Emeka Osuji
August 6th, 2022
Foundations of Programming, Python
Assignment 5

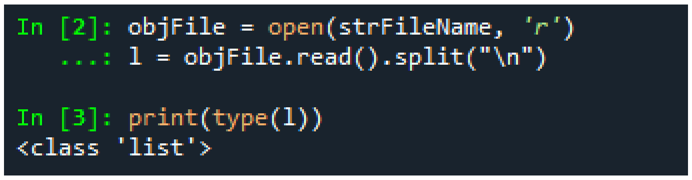
ADDING FUNCTIONALITIES TO EXISTING PYTHON DATA LIST

INTRODUCTION

In this paper, we will discuss some of the challenges involved in adding functionalities to existing python code. While this assignment is an extension of Assignment 4 from the previous week, it also introduces the concept of dictionaries and their relative features. For this assignment, we will need to generate a 2D data structure which encompasses a list of dictionaries containing user data inputs. We will also add functionalities to load data from exiting text files and delete undesired row. The script will be run in Spyder and then in the anaconda terminal as proof of successful execution.

Loading…

Loading in existing data from a text file was perhaps the hardest part of this assignment. First, it was important to be aware of the structure in which the data is saved in the existing file. Knowing this would make it easier to address method implementations such as the `split()` and `strip()` method. It was unclear how to exactly to use the strip method. Repeated efforts to use it resulted in error reading stating that the attribute was nonexistent. These errors, along with other, were quickly rectified once I realized that the data was being read in as a list data type as opposed to a string with the `split()` method applied



```
In [2]: objFile = open(strFileName, 'r')
   ...: l = objFile.read().split("\n")

In [3]: print(type(l))
<class 'list'>
```

Figure 1: Data type shown as list once read and split

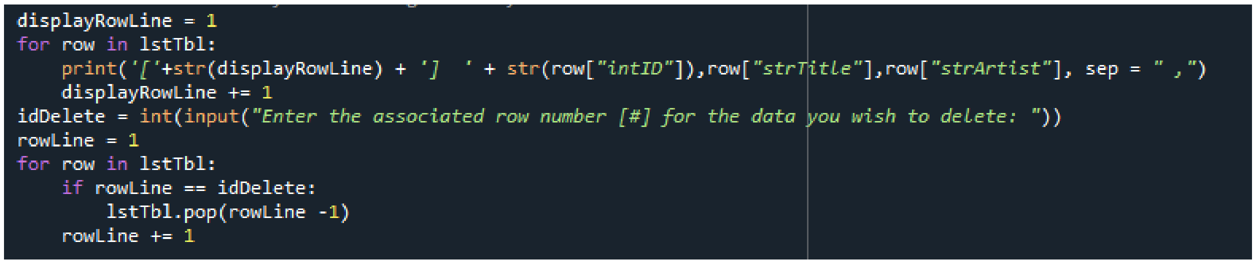
Once the read in data had been separated in list entries by newlines (‘\n’), they were further split again by the commas. This process is what allowed for the successful execution of the code while bypassing the use of the strip method which was suggested in the modules. The final step was to now convert each list into dictionaries to then be stored in a larger list. This step would fulfil the requirement of having dictionaries within a list. With the data now loaded, which were now converted to dictionaries and stored in a larger list, new dictionary data could now be appended to the larger list for further modification.

“You’ve Just been Erased”

The step of adding a data deletion functionality seemed quite straight forward until an attempt was made. It was a simple error of misaligned data types when trying to delete a row that proved to be the issue. The script was designed to request a user input on the ID number they wished to delete. This input value along with the data type of the stored ID value were stored as integers. However, when loading in existing data, there was an oversight in converting the loaded ID values to integers. Therefore, once a user attempted to delete a data row which was loaded in and stored, the script was unable to find this row because it was stored as a string instead of an integer.

Once the correction was made, further test revealed some limitations with this approach to the script. Should a user attempt to delete a data input which had an identical ID number with another data row, both rows would be deleted. This means that the script limits the autonomy of the user to delete a specific data row, even though it may have a similar ID. For the purpose of this assignment, the script was relegated to data deletion via ID number identification rather than row specificity.

However, if we wished to delete data entries based on specific rows, we could implement a script like what can be found in figure 2. Here, the script first prints the stored data in rows starting from 1 and incremented consecutively for user visualization. The user can then choose which specific row they wish to delete regardless of the row ID number



```
displayRowLine = 1
for row in lstTbl:
    print('['+str(displayRowLine) + ']  ' + str(row["intID"]),row["strTitle"],row["strArtist"], sep = " ,")
    displayRowLine += 1
idDelete = int(input("Enter the associated row number [#] for the data you wish to delete: "))
rowLine = 1
for row in lstTbl:
    if rowLine == idDelete:
        lstTbl.pop(rowLine -1)
    rowLine += 1
```

Figure 2: Alternate code for deleting data entries by specific rows rather than specified ID number

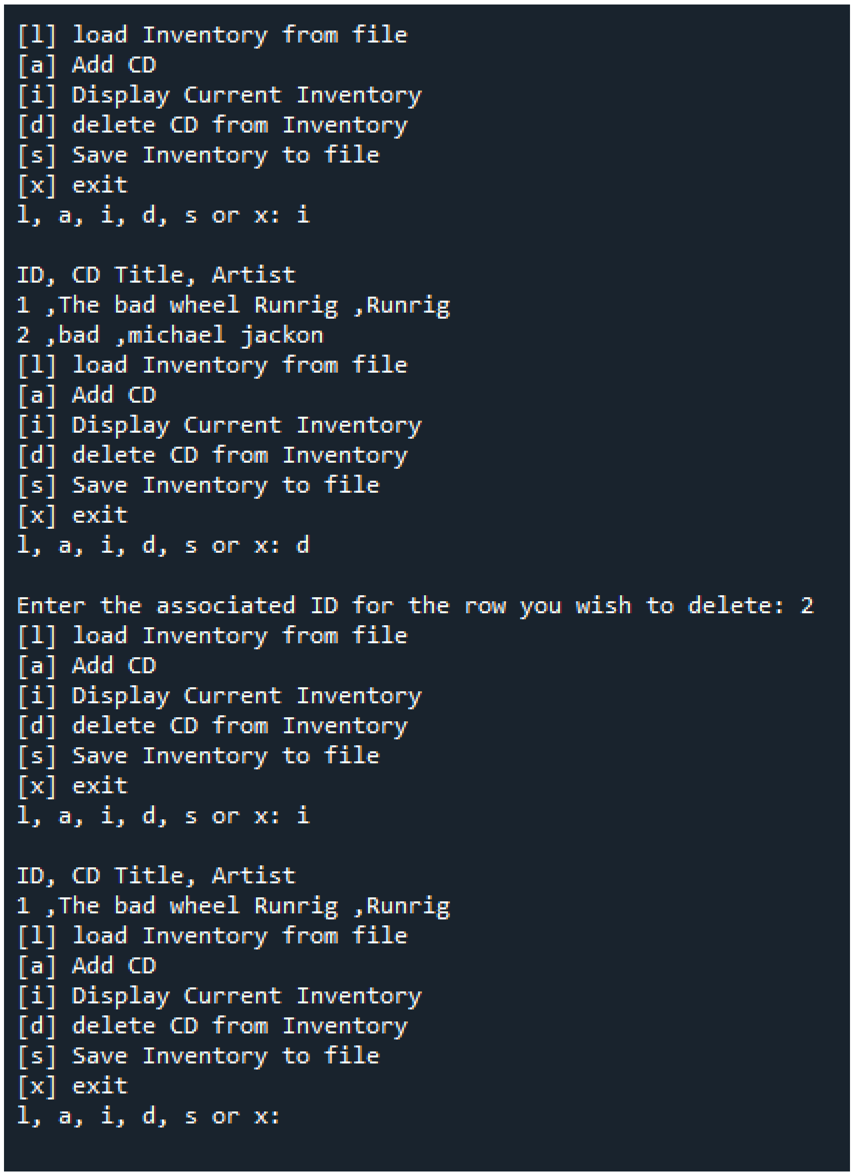
Once the script had been generated, it was tested and ran in Spyder and in the anaconda user terminal. As can be seen in Appendix I and II, the script performs as expected by loading in existing data, providing the ability to add and deleted data, and finally being able to resave the data and exit the program. The python script for this assignment along with the this paper can also be found on GitHub (https://github.com/sirRockIII/Assignment_05.git).

Summary

In this assignment we added functionalities to an existing code. These functionalities include the ability to load in data from an existing file and to delete data from the loaded in or added entries. This process required that the data being loaded in was in a format that could be read, modified and stored in dictionaries. We also had to pay close attention to the different data types being manipulated in order to make sure we could use the appropriate attributes. While we were able to add functionality to delete data, we also realized that the user may wish to delete data based on specific rows rather than the associated ID if for some reason 2 data rows share the same ID number. An alternative code for specific data row deletion was provided in figure 2.

Appendix

I)



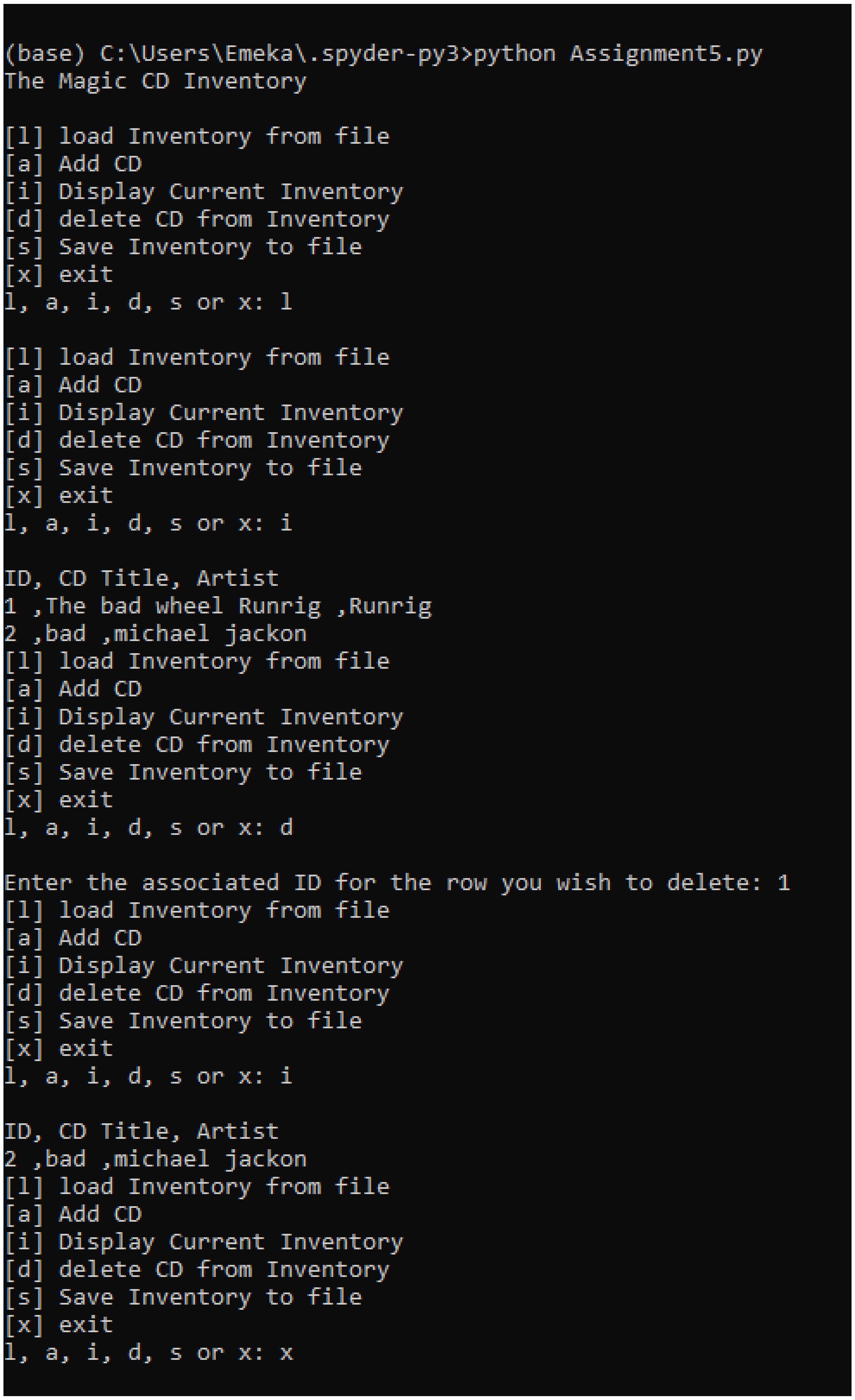
```
[l] load Inventory from file
[a] Add CD
[i] Display Current Inventory
[d] delete CD from Inventory
[s] Save Inventory to file
[x] exit
l, a, i, d, s or x: i

ID, CD Title, Artist
1 ,The bad wheel Runrig ,Runrig
2 ,bad ,michael jackon
[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

Enter the associated ID for the row you wish to delete: 2
[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 bad wheel Runrig ,Runrig
[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:
```

II)



```
(base) C:\Users\Emeka\.spyder-py3>python Assignment5.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

[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 bad wheel Runrig ,Runrig
2 ,bad ,michael jackon
[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

Enter the associated ID for the row you wish to delete: 1
[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
2 ,bad ,michael jackon
[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
```