

Martin Stallworth

12 August 2020

Foundations of Programming, Python

Assignment 05

Working with Lists and Dictionaries and Improving Scripts

Introduction

During this week's assignment we built on what we touched on last week in regards to using lists to hold collections of data. This included saving data from a list into a file and loading data from a file into a list. We then covered another method of storing data which was the dictionary mapping type. We were able to take what we learned from using lists and adjust those processes to use dictionaries in their place. Lastly, we covered different ways that we can improve on and handle errors in our scripts.

Getting the Background

The first thing that we covered in this week's module was a continuation of working with lists. Particularly we dived deeper in how to unpack 2D lists and how to save the data from lists into a file as well as loading data from a created file into a list. This was similar to what we covered last week with the addition of loading data from a file into lists which was a concept that took me a while to grasp. Working through the assignment actually gave me a eureka moment in regards to this which I will discuss later on. We then used what we learned with lists to move on to the next topic which was dictionaries. Learning about tuples and lists beforehand really helped with the understanding of the manipulation of dictionaries but there were some key differences. Mainly, the fact that dictionaries involved both a key and a value within its structure so the inner structure is slightly more complex at face value. The 'Saravji's Hut Part 2' YouTube video as well as the textbook (Dawson 139-148) really helped with understanding the differences and advantages of using dictionaries. The key of the dictionary has to be immutable while the values can be both immutable and mutable which seems like something that could open up many versatile options in your programs.

Moving on to the next section of the module we discussed many ways in which we can improve on our scripts. One underrated part the I enjoyed learning was creating the template header because it was actually getting tedious to try to type out the header every week. The separation of concerns is something that I think will be very helpful in the future because it allows you to further organize your ideas break down your code in a way that makes sense for what you are trying to accomplish. What seems like the more important part of this section for the future seems to be the discussion of functions. This allows you to define a function beforehand and call it when you may actually need it which sounds like it could be helpful. The structured error handling also can be helpful for future code because it prevents the entire program from crashing when the user makes a mistake. I often get annoyed when I have to completely restart a program when I mistake different input values. Lastly, GitHub seems like a great tool for going back into previous parts of your code. I often like to start experimenting with my code when I get stuck on something, but when it goes awry it's great to know that I can get back to something like a checkpoint in my process.

Working through Labs and Assignment

This week's labs had a different feel than those from the previous weeks. The first thing I noticed was that there were less labs than usual so I assumed these week's labs would be more difficult. While I did run into some

issues working through the labs I did not feel they were as difficult as some of the labs from the past couple of weeks. I enjoyed the structure this week in lab A where we got a starter script and had instructions in the code of what we were trying to accomplish. This lab also had some similarities to last week's assignment so I was able to use that as a bridge to work on this lab. The only thing that I found to be different was the section where we had to read the file into the list. The 'Saravji's Hut Part 1' video and the module notes helped me figure out how to work through it to get the code to work but I still did not understand what I was actually doing. An example of this part of my code is provided below.

```
1. elif strChoice == 'r':
2.     # File to print
3.     # TODO read the file line by line into in-memory list.
4.     print('ID | CD Title | Artist Name')
5.     objFile = open('CDInventory.txt', 'r')
6.     for row in lstTbl:
7.         lstRow = row
8.         print(*lstRow, sep = ' | ')
9.     objFile.close()
10.
```

Listing 1: Ex. of reading file into a list

Over the course of the next lab we moved from using lists to dictionaries. For this lab we just had to switch our inner lists into dictionaries within the 2D list. This process went by mostly smoothly except I had some difficulty with the part where we had to read the data from the file. There were some extra steps that needed to be made when switching from lists to dictionaries but after some playing around I was able to figure out what changes needed to be made. Another key difference this week was that examples of the solutions of each lab were given at the end of the module which I found to be helpful because I was able to take some of the methods that I found to be more efficient than my original methods and applied them to the assignment.

I enjoyed working on this week's assignment because it was a continuation of the one from last week so everything was at least somewhat familiar. While working on this assignment I was actually able to get a better idea of what the read file function actually does. Initially I thought that this was just another way to print your data but after paying more attention to what was going on I noticed that data that I hadn't yet entered was in the inventory after I hit the load function. I then realized that data saved into the file was being loaded into the in memory inventory which explains the need to clear the outer list. Most of the other processes I was familiar with because of the previous labs so they did not take too much time to complete. However, deleting an entry was something that I was unfamiliar with so it took so creativity and research to try to figure out how to do this. I discovered through the 'note.nknk.me' website that '.pop' was a method used to remove data. Once I gained this knowledge I used an approach that made sense to me to delete an entry. My code for this function is provided below.

```
1. elif strChoice == 'd':
2.     # Add functionality of deleting an entry
3.     print('The entries in this inventory are numbered in rows starting from 0', '\n')
4.     print('Row # = id - 1', '\n')
5.     for row in lstTbl: # unpack 2d list and print
6.         print(*row.values(), sep = ', ')
7.     delUserInput = int(input('Enter row # of the entry you would like to remove:')) # allow user
    to choose row # they want to delete
8.     print()
9.     print('You have chosen to delete row:', delUserInput) # show user which option they chose
10.    lstTbl.pop(delUserInput) # remove data with row # the user entered
11.    print()
12.    print('Your inventory is now:', '\n')
13.    for row in lstTbl: # reprint data with the entry now removed
14.        print(*row.values(), sep = ', ')
15.    print()
```

16. `print('If you deleted this option by mistake reload your data with option "l."')`

Listing 2: Ex of deleting an entry from 2D list

Once I got through this process I worked on the presentation of my program to communicate with the user. I then performed one last test run of the code in Spyder and a terminal to make sure everything worked as I intended. Lastly, I uploaded the script to my GitHub profile. Images of my program loading data in Spyder and the terminal window are provided below.

```
(base) Martins-Air: ~ marcl_0000 python _FUNPROGRAMMING/Assignments/CDInventory.py
[The Magic CD Inventory

[l] load Inventory from file
[a] Add CD
[i] Display Current Inventory
[d] delete CD from Inventory
[s] Save Inventory to file
[x] exit
l, a, i, d, s or x: l

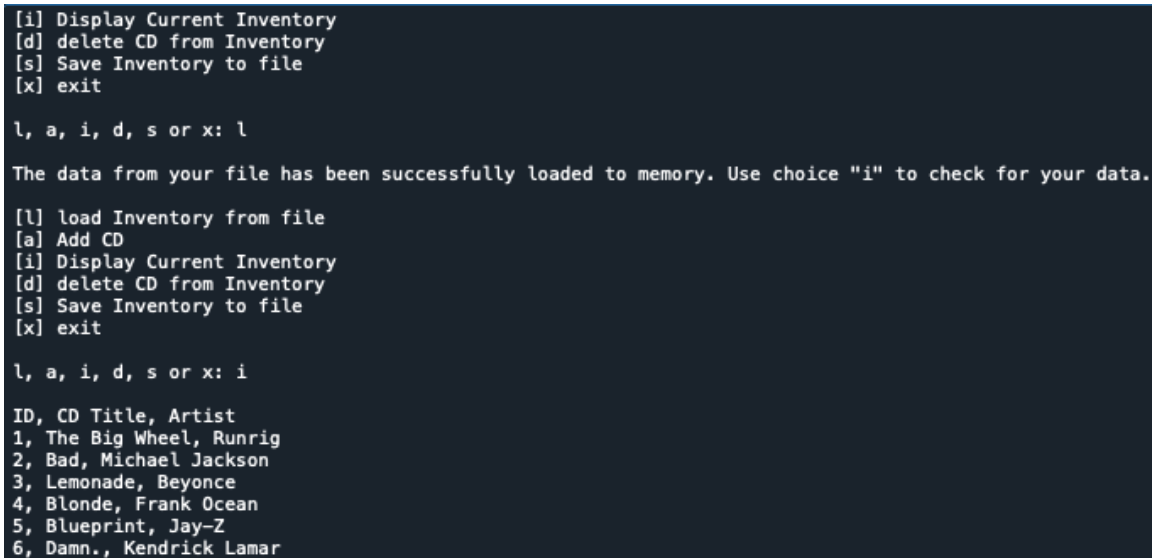
The data from your file has been successfully loaded to memory. Use choice "i" to check for your data.

[l] load Inventory from file
[a] Add CD
[i] Display Current Inventory
[d] delete CD from Inventory
[s] Save Inventory to file
[x] exit
l, a, i, d, s or x: i

ID, CD Title, Artist
1, The Big Wheel, Runrig
2, Bad, Michael Jackson
3, Lemonade, Beyonce
4, Damn., Kendrick Lamar
5, Blonde, Frank Ocean

[l] load Inventory from file
[a] Add CD
[i] Display Current Inventory
[d] delete CD from Inventory
[s] Save Inventory to file
[x] exit
l, a, i, d, s or x: █
```

Figure 1: Image of loading data in terminal



```
[i] Display Current Inventory
[d] delete CD from Inventory
[s] Save Inventory to file
[x] exit

l, a, i, d, s or x: l

The data from your file has been successfully loaded to memory. Use choice "i" to check for your data.

[l] load Inventory from file
[a] Add CD
[i] Display Current Inventory
[d] delete CD from Inventory
[s] Save Inventory to file
[x] exit

l, a, i, d, s or x: i

ID, CD Title, Artist
1, The Big Wheel, Runrig
2, Bad, Michael Jackson
3, Lemonade, Beyonce
4, Blonde, Frank Ocean
5, Blueprint, Jay-Z
6, Damn., Kendrick Lamar
```

Figure 2: Image of loading data in Spyder '

Summary

In this assignment, we worked with both lists and dictionaries to hold data in a 2D list. While we covered lists last week as well the introduction of dictionaries was a new concept and yet another way to hold data which

gives many options for future codes. We used both to save data into a file and load data from a file into a list as well as many other functions. We did not use functions in these week's assignment but we did have the opportunity to use structured error handling and upload our files into GitHub. I am excited to learn about how to utilize these processes for improving scripts in the future.

Appendix

Using [planetb's](#) webpage (external reference). Retrieved 2020-Aug-12

Using [Saravji's Hut Youtube](#) (external reference). Retrieved 2020-Aug-10

Using [Saravji's Hut Youtube](#) (external reference). Retrieved 2020-Aug-10

Using [note.nkmk.me's](#) webpage (external reference). Retrieved 2020-Aug-11

Dawson, Michael. Python Programming for the Absolute Beginner: 3rd Edition. Kindle ed., Cengage Learning, 2010

Link for GitHub: https://github.com/mstallworth513/Assignment_05

Listing for LAB05_A.py

```
11. #-----#
12. # Title: Lab05_A_starter.py
13. # Desc: Lab05-A starter script
14. # Change Log: (MStallworth, 2020-Aug-
    10, Added code for adding data, writing data to a file, reading data from the file into memory, and d
    isplaying the data as the user chooses.)
15. # DBiesinger, 2030-Jan-01, Created File. MStallworth, 2020-Aug-10, Appened File
16. #-----#
17.
18. # Declare variabls
19.
20. strChoice = '' # User input
21. # list of lists to hold data
22. lstTbl = [
23.     [1, 'The Big Wheel', 'Runrig'],
24.     [2, 'Bad', 'Michael Jackson']]
25. lstRow = [] # list of data row
26. strFileName = 'CDInventory.txt' # data storage file
27. objFile = None # file object
28.
29. # Get user Input
30. print('Write or Read file data.')
31. while True:
32.     print('\n[a] add data to list\n[w] to write data to file\n[r] to read data from file')
33.     print('[d] display data\n[exit] to quit')
34.     strChoice = input('a, w, r, d, or exit: ').lower() # convert choice to lower case at time of inp
    ut
35.     print('\n\n')
36.
37.     if strChoice == 'exit':
38.         break
39.     if strChoice == 'a': # no elif necessary, as this code is only reached if strChoice is not 'exit
    '
40.         # Add data to list in memory
41.         # TODO ask user to input data and store it in the in-memory list
42.         ID = int(input('Enter an ID:'))
43.         CD = input('Enter a CD Title:')
44.         Artist = input('Enter an artist name:')
45.         new_cd = [ID, CD, Artist]
46.         lstTbl.append(new_cd)
47.
48.     elif strChoice == 'w':
49.         # List to File
50.         # TODO add code here to write from in-memory list to file
51.         objFile = open('CDInventory.txt', 'w')
52.         for row in lstTbl:
53.             objFile.write('{} , {} , {}\n'.format(row[0], row[1], row[2]))
54.         objFile.close()
```

```

55.
56.     elif strChoice == 'r':
57.         # File to print
58.         # TODO read the file line by line into in-memory list.
59.         print('ID | CD Title | Artist Name')
60.         objFile = open('CDInventory.txt', 'r')
61.         for row in lstTbl:
62.             lstRow = row
63.             print(*lstRow, sep = ' | ')
64.         objFile.close()
65.
66.     elif strChoice == 'd':
67.         # Display data
68.         # TODO display the data to the user.
69.         print('ID | CD Title | Artist Name')
70.         for row in lstTbl:
71.             print(*row, sep= ' | ')
72.         print()
73.
74.     else:
75.         print('Please choose either a, w, r or exit!')

```

Listing for CDInventory.py

```

17. #-----#
18. # Title: CDInventory.py
19. # Desc: Starter Script for Assignment 05
20. # Change Log: MStallworth, 2020-Aug-
10, Added script loading inventory from file, adding data, displaying data, deleting data, and saving
inventory
21. # DBiesinger, 2030-Jan-01, Created File. MStallworth, 2020-Aug-10, Appended file
22. #-----#
23.
24. # Declare variabls
25.
26. strChoice = '' # User input
27. lstTbl = [] # list of lists to hold data
28. lstRow = []
29. # replace list of lists with list of dicts
30. dicRow = {} # list of data row
31. dicRow1 = {'id': 1, 'cd': 'The Big Wheel', 'artist': 'Runrig'}
32. dicRow2 = {'id': 2, 'cd': 'Bad', 'artist': 'Michael Jackson'}
33. lstTbl.append(dicRow1)
34. lstTbl.append(dicRow2)
35. strFileName = 'CDInventory.txt' # data storage file
36. objFile = None # file object
37.
38. # Get user Input
39. print('The Magic CD Inventory\n')
40. while True:
41.     # Display menu allowing the user to choose:
42.     print('[l] load Inventory from file\n[a] Add CD\n[i] Display Current Inventory')
43.     print('[d] delete CD from Inventory\n[s] Save Inventory to file\n[x] exit')
44.     strChoice = input('l, a, i, d, s or x: ').lower() # convert choice to lower case at time of input
45.     print()
46.
47.     if strChoice == 'x':
48.         # Exit the program if the user chooses so
49.         print('Goodbye.')
50.         break
51.     if strChoice == 'l':
52.         # Add the functionality of loading existing data
53.         lstTbl.clear() # clear the data from the memory
54.         objFile = open(strFileName, 'r') # open the saved file
55.         for row in objFile: # go through every line in file
56.             lstRow = row.strip().split(',') # ignore blank lines and separate data with comma

```

```

57.         dicRow = {'id': lstRow[0], 'cd': lstRow[1], 'artist': lstRow[2]} # Place all data from fi
    le into a dictionary
58.         lstTbl.append(dicRow) # place data from file into the cleared 2d list
59.         objFile.close()
60.         print('The data from your file has been successfully loaded to memory. Use choice "i" to chec
    k for your data.', '\n')
61.
62.     elif strChoice == 'a': # no elif necessary, as this code is only reached if strChoice is not 'ex
    it'
63.         # Add data to the table (2d-list) each time the user wants to add data
64.         strID = input('Enter an ID: ') # Create variable names for all parts of data
65.         strTitle = input('Enter the CD\'s Title: ')
66.         strArtist = input('Enter the Artist\'s Name: ')
67.         intID = int(strID) # convert ID into an integer
68.         dicRow = {'id': intID, 'cd': strTitle, 'artist': strArtist} # format how user data is added i
    nto 2d list
69.         lstTbl.append(dicRow) # add user data into in memory user 2d list
70.         print()
71.     elif strChoice == 'i':
72.         # Display the current data to the user each time the user wants to display the data
73.         print('ID, CD Title, Artist')
74.         for row in lstTbl: # unpack the 2d list
75.             print(*row.values(), sep = ', ')
76.         print()
77.
78.     elif strChoice == 'd':
79.         # Add functionality of deleting an entry
80.         print('The entries in this inventory are numbered in rows starting from 0', '\n')
81.         print('Row # = id - 1', '\n')
82.         for row in lstTbl: # unpack 2d list and print
83.             print(*row.values(), sep = ', ')
84.         delUserInput = int(input('Enter row # of the entry you would like to remove:')) # allow user
    to choose row # they want to delete
85.         print()
86.         print('You have chosen to delete row:', delUserInput) # show user which option they chose
87.         lstTbl.pop(delUserInput) # remove data with row # the user entered
88.         print()
89.         print('Your inventory is now:', '\n')
90.         for row in lstTbl: # reprint data with the entry now removed
91.             print(*row.values(), sep = ', ')
92.         print()
93.         print('If you deleted this option by mistake reload your data with option "l."', '\n')
94.
95.
96.     elif strChoice == 's':
97.         # Save the data to a text file CDInventory.txt if the user chooses so
98.         objFile = open(strFileName, 'w') # open file and overwrite data
99.         for row in lstTbl:
100.             strRow = ''
101.             for item in row.values(): # unpack dictionary values
102.                 strRow += str(item) + ',' # store data into variable
103.             strRow = strRow[:-1] + '\n'
104.             objFile.write(strRow) # write stored values into file
105.         objFile.close()
106.         print()
107.     else:
108.         print('Please choose either l, a, i, d, s or x!')

```