Biological Command Line Interpreter

Project Instructions:

1. Project should be done in teams of 4 to 5 students from the same groups of each TA.
    - B1 and B2 can form groups together → Dina Amr
    - B3 and B4 can form groups together → Amany Adel
    - B5 can form groups together → Alaa Adel
2. Submissions will be through google forms. Please use the below link:
   https://forms.gle/i9y92FDKkUswBotPA
3. Deadline of submitting the project is Saturday 24-December-2022 at 11P.M.
4. One student from each team should submit one file named: '"ID1-ID2-ID3-ID4-ID5_Group_TAName". This file is your code with extension .py.
    - Please don't use zip files or any other file format (only the code).
5. Copied code or cheating cases will take -3.

Project Description:

You shall use what you have learned in this course to develop a Biological Command Line Interpreter Simulator in python. It will be similar to the cmd in windows but will take commands that apply some biological operations on biological data. You can use any built-in libraries explained in the lectures.

Input format:

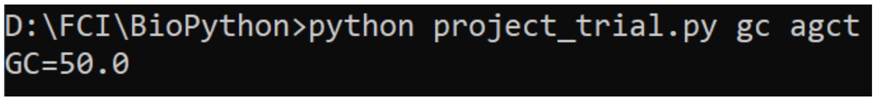
The input to your project will be a command to execute along with any parameters the command demands (this will be more clarified in the following section). You MUST use the getopt functionality that was explained in lectures to parse the input because you will run your python file as follows:

```
python myfile.py    command_name parameter_1 parameter_2 –o option_1 –x option_2
```

You can see that the command is divided into two sections the first one is your program itself as an argument to python to be parsed and interpreted. The second part is the command your program shall consume and parse to be executed.

The parsing step should be done using getopt functionality to identify the command to be execute, get each argument and each option to be execute the command.

Your program should call the correct function in your code, that correspond to the given command, and pass it the given parameters needed to execute this command. E.g. calc_gc(args[1])



```
D:\FCI\BioPython>python project_trial.py gc agct
GC=50.0
```

For the above example, the command consists of two parts: the first one is “python project_trial.py” is the mandatory step for running your program. The second part is “gc agat”, that is the command “gc” and its argument seq “agat” that your program will parse. Your program should recognize which function to call corresponding to the “gc” command and pass it the sequence “agat” and the result of this function is your output which is in this case is the gc percentage of the given sequence.

Note: you should handle any errors that may occur like “wrong command name” or “missing parameters”.
The required commands details are illustrated in the following section.

Commands Description:

gc	
Usage	gc seq
Parameters	seq: a string represents the sequence
Description	This command takes a seq as a string and returns the gc percentage of it.
Sample run	gc AGCAT

transcribe	
Usage	transcribe seq
Options and arguments	seq: a string represents the sequence
Description	This command takes a seq as a string and returns its transcription.
Sample run	transcribe AGCAT

reverse_complement	
Usage	reverse_complement seq
Options and arguments	seq: a string represents the sequence
Description	This command takes a seq as a string and returns its reverse complement.
Sample run	reverse_complement AGCAT

calc_nbases	
Usage	calc_nbases seq
Options and arguments	seq: a string represents the sequence

Description	This command takes a seq and calculates its nbases
Sample run	calc_nbases NGCTN

is_valid	
Usage	is_valid seq type
Options and arguments	seq: a string represents the sequence type: a string that represents the type of the sequence. It can be one of these keywords [protein, dna, rna]
Description	This command takes a seq and a type (protein, dna, rna) and returns a Boolean value of whether it's a valid type or not
Sample run	is_valid NGCTN protein

filter_nbases	
Usage	filter_nbases seq
Options and arguments	seq: a string represents the sequence
Description	This command takes a seq and returns the Seq after removing n bases
Sample run	filter_nbases NGCTN

seq_alignment	
Usage	seq_alignment seq1 seq2 [-o file]
Options and arguments	seq1, seq2 : strings represents the sequence -o file : specifies the path of the output file
Description	This command takes 2 sequences as input and get all their alignments along with the score. The -o is an optional parameter if we need the output to be written on a file instead of the screen.
Sample run	seq_alignment AGCCT AGC -o output.txt

seq_alignment_files	
Usage	seq_alignment_files file1 file2 [-o file3]
Options and arguments	file1, file2: specifies the paths of the files to be aligned that contain the sequences -o file3 : specifies the path of the output file
Description	This command takes 2 fasta files as input, each file contains a single sequence. It reads the 2 sequences from files and get all their alignments along with the score. The -o is an optional parameter if we need the output to be written on a file instead of the screen.
Sample run	seq_alignment s1.fasta s1.fasta -o output.txt

online_alignment	
Usage	online_align seq [-o file]

Options and arguments	seq : a string represents the sequence -o file : specifies the path of the output file
Description	This command takes a sequence and uses BLAST to search the internet for its alignments. The output should be all the information in the resultant BLAST record. The -o is an optional parameter if we need the output to be written on a file instead of the screen.
Sample run	seq_alignment ACTGCCGTCAAGTCAG -o output.txt

merge_fasta	
Usage	merge_fasta file1 file2 [file3 …] [-o output_file]
Options and arguments	file1, file2, etc. : specifies the paths of the files to be merged -o output_file : specifies the path of the output file
Description	This command takes any number of fasta files (at least two) and merge their contents into one fasta output file. There is an option to write the merge result in a file using -o option, otherwise the merge result will be displayed on the console. **Hint:** use variadic parameters in functions to handle the unknown number of parameters**. Don't** use lists.
Sample run	merge_fasta f1.fasta f2.fasta f3.fasta f4.fasta

convert_to_fasta	
Usage	convert_to_fasta file
Options and arguments	file: specifies the path of a genbank file
Description	This command converts the input genbank file with multiple records onto a fasta formatted file. The output is to be written in a different output fasta file.
Sample run	convert_to_fasta “ls-orchid.gbk”

Grading Schema:

Using Getopt	1
Error Handling	0.5
Gc	0.5
Transcribe	0.5
Reverse_complement	0.5
Calc_n_bases	0.5
Is_valid	0.5
Filter_n_bases	0.5
Seq_alignment S1 S2 [-o output_file]	1.5
seq_alignment file1 file2 [-o file name]	1.5
online_align s1 [-o file name]	1.5
merge file1 file2 ... (variadic parameters)	1.5
convert_to_fasta genbank_file	1.5
Total	**12**