Evan Malina

11/21/2021

Foundations of Programming Python

Assignment 6

# Introduction

For this assignment, I modified an existing script to change the use of a list for storing individual data entries to a dictionary. Additionally, I added code to the existing script to load the data from the inventory, and to delete an entry. Finally, I unloaded the code to my Assignment\_05 repository on gitlab. This assignment was mainly focused on processing dictionaries. That is, inputing data into a dictionary, writing a dictionary to .txt and inputing the file back into a dictionary.

# Code

## Github

Github link to my code and knowledge document here;

<https://github.com/ewmalina/Assignment_06>

## Modifying the lists to dictionaries

I started by changing the empty list to a dictionary, simply by swapping [] for {}. The ‘a’ section was fairly straightforward as well, simply by changing the existing list to a dictionary with {}, and adding the ‘key’ strings in this example (or rather, any immutable or hash object).

The ‘i’ section caught me up for a bit. I have some more learning to do regarding the use of \*row in a for row in …. Code. I ended up simply swapping \*row into \*row.values(). This worked great and makes sense but I’m still a little stuck on the idea of \*row.

The ‘s’ section was very similar to the ‘I’. I really only swapped the for loop ‘row’ to ‘row.values()’

## Loading existing Data

I debated this section quite a lot as well – mainly whether to clear the existing variable lstTbl prior to importing the data. I ended up NOT clearing lstTbl, but instead modified the writing to .txt to ‘w’ (write) instead of ‘append’. This was to solve a dilemma because when I actually ran the code, I would add a new CD Entry and then import the existing data and finally write the data to .txt. This was creating duplicates using the append, and I didn’t have a great solution. Ideally, if import happens, ‘w’ else ‘a’. But the history if whether import happens is tricky. Now that I’m verbalizing this, if the import section of code I could change some Boolean variable from false to true.

In this section of code, I mostly *plagerized* from the Module script. I used a for loop to run through each line of .txt code, and input the *stripped* and *separated* (by comma ‘,’) values into a table. The 3 indexes of the table were then dumped into a new dictionary, and appended to the master table.

## Delete an entry

I thought about this section for a while, and would love to delete a specific entry. I ended up deciding to simply delete the LAST entry. This isn’t totally true, because if an entry was added and then the existing data was loaded, my code would take and delete the final loaded data entry. Nonetheless, I simply deleted the final table item with the del command of a list using len(list)-1.

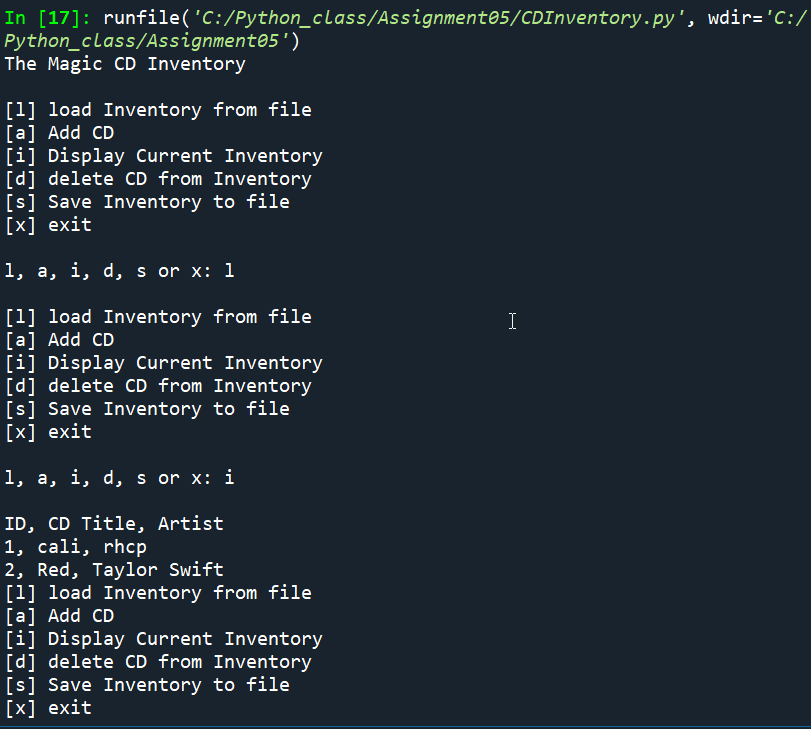


Figure 1 - Spyder script run part 1

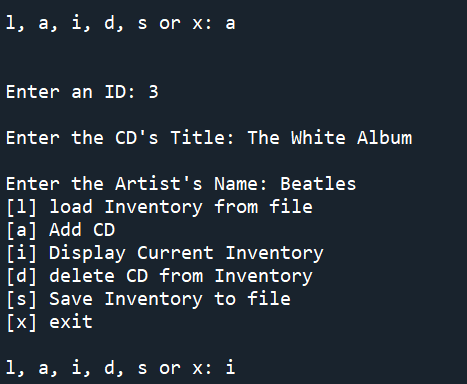


Figure 2 - Spyder script run part 2

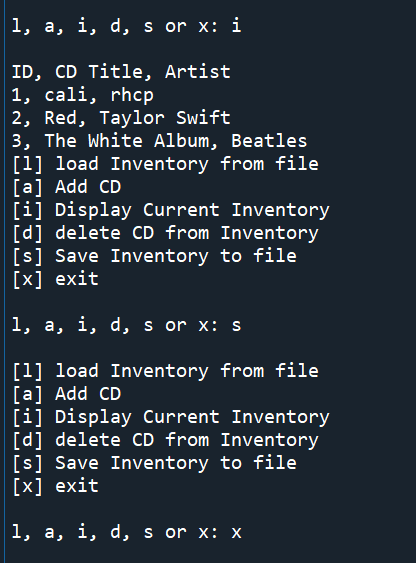


Figure 3 - Spyder script run part 3

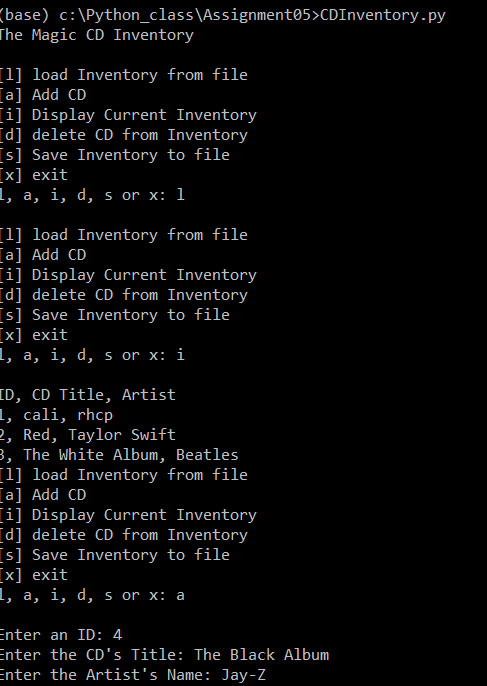


Figure 4 - shell run of script

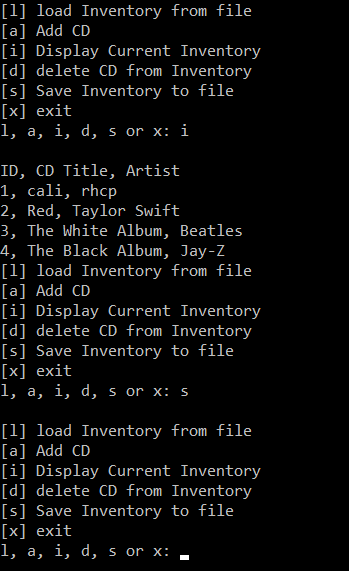


Figure 5 - shell run of script part 2

1. *#------------------------------------------#*
2. *# Title: CDInventory.py*
3. *# Desc: Starter Script for Assignment 05*
4. *# Change Log: (Who, When, What)*
5. ***# DBiesinger, 2030-Jan-01, Created File***
6. *#------------------------------------------#*
8. *# Declare variabls*
10. **strChoice = '' *# User input***
11. lstTbl = [] *# list of lists to hold data*
12. *# TODO replace list of lists with list of dicts*
13. lstRow = {} *# DICTIONARY of data row*
14. strFileName = 'CDInventory.txt' *# data storage file*
15. **objFile = None *# file object***
17. *# Get user Input*
18. **print**('The Magic CD Inventory**\n**')
19. **while** True:
20. ***# 1. Display menu allowing the user to choose:***
21. **print**('[l] load Inventory from file**\n**[a] Add CD**\n**[i] Display Current Inventory')
22. **print**('[d] delete CD from Inventory**\n**[s] Save Inventory to file**\n**[x] exit')
23. strChoice = input('l, a, i, d, s or x: ').lower() *# convert choice to lower case at time of input*
24. **print**()
26. **if** strChoice == 'x':
27. *# 5. Exit the program if the user chooses so*
28. **break**
29. **if** strChoice == 'l':
30. ***# TODO Add the functionality of loading existing data***
31. objFile = open(strFileName, 'r') *# opens existing data file in read mode*
32. lstTbl.clear()
33. **for** row **in** objFile:
34. lstRow = row.strip().split(',')
35. **dicRow = {'ID': lstRow[0], 'Title': lstRow[1], 'Artist': lstRow[2]}**
36. lstTbl.append(dicRow)
37. **elif** strChoice == 'a': *# no elif necessary, as this code is only reached if strChoice is not 'exit'*
38. *# 2. Add data to the table (2d-list) each time the user wants to add data*
39. strID = input('Enter an ID: ')
40. **strTitle = input('Enter the CD\'s Title: ')**
41. strArtist = input('Enter the Artist**\'**s Name: ')
42. intID = int(strID)
43. lstRow = {'ID': intID, 'Title': strTitle, 'Artist': strArtist}
44. lstTbl.append(lstRow)
45. **elif strChoice == 'i':**
46. *# 3. Display the current data to the user each time the user wants to display the data*
47. **print**('ID, CD Title, Artist')
48. **for** row **in** lstTbl:
49. *# print(row)*
50. **print(\*row.values(), sep = ', ')**
51. **elif** strChoice == 'd':
52. *# TODO Add functionality of deleting an entry*
53. **print**('Deleting last entry')
54. **print**(len(lstTbl))
55. **del lstTbl[(len(lstTbl)-1)]**
56. **elif** strChoice == 's':
57. *# 4. Save the data to a text file CDInventory.txt if the user chooses so*
58. objFile = open(strFileName, 'w')
59. **for** row **in** lstTbl:
60. **strRow = ''**
61. **for** item **in** row.values():
62. strRow += str(item) + ','
63. strRow = strRow[:-1] + '**\n**'
64. objFile.write(strRow)
65. **objFile.close()**
66. **else**:
67. **print**('Please choose either l, a, i, d, s or x!')

Listing 1 - CDInventory.py script

# Summary

The class material prepared me to complete this assignment. I did have some hangups on the row.values() concept, and still could use some more studying on \*row. I am enjoying learning about the applications of github, and coding in standard ways so that hundreds (or thousands) of programmers can work on the same project together. We have github at work, but it is scarcely used so I am happy to better understand it. A couple final questions – why would I ever use a tuple? I did see one nice use in dictionaries, requiring an immutable object so (0,0) could be used for the key but [0,0] could not… Otherwise, I’m struggling to find value in tuples. Also, in Lab05-went well but when I read the file, it adds an ‘\n’ to the end that doesn’t actually show up in the .txt file. I specifically added ‘\n’ to the end of the string prior to writing it to the .txt but it doesn’t show up in the .txt (which is great!), but I don’t understand why its invisible there, but pops back up when the .txt is read