Ben Streck

27 Nov 2022

Foundations of Programming (Python)

Assignment 07

Binary Data and Structured Error Handling

Introduction

In this assignment, I outline the process of creating a CD inventory script in python. This script differs from Assignment06.py because it uses structured error handling and stores data in a binary format. The script presents users with a menu that gives them six options: ‘Load Inventory from File’, ‘Add CD’, ‘Display Current Inventory’, ‘Delete CD from Inventory’, ‘Save Inventory to File’ and ‘Exit.’ Each menu option has different functionality and is accompanied by code that appends, deletes, displays, or saves data to the CDInventory.dat file. Completing this assignment requires an understanding of exception handling, binary data, classes, functions, and more. In CDInventory.py, I organized my script into three main sections: Data, Processing, and Presentation (Input/ Output). I used classes to organize my functions into logical groups such as data processing, file processing, and input/output. I used a while loop with conditional statements to display the available options and run the correct functions based on user inputs. Additionally, I used python’s pickle module to read data, add data, and save data to the .DAT binary file. Lastly, I created a GitHub repository for this assignment with the intention of peer-review activities throughout the week.

Exception Handling Research

1. https://www.programiz.com/python-programming/exception-handling[1]
2. https://www.geeksforgeeks.org/python-exception-handling/[2]
3. https://realpython.com/python-exceptions/[3]
4. https://www.w3schools.com/python/gloss_python_error_handling.asp[4]

Data Pickling Research

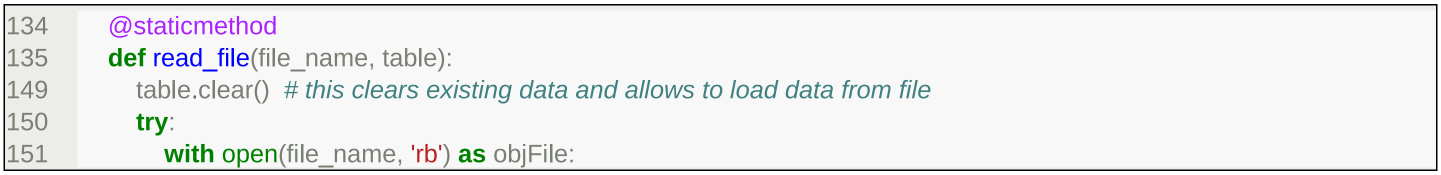
1. https://docs.python.org/3/library/pickle.html[5]
2. https://www.geeksforgeeks.org/understanding-python-pickling-example/[6]
3. https://www.tutorialspoint.com/python-pickling[7]
4. https://pythonprogramming.net/python-pickle-module-save-objects-serialization/[8]

Writing the Script

Structured Error Handling

Reading the .DAT Binary File

Listing 1 shows how I used structured error handling to prevent the program from crashing when it attempts to read data from a file that does not exist yet. The try-except construct allowed me to separate the FileNotFoundError from any other general errors that might present themselves.



```
@staticmethod
def read_file(file_name, table):
    table.clear()  # this clears existing data and allows to load data from file
    try:
        with open(file_name, 'rb') as objFile:
```

[1] Retrieved 20 Nov 2022
[2] Retrieved 20 Nov 2022
[3] Retrieved 22 Nov 2022
[4] Retrieved 22 Nov 2022
[5] Retrieved 26 Nov 2022
[6] Retrieved 26 Nov 2022
[7] Retrieved 26 Nov 2022
[8] Retrieved 26 Nov 2022

```
                dum1 = pickle.load(objFile)
            objFile.close()
            for i in range(len(dum1)):
                table.append(dum1[i])
        except FileNotFoundError as e:
            print('\n{} does not exist...'.format(file_name))
            print('Type: ', type(e), '\nError: ', e, '\nMessage: ', e.__doc__)
            print('\nCreating the File...')
            file = open(file_name, 'wb')
            file.close()
            print('The file, {}, has now been created!'.format(file_name))
        except Exception as e:
            print('\nThere was a general error...')
            print('Type: ', type(e), '\nError: ', e, '\nMessage: ', e.__doc__)
```

Listing 1 – Error Handling for File Access Operations

Type Casting (String → Integer)

Listing 2 shows how I used structured error handling to prevent the program from crashing when it can’t convert the user’s input to an integer. The try-except construct allowed me to separate the ValueError from any other general errors that might present themselves.

```
    @staticmethod
    def del_CD_choice(table):
        print('Deleting an entry from the CD Inventory...')
        print('What is the ID number of the entry you want to delete?\n')
        IO.show_inventory(table)
        try:
            intIDDel = int(input('Enter ID Number Here: ').strip())
            return intIDDel
        except ValueError as e:
            print('\nThat is not a valid ID number...')
            print('Type: ', type(e), '\nError: ', e, '\nMessage: ', e.__doc__)
            print('\nNo Entries Deleted')
        except Exception as e:
            print('\nThere was a general error...')
            print('Type: ', type(e), '\nError: ', e, '\nMessage: ', e.__doc__)
            print('\nNo Entries Deleted')
```

Listing 2 – Error Handling for Type Casting

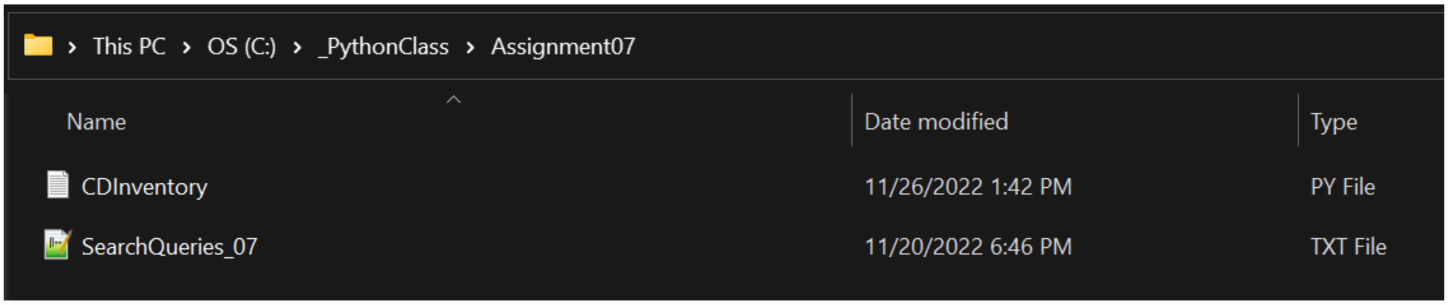
Changing to Binary Data Storage

Listing 3 shows how I used python’s pickle module to read and write binary data to the CDInventory.dat file. I used pickle.load() and pickle.dump() to complete those two tasks.

```
import pickle

class FileProcessor:
    """
    Processing the data to and from .DAT binary file
    """

    @staticmethod
    def read_file(file_name, table):
        """
        Function to manage data intake from the .DAT binary file to a list of dictionaries.
        The function reads data from the file identified by 'file_name' into a 2D table
        (list of dicts). It also includes structured error handling in case the file
        does not exist yet.

        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
        """
        table.clear()  # this clears existing data and allows to load data from file
        try:
            with open(file_name, 'rb') as objFile:
                dum1 = pickle.load(objFile)
            objFile.close()
            for i in range(len(dum1)):
                table.append(dum1[i])
        except FileNotFoundError as e:
            print('\n{} does not exist...'.format(file_name))
            print('Type: ', type(e), '\nError: ', e, '\nMessage: ', e.__doc__)
            print('\nCreating the File...')
            file = open(file_name, 'wb')
            file.close()
            print('The file, {}, has now been created!'.format(file_name))
        except Exception as e:
            print('\nThere was a general error...')
            print('Type: ', type(e), '\nError: ', e, '\nMessage: ', e.__doc__)

    @staticmethod
    def write_file(file_name, table):
        """
        Function to manage data writing from the list of dictionaries to a .DAT binary file.

        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
        """
        with open(file_name, 'wb') as objFile:
            pickle.dump(table, objFile)
        objFile.close()
```
Listing 3 – Using the 'pickle' module for binary data storage

Saving the Script

As instructed, I created a folder in C:\ _PythonClass\ called 'Assignment07' and saved my script as CDInventory.py.

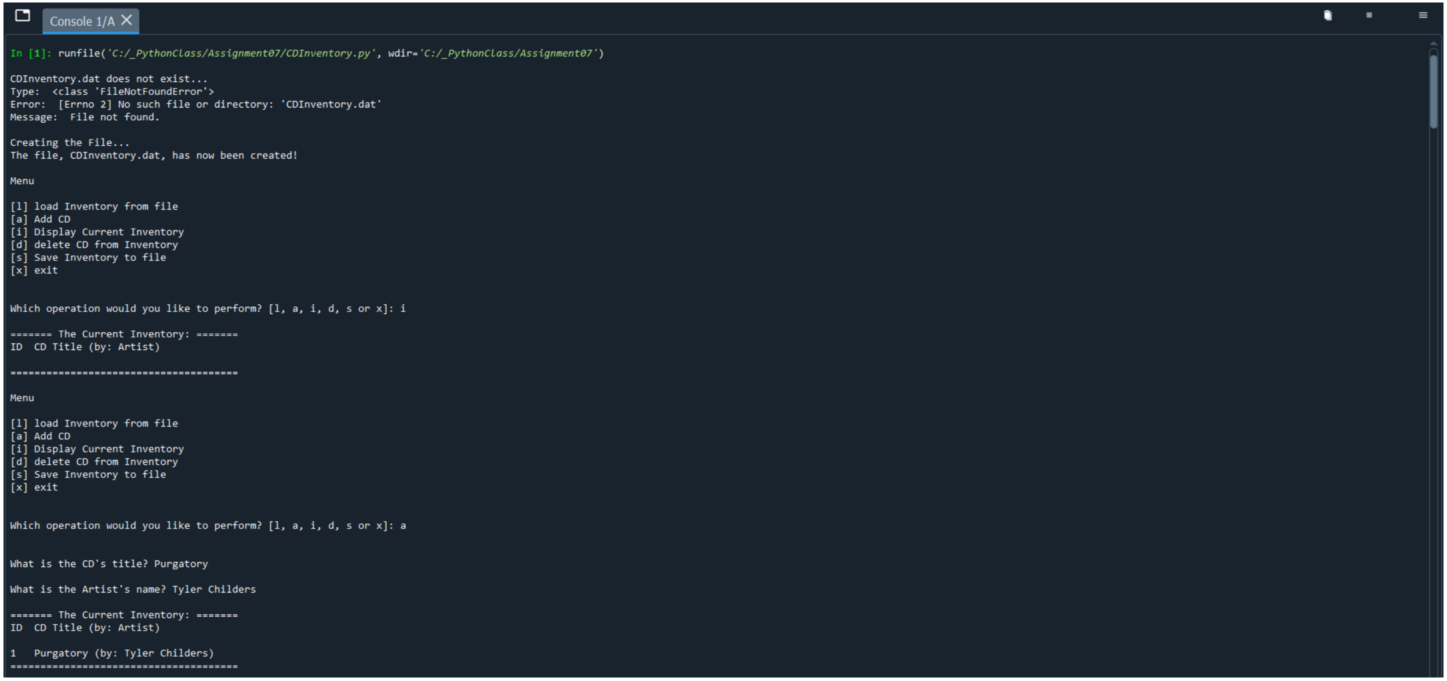


Name	Date modified	Type
CDInventory	11/26/2022 1:42 PM	PY File
SearchQueries_07	11/20/2022 6:46 PM	TXT File

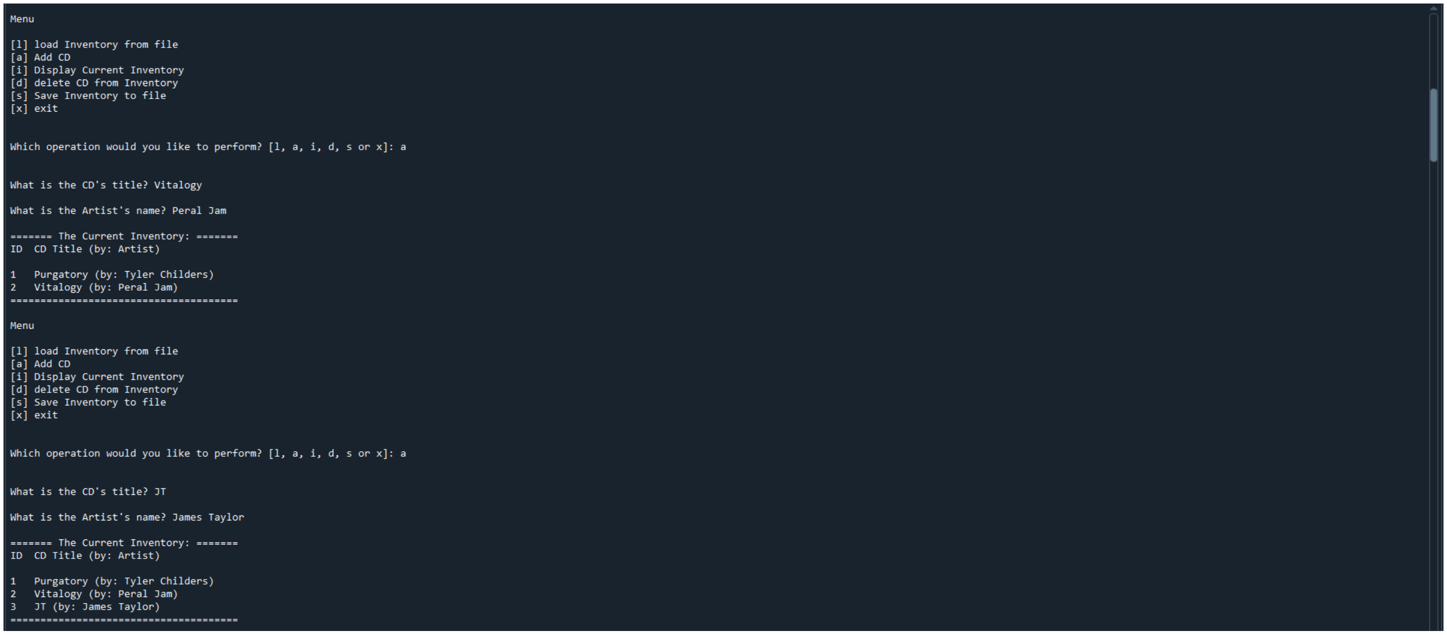
Figure 1 – Saving CDInventory.py

Running the Script

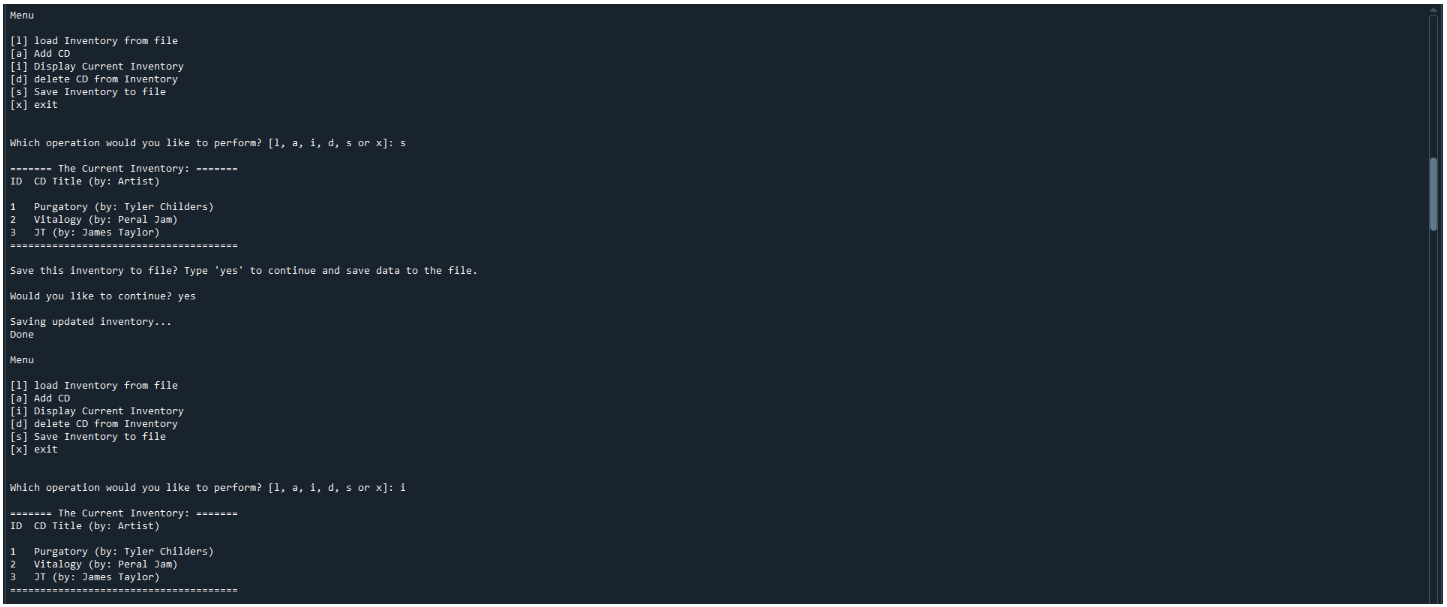
Spyder



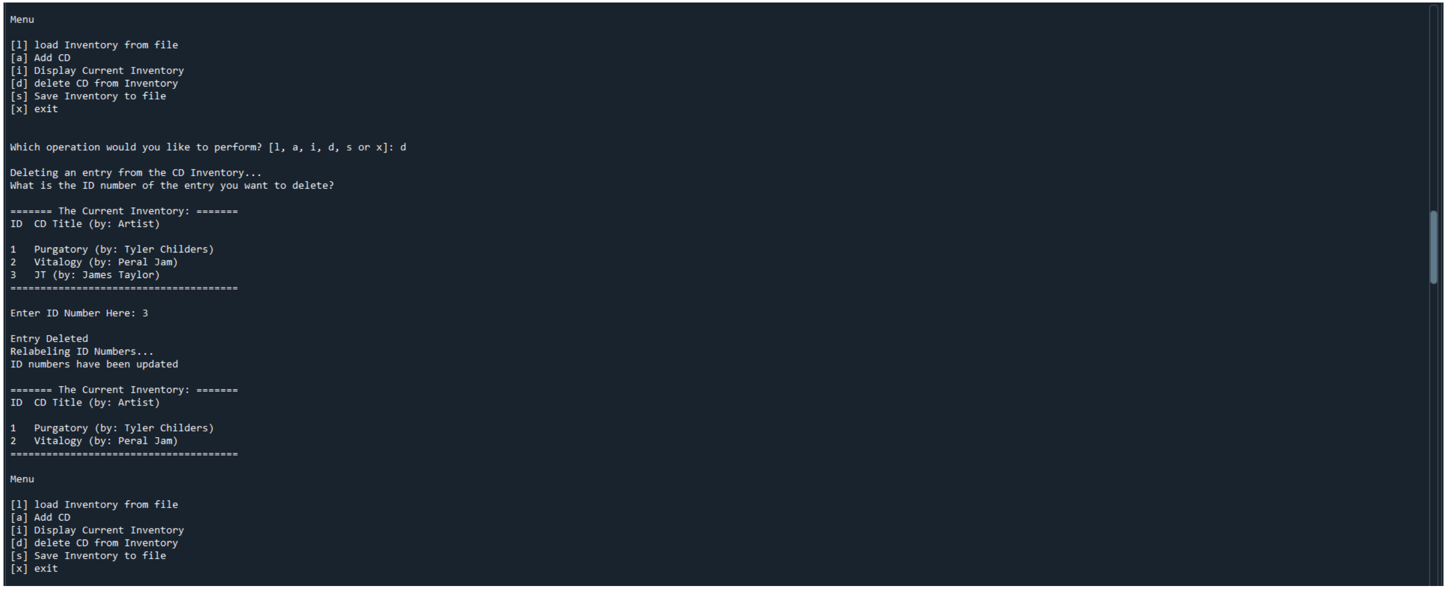
```
In [1]: runfile('C:/_PythonClass/Assignment07/CDInventory.py', wdir='C:/_PythonClass/Assignment07')

CDInventory.dat does not exist...
Type:  <class 'FileNotFoundError'>
Error:  [Errno 2] No such file or directory: 'CDInventory.dat'
Message:  File not found.

Creating the File...
The file, CDInventory.dat, has now 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]: 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


What is the CD's title? Purgatory

What is the Artist's name? Tyler Childers

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

1   Purgatory (by: Tyler Childers)
======================================
```



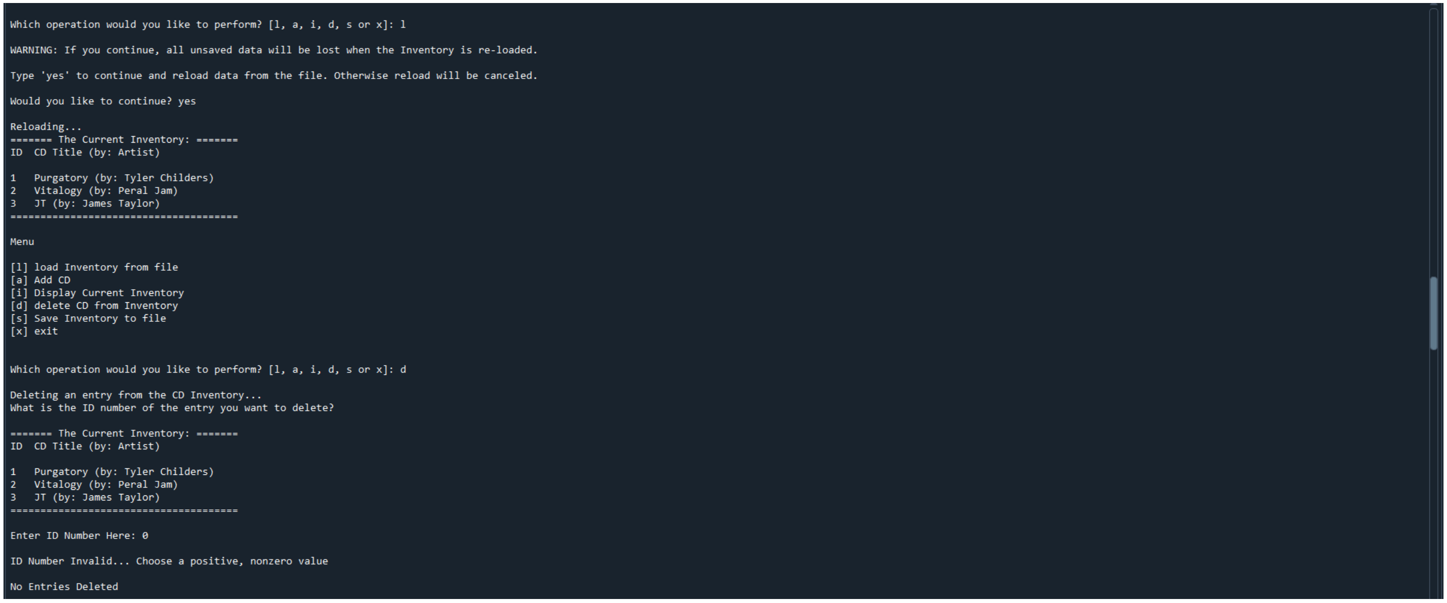
```
Menu

[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


What is the CD's title? Vitalogy

What is the Artist's name? Peral Jam

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

1   Purgatory (by: Tyler Childers)
2   Vitalogy (by: Peral Jam)
======================================

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


What is the CD's title? JT

What is the Artist's name? James Taylor

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

1   Purgatory (by: Tyler Childers)
2   Vitalogy (by: Peral Jam)
3   JT (by: James Taylor)
======================================
```



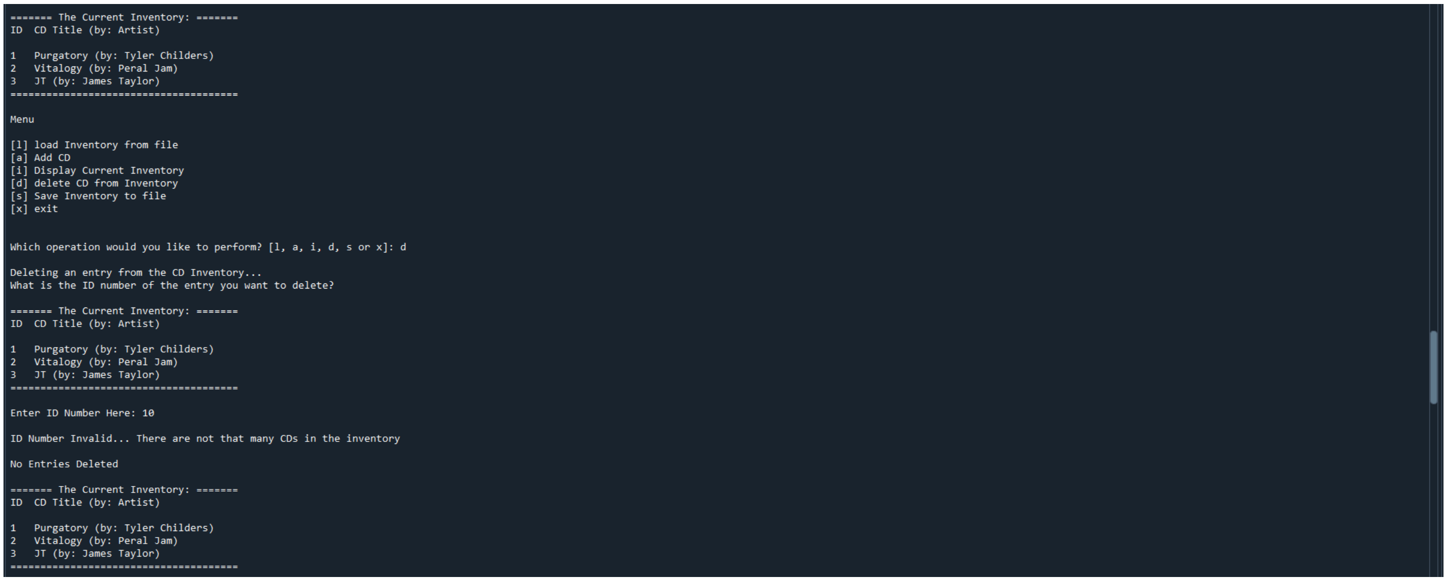
```
Menu

[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   Purgatory (by: Tyler Childers)
2   Vitalogy (by: Peral Jam)
3   JT (by: James Taylor)
======================================

Save this inventory to file? Type 'yes' to continue and save data to the file.

Would you like to continue? yes

Saving updated inventory...
Done

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)

1   Purgatory (by: Tyler Childers)
2   Vitalogy (by: Peral Jam)
3   JT (by: James Taylor)
======================================
```



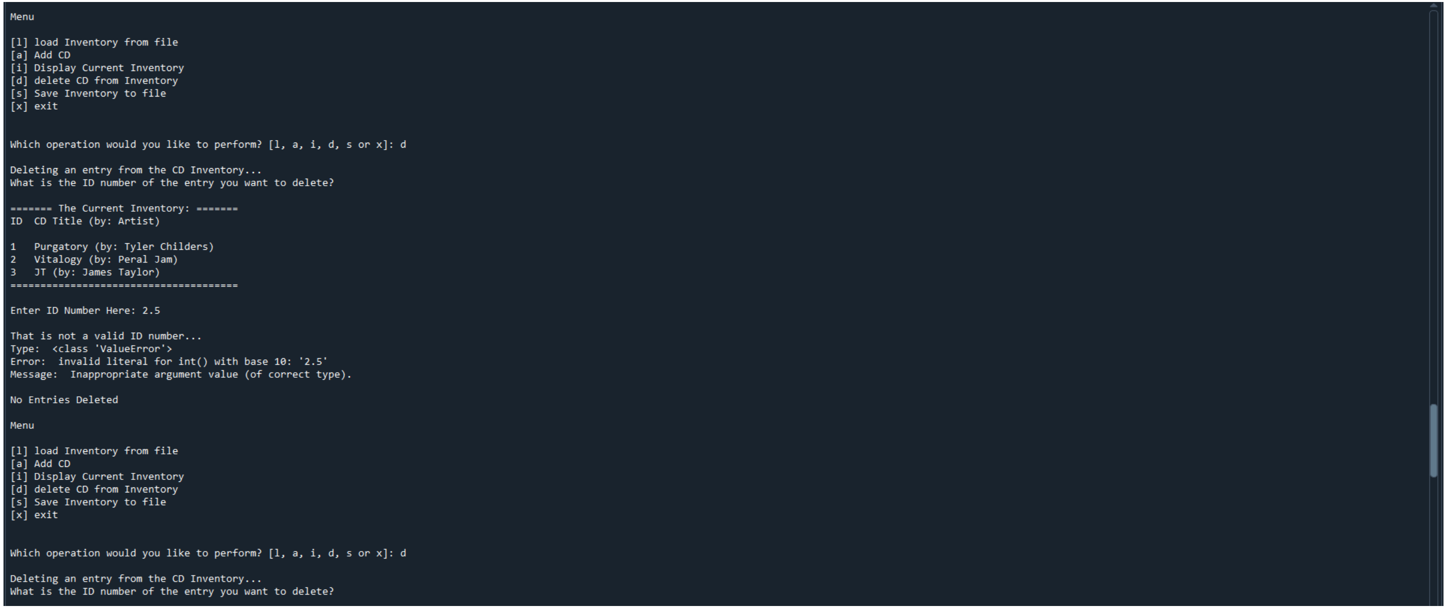
```
Menu

[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

Deleting an entry from the CD Inventory...
What is the ID number of the entry you want to delete?

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

1   Purgatory (by: Tyler Childers)
2   Vitalogy (by: Peral Jam)
3   JT (by: James Taylor)
======================================

Enter ID Number Here: 3

Entry Deleted
Relabeling ID Numbers...
ID numbers have been updated

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

1   Purgatory (by: Tyler Childers)
2   Vitalogy (by: Peral Jam)
======================================

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
```



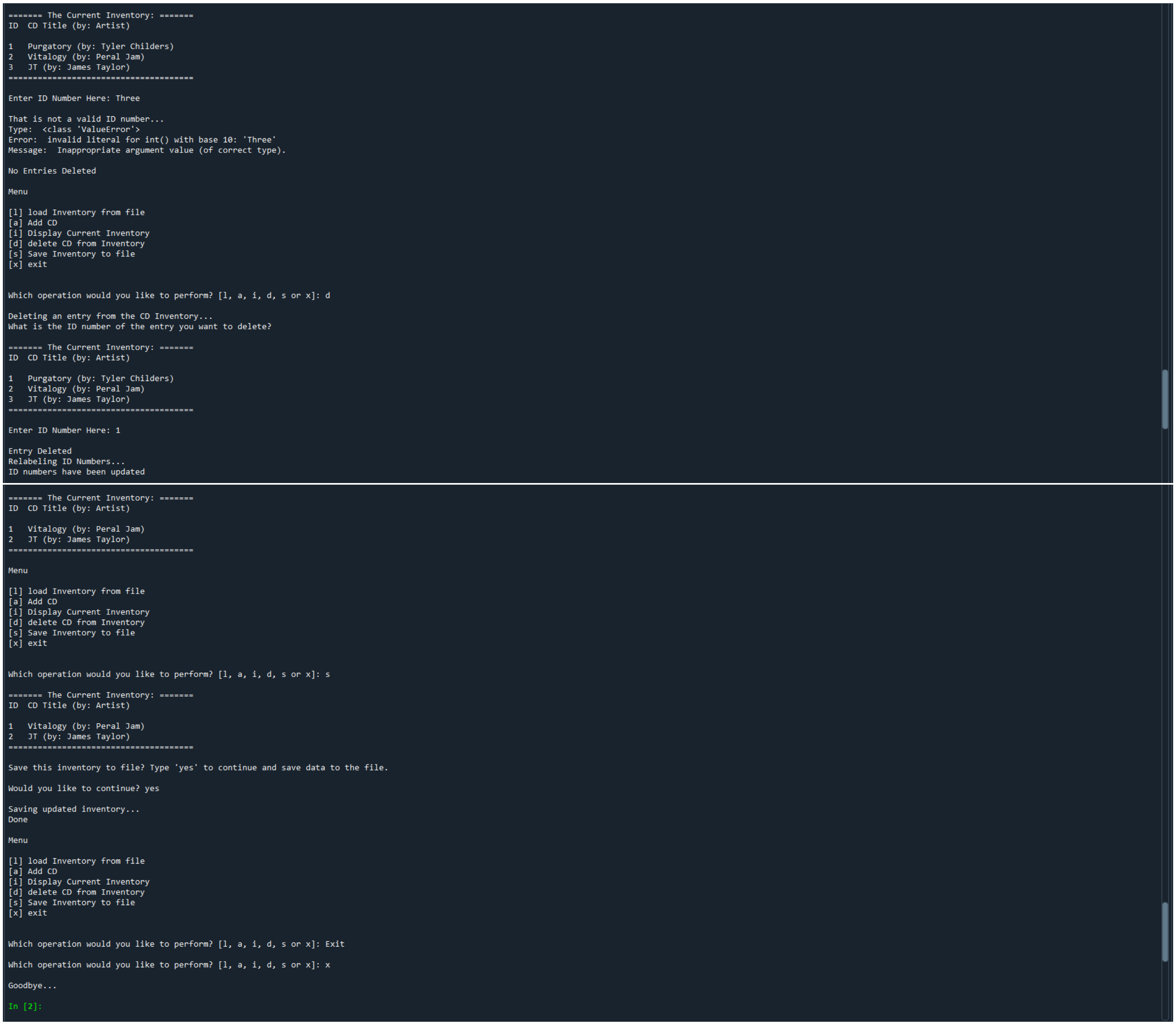
```
Which operation would you like to perform? [l, a, i, d, s or x]: l

WARNING: If you continue, all unsaved data will be lost when the Inventory is re-loaded.

Type 'yes' to continue and reload data from the file. Otherwise reload will be canceled.

Would you like to continue? yes

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

1   Purgatory (by: Tyler Childers)
2   Vitalogy (by: Peral Jam)
3   JT (by: James Taylor)
======================================

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

Deleting an entry from the CD Inventory...
What is the ID number of the entry you want to delete?

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

1   Purgatory (by: Tyler Childers)
2   Vitalogy (by: Peral Jam)
3   JT (by: James Taylor)
======================================

Enter ID Number Here: 0

ID Number Invalid... Choose a positive, nonzero value

No Entries Deleted
```



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

1   Purgatory (by: Tyler Childers)
2   Vitalogy (by: Peral Jam)
3   JT (by: James Taylor)
======================================

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

Deleting an entry from the CD Inventory...
What is the ID number of the entry you want to delete?

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

1   Purgatory (by: Tyler Childers)
2   Vitalogy (by: Peral Jam)
3   JT (by: James Taylor)
======================================

Enter ID Number Here: 10

ID Number Invalid... There are not that many CDs in the inventory

No Entries Deleted

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

1   Purgatory (by: Tyler Childers)
2   Vitalogy (by: Peral Jam)
3   JT (by: James Taylor)
======================================
```



```
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

Deleting an entry from the CD Inventory...
What is the ID number of the entry you want to delete?

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

1   Purgatory (by: Tyler Childers)
2   Vitalogy (by: Peral Jam)
3   JT (by: James Taylor)
======================================

Enter ID Number Here: 2.5

That is not a valid ID number...
Type:  <class 'ValueError'>
Error:  invalid literal for int() with base 10: '2.5'
Message:  Inappropriate argument value (of correct type).

No Entries Deleted

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

Deleting an entry from the CD Inventory...
What is the ID number of the entry you want to delete?
```



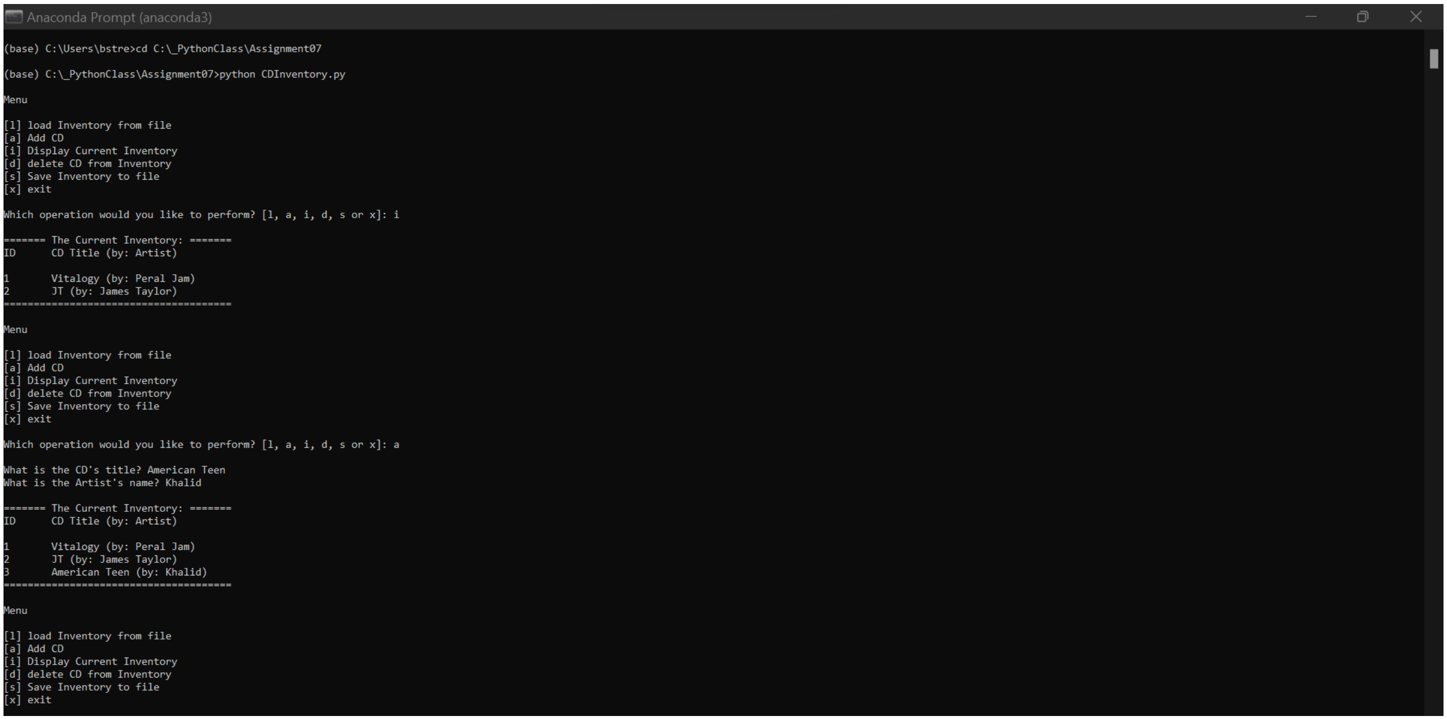
```
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Purgatory (by: Tyler Childers)
2   Vitalogy (by: Peral Jam)
3   JT (by: James Taylor)
======================================

Enter ID Number Here: Three

That is not a valid ID number...
Type:  <class 'ValueError'>
Error:  invalid literal for int() with base 10: 'Three'
Message:  Inappropriate argument value (of correct type).

No Entries Deleted

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

Deleting an entry from the CD Inventory...
What is the ID number of the entry you want to delete?

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

1   Purgatory (by: Tyler Childers)
2   Vitalogy (by: Peral Jam)
3   JT (by: James Taylor)
======================================

Enter ID Number Here: 1

Entry Deleted
Relabeling ID Numbers...
ID numbers have been updated
```

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

1   Vitalogy (by: Peral Jam)
2   JT (by: James Taylor)
======================================

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   Vitalogy (by: Peral Jam)
2   JT (by: James Taylor)
======================================

Save this inventory to file? Type 'yes' to continue and save data to the file.

Would you like to continue? yes

Saving updated inventory...
Done

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]: Exit

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

Goodbye...

In [2]:
```

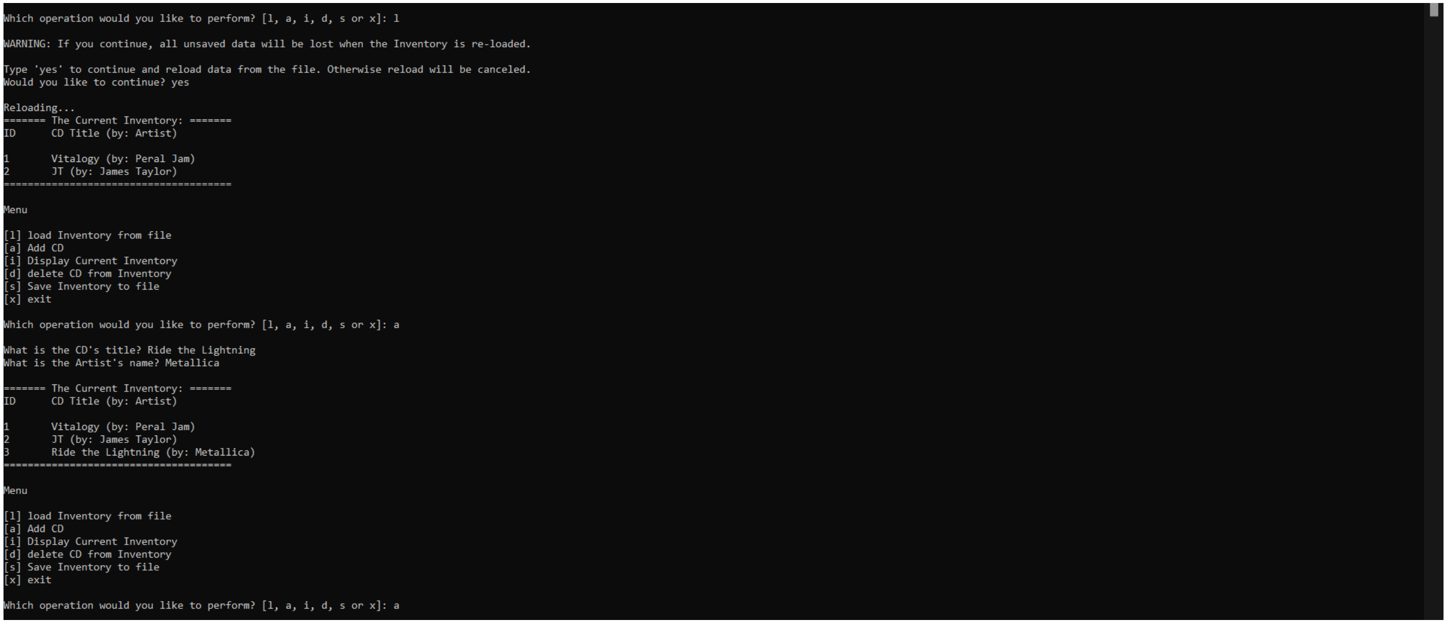
Figure 2 – Running CDInventory.py in Spyder

I opened Spyder on my Windows computer, opened CDInventory.py, and clicked F5 to execute the script. The file, CDInventory.dat did not exist yet and the program behaved correctly. It displayed the FileNotFoundError and proceeded to create the file for use later in the script. I followed the user prompts and entered 'i' to display the current CD inventory. It was empty and the program behaved correctly. Next, I used 'a' three times to add three CDs and entered the necessary information when prompted. I followed that up with the 's' command and an 'i' command to save the updated inventory and then display it to the user. Afterwards, I used 'd' to delete the third CD from the inventory. I used 'l' next to load the last save from CDInventory.txt. This undid my deletion of the third CD. Then, I showed that the delete option displays useful information when the user enters an ID number outside of the data range. I also showed that the delete option handles type casting errors correctly by inputting a float and a string for the ID number. Then, I deleted CD number one and saved my changes. Finally, I entered 'Exit' to show the script's response to an invalid input followed by 'x' to exit the program. CDInventory.py ran correctly all the way through. It accepted user inputs, read data, modified data, displayed data, and saved data as intended. Figure 2 shows that the script functions correctly while running in the Spyder IDE.

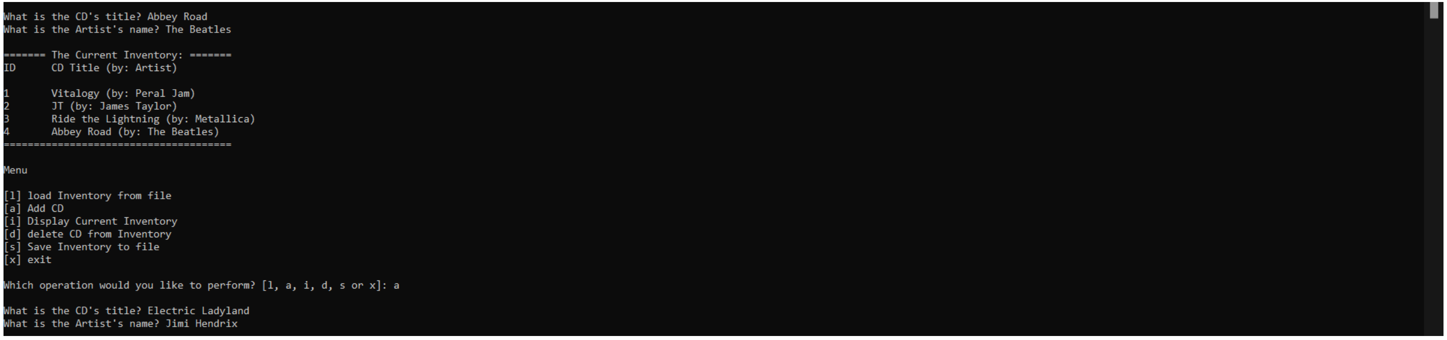
Terminal



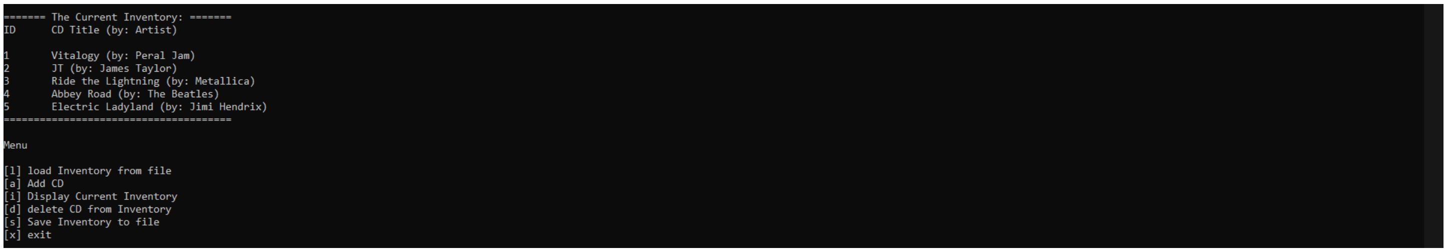
```
(base) C:\Users\bstre>cd C:\_PythonClass\Assignment07

(base) C:\_PythonClass\Assignment07>python CDInventory.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)

1       Vitalogy (by: Peral Jam)
2       JT (by: James Taylor)
======================================

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

What is the CD's title? American Teen
What is the Artist's name? Khalid

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

1       Vitalogy (by: Peral Jam)
2       JT (by: James Taylor)
3       American Teen (by: Khalid)
======================================

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
```



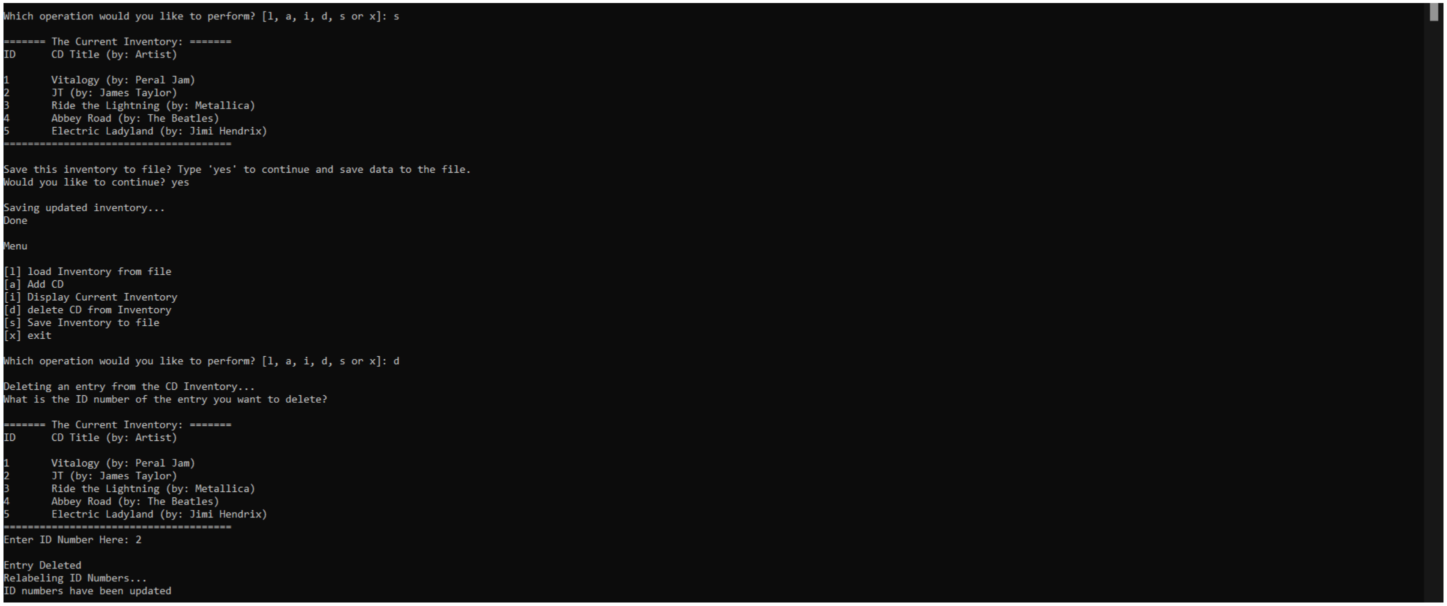
```
Which operation would you like to perform? [l, a, i, d, s or x]: l

WARNING: If you continue, all unsaved data will be lost when the Inventory is re-loaded.

Type 'yes' to continue and reload data from the file. Otherwise reload will be canceled.
Would you like to continue? yes

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

1       Vitalogy (by: Peral Jam)
2       JT (by: James Taylor)
======================================

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

What is the CD's title? Ride the Lightning
What is the Artist's name? Metallica

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

1       Vitalogy (by: Peral Jam)
2       JT (by: James Taylor)
3       Ride the Lightning (by: Metallica)
======================================

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
```



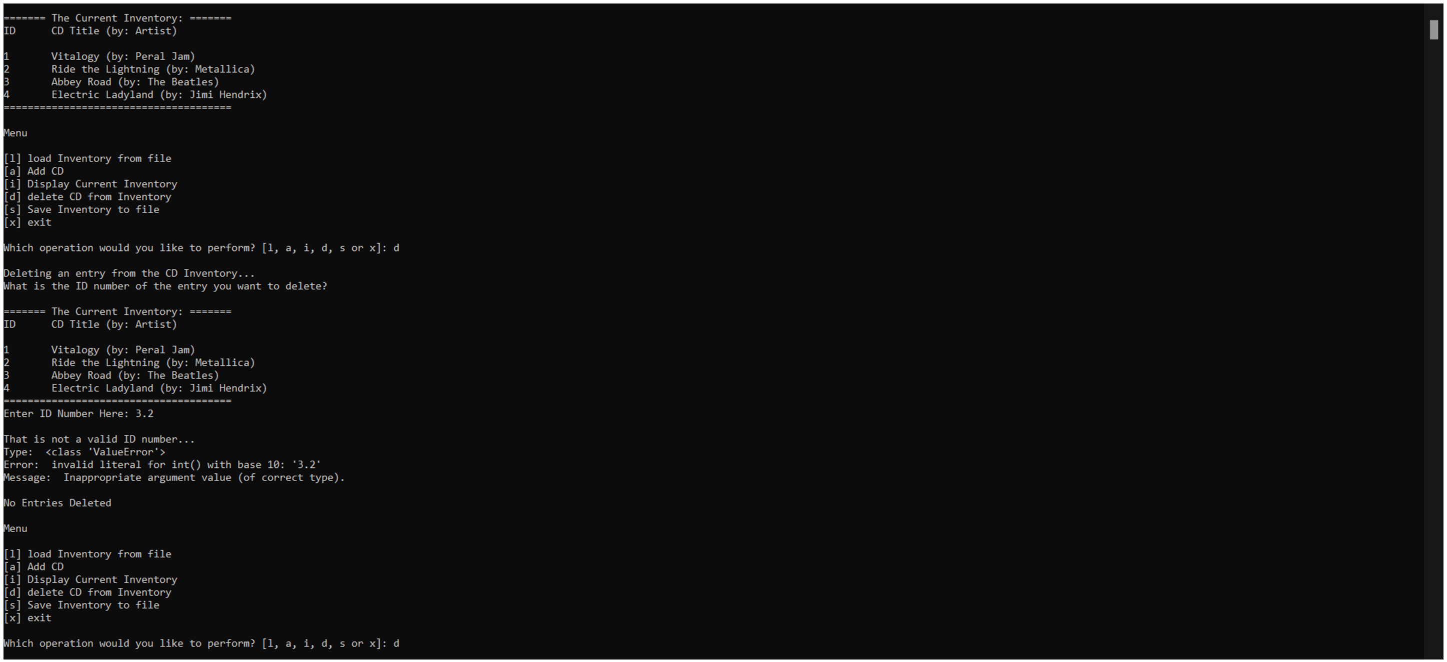
```
What is the CD's title? Abbey Road
What is the Artist's name? The Beatles

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

1       Vitalogy (by: Peral Jam)
2       JT (by: James Taylor)
3       Ride the Lightning (by: Metallica)
4       Abbey Road (by: The Beatles)
======================================

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

What is the CD's title? Electric Ladyland
What is the Artist's name? Jimi Hendrix
```



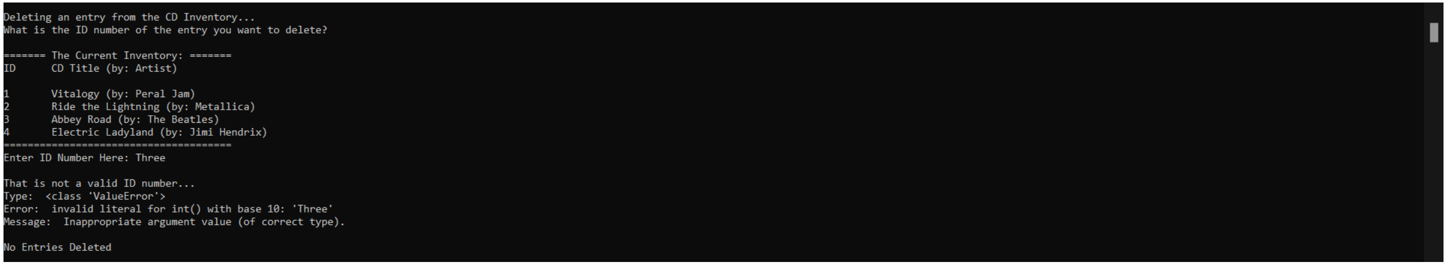
```
======= The Current Inventory: =======
ID      CD Title (by: Artist)

1       Vitalogy (by: Peral Jam)
2       JT (by: James Taylor)
3       Ride the Lightning (by: Metallica)
4       Abbey Road (by: The Beatles)
5       Electric Ladyland (by: Jimi Hendrix)
======================================

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
```



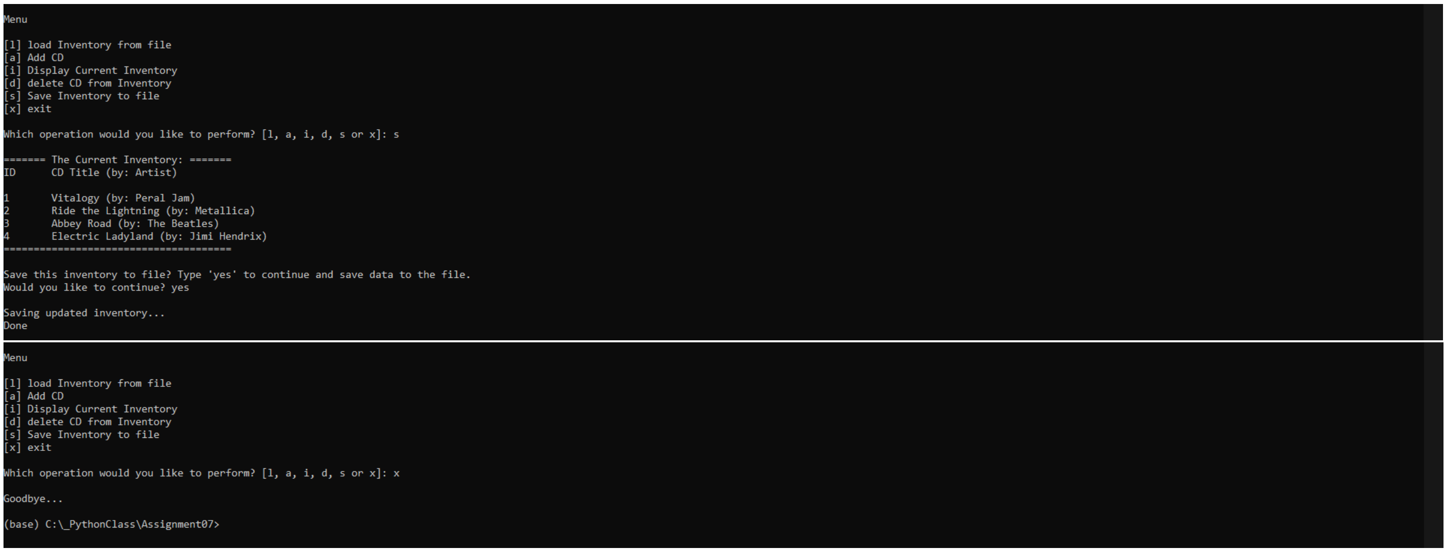
```
Which operation would you like to perform? [l, a, i, d, s or x]: s

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

1       Vitalogy (by: Peral Jam)
2       JT (by: James Taylor)
3       Ride the Lightning (by: Metallica)
4       Abbey Road (by: The Beatles)
5       Electric Ladyland (by: Jimi Hendrix)
======================================

Save this inventory to file? Type 'yes' to continue and save data to the file.
Would you like to continue? yes

Saving updated inventory...
Done

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

Deleting an entry from the CD Inventory...
What is the ID number of the entry you want to delete?

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

1       Vitalogy (by: Peral Jam)
2       JT (by: James Taylor)
3       Ride the Lightning (by: Metallica)
4       Abbey Road (by: The Beatles)
5       Electric Ladyland (by: Jimi Hendrix)
======================================
Enter ID Number Here: 2

Entry Deleted
Relabeling ID Numbers...
ID numbers have been updated
```



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

1       Vitalogy (by: Peral Jam)
2       Ride the Lightning (by: Metallica)
3       Abbey Road (by: The Beatles)
4       Electric Ladyland (by: Jimi Hendrix)
======================================

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

Deleting an entry from the CD Inventory...
What is the ID number of the entry you want to delete?

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

1       Vitalogy (by: Peral Jam)
2       Ride the Lightning (by: Metallica)
3       Abbey Road (by: The Beatles)
4       Electric Ladyland (by: Jimi Hendrix)
======================================
Enter ID Number Here: 3.2

That is not a valid ID number...
Type:  <class 'ValueError'>
Error:  invalid literal for int() with base 10: '3.2'
Message:  Inappropriate argument value (of correct type).

No Entries Deleted

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
```



```
Deleting an entry from the CD Inventory...
What is the ID number of the entry you want to delete?

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

1       Vitalogy (by: Peral Jam)
2       Ride the Lightning (by: Metallica)
3       Abbey Road (by: The Beatles)
4       Electric Ladyland (by: Jimi Hendrix)
======================================
Enter ID Number Here: Three

That is not a valid ID number...
Type:  <class 'ValueError'>
Error:  invalid literal for int() with base 10: 'Three'
Message:  Inappropriate argument value (of correct type).

No Entries Deleted
```



```
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       Vitalogy (by: Peral Jam)
2       Ride the Lightning (by: Metallica)
3       Abbey Road (by: The Beatles)
4       Electric Ladyland (by: Jimi Hendrix)
======================================

Save this inventory to file? Type 'yes' to continue and save data to the file.
Would you like to continue? yes

Saving updated inventory...
Done
```

```
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

Goodbye...

(base) C:\_PythonClass\Assignment07>
```

Figure 3 – Running CDInventory.py in Terminal

I opened terminal on my Windows computer and navigated to the correct folder using the cd (change directory) command. Then, I ran the command ‘python CDInventory.py’ to execute the script. I followed the user prompts and entered ‘i’ to show that the program loaded the data correctly. I ran the script on terminal after I had already run it on Spyder, so there were two CDs in the inventory. Next, I used ‘a’ to add a CD and entered the necessary information when prompted. I followed that up with the ‘l’ command to reload the last save from CDInventory.txt. I used ‘a’ three more times to add three new CDs and then used ‘s’ to save my changes. Afterwards, I used ‘d’ to delete the second CD from the inventory followed by ‘d’ two more times to show that the delete option handles type casting errors correctly. Finally, I used ‘s’ to save and ‘x’ to exit. CDInventory.py ran correctly all the way through. It accepted user inputs, read data, modified data, displayed data, and saved data as intended. Figure 3 shows that the script functions correctly while running in terminal.

Checking the .DAT Binary File

The binary data format is intended to be a more efficient method of storing data. For example, storing values using numeric formats instead of text characters will often use less memory. The binary format also has advantages in terms of speed of access. As a result, the .DAT file is not easily readable. However, it is clear that the data was created, read, edited, and saved properly. This is true because the script functioned correctly in the Spyder IDE and in the Anaconda Terminal above.

GitHub Repository

Link: https://github.com/BenStreck/Assignment_07

Summary

I successfully created a python script that fulfills the requirements listed in Assignment 07. I did so using information from the textbook, the Module 07 videos, and my own research outside of class. The script demonstrates my understanding of classes, functions, error handling, and binary data storage.

It wasn’t immediately clear how to “verify that the data is being written to the file correctly.” In previous assignments, I have been able to screenshot the .TXT file because it was easily readable. The .DAT file was not as clear. Other than that, no significant struggles presented themselves with this assignment.

Appendix

Full Listing – CDInventory.py

```
#------------------------------------------#
# Title: CDInventory.py
# Desc: This is a script to store CD Inventory Data
#       This script demonstrates my understanding of how to use structured
#       error handling. It also demonstrates my ability to work with
#       binary data
# Change Log: (Who, When, What)
# DBiesinger, 2030-Jan-01, Created File
# BStreck, 2022-Nov-16, Started adding functionality in the 'TO-DO' sections (Assignment06)
# BStreck, 2022-Nov-19, Finished adding functionality in the 'TO-DO' sections (Assignment06)
# BStreck, 2022-Nov-26, Added structured error handling and changed data strage to binary data (Assignment07)
#------------------------------------------#

import pickle

# -- DATA -- #
strChoice = '' # User input
lstTbl = []  # list of lists to hold data
dicRow = {}  # list of data row
strFileName = 'CDInventory.dat'  # data storage file
objFile = None  # file object


# -- PROCESSING -- #
class DataProcessor:
    """
    Processing the data during runtime
    """

    @staticmethod
    def add_CD(table, ID, strTitle, strArtist):
        """
        Function to add a new CD to the current inventory and show the updated inventory afterwards.

        Args:
            table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
            ID (integer): the ID number of the new CD being added to the inventory
            strTitle (string): the title of the new CD being added to the inventory
            strArtist (string): the artist of the new CD being added to the inventory

        Returns:
            None
        """
        dicRow = {'ID': ID, 'Title': strTitle, 'Artist': strArtist}
        table.append(dicRow)
        print()
        IO.show_inventory(table)

    @staticmethod
    def delete_CD(table, intIDDel):
        """
        Function to delete a CD from the current inventory and show the updated inventory afterwards.
        It also relabels the ID numbers to prevent discontinuities in the inventory.

        Args:
            table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
            intIDDel (integer): the ID number of the CD being deleted from the inventory

        Returns:
            table (list of dict): updated 2D data structure (list of dicts) that holds the data during runtime
        """
        if intIDDel < 1:
            print('\nID Number Invalid... Choose a positive, nonzero value\n')
            print('No Entries Deleted\n')
        elif intIDDel > len(table):
            print('\nID Number Invalid... There are not that many CDs in the inventory\n')
            print('No Entries Deleted\n')
        else:
            table = list(filter(lambda i: i['ID'] != intIDDel, table))
            print('\nEntry Deleted')
            print('Relabeling ID Numbers...')
            i = 1
            for row in table:
                row['ID'] = i
                i += 1
            print('ID numbers have been updated\n')
        IO.show_inventory(table)
        return table

    @staticmethod
    def load_inventory(file_name, table):
        """
        Function managing the FileProcessor.read_file() function.
        This helps prevent unintentional overwriting of data in the current inventory.
        It also shows the current inventory after it has been loaded.

        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
        """
        print('WARNING: If you continue, all unsaved data will be lost when the Inventory is re-loaded.\n')
        print('Type \'yes\' to continue and reload data from the file. Otherwise reload will be canceled.')
        strYesNo = input('Would you like to continue? ')
        if strYesNo.strip().lower() == 'yes':
            print('\nReloading...')
            FileProcessor.read_file(file_name, table)
            IO.show_inventory(table)
        else:
            input('Canceling... Inventory data NOT reloaded. Press [ENTER] to continue to the menu. ')
            IO.show_inventory(table)

    @staticmethod
    def save_inventory(file_name, table):
        """
        Function managing the FileProcessor.write_file() function.
        It shows the current inventory prior to saving which allows users to verify they are saving the correct data.

        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:
```

```python
            None
        """
        IO.show_inventory(table)
        print('\nSave this inventory to file? Type \'yes\' to continue and save data to the file.')
        strYesNo = input('Would you like to continue? ')
        if strYesNo.strip().lower() == 'yes':
            print('\nSaving updated inventory...')
            FileProcessor.write_file(file_name, table)
            print('Done')
        else:
            input('The inventory was NOT saved to file. Press [ENTER] to return to the menu. ')


class FileProcessor:
    """
    Processing the data to and from .DAT binary file
    """

    @staticmethod
    def read_file(file_name, table):
        """
        Function to manage data intake from the .DAT binary file to a list of dictionaries.
        The function reads data from the file identified by 'file_name' into a 2D table
        (list of dicts). It also includes structured error handling in case the file
        does not exist yet.

        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
        """
        table.clear()  # this clears existing data and allows to load data from file
        try:
            with open(file_name, 'rb') as objFile:
                dum1 = pickle.load(objFile)
            objFile.close()
            for i in range(len(dum1)):
                table.append(dum1[i])
        except FileNotFoundError as e:
            print('\n{} does not exist...'.format(file_name))
            print('Type: ', type(e), '\nError: ', e, '\nMessage: ', e.__doc__)
            print('\nCreating the File...')
            file = open(file_name, 'wb')
            file.close()
            print('The file, {}, has now been created!'.format(file_name))
        except Exception as e:
            print('\nThere was a general error...')
            print('Type: ', type(e), '\nError: ', e, '\nMessage: ', e.__doc__)

    @staticmethod
    def write_file(file_name, table):
        """
        Function to manage data writing from the list of dictionaries to a .DAT binary file.

        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
        """
        with open(file_name, 'wb') as objFile:
            pickle.dump(table, objFile)
        objFile.close()


# -- 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('\nMenu\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 string 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()  # Add extra space for layout
        return choice

    @staticmethod
    def show_inventory(table):
        """
        Displays the 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('======================================')

    @staticmethod
    def new_CD_choice(table):
        """
        Function to accept user inputs for a new CD.
        The data will be added to the current inventory using the DataProcessor.add_CD() function.

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

        Returns:
            ID (integer): the ID number of the new CD being added to the inventory
            strTitle (string): the title of the new CD being added to the inventory
            strArtist (string): the artist of the new CD being added to the inventory
        """
        ID = len(table) + 1
        strTitle = input('What is the CD\'s title? ').strip()
        strArtist = input('What is the Artist\'s name? ').strip()
        return ID, strTitle, strArtist

    @staticmethod
    def del_CD_choice(table):
        """
        Function to accept user inputs for deleting a CD.
        The chosen CD will be removed from the current inventory using the DataProcessor.delete_CD() function.
        It also includes structured error handling in case the input cannot be converted to an integer.

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

        Returns:
            intIDDel (integer): the ID number of the CD being deleted from the inventory
        """
        print('Deleting an entry from the CD Inventory...')
        print('What is the ID number of the entry you want to delete?\n')
        IO.show_inventory(table)
        try:
            intIDDel = int(input('Enter ID Number Here: ').strip())
            return intIDDel
        except ValueError as e:
            print('\nThat is not a valid ID number...')
            print('Type: ', type(e), '\nError: ', e, '\nMessage: ', e.__doc__)
            print('\nNo Entries Deleted')
        except Exception as e:
            print('\nThere was a general error...')
            print('Type: ', type(e), '\nError: ', e, '\nMessage: ', e.__doc__)
            print('\nNo Entries Deleted')


# 1. When program starts, read in the Current Inventory
FileProcessor.read_file(strFileName, lstTbl)

# 2. Start main loop
while True:

    # 3. Display menu to user and get choice
    IO.print_menu()
    strChoice = IO.menu_choice()

    # 4. Process menu selections

        # 4.1 Exit
    if strChoice == 'x':
        print('Goodbye...')
        break

        # 4.2 Load Inventory
    if strChoice == 'l':
        DataProcessor.load_inventory(strFileName, lstTbl)
        continue

        # 4.3 Add a CD
    elif strChoice == 'a':
        ID, strTitle, strArtist = IO.new_CD_choice(lstTbl)
        DataProcessor.add_CD(lstTbl, ID, strTitle, strArtist)
        continue  # start loop back at top.

        # 4.4 Display Current Inventory
    elif strChoice == 'i':
        IO.show_inventory(lstTbl)
        continue  # start loop back at top.

        # 4.5 Delete a CD
    elif strChoice == 'd':
        try:
            intIDDel = IO.del_CD_choice(lstTbl)
            lstTbl = DataProcessor.delete_CD(lstTbl, intIDDel)
            continue # start loop back at top
        except:
            continue  # start loop back at top.

        # 4.6 Save Inventory to File
    elif strChoice == 's':
        DataProcessor.save_inventory(strFileName, lstTbl)
        continue  # start loop back at top.

        # 4.7 Catch-All Error... Should not be possible because the user's choice gets vetted in IO
    else:
        print('Invalid Input...\n')
        print('Please choose one of the options listed\n')
```