CSE 231 Spring 2018

Programming Project 06

Edit 2/21: removed one line from Function Test 2 to improve clarity.
This assignment is worth 45 points and must be **completed and turned in before 11:59 on Monday, February 26, 2018.**

Assignment Overview

This assignment will give you more experience on the use of:

1. Lists and tuples
2. function
3. File manipulation

The goal of this project is to extract gene lengths from a gene annotation file. With a gene annotation GFF file, you will need to extract the gene coordinates on each chromosome and calculate the average and standard deviation of gene lengths.

Assignment Background

The eukaryotic genome is composed of multiple chromosomes. On each chromosome, there are multiple genes. In bioinformatics, the genome annotations can be saved in a file format called GFF. In NCBI genome database (https://www.ncbi.nlm.nih.gov/genome/), there are many publically available annotated organisms. These annotated genomes can be downloaded in multiple file formats, including GFF format. For this project, we will focus on a relatively simple model species: Caenorhabditis elegans. This worm has a genome of six chromosomes named chrI, chrII, chrIII, chrIV, chrV, and chrX.

We provide two input files:

```
C.elegans_small.gff    # a small file for development
C.elegans.gff          # a real BIG data file
```

Project Description

a) **`open_file()`** prompts the user to enter a filename. The program will try to open a tab- separated GFF file (a text file). An error message should be shown if the file cannot be opened. This function will loop until it receives proper input and successfully opens the file. It returns a file pointer.

b) **`read_file(fp)`** receivers a file pointer of the data file and read all the genes information. For this project, we are only interested in the following columns: the chromosome name (string) is in column 0, the gene_start is in column 3, and the gene_end is in column 4. Convert number strings to int. No other values are needed for this project. If a value is missing, use 0 as the value.
For each gene, save it in a tuple, (chromosome, gene_start, gene_end), and append each tuple to a list of genes. Sort the list and then return the sorted list of genes (sorting makes a canonical list for comparison testing on Mimir).

b) **`extract_chromosome(genes_list, chromosome)`** receives a list of genes (such as what was returned by the **read_file**() function) and a chromosome name, extract the gene information for this chromosome and save in list chrom_gene_list. Sort and return the list (sorting makes a canonical list for comparison testing on Mimir).

c) **`extract_genome(genes_list)`** receives a list of genes and extract the gene information for each chromosome. In this function, use extract_chromosome(genes_list, chromosome) to extract the genes for each chromosome (chrom_gene_list) then save the returned result in a list genome_list (a list of six chrom_gene_lists). Sort and return the list (sorting makes a canonical list for comparison testing on Mimir).

d) **`compute_gene_length(chrom_gene_list)`** This function receives chrom_gene_list, the list of genes for a specific chromosome (such as returned from the extract_chromosome() function). For each gene, compute the gene length and save the result in a list gene_length. Given the gene length list, compute the average gene length and standard deviation for these genes. Save the results in a tuple with gene length first followed by gene standard deviation. Return that tuple.

The length of one gene, **gene_len**, is calculated as: **gene_len = gene_end – gene_start + 1**
(Hint: make a list that has the lengths of the genes.)

The **gene_mean** is the average length of all the genes (Hint: consider using list `sum()` and `len()` functions—something you can do if your lengths are in a list.)

The **gene_number** is a count of all the genes.

The **gene_stddev** is the standard deviation of all the gene lengths, calculated according to the following formula. The summation in the formula sums across all genes. That is, for each gene subtract the mean from the length and square the difference; then sum those values. Take that sum and divide by the number of genes (**gene_number**); then take the square root (remember to `import math`). (Hint: "for each gene" can be done easily by walking through a list with a `for` loop, if your lengths are in a list.)

$$gene\_stddev = \sqrt{\frac{\sum (gene\_len - gene\_mean)^2}{gene\_number}}$$

e) **`display_data(chrom_gene_list, chrom)`** This function receives chrom_gene_list, the list of genes for a specific chromosome as well as chrom, the string name. It displays the chromosome name, the average length of the gene and the standard deviation (from the **`compute_gene_length()`** function that gets called within this function). The chromosome name must be displayed with the first three characters 'chr' in lower case and the remaining characters in upper case, e.g. 'chrIV'. (Hint: slicing is your friend.)

Assignment Deliverable

The deliverable for this assignment is the following file:

`proj06.py` – the source code for your Python program

Be sure to use the specified file name and to submit it for grading via the **Mimir system** before the project deadline.

Assignment Notes

1. The gff input files we provide have lines starting with ‘#’ as file annotations; skip these lines when reading the gff file.
2. To parse the lines in gff file, use split() function. You can split by the tab.
3. Use this constant for chromosome names

```
    CHROMOSOMES = ['chri','chrii','chriii','chriv','chrv','chrx']
```

4. Items 1-9 of the Coding Standard will be enforced for this project.

Test Cases

Function Test 1: read_file

```
Input file: C.elegans_small.gff
Returns:
[('chri', 3747, 3909), ('chri', 4221, 10148), ('chri', 11641, 16585),
('chrii', 25, 175), ('chrii', 25, 175), ('chrii', 1867, 4663), ('chriii',
1271, 2917), ('chriii', 4251, 11940), ('chriii', 12189, 14753), ('chriv',
695, 14926), ('chriv', 8765, 11070), ('chriv', 15499, 20899), ('chrv', 180,
329), ('chrv', 180, 329), ('chrv', 2851, 6511), ('chrx', 151, 263),
('chrx', 151, 263), ('chrx', 13494, 13643)]
```

Function Test 2: extract_chromosome

```
genes_list = [('chri', 3747, 3909), ('chri', 4221, 10148), ('chri', 11641,
16585), ('chrii', 25, 175), ('chrii', 25, 175), ('chrii', 1867, 4663),
('chriii', 1271, 2917), ('chriii', 4251, 11940), ('chriii', 12189, 14753),
('chriv', 695, 14926), ('chriv', 8765, 11070), ('chriv', 15499, 20899),
('chrv', 180, 329), ('chrv', 180, 329), ('chrv', 2851, 6511), ('chrx', 151,
263), ('chrx', 151, 263), ('chrx', 13494, 13643)]
chromosome = 'chrv'

Returns:
[('chrv', 180, 329), ('chrv', 180, 329), ('chrv', 2851, 6511)]
```

Function Test 3: extract_genome

```
genes_list = [('chri', 3747, 3909), ('chri', 4221, 10148), ('chri', 11641,
16585), ('chrii', 25, 175), ('chrii', 25, 175), ('chrii', 1867, 4663),
('chriii', 1271, 2917), ('chriii', 4251, 11940), ('chriii', 12189, 14753),
('chriv', 695, 14926), ('chriv', 8765, 11070), ('chriv', 15499, 20899),
('chrv', 180, 329), ('chrv', 180, 329), ('chrv', 2851, 6511), ('chrx', 151,
263), ('chrx', 151, 263), ('chrx', 13494, 13643)]

Returns:
[[('chri', 3747, 3909), ('chri', 4221, 10148), ('chri', 11641, 16585)],
[('chrii', 25, 175), ('chrii', 25, 175), ('chrii', 1867, 4663)],
[('chriii', 1271, 2917), ('chriii', 4251, 11940), ('chriii', 12189,
14753)], [('chriv', 695, 14926), ('chriv', 8765, 11070), ('chriv', 15499,
20899)], [('chrv', 180, 329), ('chrv', 180, 329), ('chrv', 2851, 6511)],
[('chrx', 151, 263), ('chrx', 151, 263), ('chrx', 13494, 13643)]]
```

Function Test 4: compute_gene_length

```
chrom_gene_list = [('chrii', 25, 175), ('chrii', 25, 175), ('chrii', 1867,
4663)]
```

```
Returns:
(1033.0, 1247.33636201307)
```

Test Case 1

```
Gene length computation for C. elegans.


Input a file name: C.elegans_small.gff


Enter chromosome or 'all' or 'quit': chri

Chromosome Length
chromosome      mean  std-dev
chrI         3678.67  2518.14


Enter chromosome or 'all' or 'quit': chriv

Chromosome Length
chromosome      mean  std-dev
chrIV        7313.00  5053.00


Enter chromosome or 'all' or 'quit': chrX

Chromosome Length
chromosome      mean  std-dev
chrX          125.33    17.44


Enter chromosome or 'all' or 'quit': quit
```

Test case 2

```
Gene length computation for C. elegans.


Input a file name: xxx
Unable to open file.

Input a file name: C.elegans_small.gff


Enter chromosome or 'all' or 'quit': xxx
Error in chromosome.  Please try again.


Enter chromosome or 'all' or 'quit': chrII

Chromosome Length
chromosome      mean  std-dev
chrII        1033.00  1247.34
```

```
Enter chromosome or 'all' or 'quit': CHIII
Error in chromosome.  Please try again.


Enter chromosome or 'all' or 'quit': CHRIII

Chromosome Length
chromosome      mean  std-dev
chrIII       3967.33  2658.87


Enter chromosome or 'all' or 'quit': aLL

Chromosome Length
chromosome      mean  std-dev
chrI         3678.67  2518.14
chrII        1033.00  1247.34
chrIII       3967.33  2658.87
chrIV        7313.00  5053.00
chrV         1320.33  1655.10
chrX          125.33    17.44


Enter chromosome or 'all' or 'quit': qUiT
```

Test Case 3

```
Gene length computation for C. elegans.


Input a file name: C.elegans.gff


Enter chromosome or 'all' or 'quit': all

Chromosome Length
chromosome      mean  std-dev
chrI         2542.65  4104.10
chrII        1879.71  2945.42
chrIII       2469.57  3761.81
chrIV         535.14  1949.55
chrV         1711.47  2687.29
chrX         1575.51  3110.69


Enter chromosome or 'all' or 'quit': quit
```

Test Case 4
Blind test.

Grading Rubric

```
General Requirements:
  __0__ ( 5 pts) Coding Standard 1-9
             (descriptive comments, function headers, etc...)
```

```
Implementation:
  __0__ ( 4 pts) open_file function (no Mimir test)
  __0__ ( 4 pts) read_file function
  __0__ ( 4 pts) extract_chromosome function
  __0__ ( 4 pts) extract_genome function
  __0__ ( 4 pts) computer_gene_length function

  __0__ ( 5 pts) Pass Test1
  __0__ ( 5 pts) Pass Test2
  __0__ ( 5 pts) Pass Test3
  __0__ ( 5 pts) Pass Test4 (Blind Test)
```