Rogger Tovar

11/21/2021

Foundations of Python

Assignment 06

CD Inventory Adding Functions

# Introduction

This week’s project modifying the now classic “CDInventory.py,” by properly adding functions to the class group.

# CD Inventory One More Time

Assignment 6 starts with preset code from last week’s assignment. This starter script was very confusing to me because I had no idea what I was looking at. I stared at the script for about 15 minutes to understand what was needed/TODO. Finally I decided to run the script to see what happens

Text

Description automatically generated

Idle 1 – error – file doesn’t exist.

Idle 1 shows the file mentioned in the script is not present or does not exist. This was quickly resolved when I created a file with the proper naming convention in the script, ran the script once more, and next thing I knew, it displayed the fully running CDInventory script. Seeing the result in Idle2, is always rewarding since it shows I am on the right path.

Text

Description automatically generated

Idle 2 - "MENU"

I wrote everything I needed TODO on a piece of paper, which helped move my progress along, Listing 1. Writing down has really helped me put together thought to find a solution or at minimum, get on the right track.

A hand holding a piece of paper with writing on it

Description automatically generated

Listing 1– starting notes.

Regarding to simplifying the process, I started from what I thought was the easiest function to create. I looked for the function that closes or askes the user to “exit” the program. Once, I found the “exit,” I was able to organize the idea of what to do next. Figure 1. Observing the context of my next step and what is already defined, the I/O, Data Progressor, and File Processor, was a matter of putting the pattern together and add already written code to my newly created Functions. The new functions are defined by the menu options in the code, write\_file, add\_inventory, delete\_inventory, etc., Figure2.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 1– finding the exit.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 2 - Newly created function and placed in a class.

There was a elif statement that needed to be split into the Data Processor Class and the other going to the I/O Class function. This took me a while to figure out. I knew there needed to be references somewhere to add the code. Once I figured out, how the references should be added, I cleaned up the code a little, removed the TODOs, and ran the code.

# Running CDInventory

So, running the code will display the Menu to the end user, who will then, have the option to make a selection. In these next examples, I will go through the succession of deleting a item/ID from the already saved list. Currently there are four entries in the CD Inventory, Idle 3

Text

Description automatically generated

Idle 3 - CD Inventory is run in the console.

Entry four, “Shock Value – by:Timberland,” will be removed, (pictured in Idle 4), then saved, (image Idle 5.)

Text

Description automatically generated

Idle 4 - ID four removed.

Text

Description automatically generated

Idle 5 - The Inventory is saved.

# Summary

I can say this was a little complicated, but something is bothering me. Either I am getting better at this, or this assignment was a little simpler; let’s hope for the former. I do miss entering the calculations myself, to better understand the references and building the arguments.

# Link to github

<https://github.com/roggergt/Assignment_06/>

# Appendix

1. *# Title: CDInventory.py*
2. *# Desc: Working with classes and functions.*
3. *# Change Log: (Who, When, What)*
4. *# rtovar, 2021-Nov-21, Copied from Original File*
5. *# rtovar, 2021-NOv-21, Created/added functions*
6. *# rtovar, 2021-Nove-21, 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. """ A collection of processing data fuctions:
20. Adding user inventory and delete inventory."""
22. **def** adding\_user\_inventory(strID, strTitle, stArtist):
24. """Adding user inventory grabs the user inputs of ID, Title,
25. and Artist to then add to the list of data (lstTble)
27. arg: strID - ID number selected by the end user.
28. strTitle - The name of the title of song selected by the end user.
29. strArtist -The Artist of the song selected by the end user.
31. Returns: The added inventory list will be updated and displayed to the end user.
32. """
33. intID = int(strID)
34. dicRow = {'ID': intID, 'Title': strTitle, 'Artist': stArtist}
35. lstTbl.append(dicRow)
36. IO.show\_inventory(lstTbl)
38. **def** delete\_inventory():
39. """Delete inventory function is designed to process a delete selection by the
40. end user then to update the over inventory list CDInventory text tile.
42. arg: none
44. return: updated list minus the selected ID choice from the end user, which
45. will remove the rove including, title and artist.
46. """
47. intRowNr = -1
48. blnCDRemoved = False
49. **for** row **in** lstTbl:
50. intRowNr += 1
51. **if** row['ID'] == intIDDel:
52. **del** lstTbl[intRowNr]
53. blnCDRemoved = True
54. **break**
55. **if** blnCDRemoved:
56. **print**('**\n**The CD was removed. **\n**')
57. **else**:
58. **print**('Could not find this CD!')
59. IO.show\_inventory(lstTbl)
61. **class** FileProcessor:
62. """Processing the data to and from text file"""
64. @staticmethod
65. **def** read\_file(strFileName, table):
66. """Function to manage data ingestion from file to a list of dictionaries
68. Reads the data from file identified by strFileName into a 2D table
69. (list of dicts) table one line in the file represents one dictionary row in table.
71. Args:
72. file\_name (string): name of file used to read the data from
73. table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
75. Returns:
76. None.
77. """
78. table.clear() *# this clears existing data and allows to load data from file*
79. objFile = open(strFileName, 'r')
80. **for** line **in** objFile:
81. data = line.strip().split(',')
82. dicRow = {'ID': int(data[0]), 'Title': data[1], 'Artist': data[2]}
83. table.append(dicRow)
84. objFile.close()
86. @staticmethod
87. **def** write\_file(strFileName, lstTbl):
88. """Write file saves the text file with CD Inventory data onto a destinaion on the hard drive.
90. arg: strFileName - is a variable for the CDInventory.txt file used to store the data entries.
91. lstTbl - is the current table of list with rows added with ID, Title, and Artist information.
93. return: None
94. """
95. objFile = open(strFileName, 'w')
96. **for** row **in** lstTbl:
97. lstValues = list(row.values())
98. lstValues[0] = str(lstValues[0])
99. objFile.write(','.join(lstValues) + '**\n**')
100. objFile.close()
101. **print**('**\n**You have successfully saved your file.**\n**')

104. *# -- PRESENTATION (Input/Output) -- #*
106. **class** IO:
107. """Handling Input / Output"""
109. @staticmethod
110. **def** print\_menu():
111. """Displays a menu of choices to the user
113. Args:
114. None.
116. Returns:
117. None.
118. """
120. **print**('Menu**\n\n**[l] load Inventory from file**\n**[a] Add CD**\n**[i] Display Current Inventory')
121. **print**('[d] delete CD from Inventory**\n**[s] Save Inventory to file**\n**[x] exit**\n**')
123. @staticmethod
124. **def** menu\_choice():
125. """Gets user input for menu selection
127. Args:
128. None.
130. Returns:
131. choice (string): a lower case sting of the users input out of the choices l, a, i, d, s or x
133. """
134. choice = ' '
135. **while** choice **not** **in** ['l', 'a', 'i', 'd', 's', 'x']:
136. choice = input('Which operation would you like to perform? [l, a, i, d, s or x]: ').lower().strip()
137. **print**() *# Add extra space for layout*
138. **return** choice
140. @staticmethod
141. **def** show\_inventory(table):
142. """Displays current inventory table

145. Args:
146. table (list of dict): 2D data structure (list of dicts) that holds the data during runtime.
148. Returns:
149. None.
151. """
152. **print**('======= The Current Inventory: =======')
153. **print**('ID**\t**CD Title (by: Artist)**\n**')
154. **for** row **in** table:
155. **print**('{}**\t**{} (by:{})'.format(\*row.values()))
156. **print**('======================================')
158. **def** add\_inventory():
159. """This function adds an inventory row using 3 inputs: ID, Title, Artist.
161. arg: none
163. return: An input selection is offered to the end user if they use to add a row.
164. """
166. *# 3.3.1 Ask user for new ID, CD Title and Artist*
167. strID = input('Enter ID: ').strip()
168. strTitle = input('What is the CD**\'**s title? ').strip()
169. stArtist = input('What is the Artist**\'**s name? ').strip()
170. DataProcessor.adding\_user\_inventory(strID, strTitle, stArtist)
172. *# 1. When program starts, read in the currently saved Inventory*
173. FileProcessor.read\_file(strFileName, lstTbl)

176. *# 2. start main loop*
177. **while** True:
178. *# 2.1 Display Menu to user and get choice*
179. IO.print\_menu()
180. strChoice = IO.menu\_choice()
182. *# 3. Process menu selection*
183. *# 3.1 process exit first*
184. **if** strChoice == 'x':
185. **break**
186. *# 3.2 process load inventory*
187. **if** strChoice == 'l':
188. **print**('WARNING: If you continue, all unsaved data will be lost and the Inventory re-loaded from file.')
189. strYesNo = input('type **\'**yes**\'** to continue and reload from file. otherwise reload will be canceled: ')
190. **if** strYesNo.lower() == 'yes':
191. **print**('reloading...')
192. FileProcessor.read\_file(strFileName, lstTbl)
193. IO.show\_inventory(lstTbl)
194. **else**:
195. input('canceling... Inventory data NOT reloaded. Press [ENTER] to continue to the menu.')
196. IO.show\_inventory(lstTbl)
197. **continue** *# start loop back at top.*
198. *# 3.3 process add a CD*
199. *# I have already moved ADD CD*
200. **elif** strChoice == 'a':
201. IO.add\_inventory()
202. **continue**
203. *# 3.4 process display current inventory*
204. **elif** strChoice == 'i':
205. IO.show\_inventory(lstTbl)
206. **continue** *# start loop back at top.*
207. *# 3.5 process delete a CD*
208. **elif** strChoice == 'd':
209. *# 3.5.1 get Userinput for which CD to delete*
210. *# 3.5.1.1 display Inventory to user*
211. IO.show\_inventory(lstTbl)
212. *# 3.5.1.2 ask user which ID to remove*
213. intIDDel = int(input('Which ID would you like to delete? ').strip())
214. *# 3.5.2 search thru table and delete CD*
215. DataProcessor.delete\_inventory()
216. **continue** *# start loop back at top.*
217. *# 3.6 process save inventory to file*
218. **elif** strChoice == 's':
219. *# 3.6.1 Display current inventory and ask user for confirmation to save*
220. IO.show\_inventory(lstTbl)
221. strYesNo = input('Save this inventory to file? [y/n] ').strip().lower()
222. *# 3.6.2 Process choice*
223. **if** strYesNo == 'y':
224. *# 3.6.2.1 save data*
225. FileProcessor.write\_file(strFileName, lstTbl)
226. **else**:
227. input('The inventory was NOT saved to file. Press [ENTER] to return to the menu.')
228. **continue** *# start loop back at top.*
229. *# 3.7 catch-all should not be possible, as user choice gets vetted in IO, but to be save:*
230. **else**:
231. **print**('General Error')