Susan Paulen

November 16, 2020

IT FDN 110 A

Assignment 05

GitHub: https://github.com/spaulen/IntroToProg-Python/

# The Home Inventory Program (Part 3)

# Introduction

This week, we learned about creating and manipulating dictionary objects as well as the difference between a list and a dictionary. Indices and Keys are one of the main differences between the objects. Methods of reading data from a file into a list and a dictionary are very similar. We also covered the programming pattern called “Separations of Concerns”, Script Templates, Error Handling as well as how functions can be used to organize code. Finally, we learned how to set ourselves up in GitHub in order to share code and maintain code versions.

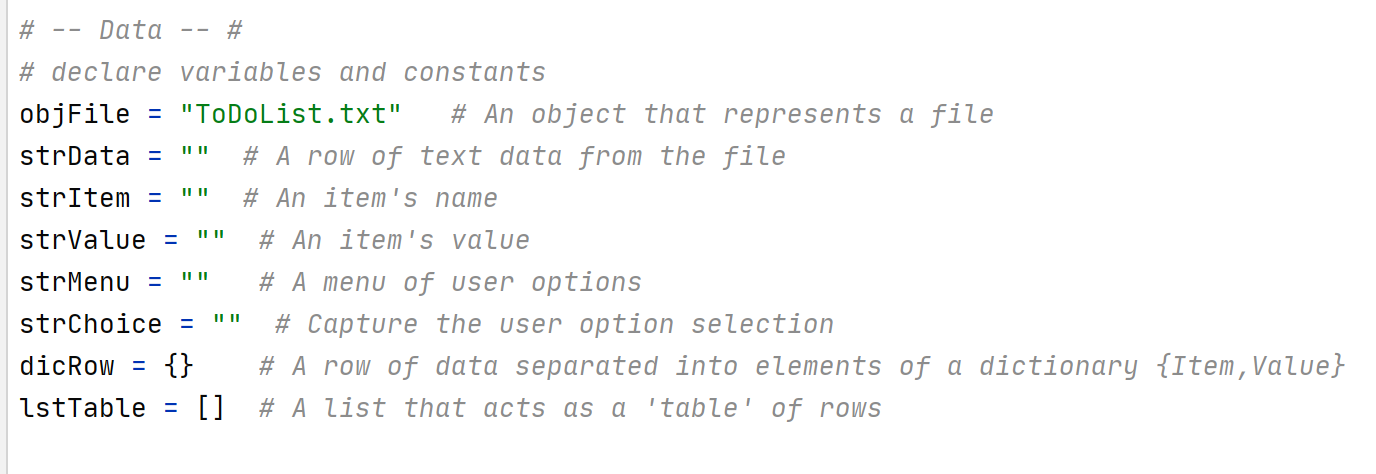
# Variable Declarations

As a best practice; file objects, strings, lists, list tables and dictionary rows are declared at the beginning of the Python code for this assignment. I plan to use dictionary rows and lists as part of this program to practice and better understand their attributes. Both the List Table (lstTable) and the Dictionary Row (dicRow) should be declared before usage to avoid errors.

Explanations for each variable and constant are provided in Python comments identified by the hashtag below in Figure 1.1.

Note that lstTable is used to contain all changes made to the ToDoList.txt data and will be saved over the ToDoList.txt file at the end of a user session if the user chooses to do so.

Figure 1.1 Variable and Constant Declaration



## List Table Creation via Dictionary Row (Step #1)

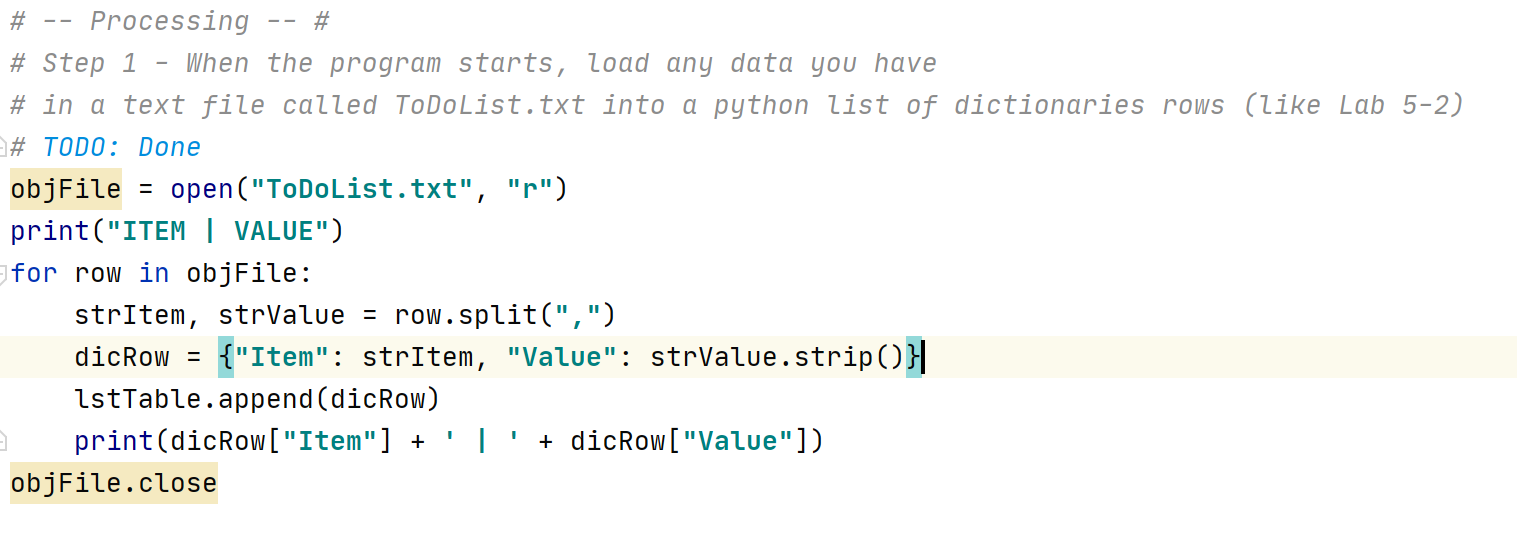
The ToDoList.txt file is opened as read only because data modifications are collected within the program in lstTable. I want to save the initial data to lstTable so that its rows are modified with any changes (additions or removals) made by the user during their program session. At the end of the program session, the user may decide whether they wish to save their changes or exit without saving changes. This avoids opening, deleting and appending the file which could impact program performance.

A header for the data is included before the “for loop”. This is due to the “for loop” iterating through each row in the file. I only want the header displayed once.

The for loop iterates through each row in the ToDoList.txt file and performs the following:

1. Each row’s data is split into the two string variables: strItem and strValue. A comma indicates the division between the two pieces of data on each row.
2. These two string variables are then used to create a dictionary row (dicRow) – assigning “Item” and “Value” as keys to indicate the two separate pieces of data. The second piece of data has the carriage return stripped off.
3. The dictionary row is then appended to the lstTable. The lstTable will contain all of the rows from the ToDoList.txt file once the for loop completes.
4. I left the print statement of the current dictionary row to demonstrate that all of the data is being collected. It provides a starting place for the user too. If this was going to production, I would remove the print row and expect the user to select Option 1 of the Menu to Show Current Data.

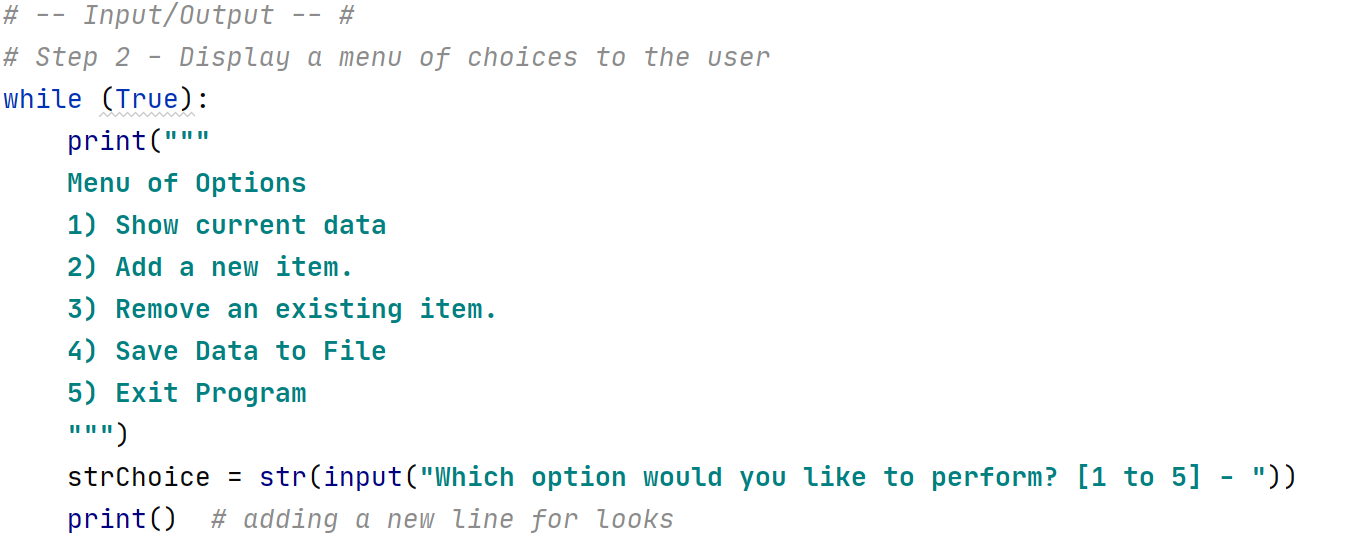
Figure 2.1 Step 1 - Load ToDoList.txt to lstTable



# While Loop, Option Menu, Capturing User Selection and Conditional Statement (Step #2)

The while loop, option menu, capturing the user and the Conditional Statement Options was created by the prior programmer – RRoot.

Figure 3.1 Step 2 - Prior Programmer's Code (RRoot)

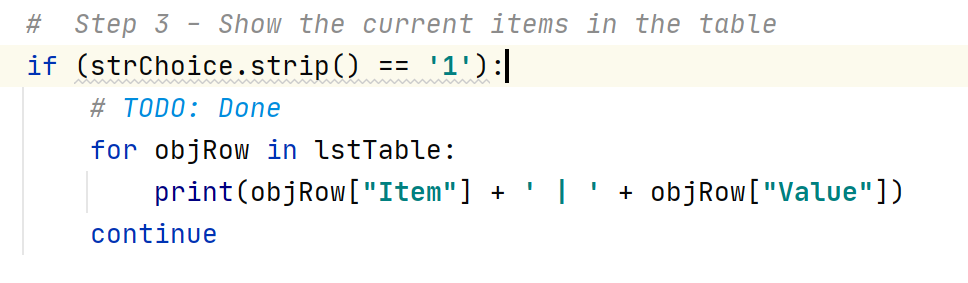


# Conditional Statement Options

## Menu Option 1: Show Current Data (Step #3)

If the user selects Option 1, the data in the lstTable (from the User’s Current Session) will be displayed to the screen. Using the lstTable (two-dimensional array), I can loop through the rows in the lstTable and unpack the Items and Values via a print statement to the screen.

Figure 4.1 Step 3 - Show Current Data



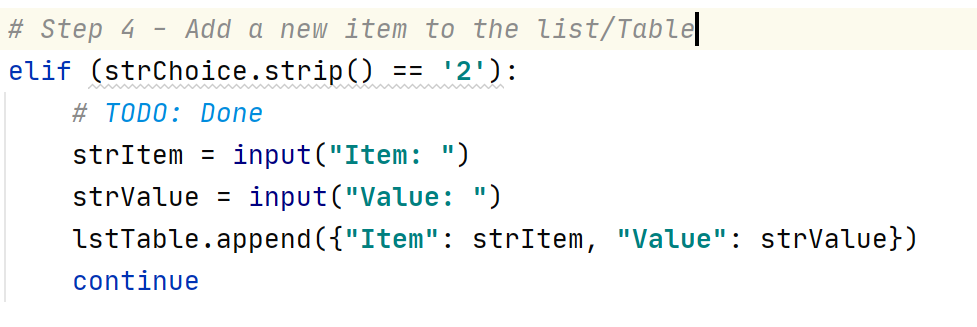
The continue statement is not required but is nice to have as it indicates that this conditional statement is in a loop that will continue to operate once the data is displayed.

## Menu Option 2: Add a New Item (Step #4)

If the user selects Option 2, I need to request the name and value of the household item to add to the lstTable. Using the variables declared at the beginning of the program, the two strings, strName and strValue, will contain the Household Item Name and the Household Item Value respectively.

These two strings are then packed into lstRow which is essentially a list array. The lstRow is then added to the variable lstTable which is a List Table array. This creates a two-dimensional table that will be added to each time the user chooses option 1 until the user chooses to exit.

Figure 5.1 Step 4 - Add a New Item

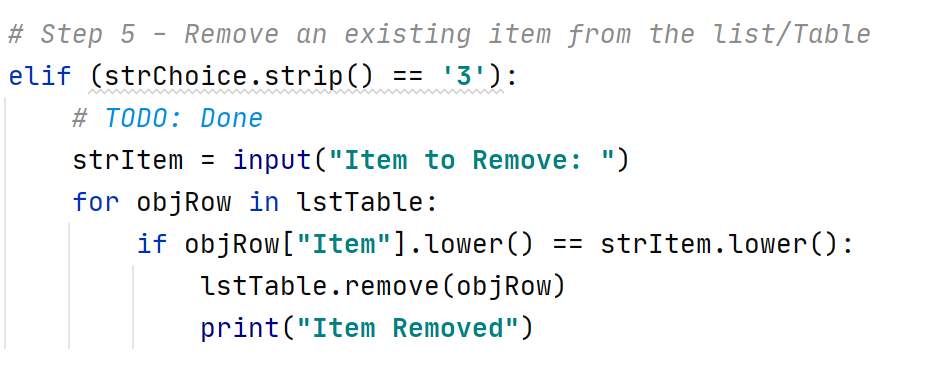


## Menu Option 3: Remove an Existing Item

If the user selects Option 3, the user will enter the item to be removed into the input variable strItem. This is most easily accomplished using a For Loop that loops through each row in the lstTable object. The case agnostic strItem is compared with every objRow["Item"]. When the row is found, it is removed from the lstTable, and the user is messaged “Item Removed”. If the item is not found, nothing happens.

This item removal will not be updated to the ToDoList.txt file until the user choses to save their work by choosing menu option 4. The user should choose menu option 1 to Show Current Data to determine if the data is accurate.

Figure 6.1 Step 5 = Remove an Existing Item



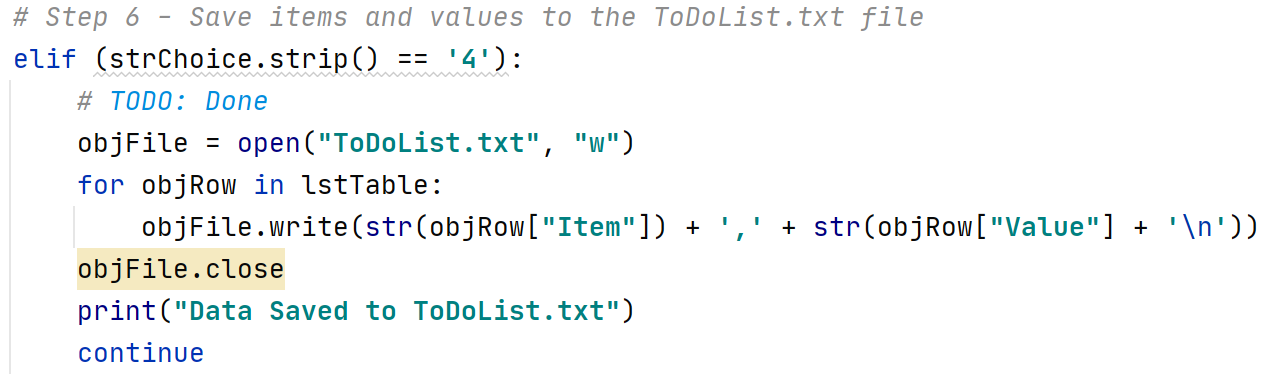
## Menu Option 4: Save Data to File

If the user selects Option 4, I need to write the data collected in lstTable to the file named ToDoList.txt. This file is saved in the same directory as the Assignment05\_Starter.py program.

Opening, writing and closing the file object named ToDoList.txt is similar to Assignment #4 where the data was written from the lstTable by row into the file object. Looping through each row in the lstTable object, the list array is packaged up by referencing the household item with objRow[“Item”] and the household value with objRow[“Value”].

Once the data has been saved, a notice is sent to the user, “Data Saved to ToDoList.txt”. The user then continues their session by choosing another option.

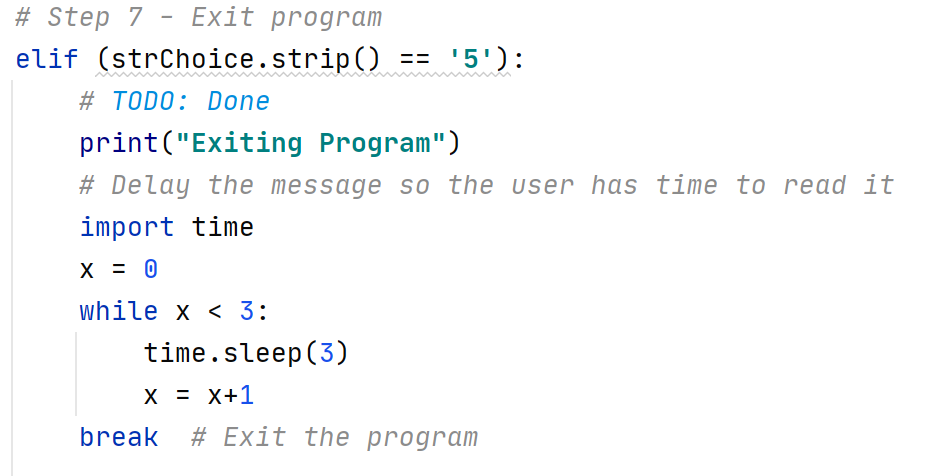
Figure 7.1 Step 6 - Save Data to File



## Menu Option 5: Exit Program (Step #7)

The last option, Option 5, allows the user to exit the program. I display the message “Exiting Program” and add a short time delay so the user has time to see the message.

Figure 8.1 Step 7 - Exit Program



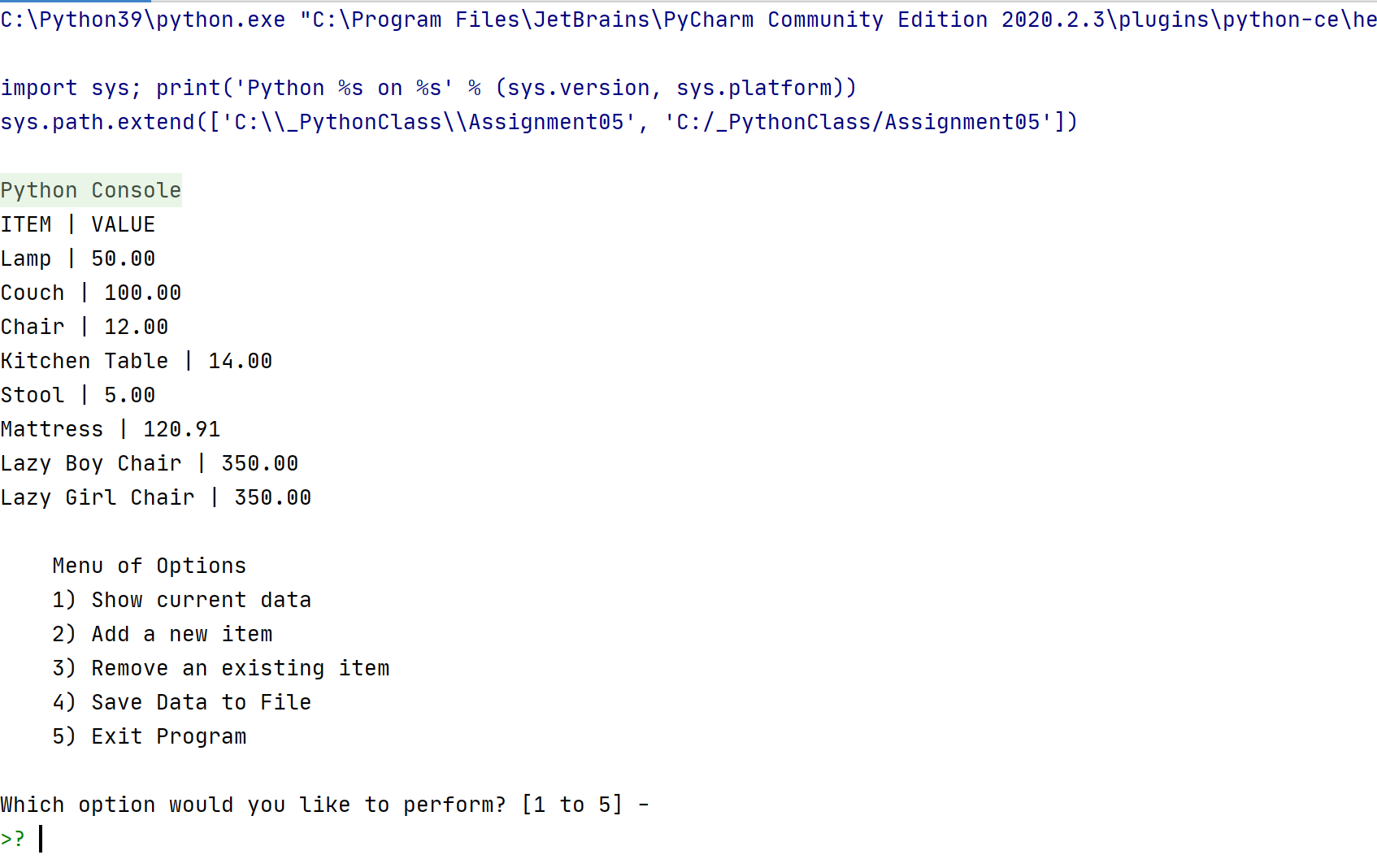
# Testing

After saving the program code to HomeInventory.py, I need to test that the program works as per the requirements.

#### **Test 1 – PyCharm Test**

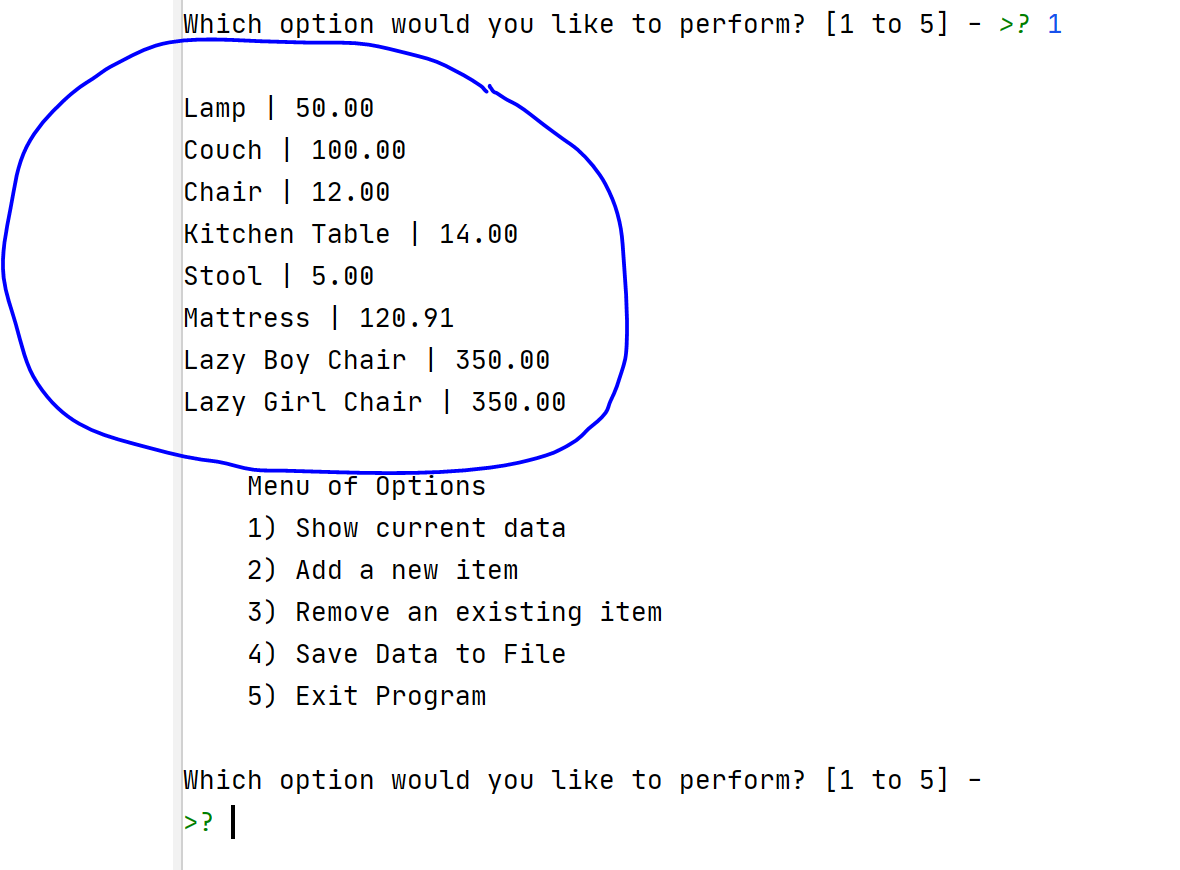
I choose Run in PyCharm and the program provides:

1. A list of items saved from the ToDoList.txt is displayed, The Menu of Options appears, and a request to select an option is shown.



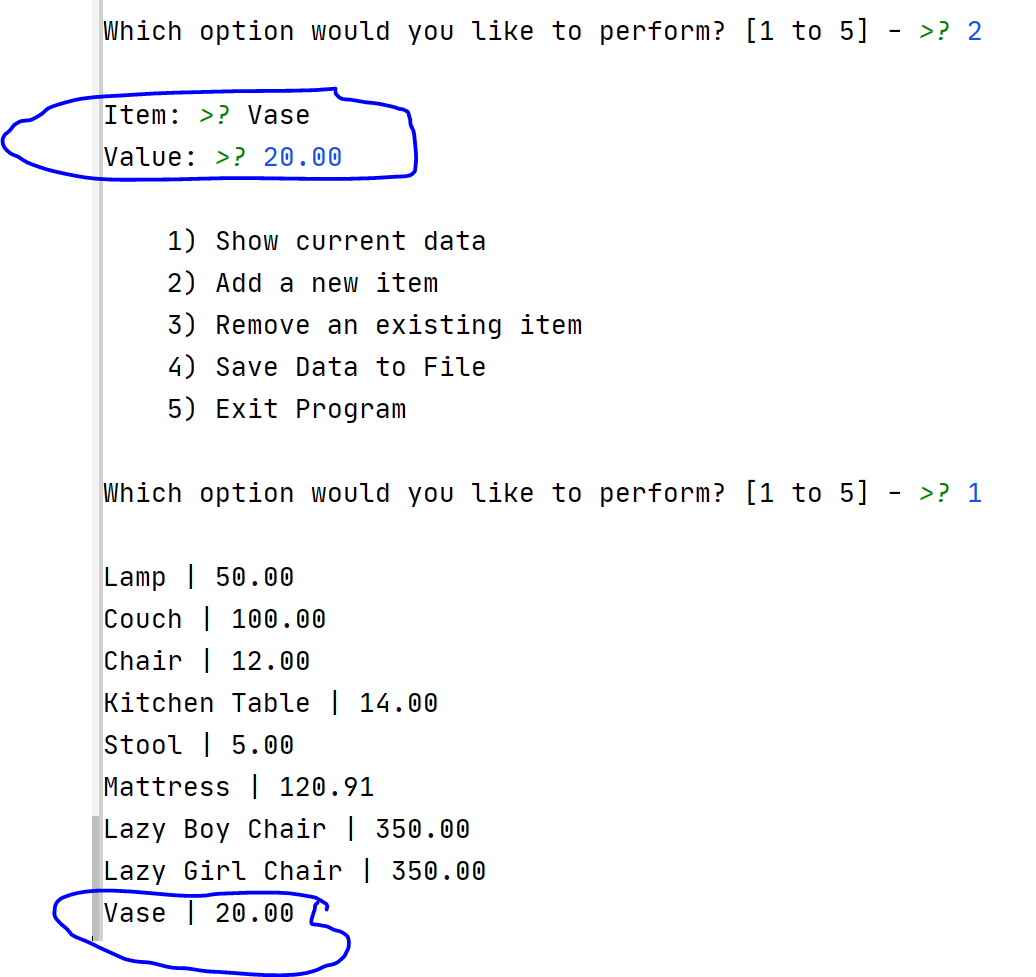
1. When Option 1 is selected, the Current Data in the lstTable is shown

This shows eight entries after choosing Option 1.

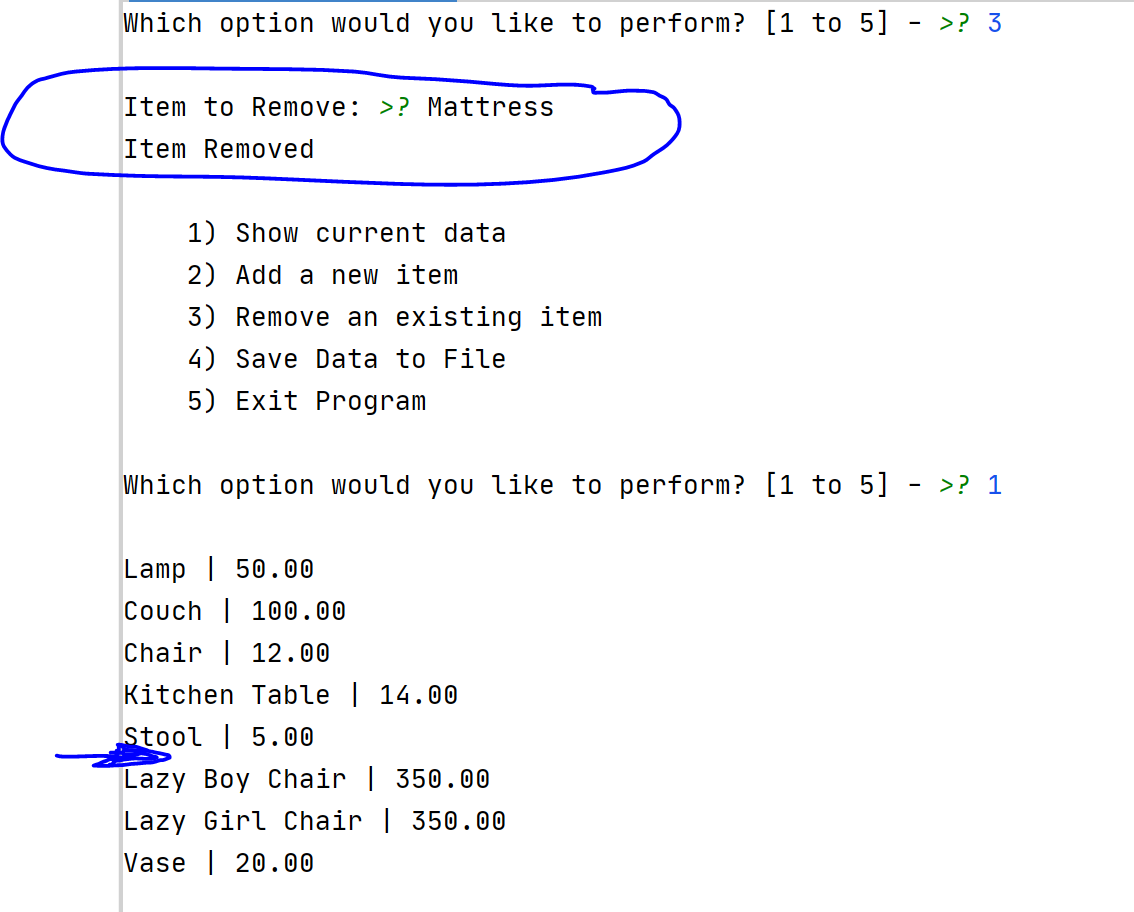


1. When Option 2 is selected, the User is prompted for an Item and a Value.

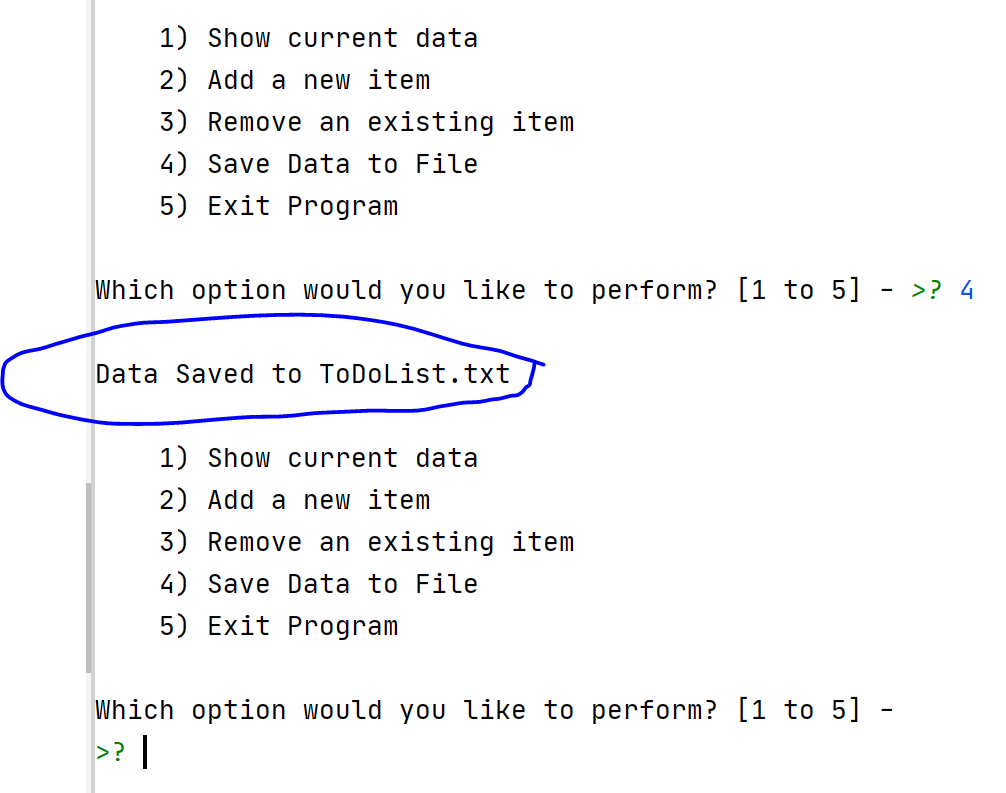
Selecting Option 1 after shows that the Vase is appended to lstTable.



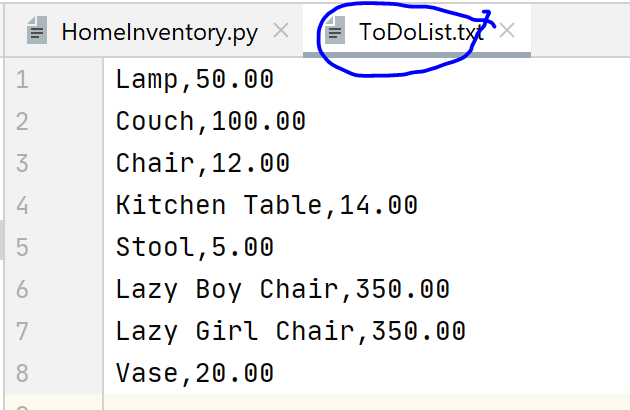
1. When Option 3 is selected, the User is prompted for an Item to Remove (Mattress). This item is found in the lstTable and removed. Mattress used to be between Stool and Lazy Boy Chair.



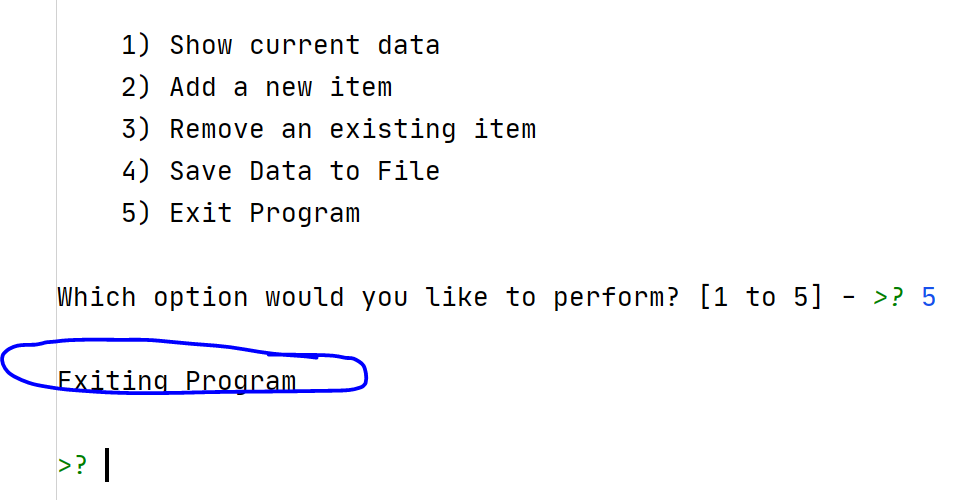
1. When Option 4 is selected:
   1. The user is informed that the data has been saved



* 1. The data from the lstTable is saved to the ToDoList.txt file in the same location as the Assignment05\_Starter.py



1. When Option 5 is selected, the Program displays “Exiting Program” and Exits.

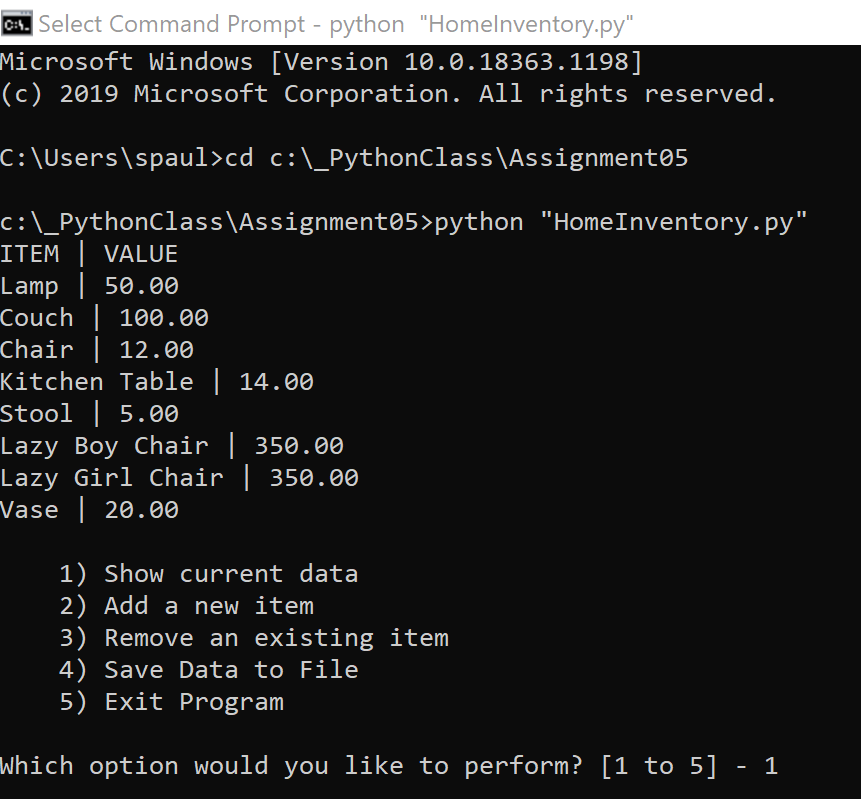


The Home Inventory program requirements have been met for PyCharm.

**Test 2 – OS Command Test**

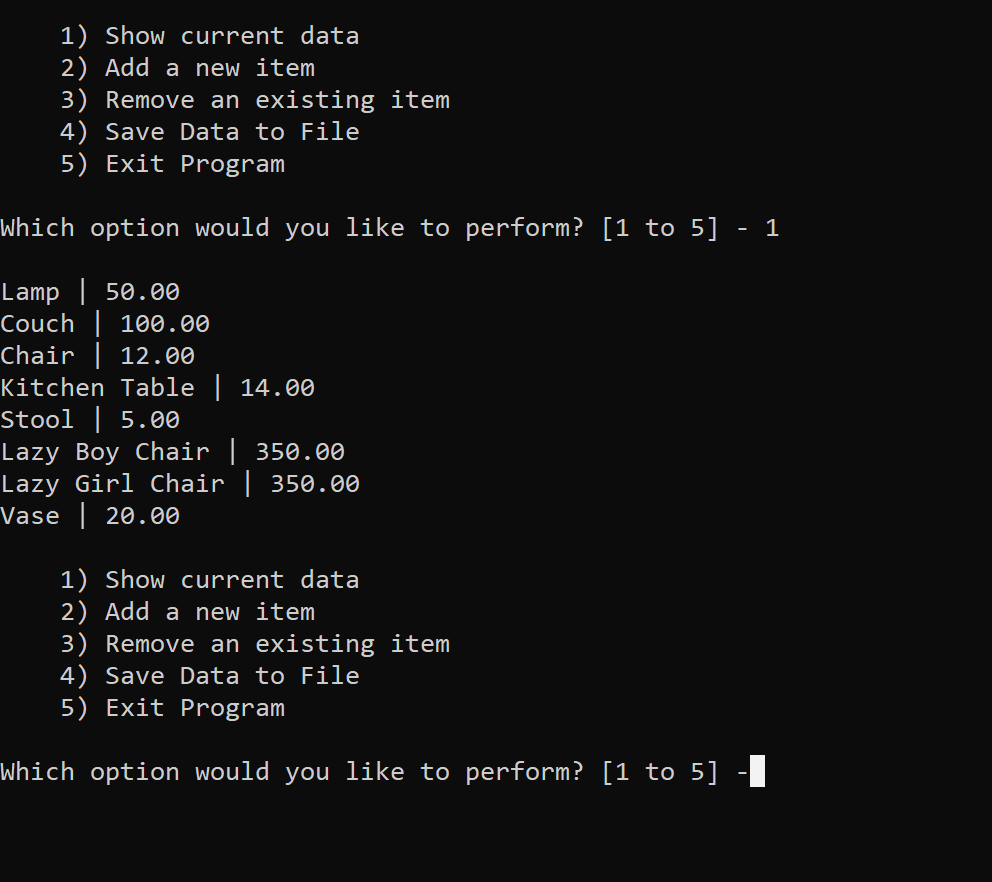
I start the program in the Command Shell and the program provides:

1. A list of items saved from the ToDoList.txt is displayed, The Menu of Options appears, and a request to select an option is shown.

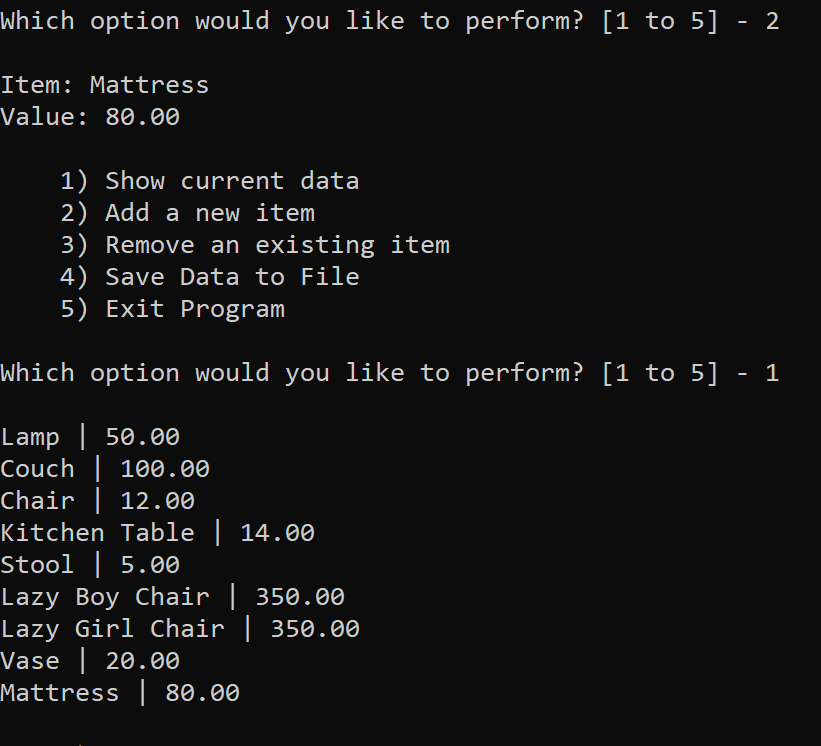


1. When Option 1 is selected, the Current Data in the lstTable is shown

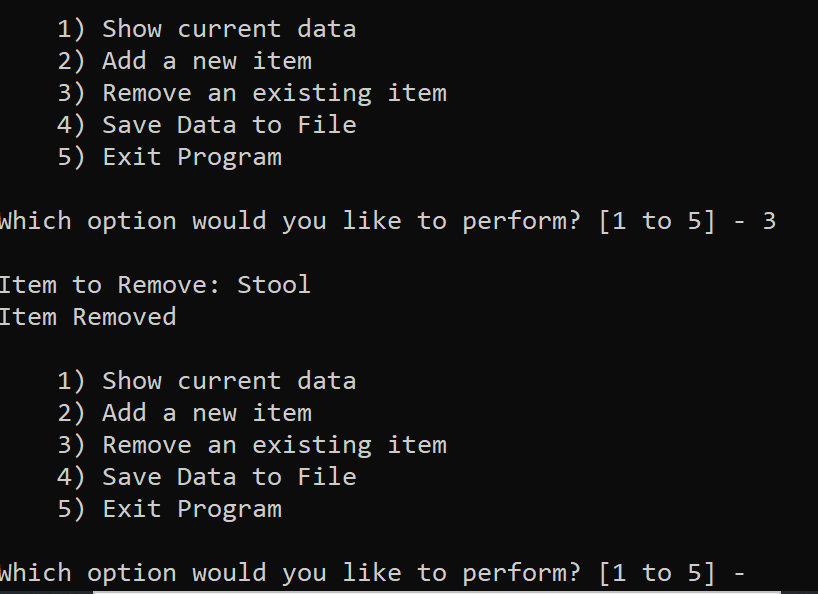
This shows seven entries after choosing Option 1.



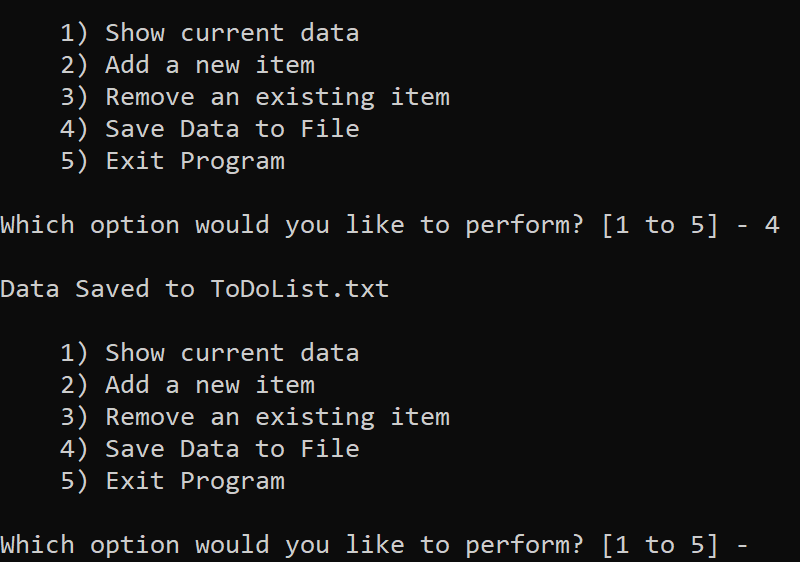
1. When Option 2 is selected, the User is prompted for an Item and a Value.

Selecting Option 1 after shows that the Mattress is appended to lstTable. 

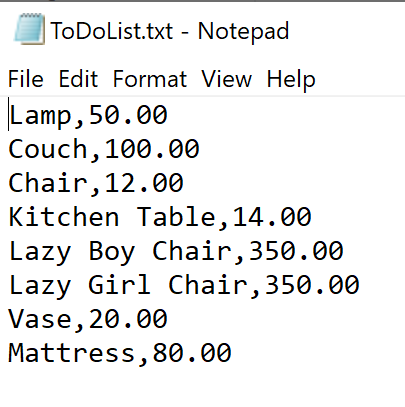
1. When Option 3 is selected, the User is prompted for an Item to Remove (Stool). This item is found in the lstTable and removed. Stool used to be between Kitchen Table and Lazy Boy Chair.



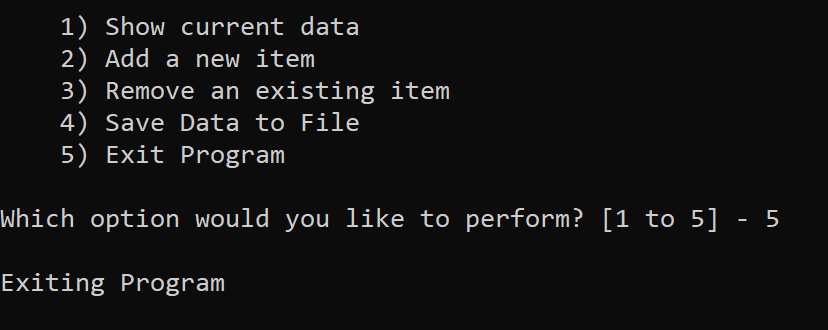
1. When Option 4 is selected:
   1. The user is informed that the data has been saved



* 1. The data from the lstTable is saved to the ToDoList.txt file in the same location as the HomeInventory.py.



1. When Option 5 is selected, the Program displays “Exiting Program” and Exits.



The Home Inventory program requirements appear to have been met for OS Command as well.

# Conclusion

Separations of Concerns was helpful within the program particularly within the Conditional Statement. List and dictionary, index and keys as well as error handling were interesting to practice. Finally, sharing code with other students in GitHub as well as reviewing other’s code is helpful to learn alternative coding practices.