BIFS 617: Advanced Bioinformatics
Dr. Alkharouf
Final Exam

Deliverable and Instructions (please READ!):

Submit your solutions to the questions below in ONE Python script file, name it using the following format please: *YourLastName*_FinalExam.py. Add comments to indicate which question each code snippet belongs to.

Note:

1) This is an individual exam, you may not seek help from anyone, or offer help to anyone!
2) Make sure your code will run on Python 3, if it does not run it won't be graded! I will use IDLE to run your code.
3) Your code should be general enough to work for any data, not just the data\files that are given to you in the questions below.

1. (10 pts) Write a program that given a DNA string, prints out the 20 characters upstream of the start codon ATG. That is, given:

dna = "CCCCATAGAGATAGAGATAGAGAACCCCGCGCGCTCGCATGGGG"

print out:

The 20 bases upstream of ATG are AGAGAACCCCGCGCGCTCGC

Write a function that accepts the DNA sequence as input, and uses a regular expression to match the desired substring and prints out the result.

2. (10 pts) Write a function that reads in a file containing two strings on each line (called Q2_input.txt, attached), and creates a dictionary with the first string as key and second string as value. Call the function and print out the dictionary pairs to the screen.

3. (15 pts) Parsing Blast results:

Attached is a file called BlastResults.txt, in it are 3 blastx searches done on three sequences. This is the format you usually get when you do a local blast search (via the stand alone blast program, not the online web version). Your job is to summarize the results to your boss (who is a biologist and has no time to scroll down and read a blast report line by line!) by showing him\her a nice table as shown here:

Query	BestHit	E-value	Identities
Contig_383	>ref|NP_758766.1| hypothetical protein pEA28_03 [Erwinia amylovora] gb|AAM94884.1| hypothetical protein [Erwinia amylovora]	1e-029	65/74(87%)
Contig_391	>gb|ABI35978.1| LacZ alpha peptide [Cloning vector pYESW29']	3e-018	49/74(66%)
Contig_114	>gb|EDR02246.1| vesicle budding-related protein [Laccaria bicolor S238N-H82]	2e-044	105/221(47%)

So you need to open the file, and use regular expressions to parse out the query, BestHit (which is the hit with the smallest E-value, listed always as the 1st one), the E-value and the Identities. Look at the table above and locate those elements in the BlastResults file so you know what you are looking for.

Parse those out and print them to a file (call it out.txt), use tab to separate them. You would want to print each blast result into a line. No need to draw borders or anything like that! Just print the Query \tab BestHit \tab E-value \tab Identities \tab and then end with \n and start with the next blast result. Use at least 1 function.

4. (15 pts) Write a program that parses out the comment lines in a FASTA file of sequences. The file MID2_454AllContigs.fna is a text file that contains FASTA sequences. Your job is to parse out the comment lines and produce a file like the MID2_out.txt that is also attached.

The comment lines look like this:

>MID2_contig00001 length=578 numreads=4

And they have to parsed out like this:

MID2 contig00001 578 4

Use a regular expression to get the job done and at least one function.

5. (20 pts) Read in a whole genome (in FASTA format – see attached) and compute the background codon frequencies. The background frequency of a codon is computed by the formula:

background_frq(codon) = 100 * N(codon)/ Total_codons

where N(codon) is the number of occurrence of the codon across the entire genome, and Total_codons is the total number of all codons in the whole genome. Print out the background frequency of each codon, from AAA to TTT. Use a dictionary in your solution.

NOTE: To simplify the problem, just count codons that appear in reading frame 1. That should give approximately the same frequencies as in all six reading frames.

6. (30 pts) Write a program that reads in a file with FASTA sequences and outputs the restriction enzymes that will cut each sequence, and their cut positions within the sequence. The file (SeqLibrary.txt) is attached and contains a number of sequences. For every sequence print out a table that displays the enzyme names and their cut sites.

Use the entire REBASE restriction site data from the *Staden* file (found at http://rebase.neb.com/rebase/link_staden -- Also attached, called staden_link.txt), use a **dictionary** to store all the data.

So basically what you'll have to do is modify the code you did in the discussions to read in the enzymes and their cut sites in this new format. Once you have done that, you'll need to write a function that takes as an argument a DNA sequence and the restriction enzyme dictionary. That function will then find all the cut sites **(for all the enzymes in the Staden file)** for a given sequence in the file. Only print out the enzymes that will cut the sequence.

Print out the table to a file, it should look something like this (just for illustrative purposes, not the actual results you will get):

Seq 1 Results

Enzyme Name	Cut positions
EcoR1	30, 45, 65
ARGII	12, 18, 89

Seq 2 …. Etc.