Tsedey Tadesse
Nov 16, 2022
IT FND 110 A AU 22
Assignment05
GitHub Link: https://github.com/tsedeytadesse/PythonClassFiles

Lists, Dictionaries, and Files

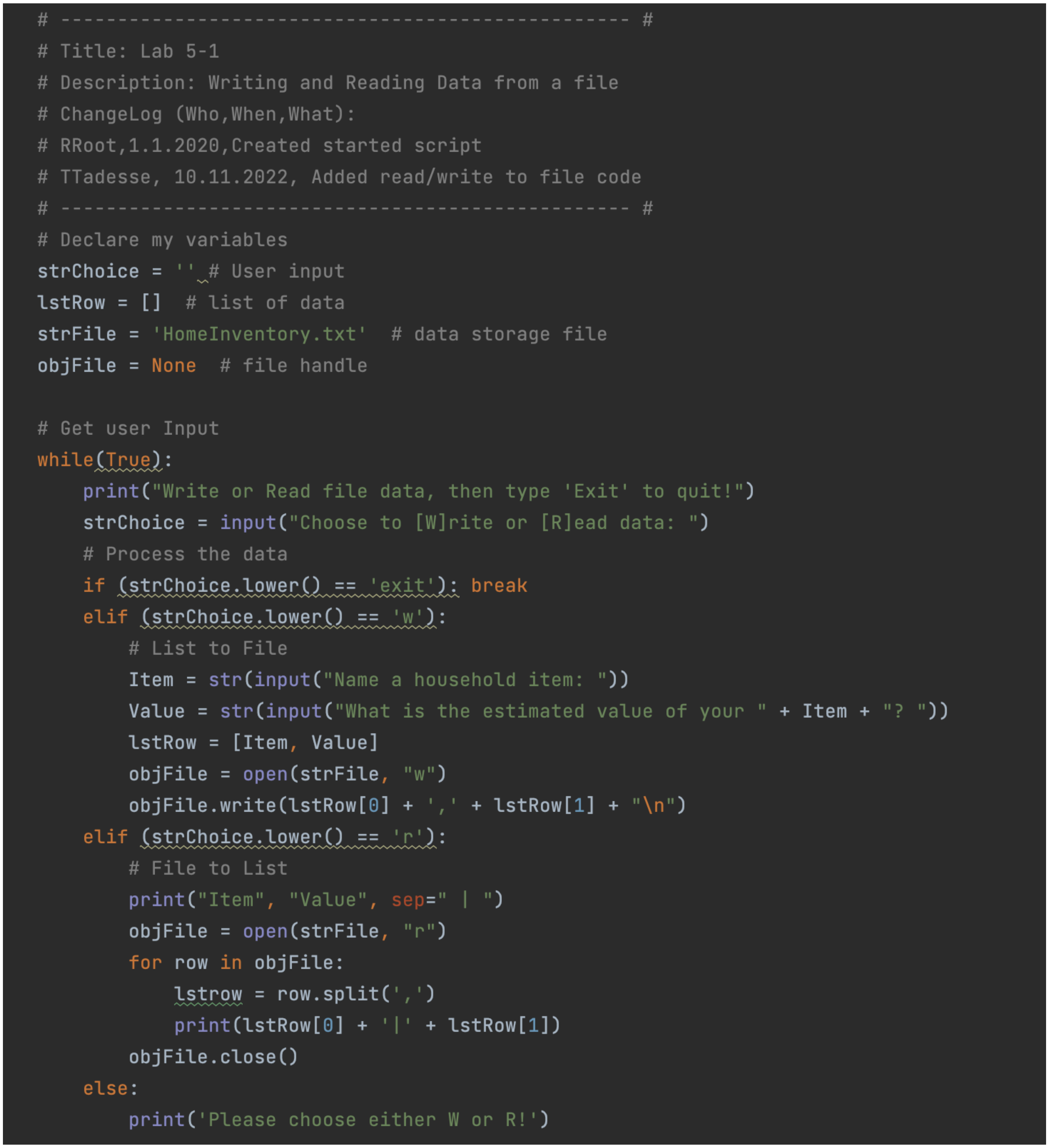
Introduction

In this assignment, I have learned more about lists, how to write them in a file, and read them from a file. I have also learned about another form of storing data in a single variable, dictionaries, which allows us to store data as a form of keys and values subscripts by using curly brackets. I have also practiced how to switch sequences from dictionaries to lists. I have also explored how to organize scripts using functions and comments as well as how to have a good practice of a professional setting by storing my data in a professionally and collaboratively organized environment.

Writing Lists to a File

In this module, I have learned more about writing data to a file and reading them in a different part of my script. This is achieved by using a combination of loops, inputs, and built in functions. For this practice I specifically used text files stored in the drive and later loaded in memory. ***Figure 1*** below is a solution to Lab 5-1 of module05 of the course. The lab prompts the user to write codes to *write* and *read* portions of the script by taking an input that allows the user to choose either to enter an input or to read the data saved in the file. Note that for this specific lab, the *write* function is in 'w' mode, meaning that whenever the user adds items and values to the list, the previous data is replaced by the new data and the *read* function only prints out the most recent data entry. I understand that the 'a' mode can be used to append all the data input, but for the purpose of this lab 'w' mode is used. And later in the assignment, I have learned a new way of writing a script to allow appending of data while having the *write* function on 'w' mode. (source: https://www.youtube.com/watch?v=m0o0CkYsDzI)

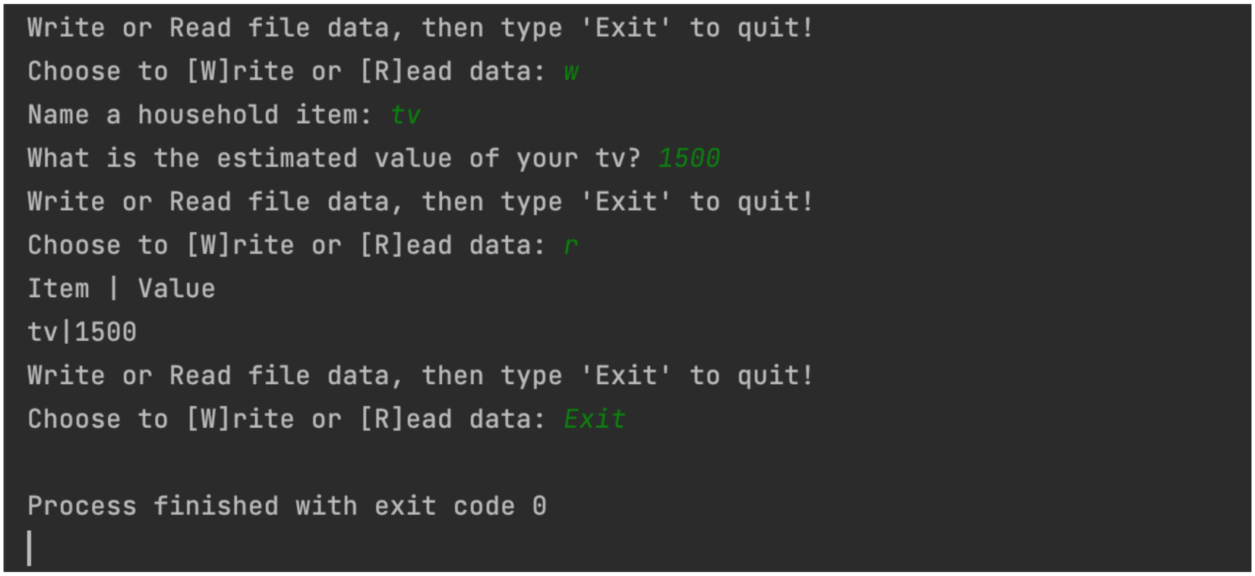
Another function that is used in the lab is the strip() function, which allows the program to read each element of the text file separated by a comma.



```
# -------------------------------------------------- #
# Title: Lab 5-1
# Description: Writing and Reading Data from a file
# ChangeLog (Who,When,What):
# RRoot,1.1.2020,Created started script
# TTadesse, 10.11.2022, Added read/write to file code
# -------------------------------------------------- #
# Declare my variables
strChoice = '' # User input
lstRow = []  # list of data
strFile = 'HomeInventory.txt'  # data storage file
objFile = None  # file handle

# Get user Input
while(True):
    print("Write or Read file data, then type 'Exit' to quit!")
    strChoice = input("Choose to [W]rite or [R]ead data: ")
    # Process the data
    if (strChoice.lower() == 'exit'): break
    elif (strChoice.lower() == 'w'):
        # List to File
        Item = str(input("Name a household item: "))
        Value = str(input("What is the estimated value of your " + Item + "? "))
        lstRow = [Item, Value]
        objFile = open(strFile, "w")
        objFile.write(lstRow[0] + ',' + lstRow[1] + "\n")
    elif (strChoice.lower() == 'r'):
        # File to List
        print("Item", "Value", sep=" | ")
        objFile = open(strFile, "r")
        for row in objFile:
            lstrow = row.split(',')
            print(lstRow[0] + '|' + lstRow[1])
        objFile.close()
    else:
        print('Please choose either W or R!')
```

***Figure 1**: Lab5-1 script*

As a result of the script in Figure 1, a user would input an item and a value for the item when they choose ‘w’ to write to file. And the program would return the most recent data when the user chooses ‘r’ to read the file. ***Figure 2*** shows the output of the script in ***FIgure 1.***

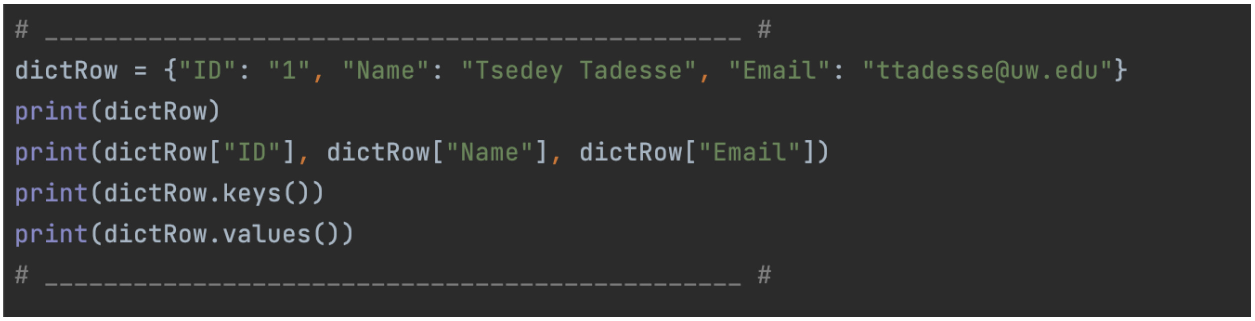


```
Write or Read file data, then type 'Exit' to quit!
Choose to [W]rite or [R]ead data: w
Name a household item: tv
What is the estimated value of your tv? 1500
Write or Read file data, then type 'Exit' to quit!
Choose to [W]rite or [R]ead data: r
Item | Value
tv|1500
Write or Read file data, then type 'Exit' to quit!
Choose to [W]rite or [R]ead data: Exit

Process finished with exit code 0
```

Figure 2**: result of script in **Figure 1

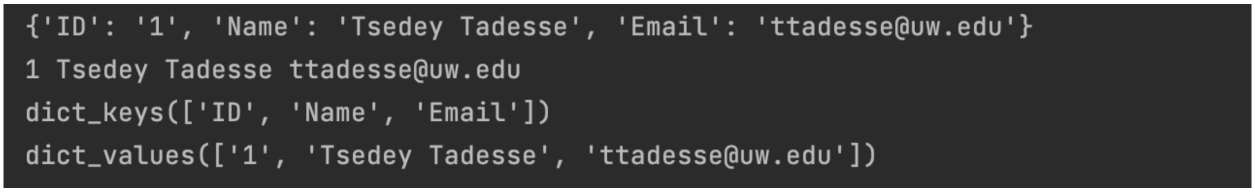
Dictionaries

Dictionaries are another way to store data in a single variable. As defined in the book, Dawson, michael, *Python Programming for the complete beginner, Third Edition*, Course Technology, 2010, "with dictionaries, you don't store information in a sequence; instead, you store it in pairs." As lists use index subscripts to identify each element of the list, dictionaries use keys and values subscripts. And dictionaries are distinguished by the use of curly brackets. ***Figure 3*** below shows how dictionaries are used in Python, how their subscripts are used in the print function, and how the keys and values subscripts are used to print each element separately. See **Figure 4** for the results of the script in ***Figure 3***.



```
# ---------------------------------------------- #
dictRow = {"ID": "1", "Name": "Tsedey Tadesse", "Email": "ttadesse@uw.edu"}
print(dictRow)
print(dictRow["ID"], dictRow["Name"], dictRow["Email"])
print(dictRow.keys())
print(dictRow.values())
# ---------------------------------------------- #
```

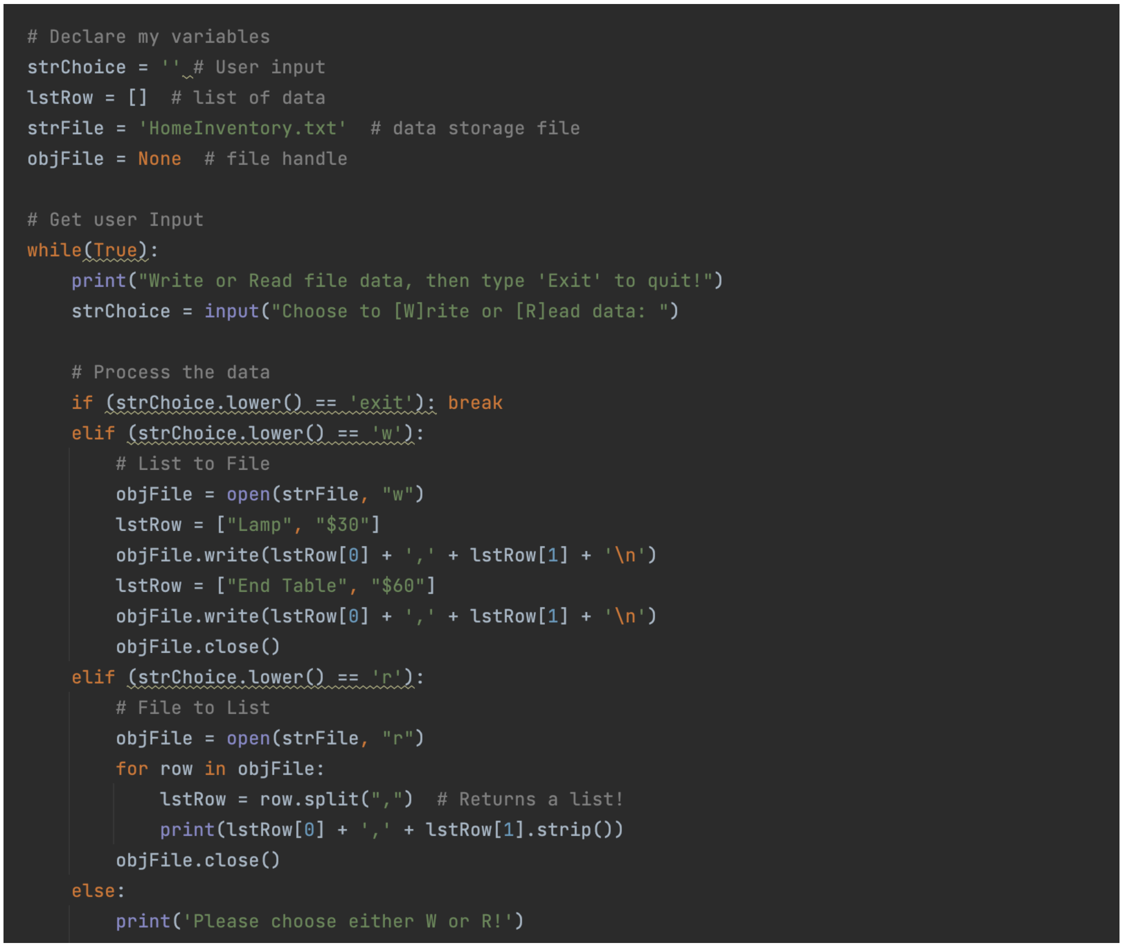
***Figure 3**: Dictionaries in Python*



```
{'ID': '1', 'Name': 'Tsedey Tadesse', 'Email': 'ttadesse@uw.edu'}
1 Tsedey Tadesse ttadesse@uw.edu
dict_keys(['ID', 'Name', 'Email'])
dict_values(['1', 'Tsedey Tadesse', 'ttadesse@uw.edu'])
```

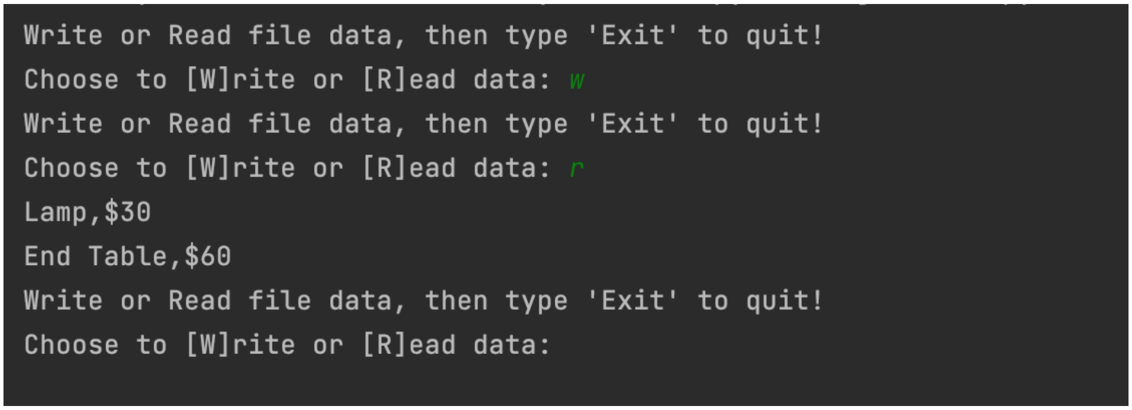
Figure 4**: result of script in **Figure 3

Like lists, dictionaries work the same way when writing data to a file and reading the data from a file. Lab 5-2 of modue05 asks to convert a script written using lists to one with dictionaries. ***Figure 5*** below shows the script that is written with lists along with the output of the program in ***Figure 6***.



```
# Declare my variables
strChoice = '' # User input
lstRow = []  # list of data
strFile = 'HomeInventory.txt'  # data storage file
objFile = None  # file handle

# Get user Input
while(True):
    print("Write or Read file data, then type 'Exit' to quit!")
    strChoice = input("Choose to [W]rite or [R]ead data: ")

    # Process the data
    if (strChoice.lower() == 'exit'): break
    elif (strChoice.lower() == 'w'):
        # List to File
        objFile = open(strFile, "w")
        lstRow = ["Lamp", "$30"]
        objFile.write(lstRow[0] + ',' + lstRow[1] + '\n')
        lstRow = ["End Table", "$60"]
        objFile.write(lstRow[0] + ',' + lstRow[1] + '\n')
        objFile.close()
    elif (strChoice.lower() == 'r'):
        # File to List
        objFile = open(strFile, "r")
        for row in objFile:
            lstRow = row.split(",")  # Returns a list!
            print(lstRow[0] + ',' + lstRow[1].strip())
        objFile.close()
    else:
        print('Please choose either W or R!')
```

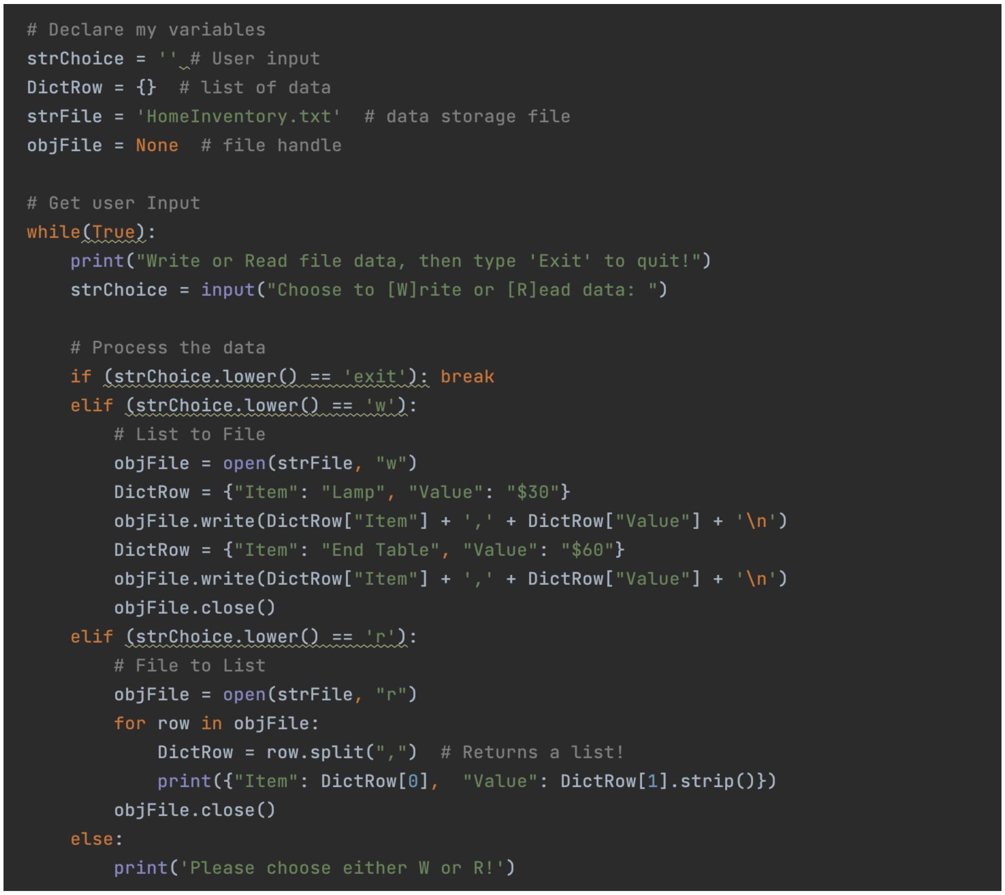
***Figure 5**: writing and reading lists in file*



```
Write or Read file data, then type 'Exit' to quit!
Choose to [W]rite or [R]ead data: w
Write or Read file data, then type 'Exit' to quit!
Choose to [W]rite or [R]ead data: r
Lamp,$30
End Table,$60
Write or Read file data, then type 'Exit' to quit!
Choose to [W]rite or [R]ead data:
```

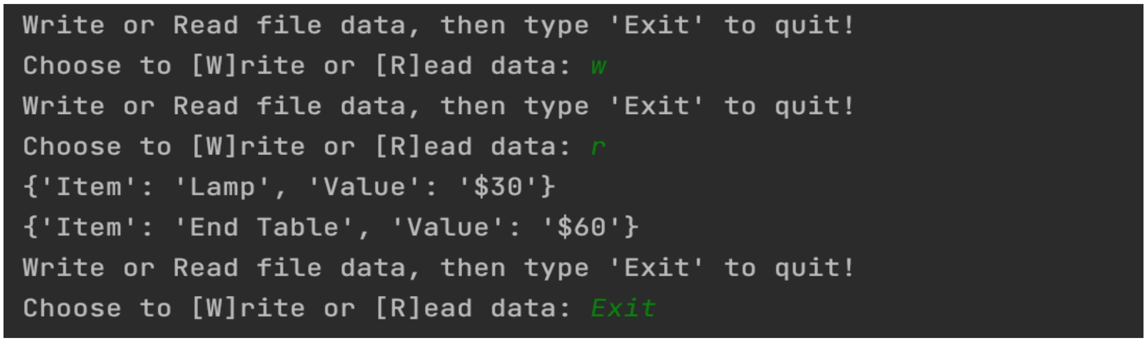
Figure 6*: result of script in* ***Figure 5***

The lists used in ***Figure 5*** can be converted to dictionaries and they work fairly the same way. The major difference is that lists use indexes to access elements of the data, but dictionaries use keys and values. ***Figure 7*** and ***Figure 8*** below shows how I updated the code in ***Figure 5*** so that dictionaries are used.



```python
# Declare my variables
strChoice = '' # User input
DictRow = {}  # list of data
strFile = 'HomeInventory.txt'  # data storage file
objFile = None  # file handle

# Get user Input
while(True):
    print("Write or Read file data, then type 'Exit' to quit!")
    strChoice = input("Choose to [W]rite or [R]ead data: ")

    # Process the data
    if (strChoice.lower() == 'exit'): break
    elif (strChoice.lower() == 'w'):
        # List to File
        objFile = open(strFile, "w")
        DictRow = {"Item": "Lamp", "Value": "$30"}
        objFile.write(DictRow["Item"] + ',' + DictRow["Value"] + '\n')
        DictRow = {"Item": "End Table", "Value": "$60"}
        objFile.write(DictRow["Item"] + ',' + DictRow["Value"] + '\n')
        objFile.close()
    elif (strChoice.lower() == 'r'):
        # File to List
        objFile = open(strFile, "r")
        for row in objFile:
            DictRow = row.split(",")  # Returns a list!
            print({"Item": DictRow[0],  "Value": DictRow[1].strip()})
        objFile.close()
    else:
        print('Please choose either W or R!')
```

Figure 7*: writing and reading lists in file*



```
Write or Read file data, then type 'Exit' to quit!
Choose to [W]rite or [R]ead data: w
Write or Read file data, then type 'Exit' to quit!
Choose to [W]rite or [R]ead data: r
{'Item': 'Lamp', 'Value': '$30'}
{'Item': 'End Table', 'Value': '$60'}
Write or Read file data, then type 'Exit' to quit!
Choose to [W]rite or [R]ead data: Exit
```

Figure 8**: result of script in **Figure

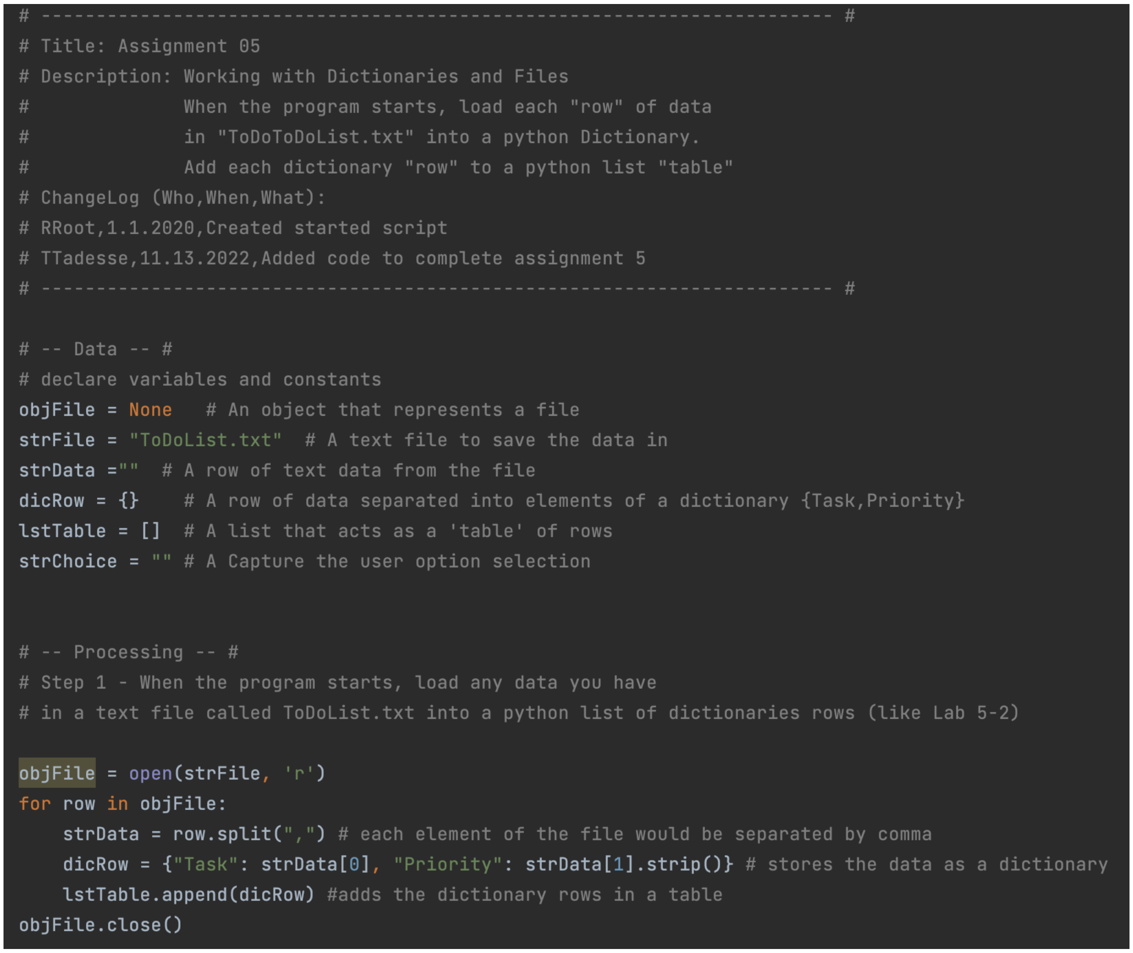
Note that the strip() function is used in Lab 5-2 to remove unwanted carriage returns in the lists/dictionaries. The result of the script in ***Figure 7*** shows the output of dictionaries. It is also possible to convert that result into any form of sequence, which would be shown in Assignment05 script.

Assignment05

Assigment05 prompts me to write a script that creates a table of "ToDo list" for the user. The script needs to have a write and read function of data saved as a text file in the drive. For this assignment, I was provided with a starter script that structures the whole script through separation of concerts, where the script is separated into distinct sections with specific tasks. For this assignment, the script is sectioned as data, processing, and I/O.

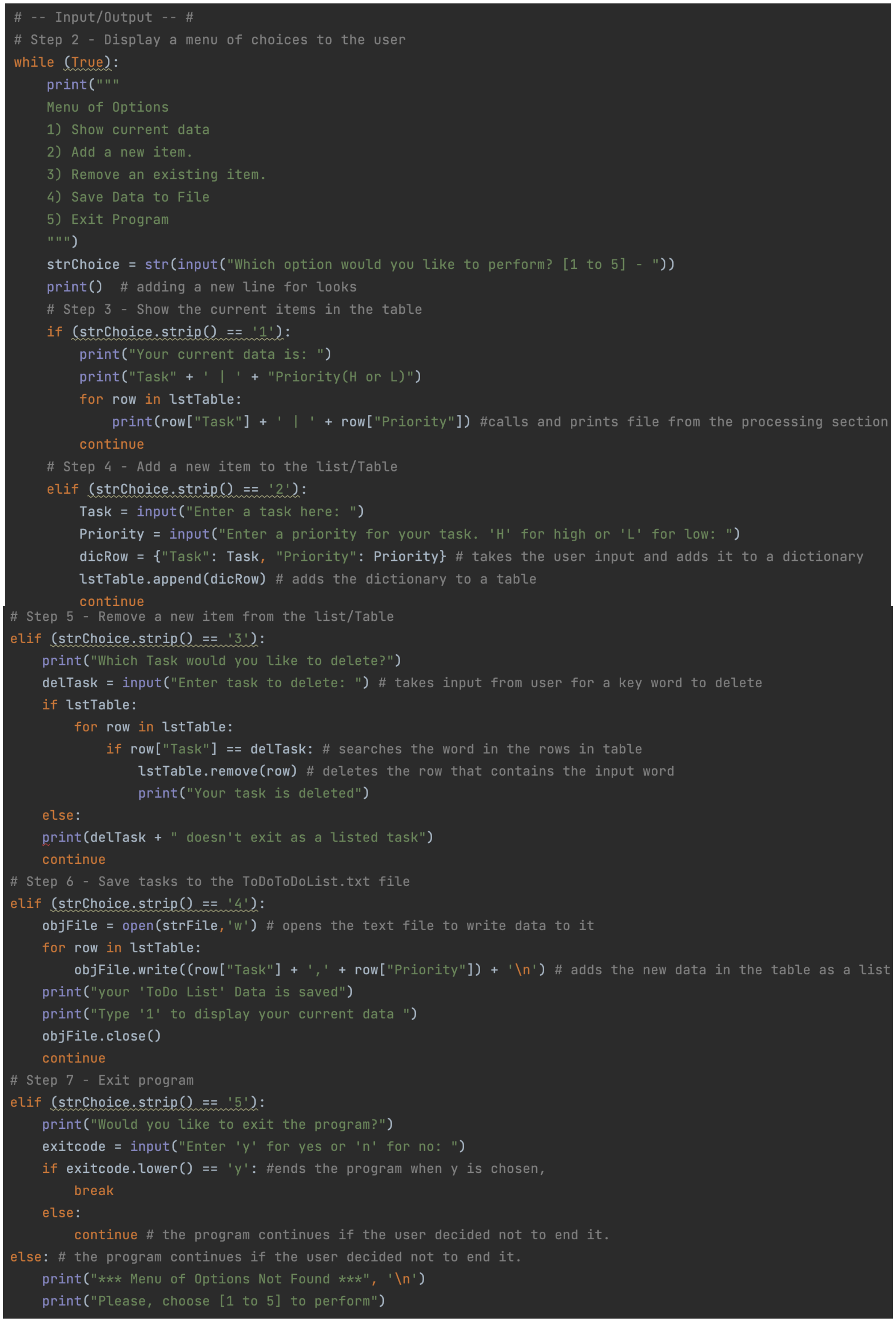
I started working on the assignment by modifying the starter script to add my name into it and change/add variable names in the data section. Then I worked on the first part of the code, where I was asked to write a code that when the program starts, it opens and reads any stored data in the text file. This is accomplished by opening the file in 'r' mode and adding a *for* loop to capture data, add it to a dictionary, and then convert it to a list file and add it to a table of data. See ***Figure 9-1*** to see the code.

In the next step, I worked on the I/O section, which has its own subsections embedded in a *while* loop. The program asks the user to choose an option from 1-5, where each option is used as a condition in if-elif-else statements to perform a task. See ***Figure 9*** to see each subsection of the code. The first subsection, Step 3, displays the data stored so far. This is accomplished by creating a *for* loop that calls the table created in the processing section of the script. Then, the next subsection, Step 4, allows the user to input data and the program takes the data and appends it to the table. The script in Step 5 allows the user to choose what they want to remove from the "ToDo List." For this step, I first created a *for* loop nested with an *if* statement to search for a word that the user inputs in order to delete the row of data where that word is contained in the table. My program loops through each row of data in the table and deletes the row where that input word is contained, but it doesn't stop there. It outputs the else statement for the other rows that don't contain the word, which is not a result that I wanted to see. So to solve this problem, I nested all what wrote so far under another *if* statement and the program runs as wanted. Step 6 asked to save the data to file. And I accomplished that by opening the text file in 'w' mode and I wrote lines of code to save the data as a list in the table. The last part of the program allows the user to exit the program. See ***Figure 9*** to see the entirety of the script.



```
# ------------------------------------------------------------------------ #
# Title: Assignment 05
# Description: Working with Dictionaries and Files
#              When the program starts, load each "row" of data
#              in "ToDoToDoList.txt" into a python Dictionary.
#              Add each dictionary "row" to a python list "table"
# ChangeLog (Who,When,What):
# RRoot,1.1.2020,Created started script
# TTadesse,11.13.2022,Added code to complete assignment 5
# ------------------------------------------------------------------------ #

# -- Data -- #
# declare variables and constants
objFile = None   # An object that represents a file
strFile = "ToDoList.txt"  # A text file to save the data in
strData =""  # A row of text data from the file
dicRow = {}    # A row of data separated into elements of a dictionary {Task,Priority}
lstTable = []  # A list that acts as a 'table' of rows
strChoice = "" # A Capture the user option selection


# -- Processing -- #
# Step 1 - When the program starts, load any data you have
# in a text file called ToDoList.txt into a python list of dictionaries rows (like Lab 5-2)

objFile = open(strFile, 'r')
for row in objFile:
    strData = row.split(",") # each element of the file would be separated by comma
    dicRow = {"Task": strData[0], "Priority": strData[1].strip()} # stores the data as a dictionary
    lstTable.append(dicRow) #adds the dictionary rows in a table
objFile.close()
```

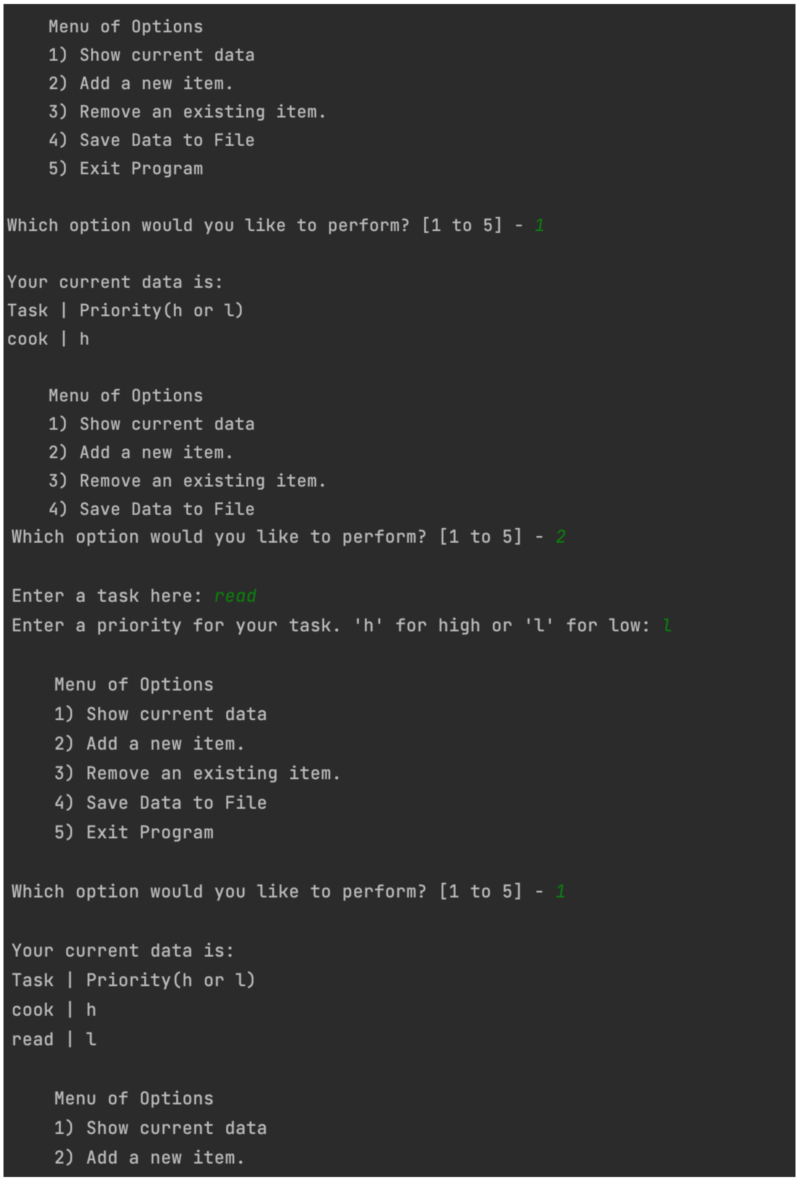
Figure 9-1*: assignment05 script*



```
# -- Input/Output -- #
# Step 2 - Display a menu of choices to the user
while (True):
    print("""
    Menu of Options
    1) Show current data
    2) Add a new item.
    3) Remove an existing item.
    4) Save Data to File
    5) Exit Program
    """)
    strChoice = str(input("Which option would you like to perform? [1 to 5] - "))
    print()  # adding a new line for looks
    # Step 3 - Show the current items in the table
    if (strChoice.strip() == '1'):
        print("Your current data is: ")
        print("Task" + ' | ' + "Priority(H or L)")
        for row in lstTable:
            print(row["Task"] + ' | ' + row["Priority"]) #calls and prints file from the processing section
        continue
    # Step 4 - Add a new item to the list/Table
    elif (strChoice.strip() == '2'):
        Task = input("Enter a task here: ")
        Priority = input("Enter a priority for your task. 'H' for high or 'L' for low: ")
        dicRow = {"Task": Task, "Priority": Priority} # takes the user input and adds it to a dictionary
        lstTable.append(dicRow) # adds the dictionary to a table
        continue
# Step 5 - Remove a new item from the list/Table
elif (strChoice.strip() == '3'):
    print("Which Task would you like to delete?")
    delTask = input("Enter task to delete: ") # takes input from user for a key word to delete
    if lstTable:
        for row in lstTable:
            if row["Task"] == delTask: # searches the word in the rows in table
                lstTable.remove(row) # deletes the row that contains the input word
                print("Your task is deleted")
    else:
    print(delTask + " doesn't exit as a listed task")
    continue
# Step 6 - Save tasks to the ToDoToDoList.txt file
elif (strChoice.strip() == '4'):
    objFile = open(strFile,'w') # opens the text file to write data to it
    for row in lstTable:
        objFile.write((row["Task"] + ',' + row["Priority"]) + '\n') # adds the new data in the table as a list
    print("your 'ToDo List' Data is saved")
    print("Type '1' to display your current data ")
    objFile.close()
    continue
# Step 7 - Exit program
elif (strChoice.strip() == '5'):
    print("Would you like to exit the program?")
    exitcode = input("Enter 'y' for yes or 'n' for no: ")
    if exitcode.lower() == 'y': #ends the program when y is chosen,
        break
    else:
        continue # the program continues if the user decided not to end it.
else: # the program continues if the user decided not to end it.
    print("*** Menu of Options Not Found ***", '\n')
    print("Please, choose [1 to 5] to perform")
```

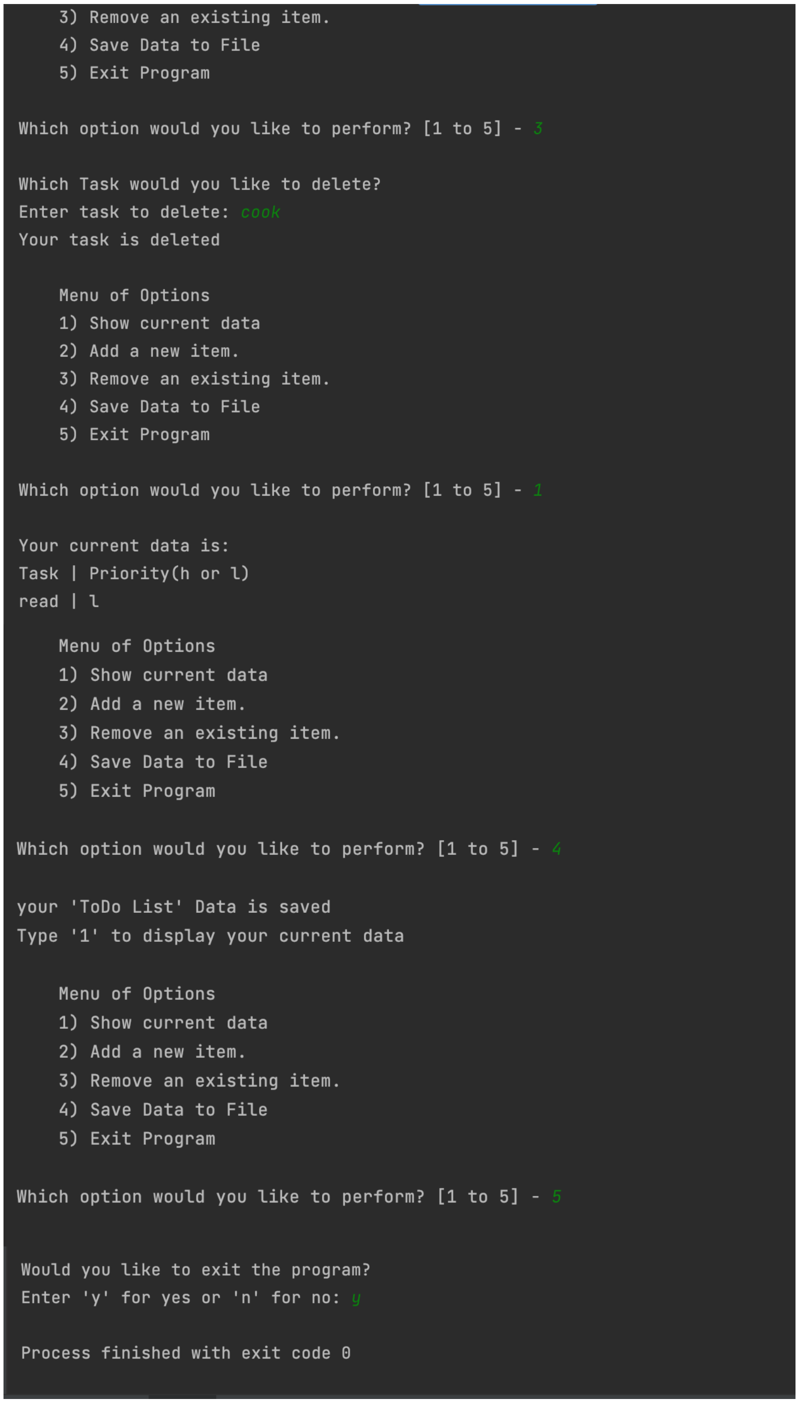
***Figure 9-2**: assignment05 script*

Figure 10 shows the result of assignment05 script as it is run on Pycharm and ***Figure 11*** shows the result run on the terminal.



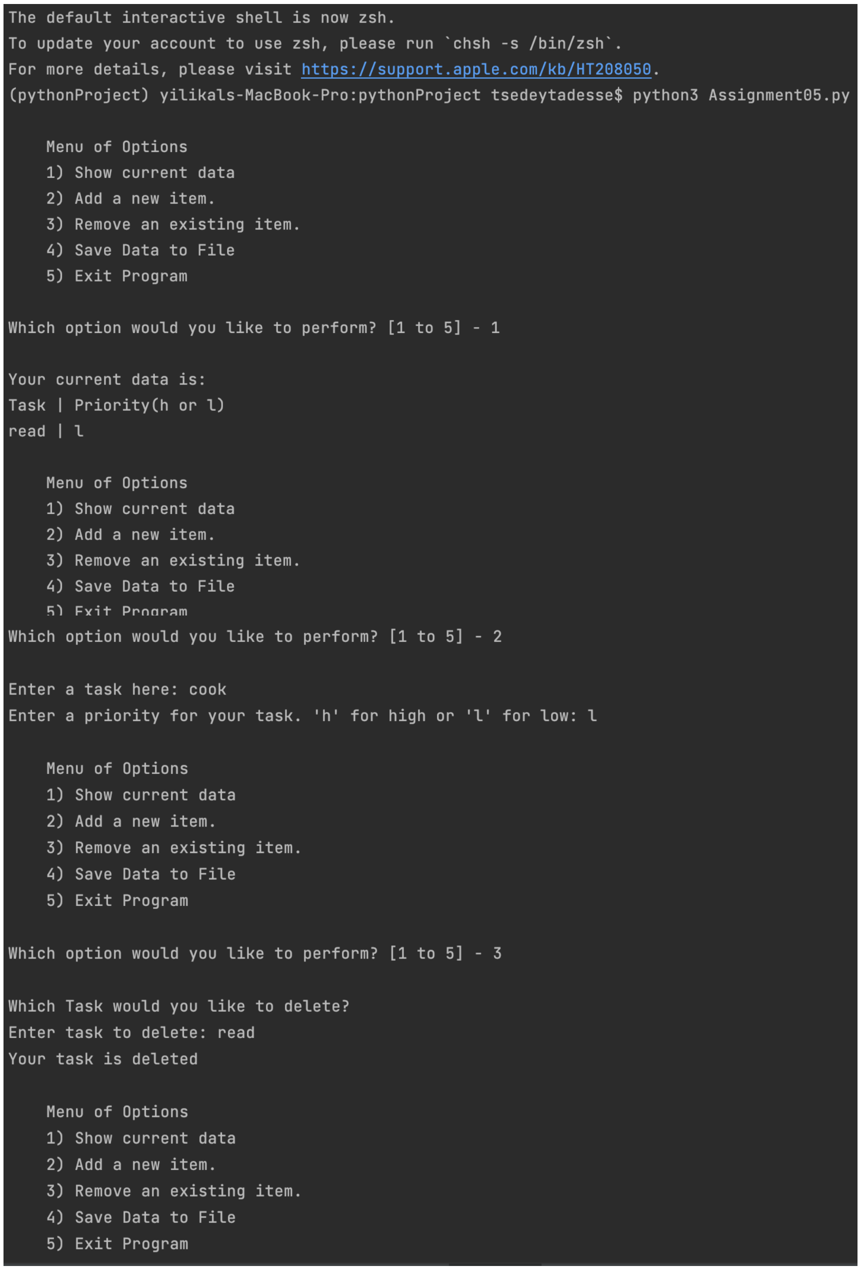
```
    Menu of Options
    1) Show current data
    2) Add a new item.
    3) Remove an existing item.
    4) Save Data to File
    5) Exit Program

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

Your current data is:
Task | Priority(h or l)
cook | h

    Menu of Options
    1) Show current data
    2) Add a new item.
    3) Remove an existing item.
    4) Save Data to File
Which option would you like to perform? [1 to 5] - 2

Enter a task here: read
Enter a priority for your task. 'h' for high or 'l' for low: l

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

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

Your current data is:
Task | Priority(h or l)
cook | h
read | l

    Menu of Options
    1) Show current data
    2) Add a new item.
```

Figure 10-1*: result of assignment script in PyCharm*



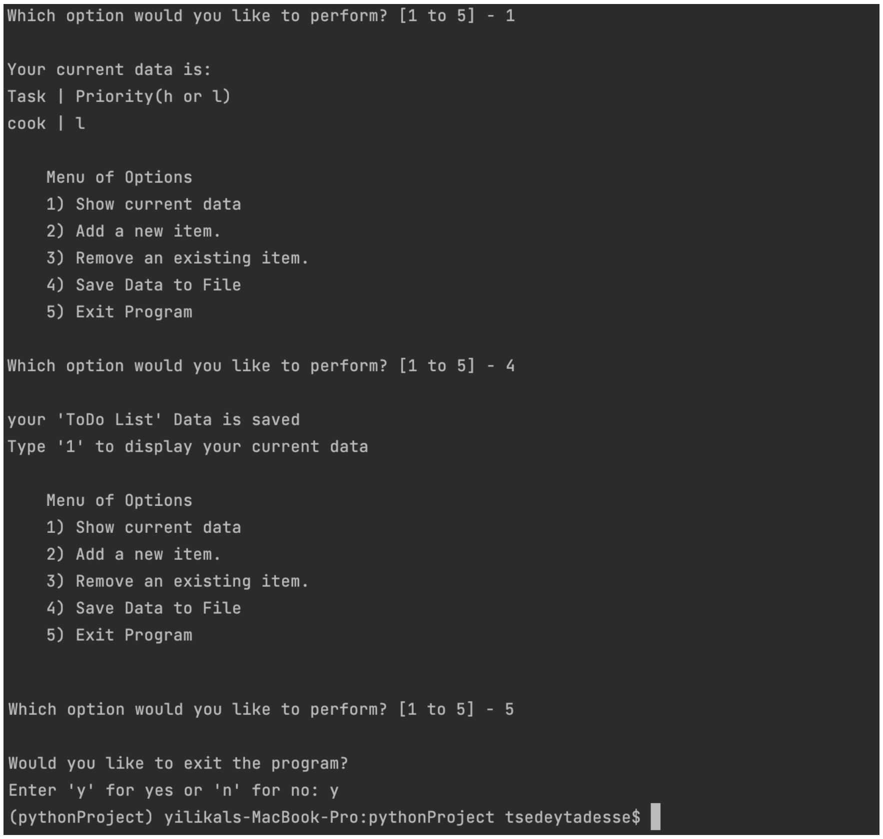
```
    3) Remove an existing item.
    4) Save Data to File
    5) Exit Program

Which option would you like to perform? [1 to 5] - 3

Which Task would you like to delete?
Enter task to delete: cook
Your task is deleted

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

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

Your current data is:
Task | Priority(h or l)
read | l

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

Which option would you like to perform? [1 to 5] - 4

your 'ToDo List' Data is saved
Type '1' to display your current data

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

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

Would you like to exit the program?
Enter 'y' for yes or 'n' for no: y

Process finished with exit code 0
```

Figure 10-2*: result of assignment script in PyCharm*



```
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(pythonProject) yilikals-MacBook-Pro:pythonProject tsedeytadesse$ python3 Assignment05.py

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

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

Your current data is:
Task | Priority(h or l)
read | l

    Menu of Options
    1) Show current data
    2) Add a new item.
    3) Remove an existing item.
    4) Save Data to File
    5) Exit Program
Which option would you like to perform? [1 to 5] - 2

Enter a task here: cook
Enter a priority for your task. 'h' for high or 'l' for low: l

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

Which option would you like to perform? [1 to 5] - 3

Which Task would you like to delete?
Enter task to delete: read
Your task is deleted

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

Figure 11-1*: result of assignment script in Terminal*



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

Your current data is:
Task | Priority(h or l)
cook | l

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

Which option would you like to perform? [1 to 5] - 4

your 'ToDo List' Data is saved
Type '1' to display your current data

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


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

Would you like to exit the program?
Enter 'y' for yes or 'n' for no: y
(pythonProject) yilikals-MacBook-Pro:pythonProject tsedeytadesse$
```

Figure 11-2*: result of assignment script in Terminal*

Summary

In module05, with the help of the class notes, the course video, the assigned textbook reading, the supplemental video and website, I was able to learn more about lists and how to write and read list data in a file. I have also learned about dictionaries and their distinguishing keys and values subscripts. I have worked on assignment05 by writing a more professionally organized format that uses the starter script provided with the assignment. I understood how the separation of concerns technique is crucial in organizing codes in a script.