

Omar Lozano

May 16, 2022

Foundations of Programming: Python

Assignment 05

<https://github.com/omega609/IntroToProg-Python>

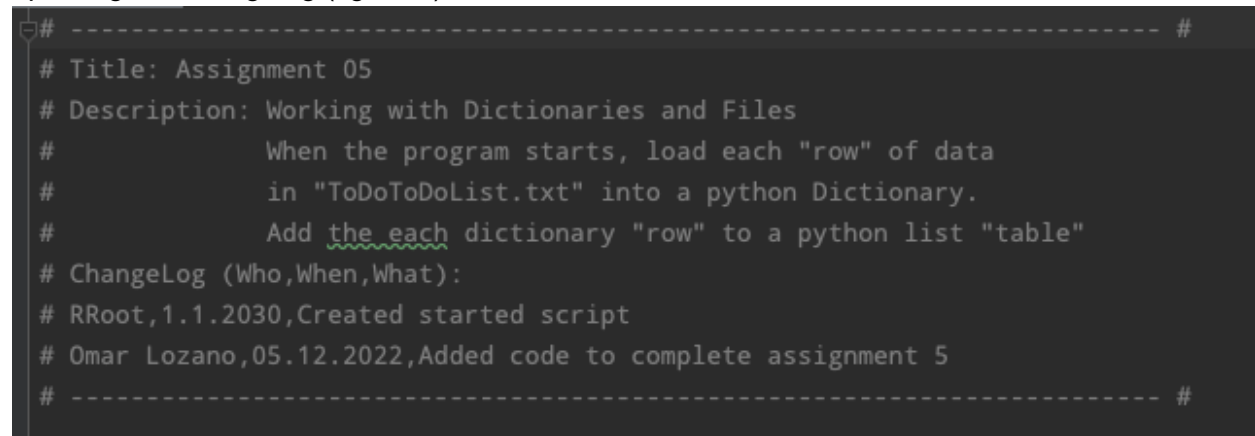
# Using a List of Dictionaries to Create a Table

## Introduction

For this assignment I will be creating a table using dictionaries. The columns of the table are representative of a task and a priority. This table will be created and manipulated through the use of a menu where the user will be able to choose if they want to add, remove, view, or save the data to a text file.

## Creating the Script

This script started with pseudo code and a template provided by our instructor outlining the steps that would need to be taken to complete this assignment. Per the instructions I began by updating the changelog (figure 1).

A screenshot of a code editor with a dark background. The code is in a light gray font. It shows a Python script header with several comments. The first line is a separator line with dashes. The second line is a comment: "# Title: Assignment 05". The third line is a comment: "# Description: Working with Dictionaries and Files". The next two lines are comments describing the program's logic: "# When the program starts, load each 'row' of data" and "# in 'ToDoToDoList.txt' into a python Dictionary.". The following line is a comment: "# Add the each dictionary 'row' to a python list 'table'", where "the each" is underlined in green. Then, there is a comment: "# ChangeLog (Who,When,What):". This is followed by two lines of changelog entries: "# RRoot,1.1.2030,Created started script" and "# Omar Lozano,05.12.2022,Added code to complete assignment 5". The final line is another separator line with dashes.

```
# ----- #  
# 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 (Who,When,What):  
# RRoot,1.1.2030,Created started script  
# Omar Lozano,05.12.2022,Added code to complete assignment 5  
# ----- #
```

**Figure 1: Update name and changes to code**

Next was the declaration of variables which would be used throughout the script. These variables were included in the template provided by our instructor (figure 2).

```

# -- 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

```

**Figure 2: Update name and changes to code**

Following the declaration of variables, I used the open function and append mode to create and load data into a text file. I then created a dictionary using “tasks” and “priorities” as my keys and included value pairs. These represent the rows of the table I am creating. This is also where I make a list of dictionaries, lstTable (figure 3).

```

# 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('ToDoList.txt', "a")
dicRow = {"Task": "Write Script", "Priority": "High"}
objFile.write(dicRow["Task"] + "," + dicRow["Priority"] + "\n")
lstTable = [dicRow] # Create list from dicRow
objFile.close()

```

**Figure 3: Generate file and append new list of dictionaries**

Presenting the user with a menu of choices to view, remove, add, or save data was next, but this was provided to me by our instructor and is done so using a while loop. The while loop allows the user to choose different options from the menu and add many dictionaries to the list created previously (figure 4).

```

# 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 4: User menu to manipulate data**

To address the first menu option, show current data, an if statement and for loop were used to iterate through the list and print to the screen the list items, dictionaries.

```
# Step 3 - Show the current items in the table
if (strChoice.strip() == '1'):
    for row in lstTable: # Iterating through items in list and printing out as a row
        print(row)
```

**Figure 5: If statement and for loop to present data**

To add to the list of tasks and priorities a conditional “elif” statement was used to ask the user to enter a new task and priority and assign these items to the “dicRow” variable to create a new list item, a dictionary. I could then append the new list item to the end of the list created above using the .append() function and show the item was added successfully (figure 6).

```
# Step 4 - Add a new item to the list/Table
elif (strChoice.strip() == '2'):
    strTask = input("Enter a task: ") # Have user enter a new task
    strPriority = input("Enter the priority: ") # Have user enter new priority
    dicRow = {"Task": strTask, "Priority": strPriority.strip()} # Creating new dictionary item
    lstTable.append(dicRow) # Append new dictionary item to list
    print(lstTable) # Show user new item added
```

**Figure 6: Add new task and priority to list**

Removing an item from the list required another conditional statement with a for loop. After asking the user to enter the task they would like to remove, a for loop is used to iterate through the list (lstTable) and compare if the string entered matches the string of a key in one of the dictionaries in the list. If the key is found .remove() function is used to remove the dictionary item (figure 7).

```
# Step 5 - Remove an existing item from the list/Table
elif (strChoice.strip() == '3'):
    strTaskRemove = input("Enter the task to remove: ")
    for row in lstTable: # Iterate through list
        if row["Task"].lower() == strTaskRemove.lower(): # Looking for a key that matches item entered by user
            lstTable.remove(row) # Remove both key and value from list
    print(strTaskRemove, "has been removed") # Letting user know item removed
```

**Figure 7: Remove task and priority from list**

Saving the newly created list of items was done almost the same way I created the file except that I did not append this time. Instead, I used “w” to write over the existing file (figure 8).

```
# Step 6 - Save tasks to the ToDoToDoList.txt file
elif (strChoice.strip() == '4'):
    objFile = open("ToDoList.txt", "w")
    objFile.write(dicRow["Task"] + "," + dicRow["Priority"] + "\n") # Appending any new dictionary items to list
    objFile.close()
    print("Your tasks have been saved to the file 'Todolist.txt' ") # Let user know item has been added to file

    continue
```

### **Figure 8: Save task list to text file**

Finally, to exit the program, I included a statement to let the user know they had exited the program (figure 9). Throughout the program the `.strip()` function was used to remove any trailing spaces that might have been entered when a user was making a choice.

```
# Step 7 - Exit program
elif (strChoice.strip() == '5'):
    # TODO: Add Code Here
    print("You have exited the program") # Let user know they have successfully exited program
    break # and Exit the program
```

### **Figure 9: Exit the program**

## **Validating the Script**

Just like the previous assignment, validating this script works as intended requires two approaches: (1) through Pycharm and (2) using the Shell Operator.

### ***Through Pycharm:***

First, I choose option 2 to enter a new task and priority, because I have included a print statement, I am shown all tasks in the priority including the new task, sleep with a priority of high (figure 10). Next I choose option three to remove a task and then option one to show the task has been removed (figure 11). Finally I save the data to a file and exit. (figure 12). Next I go to Finder >Documents >\_PythonClass >Assignment05 >ToDoList.txt and open the file and verify a file has been created with the user input (figure 13).

### ***Using the Shell Operator:***

To validate the code works in the command shell I copy the path and paste it to the command line. I run the script using different data from what was used in PyCharm, a task of run, and capture the results of the code to demonstrate that the code works as intended (figure 14, figure 15). Finally, I use Finder again to locate the new text file created (ToDoList.txt) and validate the new data is written to the file (Figure 16).

```
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

Enter a task: Sleep
Enter the priority: High
[{'Task': 'Write Script', 'Priority': 'High'}, {'Task': 'Sleep', 'Priority': 'High'}]
```

**Figure 10: New task added “sleep”**

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

Enter the task to remove: Write Script
Write Script has been removed
```

**Figure 11: Remove task of “Write Script”**

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

{'Task': 'Sleep', 'Priority': 'High'}
```

**Figure 12: Display tasks with “Write Script” removed**

  
Sleep,High

 **ToDoList.txt**

**Figure 13: File created with new task**

```
amilapatel — Assignment05.py — 85x24
((base) Amilas-MacBook-Pro:~ amilapatel$ python3 /Users/amilapatel/Documents/_PythonCl
ass/Assignment_05/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] - 2

Enter a task: Run
Enter the priority: Low
```

**Figure 14: Run code in shell**

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

Enter the task to remove: write script
write script has been removed

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

Your tasks have been saved to the file 'Todolist.txt'
```

**Figure 15: Remove existing task**

```
Run, Low
```

 **ToDoList.txt**

**Figure 16: Text file created**

## Summary

Using dictionaries in a list for me proved to be a difficult concept since each item in the list consists of the key and value pair, making it seem as though there was a larger list than needed. This led to me having trouble at first removing tasks, as I thought I had to remove both the key and value pair and the user could remove a new task, but this was clarified in the discussion board and I was able to get my script to run.