Bash Cheat Sheet

Wildcards * , ? , \ , [ ]

* -> whichever characters as many times as you like
? -> one whatever character

[xx] -> represents a group of characters
Ex:
ls [a-z]*
ls [A-Z]*
ls [a-zA-Z]*
ls [0-9]*

Escape character \

Ex:
ls ** : lists all files and directories in the current directory that have an asterisk (*) in their names.
ls /My\ Files/ : Telling the shell My Files is one single directory

Sudo command

Acknowledge excuting permission for commands that can only be run by the root user.

$ **sudo ls /root**

Navegating Directories

$ **pwd**: print current directory path
$ **whoami**: print current session user
$ **cd foo**: go to foo subdirectory (using relative route)
$ **cd** and **cd ~** : go to home directory
$ **cd ..** : go to last directory

Directories

Creating directories

$ mkdir foo   # Create a directory
$ mkdir foo bar   # Create multiple directories
$ mkdir -p foo/bar/baz   # Create nested directories and add the ones that doesn't exist yet

Moving directories

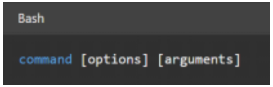
$ cp -R /home/user/docs /home/user/backups   # Copies "docs" directory content into "backups" directory
(what to copy?)   (where to copy?)

$ mv /home/user/docs /home/user/backups   # Moves "docs" directory content into "backups" directory
(what to copy?)   (where to copy?)

Deleting directories

$ rmdir foo   # Delete empty directory

rm -r   # Delete directory and it's content, ignore nonexistent files



Bash

```
command [options] [arguments]
```

Useful commands for sysadmins

ps : stands for "process status" , prints processes running in the current shell session
Options:
ps aux or **ps -ef**: Lists all processes running on the system.

w: who's on my server

I/O commands

-Taking into account the standard Input/Output process is keyboard for input, screen for output:

$ ls > listing.txt
Sets output on a new file "listing.txt"

$ ls >> listing.txt
Appends output to the existing file, instead of overwrite it

$ sort < file.txt
Gets input from a file

$ sort < file.txt > sorted_file.txt
Save the sorted results in a new file

Pipe (**|**): The standard output of the command on the left side of the pipe, is passed as the standard input to the command on the right side.
I personally like to see it as "the left command feeds the right one"
Common uses:

Filtering and summarizing

$ ps -ef | more : Displays all running processes in full format and allows you to view the output one screen at a time.
$ ps -ef | head :Displays the first 10 lines of the full list of all running processes
$ ps -ef | grep daemon : prints only the lines that contain the word daemon

Chain commands to pass their outputs as needed

$ cat file.txt | fmt | pr | lpr

Explanation:
1: **cat** file.txt reads the contents of file.txt and outputs it.
2: The output from cat is passed to **fmt**, which formats the text for better readability (like adjusting line lengths).
3: The formatted text is passed to **pr**, which adds headers, footers, and paginates the text for printing.
4: Finally, the paginated text is sent to the printer using **lpr**.

Files

Creating files

$ touch foo.txt	# Create file or update existing files modified timestamp
$ touch foo.txt bar.txt	# Create multiple files
$ touch {foo,bar}.txt	# Create multiple files

Moving files

$ cp foo.txt bar.txt	# Copy file
$ mv foo.txt /home/user/docs	# Move file
$ mv foo.txt bar.txt	# Rename file

Deleting files

$ rm foo.txt	# Delete existent file
rm -f	# Delete file, ignore nonexistent files

Finding files

Find: needs to specify a path

$ find /path -name foo.txt	# Find a file
$ find /path -iname foo.txt	# Find a file with case insensitive search

Locate: uses an index and is fast

$ updatedb	# Update the index

$ locate foo.txt	# Find a file
locate --ignore-case	# Find a file and ignore case sensitive
$ locate f*.txt	# Find a text file starting with 'f'

By: Carolina Montoya