Graham Byron

16 March 2022

IT FDN 110: Introduction to Programming (Python)

Assignment 07

Assignment 07: Functions and Classes

Intro

Assignment 07 centered around the use of text and binary files. Further, structured error handling was discussed and demonstrated as well. Lastly, we worked with saving the files in a specific manner and again posted the final write up and code to GitHib. The emphasis of this section revolved around working with text files; specifically with write, read and append. Several examples of code were given followed by how the data was saved within the system. This included pickling, which allows for the coder to save the data in a safe way as binary information. Structured error handling was lightly touched on as well, giving multiple examples of the errors that could occur when dealing with user interface, file access, etc. We were urged to do our own research on the above two topics to further our ability to find pertinent and helpful information. After this, we were tasked with altering code to include both structured error handling and altering the permanent data store to use binary data.

Topic

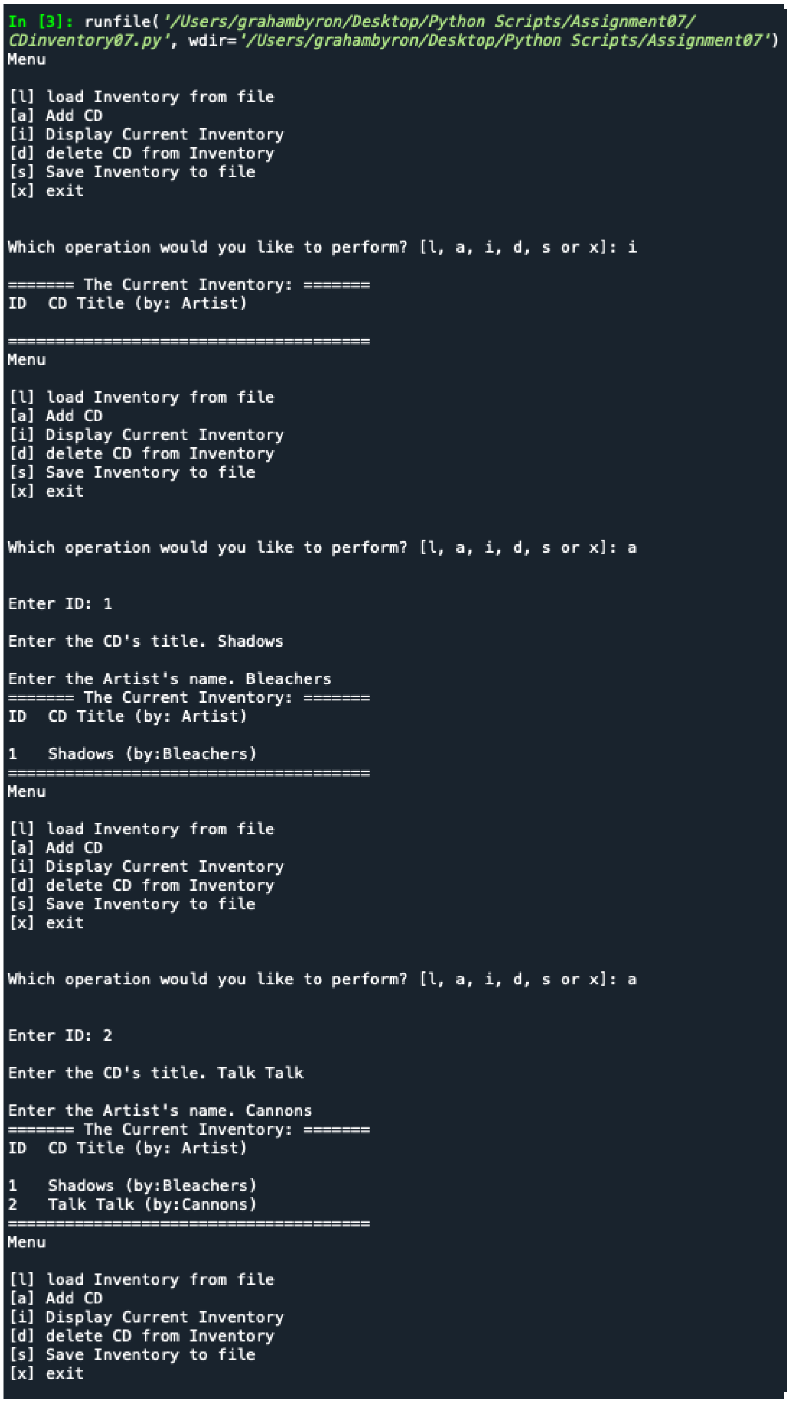
The first item touched on in this lab was working with data files. This first was discussed through base level groundwork already established about the read, write, and append functionality. This knowledge was furthered through the explanation and demonstration of the readline() function, which can be used in a while loop. A different way to go about this is the readlines() function, which is able to accomplish the same goal without a while loop. Further, the use of the for loop as well as the ‘with’ operator were discussed to give a stronger understanding of the functionalities.

The next item discussed was working with binary files. We had been working with file saving techniques that are easily readable, however this is not the same way in which data is stored in memory. The easier option is to save the information the same way that it is in memory, which is binary code. This is the art of pickling, which serializes the data in a way that can be stored as binary information. An example is given to help gain an understanding of the flow and look of pickling in action. Further, the saved output is given to illustrate the different way in which it is saved, which looks significantly different! At this point I did look at a couple of websites that helped me to gain a better understanding of the powers of pickling.

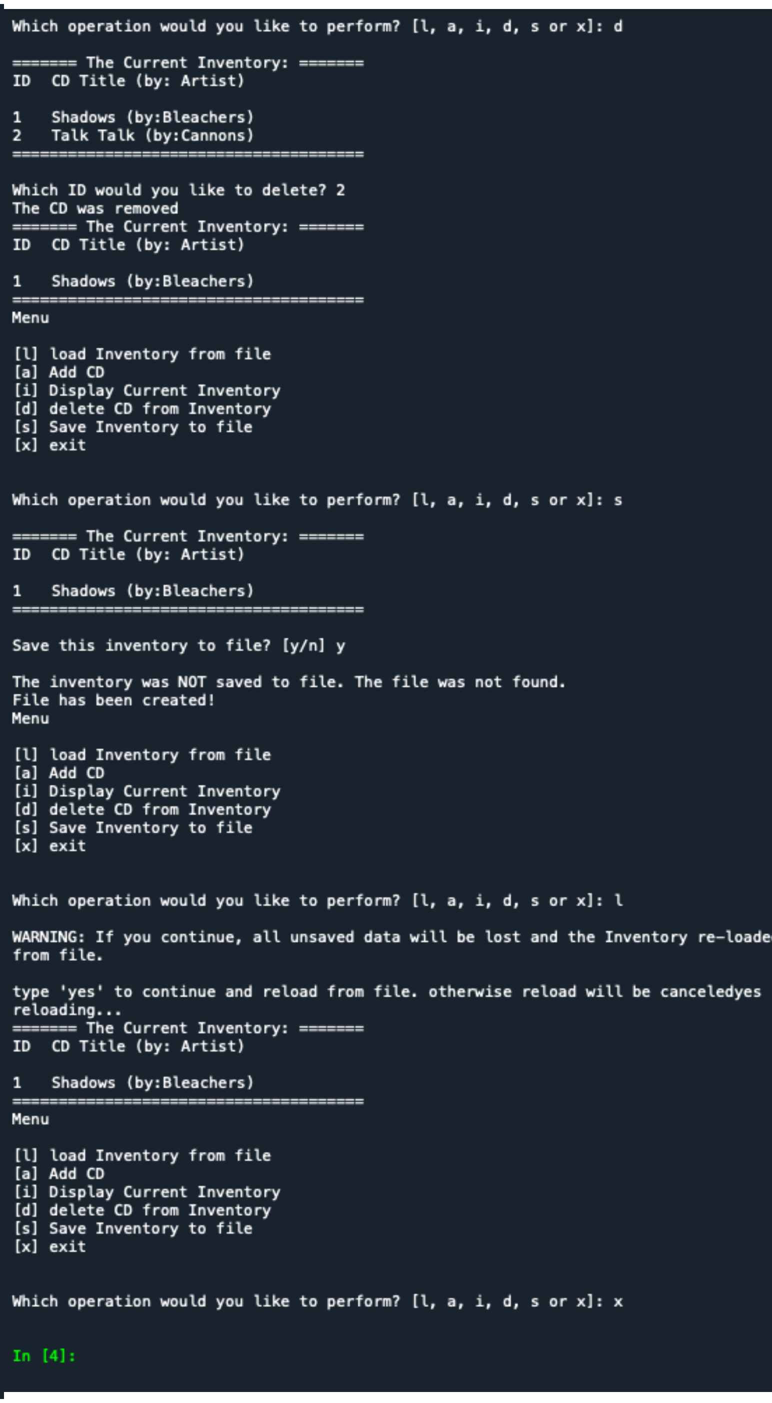
Lastly was the topic of structured error handling. A few of the potential errors are mentioned like ZeroDivisionError, ValueError, and FileNotFound. However there are many other errors that could occur. The times in which structured error handling would be helpful are times when there is interaction with the user, interaction with the file, and interaction with other coders may interact with. Times like these that can terminate the function can be stopped through error handling through a try-except block. This will allow the coder to handle the errors that may pop up and avert termination of the entire program. This is further built upon through the exception class, which can show the details about the error itself. Again, we were encouraged to do some of our own research on the subject, of which some of the links I visited are in the appendix.

After this was the knowledge portion of the assignment. I was tasked with modifying my assignment from last week to include structure error handling. Further, I needed to change the data storage to save and be used in binary data. Below is the Python script in action.

Code Output in Spyder

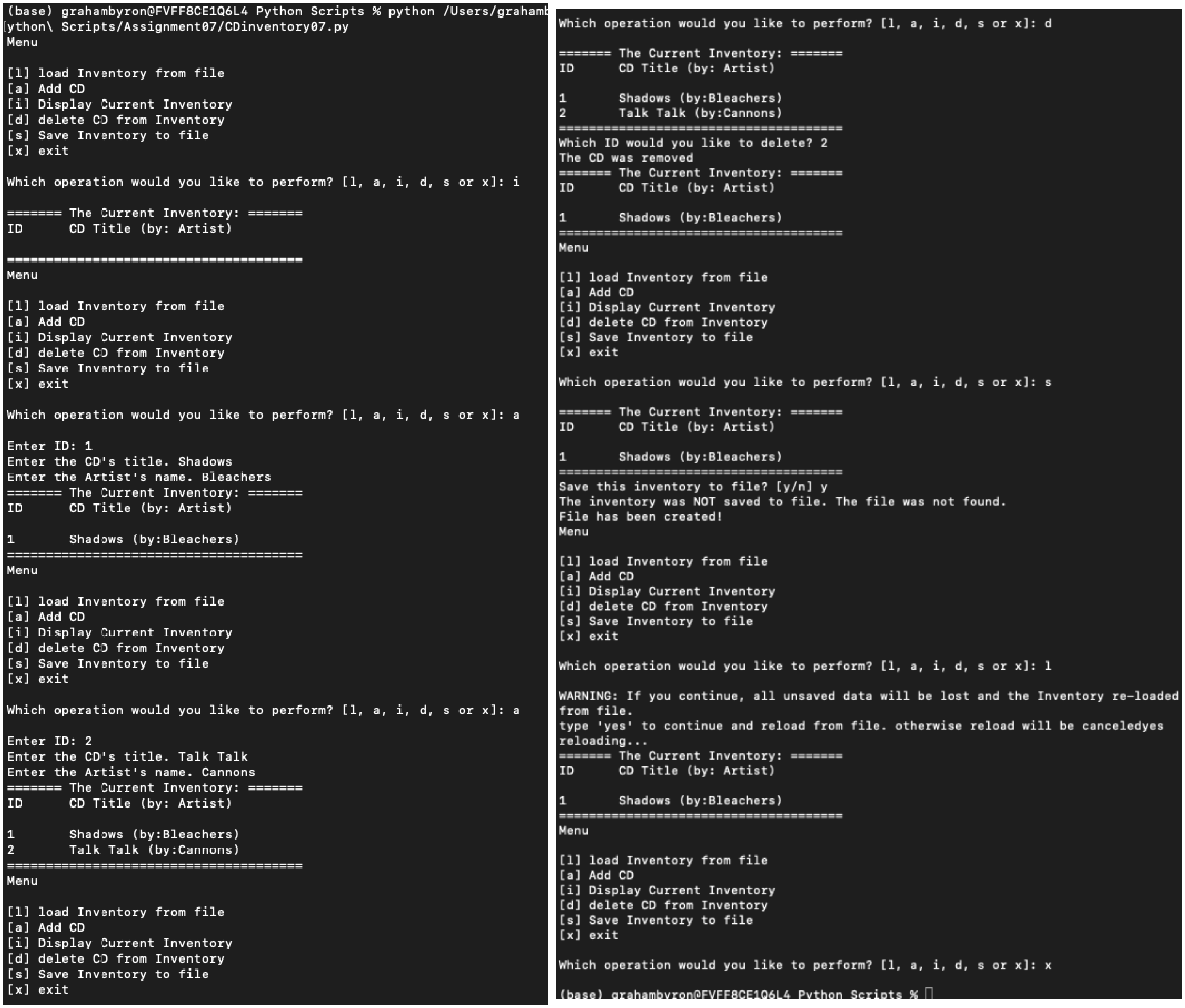


```
In [3]: runfile('/Users/grahambyron/Desktop/Python Scripts/Assignment07/
CDinventory07.py', wdir='/Users/grahambyron/Desktop/Python Scripts/Assignment07')
Menu

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


Which operation would you like to perform? [l, a, i, d, s or x]: i

======= The Current Inventory: =======
ID  CD Title (by: Artist)

======================================
Menu

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


Which operation would you like to perform? [l, a, i, d, s or x]: a


Enter ID: 1

Enter the CD's title. Shadows

Enter the Artist's name. Bleachers
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Shadows (by:Bleachers)
======================================
Menu

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


Which operation would you like to perform? [l, a, i, d, s or x]: a


Enter ID: 2

Enter the CD's title. Talk Talk

Enter the Artist's name. Cannons
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Shadows (by:Bleachers)
2   Talk Talk (by:Cannons)
======================================
Menu

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



```
Which operation would you like to perform? [l, a, i, d, s or x]: d

======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Shadows (by:Bleachers)
2   Talk Talk (by:Cannons)
======================================

Which ID would you like to delete? 2
The CD was removed
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Shadows (by:Bleachers)
======================================
Menu

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


Which operation would you like to perform? [l, a, i, d, s or x]: s

======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Shadows (by:Bleachers)
======================================

Save this inventory to file? [y/n] y

The inventory was NOT saved to file. The file was not found.
File has been created!
Menu

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


Which operation would you like to perform? [l, a, i, d, s or x]: l

WARNING: If you continue, all unsaved data will be lost and the Inventory re-loade
from file.

type 'yes' to continue and reload from file. otherwise reload will be canceledyes
reloading...
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Shadows (by:Bleachers)
======================================
Menu

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


Which operation would you like to perform? [l, a, i, d, s or x]: x


In [4]:
```

Output in the Terminal



```
(base) grahambyron@FVFF8CE1Q6L4 Python Scripts % python /Users/grahamb
[ython\ Scripts/Assignment07/CDinventory07.py
Menu

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

Which operation would you like to perform? [l, a, i, d, s or x]: i

======= The Current Inventory: =======
ID      CD Title (by: Artist)

======================================
Menu

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

Which operation would you like to perform? [l, a, i, d, s or x]: a

Enter ID: 1
Enter the CD's title. Shadows
Enter the Artist's name. Bleachers
======= The Current Inventory: =======
ID      CD Title (by: Artist)

1       Shadows (by:Bleachers)
======================================
Menu

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

Which operation would you like to perform? [l, a, i, d, s or x]: a

Enter ID: 2
Enter the CD's title. Talk Talk
Enter the Artist's name. Cannons
======= The Current Inventory: =======
ID      CD Title (by: Artist)

1       Shadows (by:Bleachers)
2       Talk Talk (by:Cannons)
======================================
Menu

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

```
Which operation would you like to perform? [l, a, i, d, s or x]: d

======= The Current Inventory: =======
ID      CD Title (by: Artist)

1       Shadows (by:Bleachers)
2       Talk Talk (by:Cannons)
======================================
Which ID would you like to delete? 2
The CD was removed
======= The Current Inventory: =======
ID      CD Title (by: Artist)

1       Shadows (by:Bleachers)
======================================
Menu

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

Which operation would you like to perform? [l, a, i, d, s or x]: s

======= The Current Inventory: =======
ID      CD Title (by: Artist)

1       Shadows (by:Bleachers)
======================================
Save this inventory to file? [y/n] y
The inventory was NOT saved to file. The file was not found.
File has been created!
Menu

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

Which operation would you like to perform? [l, a, i, d, s or x]: l

WARNING: If you continue, all unsaved data will be lost and the Inventory re-loaded
from file.
type 'yes' to continue and reload from file. otherwise reload will be canceledyes
reloading...
======= The Current Inventory: =======
ID      CD Title (by: Artist)

1       Shadows (by:Bleachers)
======================================
Menu

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

Which operation would you like to perform? [l, a, i, d, s or x]: x

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

Again, I found it difficult to modify the script without messing up any of the pre-establish diction and flow of the code. I found myself combing over the code only to find that there was a missing ‘:’ or indent that shouldn’t be where it was. That has helped me learn the importance of diligence when writing out and modifying a script.

Conclusion

Assignment 07 revolved around the functionalities around text and binary files. Further, there was additional emphasis to pickling and structured error handling in which I was guided to do self-direct research to help gain a stronger understanding of these two items This research was used to tie together the module in the knowledge portion, which tasked me with altering my own code to save files as binary data and include structured error handling where necessary.

Appendix

Links:

https://www.geeksforgeeks.org/python-exception-handling/

https://realpython.com/python-exceptions/

https://stackoverflow.com/questions/57007680/how-to-handle-the-exception-when-input-file-does-not-exists-in-python

https://docs.python.org/3/library/pickle.html#:~:text=%E2%80%9CPickling%E2%80%9D%20is%20the%20process%20whereby,back%20into%20an%20object%20hierarchy.

https://www.synopsys.com/blogs/software-security/python-pickling/

GitHub Link

https://github.com/gwiby123/Assignment_07

Python Script

```
#------------------------------------------#
# Title: Assignment06_Starter.py
# Desc: Working with classes and functions.
# Change Log: (Who, When, What)
# DBiesinger, 2030-Jan-01, Created File
# GByron, 2022-Mar-09, Modified/ File and Completed Alterations
# Gbyron, 2022-Mar-13, Added Structured Error Handling and Binary Data Storage
# Gbyron, 2022-Mar-16, Adjusted code to add more outputs to user
#------------------------------------------#

import pickle

# -- DATA -- #
strChoice = ''
lstTbl = []
dicRow = {}
strFileName = 'CDInventory.dat'
objFile = None


# -- PROCESSING -- #
class DataProcessor:
    """ Searches for ID in table, if it exists entry is deleted.
     if not user recieves 'Could not find this CD!' """

    def delete_cd_from_table(intIDDel):
        intRowNr = -1
        blnCDRemoved = False
        for row in lstTbl:
            intRowNr += 1
            if row['ID'] == intIDDel:
                del lstTbl[intRowNr]
                blnCDRemoved = True
                break
        if blnCDRemoved:
            print('The CD was removed')
        else:
            print('Could not find this CD!')

    """"This saves the data within the system"""

    def save_data(data, strFileName):
        try:
            open('CDInventory.dat')
        except FileNotFoundError:
            input('The inventory was NOT saved to file. The file was not found.')
            print('File has been created!')
            with open(strFileName, 'wb') as fileObj:
                pickle.dump(data, fileObj)

    """ Adding the CD to memory

    Args: None

    Returns:
        cd_lst, calling on below function of add_cd
        dicRow, dictionary of user input items to be saved into the file


    """

    def adding_cd():
        cd_lst= IO.add_cd()
        dicRow = {'ID': cd_lst[0], 'Title': cd_lst[1], 'Artist': cd_lst[2]}
        lstTbl.append(dicRow)

class FileProcessor:
    """Processing the data to and from text file"""
    @staticmethod
    def read_file(file_name, table):
        """Function to manage data ingestion from file to a list of dictionaries

        Reads the data from file identified by file_name into a 2D table
        (list of dicts) table one line in the file represents one dictionary row in table.

        Args:
            file_name (string): name of file used to read the data from
            table (list of dict): 2D data structure (list of dicts) that holds the data during runtime

        Returns:
            None.
        """
        try:
            open('CDInventory.dat')
        except FileNotFoundError:
            input('The inventory was NOT read. The file was not found.')
            with open(file_name, 'rb') as fileObj:
                data = pickle.load(fileObj)
            return data


# -- PRESENTATION (Input/Output) -- #

class IO:
    """Handling Input / Output"""

    @staticmethod
    def print_menu():
        """Displays a menu of choices to the user

        Args:
            None.

        Returns:
            None.
        """

        print('Menu\n\n[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\n')

    @staticmethod
    def menu_choice():
        """Gets user input for menu selection

        Args:
            None.

        Returns:
            choice (string): a lower case sting of the users input out of the choices l, a, i, d, s or x

        """
        choice = ' '
        while choice not in ['l', 'a', 'i', 'd', 's', 'x']:
            choice = input('Which operation would you like to perform? [l, a, i, d, s or x]: ').lower().strip()
        print()
        return choice

    @staticmethod
    def show_inventory(table):
        """Displays current inventory table


        Args:
            table (list of dict): 2D data structure (list of dicts) that holds the data during runtime.

        Returns:
            None.

        """
        print('======= The Current Inventory: =======')
        print('ID\tCD Title (by: Artist)\n')
        for row in table:
            print('{}\t{} (by:{})'.format(*row.values()))
        print('======================================')


        """Adds CD of user choice to the list, does not save the CD to memory

        Args: None

        Returns:
            strID, asks user for deired value to identity ID
            strTitle, asks user for title of CD associated with ID
            strArtist, asks user for CD artist

        """

    def add_cd():
        while True:
            strID = input('Enter ID: ').strip()
            try:
                intID = int(strID)
                break
            except ValueError as e:
                print('That is not an integer!')
                print(e)
        strTitle = input('Enter the CD\'s title. ').strip()
        stArtist = input('Enter the Artist\'s name. ').strip()
        return [intID, strTitle, stArtist]

# 2. start main loop
while True:
    IO.print_menu()
    strChoice = IO.menu_choice()

    if strChoice == 'x':
        break
    if strChoice == 'l':
        print('WARNING: If you continue, all unsaved data will be lost and the Inventory re-loaded from file.')
        strYesNo = input('type \'yes\' to continue and reload from file. otherwise reload will be canceled')
        if strYesNo.lower() == 'yes':
            print('reloading...')
            FileProcessor.read_file(strFileName, lstTbl)
            IO.show_inventory(lstTbl)
        else:
            input('canceling... Inventory data NOT reloaded. Press [ENTER] to continue to the menu.')
            IO.show_inventory(lstTbl)
        continue
    elif strChoice == 'a':
        DataProcessor.adding_cd()
        IO.show_inventory(lstTbl)
        continue
    elif strChoice == 'i':
        IO.show_inventory(lstTbl)
        continue
    elif strChoice == 'd':
        IO.show_inventory(lstTbl)
        intIDDel = int(input('Which ID would you like to delete? ').strip())
        DataProcessor.delete_cd_from_table(intIDDel)
        IO.show_inventory(lstTbl)
        continue
    elif strChoice == 's':
        IO.show_inventory(lstTbl)
        strYesNo = input('Save this inventory to file? [y/n] ').strip().lower()
        DataProcessor.save_data(lstTbl, strFileName)
        continue

    else:
        print('General Error')
```