Scientific Computation
Autumn 2024
Project 1

Due: November 1st, 1pm

In addition to this project description, there are 2 files for this assignment:

- project1.py: file which you will complete and submit on Blackboard (see below for details)
- report1.tex: a latex template file for the short report which you will submit. The discussion and figure(s) described below should be placed in this report.

Part 1

Consider the following problem. You are given a Python list containing n integer IDs and receive a sequence of m integer IDs. The IDs in the sequence should be processed one at a time. For each ID in the sequence, return a location in the list where it can be found or return -1000 if it is not in the list. You have been provided with two functions, *method1* and *method2*, which can be used to solve this problem. Note that *method2* relies on three other provided functions, and the input for *method2* must be set in a sensible manner so it is used efficiently for this problem.

1. (4 points) For each method: (a) briefly (in a few sentences) describe the underlying strategy used to solve the given problem and (b) discuss the worst-case asymptotic time complexity when the method is sensibly applied to the given problem. For (b), you do not need to provide a detailed operation count, and it is fine to state results from lecture.

2. (6 points) Design one or more timing tests to critically evaluate and compare the performance of *method1* and *method2*. Your test should analyze how the wall time required by each method depends on n and m and should allow you to critically compare the methods. Your code should generate up to 4 well-designed plots that illustrate what you consider to be important trends. In your report, explain the design of your test(s), discuss the trend(s) shown in your plot(s), and also carefully discuss the degree to which the trends agree with what you expect based on your earlier discussion of the time complexities of the methods. You do not need to consider values of n larger than 10000 (though you are welcome to do so), and your discussion should not be limited to trends for the largest values of n and m considered.

   Place the code used to generate the figure(s) in the function, *part1_test*. Place your discussion and plots in your report.

Notes: You will not be assessed on how well your timing test results match your theoretical analysis, however there should be at least some correspondence. Assessment will instead focus on the reasoning used in the design of your test and in the discussion of its results. A figure with x subplots will be counted as x plots.

Part 2

In part 2, you will develop code for analyzing amino acid (AA) sequences. There are 20 naturally-occurring AAs, but here we will assume sequences only contain one or more of the 9 non-polar AAs, and we will use lower case letters from 'a' to 'i' as labels for these acids. You will develop a function to search for patterns in a pair of length-n AA sequences, A_1 and A_2. The patterns are stored in the input list L which contains l 2-element sub-lists; each element of a sub-list is a length-m AA sequence. For each pair of AA sequences in L, you should find all locations where the first sequence appears in A_1 and the second sequence appears in the same location in A_2. For example, if A_1 = afgabciiabcb, A_2 = feadefdsdefe, and $L = [[abc, def], [afg, fea]]$ then $n = 12$, $m = 3$, $l = 2$, and the search should identify the locations 3 and 8 for the first pair in L and 0 for the second pair. Specifically, the function should return a list of lists, F, where F[i] is a list containing all locations in A_1 and A_2 where the ith length-m pair in L can be found at the same location (for the example above, F = [[3,8],[0]]). If no matches are found for the ith pair in L, you should have, F[i] = [ ]. A_1, A_2, and L are provided as input to the function, *part2*.

1. (6 pts) Complete the function *part2*, so that it efficiently constructs F as described above. You should design your code for input with $m \gg 1$, $l \gg 1$, and $n \gg m$ though it may be helpful to consider smaller problem sizes when developing and testing your code. You should also assume that $m < l < n-m$. Your algorithm and its implementation should be efficient with regards to time complexity, and for two methods with comparable time complexities, the method with lower memory usage should be selected. You should also assume that the cost of Python integer arithmetic is independent of the length of the integer. See the function documentation for further details on the function input/output.

2. (4 pts) Add a brief description of your code along with a clear and concise analysis of its time complexity to your report. Include an explanation of why your code should be considered to be 'efficient'. You do not need to run timing tests or discuss the wall time required by your code, and you are not being asked to optimize your code for wall time.

Further guidance

- You should submit both your completed python file (*project*1.*py*) and a pdf containing your discussion and figure(s). You are not required to use the provided latex template, any well-organized pdf is fine. To submit your assignment, go to the module Blackboard page and click on"Project 1". There will be an option to attach your files to your submission. (these should be named *project*1.*py* and *report*1.*pdf*). After attaching the files, submit your assignment.

- Marking will be based on the correctness of your work and the degree to which your submission reflects a good understanding of the material covered up through the slides of lecture 5 and corresponding labs. You should aim to keep the amount of text in the pdf version of your report to less than 2 pages.

- You may use chatgpt-4o and claude 3.5 sonnet for the assignment without restriction (other llms and code-generation tools are not allowed). Any submitted llm-generated code must be clearly referenced with comments indicating what is llm-generated and which specific llm was used (e.g. chatgpt-4o). Prompts used to generate submitted code should be included in an appendix of the submitted report. Similarly, submitted discussion which draws substantially on llm-generated content should be clearly referenced and relevant prompts should be included in a report appendix. Note that both chatgpt-4o and claude 3.5 sonnet are free but have usage limits, and it is your responsibility to manage your usage appropriately.

- Please do not modify the input/output of the provided functions without permission. For part 1, you may import and use any modules that have been used or mentioned during the term so far. Otherwise, please do not import any modules without permission. You may create additional functions as needed, and you may use any code that I have provided during the term.

- Open-ended questions require sensible time-management on your part. Do not spend so much time on this assignment that it interferes substantially with your other modules. If you are concerned that your approach to the assignment may require an excessive amount of time, please get in touch with the instructor.

- Questions on the assignment should be asked in private settings. This can be a "private" question on Ed (which is distinct from "anonymous") or by arrangement with the instructor.

- Please regularly backup your work. For example, you could keep an updated copy of your files on OneDrive.

- In order to assign partial credit, we need to understand what your code is doing, so please add comments to the code to help us.

- You have been asked to submit code in Python functions, but it may be helpful to initially develop code outside of functions so that you can easily check the values of variables in a Python terminal.

- The weightings for assignments can be found in the module overview slides.

- Your submission should be your own work, you should not collaborate with other students, and you should clearly and completely cite external sources which are used in your work (detailed citations of slides/labs from the module are not needed).