Kyle Biondich

8/7/2023

IT FDN 110 A

Assignment 05

ToDo Funs

Introduction

Week 5 of the course introduced dictionaries and reading data from a text file in python. The following paragraphs outline the methods that were used to read a text file into a python script, capture user input from a menu, and either add a new item to a list, read the total list, or write the list back to an external file.

Intended Outcome

The intended outcome of this assignment is to initially load a text file that consists of tasks and priorities if it exists and store the contents of that text file in a list and provide the ability to:

1. Display the contents of the list
2. Add new tasks and their priorities to the list
3. Remove a task and its priority from the list
4. Save the contents of the list to a file
5. Exit the program.

```
PS D:\DEV\uw\assignments\_PythonClass\Assignment05> python .\Assigment05_Starter.py

    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

Which option would you like to perform? [1 to 5] -
```

Figure 1: Intended Outcome: Assignment05_Starter.py Menu

```
Which option would you like to perform? [1 to 5] - 1

Task | Priority
dog | 1
me | 3
house | 2
you | 3
car | 4
mine | 2
ine | 2
```

Figure 2: Intended Outcome: Assignment05_Starter.py Menu 1

```
Which option would you like to perform? [1 to 5] - 2

Task description: new task
Task Priority (1 - 5): 4

    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

Which option would you like to perform? [1 to 5] -
```

Figure 3: Intended Outcome: Assignment05_Starter.py Menu 2

```
Which option would you like to perform? [1 to 5] - 3

# | Task | Priority
1 | dog | 1
2 | me | 3
3 | house | 2
4 | you | 3
5 | car | 4
6 | mine | 2
7 | ine | 2
8 | new task | 4
Choose which to delete: 2
```

Figure 4: Intended Outcome: Assignment05_Starter.py Menu 3

```
Which option would you like to perform? [1 to 5] - 4


    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

Which option would you like to perform? [1 to 5] -
```

Figure 5: Intended Outcome: Assignment05_Starter.py Menu 4

Assigment05_Starter.py ToDoList.txt M

ToDoList.txt

```
dog,1
house,2
you,3
car,4
mine,2
ine,2
new task,4
```

Figure 6: Intended Outcome: ToDoList.txt

```
Which option would you like to perform? [1 to 5] - 5

PS D:\DEV\uw\assignments\_PythonClass\Assignment05>
```

Figure 7: Intended Outcome: Assignment05_Starter.py Menu 5

Declare Variables and constants

The starter assignment python file contained a bunch of starting variables and constants, as seen in figure 8. Also in figure 8, I added a couple of my own, lstData and count.

```
# -- Data -- #
# declare variables and constants
objFile = "ToDoList.txt"   # An object that represents a file
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
lstData = []    # A list that holds the extracted task name from the txt file
count = 0       # A counting variable
```

Figure 8: Variables and Constants

Step 1 – Load a file

In this step, just passing the read argument to the open function creates an error if the file doesn't exist, so I first pass the append argument to first create the file and then read the file, as seen in figure 9. If the ToDolist.txt file does exist, a for loop loops through the file line by line, splitting the string on commas and passing the remaining strings into a list one by one. Knowing that the file contains task comma priority comma return line as the row contents, those elements are passed into a dictionary 'dicRow' and stored as key value pairs. Upon passing the lstData contents to dicRow, the return line character is stripped off. Next, the dicRow dictionary is passed to the lstTable list and stored as a list item.

```
# create a temp variable 'readFile' and pass objFile to it, create the file if it doesn't exist using the append argument, then the read argument
readFile = open(objFile, "a")
readFile = open(objFile, "r")
# use a for loop to loop through the file and do something with the things that are between the commas
for row in readFile:
    lstData = row.split(",")
    dicRow = {"Task":lstData[0],"Priority":lstData[1].strip()}
    lstTable.append(dicRow)
```

Figure 9: Load a File into a dictionary list

Step 2 – Display a Menu

Next, the starter file contained the menu that is used in the script. It displays it using a while loop.

```python
# -- Input/Output -- #
# Step 2 - Display a menu of choices to the user
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
```

Figure 10: Menu While Loop

Step 3 - See the Current Items

If the user chooses '1' at the input prompt, the script will display the contents of lstTable to the console with a pipe separator between the task and priority. I use a for loop to loop through the contents of lstTable and assign the first value of the "task" key to taskN and the value of "priority" key to taskP. I then pass those variables to the print function, as seen in figure 11.

```python
    # Step 3 - Show the current items in the table
    if (strChoice.strip() == '1'):
        print("Task | Priority ")
        for i in lstTable:
            taskN = i["Task"]
            taskP = i["Priority"]
            print(str(taskN) + " | " + taskP)
        continue
```

Figure 11: Menu Choice 1

Step 4 – Add New Items

If the user chooses '2', the script will prompt the user to enter a new task and assign it a priority between 1 and 5. The script captures the task description into a variable of taskName and then captures the priority value into taskPriority. It passes these values into dicRow as key value pairs and then appends the dicRow dictionary to lstTable.

```python
    # Step 4 - Add a new item to the list/Table
    elif (strChoice.strip() == '2'):
        taskName = input("Task description: ")
        taskPriority = input("Task Priority (1 - 5): ")
        dicRow = {"Task":taskName,"Priority":taskPriority}
        lstTable.append(dicRow)
        continue
```

Figure 12: Menu Choice 2

Step 5 – Remove an Item

If the user chooses '3', the script will print the contents of lstTable to the console, along with a number indicating the position of the task in the lstTable. A variable named 'delRow' asks the user to input a number that corresponds to the row that they would like to remove from the list, finds that row and uses the pop method to remove that entry.

```python
    # this step displays to the user each task and asks for which to remove and then finds the one the user specifies
    elif (strChoice.strip() == '3'):
        print("# | Task | Priority")
        for i in lstTable:
            count += 1
            taskN = i["Task"]
            taskP = i["Priority"]
            print(str(count) + " | " + str(taskN) + " | " + taskP)
        delRow = input("Choose which to delete: ")
        lstTable.pop(int(delRow)-1)
        continue
```

Figure 13: Menu Choice 3

Step 6 – Save Current Items to a Txt File

If the user chooses '4', the script will open the text file with the 'write' argument, effectively clearing the contents of the file and then use a for loop to loop through the lstTable list and write each list item to the text file.

```python
    # Step 6 - Save tasks to the ToDoToDoList.txt file
    elif (strChoice.strip() == '4'):
        readFile = open(objFile, "w")
        for i in lstTable:
            readFile.write(i["Task"] + "," + i["Priority"] + '\n')
        continue
```

Figure 14: Menu Choice 4

Step 7 – Exit the Script

Finally, if the user chooses '5', the script will close the file and break out of the while loop.

```
    # Step 7 - Exit program
    elif (strChoice.strip() == '5'):
        readFile.close
        break  # and Exit the program

```

Figure 15: Menu Choice 5

Observations

One observation when making this script was that I forgot to use the .strip() method when reading in the text file initially. This caused the 'priority' key to acquire a '\n' at the end of the text string and my program would error out. I used some online help to understand better how dictionaries work (Dictionaries, n.d.).

Summary

In summary, utilizing all the resources provided to the class and the online lecture, this paper outlines all the steps that were taken to create a python script that results in a successful execution of the intended outcome (Figure 1). Following the steps outlined above will allow for the audience to recreate the presented result.

References

Dictionaries. (n.d.). Retrieved from AfterHours Programming: https://www.afterhoursprogramming.com/tutorial/