Programming for Bioinformatics | BIOL7200

Week 8 Exercise

October 11, 2022

Starting this week we will be using real bioinformatics data and writing (simpler versions of) real world bioinformatics scripts.

For this week, assume that the user gives you correct inputs all the time. Your script will be graded on the output produced and not how all the errors are handled.

Again, please do not use any modules other than `sys`.

Instructions for submission

- This assignment is due Wednesday, October 19 2022 at 11:59pm. Late submissions will receive a 0.
- The *k*-mer counter script must be named: kmer_counter.py
- The vertical column script should be named: vertical_colum.py
- The three way file join script must be named: three_way_join.py
- All scripts should output their tab-separated results to STDOUT
- Please submit these three files on Canvas.

1. K-mer counter in Python

Write a script that reads in a FASTA file and a value of *k* and calculates the number of times each k-mer is observed within the genome. A k-mer is a sequence of length *k*; for example, k-mers of length 2 (*k*=2) for DNA are AA, AT, AG, AC, CC, CT, CG, CA, TT, TA, TG, TC, GG, GC, GT, and GA

You should only report k-mers with non-zero occurrences. The output should be printed on the standard output in two, tab-separated columns. The first column should contain the k-mer sequence and the second column should be the number of times it occurs within the input sequence. Do not print any extra lines. The k-mers should be printed alphabetically (i.e., sorted based on their sequence and not on their occurrence).

For the test dataset, you are given a FASTA file (NC_000913.fasta). This is the genome for *E. coli* K-12 substr. We are also providing you with an output file for this genome later in the week. If your script can reproduce this output file correctly, it should work fine on other datasets too.

A k-mer is a sequence of length *k*. If you are given a sequence AGCTTTTCA and asked to find all possible *k*-mers with *k*=5, the solution would be:

```
AGCTT 1
CTTTT 1
GCTTT 1
TTTCA 1
TTTTC 1
```

Your script should take two positional arguments (k-mer size and FASTA file), do NOT use `getopts` or any other modules, and be named `kmer_counter.py`

```
kmer_counter.py <k> <input FASTA>
```

2. Read a file vertically

On Canvas, you will find a file: knownGene.txt. You can read the first (second, third and so on..) row of this file using a normal for loop. But can you read just the first column? What about the nth column. This can be done easily using: "*cut -f <n> file_name*" in bash. But can you do this in python by reading the file?

Your task for this question is to write a script that takes in one argument k: a column number, and print just that column on stdout (your terminal screen).

For k=1, print column 1, for k=2, print column 2. If the column number does not exist in the file, then tell the user that "k" value is exceeding the file size. There is no column 0, so throw the same error for k=0.

```
read_vertical.py <k> <knownGene.txt>
```

Want to learn more? Look what List Comprehension is in Python. See if you can use that in this question. Using list comprehension is not necessary for this question.

3. Three-way file join

You are given three files:

a) knownGene.txt

b) kgXref.txt

c) InfectiousDisease-GeneSets.txt

The first two files have been downloaded from ftp://hgdownload.cse.ucsc.edu/goldenPath/hg19/database and are described in the sql files (**knownGene.sql** and **kgXref.sql**) located in the same ftp location. The third file is the result of manual curation by one of your collaborators.

While the full description of the first two files can be found in the above-noted sql files, here is the information you need to answer this question:

a) The **knownGene.txt** file is a tab-separated file that has multiple columns, but you are only interested in columns 1 (UCSC id), 2 (chromosome), 4 (transcription start position) and 5 (transcription stop position).

b) The **kgXref.txt** file is also tab-separated, and the columns we are interested in are 1 (UCSC id) and 5 (gene name). Entries with missing information are represented as blanks within this file. Try pasting the file in Excel to see how it is formatted.

Your task is to find the genomic coordinates for the genes listed in the **InfectiousDisease-GeneSets.txt file**. The output should be printed on the standard output in four, tab-separated columns and will look like this (tab-separated fields)

```
Gene    Chr     Start    Stop
ACTB    chr7    5566778      5570232
ACTG1   chr17   79476996     79479892
ADCY3   chr2    25042038     25142055
ADCY9   chr16   4012649      4166186
```

The output should be sorted alphabetically by gene name.

Your script should take three positional arguments, do NOT use **getopts**, and be named **three_way_join.py**

```
three_way_join.py  knownGene.txt  kgXref.txt  InfectiousDisease-GeneSets.txt
```

Some of the assumptions you will have to make:

1) UCSC id is the unique identifier for **knownGene.txt** and acts as a connector between **knownGene.txt** and **kgXref.txt**

2) A gene can have multiple transcripts listed in **kgXref.txt** and hence multiple UCSC ids associated with it. If this happens, pick the FIRST set of coordinates for the gene. **This is a simplifying assumption, and this is what we will be using for testing your code.**

3) Genes can be absent from the kgXref table; this is ok. The inconsistency is due to discordance in the update dates of the table and GeneSets file, but there shouldn't be a lot of these cases.

Sample output files for your reference:

1) **q1-3mer.out.txt** – This is a sample output produced from the input file (NC000913.fasta) and a k-mer size of 3
2) **q1-4mer.out.txt** – This is a sample output produced from the input file (NC000913.fasta) and a k-mer size of 4
3) **q3-geneSetCoordinates.txt** – this is the final expected output file for provided input file.