Unix Tutorial 1: Files in Linux

Log in to your Unix system with your username and password, then open a bash shell. On the Glasgow remote Linux setup, you drop into a shell automatically with your ssh login. On a Linux laptop, you should execute command `gnome-terminal` or `xterm` to start a bash shell in a window. On a Mac machine, you should execute the `Terminal` application. On Windows, if you have WSL installed, the `wsl` command will drop you into a shell.

Type a simple command, e.g.

```
$ date
```

Press Enter after typing in date. The system should print out the current date and time. Try a few more simple commands ...

```
$ hostname
$ whoami
$ cal 2023
```

and observe the output. To quit your bash session, either type exit or press ctrl+D together.

Now we are going to practise file manipulations, moving around the directory hierarchy. The special directory ~ is your home directory, and / is the root directory. Try this:

```
$ cd /
$ ls
$ cd ~
$ ls
```

Two more special directories are 'dot' `.` (the current directory) and 'dot dot' `..` (the parent directory). Another useful command is `pwd` which shows the current directory you are in.

Challenge: When you first login to your system, how many directories do you have to go up (with `cd ..`) to reach the root directory? Use `pwd` to work out when you reach the root.

Now we are going to create some files. We will use the cat command to write a string of characters into plaintext files.

```
$ cat > a.txt
The quick brown fox jumps over the lazy dog.
```

You will need to press Enter then ctrl+D after typing the words in the sentence. Here, ctrl+D means 'end of file'.

You can run ls again, to show that file a.txt is now in your home directory. Execute some more commands:

```
$ cp a.txt b.txt
$ ls
$ mv a.txt fox.txt
$ cp fox.txt f.txt
```

Now we want to create a new directory and move all the files into this directory.

```
$ mkdir tutorial1
$ ls
$ mv *.txt tutorial1
$ cd tutorial1
$ ls
$ cd ..
```

Now suppose we want to save this directory as a compressed zip archive called files.zip.

```
$ cd ~
$ zip -r files.zip tutorial1
$ ls -lh
```

The zip file has been created in your home directory. It should only be a few bytes in size. Now let's delete the original files and directory that we created.

```
$ cd tutorial1
$ rm *.txt
$ cd ..
$ rmdir tutorial1
```

If you try to execute the rmdir command when the directory is not empty, it will fail and give you an error message. However there is a quicker way to delete a directory and all its contents. Try this:

```
$ unzip files.zip
```

to restore the tutorial1 directory and its contents. Use the cd and ls commands to check the files are all present. Now, to delete the tutorial1 directory in one go:

```
$ cd ~
$ rm -rf tutorial1
```

The r flag stands for 'recursive' and f for 'force'. Use this command with caution! In his book *Creativity Inc.*, Ed Catmull describes how most of the graphics from *Toy Story 2* were wiped from Pixar's filestore by careless use of the rm command. Google for `rm toy story 2` to find out the details.

Questions

1. If `cd ..` moves to the parent directory, then what is the parent directory of root `/` ?

2. Why is the following command a *very bad idea?* `rm -rf /`

3. In your bash prompt, what does the ↑ cursor arrow do? What about ↓?

4. Some commands (like `ls` and `rm`) take extra option flags or switches. See if you can find out the switches for the following (using Google or `man`):

   - listing all files in a directory in order of modification, with most recently modified first?
   - listing all files in a directory in order of modification, with most recently modified last?
   - removing a file, with an interactive check where the user has to type `y` if they really want to remove the file?

Further Reading

For further investigation today, use the `man` command to find out the flags for all the commands you have already executed. For instance, you might run `man cat` to Find out how the cat command works in more detail—you can use cat both to print out existing text files, to create new ones and to append to existing files. You press `q` to quit the man page viewer and return to the terminal.

There are plenty of helpful file handling tutorials online, some examples below:

- `https://ryanstutorials.net/linuxtutorial/filemanipulation.php`
- `https://www.redhat.com/sysadmin/navigating-filesystem-linux-terminal`
- `https://www.redhat.com/sysadmin/Linux-file-navigation-commands`