|  |  |
| --- | --- |
| Name | Bruck Assefa |
| Date | 2021-Feb-02 |
| Course | IT FDN 110: Introduction to Programming (Python) |
| Assignment | Assignment 06 |

Assignment 06

# Introduction

This document covers steps involved in taking a script and changing scripts to a class and function format. In this document we will show the script and function call for IO.add\_inventory, FileProcessor. write\_file and DataProcessor.delete\_inventory and demonstrate the results.

The IDE we will be using is Spyder. We will also be working with Python 3.8.5 version as shown on the below console.

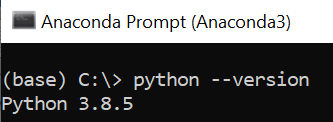


Figure 1 "Python version" script

# Create a Python Script

We will create and test a python script file that performs the following:

1. Create classes for data processing, file processing and IO (Input/Out put).
2. Following the TODO instruction take scripts and move them to functions.
3. Test your work by running the script.

First open a note pad document and write down the below scripts.

1. #------------------------------------------#
2. # Title: Assignment06\_Starter.py
3. # Desc: Working with classes and functions.
4. # Change Log: (Who, When, What)
5. # ABruck, 2021-Feb-2, Created File
6. #------------------------------------------#
8. # -- DATA -- #
9. strChoice = '' # User input
10. lstTbl = []  # list of lists to hold data
11. dicRow = {}  # list of data row
12. strFileName = 'CDInventory.txt'  # data storage file
13. objFile = None  # file object

16. # -- PROCESSING -- #
17. **class** DataProcessor:
18. **def** delete\_inventory(table):
19. # 3.5.1 get Userinput for which CD to delete
20. # 3.5.1.1 display Inventory to user
21. IO.show\_inventory(table)
22. # 3.5.1.2 ask user which ID to remove
23. intIDDel = int(input('Which ID would you like to delete? ').strip())
24. # 3.5.2 search thru table and delete CD
25. intRowNr = -1
26. blnCDRemoved = False
27. **for** row **in** table:
28. intRowNr += 1
29. **if** row['ID'] == intIDDel:
30. **del** table[intRowNr]
31. blnCDRemoved = True
32. **break**
33. **if** blnCDRemoved:
34. **print**('The CD was removed')
35. **else**:
36. **print**('Could not find this CD!')
37. IO.show\_inventory(table)
38. # 3.6 process save inventory to file
40. **class** FileProcessor:
41. """Processing the data to and from text file"""
43. @staticmethod
44. **def** read\_file(file\_name, table):
45. """Function to manage data ingestion from file to a list of dictionaries
47. Reads the data from file identified by file\_name into a 2D table
48. (list of dicts) table one line in the file represents one dictionary row in table.
50. Args:
51. file\_name (string): name of file used to read the data from
52. table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
54. Returns:
55. None.
56. """
57. table.clear()  # this clears existing data and allows to load data from file
58. objFile = open(file\_name, 'r')
59. **for** line **in** objFile:
60. data = line.strip().split(',')
61. dicRow = {'ID': int(data[0]), 'Title': data[1], 'Artist': data[2]}
62. table.append(dicRow)
63. objFile.close()
65. @staticmethod
66. **def** write\_file(file\_name, table):
67. # 3.6.1 Display current inventory and ask user for confirmation to save
68. IO.show\_inventory(table)
69. strYesNo = input('Save this inventory to file? [y/n] ').strip().lower()
70. # 3.6.2 Process choice
71. **if** strYesNo == 'y':
72. # 3.6.2.1 save data
73. objFile = open(file\_name, 'w')
74. **for** row **in** lstTbl:
75. lstValues = list(row.values())
76. lstValues[0] = str(lstValues[0])
77. objFile.write(','.join(lstValues) + '\n')
78. objFile.close()
79. **else**:
80. input('The inventory was NOT saved to file. Press [ENTER] to return to the menu.')

83. # -- PRESENTATION (Input/Output) -- #
85. **class** IO:
86. """Handling Input / Output"""
88. @staticmethod
89. **def** print\_menu():
90. """Displays a menu of choices to the user
92. Args:
93. None.
95. Returns:
96. None.
97. """
99. **print**('Menu\n\n[l] load Inventory from file\n[a] Add CD\n[i] Display Current Inventory')
100. **print**('[d] delete CD from Inventory\n[s] Save Inventory to file\n[x] exit\n')
102. @staticmethod
103. **def** menu\_choice():
104. """Gets user input for menu selection
106. Args:
107. None.
109. Returns:
110. choice (string): a lower case sting of the users input out of the choices l, a, i, d, s or x
112. """
113. choice = ' '
114. **while** choice **not** **in** ['l', 'a', 'i', 'd', 's', 'x']:
115. choice = input('Which operation would you like to perform? [l, a, i, d, s or x]: ').lower().strip()
116. **print**()  # Add extra space for layout
117. **return** choice
119. @staticmethod
120. **def** show\_inventory(table):
121. """Displays current inventory table

124. Args:
125. table (list of dict): 2D data structure (list of dicts) that holds the data during runtime.
127. Returns:
128. None.
130. """
131. **print**('======= The Current Inventory: =======')
132. **print**('ID\tCD Title (by: Artist)\n')
133. **for** row **in** table:
134. **print**('{}\t{} (by:{})'.format(\*row.values()))
135. **print**('======================================')
137. **def** add\_inventory():
138. # 3.3.1 Ask user for new ID, CD Title and Artist
139. strID = input('Enter ID: ').strip()
140. strTitle = input('What is the CD\'s title? ').strip()
141. stArtist = input('What is the Artist\'s name? ').strip()
143. # 3.3.2 Add item to the table
144. intID = int(strID)
145. dicRow = {'ID': intID, 'Title': strTitle, 'Artist': stArtist}
146. lstTbl.append(dicRow)
147. IO.show\_inventory(lstTbl)
149. # 1. When program starts, read in the currently saved Inventory
150. FileProcessor.read\_file(strFileName, lstTbl)
152. # 2. start main loop
153. **while** True:
154. # 2.1 Display Menu to user and get choice
155. IO.print\_menu()
156. strChoice = IO.menu\_choice()
158. # 3. Process menu selection
159. # 3.1 process exit first
160. **if** strChoice == 'x':
161. **break**
162. # 3.2 process load inventory
163. **if** strChoice == 'l':
164. **print**('WARNING: If you continue, all unsaved data will be lost and the Inventory re-loaded from file.')
165. strYesNo = input('type \'yes\' to continue and reload from file. otherwise reload will be canceled')
166. **if** strYesNo.lower() == 'yes':
167. **print**('reloading...')
168. FileProcessor.read\_file(strFileName, lstTbl)
169. IO.show\_inventory(lstTbl)
170. **else**:
171. input('canceling... Inventory data NOT reloaded. Press [ENTER] to continue to the menu.')
172. IO.show\_inventory(lstTbl)
173. **continue**  # start loop back at top.
174. # 3.3 process add a CD
175. **elif** strChoice == 'a':
176. IO.add\_inventory()
177. # 3.4 process display current inventory
178. **elif** strChoice == 'i':
179. IO.show\_inventory(lstTbl)
180. **continue**  # start loop back at top.
181. # 3.5 process delete a CD
182. **elif** strChoice == 'd':
183. DataProcessor.delete\_inventory(lstTbl)
184. **elif** strChoice == 's':
185. FileProcessor.write\_file(strFileName, lstTbl)
186. **continue**  # start loop back at top.
187. # 3.7 catch-all should not be possible, as user choice gets vetted in IO, but to be save:
188. **else**:
189. **print**('General Error')

Figure 2: Assignment06.py script command lines

# Save your script to a folder

Inside the course folder create another folder called Assignment06 and save the script document as “Assignment06.py” and also create a CDInventory.txt file.

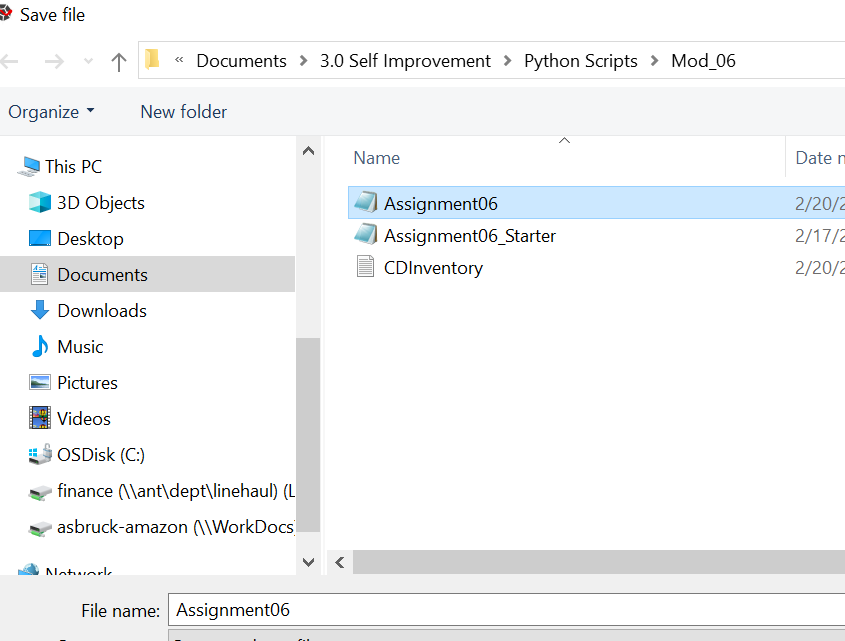


Figure 3 saving a .py extension file

# Run and Test Your Script

1. Test the IO.add\_inventory function.

We have created an IO.add\_inventory() function.

Script

The below script for the IO.add\_inventory() function will get executed

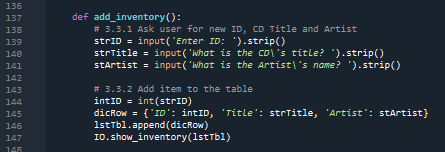


Figure 4 IO.add\_inventory function syntax

Function call



Figure 5 IO.add\_inventory function call

Result

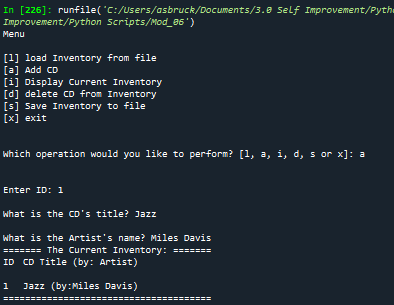


Figure 6 Script was successful for add\_inventory funtion

1. Test the FileProcessor. write\_file function.

We have created an IO.add\_inventory() function. As shown in the below image CDInventory.txt file was able to save the id, title, and artist name that we added previously. When we added the cd information it was captured in the temporary memory. Executing the FileProcessor.write\_file function was able to save it in the text file.

Script

The below script for the FileProcessor.write\_file function will get executed

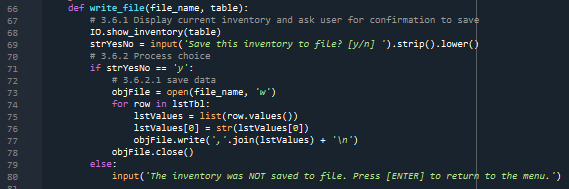


Figure 7 FileProcessor.write\_file function syntax

Function call



Figure 8 FileProcessor.write\_file function call

Result

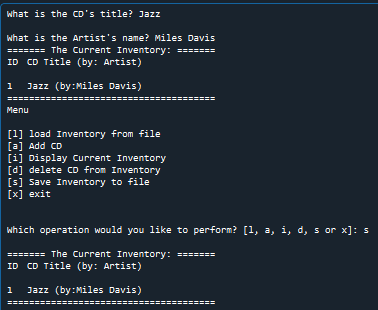


Figure 9 Script was successful for FileProcessor.write\_file

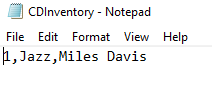


Figure 10 result evidence for FileProcessor.write\_file

1. Test the DataProcessor.delete\_inventory function.

We have created an DataProcessor.delete\_inventory function.

Script

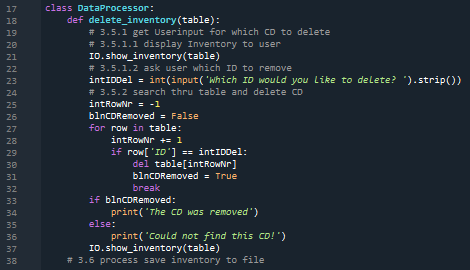


Figure 11 DataProcessor.delete\_inventory script

Function call



Figure 12 DataProcessor.delete\_inventory function call

Result

The below image shows running the DataProcessor.delete\_inventory function call and evidence of the CDInventory.txt file.

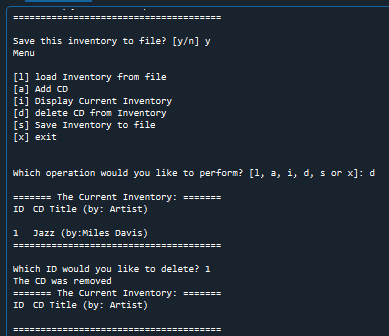
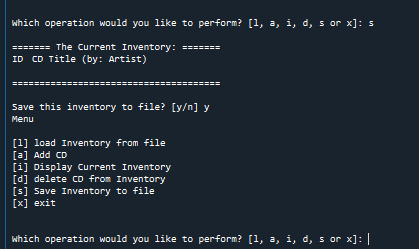
 

Figure 13 successful deletion of the cd information Figure 14 Saving the deletion to text file

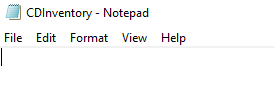


Figure 15 Saving the deletion to text file

# Summary

In this assignment I have learned about

* Creating a class and function. Working with code provided by others.
* Using positional variables in a function.
* Most of all I have also learned that taking the time to test the result of the code executed in a strategic way gives more confidence and helps to spot any errors made.

On my assignment05 I gave up after my codes were not executing as I intended it to work. Nevertheless, taking pieces of the code and executing it one step at a time and seeing what is working and what is returning errors helps to troubleshoot any errors in the code. I felt more confident after all the three functions executed in the way it was required.

# Reference

* FDN\_Py\_Module\_06 file from IT FDN 110 A Module 04 Overview materials

<https://saravji.github.io/saravjis_hut/FDN_Prog/Modules.html#module-06-materials-list>