# CSC121 Lab 05: Functions Part 2; Lists Part 1

## Goals

In this lab assignment, students will demonstrate the ability to:

* Send and receive multiple return values in functions
* Create and use functions in modules
* Create a list
* Access list elements
* Alter a list
* Iterate over a list
* Do list copying, concatenation, and slicing
* Use operators, list methods, and built-in functions on lists

NOTE: This lab document is a copyrighted work of Wake Tech Community College and the course instructor. Any posting of this document outside of Wake Tech is considered a copyright violation. Students who post these documents outside of Wake Tech are subject to academic and possible legal actions.

## Instructions

In this lab, you will demonstrate your understanding of value-returning functions. You will also demonstrate your understanding what we've learned about lists so far.

Follow the instructions in each problem and submit the specified files. All problems will require that Python code be submitted as well as screenshots that prove the programs have been executed in PyCharm.

Problems 1 includes code provided by the instructor that needs to be completed. Problem 2 consists of a program that you create from scratch that meets the problem specification.

The **main** function: As with Lesson 4 as well as future lessons, all programs that you write from scratch must have a main function that contains the top-level logic of your code. Otherwise, you should follow the instructions provided in the lab document.

## Problems

### Problem 1

Trish at Bargain Swap Shop continues to use the programs you developed for her. On the previous program you did for Lab 3, Trish wants to update the program so that it has a nicer receipt. She also wants to include a 5% discount for all customers who buy more than a certain amount of a particular item.

You expect that Trish will continue to ask you to extend the types of items that she currently sells. So, you've decided to also update the program you did for Lab 03 to use functions.

* The instructor has provided two files called **Lab05P1.py** and **itemProcessor-FillThisIn.py**. Download those files and rename the second file **itemProcessor.py**.
* Copy those files into your PyCharm project.
* Change the program headers to include your name and the date.

The partial program is made up of 4 functions in two modules (files):

**main** in **Lab05P1.py –** CHANGE ONLY THE NAME AND DATE IN THIS FILE.

This function is already complete. It imports the helper module and calls the other 3 functions to accomplish its work:

* The main function calls **get\_item\_count** and then **get\_item\_total** 3 times to get the item costs for books, DVDs, and games.
  + Notice that the main program passes in the parameters that determine the item unit price, maximum number of items allowed, and the threshold for the discount.
* Then the main function calls **calc\_and\_display\_receipt** to print the receipt.

**get\_item\_count** in**itemProcessor.py**

This function will need to be implemented.

* This function takes three parameters:
  + item\_name – name of the item being prompted about
  + max\_allowed – maximum number of items the user can enter
  + discount\_threshold – minimum number of items eligible for a 5% discount
* The function will ask the user to enter the number of items using a prompt which contains the item name passed in as a parameter.
* If the user enters a value less than 0 or more than the max allowed, an error message will be printed, and they will be asked to enter a different value. See the sample output for an example.
* The function will return the number of items the user enters.

**get\_item\_total** in **itemProcessor.py**

This function will need to be implemented.

* This function takes three parameters:
  + num\_items – number of items
  + unit\_price – the cost of each item
  + discount\_threshold – minimum number of items eligible for a 5% discount
* The function calculates the total cost for the items. If the number of items is greater than or equal to the discount threshold, the total will be discounted by 5%.
* The function returns the total cost of all the items including the discount if applicable.

**calc\_and\_display\_receipt** in **itemProcessor.py**

This function will need to be implemented.

* This function takes three parameters:
  + books\_total – total cost of books
  + dvds\_total – total cost of DVDs
  + games\_total – total cost of games
* The function will calculate the total before tax.
* The function will calculate the sales tax using the function you created. Use the provided global constant for the sales tax.
* The function will calculate the total after tax.
* Then the function will print a receipt that contains the following information:
  + Total cost of books (**if zero**, skip this step)
  + Total cost of DVDs (**if zero**, skip this step)
  + Total cost of games **(if zero**, skip this step)
  + A divider line
  + Subtotal before tax
  + Amount of tax to be charged
  + Total including tax

Here's some sample outputs of running the program:

Enter the number of books.

5 or more receive a discount: 8

Enter the number of DVDs.

8 or more receive a discount: 3

Enter the number of games.

4 or more receive a discount: 4

Books: $17.10

DVDs: $13.05

Games: $19.00

---------------------

Subtotal: 49.15

Tax: 3.19

Amount due: 52.34

Enter the number of books.

5 or more receive a discount: -5

Number of items must be between 0 and 40

Enter the number of books: 3

Enter the number of DVDs.

8 or more receive a discount: 0

Enter the number of games.

4 or more receive a discount: -2

Number of items must be between 0 and 25

Enter the number of games: 40

Number of items must be between 0 and 25

Enter the number of games: 4

Books: $6.75

Games: $19.00

---------------------

Subtotal: 25.75

Tax: 1.67

Amount due: 27.42

NOTE: We are **not** handling issues where a user types an invalid integer, that is, an entry which cannot be converted using the int() function. We will handle that in a future lab.

Run this program. Take a screenshot of the results. Name the screenshot **Lab05P1-ouput.jpg**.

Submit all 3 files, **Lab05P1.py**, **itemProcessor.py**, and **Lab05P1-output.jpg**, to Blackboard for credit.

### Problem 2

In this problem, you will create a program from scratch based to demonstrate your abilities to create and manipulate lists in Python.

Create a file named **Lab05P2.py**.

For this program:

* **Label each block of code** that performs the different steps with an appropriate comment (e.g. # Step a).
* Your output should also include indications as to what step is being displayed. See the Sample Output for an example.
* Remember to put your program in a main function and call main at the end of your file.

Write a Python program that performs the following steps:

1. Ask the user how many numbers they want in the lists. This number will determine the size of BOTH lists.
2. Ask the user for the lower and upper bound for generating random numbers.
3. Use a **for loop** and a random integer generator to generate random integers from lower to upper bounds the user specified in step B. This is an inclusive range. The program should generate the number of integers indicated by the user in step A. Store the random integers in a list. Display the list.
4. Create a second list in the same way as was done in step B. Display the second list.

* NOTE: You could create a function that generates a list like what is specified in steps C and D. To do that, pass in the numbers from steps A and B as parameters, and then return the list that is generated.

1. Use a **for loop** to display the elements in the two lists in pairs, i.e., display the first elements in both lists, display the second elements in both lists, etc.

* HINT: Have your for loop increment through the lists **using an index**. That will allow you to access the 0th elements, then the 1st elements, etc. See the Explainer at the end of this lab document for more information.

1. Combine the lists into a single list using concatenation and display the combined list.
2. Sort the combined list **in place** and display the list. Do NOT use the sorted function. Use the list method we learned about in class.
3. Use list slicing to display the first three elements in the sorted combined list and the last three elements in that list.
4. Display the total of the elements in the list. Use a single function to calculate that total.
5. Display the minimum element in the list. Use a single function to determine that element.
6. Display the maximum element in the list. Use a single function to determine that element.
7. Use a **for loop** and a random integer generator to generate 5 random integers in the range specified by the user in step a. Check if each number is in the combined list.

* If the number is in the combined list, print the number and print the index of the first element that matches.
  + NOTE: There is a list method for finding an element value which return the index and you should use that function. Do NOT use another for loop or you will lose some points.
* If the element is found, after printing the index, the element should be deleted or removed from the list.
  + NOTE: There is both a function and a list method that can delete or remove an element from a list. You should use either the function or the list method.
* If the number is not in the combined list, print the number followed by "not found in list".

1. Print the combined list now that elements may have been deleted by step L.

Sample output:

Step a: How many numbers in each list? 5

Step b:

What is the lower bound for the random number? 1

What is the upper bound for the random numbers? 15

Step c: First list: [13, 8, 5, 14, 13]

Step d: Second list: [6, 9, 15, 5, 3]

Step e:

13 6

8 9

5 15

14 5

13 3

Step f: Combined list: [13, 8, 5, 14, 13, 6, 9, 15, 5, 3]

Step g: Sorted list: [3, 5, 5, 6, 8, 9, 13, 13, 14, 15]

Step h:

First three elements: [3, 5, 5]

Last three elements: [13, 14, 15]

Step i: Sum: 91

Step j: Minimum: 3

Step k: Maximum: 15

Step l:

8 at index 4

3 at index 0

11 not found in list

14 at index 6

7 not found in list

Step m: Final list: [5, 5, 6, 9, 13, 13, 15]

Run this program and take a screenshot of the program run with the results. Name the screenshot **Lab05P2-ouput.jpg**.

Submit both files, **Lab05P2.py** and **Lab05P2-output.jpg**, to Blackboard for credit.

## Grading Rubric

### Grading rubric for Problem 1 (45 points)

* Program has a well-formatted and correct header [5 points]
* Program does execute correctly and produces correct results [35 points]
* Screenshots demonstrates student executed the faulty program and the corrected program [5 points]

### Grading rubric for Problem 2 (55 points)

* Program has a well-formatted and correct header [5 points]
* Program does execute correctly and produces correct results [45 points]
* Screenshot demonstrates student executed the program [5 points]

## Problem 2 Explainer: Understanding Indexes in Lists

In Python, each item in a list has an assigned index value. It is important to note that Python is a zero-indexed based language. This means that the first item in a list has an index of 0.

For example, consider the following list:

my\_list = ['apple', 'banana', 'cherry', 'date']

The item 'apple' has an index of 0, 'banana' has an index of 1, 'cherry' has an index of 2, and 'date' has an index of 3.

### Generating Indexes in a Loop

When you loop over a list using a for loop in Python, you often do it like this:

for fruit in my\_list:

print(fruit)

In this case, fruit is the actual item from the list, not its index. However, sometimes you need to know the index of the current item. This can be achieved using the range() and len() functions, like so:

for i in range(len(my\_list)):

print("The item at index", i, "is", my\_list[i])

Here, i is the index, which we get from the range() function. len(my\_list) gives us the total number of items in the list. So, range(len(my\_list)) generates a sequence of numbers from 0 up to (but not including) the number of items in the list. In each iteration of the loop, i takes on the next value in this sequence.