Nitasha Woodward

December 10, 2021

IT FDN 110 B Au 21: Foundations of Programming: Python

Assignment 08

Working with Software Objects

# Introduction

In this assignment I will explain the steps I used to update the pseudocode in the starter file provided for creating a CD Inventory program using objects. This assignment involved using code from previous assignments 06 and 07 and modifying them to work with a list of objects instead of a list of dictionaries, and reading/writing to and from a text file.

# Creating the Code

### Reusing Previous Code

I first looked at the pseudocode in the starter file to figure out what code I could reuse from previous assignments. I copied the code for the *load\_inventory* method from Assignment 06 *read\_file* method and updated to work with a list of objects instead of a list of dictionaries. I didn’t use Assignment 07 as the starting point, since that was working with binary files instead of text files. I also copied code for:

* Load\_inventory
  + Copied code from *read\_file* method in Assignment 06 and modified to work with the arguments provided in the starter code, as well as to work with a CD object instead of a dictionary.
* Save\_inventory
  + Coped code from *write\_file* method in Assignment 06 and modified to use the dunder string method (\_\_str\_\_) to neatly write CD object data to the text file.
* The DataProcessing class – The requirement to add this section was a little confusing at first because this class was completely missing and not addressed in the starter code. I spent time trying to figure out if this code belonged somewhere else in the starter code, before deciding to copy over as its own class, similar to prior assignments.
  + add\_cd: I simplified this method by just appending the cd object to the list. No other formatting was needed here.
  + del\_cd: This was one of the most difficult parts of the program to rewrite. Line 171 was hard to figure out how I could access the ID part of the CD object.
* The IO class
  + print\_menu: Copied with no change.
  + menu\_choice: no change but I did struggle with spacing/tabs when I copied + pasted the code.
  + show\_inventory: updated to remove the format and .values that was used to work with the dictionaries
  + get\_cd: no change

### Creating New Code

I had to create new code in the CD class. I started by first defining the attributes that I wanted each instance of a CD object to have: ID, title, artist. I made these attributes private using a \_\_ prefix. I then defined properties for each of these attributes so that the user could set/get them without interacting directly with the attributes (although I did not end up using these… not sure how this would work with the user?).

I also created a \_\_str\_\_ method to nicely format the content of my CD objects for printing to the text file.

The format of the main code was setup similar to prior assignments, with a few updates to work with objects instead of dictionaries.

# Saving the Script

After writing and testing my code in Spyder using the starter code, I navigated to the folder in my Home Folder called ‘Python’, in which I created a folder for ‘Assignment08’ and saved my text file as CD\_Inventory.py.

# Running the Script

Once I saved my final script, I ran the program successful in Spyder as well as my Terminal.

Text

Description automatically generated

Figure - Testing in Spyder [1 of 4]

Text

Description automatically generated

Figure - Testing in Spyder [2 of 4]

Text

Description automatically generated

Figure - Testing in Spyder [3 of 4]

Graphical user interface, text

Description automatically generated

Figure - Testing in Spyder [4 of 4]

Text

Description automatically generated

Figure - Testing in Terminal [1 of 3]

Text

Description automatically generated

Figure - Testing in Terminal [2 of 3]

Text

Description automatically generated

Figure - Testing in Terminal [3 of 3]

# Summary

After completing the assigned reading, labs, and videos in Module 08 I was able to modify the pseudocode starter file to a working program that uses software objects for CD to create a CD Inventory program. I have posted the assignment .py file and this document to GitHub for peer review:

# Appendix

## <https://highlight.hohli.com/index.php> (external[[1]](#footnote-1)) web page

1. *#------------------------------------------#*
2. *# Title: CD\_Inventory.py*
3. *# Desc: Assignnment 08 - Working with classes*
4. *# Change Log: (Who, When, What)*
5. ***# DBiesinger, 2030-Jan-01, created file***
6. *# DBiesinger, 2030-Jan-01, added pseudocode to complete assignment 08*
7. *# NWoodward, 2021-Dec-10, added code to create CD Inventory using classes*
8. *# NWoodward, 2021-Dec-11, changed file title, review error handling, fix delete method*
9. *#------------------------------------------#*
11. *# -- DATA -- #*
12. strFileName = 'cdInventory.txt' *#text storage file*
13. lstOfCDObjects = []
15. **class CD:**
16. """Stores data about a CD:
18. properties:
19. cd\_id: (int) with CD ID
20. **cd\_title: (string) with the title of the CD**
21. cd\_artist: (string) with the artist of the CD
22. methods:
24. """
25. ***#--Fields--#***
26. cd\_id = None
27. cd\_title = ''
28. cd\_artist = ''
30. ***#--Constructor--#***
31. **def** \_\_init\_\_(self, cdID: int, title: str, artist: str):
32. """Instantiate a new CD with ID, title, and artist values"""
33. *#---Attributes---#*
34. self.\_\_cd\_id = int(cdID)
35. **self.\_\_cd\_title = str(title)**
36. self.\_\_cd\_artist = str(artist)
38. *#--Properties--#*
39. *# CD ID*
40. **@property**
41. **def** cd\_id(self):
42. **return** self.\_\_cd\_id
44. @cd\_id.setter
45. **def cd\_id(self, cdID):**
46. **if** str(cdID).isnumeric():
47. self.\_\_cd\_id = int(cdID)
48. **else**:
49. **raise** Exception('CD ID must be a number.')
51. *# CD Title*
52. @property
53. **def** cd\_title(self):
54. **return** self.\_\_cd\_title
56. @cd\_title.setter
57. **def** cd\_title(self, title):
58. **if** str(title).isnumeric():
59. **raise** Exception('Please enter a string, not a number.')
60. **else:**
61. self.\_\_cd\_title = str(title)
63. *# CD Artist*
64. @property
65. **def cd\_artist(self):**
66. **return** self.\_\_cd\_artist
68. @cd\_artist.setter
69. **def** cd\_artist(self, artist):
70. **if str(artist).isnumeric():**
71. **raise** Exception('Please enter a string, not a number.')
72. **else**:
73. self.\_\_cd\_artist = str(artist)
75. ***#--Methods--#***
76. **def** \_\_str\_\_(self):
77. """Format CD object data into a readable string output"""
78. **return** '{}, {}, {}**\n**'.format(self.\_\_cd\_id, self.\_\_cd\_title, self.\_\_cd\_artist)
80. ***# -- PROCESSING -- #***
81. **class** FileIO:
82. """Processes data to and from a text file:
84. properties:
86. methods:
87. save\_inventory(file\_name, lst\_Inventory): -> None
88. load\_inventory(file\_name, lst\_Inventory): -> (a list of CD objects)
90. **"""**
92. *#--Methods--#*
93. @staticmethod
94. **def** load\_inventory(file\_name: str, lst\_Inventory: list) -> list:
95. **"""Method to process data from a text file to a list of CD objects.**
97. Args:
98. file\_name (string): name of file used to read data from
99. lst\_Inventory (list of objects): list that holds CD objects
101. Returns:
102. lst\_Inventory (list of objects): list that holds CD objects
103. """
104. lst\_Inventory = []
105. **try:**
106. **with** open(file\_name, 'r') **as** objFile:
107. **for** line **in** objFile:
108. data = line.strip().split(',')
109. cd = CD(data[0],data[1],data[2])
110. **lst\_Inventory.append(cd)**
111. **except** EOFError **as** e:
112. **print**('File is blank. Please add CD**\'**s')
113. lst\_Inventory = []
114. **finally**:
115. **return lst\_Inventory**
117. @staticmethod
118. **def** save\_inventory(file\_name: str, lst\_Inventory: list) -> None:
119. """Method to write data from a list of objects to a text file.
121. Writes data from the list identified by lst\_Inventory into a text file
122. identified by file\_name.
124. Args:
125. **file\_name (string): name of file used to read data from**
126. lst\_Inventory (list of objects): list that holds CD objects
128. Returns:
129. None.
130. **"""**
131. **try**:
132. **with** open(file\_name, 'w') **as** objFile:
133. **for** obj **in** lst\_Inventory:
134. cd = obj.\_\_str\_\_()
135. **objFile.write(cd)**
136. **except** Exception **as** e:
137. **print**(type(e))
139. **class** DataProcessor:
140. **"""Methods for processing data"""**
142. **def** add\_cd(cd, lst\_Inventory: list):
143. """ Method that allows user to add a CD to the inventory in memory. The CD must be
144. saved, choice 's', in order for the CD to be written to a text file.
146. Args:
147. cd (object): CD object
148. lst\_Inventory (list of objects): list that holds CD objects
150. **Returns:**
151. lst\_Inventory (list of objects): list that holds CD objects
152. """
153. lst\_Inventory.append(cd)
154. **return** lst\_Inventory
156. @staticmethod
157. **def** del\_cd(lst\_Inventory, cdID):
158. """Method that allows user to delete a CD from the inventory in memory.
160. **Args:**
161. lst\_Inventory (list of objects): list that holds CD objects
162. cdID: ID number of the CD the user would like to delete
164. Returns:
165. **message: Message to the user to tell them if their attempt to delete a CD was successful**
166. """
167. intRowNr = -1
168. blnCDRemoved = False
169. **for** row **in** lst\_Inventory:
170. **intRowNr += 1**
171. **if** row.cd\_id == cdID:
172. **del** lst\_Inventory[intRowNr]
173. blnCDRemoved = True
174. **break**
175. **if blnCDRemoved:**
176. message = **print**('The CD was removed')
177. **else**:
178. message = **print**('Could not find this CD!')
179. **return** message
181. *# -- PRESENTATION (Input/Output) -- #*
182. **class** IO:
183. """ Handling Input/Output"""
185. ***#--Methods--#***
186. @staticmethod
187. **def** print\_menu():
188. """Displays a menu of choices to the user.
190. **Args:**
191. None
192. Returns:
193. None.
194. """
195. **print('Menu\n\n[l] load Inventory from file\n[a] Add CD\n[i] Display Current Inventory')**
196. **print**('[d] delete CD from Inventory**\n**[s] Save Inventory to file**\n**[x] exit**\n**')
198. @staticmethod
199. **def** menu\_choice():
200. **"""Gets user input for menu selection**
202. Args:
203. None.
205. **Returns:**
206. choice (string): a lower case string of the users input out of the choices l, a, i, d, s or x
207. """
208. choice = ' '
209. **while** choice **not** **in** ['l', 'a', 'i', 'd', 's', 'x']:
210. **choice = input('Which operation would you like to perform? [l, a, i, d, s or x]: ').lower().strip()**
211. **print**() *# Add extra space for layout*
212. **return** choice
214. @staticmethod
215. **def show\_inventory(lst\_Inventory: list) -> None:**
216. """Displays current inventory to the screen
218. Args:
219. lst\_Inventory (list of objects): list that holds CD objects
221. Returns:
222. None.
223. """
224. **print**('======= The Current Inventory: =======')
225. **print('ID\tCD Title (by: Artist)\n')**
226. **for** row **in** lst\_Inventory:
227. **print**(row)
228. **print**('======================================')
230. **@staticmethod**
231. **def** get\_cd():
232. """ Method to enable user to add a new CD to in memory to a list of CD objects
234. Args:
235. **None.**
237. Returns:
238. cdID (string): ID number entered by user
239. artist (string): Artist name input by user
240. **title (string): CD title input by user**
242. """
243. **try**:
244. cdID = input('Enter ID: ').strip()
245. **cdID = int(cdID)**
246. title = input('What is the CD**\'**s title? ').strip()
247. artist = input('What is the Artist**\'**s name? ').strip()
248. **return** cdID, title, artist
249. **except** ValueError **as** e:
250. **print('Please enter a whole number. You entered {}'.format(cdID))**
251. **print**(type(e))
253. *# -- Main Body of Script -- #*
254. *# Load data from file into a list of CD objects on script start*
255. **try:**
256. lstOfCDObjects = FileIO.load\_inventory(strFileName,lstOfCDObjects)
257. **except** Exception **as** e:
258. **print**(type(e))

261. **while** True:
262. *# Display menu to user*
263. IO.print\_menu()
264. choice = IO.menu\_choice()
266. *# let user exit program*
267. **if** choice == 'x':
268. **break**
270. ***# show user current inventory***
271. **if** choice == 'i':
272. **try**:
273. IO.show\_inventory(lstOfCDObjects)
274. **except** TypeError **as** e:
275. **print('The CD Inventory is blank. Please add content.')**
276. **print**(type(e))
277. **finally**:
278. **continue** *#start loop back at top*
280. ***# let user add data to the inventory***
281. **elif** choice == 'a':
282. **try**:
283. cdID, title, artist = IO.get\_cd()
284. cd = CD(cdID, title, artist)
285. **lstOfCDObjects = DataProcessor.add\_cd(cd,lstOfCDObjects)**
286. IO.show\_inventory(lstOfCDObjects)
287. **except** Exception **as** e:
288. **print**(type(e))
289. **finally**:
290. **continue *#start loop back at top***
292. *# let user save inventory to file*
293. **elif** choice == 's':
294. **try**:
295. **IO.show\_inventory(lstOfCDObjects)**
296. strYesNo = input('Save this inventory to file? [y/n] ').strip().lower()
297. **if** strYesNo == 'y':
298. FileIO.save\_inventory(strFileName, lstOfCDObjects)
299. **else**:
300. **input('The inventory was NOT saved to file. Press [ENTER] to return to the menu.')**
301. **except** Exception **as** e:
302. **print**(type(e))
303. **finally**:
304. **continue** *# start loop back at top.*
306. *# let user load inventory from file*
307. **elif** choice == 'l':
308. **print**('WARNING: If you continue, all unsaved data will be lost and the Inventory re-loaded from file.')
309. strYesNo = input('type **\'**yes**\'** to continue and reload from file. otherwise reload will be canceled: ')
310. **if strYesNo.lower() == 'yes':**
311. **print**('reloading...')
312. lstOfCDObjects = FileIO.load\_inventory(strFileName,lstOfCDObjects)
313. IO.show\_inventory(lstOfCDObjects)
314. **else**:
315. **input('canceling... Inventory data NOT reloaded. Press [ENTER] to continue to the menu.')**
316. IO.show\_inventory(lstOfCDObjects)
317. **continue** *# start loop back at top.*
319. *# let user delete data from inventory*
320. **elif choice == 'd':**
321. IO.show\_inventory(lstOfCDObjects)
322. cdID = int(input('Which Id would you like to delete?').strip())
323. DataProcessor.del\_cd(lstOfCDObjects, cdID)
324. **print**() *## Add extra space for layout*
325. **IO.show\_inventory(lstOfCDObjects)**
326. **continue**
328. **else**:
329. **print**('General Error')

1. Accessed December 10, 2021 [↑](#footnote-ref-1)