Jonathan Ou

May 11, 2021

Foundations of Programming: Python

Assignment 05

<https://github.com/jayohhhh/IntroToProg-Python>

To Do List Documentation

# Introduction

This assignment goes through the concepts of utilizing a list and dictionary while creating a menu of options for the user to cycle through. Similar to Assignment 04, the user is given multiple opportunities to choose how the script will run through each case. However since this script is working with using dictionaries, additional detail will have to be made on how to call out the values in the dictionary as well as working with and saving these values.

Data Declaration

Figure 1 shows all the variables that are initially called out for later usage. If the variable stands for a list or a dictionary then the blank will be a specific bracket that will allow python to recognize the type of data that is going to replace the variable.

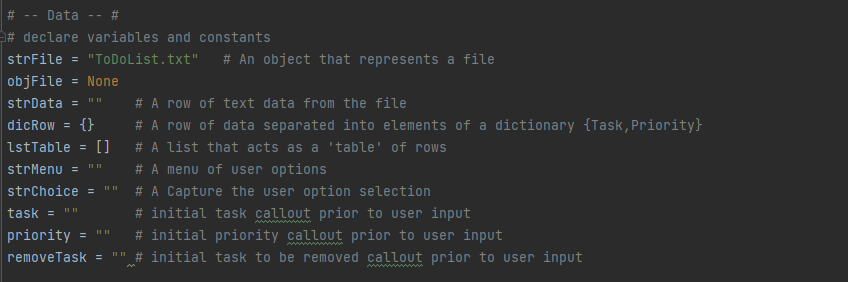


Figure 1: Variable Declaration

Data Processing

The next part of the script is to take in initial data from the text file ToDoList.txt. This is shown in Figure 2. Utilizing the entries from the text file, the script organizing each row into a dictionary with given keys of Task and Priority. The dictionary callout is specific with assigning the keys with values in the input. Afterwards, these rows are inputted into a rows of list which will be saved into the script memory for later manipulation.

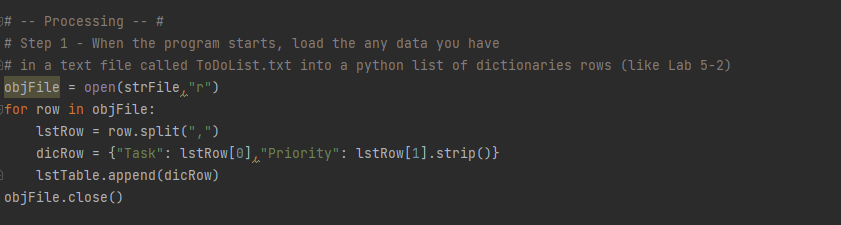


Figure 2: Data Input from Text File

Menu Creation

The menu creation starts with the print function which creates the initial appearance of the menu for the user to view. Then it offers the user an input option which will then initiate several if statements to initiate the proper action. This user input is saved as the variable strChoice and will continually be replaced every time the user is prompted with the menu selection. The rest of the script is the conditionals in which it lists out the script commands when the user inputs entries from 1-5.

Figure 3 shows the menu creation and what happens what the user selects 1 and 2. Selecting 1 displays all information stored in the variable lstTable which contain all the tasks and priorities. Since each row is a dictionary input, the callout for printing will use the corresponding keys to extract the values. Selecting 2 will allow the user to input in tasks and priorities of their own through the help of the input function. Again since the lstTable is assembled with rows of dictionaries, the creation of each row must be a dictionary. This still utilizes the format of calling out the key then the values which correspond to the user input. At the end of option 2, the user will be displayed the entire dataset to confirm that their entry was received and processed.

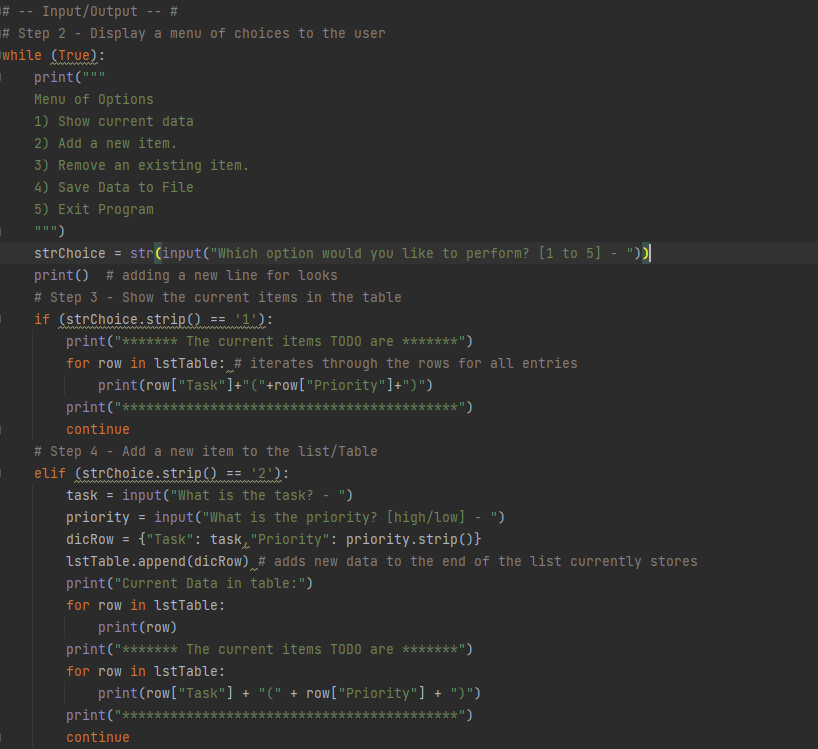


Figure 3: Menu Creation and Choice 1-2 Detail

Figure 4 shows the script which assembles the output for user options 3-5. Option 3 will allow the user to remove any entry from the dataset if their input matches one of the tasks. This is accomplished through the use of a for loop and and if statement. The for loop cycles through all rows of the dataset individually. The if statement scans the first key/value pair to check if the user input is contained in the value section. If there is a match, then the append function will remove the scanned row from the dataset and the user will receive a printed message saying that a task has been removed thanks to their input. Option 4 goes through saving the current data set, lstTable, into a text file. The user must select the option y prior to doing so otherwise the script will simply bring the user back to the menu option. When writing into the text file, the values are extracted from the dictionaries through the index callout of the keys. This will then be used to build a string which will take out any mention of the keys. The final text file will only contain values of tasks and priorities separated by a comma. The user will receive a message notifying them if they had their data saved or not. Finally choosing option 5 will break the script and exit the user from the program. It simply uses the break as that is all that is needed.

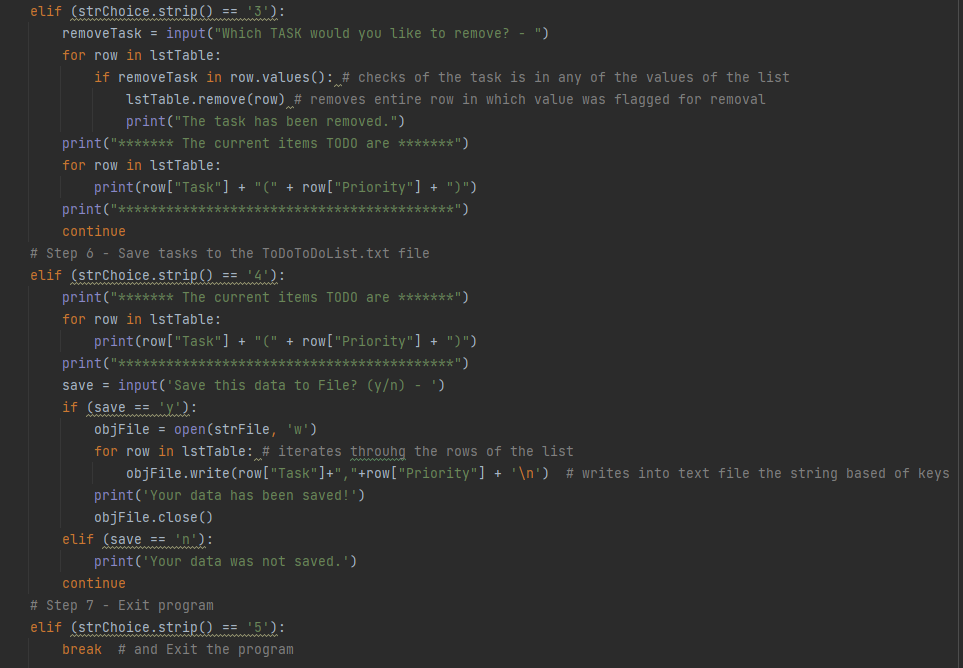


Figure 4: Choice 3-5 Detail

Pycharm Output

The following Figures 5, 6, and 7 show the output of the script into Pycharm. The output cycles through all the options of the menu.

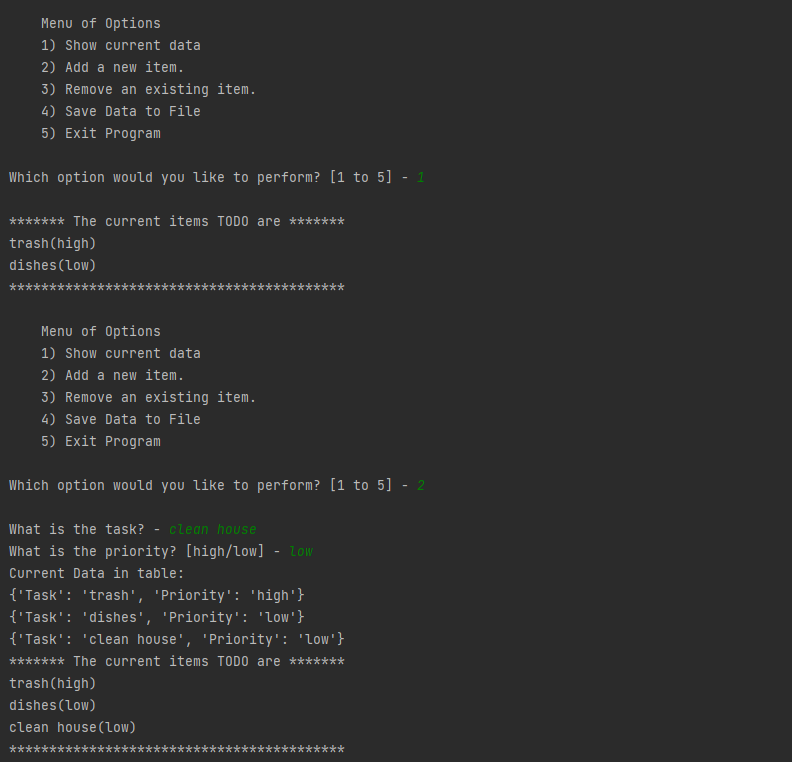


Figure 5: Pycharm Output Part 1

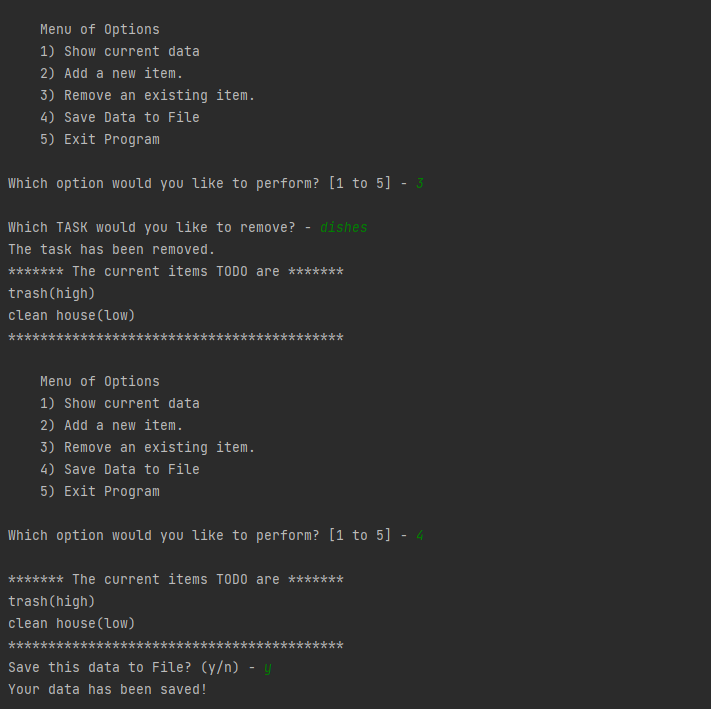


Figure 6: Pycharm Output Part 2

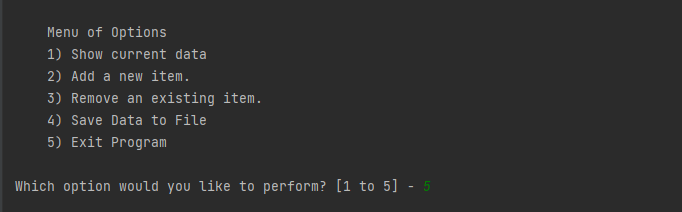


Figure 7: Pycharm Output Part 3

CMD Output

The following figures display the CMD output of the assignment script. Note that Figures 8 and 9 are exactly the same output as Figure 5, Figure 6, and Figure 7. The one difference that arises is the input text file which has been manipulated in the Pycharm output and undone in the CMD output.

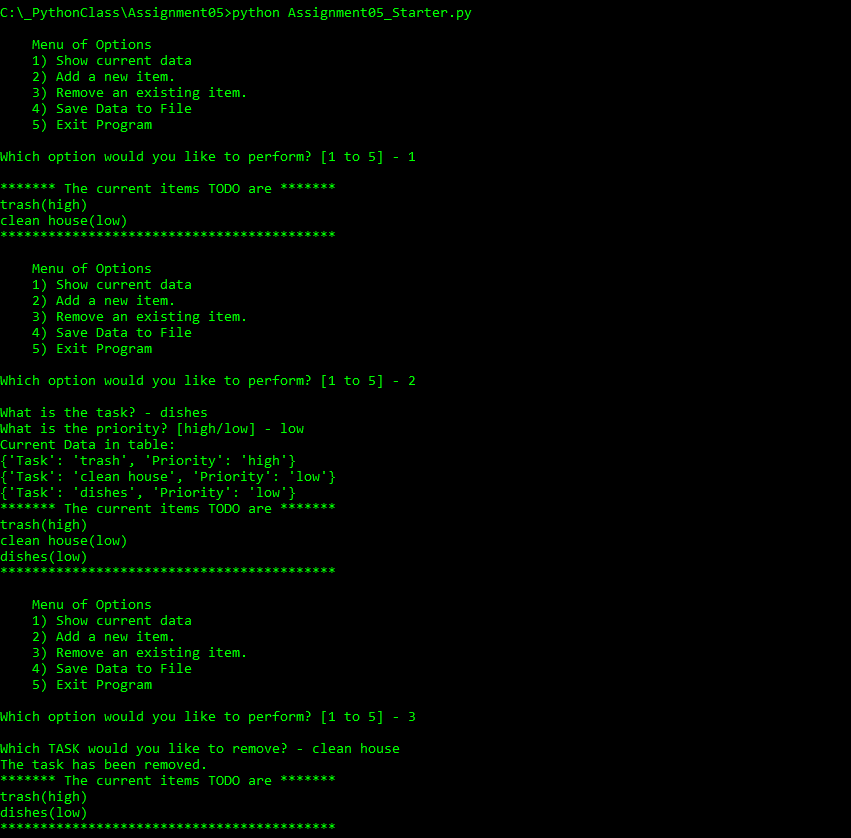


Figure 8: CMD Output Part 1

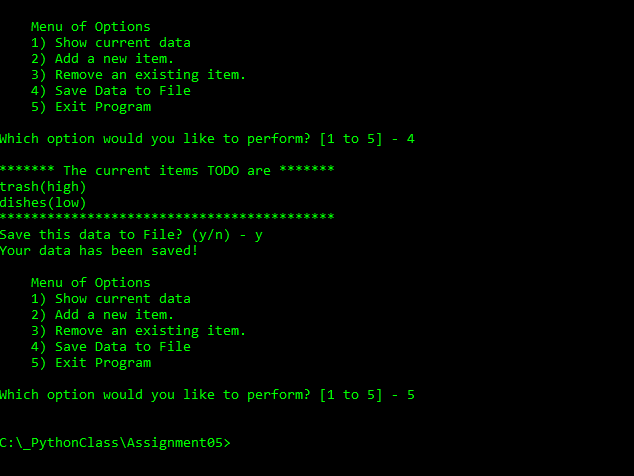


Figure 9: CMD Output Part 2

Text File Output

The final Figure 10 will show the text file output which was generated by the CMD output. This text file was created in the same file location as the python script.

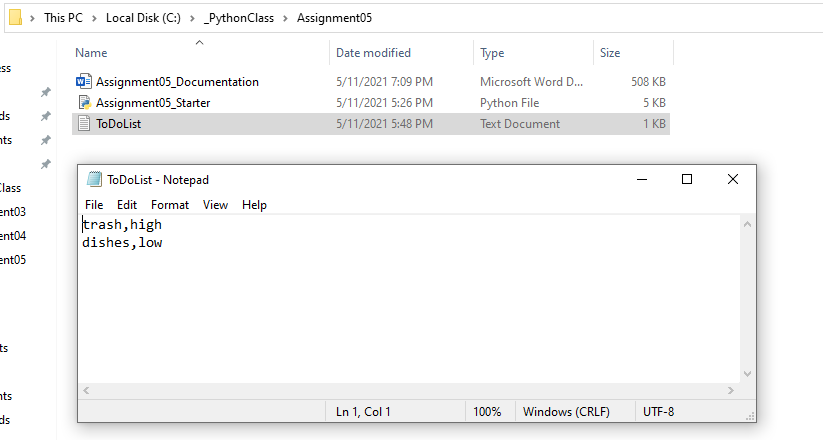


Figure : Text File Check

# Summary

The completion of this script allows me to create a menu of options which allows the user to manipulate dictionary inputs into a list. Actions for these inputs include adding new items, removing specific entries, printing current memory data, and finally writing the items to a file. This exercise continues to utilize previous knowledge such as conditionals which are increasingly useful. It will be great to keep adding to this list of practices.