Eliana Arias-Dotson
Created: 2020, March 16th 2020
Foundations of Programming, Python
Assignment 08

Introduction

The purpose of this assignment was to start simplifying our codes by using object Oriented Programming creating Classes to define objects, writing methods , defining attributes with properties that enforce how they are use in the methods that use them.

Starter Pseudocode

For this assignment we started with a pseducode that intents to guide us in the construction of the code we have been working but implementing the new concepts of Classes and attributes.

Populating the Pseudocode:

To start, I used the provided pseudo-code and took some of the pieces from the previous assignments and tried to make sense on how they are handled when attributes are previously defined. Here the basic structure of the class definition working with OOP

Class
##Fields
Constructors
Atributes
##Properties
##Methods

Defining the **Class** , **Constructors** the __init__ & __str__ methods, **Attributes & Properties in** the CD class

The first part of the assignment was to define the attributes by using the __init__ method which works as a constructor to initialize/create the variables that can be instatiated. When defining this __init__ function, this function is automatically called whenever a new object of the class is instantiated. The parameter self, that is, because when a fucntion is called using and object, the object itself is passed automatically to the function as an argument. That's why the very first argument in the function must be the object itself, which is conventionally called 'self'. (1-3)

For our case the class CD holds three of the main values that need to be initiated for the Magic Inventory, cd_id, cd_title and cd_artist. After running the labs it made more sense to me to write the attributes as private than public as below.

(1) https://intellipaat.com/blog/tutorial/python-tutorial/python-classes-and-objects/
(2) https://www.tutorialsteacher.com/python/private-and-protected-access-modifiers-in-python
(3) https://www.bogotobogo.com/python/python_classes_instances.php

Although the material provided suggested to use the __str__ method to test something useful I implemented just to return. The **__str__** is run automatically whenever an instance is converted to its print string[4].

```
# TO-DOne Add Code to the CD class

class CD:
    """Stores data about a CD:

    properties:
        cd_id: (int) with CD ID
        cd_title: (string) with the title of the CD
        cd_artist: (string) with the artist of the CD
    methods:
        TBI
    """

    #Constructor
    def __init__(self, cd_id,cd_title,cd_artist):
        #----Protected Attributes----#
        self.__cd_id = int(cd_id)
        self.__cd_title = cd_title
        self.__cd_artist = cd_artist
   #Properties
    @property
    def cd_id(self):
        return self.__cd_id

    @cd_id.setter
    def cd_id(self,value):
        try:
            self.__cd_id = int(value)
        except ValueError:
            raise ValueError("CD ID must be an integer")

    @property
    def cd_title(self):
        return self.__cd_title

    @property
    def cd_artist(self):
        return self.__cd_artist

    def __str__(self):
        return int(self.__cd_id)+'/'+str(self.__cd_title)
+'/'+str(self.__artist)


#Validating: #A simple validation to retrieve the attributes
newcd= CD(1,"Reik","Ojos Negros")
#print(newcd.cd_id,newcd.cd_artist,newcd.cd_title)
#print(newcd.cd_id)
print('{}\t{} (by:{})'.format(newcd.cd_id, newcd.cd_title, newcd.cd_artist))
```

Figure 1. Definition of Class CD, its attributes and properties with simple validation

(1) https://intellipaat.com/blog/tutorial/python-tutorial/python-classes-and-objects/
(2) https://www.tutorialsteacher.com/python/private-and-protected-access-modifiers-in-python
(3) https://www.bogotobogo.com/python/python_classes_instances.php

Working with Files & static methods

The second part of the assignment was to define the codes for to load and save the inventory from and to a file defined in the Class FileIO. For this part I recycled the existing routine from Assigment 06 and modified it to read the attributes not from the dictionary as done before but from the attributes defined in the CD class. Although I was able to access the attributes from the class and got the functions load and save inventory, I know most likely there is a more elegant way to access all the attributes with a proper method. For simplicity I removed the docstrings documenting the routine.

```
@staticmethod
    def load_inventory(file_name,lst_Inventory):
        try:
            objFile = open(file_name, 'r')
            lst_Inventory.clear()  #
           for attr in objFile:
                cd_attr=(attr.cd_id, attr.cd_title,attr.cd_artist)
                lst_Inventory.append(cd_attr)
            objFile.close()
        except FileNotFoundError as err:
            print(err, "The file {} could not be loaded".format(file_name))
        finally:
            return lst_Inventory


# TO-DOne Add code to process data to a file - WRITE
 ###
    @staticmethod
    def save_inventory(file_name, lst_Inventory):
        with open(file_name, 'w') as objFile:
            for row in lst_Inventory:
                   newcd=(str(row.cd_id),row.cd_artist,row.cd_title)
                objFile.write(','.join(newcd) + '\n')
```

Figure 2. Definition of Class CD, static methods for the FileIO class

Presentation of the Code

In this part of the assigment, I used the routines defined in previous assignments making some small adjustments to read from the list of attributes instead of the dictionary list and values.

(1) https://intellipaat.com/blog/tutorial/python-tutorial/python-classes-and-objects/
(2) https://www.tutorialsteacher.com/python/private-and-protected-access-modifiers-in-python
(3) https://www.bogotobogo.com/python/python_classes_instances.php

```
# -- PRESENTATION (Input/Output) -- #
class IO:
   # TODO add docstring
   # TODO add code to show menu to user
   # TODO add code to captures user's choice
   # TODO add code to display the current data on screen
   # TODO add code to get CD data from user
   pass
```

```
class IO:
    @staticmethod
    def print_menu():

        print('Menu\n\n[l] load Inventory from file\n[a] Add CD\n[i] Display
Current Inventory')
        print('[d] delete CD from Inventory\n[s] Save Inventory to file\n[x]
exit\n')


    @staticmethod
    def menu_choice():
        choice = ' '
        try:
            choice = input('Which operation would you like to perform? [l, a,
i, d, s or x]: ').lower().strip()
            while choice not in ['l', 'a', 'i', 'd', 's', 'x']:
                raise ValueError('That is not a valid choice!')
        except ValueError as e:
            print(type(e))
            print('Please enter one of the offered options from menu')
        else:
            print("Thank you for entering a valid choise, please continue:")
            return choice


   @staticmethod
    def show_inventory(lst_Inventory):
        print('======= The Current Inventory: =======')
        print('ID\tCD Title (by: Artist)\n')
        for row in lst_Inventory:
            print('{}\t{} (by:{})'.format(row.cd_id, row.cd_title,
row.cd_artist))


        print('======================================')
```

Figure 3. Definition of Class IO,to print_menu, display_choices and show_inventory

(1) https://intellipaat.com/blog/tutorial/python-tutorial/python-classes-and-objects/
(2) https://www.tutorialsteacher.com/python/private-and-protected-access-modifiers-in-python
(3) https://www.bogotobogo.com/python/python_classes_instances.php

Adding the Data Processor class:

In this case again, the list of attributes were access by instatiating the class as if it was a function passing the arguments defined in the __init__ () method.

```
class DataProcessor():
   @staticmethod
   def add_new(lst_Inventory,cd_id, cd_artist, cd_title):
       new_cd = CD(cd_id, cd_title, cd_artist)
       lst_Inventory.append(new_cd)
       return lst_Inventory
```

Figure 4. Definition of Class for Data Processor to add a new_cd entry, calling the attributes defined on the CD Class

The Main function __main__

I used the __main__ function as a starting point for the execution of the program, although it does not need to be necessary as I tried with or with out it seems to define better where the program executes all the methods previously defined.

```
# -- Main Body of Script -- #

def main():
    print("Hello, world")
    if __name=="__main__":
        main()
    # -- DATA -- #
    strFileName = 'cdInventory.txt'
    lstOfCDObjects = []
    strChoice = ''
    lstOfCDObjects = FileIO.load_inventory(strFileName,lstOfCDObjects)

# Display menu to user
    # show user current inventory
        # 2. start main loop
    while True:
        # 2.1 Display Menu to user and get choice
        IO.print_menu()
        strChoice = IO.menu_choice()

# let user add data to the inventory
        if strChoice == 'a':
            # 3.3.1 Ask user for new ID, CD Title and Artist
            strID, strTitle, strArtist = IO.get_CDUserInput()
            # 3.3.2 Add item to the table
            DataProcessor.add_new(lstOfCDObjects, strID, strTitle, strArtist)
            IO.show_inventory(lstOfCDObjects)
            continue # start loop back at top.

# let user save inventory to file
```

(1) https://intellipaat.com/blog/tutorial/python-tutorial/python-classes-and-objects/
(2) https://www.tutorialsteacher.com/python/private-and-protected-access-modifiers-in-python
(3) https://www.bogotobogo.com/python/python_classes_instances.php

```
        elif strChoice == 's':
            # 3.6.1 Display current inventory and ask user for confirmation
to save
            IO.show_inventory(lstOfCDObjects)
            strYesNo = input('Save this inventory to file? [y/n]
').strip().lower()
            # 3.6.2 Process choice
            if strYesNo == 'y':
                # 3.6.2.1 save data
                FileIO.save_inventory(strFileName, lstOfCDObjects)
            else:
                input('The inventory was NOT saved to file. Press [ENTER] to
return to the menu.')
            continue  # start loop back at top.

# let user load inventory from file
        elif strChoice == 'l':
            print('WARNING: If you continue, all unsaved data will be lost
and the Inventory re-loaded from file.')
            strYesNo = input('type \'yes\' to continue and reload from file.
otherwise reload will be canceled. ')
            if strYesNo.lower() == 'yes':
                print('reloading...')
                lstOfCDObjects =
FileIO.load_inventory(strFileName,lstOfCDObjects)
                IO.show_inventory(lstOfCDObjects)
            else:
                input('canceling... Inventory data NOT reloaded. Press
[ENTER] to continue to the menu.')
                IO.show_inventory(lstOfCDObjects)
            continue  # start loop back at top.

 # process display current inventory
        elif strChoice == 'i':
            IO.show_inventory(lstOfCDObjects)
            continue  # start loop back at top.

# let user exit program
        elif strChoice == 'x':
            break
        else:
            print('General Error')

main()

("\n\nPress the enter key to exit.")
```

Figure5. Main Function definition where all choices are executed and attributes and methods from the Classes are being called to execute.

(1) https://intellipaat.com/blog/tutorial/python-tutorial/python-classes-and-objects/
(2) https://www.tutorialsteacher.com/python/private-and-protected-access-modifiers-in-python
(3) https://www.bogotobogo.com/python/python_classes_instances.php

```
Menu

[l] load Inventory from file
[a] Add CD
[i] Display Current Inventory
[d] delete CD from Inventory
[s] Save Inventory to file
[x] exit


Which operation would you like to perform? [l, a, i, d, s or x]: l
Thank you for entering a valid choise, please continue:
WARNING: If you continue, all unsaved data will be lost and the Inventory re-
loaded from file.
```

Figure 6. Execution of option for loading inventory

```
Which operation would you like to perform? [l, a, i, d, s or x]: s
Thank you for entering a valid choise, please continue:
======= The Current Inventory: =======
ID      CD Title (by: Artist)

2       La vida misma (by:Aprendiendo a querer)
=======================================

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

Figure 7. Execution of option for saving entry

```
Which operation would you like to perform? [l, a, i, d, s or x]: a
Thank you for entering a valid choise, please continue:

Enter ID: 2

What is the CD's title? Aprendiendo a querer

What is the Artist's name? La vida misma
======= The Current Inventory: =======
ID      CD Title (by: Artist)

2       La vida misma (by:Aprendiendo a querer)
=======================================
```

Figure 8. Execution of option for adding new entry

(1) https://intellipaat.com/blog/tutorial/python-tutorial/python-classes-and-objects/
(2) https://www.tutorialsteacher.com/python/private-and-protected-access-modifiers-in-python
(3) https://www.bogotobogo.com/python/python_classes_instances.php

```
Which operation would you like to perform? [l, a, i, d, s or x]: i
Thank you for entering a valid choise, please continue:
======= The Current Inventory: =======
ID      CD Title (by: Artist)

2       La vida misma (by:Aprendiendo a querer)
=======================================
```

Figure 9. Execution of displaying inventory

```
Which operation would you like to perform? [l, a, i, d, s or x]: x
Thank you for entering a valid choise, please continue:
```

Figure 10. Execution of displaying Exiting

```
In [58]: more CDInventory.txt
2,Aprendiendo a querer,La vida misma
```

Figure 11.Entry saved in file CD Inventory

To Learn and Improve

There has been a step curve understanding each part of writing a code. I think the purpose of this assignment was to start putting all those pieces together and learn efficient ways to incorporate those previous concepts and add the value of the Object Oriente Programming (OOP), which seems to be a way to organize the script around the object with specific attributes and properties created.
I am aware, I did not take full advantage of incorporating the OOP into the routine so I look forward to read, learn and practice the proper use.

Github: https://github.com/elidot/Assignment08.git

(1) https://intellipaat.com/blog/tutorial/python-tutorial/python-classes-and-objects/
(2) https://www.tutorialsteacher.com/python/private-and-protected-access-modifiers-in-python
(3) https://www.bogotobogo.com/python/python_classes_instances.php

Useful resources

Python OOP Tutorial 1: Classes and Instances

https://www.youtube.com/watch?v=ZDa-Z5JzLYM&feature=emb_rel_err
https://www.tutorialsteacher.com/python/property-decorator
https://docs.python.org/2/library/exceptions.html
https://towardsdatascience.com/how-to-define-custom-exception-classes-in-python-bfa346629bca
https://realpython.com/lessons/how-and-when-use-__str__/
https://www.youtube.com/watch?v=Un_8afoyExI

(1) https://intellipaat.com/blog/tutorial/python-tutorial/python-classes-and-objects/
(2) https://www.tutorialsteacher.com/python/private-and-protected-access-modifiers-in-python
(3) https://www.bogotobogo.com/python/python_classes_instances.php