Linux Practical 1
Introduction

This session is designed to introduce you to the underlying UNIX-based Linux file system and computing infrastructure used within the School of GeoSciences and similar to many computing environments worldwide. UNIX-style operating systems are sophisticated and powerful environments for data analysis and processing, and also power much of the internet and world-wide web.

1. Connecting to Linux in GeoSciences (Windows Labs vs. Remote)
- Connecting to the School's XRDP service

2. Exploring the Remote Linux Desktop
- Finding files and programs on the remote server via a (semi-)familiar *Desktop* environment

3. Power Computing: Bash Prompt – The Command Line
- Using the command line interface: universal, quick and powerful.

4. Files and Directories
- Listing, viewing, copying, making. How your workspace is structured.

5. Help!
- How to get it. Is it helpful? Other places to look.

6. File Utilities (Big Data Downloads)
- Useful tools that use the command line - **`tar`**, **`gzip`**, **`ftp`**.

7. Further File Utilities (Zip Files, Automated Download, Web PDFs)
- More useful tools for file-handling and working remotely - **`zip`**/**`unzip`**, **`curl`**, **`wget`**, PDFs.

8. Resource Monitoring Utilities
- Finding out how much disk space you have. Finding other users.

9. Searching
- Finding files. Finding files with a specific bit of text in them.

10. Full Applications
- What are available? How do I start them? Transferring machines: **`ssh`**

11. Log Out/Off or Close Session?
- Full logout versus leaving a session suspended in memory to return to later (from anywhere!)

This practical will allow you to familiarise yourself with the basics of working with Linux in the School of GeoSciences. The practical workbook is designed for you to work through during the practical session in the lab, when there are people around to help. **It is NOT a definitive reference!** Much more information is available on the web, including this document which can be found at:

www.geos.ed.ac.uk/~gisteac/wkzero

Making sense of this Workbook

Throughout this practical workbook any commands you have to type, will be given in **`bold courier font`**, and prompts or responses from the computer will be in `plain courier font`, for example:

```
s1234567@baltic10:~$ pwd
s1234567@baltic10:~$ cd /
s1234567@baltic10:/$ ls -al
```

→ Will indicate tasks to be completed or key learning points

(→) May be used to indicate useful but optional tasks or related info

Specific questions you can answer will have a letter in the left hand margin – a), b), c) etc.

You can record answers or related notes in the spaces provided (if you choose to print these notes), or in e.g. a text file or document see the box below on making notes online.

 This symbol indicates text of high importance or direct usefulness during the practical.
 This symbol indicates further information it would or might be useful to know later.

s*1234657* represents your unique student login (matriculation number preceded by an **s**.)

Note: For staff this may be first initial immediately proceeded by surname up to 8 characters (e.g. jjohnson, jjohnso1, etc.) and for visitors with an initial v*n*fsurna (e.g. v1jjohns).

~ (known as **tilde**) represents your home directory. In the command prompt above note that this will change to reflect whichever directory you are currently working in.

Note that Linux is **case sensitive** - all commands must be typed **exactly** as they are shown on this sheet or they will not work!

How to make notes/record answers during practicals in an online age

While you may print out the practical notes and annotate them, or use the spaces provided to answer any questions, you may wish to work efficiently wholly on-screen. You can arrange multiple windows side by side (2x2 or even up to 4x4) by holding down the Windows key, then using the cursor keys to arrange the windows. This often also works for/on remote systems so you may first need to click **Restore Down**  on the remote window *blue bar* to be able to move it this way. **Note:** Use the lower horizontal *scrollbar* on the remote Linux system to access the whole desktop, or **Maximise** as required. If you need to you can also drag window sides to resize in other configurations (e.g. 3x3).

Better however is to make notes in a text file *directly* on the remote Linux desktop and arrange your program windows side by side entirely *within Linux* – very effective for working! You can use **LibreOffice Writer** from **Applications ► Office** (and can save in Word *.docx* format if preferred) or use another text program. Save to **Home** or **Home ► Documents** to save within your GeoSciences home directory space (M: under Windows).

There follows a set of attainment targets listed below. Make sure you can do everything in the list. Many students **will** be using Linux throughout the year. It is therefore enormously important that you speak to one of the demonstrators if you are confused or have been unable to complete the tasks.

Targets for this Session

You should be able to:

- Connect to the Geosciences XRDP service from a Windows lab PC or remote location
- Know where to find information regarding connecting off-campus/via other platforms
- Locate your Linux home directory
- Change directories
- Create directories
- View simple text files; use simple file viewers (pagers)
- Know how to find quick help on commands/programs
- Download and retrieve data *via FTP or similar mechanisms*
- Monitor disk space used, and find out about other users on the system
- Search for (and through) files; simple pattern matching
- Start full Linux applications and understand about different types of application
- Connect to different machines using ssh
- Log out of a Linux session and out of XRDP

And you should know:

- How your directory system is structured and the notation used
- Where to go for help on Linux (real people and the web!)
- Why you would use Linux
- When to switch off

1 Connecting to Linux in GeoSciences (Windows Labs vs. Remote)

XRDP: The School's Remote Linux Desktop Service

→ The main School-supported mechanism for connecting to GeoSciences Linux systems is by using the School's **XRDP** service. The connection method used is much the same across multiple operating systems/computing platforms and whether working on-site or remotely.

(→) The XRDP service offers a full user-friendly graphical environment with command line options, and allows you to simply disconnect without logging out in order to leave complex software running, with the ability to later resume working exactly as before logging out.

(→) The officially supported connection method on or off campus is made using Windows' **Remote Desktop Connection** client which provides a remote view of the Linux server in the form of a virtual desktop. This allows us to work directly (or *natively*) on a Linux server as if sat directly at the machine, as we would a lab PC. For the full and latest instructions, and further options including connecting Apple Macs and Linux machines to GeoSciences servers, and low bandwidth options, see the information at:
https://blogs.ed.ac.uk/geosciences-it-help/linux-servers/
https://blogs.ed.ac.uk/geosciences-it-help/remote-desktop/

→ We will assume you are connecting from a Windows PC. When using a lab PC (or other PC already within the GeoSciences network) you can of course skip step1 (VPN/eduroam wifi). The following assume you are using Windows' Remote Desktop Connection client.

1. If you are off-campus you must connect to the University's VPN first. (**Note:** If you are on-campus and connected to *eduroam* wifi no VPN connection is required!)
2. Open the **Remote Desktop Connection** client (you can type this in the *search bar*).
3. Click on **Show Options** to expand the drop-down to the bottom of the window
4. Click on the **General** tab. Specify the **Computer:** name as **xrdp.geos.ed.ac.uk**.
5. Enter your **uun** username only (e.g. *s1234567*) **without** any @ed.ac.uk part.
6. Now click **Connect**.
7. Click **Yes** to any certificate popup message (you're already safely within the network!)
8. Finally, ensure your username is again **uun** only (**no** @ed.ac.uk part) then enter the usual **EASE password** at the Linux (XRDP) prompt and click **OK**. You should now be logged into a virtual Linux desktop. If the screen goes blank due to a screen-saver or Linux lock-screen simply nudge or click the mouse to wake the machine up.

Connecting from home; connecting from various operating systems (e.g. Mac)
Apple Mac users can obtain the Windows' Remote Desktop Connection client from the Mac App Store. Linux users can use built-in or similar **RDP** functionality e.g. **Remmina** in much the same way. (Alternative RDP clients can also be used on Windows e.g. **MobaXTerm** which also permits simple **SSH** and **SFTP** connections should VPN connections be problematic.)

Following the relevant instructions below connect now to the XRDP service (we've outlined the steps for connecting from *Windows* in this document directly below).
https://blogs.ed.ac.uk/geosciences-it-help/remote-desktop/windows-to-xrdp/
https://blogs.ed.ac.uk/geosciences-it-help/remote-desktop/macos-to-xrdp/
https://blogs.ed.ac.uk/geosciences-it-help/remote-desktop/linux-to-xrdp/

Problems? Remote Desktop stuck? Use RDP alternative OR text-only SSH
This can sometimes happen with large classes all accessing the system at once, and can happen to any system! If the RDP (Remote Desktop) service temporarily reaches full capacity you may need to wait a little while, or you can start a new basic ***SSH*** command line session rather than have to wait.

Basic SSH functionality is built-into Macs, and also DOS (Windows' *cmd* command prompt – type **cmd** into the search box at the bottom of Windows' screen), though Windows PCs are better served by e.g. **MobaXterm**. *Moba* also contains all of: SSH (command line)+X Server for graphics, simple graphical Secure FTP for file transfer (good in case of any VPN/drive mapping issues) and *an alternative **RDP** client*!

Off-campus? VPN or use ssh.geos gateway first
Off-campus you will need to either start a VPN connection before connecting to **xrdp.geos.ed.ac.uk**, **or** you can make an initial connection without VPN requirement to **ssh.geos.ed.ac.uk**, then once connected you can simply hop onto a full *compute* server by typing **ssh xrdp**. You can then proceed without further delay!

Note: You can use **ssh –X** if you need to run programs that use a graphical display rather than simply running at the command line. This may also be set in *config* settings.

More on SSH Clients (for interacting with a server):
Simple SSH clients can be useful if internet bandwidth or allowances are extremely limited when working remotely. Using a program called an X Server (**ssh –X**) allows graphical programs to be launched as required but via a command line connection.

MobaXterm used this way includes all of a command line terminal, an *X server* for graphical display, and additionally a conventional Windows-like file browser making it a powerful alternative to the remote desktop client. MobaXTerm also offers an SSH-based **Secure FTP** (**SFTP**) client which allows you to connect to a Linux/UNIX server and browse files and folders with a file manager. Critically however it allows you to open and edit files using Windows' programs but, *unlike other mechanisms – e.g. samba*, it will edit files *as if working natively* on the remote Linux machine, respecting file ownership/access permissions and other settings. Very useful for development work!

MobaXterm can also work as a replacement Remote Desktop client (use **RDP** rather than *SSH*) instead of Windows' own client, however it is easier to stick with the officially supported/more flexible Windows' **Remote Desktop Connection** for this purpose.

2 Exploring the Remote Linux Desktop

The MATE Desktop Environment

Shown above is the current *Ubuntu Linux* based **MATE** environment provided upon logging into the GeoSciences XRDP service. This consists of a virtual desktop (actually four *workspaces*, i.e. screens, are available as usual with such systems – and comparable to modern smartphones), plus some shortcut *icons* and a system of menus to the top-left of the screen. A clock is shown to the top-right which expands to a calendar once clicked.

→ Note a number of useful features. Like all computer operating systems files are organised in sets of folders, or in Linux terms *directories*. In a Linux system each user has a default folder – their home directory.

→ On the desktop is a shortcut icon to your home directory. Clicking on this opens the **Caja** file manager *at* (or first located *in*) your own home directory.

→ You can see the entire network by clicking on **File System**. Note that some directories are not visible by default and need to be accessed via the command line or a search bar first (use **Ctrl+L** – i.e. hold **Ctrl** and press **L** – to specify a file location to browse) – most commonly these are the directories on the GeoSciences web server (those in **/web**)!

→ A great number of programs and tools can be found from the menus. The top-left **Applications** menu is particularly useful. The right-hand **System** menu is the one for logging out. Beside the menus are a selection of icons including one for the open-source **Firefox** web browser, and another shortcut to the **Caja** file manager – shown as a filing cabinet. The most useful icon for us though is the middle black box one! Therefore **left**-click now on this **Terminal** icon (or go to **Accessories ► System Tools ► MATE Terminal**).

(→) **Colours:** We will make extensive use of the command line during this session. Historically terminal windows tend to bave a black background with white text which may be good for easing eye strain however regular breaks are recommended with computers. A white background may be better for displaying on a projector screen. You can control colours within the terminal window should you wish from **Edit ► Profile Prefefences ► Colours**.

→ We will now look at using the command line interface to demonstrate the power of Linux. Many of you will make use of the most common of these commands to manage your work during taught classes or research. If you have not got a Terminal window to appear, or are not happy at this point – please ask for help!

3 Power Computing: Bash Prompt – The Command Line

The Bash (Shell) Prompt

→ When you open up a Terminal window you should see a largely blank window apart from a semi-cryptic bit of text known as the *bash prompt*. The command line is known in Linux terminology as a *shell* (sh for short) and for historical reasons the default shell commonly used is called bash*. The prompt shown awaits your command(s), which you should type "*at*" (i.e. just after) it. The prompt here also gives some helpful information to help you.

```
s1234567@server:~$
```

→ The prompt consists of your username (your *uun*), shown at (@) the *host* or ***server*** name (i.e. the name of the server you are logged into). Currently **xrdp** is as an alias for the server **baltic10** however it could easily be any School server such as the *compute* server **stream**.)

→ Far more usefully, the tilde (**~**) tells you that you are currently located in your home directory (just as with the file manager earlier). This directory also has a full form *address* (or *path*) which we will ask you to note in the next section in order to bolster your Linux knowledge!

(→) **Note:** For School postgraduate students, this home directory, or *homedir*, is normally the same space that also appears in Windows as **M:** (In Windows this Linux network location is *mapped* to the Windows M: drive using a software technology known as Samba.)

(→) **Note:** *If* your prompt contains a server name other than `baltic10` then you can type **`ssh xrdp`** to ensure you are on a suitable server. E.g. the address **`ssh.geos.ed.ac.uk`** points to a very basic ssh gateway server from which you will need to then connect to a *full* server.

Space – the frontier between ALL commands! Use spaces!
A key point before we proceed is that computers (despite recent advances in so-called Machine Learning) are really rather dumb and thus **all commands** and any other parameters (known as *arguments* or *flags*) **must be** specified **separated by spaces** so that the computer has a chance to identify where one command or argument ends and the next begins. In these, and other exercises, if it looks like a space then it probably is one and should be typed as such! **Note:** This is different from e.g. web browsers which simply render text as instructed using special mark-up 'tags' and which otherwise ignore spaces ("whitespace")! When working at any command line however you ***need*** spaces!

***bash? Why?**
In case you are curious, there have been various shells over the years, but the most common is a variant of the Bourne shell called the Bourne Again Shell (or bash for short!).

4 Files and Directories

Listing Files and Directories

a) One of the most useful commands in Linux is **`pwd`** - **p**rint **w**orking **d**irectory. It tells you where you are in the directory structure, and the full ***path*** to that location. On School systems the prompt already does this for you however it is worth knowing this command. **Note:** In Linux/UNNIX (and in fact in DOS, i.e. anywhere outside of Windows Explorer!) folders are known as ***directories***. A directory is actually a file; a list of groups of other files.

To use the **`pwd`** command simply type it as is at the bash prompt and press **`Return`** or **`Enter`**: This command takes no following arguments (further control information or parameters) and therefore no spaces are required here.

```
s1234567@baltic10:~$ pwd
```

What is the address of (*path* to) your home directory? (Either note this here, or in a text file).

……

……

Be aware that drive letters such as **M:** (or **P:**, **G:**, **U:** etc.) are a Windows-only feature and that when using Linux you will need to reference your home directory as /home/s1234567.

Fortunately the Linux desktop environment provides convenient shortcuts to your home directory and each time you start a Terminal session you are logged 'into' (i.e. located within) your home directory to start with automatically.

b) You will also need to know what files and directories there are within your home directory. The command for this is **`ls`** (i.e. list). Try it now. Are there any files in your home directory?

……

……

→ **`ls`** is a command which can optionally take a following **argument** (separated from the command by a space). Normally this is a folder path to list the contents of, and the output from it can be changed further by the use of extra control **flags** extra argument(s) in such cases. Conventionally flags are specified first. For examples of these we shall explore the **wkzero** directory held on our main shared teaching resource, ***netdata***.

```
/geos/netdata/wkzero
```

→ Look at the example below and try it yourself. Notice it takes a single argument (the path to a directory location) and so a single space is required *between* the command (**`ls`**) and argument (**`/geos/netdata/wkzero`**).

→
```
s1234567@baltic10:~$ ls /geos/netdata/wkzero
demodir  if.txt  jabberwock.txt  nation_data.txt  protocols  xyzpoints.txt
```

→ Now let's try a more advanced version with two spaces required; either side of the **-al** flags.

→
```
s1234567@baltic10:~$ ls -al /geos/netdata/wkzero
total 68
drwxr-xr-x  3 omacdona netdata  4096 Sep 14  2010 .
drwxrwsr-x 54 omacdona netdata  4096 May 29  17:10 ..
drwxr-xr-x  3 omacdona netdata  4096 Sep  7  2011 demodir
-rwxr-xr-x  1 omacdona netdata  1734 Sep  7  2011 if.txt
-rwxr-xr-x  1 omacdona netdata  1353 Sep  7  2011 jabberwock.txt
-rwxr-xr-x  1 omacdona netdata  8296 Sep  7  2011 nation_data.txt
drwxrwxr-x  2 omacdona netdata  4096 Sep 12  2016 protocols
-rwxr-xr-x  1 omacdona netdata 34965 Sep  7  2011 xyzpoints.txt
```

→ The first version used **ls** with an **argument** only (the *address of* or *path to* a directory), the command means "list the files that are in the directory **/geos/netdata/wkzero**".

→ The second version uses both an **argument** and **flags**. The flags are **-a** and **-l** (you can stick flags together as above **-al**). **ls -a** means show all the files, including hidden ones (hidden filenames in Linux start with a **.** character). **ls -l** means give **l**onger (more) information on each file (or l for long-form perhaps).

→ In the example above you should note the two extra entries **.** and **..** . A single **.** means "the current directory" in this case **/geos/netdata/wkzero**. A double **..** means the directory above this one – "the parent directory" or in this case **/geos/netdata**. We will come back to this notation later on…

→ So **ls -al /geos/netdata/wkzero** means "show all the files in the directory **/geos/netdata/wkzero** and give a long description of them".

In the long description you can see the read and write permissions of the files, what type of file (file or directory) they are, who owns them, how big they are etc. For example:

```
drwxrwxr-x  3 omacdona netdata    2 Dec 10  2020 demodir
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63
```

→ The explanation for this follows:

Column 1: the **d** in this column shows that **this file is a directory** – normal files have a **-** symbol here.
Columns 2-10 show the **permissions** set up for this file – see the full explanation below
Column 13 shows the number of links to the file (you need not worry about this)
Columns 15-22 record the user name of the **owner** of this file – **omacdona**
Columns 24-30 record the Linux **group** (of users) to which the file belongs – **netdata**
Columns 34-37 show the **size** of this file in bytes or characters
Columns 39-50 show the **date** and, if recent, the time this file was **last modified**
Columns 52-57 show the **name** of this file – **wkzero**

Columns 2, 3, and 4 and 5, 6, and 7 respectively show that the file *owner* and the Linux *group* both have permission to **r**ead, **w**rite/delete, and e**x**ecute (run commands on) this file. Columns 8, 9, and 10 show that all **other** users only have permission to **r**ead and e**x**ecute for this file. They cannot alter it or delete it. We commonly talk of "world read" but in reality this only extends to other users already securely logged in to the system – not like a website!

c) Using the commands above, find out if there are any files in **`/geos/netdata/wkzero/demodir`** and if there are, note their name, owner, and size – or at the very least, note the command used to find this information. Once again you can use the space below to record these if using printed notes, or put your answers in a text file.

……

……

……

……

Simple Viewing of Files

d) Viewing simple text files is easy – you can use one of these 3 *pager* commands (and others!):

`more` ***`filename`*** displays ***`filename`*** one page at a time
`head` ***`filename`*** displays the first few lines of ***`filename`***
`tail` ***`filename`*** displays the last few lines of ***`filename`***

You should have found a file called **`if.txt`** in **`/geos/netdata/wkzero/`**. What are the first few lines? Or if you are in a hurry, just tell us who the author of the file is!

(**Hint:** A filename is itself just a path to some data so when referencing files outwith the current folder (as we are doing here) the full path – including filename – should be specified and *passed* to the command as its argument – e.g. if entering a command to act on a file it would take the form below (*we'll shortly show you an invaluable shortcut to save typing!*): ***my_chosen_command /top-folder/sub-folder1/sub-folder2/…/myfile.txt***.)

……

……

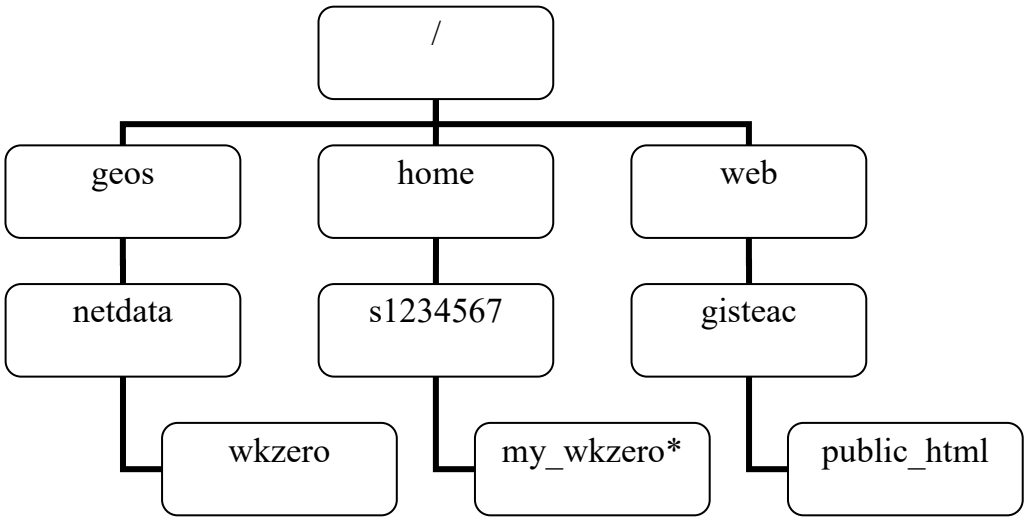
……

……

As well as `more` there is actually also a similar command `less` which is usually set as the default text viewing (pager) program on the version of Linux used within the School! It works mostly the same way but requires you to press **`q`** to quit once you are finished reading. This prevents you scrolling to the end of the file and having the text viewer program close automatically as can happen with `more`.

Linux/UNIX Filesystem Structure (Example)

→



→ The above diagram shows a simplified extract of the GeoSciences Linux filesystem. At the top (of the *inverted tree* structure) is the home directory of the root (or super) user – this is the administrator or filesystem owner. Following along the branches from this 'root' are various *folders* and *subfolders*, or *directories* in a non-graphical command line context.

→ You should recognise the shared directory/folder *netdata* as part of this with one of its subdirs/subfolders *wkzero* shown which you used earlier. The other *my_wkzero** folder marked with an asterisk (*) is meant to signify one lying within your own home directory – *we haven't actually created this yet but in fact will get you to create this next!*

(→) **Note:** The user **gisteac** (note there is no '**h**'!) is a very useful user to remember for those taking GIS/EO courses in GeoSciences. In particular, shown here are gisteac's web folders at `/web/gisteac/public_html` which contain much useful info relating to geospatial computing. Any publicly accessible info stored here can be reached *over the web* as per the URL given at the very top of this practical, i.e. at **www.geos.ed.ac.uk/~gisteac**. **Note** the special use of the tilde character (**~**) here to signify the main (home) web directory of a user.

Creating and Changing Directories; Deleting Files or Directories

→ Managing directories and files is an important aspect of working with Linux (or indeed any operating system). You only have a limited amount of disk space yet putting all your files together in one directory is only going to lead to large amounts of wasted time searching for previous work! Making a directory is easy – use the command **`mkdir`**. For example:

→ `s1234567@baltic10:~$` **`mkdir my_wkzero`**

will create a new directory called **`my_wkzero`** – **Note:** it will create this in ***your*** current directory, if your prompt matches the form above. Logically we could call this `wkzero` to ensure that any replicated file paths match exactly, however for today we will try to make things ultra-clear! Remember the need for spaces to separate commands and arguments.

e) Using **`mkdir`** create some other new directories to put *your* work in. How about **`webpages`** and **`papers`** for starters? Once you have made these new directories, check they exist using **`ls`**. Can you record how below – what is the specific variant of **`ls`** needed?

……

→ Having other directories is only useful if we can move to them. To change directory use **`cd`**.

→ `s1234567@baltic10:~$` **`cd my_wkzero`**
→ `s1234567@baltic10:~/my_wkzero$` **`pwd`**
`/home/s1234567/my_wkzero`
→ `s1234567@baltic10:~/my_wkzero$` **`cd ..`**
→ `s1234567@baltic10:~$` **`pwd`**
`/home/s1234567`

→ Remember the **`..`** notation from before? Here it is *used* (i.e. in the command) to change to the parent directory of the directory that you are currently in (i.e. to move to one level above).

→ **Note: Also notice how the prompt reflects the current directory and changes from ~ to `my_wkzero` and then back again in the above example. Keep an eye on this prompt changing and use this feature during the practical to ensure that you are issuing commands from within the *correct* folder! I.e. your prompt should always match the one given in the worksheets at each stage. *Ubuntu seems to have rather full prompts!***

f) Now use the **`cd`** command to *change **back*** into *your* **`my_wkzero`** directory and add two more new subdirectories there (i.e. create these *within* **`my_wkzero`**!) called **`data`** and **`docs`**. You can use **`ls`** to check everything is OK. Note the **`cd`** and **`mkdir`** commands used below.

……

……

……

!→ If a command fails, first check you are in **YOUR `my_wkzero`** folder and not our reference version `wkzero` on read-only **netdata**! (I.e. You should have *your username* in the path, and **not** `netdata`!) Use **`pwd`** to check, or **`cd ~/my_wkzero`** to go to the right place!

Note: You can create directories in your Linux home directory from within Windows using Windows' File Explorer. Your home directory usually appears in Windows, mapped to *M:*\ for GeoSciences postgraduates, or perhaps to another letter (usually **T:**) for undergraduates as required; useful knowledge for anyone demonstrating on UG courses.

When using Windows to access Linux network drives – **be careful!** Windows lets you put spaces in your directory names. Linux is happy enough with this, but it can make moving around in directories within Linux more awkward since you may need to enclose the directory names in quote marks **" "**. You may also find some software will object to paths with spaces in their names; much better to use an underscore **_** or hyphen **-** instead.

→ To delete a file in Linux you use the **`rm`** command. For a directory you use the **`rmdir`** command. **Be very careful when using `rm`! Once a file is gone, it is gone forever!** It is much better to **always use `rm` with the `-i` flag**. This means that Linux will ask you to confirm the deletion of a file or directory. Much safer… **Remember** that with *network* drives, (e.g. Windows' M: drive), the same rules apply – it will not go to the **Recycle Bin**, it will be gone forever! It may be best to simply mark files for deletion *in Linux* later (see *links* box!).

→ Use **`rmdir`** to delete one of your new directories and **`mkdir`** to recreate it.

Symbolic Links; Group Projects/Shared Files – Beware…
Be careful if using Linux *links* (known more fully as *symbolic links* or *symlinks*) which will appear simply as folders under Windows. If you are collaborating with several colleagues you should always use **`rm`** or, better still, **`unlink`** to remove any unwanted links so as not to accidentally delete any linked files or sub-folders which you have write access to! Deleting symlinks directly from Windows could be rather risky to say the least!! Links can be shown in cyan (light blue) when listed in a Terminal window. The long form of the ls command **`ls -l`** will also reveal these, shown *linkname -> linked_destination*.

Sneaky (useful!) tips:

1) Using the up arrow button on the keyboard will scroll through the commands you have typed before so you can re-use them… You can also use cursor (arrow) keys for editing.

2) In a similar vein, the Terminal offers a copy and paste facility even across operating systems (i.e. between Linux and Windows and vice versa). Either go to the **Edit** menu and **Copy** and **Paste** as required (or **`Shift+Ctrl+C`** for copy and **`Shift+Ctrl+V`** to paste, **note:** **`Shift`** key only required in a Linux *Terminal*. **Note:** Some clients are even more rapid allowing you to simply highlight text to copy, and **right**-click to paste. If you highlight past the end of a line a 'newline' is also included and any text (e.g. commands) submitted to the computer with one click! For `Ctrl` on Macs, use **`Cmd`**.

3) If you are typing a file or path name, you can use auto-completion with the **Tab** key usually shown on the keyboard itself as **`->|`**. Don't spend ages typing!!! Instead, hit **Tab**!

4) You can use a handy version of **`rm`** to delete any files and folders safely and in one operation. Use **`rm -ir`** (or **`rm -iR`**). The recursive –r flag allows deletion within folders.

Copying and Moving Files and Directories

→ You can copy files and directories not only within your own working area (i.e. in your home directory, or any directories below that in the hierarchy) but also from other users' directories so long as *you have the correct permissions*.

→ To **copy** a file or directory, use the command **`cp`**. For example, *once again* check you are in YOUR **`my_wkzero`** directory – check the prompt!, then type both of the following commands: **(Remember:** when we say *type*, also use the **tab auto-complete** function to ease the pain!!)

→ `s1234567@baltic10:~/my_wkzero$` **`cp /geos/netdata/wkzero/if.txt .`**
[NOTE THE SINGLE DOT AT THE END, SEPARATED BY A SPACE!!]

→ `s1234567@baltic10:~/my_wkzero$` **`cp /geos/netdata/wkzero/if.txt 2nd_if.txt`**

g) But what do they do? What has happened? *Use **`ls`** to list all the files in your directory and find out.* So what does each command above do? Be sure and ask if not clear!

……

……

……

……

→ Copying a whole directory is very similar – you just add a flag **`-R`** (or `-r`) which stands for **recurse** the directory i.e. look at all the files within the directory and copy them as well, keeping the directory structure the same as that of the original. As an example, try the following (again use **tab auto-complete** to save typing:

→ `s1234567@baltic10:~/my_wkzero$` **`cp -R /geos/netdata/wkzero geos_wkz_copy`**

h) Then use **`ls`** again. What has happened? Again, ask if not sure before moving on!

……

……

→ **Moving** and **renaming** a file or directory are the same thing in Linux and the command used is **`mv`**. If you rename a directory then the ***path*** (or address) of all the files within that directory will move as well. Try the example below:

```
→ s1234567@baltic10:~/my_wkzero$ ls geos_wkz_copy/
  demodir if.txt  jabberwock.txt  nation_data.txt  protocols  xyzpoints.txt
→ s1234567@baltic10:~/my_wkzero$ mv geos_wkz_copy/ not_wanted_now
```

→ You have now moved (renamed in this case) the directory so the following command fails:

```
→ s1234567@baltic10:~/my_wkzero$ ls geos_wkz_copy/
  geos_wkz_copy/: No such file or directory
```

→ You can also move groups of files around within your directory structure. Having renamed **`geos_wkz_copy`** we should move the files out of it to somewhere more logical: So:

```
→ s1234567@baltic10:~/my_wkzero$ mv not_wanted_now/*.txt docs/
→ s1234567@baltic10:~/my_wkzero$ ls docs/
  if.txt  jabberwock.txt  nation_data.txt  xyzpoints.txt
```

→ The **`*`** character is a **wildcard** symbol that means "all characters".

(→) **Note:** The final / is often not required, it simply indicates clearly that we are referencing a directory. Therefore the following will work just as well:

```
→ s1234567@baltic10:~/my_wkzero$ ls docs
  if.txt  jabberwock.txt  nation_data.txt  xyzpoints.txt
```

→ To keep your home directory (and thus **M:**) tidy and well-organised you should **mov**e from yesterday your **`PCInduction`** folder *IF (AND ONLY IF) YOU CREATED IT!!* (it should be in your uppermost top-level ***home*** directory *if you created it*). Logically, you should move this into *your* **`my_wkzero`** folder. This covers the important learning point of good data and folder management and also serves to give you more command line Linux practice. We can do this as follows:

```
→ s1234567@baltic10:~/my_wkzero$ mv ../PCInduction PCInduction
```

→ If you did not create this folder yesterday, no need to worry but you will get an error message to say the folder does not exist. **Note:** We could also have issued any of these alternatives:

```
s1234567@baltic10:~/my_wkzero$ mv ../PCInduction .
s1234567@baltic10:~/my_wkzero$ mv ~/PCInduction PCInduction
s1234567@baltic10:~/my_wkzero$ mv ~/PCInduction .
```

→ The tilde character (**~**) is a short-hand reference for your own home directory. Thus we can more usefully write this command as:

```
→ s1234567@baltic10:~/my_wkzero$ mv ~/PCInduction ~/my_wkzero/PCInduction
```

!→ **Note:** This last command can thus be issued from ***anywhere*** within the directory structure!

5 Help!

→ Help in Linux can be a little unfriendly until you get used to it! There are lots of sites on the web that will be clearer and easier to follow for beginners. You should, however, know how to get help for specific commands, and once you get used to the syntax and structure of the pages, the help system might even start to grow on you…

→ Help is accessed with the **man** (short for *manual*) command and the help pages are exactly that – digital *pages* of instructions. The structure is **man <command>** where **<command>** is the name of the command you want help with. Angle brackets <>: are often used in notes either for specific keypresses/buttons, or to show a user-specified and thus variable textual instruction. (Optional parameters are often shown further enclosed in square [] brackets).

→ For now try:

→ `s1234567@baltic10:~/my_wkzero$ man man`

→ Press **Space**, **Enter**, or the cursor (arrow) keys, to *scroll* through the content of the manual pages, and **q** to quit or exit. This should give you a better idea of how to use help! Notice that the help system uses the default text viewer, **less**, described earlier.

Another useful feature of man involves the **-k** flag. This allows you to specify a keyword, which man will look for in the help pages. Try:

→ `s1234567@baltic10:~/my_wkzero$ man -k transfer`

i) This will return quite a lot of possible commands – lurking in the list is one we will use next – **ftp**. What does the **man** page for this say it is for?

……

……

6 File Utilities (Big Data Downloads)

FTP (File Transfer Program/Protocol)

→ FTP is a long-standing way of transferring files from a remote computer/site – either a standalone PC, workstation, or server, to your machine. The following example uses a demonstration FTP site set up for this practical. Due to the simple open nature of FTP many sites will now require Secure FTP (sometimes known as SFTP) or will provide a more involved (registration probably required), but more modern, web interface. FTP is still a useful fundamental tool to learn though, and still used by many data providers in some form.

→ Type the following (when asked for a user name type **anonymous** and in line with 'anonymous ftp' convention, enter your ***email address*** when asked for your password):

→ `s1234567@baltic10:~/my_wkzero$ **cd data**`
→ `s1234567@baltic10:~/my_wkzero/data$ **ftp**`
→ `ftp> **open ftp.ed.ac.uk**`

```
Connected to ftp1.is.ed.ac.uk.
220-University of Edinburgh Anonymous FTP server
 ============================================================================

  When requested for a username enter 'ftp' or 'anonymous'.  If you have
  problems, try using a dash (-) as the first character of your password.
  If you still have problems or wish to make a comment then send email to
  ftpmaster@ed.ac.uk.

  All transfers are logged.
220 FTP Server
```

→ `Name (ftp.ed.ac.uk:s1234567): **anonymous**`

→ You should get a response like this:

```
331 Anonymous login ok, send your complete email address as your password
```

→ `Password: **s1234567@sms.ed.ac.uk**`

```
230-This service is managed by Information Services.  It holds information
 which may be useful to system managers and space is provided for
 individuals and groups upon request.  Upload facilities are also
 available.  Anyone can make use of this service.
.
.
.
230 Anonymous access granted, restrictions apply
Remote system type is UNIX.
Using binary mode to transfer files.
```

→ You are now connected to the Edinburgh FTP server – **ftp.geos.ed.ac.uk** - and have limited rights to list and download files. Remember that you are now a user of the FTP application; it has its own set of commands (though they are very similar to Linux commands).

→ Type the following commands:

→ `ftp> ls`

```
229 Entering Extended Passive Mode (||||54135|)
150 Opening ASCII mode data connection for file list
drwx-wx-wx   2 root     root         4096 Sep  7 02:00 edupload
drwx-wx-wx   2 root     root         8192 Sep  6 12:22 incoming
lrwxrwxrwx   1 root      root              8 Nov 24   2020 INSTRUCTIONS-FOR-USING-
THIS-SERVICE -> .message
drwxr-xr-x  35 root     root         4096 May 20  2020 pub
226 Transfer complete
```

→ `ftp> cd pub/geos`

```
250 CWD command successful
```

→ `ftp> ls`

```
229 Entering Extended Passive Mode (||||54275|)
150 Opening ASCII mode data connection for file list
drwxr-xr-x   2 309643   root           30 Sep 10  2008 wkzero
226 Transfer complete
```

→ `ftp> cd wkzero`

```
250 CWD command successful
```

→ `ftp> ls`

```
229 Entering Extended Passive Mode (||||54353|)
150 Opening ASCII mode data connection for file list
-r--r--r--   1 root     daemon        365 Sep 10  2008 jefferson.tar.gz
226 Transfer complete
```

→ **The commands you have typed above ONLY apply to the remote system – you are still located in the same directory on your local Linux server**. The command sequence above shows you some of the files you can download. Often you will first have to set the **transfer type**. You are going to download the file **`jefferson.tar.gz`** which is a compressed binary file. So just to be sure type:

→ `ftp> type binary`

```
200 Type set to I
```

→ Then, use the **`get`** command to begin transfer (since this is a file path you can use tab here):

→ `ftp> get jefferson.tar.gz`

```
local: jefferson.tar.gz remote: jefferson.tar.gz
229 Entering Extended Passive Mode (|||54290|)
150 Opening BINARY mode data connection for jefferson.tar.gz (365 bytes)
100% |**********************************|   365        7.91 MiB/s    00:00 ETA
226 Transfer complete
365 bytes received in 00:00 (717.19 KiB/s)
```

→ Then, to log off from FTP and close the network connection:

→ `ftp> bye`

```
221 Goodbye.
```

→ You should now have your own copy of the file **`jefferson.tar.gz`** transferred from the FTP Server running the Edinburgh FTP Service. While modern day FTP servers will often allow you to obtain files via a web browser, the next part is key knowledge for Linux users.

(→) **FTP problems?**
If you have FTP issues you can obtain **jefferson.tar.gz** from the **wkzero** web pages. Save this file with Firefox, available in XRDP. It will save to e.g. `/home/s1234567/Downloads` but you can then use the file manager **Caja** simply **right**-click and **copy**/**paste**, or use **`Ctrl+C`** and **`Ctrl+V`**. Alternatively you could practice using the **`mv`** or **`cp`** commands! Or, simply download *remotely* in Windows as normal (e.g. saving to **M:\my_wkzero\data**).

GZIP (GunZip)

→ So you've got a file that looks like it might have some data in it. But what kind of a file is it? And how do you get at the data? As is often the case for Linux data files, this 'file' is actually a *collection* of files specially encoded as one to save space and then compressed further to save disk space (e.g. when dealing with high-res images or large datasets). Unpacking this is thus a two stage process that will introduce you to two more useful Linux utilities.

→ First make sure the file **jefferson.tar.gz** (**mv** if necessary) is in *your* **my_wkzero/data/** as it should be. **Note:** You can launch applications (e.g. ftp) from any chosen destination directory, in this case we used the **data** directory, to make life easier once finished with FTP. Make sure you are in the **data** directory now.

→ The file **jefferson.tar.gz** has been compressed using the **GunZip** utility. This allows you to compress files so that they take up less disk space (and transfer faster across FTP). To look at the data, we first have to reverse this compression, using the command **gzip**:

→ `s1234567@baltic10:~/my_wkzero/data$ ls -l`
`total 1`
`-rw-rw-r-- 1 s1234567 s1234567 365 Sep 11 12:00 jefferson.tar.gz`
→ `s1234567@baltic10:~/my_wkzero/data$ gzip -d jefferson.tar.gz`
→ `s1234567@baltic10:~/my_wkzero/data$ ls -l`
`total 10`
`-rw-rw-r-- 1 s1234567 s1234567 10240 Sep 11 12:00 jefferson.tar`

→ Note what has happened. The **-d** flag told GunZip to decompress the file, so it loses its extension and gets bigger. You can see which flags to use to compress files by typing **gzip -h** (h for help).

→ Now you have uncompressed the file, but it is still **archived – it is one file that contains many other files**. You thus need another utility to get the archived contents out again.

TAR (Archiving Tool)

→ To get the files back out from their archived form, you need to use the **tar** command. The instructions below are a **very simple example** of using **tar** to extract files, you can do much much more with it than this!

→ To extract the files, type the following:

→ `s1234567@baltic10:~/my_wkzero/data$ tar -xvf jefferson.tar`
`jefferson.txt`

→ Ok in this case there is only one file but usually you will have many files – e.g. datasets, images, etc. (The name tar originally stood for tape archive/archiving, involving many files.)

→ The flags:
-x Extract files from the archive.
-v Verbose- provide feedback to the user.
-f The file to extract from will be specified by the user and is the next argument.

7 Further File Utilities (Zip Files, Automated Download, Web PDFs)

Working with Windows .zip files; Minimising Upload/Download

→ While the UNIX gz format is common in the scientific and computer programming worlds, often there is a need or preference for the, *possibly more common*, Windows zip format. On the School Linux servers there are therefore tools to work with files in this format. Try:

→ `s1234567@baltic10:~/my_wkzero/data$` **`zip jefferson.zip jefferson.txt`**

→ This would compress `jefferson.txt` to become `jefferson.zip`. **Note:** The command requires the output zipped file to be specified first!

(→) You could also compress a whole folder of files in the same way, or specify a set of files to zip into one compressed file. This makes it much easier to distribute multiple files intended for use by Windows users without having to download all the files to your local Windows PC, zip up and re-upload the compressed zip file to a web server. You can simply do the work on the server and move the result directly to the School web server all in Linux. This is handy if working with limited internet where only a command line Terminal connection is available, or perhaps if broadband allowances or speeds are low and files are big in size.

→ Now let's decompress the zipped file to a *new copy of the file in a new folder* `jefferson2`.

→ `s1234567@baltic10:~/my_wkzero/data$` **`unzip jefferson.zip –d jefferson2`**

→ Alternatively you can leave out the folder specification and will be asked whether you wish to overwrite any existing files. You can find more information by simply running zip with no arguments specified (this is commonly the case with Linux/UNIX commands).

Remote/Automated File and Web Page Retrieval (e.g. via WWW)

→ Instead of obtaining files or datasets from special file-servers via FTP, you may have to retrieve files via the more conventional world-wide web (WWW) and save them to your GeoSciences home directory. If you wish to automate downloads or are working with very limited internet connectivity then you can use special Linux command line tools to allow the GeoSciences server to download these files for you. You can then issue other commands via Linux to perform processing or analysis on these and obtain results *leaving your broadband allowance or internet bandwidth unaffected*, *or to automate the entire process*.

Introducing `curl` and `wget`

→ The **`curl`** and **`wget`** commands can be used to retrieve files from any web URL, e.g. with **`curl`**, [**Note:** Type these all on one line, use the up arrow to edit the (second) command.]:

→ `s1234567@baltic10:~/my_wkzero/data$` **`curl https://www.geos.ed.ac.uk/~gisteac/wkzero/protocols.txt -o protocols.txt`**

→ `s1234567@baltic10:~/my_wkzero/data$` **`curl https://www.geos.ed.ac.uk/~gisteac/wkzero/protocols.html -o protocols.html`**

(→) If you omit everything from the output –o flag onwards the file contents will just be displayed.

→ The **wget** command functions in a similar way. Each command has both advantages and disadvantages over the other, but **wget** also offers a useful *recursive* facility where it is possible to download not only a web page, but other pages or links contained within it, e.g, [**Note:** Again, type this all on one continuous line]:

→ `s1234567@baltic10:~/my_wkzero/data$` **`wget http://www.geos.ed.ac.uk/~gisteac/wkzero/protocols_all.html -r`**

→ Notice the **-r** at the end of the line. Use **ls** and **more** to see what you get from the above commands! Do you really get a directory called www.geos.ed.ac.uk? Try **ls** and check!

(→) **Note:** It is also possible to use these commands non-interactively in pre-written script files to *automate* file retrieval across the web (via HTTP/S), or by FTP as we used earlier, or in fact by a number of other data transfer protocols including Secure FTP (SFTP). This also means you can set a large download going and log out (we'll see more on this later). The ability to automate such tasks is a major strength of these tools and of sophisticated operating systems such as Linux or UNIX. Windows does offer some such sophisticated tools but is still catching up in many respects and arguably does not cater so well for multiple users (despite its everyday prevalence!)

(→) Be careful to use **wget** (and **curl**) responsibly. You may not be able to use all functions of the command without having first been given the appropriate access to a website! Also, if you try to bulk save people's websites en masse too often they may well complain!

Working with PDFs remotely (For WWW-based PDF distribution)

There are an unlimited number of tools that you may wish to use to analyse large datasets downloaded to your GeoSciences home directory, or that you might wish to run on the School servers. As a slight change however let's look at a useful tool for working with PDF files, a very common method of distributing information over the web either for local colleagues (saving clogging email inboxes) or for distribution of information world-wide.

→ While saving or printing to PDFs is now common place, often you may require to join multiple PDFs together into one document, or to separate a single PDF into multiple files. E.g. you may wish to stitch together pages scanned individually on a School or University multi-function scanner-copier machine, or perhaps scan a selection of documents in one go then split into separate documents later.

→ You perform such tasks with **pdfunite** and **pdfseparate**. Here is an example of using **pdfunite** to merge a set of PDF files, from **netdata**, into a single PDF in your current directory:

→ `s1234567@baltic10:~/my_wkzero/data$` **`pdfunite /geos/netdata/wkzero/protocols/*.pdf protocols.pdf`**

(→) Don't worry if you don't fully understand this just now – do ask though!! – the point is to know you can do it and to have a document (this one!) of where to find the tools. You can review the output of this from a file manager window and/or by running a PDF reader.

(→) Sample joined output (and then the result of this re-split by pdfseparate) can be seen at **/geos/netdata/wkzero/protocols/joined_then_separated**. You can use the Caja file manager and a suitable PDF reader to view these.

8 Resource Monitoring Utilities

→ Despite the ability to download large datasets automatically, you only have a limited amount of disk space on the Linux system. You can keep a close eye on how much disk space you are using with the command **du**. Some examples follow but see **man du** for fuller details.

→ `s1234567@baltic10:~/my_wkzero/data$ du -sk`

will give you the total amount in kilobytes of disk space used in the current directory which should still be `data` unless you have changed this yourself! (Running this command in your home directory will tell you your **total** disk usage.)

→ Replacing the `k` with an **h** will give you a more human-readable form:

→ `s1234567@baltic10:~/my_wkzero/data$ du –sh`

→ Alternatively you can use SI units (where 1 Kb = 1000 bytes instead of 1024 bytes). This may help better manage data volumes as you will appear to have more MBs or GBs etc!!

→ `s1234567@baltic10:~/my_wkzero/data$ du –s --si`

→ Note that this uses a secondary (sub) flag signified by a double-hyphen.

→ Finally, running the following command in your *home* directory will give you a breakdown of the total space used by each file and directory located there, hence type:

→ `s1234567@baltic10:~/my_wkzero/data$ cd ~`
→ `s1234567@baltic10:~$ du -sh *`

→ Alternatively you can achieve the same effect (all your files and folders) but from *anywhere* in the filesystem as follows:

→ `s1234567@baltic10:~$ du -sh ~/*`

j) Using **du**, you can find out how much storage space you are using and note it here:

……

……

→ As well as checking disk space, you may wish to know about available computer memory or running processes. Commands such as **top** (listing the top resource-consuming tasks or applications) and **ps** (for controlling running processes, i.e. programs) are used for such tasks. We'll look at these in detail in the advanced session that follows.

(→) **Miscellaneous Admin Commands**
If you are tasked with managing a system it may also be useful to know **who** is logged in! In a privacy-concerned "GDPR" age however, any further details will be securely held in a proper system well away from the average user (sysadmin access only!). You can however use a related command if you forget who you are – **whoami**. Finally, **hostname** is also useful on occasions when the command prompt changes upon running another application.

9 Searching

k) You can search for all sorts of things in Linux – usually it will be files or bits of text. The commands you would normally use are **grep** and **find**. But what do the commands do? Time for you to do some work. Use **man** to find out what **grep** and **find** actually do…

……

……

……

……

……

……

……

……

……

……

……

l) Found out? Try these commands - what do they do?

```
s1234567@baltic10:~$ find /geos/netdata/wkzero -name "*.txt"
s1234567@baltic10:~$ grep -n 'stan' /geos/netdata/wkzero/nation_data.txt
```

……

……

……

……

……

10 Full Applications

→ The way you interact with Linux/UNIX applications can vary. Some will operate simply at the command line in one go, sometimes interactively, others may take over the Terminal window with a text-based menu or *screen* system, others still may offer full-blown graphical capabilities. While you can open multiple Terminal windows easily, it is still worth knowing the fundamental principles should you ever have reason to work with a single standalone Terminal program, a small screen display or have reason to control programs in more detail.

Textual: Command Line or Screen based

→ When you start a full application from the command line, control and response is **transferred from shell to application**. You have already used one such application – **`ftp`**. Remember how the command prompt changed from `s1234567@baltic10...$` to `ftp>` ? Another program that may once have been of use in extremely low bandwidth field-research environments is the email program **Alpine**. This is an entirely textual email client! The important point to note with all these is that **once you are running the application, the commands will be specific to that application**. As a quick example, do the following:

→ `s1234567@baltic10:~$` **`alpine`**

m) Note how this time the command prompt format then changes significantly to that of a different type of program. How would you describe this?

……

→ Type **`e`** to exit the initial greeting if shown, then type **`q`** and then **`y`** to quit.

→ Another program that uses the same whole-window/screen format is the editor **`nano`** which is used as a basic default editor by some applications, such as the School Oracle databases.

Graphical: Graphical User Interface (GUI) based

n) GUI based applications are very different. Here, the application usually opens a new window for itself, and your **interaction with the application is through that window, not the command line**. For example:

`s1234567@baltic10:~$` **`xcalc`**

will start up **`xcalc`** in a new window. Your interaction with **`xcalc`** is now through the GUI, not the command line. Move the window if necessary so that you can see the Terminal window. What has happened to the command line? We will come back to this later…

……

→ Now with a rich full-featured environment such as xrdp you can simply open a second Terminal window should you require to issue commands whilst `xcalc` runs, however we can only keep opening a new Terminal for so long, so it's useful to be able to understand how applications work in Linux and how we can control running programs – more on this later!

SSH: Connecting to programs running on specific servers

→ You may require access to other Linux machines e.g. if they have different software or greater processing power. This is common when using Linux/UNIX workstations. You can achieve this by connecting with the **ssh** (secure shell) command from a Terminal window.

→ Other School servers exist which you may wish to connect to at some point, e.g. **stream**.

→ `s1234567@baltic10:~$ ssh stream`

(→) You *may* be asked for your username and password, or just your password, **each** time you make another ssh connection from an existing session. From time to time you may receive warning messages about network addresses or host keys etc. in relation to the new ssh connection. Since you are already safely within the network such warnings can often be safely read once and then disregarded however if unsure you can always ask IT by sending an email to it.geos@ed.ac.uk.

→ *Each* time you log in, you are connected to that machine via a *brand new* connection, just the same as when you first open a terminal window on a PC. You will thus once again start in your home directory each time you make a new connection. To close the connection to the remote machine use **exit**: Control will pass back to your original machine (or ssh window) and so on if you have multiple logins 'on the go' at one time.

→ `1234567@stream:~$ exit`

11 Log Out/Off or Close Session?

→? **Logout… (completely!)**
You can close and logout an individual Terminal session by typing **exit** as above. This is recommended as it will close the session 'cleanly' – i.e. without any unclosed programs or files. It is always best to clean up after yourself. To close the whole graphical XRDP session and all programs you can go to the **System** menu and select **Log Out**. You will be asked to confirm. This also avoids leaving Terminal windows open for long periods (e.g. days or weeks) during which time configuration files may be changed resulting in unpredictable or unexpected results in a Terminal window. Alternatively…

→? **Close – Disconnect (temporarily!)**
You can also go to the top of the screen and wait for the blue Remote Desktop drop-down bar to appear and click Close (X). The major strength of XRDP and closing this way is that when you login later any open programs should remain running. (**Note:** There is also a tool called screen which can be used to keep programs open when running at a command line only using an SSH client such as if having to connect to a more powerful server as described above. See: **https://blogs.ed.ac.uk/geosciences-it-help/linux-servers/**)

→ That's all for this jam-packed practical. You can use this document for guidance during the coming months, and certainly during your dissertation!