OCN318 Homework 3

Assigned March 22, 2018, due April 2 **before** class.

Review: Reusing other Python scripts

Given a Python class BMP280 defined in a file named bmp280.py (https://github.com/stanleylio/fishie/blob/master/drivers/bmp280.py), one can access the class from another Python script by

```
from bmp280 import BMP280

sensor = BMP280()
```

Another example: the Python class EZO_EC is defined in a file named ezo_ec.py (https://github.com/stanleylio/fishie/blob/master/drivers/ezo_ec.py). One can access the EZO_EC class by

```
from ezo_ec import EZO_EC

whatever = EZO_EC()

# very sophisticated computation goes here
```

Due to popular demand, here are a few exercises to refresh your memory.

Homework Questions

Q1.

OCN318 is equipped with all five Atlas Scientific sensors (on loan via MESH Lab). We used the EZO EC (Electrical Conductivity) sensor in the previous class.

Now say you want to use the Atlas Scientific EZO DO (Dissolved Oxygen) sensor with the Python drivers in the "node" (https://github.com/stanleylio/fishie) repository. Fill in the blanks (assuming you have a local copy of the repository in /home/pi/node):

```
import sys

sys.path.append('/home/pi')

from node.______________ import EZO_DO
```

Q2.

Continue from the previous question. Given the following code, complete the following code to print one reading from the EZO sensor to screen:

```
do = EZO_DO()

print(____________________)
```

Hint: https://github.com/stanleylio/fishie/blob/master/drivers/ezo_do.py

Q3.

Now let's say you'd like to collect 10 samples from the EZO DO sensors for later use (e.g. write to CSV, calculate average, plot...). Complete the following code snippet by filling in the blank:

```
# create an empty list to store the samples:

L = []

# get a reading from the sensor and append it to the list, x10

for i in range(10):

    L.________________

# an example of how to use the 10 readings in L (this calculates the sample mean):

print(sum(L)/len(L))
```

Hint: `jabuticaba3.py`

Q4. (not really Python, but stop me)

The `read()` function in the driver for EZO_DO returns readings in unit of `mg/L`. Given a reading of `X` `mg/L`, show how to convert it to micromolar (µM). (No code required, though a Python script that converts `mg/L` to µM takes only two lines or less.)

Q5.

What is the **unit** of the readings from the Atlas Scientific EZO EC sensor?

Q6.

What is the name of driver file for the BME280 Temperature, Pressure and Humidity sensor in the course's repository?

https://github.com/stanleylio/fishie

Say you want to use the "BME280" class in the aforementioned driver file. Complete the following line:

```
from node.______________________ import BME280
```

What is the name of the driver file for the TSYS01 Temperature sensor? Complete the following line:

```
from node.______________________ import TSYS01
```

Q7. (optional, for those who want to prepare for the next class / quiz):

The official website for the matplotlib plotting library has an example of plotting error bars:

https://matplotlib.org/1.2.1/examples/pylab_examples/errorbar_demo.html

Collect 100 conductivity readings for each of the six water samples we used in class. First, plot the 100 samples as time series (total of six plots). What is the standard deviation of each of the six sets of readings?

In another graph, plot the sample means of the conductivity readings with error bars (so six points with six error bars; x axis are the samples' names, y axis are conductivity/salinity). Is your largest error bar smaller than the smallest difference between the six samples? If not, how many more readings do you need to be able to resolve the difference?

Final Remarks

No new material is required to complete this homework. As usual, all answers can be found in the course's examples and in the course's repositories if you know where to look.

Python or not, you will spend a lot of more time reading someone else's code than writing your own. Often it is a piece of code that you wrote three weeks ago and you have forgotten what it does. We started reusing code from the week with the thermistor. We will continue to integrate others' code for the rest of the course so it's important to know how to quickly determine the purpose and intention of existing code.