THE UNIVERSITY *of* EDINBURGH
School of Biological Sciences

Assignment 2

Bioinformatics Processing and Systems Management

Student Exam Number: B151359-2019

The file name of your uploaded document **must** include your exam number

Taxonomic Protein Analysis.py

User Manual

Github:

Link: https://github.com/B151359-2019/Assignment2.git

Password: notsecure

Program Overview:

Taxonomic Protein Analysis.py will take a set of protein sequences associated with a protein and a subset of a taxonomic tree to produce a conservation analysis between species, to find motifs associated with a sequence, and to produce a phylogenetic tree based on the dataset. The program uses the NCBI database, clustal omega, blastp, and a variety of emboss tools such as plotcon, patmatmotifs, and cons to produce its outputs. In this user manual we will walk through how to use the program step by step from start to finish:

1. Start the program in your terminal like this: './runme.py' You should see this screen:

```
Welcome to Taxonomic Protein Analysis.py!
Type exit,quit, or q at any time to quit the program.

1.[X] Skip Redundant Sequences
2.[X] Determine/Plot Protein Conservation
3.[X] Scan Sequences for Motifs
4.[ ] Phylogenetic Tree

Everything marked with X will be run.
type numbers 1 through 4 to select/unselect a choice.
5 will select all choices and 6 will deselect all choices.
If nothing is selected, the program will download your query fasta file, generate a sorted list of your most similar sequences, and produce a protein alignment in the outputs folder.
If you would like to tweak some more intricate features, type 'advanced' to access the advanced settings.
 If you're happy with your choices, just hit enter to move on!
Value:
```

   a. At any point in the program where it asks for an input, you may type 'q', 'exit', or 'quit' to exit the program (advanced settings excluded).
   b. In this welcome menu, you may enter:
      i. 1,2,3,4 to select/unselect a process you wish to run [X] means it will run [ ] means it will not
      ii. 5 to select all processes to run
      iii. 6 to unselect all processes to run
   c. The default selections for the process are 1. Skip Redundant Sequences, 2. Determine/Plot Protein Conservation, and 3. Scan Sequences for Motifs. 4. Phylogenetic Tree is turned off due to how long it takes to process, but this can be tweaked in the advanced menu by increasing the bootstrap value to a much higher number with a hit in accuracy of your tree.
   d. Only in this welcome menu may you access the advanced settings. To do so, simply type 'advanced' and hit enter. You will be greeted with this:

```
Advanced Menu:
1.Maximum Accessions to Download (2-10000): 10000
2.Maximum Accessions to Process (0-250): 250
3.Redundancy Match Threshold (0-100): 100
4.Phylogenetic Tree Boostrap value (min: 1000): 1000
Make a selection between 1 and 4. Or, type exit to leave the advanced menu:
```

   e. Here your available inputs are:
      i. 1 to change the limit of how many accessions you can download with a range of 2 to 10,000.
      ii. Although the program will download a maximum of 10,000 accessions, it will only process the top set of a predefined number of accessions for optimization reasons, this is default set to 250, but you may adjust it to any number between 2 and 250.
      iii. 3 to adjust the similarity level of the sequences you wish to exclude from your analysis within a species. A minimum value of 50 to 100 is accepted. 100 is equivalent to 100%, so any sequence that matches another one within a species 100% will be removed.
      iv. 4 to adjust your bootstrap value (and also the alrt value) for the phylogenetic tree making process. The minimum recommended value is 1000 but feel free to go as high as you wish, but just know that the higher you go the longer the process will take, and the phylogenetic tree making process is quite long with a database larger than 100 accessions.

v. ‘Exit’ to go back to the welcome menu. This is the only input area where you cannot exit the program.

2. If you are happy with your choices, leave the input empty and hit enter. If you do not leave the input empty then you will not be able to move onto the next screen. You should be greeted with another area asking you to input your query for taxon and protein:

```
Taxonomic Group: aves
Protein family: glucose-6-phosphatase
Your inputs were:
        1. Taxon: aves
        2. Protein: glucose-6-phosphatase

To change an input, enter 1 for taxon, 2 for protein, or 3 for both. otherwise just hit enter to move on.
Value:
```

   a. Simply type your choices into the input areas displayed
      i. The program will ask for Taxonomic Group first. Make sure not to enter:
         1. Taxon id with [uid] or txid in the name, the program will handle that for you, just the numbers will work fine!
         2. A blank entry. The program will not let you continue without having typed anything in
         3. Exit, unless you wish to exit the program, because it will exit the program.
      ii. Next you will be prompted to enter your protein family. The requirements are more lax but just keep in mind not to enter:
         1. An blank entry
         2. Exit unless you wish to exit the program
      iii. NOTE: If your taxon query did not produce one taxon in its search results, you will be prompted to select one of the following displayed taxons to narrow down your query.

```
Taxonomic Group: fish
Protein family: glucose-6-phosphatase
Your inputs were:
        1. Taxon: fish
        2. Protein: glucose-6-phosphatase

To change an input, enter 1 for taxon, 2 for protein, or 3 for both. otherwise just hit enter to move on.
Value:
Taxon query too vague. Narrow down by selecting a number, or enter 0 to choose another taxon or protein:
1. Coelacanthimorpha: (lobe-finned fishes), vertebrates
2. Hyperoartia: (fishes), vertebrates
3. Hyperotreti: (fishes), vertebrates
4. Actinopterygii: (ray-finned fishes), superclass, bony fishes
5. Dipnoi: (lungfishes), lungfishes
6. Chondrichthyes: (cartilaginous fishes), class, cartilaginous fishes
Choice :4
Your inputs were:
        1. Taxon: Actinopterygii
        2. Protein: glucose-6-phosphatase
```

         1. In this menu, your choices are a number value between 1 and the last available taxon choice, in the above example with fish, its 6. You may also choose 0 to return to the previous menu to change your taxon completely if you wish.

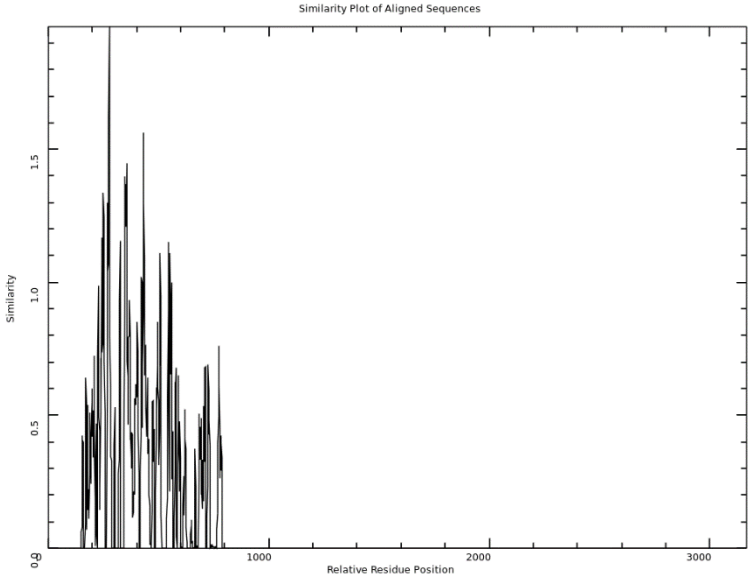
   b. You will then be shown your choices. At this point you have 4 choices:
      i. Type 1 to change your taxon entry
      ii. Type 2 to change your protein entry
      iii. Type 3 to change both
      iv. Type exit, q, or quit to leave the program.

3. If you are happy with your choices, the program will generate a summary count of the amount of accessions you will have to process:

```
your search produced 461 hits. Enter to process:
Your top 5 species in terms of protein sequence hits:
1:      Pipra filicauda         hits:    17
2:      Numida meleagris        hits:    17
3:      Lepidothrix coronata    hits:    17
4:      Columba livia   hits:    16
5:      Neopelma chrysocephalum         hits:    14
Your data shows that there are 461 sequences between 80 species. Enter to continue:
Your top 5 species in terms of protein sequence hits:
1:      Erythrura gouldiae      hits:    9
2:      Gallus gallus   hits:    7
3:      Meleagris gallopavo     hits:    7
4:      Lepidothrix coronata    hits:    7
5:      Falco cherrug   hits:    6
After filtering for redundancy, your new dataset has 309 proteins among 80 species. Enter to continue:
```

a. If you are happy with your number of hits, simply hit enter when it says 'Enter to process.' However, note that if your search produced 1 or fewer results, you will be prompted to change one or both of your search inputs.
b. If you did not have 1 or fewer results and are happy with your results, hitting enter will prompt the program to download the fasta file which will be used to generate the alignment file and in turn all other proceeding functions of the program, and a genbank file which will give you a more specific information about your data. You will be shown your top 5 species with respect to how many protein sequences are associated with that species.
c. If you are happy with that information, simply hit enter to move onto the next step. Otherwise just type anything else to return to the protein and taxon choosing menu.
d. If you chose to remove redundant protein sequences, here you will see an updated list of your top hits as well as an updated summary of total sequences and species. If you are happy with these results, hit enter, if you are not, type anything else to return to the query choosing menu. This screen is your last chance to make any changes before the program processes any following steps.

4. If you chose to proceed, the program will generate your alignment file using clustal omega, a consensus sequence based on that alignment file using emboss tools cons, and then it will produce a list of accessions sorted by similarity created by turning the downloaded fasta into a database, and running the consensus sequence against the database. These 3 items are always generated regardless of what you chose to run in the welcome screen.
    a. Clustal Omega information can be found here: https://www.ebi.ac.uk/Tools/msa/clustalo/
    b. Emboss cons information can be found here: http://www.bioinformatics.nl/cgi-bin/emboss/help/cons
    c. Blastp information can be found here: https://bio.libretexts.org/Bookshelves/Cell_and_Molecular_Biology/Book%3A_Investigations_in_Molecular_Cell_Biology_(O'Connor)/9%3A_Protein_Conservation/9.6%3A_The_BLASTP_algorithm
5. If you chose to run "Determine/Plot Protein Conservation" in the welcome screen, the program will produce a plot of the areas of conservation among your protein sequences using emboss tools plotcon. The output looks like this: Your plot will be saved as an svg in the outputs folder for your further analysis.



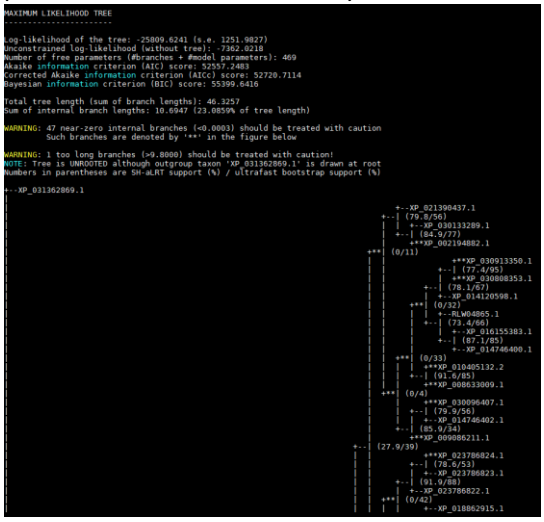
    a. Emboss plotcon information can be found here: http://www.bioinformatics.nl/cgi-bin/emboss/help/plotcon
6. If you chose "Scan Sequences for Motifs" in the welcome screen, the program will proceed to check your alignment file against the PROSITE database for motifs. If the protein sequence produced no hits according to the PROSITE database, then the accession will not be included in the final output file of this step. The

output file will be a compilation of all the motif information for all the accessions which succeeded in producing a hit. This is what your motifs output should look like. This output will be saved in the outputs/motifs file for further reference.

```
########################################
# Program: patmatmotifs
# Rundate: Sun 17 Nov 2019 23:15:59
# Commandline: patmatmotifs
#    [-sequence] ./outputs/motifs/XP_010199970.1.fasta
#    [-outfile] stdout
#    -auto
# Report_format: dbmotif
# Report_file: stdout
########################################

#=======================================
#
# Sequence: XP_010199970.1     from: 1   to: 355
# HitCount: 1
#
# Full: No
# Prune: Yes
# Data_file: /localdisk/software/EMBOSS-6.6.0/share/EMBOSS/data/PROSITE/prosite.lines
#
#=======================================

Length = 17
Start = position 214 of sequence
End = position 230 of sequence

Motif = TNF_1

KTNLFLFMFAFGFYLVLKLLDIDLLWS
     |               |
   214             230


#---------------------------------------
#---------------------------------------
########################################
# Program: patmatmotifs
# Rundate: Sun 17 Nov 2019 23:16:00
# Commandline: patmatmotifs
#    [-sequence] ./outputs/motifs/XP_009903782.1.fasta
#    [-outfile] stdout
#    -auto
# Report_format: dbmotif
# Report_file: stdout
########################################
```

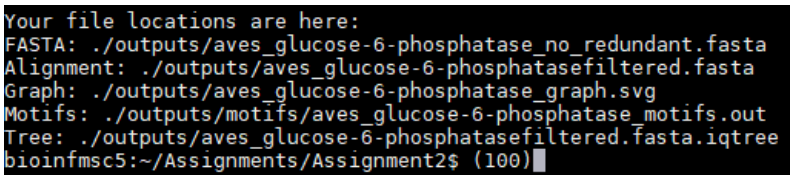
a. PROSITE database information can be found here: https://prosite.expasy.org/prosuser.html#gen_introduction
b. Emboss patmatmotifs information can be found here: http://emboss.sourceforge.net/apps/cvs/emboss/apps/patmatmotifs.html

7. If you chose “Phylogenetic Tree” as an option in the welcome screen, then the program will run iqtree to produce a phylogenetic tree based on maximum likelihood. This process has the potential to take a very long time based on the bootstrapping value that you have chosen and the size of your dataset. The minimum recommended bootstrapping value to use is 1000, but you can change this to anything higher in the advanced settings which you can access in the welcome screen. Each branch of the tree will include a value for Sh-aLRT and UltraFast Boostrap in this format (sh-aLRT,UFBootstrap). The higher your bootstrap value is, the longer it will take to run, because what bootstrap is doing is it is checking every iteration of the potential tree and forming a ratio of how many times the same version of the tree came up during the process. It is recommend to only trust a branch if sh-aLRT >= 80% and UFBootstrap >= 95%.



a. IQ-Tree information can be found here: http://www.iqtree.org/doc/Home

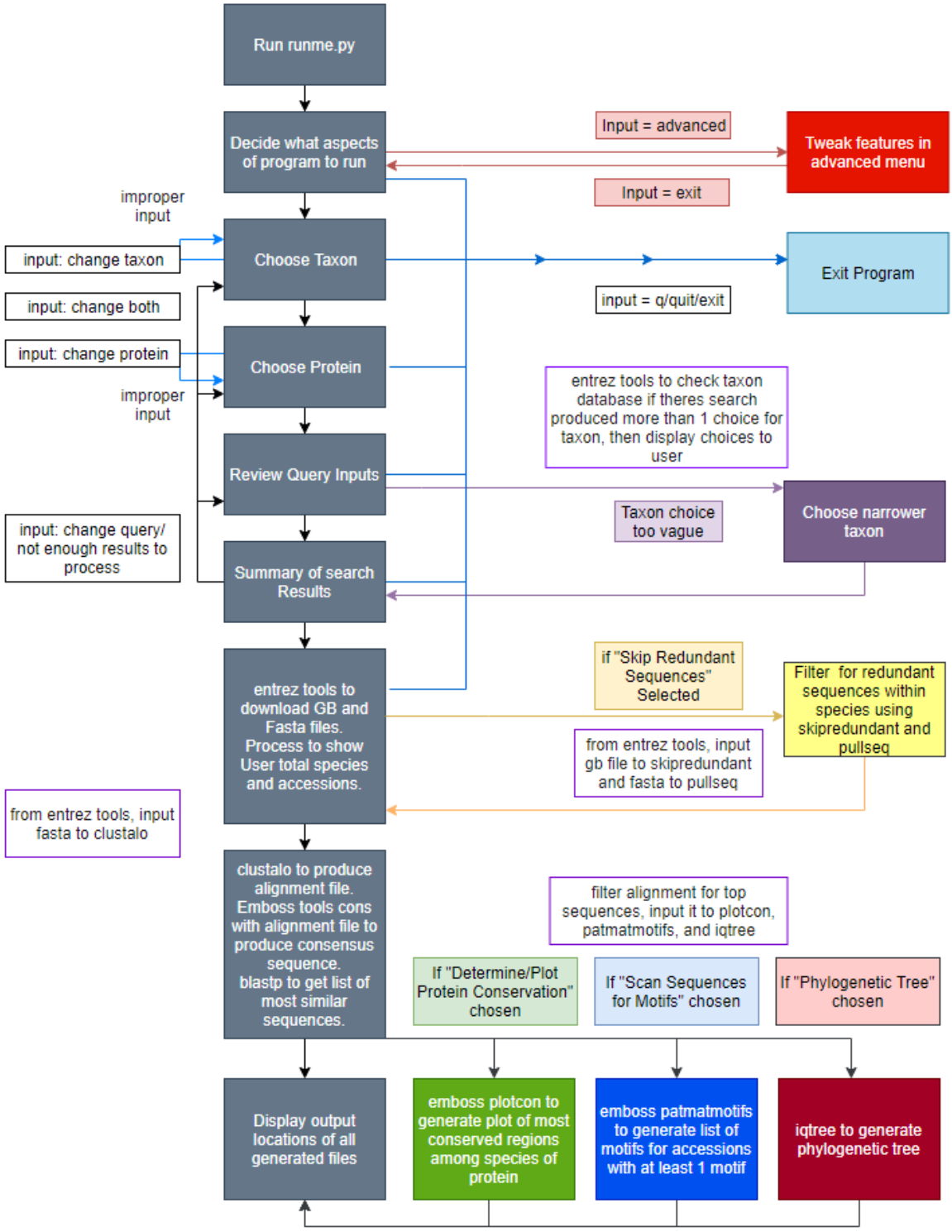
8. Once every function is complete, the program will display the file locations of the outputs of interest, depending on what was run. All files will be generated inside of the outputs folder. You may access these files for further reference and analysis.



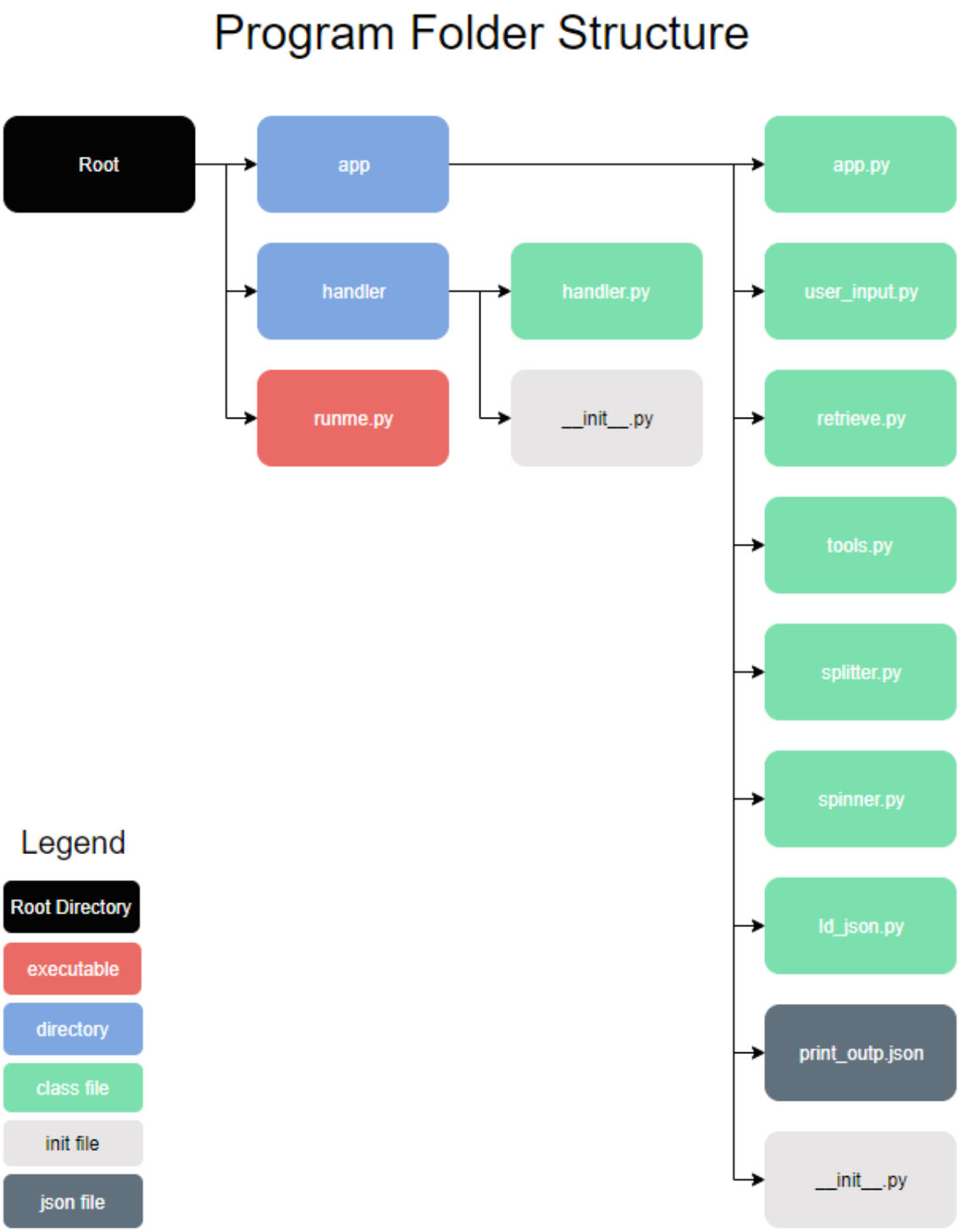
Maintenance Guide

Table of Contents

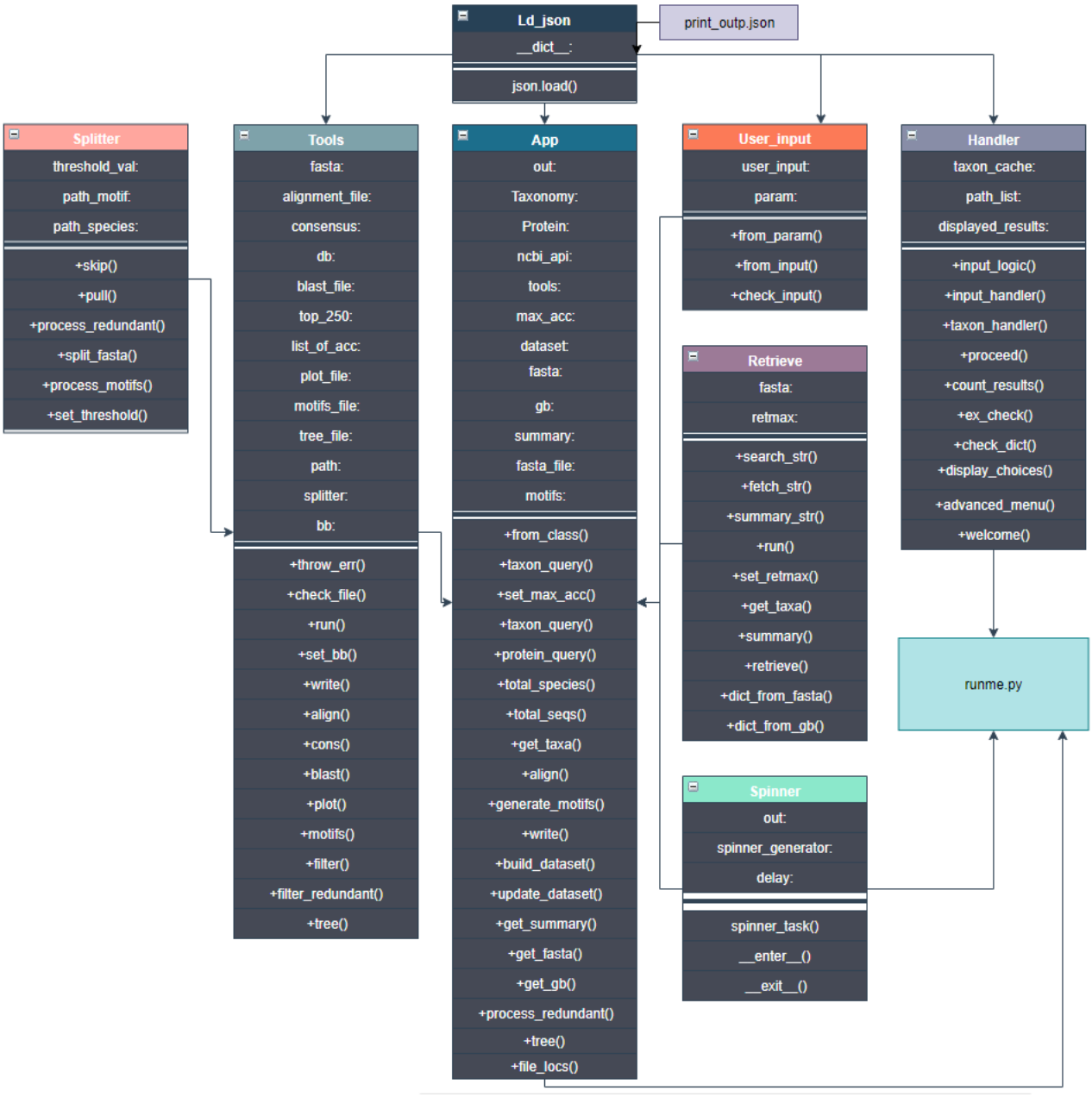
1. Program Flow Chart:



2. Folder Structure:



3. Class Diagram:



4. Class and Function Descriptions
    a. Overview:
        i. The program functionality is built and blueprinted within the class App (which is composed of the Tools, User_input, and Retrieve classes). The direction and user interface of the program is handled by the Handler class. The overall flow of the program is handled functionally in runme.py where App and Handler are instanced and App is passed into Handler. We will start with going over runme.py, then our App class and its components, the Handler class, and finally the auxiliary classes we've built, Spinner and Ld_json.
    b. runme.py
        i. Made up of 2 functions
            1. run_app()
                a. initializes App and Handler, then passes both of them into handle()

2. handle()
    a. takes 2 arguments
        i. obj, in this case app
        ii. handler, in this case handler
    b. Controls the flow of the entire program using methods and Boolean list generated in Handler class by passing in values retrieved from our app class.
    c. Controls pathing of program based on conditionals set by user input in Handler.welcome()
3. Separate from class based approach of the rest of the program as handles its flow by recursively calling itself

c. App.py
    i. Our most abstracted and main class which defines our program's functionality.
        1. Handlers user input, error traps, and improper inputs through the User_input class.
        2. Abstracts downloading data from ncbi and turning it to dictionary in self.dataset through the Retrieve class
        3. Handles writing accessions to files, generating alignment sequences, building consensus sequence, blasting, and filtering for top sequences with self.align() through the Tools class
        4. Allows modularity in building motifs, processing redundant sequences, and building the tree through the Tools class with the help of its subclass Splitter
        5. Can easily print fasta information, genebank information, dataset information, and file pathways directly to screen
    ii. Built from User_input, Retrieve, Tools. Spinner and Ld_json perform some auxiliary functions.
    iii. Maintains the application's state in terms of taxon and protein queries, instancing of subclasses, updating and initializing dictionary {species:[accessions]}
    iv. User input for the app, including input and error handling is handled by the User_input class:
        1. This class is responsible for
            a. Controlling user input of the taxon and protein query inputs
            b. Using regular expressions to block out potential wrong inputs, such as entering taxon id with [uid] or txid (handled in Retrieve class), or empty strings.
            c. Note: assigning taxon and protein can be done directly by instancing the App class such as App (taxon,protein) or by using the setter App.taxon_query = "taxon" / App.protein_query = "protein".
    v. All interfacing with NCBI is done through the Retrieve class, which is instanced in App as ncbi_api:
        1. This class is responsible for:
            a. Abstracting entrez tools to
                i. generate results for summary counts of accessions
                ii. taxon results
                iii. downloading fasta file to be processed for redundancies or in our alignment through clustalo
                iv. controlling our maximum number of accessions through retmax
                v. downloading genebank file to be turned into a dictionary with species as key and a list of accessions as values
            b. Building our dictionary from our genebank file by:

i. Extracting ORGANISM and VERSION fields from genebank using regex
ii. Handling inconsistencies in cases where species and accession lists do not match

vi. Emboss tools, blastp, clustalo, and IQ-Tree are all handled in the Tools class.*
1. This class is responsible for:
a. Keeping track of all file names and file locations/outputs.
b. Placing most of the error traps as many would have to do with files not existing or being placed in the wrong area/deleted.
c. Using subprocess.check_output to run:
i. Clustalo to build alignment sequence file
ii. Cons to build consensus sequence
iii. Makeblastdb to build our blast database
iv. Blastp to generate our blast file in order to get our top most similar sequences
v. Plotcon to generate our plot
vi. Iqtree to generate our phylogenetic tree
2. This class also Abstracts motif, skipredundancy, and pullseq functionality which is all handled by the Splitter class:
a. Filtering of fasta/alignment file is handled through pullseq and iterating through a generated list of accessions built through our dictionary which belongs to App.dataset.
b. Removes sequences that are identical within species with skipredundancies and pullseq
c. Processes motif files iterating through list of accessions and calling patmatmotifs on each one individually
3. *This class is very large and could use a good refactor. It can be split up into a few smaller classes with more focused rolls, something like Emboss.py for emboss tools, Blast.py for blast tools, etc.

d. Handler.py
i. Responsible for all interfacing with the user through controlling logic and flow of inputs
ii. Handles updating of any taxon or protein queries and updates cache of previous taxon choices
iii. Restricts user input to acceptable inputs outside of user_inputs regex handling
1. Mostly restricts user to enter a single digit integer
2. Also handles cases to exit the program
iv. Will narrow taxon choice depending on user input
v. Displays information based on summary and dataset results
vi. Responsible for building menus which user can navigate but restrict user input through while true loop manipulation.

e. Auxiliary classes:
i. Ld_json()
1. Processes print_outp.json, a json file containing all of our print statements, and returns a dictionary which can be accessed within our other classes by importing it.
2. Example: self.out = Ld_json().app will retrieve all print statements associated for the app class.
ii. Spinner()
1. Adds a spinning wheel animation through multithreading so it runs at the same time as a function or process.

For more information about each method, variable, or function, please refer to the comments in each class file.