
Lab 8: The Shell

This assignment is designed to reinforce and build on what you have learned about the shell, and demonstrate how you can write programs using command-line tools. The commands you will need to complete it are **mkdir**, **cd**, **cp**, **grep**, **cat**, **wc**, **|**, and **>**.

Figure out how to do each of these tasks, **in the order they are given** (the output from one task is often used as the input to the next task):

1. Create a new file called **command.txt** in the directory **Lab8-Shell-F20**. The commands you write for parts 2 - 12 should be stored in this file.

- You can use the normal Jupyter interface to create and edit this text file if you wish.
- You could also use the history command to find your command history and remind yourself of the commands you used.

2. Open a **Terminal** in Jupyter and enter the directory **Lab8-Shell-F20** using the command **cd**.

3. Copy the following three sequence data files to your directory **Lab8-Shell-F20**. The file paths for the three data files are:

```
~charliep/courses/cs128/first.dat
```

```
~charliep/courses/cs128/second.dat
```

```
~charliep/courses/cs128/third.dat
```

These are small subsets of the actual output (FASTQ, Illumina format) from sequencing 16S rRNA genes extracted from soil microbes at an archaeological site in Iceland. It is about 1/2,000,000th of the total sequencer output from eight samples. Each entry, called a read, contains four lines. The first starts with @ and contains the unique ID, the second is the actual read (DNA), the + is a separator, and the last line is the quality data for the read.

4. Find all the lines from **first.dat** and **second.dat** that contain the string **'AACCTTNN'** and save them into one file called **fourth.dat**. All of those sequences should end up together in one file called **fourth.dat**. **Hint:**

- **cat** has an additional use that we haven't seen yet. "**cat file1 file2**" will output the contents of both of those files, in order. By default, this output will go to the screen, but we can pipe it to another command.
- Think about how you can find all of the lines that contain a certain sequence.
- Think about how you can make the output of that command end up in a new file.

5. Append the contents of `third.dat` **to** `fourth.dat` to create `fifth.dat` . (The contents of `third.dat` should come after those of `fourth.dat` .)
6. Save the number of lines, words and characters in `fifth.dat` to a new file `fifth-count.txt` .
7. Save all the lines in `fifth.dat` that do not contain the string `'AACCTTNN'` in one file called `sixth.dat`
 - Check to see if there are special flags that can be given to `grep` to make this task easier.
8. Using one command line, display the number of **lines** in `fourth.dat` and `sixth.dat` combined. You should only display the number of lines, and nothing else.
9. Using one command line, extract sequences in `first.dat` , `second.dat` , and `third.dat` that contain the string `'ATG'` **but do not** contain the string `'TAG'` . The results should end up together in one file called `seventh.dat` .
10. Using one command line, extract sequences in `first.dat` , `second.dat` , and `third.dat` that contain `'CAT'` and contain `'TAG'` . The results should end-up together in one file called `eighth.dat` .
11. Using one command line, display only the number of **words** in `seventh.dat` and `eighth.dat` .
12. Use `history > history.txt` to create an archive of your command history. **You need to do this last; if you do other things after doing this, then you should do it again so you have an updated archive of your command history.**

Note for future: add to the assignment something about getting the tail and head of a file. Also, perhaps make them use the help (i.e. documentation). (Catch a man a fish and he'll eat one meal. Teach him to fish and he'll eat for a lifetime....)

All of these tasks can be accomplished with basic terminal commands, and while they could also be done with Python the point of this exercise is shell programming so let's stick with terminal for now.

This part is **optional**, and requires doing a lot of things we haven't discussed at all in class. Email the instructor if you are interested. You may wish to collect all the command-lines you figure out into a single Bash script, ordered as above, that can be run with the command `$./script-name.sh`