Rogger Tovar

12/6/2021

Foundations of Python

Assignment 08

CD Inventory OOP

# Introduction

In this assignment, we are to add Object Oriented Programming to the CD Inventory. This might be my favorite assignment since we started. I liked the challenge that we had to write the code from scratch but with a little guidance. The pseudocode helped us see exactly where we needed to add functions. It did make it a little easier that the classes were added.

# CD Inventory

When I first started this process of the code, I copy and pasting the code; that was a mistake. So, I continued to then add everything myself, which was a challenge but a fun one. I added statements as a result of the end user making a selected option. I ran into a couple of issues but quickly noticed and resolved the issue, Figure 1.

Text

Description automatically generated

Figure 1 – easy issue found.

Though it took me a while, I was utterly surprised I was able to run my code pretty quickly, figure 2. Now it was a matter of making the menu work. Since the start of the code displays an error stating the file has not been created, I quickly added “a+” to append the file if none exists. Once the file portion was resolved, the menu appeared.

Graphical user interface

Description automatically generated

Figure 2 Menu displays

Graphical user interface

Description automatically generated

Figure 3 File successfully created with excitement.

Text

Description automatically generated

Idle 1 Spelling errors

In some instances, I was not paying attention to the console at all. I would just notice the error and try to fix it without reading what the issue was. After, quickly paying more attention and not trying to speed run this assignment, I was able to move on to the next phase – applying what we learned in module 9 pdf.

# Trying to create objects in CDInventory

Text

Description automatically generated

Idle 2 All menus providing a response.

In the image above (Idle2) all the menus are interacting as it should, except the add option. I can have the user manually enter the three values, but I cannot make the collective values to equal an object. I can create an object outside the defined classes and encapsulations, but not in defined function. This was the real challenge and after many hours of trying, I threw in the towel. I honestly think I was really close to figuring out the my final result. There could have been one more method I may have needed to add.

Text

Description automatically generated

Idle 3 Trying to add to the list, fail.

In “Idle 3” I can finally add the entries, but I cannot store it. Here is where I believe my missing piece may lie in another function, but I wouldn’t know how to apply it; but it feels so close.

# Summary

This was my favorite project. It felt like everything I learned and applying it from scratch, which I think ok but not complete. In the end, I probably needed the “\_\_str\_\_” and another function, but again, I couldn’t figure it out. I like to think I did great in the labs and really understood what was happening in the examples. Again, I really like this assignment. Nothing felt too stressful and felt ideas flowing in the code without issue or seriousness. Of course, I forgot to add additional information, like the docstrings.

# Link to github

https://github.com/roggergt/Assignment08/

# Other resources

A couple resources to elaborate the methodology of OOB.

<https://www.youtube.com/watch?v=jCzT9XFZ5bw>

<https://www.youtube.com/watch?v=AEOuYv699K4>

# Appendix

Straight copy from Anaconda

1. *#------------------------------------------#*
2. *# Title: Assignment08.py*
3. *# Desc: Assignment 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. *# RTovar, 2021-Dec-05, added menu*
8. *#------------------------------------------#*
9. *# -- DATA -- #*
10. *strFileName = 'cdInventory.txt'*
11. *lstOfCDObjects = []*
12. *cd\_id = 0*
13. *cd\_title = ''*
14. *cd\_artist = ''*
15. *class CD:*
16. *"""Stores data about a CD:*
17. *properties:*
18. *cd\_id: (int) with CD ID*
19. *cd\_title: (string) with the title of the CD*
20. *cd\_artist: (string) with the artist of the CD*
21. *methods:*
22. *"""*
23. *# TODO Add Code to the CD class*
24. *# - - Contructor - - #*
25. *def \_\_init\_\_(self, cd\_id, cd\_title, cd\_artist):*
26. *# - - Attributes - - #*
27. *self.\_\_cd\_id = cd\_id*
28. *self.\_\_cd\_title = cd\_title*
29. *self.\_\_cd\_artist = cd\_artist*

32. *@property*
33. *def cd\_id(self):*
34. *return self.\_\_cd\_id*
35. *@cd\_id.setter*
36. *def cd\_id(self, value: int):*
37. *if str(value).isnumeric():*
38. *raise Exception('This should be a number')*
39. *else:*
40. *self.\_\_cd\_id = value*
41. *@property*
42. *def cd\_title(self):*
43. *return self.\_\_cd\_title*
44. *@cd\_title.setter*
45. *def cd\_title(self, value):*
46. *if str(value).isnumeric():*
47. *raise Exception('This should be a string')*
48. *else:*
49. *self.\_\_cd\_title = value*
50. *@property*
51. *def cd\_artist(self):*
52. *return self.\_\_cd\_artist*
53. *@cd\_artist.setter*
54. *def cd\_artist(self,value):*
55. *if str(value).isnumeric():*
56. *raise Exception('This should be a string value')*
57. *else:*
58. *self.\_\_cd\_artist = value*
60. *# -- PROCESSING -- #*
61. *class FileIO:*
62. *"""Processes data to and from file:*
63. *properties:*
64. *methods:*
65. *save\_inventory(file\_name, lst\_Inventory): -> None*
66. *load\_inventory(file\_name): -> (a list of CD objects)*
67. *"""*
68. *# TODO Add code to process data from a file*
69. *@staticmethod*
70. *def read\_file(strFileName, table):*
71. *table.clear()*
72. *objFile = open(strFileName, 'a+')*
73. *for line in objFile:*
74. *data = line.strip().split(',')*
75. *tbl = [data[0], data[1], data[2]]*
76. *lstOfCDObjects.append(tbl)*
77. *objFile.close()*
78. *# TODO Add code to process data to a file*
79. *def write\_file(strFileName, lstOfCDObjects):*
80. *objFile = open(strFileName, 'w')*
81. *for row in lstOfCDObjects:*
82. *lstValues = list(row.values())*
83. *lstValues[0] = str(lstValues[0])*
84. *objFile.write(','.join(lstValues) + '\n')*
85. *objFile.close()*
86. *print('\nYou have successfully saved your file.\n')*
87. *pass*
88. *# -- PRESENTATION (Input/Output) -- #*
89. *class IO:*
90. *# TODO add docstring*
91. *# TODO add code to show menu to user*
92. *@staticmethod*
93. *def print\_menu():*
94. *"""Displays a menu of choices to the user*
95. *Args:*
96. *None.*
97. *Returns:*
98. *None.*
99. *"""*
100. *print('Main Menu\n-----------------------------------\n\n[i] Display Current Inventory\n[a] Add CD')*
101. *print('[s] Save Inventory to File\n[l] Load Inventory\n[x] exit\n')*
103. *# TODO add code to captures user's choice*
104. *@staticmethod*
105. *def menu\_choice():*
106. *"""Gets user input for menu selection*
107. *Args:*
108. *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*
111. *"""*
112. *choice = ' '*
113. *try:*
114. *while choice not in ['l', 'a', 'i', 's', 'x']:*
115. *choice = input('Which operation would you like to perform? [l, a, i, s or x]: ').lower().strip()*
116. *print() # Add extra space for layout*
117. *return choice*
118. *except Exception as e:*
119. *print(e)*

122. *# TODO add code to display the current data on screen*
123. *@staticmethod*
124. *def show\_inventory(tbl):*
125. *print('======= The Current Inventory: =======')*
126. *print('ID\tCD Title (by: Artist)\n')*
127. *for row in tbl:*
128. *print('{}\t{} (by:{})'.format(\*row.values()))*
129. *print('======================================')*



134. *# TODO add code to get CD data from user*
135. *def add\_inventory():*
136. *# Ask user for new ID, CD Title and Artist*
137. *while True:*
138. *try:*
139. *cd\_id = int(input('Enter ID: ').strip())*
140. *break*
141. *except Exception as e:*
142. *print(e)*
143. *print('Please enter a number and try again. ')*
144. *cd\_title = input('What is the CD\'s title? ').strip()*
145. *cd\_artist = input('What is the Artist\'s name? ').strip()*
146. *cd\_entry = CD(cd\_id, cd\_title, cd\_artist)*
147. *lstOfCDObjects.append(CD)*
149. *return cd\_entry*
150. *# -- Main Body of Script -- #*
151. *# TODO Add Code to the main body*
152. *# Load data from file into a list of CD objects on script start*
153. *FileIO.read\_file(strFileName, lstOfCDObjects)*
154. *IO.show\_inventory(lstOfCDObjects)*
155. *# Display menu to user*
156. *while True:*
157. *IO.print\_menu()*
158. *choice = IO.menu\_choice()*
160. *# show user current inventory*
161. *if choice == 'i':*
162. *IO.show\_inventory(lstOfCDObjects)*
163. *continue*
164. *# let user add data to the inventory*
165. *elif choice == 'a':*
166. *IO.add\_inventory()*
167. *continue*
168. *# let user save inventory to file*
169. *elif choice == 's':*
170. *IO.show\_inventory(lstOfCDObjects)*
171. *strYesNo = input('Are you sure you want to save Inventory file? [y/n] ').strip().lower()*
172. *if strYesNo == 'y':*
173. *FileIO.write\_file(strFileName, lstOfCDObjects)*
174. *else:*
175. *input('Inventory was NOT save to file. Press any key to return to Main Menu.')*
176. *continue*
177. *# let user load inventory from file*
178. *elif choice == 'l':*
179. *print('WARNING: If you continue, all unsaved data will be lost and the Inventory re-loaded from file.')*
180. *strYesNo = input('type \'yes\' to continue and reload from file. otherwise reload will be canceled: ')*
181. *try:*
182. *if strYesNo.lower() == 'yes':*
183. *print('Reloading...')*
184. *FileIO.read\_file(strFileName, lstOfCDObjects)*
185. *elif strYesNo.lower() != '':*
186. *print('Start over...')*
187. *except Exception as e:*
188. *print(e)*
190. *FileIO.read\_file(strFileName, lstOfCDObjects)*
191. *IO.show\_inventory(lstOfCDObjects)*
192. *continue*
193. *# let user exit program*
195. *elif choice == 'x':*
196. *input('Ok, bye bye!')*
197. *break*
198. *else:*
199. *print('\nYou may want to try again\n')*