**Homework Assignment 6**

*Due date and time: 11 AM, Wednesday, February 17th, 2016*

***Please note that this particular homework assignment is a group activity. You must work with other members of your group on this assignment and submit a consolidated report that describes your findings.***

*Please submit homework assignments in hard copy and electronic formats. The python script must be uploaded as \*.py file for testing. Please also submit your python script in hard copy format.*

#### You have just discovered that the mouse protein MRG15/MORF4L1 (GenBank ID or gid: 13277348) that you have been investigating is a chromatin interacting protein that is recruited by numerous chromatin-associated complexes. You now wish to learn about the patterns of sequence conservation within this protein family for follow-up functional studies.

#### **TASK 1**: Perform a BLAST search with the goal of assembling the sequences of its orthologs in a broad range of species (use the file species.txt provided that lists all the species). Check to make sure all the species are represented in the blast output. Download the XML output file produced by BLAST. Write a python script to scan the file and designate the highest scoring hit to the query sequence as the putative ortholog in each of the species above. In the script, import the **urllib2** module and use it along the lines below (see [this](http://www.ncbi.nlm.nih.gov/books/NBK25499/#_chapter4_EFetch_) URL for information on how to construct your fetch string using the NCBI EUtils utility; note that the string “&retmode=text&rettype=fasta” needs to be appended to the fetch string). Use the following block of code to download the sequence for a given gid (a variable which will be different for each orthologous sequence) in FASTA format. The downloaded sequence is stored in the fasta variable:

#### fetch = urllib2.urlopen("http://www.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?db=protein&id=" + str(gid) + "&retmode=text&rettype=fasta")

#### fasta = fetch.read()

#### Use the script to compile the orthologous sequences into a single file, as the file would be useful for the next task. Provide a hard copy of this FASTA file. (**12 points**)

#### **TASK2:** Now generate an MSA using the FASTA file above using your favorite algorithm (i.e. CLUSTAL Omega or MUSCLE). Your studies suggest that a domain at the C-terminus is responsible for interactions with a broad range of proteins.

1. Download the alignment and use [Jalview](http://www.jalview.org/download.html) (use the desktop version, as it has more functionality) to read in the MUSCLE/CLUSTAL  generated MSA and ‘edit’ the alignment, so as to restrict it to residues that make up the C-terminal domain of the protein. Run a BLAST search to check whether there is a known structure for the C-terminal domain in the PDB; this should guide the selection of residue ranges for this domain. Generate a sub-alignment of this domain along the lines described below (see tips below) and provide a hard copy. Examine the PDB file using PyMol for conserved, solvent-exposed residues. Suggest 3 residues you would mutate so as to disrupt interactions with partner proteins. (**4 points**)

(**Tips**: clicking on any residue in Jalview will give its residue number at the bottom left of the window and you can select any region of interest using the mouse; you can then output the sub-alignment by right-clicking the mouse and outputting the selection to a text-box in FASTA format; clicking on the New Window button will give you a new window with the sub-alignment; color the alignment using the BLOSUM62 score to view the alignment in a more visually interesting format; the alignment can be exported as an EPS file for editing in Adobe Illustrator).

1. Use the FASTA output for the sub-alignment from T2i above to generate a profile HMM using the [Motif Search](http://motif.genome.jp/MOTIF3.html) web server (the program is called HMMER – pronounced ‘hammer’ and purportedly is a more precise tool than BLAST). Save the profile and click on the link that says “Jump to Search the Sequence Database”. Search the KEGG database using the profile HMM you generated for matches to protein(s) in *Saccharomyces cerevisiae* (the organism code is sce, so search for this string in your hits) that have this domain. What is the name of the protein that has this domain? (**3 points**)
2. Search the [Pfam](http://pfam.sanger.ac.uk/) database with the mouse MRG15/MORF4L1 sequence. Rationalize your results with what you found in Task 2i and ii. (**1 point**)

**Pseudocode for TASK 1**:

1. Create a list of species names of interest all collected in one file in a suitable format
2. Import the **urllib2** and **re** modules
3. Prompt user for species filename
4. Open species.txt and read species names into a list
5. Prompt user for blast xml file
6. Call a function that would return a list of accession codes using blast xml output file and species list as arguments to the function call
   1. For each element of the species list
   2. Parse blast output but open the xml output file first
      1. Scan the blast output for tags that have strings “<Hit\_id>” or “<Hit\_def>”
         1. Store gi accession code if the tag is a Hit\_id (using appropriate pattern matches); the code is the string of numbers next to the string “gi|”
         2. Store the species if the tag is Hit\_def (using appropriate pattern matches)
            1. At the first instance of the species above being a member of the species array, ‘push’ gi accession to the list of accession codes and terminate blast output parsing (this is because the first species match is the high scoring match as blast has already ordered the output by bit scores)
   3. Return the list of gi accession codes
7. For each element of accession list
   1. ‘Get’ the sequence for the element in FASTA format
   2. Append the FASTA sequence to a pre-defined output file

**Tips**: What you will be coding is a very specific parser suited to your needs.  In the parser, you would read the xml file line by line.  Look for matches to the pattern "<Hit\_id>" or "<Hit\_def>" as these are the lines with useful information (accession code and species id, respectively).