Murat Yokus
August 15th, 2021
IT FDN 110 B: Introduction to Programming (Python)
Assignment 06

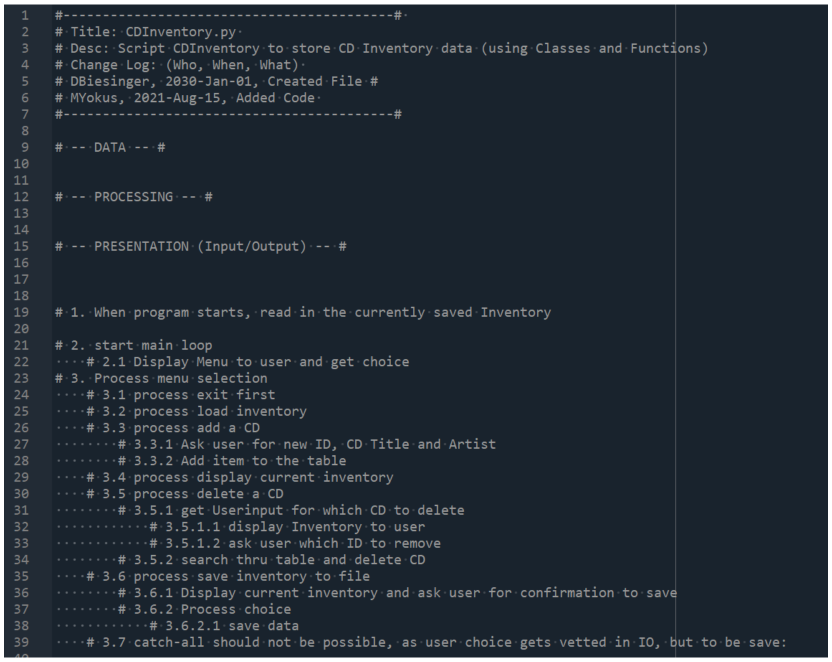
CDInventory.py Script with Functions & Classes

Introduction

This document outlines the steps required for generating a script that has a menu structure and allows user to load CD data from CDInventory.txt file, enter CD data, view the current inventory, delete DC data from inventory, save data to CDInventory.txt data file, and exit the program. The script is a continuation of Assignment 05 and includes classes and functions to simplify and organize the parts of the previous assignment. The script was written in the Spyder IDE, and its successful operation was shown in Spyder and anaconda terminal. Finally, the document summarizes my learnings from Module 6.

Steps:

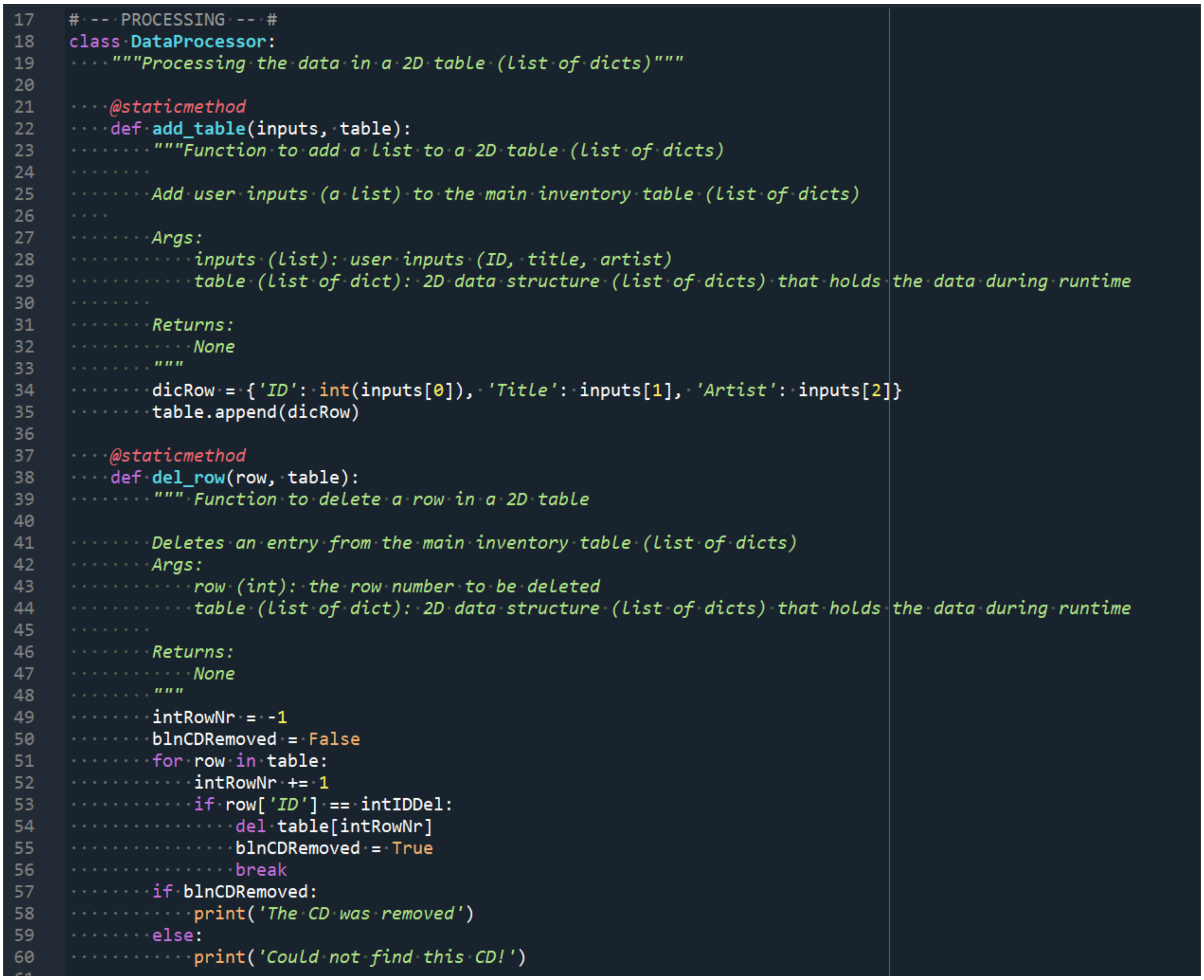
As shown in **Listing 1**, the script follows the common "separation of concerns" approach, dividing the main script into three distinct sections: Data, Processing, and Presentation (input – output). Under the Data section, the variables (data types: integer, string, list, and dictionary) were defined and the name of the data storage file was given. The processing section includes two main classes for data and file processing (defined as DataProcessor and FileProcessor).



```
#-----------------------------------------#
# Title: CDInventory.py
# Desc: Script CDInventory to store CD Inventory data (using Classes and Functions)
# Change Log: (Who, When, What)
# DBiesinger, 2030-Jan-01, Created File #
# MYokus, 2021-Aug-15, Added Code
#-----------------------------------------#

# -- DATA -- #


# -- PROCESSING -- #


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



# 1. When program starts, read in the currently saved Inventory

# 2. start main loop
    # 2.1 Display Menu to user and get choice
# 3. Process menu selection
    # 3.1 process exit first
    # 3.2 process load inventory
    # 3.3 process add a CD
        # 3.3.1 Ask user for new ID, CD Title and Artist
        # 3.3.2 Add item to the table
    # 3.4 process display current inventory
    # 3.5 process delete a CD
        # 3.5.1 get Userinput for which CD to delete
            # 3.5.1.1 display Inventory to user
            # 3.5.1.2 ask user which ID to remove
        # 3.5.2 search thru table and delete CD
    # 3.6 process save inventory to file
        # 3.6.1 Display current inventory and ask user for confirmation to save
        # 3.6.2 Process choice
            # 3.6.2.1 save data
    # 3.7 catch-all should not be possible, as user choice gets vetted in IO, but to be save:
```

Listing 1- Header and layout of the steps in the pseudocode

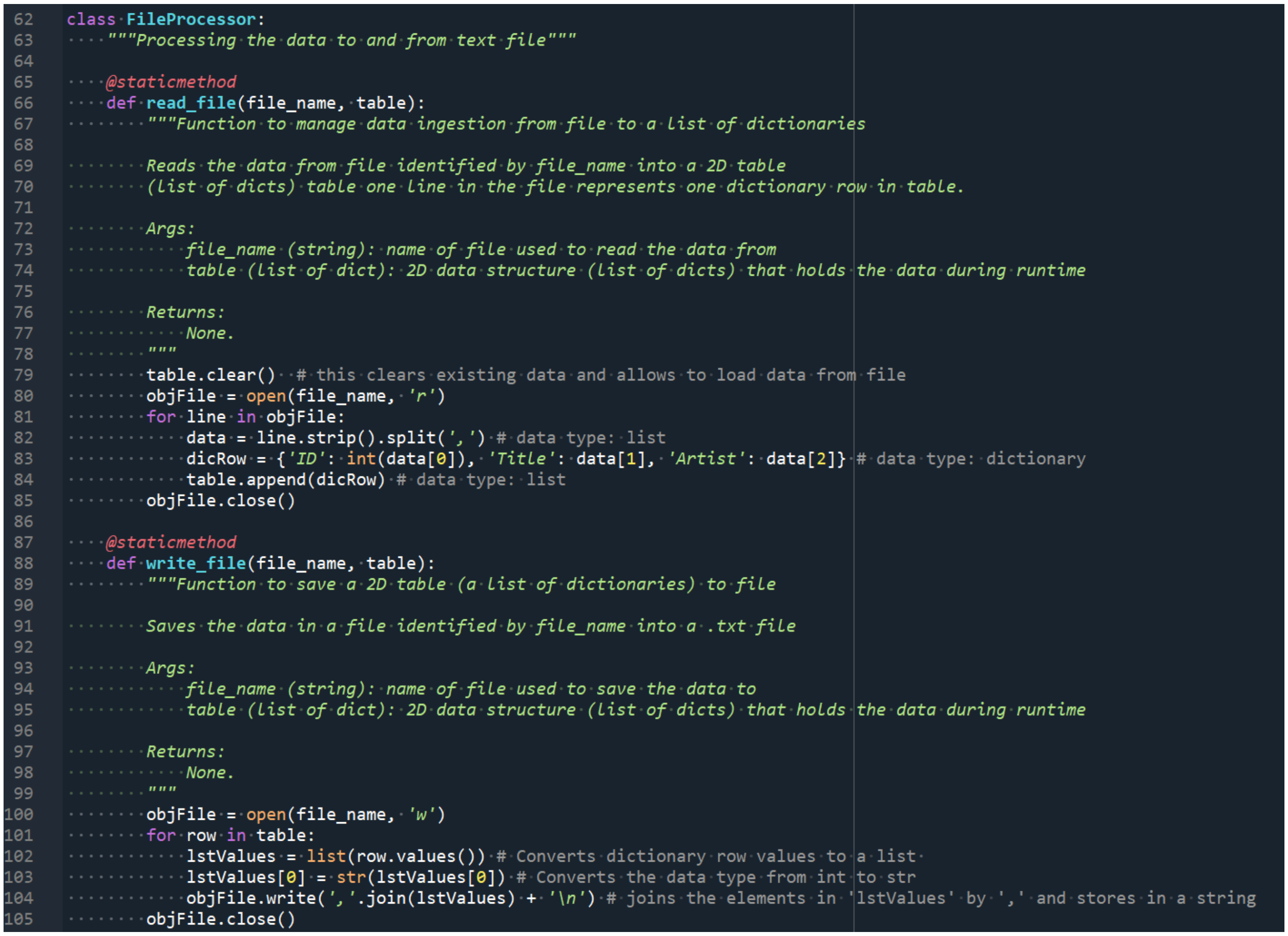
The DataProcessor class of the processing section of the script includes two functions for adding user's input to the main CD inventory and deleting a CD entry from the inventory (**Listing 2**). I created two functions under the DataProcessor class: ***(1)*** add_table() and ***(2)*** *del_row()*. The *add_table()* function takes the user input (a dictionary of CD ID, Title, and Artist) and appends it to a list of dictionaries (a 2D list of main CD inventory). Similarly, the *del_row()* function deletes a CD from the main inventory based on the ID number inputted by the user and prints "The CD was removed" on the screen if the ID number given by the user is found in the main CD inventory.



```python
# -- PROCESSING -- #
class DataProcessor:
    """Processing the data in a 2D table (list of dicts)"""

    @staticmethod
    def add_table(inputs, table):
        """Function to add a list to a 2D table (list of dicts)
        
        Add user inputs (a list) to the main inventory table (list of dicts)
    
        Args:
            inputs (list): user inputs (ID, title, artist)
            table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
        
        Returns:
            None
        """
        dicRow = {'ID': int(inputs[0]), 'Title': inputs[1], 'Artist': inputs[2]}
        table.append(dicRow)

    @staticmethod
    def del_row(row, table):
        """ Function to delete a row in a 2D table

        Deletes an entry from the main inventory table (list of dicts)
        Args:
            row (int): the row number to be deleted
            table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
        
        Returns:
            None
        """
        intRowNr = -1
        blnCDRemoved = False
        for row in table:
            intRowNr += 1
            if row['ID'] == intIDDel:
                del table[intRowNr]
                blnCDRemoved = True
                break
        if blnCDRemoved:
            print('The CD was removed')
        else:
            print('Could not find this CD!')
```

Listing 2 – Creation of a class for Data Processing. This class includes "add_table()" and "del_row()" functions to add or delete an entry to the main CD inventory, respectively.

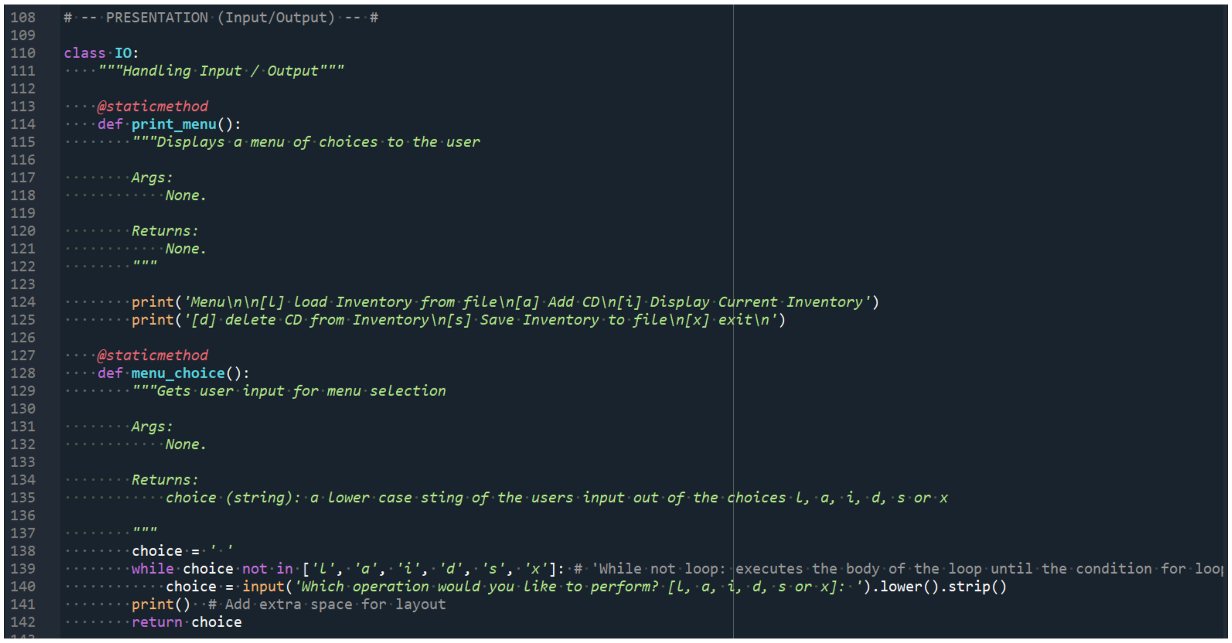
The FileProcessor class the processing section of the script includes two functions for reading from a text file and writing to a text file (**Listing 3**). The *read_file()* function opens the CDInventory.txt file, reads the content line by line, appends data to the main inventory table in the memory, and closes the text file. Similarly, the *write_file()* function, which I created, opens the CDInventory.txt file, writes the content in the main inventory table to the text file line by line, and closes the text file. Within this function, the code in Line 104 joins the elements in the list by ‘,’ and adds a new line.



```
62   class FileProcessor:
63       """Processing the data to and from text file"""
64   
65       @staticmethod
66       def read_file(file_name, table):
67           """Function to manage data ingestion from file to a list of dictionaries
68   
69           Reads the data from file identified by file_name into a 2D table
70           (list of dicts) table one line in the file represents one dictionary row in table.
71   
72           Args:
73               file_name (string): name of file used to read the data from
74               table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
75   
76           Returns:
77               None.
78           """
79           table.clear()  # this clears existing data and allows to load data from file
80           objFile = open(file_name, 'r')
81           for line in objFile:
82               data = line.strip().split(',') # data type: list
83               dicRow = {'ID': int(data[0]), 'Title': data[1], 'Artist': data[2]} # data type: dictionary
84               table.append(dicRow) # data type: list
85           objFile.close()
86   
87       @staticmethod
88       def write_file(file_name, table):
89           """Function to save a 2D table (a list of dictionaries) to file
90   
91           Saves the data in a file identified by file_name into a .txt file
92   
93           Args:
94               file_name (string): name of file used to save the data to
95               table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
96   
97           Returns:
98               None.
99           """
100          objFile = open(file_name, 'w')
101          for row in table:
102              lstValues = list(row.values()) # Converts dictionary row values to a list
103              lstValues[0] = str(lstValues[0]) # Converts the data type from int to str
104              objFile.write(','.join(lstValues) + '\n') # joins the elements in 'lstValues' by ',' and stores in a string
105          objFile.close()
```

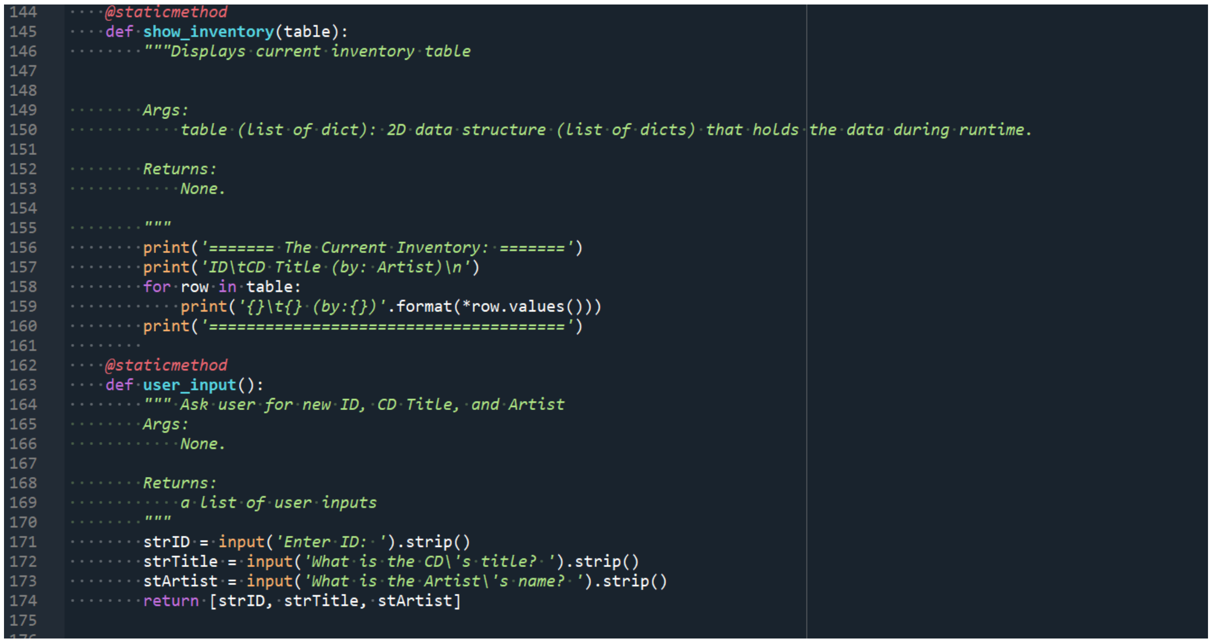
Listing 3 – Creation of a class for File Processing. This class includes “read_file()” and “write_file()” functions for reading from and writing to a text file, respectively.

The presentations section of the script has a single I/O class defined to include four different functions: ***(1)*** *print_menu()* function for displaying the menu options to the user, ***(2)*** *menu_choice()* function for getting user input for menu selection, ***(3)*** *show_inventory()* function for displaying the current inventory data, and ***(4)*** *user_input()* function for asking user input for a new CD ID, Title, and Artist name (**Listing 4** and **5)**. I created the *user_input()* function to read CD ID, title, and artist inputs from the user. The function returns CD ID, title, and artist info as strings (**Listing 5**).



```python
# -- 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']: # 'While not loop: executes the body of the loop until the condition for loo
            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
```

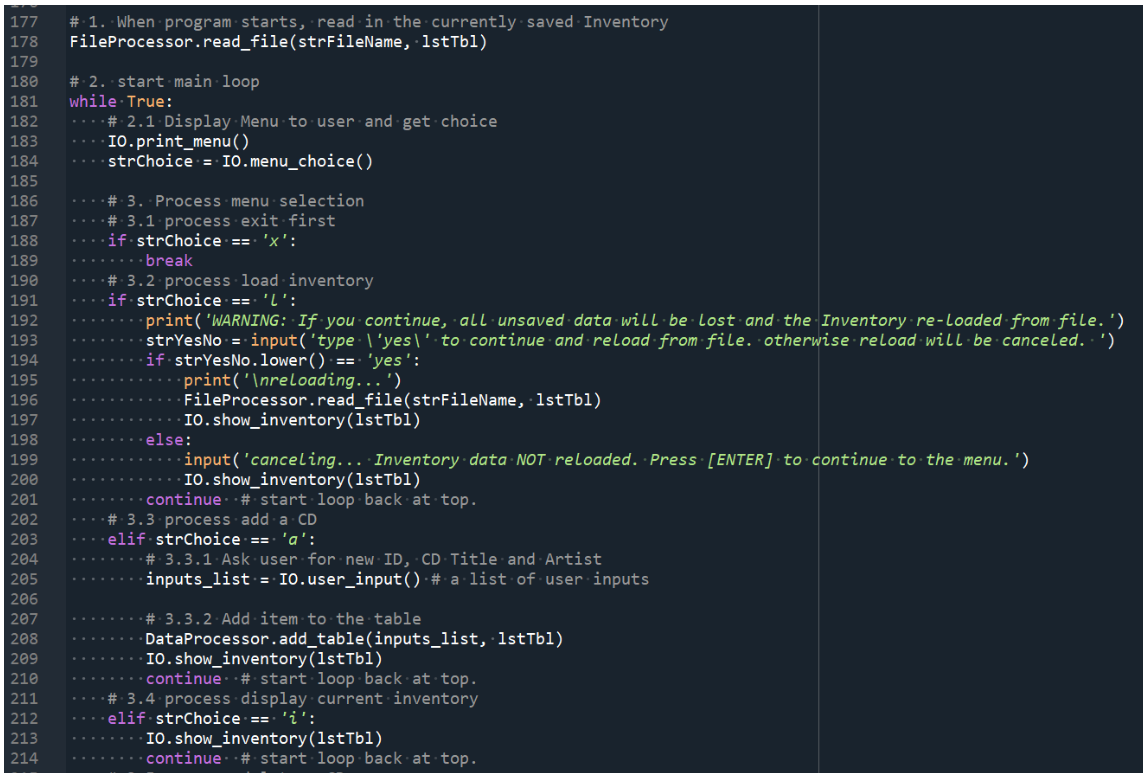
Listing 4 – Creation of I/O Class. This specific part of the presentation section includes *print_menu()* and *menu_choice()* functions for displaying menu and getting user input, respectively.



```python
    @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('======================================')
        
    @staticmethod
    def user_input():
        """ Ask user for new ID, CD Title, and Artist
        Args:
            None.

        Returns:
            a list of user inputs
        """
        strID = input('Enter ID: ').strip()
        strTitle = input('What is the CD\'s title? ').strip()
        stArtist = input('What is the Artist\'s name? ').strip()
        return [strID, strTitle, stArtist]
```

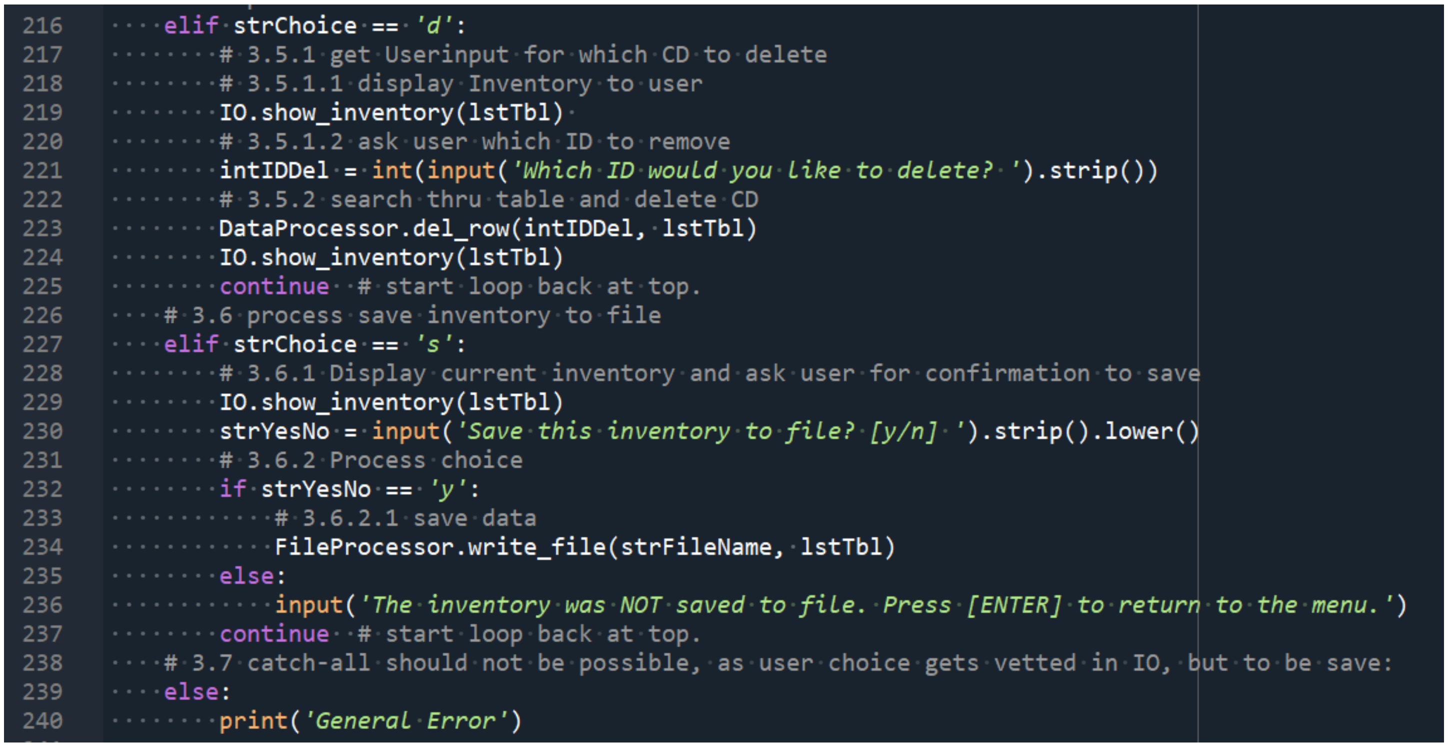
Listing 5 – Creation of I/O Class. This specific part of the presentation section includes *show_inventory()* and *user_input()* functions for displaying the main CD inventory and getting user input for CD ID-title-artist info, respectively.

The rest of the Presentation section is described below. The remaining script loads the inventory, adds CD, displays current inventory, deletes CD, saves inventory to file, and exits the program as shown in **Listing 6** and **7**. The functions that I created in the Data and File processing sections of the script were included in line 205, 208, 223, and line 234 of the script as shown in **Listing 6** and **7**. The *user_input()* function in the line 205 reads the CD ID, title, and artist info inputs from the user and saves to a list, called “*inputs_list*”. This list was then fed into the *add_table()* function in Line 208 to append the new CD data to the main CD inventory (**Listing 6**).



```
177 # 1. When program starts, read in the currently saved Inventory
178 FileProcessor.read_file(strFileName, lstTbl)
179 
180 # 2. start main loop
181 while True:
182     # 2.1 Display Menu to user and get choice
183     IO.print_menu()
184     strChoice = IO.menu_choice()
185 
186     # 3. Process menu selection
187     # 3.1 process exit first
188     if strChoice == 'x':
189         break
190     # 3.2 process load inventory
191     if strChoice == 'l':
192         print('WARNING: If you continue, all unsaved data will be lost and the Inventory re-loaded from file.')
193         strYesNo = input('type \'yes\' to continue and reload from file. otherwise reload will be canceled. ')
194         if strYesNo.lower() == 'yes':
195             print('\nreloading...')
196             FileProcessor.read_file(strFileName, lstTbl)
197             IO.show_inventory(lstTbl)
198         else:
199             input('canceling... Inventory data NOT reloaded. Press [ENTER] to continue to the menu.')
200             IO.show_inventory(lstTbl)
201         continue  # start loop back at top.
202     # 3.3 process add a CD
203     elif strChoice == 'a':
204         # 3.3.1 Ask user for new ID, CD Title and Artist
205         inputs_list = IO.user_input() # a list of user inputs
206 
207         # 3.3.2 Add item to the table
208         DataProcessor.add_table(inputs_list, lstTbl)
209         IO.show_inventory(lstTbl)
210         continue  # start loop back at top.
211     # 3.4 process display current inventory
212     elif strChoice == 'i':
213         IO.show_inventory(lstTbl)
214         continue  # start loop back at top.
```

Listing 6 – The remaining code of the Presentation Section [1/2].



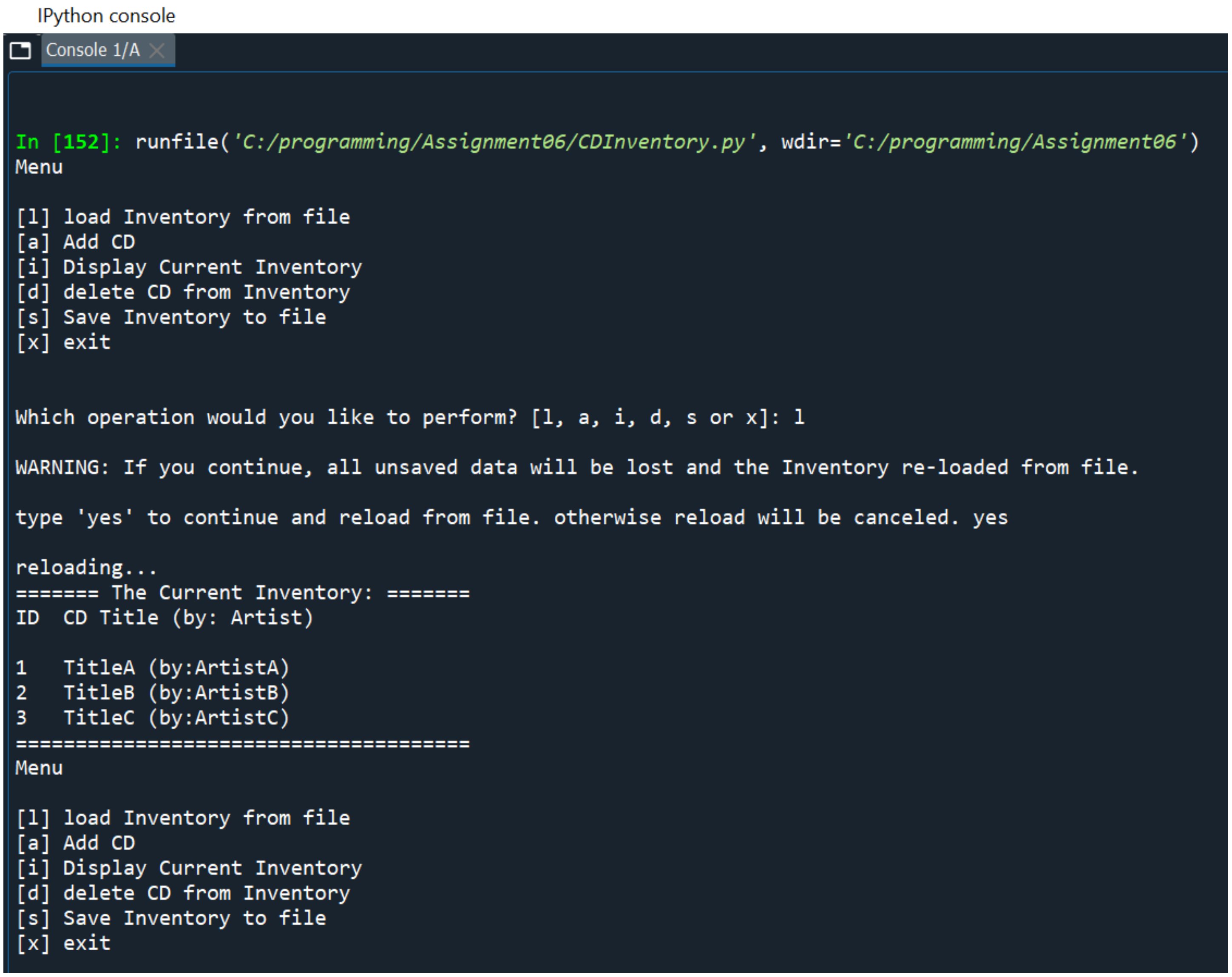
```
216     elif strChoice == 'd':
217         # 3.5.1 get Userinput for which CD to delete
218         # 3.5.1.1 display Inventory to user
219         IO.show_inventory(lstTbl) 
220         # 3.5.1.2 ask user which ID to remove
221         intIDDel = int(input('Which ID would you like to delete? ').strip())
222         # 3.5.2 search thru table and delete CD
223         DataProcessor.del_row(intIDDel, lstTbl)
224         IO.show_inventory(lstTbl)
225         continue  # start loop back at top.
226     # 3.6 process save inventory to file
227     elif strChoice == 's':
228         # 3.6.1 Display current inventory and ask user for confirmation to save
229         IO.show_inventory(lstTbl)
230         strYesNo = input('Save this inventory to file? [y/n] ').strip().lower()
231         # 3.6.2 Process choice
232         if strYesNo == 'y':
233             # 3.6.2.1 save data
234             FileProcessor.write_file(strFileName, lstTbl)
235         else:
236             input('The inventory was NOT saved to file. Press [ENTER] to return to the menu.')
237         continue  # start loop back at top.
238     # 3.7 catch-all should not be possible, as user choice gets vetted in IO, but to be save:
239     else:
240         print('General Error')
```

Listing 7– The remaining code of the Presentation Section [2/2].

The *del_row()* function in Line 223 receives the CD ID info for deletion (via the *input()* function in line 221) and removes that CD entry from the main inventory (**Listing 7**).

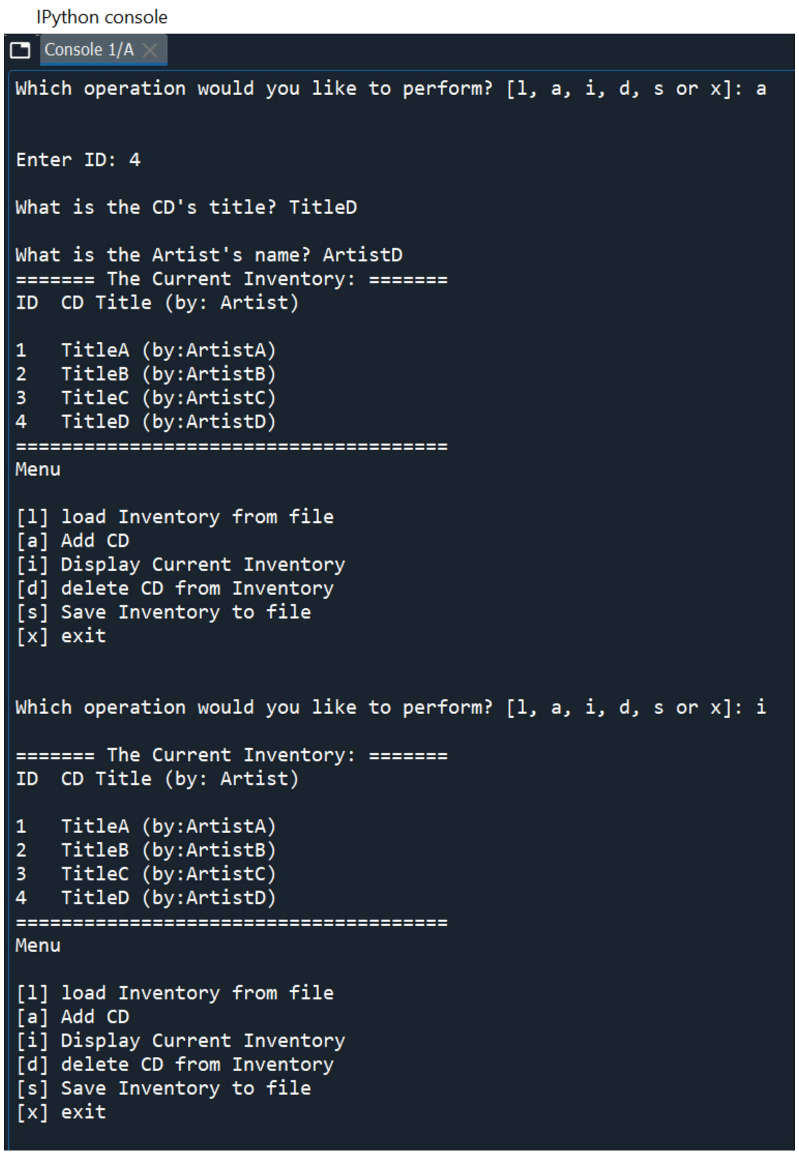
Lastly, I used the *write_file()* function in line 234 of **Listing 7**. If the user wants to save the CD inventory data in the memory to a text file, this function saves the "*lstTbl*" to the CDInventory.txt file.

Successful operation of the script in Spyder IDE was provided in **Figure 1, 2,** and **3**.



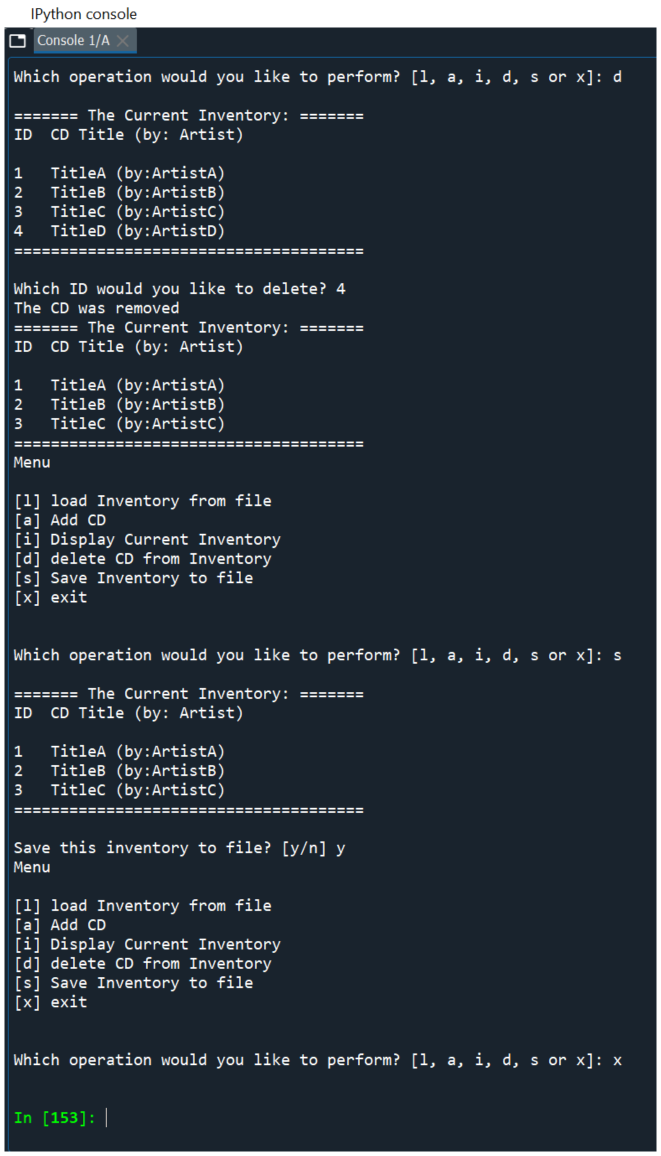
```
In [152]: runfile('C:/programming/Assignment06/CDInventory.py', wdir='C:/programming/Assignment06')
Menu

[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 canceled. yes

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

1   TitleA (by:ArtistA)
2   TitleB (by:ArtistB)
3   TitleC (by:ArtistC)
======================================
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
```

Figure 1 – Successful run of the script in Spyder IDE [1/3].



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


Enter ID: 4

What is the CD's title? TitleD

What is the Artist's name? ArtistD
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   TitleA (by:ArtistA)
2   TitleB (by:ArtistB)
3   TitleC (by:ArtistC)
4   TitleD (by:ArtistD)
======================================
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   TitleA (by:ArtistA)
2   TitleB (by:ArtistB)
3   TitleC (by:ArtistC)
4   TitleD (by:ArtistD)
======================================
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
```

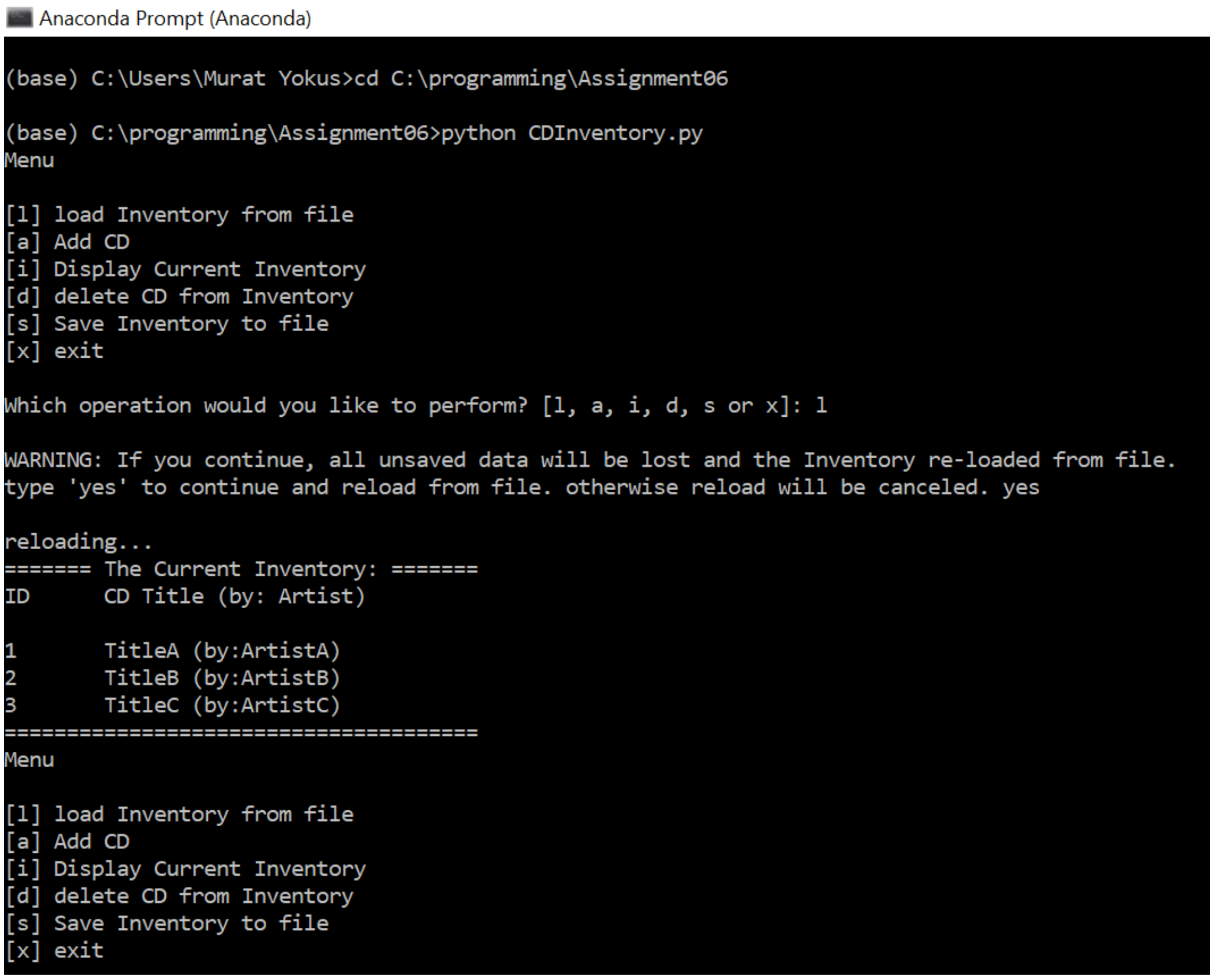
Figure 2– Successful run of the script in Spyder IDE [2/3].



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

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

1   TitleA (by:ArtistA)
2   TitleB (by:ArtistB)
3   TitleC (by:ArtistC)
4   TitleD (by:ArtistD)
======================================

Which ID would you like to delete? 4
The CD was removed
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   TitleA (by:ArtistA)
2   TitleB (by:ArtistB)
3   TitleC (by:ArtistC)
======================================
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   TitleA (by:ArtistA)
2   TitleB (by:ArtistB)
3   TitleC (by:ArtistC)
======================================

Save this inventory to file? [y/n] y
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 [153]:
```

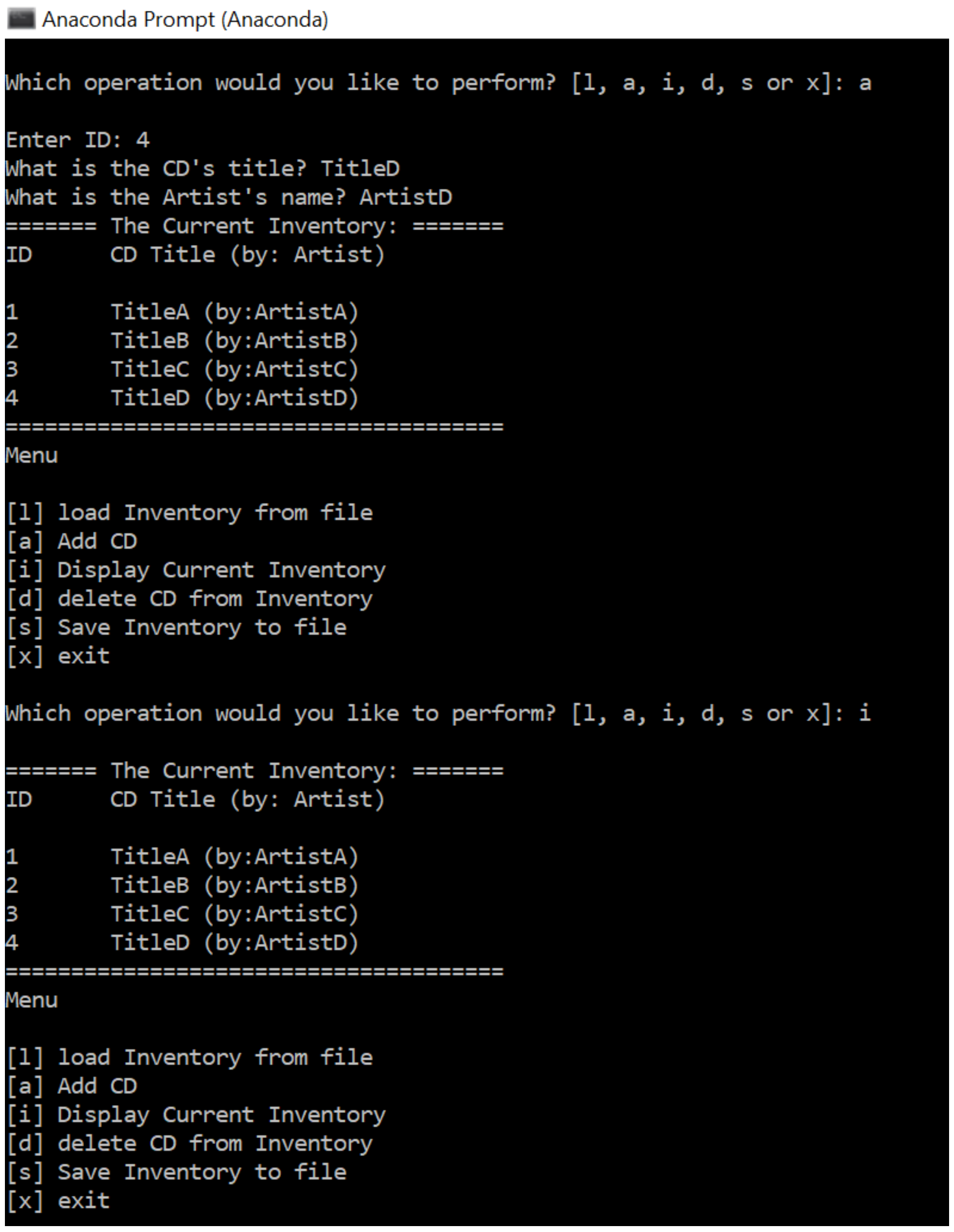
Figure 3– Successful run of the script in Spyder IDE [3/3].

Successful operation of the script in Anaconda Terminal was provided in **Figure 4, 5,** and **6**. The screenshot of the CDInventory.txt file is given in **Figure 7**.



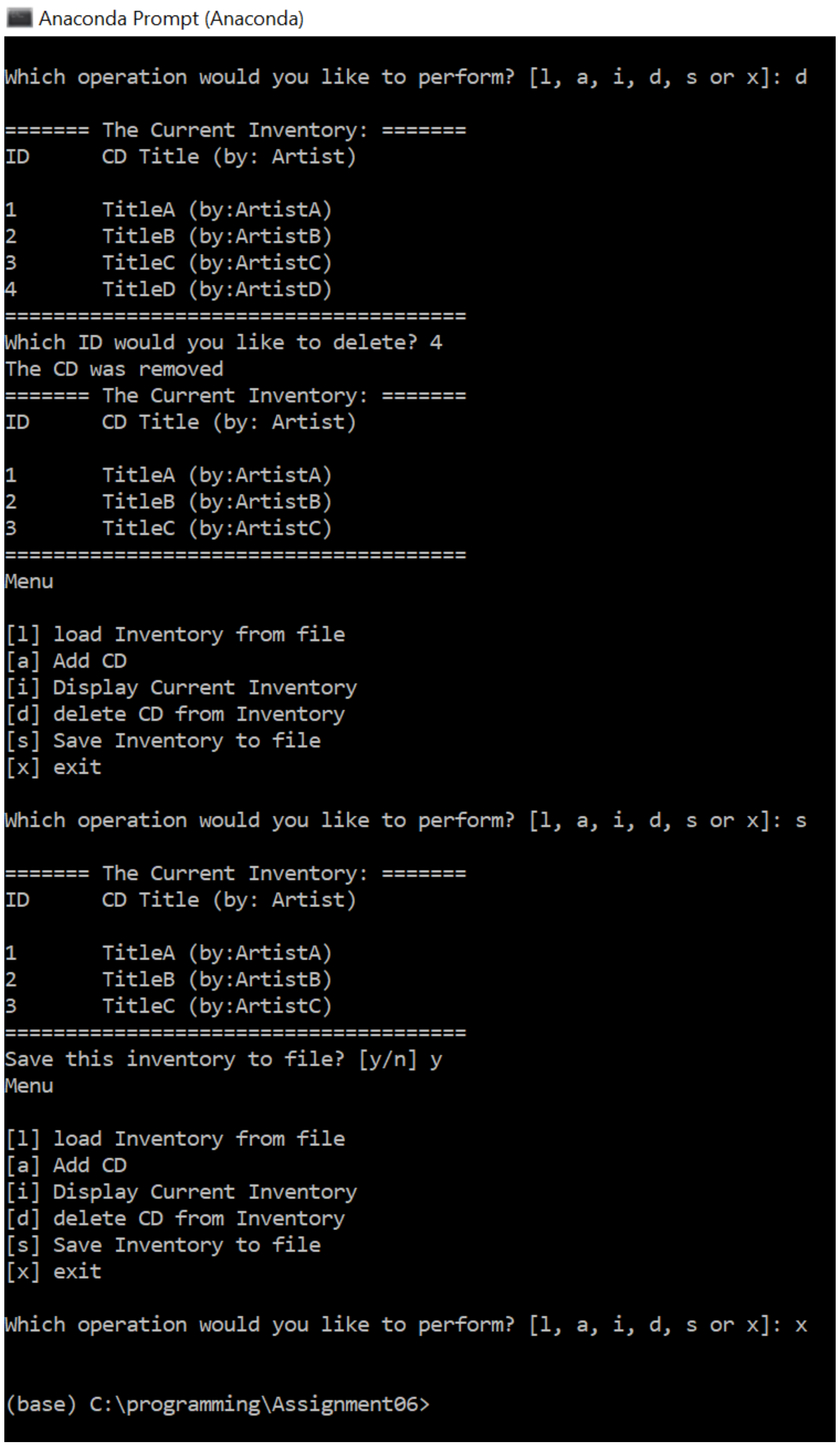
```
(base) C:\Users\Murat Yokus>cd C:\programming\Assignment06

(base) C:\programming\Assignment06>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]: 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 canceled. yes

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

1       TitleA (by:ArtistA)
2       TitleB (by:ArtistB)
3       TitleC (by:ArtistC)
======================================
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
```

Figure 4 – Successful run of the script in terminal [1/3].



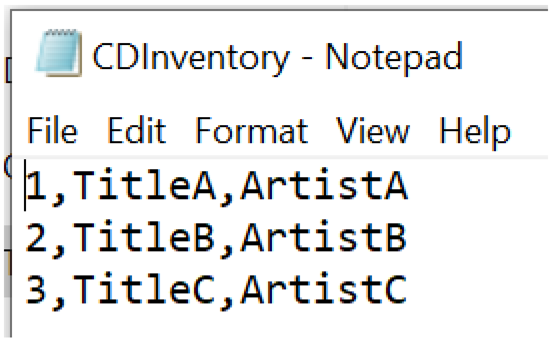
```
Which operation would you like to perform? [l, a, i, d, s or x]: a

Enter ID: 4
What is the CD's title? TitleD
What is the Artist's name? ArtistD
======= The Current Inventory: =======
ID      CD Title (by: Artist)

1       TitleA (by:ArtistA)
2       TitleB (by:ArtistB)
3       TitleC (by:ArtistC)
4       TitleD (by:ArtistD)
======================================
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       TitleA (by:ArtistA)
2       TitleB (by:ArtistB)
3       TitleC (by:ArtistC)
4       TitleD (by:ArtistD)
======================================
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
```

Figure 5 – Successful run of the script in terminal [2/3].



```
Anaconda Prompt (Anaconda)

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

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

1       TitleA (by:ArtistA)
2       TitleB (by:ArtistB)
3       TitleC (by:ArtistC)
4       TitleD (by:ArtistD)
======================================
Which ID would you like to delete? 4
The CD was removed
======= The Current Inventory: =======
ID      CD Title (by: Artist)

1       TitleA (by:ArtistA)
2       TitleB (by:ArtistB)
3       TitleC (by:ArtistC)
======================================
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       TitleA (by:ArtistA)
2       TitleB (by:ArtistB)
3       TitleC (by:ArtistC)
======================================
Save this inventory to file? [y/n] y
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) C:\programming\Assignment06>
```

Figure 6 – Successful run of the script in terminal [3/3].


CDInventory - Notepad

File Edit Format View Help

```
1,TitleA,ArtistA
2,TitleB,ArtistB
3,TitleC,ArtistC
```

Figure 7 – Content of the CDInventory.txt file after saving the CD inventory table to a text file.

GitHub Link

The knowledge document, the script, and CDInventory.txt file were uploaded to GitHub/Assignment_06 repository. Link: https://github.com/myokus/Assignment_06

Module 6: Learnings

In the Module 6, I learned and practiced the following topics.

- functions & classes
- parameters, arguments, and return values
- local and global variables

Summary

Overall, the objective of this assignment is to implement functions and classes in the script of Assignment05. The inventory program that has a menu structure and allows user to load CD data from CDInventory.txt file, enter CD data, view the current inventory, delete DC data from inventory, save data to CDInventory.txt data file, and exit the program. The starter code that was provided with the Assignment06 was modified to include functions and classes to simplify and organize the overall script. This document demonstrate successful implementation and operation of functions and classes along with my learnings from the Module 6.

Appendix

Listing CDInventory.py

```
#------------------------------------------#
# Title: CDInventory.py
# Desc: Script CDInventory to store CD Inventory data (using Classes and Functions)
# Change Log: (Who, When, What)
# DBiesinger, 2030-Jan-01, Created File #
# MYokus, 2021-Aug-15, Added Code
#------------------------------------------#

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


```

```python
# -- PROCESSING -- #
class DataProcessor:
    """Processing the data in a 2D table (list of dicts)"""

    @staticmethod
    def add_table(inputs, table):
        """Function to add a list to a 2D table (list of dicts)

       Add user inputs (a list) to the main inventory table (list of dicts)

       Args:
           inputs (list): user inputs (ID, title, artist)
           table (list of dict): 2D data structure (list of dicts) that holds the data during runtime

       Returns:
           None
       """
        dicRow = {'ID': int(inputs[0]), 'Title': inputs[1], 'Artist': inputs[2]}
        table.append(dicRow)

    @staticmethod
    def del_row(row, table):
        """ Function to delete a row in a 2D table

       Deletes an entry from the main inventory table (list of dicts)
       Args:
           row (int): the row number to be deleted
           table (list of dict): 2D data structure (list of dicts) that holds the data during runtime

       Returns:
           None
       """
        intRowNr = -1
        blnCDRemoved = False
        for row in table:
            intRowNr += 1
            if row['ID'] == intIDDel:
                del table[intRowNr]
                blnCDRemoved = True
                break
        if blnCDRemoved:
            print('The CD was removed')
        else:
            print('Could not find this CD!')

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.
       """
        table.clear()  # this clears existing data and allows to load data from file
        objFile = open(file_name, 'r')
        for line in objFile:
            data = line.strip().split(',') # data type: list
            dicRow = {'ID': int(data[0]), 'Title': data[1], 'Artist': data[2]} # data type: dictionary
            table.append(dicRow) # data type: list
        objFile.close()

    @staticmethod
    def write_file(file_name, table):
        """Function to save a 2D table (a list of dictionaries) to file

       Saves the data in a file identified by file_name into a .txt file

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

       Returns:
           None.
       """
        objFile = open(file_name, 'w')
        for row in table:
            lstValues = list(row.values()) # Converts dictionary row values to a list
            lstValues[0] = str(lstValues[0]) # Converts the data type from int to str
            objFile.write(','.join(lstValues) + '\n') # joins the elements in 'lstValues' by ',' and stores in a string
        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('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']: # 'While not loop: executes the body of the loop until the condition for loop termination is met'
            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 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 user_input():
        """ Ask user for new ID, CD Title, and Artist
       Args:
           None.

       Returns:
           a list of user inputs
       """
        strID = input('Enter ID: ').strip()
        strTitle = input('What is the CD\'s title? ').strip()
        stArtist = input('What is the Artist\'s name? ').strip()
        return [strID, strTitle, stArtist]


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

# 2. start main loop
while True:
    # 2.1 Display Menu to user and get choice
    IO.print_menu()
    strChoice = IO.menu_choice()

    # 3. Process menu selection
    # 3.1 process exit first
    if strChoice == 'x':
        break
    # 3.2 process load inventory
    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('\nreloading...')
            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  # start loop back at top.
    # 3.3 process add a CD
    elif strChoice == 'a':
        # 3.3.1 Ask user for new ID, CD Title and Artist
```

```
            inputs_list = IO.user_input() # a list of user inputs

            # 3.3.2 Add item to the table
            DataProcessor.add_table(inputs_list, lstTbl)
            IO.show_inventory(lstTbl)
            continue  # start loop back at top.
        # 3.4 process display current inventory
        elif strChoice == 'i':
            IO.show_inventory(lstTbl)
            continue  # start loop back at top.
        # 3.5 process delete a CD
        elif strChoice == 'd':
            # 3.5.1 get Userinput for which CD to delete
            # 3.5.1.1 display Inventory to user
            IO.show_inventory(lstTbl)
            # 3.5.1.2 ask user which ID to remove
            intIDDel = int(input('Which ID would you like to delete? ').strip())
            # 3.5.2 search thru table and delete CD
            DataProcessor.del_row(intIDDel, lstTbl)
            IO.show_inventory(lstTbl)
            continue  # start loop back at top.
        # 3.6 process save inventory to file
        elif strChoice == 's':
            # 3.6.1 Display current inventory and ask user for confirmation to save
            IO.show_inventory(lstTbl)
            strYesNo = input('Save this inventory to file? [y/n] ').strip().lower()
            # 3.6.2 Process choice
            if strYesNo == 'y':
                # 3.6.2.1 save data
                FileProcessor.write_file(strFileName, lstTbl)
            else:
                input('The inventory was NOT saved to file. Press [ENTER] to return to the menu.')
            continue  # start loop back at top.
        # 3.7 catch-all should not be possible, as user choice gets vetted in IO, but to be save:
        else:
            print('General Error')
```