George Kim

February 14, 2019

Foundations of Programming: Python

Assignment 06

GitHub Link: <https://github.com/gjkim44/IntroToProg-Python>

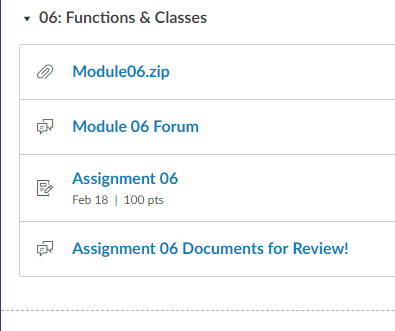
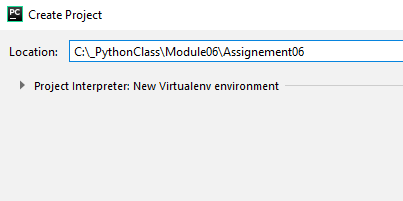
Function and Classes

# Introduction

Coming off one of the more challenging assignments (Assignment 5), we are asked to build off of Professor Roots version of Assignment 5 and use Function and Classes for this next project. The challenging piece of this assignment will be using someone else’s code and assigning functions and classes to them. I will be going over the steps of how I accomplished this assignment with the use of screen shots and descriptions of my process.

**Starting the Script**

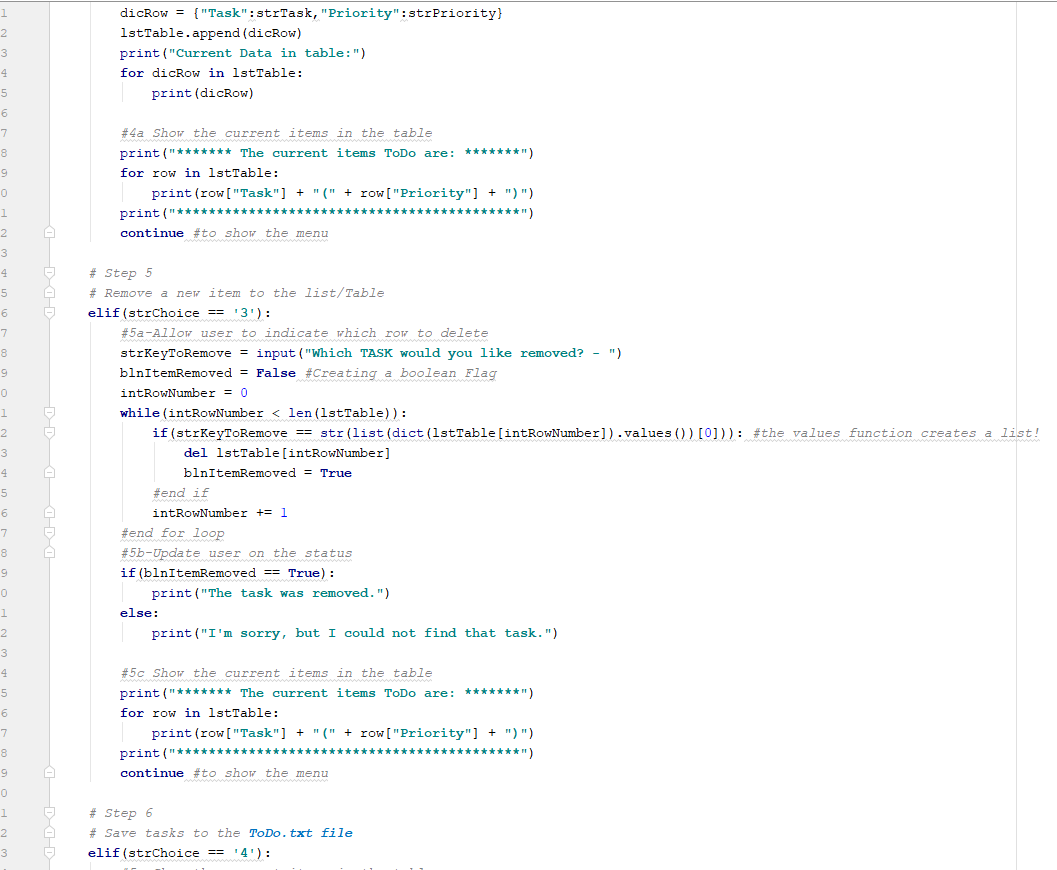
We start this assignment like the others in the previous assingments. Open up PyCharm and start a new project called Module 6 and a sub folder called Assignment 6 (**Figure 1**). We then go to module 6 on the canvas page and download the zip file (**Figure 1a**) and save that file in the folder Assignment06.

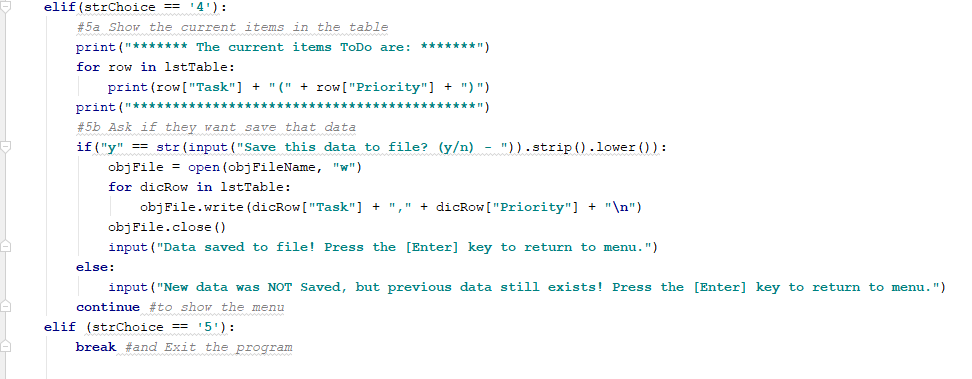


***Figure 1, 1a. Creating a project and downloading zip file.***

We open the contents and use Professor Roots Assignment05\_Answer.py file. Here is what his code looks like:

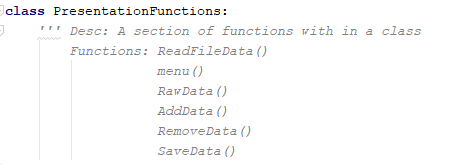






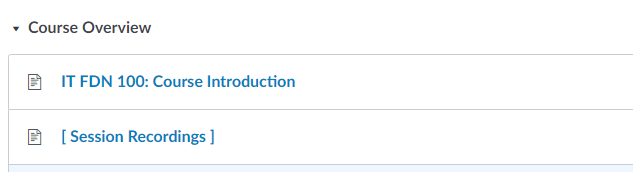
**Classes and Functions**

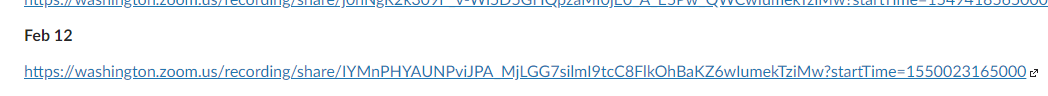
The assignment is to assign a Class and Functions to the lines of code before the conditionals and in them( *if-elif statements*). Professor Roots \_Mod06PythonProgrammingNotes, states for Class and Methods(Functions), “*Classes are a way of grouping functions, variables, and constants*.” It is actually pretty simple to assign a class. You put *class* with a *name* for the class followed by “:”



For each class and function we are to give a header or *docstring* , to describe what the class and functions are about. I went ahead a gave a full description for each one, as he just asked for a basic *docstring* for this assignment, just for practice. As you can see above from my script, I gave a basic description on what the *class* is for and what is contained in the *class*, which are my *functions*. To call on a function within a class you call the NAME of the CLASS followed by a “ . ”, then the NAME of the FUNCTION.

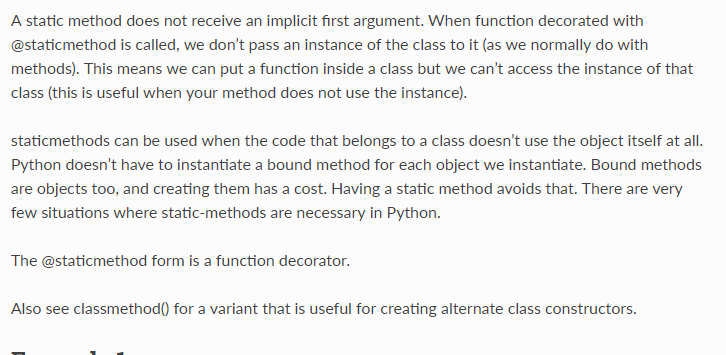
I then went back to the online session recordings in Canvas, (**Figure 2a,b**) to watch the section of how to put the first function in a class, which was the text file code for reading/writing to a python dict/list. It starts at the 2:29:25 mark of the video I incorporated a link to go right to the video. <https://washington.zoom.us/recording/play/jgDdjHD7moi2yih7PFKbxwl_Iuar7zxydOx9AZy5us3Xv9yV7S6QvPT5IwWeSRNc?startTime=1550023165000>



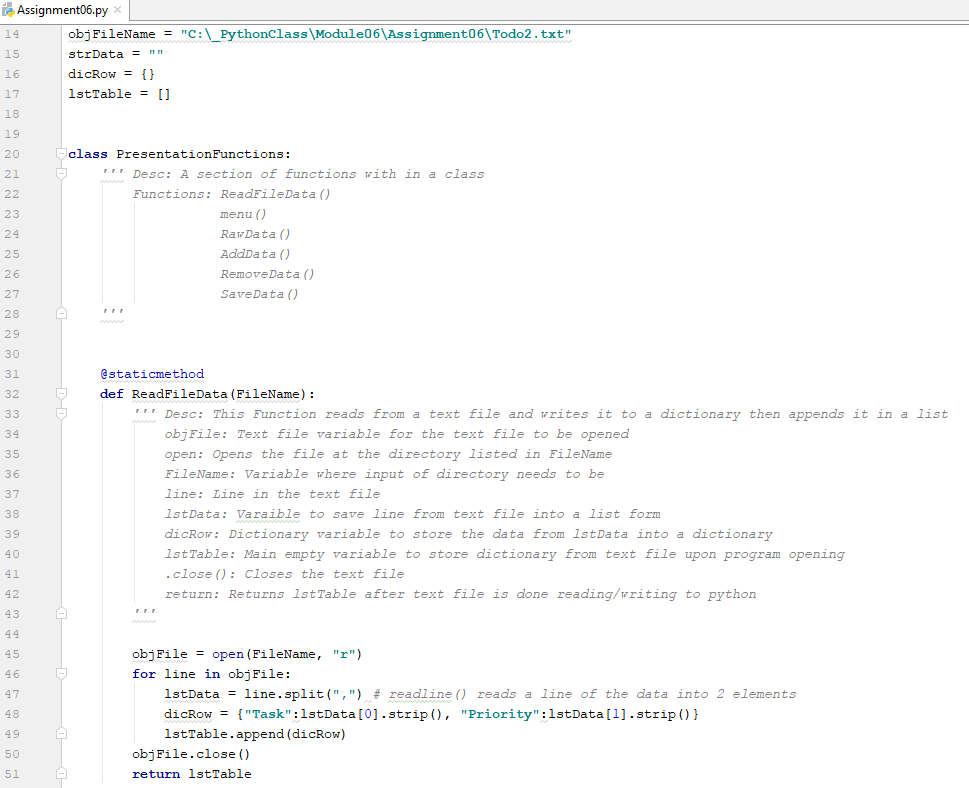


***Figure 2a,b. Session recordings seciton on course overview and which session.***

As you will see in the definition and code below (**Figure 3a,b**) we use the *@staticmethod* above the function. Here is the best definiton that made sense to me on the web about this method:



(python-reference.readthedoc.io, <https://python-reference.readthedocs.io/en/latest/docs/functions/staticmethod.html>, 2019)(External Site)



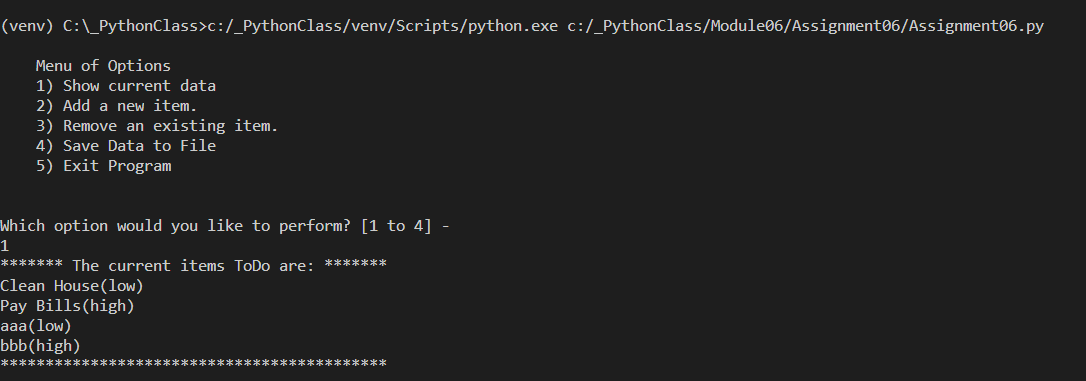


***Figure 3a,b. Shows Class and first Function, and 3b show varible lstTable that calls the function within the class to open the text file and write it to a list.***

Now in **Figure 3a,b**, you see an *function* that has an *argument* within the parentheses called *FileName*. This makes it for cleaner code as we are telling other developers who are looking at our code, what we are doing here. You see in **Figure 3b**, that the *FileName = objFileName*. It is grabbing the variable at the beginning of the page which has the path to the text file which the FileName now becomes, to start the function.

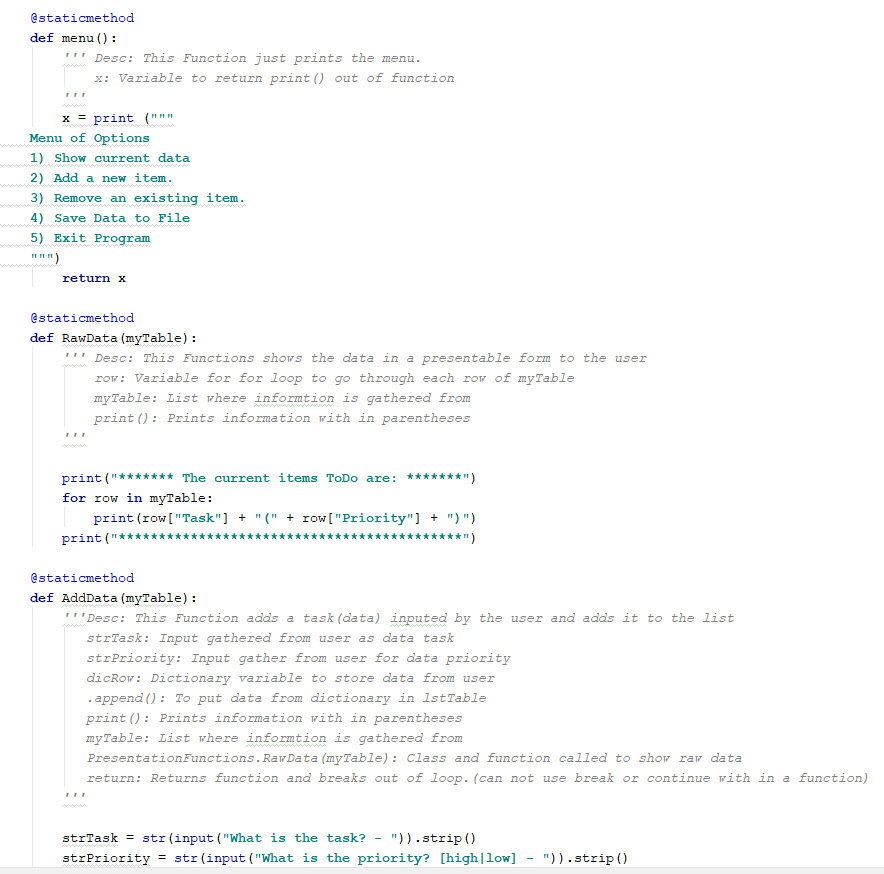
I believe Prefessor Root helped us out on the toughest piece of the assignment in my opinion. I know personally I would have struggled with this for a little while. Now you might ask, Why? I would have been the simple *return lstTable* , that would have made me stumble with this code a bit. After thinking about it for a few minutes it made sense. The text file opens with in the function when it is called later ***lstTable = PresentationFunctions.ReadFileData(FileName = objFileName***). So within the function it runs as normal appending to the empty lstTable = [] variable up top. When the program is done it returns that information to another variable called lstTable, which opens when the script is initiated with *PresentationFunctions.ReadFileData()*. That is how I am making sense of it.

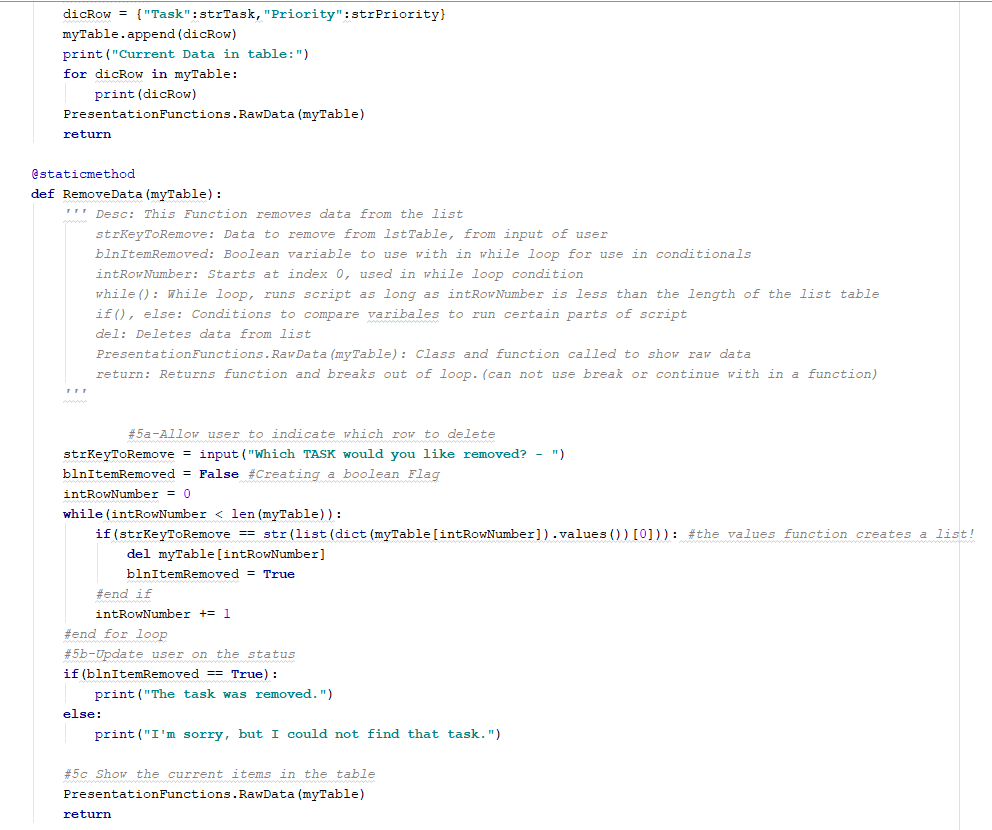
I then tested out the function and got my results in the output below:

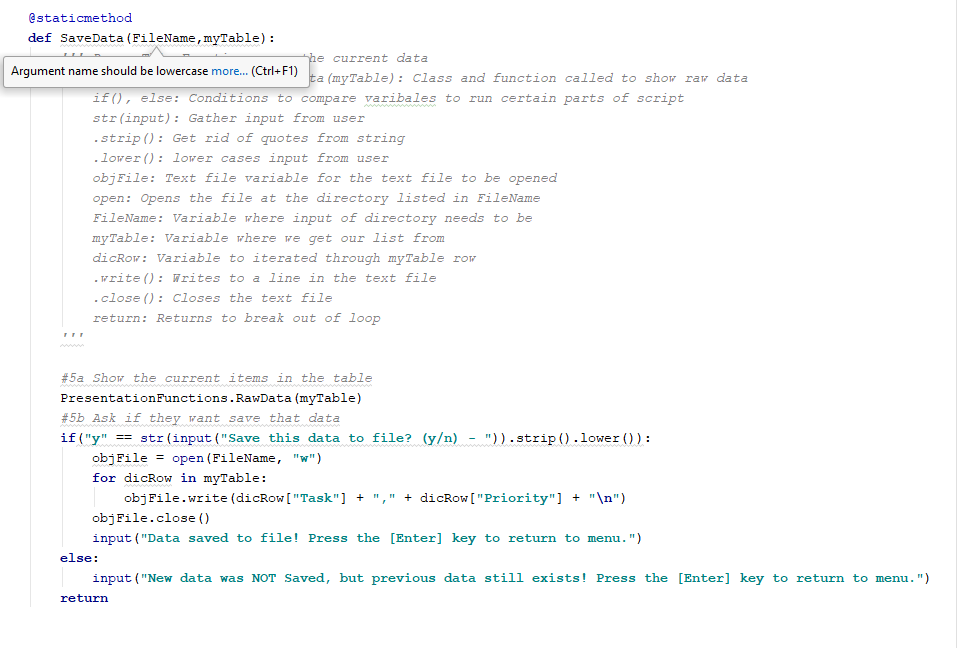


It worked!! I did this for every code that was in the *if-elif statements* and even the menu. I made functions out of each one of them with in the *class*. I only had to make minor changes to each code and add an *argument* to each function of *FileName* and *myTable*. I also changed within the codes anything that had *objFileName* and *lstTable* to *FileName* and *myTable* , to have it make sense to other coders of what each variable is.

Below, in the ***menu()*** function, I was having a hard time printing the menu. I remembered *to set the print statement to a variable(x)* and *return the variable(x)* to show the print statement. Now this is for the rest of the code below. I found out the hard way, from the last assignment, that you can not *break* within a function that is in a loop. I used a global variable last time and it made my code a bit more complicated than it should have been. After talking with Professor Root about this he mentioned that you can *return* within a function which acts the same way as a *break*. So for any part of the original code that has a *break,* I replaced it with *return*.

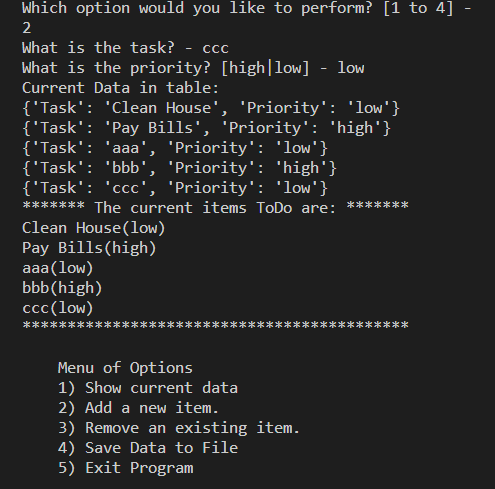




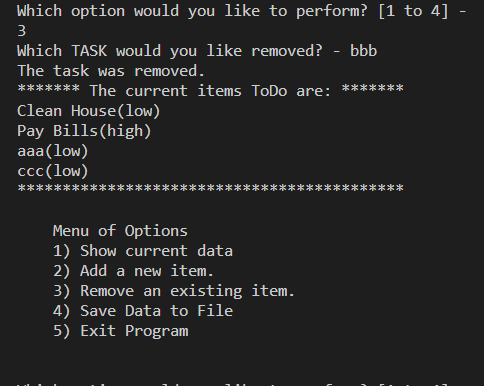


**Outputs**

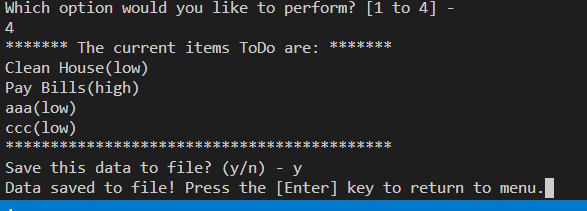
Adding a task:



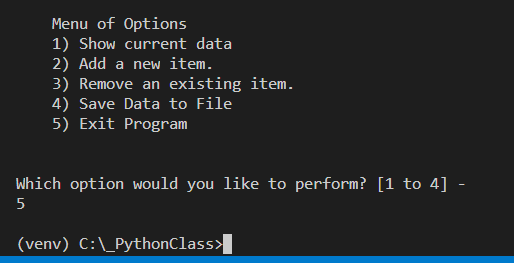
Deleting a task:



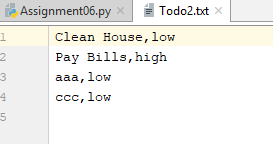
Saving the data file:



Exiting the python script:



Saved data to the text file:



**Summary**

When we finally look at the body of the script you will see how clean and short it is compared to what it originally looked like. The class and functions are able to be reused, if written properly,through out your body of code. For example PresentationFunction.RawData() was used within the other functions to show the data within the list, so you wouldn’t have to retype the code out again. When you are writing complex lines of code having classes and functions will help with the amount of code typing one would have to do. This was an excellent exercise to help understand how classes and functions work and improve our python skills in not repeating our code in our script.

