CS201 Programming for AI
Assignment 02: Using builtin Python structures.
Due : 23h55 Friday 12 November, 2021.

FCSE
GIKI

Fall 2021

1 Objectives

(a) Test the student's ability to use builtin structures.

(b) Test the student's ability to write functions.

(c) Test the student's ability do error handling in code.

(d) Test the student's ability to follow written instructions.

2 Description

You will be using the Python language to do this assignment. You can't use any external libraries. Only the Python Standard Library is allowed.

This assignment weighs 3% of your overall grade.

3 Task 1

T1a. Count 2-letter sequences.

A DNA sequences consists of sequence of letter representing genes. You should write a function that takes one argument - a DNA sequence in the form of a string, and returns a dictionary containing the frequencies of all 2-letter sequences found in that string.

For example take the following DNA string:
"ACCTAGCCCTA"

If your function is passed the above string, it should return the following dictionary:
{'AC': 1, 'CC': 3, 'CT': 2, 'TA': 2, 'AG': 1, 'GC': 1}

The passed string could be arbitrarily long and can have genes letters arranged in arbitrary order.

T1b. Count n-letter sequences.

A DNA sequences consists of sequence of letter representing genes. You should write a function that takes two argumnts: 1) a DNA sequence in the form of a string, and 2) an inteter `n`. It should return a dictionary containing the frequencies of all `n`-letter sequences found in that string.

For example when given the string **"ACCTAGCCCTA"** and the integer **3**, it should return the following dictionary:
{'ACC': 1, 'CCT': 2, 'CTA': 2, 'TAG': 1, 'AGC': 1, 'GCC': 1, 'CCC':1}

The passed string could be arbitrarily long and can have genes letters arranged in arbitrary order.

4 Task 2

T2a. .

You are given a file which contains molecular data. Each line that begins with `ATOM` contains information about a molecule. The third column (assuming the first column is numbered 1) contains the molecule name. For example in the figure below for the first line the molecule name is `CA`. Similarly each columns contain a certain information about that molecule.

```
ATOM    114  CA   GLU A   8      13.946   1.022   5.505  1.00  0.43           C
ATOM    115  C    GLU A   8      13.659  -0.390   6.019  1.00  0.39           C
ATOM    116  O    GLU A   8      14.559  -1.170   6.260  1.00  0.45           O
ATOM    117  CB   GLU A   8      14.207   1.950   6.696  1.00  0.53           C
ATOM    118  CG   GLU A   8      14.064   3.409   6.256  1.00  1.39           C
ATOM    119  CD   GLU A   8      14.660   4.324   7.328  1.00  1.74           C
ATOM    120  OE1  GLU A   8      14.225   4.232   8.464  1.00  2.30           O
ATOM    121  OE2  GLU A   8      15.540   5.100   6.994  1.00  2.19           O
ATOM    122  H    GLU A   8      12.430   2.419   4.873  1.00  0.47           H
ATOM    123  HA   GLU A   8      14.814   1.003   4.863  1.00  0.46           H
ATOM    124  HB2  GLU A   8      13.494   1.739   7.479  1.00  1.12           H
ATOM    125  HB3  GLU A   8      15.208   1.785   7.067  1.00  1.14           H
ATOM    126  HG2  GLU A   8      14.587   3.557   5.322  1.00  2.04           H
ATOM    127  HG3  GLU A   8      13.019   3.645   6.127  1.00  2.01           H
ATOM    128  N    GLN A   9      12.408  -0.720   6.195  1.00  0.35           N
ATOM    129  CA   GLN A   9      12.051  -2.069   6.699  1.00  0.38           C
ATOM    130  C    GLN A   9      11.745  -3.001   5.523  1.00  0.35           C
ATOM    131  O    GLN A   9      12.213  -4.121   5.468  1.00  0.40           O
```

Figure 1: Snapshot of part of a file containing molecular data

Write a function that takes a filename as a string argument and then a series of keyword arguments. Each keyword argument will be of the form `'XXX'=N` where 'XXX' is a string denoting a molecule name and N will be an integer containing the column number. For each such keyword argument, your function should find all lines containing information about molecule 'XXX' and return the information contained in column `N` for each such line.

It should return such information about all the keyword arguments in the form of a dictionary. Each line of a dictionary will be a `(key, value)` pair where the key will be the molecule name `'XXX'` and the value will be a list of tuple `(x,y)` where `x` is the line number on which the molecule `'XXX'` was found and `y` will be information contained in column `N` of that line. For example if the filename for above figure was `"fname.pdb"` the function call with the filename and two keyword arguments:

```
get_molecule_data("fname.pdb", 'CA'=7, 'N'=11)
```

would return:

```
{'CA':[(1,13.946), (16, 12.051)], 'N':[(15, 0.35)] }
```

Since `'CA'` occurs twice and `N` only once in the shown figure, the returned dictionary contains two entries: basically the values of the 7th column for `'CA'` on lines 1 and 16, and the value of the 11th column for `'N'` on line 15. Assume both the line numbers and column numbers start with 1.

The function should be able to handle any number of keyword arguments provided after the filename. In case of no keyword arguments after the filename, it should return an empty dictionary.

You can test your code on `1bta.pdb` file provided in attachment.

5 Error Checking and Clean Code instructions

It is extremely important that your program handles errors correctly, i.e., it should be able to detect when something goes wrong and should exit gracefully after displaying a useful error message. Under no circumstances shall it CRASH!!

Things that can go wrong may include, but are not limited to, :

- file names `input.txt` is not present
- file names `input.txt` is present but is empty
- ..., etc.

You would lose marks if your program crashes during use.

It is the programmer's responsibility to free any system resources.In this case you will see that all files opened by a program should be closed before the program ends. Your program should always close any and all opened files.

Code should be properly indented, readable and commented.

If your program crashes, you will get a 0.

6 Submission Instructions

1. You submission should consist of one .py file only. This file should contain all three functions.
2. You shall name your submission as u2020xxx_a2.py where xxx are the last three digits of your registration number.
3. You will submit on MS Teams.
4. Missing submission deadline on will cost you (50%) marks. Submissions received more than 24 hours after submission deadline will get a 0.

7 Rubric

This is an individual assignment. Any form of collaboration, cheating, plagiarism will get you a 0. Giving your code to somebody else, even if it is for their understanding only, is not allowed. You may be called for a viva; if you are unable to explain any line of the submitted code, you'll get a 0.

Any form of plagiarism or collusion will get you at least a 0 in the assignment and, potentially, an F in the course.

To discourage plagiarism and encourage academic honesty, if you've been unable to do any thing you can submit a program saying Hello World before the deadline by following submission instructions (name your file u2020xxx_a02_hw.py), and get the submission marks. This way you are sure to get at least 25% of the marks.

Category	Marks
Followed submission insns	05 marks
Code was readable + Compiled without warnings + Does not crash + Program handles errors well	05 marks
T1 working properly	10 marks
T2 working properly	10 marks
Total	30 marks