Cheryl Montejo

August 28, 2022

Foundations of Programming, Python

Assignment 8

CD Inventory: Creating Classes

# Introduction

For this assignment, I will explain the steps I took to update an existing Python script for a CD inventory system to create a class specifically for CDs.

# Learning Through the Module

For this assignment, I read through the module labs prior to trying the assignment. For this assignment, I used the module to guide me, but I mostly relied on the internet for examples and trial and error to complete the assignment (W3Schools, <https://www.w3schools.com/python/python_classes.asp>) (external site)[[1]](#footnote-1) and (GeeksForGeeks, <https://www.geeksforgeeks.org/python-classes-and-objects/>) (external site)[[2]](#footnote-2).

# Drafting the Script

For this assignment, we were asked to rebuild our script from previous assignments to accommodate a custom class.

To draft the script, I pulled sections from previous iterations of the CDInventory script. Based on the variable for the file name using a txt file, I pulled sections of CDInventory script from Assignment 6 since the inventory in this situation wouldn’t be pickled. I did make sure to bring over the error handling from Assignment 7. I also removed the portions of the script related to deletion since that wasn’t called for in the psuedoscript. For setting up the CD class, I followed along with the example from the module.

# Running the Script in Spyder and Anaconda Prompt

For running the script in Spyder, I launched Spyder and opened CD\_Inventory.py (File -> Open…->CD\_Inventory.py). The error handling around having an existing txt file automatically ran when I ran the script. Through the menu, I tested out the options. I started by loading multiple CDs into the inventory and then saving it to a txt file. I ran the script again and loaded the inventory from the txt file to make sure the inventory would load at the start. I also tested out the error handling for the menu and CD ID inputs. I then deleted the text file so I could start from scratch while testing the script in the Terminal console.

Text

Description automatically generated

Figure - Screenshot showing CD\_Inventory.py in Spyder being used attempt the load the inventory and to add CDs

Text

Description automatically generated

Figure - Screenshot showing CD\_Inventory.py in Spyder being used to save and show the inventory

Graphical user interface, application

Description automatically generated with medium confidence

Figure - Screenshot showing the data in the .txt file

Text

Description automatically generated

Figure - Screenshot showing CD\_Inventory.py in Spyder being used to reload the inventory and exit

Text

Description automatically generated

Figure - Screenshot showing CD\_Inventory.py in Spyder being used to reload the inventory and demonstrate error handling

For running the script in Anaconda Prompt, I opened the Terminal console and navigated to the correct folder using the cd command. I then used the python command with the file name, CDInventory.py, to run the script. Using the menu, I repeated the steps from when I tested the script in Spyder.

Text

Description automatically generated

Figure - Screenshot showing CD\_Inventory.py in the terminal being used to attempt to load the inventory and add CDs

Text

Description automatically generated

Figure - Screenshot showing CD\_Inventory.py in the terminal being used to show and save the inventory and add another CD

Text

Description automatically generated

Figure - Screenshot showing CD\_Inventory.py in the terminal being used to reload the inventory and exit

Text

Description automatically generated

Figure - Screenshot showing CD\_Inventory.py in the terminal being used to reload the inventory and demonstrate error handling

# Summary

In this assignment, we played around with adding our own class and building a script based on previous assignments. It’s been interesting and fun to continue to build off the same script each week and further refine it to make it a more useable product.

# Appendix

## Link to GitHub Repo

https://github.com/cmonte2uw/Assignment\_08[[3]](#footnote-3)

## Listing AddressBook.py (pasted from Notepad++)

#------------------------------------------#

# Title: Assignmen08.py

# Desc: Assignnment 08 - Working with classes

# Change Log: (Who, When, What)

# DBiesinger, 2030-Jan-01, created file

# DBiesinger, 2030-Jan-01, added pseudocode to complete assignment 08

# CMontejo, 2022-Aug-27, copied file

# CMontejo, 2022-Aug-28, created CD class, updated docstrings, added code for the rest of the sections

#------------------------------------------#

# -- DATA -- #

strFileName **=** 'cdInventory.txt'

lstOfCDObjects **=** **[]**

**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:

\_\_init\_\_ (cd\_id, cd\_title, cd\_artist): -> None

getcd\_id(): -> cd\_id

getcd\_title(): -> cd\_title

getcd\_artist(): -> cd\_artist

\_\_str\_\_(): -> None

add\_cd(lst\_Inventory, cd\_id, cd\_title, cd\_artist): -> (new CD inventory item)

"""

#--Constructor--#

**def** \_\_init\_\_**(**self**,** cd\_id**,** cd\_title**,** cd\_artist**):**

self**.**cd\_id **=** cd\_id

self**.**cd\_title**=** cd\_title

self**.**cd\_artist **=** cd\_artist

#--Properties--#

*@property*

**def** getcd\_id**(**self**):**

**return** self**.**cd\_id

*@property*

**def** getcd\_title**(**self**):**

**return** self**.**cd\_title

*@property*

**def** getcd\_artist**(**self**):**

**return** self**.**cd\_artist

#--Methods--#

**def** \_\_str\_\_**(**self**):**

**pass**

*@staticmethod*

**def** add\_cd**(**lst\_Inventory**,** cd\_id**,** cd\_title**,** cd\_artist**):**

"""Adds new CD to the table

Args:

lstOfCDObjects (list of lists): 2D table to hold CD Inventory data

strID (string): ID number for the CD

strTitle (string): Title of the CD

strArtist (string): Artist for the CD

Returns:

None.

"""

intID **=** **int(float(**cd\_id**))**

dicRow **=** **{**'ID'**:** intID**,** 'Title'**:** cd\_title**,** 'Artist'**:** cd\_artist**}**

lst\_Inventory**.**append**(**dicRow**)**

IO**.**show\_inventory**(**lst\_Inventory**)**

# -- PROCESSING -- #

**class** **FileIO:**

"""Processes data to and from file:

properties:

None.

methods:

save\_inventory(file\_name, lst\_Inventory): -> None

load\_inventory(file\_name,lst\_Inventory): -> (a list of CD objects)

"""

*@staticmethod*

**def** load\_inventory**(**file\_name**,** lst\_Inventory**):**

"""Function to manage data ingestion from file to a list of dictionaries

Reads the data from file identified by file\_name into a 2D table

(list of dicts) table one line in the file represents one dictionary row in table.

Args:

file\_name (string): name of file used to read the data from

lstOfCDObjects (list of dict): 2D data structure (list of dicts) that holds the data during runtime

Returns:

None.

"""

**try:**

lst\_Inventory**.**clear**()** # this clears existing data and allows to load data from file

objFile **=** **open(**file\_name**,** 'r'**)**

**for** line **in** objFile**:**

data **=** line**.**strip**().**split**(**','**)**

dicRow **=** **{**'ID'**:** **int(**data**[**0**]),** 'Title'**:** data**[**1**],** 'Artist'**:** data**[**2**]}**

lst\_Inventory**.**append**(**dicRow**)**

objFile**.**close**()**

**except** **FileNotFoundError:**

**print(**'Existing inventory file not found.\n'**)**

**pass**

*@staticmethod*

**def** save\_inventory**(**file\_name**,** lst\_Inventory**):**

"""Function to manage data ingestion from a list of dictionaries to a file

Reads the data from a 2D table (list of dictionaries) identified as lstTbl into a file.

Args:

file\_name (string): name of file used to read the data from

lstOfCDObjects (list of dict): 2D data structure (list of dicts) that holds the data during runtime

Returns:

None.

"""

objFile **=** **open(**strFileName**,** 'w'**)**

**for** row **in** lstOfCDObjects**:**

lstValues **=** **list(**row**.**values**())**

lstValues**[**0**]** **=** **str(**lstValues**[**0**])**

objFile**.**write**(**','**.**join**(**lstValues**)** **+** '\n'**)**

objFile**.**close**()**

**pass**

# -- PRESENTATION (Input/Output) -- #

**class** **IO:**

"""Handling Input / Output

properties:

None.

methods:

print\_menu(): -> None

menu\_choice(): -> (menu choice)

show\_inventory(lst\_Inventory): -> None

new\_cd(): -> (new CD Inventory item)

"""

*@staticmethod*

**def** print\_menu**():**

"""Displays a menu of choices to the user

Args:

None.

Returns:

None.

"""

**print(**'Menu\n\n[l] Load Inventory from File\n[a] Add CD\n[i] Display Current Inventory'**)**

**print(**'[s] Save Inventory to File\n[x] Exit\n'**)**

*@staticmethod*

**def** menu\_choice**():**

"""Gets user input for menu selection

Args:

None.

Returns:

choice (string): a lower case sting of the users input out of the choices l, a, i, s or x

"""

choice **=** ' '

**while** choice **not** **in** **[**'l'**,** 'a'**,** 'i'**,** 's'**,** 'x'**]:**

choice **=** **input(**'Which operation would you like to perform? [l, a, i, s or x]: '**).**lower**().**strip**()**

**print()** # Add extra space for layout

**return** choice

*@staticmethod*

**def** show\_inventory**(**lst\_Inventory**):**

"""Displays current inventory table

Args:

lstOfCDObjects (list of dict): 2D data structure (list of dicts) that holds the data during runtime.

Returns:

None.

"""

**print()** # Add extra space for layout

**print(**'======= The Current Inventory: ======='**)**

**print(**'ID\tCD Title (by: Artist)\n'**)**

**for** row **in** lst\_Inventory**:**

**print(**'{}\t{} (by:{})'**.format(\***row**.**values**()))**

**print(**'======================================'**)**

**print()** # Add extra space for layout

*@staticmethod*

**def** new\_cd**():**

"""Get user input for a new CD

Args:

None.

Returns:

None.

"""

**while** **True:** #Added error handling to force user to enter an integer for the ID

strID **=** **input(**'Enter ID: '**).**strip**()**

**try:**

intID **=** **(int(float(**strID**)))**

**break**

**except** **ValueError:**

**print(**'Please enter an integer\n'**)**

strTitle **=** **input(**'What is the CD\'s title? '**).**strip**()**

strArtist **=** **input(**'What is the Artist\'s name? '**).**strip**()**

CD**.**add\_cd**(**lstOfCDObjects**,** intID**,** strTitle**,** strArtist**)**

# -- INTERFACE -- #

# 1. When program starts, read in the currently saved Inventory

FileIO**.**load\_inventory**(**strFileName**,** lstOfCDObjects**)**

# 2. Start main loop

**while** **True:**

# 2.1 Display Menu to user and get choice

IO**.**print\_menu**()**

strChoice **=** IO**.**menu\_choice**()**

# 3. Process menu selection

# 3.1 Process exit first

**if** strChoice **==** 'x'**:**

**break**

# 3.2 Process to load inventory

**if** 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\n'**)**

**if** strYesNo**.**lower**()** **==** 'yes'**:**

**print(**'reloading...'**)**

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

# 3.3 Process to add a CD

**elif** strChoice **==** 'a'**:**

# 3.3.1 Ask user for new ID, CD Title and Artist and add item to the table

IO**.**new\_cd**()**

**continue** # Start loop back at top

# 3.4 Process to display current inventory

**elif** strChoice **==** 'i'**:**

IO**.**show\_inventory**(**lstOfCDObjects**)**

**continue** # Start loop back at top

# 3.5 Process to save inventory to file

**elif** strChoice **==** 's'**:**

# 3.5.1 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**()**

# 3.5.2 Process choice

**if** strYesNo **==** 'y'**:**

# 3.5.2.1 Save data

FileIO**.**save\_inventory**(**strFileName**,** lstOfCDObjects**)**

**pass**

**else:**

**input(**'The inventory was NOT saved to file. Press [ENTER] to return to the menu.'**)**

**continue** # Start loop back at top

# 3.6 Catch-all should not be possible, as user choice gets vetted in I/O, but to be safe:

**else:**

**print(**'General Error'**)**

1. Retrieved 2022-Aug-28 [↑](#footnote-ref-1)
2. Retrieved 2022-Aug-28 [↑](#footnote-ref-2)
3. Retrieved 2022-Aug-28 [↑](#footnote-ref-3)