Dmitriy Alexandrov

November 12th, 2019

IT FDN 100 A

Assignment06

Coding a To Do Task List Script with Functions

Introduction

In this paper I am going to go through the steps that I took in adding code to a starter script to create a fully functional To Do Task List script, which opens up a ToDoList.txt file (or creates it, if the files doesn't already exist) and is able to read, write and update tasks to that file using input that the user provides but this time using functions.

Steps Taken in Adding Code to the To Do Task List Script

I started out the process by launching PyCharm and creating a new project called Assignment06 in my _PythonClass folder on my C: drive using a virtual Python interpreter. After the project was created, I imported the starting code for the assignment called "Assignment06_Starter.py".

With the file in PyCharm, the first thing I added to the document was a change log entry in the header of the code to identify who was working on the code after the initial creation of it.

```
# ------------------------------------------------------------------------ #
# Title: Assignment 06
# Description: Working with functions in a class,
#              When the program starts, load each "row" of data
#              in "ToDoToDoList.txt" into a python Dictionary.
#              Add the each dictionary "row" to a python list "table"
# ChangeLog (Who,When,What):
# RRoot,1.1.2030,Created started script
# RRoot,1.1.2030,Added code to complete assignment 5
# RRoot,1.1.2030,Fixed bug by clearing the list before it was refilled
# DAlexandrov,11.11.2019,Modified code to complete assignment 6
# ------------------------------------------------------------------------ #
```

After taking a look at the declared variables that were already in the code, I identified what I already had to work with. The only change that I made to the already declared variables was adding a lstMenu variable to keep track of user inputs as an added feature that I also had in Assignment05.

```
strChoice = ""  # Capture the user option selection
lstMenu = []# Keep track of user inputs for exit menu
# Data ---------------------------------------------------------------------- #
```

The first change that I made was to the ReadFileDataToList function, because when I initially ran the script it gave me an error that the file that the file didn't exist. So I added a try except statement to create the file if it doesn't exist before trying to open it. This code tries to open the file, if an error is generated then it creates the file first then tries again.

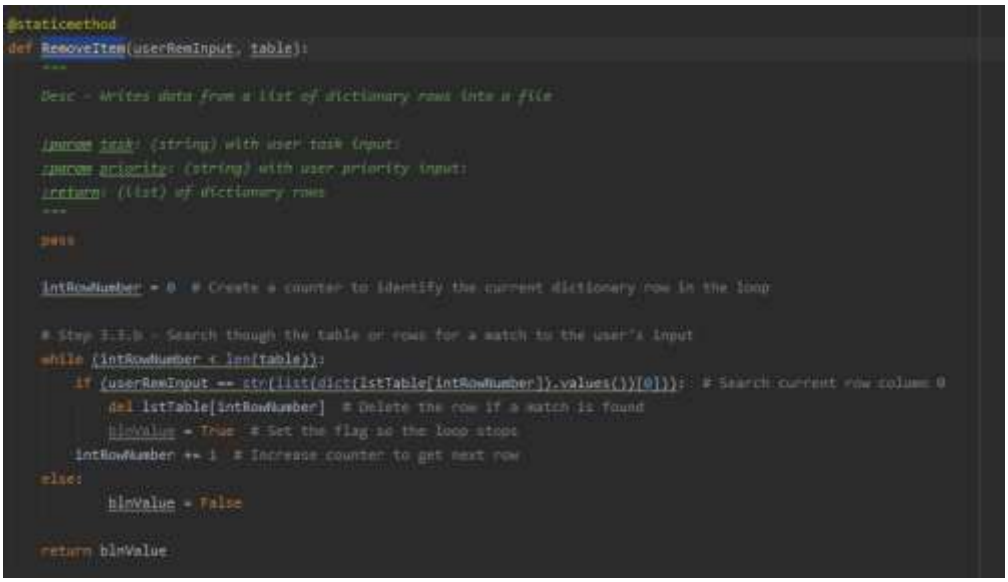
```
while(True):
    try:
        file = open(file_name, "r")  # Tries to read the file
        break
    except FileNotFoundError:  # If the file isn't found, throws an exception and creates the file and restarts the program.
        print()
        print("Cannot find file, creating the " + file_name + " file and restarting...")
        file = open(file_name, "x")
        continue
    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
```

After fixing the ReadFileDataToList function, I moved to the WriteListDataToFile function and added the following code to it:

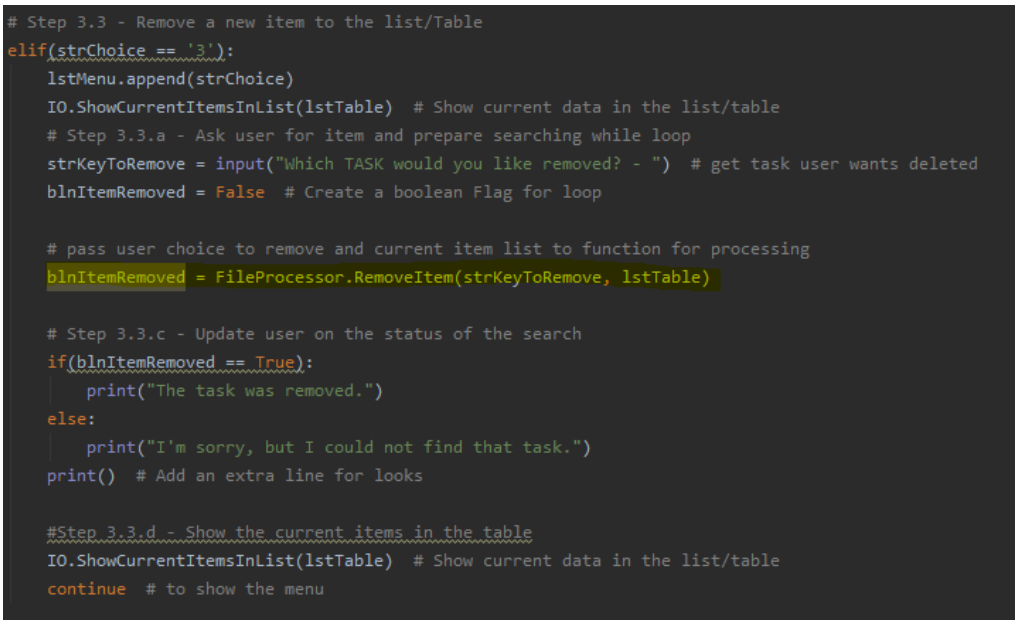
```
@staticmethod
def WriteListDataToFile(file_name, list_of_rows):
    """
    Desc - Writes data from a list of dictionary rows into a file

    :param file_name: (string) with name of file:
    :param list_of_rows: (list) you want filled with file data:
    :return: (list) of dictionary rows
    """
    pass
    file = open(file_name, "w")
    for row in list_of_rows:  # Write each row of data to the file
        file.write(row["Task"] + "," + row["Priority"] + "\n")
    file.close()
```

First, I added the Description and parameters to show the tooltip in PyCharm if someone wanted to see what this function does. After that I took the code from the Step #3.4 in the starter code and placed it into this function so that it can be reused in the future. This function took in two parameter (file_name and list_of_rows) and writes the data to a file.

Continuing looking through the starter code, the remove task code needed to be a function so that it can be reused rather than being used once in the main code. For this I created a new function called RemoveItem which received two parameters userRemInput and table. userRemInput was what the user input for which task they wanted removed and “table” was the list of current tasks. When the function runs and receives these inputs it looks through the list of tasks and if it finds the task that the user wants to delete, it deletes it and returns a Boolean value to let the user know if the item was deleted or not.



```
@staticmethod
def RemoveItem(userRemInput, table):
    """
    Desc - Writes data from a list of dictionary rows into a file

    :param task: (string) with user task input:
    :param priority: (string) with user priority input:
    :return: ((list) of dictionary rows
    """
    pass

    intRowNumber = 0  # Create a counter to identify the current dictionary row in the loop

    # Step 3.3.b - Search though the table or rows for a match to the user's input
    while (intRowNumber < len(table)):
        if (userRemInput == str(list(dict(lstTable[intRowNumber]).values())[0])):  # Search current row column 0
            del lstTable[intRowNumber]  # Delete the row if a match is found
            blnValue = True  # Set the flag so the loop stops
        intRowNumber += 1  # Increase counter to get next row
    else:
            blnValue = False

    return blnValue
```



```
# Step 3.3 - Remove a new item to the list/Table
elif(strChoice == '3'):
    lstMenu.append(strChoice)
    IO.ShowCurrentItemsInList(lstTable)  # Show current data in the list/table
    # Step 3.3.a - Ask user for item and prepare searching while loop
    strKeyToRemove = input("Which TASK would you like removed? - ")  # get task user wants deleted
    blnItemRemoved = False  # Create a boolean Flag for loop

    # pass user choice to remove and current item list to function for processing
    blnItemRemoved = FileProcessor.RemoveItem(strKeyToRemove, lstTable)

    # Step 3.3.c - Update user on the status of the search
    if(blnItemRemoved == True):
        print("The task was removed.")
    else:
        print("I'm sorry, but I could not find that task.")
    print()  # Add an extra line for looks

    #Step 3.3.d - Show the current items in the table
    IO.ShowCurrentItemsInList(lstTable)  # Show current data in the list/table
    continue  # to show the menu
```

Jumping back above the delete task function, I added a function that would handle the adding of a new task by taking user input and after the function is executed return what the user put in, into a new local variable (strInput). After that I created two more local variables (strTask, strPriorityt) which took the 1st and 2nd values of strInput which were the task and priority that the user input. After the user input is received and stored, I created a ProcessUserInput function which added the user input to the task list.

```
@staticmethod
def AddNewTask():
    """ Gets the input from a user
    :return: string
    """
    # Ask user for input on Task Name // Task Priority
    strTaskName = str(input("What is the task? - ")).strip()  # Get task from user
    strTaskPriority = str(input("What is the priority? [high|low] - ")).strip()  # Get priority from user
    return strTaskName, strTaskPriority
```

```
# Step 3.2 - Add a new item to the list/Table
elif(strChoice.strip() == '2'):
    lstMenu.append(strChoice)
    # Step 3.2.a - Ask user for new task and priority
    userInput = IO.AddNewTask()# get user input from AddNewTask function
    userTask = userInput[0]# set task to the task that user input
    userPriority = userInput[1]# set priority to level that user input
    print()  # Add an extra line for looks

    # Step 3.2.b  Add item to the List/Table
    FileProcessor.ProcessUserInput(userTask, userPriority)# pass the user's task/priority to add to list
    IO.ShowCurrentItemsInList(lstTable)  # Show current data in the list/table
    continue  # to show the menu
```

```
@staticmethod
def ProcessUserInput(task, priority):
    """
    Desc - Writes data from a list of dictionary rows into a file

    :param task: (string) with user task input:
    :param priority: (string) with user priority input:
    :return: (list) of dictionary rows
    """
    pass
    dicNewRow = {"Task": task, "Priority": priority}  # Create a new dictionary row
    lstTable.append(dicNewRow)  # Add the new row to the list/table
    return lstTable
```

Lastly, I added the ExitCheck function that would run if the user chose to exit the script (option 6). This check would see what values were in the lstMenu variable that I added at the beginning and appended to whenever the user chose an option. With this list if the user ever added a task (option 2) or deleted a task from the list (option 3) and did not save their progress (option 4) then it would ask the user if they

want to save their progress before exiting. The function then returned the user's response and depending on if they chose to exit or not would exit the program or continue running until a save had been completed or the user just chose to exit.

```
@staticmethod
def ExitCheck(userMenu):
    """ Gets the input from a user
    :return: string
    """

    if (("2" in userMenu or "3" in userMenu) and "4" not in userMenu):
        # if the user has added or deleted tasks from the list and they have not saved the data, ask them if they want to save
        strExit = input("You added or deleted task(s) without saving, are you sure you want to quit? (y/n) ")
        if (strExit.lower().strip() == 'y'):
            return strExit
        elif (strExit.lower().strip() == 'n'):
            return strExit
        else:
            strExit = "n"
            return strExit
    else:
        strExit = "y"
        return strExit
```

```
# Step 3.6 - Exit the program
elif (strChoice == '6'):
    strReturn = IO.ExitCheck(lstMenu)
    if (strReturn == "y"):
        break # and Exit
    if (strReturn == "n"):
        continue # Continue running the script
    else:
        print("You did not choose yes or no.")
        continue
```

Summary

Overall, the script worked as intended and reflected what was being asked for in the assignment. I added extra functionality to the script outside of the original assignment to help add some more functionality to the program as stated in the steps above.

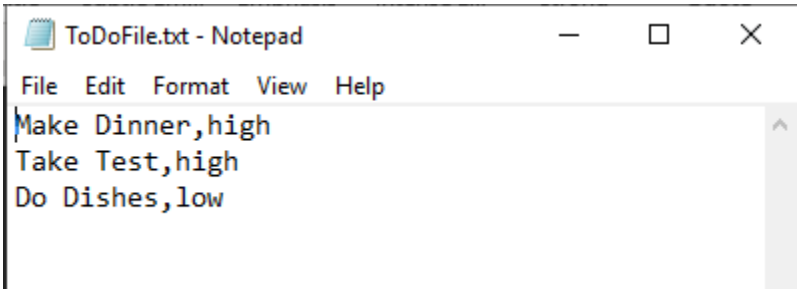
The results from running the script from PyCharm were:

```
C:\_PythonClass\Assignment06\venv\Scripts\python.exe C:/_PythonClass/Assignment06/Assignment06_Starter.py

        Menu of Options
        1) Show current data
        2) Add a new item.
        3) Remove an existing item.
        4) Save Data to File
        5) Reload Data from File
        6) Exit Program


Which option would you like to perform? [1 to 6] - 1

******* The current items ToDo are: *******
Make Dinner (high)
Take Test (high)
Do Dishes (low)
*******************************************


        Menu of Options
        1) Show current data
        2) Add a new item.
        3) Remove an existing item.
        4) Save Data to File
        5) Reload Data from File
        6) Exit Program


Which option would you like to perform? [1 to 6] - 2

What is the task? - Testing1
What is the priority? [high|low] - high

******* The current items ToDo are: *******
Make Dinner (high)
Take Test (high)
Do Dishes (low)
Testing1 (high)
*******************************************
```

```
        Menu of Options
        1) Show current data
        2) Add a new item.
        3) Remove an existing item.
        4) Save Data to File
        5) Reload Data from File
        6) Exit Program


Which option would you like to perform? [1 to 6] - 3

******* The current items ToDo are: *******
Make Dinner (high)
Take Test (high)
Do Dishes (low)
Testing1 (high)
*****************************************

Which TASK would you like removed? - Testing1
I'm sorry, but I could not find that task.

******* The current items ToDo are: *******
Make Dinner (high)
Take Test (high)
Do Dishes (low)
*****************************************


        Menu of Options
        1) Show current data
        2) Add a new item.
        3) Remove an existing item.
        4) Save Data to File
        5) Reload Data from File
        6) Exit Program


Which option would you like to perform? [1 to 6] - 4
```

```
******* The current items ToDo are: *******
Make Dinner (high)
Take Test (high)
Do Dishes (low)
*****************************************

Save this data to file? (y/n) - y
Data saved to file! Press the [Enter] key to return to menu.

        Menu of Options
        1) Show current data
        2) Add a new item.
        3) Remove an existing item.
        4) Save Data to File
        5) Reload Data from File
        6) Exit Program


Which option would you like to perform? [1 to 6] - 5

Warning: This will replace all unsaved changes. Data loss may occur!
Reload file data without saving? [y/n] - n
File data was NOT reloaded! Press the [Enter] key to return to menu.
******* The current items ToDo are: *******
Make Dinner (high)
Take Test (high)
Do Dishes (low)
*****************************************


        Menu of Options
        1) Show current data
        2) Add a new item.
        3) Remove an existing item.
        4) Save Data to File
        5) Reload Data from File
        6) Exit Program


Which option would you like to perform? [1 to 6] - 6


Process finished with exit code 0
```

The output in the ToDoList.txt file was:



And running the script from the OS shell looked like this:

```
c:\_PythonClass\Assignment06>Python Assignment06_Starter.py

        Menu of Options
        1) Show current data
        2) Add a new item.
        3) Remove an existing item.
        4) Save Data to File
        5) Reload Data from File
        6) Exit Program


Which option would you like to perform? [1 to 6] - 1

******* The current items ToDo are: *******
Make Dinner (high)
Take Test (high)
Do Dishes (low)
*******************************************



        Menu of Options
        1) Show current data
        2) Add a new item.
        3) Remove an existing item.
        4) Save Data to File
        5) Reload Data from File
        6) Exit Program


Which option would you like to perform? [1 to 6] - 2

What is the task? - Test
What is the priority? [high|low] - low

******* The current items ToDo are: *******
Make Dinner (high)
Take Test (high)
Do Dishes (low)
Test (low)
*******************************************



        Menu of Options
        1) Show current data
        2) Add a new item.
        3) Remove an existing item.
        4) Save Data to File
        5) Reload Data from File
        6) Exit Program


Which option would you like to perform? [1 to 6] - 3
```

```
Which option would you like to perform? [1 to 6] - 3

******* The current items ToDo are: *******
Make Dinner (high)
Take Test (high)
Do Dishes (low)
Test (low)
*******************************************

Which TASK would you like removed? - Test
I'm sorry, but I could not find that task.

******* The current items ToDo are: *******
Make Dinner (high)
Take Test (high)
Do Dishes (low)
*******************************************


        Menu of Options
        1) Show current data
        2) Add a new item.
        3) Remove an existing item.
        4) Save Data to File
        5) Reload Data from File
        6) Exit Program


Which option would you like to perform? [1 to 6] - 4

******* The current items ToDo are: *******
Make Dinner (high)
Take Test (high)
Do Dishes (low)
*******************************************

Save this data to file? (y/n) - y
Data saved to file! Press the [Enter] key to return to menu.6

        Menu of Options
        1) Show current data
        2) Add a new item.
        3) Remove an existing item.
        4) Save Data to File
        5) Reload Data from File
        6) Exit Program


Which option would you like to perform? [1 to 6] - 6
```

The output in the ToDoList.txt file was: