# Linux commands

By: Ashish Singh Rathore

**Remember that in Linux, folders end with a forward slash (/).**

## 1. Mkdir

To create folders in the shell, you use the mkdir command. Just specify the new folder's name, ensure it doesn't exist, and you're ready to go.

For example, to make a directory to keep all of your images, just type:

```
mkdir images/
```

To create subdirectories with a simple command, use the parent (-p) flag:

```
mkdir -p movies/2004/
```

## 2. ls

Description: It allows you to list the contents of the directory you want (the current directory by default), including files and other nested directories.

```
ls <enter>
```

## 2.Alias

Description: The alias command lets you define temporary aliases in your shell session. When creating an alias, you instruct your shell to replace a word with a series of commands.

```
alias ls="ls --color=auto"<enter>
```

## 3.unalias

Description: As the name suggests, the unalias command aims to remove an alias from the already defined aliases. To remove the previous ls alias, you can use:

```
unalias ls
```

# 4.pwd

Description: The pwd command stands for “print working directory,” and it outputs the absolute path of the directory you’re in.  you can use:

```
Pwd <enter>
```

# 5.cd

Description:The cd command is highly popular, along with ls. It refers to “change directory” and, as its name suggests, switches you to the directory you’re trying to access. you can use:

```
Cd ashish <enter>
```

1) Go to the home folder

```
Cd
```

2) Move a level up

```
cd ..
```

3) Return to the previous directory

```
cd -
```

# 6.cp

It’s so easy to copy files and folders directly in the Linux terminal that sometimes it can replace conventional file managers.

To use the cp command, just type it along with the source and destination files:

```
cp file_to_copy.txt new_file.txt
```

You can also copy entire directories by using the recursive flag:

```
cp -r dir_to_copy/ new_copy_dir/
```

**Remember that in Linux, folders end with a forward slash (/).**

# 7. rm

Now that you know how to copy files, it'll be helpful to know how to remove them.

You can use the rm command to remove files and directories. Be careful while using it, though, because it's very difficult (yet not impossible) to recover files deleted this way.

To delete a regular file, you'd type:

```
rm file_to_copy.txt
```

# 8. mv

You use the mv command to move (or rename) files and directories through your file system.

To use this command, you'd type its name with the source and destination files:

```
mv source_file destination_folder/
```

```
mv command_list.txt commands/
```

To utilize absolute paths, you'd use:

```
mv /home/kinsta/BestMoviesOfAllTime ./
```

…where ./ is the directory you're currently in.

You also can use mv to rename files while keeping them in the same directory:

```
mv old_file.txt new_named_file.txt
```

# 9. touch Command

The touch command allows you to update the access and modification times of the specified files.

For example, I have an old file that was last modified file.

```
touch abc <enter>
```

most of the time, you won't use touch to modify file dates, but rather to create new empty files:

```
touch new_file_name
```

## 10. chmod Command

The chmod command lets you change the mode of a file (permissions) quickly. It has a lot of options available with it.
The basic permissions a file can have are:

## r (read)
## w (write)
## x (execute)

One of the most common use cases for chmod is to make a file executable by the user. To do this, type chmod and the flag +x, followed by the file you want to modify permissions on:

```
chmod +x script
```

You use this to make scripts executable, allowing you to run them directly by using the ./ notation.

## 11. exit Command

The exit command does exactly what its name suggests: With it, you can end a shell session and, in most cases, automatically close the terminal you're using:

```
Exit
```

## 12. man Command

Another essential Linux command is man. It displays the manual page of any other command (as long as it has one).
To see the manual page of the mkdir command, type:

```
man mkdir
```

You could even refer to the man manual page:

```
man man
```

# 13. shutdown Command

As you may guess, the shutdown command lets you power off your machine. However, it also can be used to halt and reboot it.

To power off your computer immediately (the default is one minute), type:

```
shutdown now
```

You can also schedule to turn off your system in a 24-hour format:

```
shutdown 20:40
```

To cancel a previous shutdown call, you can use the -c flag:

```
shutdown -c
```

# 14. echo command

The echo command displays defined text in the terminal — it's that simple:

```
echo "you all are good"
```

Its primary usage is to print environmental variables inside those messages:

```
echo "Hey $USER"

# Hey ASR
```

# 15. cat Command

Cat, short for "concatenate," lets you create, view, and concatenate files directly from the terminal. It's mainly used to preview a file without opening a graphical text editor:

```
cat long_text_file.txt
```

# 16. kill command

It's annoying when a program is unresponsive, and you can't close it by any means. Fortunately, the kill command solves this kind of problem.

Simply put, kill sends a TERM or kill signal to a process that terminates it.

You can kill processes by entering either the PID (processes ID) or the program's binary name:

```
kill 533494
```

```
kill firefox
```

Be careful with this command — with kill, you run the risk of accidentally deleting the work you've been doing.

# 17. Ping  command

ping is the most popular networking terminal utility used to test network connectivity. ping has a ton of options, but in most cases, you'll use it to request a domain or IP address:

```
ping google.com
```

```
ping 8.8.8.8
```

# 18. history Command

If you're struggling to remember a command, history comes in handy. This command displays an enumerated list with the commands you've used in the past:

```
History
```

# 19. passwd Command

passwd allows you to change the passwords of user accounts. First, it prompts you to enter your current password, then asks you for a new password and confirmation.

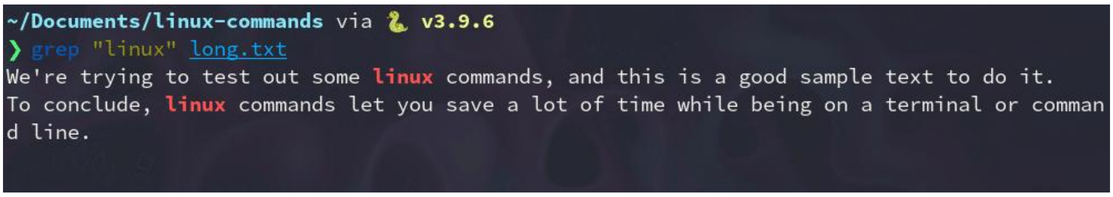
It's similar to any other change of password you've seen elsewhere, but in this case, it's directly in your terminal:

```
Passwd
```

# 20. grep Command

Grep is one of the most powerful utilities for working with text files. It searches for lines that match a regular expression and print them:

```
grep "linux" long.txt
```



```
~/Documents/linux-commands via 🐍 v3.9.6
❯ grep "linux" long.txt
We're trying to test out some linux commands, and this is a good sample text to do it.
To conclude, linux commands let you save a lot of time while being on a terminal or comman
d line.
```

## 21. whoami Command

The whoami command (short for “who am i”) displays the username currently in use:

**whoami**

**# ASR**

You would get the same result by using echo and the environmental variable

**$USER:**

**echo $USER**

**# ASR**

## 22. wc Command

Wc stands for “word count,” and as the name suggests, it returns the number of words in a text file:

wc long.txt

Let’s breakdown the output of this command:

37 lines
207 words
1000 byte-size
The name of the file (long.txt)

If you only need the number of words, use the -w flag:

```
Wc -w long.text
207 long.text
```

# 23. uname Command

uname(short for “Unix name”) prints the operative system information, which comes in handy when you know your current Linux version.

Most of the time, you’ll be using the -a (–all) flag, since the default output isn’t that useful:

```
uname

# Linux

uname -a
```

# 24. neofetch Command

Neofetch is a CLI (command-line interface) tool that displays information about your system — like kernel version, shell, and hardware — next to an ASCII logo of your Linux distro:

```
Neofetch
```

In most machines, this command isn’t available by default, so make sure to install it with your package manager first.