Mark Daza
February 15th, 2022
Foundations of Programming
Assignment05
https://github.com/markdazab/IntroToProg-Python

ToDo List

Introduction
This document will record the steps that I took in performing Assignment05. Briefly, for this first assignment, we will modify a python script provided by Randal and to manage a "ToDo list" that would provide the user with multiple menu options. The user will be allowed to add data to a list, display its contents, or exit the program with an opportunity to save the list to a file.

Process
After completing tasks 1-4, I started working on this week's assignment script. As mentioned in the introduction, we were asked to modify a script to ask the user to select from a menu of options. The user can choose to show the ToDo list, add a new "Task," remove a "Task," save the file or exit the program. The assignment notes did not provide an example of expected results, but the backbone and concept were covered during Lab5-1 and Lab5-2.

Following the pseudo-code provided in the starting script, I planned to tackle the assignment step-by-step. First, I would load the data to a table of rows (a list) from the ToDoList.txt file also provided. To do so, I had the script open and read each row in the file and, in that order, load it to elements of a dictionary that would then get loaded to a list that would act as a table of rows, which I called lstTable. Every time the scrip went through a row from the text file, it would append and add the values to lstTable. (Figure 1)

```
# 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(",")
    dicRow = {"Task": lstRow[0], "Priority": lstRow[1].strip()}
    lstTable.append(dicRow)
objFile.close()
```

Figure 1. Scrip for ToDoList Step 1

The next step was all about input and output. I believe step 2 was provided in the script, so I moved on to step 3, which would show the current contents of the lstTable. If the user selected option one from the menu, it printed a header which I named "Task" and "Priority." Then using a for loop, I had it go through each row in lstTable and print the contents of each

row of data which were a dictionary using a key to print the values of the “Task” and “Priority” keys(). (Figure 2)

```
# Step 3 - Show the current items in the table
if strChoice.strip() == '1':
    print(("Task"+"\t|\t"+"Priority"))
    for row in lstTable:
        print(row["Task"]+"\t|\t"+row["Priority"])
    continue
```

Figure 2. Scrip for ToDoList Step 3

Next, we had step 4, to let the user add a new item to the list/table. By far, the easiest step of the assignment since I only had to ask the user’s input to create a new task and priority. For this, I used the input function to add data to NewTask and NewPriority variables and pass them on as values to a dictionary with “Task” and “Priority” keys. Finally, using the append () list method, I added the dictionary row to lstTable. (Figure 4)

```
# Step 4 - Add a new item to the list/Table
elif strChoice.strip() == '2':
    NewTask = input("Enter name of task: ")
    NewPriority = input("Enter priority of task: [1-5] - ")
    dicRow = {"Task": NewTask, "Priority": NewPriority}
    lstTable.append(dicRow)
    continue
```

Figure 3. Scrip for ToDoList Step 4

Step 5 was probably the hardest one to figure out for this assignment, mainly because I have a terrible memory of remembering all the list methods. In this case, I knew that to remove an item from the list/table, I had to get the user’s input and use a conditional statement that would trigger its removal if found in the list. I started with getting the user’s input to request the name of the task they wanted to remove and assigned it to the variable removeChoice. Then, using a for loop, I had my code run through each row and, using an if condition, find if removeChoice matched the value of any “Task” key. Because the value matching is case-sensitive, I used the lower() string method to convert both removeChoice and key-value lowercase. In case of a match, using the remove() method deleted the entire row from lstTable, and printed a text informing the user a task had been removed. (Figure 4)

```
# Step 5 - Remove a new item from the list/Table
elif strChoice.strip() == '3':
    removeChoice = input("What task would you like to remove? ")
    for row in lstTable:
        if row["Task"].lower() == removeChoice.lower():
            lstTable.remove(row)
            print("Task removed")
    continue
```

Figure 4. Scrip for ToDoList Step 5

Step 6 was saving the ToDo list to a file. If the user selected this option, I had the scrip open and write on the ToDoList.txt file. Using a for loop to go through each row in lstTable, I wrote a row to the file making sure I created a new line after, so when the loop started again, it would write the next row in a new line. Once done with the loop, I printed a text letting the user know it had saved the changes. (Figure 5)

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

Figure 5. Scrip for ToDoList Step 5

Step 7 was pretty straightforward, and initially, I had nothing major or complicated planned for it. However, I once tried it and exited the script without saving my changes; very annoying if you forget to save your list. I wanted to create an option for the user to decide if they wanted to exit without saving their changes. So I made a finalChoice variable to get the user's input if they would like to keep their changes. If they entered "Y," the script would perform pretty much step 4. But if they entered "N," it would break the while loop. (Figure 7)

```
# Step 7 - Exit program
elif strChoice.strip() == '5':
    finalChoice = input("Would you like to save your changes? ['Y' or 'N'] - ")
    if finalChoice.lower() == "y":
        objFile = open(strFile, "w")
        for row in lstTable:
            objFile.write(row["Task"] + "," + row["Priority"] + "\n")
        objFile.close()
        print("Changes saved. Good Bye")
    elif finalChoice.lower() == "n":
        print("Changes not saved. Good Bye")
    break  # and Exit the program
```

Figure 6. Scrip for ToDoList Step

Summary

List and dictionaries were the focus of this week's Foundations of Programming. These are pretty exciting concepts, and I agree with Randal and Grace's comment about this week's more complex course assignment. Below are the screenshots of the script running both in PyCharm and IDLE. (Figure 8 and Figure 9)

I can see how this week's dictionary and list concept can get confusing, but luckily I think I got the hang of it thanks to the lectures and supplemental content provided by the book and videos this week. Not much to add other than I am very excited to see what other people in the class came up with and looking forward to learning from their thought processes.

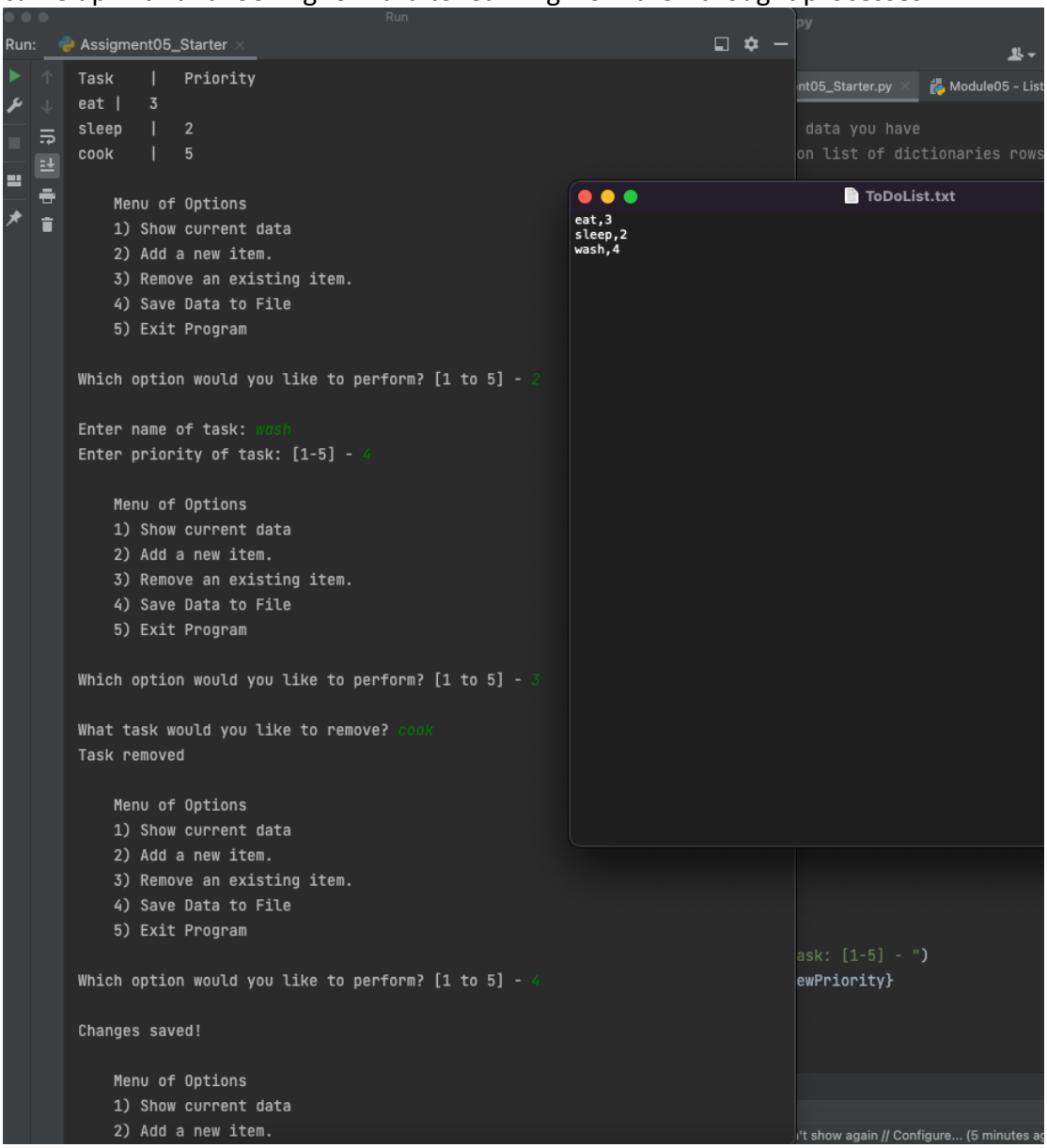

Figure 8. Final script for Assignment05 running on PyCharm and ToDoList.txt.

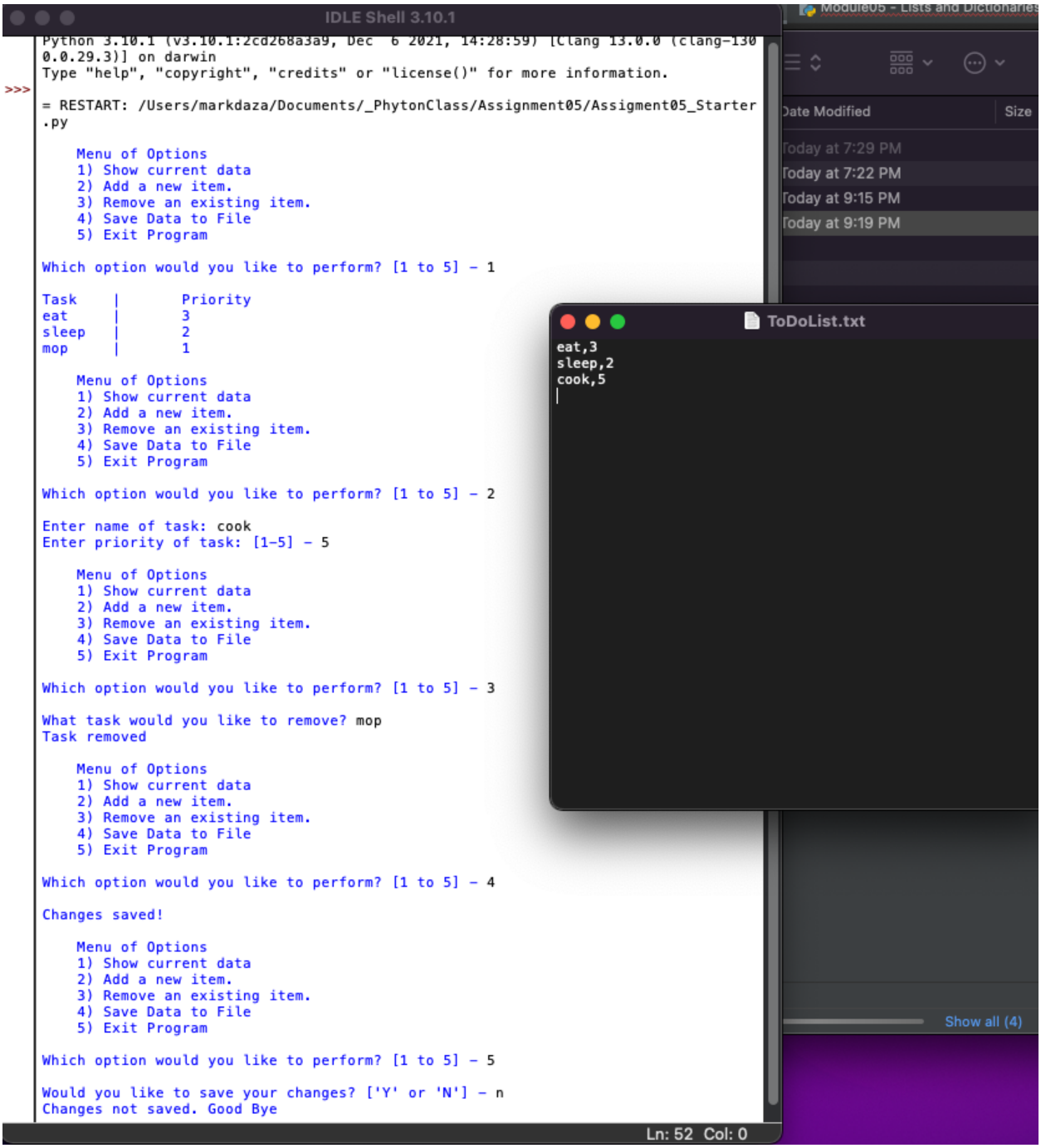

Figure 9. Final script for Assignment05 running on IDLE and ToDoList.txt.