Session 3

The file system is the part of the operating system (OS) that manages files and directories. It is responsible for organising data into files (information stores), and directories (or folders), which hold files/other directories.

To find out what directory we are in we can use the pwd command (which stands for ‘print working directory’). This is where we are in the computer. Most commands that we will cover perform actions on files in the directory they are executed in.

$ pwd

(base) /Users/Sam2u17

This is my home directory. Yours may look different depending if you are using Anaconda and if you are using a different virtual environment.

### What is the home directory?

To understand what a ‘home directory’ is, let’s analyse how the file system as a whole is organized.

The root directory is at the top and we refer to it using a slash character, / , on its own. The first slash in /Users/Sam refers to root.

Inside root are several other directories: bin (some built-in programs are stored here), data (miscellaneous data files), Users (your personal directories), tmp (temporary files).

We can tell that the current working directory /Users/Sam is stored inside /Users because /Users comes first. Similarly, we know that /Users is stored within root / as it begins with a /.

### What are slashes?

Slashes, / , have two meanings. A slash at the front of a file or directory name refers to the root directory as we have seen. However, when it appears after this, so inside a path, it just acts to separate up words.

Now look at your own filesystem. Use ls:

$ ls

Applications Documents Library Music Public

Desktop Downloads Movies Pictures

(Again, your results may be slightly different depending on your operating system and how you have customized your filesystem.)

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, or flag. This tells ls to classify the output by adding a marker

* a trailing / indicates that this is a directory
* @ indicates a link
* \* indicates an executable

Some shells may also use different colours.

$ ls **-F**

Applications/ Documents/ Library/ Music/ Public/

Desktop/ Downloads/ Movies/ Pictures/

### Decluttering your terminal

Use the clear command to clear everything currently showing on your terminal screen.

You can still access previous commands using ↑ and ↓ to move line-by-line, or by scrolling in the terminal.

### General syntax of a shell command

If we look at the following:

$ ls **-F** /

ls is the command, the option is  -F and the argument is  /. Options (also called switches or flags) either start with a single or double dash (-) (--). They change the behaviour of a command.

Arguments tell the command what to operate on (e.g. files and directories). 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.

Separate each part with a space. If you don’t put a space between ls and -F the shell will look for the command ls-F, which doesn’t exist.

Capitalisation is important. ls -s will display the size of files and directories alongside the names, while ls -S will sort the files and directories by size:

$ ls -s Desktop/Data\_Files\_TMCS/data

total 208

8 amino-acids.txt 0 elements 24 planets.txt

0 animal-counts 8 morse.txt 8 salmon.txt

8 animals.txt 0 pdb 152 sunspot.txt

$ ls -S Desktop/Data\_Files\_TMCS/data

sunspot.txt elements morse.txt animals.txt salmon.txt

planets.txt pdb amino-acids.txt animal-counts

You should be able to run the above commands and see similar results if you have downloaded the data file and put it on desktop.

Putting all that together, our command above gives us a listing of files and directories in the root directory /. An example of the output you might get from the above command is given below:

$ ls **-F** /

Applications/ System/

Library/ Users/

Network/ Volumes/

### Help

ls has many options. A common way of finding out how to use a command and what options it accepts:

Read the manual with man:

$ man ls

This will turn your terminal into a page with a description of the ls command and its options and, if you’re lucky, some examples of how to use it.

To navigate through the man pages, you may use ↑ and ↓ to move line-by-line. Use space bar to skip whole pages. To quit the man pages, press Q.

### Quick Questions:

### Find some more ls flags. What do they do?

### Exploring Other Directories

### We can use ls to list the content of other directories. We simply put the directory that we want (the argument) after the command (ls) and the option.

$ ls **-F** Desktop

Data\_Files\_TMCS /

Your output will be a list of your own files and sub-directories in Desktop.

Now we can look at the contents of Data\_Files\_TMCS, using the same strategy as before, passing a directory name to ls:

$ ls **-F** Desktop/Data\_Files\_TMCS

creatures/ north-pacific-gyre/ solar.pdf

data/ notes.txt writing/

molecules/ pizza.cfg

We can also move our working directory into a new directory by using the cd command, followed by a directory name. cd stands for ‘change directory’.

To move to the Data\_Files\_TMCS directory we can use the following commands:

$ cd Desktop

$ cd Data\_Files\_TMCS

$ cd data

You’ll notice that cd doesn’t print anything. Most shell commands won’t output anything to the screen when successfully executed.

To go up directories use:

$ cd ..

.. refers to the parent directory to the one you are in.

The directory .. doesn’t usually show up when running ls. To display it, add the -a option to ls -F:

$ ls **-F** **-a**

./ data/ pizza.cfg

../ molecules/ solar.pdf

.bash\_profile north-pacific-gyre/ writing/

creatures/ notes.txt

-a stands for ‘show all’. As you can see, it also displays another special directory that’s just called ., which means ‘the current working directory’.

Note that in most command line tools, multiple options can be combined with a single - and no spaces between the options: ls -F -a is equivalent to ls -Fa.

**Other Hidden Files**

In addition to the hidden directories .. and ., you may also see a file called .bash\_profile. This file usually contains shell configuration settings. You may also see other files and directories beginning with .. These are usually files and directories that are used to configure different programs on your computer. The prefix . is used to prevent these configuration files from cluttering the terminal when a standard ls command is used.

**Orthogonality**

The special names . and .. don’t belong to cd; they are interpreted the same way by every program. For example, if we are in /Users/Sam/data, the command ls ..will give us a listing of /Users/Sam. When the meanings of the parts are the same no matter how they’re combined, programmers say they are orthogonal: Orthogonal systems tend to be easier for people to learn because there are fewer special cases and exceptions to keep track of.

### Quick Questions:

1. What happens If you just use cd
2. Can you move to a directory that isn’t one above, or one below, in one line of code (hint: use ‘absolute paths’

### More shortcuts

### The shell interprets the character ~ (tilde) at the start of a path to mean “the current user’s home directory”.

### Another shortcut is the - (dash) character. cd will translate - into the previous directory I was in. This is a veryefficient way of moving back and forth between directories. The difference between cd .. and cd - is that the former brings you up, while the latter brings you back.

### Quick Questions:

1. Use the following commands. What do they do?
2. cd .
3. cd /
4. cd ../..
5. cd ~
6. cd home
7. cd ~/data/..
8. cd
9. cd ..
10. What do the following commands do? -r tells ls to display things in reverse order. Try to figure out what they will do before inputting them in terminal.

**Takeaways**

1. The file system is responsible for managing information on the disk.
2. Information is stored in files, which are stored in directories (folders).
3. Directories can also store other directories, which forms a directory tree.
4. cd path changes the current working directory.
5. ls path prints a listing of a specific file or directory; ls on its own lists the current working directory.
6. pwd prints the user’s current working directory.
7. / on its own is the root directory of the whole file system.
8. A relative path specifies a location starting from the current location.
9. An absolute path specifies a location from the root of the file system.
10. Directory names in a path are separated with / on Unix, but \ on Windows.
11. .. means ‘the directory above the current one’; . on its own means ‘the current directory’.