Selina Serphin

February 13, 2021

Foundations of Programming: Python

Assignment 05

<https://github.com/sserphin1208/IntroToProg-Python>

# Creating the TODO List Script

### Introduction

In this paper I will outline how I prepared for and built the TODO List script. This script is part of Module 05 of the Foundations of Programming: Python class. This script allows the user to add tasks and their respective priorities to a todo list, the user can also remove items from the list and save the list to a text file.

### Preparing to Write the Script

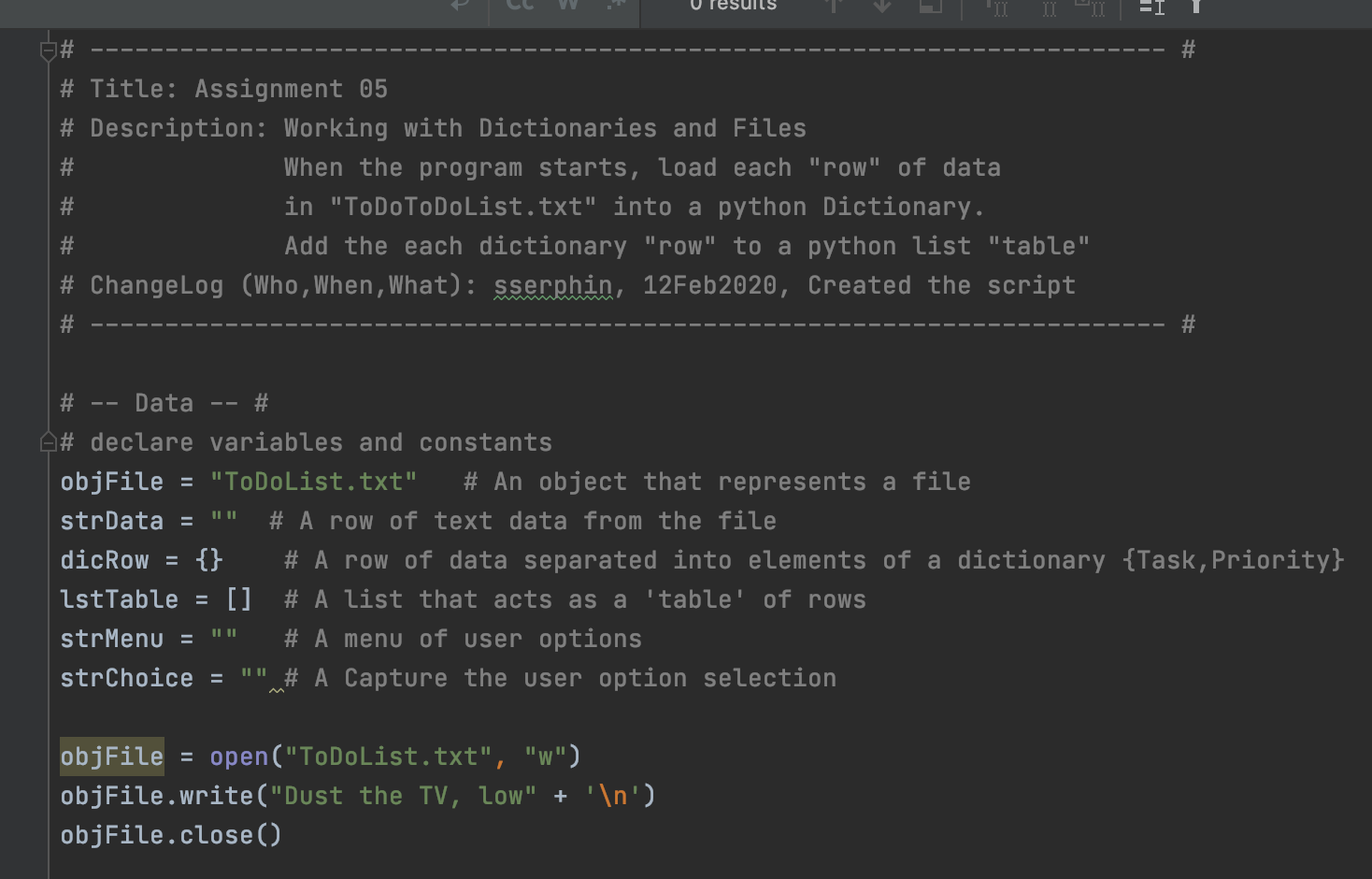
In preparation for this assignment I attended the Module 05 lecture last Tuesday and read “Chapter Five” of *Python Programming for the Absolute Beginner, 3rd Edition by Michael Dawson.* The reading imparted a lot of helpful information about lists and dictionaries. The section that explained list mutability helped me start to understand the difference between tuples and lists.

The section of the reading that outlined some of the most commonly used list methods was helpful because I knew I would be using the .remove() method in the script. It also helped me connect lists’ mutability with their greater flexibility, when compared to tuples.

The concept of nested sequences was helpful for me to review before starting the script, because I would need to create a list table made up of dictionary rows. I felt that I had a solid, high level understanding of nested sequences, but I ended up reviewing them a bit when I was struggling to get my list table to write to my text file.

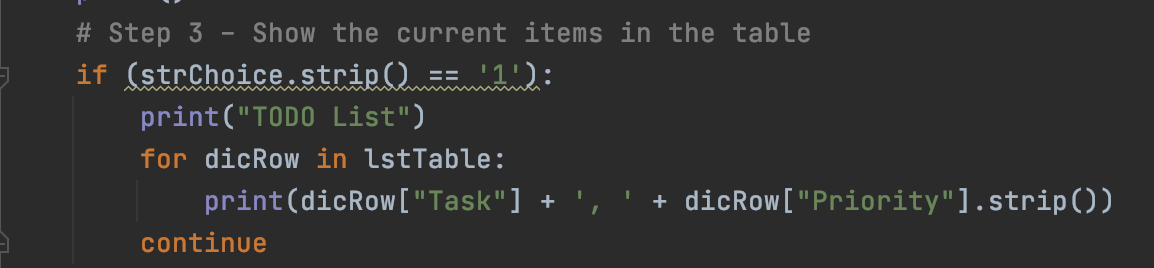
### Creating the Home Inventory Script

I started my script by adding a change log section and pasted in the starter code. I also soon realized that I wanted to create the TodoList.txt file and hard code some data into it. I thought creating the file before the While loop starts would make the script more user friendly by ensuring that the file exists before the code tries to read from it. Please see **Figure 1**.



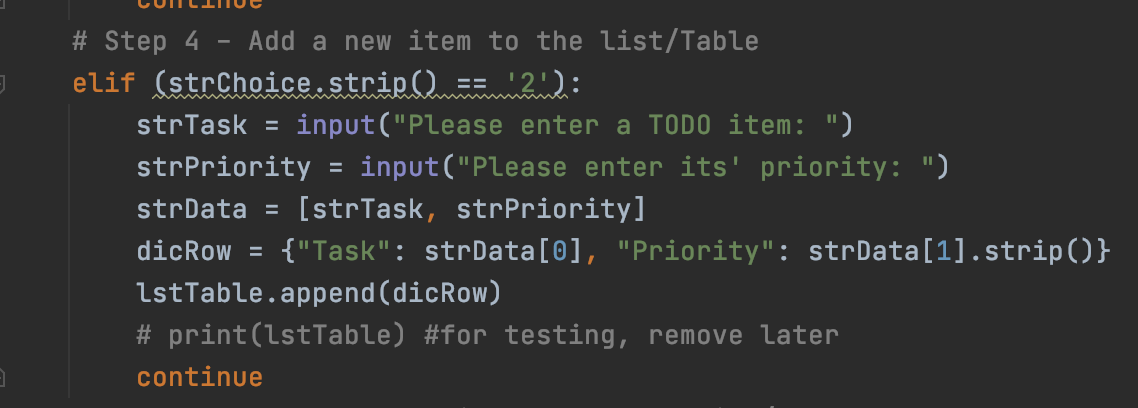
***Figure 1: The change log and creating the TodoList.txt file with hardcoded data***

I then started on the first menu item 1) Show current data. Based on the class labs I knew I would need a for loop, because I wanted to display the data in a way that would be visually organized. For each dictionary row in my list table I printed the Task and Priority key and stripped off the carriage return. Please see **Figure 2.**



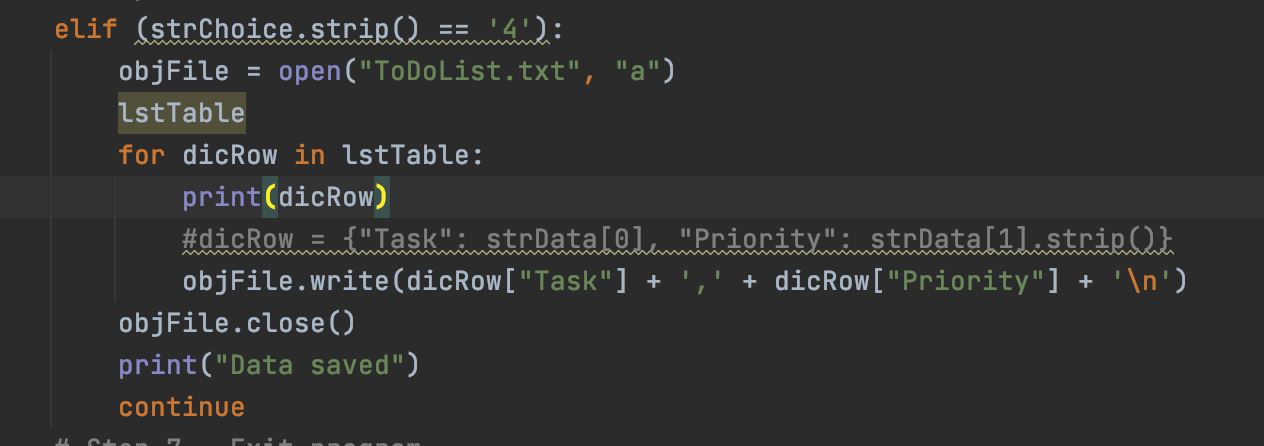
***Figure 2: The code for the first menu choice: show current data***

The second menu option was to add a new item to the TODO list. I used input() to get the user data and stored them to two separate variables, one for the task itself and one for the task’s priority. I saved those values together in a list. But to add the item to my list table, which was made of dictionary rows, I needed to save the task and priority data in a dictionary. I pulled out the zeroth and first index elements from my list and saved them with the keys, Task and Priority in my dictionary row variable. I used the print() method so I could test that my code was working as I built out the script. Please see **Figure 3.**

******

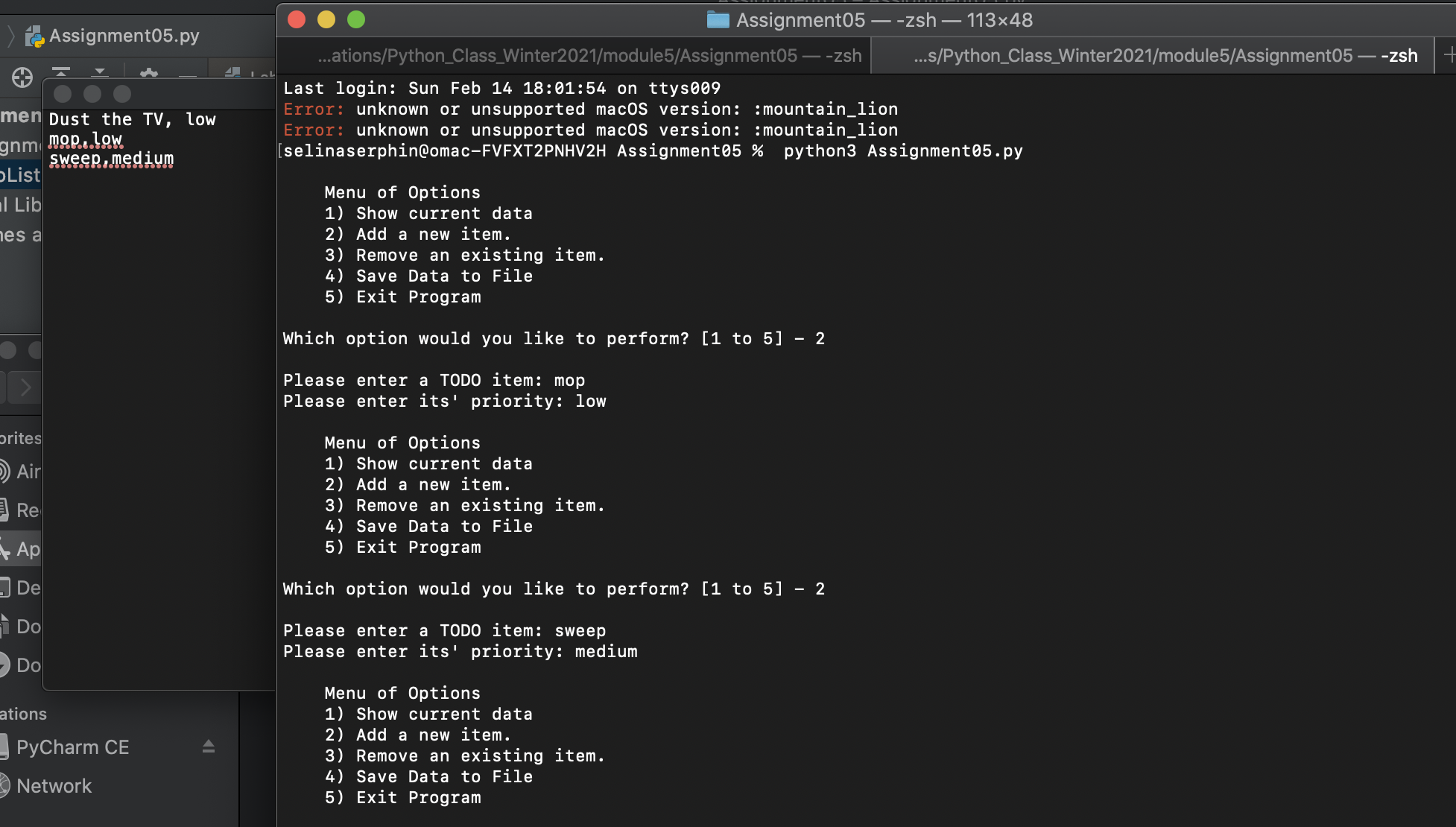
***Figure 3: Saving my list data as a dictionary row***

The fourth menu option, Saving data to the file, was the most difficult for me. I was running into an issue where my code was only saving the last (most recently created) dictionary row to the file instead of the entire list table. I finally realized this bug was happening because I was setting my dicRow objet back to the last instance of my todo list variable (strData). Once I removed that line, all of the items in my list table saved to the Todo text file as expected. Please see **Figure 4.**



***Figure 4: Needed to remove the commented out line so my list saved to the file correctly***

I verified that the code runs correctly in PyCharm and the Terminal, please see **Figure 5**.



***Figure 5: Assignment 05 runs in the terminal without error***

### Summary

Assignment 05 really increased my understanding of lists and dictionaries. The reading’s game based code examples were a fun way to illustrate the practical uses of lists and nested sequences. The flexibility of lists make them a great option for storing and manipulating data. Dictionaries were a new concept to me, but during the class lecture the SQL/primary key example made me realize I had worked with dictionaries before. The assignment really brought these two concepts together when I used dictionaries to build a list table of todo items. I look forward to learning more about each concept.