
Mariia Kaminskaia

February 14, 2023

IT FDN 110 A Wi 23: Foundations Of Programming: Python

Assignment_05

Link to GitHub: <https://github.com/moriia/IntroToProg-Python>

Create a ToDo list

Introduction

In this assignment, I had to modify a new script that manages a ToDo list, using a dictionary object. Add, remove, display entered data, and save entered data to the file. This document summarizes the steps I performed to complete the assignment.

Upload a file to GitHub

I uploaded a file to GitHub as it was requested, and made a connection between Pycharm and GitHub by opening the project from the remote repository.

Processing data

As a first step in the assignment, I needed to load any data to a text file called ToDoList.txt using a python list of dictionary rows. For this purpose, I opened a file “ToDoList.txt”, and declared the dicRow with two keys: ‘task’ and ‘priority’ each pair with its value. After the dictionary row was declared, I added it to the table. At the end of this block, the table was saved to the file (**Figure 1**).

```
# -- 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 (strFileName, 'w')
dicRow = {"task": "Do homework" , "priority": "Medium"}
lstTable.append(dicRow)
objFile.write(dicRow["task"]+', '+dicRow["priority"]+'\n')
objFile.close()
```

Figure 1. Load data to the text file at the beginning of the program.

Input/Output

When the user will start to interact with the program the 1st thing he will see will be a menu of options with the request to the user to choose one of them to perform the desired action.

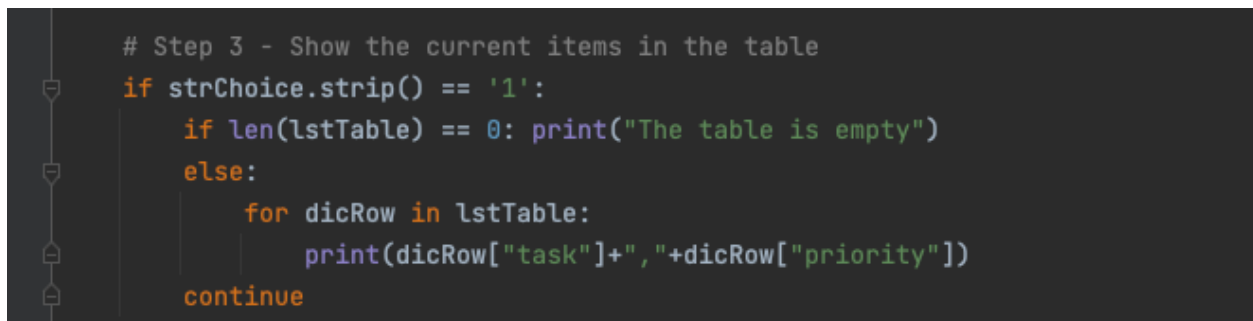
User can:

- See the items in the table
- Add a new item to the table
- Delete an item from the table
- Save changes to the file
- Exit the program

Each block, except the exit program, returns the user to the main menu.

Step 1. Show the current items in the table

In our case, if the user selects '1' he will be able to see data in the table. I declared `lstTable` variable at the beginning of the program, and I know it does exist. In my program, I do not check if a file exists or if there is data in the file. I assumed that if the user will add data he will expect to see it in the data displayed, so I am working only with the data existing in the table. However, since we have the option to remove the data, I can assume that `lstTable` can be empty in this case. To catch this scenario first I check what is the length of the table, or in other words how many rows exist in the table (**Figure 2**)



```
# Step 3 - Show the current items in the table
if strChoice.strip() == '1':
    if len(lstTable) == 0: print("The table is empty")
    else:
        for dicRow in lstTable:
            print(dicRow["task"]+" "+dicRow["priority"])
        continue
```

Figure 2. The block of code to show the user the data which exists in the table (not in the file).

If there are 0 rows in the table, it means that there is nothing that can be displayed, the user will be notified and returned to the main menu.

In case there is data in the table, I will run the loop for each dictionary row, so as a result I will print values existing in the table according to its key (in our case 'task' and 'priority' (**Figure 2**).

Step 2. Add a new item to the list/Table

If the user selects '2' the block of code which will add a new item to the table will be started. Instead of declaring two variables that will store entered data, I am using the global variable `dicRow` where I call `input()` and pair values to specified keys (**Figure 3**).

```
# Step 4 - Add a new item to the list/Table
elif strChoice.strip() == '2':
    dicRow={"task":input("Please enter a task: "),
            "priority":input("Please enter a priority: ")}
    lstTable.append(dicRow)
    print("Entered items are added to the list, but not saved\n")
    continue
```

Figure 3. Adding new data to the table.

After storing the user input as a dictionary row, I added it to the table. Since I do not want the existing item will be deleted or overwritten, I am using an append method to add a new dictionary row to the table. After it's done, the user will be notified that data was added to the table, but was not saved, and the user returns to the main menu (**Figure 3**).

Step 3. Remove a new item from the list/Table

In case the user chooses option '3' the block of code which deletes the last row in the table is used. It is possible that the user will try to delete an item from the empty table, to prevent it I do check if there is objects in the table. If there is no at least one row in the table, the user will be informed that there is no data in the list and will be redirected to the main menu (**Figure 4**).

```
# Step 5 - Remove a new item from the list/Table
elif strChoice.strip() == '3':
    if len(lstTable) == 0: print("The table is empty\n")
    else:
        del lstTable[-1]
        print("The last added item removed\n")
    continue
```

Figure 4. Remove data from the table if there is at least one item in the table.

There are several methods that can delete the last item from the table, in my work I used the del statement. The del operator deletes the item at the specified index location from the list. The last item can be deleted without calculating the length, by using the negative index -1. (**Figure 4**). As soon as the item will be removed a prompt message will be sent to the user, and the user returns to the main menu.

Step 4. Save tasks to the ToDoToDoList.txt file

If the user chooses option '4' the data existing in the table will be written to the file. For this time I do not check if the table of data is empty or not, since if the user wants to save an empty file he absolutely can do it. To save the data to the file, first I open it in a mode w - writing - the file will be overwritten. For each dictionary row from the table, I will write values to the file one by one, by using their keys (**Figure 5**).

```
# Step 6 - Save tasks to the ToDoToDoList.txt file
elif strChoice.strip() == '4':
    objFile = open(strFileName, 'w')
    for dicRow in lstTable:
        objFile.write(dicRow["task"] + ',' + dicRow["priority"] + '\n')
    objFile.close()
    print("File updated!\n")
    continue
```

Figure 5. Write a dictionary row to the file, by getting values from their keys.

As soon as the last object from the table will be written to the file, the user will be notified that "File updated" and the user returns to the main menu.

Step 5. Exit program

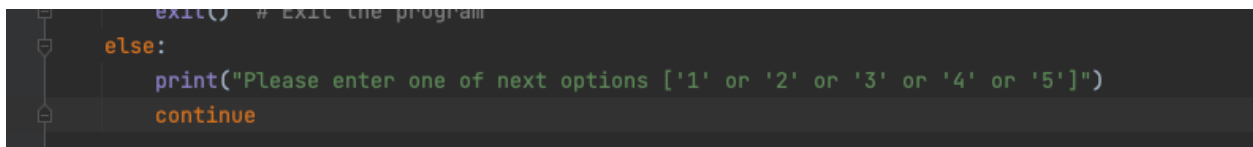
If the user chooses option '5' - the user will exit the program, I also added a prompt message to inform him that the program is finished and will be closed (**Figure 6**).

```
# Step 7 - Exit program
elif strChoice.strip() == '5':
    print("See you next time \nProgram finished\n")
    exit() # Exit the program
```

Figure 6. Exit the program block.

Step 6. Handle the user's mistake

In this assignment, I do not catch any error from the user's input. I assumed the user can enter any text for a task and any text for a priority, so there is no need to do anything there. Everything the user enters as a task or as a task priority can be added to the file. However, to prevent the program from crashing as a last step I added an 'else' block where the user is notified of what options he can enter when the user is in the main menu (Figure 7).



```
exit() # Exit the program
else:
    print("Please enter one of next options ['1' or '2' or '3' or '4' or '5']")
    continue
```

Figure 7. A prompt message to ask the user to choose one of the valid options.

After completing the script I run the program from the PyCharm and Terminal to be sure it works expectedly.

Summary

While completing the assignment I learned about: the difference between a List and a Dictionary, what is difference between index and a key, and how to read data from and write data to the file.