Christopher J. Scott

15-Feb-2021

IT FDN 110: Introduction to Programming (Python)

Assignment 05

Working with dictionaries and utilizing GitHub

# Introduction

Module 05 expanded on manipulating data within files and lists, introduced dictionaries and multiple ways to improve scripts including separation of concerns and functions. The online repository GitHub was also introduced and the collaborative nature of giving and receiving feedback on coding was demonstrated. This module culminated with expanding on the previously introduced programs for handling a CD inventory text file.

# Reading and Writing to/from Files

A key piece of making data handleable within a python script is extracting information from a file and ensuring it is in a digestible format. The various arguments within the open() function including “a” (append), “r” (read) and “w” (write) must be specified depending on the intent. For this module specifically, all are used depending on whether information is being read or written into the data file.

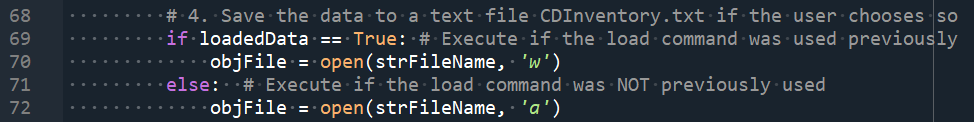


Figure . Various arguments within open() command

Since dictionaries are used, additional functions including split() and strip() are used to take data that is stored in dictionaries within lists to make the individual elements. Using for loops, one is able to extract the individual elements from a dictionary into the working directory.

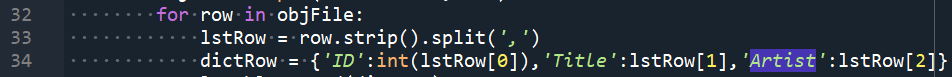


Figure . Extracting information from data files into dictionaries

# Using Dictionaries

Dictionaries are mapping type data sets that use the {key:value} pair relationship as sort of a row-column relationship. There are limitations of the types of data that can be contained in dictionaries; those are strings, numbers and tuples (assuming those tuples also only contain numbers, strings or tuples themselves).

In Assignment05, the individual dictionary was created in a similar way to previously explored lists and tuples. The individual elements were input and then assigned by prescribing the keys and assigning the particular values to those keys as shown in Figure 3.

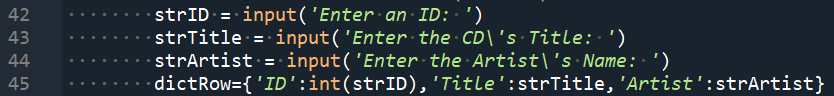


Figure . Assigning dictionary with user input

Additional functionality was added to this script for Assignment 05 including deleting information that presented some challenges. In order to determine which row to delete, the user was presented with the data stored in the working directory for reference. The user then input a particular ID. The program then cycled through all of the dictionaries contained in the outer list until the specific ID was found. At that point, the row was deleted by using the remove() command.

Combining these new features with some of the holdover script that was previously provided yielded a program that allows the user to load, add, display, delete, save and ultimate exit.

# Demonstrating CDInventory.py

To demonstrate the functionality of the CDInventory.py program, an initial CDInventory.txt file was created with some information contained as shown in Figure 4.

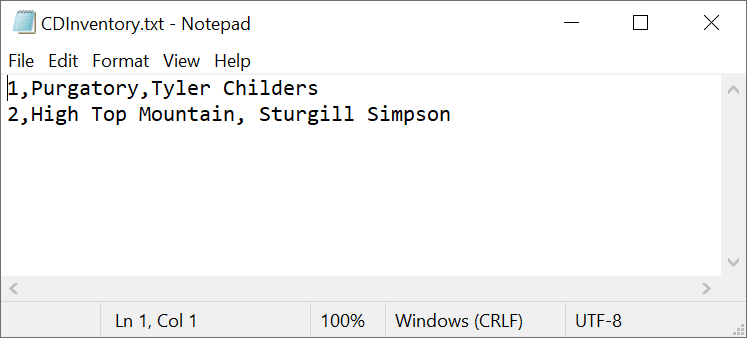


Figure . CDInventory.txt before running CDInventory.py

Figure 5 shows the CDInventory.py program being run within Spyder and shows that all of the functionalities are working as expected.

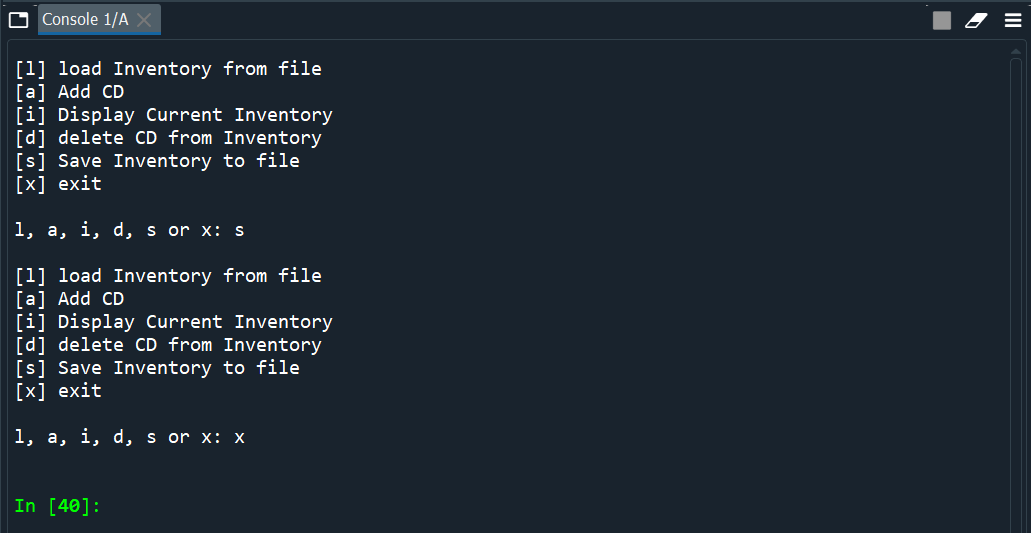
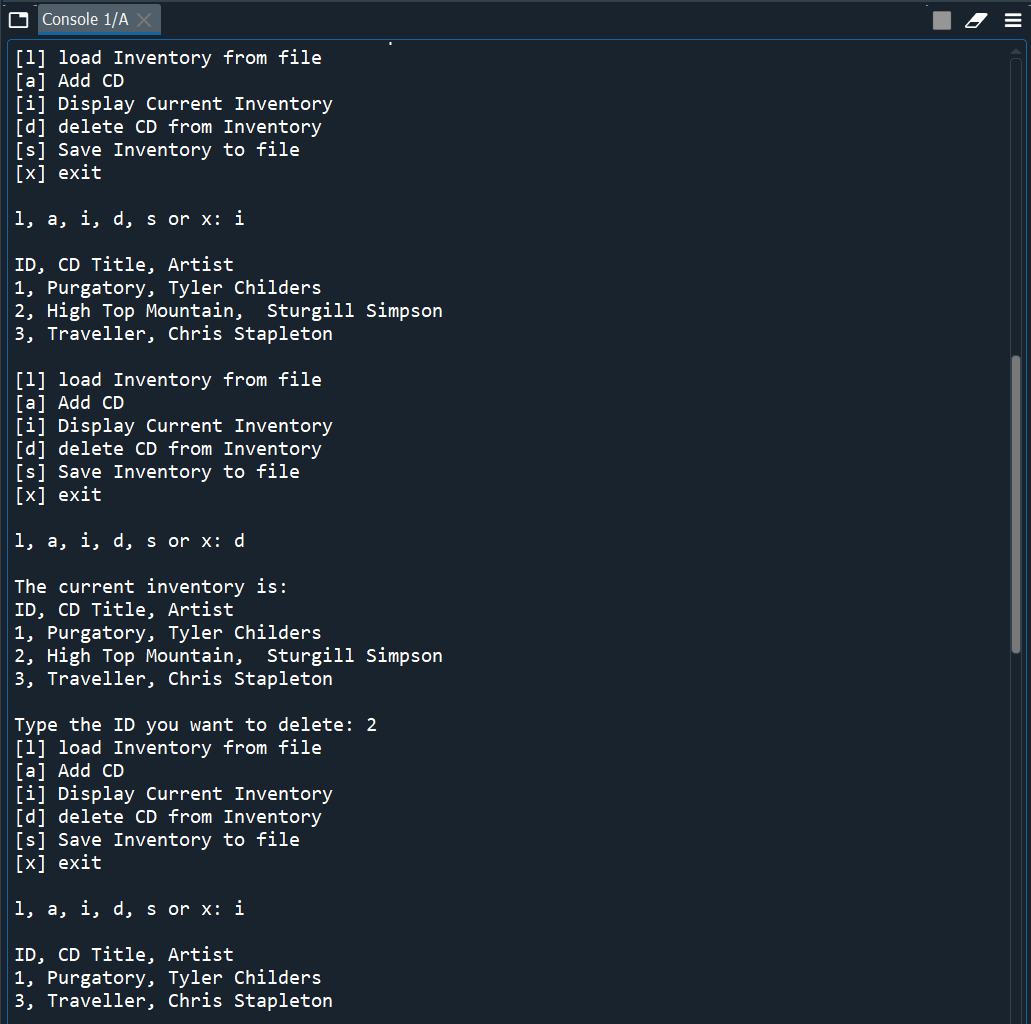
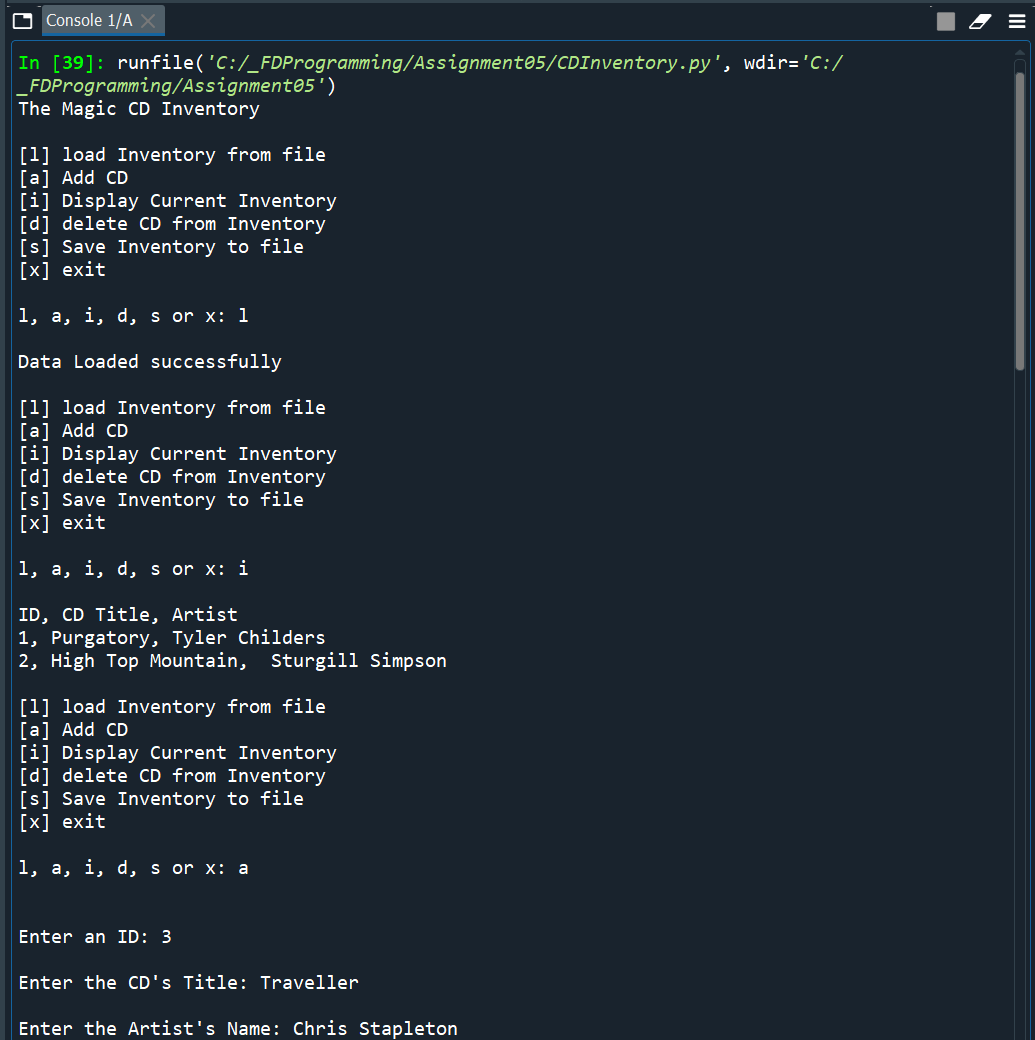


Figure . CDInventory.py being executed within Spyder

Figure 6 shows the resulting CDInventory.txt file after being saved within the spyder application.

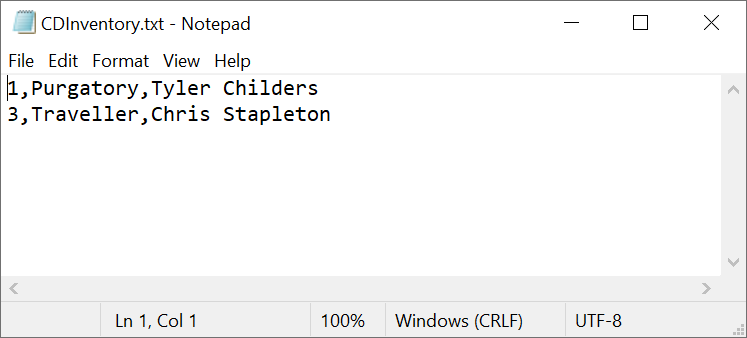


Figure . Result of CDInventory.txt after execution

# Summary

By completing Module 5 and associated Lab exercises, a user is able to create a program that is interactive, utilizes dictionaries as the inner data structure and allows for adding, deleting, displaying and saving of information. A user is also able to upload documents to GitHub and learn the benefits of peer review. My repository that contains this specific CDInventory.py can be found [here](https://github.com/cscott08vt/Assignment05/blob/main/CDInventory.py).

# Appendix

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. # CScott, 2021-Feb-14, Modified the program
7. #------------------------------------------#
9. # Declare variables
11. strChoice = '' # User input
12. deleteID = '' # User input for deleting specific ID
13. lstTbl = []  # list of dictionaries to hold data
14. dictRow = {}  # dictionary of data row
15. strFileName = 'CDInventory.txt'  # data storage file
16. objFile = None  # file object
18. # Get user Input
19. **print**('The Magic CD Inventory\n')
20. **while** True:
21. # 1. Display menu allowing the user to choose:
22. **print**('[l] load Inventory from file\n[a] Add CD\n[i] Display Current Inventory')
23. **print**('[d] delete CD from Inventory\n[s] Save Inventory to file\n[x] exit')
24. strChoice = input('l, a, i, d, s or x: ').lower()  # convert choice to lower case at time of input
25. **print**()
27. **if** strChoice == 'x':
28. # 5. Exit the program if the user chooses so
29. **break**
30. **if** strChoice == 'l':
31. objFile = open(strFileName, 'r') # Read in txt file
32. **for** row **in** objFile:
33. lstRow = row.strip().split(',')
34. dictRow = {'ID':int(lstRow[0]),'Title':lstRow[1],'Artist':lstRow[2]} # Assign indexed data into dictionary values
35. lstTbl.append(dictRow)
36. objFile.close()
37. **print**('Data Loaded successfully\n')
38. loadedData = True
39. **pass**
40. **elif** strChoice == 'a':  # no elif necessary, as this code is only reached if strChoice is not 'exit'
41. # 2. Add data to the table (2d-list) each time the user wants to add data
42. strID = input('Enter an ID: ')
43. strTitle = input('Enter the CD\'s Title: ')
44. strArtist = input('Enter the Artist\'s Name: ')
45. dictRow={'ID':int(strID),'Title':strTitle,'Artist':strArtist}
46. lstTbl.append(dictRow)
47. **elif** strChoice == 'i':
48. # 3. Display the current data to the user each time the user wants to display the data
49. **print**('ID, CD Title, Artist')
50. **for** row **in** lstTbl:
51. **print**(\*row.values(), sep = ', ')
52. **print**()
53. **elif** strChoice == 'd':
54. **print**('The current inventory is:')
55. **print**('ID, CD Title, Artist')
56. **for** row **in** lstTbl:
57. **print**(\*row.values(), sep = ', ')
58. deleteID = int(input('Type the ID you want to delete: '))
59. foundID = False
60. **for** row **in** lstTbl:
61. **if** deleteID == row['ID']:
62. lstTbl.remove(row)
63. foundID = True
64. **if** foundID == False:
65. **print**('\nID ' + str(deleteID) + ' not found in inventory')
66. **print**()
67. **elif** strChoice == 's':
68. # 4. Save the data to a text file CDInventory.txt if the user chooses so
69. **if** loadedData == True: # Execute if the load command was used previously
70. objFile = open(strFileName, 'w')
71. **else**:  # Execute if the load command was NOT previously used
72. objFile = open(strFileName, 'a')
73. **for** row **in** lstTbl:
74. strRow = ''
75. **for** item **in** row.values():
76. strRow += str(item) + ','
77. strRow = strRow[:-1] + '\n'
78. objFile.write(strRow)
79. objFile.close()
80. **else**:
81. **print**('Please choose either l, a, i, d, s or x!')

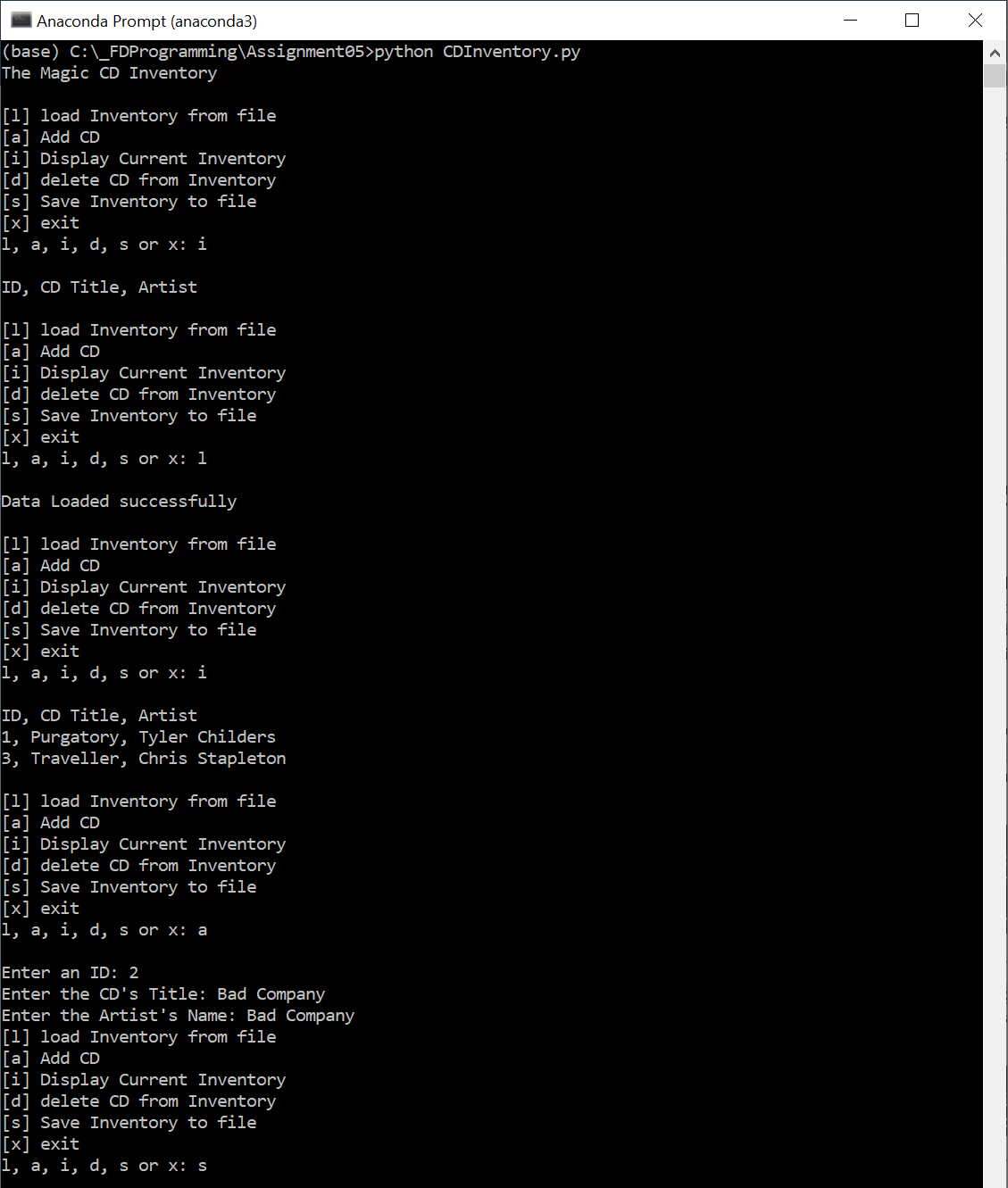


Figure . CDInventory.py being used in terminal