BCB/EEOB546: UNIX Exercise 1

While you may be able to complete this exercise without issue on your personal laptops using GitBash for Windows or in your Mac Terminal application, I would recommend copying the `SNPs` folder in the `Week_01` folder to the student cluster hpc-class. I have tested this assignment in this UNIX environment, and you should not run into problems there.

First, log into hpc-class by typing the following into your Terminal or GitBash application:

```
 ssh <ISU NetID>@hpc-class.its.iastate.edu
```

The first time you login to hpc-class, it will respond with "host key not found, generate hostkey?(yes/no)". Answer yes.

The password for hpc-class.its.iastate.edu is the same as that for CyMail.

To add the `SNPs` folder necessary for this exercise to hpc-class, let's clone the course repository in your directory:

```
git clone https://github.com/EEOB-BioData/BCB546-Spring2022
```

Now that you have the data you need in hpc-class, let's do something with it!

- Use the `cd` command to navigate your way into the `SNPs` folder
- Make sure you are inside the folder by using the `pwd` command
- Now use the `ls` (list) command to list the contents of the `SNPs` folder; you should see four yabba-dabba-doo subfolders
- These folders each contain a single text file. Each of your collaborators (Fred, Wilma, Barney and Betty) have identified the number of SNPs in fifteen exons. Their files contain two columns: one with a unique ID for their exons and one with the number of SNPs/exon. Our goal is to determine which of the 60 exons investigated by our four collaborators has the most SNPs (and to learn some UNIX along the way!).
- from your current location in the `SNPs` folder, `cd` into the BarneySNPs folder
- use `ls` to observe the contents of the folder
- Use the `cat` (concatenate) command (`cat BarneySNPs.txt`) to print the entire contents of the

text file

- try the `head` command on your text file (`head BarneySNPs.txt`) to print just the first portion of the file. How many lines of text are printed?
- Now try `head -5` . Any difference? How does this change if you vary the number? Understand what's going on here?
- Now let's try `tail` . What portion of the file is printed when you run this command on the text file?
- Modify your command to `tail -n +2` . This should print everything but the first row of your text file (*i.e.*, the header). This command will come in handy later on.
- Now let's `cd` back into the SNPs folder. If you type `cd ..` , this will move you up one directory relative to your current directory; type `pwd` to verify this is the case; are you in the `SNPs` folder?
- It's a little messy with all these folders and files. First, let's move all our text files into a single new folder we will name `AllSNPs` . To create this folder use the command `mkdir` (make directory); type `mkdir AllSNPs`
- now use `ls` to see if your new directory has appeared
- you can copy all of your text files in the Flintstones' folders using the `cp` (copy) command in a single step by utilizing wild-card characters as follows:

```
cp *SNPs/*.txt AllSNPs/
```

- The wild asterisk matches all folders with names ending in "SNPs" and also the .txt files therein and these are then copied to the `AllSNPs` folder
- `cd` into your `AllSNPs` folder and with an `ls` verify that all the text files are now there
- O.k., all of our files are now in a single folder, let's clean house. We're going to use a very handy but potentially dangerous command here, so be careful!
- First `cd` up a directory (you can use the `cd ..` syntax we learned above), when you `ls` now you should see all those messy folders
- Try using the `rmdir` (remove directory) command on the `BarneySNPs` folder. Won't let you, will it? That's because `rmdir` only works on empty directories.
- There's a way around this. **MAKE SURE** you are in the SNPs folder by typing `pwd` . You should see: `/home/<your user name>/SNPs` . Now type: `rm -r BarneySNPs` . You just recursively removed

`BarneySNPs` ...that means you removed that directory and all of its subdirectories and files (which is just a single text file here). Use the `ls` command to make sure BarneySNPs has disappeared. Now, imagine the carnage if you typed this command on your User directory on your personal machine! Goodbye years of hard work and data!!! Let's take a few deep breaths after imagining that and use this handy but scary command on Betty, Fred and Wilma. Finally, use `ls` to confirm all these folders have been removed.

- Now `cd` into the `AllSNPs` folder and print to your screen the contents of all files simultaneously by typing `cat *`
- We could just look through these data very carefully at this point and find the exon with the most SNPs, but let's eliminate potential human error by going a bit further; we want to produce a single file with everyone's data and sort this by the second column which includes the SNP data
- First, we need to remove the header line at the top of each file (the line with the column names). We'll do this with syntax that could greatly improve your quality of life, a `for loop`!
- Heres the syntax:

```
for i in *.txt; do tail -n +2 $i >> $i.tail; done
```

- And here's what it's doing: each of the files (matched with the wild asterisk, `*.txt`) is temporarily assigned to the variable `i`, then, in the `do` portion of the loop, the `tail -n +2` command is applied (remember, that leaves us with everything but the header). The output of that command is then written to a new file that will have the same name as the original file but with `.tail` appended at the end. The loop will continue across all files matched by `*.txt` and then quit
- After running your for loop, type `ls` and see if your brand new "headless" files are in your directory
- Check to see that the header has been removed by using `cat` on one of the `.tail` files
- Now let's combine these headless files into a single file by redirecting the output of `cat` to a text file. Here's the syntax:

```
cat *.tail > AllSNPs.txt
```

- The syntax above matches all files ending in `.tail` (the files we're interested in) and through `cat` directs their content to the file we create called `AllSNPs.txt`
- Let's convince ourselves this has really worked using a new command: `wc` (stands for word count). Type `wc *` from your current directory; the first column of the output tells you how many lines there are in each file in your directory. How many lines does `AllSNPs.txt` have? Convinced?

- Great, we have the file we need to figure out which exon has the most SNPs, now we need to sort it. Let's use the `sort` command for this as follows:

```
sort -k2 AllSNPs.txt
```

- the `-k2` argument tells your UNIX kernel to sort the file based on the second column (the SNP column); is this the sort we were after? **NO**! What's wrong? Sadly, we've sorted numbers as if they were text

- Let's try again, now with an additional argument:

```
sort -k2 -n AllSNPs.txt
```

- the `-n` argument tells your UNIX kernel to sort these values numerically; bingo, that's what we wanted, but we had to scroll down through a bunch of values to find that Fred's eighth exon had the most SNPs (198). Let's use a pipe to get only the value we're after:

```
sort -k2 -n AllSNPs.txt | tail -1
```

- The pipe is the character `|`. This tells your UNIX kernel to take the output of your `sort` and send that to the `tail -1` command which leaves you with only one exon; Fred wins with 198 SNPs in his eight exon....yabba dabba doo!!

- **Congratulations, you're on your way with UNIX!**