Homework 3

Due: November 13, 2019 **Points:** 100

1. (*100 points*) This problem asks you to calculate the atomic weight of molecules.
   The file "atomic_weights.txt" contains lines with three fields separated by tabs. The first field is the atomic weight, the second field is the symbol for the element, and the third field is the name of the element (which you can ignore for this problem).
   We will proceed in stages, to make life easier. Remember, if this seems overwhelming, take some deep breaths and do something relaxing — take a walk, read, work on your hobby, . . .

   (a) Write a function to load the contents of the file into a dictionary. The key is to be the chemical symbol. The value is to be the atomic weight. As noted above, you can ignore the third field.

   (b) Now write a function that takes a chemical compound and breaks it into elements and numbers. The basic unit of a chemical formula is an element's symbol followed by a number (1 or more digits); the chemical compound's formula is a sequence of one or more units. For example, the chemical formula for ethanol, `C2H5OH`, is 2 C (carbon) atoms, 5 H (hydrogen) atoms, an oxygen atom, and another hydrogen atom; and the chemical formula for water, `H2O`, is 2 H (hydrogen) atoms and an oxygen atom.

   A good way to check your program is to have it print out each atom's symbol and the number that follows it, if any.

   Hint: Element symbols are either 1 or 2 letters. The first letter is *always* capitalized; if there is a second letter, it is *always* lower case. So "HO" is a hydrogen atom (H) and an oxygen atom (O), and "Ho" is the symbol for holmium. Similarly, "SN" is a sulfur atom (S) and a nitrogen atom (N), and "Sn" is the symbol for tin. Similarly, if no number follows an element's name, treat it as 1.

   (c) Using the functions you wrote in the above two parts, write a program that reads in a chemical compound and prints its atomic weight. Your program is to continue reading input until the user types an end of file.

   Your output is to look like this (input is in red).

```
Chemical composition? C2H5OH↵
The atomic weight of C2H5OH is 46.08
Chemical composition? H2O↵
The atomic weight of H2O is 18.02
Chemical composition? HO↵
The atomic weight of HO is 17.01
Chemical composition? Ho↵
The atomic weight of Ho is 164.93
Chemical composition? SN3↵
The atomic weight of SN3 is 74.1
Chemical composition? Sn3↵
The atomic weight of Sn3 is 356.13
Chemical composition? control-D
```

   To turn in: Please call your program *chem.py* and submit it to Canvas

Extra Credit

E1. (*50 points*) The *birthday problem* asks how many people must be in a room so that the probability of two of them having the same birthday is 0.5. This problem has you explore it by simulation. Basically, you will create a series of lists of random numbers of length $n = 2, \ldots$, and look for duplicates. You will do this 5000 times for each length. For each length, count the number of lists with at least 1 duplicate number; then divide that number by 5000. That is the (simulated) probability that a list of n generated numbers has at least one duplicate. As the random numbers you generate are between 1 and 365 (each one corresponding to a day of the year), this simulates the birthday problem.
   Now, breathe deeply and calm down. We will do this in steps!
   (a) First, detecting duplicates. Write a function called `hasduplicates(l)` that takes a list `l` and returns `True` if it contains a duplicate element, and `False` if it does not. For example:

```
>>> hasduplicates([1, 2, 3, 4, 5, 5, 2])␣
True
>>> hasduplicates([1, 2, 3, 4, 5, 6, 7])␣
False
```

(b) Now, deal with one set of birthdays. Write a function called `onetest(count)` that generates a list of *count* random integers between 1 and 365 inclusive, and returns `True` if it contains a duplicate element, and `False` if it does not. Please use the function `hasduplicates(l)` to test for duplicates.
Hint: To generate a random number between *a* and *b* inclusive, put

```
import random
```

at the top of the program, and then call the function `random.randint(a, b)`.

(c) Now for the probability for *count* people. Write a function `probab(count, num)` that runs *num* tests of *count* people, and counts the number of tests with duplicates. It returns the fraction of the tests with duplicates; that is, the number of duplicates divided by *num*.

(d) Now for the demonstration. Start with 2 people, and begin adding people until the probability of that many people having two people with a birthday in common is over 0.9. (In other words, start with a list of 2 elements, and increase the number of elements in the list until the simulation shows a probability of 0.9 that a number in the list is duplicated.) Print each probability; your output should look like this:

```
For  2 people, the probability of 2 birthdays is 0.00220
For  3 people, the probability of 2 birthdays is 0.00880
For  4 people, the probability of 2 birthdays is 0.01680
For  5 people, the probability of 2 birthdays is 0.02940
For  6 people, the probability of 2 birthdays is 0.03940
For  7 people, the probability of 2 birthdays is 0.05900
For  8 people, the probability of 2 birthdays is 0.06840
For  9 people, the probability of 2 birthdays is 0.09700
For 10 people, the probability of 2 birthdays is 0.12360
```

How many people are needed so that the probability of two of them with a birthday in common is over 0.9? How many are needed such that the probability of two of them having the same birthday is at least 0.5? Put these answers into a comment at the head of the file.
Hint: Don't be surprised if your probabilities are slightly different than the ones shown in the sample output. As randomness is involved, it is very unlikely your numbers will match the ones shown here.

To turn in: Please call your program *bday.py* and submit it to Canvas