Programming for Bioinformatics | BIOL7200

Week 3 Exercise

September 1, 2020

The goal of these exercises is to get you used to working with some basic UNIX commands and their options. Try and think about what each command does, and where it can be used. Reading the `man` page for each of these is **recommended**.

We're going to cover the following commands:

Command	Explanation
`cut`	Extract some columns/characters/bytes from a file
`paste`	Paste together files side by side
`grep`	Find patterns in a file/files/stream
`sed`	Find and replace patterns in a file/files/stream
`tr`	Translate – *not covered this week* but has (almost) similar behavior to `sed`; `tr` makes changing or deleting characters easier than `sed`
`apt-get`	Handle system packages
`make`	Build executable programs
`sudo`	Execute a command with elevated privileges

Instructions for submission

- Prepare two solution sheets for this exercise - one for submission and another for your own reference. **Only submit the submission sheet.** The solution sheet you create for your reference should help you going forward - make it as detailed/brief as you'd like for your own learning style. For the submission sheet, copy the question and write the correct answers below the question.

Exercises

Regular expressions is one of the most testing topics in bioinformatics - but if you can master it, you can master the art of file processing. This week we will learn efficient ways of processing files using advanced techniques.

Regardless of your specific areas of research, file manipulations techniques will always be useful. If there is only one thing that you can learn from this class, let it be this. We'll be happy to go over any concept that you are still struggling with in Thursday's discussion session.

As you do these regex exercises, please keep in mind that most of them are designed to have simple commands. They are designed to challenge your conceptual understanding of the subject.

General regex

1. Create regular expressions for the following; this is a theoretical exercise, but you're welcome to try out the regex using `grep`:
   1. Only a number that is a multiple of 5
   2. Exactly 5 characters
   3. Any letter followed by a number
   4. The first 3 columns of a BED file (Google UCSC BED format to find out the specifications of the standard BED format)
   5. The first 3 bases in a DNA sequence
   6. The last 3 bases in a DNA sequence
   7. Two numbers followed by 2 lower case letters
   8. What does this regular expression match? `\d*\.\d{3}`

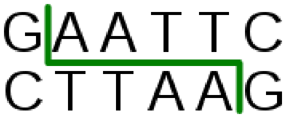
Regular expression command exercises

2. Searching a file with `grep`
   1. Extract the `knownGene.txt.gz` from the files you downloaded from Canvas. Google the command if you don't know how to extract it.
   2. Use `grep` to get all genes on chr22
   3. Use `grep` to get all and only those genes that occur on chr1

3. Editing data streams with `sed`
   1. Take the results from `2.2` and duplicate each line
   2. Change the `chr` position of every other line to `cow`
   3. Delete the lines that have `cow` in them
   4. Repeat `1-3`, but this time do it "in-place". Read the `man` page to figure out what this means.

Biologically-inspired problem

4. An *in silico* restriction enzyme digestion.
   In a parallel universe, restriction enzymes are called `sed`, and cut microbial genomes on specific patterns. One such enzyme has magically found its way to your computer. Download the `M07149.fasta` from Canvas; we've got some cutting to do!
   1. The restriction enzyme works on the pattern `GAATTC` and cuts right after the G like this:



Cut the genome into pieces using this restriction enzyme (**sed**)! Store the fragmented genome in a new file. How many pieces did you get? (Don't count this manually – use a command like **wc**).

2. Upon further investigation, you found that the restriction enzyme is a little flexible. It can actually cut after the first base in the following patterns:

   ```
   GAATTC, GAATTG,
   GATTTC, GATTTG,
   CAATTC, CAATTG,
   CATTTC, CATTTG
   ```

   Update your pattern to cut the genome accordingly. How many pieces did you get this time?

3. You underestimated the strength of this enzyme – it can also vary its length. The updated list of patterns has the following letters being optional: third (A or T), fourth (T) and last (C or G). Update the pattern to get the new number of pieces. How many did you get this time?

Harder installation problem

Continuing our installation discussion from last week, this week we will install MySQL without using root. MySQL is a relational database management system. If that doesn't mean anything to you right now that's okay, but databases are extremely useful in bioinformatics. I recommend relational databases (taught in CS 4400) for everybody. MySQL is also a good example for typical compilation/installation.

1. Download the latest source code for MySQL (http://dev.mysql.com/downloads/mysql/), not the precompiled binaries.
2. Next step requires **cmake**. What is **cmake**?
3. Unpack the source and run **cmake** . in the directory you just created. If you don't have **cmake** in your system, get it using **apt-get**. Don't attempt **cmake** install without root, it's a harder install.
4. Build the MySQL executables with **make**
5. Try to install them with **make install**
6. That should have failed. Why?
7. How would you get around this with **sudo**? How would you get around this with **cmake**? (Hint: you have to tell **cmake** where YOUR bin directory is. Run **cmake --help**)