**Cas Carroll – BIOL 792 – Unix HW 1**

**Part I: Questions**

1. *What is a Graphical User Interface (GUI) and how does that differ from the Command Line Interface (CLI)?*

A GUI is something like what you encounter when you open your finder on a mac – the computer presents you with directories which you can click on and use to move through all of your files and directories. GUIs can also The CLI is what you encounter when you open the terminal. You can navigate through your files and directories as you can with the GUI, but you can also use the CLI to interact with files that would be too large for you to open and work with. When using the CLI, you must be able to communicate with your computer using the unix/linux languages.

1. *What does the shell do?*

Users can run commands in the shell, which the computer executes and usually gives an output (i.e. using the command ls will return a list of directories and files within the directory you are currently in).

1. *What is the command used for listing things in a directory?*

**ls**

1. *What is the command used for changing directories?*

**cd**

1. *What command would you use to get your current working directory?*

**pwd**

1. *How do you get the manual for these commands?*

**man [command]**

1. *What does the shell prompt look like?*

For bash, the shell prompt is “$”, but for zsh it is “%”.

1. *How would you list things in a directory in chronological order?*

**ls -tl** lists your directory contents in chronological order. “t” lists your directory contents by time, and “l” prints the long form so that you can see the dates.

1. *Name two ways to get to the ‘home’ or ‘root’ directory.*

**~** and **cd ~** take you to your home directory. If you know where you are relative to your home directory (I’m assuming you are below your home directory in this situation) you can also use **cd ../../** to navigate up directories and return to home. That command would work if you were two directories below your home directory (i.e. if you were in Users/cascarroll/Desktop/CWSsdm/ you could use that command to return to home).

1. *What is the difference between an absolute path and a relative path?*

An absolute path is what your computer would return if you input the command **pwd**. A relative path is relative to where you are in your file system. **.** allows you to stay in your current directory while **..** allows you to go up one directory.

1. *What are the two relative path directories we talked about and what do they mean?*

**.** allows you to stay in your current directory while **..** allows you to go up one directory.

1. *Why shouldn’t you put spaces in filenames?*

It is difficult for unix/linux to interpret filenames with spaces.

1. *Name a way to have multiple readable words in a filename without spaces?*

You can use underscores (“\_”) between words or you can capitalize the first letter of each word (i.e. Butterfly\_counts\_2021.xlsx or ButterflyCounts2021.xlsx)

1. *What is the program nano, what does it do? Do you use nano? If not what do you use?*

Nano is the built-in text editor for the terminal/shell. I used to use nano and I’m comfortable with it, but it isn’t the most efficient text editor. I usually use Visual Studio Code or Spyder. I’ve also used BBedit and Atom.

1. *\* is a form of a wildcard. What does it mean?*

The wildcard can be used to return all the files that follow a pattern. You could use \*.fastq to do something with all the fastq files in a directory, or you could butterfl\* to return any files that start with that string of characters (i.e. butterfly\_sequences.fastq, butterflies.txt, etc.)

1. *The following code would match what? ls \*.txt*

The code would print all the files in the directory that end with .txt.

1. *Why do we need to be careful with the mv command?*

You could accidentally rename the wrong file unless you have a safety in place. For example, I have an alias set up for mv (mv -i) so that it prompts me for confirmation before renaming a file.

1. *Why do we need to be careful with the rm command?*

You can accidentally delete files if you don’t have any safeties in place. For example, I have an alias set up in my .zshrc file so that I need to confirm that I want to delete files before they are deleted (rm -i).

1. *What is the difference between > and >>?*

*>* writes the results of the command to a file and >> appends the results of a command to a file.

1. *What does head do?*

head prints the first few lines of a file.

1. *What is the purpose of | (pipe)?*

A pipe allows you to run a sequence of commands instead of one command per line of code. For example, you could use a pipe between two commands that allow you to call and filter SNPs using bcftools.

1. *Interpret the following command:*

cat huge\_file.fasta | uniq | head -n 5 >proteins.fasta

The first part of the code (cat to the first pipe) reads through huge\_file.fasta, uniq filters out duplicates in the fasta file, head -n 5 prints the first five lines of the file and >proteins.fasta writes those lines to a file called proteins.fasta.

1. *What is a loop and when would you use it?*

A loop is used to process many files at a time. If you wanted to, for example, search for a certain sequence of nucleotides in many .fasta files, you would use a for loop.

1. *In a directory with the following files. What would the following loop do? apple.txt, banana.txt, orange.txt $ for filename in \*.txt; do > cat ${filename} >>fruits.txt > done*

The loop combines apple.txt, banana.txt, and orange.txt into one file called fruits.txt.

1. *What is a shell script?*

A shell script script is a program, which is a series of instructions that are interpreted by bash or, in the case of new macs, zsh. You can save scripts as .sh files to use repeatedly.

1. *What are the benefits of writing and running a script over typing the code in?*

Scripts can be long and complicated instructions. If you use a script instead of typing the code in, you can save time on routine tasks that you do regularly.

1. *When looking at a script what does # mean? Why would you use one?*

# is the pound sign, and it is usually followed by an exclamation point on the first line of a script. Every bash script starts with #!/usr/bin/[the interpreter you want to use]. You don’t necessarily need to use this for scripts you aren’t sharing with folks, because you will know which interpreter to use based on the file extension (i.e. .py for Python, .sh for bash/zsh) and you can run something like “python3 myscript.py” from the command line, but if you are sharing with folks, it is good practice as it tells the user which language needs to interpret the script.

1. *What does grep stand for and what does it do?*

grep stands for “Global Regular Expression Print”, and it is used to search for patterns in a file (i.e. a sequence of interest in a genome).

1. *What does find do?*

find searches through the files below you in the file hierarchy.

## Part II - Writing Code

1. Open the shell and change directories to the desktop and list the contents, paste your code here.

cd Desktop

ls

1. Draw the file structure from the root directory into one of the folders on the desktop. You can submit a pdf with a drawing. Select three locations and type the commands for changing directories to those locations use a combination of absolute and relative paths.

Diagram

Description automatically generated

(from home) cd Desktop/spring22/

(from GrantApps to home) cd ../../

(from CWSproj to CWS\_SDM) cd ../../Documents/CWS\_SDM/

1. On your computer create a directory on your desktop. Create two files without opening them that have file extensions .txt. Then use wildcards to list the files in that directory. Paste you code here.

mkdir test

cd test/

touch hello.txt goodbye.txt

ls \*.txt

1. Download the file [Hutia\_DNA.fasta](https://www.dropbox.com/s/07y610c8zt63nt5/Hutia_DNA.fasta?dl=0) from the Github repo. Make a new directory on the desktop. Move the file into that directory. List the contents of the directory. How many lines are in this file? Fasta files are a type of data file that holds DNA sequences. They are all formatted the same. The first line starts with a > and a name followed by a line break, and then DNA sequence followed by a line break. The next line starts with an > and another name followed by a line break and the DNA sequence. With that in mind, how could you use linux commands to determine how many sequences are in this file? How many sequences are in this file?

I used wc -l Hutia\_DNA.fasta to count the number of lines. There are 2382004 lines in the file.

I used grep -cn “>” Hutia\_DNA.fasta to fine the number of DNA sequences and the output was 1191002.

1. Build a single line of code that would take the first 100 sequences of this file and put them in a new file called ‘MyOutputFile.fasta’. Use a combination of wc, sort and uniq in a single line of code to tell me how many unique lines are in this file. Paste your commands and the answer here.

head -n 200 > MyOutputFile.fasta #I use “-n 200” because I want the name and the sequence.

sort -u Hutia\_DNA.fasta | uniq | wc -l

There are 2324870 unique lines in the file.

1. Write code that would create a loop to copy all the files in one directory ending in .txt to another directory. Paste the code here.

I used the following code to move three .txt files in Users/cascarroll/Desktop/test to Users/cascarroll/Desktop/targetdest:

mv \*.txt ../targetdest/

1. Write a bash script with in-line documentation (hint #) to show how to run a made up python program (script) called 'find\_taxa.py' on set of files ending in ‘.fasta’. Show in the script how to run the program on each of those files. What is the name of your script? Type the script here.

Script name: run\_find\_taxa.sh

#!/usr/bin/bash

#tells the computer to use bash as the interpreter

for file in \*.fasta

#initiating for loop to loop through .fasta files

do

python3 find\_taxa.py $file

done

#runs the python program “find\_taxa.py”

#I add python 3 before the script because python will be the interpreter

#not bash

1. From the Hutia\_DNA.fasta file tell me how on how many lines do we find the pattern ‘GAGA’. What was the code used to find this?

grep -cn "GAGA" Hutia\_DNA.fasta

304806