Gavin Chan

03/17/2020

IT FDN 100

Assignment 08

# Introduction

In this week’s module and assignment, we went more in depth into classes and how they are critical to OOP, or object oriented programming.

# Assignment 08: Modifying the CDInventory Script

For Assignment 8, we were tasked with the classic CD Inventory program.

This script was modified from last week’s script with the addition of classes, as well as constructors and methods. This helped streamline the CD information collection steps of the code. Figure 1 is a snippet of the new classes, constructors, and methods:

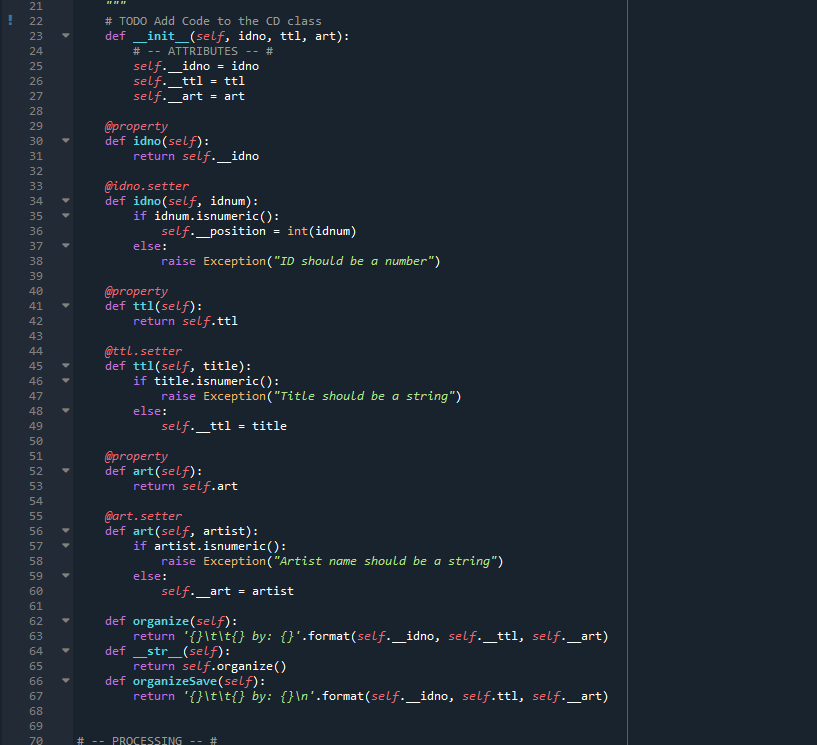


Figure - New Classes, Constructors, and Methods

A lot of the code could be copied from the previous week’s assignment. However, one of the major changes can be found in the way we append the CD information to the table with our new classes. Below is a snippet of the additional code that tackles this:

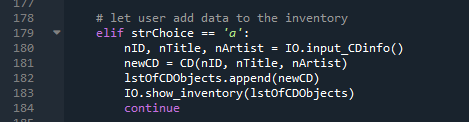


Figure - New Append Code

Because this code is too long to include in one snippet, the code will be found in the appendix.

# Saving the Python File

The python script was saved in the following directory:

C:\\_FDNProgramming\Assignment08

The code itself was named Assignment08.py. The text file with the results is named cdInventory.txt, as named by the script.

The link to the Github is as follows:

<https://github.com/NotGavin/Assignment08/blob/master/CDInventory.py>

# Summary

In this module, we learned about the importance of classes and constructors as they tie into OOP, or Object Oriented Program. Destructors were also touched upon in the module, but were not required for the assignment this week.

# Appendix

Assignment 08 Script:

1. #------------------------------------------#
2. # Title: Assignmen08.py
3. # Desc: Assignnment 08 - Working with classes
4. # Change Log: (Who, When, What)
5. # GChan, 2020-Mar-15, created file
6. #------------------------------------------#
8. # -- DATA -- #
9. strFileName = 'cdInventory.txt'
10. lstOfCDObjects = []
12. **class** CD():
13. """Stores data about a CD:
15. properties:
16. cd\_id: (int) with CD ID
17. cd\_title: (string) with the title of the CD
18. cd\_artist: (string) with the artist of the CD
19. methods:
21. """
22. # TODO Add Code to the CD class
23. **def** \_\_init\_\_(self, idno, ttl, art):
24. # -- ATTRIBUTES -- #
25. self.\_\_idno = idno
26. self.\_\_ttl = ttl
27. self.\_\_art = art
29. @property
30. **def** idno(self):
31. **return** self.\_\_idno
33. @idno.setter
34. **def** idno(self, idnum):
35. **if** idnum.isnumeric():
36. self.\_\_position = int(idnum)
37. **else**:
38. **raise** Exception("ID should be a number")
40. @property
41. **def** ttl(self):
42. **return** self.ttl
44. @ttl.setter
45. **def** ttl(self, title):
46. **if** title.isnumeric():
47. **raise** Exception("Title should be a string")
48. **else**:
49. self.\_\_ttl = title
51. @property
52. **def** art(self):
53. **return** self.art
55. @art.setter
56. **def** art(self, artist):
57. **if** artist.isnumeric():
58. **raise** Exception("Artist name should be a string")
59. **else**:
60. self.\_\_art = artist
62. **def** organize(self):
63. **return** '{}\t\t{} by: {}'.format(self.\_\_idno, self.\_\_ttl, self.\_\_art)
64. **def** \_\_str\_\_(self):
65. **return** self.organize()
66. **def** organizeSave(self):
67. **return** '{}\t\t{} by: {}\n'.format(self.\_\_idno, self.ttl, self.\_\_art)

70. # -- PROCESSING -- #
71. **class** FileIO:
72. """Processes data to and from file:
74. properties:
76. methods:
77. save\_inventory(file\_name, lst\_Inventory): -> None
78. load\_inventory(file\_name): -> (a list of CD objects)
80. """
81. # TODO Add code to process data from a file
82. **def** read\_file(file\_name, lst\_Inventory):
83. objFile = open(file\_name, 'r')
84. **for** line **in** objFile:
85. data = line.strip().split(',')
86. cd = CD(int(data[0]),data[1],data[2])
87. lst\_Inventory.append(cd)
88. objFile.close()
89. # TODO Add code to process data to a file
90. **def** write\_file(file\_name, table):
91. objFile = open(file\_name, 'w')
92. **for** cd **in** table:
93. objFile.write(cd.organizeSave())
94. objFile.close()
96. # -- PRESENTATION (Input/Output) -- #
97. **class** IO:
98. # TODO add docstring
99. # TODO add code to show menu to user
100. @staticmethod
101. **def** print\_menu():
102. """Displays a menu of choices to the user
104. Args:
105. None.
107. Returns:
108. None.
109. """
111. **print**('Menu\n\n[l] load Inventory from file\n[a] Add CD\n[i] Display Current Inventory')
112. **print**('[s] Save Inventory to file\n[x] exit\n')
113. # TODO add code to captures user's
114. @staticmethod
115. **def** menu\_choice():
116. """Gets user input for menu selection
118. Args:
119. None.
121. Returns:
122. choice (string): a lower case sting of the users input out of the choices l, a, i, s or x
124. """
125. choice = ' '
126. **while** choice **not** **in** ['l', 'a', 'i', 's', 'x']:
127. choice = input('Which operation would you like to perform? [l, a, i, s or x]: ').lower().strip()
128. **print**()  # Add extra space for layout
129. **return** choice
130. # TODO add code to display the current data on screen
131. @staticmethod
132. **def** show\_inventory(table):
133. """Displays current inventory table

136. Args:
137. table (list of dict): 2D data structure (list of dicts) that holds the data during runtime.
139. Returns:
140. None.
142. """
143. **print**('======= The Current Inventory: =======')
144. **print**('ID\tCD Title (by: Artist)\n')
145. **for** CD **in** table:
146. **print**(CD)
147. **print**('======================================')
148. # TODO add code to get CD data from user
149. @staticmethod
150. **def** input\_CDinfo():
151. """Gets CD information as input from user
153. Args:
154. none
156. Returns:
157. intID, StrTitle, strArtist
159. """
160. intID = int(input('Enter ID: ').strip())
161. strTitle = input('What is the CD\'s title? ').strip()
162. strArtist = input('What is the Artist\'s name? ').strip()
163. **return** intID, strTitle, strArtist
165. # -- Main Body of Script -- #
166. # TODO Add Code to the main body
167. # Load data from file into a list of CD objects on script start
168. # Display menu to user
169. **while** True:
170. # show user current inventory
171. IO.print\_menu()
172. strChoice = IO.menu\_choice()
174. # let user exit program
175. **if** strChoice == 'x':
176. **break**
178. # let user add data to the inventory
179. **elif** strChoice == 'a':
180. nID, nTitle, nArtist = IO.input\_CDinfo()
181. newCD = CD(nID, nTitle, nArtist)
182. lstOfCDObjects.append(newCD)
183. IO.show\_inventory(lstOfCDObjects)
184. **continue**
186. # let user save inventory to file
187. **elif** strChoice == 's':
188. IO.show\_inventory(lstOfCDObjects)
189. strYesNo = input('Save this inventory to file? [y/n] ').strip().lower()
190. **if** strYesNo == 'y':
191. FileIO.write\_file(strFileName, lstOfCDObjects)
192. **else**:
193. input('The inventory was NOT saved to file. Press [ENTER] to return to the menu.')
194. **continue**
196. # let user load inventory from file
197. **elif** strChoice == 'l':
198. **print**('WARNING: If you continue, all unsaved data will be lost and the Inventory re-loaded from file.')
199. strYesNo = input('type \'yes\' to continue and reload from file. otherwise reload will be canceled')
200. **if** strYesNo.lower() == 'yes':
201. **print**('reloading...')
202. lstOfCDObjects = []
203. FileIO.read\_file(strFileName, lstOfCDObjects)
204. IO.show\_inventory(lstOfCDObjects)
205. **else**:
206. input('canceling... Inventory data NOT reloaded. Press [ENTER] to continue to the menu.')
207. IO.show\_inventory(lstOfCDObjects)
208. **continue**