Bioinformatics DevOps Engineer Coding Challenge

Kelly Harrington
December 3, 2019

Linux:

1. What does ls -alt do?

   ls -alt provides a list of files in a directory with three commands specifying how the list is presented. -a is a command that allows all files including those beginning with '.' to appear in the list. -l provides the long list format, describing properties of the file such as including username, file name and timestamp of last modification. -t sorts by the timestamp.

2. What command would you use to list all files starting with 'Run' and ending with '.txt' in a directory and all of its subdirectories?

   Once in the directory you want to search, use:

   printf '%s\n' Run*.txt

3. How would you append the contents of 'exampleFile1.txt' to 'exampleFile2.txt'?

   cat exampleFile2.txt >> exampleFile1.txt

4. How would you (1) sort the contents of 'exampleFile1' and (2) redirect the sorted content to 'exampleFile2.txt' in one line using the pipe operator?

   sort exampleFile1 | cat exampleFile1 >> exampleFile2.txt

5. Which commands would you use to find files whose name match a certain pattern, and to find files containing a certain text?

   To find files whose name matched a certain pattern, I would incorporate wildcards (*) into the desired pattern. So, if I wanted a list of all the files that had the letters 'an' in it, I would use:

   printf '%s\n' *an*

   The find command can also be used here:

   Find ~ -type f -print | grep an

   If I wanted to explore a different pattern, for example, finding a list of all the filenames that rhymed with 'tail', I would use:

   printf '%s\n' *ail

   To find all files containing a certain text, I would use:

   grep -r "TEXT" ~

   and replace TEXT with the appropriate text.
   If only looking in a certain directory for that text, ~ could be replaced with the relative path.

SQL:

1. For the following SQL statement, what is wrong with it and how would you fix it:

```
-- Question:
SELECT UserId, AVG(Total) AS AvgOrderTotal
FROM Invoices
HAVING COUNT(OrderId) >= 1
```

As written, this statement is not properly calculating an average as there is no clause grouping the data to aggregate an average. Adding a GROUPBY clause can remedy this. Also, a semicolon is needed following the statement.

```
SELECT
        UserId,
        AVG(Total) AS AvgOrderTotal
FROM
        Invoices

GROUPBY
        UserID

HAVING COUNT(OrderId) >= 1;
```

Bioinformatics:

1. Recursively find all FASTQ files in a directory and report each file name and the percent of sequences in that file that are greater than 30 nucleotides long.

```
from Bio import SeqIO

fastq_seq = SeqIO.parse(r"FILEPATH\Sample_R2.fastq", "fastq")

def greater_than_30(file):
   long_l = []
   total_l = []
   for each in file:
      total_l.append(each)
      if len(each) > 30:
         long_l.append(each)
   for each in long_l:
      print(each.name)

   return ('% of sequences with more than 30 nucleotides:', (len(long_l)/len(total_l))*100)
greater_than_30(fastq_seq)
```

To run in Linux Command Line, change the directory to the directory that includes the .py file (use ‘cd’ command to do so). Then:

python3 FILENAME.py

Assuming the correct dependencies are installed, the program will run on Linux or Windows.