GNBF5010 Homework 2

Deadline: Sunday 17 October 2021

Remarks:

1. Write a single python script for each question. Use filenames `q1-xxxxx.py`, `q2-xxxxx.py` and so forth, where xxxxx is the last five digits of your student ID.
2. Zip all your files for this homework in a single file and upload to the Blackboard before the deadline. Use a `readme.txt` file if you have any additional notes to the grader. *Late submission will not be accepted.*

3. It's suggested to use the provided template in Homework 1 (`script-template.py`) for coding.
4. Like homework 1, your coding will be graded based on program design, program execution, specification, coding style and comments.

5. Add brief comments to explain the reasoning behind important or complex statements.
6. Have a second thought if the design of your program gets complicated.
7. Test your program with various test cases to ensure that it works properly.

Questions

1. **Unknown Letters**
   The input file `seqs.txt` contains 516 sequences, with each sequence in a single line. Write a program to list which letters in the file `seqs.txt` are not A, T, C, or G. It should only list each letter once. *Hint: Start with an empty list for unknown letters. Then use two loops to scan letters in each sequences.*

2. **Sequence Properties**
   Write a program, **1)** read all sequences in `seqs.txt` and store them into a list called `seqs`, **2)** prompt the user a menu for selection of various properties of the seuqences, and **3)** show the corresponding results based on user's choice.

   The selection menu should include the following items. Please use **separate functions** in your program for the processes in options 1 to 4.

   1) Number of sequences in the input file
   2) Number of occurrences of a specific sequence, e.g. GGATC. The program will then prompt another message for the input of the target sequence.
   3) Number of sequences with length ≥ a particular length, e.g. 500. The program will then ask for the minimum length.

4) Number of sequences with GC content ≥ a given value, e.g. 50%. The GC content could be calculated as `(num_of_G + num_of_C) / seq_total_len`. You can use web tools like https://jamiemcgowan.ie/bioinf/gc.html to check your calculation.
5) The combination of choices 3 and 4: Number of sequences with length ≥ a particular length and with GC content ≥ a particular value.

The output of your program should look like this:

```
============================
Please select the sequences property that you want to display, or press 0 to
exit the program.
1) Total number of sequences
2) Number of pattern occurrences
3) Number of sequences with length >= min_len
4) Number of sequences with GC% >= min_GC
5) Number of sequences with length >= min_len and GC% >= min_GC

Enter the choice: 4
min_GC (e.g. 50%) = 50
Num of seqs with gc >= min_gc is 255.

============================
Please select the sequences property that you want to display, or press 0 to
exit the program.
1) Total number of sequences
2) Number of pattern occurrences
3) Number of sequences with length >= min_len
4) Number of sequences with GC% >= min_GC
5) Number of sequences with length >= min_len and GC% >= min_GC

Enter the choice: 5
min_len = 600
min_GC (e.g. 50%) = 60
Number of seqs with len >= min_len and gc >= min_gc is 11.

============================
Please select the sequences property that you want to display, or press 0 to
exit the program.
1) Total number of sequences
2) Number of pattern occurrences
3) Number of sequences with length >= min_len
4) Number of sequences with GC% >= min_GC
5) Number of sequences with length >= min_len and GC% >= min_GC

Enter the choice: 0
Exiting ...
```

3. Average of tests

Suppose `test1.txt`, `test2.txt` and `test3.txt` are results from three different experiments. The gene set used in each experiment is slightly different. Each row in the data file represents one gene, with the following format:

```
Accession_Number        Number        Number        …
```

1) First, load the data from all files to **a single dictionary**, with the accession number as the key and the reference to a list of data from all experiments as the value.
2) Then calculate the average for each gene and store them in another dictionary.
3) At the end, write this new dictionary to an output file called `test_averages.txt`, where the first column is the accession number of gene and the second column is the corresponding average value.

4. Molecular Weight

Write a program with the following steps.

1) Make a python dictionary of one-letter amino acids codes (the keys) to their molecular weight (the values), for all 22 amino acids. The molecular weight of 22 amino acids can be found in provided table. For example, the molecular weight of C (Cysteine) is 121.

2) Print out a list of all the amino acids sorted by their molecular weights from the heaviest to the lightest. *Hint: Just sort the items in Question (1) dictionary based on values. The output will look like:*

```
a. AA  MW
b. W   204Da
c. Y   181Da
d. R   174Da
e. F   165Da
f. …   …
```

3) Read the protein sequence from `lysozyme.fasta` and calculate the molecular weight of this protein using the dictionary created in question (1).

Amino Acid Abbreviations and Molecular Weights.

Amino Acid	Three-Letter Abbreviation	One-Letter Symbol	Molecular Weight
Alanine	Ala	A	89Da
Arginine	Arg	R	174Da
Asparagine	Asn	N	132Da
Aspartic acid	Asp	D	133Da
Asparagine or aspartic acid	Asx	B	133Da
Cysteine	Cys	C	121Da
Glutamine	Gln	Q	146Da
Glutamic acid	Glu	E	147Da
Glutamine or glutamic acid	Glx	Z	147Da
Glycine	Gly	G	75Da
Histidine	His	H	155Da
Isoleucine	Ile	I	131Da
Leucine	Leu	L	131Da
Lysine	Lys	K	146Da
Methionine	Met	M	149Da
Phenylalanine	Phe	F	165Da
Proline	Pro	P	115Da
Serine	Ser	S	105Da
Threonine	Thr	T	119Da
Tryptophan	Trp	W	204Da
Tyrosine	Tyr	Y	181Da
Valine	Val	V	117Da