SENSOR DATA READER

PURPOSE

In this exercise, string formatting and parsing is explored in Python. Sensor readings are read in from a text file, and the average is calculated per sensor and printed out in a formatted manner.

OBJECTIVES

After completing this exercise, you should be able to:

- Parse strings and extract relevant information
- Format strings to control the displayed output

PROCEDURE

PREPARE SUBMISSION FILE

1. Create a copy of the submission template called COMP6060**INIT**Lab8.docx where **INIT** is replaced with your own initials. So if your name is John Smith, the document will be called COMP6060**JS**Lab8.docx

PREPARE PYTHON FILE

1. Create a Python file called COMP6060**INIT**Lab8.py where **INIT** is replaced with your own initials. So if your name is John Smith, the document will be called COMP6060**JS**Lab8.py
2. Download the data.txt file from FOL under Week 9, and place in the same location as the Python file
3. In the Python file, print out the following to the console, replacing NAME with your name:
   Welcome to NAME's sensor data reader!

INSPECT DATA FILE

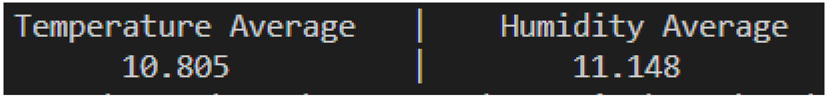
1. Open the data.txt file in a text editor
2. Notice that the data is split into 3 columns separated by two tab (`\t`) characters:
   timestamp		*sensor_name*		*sensor_reading*
   a. Timestamp in seconds as an integer value
   b. Sensor name
   c. Sensor reading as a float value

READ DATA FILE

1. In the Python file, create 4 variables as follows, and assign them all the value of 0:
    a. `tempSum`
    b. `tempNum`
    c. `humidSum`
    d. `humidNum`
2. Open the data.txt file in reading mode
3. Use a ranged-for loop to iterate over all the lines in the file
4. Inside the first ranged-for loop:
    a. Split the line into the 3 parts separated by 2 tabs. Store the resulting array in a new variable called entry
    This will result in the 3 parts of a single line in the file stored as string values in the entry array. Example: `[‘1672605261’, ‘Temp’, ‘5.2309832’]`
    b. Check if the entry is a temperate or humidity sensor reading:
        i. If the entry is for the temperature sensor:
            1. Increment the variable `tempNum`
            2. Cast the temperature reading value to a float, and add it to `tempSum`
        ii. Otherwise, if the entry is for the humidity sensor:
            1. Increment the variable `humidNum`
        iii. Cast the humidity reading value to a float, and add it to `humidSum`
        iv. If it’s neither, print out an error message with the following format: `sensor_name is not supported.`
5. After the for loop is done, close the file

CALCULATE SENSOR AVERAGES

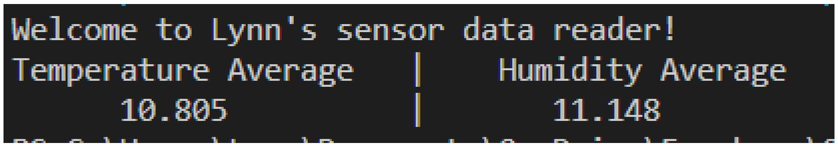
1. Calculate the average of the temperature sensor readings, and store the results in a new variable called `tempAvg`
2. Calculate the average of the humidity sensor readings, and store the results in a new variable called `humidAvg`
3. Print the averages in the following format:



```
Temperature Average  |     Humidity Average
      10.805         |         11.148
```

    a. Ensure that the float values only display 3 decimal values
    b. Ensure that the values have a column width of 12.

Expected Output



```
Welcome to Lynn's sensor data reader!
Temperature Average    |     Humidity Average
      10.805           |        11.148
```

Show results to Instructor.

Student Name: _______________________ Instructor: _______________________

Date: _______________________