is executed as follows:

`wc -m <file name>`

Consider the below example:

`wc -m exm.txt`

The above command will display the number of words from the file 'exm.txt'. Consider the below output:

```
javatpoint@javatpoint-Inspiron-3542:~$ wc -m exm.txt
80 exm.txt
```

## Display the number of bytes in a file

The '-c' option is used to display the number of bytes in a file. It is executed as follows:

`wc -c <file name>`

Consider the below example:

`wc -c exm.txt`

The above command will display the number of bytes in a file. Consider the below output:

```
javatpoint@javatpoint-Inspiron-3542:~$ wc -c exm.txt
80 exm.txt
```

## Display the number of words in a file

The '-w' option is used to display the total number of words from a file. It is executed as follows:

`wc -w <file name>`

Consider the below example:

`wc -w exm.txt`

The above command will display the total number of words from the file 'exm.txt'. Consider the below output:

```
javatpoint@javatpoint-Inspiron-3542:~$ wc -w exm.txt
16 exm.txt
```

## Count the number files in a directory

To count the number of files and folders in a directory, combine the `wc` command with the `ls` command. Execute it as follows:

`ls | wc -l`

The above command will display the count of the files from the current working directory. Consider the below output:

```
javatpoint@javatpoint-Inspiron-3542:~$ ls | wc -l
50
```

## Display the length of the longest line

The '-L' option is used to display the length of the longest line from a file. It is executed as follows:

```
wc -L <file name>
```

Consider the below example:

```
wc -L exm.txt
```

The above command will display the length of the longest line of the file 'exm.txt'. Consider the below output:

```
javatpoint@javatpoint-Inspiron-3542:~$ wc -L exm.txt
24 exm.txt
```

