Using List And Dictionaries Interacting with a File

**Introduction and Script along with Proof it Works**

Let me start with it was not an straightforward one for me to create and get running correctly. A hurdle for me is interacting with text files and having them print correctly by using code, and I’ve always known interacting with text files is my current weak point. A good deal of information I had to check online sources to see what others have done and their explanations as to why.

# ------------------------------------------------------------------------ #

# Title: Assignment 05

# Description: Working with Dictionaries and Files

# When the program starts, load each "row" of data

# in "ToDoToDoList.txt" into a python Dictionary.

# Add the each dictionary "row" to a python list "table"

# ChangeLog (Jack Kennedy,11/17/2020,Editing Script for Assignment):

# RRoot,1.1.2030,Created started script

# <Jack Kennedy>,<11/17/2020>,Added/Edited code to complete assignment 5

# ------------------------------------------------------------------------ #

# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ #

# Personal Notes to Translate Meaning to Self (Jack):

# The purpose of the script is for the user to input two pieces of information

# that will be into an \*Dictionary\*. There will be !!1 Dictionary, 1 ROW!!

# representing each data set. These will then be put into a SINGLE list to create

# the equivilant of a table of data. All this will be saved to a file named

# "mytodolist" where the data will be appended as to not delete previous data.

# Path to file is - "C:\\_PythonClass\mytodolist.txt"

# Flow of work:

# Open "ToDo - Put input data into Dictionary - Put Dictionary into a List - Use list to print and save

# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ #

# -- Data -- #

# declare variables and constants

strData = "" # A row of text data from the file

dicRow = {} # A row of data separated into elements of a dictionary {Task,Priority}

lstTable = [] # A list that acts as a 'table' of rows

strMenu = "" # A menu of user options

strChoice = "" # A Capture the user option selection

txtdata = [] # ADDED = Data for the text document

# -- Processing -- #

# Step 1 - When the program starts, load the any data you have

# in a text file called ToDoList.txt into a python list of dictionaries rows (like Lab 5-2)

# This took forever to figure out. Didn't realize for a while how to put a list into a dictionary.

# These lines take the data in the file, then format them into a dictionary split with the two data

# sets. The dictionary names are decided at the start then the list data is put in. Trying to explain

# it to myself because it was really confusing.

# At the end "dicRow" is given a value to be used later on. This is to take data out of the

# text file to save as a variable.

objFile = open("C:\\_PythonClass\mytodolist.txt")

for row in objFile:

txtdata = row.split(",")

dicRow = {"Task": txtdata[0], "Priority": txtdata[1].strip()}

lstTable.append(dicRow)

objFile.close()

# -- Input/Output -- #

# Step 2 - Display a menu of choices to the user

print("==============================================\n"

"\t Hello amazing user!\n"

"\t This script is made to allow you to\n"

"\t view, add, and remove items of a\n"

"\t list made in -mytodolist.txt-. There is\n"

"\t also an option to save that data.\n\n"

"\t First it will open the file to keep\n"

"\t previously existing data then you the\n"

"\t user may interact with that data.\n\n"

"\t !IMPORTANT!\n"

"\t DATA IS ONLY SAVED WHEN THE OPTION\n"

"\t IS CHOSEN IN THE MENU!\n"

"==============================================")

while (True):

print("""

\*\*\* Menu of Options \*\*\*

1) Show current data

2) Add a new item.

3) Remove an existing item.

4) Save Data to File

5) Exit Program

""")

strChoice = str(input("Which option would you like to perform? [1 to 5] - "))

print() # adding a new line for looks

# Step 3 - Show the current items in the table

if (strChoice.strip() == '1'):

# NOTE - Use "read" to display the data. Will need to use

# a small loop to make it go vertical, refer to previous

# assignment on the code needed. The data does not change

# here, it's only displayed then goes back to loop.

print("--------------------------------------------\n"

"\tThis is the current data in the table.\n"

"--------------------------------------------\n"

"==============")

print("Row - Task - Priority")

counter = 0

for row in lstTable:

print(f'{counter} | {row["Task"]} | {row["Priority"]} |')

counter += 1

print("==============")

continue

# Step 4 - Add a new item to the list/Table

elif (strChoice.strip() == '2'):

newtask = input("-------------------------------------------------\n"

"\tYou've chosen to add to the current table.\n"

"\tWhat task do you wish to add?\n"

"-------------------------------------------------\n"

"\tEnter New Task: ")

newpriority = input("\n---------------------------------------------------------------\n"

"\tWhat priority do you wish to designate to this task?\n"

"\tExamples: 1-10, Not Important to Important, Low to High.\n"

"---------------------------------------------------------------\n"

"\tEnter Priority for New Task: ")

dicRow = {"Task":newtask, "Priority":newpriority}

lstTable.append(dicRow)

print("\n=====",newtask,"has been added with priority",newpriority,"=====")

continue

# Step 5 - Remove a new item from the list/Table

elif (strChoice.strip() == '3'):

# Previous lines before rereading the assignment. Works, just not to the extent needed.

# del lstTable[-1]

# print("--------------------------------------------------\n"

# "\t You've selected to remove\n"

# "\t The newest or last task and priority has\n"

# "\t been removed from the current table\n"

# "--------------------------------------------------\n")

removal = int(input("--------------------------------------------------\n"

"\tThe option to remove a row has been chosen\n"

"\tWhich row do you wish to delete?\n"

"--------------------------------------------------\n"

"\tRow to Delete: "))

lstTable.pop(removal) # NEEDS TO BE AN INTERGER!!

print("\n===== Row",removal,"has been taken out of the table =====")

continue

# Step 6 - Save tasks to the ToDoToDoList.txt file

elif (strChoice.strip() == '4'):

# NOTE - ALL DATA BEFORE THIS POINT IS NOT SAVED TO THE TEXT FILE!!!!!!

# This is the point at which that's done. Open the file then write the

# list inside. Before this point the data is saved in memory, NOT HARDWARE!!

# NOTE TO SELF - Figured out the problem. The dictionary sections needed

# to be in lists not strings.

objFile = open("C:\\_PythonClass\mytodolist.txt","w")

for row in lstTable:

objFile.write(row["Task"] + "," + row["Priority"] + "\n")

objFile.close()

print("===== The current table has been saved =====\n")

continue

# Step 7 - Exit program

elif (strChoice.strip() == '5'):

input("======================================\n"

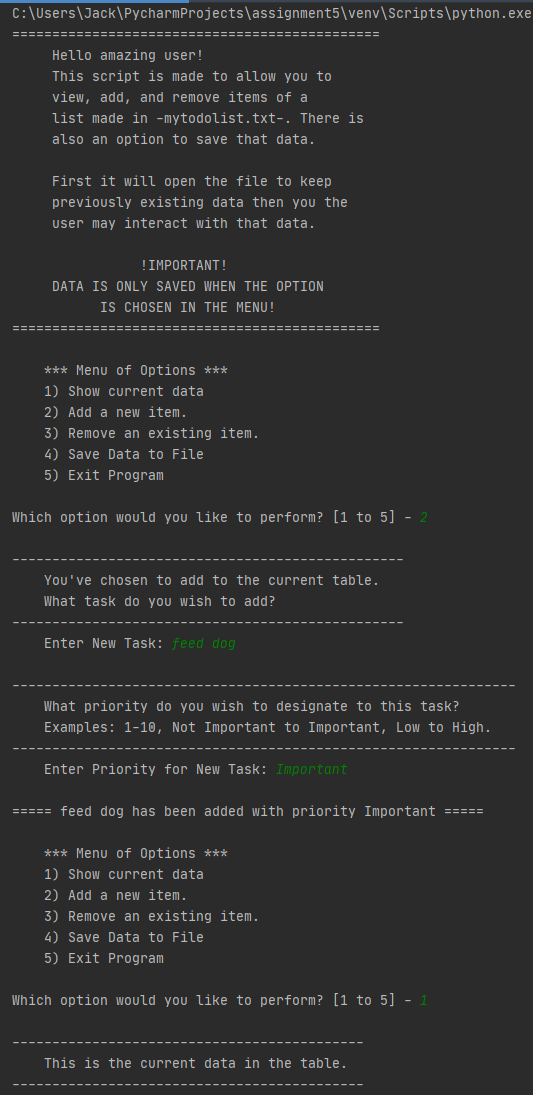
" Thank you for using the script.\n"

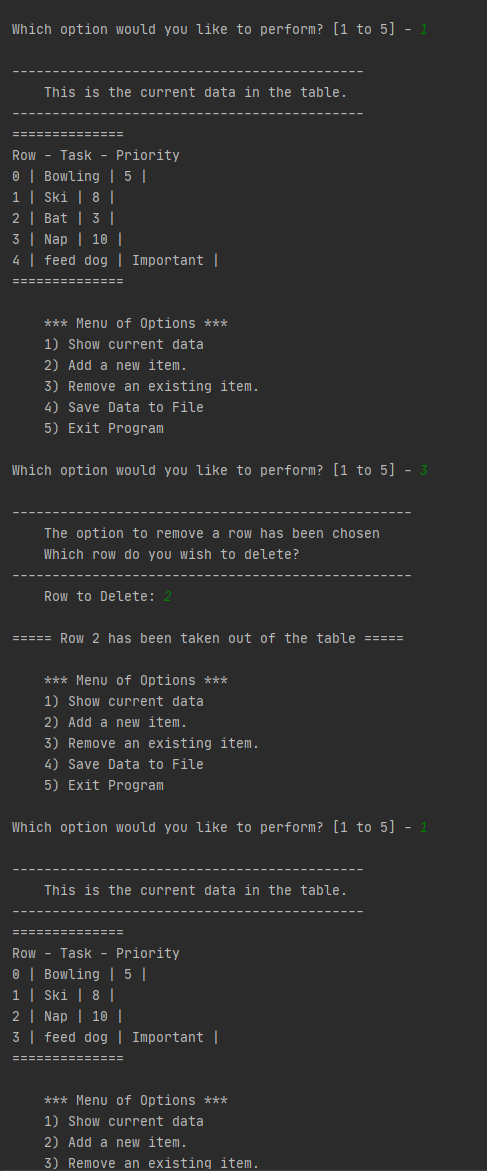
" The script has ended. Please hit any\n"

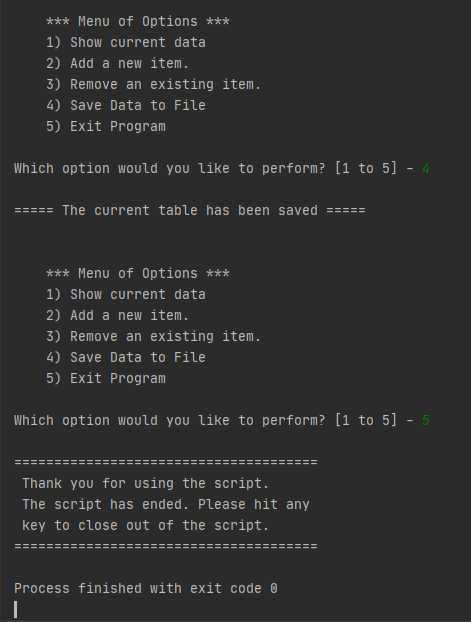
" key to close out of the script.\n"

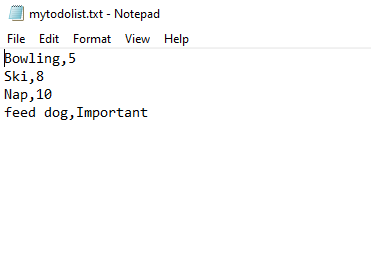
"======================================")

break # and Exit the program









**Topic 1**

The first part was translating the assignment into my own works to check back to while writing the lines of code. There was conversion of code I wanted to keep straight in my head so they were properly used, or avoided errors occurring. Along with that a template was used to keep a log of who originally created it, general changes made, and dates to keep track of. After that are important variables that are used throughout the code. It all served as help for when I needed to reference something again.

**Topic 2**

The processing was the hardest part out of every part of code for the assignment. I for a really long time didn’t realize I could take parts out of the list and put them into a dictionary. I was originally looking up how to convert lists into dictionaries or put the info into dictionaries, which was more complicated than what is there currently. The code in processing makes sense and it’s straight forward: it opens the file, takes data from either side of the comma, and adds it to a preexisting variable.

**Topic 3**

The input output is an introduction that will not loop with a menu that does loop. The introduction tries to communicate how the script is going to work along with the importance that no data saves unless the user chooses to save. Otherwise the text file will stay the same, all the changes beforehand are done in memory.

**Topic 4**

Adding an item to the list was probably the easiest code to put in. This part of the script asks for two inputs from the user, being a task and priority for it. Once those are put in the code adds it to the dictionary, as there is no way to append directly to one, and then takes that info to append to the “lstTable” serving as the primary overarching list for all the data. It then goes back to the menu.

**Topic 5**

Removing a piece of data was actually pretty fun to learn. Working on it I learned the different functions that can be done with lists. The one used is “.pop” to remove a row that the user designates. The older code is kept in, commented out to show previously what it was doing. At the time I was only reading the Step 5 instructions, which said “Remove new item” which I interpreted as the last item. It still deletes an item, but only the last one in the list.

**Topic 6**

This was the second most challenging part of making the script. I feel like I made it similar several ways, but got quite a lot of error messages. The first part was making sure the opening of the file worked, which for some reason wasn’t. I believe it had to do with a pre-existing variable near the top, which I removed. The second part was getting it to write the way I needed it to so the text in the file may be used again. That took reading up more on dictionaries and how they are used, I was unaware originally that I didn’t need a loop. After that I got errors concerning the file closing, which was simply me not having the closing function unindented. After that, the last option allows the user to close the script. Not too much to say there, it’s not as complicated as the rest.

**Summary**

This was the most challenging assignment by far for me in the course. At least up to this date. I have a really big weak point for working with data files, it’s just going to take practice and more learning to get over those hurdles. It was difficult to start since I didn’t even know originally how to get information out of the text file correctly in the first place. Thankfully now I understand more clearly why it’s working, a good deal of it due to having errors over and over again. It also helped to keep notes all over the place, which as I gain more experience will most likely keep in more organized locations. They aren’t too unorganized now, but I’m sure better places exist.