Chrissy Henderson

November 13, 2020

IT FDN 110 A Au 20: Foundations Of Programming: Python

Assignment 05

# Creating a Multi-Option Task List Script in Python

## Introduction

In this module, we learned about working with dictionaries, importing data from text files, and loading code to GitHub. This particular assignment involved loading in text file data and then the user selecting from several menu options and taking file contents as well as new input from the user as a dictionary and removing items and saving items to a text file.

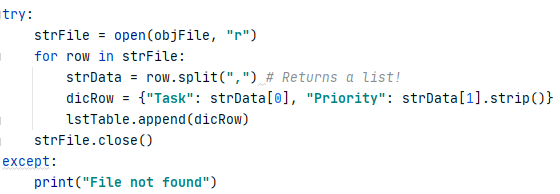
## Script Creation

I was asked to edit a script template that loads in data from a text file and places it into a table. The program then gives the user five different menu options to select from which includes viewing the data, adding a new data item, removing an existing item, saving the data to a file, and exiting the program.

### Before the menu

Although already written, the first step of the program is to declare variables to be used later on. This way, the data type of the variable is already determined and future data can be easily appended and added to the variables without having to declare them later. This also gives an idea to a future reader of this code what variables are present.

My first step after the variable declaration was to write code to load in a text file as a table that could be referenced later in the program (Figure 1). I first wanted to tell the program to “try” to load in the file and then run through the file line by line using a loop. For each line, the loop would read in the data as a string, convert it to a dictionary line, and append the data as dictionary data to the table “lstTable” which had been created previously in the variable declaration section. If the text file was not found, the program would print “File not found” and move on to the next step.



***Figure 1. Code line creating a variable that will open a text file and load the data into a table.***

### Step 1

To start the loop, I wanted to only loop the program when it had not been broken and was still “true” so started the loop with “while(True)”. At the start of the loop, the user needs to know what the program does and so is presented a menu that prints five menu options and then prompts the user to enter in a number between 1 and 5 which the program will store with the variable name of “strChoice” (Figure 2).



***Figure 2. Code line creating a variable that will be used in the loop to select an action to complete***

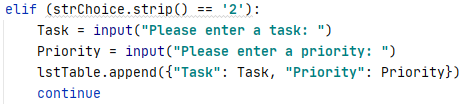
After entering the number “1” the program will print the data located in the table “lstTable” created in the “try” statement with all of the formatting of dictionary data. Figure 3 is an example of how this dictionary data would be viewed from the user’s perspective.



***Figure 3. The previously created table formatted as dictionary data***

### Step 2

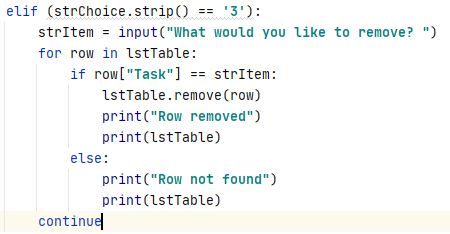
After data has been printed, the program continues and loops again. If the user enters in “2”, the program will prompt the user for a task and priority item and will append the data to the table that has been imported (Figure 4). If no text file had existed, this table would be empty but would exist as it had been created in the variable declaration step.



***Figure 4. If a user inputs “2” into the program, the program will append the new data to the table.***

### Step 3

After the program appends the data, the program will prompt the user to enter in another menu item. If the number “3” is entered, the program will print a question asking what data the user would like to remove (Figure 5). The program will save the input as “strItem” and then a loop will begin for each of the rows in the “1stTable”. If the loop hits a task in the dictionary that matches the input from the user, the row will be removed and the loop will print a statement saying the row has been removed and then will print the whole table. If it does not find it, the loop will print that the row has not been found and the program will continue back to the menu.

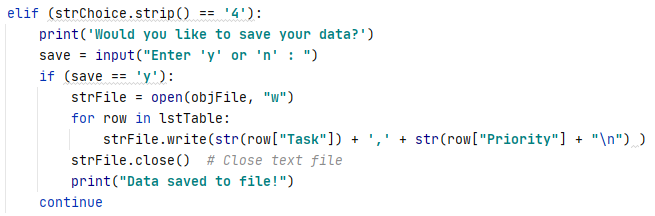


***Figure 5. If a user input “3” into the program, the program will ask what item the user wants to remove an then remove it from the table.***

### Step 4

After the program continues back to the menu, the program will prompt the user to enter in another menu item. If the number “4” is entered, the program will print a question asking if the user wants to save their data (Figure 6). The program will then accept input of “y” before formatting the data and then writing it to the file variable created earlier. When opening a file using Python, if the file does not exist, the program will create it.

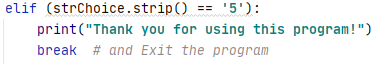
After this, the program prints “Data saved to file!” to let the user know that the data was saved. I made sure to put this line after the “write” and “close” lines so that if something goes wrong with writing the file, the user does not see the printed statement before this error. After this step, the loop continues back to the menu.



***Figure 6. When entering a choice of “4”, the program asks the user if they would like to save the data to a text file.***

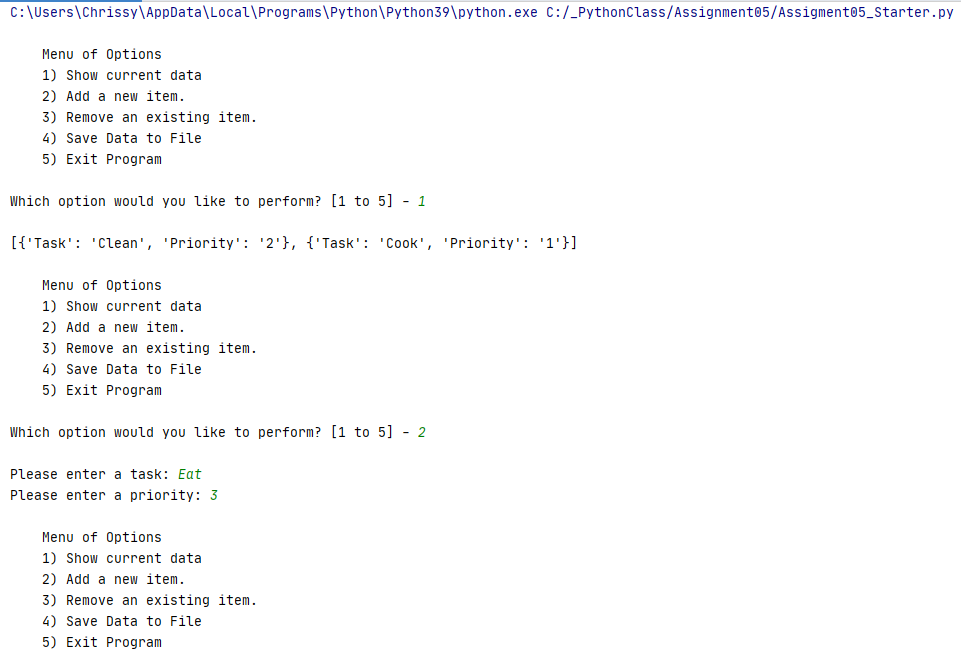
### Step 5

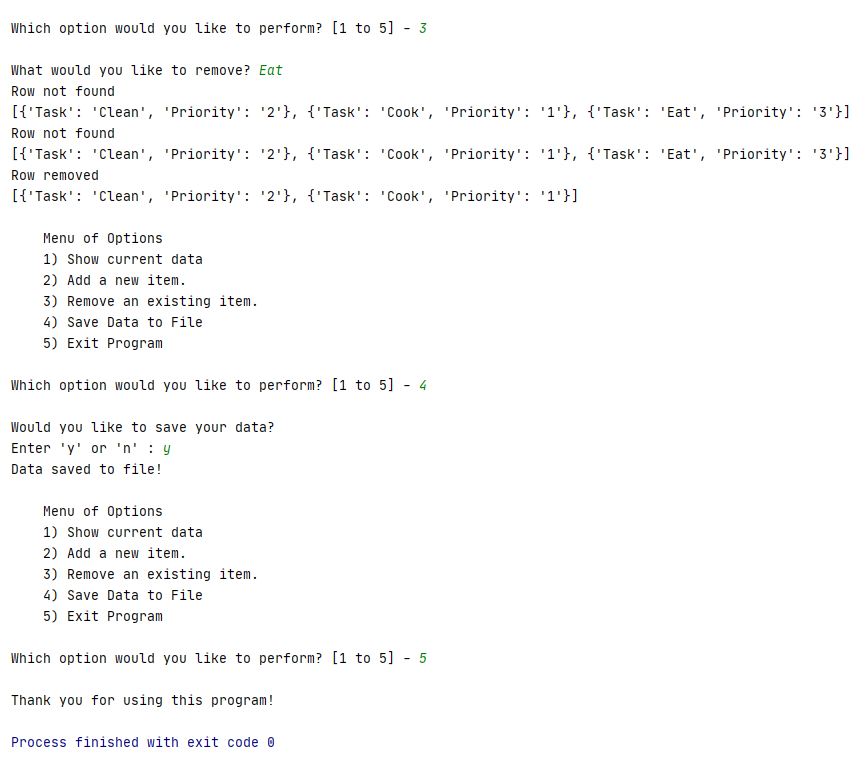
The last option I put in in is if the user enters in “5”, the program will print out a message and break (Figure 7).



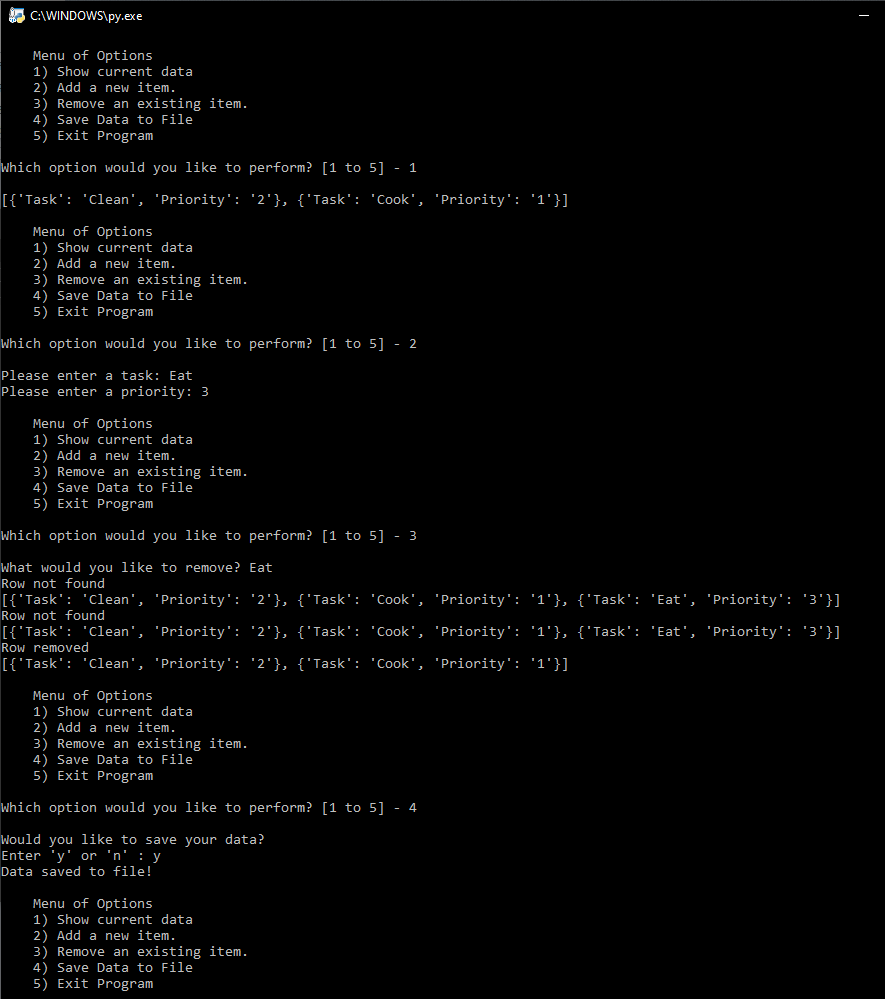
***Figure 7. Printed message before the loop breaks and the program ends***

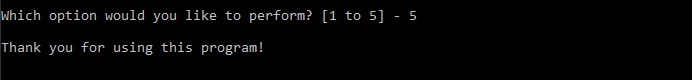
The task list program was successfully executed in PyCharm IDE and in the Windows command line, respectively (Figure 8 and 9). When I checked the text file that was created, the data was successfully written to the file (Figure 10).



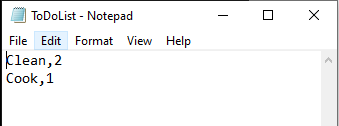


***Figure 8. The program working in PyCharm***





***Figure 9. The program working in the Windows command shell***



***Figure 10. The program output of the user’s data before and after running***

## Summary

In this particular assignment, I created a program to create a text file of tasks and priorities of these tasks. The program prints instructions with five different options. The five options allow the user to view table data, append task and priority data to a table, remove data from the table, writes the information to a text file, and finally exit the program. Creating and writing to a text file is a great way to collect and store user information and can be beneficial for streamlining data entry and preventing users from accidentally changing past data. The program in particular is handy for entering and viewing data in a dictionary format to properly store information. This can save time as the user does not have to find the file and open it to see that they might have made a mistake. Overall, the program is very useful for cataloging.