Tony Le

11/05/2019

IT FDN 100

Assignment 05

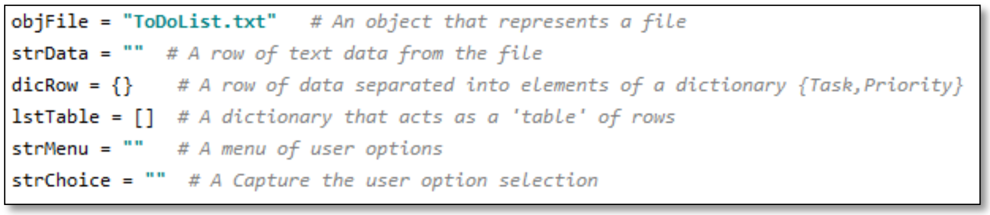
List and Dictionaries

Introduction

The assignment for this week is to create a program that displays a menu to the user that allows them to create a To-Do list. This user has options to view the data, save the data, add an item, remove an item or exit the program. We were given a starter file and we add to it to accomplish this task.

Declare Variables and Constants

The script in Figure 1 begins by declaring the variables and constants to be used at a later point.

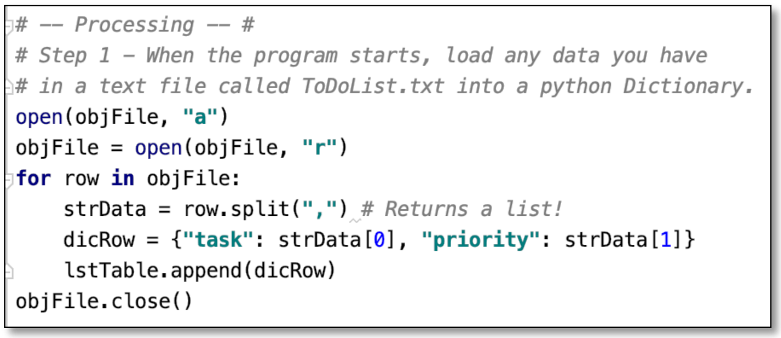


```
objFile = "ToDoList.txt"   # An object that represents a file
strData = ""  # A row of text data from the file
dicRow = {}    # A row of data separated into elements of a dictionary {Task,Priority}
lstTable = []  # A dictionary that acts as a 'table' of rows
strMenu = ""   # A menu of user options
strChoice = ""  # A Capture the user option selection
```

Figure 1: Declaring variables

Loading data from previous text file

Figure 2 shows the script that reads the text file **ToDoList.txt** and reads the data within the file and loads it into a python dictionary **dicRow**. We begin by opening the file and reading the data into the memory. We follow by iterating through each entry in the txt file and extract with [0] and [1] entries to be inserted into a dictionary with keys **task** and **priority.** This dictionary row is then appended onto our list table.

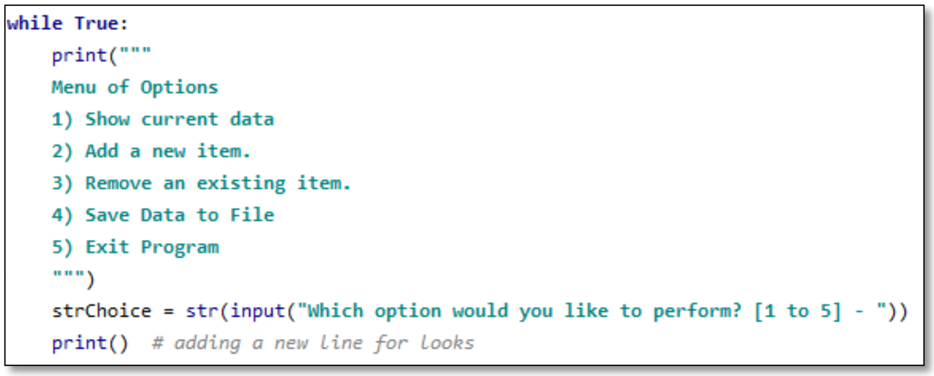


```
# -- Processing -- #
# Step 1 - When the program starts, load any data you have
# in a text file called ToDoList.txt into a python Dictionary.
open(objFile, "a")
objFile = open(objFile, "r")
for row in objFile:
    strData = row.split(",") # Returns a list!
    dicRow = {"task": strData[0], "priority": strData[1]}
    lstTable.append(dicRow)
objFile.close()
```

Figure 2: Declaring variables

Creating the menu

The next step in the script is to create the menu that allows to use to control what the program does as shown in Figure 3. This menu utilizes a "**while true**" command in order to continually run until the user decides to exit the script on their own accord. An input is requested from the user to determine which action they'd like to perform.

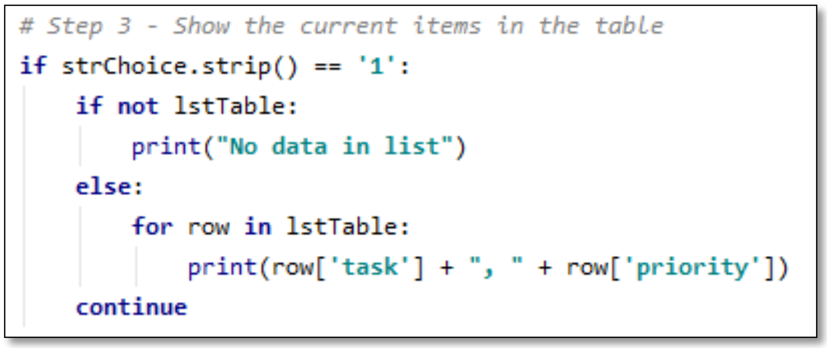


```
while True:
    print("""
    Menu of Options
    1) Show current data
    2) Add a new item.
    3) Remove an existing item.
    4) Save Data to File
    5) Exit Program
    """)
    strChoice = str(input("Which option would you like to perform? [1 to 5] - "))
    print()  # adding a new line for looks
```

Figure 3: Menu script

Displaying current data

The first option in the menu when the user enters "1" is to display the current items in the table **(lstTable)** created. I create an **if statement** in the case where there is no data in the table at the time. If there is no data, it will return a "No data in list" notification. An **else command** is placed in so that if there is data that exists in the table, then it will iterate for each row that exists and displays the "task" and the "priority" level. The task and priority are defined in the following section where the user inputs them into a dictionary. Figure 4 displays the code for this section.

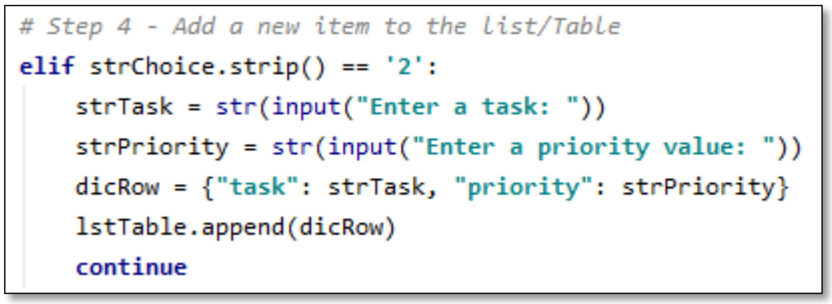


```
# Step 3 - Show the current items in the table
if strChoice.strip() == '1':
    if not lstTable:
        print("No data in list")
    else:
        for row in lstTable:
            print(row['task'] + ", " + row['priority'])
    continue
```

Figure 4: Option 1 showing current data

Adding new tasks

Figure 5 shows the script to add a new item to the table. If the user enters in option 2 to the menu input, then they will be asked to add a task and priority to the list. These values are inserted into a dictionary with keys **"task** and **"priority"** respectively. The dictionary created is then appended as a row onto the list table with **lstTable.append()**.

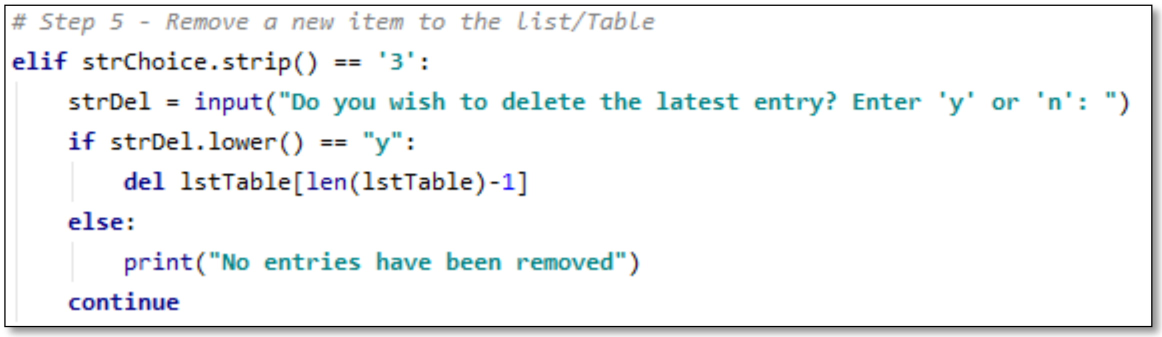


```
# Step 4 - Add a new item to the list/Table
elif strChoice.strip() == '2':
    strTask = str(input("Enter a task: "))
    strPriority = str(input("Enter a priority value: "))
    dicRow = {"task": strTask, "priority": strPriority}
    lstTable.append(dicRow)
    continue
```

Figure 5: Add new task to table

Removing new tasks

The next section of the script is to provide a method for the user to remove a "new item" that they've just added to the script as shown in Figure 6. This script executes if the user enters 3 in the menu. Then are they prompted to either enter in 'y' or 'n' if they would like to delete the latest entry in the to do list. If 'y' is entered, then the last row of **lstTable** is deleted using the **del** command. The index that is removed is determined by the **len()** command minus one. This removes the latest entry in the list.

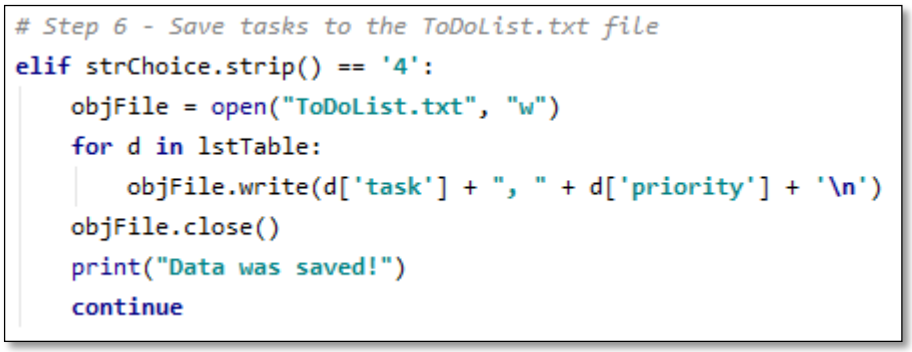


```
# Step 5 - Remove a new item to the list/Table
elif strChoice.strip() == '3':
    strDel = input("Do you wish to delete the latest entry? Enter 'y' or 'n': ")
    if strDel.lower() == "y":
        del lstTable[len(lstTable)-1]
    else:
        print("No entries have been removed")
    continue
```

Figure 6: Removing a new task from table

Saving tasks entered

If the user would like to save the entries they've already provided, they can use option 4 as shown in Figure 7. This option opens up the ToDoList.txt file and will write to it. The write function is done with a for loop where it will run for every iteration in the table and calls upon the **task** and **priority** keys. The file is then closed and saved.

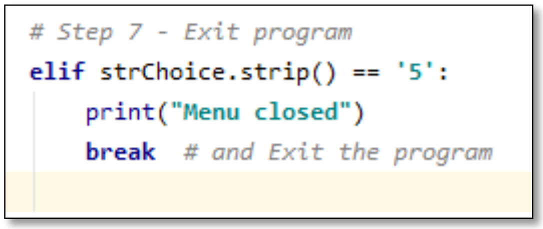


```
# Step 6 - Save tasks to the ToDoList.txt file
elif strChoice.strip() == '4':
    objFile = open("ToDoList.txt", "w")
    for d in lstTable:
        objFile.write(d['task'] + ", " + d['priority'] + '\n')
    objFile.close()
    print("Data was saved!")
    continue
```

Figure 7: Saving new tasks from table

Exit program

The exiting of the program through input 5 is very simple as shown in Figure 8. It simply prints out a notification that the program is closed and a **break** command completes the **while true** loop at the beginning of this code.

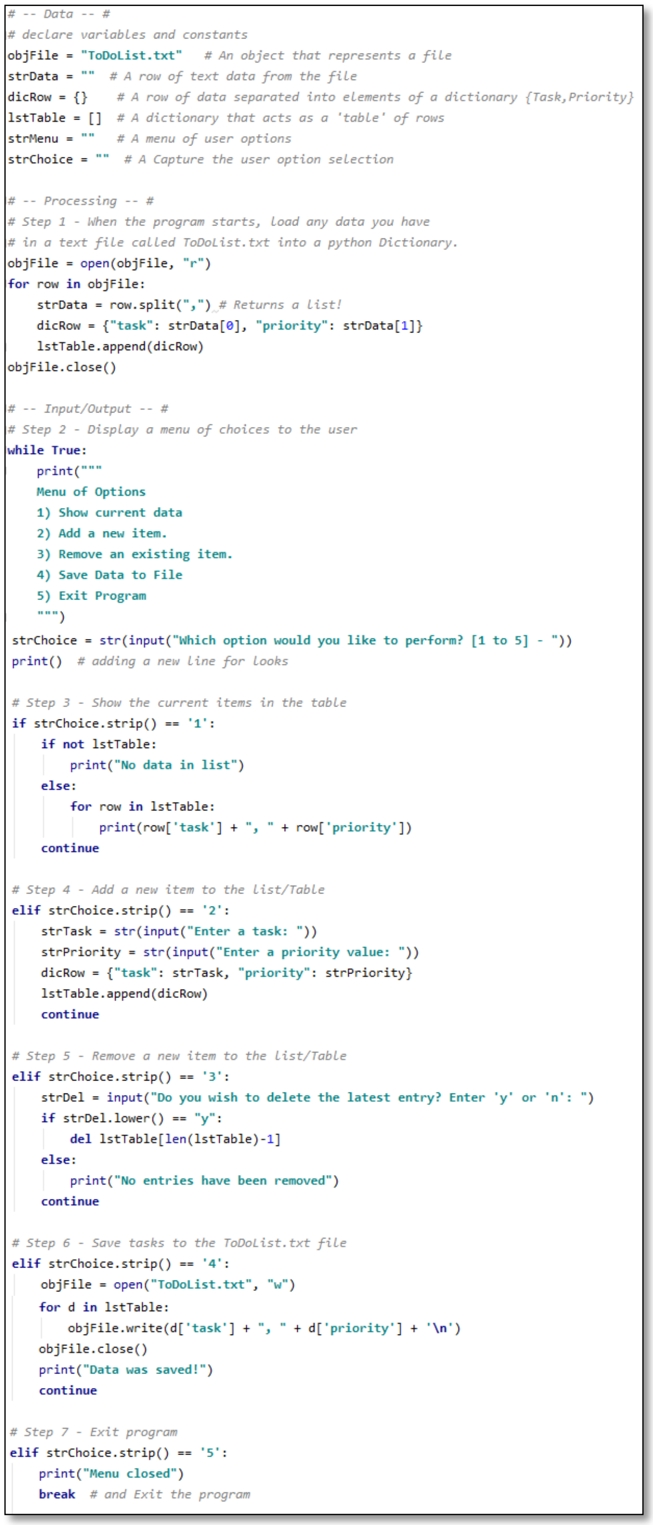


```
# Step 7 - Exit program
elif strChoice.strip() == '5':
    print("Menu closed")
    break  # and Exit the program
```

Figure 8: Exit program

Summary

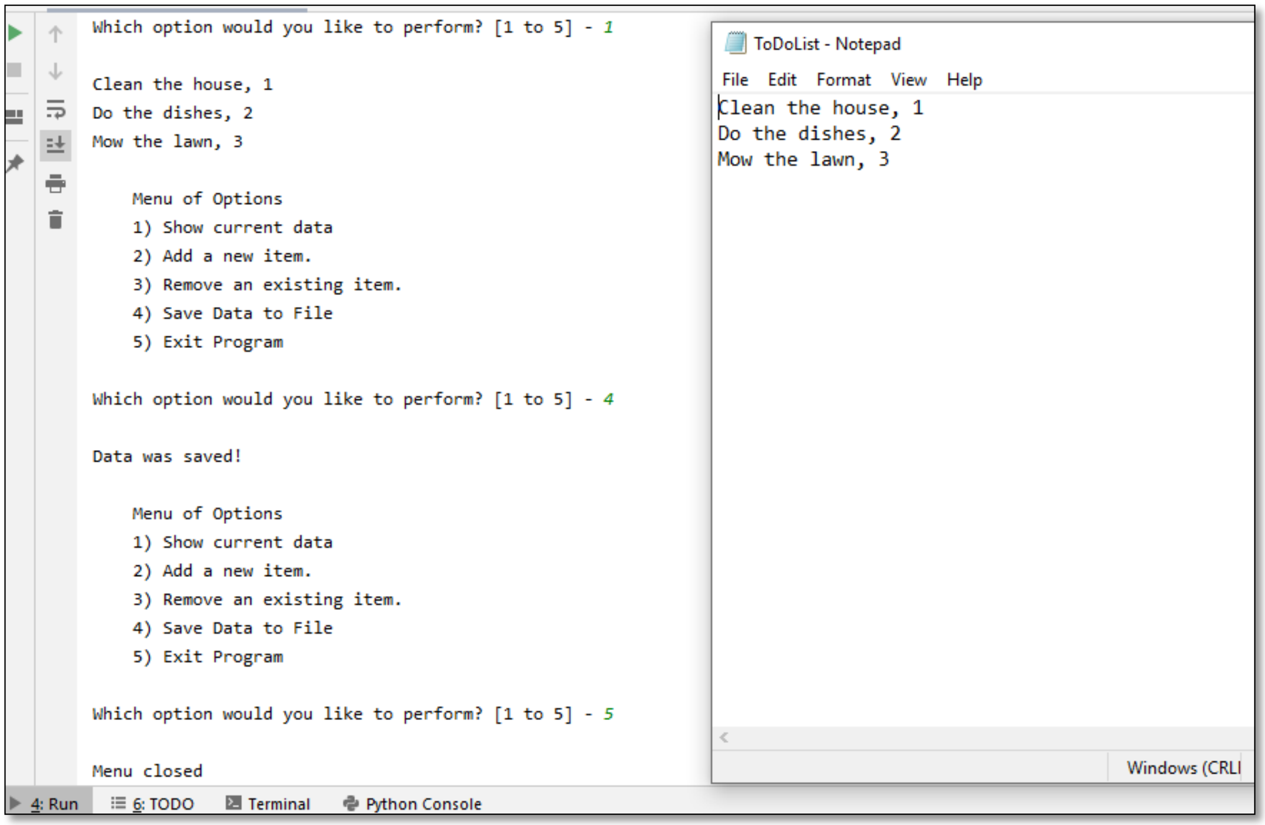
Figure 9 shows the completed script.



```
# -- 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 dictionary 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 Dictionary.
objFile = open(objFile, "r")
for row in objFile:
    strData = row.split(",") # Returns a list!
    dicRow = {"task": strData[0], "priority": strData[1]}
    lstTable.append(dicRow)
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':
    if not lstTable:
        print("No data in list")
    else:
        for row in lstTable:
            print(row['task'] + ", " + row['priority'])
    continue

# Step 4 - Add a new item to the list/Table
elif strChoice.strip() == '2':
    strTask = str(input("Enter a task: "))
    strPriority = str(input("Enter a priority value: "))
    dicRow = {"task": strTask, "priority": strPriority}
    lstTable.append(dicRow)
    continue

# Step 5 - Remove a new item to the list/Table
elif strChoice.strip() == '3':
    strDel = input("Do you wish to delete the latest entry? Enter 'y' or 'n': ")
    if strDel.lower() == "y":
        del lstTable[len(lstTable)-1]
    else:
        print("No entries have been removed")
    continue

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

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

Figure 9: Completed script

The intent of this script was to create an interactive menu and allow the user to create a a to-do list of task and their respective priority values. This assignment utilizes a lot of new skills we've learned in Module 5. We got a sneak peek of lists in the past module and learned to utilize them with dictionaries in this one. We also gained more experience in writing to text files through full control of user input. The output is shown below in Figure 10 that shows the results of the data saved in a ToDoList.txt file created as a result of this script.



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

Clean the house, 1
Do the dishes, 2
Mow the lawn, 3

    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 was 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

Menu closed
```

ToDoList - Notepad

```
Clean the house, 1
Do the dishes, 2
Mow the lawn, 3
```

Figure 10: Print out of full name entered by the user