**Assignment #11**

**Objective**

Learn about Regular Expressions and UNIX command-line data wrangling tools

**Due**

Before 9am on Tuesday March 27th. Commit your results to your GitHub repository. And when completed let Dr. Ficklin know via Slack that it is available. Please indicate the amount of time the assignment took to complete.

**Setup**

This assignment requires a UNIX (Linux) terminal. If you have been strictly using Windows for this course you will need access to some UNIX-based machine. The simplest method is to download and install the MobaXterm at <https://mobaxterm.mobatek.net/download.html>. Download the free Home Edition. After you download and install MobaXterm you can gain access to a linux-style terminal by clicking the button in the center of the program with the label “Start local terminal”.

**Tasks**

Task A: Regular Expressions

Read through the tutorial on regular expressions at this site: <https://ryanstutorials.net/regular-expressions-tutorial/>. This tutorial has no exercises but does have plenty of examples. Take time to examine each example and make sure you understand it.

Unfortunately, there is no set syntax for regular expressions. Each programming language may introduce differences in the syntax of regular expressions. Therefore, the tutorial above is great as an introduction but it is a good idea to review the rules for the language you are using. Next, read through the instructions for to use regular expressions in Python at this page: <https://www.machinelearningplus.com/python-regex-tutorial-examples/> . In section 10 of this tutorial are several exercises that have both questions and answers. Create a new Jupyter notebook and perform these exercises. The answers are provided but try to complete the exercise without looking at the solution. When completed, upload your notebook to your GitHub repository.

Task B: Using grep, sed and awk

The UNIX tools **grep**, **sed** and **awk** are powerful command-line tools that allow you to find, replace and reformat data. You can do a lot of text manipulation and data wrangling on the UNIX command-line using quick “one-liners” that combine these tools. Often this can be faster for some tasks than writing Python scripts.

Before we learn about grep, sed and awk we need to learn a bit about pipes and redirects on the UNIX command-line. Read the following reference: <http://www.westwind.com/reference/os-x/commandline/pipes.html>. This reference is for Mac’s OSX UNIX system but is sufficiently generic for all UNIX varieties.

The first tool we will examine is grep. It allows for searching of text within files. Read about grep here: <http://www.compciv.org/recipes/grep/basics-of-grep/> . Note that regular expressions in grep are somewhat different from Python.

Next, the sed tool provides a bit more functionality over grep in that it supports altering of text files. Read about sed here: <https://www.digitalocean.com/community/tutorials/the-basics-of-using-the-sed-stream-editor-to-manipulate-text-in-linux>. Note that regular expressions in sed are somewhat different from Python.

Next, awk is a tool that treats contents of files as “records” (i.e. tab-delimited files) and provides methods for manipulating those files. Read this introduction to awk here: <http://tldp.org/LDP/Bash-Beginners-Guide/html/chap_06.html>

Together, grep, awk and sed allow you to “wrangle” data on the command-line. The tutorials you read here provided a basic introduction to the most common applications for these tools. They can do much more!

**Exercise:** The gene ontology provides a controlled vocabulary of terms that describe the molecular function, cellular component and biological processes in which genes are involved. We have discussed this briefly in class. These terms can be assigned to genes to indicate the function and locations of their products. You can download the entire gene ontology in the format of an OBO file. Subsets (or slims) of the gene ontology can also be downloaded. As part of this assignment a GO slim in the OBO format has been provided for you on slack (goslim\_plant.obo). Take a look at this file. Each term is defined using a unique [Term] section. As you scroll to the bottom there are some relationship terms within [Typedef] sections.

Using the knowledge of the tools you learned, convert the contents of that file into a tab-delimited file containing four columns: The ID, name, namespace and definition of each term. You will see these items as the ‘id:’, ‘name:’, ‘namespace:’ and ‘def:’ respectively. Do not include the terms that are of [Typedef]. This will take some trial and error, some creativity and time. But spending the time thinking on it will help solidify the use of these tools and regular expressions. Try to do this all on one command-line using the UNIX pipe command ‘|’. Save your solution in a file named **obo\_convert.sh** and upload it to your GitHub repository. If you are not able to find a solution, upload as far as you got.

**Hints:**

1. You can use the command **egrep** to search for multiple matches at once. Just separate each item with the ‘|’ (bar) character… it means “or”. For example, egrep 'cow|dog' will find either cow or dog in any line.
2. At some point you may want to remove the invisible line-feeds at the end of each line. Sed does not make this easy to do, but here’s the command to do it: sed ':a;N;$!ba;s/\n//g'. That’s not pretty, but it works.
3. For this you will need both **sed** and **grep**. But remember that both have slight differences in how they use regular expressions. If you get confused about how their regular expressions are formed, take some time to look it up.

**Solution:** This is just one solution (upside down), there are probably other ways to do this exercise. If you need to look, take time to examine each step of the full command to make sure you understand.

egrep "\[Term\]|\[Typedef\]|^id:|^name:|^def:|^namespace:" goslim\_plant.obo | sed 's/\[Term\]/||/g' | sed 's/\[Typedef\]/==/g' | sed 's/id: /--/g' | sed 's/name: /--/g' | sed 's/def: /--/g' | sed 's/namespace: /--/g' | sed ':a;N;$!ba;s/\n//g' | sed 's/||/\n/g' | sed 's/==/\n==/g' | sed 's/--/\t/g' | grep -v '^==' | sed 's/^\t//' > goslim\_plant.terms.txt