Nary Hang

March 5, 2022

Introduction to Programming (Python)

Assignment 06

How to modify an existing CD Inventory program with Functions and Classes

# Introduction

The objective for this sixth assignment is focused on modifying an existing CD Inventory program by working on the provided TODOs on functions. The process to complete the assignment already includes a few functions and we were tasks to manipulate existing code to add the remaining functions. This knowledge document will detail the steps taken to modify the program.

# Steps Involved

After opening script in Anaconda Spyder, we were tasked with modifying an existing CD Inventory program, which asks a user to choose from a menu. The first option on the menu allows the user to load the inventory from a file. The other choices on the menu includes adding information on a CD such as ID number, CD title, and artist name as well as displaying current inventory, deleting a CD from inventory, saving the inventory to a file, and exiting.

The following steps were taken:

1. The first TODO that I tackled was moving processing code into function for the save inventory to a file selection. The def and attributes were already provided under class FileProcessor. I ended up copying and pasting existing code to the defining function, write\_file. Next, I replaced the existing code by creating a call to the new location using class, def, and attributes. The attributes included the location of the text file and the data list. See below for the following code:

**def** write\_file(file\_name, table):

objFile = open(file\_name, 'w')

**for** row **in** table:

lstValues = list(row.values())

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

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

objFile.close()

FileProcessor.write\_file(strFileName, lstTbl)

**Figure 1:**

*Results after selecting Saving Inventory to file from the menu in Spyder*

*Text

Description automatically generated*

1. The next TODO that I worked on was moving input/output code into function when user is asked to insert a new ID, CD title and artist. I defined the function with an empty parameter under class IO called user\_inputs. Next, I copied and paste existing code for user inputs. I added the three variables under data and assigned it as none. None is used to define a null value or an object. I identified variable strID as an integer. Then, I created a return statement for the values of the three variables. See below for the following code:

@staticmethod

**def** user\_inputs():

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

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

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

intID = int(strID)

**return** intID, strTitle, strArtist

Another TODO task was to move processing code into function for adding an item to the table. I defined the function called adding\_cd with table parameter under class DataProcessor. According to docstring, the parameter table represents the list of dictionaries. I copied and paste the existing code to the function. I made some edits to include IO.user inputs() for the three variables. I also, wanted the code to show the current inventory. Next, I called it to the new location. See below for the following code:

**def** adding\_cd(table):

intID, strTitle, strArtist = IO.user\_inputs()

dicRow = {'ID': intID, 'Title': strTitle, 'Artist': strArtist}

lstTbl.append(dicRow)

IO.show\_inventory(table)

DataProcessor.adding\_cd(lstTbl)

**Figure 2:**

*Results after selecting add CD from the menu in Spyder*

*Graphical user interface, text

Description automatically generated*

1. The last task was to move processing code into function when searching through the table and deleting a CD. I defined the function called deleting\_cd with table parameter under class DataProcessor. I copied and paste the existing code to the function. Since we are deleting, I had to update the code to search for the stored inventory data in table. Next, I called it to the new location. See below for the following code.

**def** deleting\_cd(table):

intRowNr = -1

blnCDRemoved = **False**

**for** row **in** lstTbl:

intRowNr += 1

**if** row['ID'] == intIDDel:

**del** table[intRowNr]

blnCDRemoved = **True**

**break**

**if** blnCDRemoved:

print('The CD was removed')

**else**:

print('Could not find this CD!')

IO.show\_inventory(table)

DataProcessor.deleting\_cd(lstTbl)

**Figure 3:**

*Results after selecting delete CD from Inventory to file from the menu in Spyder*

*Graphical user interface, text

Description automatically generated*

1. After successfully running the program in Terminal, see below for screenshots:

**Figure 4:**

*Results after running python script in Terminal*

Text

Description automatically generated

**Figure 4.1:**

*Results after selecting Option a in Terminal*

Text

Description automatically generated

**Figure 4.2:**

*Results after selecting Option s in Terminal*

Text

Description automatically generated

**Figure 4.3:**

*Results after selecting Option i in Terminal*

Text

Description automatically generated

**Figure 4.4:**

*Results after selecting Option l in Terminal*

Text

Description automatically generated

**Figure 4.5:**

*Results after selecting Option d in Terminal*

*Text

Description automatically generated*

1. After completing this assignment, the files are posted in GitHub. Here is the link to the repository:

<https://github.com/naryhang/Assignment_06>

# Summary

I had a couple setbacks in this assignment. First, I was overwhelmed by the abundent of codes from Assignment06\_Starter.py. After going through the recording on Friday class, it helped me organize my thoughts on how to approach the assignment and Lambda jokes made me laugh. Another issue is when I was getting an error, “int() argument must be a string, a bytes-like object or a number, not ‘list” while working on I/O code for the user inputs. After staring at the code for 4+ hours, I realized I needed to create a string under processing def adding\_cd.

In completing this assignment, I gained experience in modifying existing code to create process and input/output functions. Also, I learned on how to resolve errors.

# Appendix