**Here’s some practice as a Ch 4 Review (Use anything that you’ve learned in Ch 5 though.)**

***Though I do not state it in each problem, you should call your function in a variety of ways to test it.***

1. Write a function called getCorrectInput1. The function should ask the user for a number between 1 and 10 over using input validation. The function should return user input in the correct range.

**For #2, I am testing to see if you remember how to deal with default parameters.**

1. Write a function called getCorrectInputV2 that takes in a starting point and ending point. The function should ask the user for a number between this starting point and ending point and should perform input validation on the user's input. The function should return user input in the correct range, that is, between the starting and ending point. The function should default to a starting point of 1 and an ending point of 10 if no parameters are passed. (HINT: COPY AND PASTE getCorrectInput1 and update it as needed.)

**For #3, I am testing to see if you remember that every function returns something, even if that value is “None.”**

1. Write a function typeOfMember that takes in a total amount of purchases that a customer has made at a store. The function should return:

"GOLD" if the person has spent at least $1000

"SILVER" if the person has spent $500-$1000

"STANDARD" if the person has spent $250-$500

Nothing otherwise.

Test the nothing case as such: If a customer is deemed to not have a membership, the customer is told how much more they have to spend to get a standard membership. Do this by making use what is returned in the final case above. Remember: Every function will return a value even if you don’t ask it too. What is that value?

**For #4/5, I am testing to see if you know when/how to use “\*” to pass values into functions and to create functions that can handle an arbitrary list of elements. It’s also a nice Ch. 5 Review because you are manipulating lists.**

1. Write a function tallyWordsArb that accepts an arbitrary number of words and prints a table tallying how many times each word occurs. Test your code with these input

TEST 1 (call your function with an arbitrary listing):

turkey, cranberry, stuffing, stuffing, pie, turkey, turkey, turkey, gobble

TEST 2 (call your function an actual list object):

[turkey, cranberry, stuffing, stuffing, pie, turkey, turkey, turkey, gobble]

1. Write a function tallyWordsList that accepts a SINGLE list of words and prints a table tallying how many times each word occurs. (HINT: Start with a copy and paste of version 1) Test your code with the input of these words: turkey, cranberry, stuffing, stuffing, pie, turkey, turkey, turkey, gobble.

TEST 1: Test your code with an arbirtray list object:

[turkey, cranberry, stuffing, stuffing, pie, turkey, turkey, turkey, gobble]

TEST 2: Test your code by asking the user to enter a new word or -1 when done entering words.

**For #6, I am testing to see if you can handle arbitrary inputs, if you know some helpful list functions, and if you can think logically through a mathematical algorithm.**

1. Write a function isFib that determines whether a number is in the Fibonnaci sequence, and if it is, what it's position is (or -1 if it is not in the sequence). The Fibonacci sequence is 1, 1, 2, 3, 5, 8, 13, 21, 34, etc.

Example:

isFib(3) returns 4.

isFib(21) returns 8.

isFib(9) returns -1.

1. Write a function isFibSubSequence that determines whether an arbitrary sequence of integers is in the Fibonnaci sequence (1, 1, 2, 3, 5, 8, 13, 21, 34, etc):

Example:

isFibSequence(3, 5, 8) returns true.

isFibSequence(3, 5, 8, 16) returns false.

isFibSequence(9, 11, 20, 31) returns false as well.

HINT: Use isFib to help. (Challenge for the more experienced: Try to save processing power by only calling isFib on the first two numbers.)