DATA 301 - Introduction to Data Analytics
Lab 4: Introduction to Python

This lab can be done in pairs; however, pairs are not required. If working in a pair, only submit one file on CANVAS with a note of your partner in the comments.

Total marks: 20 marks + bonus 2 marks

In this lab, we will create some Python programs with variables, decisions, and loops.

Instructions

1. Create a Python program that processes a list of data to return the minimum, maximum, and average values in the list for all data values in a user-specified range. Details:
   a. (1 mark) Put a comment at the top of the Python file called **lab4q1** with your name and student number(s).
   b. The list of data is stored in the Python code itself (not read in from the user or a file).
   c. The data are numbers between 1 and 100 (perhaps representing sensor readings).
   d. Your code must work on any list of data in that range. Two samples:

```
data = list(range(1,101))
data = [1, 9, 13, 25, 42, 55, 59, 63, 68, 69, 70, 70, 70, 70, 72, 75,
85, 99, 90, 85, 44, 23, 55, 66, 77]
```

   e. (2 marks) Prompt the user for two numbers: a lower bound (inclusive) and an upper bound (inclusive). Convert the values to numbers using `int()`.
   f. (1 mark) Create variables to store maximum, minimum, sum, and count of data items.
   g. (1 mark) Use a for loop to examine each data item in the list.
   h. (2 marks) For each data item only process it if it is within the lower and upper bounds (inclusive).
   i. (2 marks) Compute the minimum, maximum, and average values (in the bounded range).
   j. (1 mark) Print out the list of data items, the minimum, maximum, and average values.

Sample Output 1:

Uses first data list `data = list(range(1,101))`

```
Enter lower bound: 10
Enter upper bound: 60
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21,
22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40,
41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59,
60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78,
79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97,
98, 99, 100]
```

```
Maximum: 60
Minimum: 10
Average: 35.0
```

Sample Output 2:

Uses second data list `data = [1, 9, 13, 25, ..., 55, 66, 77]`

```
Enter lower bound: 20
Enter upper bound: 90
[1, 9, 13, 25, 42, 55, 59, 63, 68, 69, 70, 70, 70, 70, 72, 75, 85, 99, 90,
85, 44, 23, 55, 66, 77]
Maximum: 90
Minimum: 23
Average: 63.476190476190474
```

2. Create a Python program that takes a badly formatted data set and converts it into CSV. Details:
   a. The data has three fields which we will assume are always present: a number, a name (that contains no spaces), and a salary. The data may have multiple spaces between the fields and all fields for the same row may be on different lines. Data set is here:

   ```
   2   Joe    95000 4 Steve
      35000    1  Samantha   150000     10 Leah   99000
    6   Riley   53215     7   Ashley   23424
    15    Sheyanne 225000   9  Dave  35235
   ```

   b. (½ mark) Put a comment at the top of the Python file called **lab4q2** with your name and student number(s).
   c. (½ mark) Put the data above into the Python code file using a triple quoted string.
   d. (½ mark) Use split to divide up the messy data where fields are separated by spaces.
   e. (1½ marks) Create individual lists for the ids, names, and salaries and print out each list.
   f. (1 mark) Output the data in CSV form: id,name,salary.
   g. (1 mark) Create a new list where each element is a list representing a row in the CSV file. Print this list.
   h. (2 marks) Create and print a new list which includes only rows where the id >= 4 and id <= 8, or the person has a salary > 50000.
   i. (1 mark) Output the original list in CSV form sorted by id ascending.
   j. (2 marks) Create and print a new list which for each row increases the salary by 50% if the length of the person's name is 3 characters or less or starts with 'S'.
   k. (1 mark (bonus)) Output the original list in CSV form sorted by salary descending.
   l. (1 mark (bonus)) Create a function to output in CSV form and use it multiple times rather than duplicating code.

   Note: If a string has a single quote (') in it, you must escape it by putting another single quote in front of it. Example: Joe's would be `'Joe''s'`.

   Sample Output

```
Ids: [,2,, ,4,, ,1,, ,10,, ,6,, ,7,, ,15,, ,9,]
Names: [,Joe,, ,Steve,, ,Samantha,, ,Leah,, ,Riley,, ,Ashley,, ,Sheyanne,,
,Dave,]
Salaries: [,95000,, ,35000,, ,150000,, ,99000,, ,53215,, ,23424,, ,225000,,
,35235,]

CSV Output:
id,name,salary
2,Joe,95000
4,Steve,35000
1,Samantha,150000
10,Leah,99000
6,Riley,53215
7,Ashley,23424
15,Sheyanne,225000
9,Dave,35235

List with each row a list:
[[2, ,Joe,, 95000], [4, ,Steve,, 35000], [1, ,Samantha,, 150000], [10,
,Leah,, 99000], [6, ,Riley,, 53215], [7, ,Ashley,, 23424], [15, ,Sheyanne,,
225000], [9, ,Dave,, 35235]]

Filter with only rows where id>=4 and id<=8 or salary>50000:

CSV Output:
id,name,salary
2,Joe,95000
4,Steve,35000
1,Samantha,150000
10,Leah,99000
6,Riley,53215
7,Ashley,23424
15,Sheyanne,225000

Data sorted ascending by id:

CSV Output:
id,name,salary
1,Samantha,150000
2,Joe,95000
4,Steve,35000
6,Riley,53215
7,Ashley,23424
9,Dave,35235
10,Leah,99000
15,Sheyanne,225000

Increase salary where the length of the name is <= 3 or starts with ,S,:

CSV Output:
```

```
id,name,salary
1,Samantha,225000.0
2,Joe,142500.0
4,Steve,52500.0
6,Riley,53215
7,Ashley,23424
9,Dave,35235
10,Leah,99000
15,Sheyanne,337500.0

Bonus: Sort descending by salary

CSV Output:
id,name,salary
15,Sheyanne,225000
1,Samantha,150000
10,Leah,99000
2,Joe,95000
6,Riley,53215
9,Dave,35235
4,Steve,35000
7,Ashley,23424
```

When complete, submit your two code files to Canvas:

- lab4q1 - Python code for question 1. May be a .txt, .py, or .ipynb file.
- lab4q2 - Python code for question 2. May be a .txt, .py, or .ipynb file.