Programming for Bioinformatics | BIOL 7200

Exercise 1

The goal of these exercises is to get you used to working with some basic UNIX commands and their options. In addition to the exercises here you are strongly encouraged to experiment with these commands with your own data (either real or made up) to better understand how they work.

Most of the questions here can be answered using commands we covered in class. However, some questions can only be answered by looking online. When searching for information about how to perform certain tasks in Bash, pay attention to which keywords in your search yield the most relevant results. Sometimes the key is learning which words others use to refer to a function or task.

Here is a reminder of commands covered in class this week:

Command	**Function**
`ls`	List files and directories at a path
`pwd`	Print working directory absolute path
`cd`	Change directory
`mkdir`	Make directory
`touch`	Create an empty file
`rm`	Remove files or directories
`mv`	Move files or directories
`cp`	Copy files or directories
`echo`	Print something
`cat`	Concatenate files
`>`, `>>`, `|`	Redirect, append, and pipe stdout
`man`	Display manual page for command
`clear`	Clear commands and outputs from your terminal
`head`	Return top N lines (default 10)
`tail`	Return bottom N lines (default 10)
`wc`	Count lines, words, or bytes in input
`paste`	Combine input in columns
`join`	Combine input using key column
`grep`	Search input for strings or patterns

Command	Function
`cut`	Extract a range of columns or characters from input
`sort`	Sort input
`uniq`	Return unique elements in input
`sed`	Edit a stream of text
`awk`	Perform complex operations on input

Instructions for submission

- Run a Linux or Mac terminal on your computer
- You may want to create a directory to work in (e.g., "~/biol7200/exercises/ex1")
- Download "ex1.bed" into your working directory (e.g., "~/biol7200/exercises/ex1")
- Prepare a file containing answers to the below questions. Copy the question and either write the correct answers or provide a screenshot of your working below the question
- In cases where you need to paste a screenshot of your terminal, you can do that as follows. Screenshot a selected area (Windows: win+shift+S, Mac: cmd+shift+4) and then paste in your submission sheet
- Use the `clear` command to tidy up your terminal so you show only relevant commands and outputs
- Name your submission sheet: "gtusername.pdf", or "gtusername.docx"
- Questions 1-4 cover Tuesday's material
- Questions 5-8 cover Thursday's material
- All questions must be answered using Bash utilities (including `awk`). Solutions that use other scripting languages (e.g., perl, python, etc.) will not be accepted.

Grading Rubric

This assignment will be graded out of 100.

- 20 points for correctly naming and submitting your completed assignments
- 80 points for questions (10 per question)

Questions

1. Using documentation to explore functionality of `ls`
    - List the files in your home directory
    - Create two empty files in your home directory. One named "file1" and one named ".hidden_file" (note the dot in the second name)
    - List **all** the files in your home directory sorted with oldest first (hint: "all" means the hidden file you made should be listed)
    - What is the size of file1? Show your working
    - What is the size of ".hidden_file"?
2. Creating and viewing file contents using the terminal
    - Add two lines of text to "file1"
    - View the contents of "file1" in your terminal

3. Copying and removing files

    - Use `cp` to copy "file1" to "file1_copy.txt"
    - Has the addition of ".txt" to the file name changed how the file contents look? Are file extensions significant in Unix systems?
    - Use `rm` to remove "file1"
    - Create an empty file named "file2"
    - Run the command `cp -n file1_copy.txt file2`. Does "file2" now contain the same contents as "file1_copy.txt"? Explain why or why not

4. Using documentation to explore useful commands. State the command and options you could use to perform the following tasks:

    - Create a directory structure "~/a/b/c" in a single command (i.e., create a directory and any missing parent directories)
    - Check if a file has Windows- or Unix-line endings
    - Copy files but only replace existing files if they are older than the source file
    - Check if whitespace characters in a file are tabs or spaces
    - View the last 5 commands you issued

5. For each row of the below table, provide a glob or extended glob pattern that would match the set of filenames in the second column and not match the set of filenames in the third column. Give 1 pattern for each row of the table

#	**Match these strings**	**Don't match these strings**
1	README.txt, data.tsv, figure.tiff	Homework.pdf, data_to_analyze/, doc.rtf
2	SRR124515, ERR123252, SRR3161371316	PRR161356, LRR124636, error.txt
3	File.txt, another.pdf	temp.csv, data.csv
4	sample_reads_1.fastq, sample_reads_2.fastq, SRR1352235_1.fq, SRR1352235_2.fq	sample_assembly.fasta, SRR1352235_assembly.fasta, sample_feats.bed, SRR1352235_feats.bed, longreads.fastq
5	Samples/a/assembly.fasta, Samples/b/assembly.fasta	assembly.fasta, Samples/assembly.fasta
4	sample_reads_1.fastq, sample_reads_2.fastq, SRR1352235_1.fq, SRR1352235_2.fq	sample_assembly.fasta, SRR1352235_assembly.fasta, sample_feats.bed, SRR1352235_feats.bed, longreads.fastq
5	Samples/a/assembly.fasta, Samples/b/assembly.fasta	assembly.fasta, Samples/assembly.fasta

6. Redirecting outputs

    - Pick a command that produces stdout, run it, and direct its stdout to a file
    - `ls` a path that does not exist in your current directory. Which output stream does the message you see come from?

- Rerun the command from step 2, but now direct the output to a file
- In a single command, `ls` both a path that does not exist (e.g., "~/not/a/real/path") and "./" (i.e., provide two positional inputs). Direct the stdout to one file and the stderr to another file (still within a single command - no use of `;`).
- Use `grep` to find the help message entry for the "-l" option of `ls` (hint: "-" is a special character interpreted by bash so you need to get around that somehow)
- How many commands are there in your "/bin" dir?

7. Data cleaning. Bioinformaticians often have to work with data generated by others. Perform the following operations to tidy the data in file "ex1.bed" provided on Canvas

    - Check if the file uses windows line endings instead of unix line endings
    - Remove the windows line endings and output the new version to a new file, preserving the original file (always good practice)
    - Remove the header lines starting with "#" and output the new version to a new file

8. Summarizing real data using bash commands. The following questions relate to the cleaned version of the "ex1.bed" file you generated above. BED format is a commonly used format for storing the location of features in an assembly. The provided file includes the three mandatory columns of a BED file: Sequence ID, start, and stop positions (further description of BED format can be found here). Using Bash commands answer the following questions about these data

    - How many unique sequence IDs are present in the file?
    - How many different start positions are there?
    - What is the highest number of features starting at the same start position?
    - How many features start in the first 10Kbp of the sequence?
    - How many features start and end in the first 10Kbp of the sequence?

      EXTRA CREDIT (5 points)

    - Which feature is the largest? Show your work