

A related command is **cal** which, by default, displays a calendar of the current month.

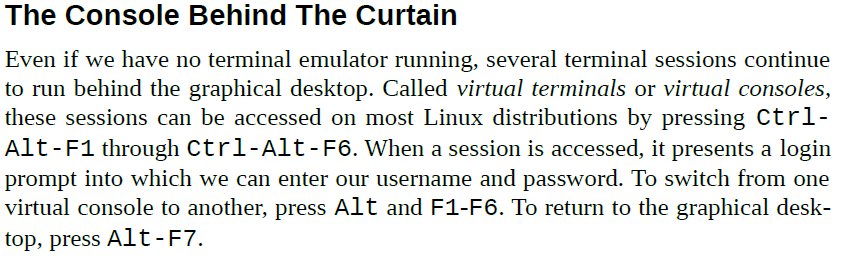
**bash-3.2$ cal**

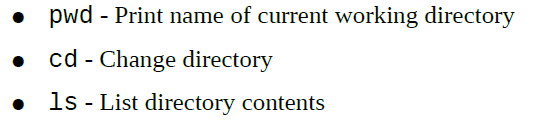
To see the current amount of free space on your disk drives, enter df:

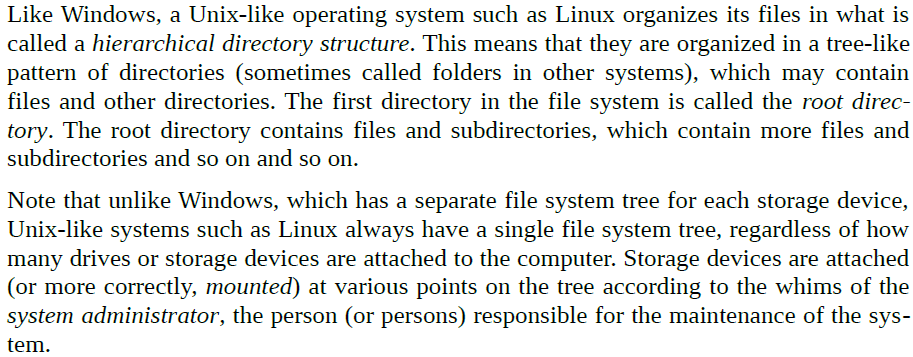
**bash-3.2$ df**

Likewise, to display the amount of free memory, enter the free command

**bash-3.2$ free**

****

****

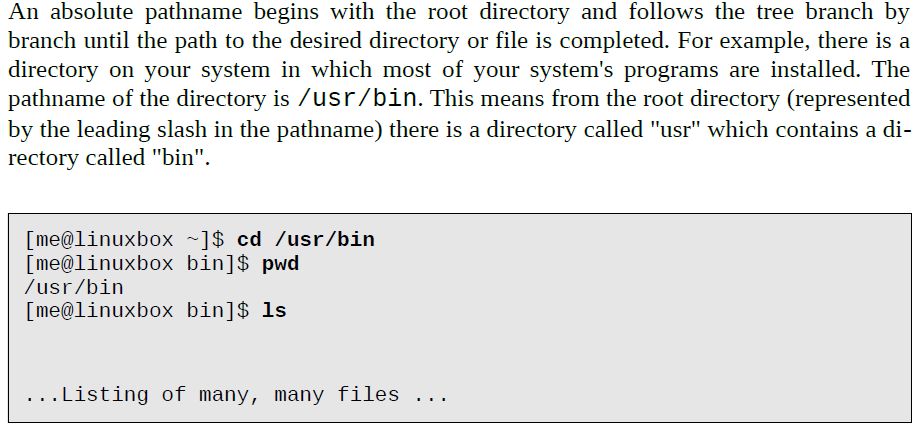
****

To display the current working directory, we use the pwd (print working directory) command

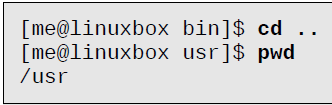
**bash-3.2$ pwd**

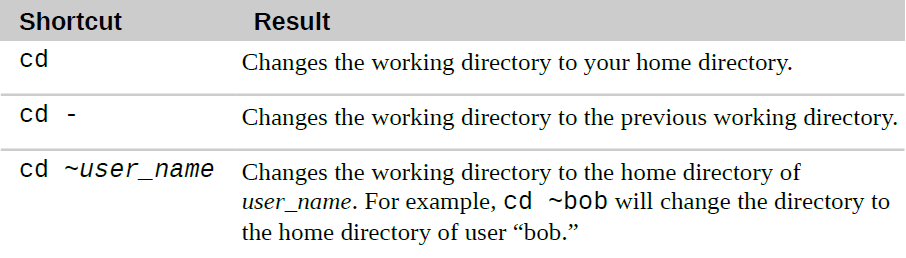
To list the files and directories in the current working directory, we use the ls command

**bash-3.2$ ls**

****

**The "."** symbol refers to the working directory and **the ".."** symbol refers to the working directory's parent directory.

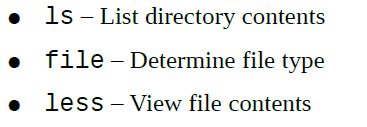
****



Filenames that begin with a period character are hidden. This only means that ls will not list them unless you say **ls -a**. When your account was created, several hidden files were placed in your home directory to configure things for your account.

Linux has no concept of a “file extension” like some other operating systems. You may name files any way you like. The contents and/or purpose of a file is determined by other means.

Though Linux supports long filenames which may contain embedded spaces and punctuation characters, limit the punctuation characters in the names of files you create to period, dash, and underscore. ***Most importantly, do not embed spaces in filenames.***If you want to represent spaces between words in a filename, use underscore characters.



We can simply enter ls to see a list of files and subdirectories contained in the current working directory:

**bash-3.2$ ls**

Specify the directory to list the contents of the directory, like:

**bash-3.2$ ls /usr**

To get the specified directory files in the given directory:

**bash-3.2$ ls ~/scr**

**To change the output to long format we can use the command:**

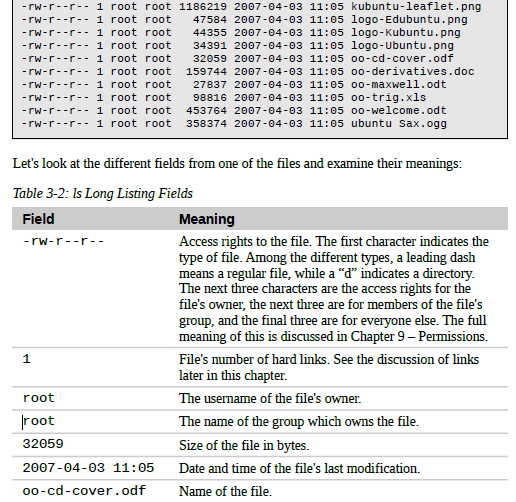
**bash-3.2$ ls –l**

In this example, the ls command is given two options, the “l” option to produce long format output, and the “t” option to sort the result by the file's modification time:

**bash-3.2$ ls –lt**

Add the long option “--reverse” to reverse the order of the sort

**bash-3.2$ ls –lt –reverse**

****

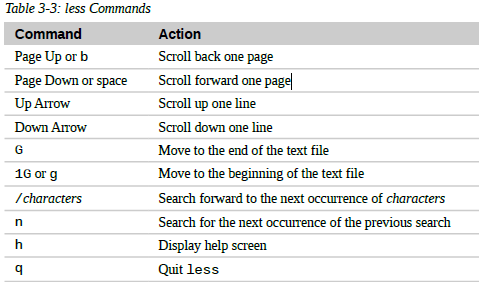
we explore the system it will be useful to know what files contain. To do this we will use the file command to determine a file's type:

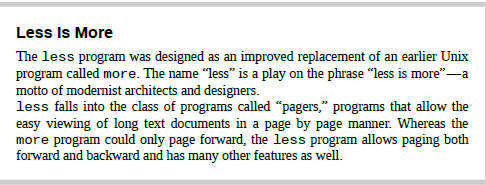
**bash-3.2$ file math3**

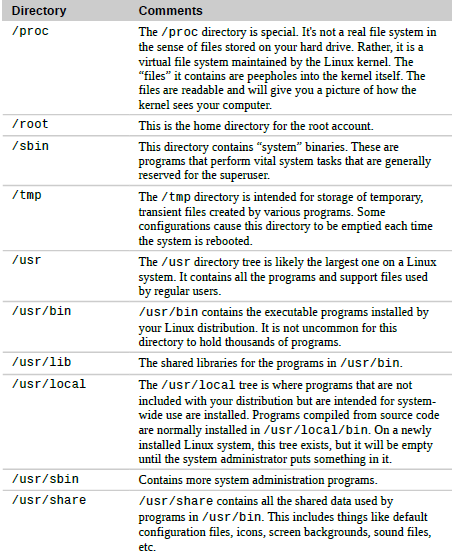
**result will be: ASCII text**

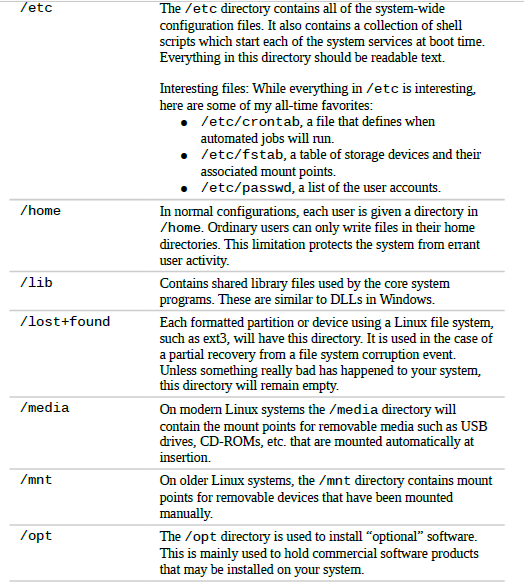
The less command is a program to view text files. Throughout our Linux system, there are many files that contain human-readable text. The less program provides a convenient way to examine them.

**bash-3.2$ less math3**

****

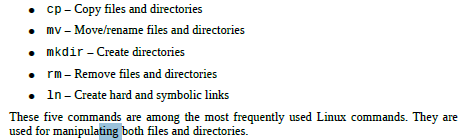
****

****

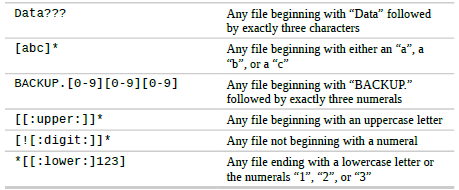
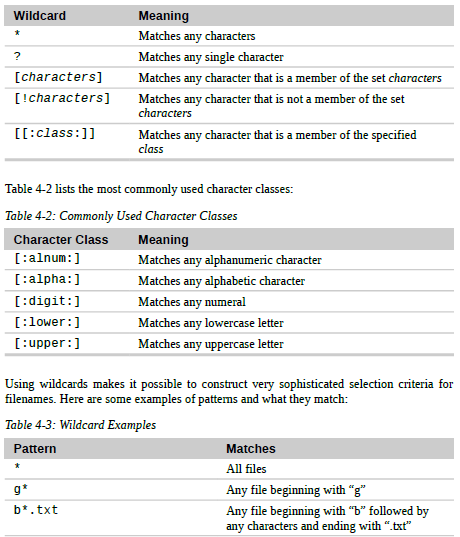
****

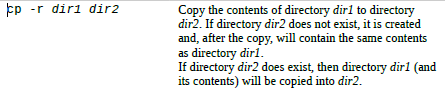
**lrwxrwxrwx 1 root root 11 2007-08-11 07:34 libc.so.6 -> libc-2.6.so**

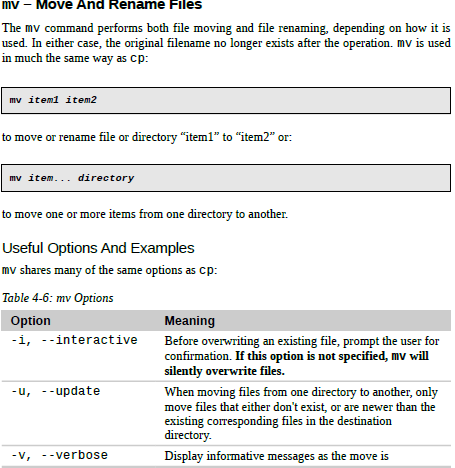
The above line demonstrate the symbolic link.The directory listing above (from the /lib directory of a Fedora system) shows a symbolic link called “libc.so.6” that points to a shared library file called “libc-2.6.so.” This means that programs looking for “libc.so.6” will actually get the file “libc-2.6.so.”

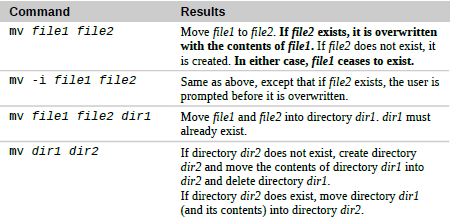


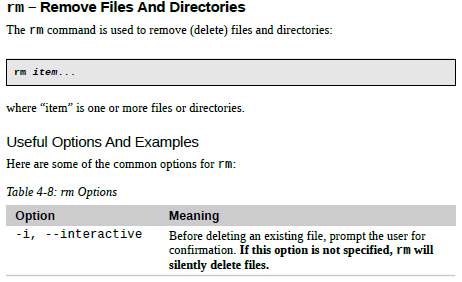
The shell uses filenames so much, it provides special characters to help you rapidly specify groups of filenames. These special characters are called wildcards.

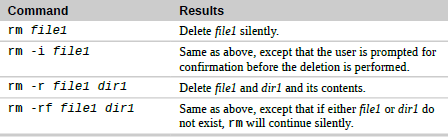
****

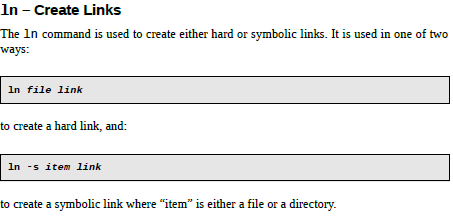
****

****

****

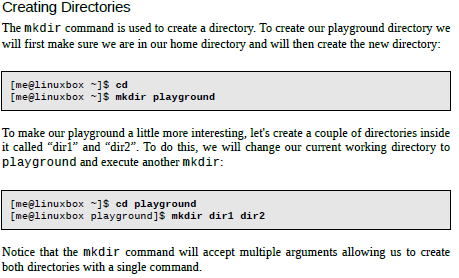
****

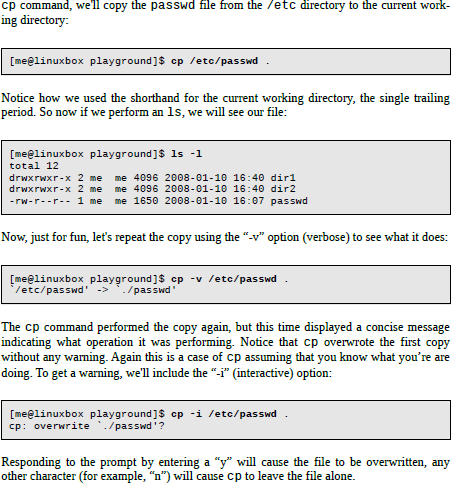
****

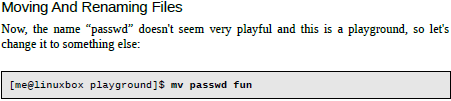
****

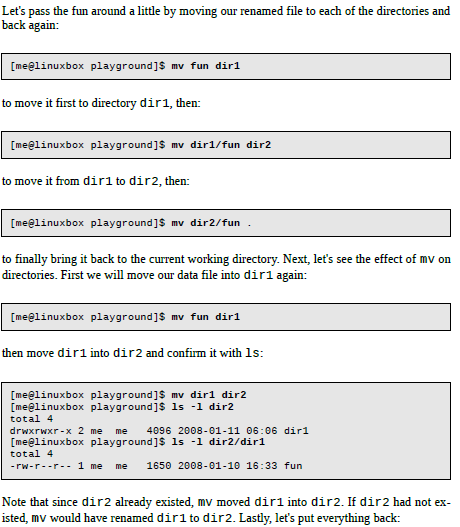
**bash-3.2$ cp /etc/passwd .** This command is used to copy the passwd file from etc directory into the current directory and the period is used to represent the current directory

**example:**

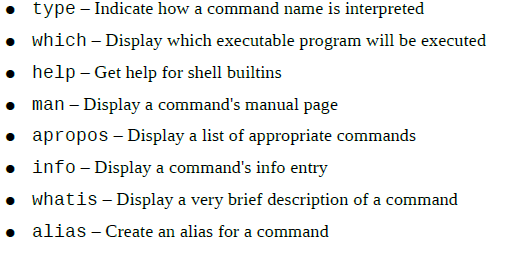






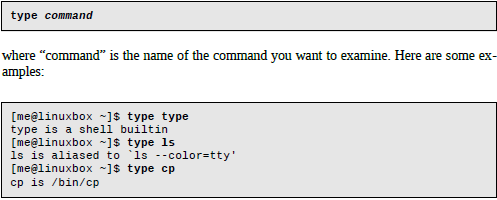






**A command built into the shell itself**. bash supports a number of commands internally called *shell builtins*. The cd command, for example, is a shell builtin.

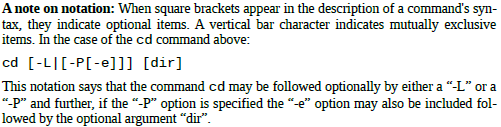
**An alias.** Commands that we can define ourselves, built from other commands.



To determine the exact location of a given executable, the which command is used:

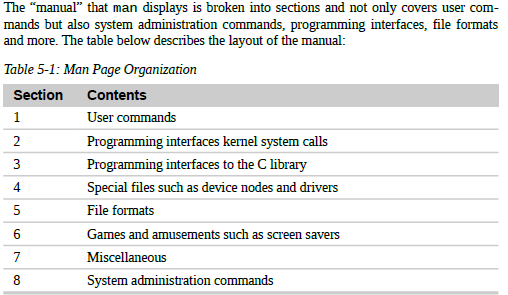


which only works for executable programs, not builtins nor aliases that are substitutes for actual executable programs.



The man command is used to check the manual of any command

Eg. Man ls will show ls command manual.





The whatis program displays the name and a one line description of a man page matching a specified keyword:

**bash-3.2$ whatis ls**

It's possible to put more than one command on a line by separating each command with a semicolon character. It works like this:

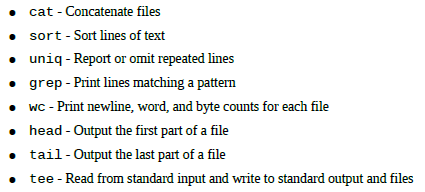
**bash-3.2$ whatis ls; ls; cd ..; ls**

**bash-3.2$ alias foo='whatis ls; ls'**

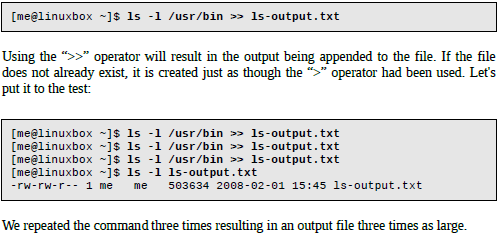
This command is used to create alias and define multiple commands inside it.

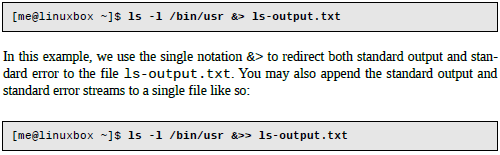
**bash-3.2$ unalias foo**

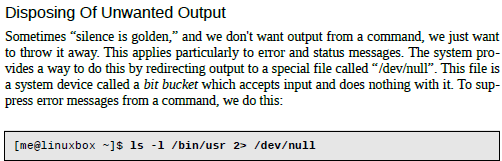
This command is used to unalias the created alias



I/O redirection allows us to change where output goes and where input comes from. Normally, output goes to the screen and input comes from the keyboard, but with I/O redirection, we can change that. I/O redirection allows us to redefine where standard output goes. To redirect standard output to another file instead of the screen, we use **the “>”** redirection operator followed by the name of the file. we append redirected output to a file instead of overwriting the file from the beginning? For that, we use the **“>>”** redirection operator.







cat – Concatenate Files

The cat command reads one or more files and copies them to standard output like so:



will display the contents of the file ls-output.txt. cat is often used to display short

text files.

The ability of commands to read data from standard input and send to standard output is utilized by a shell feature called *pipelines*. Using the pipe operator “|” (vertical bar), the standard output of one command can be *piped* into the standard input of another:

**bash-3.2$ ls -l /usr/bin | less**

**bash-3.2$ ls -l /usr/bin | sort | less**

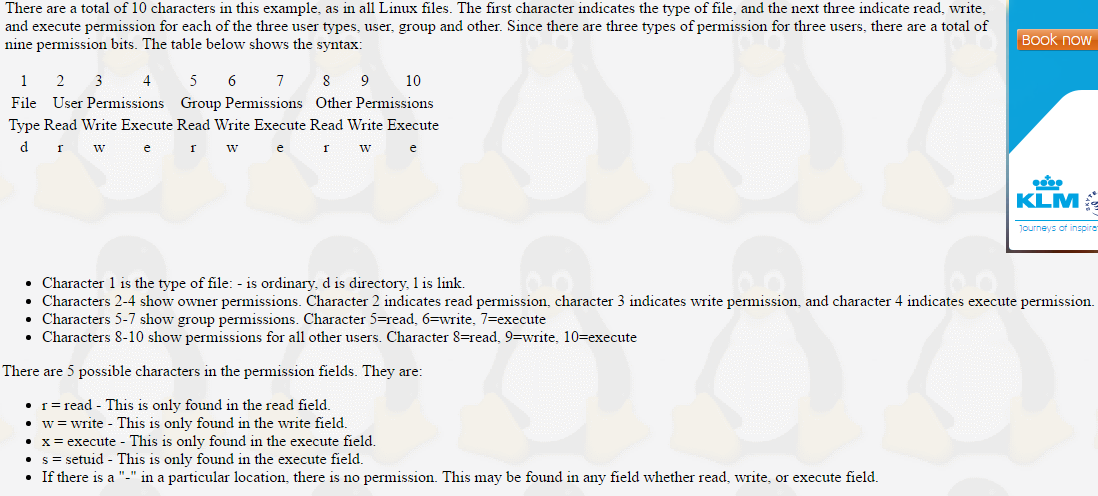
The uniq command is often used in conjunction with sort. uniq accepts a sorted list

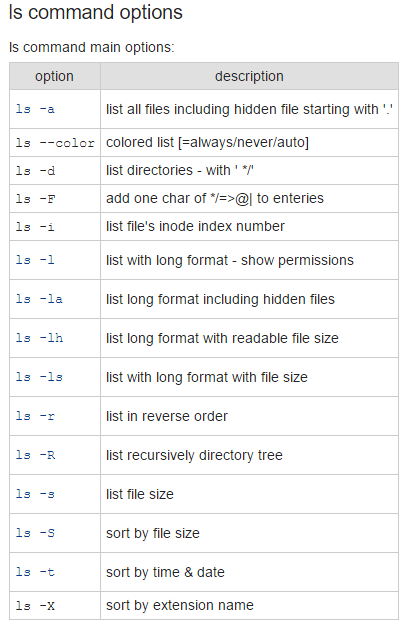
of data from either standard input or a single filename argument

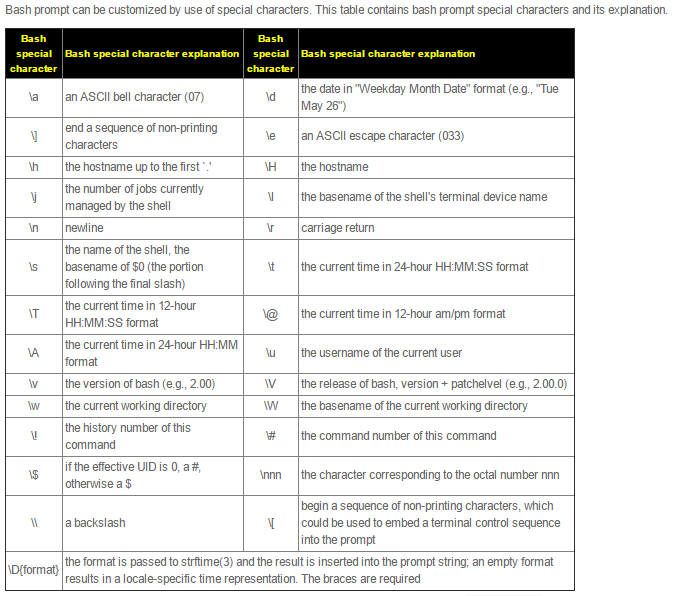
**bash-3.2$ ls -l /usr/bin | sort |uniq | less**

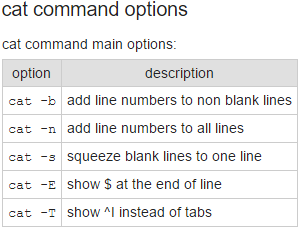
* The following command displays a message (use banner on Unix): **figlet hello world**
* The script can be tested as follows: **bash hello or . hello**
* The current path can be viewed by using the **echo $PATH** to view current path variable
* If the current working directory is not present, the PATH variable may be amended to add the directory: export PATH=$PATH**:. (The dot in the end after colon means to add all scripts)**
* To find bashrc file in the current profile or directory use: **ls -al .bashrc**
* To edit .bashrc file just do: **vi .bashrc**
* To login again without going out use:**. .bashrc**
* To count number of files in current directory: ls | wc –l
* To count the number of files in a directory, I typically use

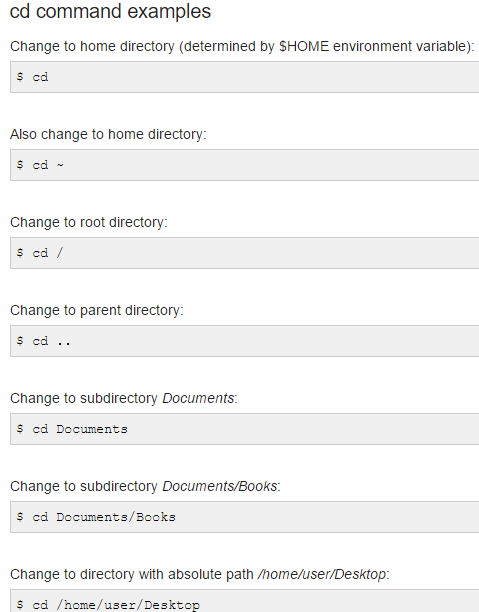
ls directory | wc -l

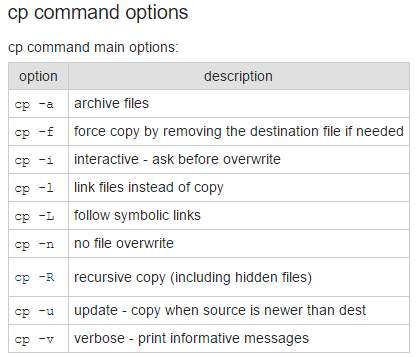




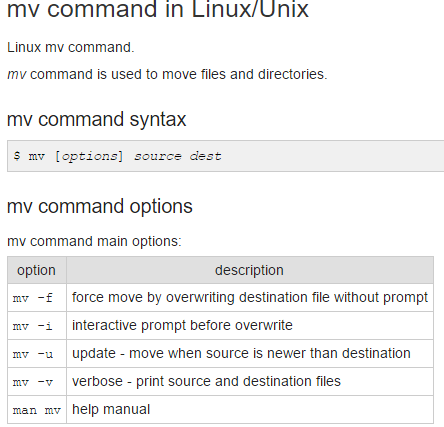


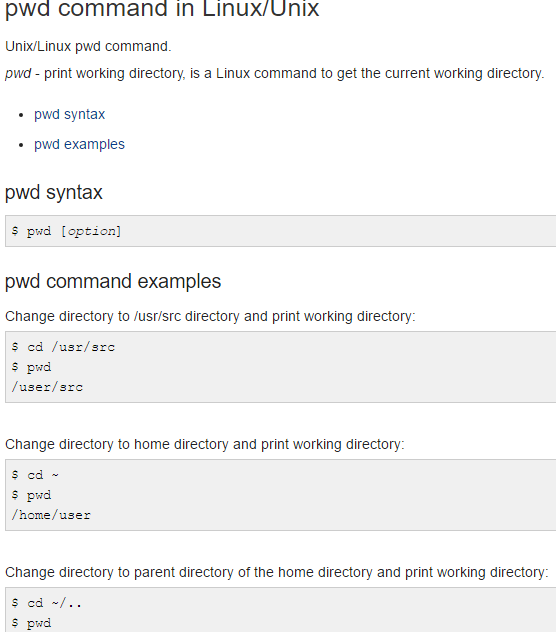
****

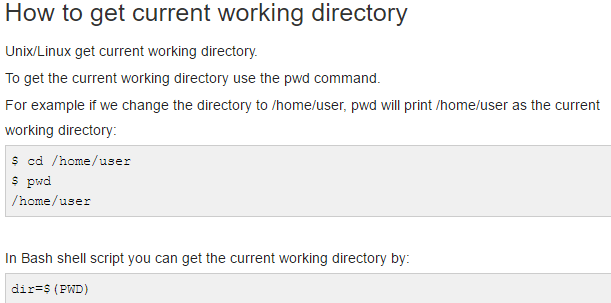
****

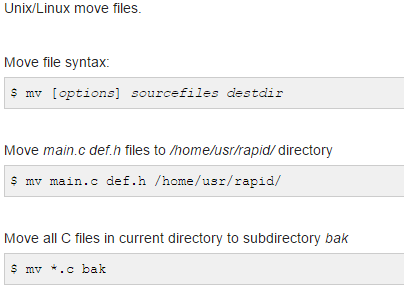
****

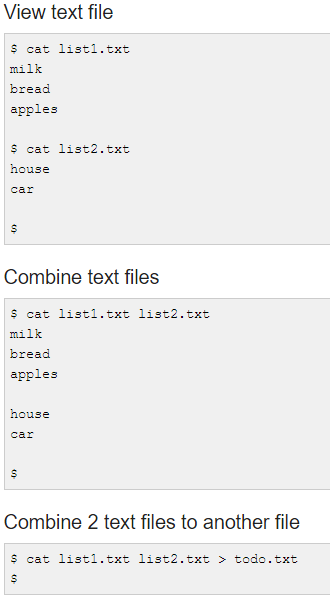
****

****

****

****

****

**ls**