CSC 314, Group Project
Cancer Bioinformatics

Background: The *C. elegans* gene *cep-1* is related to *TP53*, a transcription factor and tumor suppressor gene in humans that regulates the cell cycle. Mutations in *TP53* can result in uncontrolled cell growth, which can result in cancer.

Objective: Identify candidate targets of cep-1, which are genes that have a TP53 binding consensus sequence in their promoters (up to 1000 bp upstream of the gene). Identify all *C. elegans* genes with at least 2 binding sites, and provide a table with the following information: the worm gene name, the number and locations of the binding sites, the human orthologs (if they exist), a summary of the human gene. (You may be asked to identify Gene Ontology (GO) terms at a later date.

The TP53 binding site has the consensus sequence:
G/A, G/A, C, A/T, A/T, G, T/C, T/C

Requirements:

I. Identification of candidate cep-1 targets [30 points]

1. Download the promoter regions of all *C. elegans* genes using the UCSC Table Viewer (and the RefSeq Genes track and RefGene table) and save to a file.
2. Write a python script that does the following:
    a. Reads in each sequence, and identifies all sequences containing at least 2 occurrences of the consensus sequence. (Note: There are 8,874 genes that contain 2 binding sites, out of 53764 genes)
    b. For each sequence containing at least 2 occurrences, output the following to a file:
        i. The transcript ID (e..g, NM_001306315) of the sequence, which needs to be extracted from the seq object ID
        ii. The number of candidate binding sites
        iii. The positions of the candidate binding sites

II. Mapping from RefSeq accession numbers to worm gene names [25 points]

1. From the UCSC Table Viewer, download a table showing the RefSeq IDs and corresponding gene names. This is accomplished by using the same settings as in Part I, except the "output format" is"selected fields…" and the selected fields are "name" and "name2".
2. Write a python script that reads in the file produced from Part I and for each match adds the corresponding gene name. The new results are output to a file. The recommended way to do this is as follows: read in the mapping file from (1) and create a python dictionary using the RefSeq ID

as the *key* and the gene name as the *value*. Note that there will be multiple RefSeq IDs for some genes.

III. Finding the human orthologs [25 points]

1. Using Biomart, create a file containing a list of worm gene names and human orthologs. For me, this file contains 9028 results (including genes with no orthologs and genes with multiple orthologs). Note that Biomart advises against queries with more than 500 records, but you can probably include all results in a single file.
2. Write a python script that reads in the file produced from Part II and for each match adds the corresponding human ortholog, if it exists and if the confidence for the ortholog is high (i.e., it is 1). This script creates a new file for only those matches with the orthologs, that contains all previous information as well as the ortholog gene name. Note that if a worm gene has multiple orthologs, only one of the orthologs needs to be included (although you may include multiple orthologs in the output if you prefer). For my results, I end up with orthologs for 547 worm genes, though several genes are included multiple times in my results.

IV. Find human gene summaries [20 points]

1. Read in the results file produced from Part III, and add a description and summary for each human ortholog. The description and summary should be found by using (and possibly modifying) the script *getEntrezGeneSummary.py* provided in class, which queries the Entrez Gene database.

Submission: All Python scripts used must be submitted through Blackboard, as well as all input/output files, with the exception of the file containing the promoter sequences. At the top of each Python script, you must include a comment that contains the author or authors, as well as a brief explanation of what the script does, including a description of any files that are read by or that are created by the script.