UNIX (LINUX) PRACTICAL 1
INTRODUCTION

1. **CONNECTING TO UNIX (LOGGING ON)**

2. **FILES AND DIRECTORIES**
   - Listing, viewing, copying, making. How your workspace is structured.

3. **HELP!**
   - How to get it. Is it helpful? Other places to look.

4. **FILE UTILITIES**
   - Useful tools that use the command line - **`tar`**, **`gzip`**, **`ftp`**.

5. **RESOURCE MONITORING UTITILITIES**
   - Finding out how much disk space you have. Finding other users.

6. **SEARCHING**
   - Finding files. Finding files with a specific bit of text in them.

7. **APPLICATIONS**
   - What are available? How do I start them? Transferring machines: **`ssh`**

This practical will allow you to familiarise yourself with the basics of working with UNIX (Linux) in the School of GeoSciences. The practical worksheet 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 at:

`http://unixhelp.ed.ac.uk/`

There is a set of attainment targets on the next page. Make sure you can do everything in the list. Many students **will** be using UNIX 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.

Throughout this worksheet 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:

```
[snnnnnnn@adder ~]$ pwd
[snnnnnnn@adder ~]$ cd /
[snnnnnnn@adder /]$ ls -al
```

s*nnnnnnn* represents your unique login (your matriculation number preceded by an 's'.)

~ represents your home directory. In the command prompt above note how this will change to reflect whichever directory you are currently working in.

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

Tasks for you to do will have a letter in the left hand margin - a) to p).

TARGETS FOR THIS SESSION

You should be able to:

- **Log on** to any UNIX/Linux machine in the School of GeoSciences
- Locate your **home directory**
- Change directories
- Create directories
- Find out how much **disk space** you are using
- View simple **text files**
- Find out about other **users** on the system
- **Search** for (and through) files
- Start UNIX **applications**
- Connect to different machines using **ssh**

And you should know:

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

1 CONNECTING TO UNIX (LOGGING-ON)

There are several UNIX/Linux systems in the School and University, and you can connect to any which you have been authorised to use. You will need a username and password (which, at least in GeoSciences, are the same university Universal UserName (UUN) you have used before, and your EASE password.)

You most usually connect (log on) to UNIX servers remotely, in that these tend to be hidden away in air-conditioned machine rooms, although occasionally you may sit down in front of a UNIX/Linux workstation.

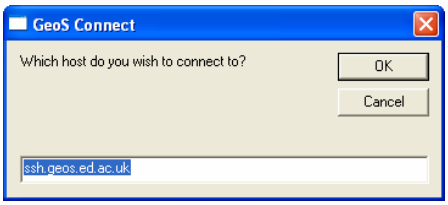
Linux is the form of UNIX now used throughout the School although some older software is only available on commercial UNIX variants e.g. Sun's Solaris. Linux is an open-source 'POSIX-compliant' operating system, often available at no (retail) cost, and is based on the earlier UNIX operating system developed in the late 1960s.

Connecting remotely from a lab PC is best achieved using PuTTY. **NB** If you require UNIX programs that have a graphical display, you should use one of the specially-created menu shortcuts where provided in order to ensure that the Xming 'X-Windowing' program is set-up when you connect to the server. In School and University labs go to:

→ **U:\SCE\GEOS\GeoS Connect**

The GeoS Connect script will automatically start the X-Server and launch PuTTY.

→ A small window will appear with **ssh.geos.ed.ac.uk** shown in the text input box. Click **OK**.



A window will appear asking you to **login as:**

→ Type your username, press **<return>** then type your EASE password and press **<return>** again. Note that your password will not appear as you type in order to be completely secure by not revealing how many characters are contained within it!

```
login as: snnnnnnn
Sent username "snnnnnnn"
snnnnnnn@ssh.geos.ed.ac.uk's password:
```

You will then be presented with the following prompt and you need to enter appropriate commands:

```
[snnnnnnn@adder ~]$
```

PuTTY is an example of a type of **Telnet**/**SSH** program, further discussed later.

2 FILES AND DIRECTORIES

Listing Files and Directories

a) One of the most useful commands in UNIX is **pwd** - **p**rint **w**orking **d**irectory. It tells you where you are in the directory structure. Type:

```
[snnnnnnn@adder ~]$ pwd
```

What is the address of your home directory?

……

Be aware that drive letters such as **M:** (or **G:** or **T:** etc.) are a Windows-only feature and that when using UNIX you will need to reference your home directory as /home/s*nnnnnnn*.

(This is also the form used when mapping your home directory to a Windows drive letter, perhaps when connecting from outside the university, i.e. of the form //geoshome*n*/home/s*nnnnnnn*.)

Fortunately you can easily access command-line Linux with PuTTY simply with your usual username and EASE password, and are logged into your home directory 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. Are there any files in your home directory?

……

ls can take a number of **arguments** and the output from it can be changed by the use of **flags**. For an example 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:

→
```
[snnnnnnn@adder ~]$ ls /geos/netdata/wkzero
demodir  if.txt  jabberwock.txt  nation_data.txt  xyzpoints.txt
```

→ `[snnnnnnn@adder ~]$ ls -al /geos/netdata/wkzero`

```
total 68
drwxr-xr-x  3 omacdona netdata  4096 Sep 14  2010 .
drwxrwsr-x 42 omacdona netdata  4096 Aug 31 11:09 ..
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
-rwxr-xr-x  1 omacdona netdata 34965 Sep  7  2011 xyzpoints.txt
```

The first version used **`ls`** with an **argument** (the address of 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 UNIX start with a **`.`** character). **`ls -l`** means give **l**onger (more) information on each file.

In the example above you should note the two extra entries **`.`** and **`..`** . A single **`.`** means "the present 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:

```
drwxr-xr-x  3 omacdona netdata  4096 Sep  7 13:15 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 UNIX **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 show that the user (**owner**) has permission to **r**ead, **w**rite/delete, and e**x**ecute (run commands on) this file. Columns 5, 6, 7 and 8, 9, and 10 respectively show that both the UNIX **group** and all **other** users only have permission to **r**ead and e**x**ecute for this file. They cannot alter it or delete it. We also commonly say that the file is world-readable (i.e. readable only by anyone in who has access to the Geosciences server that is!)

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 below.

……

Simple Viewing of Files

d) Viewing simple text files is easy - you can use one of 3 commands:

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?

……

……

There is actually also a similar command `less` which is set as the default text viewing 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 with `more`.

Creating and Deleting Files and Directories

Managing your directories and files is an important aspect of working with UNIX (or indeed any computer operating system). You only have a limited amount of disk space and 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:

→ `[snnnnnnn@adder ~]$` **mkdir wkzero**

will create a new directory called **wkzero** – **NB** in *your* current directory . Using **mkdir** create some other new directories to put *your* work in. How about **webpages** and **papers** for starters?

e) Once you have made these new directories, check their properties using **ls**.

Having directories isn't much use if you are stuck in your home directory. To change directory use **cd**. For example:

→ `[snnnnnnn@adder ~]$ cd wkzero`
→ `[snnnnnnn@adder wkzero]$ pwd`
`/home/snnnnnnn/wkzero`
→ `[snnnnnnn@adder wkzero]$ cd ..`
→ `[snnnnnnn@adder ~]$ pwd`
`/home/snnnnnnn`

Remember the **..** notation from before? Here it is *used* to change to the parent directory of the directory that you are currently in.

f) Now change (**cd**) to the **wkzero** directory and add two more new subdirectories called **ftpdata** and **docs**. Use **ls** to check everything is OK.

Note! You can create directories in your home directory from within Windows using Windows Explorer (My Computer). Your home directory usually appears in Windows, mapped to **M:**\ for most postgraduates, or perhaps to another letter (usually T:) for undergraduates or students formerly undergraduates at Edinburgh.

When using Windows to access UNIX 'graphically' – **be careful!** Windows lets you put spaces in your directory names. UNIX is happy enough with this, but it can make moving around in directories within UNIX more awkward since you may need to enclose the directory names in **" "** marks. You may also find some software will object to UNIX paths with spaces in their names. It is much better to use a dash _ or hyhen – instead of a space.

To delete a file in UNIX 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 UNIX will ask you to confirm the deletion of a file or directory. Much safer… **Remember** that with *network* drives, e.g. if you use Windows Explorer to delete a file or directory on your **M:**\ drive, the same rules apply – it will not go to the Recycle Bin, it will be gone forever!

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

Sneaky 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 move the cursor along a recalled command to edit it with minimal effort.

2) If you are typing a file or path name, you can use auto-completion by hitting **tab**. Try it!

3) 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**).

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 the **wkzero** folder then do the following:

→ [s*nnnnnnn*@adder wkzero]$ **cp /geos/netdata/wkzero/if.txt .**
→ [s*nnnnnnn*@adder wkzero]$ **cp /geos/netdata/wkzero/if.txt 2nd_if.txt**

g) But what do they do? Use **ls** to list all the files in your directory. What has happened? So what does each command above do?

……

……

……

Copying a whole directory is very similar - you just add a flag **-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. Try:

→ [s*nnnnnnn*@adder wkzero]$ **cp -R /geos/netdata/wkzero wkzero_copy**

h) Then use **ls** again. What has happened?

……

……

Moving and **renaming** a file or directory are the same thing in UNIX 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:

→ [s*nnnnnnn*@adder wkzero]$ **ls wkzero_copy/**
if.txt nation_data.txt xyzpoints.txt jabberwock.txt
→ [s*nnnnnnn*@adder wkzero]$ **mv wkzero_copy/ not_wanted_now**

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

[s*nnnnnnn*@adder wkzero]$ **ls wkzero_copy/**
wkzero_copy/: No such file or directory

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

→ [s*nnnnnnn*@adder wkzero]$ **mv not_wanted_now/*.txt docs/**
→ [s*nnnnnnn*@adder wkzero]$ **ls docs/**
if.txt jabberwock.txt nation_data.txt xyzpoints.txt

The * character is a **wildcard** that means "all names" or "all characters".

NB The final / is often not required, it simply indicates to the user that they are referencing a directory. Thus:

→ [s*nnnnnnn*@adder wkzero]$ **ls docs**
if.txt jabberwock.txt nation_data.txt xyzpoints.txt

will work just as well!

To keep your home directory (and i.e. thus **M:** or **T:** drive) tidy and well-organised you should **mo**v**e** your **PCInduction** folder from yesterday (which should be in your uppermost top-level ***home*** directory) into your **wkzero** folder.

We can do this as follows:

→ [s*nnnnnnn*@adder wkzero]$ **mv ../PCInduction PCInduction**

Note we could also have issued this as any of these:

[s*nnnnnnn*@adder wkzero]$ **mv ../PCInduction .**
[s*nnnnnnn*@adder wkzero]$ **mv ~/PCInduction PCInduction**
[s*nnnnnnn*@adder 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:

[s*nnnnnnn*@adder wkzero]$ **mv ~/PCInduction ~/wkzero/PCInduction**

NB This last command can be issued from ***anywhere*** within the directory structure!

3 HELP

Help in UNIX 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 – online versions of the UNIX manual. The structure is **man <command>** where **<command>** is the name of the command you want help with. Try:

→ `[snnnnnnn@adder 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:

→ `[snnnnnnn@adder 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?

……

……

4 FILE UTILITIES

FTP (File Transfer Program/Protocol)

FTP is a way of transferring files from a remote computer/site - either a standalone PC, workstation, or server, to your machine. It is a very useful facility, utilised by a large number of internet sites and geographic data providers. The following example uses a demonstration FTP site set up for this practical. Due to the insecure nature of FTP many sites will now require Secure FTP (sometimes known as SFTP).

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):

```
[snnnnnnn@adder wkzero]$ cd ftpdata
[snnnnnnn@adder ftpdata]$ ftp
ftp> open ftp.ed.ac.uk
Connected to lewis.ucs.ed.ac.uk.
220-    Welcome to the University of Edinburgh Anonymous FTP server
220-============================================================================
220-
220-The following anonymous ftp servers are also available:
220-
220-    ftp.ed.ac.uk            University of Edinburgh (this server)
220-    ftp.ucs.ed.ac.uk        Information Services (old EUCS)
220-    ftp.epcc.ed.ac.uk       Edinburgh Parallel Computing Centre
220-
220-When requested for a username enter 'ftp' or 'anonymous'.  If you have
220-problems, try using a dash (-) as the first character of your password.
220-If you still have problems or wish to make a comment then send email to
220-ftpmaster@ed.ac.uk. The local time is Wed Sep 10 13:19:15 2008.
220-
220-All transfers are logged.  This server supports automatic taring and
220-compressing/uncompressing of files during transfer.
220-
220-
220 lewis.ucs.ed.ac.uk FTP server (Version wu-2.6.2(2) Mon Oct 12 14:39:33 BST
2009) ready.
530 Please login with USER and PASS.
530 Please login with USER and PASS.
KERBEROS_V4 rejected as an authentication type
Name (ftp.ed.ac.uk:snnnnnnn): anonymous
```

You should get a response like this:

```
331 Guest login ok, send your complete e-mail address as password.
Password: snnnnnnn@sms.ed.ac.uk

230-This service is managed by Information Services.  It holds information
230-which may be useful to system managers and space is provided for
230-individuals and groups upon request.  Upload facilities are also
230-available.  Anyone can make use of this service.
230-
.
.
.
230-
230 Guest login ok, access 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 UNIX commands).

→ Type the following:

```
ftp> ls
227 Entering Passive Mode (129,215,70,239,201,118)
150 Opening ASCII mode data connection for /bin/ls.
total 1100
-rw-------   1 root          38 Dec 16  2010 .bash_history
lrwxrwxrwx   1 root           6 Jan 27  2006 .login_message -> README
----------   1 root           0 Oct 11  2005 .notar
-r--r--r--   1 other          0 Oct 19  2005 .www_browsable
-r--r--r--   2 root        2647 Sep 24  2010 INSTRUCTIONS-FOR-USING-THIS-SERVICE
-r--r--r--   2 root        2647 Sep 24  2010 README
lrwxrwxrwx   1 bin            9 Oct 13  2005 bin -> ./usr/bin
drwxr-xr-x   2 other        512 Oct 13  2005 dev
drwx-wx-wx   2 other       5632 Sep  8 00:00 edupload
drwxr-xr-x   5 other        512 Oct 13  2005 etc
drwx-wx-wx   2 other     221184 Sep 10 11:55 incoming
lrwxrwxrwx   1 root           5 Nov 17  2005 index -> ls-lR
d--x--x--x   2 other        512 Oct 13  2005 lib
-rw-r--r--   1 other     255744 Sep 11 00:00 ls-lR
-rw-r--r--   1 root       40777 Sep 11 00:00 ls-lR.Z
drwxr-xr-x  56 other       1536 Jun  9  2011 pub
drwx--x--x   6 other        512 Oct 13  2005 usr
226 Transfer complete.
ftp> cd pub/geos
250 CWD command successful.
ftp> ls
227 Entering Passive Mode (129,215,146,5,89,98)
150 Opening ASCII mode data connection for /bin/ls.
total 2
drwxr-xr-x   2 root         512 Sep 10  2008 wkzero
226 Transfer complete.
ftp> cd wkzero
250 CWD command successful.
ftp> ls
227 Entering Passive Mode (129,215,146,5,78,174)
150 Opening ASCII mode data connection for /bin/ls.
total 2
-r--r--r--   1 12           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 in the same directory on your local UNIX server. The command sequence above shows you some of the files you can download. Often you will first byehave 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:

```
ftp> get jefferson.tar.gz
local: jefferson.tar.gz remote: jefferson.tar.gz
227 Entering Passive Mode (129,215,146,5,249,166)
150 Opening BINARY mode data connection for jefferson.tar.gz (365 bytes).
226 Transfer complete.
365 bytes received in 0.0091 seconds (39 Kbytes/s)
```

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

```
ftp> bye
→ 221-You have transferred 365 bytes in 1 files.
221-Total traffic for this session was 6074 bytes in 4 transfers.
221-Thank you for using the FTP service on lewis.ucs.ed.ac.uk.
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.

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 data files on UNIX, this file is actually a collection of files specially encoded to save space. This is a two stage process that will introduce you to two more useful UNIX utilities.

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

```
→ [snnnnnnn@adder ftpdata]$ ls -l
  total 4
  -rw-rw-r-- 1 omacdona omacdona 365 Sep  7 15:40 jefferson.tar.gz
→ [snnnnnnn@adder ftpdata]$ gzip -d jefferson.tar.gz
→ [snnnnnnn@adder ftpdata]$ ls -l
  total 12
  -rw-rw-r-- 1 omacdona omacdona 10240 Sep  7 15:40 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 more with it! Have a look at either the online help or the UNIXHelp web pages…

To extract the files, type the following:

→ `[snnnnnnn@adder ftpdata]$ 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 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.

5 RESOURCE MONITORING UTILITIES

You only have a limited amount of disk space on UNIX/Linux. You can find out what disk space you have available by using the **quota** command:

→ `[snnnnnnn@adder ftpdata]$ quota`

Using the **-s** option/flag will give you a more useful breakdown.

→ `[snnnnnnn@adder ftpdata]$ quota -s`

Due to the peculiarities of this particular system perhaps, the quota command can display misleading user information (i.e. home dir) however the numbers should reflect *your* quota! To find out how much disk storage space you are using, use the **du** command. See **man du** for fuller details.

→ `[snnnnnnn@adder ftpdata]$ du -sk`

will give you the total amount in kilobytes of disk space used in the current directory which should still be `ftpdata` 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 readable form.

→ `[snnnnnnn@adder ftpdata]$ du -sh`

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

→ `[snnnnnnn@adder ftpdata]$ 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:

→ [s*nnnnnnn*@adder ftpdata]$ **cd ~**
→ [s*nnnnnnn*@adder ~]$ **du -sh ***

Alternatively you can achieve the same anywhere as follows:

→ [s*nnnnnnn*@adder ~]$ **du -sh ~/***

j) Using **quota** and **du**, find out how much storage space you are using and how much you have left:

……

k) You can find out about other users on the system in two ways – **who** and **finger**.

[s*nnnnnnn*@adder ~]$ **who**

will give you a list of who is currently logged on to the machine you are working on - it will also tell you how they are logged in. Who is logged on to the same UNIX machine as you? And which remote machine are they using to access it?

……

……

……

You can use a related command if you forget who you are – **whoami**

l) **finger** gives a much bigger set of information about a user. It allows you to search for users not only with their user name but also with either their real last name or real first name – the example bellow shows the result for **gisteac**.

```
[snnnnnnn@adder ~]$ finger gisteac
Login: gisteac                          Name: MSc in GIS teaching
Directory: /home/gisteac                Shell: /bin/bash
Last login Tue Sep 14 17:29 (BST) on pts/34 from adder.geos.ed.ac.uk
No mail.
No Plan.
```

See if you can find out the login name and home directory of another user familiar to you by 'fingering' their *surname*.

……

……

The information in the **Plan:** comes from a special file in the user's home directory called **.plan** (a hidden file) – not everybody has a plan… ☺

6 SEARCHING

m) You can search for all sorts of things in UNIX – 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 **http://unixhelp.ed.ac.uk/** and maybe **man** to find out what **grep** and **find** actually do…

……

……

……

……

……

……

……

……

……

……

……

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

```
[snnnnnnn@adder ~]$ find /geos/netdata/wkzero -name "*.txt"
[snnnnnnn@adder ~]$ grep -n 'stan' /geos/netdata/wkzero/nation_data.txt
```

……

……

……

……

……

7 APPLICATIONS

The way you interact with UNIX software applications is slightly different, depending on their usual mode of operation…

Text (Command Line) based

When you start a command line based application, control and response is **transferred from UNIX to the application**. You have already used one application of this sort – **`ftp`**. Remember how the command prompt changed from `[snnnnnnn@adder ~]$` to `ftp>` ?

Another program that you may wish to use is the UNIX email program **Alpine**. 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:

→ `[snnnnnnn@adder ~]$` **`alpine`**

o) 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.

GUI (Graphical User Interface) based

p) 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:

`[snnnnnnn@adder ~]$` **`xclock`**

will start up **X-Clock** in a new window. Your interaction with X-Clock is now through the GUI, not the command line. What has happened to the command line? We will come back to this in later practicals…

……

……

SSH: Connecting from Home and Connecting to other Servers

You may require access to other UNIX/Linux machines e.g. if they have different software or greater processing power. This is common when using UNIX workstations.

You can also connect to a School UNIX server from home. The way to do this is to connect to **`ssh.geos.ed.ac.uk`** using PuTTY (or the built-in ssh software if using an Apple Mac). This ensures that you will have proper access to the correct School *compute* server which may change in future while the address, `ssh.geos.ed.ac.uk`, should remain the same.

We will briefly simulate connecting from home and then using **`ssh`** from the command line to connect to another UNIX machine.

Run a plain copy of PuTTY without any connection script as follows:

Start ► All Programs ► XMing ► Portable PuTTY ► PuTTY.

Running PuTTY the long-way best demonstrates typical behaviour shown by any ssh program that you might install on a home computer.

In the **Host Name (or IP address)** box type **`ssh.geos.ed.ac.uk`**. Ensure that **22** can be seen in the **Port** box and that **SSH** is selected under **Connection type:** then click **Open**. A small window will appear as usual and prompt for your Universal UserName (**UUN**) and (**EASE**) password.

You are connected to the GeoSciences Linux Cluster. Now connect to another UNIX or Linux server.

```
[snnnnnnn@adder ~]$ ssh loyal
```

You may be asked for your username and password, or just password, **each** time you make another ssh connection from an existing session. Don't worry if you are given a message about host authenticity each time you connect to a new *host*; when asked if you wish to continue, you can safely type **`yes`** then press **`<return>`**.

When you log in, you are connected to that machine via a new connection, just the same as when you first open a terminal window on a PC. You will once again start in your home directory each time you make a new connection.

To close the connection to the remote machine use **`exit`**:

```
[snnnnnnn@adder ~]$ exit
```

Control will pass back to your original machine (or ssh window.)

If you wish to run graphical applications at home you will need a program called an X-Server such as XMing. You ***must*** also enable *X11 forwarding* in PuTTY via **File ► New Session** (or from the main PuTTY start-up screen) then in the **Category:** window on the left-hand side go to **Connection ► SSH ► X11**. You can save these settings in **Saved Sessions**.

To launch XMing for any reason in a lab go to **Start ► All Programs ► XMing ► Xming**. You need not start XMing first (i.e. before PuTTY) so can simply run this as/when required.

Alternatively if you come across a program that does not require a graphical display, but appears to hang at the command prompt (or fails to provide a prompt) then you may need to issue the following command before running the program: **export DISPLAY=**. This ensures that the computer does not try to force a graphical display when there is no X software there to interpret graphics drawing commands. To stop a program that has 'hung' or frozen use pres **CONTROL** and **C** together **([Ctrl-C])**.

That's all for this practical. You should safely file these worksheets and use them for guidance during the coming months, and certainly during your dissertation!