README.md

Class 1 – Setting up Great Lakes account and Introduction to UNIX

Goal

- In this module, we will log into Great Lakes and make sure we all have access.
- Get acquanted with the basics of UNIX file system.
- Learn how to navigate course directory structure using command line interface.
- Learn some foundational commands and UNIX shortcuts

Sign up for Great Lakes account

Log in to great lakes

Log into your great lakes account using the ssh command below. SSH stands for secure shell and can be used to connect or transfer data to/from remote hosts such as Great Lakes.

Replace username with your umich uniqname and run the ssh command to log in to Great Lakes.

```
ssh username@greatlakes.arc-ts.umich.edu
```

Note that if you are off-campus you will need to be connected to the VPN to login to Great Lakes!

Intro to Unix and your first command!

UNIX systems allows users to perform complex and powerful tasks, often with just a few keystrokes/commands or lines of code. It helps users automate repetitive tasks and easily combine smaller tasks into larger, more powerful workflows. Bioinformatics data analyis involves generation of lots of files with different types of extensions. Exploring these files using a GUI becomes cumbersome when you have different combination of files/folder for different samples. Therfore, use of the shell is fundamental to a wide range of advanced data analysis and high-performance computing tasks.

Before we get started executing our first UNIX commands, here are some important tips that can help you avoid some frustration:

1. UNIX is case sensitive (typically all lowercase for commands), so check your case if a command is not working
2. Oftentimes there are spaces between different parts of a UNIX command. Having spaces (or not having them) will impact whether a command runs as intended.

Typing the ssh command above gave you remote access to the Great Lakes login node. So, while you are still typing commands on your own computer, they are actually being executed on the login node! In a UNIX system each user typically has their own home directory, which is where you are placed on login. Your home directory on Great Lakes is `/home/uniqname` where uniqname will be your username that we used for logging in.

Next, we are going to learn our first UNIX command, which is among the most important and heavily used commands for beginners. `pwd` stands for print working directory and is a very handy command to quickly find out where you are in the file system. This is important for making sure you are executing commands in the correct directory (e.g. not deleting files from the wrong directory) and also for re-orienting yourself after moving around. Lets run this command and find where we actually are.

```
pwd
```

To execute a UNIX command you hit enter after typing it

My present working directory is `/home/esnitkin` , and yours should be '/home/uniqname'.

What is this actually telling us? Well, it's the path from the top of the directory structure (called the root) to the directory we are currently in. So, for me the path to the current directory takes me down the following path:

/ -> home -> esnitkin

Note that first '/' is actually a directory, and is called the root directory. Other /'s are spacers separating the names of directories along the path. In the next section we will learn how to move up and down the directory structure using the `cd` command.

Before that, let's learn one more UNIX trick. First, let's get our first error by typing pwd incorrectly:

```
#Whoops - forgot the d!

pw
```

With a short command like pwd it is pretty easy to visually see what the problem is. However, with longer commands it can be hard (and frustrating) to figure out what is wrong with your command. One trick to ensure we are typing out a valid command is to use tab complete. To see how this works type 'pw' and then 'tab-tab' (tab key two times). What hitting tab-tab does is show you valid commands or files that your shell is aware of. Now, type pwd and hit tab-tab. How does the output change?

Examining the contents of directories with ls

Now that we know how to determine what directory we are in, let's next learn how to see what files and directories are in our current directory. First, execute the following command to copy over some directories and files for us to play with. Don't worry about what this command is doing for the time being, but rest assured that you will be learning the 'cp' command very soon!

```
cp -r /scratch/epid582w22_class_root/epid582w22_class/shared_data/data/class1/shell_data/ .
```

Hmmm, when we type the command we get no feedback from the shell, so how do we know that it worked? Well, let's learn our next command, which will help us determine if the files were copied over.

The `ls` command lists the names of files in a particular Unix directory. If you type the ls command with no parameters or qualifiers, the command displays the files listed in your current working directory.

```
ls
```

You should now see that "shell_data" is in your directory. Now, the ls command might seem pretty simple, but like most UNIX commands, it actually has a lot of bells and whistles. You can access these different features by adding a flag to the command. Let's now use the '-F' flag, which will tell us which things in our directory are files and which are directories.

```
ls -F
```

Now you should see that directories end with a '/', and files (if there are any) do not. This is great, but how can you find other useful flags if you don't already know about them? To learn more about a command you can look at the man (manual) page. Let's do this for ls:

```
man ls
```

You should now see a screen that gives a short description of what ls does, along with lots of flag options to choose from! When in a man page, you have the following navigational keys available to you:

1. Up/down arrows to move up or down one line at a time
2. Space bar to move forward a full screen of text at a time
3. ctl-b to move back a full screen of text at a time
4. q to exit the man page

Let's try one more flag, the '-a' flag. What do you expect this to do based on the man page entry? How does it alter the output of the ls command?

In addition to providing ls with flags, you can also specify what directory you would like to list the contents of. For example, let's list the contents of the shell_data directory with the followng command:

```
ls shell_data
```

What's in the shell_data directory? Are the entries files or directories?

Finally, let's learn about some special UNIX characters that are super useful for directory navigation:

1. . - One dot is a special directory, namely the directory you are currently in. You'll notice in the cp command above that we used this to move something to our current directory.
2. .. - Two dots is a special directory, namely the directory above the directory you are currently in. This will be helpful in getting out of a directory once we've entered it.
3. ~ - Tilda is a stand in for your home directory.

Let's try out these special characters with the ls command.

```
#Look at the contents of the current directory
ls .

#Look at the contents of the directory above the current one
ls ..

#Look at the contents of the home directory
ls ~
```

Note the lines that start with # are comments, you do not need to execute these

Navigating directory structure with cd

So far we've learned how to figure out what directory we are in and what is in a directory. Next, let's learn how to really get into trouble by moving between different directories :). To do this we will use the cd command, which stands for change directory.

First, let's go into the shell_data directory with the following command:

```
cd shell_data
```

How do we know if the command worked? Can you now list the files in the shell_data directory?

OK - now let's traverse the file system a bit. We are going to use the special directory '..' to move up the file system until we get to the root. To do this execute the following command repeatedly until you are in the root directory, verifying your moves with 'pwd' each time:

```
cd ..
```

Pro tip - if you hit the up arrow at the prompt it will bring up previously executed commands, which prevents you from having to type them again!

Alright, you've traversed to the root of the directory tree, but now you want to go back to your home directory. How to do this? Well, there are actually several ways to get home :)

1. One directory at a time (cd home and then cd esnitkin)
2. Relative paths (cd home/esnitkin)
3. Absolute path (cd /home/esnitkin)
4. Special home character (cd ~)
5. Default cd behavior (cd by itself will default to home directory)!

Viewing files with less

Finally, let's use the less command to look at the contents of a file, and get some more experience with directory navigation. First, go to your home directory using your desired approach. Now, use 'cd' to enter the 'shell_data/sra_metadata' folder:

```
cd shell_data/sra_metadata
```

Now, let's look at the 'SraRunTable.txt' file with less:

```
less SraRunTable.txt
```

Tip - try hitting tab after hitting Sra, I think you will be pleased with the result :)

Once inside less you can use the same navigation keys as when in the man pages (e.g. arrows, space, ctl-b, q). I also want to point out one other really useful trick, which works in both less and man pages, which is searching with '/'. If you hit '/', you will see a cursor appear at the bottom of the screen. Now type WGS to search for that text. You can now see that it highlighted instances of WGS appearing. Moreover, you can move between instances by hitting 'n'.

OK, now let's go to our home directory, and use different strategies to viewing the file without entering the directory. First, go to your home directory with your command of choice. First, let's view the file using the relative path:

```
less shell_data/sra_metadata/SraRunTable.txt
```

Now see if you can view the file using:

1. Absolute pathways (i.e. the full path to the file)
2. The special ~ character

▾ Solution 1

```
less /home/uniqname/shell_data/sra_metadata/SraRunTable.txt
```

▾ Solution 2

```
less ~/shell_data/sra_metadata/SraRunTable.txt
```

To finish up let's get a little more practice with navigation and directories.

What are different ways to get into the directory '/home/esnitkin/shell_data/untrimmed_fastq' from the home directory '/home/esnitkin'?

▾ Solution

```
   #Relative path
   cd shell_data/untrimmed_fastq

   #Absolute path
  cd /home/esnitkin/shell_data/untrimmed_fastq

   #Tilda home shortcut
  cd ~/shell_data/untrimmed_fastq
```

How can we view the file '/home/esnitkin/shell_data/sra_metadata/SraRunTable.txt', while we are in the untrimmed_fastq directory?

▾ Solution

```
  less ../sra_metadata/SraRunTable.txt
```

Data Carpentry: Introducing the Shell & Navigating Files and Directories

To learn more about today's lesson check out the following resources Introducing the Shell and Navigating Files and Directories to go over some basic unix commands and learn how to navigate directories. On the Canvas site we also have links to videos from a workshop we co-led where the instructors go through these and other lessons that will be useful in this course.