Kelsey Sorge-Toomey  
2020 Sept 3  
IT FDN 110 B Su 20: Foundations Of Programming: Python  
Assignment 08

Module 08 Assignment 08

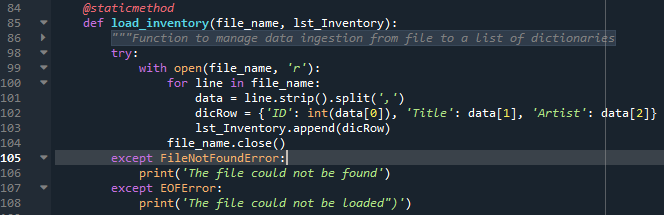
# Introduction

The goal of the seventh’s week’s homework was to expolore Object Oriented Programming. We learned how to use classes to define objects, create methods and attributes for objects and instantiate objects.

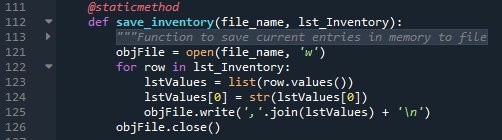
This week was another difficult one for me, but I tried really hard to understand the new concepts that were introduced even if that meant turning it in late.

# Assignment 08:

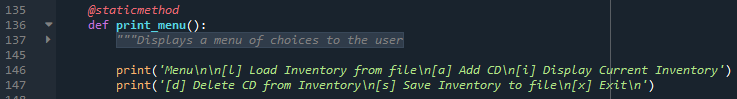
I started by grabbing some functions from a corrected version of Assignment 06 and Assignment 07. This included load\_inventory (Figure 1) that grabs data from a .txt file, save\_inventory (Figure 2) that saves data out to a .txt file, print\_menu (Figure 3), menu\_choice Figure 4), show\_inventory (Figure 5) to show the inventory back to the user, and user\_entry (Figure 6) to accept user input.



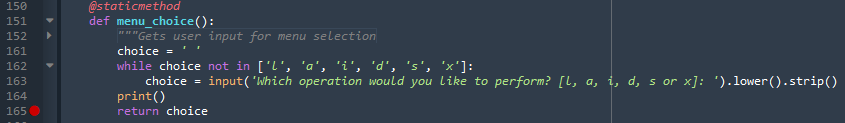
Figure



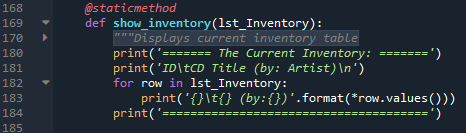
Figure



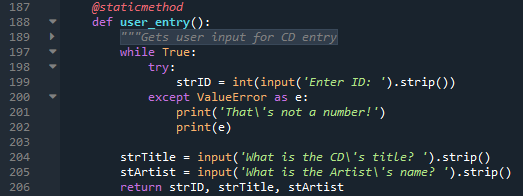
Figure



Figure

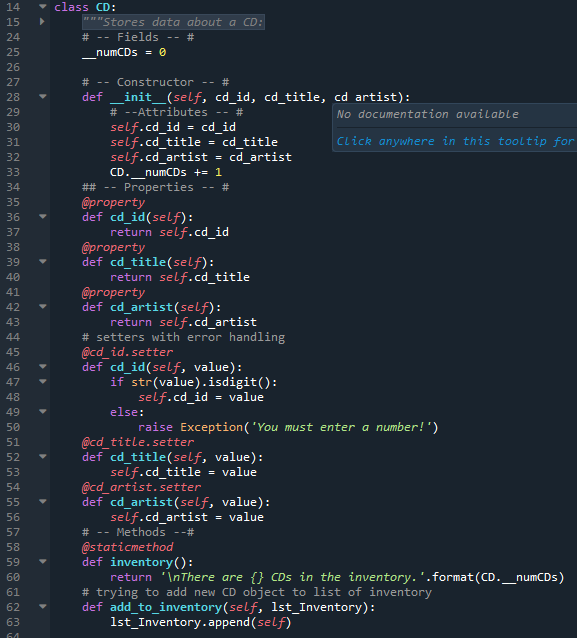


Figure



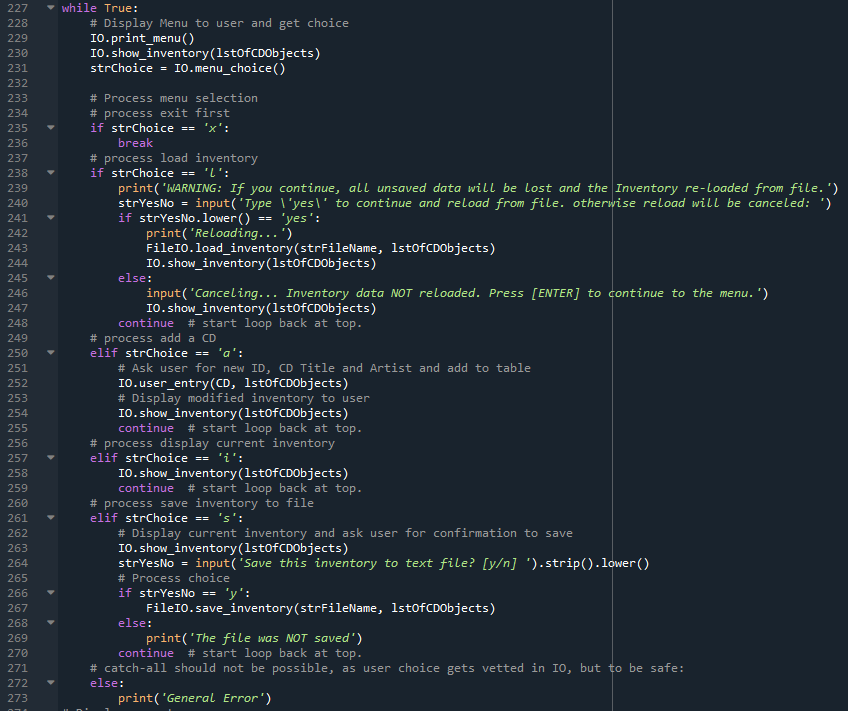
Figure

Next, I did the first pass on creating the CD class (Figure 7). I included attributes for the CD ID, CD Title, and CD Artist. I included error handling if the user enters a non-integer for the CD ID. I also included a method for saving the CD object to the inventory list.



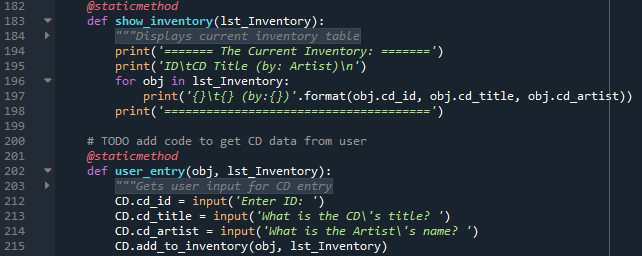
Figure

I also copied over the main body of the while loop from a corrected version of assignment 6, updating variables where needed and removing the delete option (Figure 8).



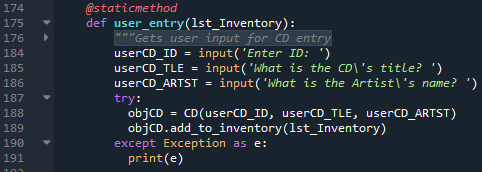
Figure

I circled back to the IO and FileIO functions to make sure that they were updated for handling the CD objects and its attributes. I changed the formatting for IO.show\_inventory to appropriately display the attributes for CD objects and updated IO.user\_entry to update class CD attributes with user input (Figure 9). However, I realized that is what changing the attributes for all CD objects because it wasn’t instantiating a new object to be updated.



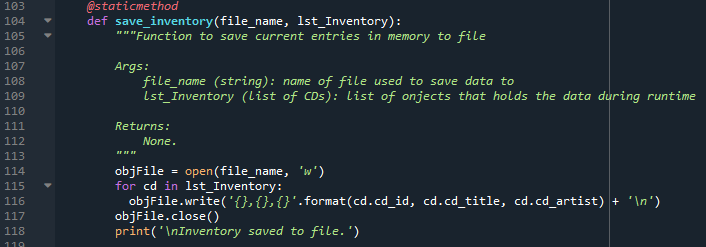
Figure

I changed IO.user\_entry take inputs from the user, then to try assigning those to a new CD, which was then appended to the list inventory using the method from the CD class (Figure 10). Then if the user entered a non-integer for the CD ID, the exception error from CD class would pop once it attempted to add the data to the list inventory.



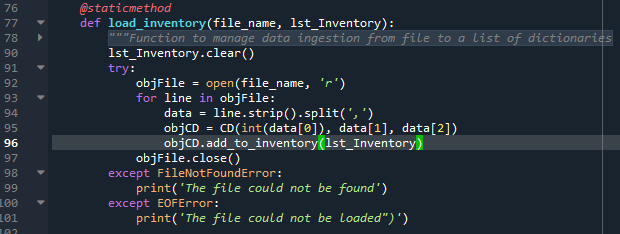
Figure

FileIO.load\_inventory and FileIO.save\_inventory were especially challenging for me to figure out the proper way to format writing the data to the file and reading the data from the file. For save\_inventory, I realized I was over thinking it and needed to use similar formatting to IO.show\_inventory (Figure 11). That way, the CD ID, CD Title, and CD Artist would be saved to the text file separated by commas, similar to the way the CD inventory was save in Assignment06.



Figure

Once I had FileIO.save\_inventory, it was a matter of more or less reversing it to get FileIO.load\_inventory to work. With adding try except, it also prevented the program from crashing if there wasn’t a text file yet.



Figure

# Final Code

1. #------------------------------------------#
2. # Title: Assignment08.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. # KSorge-Toomey, 2020-Aug-31, added functions, CD class
8. # KSorge-Toomey, 2020-Sept-2, edited functions and CD class
9. # KSorge-Toomey, 2020-Sept-3, polished code
10. #------------------------------------------#
12. # -- DATA -- #
13. strFileName = 'cdInventory.txt'
14. lstOfCDObjects = []
16. **class** CD:
17. """Stores data about a CD:
19. properties:
20. cd\_id: (int) with CD ID
21. cd\_title: (string) with the title of the CD
22. cd\_artist: (string) with the artist of the CD
23. methods:
24. inventory:
25. """
26. # -- Fields -- #
27. \_\_cd\_id = 0
28. \_\_cd\_title = ''
29. \_\_cd\_artist = ''
31. # -- Constructor -- #
32. **def** \_\_init\_\_(self, ID, title, artist):
33. # --Attributes -- #
34. self.cd\_id = ID
35. self.cd\_title = title
36. self.cd\_artist = artist
38. ## -- Properties -- #
39. @property
40. **def** cd\_id(self):
41. **return** self.\_\_cd\_id
42. @property
43. **def** cd\_title(self):
44. **return** self.\_\_cd\_title
45. @property
46. **def** cd\_artist(self):
47. **return** self.\_\_cd\_artist
48. @cd\_id.setter
49. **def** cd\_id(self, value):
50. **if** str(value).isdigit():
51. self.\_\_cd\_id = value
52. **else**:
53. **raise** Exception('You must enter a number!')
54. @cd\_title.setter
55. **def** cd\_title(self, value):
56. self.\_\_cd\_title = value
57. @cd\_artist.setter
58. **def** cd\_artist(self, value):
59. self.\_\_cd\_artist = value
61. # -- Methods --#
62. **def** add\_to\_inventory(self, lst\_Inventory):
63. lst\_Inventory.append(self)

66. # -- PROCESSING -- #
67. **class** FileIO:
68. """Processes data to and from file:
70. properties:
72. methods:
73. save\_inventory(file\_name, lst\_Inventory): -> None
74. load\_inventory(file\_name): -> (a list of CD objects)
75. """
77. @staticmethod
78. **def** load\_inventory(file\_name, lst\_Inventory):
79. """Function to manage data ingestion from file to a list of CDs
81. Reads the data from file identified by file\_name into a
82. (list of CDs) one line in the file represents one CD in list.
84. Args:
85. file\_name (string): name of file used to read the data from
86. lst\_Inventory (list of CDs): list of objects that holds the data during runtime
88. Returns:
89. None.
90. """
91. lst\_Inventory.clear()
92. **try**:
93. objFile = open(file\_name, 'r')
94. **for** line **in** objFile:
95. data = line.strip().split(',')
96. objCD = CD(int(data[0]), data[1], data[2])
97. objCD.add\_to\_inventory(lst\_Inventory)
98. objFile.close()
99. **except** FileNotFoundError:
100. **print**('The file could not be found')
101. **except** EOFError:
102. **print**('The file could not be loaded")')
104. @staticmethod
105. **def** save\_inventory(file\_name, lst\_Inventory):
106. """Function to save current entries in memory to file
108. Args:
109. file\_name (string): name of file used to save data to
110. lst\_Inventory (list of CDs): list of onjects that holds the data during runtime
112. Returns:
113. None.
114. """
115. objFile = open(file\_name, 'w')
116. **for** cd **in** lst\_Inventory:
117. objFile.write('{},{},{}'.format(cd.cd\_id, cd.cd\_title, cd.cd\_artist) + '\n')
118. objFile.close()
119. **print**('\nInventory saved to file.')

122. # -- 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**('\nMenu\n\n[l] Load Inventory from file\n[a] Add CD\n[i] Display Current Inventory')
138. **print**('[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', 's', 'x']:
153. choice = input('Which operation would you like to perform? [l, a, i, s or x]: ').lower().strip()
154. **print**()
155. **return** choice
157. @staticmethod
158. **def** show\_inventory(lst\_Inventory):
159. """Displays current inventory table

162. Args:
163. lst\_Inventory (list of CDs): list of objects that holds the data during runtime.
165. Returns:
166. None.
168. """
169. **print**('\n======= The Current Inventory: =======')
170. **print**('ID\tCD Title (by: Artist)\n')
171. **for** objCD **in** lst\_Inventory:
172. **print**('{}\t{} (by:{})'.format(objCD.cd\_id, objCD.cd\_title, objCD.cd\_artist))
173. **print**('======================================')
175. @staticmethod
176. **def** user\_entry(lst\_Inventory):
177. """Gets user input for CD entry
179. Args:
180. lst\_Inventory (list of CDs): list of objects that hold the data during runtime
182. Returns:
183. None:
184. """
185. userCD\_ID = input('Enter ID: ')
186. userCD\_TLE = input('What is the CD\'s title? ')
187. userCD\_ARTST = input('What is the Artist\'s name? ')
188. **try**:
189. objCD = CD(userCD\_ID, userCD\_TLE, userCD\_ARTST)
190. objCD.add\_to\_inventory(lst\_Inventory)
191. **except** Exception as e:
192. **print**(e)

195. # -- Main Body of Script -- #
197. # Load data from file into a list of CD objects on script start
198. FileIO.load\_inventory(strFileName, lstOfCDObjects)
199. IO.show\_inventory(lstOfCDObjects)
201. **while** True:
202. # Display Menu to user and get choice
203. IO.print\_menu()
204. strChoice = IO.menu\_choice()
206. # Process menu selection
207. # process exit first
208. **if** strChoice == 'x':
209. **break**
210. # process load inventory
211. **if** strChoice == 'l':
212. **print**('WARNING: If you continue, all unsaved data will be lost and the Inventory re-loaded from file.')
213. strYesNo = input('Type \'yes\' to continue and reload from file. Otherwise reload will be canceled: ')
214. **if** strYesNo.lower() == 'yes':
215. **print**('Reloading...')
216. FileIO.load\_inventory(strFileName, lstOfCDObjects)
217. IO.show\_inventory(lstOfCDObjects)
218. **else**:
219. input('Canceling... Inventory data NOT reloaded. Press [ENTER] to continue to the menu.')
220. IO.show\_inventory(lstOfCDObjects)
221. **continue**  # start loop back at top.
222. # process add a CD
223. **elif** strChoice == 'a':
224. # Ask user for new ID, CD Title and Artist and add to table
225. IO.user\_entry(lstOfCDObjects)
226. # Display modified inventory to user
227. IO.show\_inventory(lstOfCDObjects)
228. **continue**  # start loop back at top.
229. # process display current inventory
230. **elif** strChoice == 'i':
231. IO.show\_inventory(lstOfCDObjects)
232. **continue**  # start loop back at top.
233. # process save inventory to file
234. **elif** strChoice == 's':
235. # Display current inventory and ask user for confirmation to save
236. IO.show\_inventory(lstOfCDObjects)
237. strYesNo = input('Save this inventory to text file? [y/n] ').strip().lower()
238. # Process choice
239. **if** strYesNo == 'y':
240. FileIO.save\_inventory(strFileName, lstOfCDObjects)
241. **else**:
242. **print**('The file was NOT saved')
243. **continue**  # start loop back at top.
244. # catch-all should not be possible, as user choice gets vetted in IO, but to be safe:
245. **else**:
246. **print**('General Error')

# Running

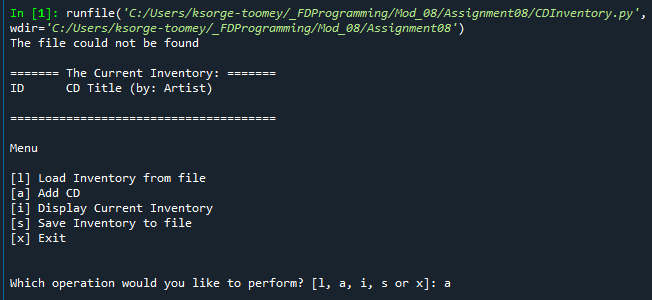


Figure - Spyder output, .txt file not found to load inventory

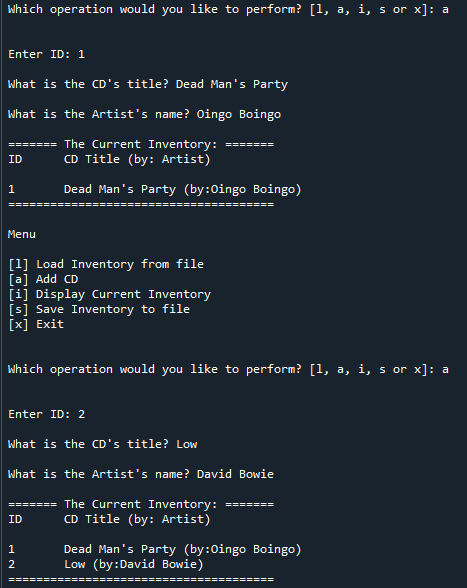


Figure - Added entries to CD inventory

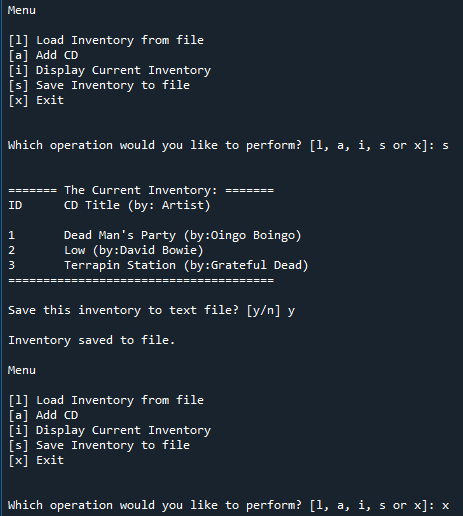


Figure - Saving inventory to .txt file and exiting

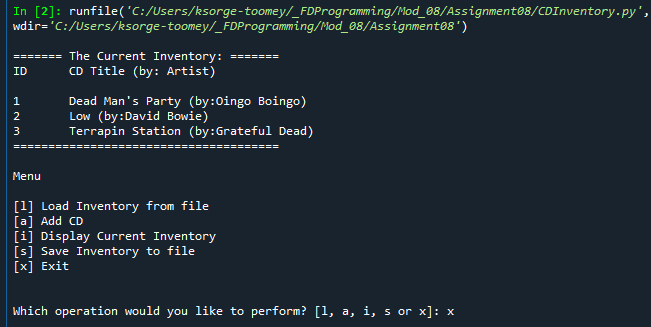


Figure - Starting program and loading inventory from .txt file

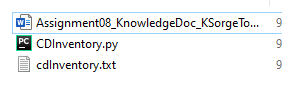


Figure - Saved .txt file

# Summary

In this week, I learned about Object Oriented Programming, including how to add attributes and methods, creating new classes, instantiating new objects and restricting access to object’s attributes. Applying this new knowledge, I successfully completed this week’s homework assignment.