

Steven Simpers

May 17, 2023

Foundations of Programming: Python

Assignment 05

<https://github.com/ssimpers/IntroToProg-Python.git> (External Site)

# To-Do List Script

## Introduction

This week's lesson focused on lists and dictionaries; comparing how data is accessed from and assigned to them. We learned about general programming concepts such as separations of concerns, error handling, script templates, and source control.

## Problem Statement

The assignment problem statement is to update the provided starter code to load task and priority data from a file and allow the user to show, add to, remove from, and save the data back to the file. The starter code contains sections for variable and constant data declarations, pseudo code for processing, and pseudo code for input/output operations.

## Data

The first section that was updated from the starter script is the data section, seen in Listing 1. A list variable "lstRow" was added to allow for storing one line of data at a time. A few string variables "strInput1", "strInput2", and "strRemove" were created to capture input from the user. Finally, an integer variable "intBreak" was included to indicate within a nested loop that the outer loop needs to be broken out of.

*Listing 1: "Assignment05\_Updated.py" data*

```
12 # -- Data -- #
13 # declare variables and constants
14 strFile = "ToDoList.txt" # A string that contains the file name
15 objFile = None # An object that represents a file
16 strData = "" # A row of text data from the file
17 dicRow = {} # A row of data separated into elements of a dictionary {Task,Priority}
18 lstRow = [] # A list representing a row of data
19 lstTable = [] # A list that acts as a 'table' of rows
20 strMenu = "" # A menu of user options
21 strChoice = "" # A capture of the user option selection
22 strInput1 = "" # A capture of the user task name to add
23 strInput2 = "" # A capture of the user task priority to add
24 strRemove = "" # A capture of the user task to remove
25 intBreak = 0 # An integer that acts as a flag within a nested loop to initiate a break from an outer loop
```

## Processing

The processing section of the code in Listing 2 opens a file in read mode and loads the data into a list. This process occurs by looping through each row of "objFile", an object that represents the file,

separating the string where commas appear, and storing the row data as a list. The list is then stored in a dictionary by defining keys for each comma separated value and stripping the carriage return off of the last value in each row. Each dictionary row is then added to a list table using the append list method and the file is closed.

**Listing 2: "Assignment05\_Updated.py" processing**

```
27 # -- Processing -- #
28 # Step 1 - When the program starts, load any data you have
29 # in a text file called ToDoList.txt into a python list of dictionaries rows (like Lab 5-2)
30 objFile = open(strFile, "r") # open file in read mode
31 for row in objFile:
32     lstRow = row.split(",") # row list
33     dicRow = {"Task": lstRow[0], "Priority": lstRow[1].strip()} # row dictionary (strip \n off)
34     lstTable.append(dicRow)
35 objFile.close()
```

## Input/Output – Show Current Items

The input and output section of code contains several subsections that perform specific actions within a while loop. The first subsection was already present in the starter code and it outputs a menu of options to the user. The next subsection displays the current data to the user, shown in Listing 3. A header is printed to the command prompt to aid the user in interpreting the information. The script prints each row of the list table by providing the key corresponding to the desired dictionary entry. The program then reaches the "continue" statement and code execution resumes at the beginning of the input/output while loop that was present in the starter script.

**Listing 3: "Assignment05\_Updated.py" input/output – show current items**

```
51 # Step 3 - Show the current items in the table
52 if (strChoice.strip() == "1"):
53     print(" " + "="*20) # header line
54     print(" Task" + "," + "Priority") # header text
55     print(" " + "="*20) # header line
56     for dicRow in lstTable:
57         print(" " + dicRow["Task"] + "," + dicRow["Priority"])
58     continue
```

## Input/Output – Add New Item

Presented in Listing 4 is the next subsection of the input and output section where new entries are added to the list. The user is asked, using input functions, to enter a task name and priority which are then stored separately as strings. The task and priority strings are then assigned to a dictionary along with the keys, then appended to the list table containing the original task entries. The script then continues at the beginning of the input/output while loop.

**Listing 4: "Assignment05\_Updated.py" input/output – add new item**

```
60 # Step 4 - Add a new item to the list/Table
61 elif (strChoice.strip() == "2"):
62     print(" Enter a task name and its priority...")
63     strInput1 = input(" Enter task name: ")
64     strInput2 = input(" Enter priority: ")
65     dicRow = {"Task": strInput1, "Priority": strInput2}
66     lstTable.append(dicRow)
67     continue
```

## Input/Output – Remove Item

Another menu option allows the user to remove an item from the current list, see Listing 5. The input function prompts the user to enter a task to remove or type 'c' to cancel. A while loop is used to allow statements to be repeated and then broken out of when initiated by break statements. An if statement on line 73 checks if the user entered 'c' and breaks out of the while loop if the expression is true.

**Listing 5: "Assignment05\_Updated.py" input/output – remove item**

```
69 # Step 5 - Remove an item from the list/Table
70 elif (strChoice.strip() == "3"):
71     strRemove = input(" Enter a task name that you would like to remove or type 'c' to cancel: ")
72     while (True):
73         if strRemove.lower() == "c":
74             break
75         intBreak = 0 # reset flag
76         for dicRow in lstTable:
77             if strRemove.lower() == dicRow["Task"].lower():
78                 lstTable.remove(dicRow)
79                 intBreak = 1 # flag to break out of outer 'while' loop
80         if intBreak == 1:
81             break
82         strRemove = input(" Task not found, enter a task name to remove or type 'c' to cancel: ")
83     continue
```

A variable "intBreak" is set equal to zero and then a for loop is used to loop through each dictionary row in the list table. Each dictionary row is tested in an if statement's expression to determine if the user's input matches any task value. If true, the dictionary row is removed from the list table and the variable "intBreak" is set equal to one. When "intBreak" equals one, the program breaks out of the while loop on line 81. The "intBreak" variable was created to indicate within a nested loop that the outer loop needs to be broken out of. If the break statement were used in the nested for loop, it would only break the program out of the nested for loop on line 76 instead of the desired action to break out of the outer while loop on line 72. If the program hasn't broken out of the while loop by line 82, the user is notified that their task was not found, and they can re-enter a task or type 'c' to cancel. After the user enters an accepted task name or cancels the action, the program continues at the beginning of the outer-most input/output while loop.

## Input/Output – Save to File

When the user enters "4", the program saves the current data to a file as seen in Listing 6. The file is opened in write mode and each dictionary row in the list table is written to the file by specifying the key representing the desired values. The file is closed and the user is notified, using a print function, that the data was saved. The program continues at the start of the input/output while loop.

**Listing 6: "Assignment05\_Updated.py" input/output – save to file**

```
85 # Step 6 - Save tasks to the ToDoList.txt file
86 elif (strChoice.strip() == "4"):
87     objFile = open(strFile, "w") # open file in write mode
88     for dicRow in lstTable:
89         objFile.write(dicRow["Task"] + "," + dicRow["Priority"] + "\n")
90     objFile.close()
91     print(" Data saved to file!")
92     continue
```

## Input/Output – Exit Program

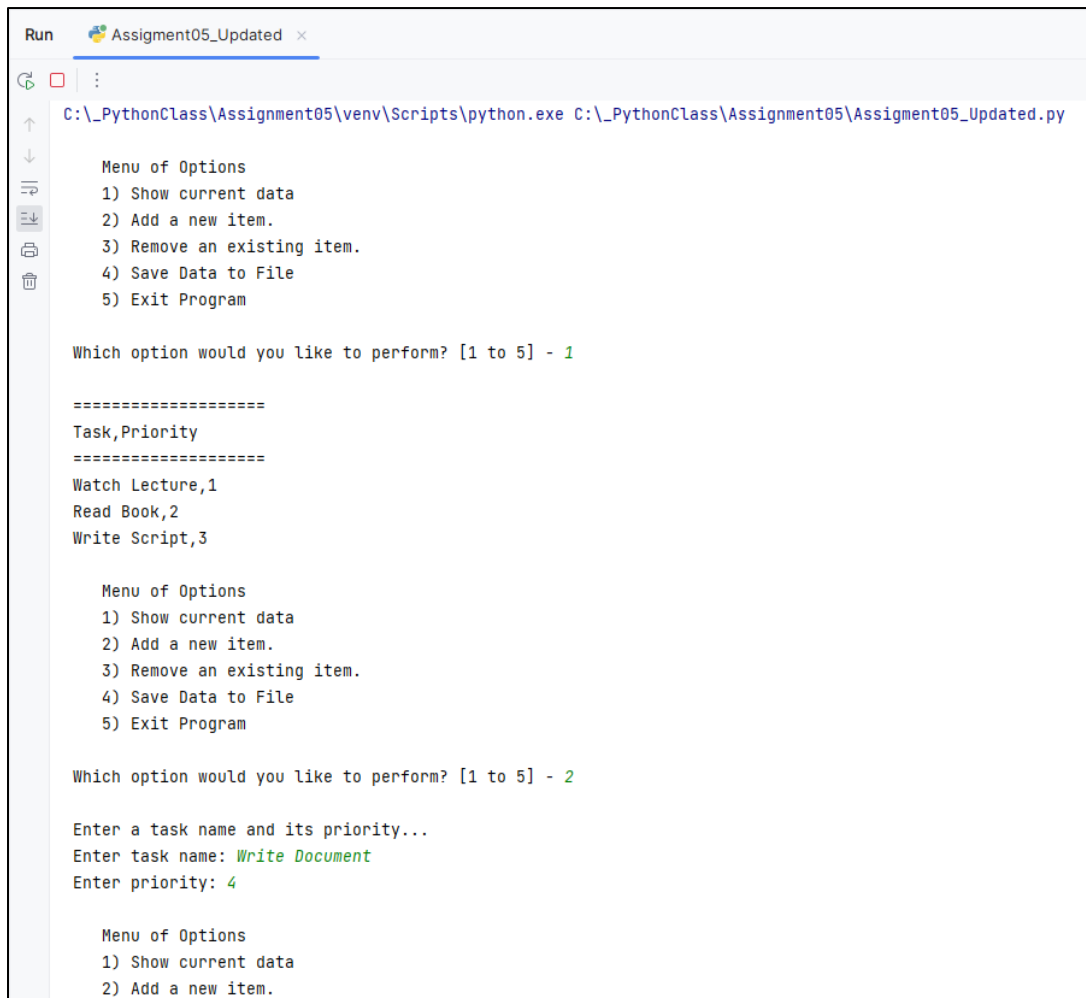
The final menu option in Listing 7 is to exit the program when the user enters “5”. A print function is used to state in the command line that the program is exiting. However, this code won’t be seen if the script is ran in the command prompt because the window will close before it can be read.

**Listing 7: “Assignment05\_Updated.py” input/output – exit program**

```
94 # Step 7 - Exit program
95 elif (strChoice.strip() == "5"):
96     print(" Exiting...")
97     break # and Exit the program
```

## Results

An example of the assignment script running in PyCharm is provided below in Figure 1. The user first chooses option 1 and the program returns the current data. The user then requests option 2 and is asked to enter a task name and priority to add to the list.



```
Run Assignment05_Updated x
C:\_PythonClass\Assignment05\venv\Scripts\python.exe C:\_PythonClass\Assignment05\Assignment05_Updated.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] - 1

=====
Task,Priority
=====
Watch Lecture,1
Read Book,2
Write Script,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

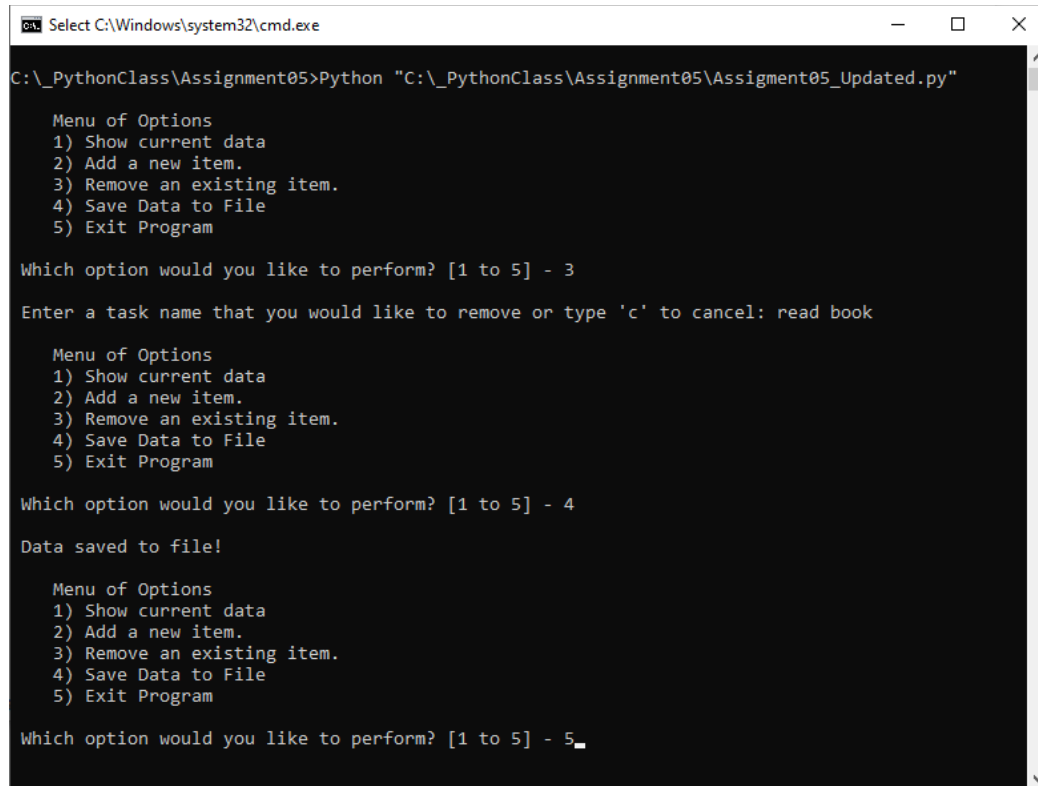
Which option would you like to perform? [1 to 5] - 2

Enter a task name and its priority...
Enter task name: Write Document
Enter priority: 4

Menu of Options
1) Show current data
2) Add a new item.
```

**Figure 1: “Assignment05\_Updated.py” Python script running in PyCharm**

Figure 2 below shows the script running in the command window. Menu option 3 is selected by the user and they are asked to enter a task name to remove from the list. The next option 4 is chosen by the user to save the file and they receive confirmation that the data was saved. Finally, option 5 is requested to exit the program.



```
Select C:\Windows\system32\cmd.exe

C:\_PythonClass\Assignment05>Python "C:\_PythonClass\Assignment05\Assignment05_Updated.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] - 3

Enter a task name that you would like to remove or type 'c' to cancel: read book

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

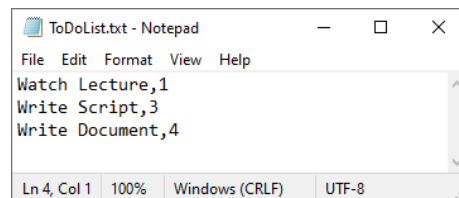
Data saved to file!

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_
```

**Figure 2: "Assignment05\_Updated.py" Python script running in Command Window**

The data file "ToDoList.txt" in Figure 3 shows the data stored as comma separated values.



```
ToDoList.txt - Notepad
File Edit Format View Help
Watch Lecture,1
Write Script,3
Write Document,4
Ln 4, Col 1 100% Windows (CRLF) UTF-8
```

**Figure 3: Data Saved in "ToDoList.txt" File**

## Summary

The assignment further developed skills working with lists and introduced dictionaries, another tool to store data. The class started using GitHub for source control and I'm looking forward to writing custom functions in the next assignment.