Jessica Chen

November 12, 2022

Foundations Of Programming: Python

Assignment 05

GitHub URL: https://github.com/jesscUW/IntroToProg-Python

Modify a script that manages a To Do List

Introduction

In this assignment, I will document how I modified a Python script that displays a menu of choices to a user for managing their To-Do list. The user can check current data, add new items, remove items, and also save the data to a text file.

Completing the Script

Creating a New Project in PyCharm

To begin with this assignment, I created a new project in PyCharm with the folder location "_PythonClass/Assignment05" (Figure 1).

Figure 1: Creating a new project in PyCharm

Adding the Starter File

I opened the starter file "Assignment05_Starter.py" and saved it as "Assignment05.py" within my project (Figure 2).

Figure 2: Adding the starter file and saving it as "Assignment05.py" in the project

Script Header

First, I updated the change log in the header (Figure 3).

```
# ChangeLog (Who,When,What):
# RRoot,1.1.2030,Created started script
# Jessica Chen, 11/12/2022, Added code to complete assignment 5
# ------------------------------------------------------------------------ #
```

Figure 3: Screenshot of the updated change log in the header

Declaring variables

Most of the variables needed were already declared in this block so I only modified a bit: I changed “objFile = ‘ToDoList.txt’” to “objFile = None” to set it as a file handle, and created “strFile = ‘ToDoList.txt’” and “lstRow = [ ]” (Figure 4).

```
# -- Data -- #
# declare variables and constants
objFile = None  # File handle
strFile = 'ToDoList.txt'  # Data storage file
strData = ''  # A row of text data from the file
lstRow = []  # List of data
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
```

Figure 4: Declaring the variables

Creating “ToDoList.txt”

Step 1 is to load the data from a text file named “ToDoList” into a dictionary and then create a table of data. I created “ToDoList.txt” with a row of data “Wash dishes, Low” (Task, Priority) so that there will be data loaded when the program starts.

Processing

After opening the file in read mode, a for loop was used to extract the stored data. I used split() method to separate elements based on comas found in the text file. A list would first be returned and then the list would be added into a dictionary. The strip() method was used here to remove any unwanted carriage return. After loading the data, I added the dictionary to “lstTable” using the append() method. The last line in step 1 was written to close the file by close() method (Figure 5).

```
# -- 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)
objFile = open(strFile, 'r')
for row in objFile:
    lstRow = row.split(', ')  # Return a list
    dicRow = {'Task': lstRow[0], 'Priority': lstRow[1].strip()}  # Add list data to dictionary
    lstTable.append(dicRow)  # Create the table
objFile.close()
```

Figure 5: Screenshot of step 1

Input/Output

Step 2 is to display a menu to a user. To allow the user to work with different options several times, the while loop was used to execute the statements repeatedly until the user quits the program. This part already existed in the starter file and there was nothing to change so I kept it as is.

There are five options for a user to choose. if-elif statements were added to evaluate specified conditions. Step 3 is to print data when the user wants to see current data. I created a for loop, followed by the loop body to extract data from “lstTable”. To display the data in the format of “Task, Priority”, I used key subscripts with the separator of a comma and space (Figure 6).

```
# Step 3 - Show the current items in the table
if (strChoice.strip() == '1'):
    print('Your current data is: ')
    for row in lstTable:
        print(row['Task'], row['Priority'], sep=', ')
    continue
```

Figure 6: Screenshot of step 3

Step 4 is to add data to the table when the user selects option 2 from the menu. append() method was used here for adding the dictionary to “lstTable”. Every time a user chooses option 2 and enters another task and its priority, both pieces of data would be added (Figure 7).

```
# Step 4 - Add a new item to the list/Table
elif (strChoice.strip() == '2'):
    print('Type in a task and its priority.')
    strTask = input('Enter a Task: ')
    strPriority = input('Enter its Priority : ')
    dicRow = {'Task': strTask, 'Priority': strPriority}
    lstTable.append(dicRow)
    continue
```

Figure 7: Screenshot of step 4

Step 5 is for the user to remove a task from the list. I used the print() function to print out the list for reference–the user can see all the tasks in the list and then decides which one to remove. “strRemove” was assigned the user’s input. A for loop was added and a nested if statement was put in the loop body. If the value of “Task” in a row equals to the user’s input (“strRemove”), the row would be removed; the remove() method was used to do the removing job (Figure 8).

```
# Step 5 - Remove a new item from the list/Table
elif (strChoice.strip() == '3'):
    print(lstTable)  # Display the list to user for reference
    strRemove = input('Enter the Task to be removed: ')
    for row in lstTable:
        if row['Task'] == strRemove:
            lstTable.remove(row)
    continue
```

Figure 8: Screenshot of step 5

The part of step 6 was written for the option to save the user’s entered data. In order to add the data gotten from the user to a file, the first thing I did was open a file in write mode. Then I used a for loop to format the data shown in the text file–it would look like this: “Task, Priority”. The print() function was used here to print out the message letting the user know that the data has been saved (Figure 9).

```
# Step 6 - Save tasks to the ToDoList.txt file
elif (strChoice.strip() == '4'):
    objFile = open('ToDoList.txt', 'w')
    for row in lstTable:
        objFile.write(row['Task'] + ', ' + row['Priority'] + '\n')
    objFile.close()
    print('Data saved!')
    continue
```

Figure 9: Screenshot of step 6

The last step is to exit the program. The break statement used here allows a user to quit the program. When the user chooses this option, they will see the message of “Bye!” and then exit (Figure 10).

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

Figure 10: Screenshot of step 7

Comments

To help other people understand my code and also be able to remind myself my original intentions in the future, I added some notes. My script is shown as below (Figure 11):

```
# ------------------------------------------------------------------------ #
# Title: Assignment 05
# Description: Working with Dictionaries and Files
#              When the program starts, load each "row" of data
#              in "ToDoList.txt" into a python Dictionary.
#              Add the each dictionary "row" to a python list "table"
# ChangeLog (Who,When,What):
# RRoot,1.1.2030,Created started script
# Jessica Chen, 11/12/2022, Added code to complete assignment 5
# ------------------------------------------------------------------------ #

# -- Data -- #
# declare variables and constants
objFile = None  # File handle
strFile = 'ToDoList.txt'  # Data storage file
strData = ''  # A row of text data from the file
lstRow = []  # List of data
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
```

```
# -- Processing -- #
# Step 1 - When the program starts, load any data you have
# in a text file called ToDoList.txt into a python list of dictionaries rows (like Lab 5-2)
objFile = open(strFile, 'r')
for row in objFile:
    lstRow = row.split(', ')  # Return a list
    dicRow = {'Task': lstRow[0], 'Priority': lstRow[1].strip()}  # Add list data to dictionary
    lstTable.append(dicRow)  # Create the table
objFile.close()


# -- 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
 # Step 3 - Show the current items in the table
 if (strChoice.strip() == '1'):
     print('Your current data is: ')
     for row in lstTable:
         print(row['Task'], row['Priority'], sep=', ')
     continue
 # Step 4 - Add a new item to the list/Table
 elif (strChoice.strip() == '2'):
     print('Type in a task and its priority.')
     strTask = input('Enter a Task: ')
     strPriority = input('Enter its Priority : ')
     dicRow = {'Task': strTask, 'Priority': strPriority}
     lstTable.append(dicRow)
     continue
 # Step 5 - Remove a new item from the list/Table
 elif (strChoice.strip() == '3'):
     print(lstTable)  # Display the list to user for reference
     strRemove = input('Enter the Task to be removed: ')
     for row in lstTable:
         if row['Task'] == strRemove:
             lstTable.remove(row)
     continue
 # Step 6 - Save tasks to the ToDoList.txt file
 elif (strChoice.strip() == '4'):
     objFile = open('ToDoList.txt', 'w')
     for row in lstTable:
         objFile.write(row['Task'] + ', ' + row['Priority'] + '\n')
     objFile.close()
     print('Data saved!')
     continue
 # Step 7 - Exit program
 elif (strChoice.strip() == '5'):
     print('Bye!')
     break  # and Exit the program
```

Figure 11: Screenshot of my script

Running the Script

I ran Assignment05.py in both PyCharm and Shell. Following the prompts, I chose option 1 to check the current data. The data was correctly loaded as “Wash dishes, Low” from the “ToDoList.txt” I created at the beginning. Then I chose option 2 to add a task and its priority. When I selected option 3, the list showed all my data. I entered “Wash dishes” to remove this task. And then I typed 4 to save the data. Finally, I chose option 5 to exit the program. The script ran as expected (Figure 12 and 13); the data was correctly updated (“Wash dishes” deleted) and written to the text file “ToDoList.txt” (Figure 14).

```
/Users/yunjuchen/Documents/_PythonClass/Assignment05/venv/bin/python /Users/yunjuchen/Documents/_PythonClass/Assignment05/Assignment05.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] - 1

Your current data is:
Wash dishes, Low

    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] - 2

Type in a task and its priority.
Enter a Task: Pay bills
Enter its Priority : High

    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] - 3

[{'Task': 'Wash dishes', 'Priority': 'Low'}, {'Task': 'Pay bills', 'Priority': 'High'}]
Enter the Task to be removed: Wash dishes

    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] - 4

Data saved!

    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] - 5

Bye!

Process finished with exit code 0
```

Figure 12: Screenshot of Assignment05.py running in PyCharm

```
= RESTART: /Users/yunjuchen/Documents/_PythonClass/Assignment05/Assignment05.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] - 1

Your current data is:
Wash dishes, Low

    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] - 2

Type in a task and its priority.
Enter a Task: Pay bills
Enter its Priority : High

    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] - 3

[{'Task': 'Wash dishes', 'Priority': 'Low'}, {'Task': 'Pay bills', 'Priority': 'High'}]
Enter the Task to be removed: Wash dishes

    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] - 4

Data saved!

    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] - 5

Bye!
```

Figure 13: Screenshot of Assignment05.py running in Shell

Figure 14: Verifying that the file has correct data

Summary

Through Module 05, I learned more about Lists and gained an understanding of Dictionaries and loading data from a text file. In addition, I got to work with a template created by another person and modified it. I added code to complete a script displaying a menu to a user–the user can choose which action to be performed until they exit the program.