Andrew Li

Nov. 24, 2020

Foundations of Programming: Python

Assignment06

<https://github.com/idrew4u/IntroToProg-Python-Mod06>

Working with Functions

# Introduction

For this assignment, we were asked to organize a python script. We were given a starting template, and then tasked to use functions to help organize the code. The overall code was basically given from the previous assignment, and it just needed to be converted to the newly provided script.

# Creating the Script

## Script Header

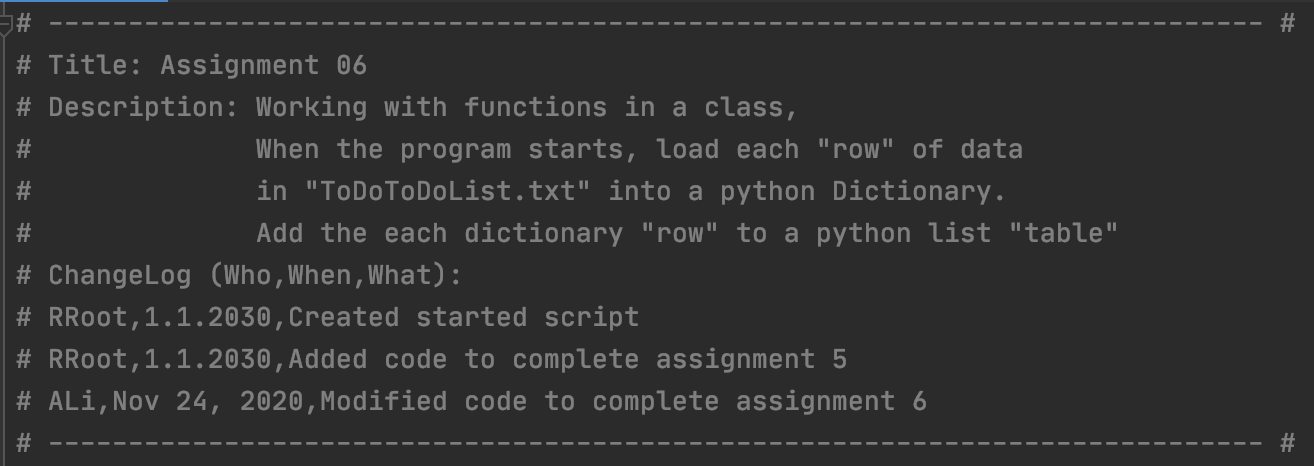
The first thing we should do is write a header for the script. The header is the first thing you see and gives us a basic idea what the script is trying to accomplish. It asks for the title of the program, a description of what it does, and logs changes to the script (Figure 1). The logs should include who made the change, the date it was changed, and what was changed in the script. In this situation, the originator already started the script header, we just had to modify the information to allow other viewers we made updates and changes to the script.

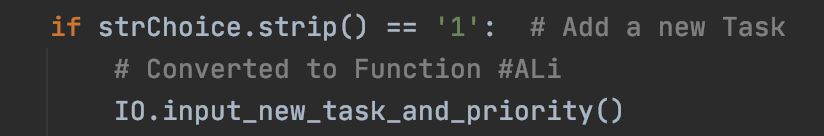
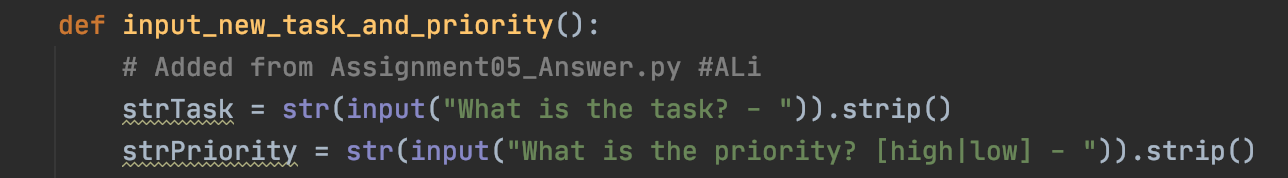
Figure 1: The header includes the title, description, and log changes.

## Functions

We use functions in scripts to help group one or more statements. Functions basically works in two parts. The first part of the function is defining it. When you define the function, you are giving the function a script to run (Figure 2).

Figure 2: An example of defining the function. The function “input\_new\_task\_and\_priority executes the code below.

Secondly, you must call the function in order to execute it (Figure 3). When you call the function, you are using the script that was defined and running it.

Figure 3: An example of calling the function to execute. We execute the “input\_new\_task\_and\_priority function that we defined earlier.

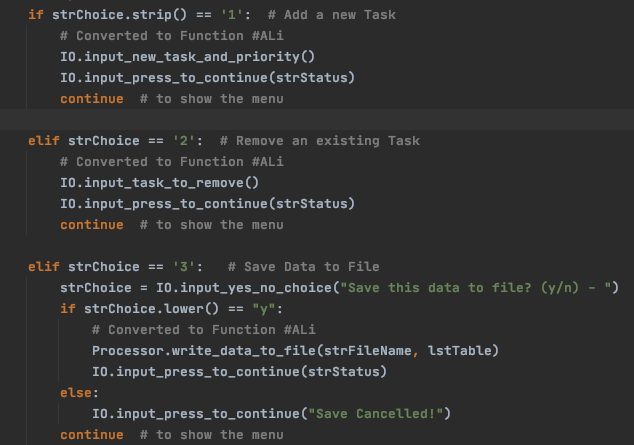
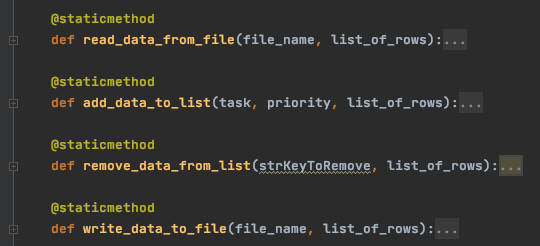
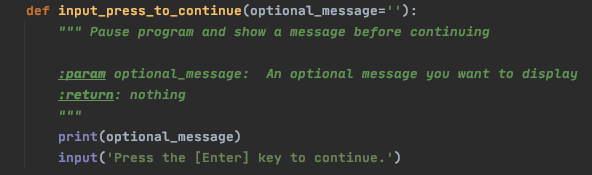
Using functions can help declutter a messy script and allow you to reuse the script without having the rewrite the full script again (Figure 4a-b.).

Figure 4a: In this example, you can see us calling the “input\_press\_to\_continue” multiple times.

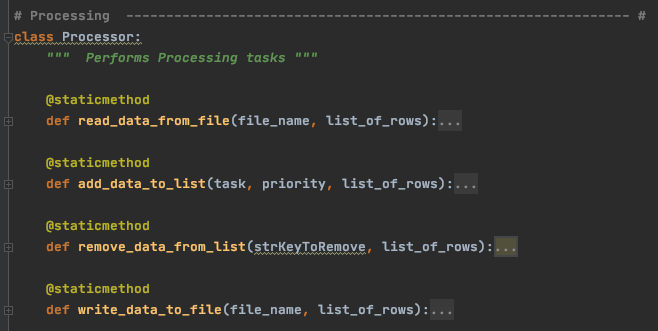
Figure 4b: When we call the function “input\_press\_to\_continue”, it would print out “Press the [Enter] key to continue.” without us having the copy and paste that script multiple times.

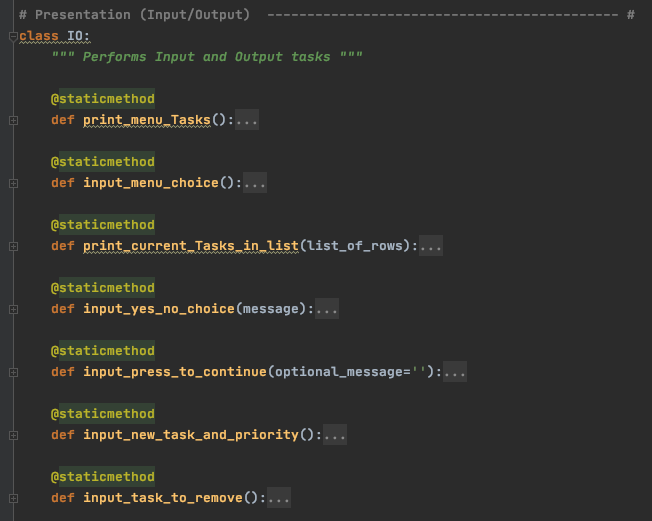
The assignment gave us the two resources: 1) the code which needed to be reorganized with functions and 2) the “answers” where the script was fully written out. Although we have the solution, it can be very daunting to look at at first sight. It is a lot of code and may take a while to decipher. By converting the code into functions, we are able to cleanly label the code and easily hide the script when we are done with it (Figure 5).

Figure 5: Some of the functions collapsed to hide the lines of code that the function executes.

## Classes

Classes are another way we can help organize the code. They can be used to help group functions, variables, and constants. It allows us to group together the script by sections. In our case, the originator broke up the code into Processor (processing) and IO (presentation, input/output) classes. It helps us compartmentalize each section, making it easier to navigate and clarify what section we are working on (Figure 6a-b). By defining these classes, it gives us an idea what the section is responsible for, allowing us to pinpoint what section we need to tackle for certain tasks.

Figure 6a:The Processor (processing) class and the functions included within the class.

Figure 6b: The IO (presentation, input/output) class and the functions included within the class.

# Running the Program

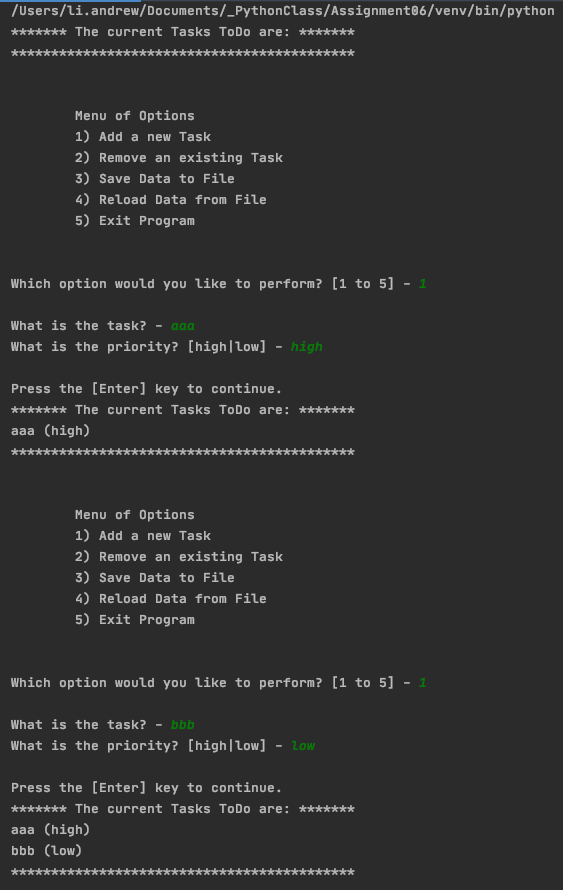
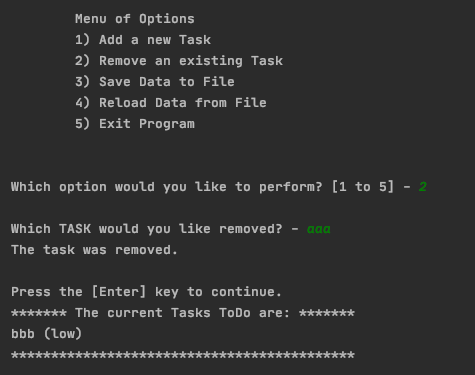
After creating the script, we verified the program is working properly by running it in PyCharm (Figure 7a-c) and the terminal command window on macOS (Figure 11a-b).

Figure 7b

Figure 7a

Figure 7a-b: The results after running our program in the PyCharm IDE, first and second options   
from menu.

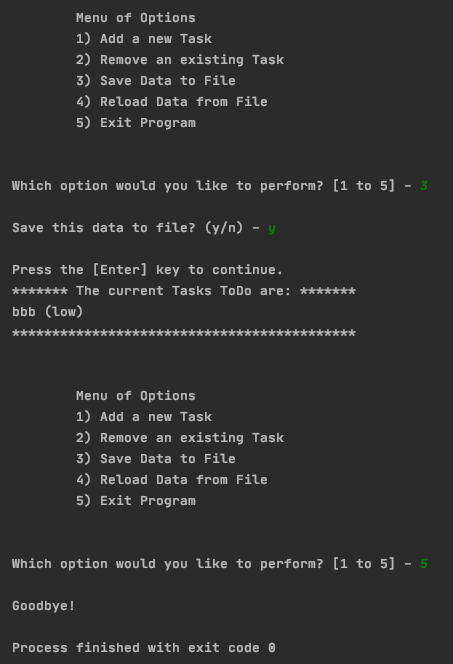
Figure 7c: The results after running our program in the PyCharm IDE, third and fifth options   
from menu.

Figure 8b

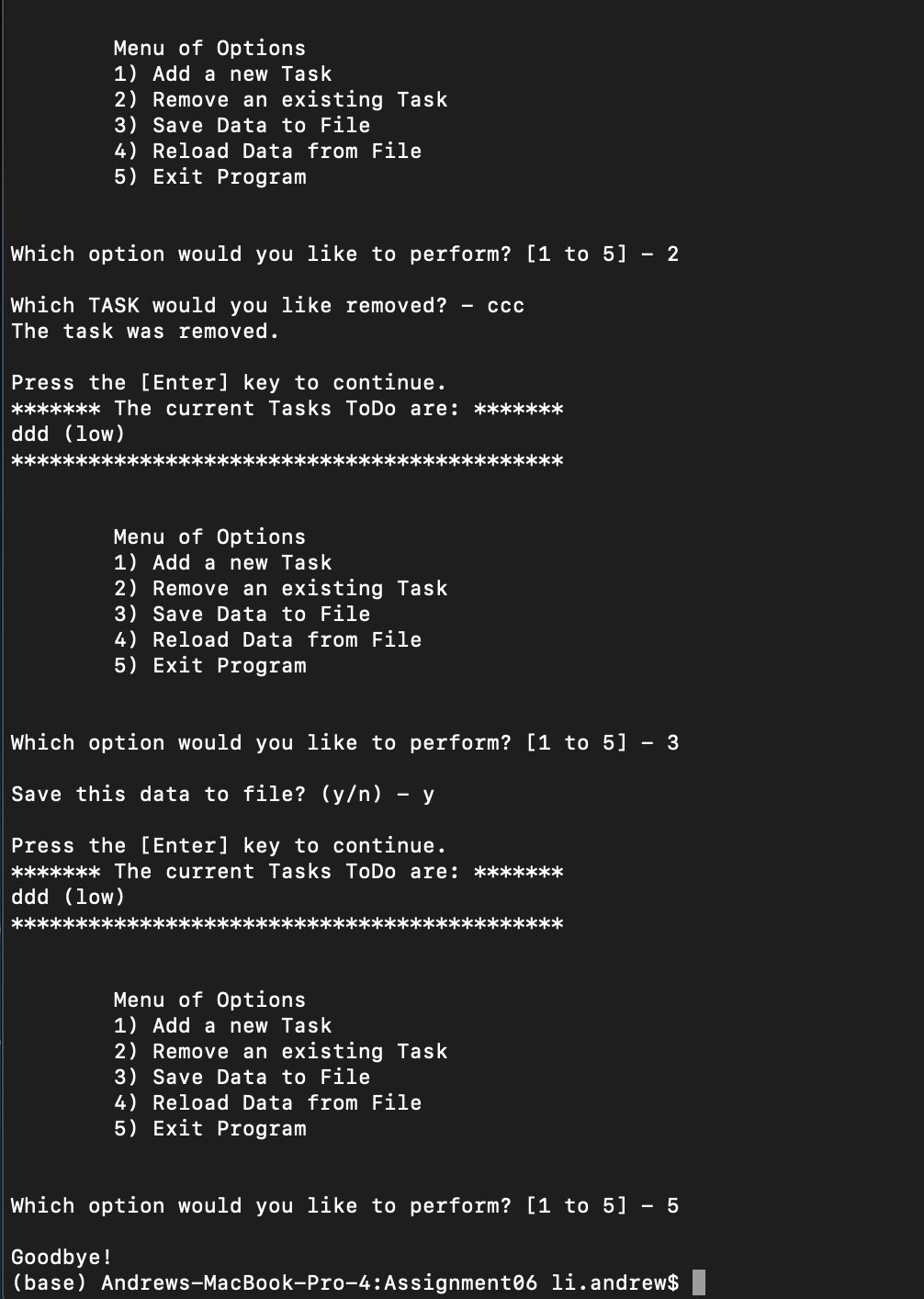
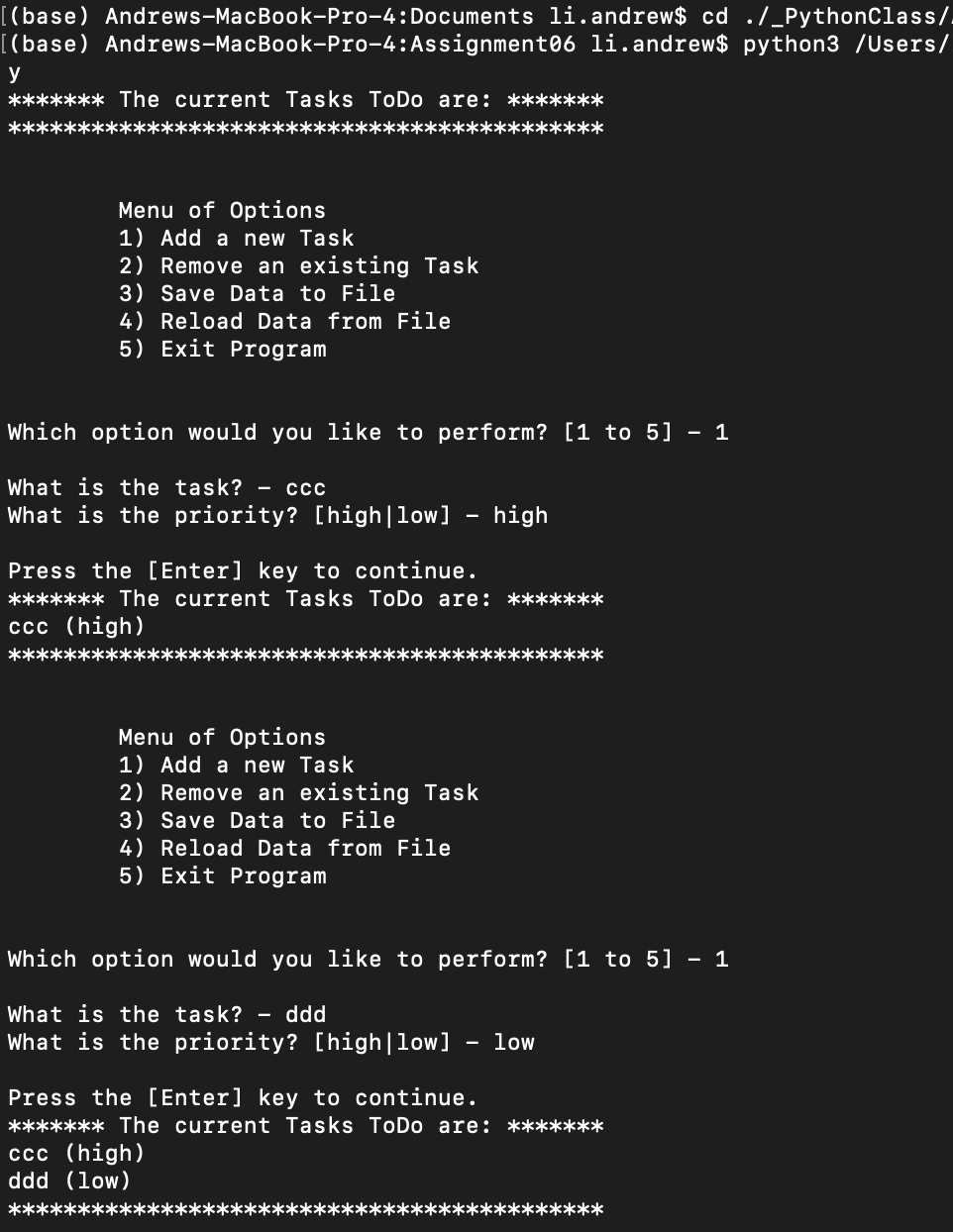


Figure 8a

Figure 8a-b: The results after running our program in the terminal command shell.

# Summary

We were able to complete and “clean up” the code from the originator using functions and classes. Both were used to help organize the script to be more legible. Functions allow us to define the script which we later called and pulled to execute. It also allowed us to use the same script without having the copy and paste or retype the whole script, making the code easier to read and decipher. Classes allowed us to organize our thoughts and compartmentalize parts of the script so it is easy to navigate.