**Introduction to Unix/Linux**

* In this set of exercises, we will introduce the Unix/Linux operating system. You will learn about command shells (terminal windows) and how to use them to more easily achieve data analysis tasks than is possible in PC graphical user interfaces (GUIs).
* We will also utilize a series of command-line instruction to explore the Unix environment and do some small tasks.
* It is common for the raw data files that are provided from High-throughput sequencing to be very large (~32 GB for a whole-exome or whole-genome sequence). This means the files cannot be easily manipulated on a normal PC. However, Unix is well suited for large file sizes, so familiarity with the Unix command line is an important skill to learn.
* At the end of the session, you should write a short paragraph regarding what you learned and place this in a folder. This is just an opportunity for you to take mental notes about what you just have just done, while it is fresh in your mind. It is good practice to write things down, especially if you may need it later.
* Please keep a detailed *e-*journal for results that you get throughout the practical. *Screen shots (****Ctrl-print Scrn*** *on your computer*) are really handy for this.
* For practical exams you may be asked to re-create some of the commands in red, so make sure you document your activities (the same as you would in the lab).

**What is Unix?**

Unix is a computer operating system first developed in the 1960s, and has been continuously developed ever since. When defining an operating system, we mean a suite of programs that make the computer function. It is a stable, multi-user, multi-tasking system designed for servers, desktops and laptops.

Unix systems also have a graphical user interface (GUI) similar to Windows which provides an easy to use environment. However, a basic knowledge of Unix is required for operations which aren't covered by a graphical program, or for when there is no windows interface available, for example, a session with a remote computer.

There are many different flavours of Unix, although they share common similarities. The most popular varieties of Unix are Sun Solaris, GNU/Linux, and MacOS X.

For this unit, to provide students a Unix environment we will utilize a virtual machine running a Linux operating system. In the building 308 computing lab, the 'Bioinformatics VM' appears on the desktop. Students who wish to try this at home or on their personal laptop can get a copy from me (Phillip). You will also have to install software that is able to run the VM, e.g. VirtualBox or VMware Player if using your own computer (for which I will happily provide instructions).

**Structure of the Unix operating system.**

The UNIX operating system is made up of three parts; the kernel, the shell and the programs.

**The kernel:** The kernel of Unix is the hub of the operating system: it allocates time and memory to programs and handles the filestore and communications in response to system calls.

For an example of the way that the shell and the kernel work together, suppose a user types the command, *rm myfile* (which has the effect of removing the file myfile). The shell searches the filestore for the file containing the program rm, and then requests the kernel, through system calls, to execute the program rm on myfile. When the process rm myfile has finished running, the shell then returns the Unix prompt $ to the user, indicating that it is waiting for further commands.

**The shell**

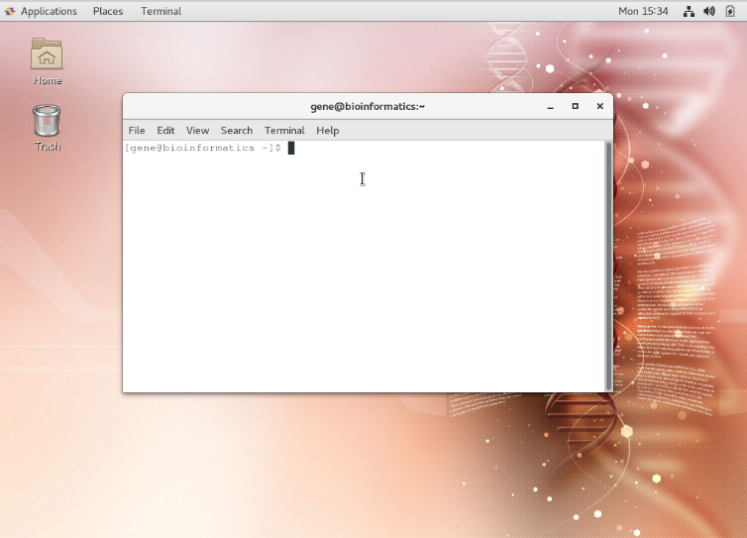
The shell acts as an interface between the user and the kernel. When a user logs in, the login program checks the username and password, and then starts another program called the shell. The shell is a command line interpreter (CLI). It interprets the commands the user types in and arranges for them to be carried out. The commands are themselves programs: when they terminate, the shell gives the user another prompt (% on our systems).

The adept user can customise his/her own shell, and users can use different shells on the same machine. Staff and students in the school have the tcsh shell by default.

{**ProTip, the command** *History*, - keeps a list of the commands you have typed in. If you need to repeat a command, use the cursor keys to scroll up and down the list or type history for a list of previous commands}

**Intro to the Command Line – Bioinformatics Machine**

Start the Bioinformatics VM and wait for the machine to come up. You should see a desktop with a shell window in the middle where you will type commands.



When you click on the VM window in your Desktop, you may be asked if you want to 'capture' the mouse. You do want to do this so that you can interact fully with the Linux operating system. Move the shell window around and explore the Applications menu to make sure you can interact with the system. To get out of the VM, you will then have to press specific keys to 'release' the mouse. You should see a message that tells you what keys to press. Under Windows it is the 'Alt' and 'Ctrl' keys. Under Mac OSX it is a mouse-click plus the command (apple) key. Practice this a few times to make sure that you can get in and out of the VM.

**Part 1: Basic Commands**

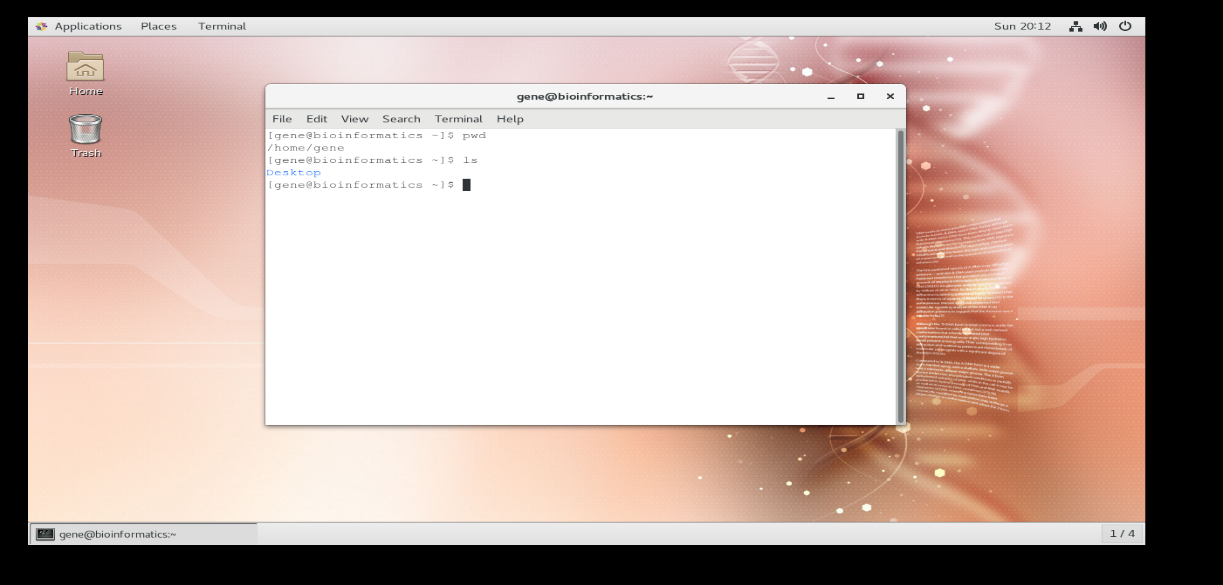
1. **ls (list):** When you first login, your current working directory is your home directory. Your home directory has the same name as your user-name, for example, *gene*, and it is where your personal files and subdirectories are saved.

*To find out what is in your home directory, type*

***[gene@bioinformatics ~]$* ls**

NB: don’t type [gene@bioinformatics ~]$ as that will already be there.

You should see a screen similar to this, there are no files but there is a director called Desktop

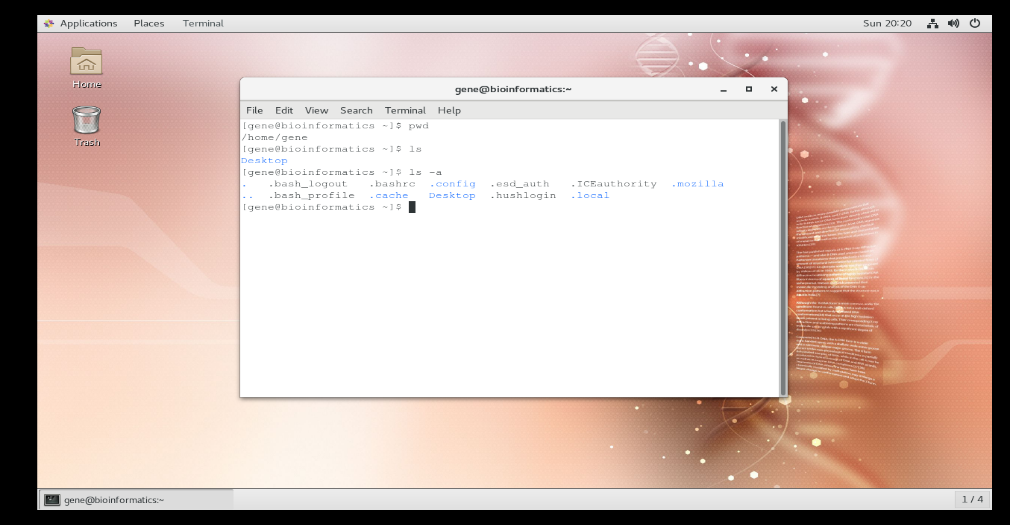
****

There may be no files visible in your home directory, in which case, the Unix prompt will be returned. Alternatively, there may already be some files inserted by the System Administrator when your account was created.

*ls* does not, in fact, cause all the files in your home directory to be listed, but only those ones whose name does not begin with a dot (.) Files beginning with a dot (.) are known as hidden files and usually contain important program configuration information. ***They are hidden because you should not change them unless you are very familiar with Unix!!!***

To list all files in your home directory including those whose names begin with a dot, type

***[gene@bioinformatics ~]$* ls -a**



1. **mkdir (make directory):** We will now make a subdirectory in your home directory to hold the files you will be creating and using in the course of this tutorial. To make a subdirectory called Myunixstuff in your current working directory type.

***[gene@bioinformatics ~]$* mkdir Myunixstuff**

To see the directory you have just created, type.

***[gene@bioinformatics ~]$* ls**

1. **cd (change directory):** The command *cd* ‘directory’ means change the current working directory to 'directory'. The current working directory may be thought of as the directory you are in. To change to the directory you have just made, type.

***[gene@bioinformatics ~]$* cd Myunixstuff**

1. **pwd (print working directory):** Pathnames enable you to work out where you are in relation to the whole file-system. For example, to find out the absolute pathname of your home-directory, type cd to get back to your home-directory and then type.

***[gene@bioinformatics/Myunixstuff ~]$* pwd**

This should return the following /home/gene/Myunixstuff

**EXERCISE 1a**

*Make another directory inside the Myunixstuff directory called backups (Take a screenshot).*

**EXERCISE 1b**

*Use the commands* ***cd, ls,*** *and* ***pwd*** *to explore the file system.*

***(NB: Remember, if you get lost, type* cd *by itself to return to your home-directory)***

**Getting Help**

**On-line Manuals:** Unix provides numerous on-line manuals which provide information about most commands. These manual pages tell you which options a particular command can take, and how each option modifies the behaviour of the command.

Type man command to read the manual page for a particular command, such as .

***[gene@bioinformatics/Myunixstuff ~]$* man ls**

Go ahead and look through this information. What does the following command adding the “-l” to my “ls” command do?

**Part 2: Commands to Move/View Files**

1. **cp (copy):** cp file1 file2 is the command which makes a copy of file1 in the current working directory and calls it file2.

What we are going to do now, is to take a file stored in an open access area of the file system, and use the cp command to copy it to your Myunixstuff directory.

***[gene@bioinformatics ~]$* cd ~/Myunixstuff**

***[gene@bioinformatics/Myunixstuff ~]$* cp /data/human\_chr20\_reference.fa .**

***(NB: Don't forget the dot . at the end. Remember, in Unix, the dot means “the current directory”.)***

The above command means copy the file science.txt to the current directory, keeping the name the same.

**EXERCISE 2a**

*Create a backup of your human\_chr20\_reference.fa file by copying it to a file called human\_chr20\_reference.fa.bak. What is the command for this?*

1. **mv (move):** mv file1 file2 moves (or renames) file1 to file2. To move a file from one place to another, use the mv command. This has the effect of moving rather than copying the file, so you end up with only one file rather than two. It can also be used to rename a file, by moving the file to the same directory, but giving it a different name. We are now going to move the file human\_chr20\_reference.fa.bak to your backup directory.

*First, change directories to your Myunixstuff directory (can you remember how?). Then, inside the Myunixstuff directory, type*

***[gene@bioinformatics/Myunixstuff ~]$* mv *human\_chr20\_reference.fa.bak* backups/.**

*Type ls and ls backups to see if it has worked.*

1. **rm (remove), rmdir (remove directory):** To delete (remove) a file, use the rm command. As an example, we are going to create a copy of the chr20\_reference.fa file then delete it.

***[gene@bioinformatics/Myunixstuff ~]$* cp human\_chr20\_reference.fa tempfile.txt**

**ls**

**rm tempfile.txt**

**ls**

You can use the rmdir command to remove a directory (make sure it is empty first).

Try to remove the ‘backups’ directory. You will not be able to since Unix will not let you remove a non-empty directory.

**EXERCISE 2b (Make sure you are still in the Myunixstuff directory)**

*Create a directory in your ‘Myunixstuff’called tempstuff using mkdir, then remove it using the rmdir command.*

1. **clear (clear screen)**: Before you start the next section, you may like to clear the terminal window of the previous commands so the output of the following commands can be clearly understood.

*At the prompt, type*

***[gene@bioinformatics ~]$ clear***

This will clear all text and leave you with the % prompt at the top of the window.

1. **cat (concatenate):** The command cat can be used to display the contents of a file on the screen. Type

***[gene@bioinformatics ~]$ cat human\_chr20\_reference.fa***

*As you can see, the file is longer than than the size of the window, so it scrolls past too quickly, making it unreadable.*

1. **head:** The head command writes the first ten lines of a file to the screen. Type

***[gene@bioinformatics ~]$ head human\_chr20\_reference.fa***

Then type

***[gene@bioinformatics ~]$ head -500 human\_chr20\_reference.fa***

*What difference did the -500 do to the head command?*

1. **tail:** The tail command writes the last ten lines of a file to the screen. Type

***[gene@bioinformatics ~]$ tail human\_chr20\_reference.fa***

*What letters does the last line contain? Are all the letters the same or are they different?*

**Part 3: Common Unix Utilities**

As mentioned earlier, often the files used in Genomics are very large, but what happens if we want to see if something is present in a file?

Hopefully now you are comfortable with some of the more common files – so in this section we are going to look though and find some information within some files and compile some data on them.

First we need to copy some new data files into our Myunixstuff directory

**{Protip-Make sure what directory you are in}**

Now we want to copy all of the files from a specific directory that end in ‘.txt’

***[gene@bioinformatics ~]$ cp /data/data-shell/data/\*.txt .***

*How many files were copied? What are the file names?*

***So say now we want to find any of the files that contain the word “rabbit” but we don’t want to open each of them and scan through them. To do this we can use a really useful Unix command called ‘grep’***

1. **Grep (don’t ask why is called grep)\*:** grep is one of many standard Unix utilities (NB: Utilities are small programs that do useful things that are available on all different flavours of Unix/Linux). It searches in files for specified words or patterns. First clear the screen then type.

***[gene@bioinformatics ~]$ grep rabbit \*.txt***

*How many times does the word “rabbit” appear?*

*How many files have the word rabbit in it?*

1. **wc (word count):** g A handy little utility is the ‘wc’ command, short for word count. To do a word count on morse.txt, type

***[gene@bioinformatics ~]$ wc -w morse.txt***

To find out how many lines the file has, type

***[gene@bioinformatics ~]$ wc -l morse.txt***

**Exercise 3a**

Which of your \*.txt files has the most words, which has the most lines.

Do the same for the human\_chr20\_reference.fa file –

How many lines?

How many words?

Are they the same or different?

What type of file do you think this?

**Exercise 3b**

Find out more about the “grep” command using “man”.

What do the “–v” and “^” options do?

What do they tell us about the human\_chr20\_reference.fa file?

*\*If you really want to know – grep stands for {global/regular expression/print} – hence grep. The creators of Unix utilities did not use imaginative titles. There is a historic reason for this as until recently most commands could be only 8 characters long, so things were often shortened as much as possible.*