Graham Byron

23 March 2022

IT FDN 110: Introduction to Programming (Python)

Assignment 08

Assignment 08: Objects and Classes

Intro

Assignment 08 centered around the concept of Object Oriented Programming (OOP), essentially meaning that everything is an object. To go about understanding this concept, emphasis was placed on creating classes to define objects, writing methods and creation attributes for said objects, instantiating objects, and restricting access to attributes of the object. These topics were broken down step by step provided by several examples that built upon one another, and were further established through the subsequent labs that followed the same cadence. This was built on through the CD inventory work that had already been established, so I was well familiar with the objective and intention of the code. Once this was worked through, we were tasked with adding code to a starter base that was provided to us. This was completed through use of previously written code as well as the new content that was covered in this module. Lastly, the code was again posted as a repository to the GitHub page.

Topic

Several sub topics were discussed in this module to break down the ideas stated above, starting with classes. Classes are what provide the blueprint (or base) for the object. The main intention is to provide the starting point to add and customize the class to the needed action. Building on classes are fields, which function as a data storage for the class itself. While it will evolve in the future, these fields are created the same way that variables are. The next item that was discussed was constructors. These are a specific method that is invoked when creating an object and are a good way to ensure that proper data types are used as well as pre-population of fields. This is established through __init__(). In contrast to this is a destructor which sounds exactly like the role it takes. A destructor is used when an object is destroyed and can be called by __def__(self) for example. One important note that is made regarding the diction in this module is regarding the keyword *self.* While it is not officially the word that is necessary in the function call, it is universally used and essentially required to be used.

Continuing on are the use of attributes, which are internal fields/variables that hold data. One important item to note is that attributes function as variables, meaning that there is no control over how they change during the code's runtime unless specifically established. Finally are properties,

which helps control and enforce interaction of values through built in control mechanisms. This is established usually through a getter (that accesses) and a setter (or mutator).

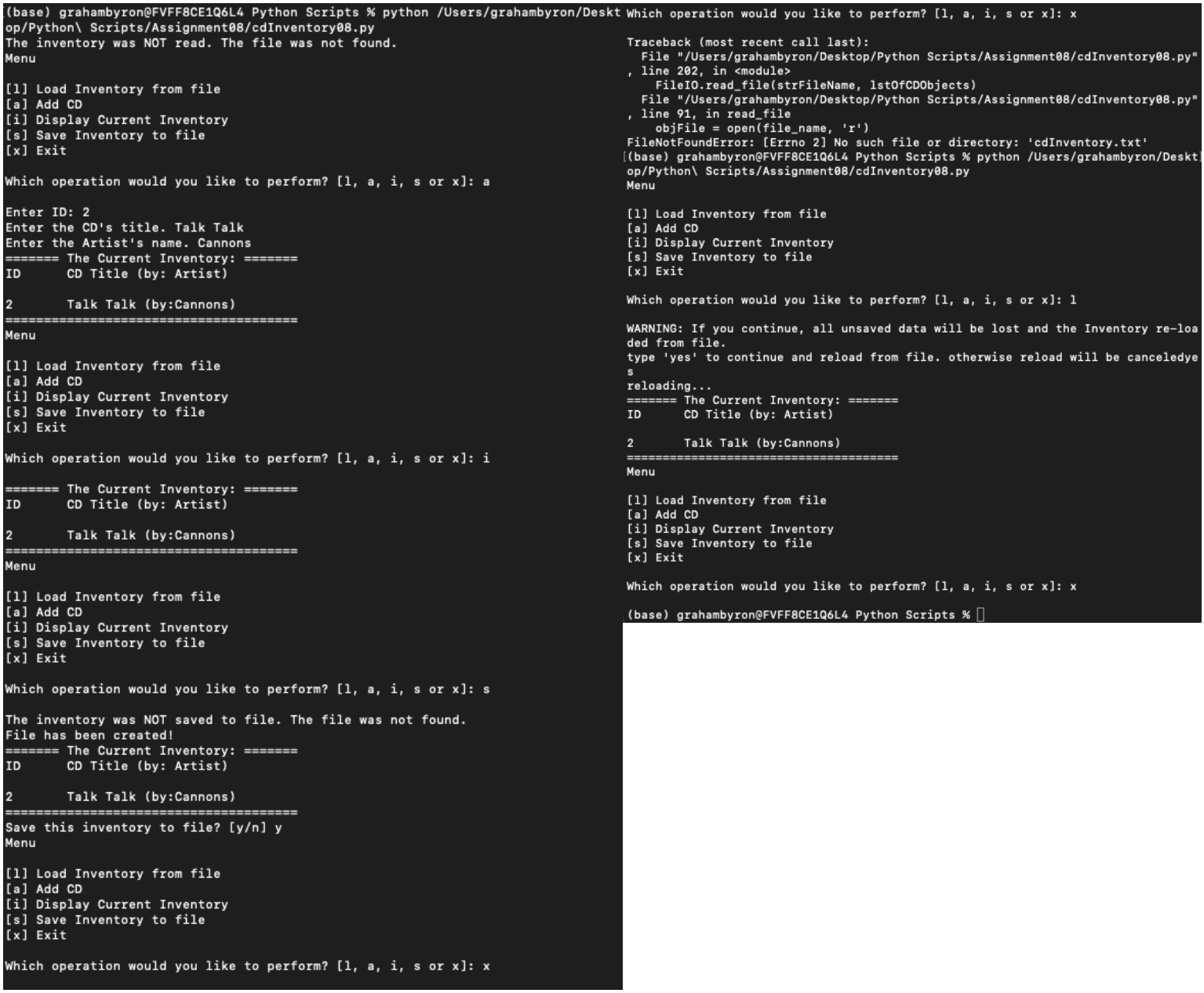
Once this is established, methods can be used in order to organize all of these statements into blocks that can be run by calling upon the name of the method. Further, __str__() is used to return some or all of the objects in the object as a string. If this is not defined Python will inherit the method from the class object.

Private methods can be called upon to have methods for internal processes only. This is the same quality that attributes hold. The way to establish this is by prefacing the method with two underscores. Once a method is private it can only be read through a public method that explicitly calls for it. After this was discussed, we were given a base set of instructions and TODOs in order to build out a script including parts of the new ideas discussed above. This build out required use of previously established script from the CD Inventory as well as additional use of classes, constructors, and attributes. This required altering previous code where necessary to hold pivotal variable names in order to function. The output of this code is shown below:

Python Output of Code:

```
In [42]: runfile('/Users/grahambyron/Desktop/cdInventory08.py', wdir='/Users/
grahambyron/Desktop')

The inventory was NOT read. The file was not found.
Menu

[l] Load Inventory from file
[a] Add CD
[i] Display Current Inventory
[s] Save Inventory to file
[x] Exit


Which operation would you like to perform? [l, a, i, s or x]: a


Enter ID: 1

Enter the CD's title. Shadows

Enter the Artist's name. Bleachers
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Shadows (by:Bleachers)
=====================================
Menu

[l] Load Inventory from file
[a] Add CD
[i] Display Current Inventory
[s] Save Inventory to file
[x] Exit


Which operation would you like to perform? [l, a, i, s or x]: i

======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Shadows (by:Bleachers)
=====================================
Menu

[l] Load Inventory from file
[a] Add CD
[i] Display Current Inventory
[s] Save Inventory to file
[x] Exit


Which operation would you like to perform? [l, a, i, s or x]: s


The inventory was NOT saved to file. The file was not found.
File has been created!
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Shadows (by:Bleachers)
=====================================

Save this inventory to file? [y/n] y
Menu

[l] Load Inventory from file
[a] Add CD
[i] Display Current Inventory
[s] Save Inventory to file
[x] Exit


Which operation would you like to perform? [l, a, i, s or x]: x

Traceback (most recent call last):

  File "/Users/grahambyron/Desktop/cdInventory08.py", line 282, in <module>
    FileIO.read_file(strFileName, lstOfCDObjects)

  File "/Users/grahambyron/Desktop/cdInventory08.py", line 91, in read_file
    objFile = open(file_name, 'r')

FileNotFoundError: [Errno 2] No such file or directory: 'cdInventory.txt'


In [43]: runfile('/Users/grahambyron/Desktop/cdInventory08.py', wdir='/Users/
grahambyron/Desktop')
Menu

[l] Load Inventory from file
[a] Add CD
[i] Display Current Inventory
[s] Save Inventory to file
[x] Exit


Which operation would you like to perform? [l, a, i, s or x]: l

WARNING: If you continue, all unsaved data will be lost and the Inventory re-loaded fro
file.

type 'yes' to continue and reload from file. otherwise reload will be canceledyes
reloading...
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Shadows (by:Bleachers)
=====================================
Menu

[l] Load Inventory from file
[a] Add CD
[i] Display Current Inventory
[s] Save Inventory to file
[x] Exit


Which operation would you like to perform? [l, a, i, s or x]: x


In [44]:
```

Terminal Output of Code:



Most of the output went without a hitch, however closing the program for the first time after creating the file did yield me that error. While technically I was already trying to exit the program, this was not ideal.

Conclusion

Assignment 08 revolved around the idea and implementation of Object Oriented Programming (OOP). This was further built on through the use for class creation, method writing, object instantiation, and object access restriction. The knowledge show required use of previous code and addition of the freshly covered details. While most ran smoothly, I did run into one issue surrounding closure of the script after just creating and saving the file. GitHub: https://github.com/gwiby123/Assignment08

Appendix:

```
#------------------------#
# Title: Assignment08.py
# Desc.: Working with classes and functions, and OOP
# Change Log: (Who, When, What)
# <GByron, 2022-Mar-23 Edited, added, and adjusted code>
#------------------------#

# -- DATA -- #
strFileName = 'cdInventory.txt'
lstOfCDObjects = []

class CD:
    """Stores data about a CD:
    properties:
        cdID: (int) with CD ID
        cdTitle: (string) with the title of the CD
        cdArtist: (string) with the artist of the CD
    methods:
    """
    #-Fields-#
        # N/A
    #-Constructor-#
    def __init__(self, cdID, cdTitle, cdArtist):
    #-Attributes-#
        self.cdID = cdID
        self.cdTitle = cdTitle
        self.cdArtist = cdArtist

    def cdID(self):
        return self.cdID

    def cdTitle(self):
        return self.cdTitle

    def cdArtist(self):
        return self.cdArtist

    """Function to add CD to table

    Args:
        None

    Returns:
        None

    """

    def adding_cd():
        cd_lst= IO.add_cd()
        dicRow = {'ID': cd_lst[0], 'Title': cd_lst[1], 'Artist': cd_lst[2]}
        lstOfCDObjects.append(dicRow)


# -- PROCESSING -- #
```

```
55          class FileIO:
56              """Processes data to and from file:
57              properties:
58              methods:
59                  save_inventory(file_name, lst_Inventory): -> None
60                  load_inventory(file_name): -> (a list of CD objects)
61              """
62
63              """Processing the data to and from text file"""
64
65              @staticmethod
66              def read_file(file_name, table):
67
68                  """Function to manage data ingestion from file to a list of dictionaries
69                  Reads the data from file identified by file_name into a 2D table
70                  (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
74                  Returns:
75                      None.
76                  """
77
78                  try:
79                      open('CDInventory.txt')
80                  except FileNotFoundError:
81                      input('The inventory was NOT read. The file was not found.')
82                  table.clear()
83                  objFile = open(file_name, 'r')
84                  for line in objFile:
85                      data = line.strip().split(',')
86                      dicRow = {'ID': int(data[0]), 'Title': data[1], 'Artist': data[2]}
87                      table.append(dicRow)
88                  objFile.close()
89
90              def save_data(data, strFileName):
91
92                  """Function to save the CD to a file
93
94                  Args:
95                      file_name: the name of the file the CD information will be saved to
96                      table: the list of the CD information that will be saved
97
98                  Returns:
99                      None
100
101                 """
102
103                 if strYesNo == 'y':
104                     objFile = open(strFileName, 'w')
105                     for row in lstOfCDObjects:
106                         lstValues = list(row.values())
107                         lstValues[0] = str(lstValues[0])
108                         objFile.write(','.join(lstValues) + '\n')
```

```
109                 objFile.close()
110             else:
111                 input('The inventory was NOT saved to file. Press [ENTER] to return to the menu.')
112 
113     # -- PRESENTATION (Input/Output) -- #
114 
115     class IO:
116 
117         """Handling Input / Output"""
118 
119         @staticmethod
120         def print_menu():
121             """Displays a menu of choices to the user
122             Args:
123                 None.
124             Returns:
125                 None.
126             """
127 
128             print('Menu\n\n[l] Load Inventory from file\n[a] Add CD\n[i] Display Current Inventory')
129             print('[s] Save Inventory to file\n[x] Exit\n')
130 
131         def menu_choice():
132             """Gets user input for menu selection
133             Args:
134                 None.
135             Returns:
136                 choice (string): a lower case sting of the users input out of the choices l, a, i, d, s or x
137             """
138             choice = ' '
139             while choice not in ['l', 'a', 'i', 's', 'x']:
140                 choice = input('Which operation would you like to perform? [l, a, i, s or x]: ').lower().strip()
141             print()
142             return choice
143 
144         @staticmethod
145         def show_inventory(table):
146             """Displays current inventory table
147             Args:
148                 table (list of dict): 2D data structure (list of dicts) that holds the data during runtime.
149             Returns:
150                 None.
151             """
152             print('======= The Current Inventory: =======')
153             print('ID\tCD Title (by: Artist)\n')
154             for row in table:
155                 print('{}\t{} (by:{})'.format(*row.values()))
156             print('======================================')
157 
158 
159             """Adds CD of user choice to the list, does not save the CD to memory
160 
161             Args: None
162 
```

```
163             Returns:
164                 strID, asks user for deired value to identity ID
165                 strTitle, asks user for title of CD associated with ID
166                 strArtist, asks user for CD artist
167 
168             """
169 
170         def add_cd():
171             while True:
172                 strID = input('Enter ID: ').strip()
173                 try:
174                     intID = int(strID)
175                     break
176                 except ValueError:
177                     print('That is not an integer!')
178             strTitle = input('Enter the CD\'s title. ').strip()
179             stArtist = input('Enter the Artist\'s name. ').strip()
180             return [intID, strTitle, stArtist]
181 
182     # -- Main Body of Script -- #
183 
184     try:
185         FileIO.read_file(strFileName, lstOfCDObjects)
186 
187     finally:
188         while True:
189 
190             IO.print_menu()
191             strChoice = IO.menu_choice()
192 
193             # let user exit program
194             if strChoice == 'x':
195                 break
196 
197             # let user load inventory from file
198             if strChoice == 'l':
199                 print('WARNING: If you continue, all unsaved data will be lost and the Inventory re-loaded from file.')
200                 strYesNo = input('type \'yes\' to continue and reload from file. otherwise reload will be canceled')
201                 if strYesNo.lower() == 'yes':
202                     print('reloading...')
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
209 
210             # let user add data to the inventory
211             elif strChoice == 'a':
212                 CD.adding_cd()
213                 IO.show_inventory(lstOfCDObjects)
214                 continue
215 
216             # show user current inventory
```

```
217             elif strChoice == 'i':
218                 IO.show_inventory(lstOfCDObjects)
219                 continue
220
221             # let user save inventory to file
222             elif strChoice == 's':
223                 try:
224                     open('CDInventory.txt')
225                 except FileNotFoundError:
226                     input('The inventory was NOT saved to file. The file was not found.')
227                     print('File has been created!')
228                 IO.show_inventory(lstOfCDObjects)
229                 strYesNo = input('Save this inventory to file? [y/n] ').strip().lower()
230                 FileIO.save_data(lstOfCDObjects, strFileName)
231                 continue
232
233         else:
234             print('General Error')
```