

Artem Aleshin
August 14th, 2022
IT FDN 110 A
Assignment 06

Assignment 06 Knowledge Document

Introduction

The Knowledge Document covers Points of Interest from Module 06, the work done as part of the sixth set of Laboratories and the completed Assignment. As part of working through the tasks, I learned about Functions, Classes, DocStrings and Variable Scope.

Module 06 Points of Interest

A topic that I found interesting from Module 06 is docstrings. Spyder does a great job of letting the user know what the information for the function is. I am also interested in classes and would like to know how the script knows when a class ends. Lastly, I think that the tic tac toe program in the book is interesting, albeit quite complicated. I would be interested in learning more about how one can improve the AI for the program.

Module 06 Laboratories

LAB 06_A

1. The script was duplicated.
2. The script was modified and is as follows:

```
1  #-----#
2  # Title: Lab06_A.py
3  # Desc: Script demonstrating Functions concept, based on Basic_Math.py (Assignment02)
4  # Change Log: (Who, When, What)
5  # DBiesinger, 2030-Jan-01, Created File
6  # DBiesinger, 2030-Jan-01, Modified to demonstrate SoC
7  # DBiesinger, 2030-Jan-01, Modified to demonstrate Functions
8  # aaleshin, 2022-Aug-13, Modified to demonstrate Attributes and Return Values
9  #-----#
10
11 # -- DATA -- #
12 intNumA = None
13 intNumB = None
14
15 # -- PROCESSING -- #
16 # Process the data
17 def getSum(intNumA, intNumB):
18     resultS = intNumA + intNumB
19     return resultS
20
21 def getDif(intNumA, intNumB):
22     resultD = intNumA - intNumB
23     return resultD
24
25 def getPro(intNumA, intNumB):
26     resultP = intNumA * intNumB
27     return resultP
28
29 def getQuo(intNumA, intNumB):
30     resultQ = intNumA / intNumB
31     return resultQ
32
33 # -- PRESENTATION (Input/Output) -- #
34 # Get User input data
35 print('Basic Math script. Calculating the Sum, DIfference, Product and Quotient of two numbers.')
36 intNumA = int(input('Please enter the 1st Number: '))
37 intNumB = int(input('Please enter the 2nd number: '))
38 # Display the Results
39 print('\n\nThis script calculated using the Numbers', intNumA, 'and', intNumB)
40 print('The Results are:\n')
41 print('Sum:\t\t', getSum(intNumA, intNumB), '\nDifference:\t', getDif(intNumA, intNumB))
42 print('Product:\t', getPro(intNumA, intNumB), '\nQuotient:\t', getQuo(intNumA, intNumB))
```

Figure 1: Lab 06-A Script.

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

```
In [1]: runfile('C:/_FDProgramming/Lab_06/Lab06_A.py', wdir='C:/_FDProgramming/Lab_06')
Basic Math script. Calculating the Sum, Difference, Product and Quotient of two numbers.
Please enter the 1st Number: 12
Please enter the 2nd number: 4

This script calculated using the Numbers 12 and 4
The Results are:

Sum:          16
Difference:    8
Product:       48
Quotient:     3.0
```

Figure 2: Lab 06-A Output.

- b. The script works by first initializing two variables that will be used throughout the rest of the code. Four functions are then defined with each one returning a result based on the algebraic calculation that was performed. The user is then prompted to enter two values which then get assigned to the variables initialized in the beginning of the script. The two values then get plugged into the functions which return the results on separate lines along with accompanying text describing which mathematical calculation was performed.

LAB 06-B

1. A copy of LAB06-A was made.
2. The script was modified and is as follows:

```
1  #-----#
2  # Title: Lab06_B.py
3  # Desc: Script demonstrating Functions concept, based on Basic_Math.py (Assignment02)
4  # Change Log: (Who, When, What)
5  # DBiesinger, 2030-Jan-01, Created File
6  # DBiesinger, 2030-Jan-01, Modified to demonstrate SoC
7  # DBiesinger, 2030-Jan-01, Modified to demonstrate Functions
8  # aaleshin, 2022-Aug-13, Modified to demonstrate Attributes and Return Values
9  # aaleshin, 2022-Aug-13, Modified to demonstrate Tuple packing
10 #-----#
11
12 # -- DATA -- #
13 intNumA = None
14 intNumB = None
15
16 # -- PROCESSING -- #
17 # Process the data
18 def doAlgebra(intNumA, intNumB):
19     resultS = intNumA + intNumB
20     resultD = intNumA - intNumB
21     resultP = intNumA * intNumB
22     resultQ = intNumA / intNumB
23     return resultS, resultD, resultP, resultQ
24
25 # -- PRESENTATION (Input/Output) -- #
26 # Get User input data
27 print('Basic Math script. Calculating the Sum, Difference, Product and Quotient of two numbers.')
28 intNumA = int(input('Please enter the 1st Number: '))
29 intNumB = int(input('Please enter the 2nd number: '))
30
31 # -- PROCESSING -- #
32 # Run function with inputs and assign to Tuple
33 doAlgebra(intNumA, intNumB)
34 answerS, answerD, answerP, answerQ = doAlgebra(intNumA, intNumB)
35
36 # Display the Results
37 print('\n\nThis script calculated using the Numbers', intNumA, 'and', intNumB)
38 print('The Results are:\n')
39 print('Sum:\t\t', answerS, '\nDifference:\t', answerD)
40 print('Product:\t', answerP, '\nQuotient:\t', answerQ)
```

Figure 3: Lab 06-B Script.

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

```
In [1]: runfile('C:/_FDProgramming/Lab_06/Lab06_B.py', wdir='C:/_FDProgramming/Lab_06')
Basic Math script. Calculating the Sum, Difference, Product and Quotient of two numbers.
Please enter the 1st Number: 12
Please enter the 2nd number: 4

This script calculated using the Numbers 12 and 4
The Results are:

Sum:      16
Difference: 8
Product:   48
Quotient:  3.0
```

Figure 4: Lab 06-B Output.

- b. The script works by first initializing two variables that will be used throughout the rest of the code. One function is then defined to perform four algebraic calculations. The function returns four results in a single line corresponding to each algebraic operation. The user is then prompted to enter two values which then get assigned to the variables initialized in the beginning of the script. The two values then get plugged into the function and the four results get assigned to a Tuple. The script then returns each one of the values in the Tuple along with accompanying text describing which mathematical calculation was performed.

LAB 06-C

1. A copy of LAB06-A was made.
2. The script was modified per the instructions and is as follows:

```
1  #-----#
2  # Title: Lab06_C.py
3  # Desc: Script demonstrating Functions concept, based on Basic_Math.py (Assignment02)
4  # Change Log: (Who, When, What)
5  # DBiesinger, 2030-Jan-01, Created File
6  # DBiesinger, 2030-Jan-01, Modified to demonstrate SoC
7  # DBiesinger, 2030-Jan-01, Modified to demonstrate Functions
8  # aaleshin, 2022-Aug-13, Modified to demonstrate Attributes and Return Values
9  # aaleshin, 2022-Aug-13, Modified to demonstrate Classes and Docstrings
10 #-----#
11
12 # -- DATA -- #
13 intNumA = None
14 intNumB = None
15
16 # -- PROCESSING -- #
17 # Process the data
18 class SimpleMath():
19     """A collection of simple math processing functions"""
20
21     @staticmethod
22     def add_values(val1 = 0.0, val2 = 0.0):
23         """Function for adding two values
24
25         Args:
26             val1: the first number to add
27             val2: the second number to add
28
29         Returns:
30             A float corresponding to the sum of val1 and val2
31         """
32         return float(val1 + val2)
33
34     @staticmethod
35     def subtract_values(val1 = 0.0, val2 = 0.0):
36         """Function for subtracting two values
37
38         Args:
39             val1: the number to subtract from
40             val2: the number to subtract
41
42         Returns:
43             A float corresponding to the difference of val1 and val2
44         """
45         return float(val1 - val2)
46
47     @staticmethod
48     def multiply_values(val1 = 0.0, val2 = 0.0):
49         """Function for multiplying two values
50
51         Args:
52             val1: the first number to multiply
53             val2: the second number to multiply
54
55         Returns:
56             A float corresponding to the product of val1 and val2
57         """
58         return float(val1 * val2)
59
60     @staticmethod
61     def divide_values(val1 = 0.0, val2 = 0.0):
62         """Function for dividing two values
63
64         Args:
65             val1: the number to divide
66             val2: the number to divide by
67
68         Returns:
69             A float corresponding to the quotient of val1 and val2
70         """
71         return float(val1 / val2)
72
73 # -- PRESENTATION (Input/Output) -- #
74 # Get User input data
75 print('Basic Math script. Calculating the Sum, Difference, Product and Quotient of two numbers.')
76 intNumA = int(input('\nPlease enter the 1st Number: '))
77 intNumB = int(input('\nPlease enter the 2nd number: '))
78 # Display the Results
79 print('\n\nThis script calculated using the Numbers', intNumA, 'and', intNumB)
80 print('\n\nThe Results are:\n')
81 print('Sum:\t\t', SimpleMath.add_values(intNumA, intNumB), '\nDifference:\t', SimpleMath.subtract_values(intNumA, intNumB))
82 print('Product:\t\t', SimpleMath.multiply_values(intNumA, intNumB), '\nQuotient:\t', SimpleMath.divide_values(intNumA, intNumB))
```

Figure 5: Lab 06-C Script.

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

```
In [1]: runfile('C:/_FDProgramming/Lab_06/Lab06_C.py', wdir='C:/_FDProgramming/Lab_06')
Basic Math script. Calculating the Sum, Difference, Product and Quotient of two numbers.

Please enter the 1st Number: 12

Please enter the 2nd number: 4

This script calculated using the Numbers 12 and 4

The Results are:

Sum:          16.0
Difference:    8.0
Product:      48.0
Quotient:     3.0
```

Figure 6: Lab 06-C Output.

- b. The script works by first initializing two variables that will be used throughout the rest of the code. A class SimpleMath() is then created containing four functions with each function corresponding to one of four algebraic calculations. The class and each of the four functions contain docstrings that add descriptions about what the functions do. The functions all return a float. The user is then prompted to enter two values which then get assigned to the variables initialized in the beginning of the script. The two values then get plugged into each of the four functions and the four results get returned along with accompanying text describing which mathematical calculation was performed.

Assignment 06

1. Folder called Assignment06 was created.
2. The code was modified and is as follows:

```
1  #-----#
2  # Title: Assignment06_Starter.py
3  # Desc: Working with classes and functions.
4  # Change Log: (Who, When, What)
5  # DBiesinger, 2030-Jan-01, Created File
6  # aaleshin, 2022-Aug-13, Modified File to move code into functions under classes
7  # aaleshin, 2022-Aug-14, Modified File to include docstrings
8  #-----#
9
10 # -- DATA -- #
11 strChoice = '' # User input
12 lstTbl = [] # list of lists to hold data
13 dicRow = {} # list of data row
14 strFileName = 'CDInventory.txt' # data storage file
15 objFile = None # file object
16
17 file = open(strFileName, 'a+') # Creates CDInventory.txt if it does not exit.
18 file.close()
19
20 # -- PROCESSING -- #
21 class DataProcessor:
22     # TODO: add functions for processing here
23     """Manipulating the data in memory"""
24
25     @staticmethod
26     def delete_CD(table):
27         """Deletes CD from in memory table
28
29         Args:
30             table (list of dict): In memory 2D data structure (list of dicts) that holds the data during runtime
31
32         Returns:
33             None.
34         """
35         intRowNr = -1
36         blnCDRemoved = False
37         for row in table:
38             intRowNr += 1
39             if row['ID'] == intIDDel:
40                 del table[intRowNr]
41                 blnCDRemoved = True
42                 break
43         if blnCDRemoved:
44             print('The CD was removed')
45         else:
46             print('Could not find this CD!')
47
48     @staticmethod
49     def add_CD(table, ID, Title, Artist):
50         """Adds CD information to in memory table
51
52         Args:
53             table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
54             ID (String): String containing CD ID.
55             Title (String): String containing CD Title.
56             Artist (String): String containing CD Artist.
57
58         Returns:
59             None.
60         """
61         intID = int(ID)
62         dicRow = {'ID': intID, 'Title': Title, 'Artist': Artist}
63         table.append(dicRow)
64         IO.show_inventory(table)
65
66 class FileProcessor:
67     """Processing the data to and from text file"""
68
69     @staticmethod
70     def read_file(file_name, table):
71         """Function to manage data ingestion from file to a list of dictionaries
72
73         Reads the data from file identified by file_name into a 2D table
74         (list of dicts) table one line in the file represents one dictionary row in table.
75
76         Args:
77             file_name (string): name of file used to read the data from
78             table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
79
80         Returns:
```

```

81         None.
82     """
83     table.clear() # this clears existing data and allows to load data from file
84     objFile = open(file_name, 'r')
85     for line in objFile:
86         data = line.strip().split(',')
87         dicRow = {'ID': int(data[0]), 'Title': data[1], 'Artist': data[2]}
88         table.append(dicRow)
89     objFile.close()
90
91     @staticmethod
92     def write_file(file_name, table):
93         # TODO: Add code here
94         """Function to manage writing data from table to file
95
96         Writes data from 2D table (list of dicts) to file one row at a time.
97
98         Args:
99             file_name (string): name of file used to read the data from
100             table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
101
102         Returns:
103             None.
104         """
105         objFile = open(file_name, 'w')
106         for row in table:
107             lstValues = list(row.values())
108             lstValues[0] = str(lstValues[0])
109             objFile.write(','.join(lstValues) + '\n')
110         objFile.close()
111
112 # -- PRESENTATION (Input/Output) -- #
113
114 class IO:
115     """Handling Input / Output"""
116
117     @staticmethod
118     def print_menu():
119         """Displays a menu of choices to the user
120
121         Args:
122             None.
123
124         Returns:
125             None.
126         """
127
128         print('Menu\n\n[L] Load Inventory from file\n[a] Add CD\n[i] Display Current Inventory')
129         print('[d] delete CD from Inventory\n[s] Save Inventory to file\n[x] exit\n')
130
131     @staticmethod
132     def menu_choice():
133         """Gets user input for menu selection
134
135         Args:
136             None.
137
138         Returns:
139             choice (string): a lower case string of the users input out of the choices l, a, i, d, s or x
140
141         """
142         choice = ' '
143         while choice not in ['l', 'a', 'i', 'd', 's', 'x']:
144             choice = input('Which operation would you like to perform? [l, a, i, d, s or x]: ').lower().strip()
145         print() # Add extra space for layout
146         return choice
147
148     @staticmethod
149     def show_inventory(table):
150         """Displays current inventory table
151
152         Args:
153             table (list of dict): 2D data structure (list of dicts) that holds the data during runtime.
154
155         Returns:
156             None.
157
158         """
159

```



```

160         print('==== The Current Inventory: =====')
161         print('ID\tCD Title (by: Artist)\n')
162         for row in table:
163             print('{ }\t{ } (by: { })'.format(*row.values()))
164         print('=====')
165
166     # TODO add I/O functions as needed
167
168
169     @staticmethod
170     def get_ID():
171         """Gets user input for CD ID.
172
173         Args:
174             None.
175
176         Returns:
177             Returns the CD ID as a string.
178         """
179         ID = input('Enter ID: ').strip()
180         return ID
181
182     @staticmethod
183     def get_Title():
184         """Gets user input for CD Title.
185
186         Args:
187             None.
188
189         Returns:
190             Returns the CD Title as a string.
191         """
192         Title = input('What is the CD\'s title? ').strip()
193         return Title
194
195     @staticmethod
196     def get_Artist():
197         """Gets user input for CD Artist.
198
199         Args:
200             None.
201
202         Returns:
203             Returns the CD Artist as a string.
204         """
205         Artist = input('What is the Artist\'s name? ').strip()
206         return Artist
207
208     # 1. When program starts, read in the currently saved Inventory
209     FileProcessor.read_file(strFileName, lstTbl)
210
211     # 2. start main loop
212     while True:
213         # 2.1 Display Menu to user and get choice
214         IO.print_menu()
215         strChoice = IO.menu_choice()
216
217         # 3. Process menu selection
218         # 3.1 process exit first
219         if strChoice == 'x':
220             break
221         # 3.2 process load inventory
222         if strChoice == 'l':
223             print('WARNING: If you continue, all unsaved data will be lost and the Inventory re-loaded from file.')
224             strYesNo = input('type \'yes\' to continue and reload from file. otherwise reload will be canceled')
225             if strYesNo.lower() == 'yes':
226                 print('reloading...')
227                 FileProcessor.read_file(strFileName, lstTbl)
228                 IO.show_inventory(lstTbl)
229             else:
230                 input('canceling... Inventory data NOT reloaded. Press [ENTER] to continue to the menu.')
231                 IO.show_inventory(lstTbl)
232             continue # start loop back at top.
233         # 3.3 process add a CD
234         elif strChoice == 'a':
235             # 3.3.1 Ask user for new ID, CD Title and Artist
236             strID = IO.get_ID()
237             strTitle = IO.get_Title()
238             strArtist = IO.get_Artist()
239             # 3.3.2 Add item to the table

```

```

240         # ToDone move processing code into function
241         DataProcessor.add_CD(lstTbl, strID, strTitle, strArtist)
242         continue # start loop back at top.
243     # 3.4 process display current inventory
244     elif strChoice == 'i':
245         IO.show_inventory(lstTbl)
246         continue # start loop back at top.
247     # 3.5 process delete a CD
248     elif strChoice == 'd':
249         # 3.5.1 get Userinput for which CD to delete
250         # 3.5.1.1 display Inventory to user
251         IO.show_inventory(lstTbl)
252         # 3.5.1.2 ask user which ID to remove
253         intIDDel = int(input('Which ID would you like to delete? ').strip())
254         # 3.5.2 search thru table and delete CD
255         # ToDone move processing code into function
256         DataProcessor.delete_CD(lstTbl)
257         IO.show_inventory(lstTbl)
258         continue # start loop back at top.
259     # 3.6 process save inventory to file
260     elif strChoice == 's':
261         # 3.6.1 Display current inventory and ask user for confirmation to save
262         IO.show_inventory(lstTbl)
263         strYesNo = input('Save this inventory to file? [y/n] ').strip().lower()
264         # 3.6.2 Process choice
265         if strYesNo == 'y':
266             # 3.6.2.1 save data
267             # ToDone move processing code into function
268             FileProcessor.write_file(strFileName, lstTbl)
269         else:
270             input('The inventory was NOT saved to file. Press [ENTER] to return to the menu.')
271             continue # start loop back at top.
272     # 3.7 catch-all should not be possible, as user choice gets vetted in IO, but to be save:
273     else:
274         print('General Error')

```

Figure 7: Assignment 06 Script

3. The script was tested:

a. In Spyder:

```

In [1]: runfile('C:/_FDProgramming/Assignment06/CDInventory.py', wdir='C:/_FDProgramming/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 canceledyes
reloading...
===== The Current Inventory: =====
ID  CD Title (by: Artist)

=====
Menu

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

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

Enter ID: 1
What is the CD's title? Ten
What is the Artist's name? Pearl Jam
===== 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
[d] delete CD from Inventory
[s] Save Inventory to file
[x] exit

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

Enter ID: 2
What is the CD's title? Nevermind
What is the Artist's name? Nirvana
===== 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
[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   Ten (by:Pearl Jam)
2   Nevermind (by:Nirvana)
=====
Menu

```

```

===== 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
[d] delete CD from Inventory
[s] Save Inventory to file
[x] exit

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

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

1   Ten (by:Pearl Jam)
2   Nevermind (by:Nirvana)
=====
Which ID would you like to delete? 1
The CD was removed
===== The Current Inventory: =====
ID  CD Title (by: Artist)

2   Nevermind (by:Nirvana)
=====
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)

2   Nevermind (by:Nirvana)
=====
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)

2   Nevermind (by:Nirvana)
=====
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

```

Figure 8: Assignment 06 Spyder Output.

b. In Anaconda Prompt:

```
(base) C:\_FDPProgramming\Assignment06>python CDInventory.py
Menu

[1] 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? [1, a, i, d, s or x]: 1

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)

2      Nevermind (by:Nirvana)
=====
Menu

[1] 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? [1, a, i, d, s or x]: a

Enter ID: 1
What is the CD's title? Ten
What is the Artist's name? Pearl Jam
===== The Current Inventory: =====
ID      CD Title (by: Artist)

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

[1] 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? [1, a, i, d, s or x]: d

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

2      Nevermind (by:Nirvana)
1      Ten (by:Pearl Jam)
=====
Which ID would you like to delete? 2
The CD was removed
===== The Current Inventory: =====
ID      CD Title (by: Artist)

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

[1] 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? [1, a, i, d, s or x]: s

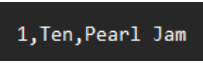
===== The Current Inventory: =====
ID      CD Title (by: Artist)

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

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

Figure 9: Assignment 06 Anaconda Prompt Output

4. The final inventory after running the script in Spyder and Anaconda Prompt is as follows:



```
1,Ten,Pearl Jam
```

Figure 10: Assignment 06 Resulting CDInventory.txt

In order to perform the assignment, I first read through the code and took note that there are three different classes. I then started with the code backwards, first moving the portion of the script that saves the 2D Table into the `write_file` function in the `FileProcessor` class. Next, I moved the script that deletes a row from the 2D Table based on ID to its own function in the `DataProcessor` class. I then created 3 functions under the `IO` class that request the user to input the ID, Title and Artist of the CD, respectively and return the strings containing the information. I then moved the portion of the script that adds a new CD to its own function under the `DataProcessor` class. For each function that was created, I made sure to properly call it in the sections where the script for the function was moved from. Lastly, I wrote the docstrings for each of the functions and confirmed that the code works both in Spyder and Anaconda Prompt.

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

https://github.com/aaleshinUW/Assignment_06

Summary

The document covers the Points of Interest, the Laboratories, and the Assignment from Module 06. By completing Module 06, I learned about Functions, Classes, DocStrings and Variable Scope.