ASSIGNMENT #6 - PROJECT: Data Processing. DESCRIPTION, INCLUDING SUBMISSION INSTRUCTIONS

READ ALL THIS DOCUMENT INCLUDING DESCRIPTION AND REQUIREMENTS

This is an up to 3 people team work (you can also work in a team of 2 people or individually if you prefer so). While you may discuss generalities with colleagues from other teams about this exercise, you cannot develop the same code, nor share code among teams, nor obtain code from other sources.

Being a team exercise, it places a big responsibility on each individual. You want to respect and be honest with your partners and with yourself: DO YOUR SHARE, and BE KNOWLEADGEABLE OF THE WHOLE ASSIGNMENT. Working on this assignment is also a way for you to prepare for the final exam.

Deadline: Sunday December 1, 11:59 pm. No extensions are possible.
Clarifications/solutions about the problem will be discussed in the last class, on Monday December 2.

Associated to this assignment/project there is:

- provided data files
- some provided code that you can use directly or adapt (leaving the comments)
- a Codewrite Assignment, CW #6.2. Some functions that you will solve in the CW assignment will be directly useful for this assignment. A solution to those CW functions will be provided on Wednesday Nov 27. You may use or adapt those functions as needed.

A. PROBLEM SOLVING

1. FIRST: Read the Problem Solving Suggestions document. IT IS BRIEF AND HIGHLY RECOMMENDED.
2. **UNDERSTAND WHAT YOU ARE ASKED TO DO** AND WHAT YOU ARE ASKED TO SUBMIT.
3. START WORKING ON THIS AS SOON AS POSSIBLE

B. GENERAL PROGRAM DESCRIPTION

You are asked to implement a Python program, which processes students' survey data. The data is provided in a text file with some specific format. The user will have the program process all the students data, and then the user will have the possibility to check how similar specific pairs of students' responses are. The program will also show some statistics to the user, and will generate a csv (comma-separated values) file with information associated to each student. More details are provided below. You should also check the sample runs as part of the problem description.

C. DEVELOPING YOUR PROGRAM IN STAGES

***Planning stage*:**

- What is the main level structure of the whole program? Which functions would be useful to develop (and call from the main level)? (Notice that some functions are provided or may be in the Codewrite assignment) What parameters would be useful that those functions have? What should those functions return?
- Which major data structures will you use? Lists of lists? Lists of strings? Lists of numeric variables? Which major numeric or string variables will you need at the main level?
- Create a flowchart or pseudocode (at a high level). Come up with descriptive names for variables and functions. Perhaps distinguish in the naming what is a string, what is a list.

Some recommended implementation and testing stages:
***Stage I*:** Implement processing of all the students data, only do the most basic stats. Use the smaller data file to make sure that the calculations are correct.
***Stage II*.** Implement the processing of specific pairs of students. Validate numeric inputs.
***Stage III*.** Provide additional statistics, write information in the output file, test with additional data.

D. FILES DESCRIPTIONS

1) Input to the program, STUDENTS ANSWERS **file: IN_all_data.txt**

- Two data files are provided, with 3 and with 14 students, with 21 responses each. You can invent other files, with different number of students and/or answers. Make sure to rename appropriately. *Marking will be done using a different file, with name **IN_all_data.txt.***
- This description uses 'line' to refer to one student's data. One 'line' is a string, where the last character is '\n'. (Note that most editors will not show the '\n' but rather include a new line)
- Each line contains the student's name, then at least one space, and then the student's answers coded from 1 to 5, separated by spaces (and then the '\n' character).
- Hence the whole file is a long string with letters, numbers, spaces , and characters '\n'
- Assumptions:
    - Students' names do not include spaces (e.g the name is one word only)
    - Answers are ordered by question number, all students answered correctly all questions

Example with 3 students and 10 answers each
name 4 1 5 1 4 2 1 5 4 3
otherName 3 1 4 3 1 4 4 5 5 2
anotherName 4 4 1 3 2 2 3 5 3 5

"name 4 1 5 1 4 2 1 5 4 3\notherName 3 1 4 3 1 4 4 5 5 2\nanotherName 4 4 1 3 2 2 3 5 3 5\n"

2) Output from the program. AVERAGE PER STUDENT: **OUT_perstudent.csv**

The output file will be also a string, with one 'line' per student, each line including the name, then a comma, then the average of the responses (and each line ending in "\n")

Example (continuing the previous example)
name,30.0
otherName,3.2
anotherName,3.2

As a string this would be: "name,30.0\notherName,3.2\nanotherName ,3.2\n"

E. HOW DATA IS READ FROM AND WRITTEN TO FILES

You are encouraged to use the interactive textbook as reference to see more details about reading and writing data files, sections in Chapter 11 are recommended as a no-points reading assignment. Still, Python code is provided for both, reading and writing data from/to files.

Python code (the function *read_string_list_from_file(…)*) is provided to read data from a text file. When this function is called (or invoked), the function will return a list of strings, where each string contains the data associated to one student, in the same order that student data is placed in the data file. You may develop a different function to read the data. You may incorporate this function verbatim in your code, and including the comments. If you revise the function, clarify how you do it.

Following the previous example, when calling the function, and providing the file name as argument, the function provided would return:

["name 4 1 5 1 4 2 1 5 4 3", "otherName 3 1 4 3 1 4 4 5 5 2", "anotherName 4 4 1 3 2 2 3 5 3 5"]

Python code (the function *write_perstudent_to_file (…)*) is provided to write data to a file. Check the requirements and assumptions in the function code and comments.

F. OUTPUT TO BE SHOWN TO THE USER:

The information that your program asks from and shows to the user should be analogous to and in the same order and detail as in the "COMPLETE FORMAT" sample runs. Ask if in doubt.

Messages to be provided when comparing pairs of students' responses:

90% or more same responses (in the same positions)	really have a lot in common (>90%)!
50% or more	have about half opinions in common!
2 or more	have just few opinions in common (<50%)
Less than 2	have nothing in common!

G. REQUIREMENTS IN DETAIL (anything 'required' gets points)

Requirements

a) The program should execute

b) The program should have a dialog and options analogous to the one presented in the COMPLETE FORMAT sample runs

c) The results obtained by your program with the data files provided should be the same as the results shown in the sample runs.

d) Your program should work well with other data files as long as the formatting conventions in the data files are respected

e) When processing pairs, the program should validate that maximum number of pairs is a number and that names are in the data. A dialog similar to the COMPLETE FORMAT sample runs should be used.

Requirements – coding details and style

Note: The provided functions do not count, functions taken or adapted from the Codewrite assignment CW #6.2 count. For functions to count they need to be called (i.e. invoked)

f) Your program should have at least 4 "fruitful" or "productive" functions.

g) Your program should have at least 2 "void functions"

h) Your program should have at least 5 functions receiving parameters (and so that the parameters are correctly used inside the function) (these functions may be productive or void)

i) Your program should have a reasonable main level which shows the general structure of the program. The main level can be the program top level or it can be inside a "main" function, (and in the latter case at the top level you just call the main function).

j) You may use some variables as global (i.e. defined at the top level and not passed as parameters). The reason to have these variables as global would be that they relevant at the main level and may be frequently used by many functions. Yet, given the requirements above you will also have to have local variables in your functions (including parameters) and return values also.

k) All the global variables used in the program should be initialized at the top of the code with some appropriate value, 0 for integers, empty list for lists, etc, (even before the functions are defined) including a brief comment of what each variable role is. No comments are needed if the names of the variables are self-explanatory.

l) Name your variables and functions appropriately

m) At the top of the program file include as comment the authors names and dates of the versions

n) Include comments, with general descriptions of functions, special situations being true at a certain place in the program, etc. On the other hand, do not include redundant comments. For example the statement "i = i + 1" does not need the comment "i is increased by 1". Keep in mind that good naming of variables and functions reduce the need of comments.

o) Include "Trace printing" as you debug your code. When you submit your solution you may comment out some tracing prints. **However, you need to leave in your program tracing print analogous to THE SAMPLE RUNS NAMED AS COMPLETE FORMAT.**

p) Bonus points will be given if you do not break loops (with break or return statements), and rather use while statements with one or more conditions.

Requirement –clarification file (txt)

q) You need to submit an admin file (as in previous assignments). Name your admin file **"group.txt"** when you are a group of 2 or 3 members. In this case you need to include the group members names and clarify how you distributed tasks among the team members. If you are working individually, name the file **"individual.txt"**. In this latter case the file may be empty or it may include any comments about how you worked with this exercise.

r) It would be useful for you if you keep track of the time you spend on this exercise. You are asked but not required to share this information, including the total hours dedicated, considering the total time by all the team members. If you submit this information, include it in the admin file.

Requirement – Flowchart for top level

s) Submit a flowchart describing only the main/top level possibly referring to some of the global variables. You do NOT need to do a flowchart for all the details!! You may use flowgorithm or draw the flowchart by hand and take a picture/capture the screen and submit a jpg or png file.

H. WHAT YOU ARE PROVIDED

a. This description
b. Code with two functions, *read_string_list_from_file(…)* and *write_perstudent_to_file (…)*
c. Sample runs
d. Input data files for you to debug your code
e. Make sure that you check email and Canvas announcements in case that additional clarifications are provided.

I . WHAT YOU NEED TO SUBMIT

a. The code (Python file) of your final submission (allowing to see the Trace printing as indicated in the COMPLETE FORMAT sample runs)
b. Flowchart of your main level program
c. A sample run of your most polished version
d. The admin text file (group.txt or individual.text)

If you have any questions consult with the Teaching Team.
End of description of the data processing assignment #6-project.