APS106 - LAB # 6

This lab will test your ability to read in data from a file and then respond to a query about that data. ***Due: 11:59pm, Friday, Mar. 8, 2019***

Note: If you need help with file I/O and csv files, read the second page first.

Question: Write a Python function, `compound_properties()`, that is passed a file name of a CSV file (assume the file is in the same folder as your Python file) and a compound's formula unit structure**. The file will contain metal elements and their properties. In the function, you are tasked to read the compound formulation (by the `molform()` function described below), find out properties of those constituent in the compound in the CSV file, and return a tuple of:

1. The element in the compound with the lowest boiling point (full name, like 'Hydrogen' for 'H')
2. The average boiling point of all elements, round to the nearest integer
3. The molecular mass of that compound (the atomic mass of each element, multiplied by the number of atoms in the molecule), round to the nearest integer

For example,

```
>>> compound_properties ('input.csv', 'Fe1S1O4')   # FeSO4
('Oxygen', 1314, 152)
```

Write a second function, `molform()`, that will be called by the first function, to take the compound's formula and return it as a dictionary with the elements as the keys and the number of atoms as the values. Note that, you can assume element names won't exceed two letters in length, but the number of atoms might be huge, for example, $C_{132983}H_{211861}N_{36149}O_{40883}S_{693}$ (in case you wonder, this thing does exist, it's called a 'Titin', a very important protein in muscle fibers. Pronouncing the full name of it will take you a good four hours' time. See here for the full name).

You may assume the compound structure will specify the number of atoms, even if the IUPAC formula would omit the subscript 1. For instance, ethanol, C_2H_6O, would be passed to the program as the string "`C2H6O1`".

Your function must read files with the format of `atoms.csv`. Each new column of data is separated by a comma (CSV). Don't be scared off by this long file, every line has the same format. You can test your function using a subset of the file `atoms.csv` (create the subset by removing many lines from it). Testcases to understand the inputs and outputs of each function can be found on the next page.

TO DO:

Download the file `lab6.py`, complete the functions inside according to their descriptions and upload your version of `lab6.py` to MarkUs.

Do not change the file name or function names. Do not use input() or print(). Your file should not contain any additional lines of code outside the function definitions. Some test cases are included in the docstrings.

**Note: If you do not know how to read chemical compounds, watch this short video: https://www.youtube.com/watch?v=mvGOOyqwdhc

Help with files/CSVs

Reading files

Reading files is made very easy in Python. There are only 3 steps to it: open file, read it and close the file.
You can use the following code to read a file.

```
f = open(path_to_file, mode='r')

lines = f.readlines()

f.close()
```

Now, `lines` is a list of all the lines in the file you just read, with each element corresponding to one line in the file.

Reading CSV files

CSV files are just regular text files. You can open then in Microsoft Word or equivalent spreadsheet processor to see what it is like inside. Basically, every line in the CSV is a row in the table. Columns of the table are separated by comma, thus the name Comma Separated Values, aka, CSV.

You can read the csv as a normal file and split by comma to get the values on your own, or you can use python's csv module to save some work, although it won't make much difference regarding the CSV you are reading in this lab.

For the usage of python's csv module, check here.

Sample Input and Output:

```
>>>isinstance(molform("C2H6O1"), dict)

True

>>>isinstance(molform("C2H6"), dict)

True

>>>isinstance(molform("C2"), dict)

True

>>>molform("C2H6O1")

{'C': 2, 'H': 6, 'O': 1}

>>>molform("C1H4")

{'C': 1, 'H': 4}

>>>isinstance(compound_properties("atoms.csv", "K1Fe4"), tuple)

True

>>>isinstance(compound_properties("atoms.csv", "K1Fe4")[0], str)

True

>>>isinstance(compound_properties("atoms.csv", "K1Fe4")[1], int)

True

>>>compound_properties("atoms.csv", "K1Cl1O3")

('Oxygen', 454, 123)

>>>compound_properties ("atoms.csv", "C6H12O6")

('Hydrogen', 1470, 180)
```