

Andrew Yeo

March 11, 2020

Foundations of Programming: Python

Github Repository: https://github.com/andyhyeo/assignment_08

Module 08

Knowledge Document

Intro

In this week's assignment, we go over Object Oriented Programming. Object Oriented Programming makes use of classes which are the blueprints for objects. The class maintains then data and the operations of the object. The object is an “instantiation” of the class. And the “features” of the objects are inherited from its class (attributes, properties, and methods)

Details

In this week's assignment we are instructed to build the CD inventory script from scratch. I largely already used what the script that we have been using in order to focus on the parts that were the topics of this weeks modules: object oriented programming, defining classes, methods, properties, and attributes.

Class CD

```
11 # -- DATA -- #
12 # DBiesinger, 2030-Jan-01, created file
13 from os import path
14 strFileName = 'cdInventory.txt'
15 lstOfCDObjects = []
16
17 class CD:
18     """Stores data about a CD:
19
20     properties:
21         cd_id: (int) with CD ID
22         cd_title: (string) with the title of the CD
23         cd_artist: (string) with the artist of the CD
24     methods:
25
26     """
27     # -- Contructor / Initializer -- #
28     def __init__(self, cd_id, cd_title, cd_artist):
29         self.id = cd_id
30         self.title = cd_title
31         self.artist = cd_artist
32
```

Figure 1: Building the CD class with attributes of id, title, and artist.

The CD class was built simply by adding parameters. Even though I have gotten a little bit of purchase on Object Oriented Programming concepts, I still had to build this largely by emulating the script within the module notes. I tested it within the console to make sure that it probably stored data when called.

```
58 @staticmethod
59 def load_inventory(file_name, lstOfCDObjects):
60     """Function to manage data ingestion from file to a list of dictionaries
61
62     Reads the data from file identified by file_name into a 2D table
63     (list of dicts) table one line in the file represents one dictionary row in table.
64
65     Args:
66         file_name (string): name of file used to read the data from
67         table (list of dict): 2D data structure (list of dicts) that holds the data during
68
69     Returns:
70         None.
71     """
72     try:
73         if path.exists('cdInventory.txt'): #Corrects for a bug in the original file whereby
74             lstOfCDObjects.clear() # this clears existing data and allows to load data from
75             objFile = open(file_name, 'r')
76             for line in objFile:
77                 data = line.strip().split(',')
78                 dicRow = {'ID': int(data[0]), 'Title': data[1], 'Artist': data[2]}
79                 lstOfCDObjects.append(dicRow)
80             objFile.close()
81             return lstOfCDObjects
82     except FileNotFoundError:
83         pass
```

Figure 2: Building the FileIO class with the method of load_inventory.

Class File IO with Method load_inventory

Figure 2 shows the part of the script where I built the load_inventory method of within the class FileIO. The module goes over concepts and best practices of writing class such as writing constructors, attributes, and properties but I was not precisely sure how to apply that and if I should have applied it. It was largely a copy and paste from the previous scripts. For simplicity, I opted to use the non-pickling loading strategy. And as shown, I wrote in the structured error handling to continue with the script if the file is not found.

Class File IO with Method save_inventory

```
86 @staticmethod
87 def save_inventory(file_name, table):
88     """Writes the inventory of IDs, CD Names, and Artists to a text file
89
90     Args:
91         file_name (string): The name of the file that it will write to
92         table (list of dict): 2D data structure (list of dicts) that holds the data during
93
94     Returns:
95         None but saves a file in the directory of the python script
96
97     """
98     objFile = open(file_name, 'w')
99     for row in table:
100         strRow = ''
101         for item in row.values():
102             strRow += str(item) + ','
103         strRow = strRow[:-1] + '\n'
104         objFile.write(strRow)
105     objFile.close()
```

Figure 3: Building the FileIO class with the method of save_inventory.

The load_inventory script was an amalgamation of previous scripts and so to was the the save_inventory.

Try-Except Error Structure Handling for ValueError in IO.ask() function

I also successfully substituted the try-except structure within the IO.ask() function which took a fair amount of trial error. I am learning that I can use all of the Spyder console in order to help me debug a code. For instance, I can run just line by line of the code. I can use the variable explorer as feedback if a variable was defined or changed the way that I wanted to.

```
181 @staticmethod
182 def ask():
183     """Ask user for new ID, CD Title and Artist
184
185     Args:
186         None
187
188     Returns:
189         dicRow (dictionary): A dictionary entry with ID (int): integer that holds
190         the ID tag,title (string): string that holds the name of the CD
191         and an artist (string): string that holds the name of the Artist.
192     """
193     while True:
194         strID = input('Enter ID: ').strip()
195         try:
196             strID = int(strID)
197             break
198         except ValueError:
199             print('That is not an integer')
200     strTitle = input('What is the CD\'s title? ').strip()
201     stArtist = input('What is the Artist\'s name? ').strip()
202     dicRow = {'ID': strID, 'CD Title': strTitle, 'Artist': stArtist}
203
204     return dicRow
---
```

Figure 4: Substituting Structured Error Handling in the Ask Function

Try-Except Error Structure Handling to handle not-integers when deleting

Figure 5 shows the part of the CD Inventory script which prompts the user to that to put in an integer when they have not done so successfully.

```

247 elif strChoice == 'd':
248     # 3.5.1 get Userinput for which CD to delete
249     # 3.5.1.1 display Inventory to user
250     IO.show_inventory(lstTbl)
251     # 3.5.1.2 ask user which ID to remove
252     while True:
253         try:
254             intIDDel = int(input('Which ID would you like to delete? ').strip())
255             break
256         except ValueError:
257             print('That is not an integer')
258     # 3.5.2 search thru table and delete CD
259     DataProcessor.delete_cd(intIDDel, lstTbl)
260     IO.show_inventory(lstTbl)
261     continue # start loop back at top.

```

Figure 4: Error Structure Handling to handle not-integers when prompted to delete

FileIO.load_inventory

Figure 5: Reading Files into the Script

```

85 # -- PROCESSING -- #
86 class FileIO:
87     """Processes data to and from file:
88
89     properties:
90
91     methods:
92         save_inventory(file_name, lst_Inventory): -> None
93         load_inventory(file_name): -> (a list of CD objects)
94
95     """
96     @staticmethod
97     def load_inventory(file_name):
98         """Function to manage data ingestion from file to a list of dictionaries
99
100         Reads the data from file identified by file_name into a 2D table
101         (list of dicts) table one line in the file represents one dictionary row in tab
102
103         Args:
104             file_name (string): name of file used to read the data from
105             table (list of dict): 2D data structure (list of dicts) that holds the data
106
107         Returns:
108             None.
109
110         """
111         try:
112             with open(file_name, 'rb') as objFile:
113                 table = pickle.load(objFile)
114                 return table
115         except FileNotFoundError:
116             pass

```

The FileIO class is populated with the load_inventory function which was largely a derivative of the pickle method. After researching online, it looks as if pickling was the best way to manage the reading of files into the script. This is corroborated by the fact that we went over pickling in preceding courses.

FileIO.save_inventory - The File Saving Function

```
117 @staticmethod
118 def save_inventory(file_name, table):
119     """Writes the inventory of IDs, CD Names, and Artists to a text file
120
121     Args:
122         file_name (string): The name of the file that it will write to
123         table (list of dict): 2D data structure (list of dicts) that holds the
124
125     Returns:
126         None but saves a file in the directory of the python script
127
128     """
129     with open(file_name, 'wb') as objFile:
130         table = pickle.dump(table, objFile)
131
```

Figure 6: Writing Files from the Script

The save_inventory method of the FileIO class is largely a derivative from a subsequent scripts pickling method. This seemed to be the most viable solution for handle class and objects.

FileIO.show_inventory - The Inventory Showing Function

Figure 7: Showing Inventory

```
167 @staticmethod
168 def show_inventory(table):
169     """Displays current inventory table
170
171
172     Args:
173         table (list of dict): 2D data structure (list of dicts) that holds the
174
175     Returns:
176         None.
177
178     """
179     print('==== The Current Inventory: =====')
180     print('ID\tCD Title (by: Artist)\n')
181     for CD in table:
182         print('{ }\t{ } (by:{ })'.format(CD.ID, CD.title, CD.artist))
183     print('=====')
```

Douglas, our class TA, wrote all of us to address a question that many of us must have posed to him - “Should we start from scratch or should we adjust previously written code?” As you can probably tell, I took the latter strategy. This code was largely readjusted to call on the CDs to display.

IO.addCD - The CD adding method

The CD adding method, ask Douglas graciously explained, is a simple method of the table object.

Figure 8: Add CD

```
184
185     @staticmethod
186     def add_cd(row, table):
187         """Adds a dictionary row to the inventory
188
189         Args:
190             row (dictionary): dictionary that holds the name of the ID, cd, and a
191             table (list of dict): 2D data structure (list of dicts) that holds th
192
193         Returns:
194             None.
195
196         """
197         table.append(row)
198         return table
199
```

IO.ask - The CD information entering method

Figure 9 : Add CD

```
200
201     @staticmethod
202     def ask():
203         """Ask user for new ID, CD Title and Artist
204
205         Args:
206             None
207
208         Returns:
209             CD (Object): A dictionary entry with ID (int): integer that holds
210             the ID tag,title (string): string that holds the name of the CD
211             and an artist (string): string that holds the name of the Artist.
212
213         """
214         title = ''
215         artist = ''
216         while True:
217             CD_id = input('Enter ID: ').strip()
218             try:
219                 CD_id = int(CD_id)
220                 break
221             except ValueError:
222                 print('That is not an integer')
223         title = input('What is the CD\'s title? ').strip()
224         artist = input('What is the Artist\'s name? ').strip()
225         CD_row = CD(CD_id, title, artist)
226         return CD_row
```

The CD adding method takes advantage of the CD class by entering information through the CD object.

Summary

As Dirk mentions in his notes and lectures, I do not yet follow appreciate the power of Object Oriented Programming. Dirk mentions that it takes more than just one time using it appreciate its power and beauty. I figure that some of the ways that I can come to appreciate it is to try and building really difficult code without it. Also as Dirk mentions, its usage is probably so standardized and par for the course that it might be difficult to “learn” it again.

Some of the future investigations can include adding the features of pickling to this code in order to increase efficiency of data storage and management. Another feature that can be looked into adding is the feature of preventing duplicate IDs to be stored.

References

<https://www.youtube.com/watch?v=IkXfgP-fAkY&feature=youtu.be>
<https://www.youtube.com/watch?v=KZdFvyCOLUQ&feature=youtu.be>
<https://www.youtube.com/watch?v=qvRyls8NX-E&feature=youtu.be>
<https://www.youtube.com/watch?v=QuwU34OT4XA&feature=youtu.be>
<https://www.youtube.com/watch?v=swAXTwW6xoA&feature=youtu.be>
<https://www.youtube.com/watch?v=5MM6IaESdQ0&feature=youtu.be>
<https://realpython.com/python3-object-oriented-programming/>
<https://www.youtube.com/watch?v=IHaTbJPdB-s&feature=youtu.be>

Appendix: The Code

Made using [Planet B's Syntax Highlighter](#)

```
1. #-----#
2. # Title: Assignmen08.py
3. # Desc: Assignment 08 - Working with classes
4. # Change Log: (Who, When, What)
5. # AYeo, 2020-Mar-10 9:06 PM, Added working CD class
6. # AYeo, 2020-
   Mar-10 1:52 PM, Added Main Body from previous script
7. # AYeo, 2020-Mar-10 2:99 PM, Added Structured Error Handling
8. # AYeo, 2020-
   Mar-15 9:29 PM, Fixed CD class to have property, private methods,
   setters
9. # AYeo, 2020-
   Mar-15 9:29 PM, Edited FileIO to use CD class, HAVE NOT TESTED
10. # AYeo, 2020-
   Mar-15 9:29 PM, Edited FileIO to use CD class, Successfully Teste
   d
11. # AYeo, 2020-
   Mar-16 5:00 PM, Correct FileIO.load_inventory to use pickle, Succ
   essfully Tested
12. # AYeo, 2020-
   Mar-16 5:00 PM, Correct FileIO.save_inventory to use pickle, NOT
   TESTED
13. # AYeo, 2020-
   Mar-16 5:00 PM, Correct FileIO.save_inventory to use pickle, Test
   ed
14.
15.
16. #-----#
17.
18. # -- DATA -- #
19. # DBiesinger, 2030-Jan-01, created file
20. from os import path
21. import pickle
22.
23. strFileName = 'cdInventory.txt'
24. lstOfCDObjects = []
25.
26.
27. class CD:
28.     """Stores data about a CD:
29.
30.     properties:
```

```

31.         cd_id: (int) with CD ID
32.         cd_title: (string) with the title of the CD
33.         cd_artist: (string) with the artist of the CD
34.     methods:
35.
36.     """
37.     # -- Contructor -- #
38.     def __init__(self, cd_id, cd_title, cd_artist):
39.         # -- Attributes -- #
40.         self.__ID = cd_id
41.         self.__title = cd_title
42.         self.__artist = cd_artist
43.
44.     # -- Properties -- #
45.     @property
46.     def ID(self):
47.         return self.__ID
48.
49.     @property
50.     def title(self):
51.         return self.__title.title()
52.
53.     @property
54.     def artist(self):
55.         return self.__artist.title()
56.
57.     @ID.setter
58.     def ID(self, value):
59.         if str(value).isnumeric():
60.             self.__ID = value
61.         else:
62.             raise Exception('The ID must be numeric')
63.
64.     @title.setter
65.     def title(self, value):
66.         if str(value).isnumeric():
67.             raise Exception('The title must be a string')
68.         else:
69.             self.__title = value
70.
71.     @artist.setter
72.     def artist(self, value):
73.         if str(value).isnumeric():
74.             raise Exception('The artist must be a string')
75.         else:
76.             self.__artist = value

```

```

77.
78.     # -- Methods -- #
79.     def __str__(self):
80.         return self.ID
81.         return self.title
82.         return self.artist
83.
84.
85. # -- PROCESSING -- #
86. class FileIO:
87.     """Processes data to and from file:
88.
89.     properties:
90.
91.     methods:
92.         save_inventory(file_name, lst_Inventory): -> None
93.         load_inventory(file_name): -> (a list of CD objects)
94.
95.     """
96.     @staticmethod
97.     def load_inventory(file_name):
98.         """Function to manage data ingestion from file to a list
of dictionaries
99.
100.        Reads the data from file identified by file_name into a 2
D table
101.        (list of dicts) table one line in the file represents one
dictionary row in table.
102.
103.        Args:
104.            file_name (string): name of file used to read the dat
a from
105.            table (list of dict): 2D data structure (list of dict
s) that holds the data during runtime
106.
107.        Returns:
108.            None.
109.
110.        """
111.        try:
112.            with open(file_name, 'rb') as objFile:
113.                table = pickle.load(objFile)
114.                return table
115.        except FileNotFoundError:
116.            pass
117.
118.    @staticmethod

```

```

118.     def save_inventory(file_name, table):
119.         """Writes the inventory of IDs, CD Names, and Artists to
120.         a text file
121.         Args:
122.             file_name (string): The name of the file that it will
123.             write to
124.             table (list of dict): 2D data structure (list of dict
125.             s) that holds the data during runtime
126.         Returns:
127.             None but saves a file in the directory of the python
128.             script
129.         """
130.         with open(file_name, 'wb') as objFile:
131.             table = pickle.dump(table, objFile)
132.
133. # -- PRESENTATION (Input/Output) -- #
134. class IO:
135.     """Handling Input / Output"""
136.     @staticmethod
137.     def print_menu():
138.         """Displays a menu of choices to the user
139.
140.         Args:
141.             None.
142.
143.         Returns:
144.             None.
145.         """
146.
147.         print('Menu\n\n[l] load Inventory from file\n[a] Add CD\n
[i] Display Current Inventory')
148.         print('[d] delete CD from Inventory\n[s] Save Inventory t
o file\n[x] exit\n')
149.
150.     @staticmethod
151.     def menu_choice():
152.         """Gets user input for menu selection
153.
154.         Args:
155.             None.
156.
157.         Returns:

```

```

158.         choice (string): a lower case sting of the users input
159.         out of the choices l, a, i, d, s or x
160.         """
161.         choice = ' '
162.         while choice not in ['l', 'a', 'i', 'd', 's', 'x']:
163.             choice = input('Which operation would you like to perform? [l, a, i, d, s or x]: ').lower().strip()
164.         print() # Add extra space for layout
165.         return choice
166.
167.     @staticmethod
168.     def show_inventory(table):
169.         """Displays current inventory table
170.
171.         Args:
172.             table (list of dict): 2D data structure (list of dictionaries) that holds the data during runtime.
173.
174.         Returns:
175.             None.
176.
177.         """
178.         print('==== The Current Inventory: =====')
179.         print('ID\tCD Title (by: Artist)\n')
180.         for CD in table:
181.             print('{}\t{} (by: {})'
182.                   .format(CD.ID, CD.title, CD.artist))
183.         print('=====')
184.
185.     @staticmethod
186.     def add_cd(row, table):
187.         """Adds a dictionary row to the inventory
188.
189.         Args:
190.             row (dictionary): dictionary that holds the name of the ID, cd, and artist
191.             table (list of dict): 2D data structure (list of dictionaries) that holds the data during runtime
192.
193.         Returns:
194.             None.
195.
196.         """
197.         table.append(row)

```

```

198.         return table
199.
200.
201.     @staticmethod
202.     def ask():
203.         """Ask user for new ID, CD Title and Artist
204.
205.         Args:
206.             None
207.
208.         Returns:
209.             CD (Object): A dictionary entry with ID (int): integer that holds
210.                 the ID tag,title (string): string that holds the name
211.                 of the CD
212.                 and an artist (string): string that holds the name of the Artist.
213.         """
214.         title = ''
215.         artist = ''
216.         while True:
217.             CD_id = input('Enter ID: ').strip()
218.             try:
219.                 CD_id = int(CD_id)
220.                 break
221.             except ValueError:
222.                 print('That is not an integer')
223.         title = input('What is the CD\'s title? ').strip()
224.         artist = input('What is the Artist\'s name? ').strip()
225.         CD_row = CD(CD_id, title, artist)
226.         return CD_row
227.
228. class DataProcessor:
229.     @staticmethod
230.     def delete_cd(intIDDel,table):
231.         """Deletes a CD row from the table
232.
233.         Args:
234.             intIDDel (int): ID which indicate which entry user would like to delete
235.             table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
236.
237.         Returns:
238.             table (list of dict): 2D data structure (list of dicts) that holds the data during runtime

```

```

238.         """
239.         intRowNr = -1
240.         blnCDRemoved = False
241.         for CD in table:
242.             intRowNr += 1
243.             if CD.ID == intIDDel:
244.                 del table[intRowNr]
245.                 blnCDRemoved = True
246.                 break
247.             if blnCDRemoved:
248.                 print('The CD was removed')
249.             else:
250.                 print('Could not find this CD!')
251.         return table
252.
253. # -- Main Body of Script -- #
254. # Load data from file into a list of CD objects on script start
255. FileIO.load_inventory(strFileName)
256.
257. # 2. start main loop
258. while True:
259.     # 2.1 Display Menu to user and get choice
260.     IO.print_menu()
261.     strChoice = IO.menu_choice()
262.
263.     # 3. Process menu selection
264.     # 3.1 process exit first
265.     if strChoice == 'x':
266.         break
267.     # 3.2 process load inventory
268.     if strChoice == 'l':
269.         print('WARNING: If you continue, all unsaved data will be
lost and the Inventory re-loaded from file.')
270.         strYesNo = input('type \'yes\' to continue and reload fro
m file. otherwise reload will be canceled')
271.         if strYesNo.lower() == 'yes':
272.             print('reloading...')
273.             lstOfCDObjects = FileIO.load_inventory(strFileName)
274.             IO.show_inventory(lstOfCDObjects)
275.         else:
276.             input('canceling... Inventory data NOT reloaded. Pres
s [ENTER] to continue to the menu.')
277.             IO.show_inventory(lstOfCDObjects)
278.             continue # start loop back at top.
279.     # 3.3 process add a CD
280.     elif strChoice == 'a':

```

```

281.         # 3.3.1 Ask user for new ID, CD Title and Artist
282.         row = IO.ask()
283.         # 3.3.2 Add item to the table
284.         IO.add_cd(row,lstOfCDObjects)
285.         IO.show_inventory(lstOfCDObjects)
286.         continue # start loop back at top.
287.     # 3.4 process display current inventory
288.     elif strChoice == 'i':
289.         IO.show_inventory(lstOfCDObjects)
290.         continue # start loop back at top.
291.     # 3.5 process delete a CD
292.     elif strChoice == 'd':
293.         # 3.5.1 get Userinput for which CD to delete
294.         # 3.5.1.1 display Inventory to user
295.         IO.show_inventory(lstOfCDObjects)
296.         # 3.5.1.2 ask user which ID to remove
297.         while True:
298.             try:
299.                 intIDDel = int(input('Which ID would you like to
delete? ').strip())
300.                 break
301.             except ValueError:
302.                 print('That is not an integer')
303.         # 3.5.2 search thru table and delete CD
304.         DataProcessor.delete_cd(intIDDel,lstOfCDObjects)
305.         IO.show_inventory(lstOfCDObjects)
306.         continue # start loop back at top.
307.     # 3.6 process save inventory to file
308.     elif strChoice == 's':
309.         # 3.6.1 Display current inventory and ask user for confir
mation to save
310.         IO.show_inventory(lstOfCDObjects)
311.         strYesNo = input('Save this inventory to file? [y/
n] ').strip().lower()
312.         # 3.6.2 Process choice
313.         if strYesNo == 'y':
314.             # 3.6.2.1 save data
315.             FileIO.save_inventory(strFileName, lstOfCDObjects)
316.         else:
317.             input('The inventory was NOT saved to file. Press [EN
TER] to return to the menu.')
318.             continue # start loop back at top.
319.     # 3.7 catch-
all should not be possible, as user choice gets vetted in IO, but
to be save:
320.     else:

```



```
321.     print('General Error')
322.
```

Appendix: The Spyder Output

```
1. runfile('/Users/andyyeo/Google Drive/Intro to Python/Mod_08/
   Assignment08/Assignment_08.py', wdir='/Users/andyyeo/
   Google Drive/Intro to Python/Mod_08/Assignment08')
2. Menu
3.
4. [l] load Inventory from file
5. [a] Add CD
6. [i] Display Current Inventory
7. [d] delete CD from Inventory
8. [s] Save Inventory to file
9. [x] exit
10.
11.
12. Which operation would you like to perform? [l, a, i, d, s or x]:
    a
13.
14.
15. Enter ID: 1
16.
17. What is the CD's title? asdfa
18.
19. What is the Artist's name? vasdc
20. ===== The Current Inventory: =====
21. ID      CD Title (by: Artist)
22.
23. 1      Asdfa (by:Vasdc)
24. =====
25. Menu
26.
27. [l] load Inventory from file
28. [a] Add CD
29. [i] Display Current Inventory
30. [d] delete CD from Inventory
31. [s] Save Inventory to file
32. [x] exit
33.
34.
35. Which operation would you like to perform? [l, a, i, d, s or x]:
    a
36.
37.
38. Enter ID: 2
39.
```

```

40. What is the CD's title? asc
41.
42. What is the Artist's name? casdg
43. ===== The Current Inventory: =====
44. ID      CD Title (by: Artist)
45.
46. 1      Asdfa (by:Vasdc)
47. 2      Asc (by:Casdg)
48. =====
49. Menu
50.
51. [l] load Inventory from file
52. [a] Add CD
53. [i] Display Current Inventory
54. [d] delete CD from Inventory
55. [s] Save Inventory to file
56. [x] exit
57.
58.
59. Which operation would you like to perform? [l, a, i, d, s or x]:
    s
60.
61. ===== The Current Inventory: =====
62. ID      CD Title (by: Artist)
63.
64. 1      Asdfa (by:Vasdc)
65. 2      Asc (by:Casdg)
66. =====
67.
68. Save this inventory to file? [y/n] y
69. Menu
70.
71. [l] load Inventory from file
72. [a] Add CD
73. [i] Display Current Inventory
74. [d] delete CD from Inventory
75. [s] Save Inventory to file
76. [x] exit
77.
78.
79. Which operation would you like to perform? [l, a, i, d, s or x]:
    d
80.
81. ===== The Current Inventory: =====
82. ID      CD Title (by: Artist)
83.

```

```

84. 1      Asdfa (by:Vasdc)
85. 2      Asc (by:Casdg)
86. =====
87.
88. Which ID would you like to delete? 1
89. ===== The Current Inventory: =====
90. ID      CD Title (by: Artist)
91.
92. 2      Asc (by:Casdg)
93. =====
94. Menu
95.
96. [l] load Inventory from file
97. [a] Add CD
98. [i] Display Current Inventory
99. [d] delete CD from Inventory
100.[s] Save Inventory to file
101.[x] exit
102.
103.
104.Which operation would you like to perform? [l, a, i, d, s or x]:
    d
105.
106.===== The Current Inventory: =====
107.ID      CD Title (by: Artist)
108.
109.2      Asc (by:Casdg)
110.=====
111.
112.Which ID would you like to delete? 2
113.===== The Current Inventory: =====
114.ID      CD Title (by: Artist)
115.
116.=====
117.Menu
118.
119.[l] load Inventory from file
120.[a] Add CD
121.[i] Display Current Inventory
122.[d] delete CD from Inventory
123.[s] Save Inventory to file
124.[x] exit
125.
126.
127.Which operation would you like to perform? [l, a, i, d, s or x]:
    1

```

```

128.
129.WARNING: If you continue, all unsaved data will be lost and the I
    nventory re-loaded from file.
130.
131.type 'yes' to continue and reload from file. otherwise reload wil
    l be canceledyes
132.reload...
133.===== The Current Inventory: =====
134.ID      CD Title (by: Artist)
135.
136.1      Asdfa (by:Vasdc)
137.2      Asc (by:Casdg)
138.=====
139.Menu
140.
141.[l] load Inventory from file
142.[a] Add CD
143.[i] Display Current Inventory
144.[d] delete CD from Inventory
145.[s] Save Inventory to file
146.[x] exit
147.
148.
149.Which operation would you like to perform? [l, a, i, d, s or x]:
    i
150.
151.===== The Current Inventory: =====
152.ID      CD Title (by: Artist)
153.
154.1      Asdfa (by:Vasdc)
155.2      Asc (by:Casdg)
156.=====
157.Menu
158.
159.[l] load Inventory from file
160.[a] Add CD
161.[i] Display Current Inventory
162.[d] delete CD from Inventory
163.[s] Save Inventory to file
164.[x] exit
165.
166.
167.Which operation would you like to perform? [l, a, i, d, s or x]:
    x

```

Appendix: The Terminal Output

```
1. Last login: Wed Mar 11 14:14:33 on ttys000
2. ^[[A^[[A
3. The default interactive shell is now zsh.
4. To update your account to use zsh, please run `chsh -s /bin/
   zsh`.
5. For more details, please visit https://support.apple.com/kb/
   HT208050.
6. (base) Andys-MacBook-Pro:~ andyyeo$ cd desktop/assignment08
7. (base) Andys-MacBook-Pro:assignment08 andyyeo$ zip -r dir.zip . -
   x ".*" -x "_MAXOSX"
8.   adding: Assignment08 Knowledge Document .pdf (deflated 29%)
9.   adding: Assignment_08_Starter.py (deflated 71%)
10. (base) Andys-MacBook-Pro:assignment08 andyyeo$
11. [Restored Mar 16, 2020 at 7:00:13 PM]
12. Last login: Wed Mar 11 15:00:02 on ttys000
13.
14. The default interactive shell is now zsh.
15. To update your account to use zsh, please run `chsh -s /bin/
   zsh`.
16. For more details, please visit https://support.apple.com/kb/
   HT208050.
17. (base) Andys-MacBook-Pro:assignment08 andyyeo$ clear
18.
19.
20.
21.
22.
23.
24. (base) Andys-MacBook-Pro:assignment08 andyyeo$ cd desktop/
   assignment08
25. -bash: cd: desktop/assignment08: No such file or directory
26. (base) Andys-MacBook-
   Pro:assignment08 andyyeo$ python Assignment_08.py
27. Menu
28.
29. [l] load Inventory from file
30. [a] Add CD
31. [i] Display Current Inventory
32. [d] delete CD from Inventory
33. [s] Save Inventory to file
34. [x] exit
35.
36. Which operation would you like to perform? [l, a, i, d, s or x]:
   a
```

```

37.
38. Enter ID: 1
39. What is the CD's title? asdfd
40. What is the Artist's name? casd
41. ===== The Current Inventory: =====
42. ID  CD Title (by: Artist)
43.
44. 1  Asdfd (by:Casd)
45. =====
46. Menu
47.
48. [l] load Inventory from file
49. [a] Add CD
50. [i] Display Current Inventory
51. [d] delete CD from Inventory
52. [s] Save Inventory to file
53. [x] exit
54.
55. Which operation would you like to perform? [l, a, i, d, s or x]:
    a
56.
57. Enter ID: 2
58. What is the CD's title? asdfasdg
59. What is the Artist's name? asgdg
60. ===== The Current Inventory: =====
61. ID  CD Title (by: Artist)
62.
63. 1  Asdfd (by:Casd)
64. 2  Asdfasdg (by:Asgdg)
65. =====
66. Menu
67.
68. [l] load Inventory from file
69. [a] Add CD
70. [i] Display Current Inventory
71. [d] delete CD from Inventory
72. [s] Save Inventory to file
73. [x] exit
74.
75. Which operation would you like to perform? [l, a, i, d, s or x]:
    s
76.
77. ===== The Current Inventory: =====
78. ID  CD Title (by: Artist)
79.
80. 1  Asdfd (by:Casd)

```

```

81. 2   Asdfasd (by:Asgdg)
82. =====
83. Save this inventory to file? [y/n] y
84. Menu
85.
86. [l] load Inventory from file
87. [a] Add CD
88. [i] Display Current Inventory
89. [d] delete CD from Inventory
90. [s] Save Inventory to file
91. [x] exit
92.
93. Which operation would you like to perform? [l, a, i, d, s or x]:
    d
94.
95. ===== The Current Inventory: =====
96. ID   CD Title (by: Artist)
97.
98. 1   Asdfd (by:Casd)
99. 2   Asdfasd (by:Asgdg)
100. =====
101. Which ID would you like to delete? 1
102. ===== The Current Inventory: =====
103. ID   CD Title (by: Artist)
104.
105. 2   Asdfasd (by:Asgdg)
106. =====
107. Menu
108.
109. [l] load Inventory from file
110. [a] Add CD
111. [i] Display Current Inventory
112. [d] delete CD from Inventory
113. [s] Save Inventory to file
114. [x] exit
115.
116. Which operation would you like to perform? [l, a, i, d, s or x]:
    d
117.
118. ===== The Current Inventory: =====
119. ID   CD Title (by: Artist)
120.
121. 2   Asdfasd (by:Asgdg)
122. =====
123. Which ID would you like to delete? 2
124. ===== The Current Inventory: =====

```



```

125.ID  CD Title (by: Artist)
126.
127.=====
128.Menu
129.
130.[l] load Inventory from file
131.[a] Add CD
132.[i] Display Current Inventory
133.[d] delete CD from Inventory
134.[s] Save Inventory to file
135.[x] exit
136.
137.Which operation would you like to perform? [l, a, i, d, s or x]:
    l
138.
139.WARNING: If you continue, all unsaved data will be lost and the I
    nventory re-loaded from file.
140.type 'yes' to continue and reload from file. otherwise reload wil
    l be canceledyes
141.reload...
142.===== The Current Inventory: =====
143.ID  CD Title (by: Artist)
144.
145.1   Asdfd (by:Casd)
146.2   Asdfasdg (by:Asgdg)
147.=====
148.Menu
149.
150.[l] load Inventory from file
151.[a] Add CD
152.[i] Display Current Inventory
153.[d] delete CD from Inventory
154.[s] Save Inventory to file
155.[x] exit
156.
157.Which operation would you like to perform? [l, a, i, d, s or x]:
    i
158.
159.===== The Current Inventory: =====
160.ID  CD Title (by: Artist)
161.
162.1   Asdfd (by:Casd)
163.2   Asdfasdg (by:Asgdg)
164.=====
165.Menu
166.

```

```
167.[l] load Inventory from file
168.[a] Add CD
169.[i] Display Current Inventory
170.[d] delete CD from Inventory
171.[s] Save Inventory to file
172.[x] exit
173.
174.Which operation would you like to perform? [l, a, i, d, s or x]:
    x
175.
176.(base) Andys-MacBook-Pro:assignment08 andyyeo$
177.
```