Homework 3 -- 150

For this homework, you'll be **writing 5 Perl scripts (Perl 5)**. The requirements for each are given below. You'll turn in each script as well as a screenshot (or more than one) for each script's execution.

If you have any questions please email me or post in the Homework 3 discussion topic in the Questions forum.

.

Functional Requirements

Script 1 (10 points):

1. Get a copy of web_blast.pl from Module 3, mod03_example_files.zip.
2. Get the file protein.fasta from mod03_example_files.zip.
3. Edit web_blast.pl to
   a. Print the value of $rid.
   b. Make sure the $rtoe is printed as well as the status lines (by default the statuses go to STDERR, which is fine). If any are commented, uncomment them. You should make sure you print the "Search is complete" line.
   c. Edit the Blast.cgi call that gets the BLAST results (last call) to return the format type XML.
   d. Output the response->content (provided it is successful) to a file called "web_blast_output.xml"
   e. Print a statement that says "Success" after you have written to the output file.
4. Execute your modified copy of web_blast.pl to search for protein.fasta using blastp and nr (non-redundant database).
5. Take a screenshot of your executing web_blast.pl to show what the values of $rid and $rtoe are and all the occurrences of "Searching…", "Search complete", "Success", and any errors. THIS MAY TAKE A LONG TIME and you may have many "Searching…" lines. Therefore, you may need to take 2 screenshots, one of the beginning of the run and one at the end.

Script 2 (15 points):

1. Obtain the data from this URL (it is a GET and uses the newest PDB API) using a web service call (LWP):
   http://data.rcsb.org/rest/v1/core/entry/4HHB

2. The data will be returned in JSON format. For debug purposes, you may want to use Data::Dumper on the decoded JSON, but comment that out for your final script.
3. Print the struct title. Those are the names of the nested fields – struct and then title.
4. Loop through the array that is in {citation} (the field name is citation) and do the following:
    a. If the year (year is the field name) is 1975, get the citation title (title is the field name here too)
    b. Print the year and title. You should only print out the title for the citation from 1975.
5. Take a screenshot of your output.

Script 3 (40 points):

1. Get the mod06_homework_files.zip file and extract. You'll see there is a data subdirectory with XML files in it. These are the results of doing a blastp search with each of the UniProt IDs from Homework 2.
2. Get your Homework 2 SQLite database file.
3. Use a Perl "glob" to get all the filenames from the data dir. I recommend that you keep the data directory as a subdirectory of the location where your script will run.
4. Use Perl to create a subdirectory under where your script executes called "output_data". Check if the subdirectory "output_data" already exists and do not try to create it a second time if it is already there.
5. Loop through the file names you got from "glob" and do the following for **each file name**:
    a. Print the file name you are currently processing to the terminal/screen/console.
    b. Extract the UniProt ID from the file name.
    c. Create a name for an output file that looks like the example below, but use each UniProt ID as appropriate. There will be one output file for each input file, but remember to put the output file in the output_data directory and change the extension to .txt. Note: you can use a / (forward slash) in the directory path even if you are running on Windows in the filename.

       output_data/P13773.txt

    d. Print a statement that shows the UniProt ID and the output file name.
    e. Read in the XML file.
    f. Parse the XML file to obtain the **top 5** matches, in particular their **accession ids**, the **e-value** (called Hsp_evalue), and the **number of identities** (called Hsp_identity).
        i. **NOTE**: If one accession number has more than one e-value or identity, you may choose one (first, last, best -- you decide, and document what you do in your code).

   g. Print out the uniprot id and each accession id.
   h. Output to the output file the hit number and the accession number in the following format, one hit per line. There will be 5 lines per output file.

      Hit: <put hit number><tab>AccessionID: <accession id>

      For example:
      Hit: 1	AccessionID: XP_644603

6. Open the SQLite database from homework 2.
   a. Create a new table with columns AccessionID, UniProtID, EValue, and Identity. Do not create the table again if it already exists.
   b. Save the top 5 matches' data for each of the files you process.
   c. Print out a statement that indicates the accession id you are saving to the database.
7. Take a screenshot of each step in the above process. This is why there are the print statements requested above. You can add other print statements if you wish.

Script 4 (65 points):

1. Use any web service you like EXCEPT UniProt or a BLAST search.
   a. The web service needs to be able to return information given an identifier (any type), text, or protein sequence.
   b. You may use eUtils.
2. Use any of the identifiers you've collected in your database, such as UniProt accession number, PDBID, or the sequence data.
   a. If you want to collect another identifier like a gene id, or use text from a name or description to do a text search, that is OK, too.
3. Take all the data elements for the type you picked above and use each one to get data from the web service you've chosen.
   a. Print a statement that shows what data you are requesting
4. Extract some data from the results (you can choose). One or two elements is fine.
   a. Print a statement that shows the request is complete
5. Create a new table in your database to hold this information, or if it makes sense, modify one of the tables to add a new column. If you want add some additional tables or data into your database, you can.
   a. Print a statement indicating that you are adding a piece of information (once per insert)
6. Take a screenshot of each step where there are print statements. You can add other print statements if you wish.

Script 5 (15 points):

1. Allow the user to enter a query (can be an identifier or text) to get some rows from the table with the web service information you extracted in script 2.
    a. If you want to do a join with other tables in your database, you can.
2. Display the output to the user. A simple command line input and output to the screen is fine.

Other Requirements

1. The scripts need to be in Perl 5 and use SQLite. No exceptions.
2. They need to run and you must submit evidence (screen shots) of execution. Also, I will likely also try running your code.
3. Good program structure and comments. I will be reading the code. I must be able to understand it without undue mental effort.
4. You must zip up your scripts, screenshots, database file, and any files created by your scripts into a zip file with the following naming convention: **your-lastname_HW3.zip**. I will return any submissions with the wrong file name/format and will not grade your homework until it is submitted correctly. Late penalties will apply.
5. Don't forget to follow the instructions closely so you won't lose any points.

Homework assignment 3 is worth **150** points. See the Calendar for the due date.

Please plan accordingly, and **start early** if you think you will have questions. Let me know right away if you have questions about what to do regarding the assignment.

You will find the link to upload the assignment in **Module 7**, when the assignment is due. If you need to submit earlier than this, please let me know.