Thomas Farmer

2-14-2022

Foundations Of Programming

Assignment05

https://github.com/MtnWolf82/IntroToProg-Python

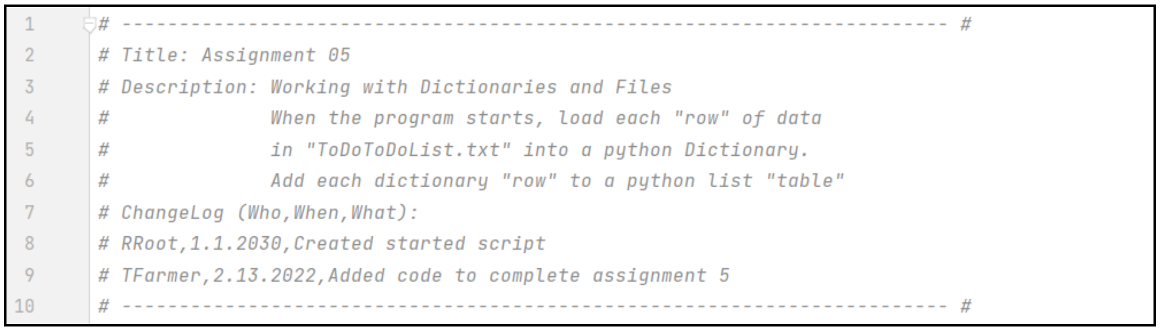
Working with Dictionaries & Files

INTRODUCTION

This week we explored the intricicies of dictionaries, and how they differ from lists and tuples. By utilizing dictionaries, we can now collect titled date from the user. Dictionaries, versus lists, will allow us to sort the data more efficiently for recall at a later time.We will present the user with several different menu options for interacting with the program. The user will be able to review a file's current table data, add or delete rows, save the data to an external Text document, and exit the program.

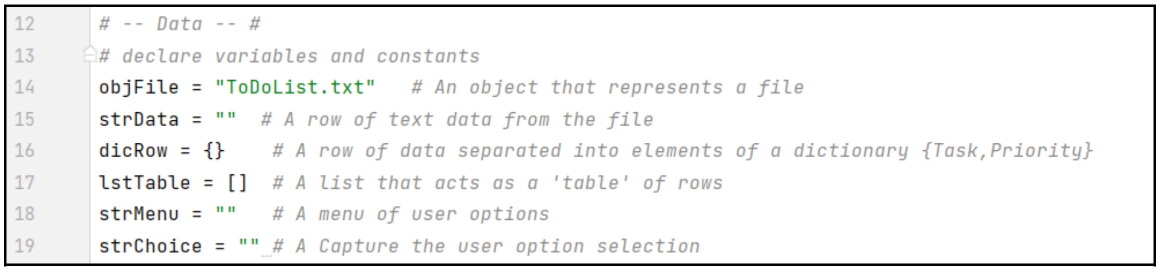
THE PROCESS

1. Open a new PyCharm file and enter your program's header comments. An example of this assignment's comments can be seen below. Note: A starter script was provided in this instance.



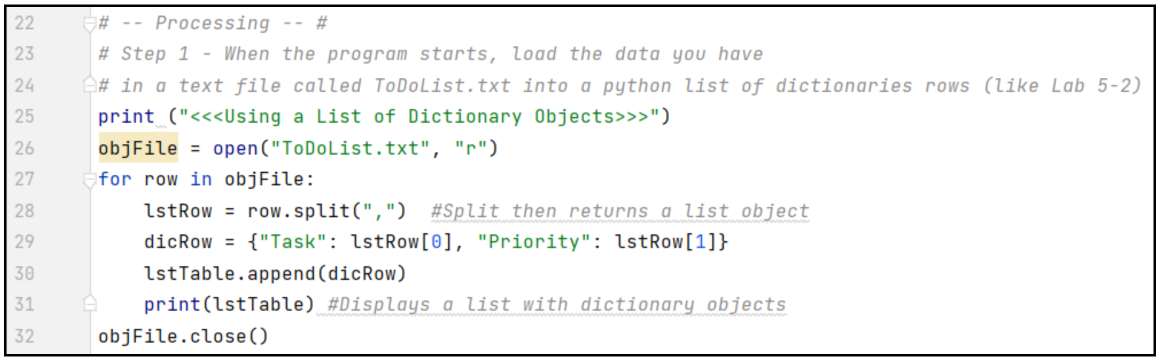
```
# ------------------------------------------------------------------------ #
# Title: Assignment 05
# Description: Working with Dictionaries and Files
#              When the program starts, load each "row" of data
#              in "ToDoToDoList.txt" into a python Dictionary.
#              Add each dictionary "row" to a python list "table"
# ChangeLog (Who,When,What):
# RRoot,1.1.2030,Created started script
# TFarmer,2.13.2022,Added code to complete assignment 5
# ------------------------------------------------------------------------ #
```

2. First, let's establish our variables and row/table lists.



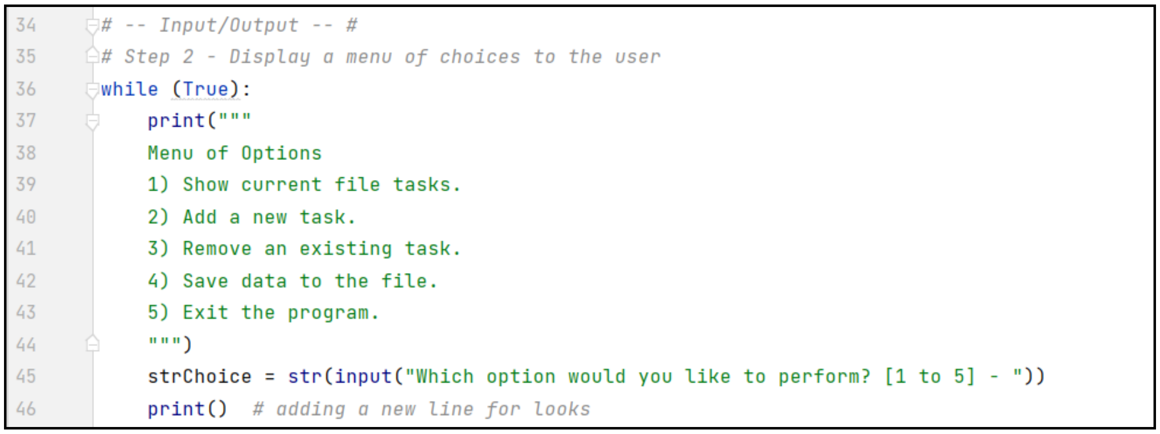
```
# -- Data -- #
# declare variables and constants
objFile = "ToDoList.txt"   # An object that represents a file
strData = ""  # A row of text data from the file
dicRow = {}    # A row of data separated into elements of a dictionary {Task,Priority}
lstTable = []  # A list that acts as a 'table' of rows
strMenu = ""   # A menu of user options
strChoice = "" # A Capture the user option selection
```

3. Our first step is to read our current Text file, saved as “ToDoList”, and provide the user with its table contents. We do this by opening the Text file to be read and pulling data from each row. In this case, each row is a separate task. The task's priority will then be listed to the right of the task. These rows are listed for user to review. Enter the code shown below.



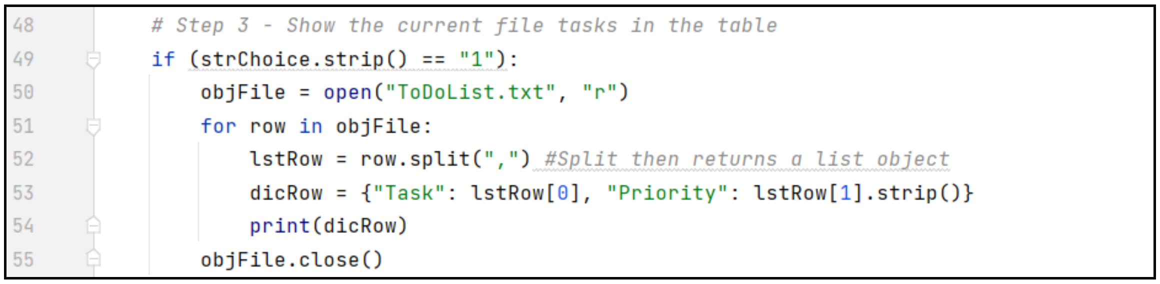
```python
# -- Processing -- #
# Step 1 - When the program starts, load the data you have
# in a text file called ToDoList.txt into a python list of dictionaries rows (like Lab 5-2)
print ("<<<Using a List of Dictionary Objects>>>")
objFile = open("ToDoList.txt", "r")
for row in objFile:
    lstRow = row.split(",")  #Split then returns a list object
    dicRow = {"Task": lstRow[0], "Priority": lstRow[1]}
    lstTable.append(dicRow)
    print(lstTable) #Displays a list with dictionary objects
objFile.close()
```

4. Once again, we will need to put together a menu that will allow the user to access various options. In this instance, we want to the user to be able to show current file data, add a new task, remove an existing task, save the entered data to a file, and exit the program. Enter the code shown below.



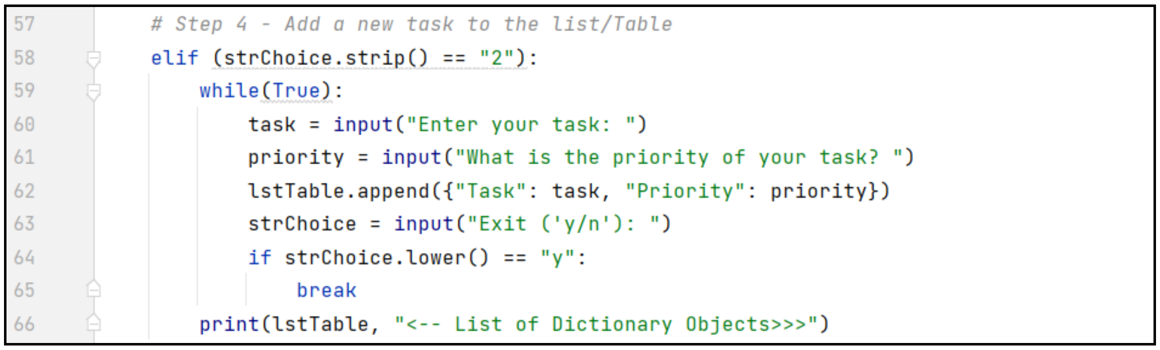
```python
# -- Input/Output -- #
# Step 2 - Display a menu of choices to the user
while (True):
    print("""
    Menu of Options
    1) Show current file tasks.
    2) Add a new task.
    3) Remove an existing task.
    4) Save data to the file.
    5) Exit the program.
    """)
    strChoice = str(input("Which option would you like to perform? [1 to 5] - "))
    print()  # adding a new line for looks
```

5. Selecting option 1 from the menu will allow the user to see tasks currently saved in the referenced ToDoList Text file. Enter the code show below.



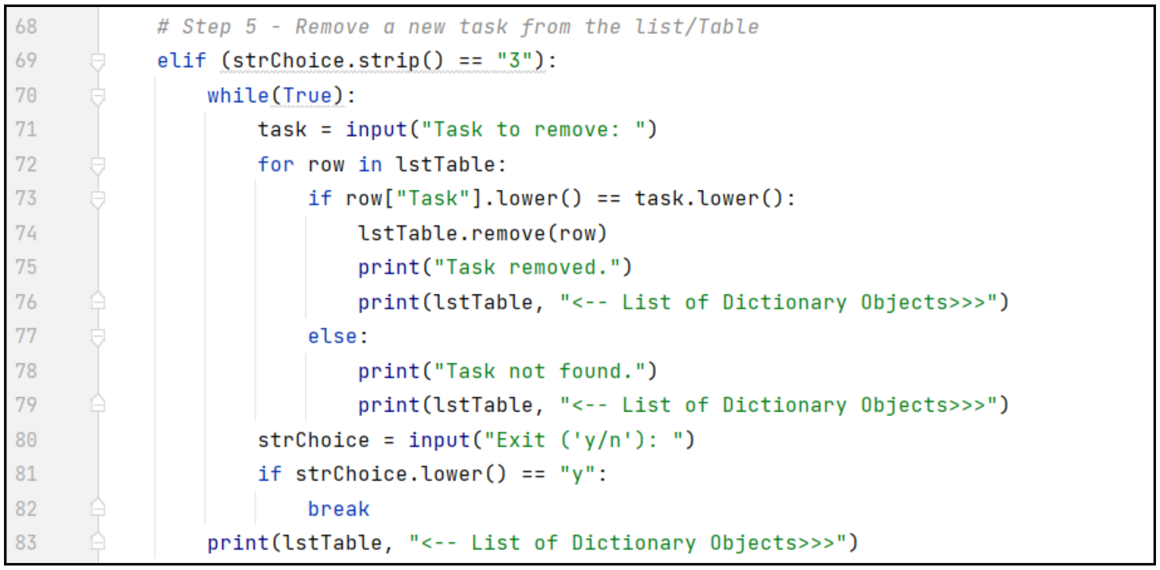
```python
    # Step 3 - Show the current file tasks in the table
    if (strChoice.strip() == "1"):
        objFile = open("ToDoList.txt", "r")
        for row in objFile:
            lstRow = row.split(",") #Split then returns a list object
            dicRow = {"Task": lstRow[0], "Priority": lstRow[1].strip()}
            print(dicRow)
        objFile.close()
```

6. Selecting option 2 from the menu will allow the user to add a new task. They'll be asked for two inputs – the task and the task's priority. The received user input for the “Task” and “Priority” variables is then added to a row, which is then added to the table. Enter the code shown below.



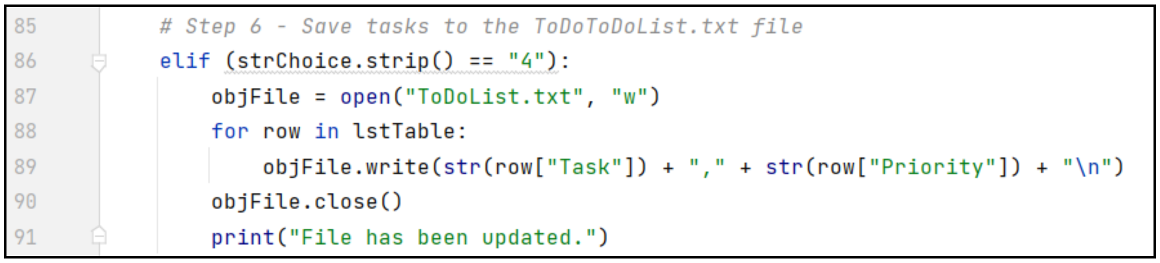
```python
# Step 4 - Add a new task to the list/Table
elif (strChoice.strip() == "2"):
    while(True):
        task = input("Enter your task: ")
        priority = input("What is the priority of your task? ")
        lstTable.append({"Task": task, "Priority": priority})
        strChoice = input("Exit ('y/n'): ")
        if strChoice.lower() == "y":
            break
    print(lstTable, "<-- List of Dictionary Objects>>>")
```

7. Selecting option 3 from the menu will allow the user to remove a new task. As the “Priority” is irrelevant if the task is being removed, only the name of the “Task” is required. Only the row containing the entered “Task” will be removed. Enter the code shown below.



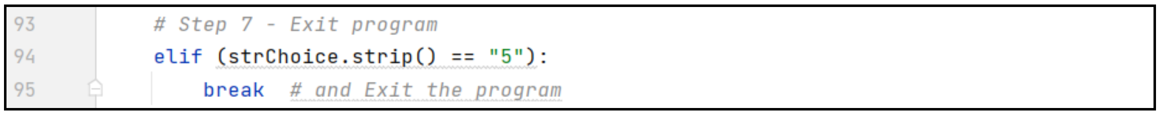
```python
# Step 5 - Remove a new task from the list/Table
elif (strChoice.strip() == "3"):
    while(True):
        task = input("Task to remove: ")
        for row in lstTable:
            if row["Task"].lower() == task.lower():
                lstTable.remove(row)
                print("Task removed.")
                print(lstTable, "<-- List of Dictionary Objects>>>")
            else:
                print("Task not found.")
                print(lstTable, "<-- List of Dictionary Objects>>>")
        strChoice = input("Exit ('y/n'): ")
        if strChoice.lower() == "y":
            break
    print(lstTable, "<-- List of Dictionary Objects>>>")
```

8. Selecting option 4 will allow the user to save their entered table data to the ToDoList Text file. This option will fully overwrite the prior contents of the file, however, the prior contents should be inluded, unless the user decided to delete them using prior menu option 3. Enter the code shown below.



```python
# Step 6 - Save tasks to the ToDoToDoList.txt file
elif (strChoice.strip() == "4"):
    objFile = open("ToDoList.txt", "w")
    for row in lstTable:
        objFile.write(str(row["Task"]) + "," + str(row["Priority"]) + "\n")
    objFile.close()
    print("File has been updated.")
```

9. Last, but not least, selecting option 5 will simply allow the user to exit the program. Nothing too fancy here. Enter the code shown below.



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

SUMMARY

I found the functionality of dictionaries to be pretty logical. I like that they operate similar to other list options, but with the structure of a spreadsheet. This assignment did put to the test a lot of what I learned from the prior assignments, and required me to refer to those notes quite a bit. That said, seeing what I've learned so far come together in one program is pretty nice. Shows me I need to keep working with this material, too, in order to more thoroughly understand how it works.