1. ls- which is short for listing. This command will list the contents of the current directory.

ls prints the names of the files and directories in the current directory. We can make its output more comprehensible by using the -F **option** (also known as a **switch** or a **flag**) , which tells ls to classify the output by adding a marker to file and directory names to indicate what they are:

* 1. a trailing / indicates that this is a directory
  2. @ indicates a link
  3. indicates an executable

ls is the **command**, with an **option** -F and an **argument** /. We’ve already encountered options (also called **switches** or **flags**) which either start with a single dash (-) or two dashes (--), and they change the behaviour of a command. Arguments tell the command what to operate on (e.g. files and directories). Sometimes options and arguments are referred to as **parameters**. A command can be called with more than one option and more than one argument: but a command doesn’t always require an argument or an option.

Each part is separated by spaces: if you omit the space between ls and -F the shell will look for a command called ls-F, which doesn’t exist. Also, capitalization can be important. For example,

1. ls -s will display the size of files and directories alongside the names, while ls -S will sort the files and directories by size

Getting help

1. ls has lots of other **options**. There are two common ways to find out how to use a command and what options it accepts:

We can pass a --help option to the command ls – help

1. ls -t lists items by time of last change instead of alphabetically.
2. The command ls -r lists the contents of a directory in reverse order.
3. ***cd*** stands for ‘change directory’, which is a bit misleading: the command doesn’t change the directory, it changes the shell’s idea of what directory we are in. The command to change locations is cd followed by a directory name to change our working directory. , The cd command is akin to double clicking a folder in a graphical interface to get into a folder.
4. -a stands for ‘show all’; it forces ls to show us file and directory names that begin with ., such as .. The special directory ’ ..’ doesn’t usually show up when we run ls. If we want to display it, we can add the -a option to ls -F. The command is ***ls -F -a***
5. pwd - which stands for ‘print working directory’). Directories are like *places* - at any time while we are using the shell we are in exactly one place, called our **current working directory**. Commands mostly read and write files in the current working directory, i.e. ‘here’, so knowing where you are before running a command is important. pwd shows you where you are.
6. If your screen gets too cluttered, you can clear your terminal using the ***clea****r* command. You can still access previous commands using ↑ and ↓ to move line-by-line, or by scrolling in your terminal.

### Create a directory

Let’s create a new directory called thesis using the command mkdir thesis (which has no output):

1. mkdir means ‘make directory’
2. Note that mkdir is not limited to creating single directories one at a time. The -p option allows mkdir to create a directory with any number of nested subdirectories in a single operation:

mkdir **-p** thesis/chapter\_1/section\_1/subsection\_1

1. The -R option to the ls command will list all nested subdirectories wtihin a directory. Let’s use ls -FR to recursively list the new directory hierarchy we just created beneath the thesis directory

### Create a text file

Let’s change our working directory to thesis using cd, then run a text editor called Nano to create a file called draft.txt:

1. nano draft.txt
2. touch my\_file.txt