<Hongyi Zhu>

<2021-11-21>

<IT FDN 110>

<Assignment\_Module\_06>

Functions

# Introduction

In this module, we learned the reason why we need functions in our code, the elements in the functions and more detals about functions. First, the function in python can be similarly understood as function in math: we need some variables (arguments) as the input of function. After processing, the function returns the output. In the code, after defining the function, we can call the function when we need such process of data. In Module 6, we systematically learned the parameter and variables. For input, there are several techniques to set the input for dealing with different problems, like positional/named arguments, default parameter values, and reference datatype. Also, we learned the difference between local and global variables, and their scope. Plus, we first contacted one of the core parts of object-oriented programming--class.

# Topic 1

I used [Syntax Highlighter](https://highlight.hohli.com/index.php) (external reference)[[1]](#footnote-1) web page to highlight the script as shown in Listing 1 and highlighted what I revised/added in the example solution. In this topic, I read the solution example and understood the coding logic. What I did is to define and call the following four functions:

Then constructs them into one dictionary and adds it into the existed table (list of dicts).

1. strID, strTitle, stArtist = IO.ask\_newID()
2. lstTbl = DataProcessor.add\_item(strID, strTitle, stArtist, lstTbl)
3. lstTbl = DataProcessor.delete\_item(lstTbl)
4. FileProcessor.write\_file(strFileName, lstTbl)

Notes (which has also been noted in the complete version of the code):

1. IO.ask\_newID()

"""Function to ask user to input a new item.

Receives the input from user including (1) ID (2) CD name (3) CD artist.

Args:

None

Returns:

strID (string): an integer string of the user’s input;

strTitle (string): a string of the user’s input;

stArtist (string): a string of the user’s input.

"""

1. DataProcessor.add\_item(ID, Title, Artist, table)

"""Function to add a new item into the existed table.

Combines the user’s input including (1) ID (2) CD name (3) CD artist to a dict and then add it into the existed table

Args:

ID (string): an integer string of the user’s input;

Title (string): a string of user’s input

Artist (string): a string of user’s input

table (list of dicts): a list of dicts (the dicts include 3 key(‘ID’,’Title’,’Artist’))

Returns:

table (new list of dicts): a list of dicts (the dicts include 3 key(‘ID’,’Title’,’Artist’)) which has been added the new item from the user’s input.

"""

1. DataProcessor.delete\_item(table,IDDel)

"""Function to delete the item whose ID is provided by user.

Delete the corresponding item of the existed table.

Args:

table (list of dicts): a list of dicts which is the existed CDInventory

IDDel (int): an integer which corresponds to the ID that the item the user wants to delete

Returns:

table (list of dicts): a list of dicts whose corresponding item has been deleted.

"""

1. FileProcessor.write\_file(file\_name, table)

"""Function to write the table to the file.

Write the existed table into the file whose name is input by user

Args:

file\_name (string): an string which is the file name that the user want to use for saving the data.

table (list of dicts): a list of dicts which is the existed CDInventory

Returns:

None.

"""

And here is the listing of the script, I also highlighted what I revised/added in the listing below:

1. # -- DATA -- #
3. strChoice = '' # User input
4. lstTbl = []  # list of lists to hold data
5. dicRow = {}  # list of data row
6. strFileName = 'CDInventory.txt'  # data storage file
7. objFile = None  # file object
9. # -- PROCESSING -- #
11. class DataProcessor:
13. @staticmethod
14. def add\_item(ID, Title, Artist, table):
15. intID = int(ID)
16. dic = {'ID': intID, 'Title': Title, 'Artist': Artist}
17. table.append(dic)
18. return table
20. @staticmethod
21. def delete\_item(table):
22. intRowNr = -1
23. blnCDRemoved = False
24. for row in table:
25. intRowNr += 1
26. if row['ID'] == intIDDel:
27. del table[intRowNr]
28. blnCDRemoved = True
29. break
31. if blnCDRemoved:
32. print('The CD was removed')
33. else:
34. print('Could not find this CD!')
35. return table
37. class FileProcessor:
38. @staticmethod
39. def read\_file(file\_name, table):
40. table.clear()  # this clears existing data and allows to load data from file
41. objFile = open(file\_name, 'r')
42. for line in objFile:
43. data = line.strip().split(',')
44. dicRow = {'ID': int(data[0]), 'Title': data[1], 'Artist': data[2]}
45. table.append(dicRow)
46. objFile.close()
47. @staticmethod
48. def write\_file(file\_name, table):
49. objFile = open(file\_name, 'w')
50. for row in table:
51. lstValues = list(row.values())
52. lstValues[0] = str(lstValues[0])
53. objFile.write(','.join(lstValues) + '\n')
54. objFile.close()
56. class IO:
57. @staticmethod
58. def print\_menu():
59. print('Menu\n\n[l] load Inventory from file\n[a] Add CD\n[i] Display Current Inventory')
60. print('[d] delete CD from Inventory\n[s] Save Inventory to file\n[x] exit\n')
62. @staticmethod
63. def menu\_choice():
64. choice = ' '
65. while choice not in ['l', 'a', 'i', 'd', 's', 'x']:
66. choice = input('Which operation would you like to perform? [l, a, i, d, s or x]: ').lower().strip()
67. print()  # Add extra space for layout
68. return choice
70. @staticmethod
71. def show\_inventory(table):
72. print('======= The Current Inventory: =======')
73. print('ID\tCD Title (by: Artist)\n')
74. for row in table:
75. print('{}\t{} (by:{})'.format(\*row.values()))
76. print('======================================')
78. @staticmethod
79. def ask\_newID():
80. strID = input('Enter ID: ').strip()
81. strTitle = input('What is the CD\'s title? ').strip()
82. stArtist = input('What is the Artist\'s name? ').strip()
83. return strID, strTitle, stArtist
85. FileProcessor.read\_file(strFileName, lstTbl)
87. while True:
88. IO.print\_menu()
89. strChoice = IO.menu\_choice()
91. if strChoice == 'x':
92. break
94. if strChoice == 'l':
95. print('WARNING: If you continue, all unsaved data will be lost and the Inventory re-loaded from file.')
96. strYesNo = input('type \'yes\' to continue and reload from file. otherwise reload will be canceled')
97. if strYesNo.lower() == 'yes':
98. print('reloading...')
99. FileProcessor.read\_file(strFileName, lstTbl)
100. IO.show\_inventory(lstTbl)
101. else:
102. input('canceling... Inventory data NOT reloaded. Press [ENTER] to continue to the menu.')
103. IO.show\_inventory(lstTbl)
104. continue  # start loop back at top.
106. elif strChoice == 'a':
107. strID, strTitle, stArtist = IO.ask\_newID()
108. lstTbl = DataProcessor.add\_item(strID, strTitle, stArtist, lstTbl)
109. IO.show\_inventory(lstTbl)
110. continue  # start loop back at top.
112. elif strChoice == 'i':
113. IO.show\_inventory(lstTbl)
114. continue  # start loop back at top.
116. elif strChoice == 'd':
117. IO.show\_inventory(lstTbl)
118. intIDDel = int(input('Which ID would you like to delete? ').strip())
119. lstTbl = DataProcessor.delete\_item(lstTbl)
120. IO.show\_inventory(lstTbl)
121. continue  # start loop back at top.
123. elif strChoice == 's':
124. IO.show\_inventory(lstTbl)
125. strYesNo = input('Save this inventory to file? [y/n] ').strip().lower()
126. if strYesNo == 'y':
127. FileProcessor.write\_file(strFileName, lstTbl)
128. else:
129. input('The inventory was NOT saved to file. Press [ENTER] to return to the menu.')
130. continue  # start loop back at top.
132. else:
133. print('General Error')

Listing 1 - Script of Assignment 06

# Topic 2

In this topic, I ran the script and verified the correctness in spyder with the results of the first run are shown in Figure 1 – 2 and the second run are shown in Figure 3 – 5. I also repeated it in a terminal window and the results are shown in Figure 6 – 10.

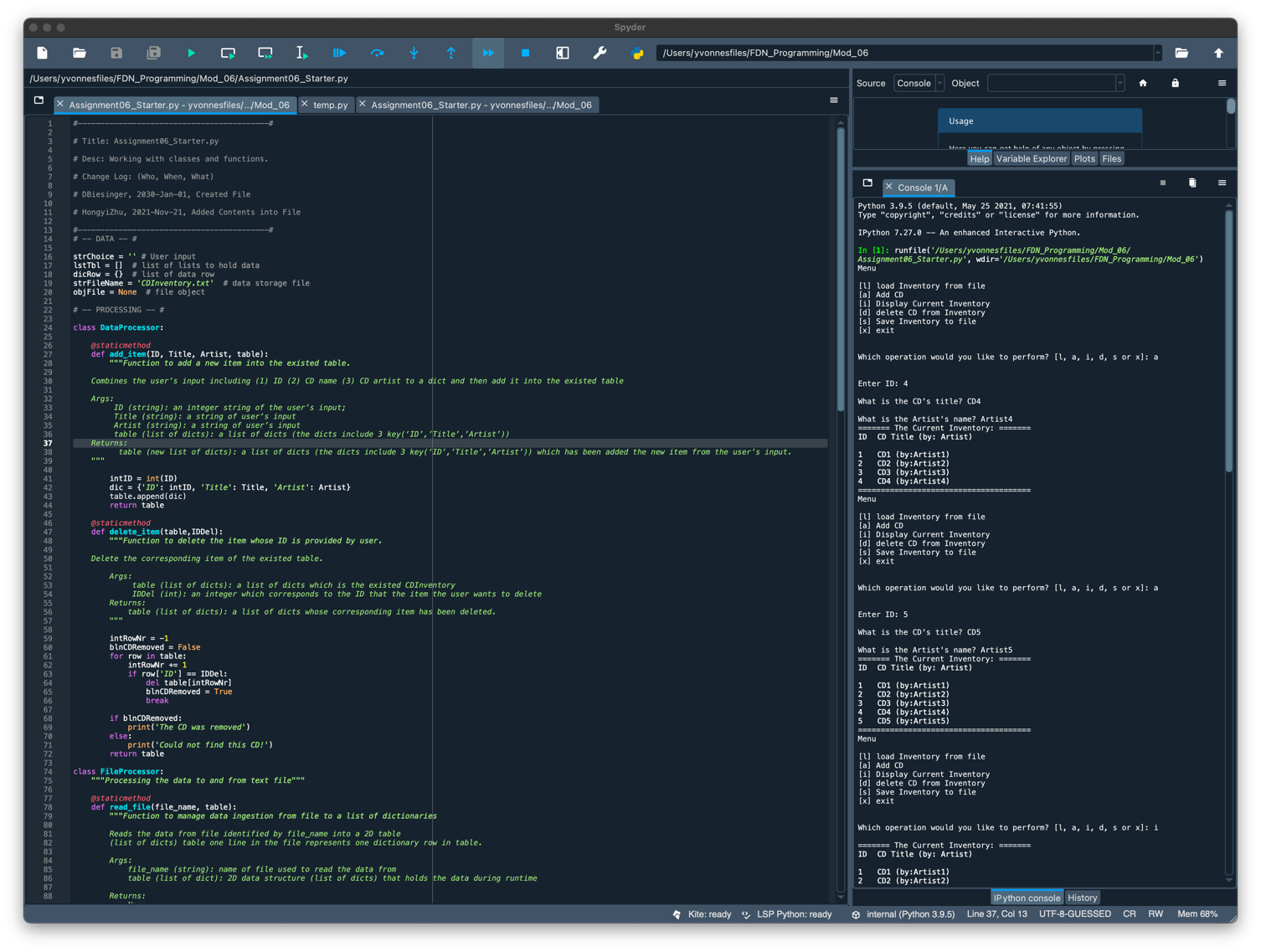


Figure 1 - Running Results of the First Run in Spyder

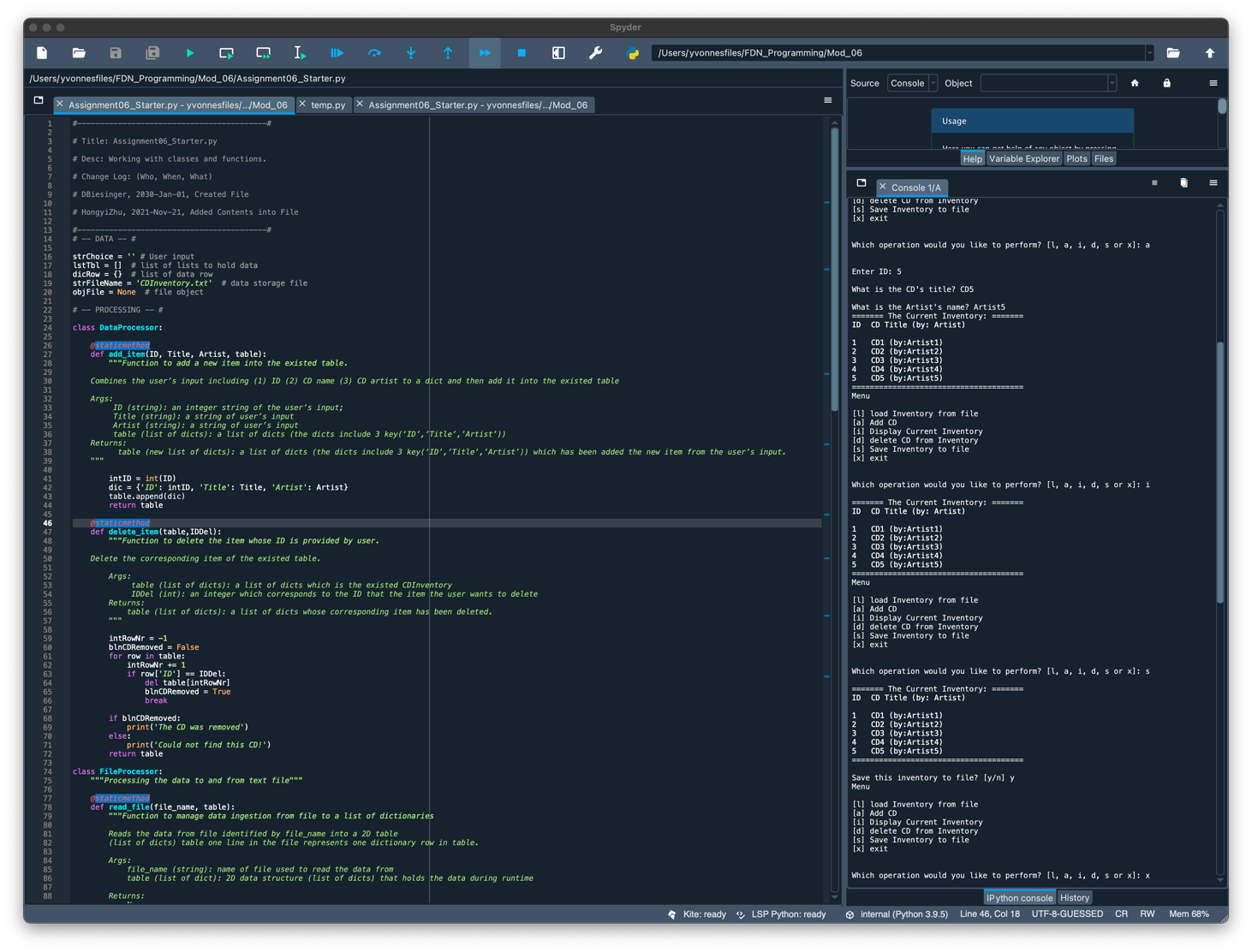


Figure 2 - Running Results of the First Run in Spyder (Continue)

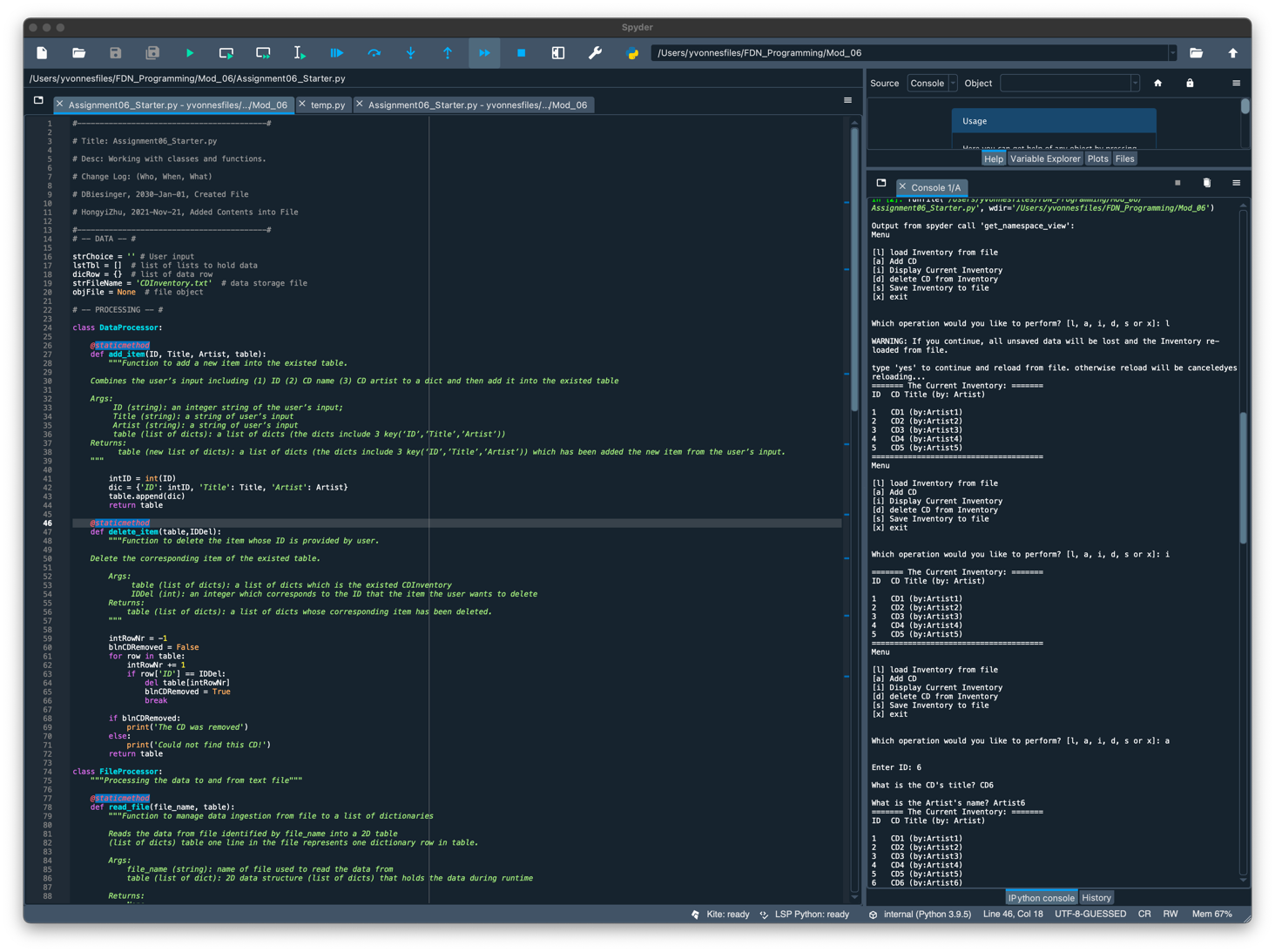


Figure 3 - Running Results of the Second Run in Spyder

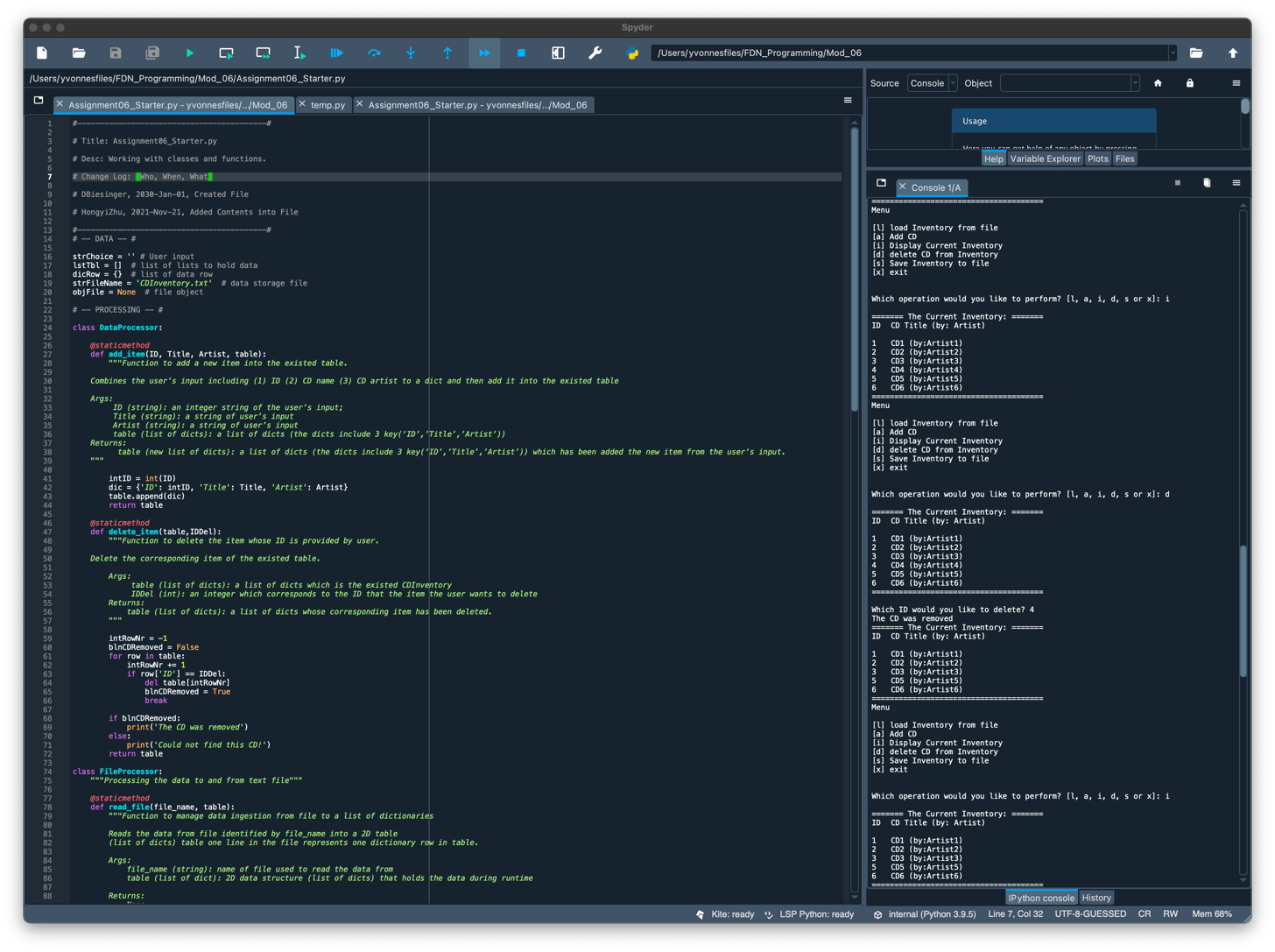


Figure 4 - Running Results of the Second Run in Spyder (Continue)

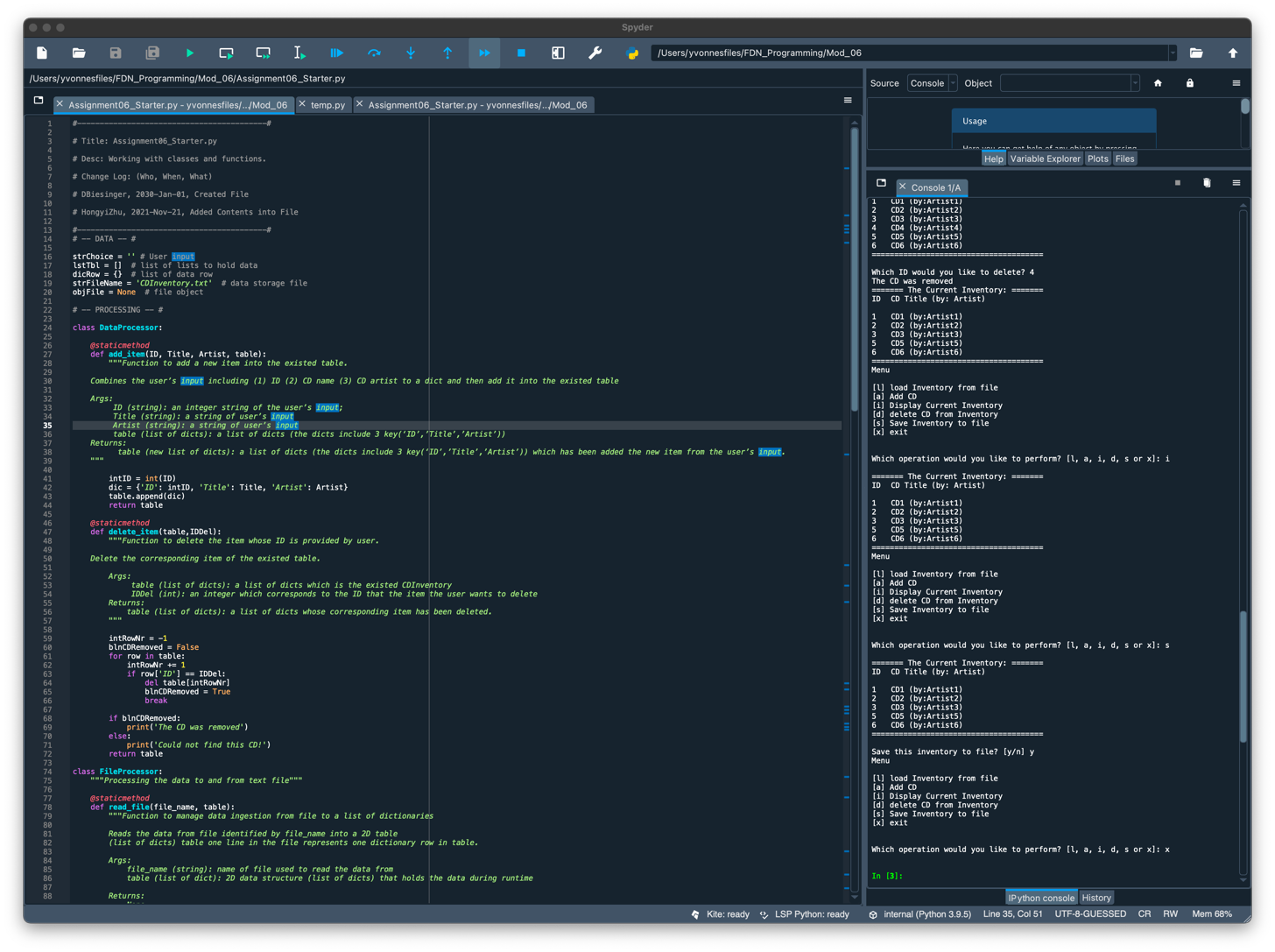


Figure 5 - Running Results of the Second Run in Spyder (Continue)



Figure 6 - Running Resultls of the First Run in Terminal



Figure 7 - Running Resultls of the First Run in Terminal (Continue)



Figure 8 - Running Resultls of the Second Run in Terminal



Figure 9 - Running Resultls of the Second Run in Terminal (Continue)

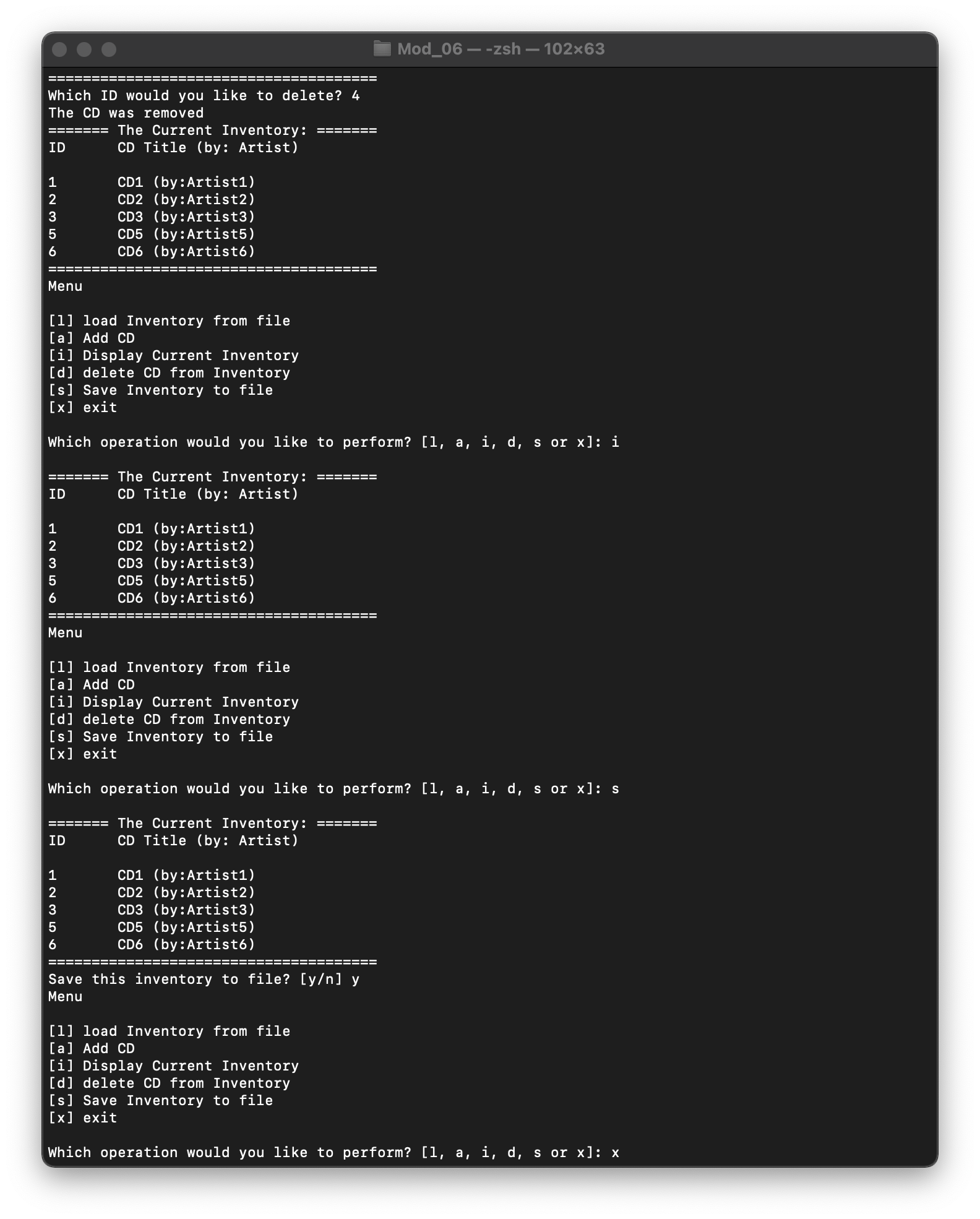


Figure 10 - Running Resultls of the Second Run in Terminal (Continue)

# Topic 3

In this topic, the text file outcomes of the script (including both spyder and terminal) are shown below. Figure 11 – 12 are the text file outcomes of run 1 and run 2 using spyder, and Figure 13 – 14 are the text file outcomes of run 1 and run 2 using terminal.

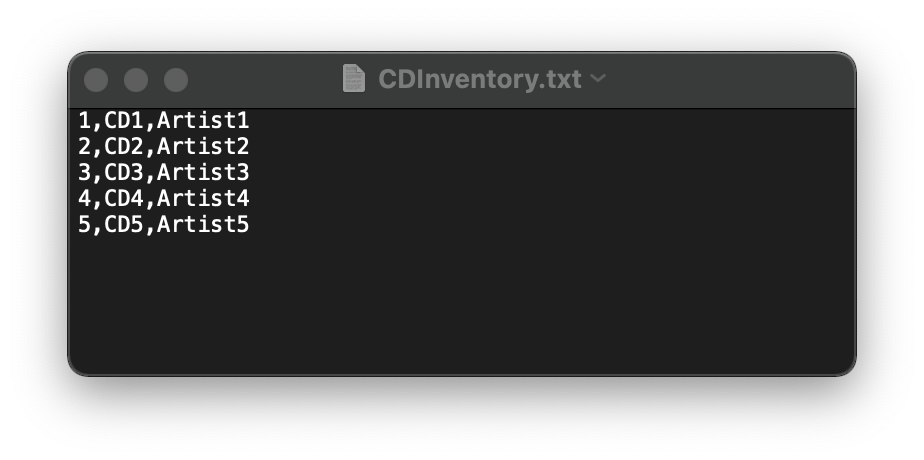


Figure 11 - Text File Caption of the First Run Using Spyder

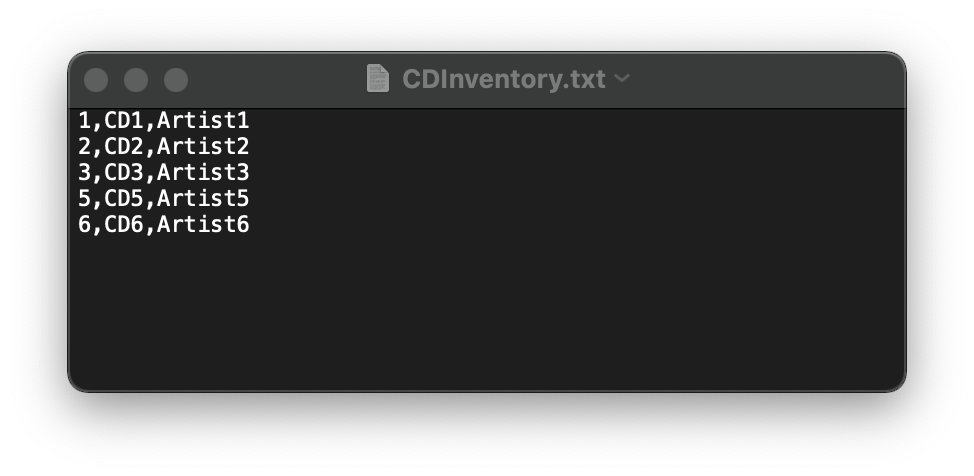


Figure 12 - Text File Caption of the Second Run Using Spyder

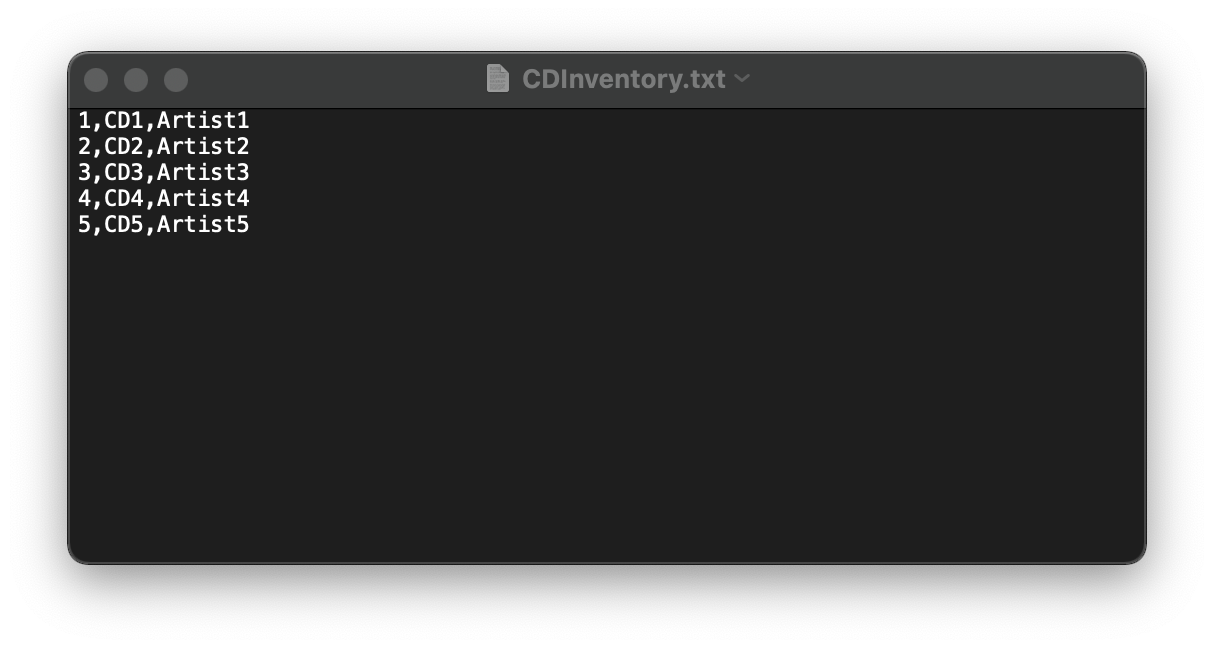


Figure 13 - Text File Caption of the First Run Using Terminal

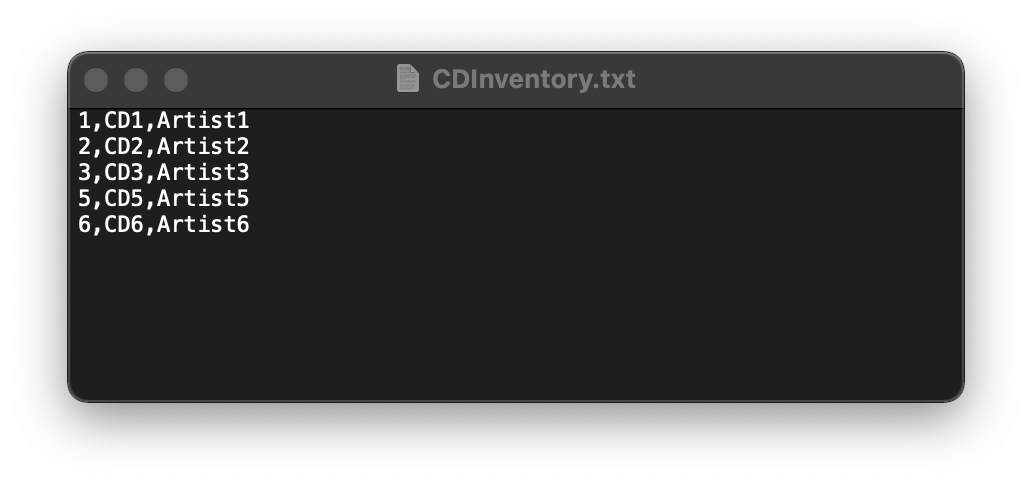


Figure 14 - Text File Caption of the Second Run Using Terminal

# Summary

In this assignment, first I strengthened the understanding of the variable scope. Meanwhile, during defining and calling functions, I also learned the rules of writing functions in code. Plus, when writing the docstring, I realized the importance of such explanation in communication between developer. We can imagine that good explanation in the code can largely increase the working efficiency.

# Appendix

# Listing CDInventory.py

1. #------------------------------------------#
3. # Title: Assignment06\_Starter.py
5. # Desc: Working with classes and functions.
7. # Change Log: (Who, When, What)
9. # DBiesinger, 2030-Jan-01, Created File
11. # HongyiZhu, 2021-Nov-21, Added Contents into File
13. #------------------------------------------#
14. # -- DATA -- #
16. strChoice = '' # User input
17. lstTbl = []  # list of lists to hold data
18. dicRow = {}  # list of data row
19. strFileName = 'CDInventory.txt'  # data storage file
20. objFile = None  # file object
22. # -- PROCESSING -- #
24. class DataProcessor:
26. @staticmethod
27. def add\_item(ID, Title, Artist, table):
28. """Function to add a new item into the existed table.
30. Combines the user’s input including (1) ID (2) CD name (3) CD artist to a dict and then add it into the existed table
32. Args:
33. ID (string): an integer string of the user’s input;
34. Title (string): a string of user’s input
35. Artist (string): a string of user’s input
36. table (list of dicts): a list of dicts (the dicts include 3 key(‘ID’,’Title’,’Artist’))
37. Returns:
38. table (new list of dicts): a list of dicts (the dicts include 3 key(‘ID’,’Title’,’Artist’)) which has been added the new item from the user’s input.
39. """
41. intID = int(ID)
42. dic = {'ID': intID, 'Title': Title, 'Artist': Artist}
43. table.append(dic)
44. return table
46. @staticmethod
47. def delete\_item(table,IDDel):
48. """Function to delete the item whose ID is provided by user.
50. Delete the corresponding item of the existed table.
52. Args:
53. table (list of dicts): a list of dicts which is the existed CDInventory
54. IDDel (int): an integer which corresponds to the ID that the item the user wants to delete
55. Returns:
56. table (list of dicts): a list of dicts whose corresponding item has been deleted.
57. """
59. intRowNr = -1
60. blnCDRemoved = False
61. for row in table:
62. intRowNr += 1
63. if row['ID'] == IDDel:
64. del table[intRowNr]
65. blnCDRemoved = True
66. break
68. if blnCDRemoved:
69. print('The CD was removed')
70. else:
71. print('Could not find this CD!')
72. return table
74. class FileProcessor:
75. """Processing the data to and from text file"""
77. @staticmethod
78. def read\_file(file\_name, table):
79. """Function to manage data ingestion from file to a list of dictionaries
81. Reads the data from file identified by file\_name into a 2D table
82. (list of dicts) table one line in the file represents one dictionary row in table.
84. Args:
85. file\_name (string): name of file used to read the data from
86. table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
88. Returns:
89. None.
90. """
91. table.clear()  # this clears existing data and allows to load data from file
92. objFile = open(file\_name, 'r')
93. for line in objFile:
94. data = line.strip().split(',')
95. dicRow = {'ID': int(data[0]), 'Title': data[1], 'Artist': data[2]}
96. table.append(dicRow)
97. objFile.close()
98. @staticmethod
99. def write\_file(file\_name, table):
100. """Function to write the table to the file.
101. Write the existed table into the file whose name is input by user
103. Args:
104. file\_name (string): an string which is the file name that the user want to use for saving the data.
105. table (list of dicts): a list of dicts which is the existed CDInventory
106. Returns:
107. None.
108. """
110. objFile = open(file\_name, 'w')
111. for row in table:
112. lstValues = list(row.values())
113. lstValues[0] = str(lstValues[0])
114. objFile.write(','.join(lstValues) + '\n')
115. objFile.close()
117. class IO:
118. """Handling Input / Output"""
119. @staticmethod
120. def print\_menu():
121. """Displays a menu of choices to the user
123. Args:
124. None.
126. Returns:
127. None.
128. """
129. print('Menu\n\n[l] load Inventory from file\n[a] Add CD\n[i] Display Current Inventory')
130. print('[d] delete CD from Inventory\n[s] Save Inventory to file\n[x] exit\n')
132. @staticmethod
133. def menu\_choice():
134. """Gets user input for menu selection
136. Args:
137. None.
139. Returns:
140. choice (string): a lower case sting of the users input out of the choices l, a, i, d, s or x
142. """
143. choice = ' '
144. while choice not in ['l', 'a', 'i', 'd', 's', 'x']:
145. choice = input('Which operation would you like to perform? [l, a, i, d, s or x]: ').lower().strip()
146. print()  # Add extra space for layout
147. return choice
149. @staticmethod
150. def show\_inventory(table):
151. """Displays current inventory table

154. Args:
155. table (list of dict): 2D data structure (list of dicts) that holds the data during runtime.
157. Returns:
158. None.
160. """
161. print('======= The Current Inventory: =======')
162. print('ID\tCD Title (by: Artist)\n')
163. for row in table:
164. print('{}\t{} (by:{})'.format(\*row.values()))
165. print('======================================')
167. @staticmethod
168. def ask\_newID():
169. """Function to ask user to input a new item.
171. Receives the input from user including (1) ID (2) CD name (3) CD artist.
173. Args:
174. None
175. Returns:
176. strID (string): an integer string of the user’s input;
177. strTitle (string): a string of the user’s input;
178. stArtist (string): a string of the user’s input.
179. """
181. strID = input('Enter ID: ').strip()
182. strTitle = input('What is the CD\'s title? ').strip()
183. stArtist = input('What is the Artist\'s name? ').strip()
184. return strID, strTitle, stArtist
186. FileProcessor.read\_file(strFileName, lstTbl)
188. while True:
189. IO.print\_menu()
190. strChoice = IO.menu\_choice()
192. if strChoice == 'x':
193. break
195. if strChoice == 'l':
196. print('WARNING: If you continue, all unsaved data will be lost and the Inventory re-loaded from file.')
197. strYesNo = input('type \'yes\' to continue and reload from file. otherwise reload will be canceled')
198. if strYesNo.lower() == 'yes':
199. print('reloading...')
200. FileProcessor.read\_file(strFileName, lstTbl)
201. IO.show\_inventory(lstTbl)
202. else:
203. input('canceling... Inventory data NOT reloaded. Press [ENTER] to continue to the menu.')
204. IO.show\_inventory(lstTbl)
205. continue  # start loop back at top.
207. elif strChoice == 'a':
208. strID, strTitle, stArtist = IO.ask\_newID()
209. lstTbl = DataProcessor.add\_item(strID, strTitle, stArtist, lstTbl)
210. IO.show\_inventory(lstTbl)
211. continue  # start loop back at top.
213. elif strChoice == 'i':
214. IO.show\_inventory(lstTbl)
215. continue  # start loop back at top.
217. elif strChoice == 'd':
218. IO.show\_inventory(lstTbl)
219. intIDDel = int(input('Which ID would you like to delete? ').strip())
220. lstTbl = DataProcessor.delete\_item(lstTbl, intIDDel)
221. IO.show\_inventory(lstTbl)
222. continue  # start loop back at top.
224. elif strChoice == 's':
225. IO.show\_inventory(lstTbl)
226. strYesNo = input('Save this inventory to file? [y/n] ').strip().lower()
227. if strYesNo == 'y':
228. FileProcessor.write\_file(strFileName, lstTbl)
229. else:
230. input('The inventory was NOT saved to file. Press [ENTER] to return to the menu.')
231. continue  # start loop back at top.
233. else:
234. print('General Error')

1. Retrieved 2021-Nov-14 [↑](#footnote-ref-1)