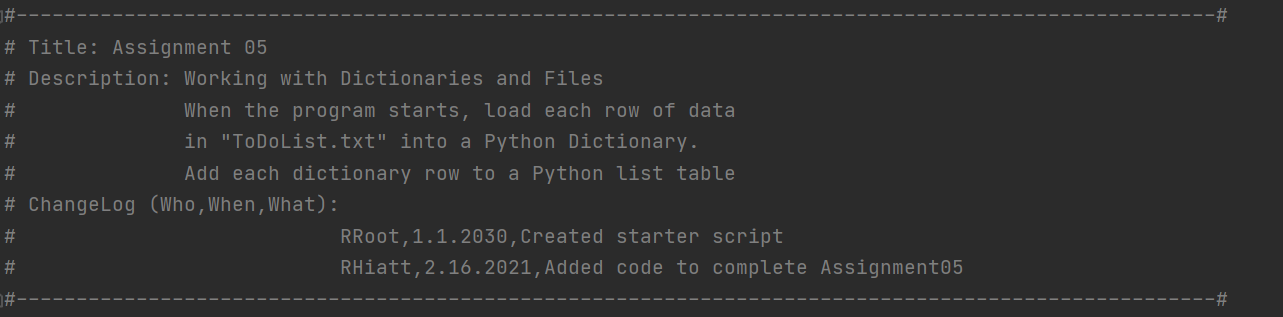
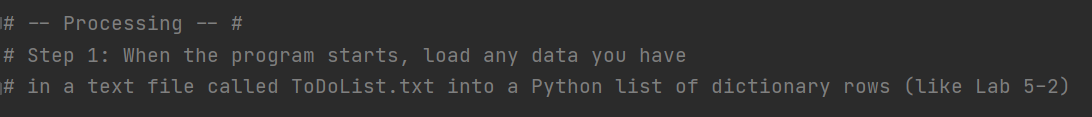
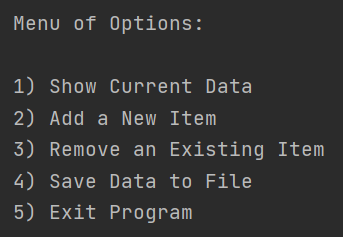
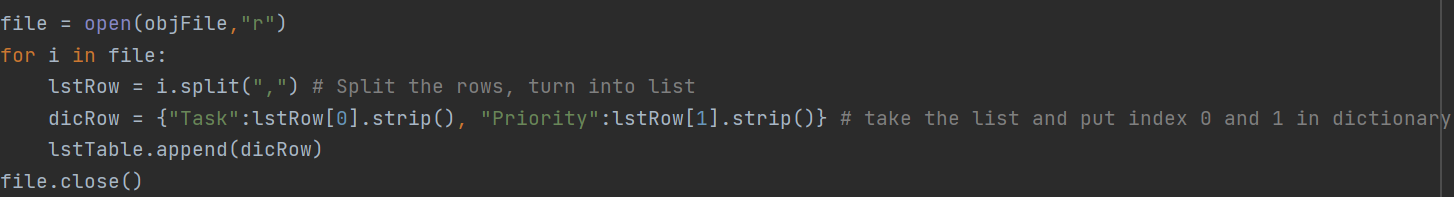
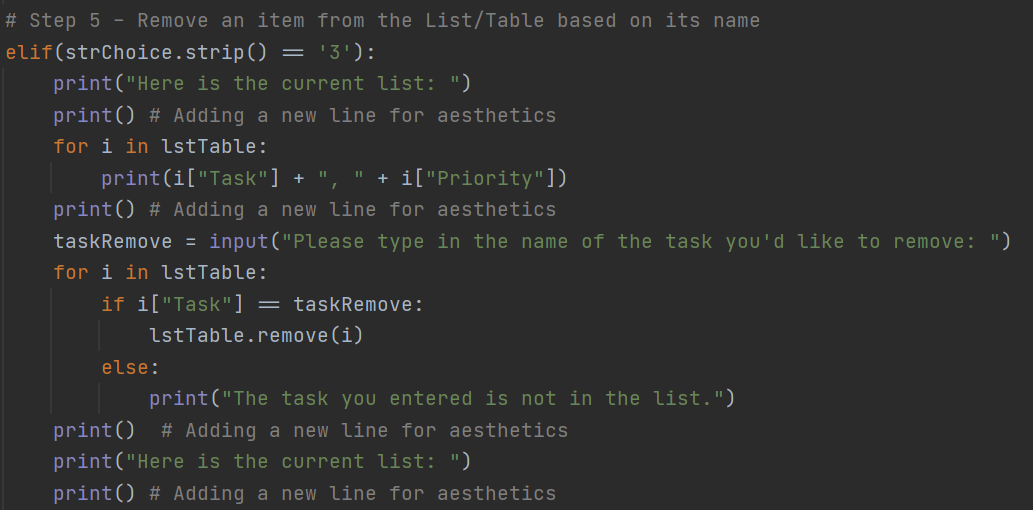
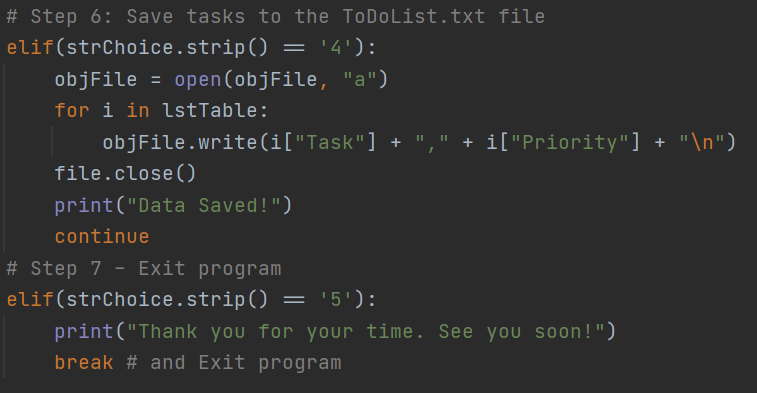
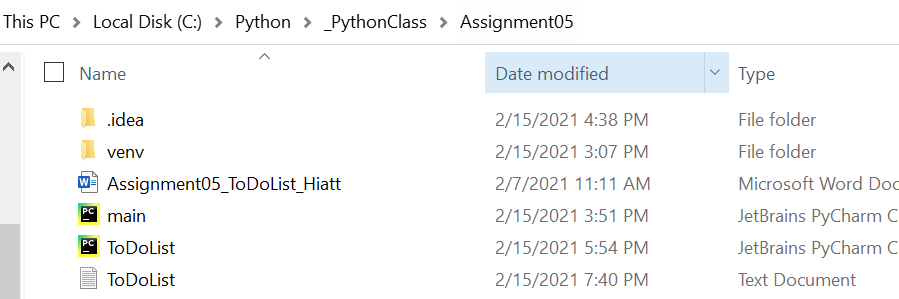
Robert F. Hiatt  
  
February 16, 2021  
  
IT FDN 110 B  
  
Assignment05

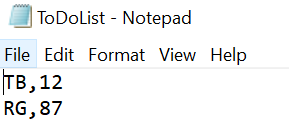
To-Do List Program in Python

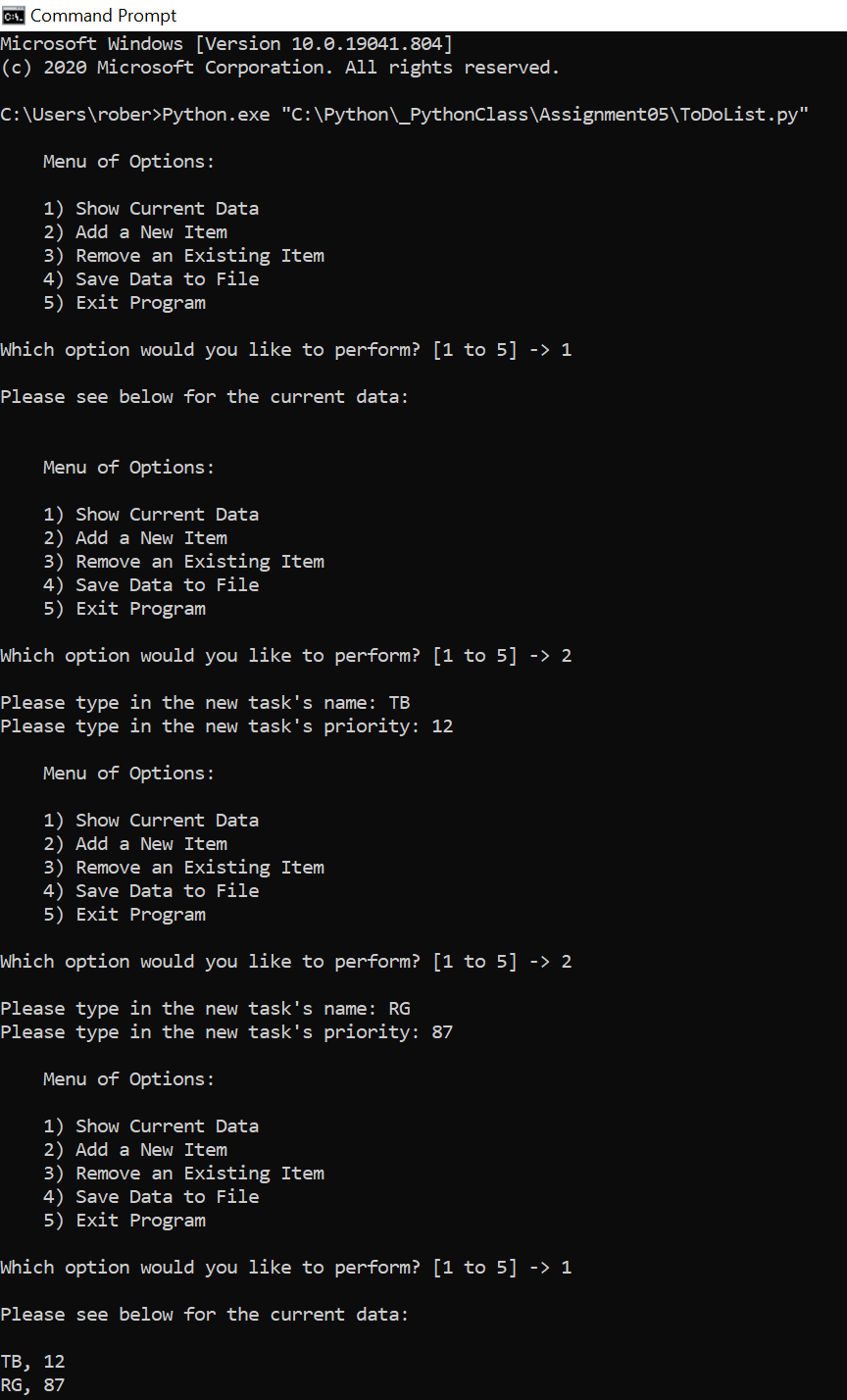
1. Introduction  
This assignment reviews how to create a Python script that allows the user to create and manage a to-do list. The skills incorporated into this Assignment include: creating a menu with lists; slicing lists; creating a dictionary; slicing keys and values in a dictionary; dictionary methods; appending data to lists; creating/writing a file that may be updated at the user’s discretion; taking another person’s code and adding to it; properly using the ChangeLog; adding useful comments; practicing the principle of *separation of concerns*; the *split, strip,* and *remove* methods; and incorporating aesthetic considerations into the user interface. You also create a GitHub account and store your code in GitHub.  
  
2. Using Someone Else’s Code  
Copy the code provided by Randal Root into your dev environment—the file’s title is Assigment05\_Starter.py. Read through the code to get a sense of what the program does. Amend the ChangeLog.  
  
  
***Script header with updated ChangeLog.***Randal provides comments and algorithms already. There are lines where Randal asks you to add data. Note that Randal uses what’s called *separation of concerns* in order to organize the script in a way that’s easier to read, discern, and potentially use later on for other endeavors.   
  
  
***Example of* separation of concerns *with the use of “Processing” in the comments.***  
  
Now let’s begin programming: Randal’s variables are a big help in getting started. For the most part, the concepts in this assignment are a continuation of prior lessons. But let’s first discuss the use of a *dictionary*. In Python, a *dictionary* is a collection of *keys* and *values*. In like manner to a conventional dictionary, *keys* refer to stored terms and *values* refers to their respective definitions. So in this lesson you’re creating a to-do list by building out and managing a *dictionary*. You can add and remove *keys* and *values* to your *dictionary* in Python.   
  
The first addition I made to Randal’s code is to create a file titled ToDoList.txt. I used the same code as Assignment04, but amended the file path:

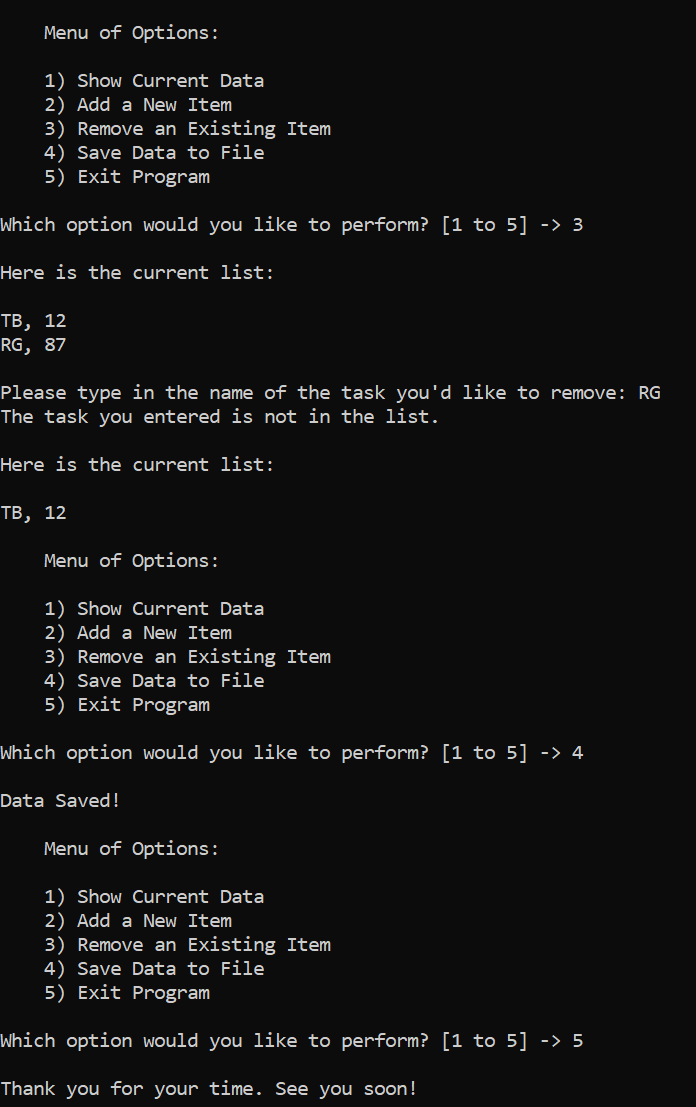
 ***Basic approach to create a text file in the Assignment05 folder.***The to-do list program has five user options, seen here:

  
***Menu options for your to-do list***The code’s first ask is to “load” the text file, and we use the *split* method as well as indexing the *dictionary* in order to load the data.   
  
  
***Note the usage of the* split *method* *and the first foray into a* dictionary*.***To meet the ask, we want to loop through the available data, split and index the information, put that information into our table. The *split* table simply allows you to pick the *keys* and *values* without necessarily taking other characters.   
  
For the next part of the *separation of concerns*, we’ll write the script for what each respective user option should produce in the program. As far as the “new” in this section, there is an additional new method *strip* and the *remove* method. *Strip* is for aesthetic purposes, such as removing a carriage return. The *remove* method is what allows the user to enter a *key* only and thereby remove the key and value from the to-do list.

  
***In the above image you can see the* strip *method and the* remove *method employed. The focus is to allow the user to remove tasks as they wish. Note the ability to print and concatenate independent* keys *and* values*.***  
The main “new” to the menu options was the ability to edit the to-do list. The additional methods were mainly tools for cleaning up the data and visual considerations. The last two menu options, to save the file and to exit the program, are familiar from the previous lesson:  
  
  
***The familiar menu options of saving information and exiting the program.***Now let’s run the script and confirm that we have a text file in our designated file path:  
  
***The text file successfully saved to the designated file path.***

And of course we’ll confirm that the text file has the data we’ve input:  
  
  
***The input was correctly written into the text file.***

Lastly, let’s run our program on the Command Line:  




3. Summary  
By this exercise’s completion you now have a program for the user to create and manage a to-do list. The core Pythonic feature deployed is a *dictionary*. You used some new methods, including *split*, *strip*, and *remove*, the latter of which allows the user to erase *keys* and *values* from their to-do list. And you’ve continued your education in creating and writing objects, with the bonus component now of editing them.