UPRM CSE Department

CIIC 3011 - Introduction to Computer Programming I

Fall 2018-2019

Project 2: DNA Consensus Sequence

A. BACKGROUND

Our genetic blueprint is encoded in our DNA (Deoxyribonucleic Acid). DNA is made up of nucleotides, which include (among other things) one of four types of nitrogen bases**: adenine (A), thymine (T), guanine (G) and cytosine (C).**

A DNA sequence is represented by these nitrogen bases, for example: AATCCGCTG. The order of these bases in our DNA determines our individual genetic code. If you're interested, you can watch a 5-minute animated video on What is DNA and How Does it Work?

Sometimes to process DNA data files a *consensus sequence* is created when there are many similar DNA sequences in a file. A consensus sequence is obtained by choosing the most highly-occurring nucleotide in each position. Consider the following example:

Sequence 1:	GATCAGCTAG
Sequence 2:	AATCCGATCG
Sequence 3:	AATGCGCTAG
Sequence 4:	ACTCTGCGTG
Consensus:	**AATCCGCTAG**

⇔

The nucleotides in the first column are G, A, A, A, A. Since A is the highest occurring nucleotide in position 1 (from left to right), that's the nucleotide that will be used in position 1 of the consensus sequence. In general, the i^{th} character of the consensus sequence is the highest occurring nucleotide of the i^{th} column of the sequences.

Your task is to create a Python program that reads DNA sequences from an input file (below is a file example illustrating how the data is organized) and generates the consensus sequence. Additionally, an output file will be created that stores the nucleotide frequencies for each position, so as to help determine whether the consensus is indeed an accurate representation of the sequences.

B. DESCRIPTION:

Basic Algorithm: Here is the basic algorithm:

```
Load data from the input file into appropriate data structures
Count the nucleotide frequencies for all positions in all positions
Determine the consensus
Process results
     Print consensus string to the screen
     Write consensus string to output file
     Write to output file frequencies of each nucleotide for each column
```

C. TECHNICAL REQUIREMENTS AND DATA DESCRIPTION:

Input file: DNA strings to be processed are to be read from a file named DNAInput.txt. The files have the following format: Description line, sequence line, description line, sequence line, and so on. Here's a sample file:

```
>biological_description_1
GATCAGCTAG
>biological_description_2
AATCCGATCG
>biological_description_3
AATGCGCTAG
>biological_description_4
```

```
ACTCTGCGTG
```
… and so on …

Description lines always start with the ">" character; you may disregard these lines.

Note that a FASTA file is a plain text file, except that the file extension is either `.fa` or `.fasta`. (i.e. read the file in the same way you would read a .txt file)

Output file: You will store the consensus sequence and the frequencies of the nucleotides in a file called DNAOutput.txt. For the sample input file provided above, here's what the output file would contain:

```
Consensus: AATCCGCTAG
Pos 1:     A:3   G:1   C:0   T:0
Pos 2:     A:3   C:1   G:0   T:0
Pos 3:     T:4   A:0   C:0   G:0
Pos 4:     C:3   G:1   A:0   T:0
Pos 5:     C:2   A:1   T:1   G:0
Pos 6:     G:4   A:0   C:0   T:0
Pos 7:     C:3   A:1   G:0   T:0
Pos 8:     T:3   G:1   A:0   C:0
Pos 9:     A:2   C:1   T:1   G:0
Pos 10:    G:4   A:0   C:0   T:0
```

Note that the nucleotide sequences listed for each column are in non-increasing order by frequency. In case of a tie (when 2 different nucleotides have the same frequency) it doesn't matter which one comes first in the output. For example, the last line in the previous example could have also been:

```
Col 10:    G:4   C:0   T:0   A:0
```

You may assume that:

- *Every combination of description+sequence takes up 2 lines (1 line for each).*
- *All sequences in the file have the same length. The exact length is not initially known; you may determine it from any of the sequences.*
- *All nucleotides are in capital letters.*
- *There will be no characters other than A, C, T, and G in the sequences.*
- *There will be no ties for the most highly-occurring nucleotide in any column. This means that, when determining the consensus, there will be a single nucleotide that is the highest occuring.*

Your code:

1. Create a function called `load_data`.
    a. It takes as argument the name of the file to be used (a string).
    b. It returns a data structure (or more than one) that contains all of the information from the input file.
2. Create a function called `count_nucl_freq`.
    a. It takes as argument the data structure(s) generated by `load_data`.
    b. It returns a new data structure (or more than one) that contains the frequencies of the nucleotides for each column in each sequence.
3. Create a function called `find_consensus`.
    a. It takes as argument the data structure(s) generated by `count_nucl_freq`.

   b. It returns a string; the consensus sequence.
4. Create a function named `process_results`.
   a. It takes as arguments the data structure(s) created by `count_nucl_freq` and the name of the output file (a string).
   b. It writes the results, in the format previously described, to the output file.
   c. It doesn't return anything.

Other Important information

1. Sample files are provided, but they are for testing purposes only. In other words, the sample `DNAOutput.txt` provided should be the result of executing your program with the sample file provided (`DNAInput.fasta`). Your program should be able to work with any FASTA file where all sequences are of the same length.
2. You should **NOT** prompt for the file name; you should **ALWAYS** *try* to open a file named `DNAInput.fasta` and your output should ALWAYS be to a file named `DNAOutput.txt`.
3. Your code should not look specifically for the letters A, C, T or G. It should just use whatever characters it finds in the strings.

D. SOURCE CODE

The starter file, named `p2_fall2018.py`, can be found in the eCourses entry for the project. Modify the `p2_fall2018.py` file and add all the necessary code to complete the project. However, keep in mind that the file must be renamed prior to submission in order to meet the naming convention specified below in Submission Details. *You **must** use this starter file as a base for your project.*

E. SUBMISSION DETAILS

You must submit the .py file for your project through the assignment entry in eCourses in the "Project 2" section. Your file name must have the following format: LastnameFirstname_Section_P2. For example, a student named Juan López from section 016 would submit the following file: LopezJuan_016_P2.py (please use only one name and one last name).

YOU MUST NOT SHARE/COPY CODE WITH/FROM ANOTHER PERSON OR ONLINE SOURCE. YOU CANNOT USE ADVANCED STRATEGIES THAT HAVE NOT BEEN COVERED IN CLASS.

This project must be completed and turned in by 11:59 on Tuesday, November 13, 2018.