Artem Aleshin
August 30th, 2022
IT FDN 110 A
Assignment 08

Assignment 08 Knowledge Document

Introduction

The Knowledge Document covers Points of Interest from Module 08, the laboratories, and the completed Assignment. As part of working through the tasks, I learned about Object Oriented Programming, Classes, Objects, Constructors, Attributes, Properties, Fields and Methods.

Module 08 Points of Interest

All in all, Module 08 covered an incredible amount of content. I think I will need more help understanding the difference between using fields, constructors and setters, and when to use which one, as well as decorators. I did like that setters can be used to catch Exceptions. It was also good that a lot of code from the previous assignments could be reused if properly modified.

Module 08 Laboratories

LAB 08_A

1. The script is as follows:

```python
#-----------------------------------------#
# Title: LAB08_A.py
# Desc: Working with Classses
# Change Log: (Who, When, What)
# aaleshin, 2022-Aug-29, Created File
#-----------------------------------------#


class TrackInfo():

    # -- Fields -- #

    position = 0
    title = ''
    length = ''

    # -- Constructor -- #
        # -- Attributes -- #
    # -- Methods -- #

objTrack1 = TrackInfo()
objTrack1.position = 1
objTrack1.title = 'Once'
objTrack1.length = '3:51'

print('Position: {}'.format(objTrack1.position))
print('Title: {}'.format(objTrack1.title))
print('Length: {}'.format(objTrack1.length))
```

Figure 1: LAB 08_A Script.

2. The output is as follows:

```
In [5]: runfile('C:/_FDProgramming/Lab_08/Lab08_A.py', wdir='C:/_FDProgramming/Lab_08')
Position: 1
Title: Once
Length: 3:51
```

Figure 2: LAB 08_A Output.

3. The script defines a class and the fields of the class. An object instance of the class is then created and the fields are set.

LAB 08-B

1. A copy of LAB08-A was made.
2. The script was modified and is as follows:

```
#------------------------------------------#
# Title: LAB08_B.py
# Desc: Working with Classses
# Change Log: (Who, When, What)
# aaleshin, 2022-Aug-29, Created File
# aaleshin, 2022-Aug-29, Updated with Constructor
#------------------------------------------#


class TrackInfo():

    # -- Fields -- #

    position = 0
    title = ''
    length = ''

    # -- Constructor -- #
    def __init__(self, p, t, l):
        # -- Attributes -- #
        self.position = p
        self.title = t
        self.length = l
    # -- Methods -- #

objTrack1 = TrackInfo(1, 'Once', '3:51')

print('Position: {}'.format(objTrack1.position))
print('Title: {}'.format(objTrack1.title))
print('Length: {}'.format(objTrack1.length))
```

Figure 3: LAB 08_B Script.

3. The script was tested,
    a. The output is as follows:

```
In [6]: runfile('C:/_FDProgramming/Lab_08/Lab08_B.py', wdir='C:/_FDProgramming/Lab_08')
Position: 1
Title: Once
Length: 3:51
```

Figure 4: LAB 08_B Output.

    b. The script works in a similar way to Lab A except the class now has a constructor and the constructor is used to set the attributes.

LAB 08-C

1. A copy of LAB08-B was made and was not modified because I used attributes in Lab B.

LAB 08-D

1. A copy of LAB08-C was made and was modified to be as follows:

```python
#-----------------------------------------#
# Title: LAB08_D.py
# Desc: Working with Classses
# Change Log: (Who, When, What)
# aaleshin, 2022-Aug-29, Created File
# aaleshin, 2022-Aug-29, Updated with Constructor
# aaleshin, 2022-Aug-29, No Changes
# aaleshin, 2022-Aug-29, Added Properties
#-----------------------------------------#


class TrackInfo():

    # -- Fields -- #
    # -- Constructor -- #
    def __init__(self, p, t, l):
        # -- Attributes -- #
        self.__position = p
        self.__title = t
        self.__length = l

    @property
    def position(self):
        return self.__position

    @property
    def title(self):
        return self.__title

    @property
    def length(self):
        return self.__length

    @position.setter
    def position(self, value):
        if str(value).isnumeric():
            self.__position = value
        else:
            raise Exception('The Position has to be an Integer!')

    @title.setter
    def title(self, value):
        if str(value).isnumeric():
            raise Exception('The Title has to be a String!')
        else:
            self.__title = value

    @length.setter
    def length(self, value):
        if str(value).isnumeric():
            raise Exception('The Length has to be a String!')
        else:
            self.__length = value

```

Figure 5: LAB 08_D Script Part 1.

```
56      # -- Methods -- #
57
58  objTrack1 = TrackInfo(1, 'Once', '3:51')
59  print(type(objTrack1))
60  objTrack1.position = 1
61  objTrack1.title = 'Once'
62  objTrack1.length = '3:51'
63
64  print('Position: {}'.format(objTrack1.position))
65  print('Title: {}'.format(objTrack1.title))
66  print('Length: {}'.format(objTrack1.length))
67
68  objTrack2 = TrackInfo('two', 1, 3)
69  print(type(objTrack2))
70  objTrack2.position = 'two'
71  objTrack2.title = 1
72  objTrack2.length = 3
73
74  print('Position: {}'.format(objTrack2.position))
75  print('Title: {}'.format(objTrack2.title))
76  print('Length: {}'.format(objTrack2.length))
```

Figure 6: LAB 08_D Script Part 2.

2. The output of the code is as follows:

```
In [24]: runfile('C:/_FDProgramming/Lab_08/Lab08_D.py', wdir='C:/_FDProgramming/Lab_08')
<class '__main__.TrackInfo'>
Position: 1
Title: Once
Length: 3:51
<class '__main__.TrackInfo'>
Traceback (most recent call last):

  File "C:\Users\artem\AppData\Local\Programs\Spyder\pkgs\spyder_kernels\py3compat.py", line 356, in compat_exec
    exec(code, globals, locals)

  File "c:\_fdprogramming\lab_08\lab08_d.py", line 70, in <module>
    objTrack2.position = 'two'

  File "c:\_fdprogramming\lab_08\lab08_d.py", line 39, in position
    raise Exception('The Position has to be an Integer!')

Exception: The Position has to be an Integer!
```

Figure 7: LAB 08_D Output.

3. The script works by adding getters and setters to the class. The attributes are then set using the setters and the getters are used to print out the object attributes. In addition, errors are purposely introduced in the second object to test the setters.

LAB 08-E

1. A copy of LAB08-D was made and was modified to be as follows:

```python
#-----------------------------------------#
# Title: LAB08_E.py
# Desc: Working with Classses
# Change Log: (Who, When, What)
# aaleshin, 2022-Aug-29, Created File
# aaleshin, 2022-Aug-29, Updated with Constructor
# aaleshin, 2022-Aug-29, No Changes
# aaleshin, 2022-Aug-29, Added Properties
# aaleshin, 2022-Aug-29, Added a __str__ Method
#-----------------------------------------#


class TrackInfo():

    # -- Fields -- #
    # -- Constructor -- #
    def __init__(self, p, t, l):
        # -- Attributes -- #
        self.__position = p
        self.__title = t
        self.__length = l

    @property
    def position(self):
        return self.__position

    @property
    def title(self):
        return self.__title

    @property
    def length(self):
        return self.__length

    @position.setter
    def position(self, value):
        if str(value).isnumeric():
            self.__position = value
        else:
            raise Exception('The Position has to be an Integer!')

    @title.setter
    def title(self, value):
        if str(value).isnumeric():
            raise Exception('The Title has to be a String!')
        else:
            self.__title = value

    @length.setter
    def length(self, value):
        if str(value).isnumeric():
            raise Exception('The Length has to be a String!')
        else:
            self.__length = value


```

Figure 8: LAB 08_E Script.

```python
    # -- Methods -- #
    def __str__(self):
        return 'Position: {}, Title: {}, Length: {}'.format(self.position, self.title, self.length)

objTrack1 = TrackInfo(1, 'Once', '3:51')
print(type(objTrack1))
objTrack1.position = 1
objTrack1.title = 'Once'
objTrack1.length = '3:51'

print('Position: {}'.format(objTrack1.position))
print('Title: {}'.format(objTrack1.title))
print('Length: {}'.format(objTrack1.length))

print(objTrack1.__str__())
```

Figure 9: LAB 08_E Output.

2. The output of the code is as follows:

```
In [1]: runfile('C:/_FDProgramming/Lab_08/Lab08_E.py', wdir='C:/_FDProgramming/Lab_08')
<class '__main__.TrackInfo'>
Position: 1
Title: Once
Length: 3:51
Position: 1, Title: Once, Length: 3:51
```

Figure 10: LAB 08_E Output.

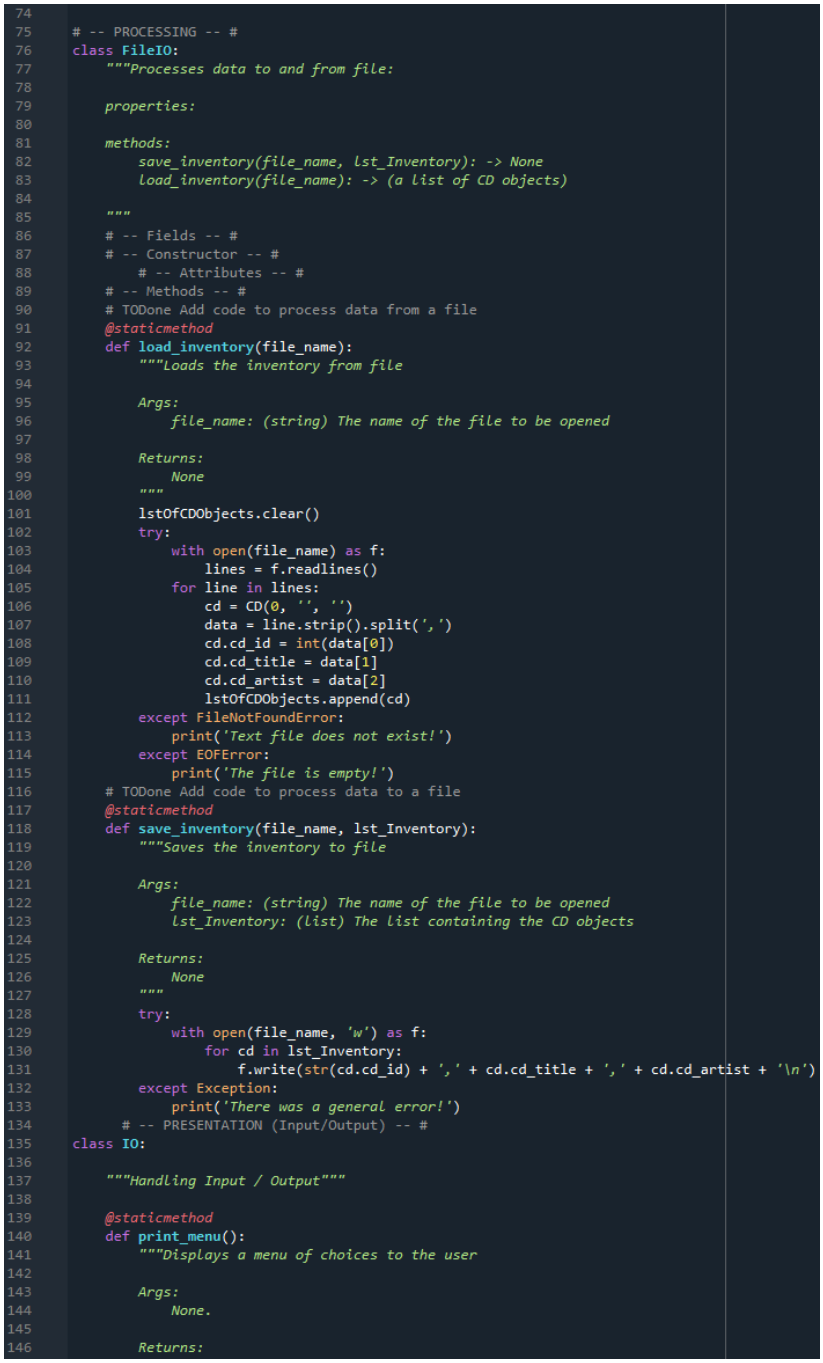
3. The code introduces the __str__(self) method to the class and then uses the method to extract information from the object.

Assignment 08

1. Folder called Assignment08 was created.
2. The code was modified and is as follows:

```python
#------------------------------------------#
# Title: CD_Inventory.py
# Desc: Assignnment 08 - Working with classes
# Change Log: (Who, When, What)
# DBiesinger, 2030-Jan-01, created file
# DBiesinger, 2030-Jan-01, added pseudocode to complete assignment 08
# aaleshin, 2022-Aug-30, Modified File; Replaced Pseudocode with Code.
#------------------------------------------#

# -- DATA -- #
strFileName = 'cdInventory.txt'
lstOfCDObjects = []

class CD:
    """Stores data about a CD:

    properties:
        cd_id: (int) with CD ID
        cd_title: (string) with the title of the CD
        cd_artist: (string) with the artist of the CD
    methods:

    """
    # TODone Add Code to the CD
    # -- Fields -- #
    # -- Constructor -- #
    def __init__(self, ID, title, artist):
        """Constructor for the CD class"""
        # -- Attributes -- #
        self.__cd_id = ID
        self.__cd_title = title
        self.__cd_artist = artist

    @property
    def cd_id(self):
        """Getter for CD ID"""
        return self.__cd_id

    @property
    def cd_title(self):
        """Getter for CD Title"""
        return self.__cd_title

    @property
    def cd_artist(self):
        """Getter for CD Artist"""
        return self.__cd_artist

    @cd_id.setter
    def cd_id(self, value):
        """Setter for CD ID"""
        if str(value).isnumeric():
            self.__cd_id = value
        else:
            raise Exception('The ID has to be an Integer!')

    @cd_title.setter
    def cd_title(self, value):
        """Setter for CD Title"""
        if str(value).isnumeric():
            raise Exception('The Title has to be a String!')
        else:
            self.__cd_title = value

    @cd_artist.setter
    def cd_artist(self, value):
        """Setter for CD Artist"""
        if str(value).isnumeric():
            raise Exception('The Artist has to be a String!')
        else:
            self.__cd_artist = value

    # -- Methods -- #
```

Figure 11: Assignment 08 Script Part 1.

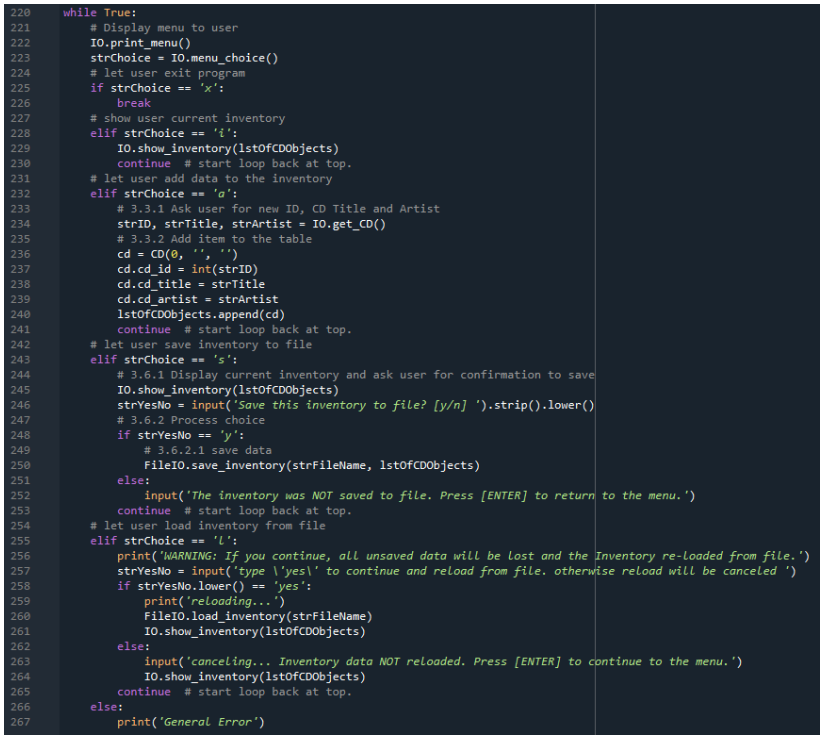


```
74
75  # -- PROCESSING -- #
76  class FileIO:
77      """Processes data to and from file:
78
79      properties:
80
81      methods:
82          save_inventory(file_name, lst_Inventory): -> None
83          load_inventory(file_name): -> (a list of CD objects)
84
85      """
86      # -- Fields -- #
87      # -- Constructor -- #
88          # -- Attributes -- #
89      # -- Methods -- #
90      # TODone Add code to process data from a file
91      @staticmethod
92      def load_inventory(file_name):
93          """Loads the inventory from file
94
95          Args:
96              file_name: (string) The name of the file to be opened
97
98          Returns:
99              None
100         """
101         lstOfCDObjects.clear()
102         try:
103             with open(file_name) as f:
104                 lines = f.readlines()
105             for line in lines:
106                 cd = CD(0, '', '')
107                 data = line.strip().split(',')
108                 cd.cd_id = int(data[0])
109                 cd.cd_title = data[1]
110                 cd.cd_artist = data[2]
111                 lstOfCDObjects.append(cd)
112         except FileNotFoundError:
113             print('Text file does not exist!')
114         except EOFError:
115             print('The file is empty!')
116     # TODone Add code to process data to a file
117     @staticmethod
118     def save_inventory(file_name, lst_Inventory):
119         """Saves the inventory to file
120
121         Args:
122             file_name: (string) The name of the file to be opened
123             lst_Inventory: (list) The list containing the CD objects
124
125         Returns:
126             None
127         """
128         try:
129             with open(file_name, 'w') as f:
130                 for cd in lst_Inventory:
131                     f.write(str(cd.cd_id) + ',' + cd.cd_title + ',' + cd.cd_artist + '\n')
132         except Exception:
133             print('There was a general error!')
134     # -- PRESENTATION (Input/Output) -- #
135 class IO:
136
137     """Handling Input / Output"""
138
139     @staticmethod
140     def print_menu():
141         """Displays a menu of choices to the user
142
143         Args:
144             None.
145
146         Returns:
```

Figure 12: Assignment 08 Script Part 2.

```
            None.
        """

        print('Menu\n\n[l] Load Inventory from file\n[a] Add CD\n[i] Display Current Inventory')
        print('[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, s or x

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

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


        Args:
            lstInventory (list of strings) that holds the data during runtime.

        Returns:
            None.

        """
        print('======= The Current Inventory: =======')
        print('ID\tCD Title (by: Artist)\n')
        for row in lstInventory:
            print('{}\t{} (by:{})'.format(str(row.cd_id), row.cd_title, row.cd_artist))
        print('======================================')

    @staticmethod
    def get_CD():
        """Gets user input for CD ID, Title and Artist.

        Args:
            None.

        Returns:
            Returns the CD ID as a string.
            Returns the CD Title as a string.
            Returns the CD Artist as as string.
        """
        try:
            ID = input('Enter ID: ').strip()
        except ValueError:
            print('ID is not an integer!')
        Title = input('What is the CD\'s title? ').strip()
        Artist = input('What is the Artist\'s name? ').strip()

        return ID, Title, Artist

# -- Main Body of Script -- #
# TODone Add Code to the main body
# Load data from file into a list of CD objects on script start

try:
    FileIO.load_inventory(strFileName)
except FileNotFoundError:
    print('Text file does not exist!')
except EOFError:
    print('The file is empty!')

```

Figure 13: Assignment 08 Script Part 3.



```
while True:
    # Display menu to user
    IO.print_menu()
    strChoice = IO.menu_choice()
    # let user exit program
    if strChoice == 'x':
        break
    # show user current inventory
    elif strChoice == 'i':
        IO.show_inventory(lstOfCDObjects)
        continue  # start loop back at top.
    # let user add data to the inventory
    elif strChoice == 'a':
        # 3.3.1 Ask user for new ID, CD Title and Artist
        strID, strTitle, strArtist = IO.get_CD()
        # 3.3.2 Add item to the table
        cd = CD(0, '', '')
        cd.cd_id = int(strID)
        cd.cd_title = strTitle
        cd.cd_artist = strArtist
        lstOfCDObjects.append(cd)
        continue  # start loop back at top.
    # let user save inventory to file
    elif strChoice == 's':
        # 3.6.1 Display current inventory and ask user for confirmation to save
        IO.show_inventory(lstOfCDObjects)
        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
            FileIO.save_inventory(strFileName, lstOfCDObjects)
        else:
            input('The inventory was NOT saved to file. Press [ENTER] to return to the menu.')
        continue  # start loop back at top.
    # let user load inventory from file
    elif 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...')
            FileIO.load_inventory(strFileName)
            IO.show_inventory(lstOfCDObjects)
        else:
            input('canceling... Inventory data NOT reloaded. Press [ENTER] to continue to the menu.')
            IO.show_inventory(lstOfCDObjects)
        continue  # start loop back at top.
    else:
        print('General Error')
```

Figure 14: Assignment 08 Script Part 4.

3. The script was tested in both Spyder and Anaconda Prompt.
    a. The output in Spyder:

```
In [4]: runfile('C:/_FDProgramming/Assignment08/CD_Inventory.py', wdir='C:/_FDProgramming/Assignment08')
Text file does not exist!
Menu

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

Which operation would you like to perform? [l, a, i, 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...
Text file does not exist!
======= The Current Inventory: =======
ID  CD Title (by: Artist)

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

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

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

Enter ID: 1
What is the CD's title? Ten
What is the Artist's name? Pearl Jam
Menu

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

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

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

1   Ten (by:Pearl Jam)
======================================
Menu

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

Figure 15: Assignment 08 Spyder Output Part 1.

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

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

1   Ten (by:Pearl Jam)
=======================================
Save this inventory to file? [y/n] y
Menu

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

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

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

1   Ten (by:Pearl Jam)
=======================================
Menu

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

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

Enter ID: 2
What is the CD's title? Nevermind
What is the Artist's name? Nirvana
Menu

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

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

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

1   Ten (by:Pearl Jam)
2   Nevermind (by:Nirvana)
=======================================
Save this inventory to file? [y/n] y
Menu

[l] load Inventory from file
[a] Add CD
```

Figure 16: Assignment 08 Spyder Output Part 2.

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

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

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

1   Ten (by:Pearl Jam)
2   Nevermind (by:Nirvana)
=======================================
Menu

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

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

Figure 17: Assignment 08 Spyder Output Part 3.

b. The output in Anaconda Prompt:

```
(base) C:\_FDProgramming\Assignment08>python CD_Inventory.py
Menu

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

Which operation would you like to perform? [l, a, i, 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       Ten (by:Pearl Jam)
2       Nevermind (by:Nirvana)
=======================================
Menu

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

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

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

1       Ten (by:Pearl Jam)
2       Nevermind (by:Nirvana)
=======================================
Menu

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

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

Enter ID: 3
What is the CD's title? Dirt
What is the Artist's name? Alice in Chains
Menu

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

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

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

1       Ten (by:Pearl Jam)
2       Nevermind (by:Nirvana)
3       Dirt (by:Alice in Chains)
=======================================
Save this inventory to file? [y/n] y
Menu

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

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

Figure 18: Assignment 08 Anaconda Prompt Output.

4. All options of the script were tested. The text file after the operations shown above is:

```
1,Ten,Pearl Jam
2,Nevermind,Nirvana
3,Dirt,Alice in Chains
```

Figure 19: Assignment 08 Text Output.

Same as with the previous assignment, I relied a lot on the lessons learned from the labs and reused from them as well as previous assignments. I started by populating the CD class with a constructor, getters and setters. I then worked on the methods for the class FileIO. Throughout the process I tested the code. Next, I reused code from the previous assignment for the class IO and repurposed the code both in the class and in the main body to use the CD class.

This Knowledge Document and the script for the assignment have been uploaded to the following GitHub Repository:

https://github.com/aaleshinUW/Assignment_08

Summary

The document covers the Points of Interest, the Laboratories, and the Assignment from Module 07. By completing Module 07, I learned about working with Binary Files, Text Files and Structured Error Handling.