Brenden Britt

5/18/21

Foundations of Programming: Python

Assignment05

Working With Functions in a Class

# Introduction

In this write-up, I will be discussing how I created a Python Script that asks a user to input one of five different options to fulfill a task. The options are to add to a file, delete from the file, save what you added to the file, reload data from the file, and end the script. The script in this assignment uses classes and functions, among other things learned in previous assignments, to create a working program that solves this task.

# Learning Functions and Classes:

An important part of this assignment was learning what a function and a class are. Functions are used in python to group statements together and must be defined calling the function in the code. Classes are used as a way of grouping functions, variables, and constants. Using both classes and functions are useful for maintaining an organized script.

Text

Description automatically generated

***Figure 1. Learning Functions and Classes. LAB 6-1***

Figure 1 shows the script I created to learn what functions and classes are. In this script, I defined several functions to add, subtract, multiple, and divide two values. These were defined under one class. I then called the functions lower in the code after defining them.

# Creating a script that works with classes and functions to modify an existing script

For this assignment, I used functions and tuples to modify and organize an existing script. I began by defining functions in the class IO (input/output) section. For this, I added code to a function that asked for user input to create a new task and priority. I also added code to function that prints whether or not removing a task was successful. Next, I moved onto the Presentation class. I added code to a number of functions that add data to a file, remove data from a file, and writes data to a file. I did this by using information learned in Assignment05 about lists, tables, dictionaries, and writing data to existing files. After this, I added code to the main body of the script that calls the functions I completed above. Again, I used information learned from Assignment05 to complete this. Something new that I used was dot format, which calls a specific function within a class to fulfill a task. When complete, the user can enter 1, 2, 3, 4, and/or 5 to complete one of five different menu options.

class Processor:  
 *""" Performs Processing tasks """* @staticmethod  
 def read\_data\_from\_file(file\_name):  
 *""" Reads data from a file into a list of dictionary rows* ***:param*** *file\_name: (string) with name of file:* ***:param*** *list\_of\_rows: (list) you want filled with file data:* ***:return****: (list) of dictionary rows  
 """* list\_of\_rows = []  
 file = open(file\_name, "r")  
 for line in file:  
 data = line.split(",")  
 row = {"Task": data[0].strip(), "Priority": data[1].strip()}  
 list\_of\_rows.append(row)  
 file.close()  
 return list\_of\_rows  
  
 @staticmethod  
 def add\_data\_to\_list(list\_of\_rows, task, priority):  
 dicRow = {'Task': str(task).strip(), 'Priority': str(priority).strip()}  
 list\_of\_rows.append(dicRow)  
 return list\_of\_rows, 'Success'  
  
 @staticmethod  
 def remove\_data\_from\_list(taskToRemove, list\_of\_rows):  
 successRemove = False  
 rowNumber = 0  
 for dicRow in list\_of\_rows:  
 if dicRow['Task'].lower() == taskToRemove.lower():  
 list\_of\_rows.remove(dicRow)  
 successRemove = True  
 rowNumber += 1  
 return list\_of\_rows, successRemove  
  
 @staticmethod  
 def write\_data\_to\_file(file\_name, list\_of\_rows):  
 objFile = open(file\_name, 'w')  
 for dicRow in lstTable:  
 objFile.write(dicRow['Task'] + ',' + dicRow['Priority'] + '\n')  
 objFile.close()  
 return list\_of\_rows, 'Success'

***Figure 3. Code for this assignment***

@staticmethod  
 def input\_new\_task\_and\_priority():  
 strTask = str(input('Enter a new task: ')).strip()  
 strPriority = str(input('Enter a new priority (high/low): ')).strip()  
 return strTask, strPriority  
  
 @staticmethod  
 def input\_task\_to\_remove(successRemove):  
 if successRemove:  
 print('Removed task')  
 else:  
 print('Task not found')  
  
# Main Body of Script ------------------------------------------------------ #  
  
# Step 1 - When the program starts, Load data from ToDoFile.txt.  
lstTable = Processor.read\_data\_from\_file(strFileName) # read file data  
  
# Step 2 - Display a menu of choices to the user  
while (True):  
 # Step 3 Show current data  
 IO.print\_current\_Tasks\_in\_list(lstTable) # Show current data in the list/table  
 IO.print\_menu\_Tasks() # Shows menu  
 strChoice = IO.input\_menu\_choice() # Get menu option  
   
 # Step 4 - Process user's menu choice  
 if strChoice.strip() == '1': # Add a new Task  
 addData = IO.input\_new\_task\_and\_priority()  
 Processor.add\_data\_to\_list(lstTable, addData[0], addData[1])  
 IO.input\_press\_to\_continue(strStatus)  
 IO.print\_current\_Tasks\_in\_list(lstTable)  
 continue # to show the menu  
  
 elif strChoice == '2': # Remove an existing Task  
 removeTask = input('Enter task to be removed: ')  
 taskDataRemoved = Processor.remove\_data\_from\_list(removeTask, lstTable)  
 IO.input\_task\_to\_remove(taskDataRemoved)  
 IO.input\_press\_to\_continue(strStatus)  
 IO.print\_current\_Tasks\_in\_list(lstTable)  
 continue # to show the menu  
  
 elif strChoice == '3': # Save Data to File  
 strChoice = IO.input\_yes\_no\_choice("Save this data to file? (y/n) - ")  
 IO.print\_current\_Tasks\_in\_list(lstTable)  
 if strChoice.lower() == "y":  
 Processor.write\_data\_to\_file(strFileName, lstTable)  
 IO.input\_press\_to\_continue(strStatus)  
 else:  
 IO.input\_press\_to\_continue("Save Cancelled!")  
 continue # to show the menu  
  
 elif strChoice == '4': # Reload Data from File  
 print("Warning: Unsaved Data Will Be Lost!")  
 strChoice = IO.input\_yes\_no\_choice("Are you sure you want to reload data from file? (y/n) - ")  
 if strChoice.lower() == 'y':  
 lstTable.clear()  
 lstTable = Processor.read\_data\_from\_file(strFileName)  
 IO.input\_press\_to\_continue(strStatus)  
 IO.print\_current\_Tasks\_in\_list(lstTable)  
 else:  
 IO.input\_press\_to\_continue("File Reload Cancelled!")  
 continue # to show the menu

***Figure 4. Code for this assignment cont.***

***Text

Description automatically generated***

***Figure 5. Successfully funning my script***

***Text

Description automatically generated***

***Figure 6. Successfully funning my script***

***Text

Description automatically generated***

***Figure 7. Successfully funning my script cont.***

***Text

Description automatically generated with medium confidence***

***Figure 8. Successfully funning my script cont.***

Text

Description automatically generated with medium confidence

***Figure 9. Testing my code in the terminal***

Figures 1, 2, 3, 4, 5, 6, 7, 8, and 9 show my code take user input data to interact with the menu and work with an existing script. These images show the code running in PyCharm and the terminal, and the script working.

# Summary

This assignment was very difficult as it had many different sections that needed attention. The notes provided by Professor Root gave good examples of how to create and use classes and function, and working with dictionaries, tuples, and existing files. Despite this, I still struggled with this assignment and had to use Professor Root’s ‘help’ video. However, I was able to complete a working script using all the available help, and work I did on previous assignments.

Link to repository: https://github.com/Brenden1354/IntroToProg-Python-Mod06