Glenn B. Dacones

May 17, 2022

Foundation of Programming: Python

Assignment 05

https://github.com/uwpce-Dgbjccd05/IntroToProg-Python

To Do List Script

Introduction

Using the provided template script, I will show you how I will create and execute a script using the inputs from the user and save the user inputs to a file. The user inputs will be placed in dictionary collection, and eventually into a list/Table collection. The inputs shall consist of task and priority e.g., low, medium, high.

How to Load the Data When the Program Runs

In this example, the collected data is placed into a dictionary collection. Then it is placed into a list/Table, and finally written into a text file (Figure 1).

```
dicRow = {"Task":"Mow", "Priority":"High"}
objFile = open(strFile, "w")
objFile.write(dicRow["Task"] + "," + dicRow["Priority"] + "\n")
lstTable = [dicRow]
objFile.close()
```

Figure 1. An example of how to load the data using a dictionary collection, then into a list/Table, and written to a text file.

How to Show the Current Items in the List/Table

In this step, when the user chooses choice 1), the user is shown the current items in the list/Table. To do this, once the choice is confirmed, a for loop is used to extract the row(s) from list/Table. Once the rows have been extracted, display the current items with the print() function (Figure 2).

```
if (strChoice.strip() == '1'):
    for row in lstTable:
        print(row)
    continue
```

Figure 2. An example of how to show the current items in the list/Table.

How to Add a New Item to the List/Table

In this step, when the user chooses choice 2), the user will be asked to add a new item (task and priority). In this example, the user inputs are assigned to variables. Also, I decided to capitalize the first letter of the inputs. I then proceeded to place the user inputs into dictionary collection and eventually appended to the list/Table (Figure 3).

```
elif (strChoice.strip() == '2'):
    strTask = input("Enter a Task: ")
    strTask = strTask.capitalize()
    strPriority = input("Enter a Priority, High, Medium, Low: ")
    strPriority = strPriority.capitalize()
    dicRow = {"Task":strTask, "Priority":strPriority}
    lstTable.append(dicRow)
    continue
```

Figure 3. An example of how add a new item to the list/Table.

How to Remove an item from the List/Table

When the user chooses choice 4), the user will be asked which task to be removed from the list/Table.

First, the user input is saved to a variable. Since the items in the list/Table are capitalized, I modified the variable content to be capitalized as well. To determine the items in the list/Table, I used the for loop to extract the items in the list/Table. Using the if statement, I compared the task to be removed and the items from the list/Table. If a match is found, the item/task is removed from the list/Table (Figure 4).

```
elif (strChoice.strip() == '3'):
    strRemove = input("Which task would you like to remove?: ")
    strRemove = strRemove.capitalize()
    for row in lstTable:
        if (row["Task"] == strRemove):
            lstTable.remove(row)
            print("\nThe task has been removed from the list/Table!")
    continue
```

Figure 4. An example of how to remove an item from the list/Table.

How to Save the Items to the ToDoList.txt File

To save the items to the ToDoList.txt file, the user must choose choice 5) from the Menu of Options. Once confirmed, the items in the list/Table are written to the ToDoList.txt file. Using the handle objFile, the ToDoList.txt file is created using the open() function with attribute “w”. Using the for loop, the row is extracted from this list/Table. Once extracted, the keys (e.g., Task, Priority) are used to write the Task, Priority combination to the ToDoList.txt file (Figure 5).

```
elif (strChoice.strip() == '4'):
    objFile = open(strFile, "w")
    for row in lstTable:
        objFile.write(row["Task"] + "," + row["Priority"] + "\n" )
    objFile.close()
    print("Your data has been saved to", strFile + "!")
    continue
```

Figure 5. An example of how to save the items to the ToDoList.text file.

How to Exit the Program

To exit the program, the user must choose choice 5). Once confirmed, the program is exited using the break command (Figure 6).

```
elif (strChoice.strip() == '5'):
    print("You are now exiting the program!")
    break  # and Exit the program
```

Figure 6. An example of how to exit the program.

Below shows the script running in PyCharm (Figure 7).

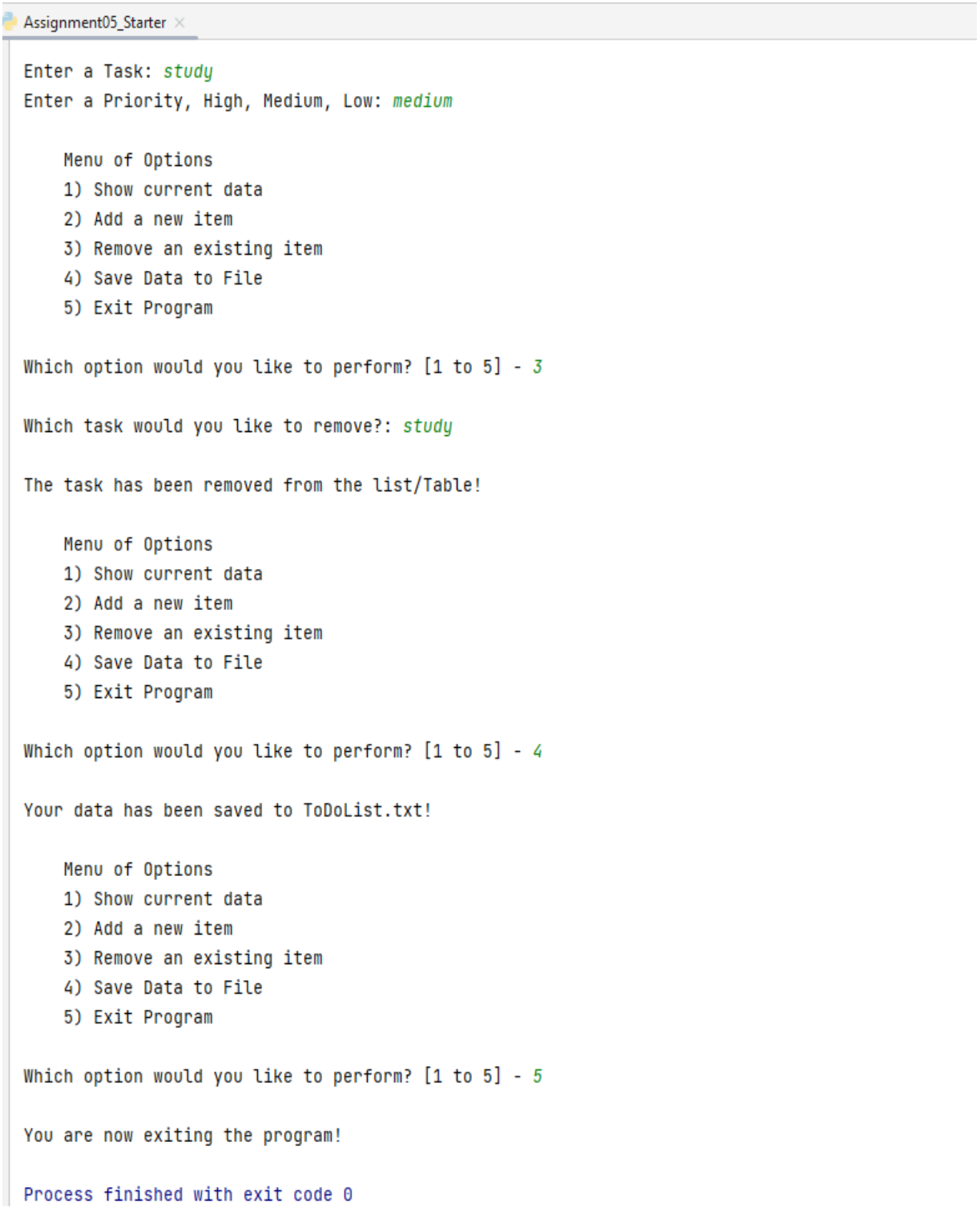

Figure 7. A screenshot of the script running in PyCharm.

Below shows the script running in the command shell (Figure 8).

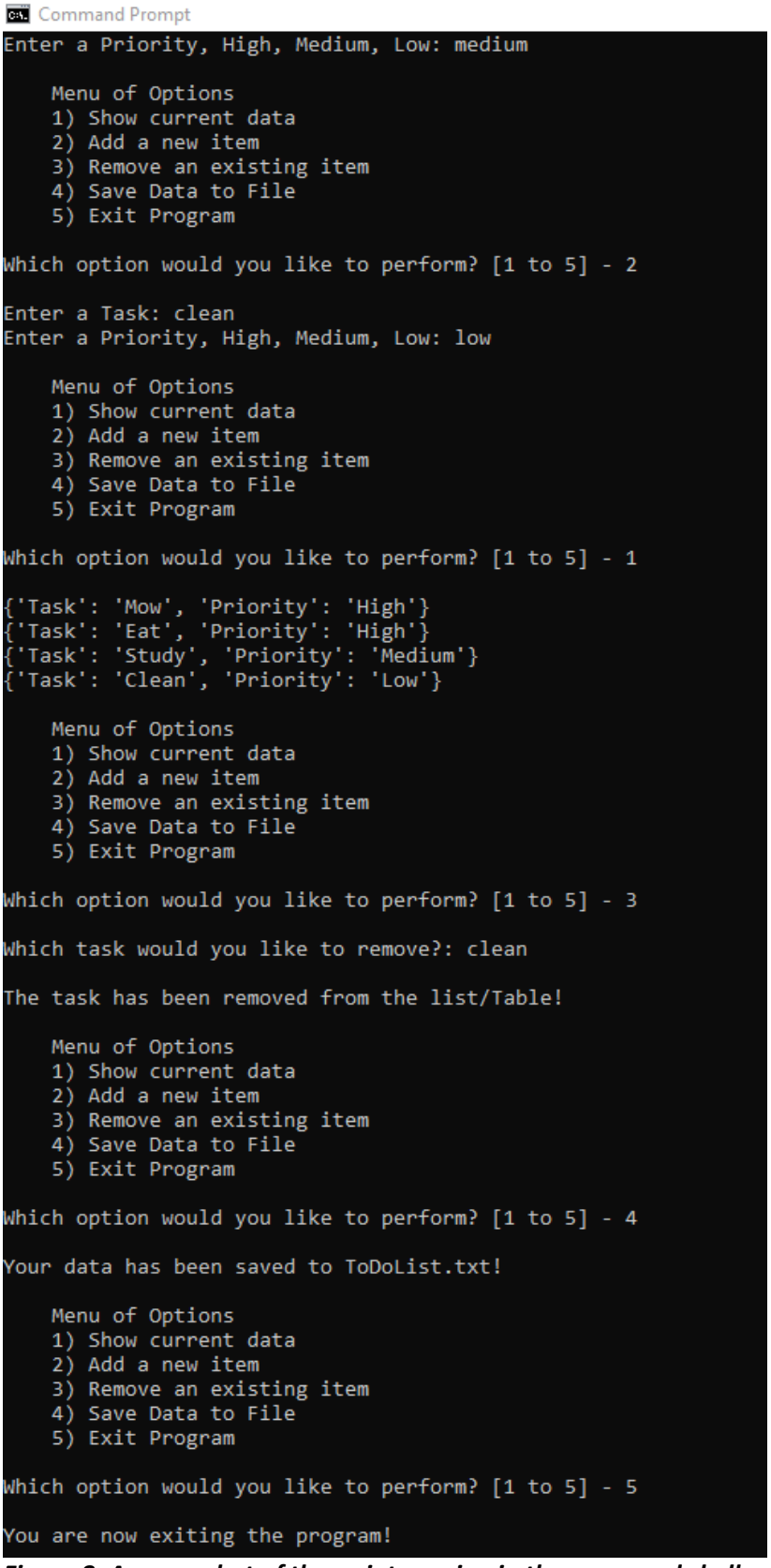

Figure 8. A screenshot of the script running in the command shell.

Below shows the tasks and priorities written to the ToDoList.text file (Figure 9).

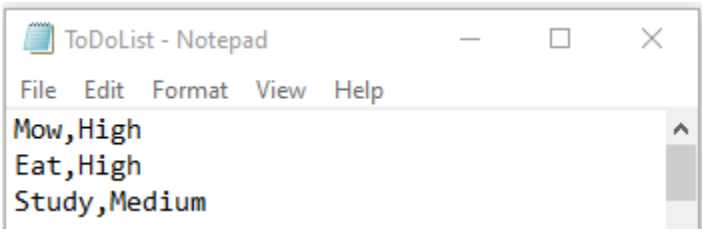

Figure 9. Verifying that the file has data.

Summary

In this assignment, I was able to complete the assigned tasks. Using the provided template, I added the necessary codes to complete the tasks. In this assignment, we used a combination of dictionary and list collections. Understanding the difference between key for dictionary, and index for list, I was able to maneuver through some tight road. It started out rough, but I was able bring this assignment home.