Graham Byron

27 February 2022

IT FDN 110: Introduction to Programming (Python)

Assignment 05

Assignment 05: Tuples, Lists, and Dictionaries

Intro

Assignment 05 focused around the use of lists and newly introduced dictionaries. Further, formatting and best practices were covered as well. This was touched on with separation of concerns, error handling, functions, and script templates. Lastly we were introduced to GitHub, an online platform that allows users to post scripts, view others' work, and collaborate together. The emphasis in this chapter revolved around the use of dictionaries in function use. Several examples were given of code in list form that could then be modified to be held in dictionary form while touching on saving these files correctly. Further, examples of how to cleanly format one's code were given to illustrate how to efficiently and cleanly convey script, while also showing how to handle a situation when the user doesn’t enter functional data. This culminated into the knowledge check that asked to modify code to be structured by dictionaries, adding two additional functionalities, and posting it to my newly created GitHub profile.

Topic

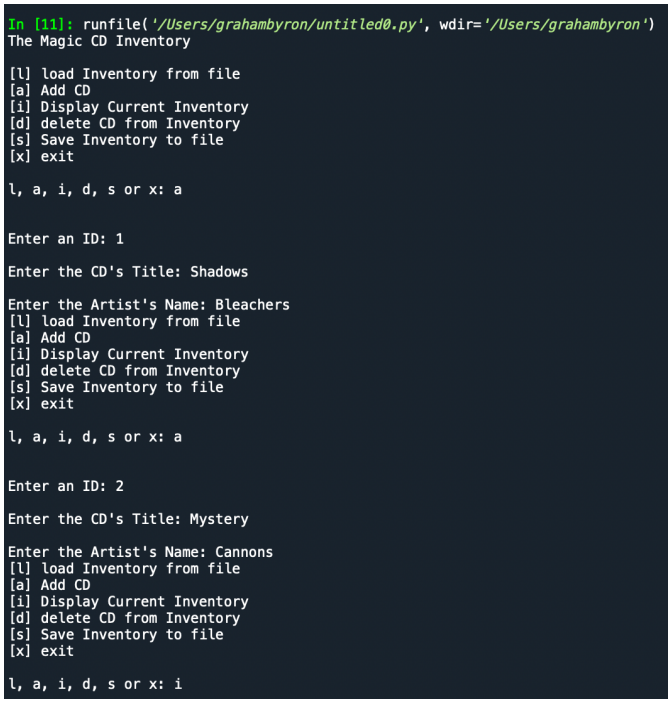
Before reaching the most notable portion of this assignment, a continuation of lists was discussed regarding how to write and load lists into a file. Following this was a lab which ran similar to the knowledge application from Assignment 04. While it was mostly the same, I was tasked with adding a functionality to read the data from the file as well. While simple, I found it difficult to be satisfied with the functionality of it, as the core functionality of ‘read’ would not return any output.

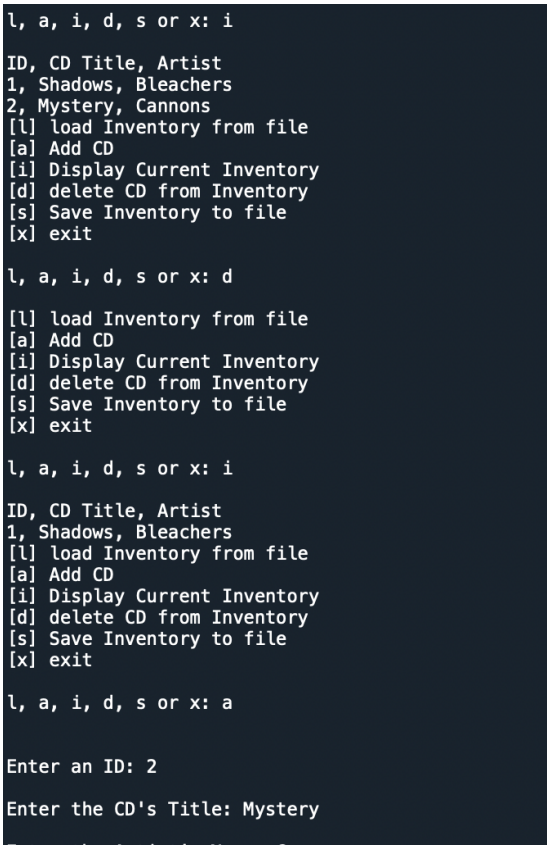
After running this a few times it was time to turn attention to the use of dictionaries. While they hold information in a similar way to the other collection types already learned, the way in which they contain and access data differ. While the previously discussed types are sequence types, dictionaries are mapping types, which means they replace the index storing with key:value pairs. This means that a piece of information is not referenced by it’s sequence number, but instead by it’s key. As with the other functions, there are several different operations that can be made on dictionaries, too many to go in depth here. Further, many examples were given showing the different ways in which the information within the dictionary could be printed. This led to the second lab, which tasked myself with replacing the inner lists with dictionaries instead. In doing so, there were a couple of important changes that needed to be made in order for it to save and print correctly. This will be shown in the ‘Apply Your Knowledge’ section.

After a groundwork was built on dictionaries, emphasis was put on formatting. This can be done a number of ways including separation of concerns, functions, script templates, and structured error handling. Separation of concerns revolves around the structure of the code itself, breaking it down into three categories: Data, Processing, and Presentation (Input/Output). Functions make it easier to section code. It allows the creator to define a function that can be called upon later for the output, this cleans up the flow and image of the code itself. Script templates are tangential to this, as they provide the structure of the separation of concerns mentioned above. Lastly is structured error handling, which ensures that the script does not get aborted when the user doesn't input functional values. This keeps the script running and flows on track. Finally, a walkthrough was given about joining GitHub, where the script discussed below is posted.

After going through that, it was time for the knowledge application portion. I was tasked with altering the script to change the inner data structures to dictionaries while also adding two additional functionalities. While having the initial code was nice, just like the '**Hint**' said, it was difficult to keep track of the other necessary changes that were needed to keep the script running properly. After the changes were made, the code was able to add user input, load the current inventory, display the current inventory, delete from the inventory, save the inventory, and exit all on user command. Below are the photos of the script in action:

Output in Spyder





```
Enter the CD's Title: Mystery

Enter the Artist's Name: Cannons
[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: s

[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: r

Please choose either l, a, i, d, s or x!
[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: x


In [12]:
```

And now in the terminal

Output in Terminal

```
(base) grahambyron@FVFF8CE1Q6L4 Python Scripts % python /Users/grahambyron/De
sktop/Python\ Scripts/Assignment05/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: a

Enter an ID: 1
Enter the CD's Title: Shadows
Enter the Artist's Name: Bleachers
[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: a

Enter an ID: 2
Enter the CD's Title: Mystery
Enter the Artist's Name: Cannons
[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, Shadows, Bleachers
2, Mystery, Cannons
[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: d
```

```
[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, Shadows, Bleachers
[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: a

Enter an ID: 2
Enter the CD's Title: Mystery
Enter the Artist's Name: Cannons
[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: s

[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

[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: x

(base) grahambyron@FVFF8CE1Q6L4 Python Scripts %
```

Success! The user was able to perform all of the necessary functions for the code.

Conclusion

Assignment 05 focused on the use of lists and more importantly (in this section at least) the use of dictionaries. Further, additional emphasis was given to adding additional functionalities/options for a user. This was brought together in altering previous code to accompany dictionaries, adding two additional functionalities, and saving the user input to a file.

(Apologies on the white space, formatting did not want to agree with me)

Appendix

Code for Knowledge Test

```
#------------------------------------------#
# Title: CDInventory.py
# Desc: Starter Script for Assignment 05
# Change Log: (Who, When, What)
# DBiesinger, 2030-Jan-01, Created File
# GByron, 2022-Feb-27, Modified File to List of Dictionaries
#------------------------------------------#

# Declare variables

strChoice = '' # User input
lstTbl = []  # list of lists to hold data
# TODO replace list of lists with list of dicts
dicRow = {}  # dict of data row
strFileName = 'CDInventory.txt'  # data storage file
objFile = None  # file object

# Get user Input
print('The Magic CD Inventory\n')
while True:
    # 1. Display menu allowing the user to choose:
    print('[l] load Inventory from file\n[a] Add CD\n[i] Display Current Inventory')
    print('[d] delete CD from Inventory\n[s] Save Inventory to file\n[x] exit')
    strChoice = input('l, a, i, d, s or x: ').lower()  # convert choice to lower case at time of input
    print()

    if strChoice == 'x':
        # 5. Exit the program if the user chooses so
        break
    if strChoice == 'l':
        # TODO Add the functionality of loading existing data
        objFIle = open(strFileName, 'r')
        for row in objFIle:
            dicRow = row.strip().split(',')
        objFIle.close()
        pass
    elif strChoice == 'a':  # no elif necessary, as this code is only reached if strChoice is not 'exit'
        # 2. Add data to the table (2d-list) each time the user wants to add data
        strID = input('Enter an ID: ')
        strTitle = input('Enter the CD\'s Title: ')
        strArtist = input('Enter the Artist\'s Name: ')
        intID = int(strID)
        dicRow = {'ID': intID, 'Title': strTitle, 'Artist': strArtist}
        lstTbl.append(dicRow)
    elif strChoice == 'i':
        # 3. Display the current data to the user each time the user wants to display the data
        print('ID, CD Title, Artist')
        for row in lstTbl:
            print(*row.values(), sep = ', ')
    elif strChoice == 'd':
        # TODO Add functionality of deleting an entry
        lstTbl.remove(dicRow)
        pass
    elif strChoice == 's':
```

```
55              # 4. Save the data to a text file CDInventory.txta if the user chooses so
56              objFile = open(strFileName, 'w')
57              for row in lstTbl:
58                  strRow = ''
59                  for item in row.values():
60                      strRow += str(item) + ','
61                  strRow = strRow[:-1] + '\n'
62                  objFile.write(strRow)
63              objFile.close()
64          else:
65              print('Please choose either l, a, i, d, s or x!')
```