Kelsey Sorge-Toomey  
2020 Aug 19  
IT FDN 110 B Su 20: Foundations Of Programming: Python  
Assignment 06

Module 05 Assignment 06

# Introduction

The goal of the sixth week’s homework was to continue learning about functions and introduce Classes while also learning about Variable Scope and DocString.

This week was easier for me than the previous weeks. I had already been familiar with functions and introduced to classes. I also was familiar with variable scope and global vs local variables.

# Assignment 06:

I started by moving the code for adding user data, deleting entry, and saving an entry into the DataProcessor class (Figure 1) and moving the code for the user entry to the IO Class (Figure 2)

A screenshot of a cell phone

Description automatically generated

Figure - Data processing functions

A screenshot of a cell phone

Description automatically generated

Figure - user\_entry() moved to IO class

I tried running through the code but was getting an error when it started running because FileProcessor.read\_file() was trying to read a file that didn’t exist yet (Figure 3). I realized I needed code that can create the .txt file when the program initializes if there isn’t already CDInventory.txt in the folder location. Referencing my code from CDInventory.py from Module 05, I added in the code for the write\_file function. After adding it, I realized I only needed it to create and close the file so I could remove everything from it except objFile = open(strFileName, ‘a’) and objFile.close() (Figure 4). I then added FileProcessor.write\_file() to initialize before FileProcessor.read\_file() (Figure 5).

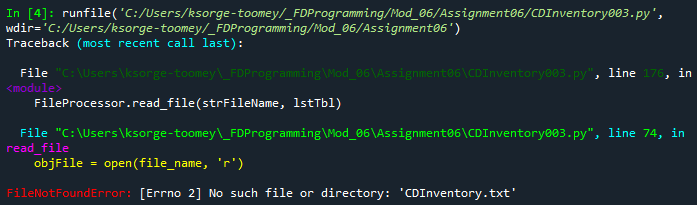


Figure - Error when trying to run

A screenshot of a cell phone

Description automatically generated

Figure - Write\_file function

A picture containing orange, holding, person, player

Description automatically generated

Figure - Added FileProcessor.write\_file()

Looking at the save\_entry() function, I realized that since it interacts with saving data to the .txt, it should be moved to the FileProcessor class (Figure 6).

A screenshot of a cell phone screen with text

Description automatically generated

Figure - Save\_entry function moved to FileProcessor Class

# Final Code

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. # KSorge-Toomey. 2020-Aug-19, Turned some tasks from while loops into functions within the appropriate class, Added docstrings
7. #------------------------------------------#
9. # -- DATA -- #
10. strChoice = '' # User input
11. lstTbl = []  # list of lists to hold data
12. dicRow = {}  # list of data row
13. strFileName = 'CDInventory.txt'  # data storage file
14. objFile = None  # file object

17. # -- PROCESSING -- #
18. **class** DataProcessor:
19. """Handling data in memory"""
21. @staticmethod
22. def add\_user\_data():
23. """Function to receive **new** entry data
25. Args:
26. None.
28. Returns:
29. None.
30. """
31. intID = **int**(strID)
32. dicRow = {'ID': intID, 'Title': strTitle, 'Artist': stArtist}
33. lstTbl.append(dicRow)
35. @staticmethod
36. def delete\_entry():
37. """Function to **delete** entry chosen by ID number
39. Args:
40. None
42. Returns:
43. None.
44. """
45. intRowNr = -1
46. blnCDRemoved = False
47. **for** row in lstTbl:
48. intRowNr += 1
49. **if** row['ID'] == intIDDel:
50. del lstTbl[intRowNr]
51. blnCDRemoved = True
52. **break**
53. **if** blnCDRemoved:
54. print('The CD was removed')
55. **else**:
56. print('Could not find this CD!')


60. **class** FileProcessor:
61. """Processing the data to and from text file"""
63. @staticmethod
64. def read\_file(file\_name, table):
65. """Function to manage data ingestion from file to a list of dictionaries
67. Reads the data from file identified by file\_name into a 2D table
68. (list of dicts) table one line in the file represents one dictionary row in table.
70. Args:
71. file\_name (string): name of file used to read the data from
72. table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
74. Returns:
75. None.
76. """
77. table.clear()  # **this** clears existing data and allows to load data from file
78. objFile = open(file\_name, 'r')
79. **for** line in objFile:
80. data = line.strip().split(',')
81. dicRow = {'ID': **int**(data[0]), 'Title': data[1], 'Artist': data[2]}
82. table.append(dicRow)
83. objFile.close()
85. @staticmethod
86. def write\_file(file\_name):
87. """Function to create CDInventory.txt **if** file does not already exist
89. Args:
90. file\_name (string):name of file to create .txt
92. Returns:
93. None.
94. """
95. objFile = open(strFileName, 'a')
96. objFile.close()

99. @staticmethod
100. def save\_entry():
101. """Function to save current entries in memory to file
103. Args:
104. None.
106. Returns:
107. None.
108. """
109. **if** strYesNo == 'y':
110. # 3.6.2.1 save data
111. objFile = open(strFileName, 'w')
112. **for** row in lstTbl:
113. lstValues = list(row.values())
114. lstValues[0] = str(lstValues[0])
115. objFile.write(','.join(lstValues) + '\n')
116. objFile.close()
117. **else**:
118. input('The inventory was NOT saved to file. Press [ENTER] to return to the menu.')

121. # -- PRESENTATION (Input/Output) -- #
123. **class** IO:
124. """Handling Input / Output"""
126. @staticmethod
127. def print\_menu():
128. """Displays a menu of choices to the user
130. Args:
131. None.
133. Returns:
134. None.
135. """
137. print('Menu\n\n[l] Load Inventory from file\n[a] Add CD\n[i] Display Current Inventory')
138. print('[d] Delete CD from Inventory\n[s] Save Inventory to file\n[x] Exit\n')
140. @staticmethod
141. def menu\_choice():
142. """Gets user input **for** menu selection
144. Args:
145. None.
147. Returns:
148. choice (string): a lower **case** sting of the users input out of the choices l, a, i, d, s or x
150. """
151. choice = ' '
152. **while** choice not in ['l', 'a', 'i', 'd', 's', 'x']:
153. choice = input('Which operation would you like to perform? [l, a, i, d, s or x]: ').lower().strip()
154. print()  # Add extra space **for** layout
155. **return** choice
157. @staticmethod
158. def show\_inventory(table):
159. """Displays current inventory table

162. Args:
163. table (list of dict): 2D data structure (list of dicts) that holds the data during runtime.
165. Returns:
166. None.
168. """
169. print('======= The Current Inventory: =======')
170. print('ID\tCD Title (by: Artist)\n')
171. **for** row in table:
172. print('{}\t{} (by:{})'.format(\*row.values()))
173. print('======================================')
175. @staticmethod
176. def user\_entry():
177. """Gets user input **for** CD entry
179. Args:
180. None.
182. Returns:
183. None.
184. """
185. global strID
186. global strTitle
187. global stArtist
188. strID = input('Enter ID: ').strip()
189. strTitle = input('What is the CD\'s title? ').strip()
190. stArtist = input('What is the Artist\'s name? ').strip()

193. # 1. When program starts,
194. FileProcessor.write\_file(strFileName)
195. FileProcessor.read\_file(strFileName, lstTbl)
197. # 2. start main loop
198. **while** True:
199. # 2.1 Display Menu to user and get choice
200. IO.print\_menu()
201. strChoice = IO.menu\_choice()
203. # 3. Process menu selection
204. # 3.1 process exit first
205. **if** strChoice == 'x':
206. **break**
207. # 3.2 process load inventory
208. **if** strChoice == 'l':
209. print('WARNING: If you continue, all unsaved data will be lost and the Inventory re-loaded from file.')
210. strYesNo = input('Type \'yes\' to continue and reload from file. otherwise reload will be canceled: ')
211. **if** strYesNo.lower() == 'yes':
212. print('Reloading...')
213. FileProcessor.read\_file(strFileName, lstTbl)
214. IO.show\_inventory(lstTbl)
215. **else**:
216. input('Canceling... Inventory data NOT reloaded. Press [ENTER] to continue to the menu.')
217. IO.show\_inventory(lstTbl)
218. **continue**  # start loop back at top.
219. # 3.3 process add a CD
220. elif strChoice == 'a':
221. # 3.3.1 Ask user for new ID, CD Title and Artist
222. IO.user\_entry()
223. # 3.3.2 Add item to the table
224. DataProcessor.add\_user\_data()
225. # 3.3.3 Display modified inventory to user
226. IO.show\_inventory(lstTbl)
227. **continue**  # start loop back at top.
228. # 3.4 process display current inventory
229. elif strChoice == 'i':
230. IO.show\_inventory(lstTbl)
231. **continue**  # start loop back at top.
232. # 3.5 process delete a CD
233. elif strChoice == 'd':
234. # 3.5.1 get Userinput for which CD to delete
235. # 3.5.1.1 display Inventory to user
236. IO.show\_inventory(lstTbl)
237. # 3.5.1.2 ask user which ID to remove
238. intIDDel = **int**(input('Which ID would you like to delete? ').strip())
239. # 3.5.2 search thru table and delete CD
240. DataProcessor.delete\_entry()
241. # 3.5.3 Display modified inventory to user
242. IO.show\_inventory(lstTbl)
243. **continue**  # start loop back at top.
244. # 3.6 process save inventory to file
245. elif strChoice == 's':
246. # 3.6.1 Display current inventory and ask user for confirmation to save
247. IO.show\_inventory(lstTbl)
248. strYesNo = input('Save this inventory to file? [y/n] ').strip().lower()
249. # 3.6.2 Process choice
250. FileProcessor.save\_entry()
251. **continue**  # start loop back at top.
252. # 3.7 catch-all should not be possible, as user choice gets vetted in IO, but to be safe:
253. **else**:
254. print('General Error')

# Testing and Output

A screenshot of a cell phone

Description automatically generated

Figure - Testing loading data from existing CDInventory.txt file

A screenshot of a cell phone

Description automatically generated

Figure - Testing deleting entry from inventory

A screenshot of a cell phone

Description automatically generated

Figure - Testing deleting entry from inventory part 2

A screen shot of a computer

Description automatically generated

Figure - Saving data in memory to CDInventory.txt file

A picture containing bird

Description automatically generated

Figure - Updated saved data in CDInventory.txt

A screenshot of a cell phone

Description automatically generated

Figure - Testing adding additional entries

A screen shot of a computer

Description automatically generated

Figure - Testing adding additional entries part 2

A screen shot of a computer

Description automatically generated

Figure - Testing saving updated data to CDInventory.txt

A picture containing bird

Description automatically generated

Figure - Updated saved data in CDInventory.txt

A screen shot of a computer

Description automatically generated

Figure - Testing exiting the program

I also tested out running the program without CDInventory.txt in the folder and the program worked correctly to create CDInventory.txt.

# Summary

In this week, I continued to learn more about functions and classes as well as variable scope and docstrings. I was already familiar with functions, classes, and variable scope and this week allowed me to learn more about these topics. Applying this new knowledge, I successfully completed this week’s homework assignment.