Michael Clark

February 14, 2021

IT FDN 110 B

Assignment 05

https://github.com/MClark89/IntroToProg-Python

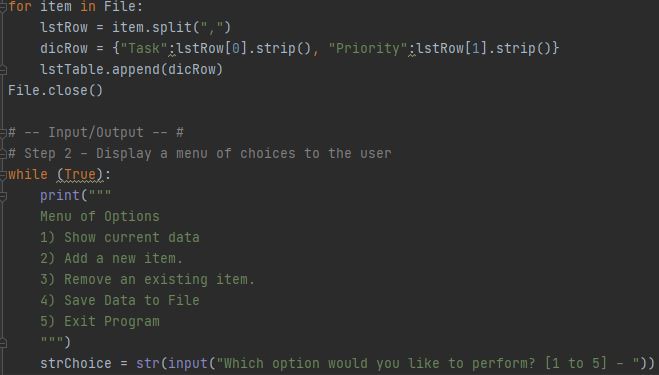
Creating a Python Script that Gathers User Data

Introduction

As we delve deeper into this course, we can learn more advanced ways of gathering and processing user input. One of these ways is through using dictionaries, which allow a user to create a list based off key-values instead of just ones and zeros. Dictionaries allow the programmer to assign specific names to columns to each input. Through this method you can organize a user’s data more efficiently.

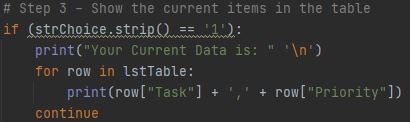
Writing the Script in Steps

The goal of this script was to develop a program that gathered user data and allowed the end user to either edit the information or save and exit the program. The first step was creating the script that opened the text file. We only wanted to first read the text and that is why the “r” function was used in Figure 1. This was followed by the defining of the loop where we used the dictionary function. An important note about dictionaries is when defining them, you need to provide each row with a specific name like “Name”:lstRow (Figure 1) so the program can read the specific row. After this, the main menu needed to be presented similar to previous assignments, where each number has a loop function associated with the specific task.



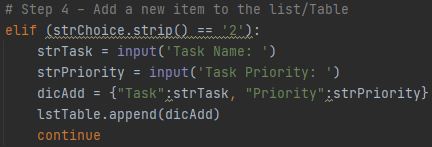
***Figure 1. Menu Screen Script***

For the first loop, we wanted to allow the user to see what current data was saved onto the ToDoList.txt. We accomplished this goal by using an if statement (Figure 2) that pulled the corresponding data from the .txt file described in Figure 1 and presented it to the end user.



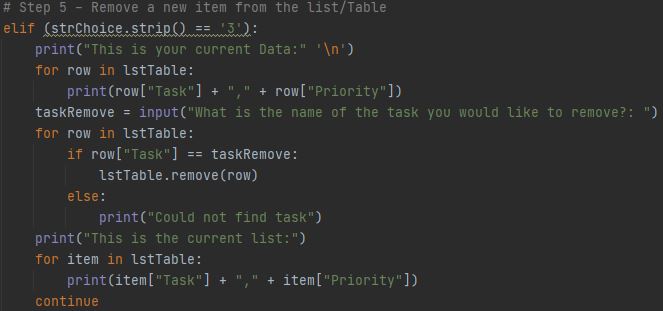
***Figure 2. Current Items in Table Script***

After the first loop we used elif loops which allowed for the script to continue repeating after each section was completed. For the next task we needed to be able to add a new task to the dic file. This script is similar in fashion to adding items to a list but with dictionaries you need to specifically define the columns with names before entering the specific strings (Figure 3).



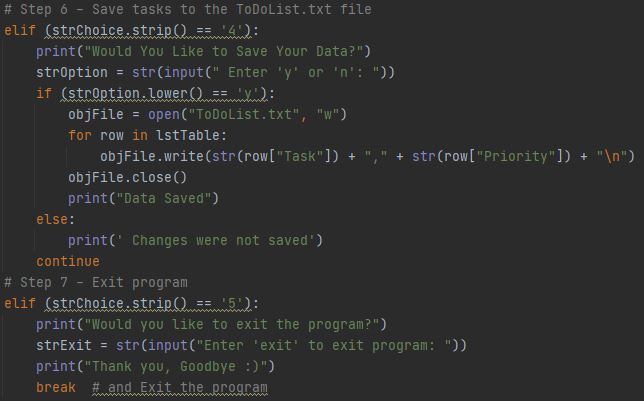
***Figure 3. Adding Items Script***

The next task was by far the most challenging when it came to creating a solid script with good order of operations. The script was created so the end user could remove previously entered data. The first portion of the script was relatively straight forward, similar to Figure 2 I just presented the current data and asked what item the user wanted to delete. Then to delete a row I decided that it would make more sense to delete a task rather than to delete the priority. So, I assigned the “Task” name describing the row to be the list to be reomoved.



***Figure 4. Remove a Task from the List Script***

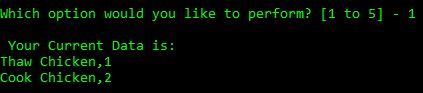
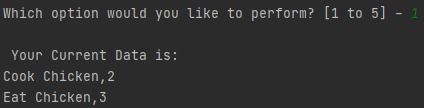
The final steps were to allow the user to save the data onto a .txt file. By using an if statement, we allowed the user to select from either ‘y’ or ‘n’ where ‘y’ would save the data and ‘n’ or anything other than ‘y’ would direct the user to changes were not saved (Figure 5). Then to exit the program the user just ended to select enter, but I wanted the end user to feel more involved, so I prompted them to enter in ‘exit’.



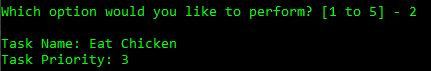
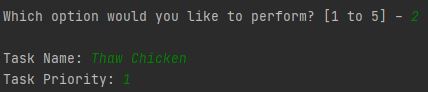
***Figure 5. Saving Data and Exiting the Program***

## Testing the Final Product

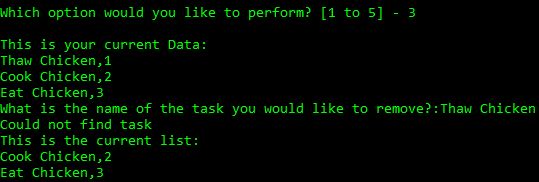
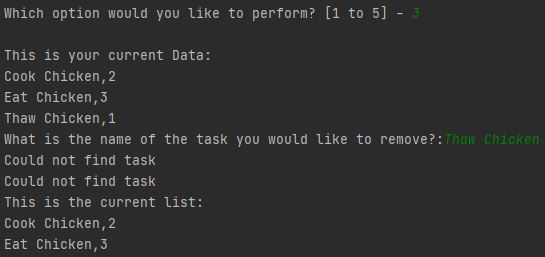
After the Script was written each step was tested in both PyCharm and the Command Panel, the following figures show how each task ran correctly (Figure 6 -10).



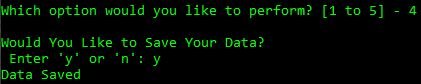
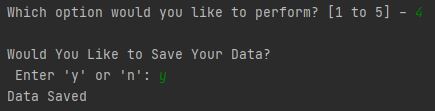
***Figure 6. Showing Current Data***



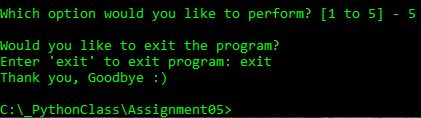
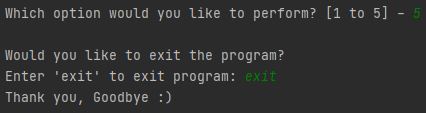
***Figure 7. Adding Data***



***Figure 8. Removing Data***



***Figure 9. Saving Data***



***Figure 10. Exiting Program***

Summary

Having a script that can gather user data and process it affectively can make a huge difference when gathering large amounts of data. Where simple lists fail, the dictionary function is a step above. Throughout the script above, dictionaries provided a more organized storage of data than lists.