Assignment #1

This assignment is due on Wednesday, October 4th 23:59:59 class via email to christian.wallraven+AMF2023@gmail.com.

Important: You need to name your file properly. If you do not adhere to this naming convention, I may not be able to properly grade you!!!

If you are done with the assignment, make one zip-file of the assignment1 directory and call this zip-file `STUDENTID_A1.zip` (e.g.: `2016010001_A1.zip`). The correctness of the IDs matters! **Please double-check that the name of the file is correct!!**

Also: Please make sure to comment all code, so that I can understand what it does. Uncommented code will reduce your points!

Finally: please read the assignment text carefully and make sure to implement EVERYTHING that is written here – if you forget to address something I wrote, this will also reduce your points! Precision is key ☺!

Part1 File handling (30 points):

Make a python file called called `<YOURLASTNAME_YOURFIRSTNAME.py>` (e.g.: `HONG_GILDONG.py`). The first lines of the file should contain a comment that explains what the file does in detail (see below).

In your python main function, create a dictionary variable called `myInfo`. Assign the following pieces of information to this variable:

your firstname (a string)

your lastname (a string)

your birth-year (a number)

your birth-month (a number)

your birth-day (a number)

The variable `myInfo` should now have five entries, two strings and three numbers.

Add commands to the script that calculates the number of seconds that you have been alive and assign this value to a sixth entry in myInfo. Assume (even if you know the exact time on which you are born) that you were born at midnight (12 AM) on your birthday. Assume that there are no leap years!

To calculate your age in seconds, you are ONLY allowed to use the `time.time()` function from python and simple math. Note, the weird time that `time.time()` is based on (it's a UNIX thing ^^)

```
# import the time module

import time
```

```
# get the current time in seconds since January 1st, 1970 at
UTC.

seconds = time.time()
```

The script should calculate your age in seconds every time you execute it, so do not put in fixed numbers, but use the output of `time.time()`!

Insert a command that displays your age in seconds in a nice format in the command window using print (such as: "At the time of calling this script, I was NNNN seconds old.").

The last commands in the script now will need to save the contents of the variable myInfo into a text file called `<LASTNAME_FIRSTNAME_myInfo.txt>` (e.g.: `HONG_GILDONG_myInfo.txt`). The text file should get the following structure:

firstname

lastname

birth-year / birth-month / birth-day

age in seconds

Note that the file has 4 lines and that the third line also contains the character "/".

Use only the following commands that demonstrate file writing for a simple case:

```
f = open("demofile.txt", "w")

f.write("Hello")

f.close()
```

Part2 Functions, statistics, plotting (30 points):

In this assignment you will use a script and a function to calculate statistics of some example data. We have the heart rate, weight and amount of exercise in hours per week of 15 participants. This data is given in the following table.

Participant	Heart rate(bpm)	Weight (kg)	Exercise (hrs)
1	72	61	3.2
2	82	91	3.5
3	69	71	7.1
4	82	67	2.4

5	75	77	1.2
6	56	80	8.5
7	93	101	0.1
8	81	75	3
9	75	55	2.7
10	59	65	3.1
11	95	79	1.5
12	66	62	6.3
13	80	75	2
14	65	81	6.5
15	69	57	4.8

Make a new script called `exercise.py` and add the data as a numpy array called `data` to the script. It is not necessary to add the participant column to the array.

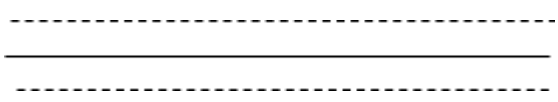
Add a function to the script that will calculate two values from the data: mean and standard deviation (STD). Those should be clear to everyone, I hope. The function should be called `meanSTD` and accept one input variable and return two output variables (the mean and the standard deviation). Make sure to comment the function properly. Use `numpy` commands to calculate the mean and standard deviations as arrays (because there will need to be three values each!).

Write a function call to `meanSTD` into the main part of your script.

Next, with this result, make a figure with three subplots (that is a single plot which contains three subplots) using matplotlib.

Each subplot will plot one original `data` variable (remember, we have three variables, heart rate, weight, and exercise – please also make sure to provide proper axis labels, plot titles, tick marks, and axis limits!) for all 15 participants. That's 15 data points for each plot (Hint: think about how to plot this information properly!).

Add the mean of each variable to each of the plots as a line. Then plot the STD around the mean of each data subset into the subplot as two dashed lines (one above the mean, one below the mean) like so:



Insert a command that saves the resulting figure as `analysis.png`.

Next, insert commands to create a SECOND figure with three subplots. In the first subplot, plot the heart rate against the weight. In the second subplot, plot the heart rate against the exercise rate. In the third subplot, plot the weight against the exercise rate. Note that these are scatter-plots, so use the command `scatter`! Again make sure to add proper titles, labels, tick marks etc!

Save this figure as `analysis2.png`.

Finally, insert detailed comments about the two plots you have created into the script. Importantly, I want to know what is the resulting (cautious) interpretation of the data?