Department of Computer Science and Software Engineering
The University of Western Australia

CITS1401
Computational Thinking with Python
Project 1, Semester 2, 2023
(Individual project)

Submission deadline: 15th September 2023 6:00 PM
Value: 15% of CITS1401

Project description:
You should construct a Python 3 program containing your solution to the following problem and submit your program electronically on Moodle. The name of the file containing your code should be your student ID e.g., `12345678.py`. No other method of submission is allowed. Please note that this is an individual project. Your program will be automatically run on Moodle for some sample test cases provided in the project sheet if you click the "check" link. However, your submission will be tested thoroughly for grading purposes after the due date. Remember you need to submit the program as a single file and copy-paste the same program in the provided text box. You have only one attempt to submit, so do not submit until you are satisfied with your attempt. All open submissions at the time of the deadline will be automatically submitted. Once your attempt is submitted, there is no way in the system to open/reverse/modify it.

You are expected to have read and understood the University's guidelines on academic conduct. In accordance with this policy, you may discuss with other students the general principles required to understand this project, but the work you submit must be the result of your own effort. Plagiarism detection, and other systems for detecting potential malpractice, will therefore be used. Besides, if what you submit is not your own work then you will have learned little and will therefore, likely, fail the final exam.

You must submit your project before the deadline listed above. Following UWA policy, a late penalty of 5% will be deducted for each day i.e., 24 hours after the deadline, that the assignment is submitted. No submissions will be allowed after 7 days following the deadline except approved special consideration cases.

Project Overview:
The ABC research institute collected information of different organisations from all over the world for their future investment purposes. The collected dataset contains several parameters about each organisation such as name of the organisation, country of the organisation registration, type of work, foundation year, number of employees, median salary, profit in 2020 and profit in 2021.

You are required to write a Python 3 program that will read a CSV file. After reading the file, your program is required to complete the following statistical tasks:

1) Find the organisations which have the highest and lowest number of employees for a specific country which are founded in the year range of 1981 to 2000 (inclusive).

2) Calculate the standard deviations for the median salary for the organisations of a specific country and all the organisations in the dataset.

3) Calculate the ratio of a specific country's net profit increase vs net profit decrease between the years 2021 and 2020. The ratio should be a positive number, where a value greater than 1 will indicate an increase in profits for a country from 2020 to 2021 and vice versa (More details given on page #03).

4) Calculate the correlation between the median salary for organisations in a specific country and their profits in the year 2021. Only use the organisations which show an increase in profits from 2020 to 2021.

Requirements:

1) **You are not allowed to import any external or internal module in python**. While use of many of these modules, e.g., csv or math is a perfectly sensible thing to do in a production setting, it takes away much of the point of different aspects of the project, which is about getting practice opening text files, processing text file data, and use of basic Python structures, in this case lists and loops.
2) Ensure your program does NOT call the *input()* function at any time. Calling the input() function will cause your program to hang, waiting for input that the automated testing system will not provide (in fact, what will happen is that if the marking program detects the call(s), it will not test your code at all which may result in zero grade).
3) Your program should also not call *print()* function at any time except for the case of graceful termination (if needed). If your program has encountered an error state and is exiting gracefully then your program needs to return zero for numerical values such as correlation, ratio and standard deviation, otherwise empty list and print an appropriate message. At no point should you print the program's outputs instead of (or in addition to) returning them or provide a printout of the program's progress in calculating such outputs.
4) Do not assume that the input file names will end in .csv. File name suffixes such as .csv and .txt are not mandatory in systems other than Microsoft Windows. Do not enforce that within your program that the file must end with a .csv or any other extension (or try to add an extension onto the provided csv file argument), doing so can easily lead to loosing marks and syntax error.

Input:

Your program must define the function **`main`** with the following syntax:

```
def main(csvfile, country):
```

The input arguments for this function are:

- `csvfile`: The name of the CSV file (as string) containing the record of the organisations around the world. The first row the CSV file will contain the headings of the columns. A sample CSV file "organisations" is provided with project sheet on LMS and Moodle.

- `country:` It will be a string parameter containing the name of a country.

Output:
We expect 4 outputs in the order below.

i) **Maximum and Minimum:** A list containing the names of organisations having highest and lowest number of employees in the input `country`. Your output should be stored in a list in the following order:
[organisation with maximum number of employees, organisation with minimum number of employees]
For example: `['weaver-barnett', 'bolton diaz and kent']`

ii) **Standard Deviation**: A list containing the standard deviation of median salary of organisations for the input `country` and the standard deviation of median salary for all the organisations in the dataset. Your output should be stored in a list in the following order:
[standard deviation for the organisations of the input country, standard deviation for all the organisations]
For example: `[300.5001, 40.0045]`

iii) **Ratio:** A numeric value that is the ratio of 'sum of profit increases (positive changes)' to 'sum of profit decreases (negative changes)', from 2020 to 2021.

For example:
Let's suppose that the changes in profit for different organisations in a country A from 2020-2021 are: `-21, 34, 245, -32`, then:

sum of profit increases= `34+245 = 279`
sum of profit decreases= `21+32 = 53`

ratio= `279/53` = `5.26`

iv) **Correlation:** A numeric value which is a correlation between median salaries and profits in 2021 for all the organisations in the input country which have an increase in profits from 2020 to 2021. The expected output is a single float value.
For example: `-0.1013`

All returned numeric outputs (both in lists and individual) must contain values rounded to four decimal places (if required to be rounded off). Do not round the values during calculations. Instead round them only at the time when you save them into the final output variables.

Examples:

Download `Organisations.csv` file from the folder of Project 1 on LMS or Moodle. Some example of how you can call your program from the Python shell (and examine the results it returns) are provided below:

```
>>> maxMin,stdv,ratio,corr=main("Organisations.csv", "Belgium")

The output variables returned are:

>>> maxMin
['weaver-barnett', 'bolton diaz and kent']

>>> stdv
[936.9468, 875.2285]

>>> ratio
2.2186

>>> corr
-0.1013
```

Assumptions:
Your program can assume the following:

- Anything that is meant to be string (e.g., header) will be a string, and anything that is meant to be numeric will be numeric in the CSV file.
- All string data in the CSV file is case-insensitive, which means “Asia” is same as “asia”. Your program needs to handle the situation to consider both to be the same. Similarly, your program needs to handle the input `country` in the same way.
- The order of columns in each row will follow the order of the headings provided in the first row. However rows can be in random order except the first row containing the headings.
- No data will be missing in the CSV file; however values can be zero and must be accounted for mathematical calculations.
- The `main` function will always be provided with valid input parameters.
- The necessary formulae are provided at the end of this document.

Important grading instruction:
Note that you have not been asked to write specific functions. The task has been left to you. However, it is essential that your program defines the top-level function `main(csvfile, country)` (hereafter referred to as “`main()`” in the project documents to save space when writing it. Note that when main() is written, it still implies that it is defined with its two input arguments). The idea is that within main(), the program calls the other functions. (Of course, these functions may then call further functions.) This is important because when your code is tested on Moodle, the testing program will call your main() function. So if you fail to define main(), the testing program will not be able to test your code and your submission will be graded zero. Don’t forget the submission guidelines provided at the start of this document.

Marking rubric:

Your program will be marked out of 30 (later scaled to be out of 15% of the final mark).

22 out of 30 marks will be awarded automatically based on how well your program completes a number of tests, reflecting normal use of the program, and also how the program handles various states including, but not limited to, different numbers of rows in the input file and / or any error or corner states/cases. You need to think creatively what your program may face. Your submission will be graded by data files other than the provided data file. Therefore, you need to be creative to look into corner or worst cases. I have provided few guidelines from ACS Accreditation manual at the end of the project sheet which will help you to understand the expectations.

8 out of 30 marks will be awarded on style (5/8) “the code is clear to read” and efficiency (3/8) “your program is well constructed and run efficiently”. For style, think about use of comments, sensible variable names, your name and student ID at the top of the program, etc. (Please watch the lectures where this discussed)

Style Rubric:

0	Gibberish, impossible to understand
1-2	Style is really poor or fair
3-4	Style is good or very good, with small lapses
5	Excellent style, really easy to read and follow

Your program will be traversing text files of various sizes (possibly including large csv files) so you need to minimise the number of times your program looks at the same data items.

Efficiency rubric:

0	Code too complicated to judge efficiency or wrong problem tackled
1	Very poor efficiency, additional loops, inappropriate use of readline()
2	Acceptable or good efficiency with some lapses
3	Excellent efficiency, should have no problem on large files, etc

Automated testing is being used so that all submitted programs are being tested the same way. Sometimes it happens that there is one mistake in the program that means that no tests are passed. If the marker can spot the cause and fix it readily, then they are allowed to do that and your - now fixed - program will score whatever it scores from the tests, minus 4 marks per intervention, because other students will not have had the benefit of marker intervention. Still, that's way better than getting zero. On the other hand, if the bug is hard to fix, the marker needs to move on to other submissions.

Extract from Australian Computing Society Accreditation manual 2019:
As per Seoul Accord section D, a complex computing problem will normally have some or all of the following criteria:

- involves wide-ranging or conflicting technical, computing, and other issues;
- has no obvious solution, and requires conceptual thinking and innovative analysis to formulate suitable abstract models;

- a solution requires the use of in-depth computing or domain knowledge and an analytical approach that is based on well-founded principles;
- involves infrequently-encountered issues;
- is outside problems encompassed by standards and standard practice for professional computing;
- involves diverse groups of stakeholders with widely varying needs;
- has significant consequences in a range of contexts;
- is a high-level problem possibly including many component parts or sub-problems;
- identification of a requirement or the cause of a problem is ill defined or unknown.

Necessary formulas:

i) **Correlation coefficient:**

Mathematical formula to calculate correlation is as follows:

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

Where x_i and y_i are the values of profits in 2021 and median salaries for the organisation in a country respectively. $\bar{x}$ is the mean of profits in 2021 and $\bar{y}$ is the mean of median salary.

ii) **Standard deviation:**

$$s = \sqrt{\frac{\sum_{i=1}^{N} (x_i - \overline{x})^2}{N-1}}$$

where $x_1, x_2, x_3 \ldots\ldots x_n$ are observed value in sample data. $\bar{x}$ is the mean value of observations and N is the number of sample observations.

iii) **Profit change:**

Profit change for an organisation = profit in 2021 – profit in 2020

iv) **Ratio:**

Ratio = (sum of positive changes in profits) / (sum of negative changes in profits)