Omar Lozano

May 16, 2022

Foundations of Programming: Python

Assignment 05

https://github.com/omega609/IntroToProg-Python

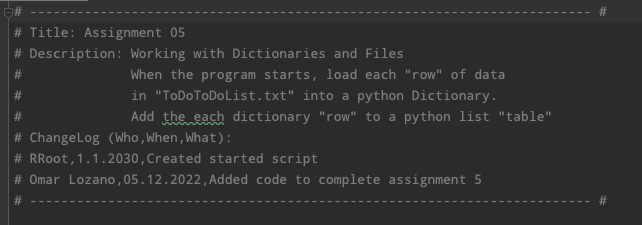
Using a List of Dictionaries to Create a Table

# Introduction

For this assignment I will be creating a table using dictionaries. The columns of the table are representative of a task and a priority. This table will be created and manipulated through the use of a menu where the user will be able to choose if they want to add, remove, view, or save the data to a text file.

# Creating the Script

This script started with pseudo code and a template provided by our instructor outlining the steps that would need to be taken to complete this assignment. Per the instructions I began by updating the changelog (figure 1).

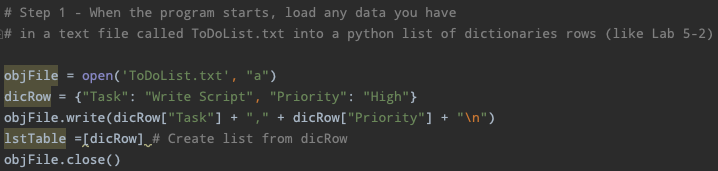


## Figure 1: Update name and changes to code

Next was the declaration of variables which would be used throughout the script. These variables were included in the template provided by our instructor (figure 2).

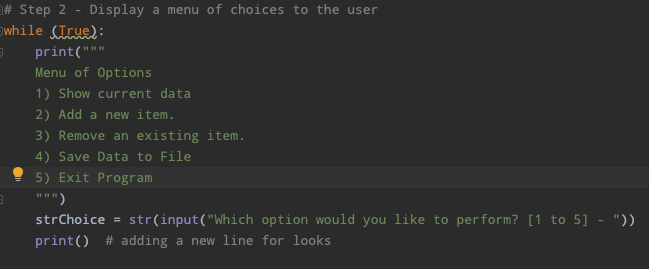
## Figure 2: Update name and changes to code

Following the declaration of variables, I used the open function and append mode to create and load data into a text file. I then created a dictionary using “tasks” and “priorities” as my keys and included value pairs. These represent the rows of the table I am creating. This is also where I make a list of dictionaries, lstTable (figure 3).



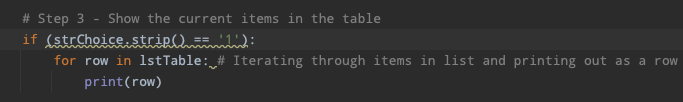
## Figure 3: Generate file and append new list of dictionaries

Presenting the user with a menu of choices to view, remove, add, or save data was next, but this was provided to me by our instructor and is done so using a while loop. The while loop allows the user to choose different options from the menu and add many dictionaries to the list created previously (figure 4).



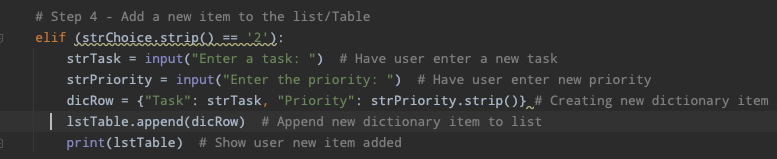
## Figure 4: User menu to manipulate data

To address the first menu option, show current data, an if statement and for loop were used to iterate through the list and print to the screen the list items, dictionaries.



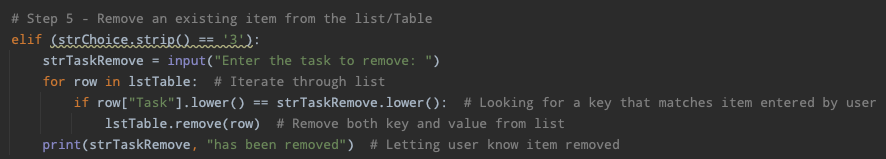
## Figure 5: If statement and for loop to present data

To add to the list of tasks and priorities a conditional “elif” statement was used to ask the user to enter a new task and priority and assign these items to the “dicRow” variable to create a new list item, a dictionary. I could then append the new list item to the end of the list created above using the .append() function and show the item was added successfully (figure 6).

******

## Figure 6: Add new task and priority to list

Removing an item from the list required another conditional statement with a for loop. After asking the user to enter the task they would like to remove, a for loop is used to iterate through the list (lstTable) and compare if the string entered matches the string of a key in one of the dictionaries in the list. If the key is found .remove() function is used to remove the dictionary item (figure 7).



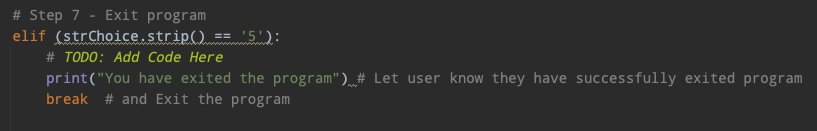
## Figure 7: Remove task and priority from list

Saving the newly created list of items was done almost the same way I created the file except that I did not append this time. Instead, I used “w” to write over the existing file (figure 8).



## Figure 8: Save task list to text file

Finally, to exit the program, I included a statement to let the user know they had exited the program (figure 9). Throughout the program the .strip() function was used to remove any trailing spaces that might have been entered when a user was making a choice.



## Figure 9: Exit the program

# Validating the Script

Just like the previous assignment, validating this script works as intended requires two approaches: (1) through Pycharm and (2) using the Shell Operator.

## 

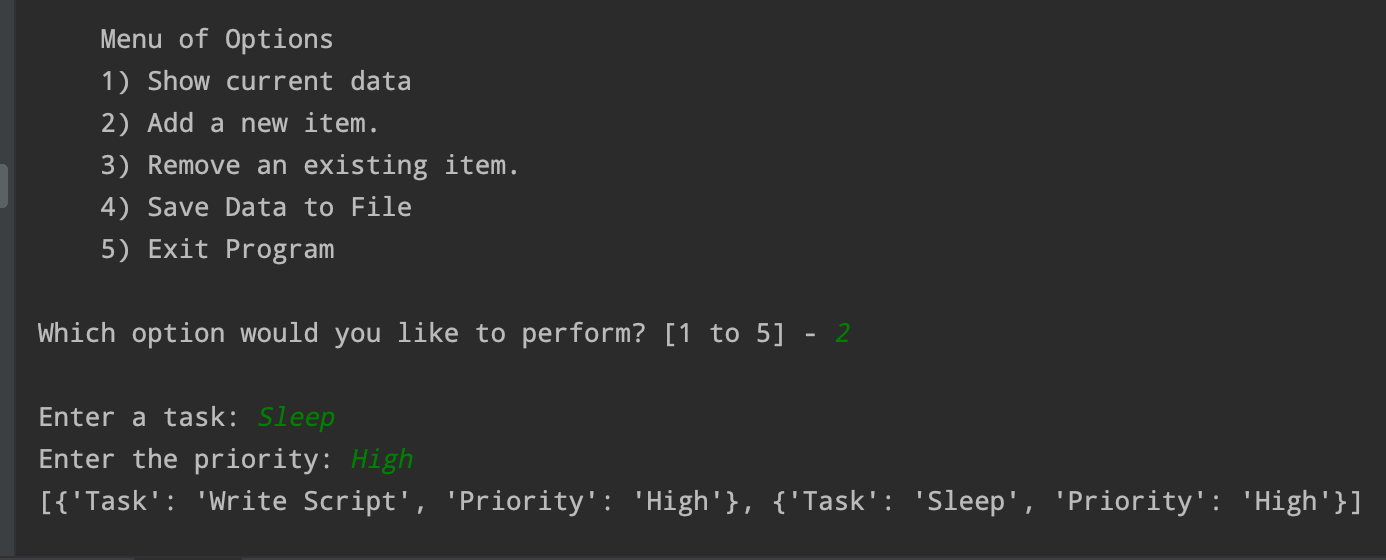
## Through Pycharm:

First, I choose option 2 to enter a new task and priority, because I have included a print statement, I am shown all tasks in the priority including the new task, sleep with a priority of high (figure 10). Next I choose option three to remove a task and then option one to show the task has been removed (figure 11). Finally I save the data to a file and exit. (figure 12). Next I go to Finder >Documents >\_PythonClass >Assignment05 >ToDoList.txt and open the file and verify a file has been created with the user input (figure 13).

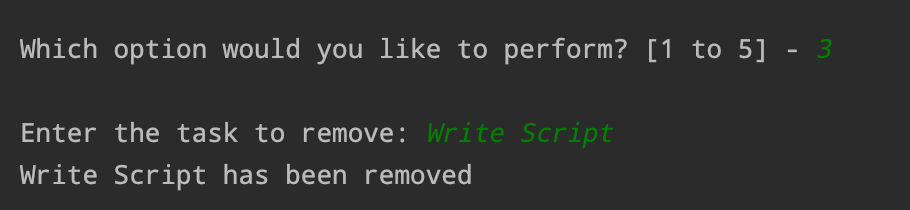
## 

## Using the Shell Operator:

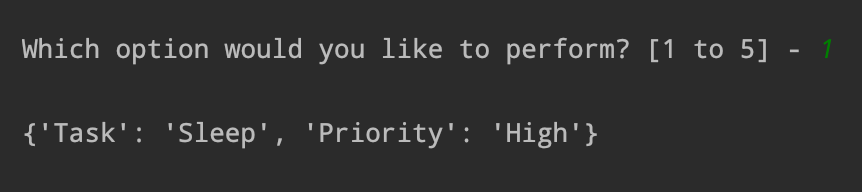
To validate the code works in the command shell I copy the path and paste it to the command line. I run the script using different data from what was used in PyCharm, a task of run, and capture the results of the code to demonstrate that the code works as intended (figure 14, figure 15). Finally, I use Finder again to locate the new text file created (TodoList.txt) and validate the new data is written to the file (Figure 16).



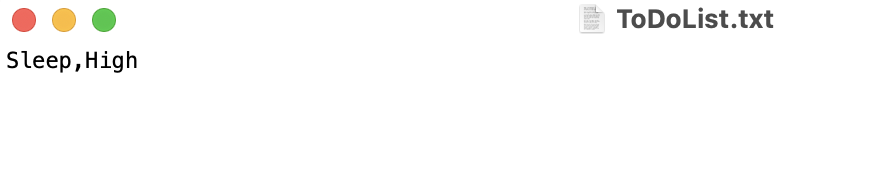
***Figure 10: New task added “sleep”***



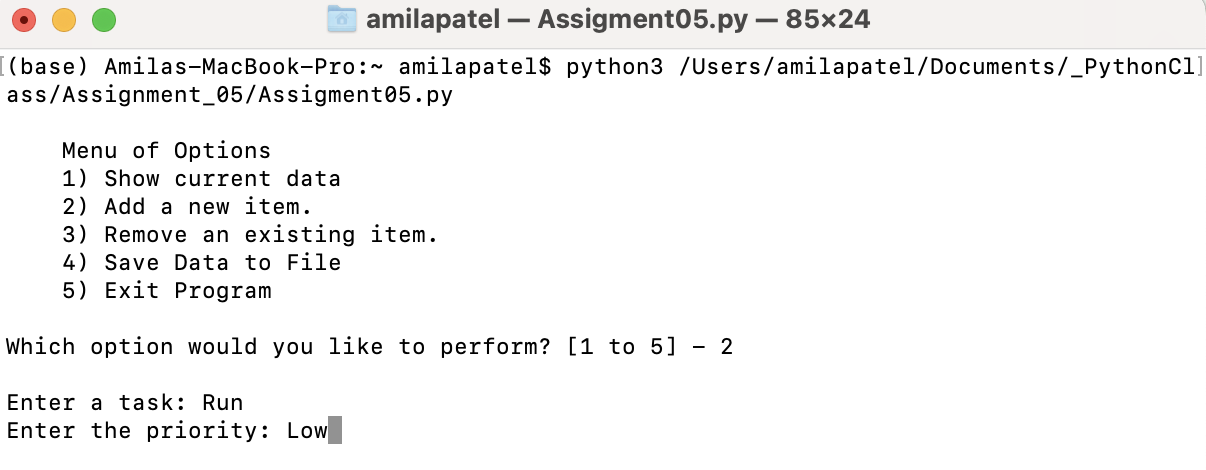
***Figure 11: Remove task of “Write Script”***



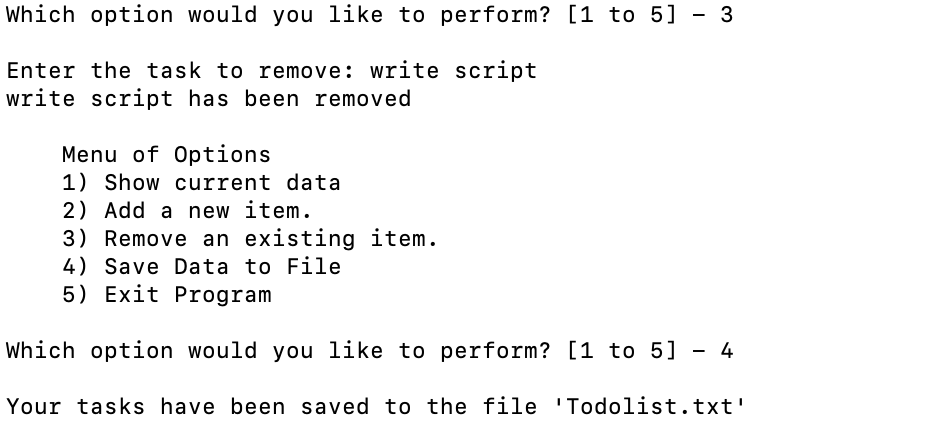
***Figure 12: Display tasks with “Write Script” removed***

******

***Figure 13: File created with new task***

******

***Figure 14: Run code in shell***

******

***Figure 15: Remove existing task***

******

***Figure 16: Text file created***

# Summary

Using dictionaries in a list for me proved to be a difficult concept since each item in the list consists of the key and value pair, making it seem as though there was a larger list than needed. This led to me having trouble at first removing tasks, as I thought I had to remove both the key and value pair and the user could remove a new task, but this was clarified in the discussion board and I was able to get my script to run.