Class 1: HW
Assigned: 2020/04/09
Due: Please complete by the beginning of Class 2 (2020/04/14)
**The primary purpose of this homework is to identify areas that you don't feel totally comfortable operating yet in Python! While it's optional, we strongly recommend at least skimming it to make sure that you feel you could complete each of the problems without much trouble.*

1.) Create two lists:

```
a = [1,3,5,7,9,11,13,15]
b =  [2,4,6,8,10,12,14,16]
```

a.) Create a list "c" that contains all of the elements from both a and b (many ways to do this) in any order.

b.) Using slicing techniques, create a list "d" that contains all of the elements from a and b in numerical order

-HINT 1: You'll need to create an empty list "c" before you can assign elements to it.
-HINT 2: If you're going to assign values to this list using slicing, you'll want it to be full of the same number of [None] values as you ultimately expect to fill it with. Try `c = [None] * _____` *<- length you'll want to the list to eventually be. Is there a way to find this value using len( ) rather than "hard-coding" it?*

c.) Using a **for** loop, create a list "e" that contains all of the elements of a and b in numerical order

-HINT 1: You'll need to create an empty list "c" before you can assign elements to it!
-HINT 2: Are there any list-associated methods that would make this easier?

d.) Reassign "e" in a new window by writing a **list comprehension** that performs the same task as your for loop

e.) Using a **while** loop, create a list "f" that contains all of the elements of a and b in order

2.) Create a dict filled with 5 arbitrary username (containing any numbers/**lowercase** letters you wish)/ passwords as key/value pairs.

a.) Write a **well-documented** function called passwordEval that takes in a username and password as input, prints "Username and password accepted!" if the username/password combo are in the dict, and prints "Username/password not found!" if they are not.

-HINT 1: When we say well-documented, we refer to both creating a docstring for the function AND commenting on your code. Comments in code can be created by typing "#" in front of whatever you don't want evaluated – just keep in mind that everything following the "#" on the same line will also be commented out

b.) We specified that we only want users to have passwords with lowercase letters. How can we always make sure that users are entering only lowercase? One method is to **assert**. We didn't learn this in class, but it's pretty straightforward. The format is

```
assert (CONDITION), 'STRING TO PRINT IF FALSE'
```

Using the str.islower() method, add an assertion that usernames must be lowercase into your function. Now test your function to see what happens if you accidentally type a username containing a capital letter!

c.) In c, we used an assertion to make sure users enter correct input. Is there a way to do this that takes the burden off the user? Remove your assertion, and instead use the str.lower( ) method to correct for this sort of error. Test it out.

d.) Finally, using your text editor, open the "week1.py" file we created in class, and copy/paste in your well-documented, tested "passwordEval" function. Save, and restart your notebook's kernel. Can you import it and use it? What does your docstring look like when you type passwordEval?

e.) Create a new dictionary that contains **all** of the username/password pairs you created in your first dict (is there a Pythonic way to do this instead of retyping it all?), plus 4 new pairs. Write a for loop that runs your passwordEval function on the new dictionary!

-HINT1: How do we iterate through a dictionary? Don't forget about dict.keys()!