PADMAVATHY BHAMIDIPATI

FD IT FDN 110 B Au 22: Foundations Of Programming: Python

11/20/2022

Assignment 06

Functions and Classes

What we learnt in Module 6

In this module, we learned more about working with ***functions*** and started on ***Classes***. We used *Spyder* as our IDE and continued using *GitHub* as our code and other documents' repository.

A brief of what we learnt in this module and applied in Assignment 6 are as follows:

- **Functions:** are a grouping of statements that can be called by a programmer using a unique name (in Python). In Python, a function must be defined ahead of it being called and typically, upon execution, returns results and continues to the calling point.
- **Parameters:** Functions give us the option of passing parameters that allow values to be passed in for processing. We can set default parameter values too.
- Importance of **naming convention.**
- Using variables as arguments; working with arguments; Positional vs Named Arguments
- **Return** values in *Functions*; can return *None*! In Python, the return of multiple values is further helped by the implicit packing and unpacking of tuples.
- We learnt the use of the **None** keyword.
- An important point learnt was the Value Type behavior vs Reference Type behavior in the context of data types passed to a Function.
- **Variable Scope:** Local vs Global; this was very useful, especially when coding for Assignment 06.
- **Shadowing Global variable** and why we must avoid it!
- **Function Document Headers or Doc Strings:** we learnt the Google standard.
- **Classes:** are a way of grouping functions, variables and constants; we learnt how to both efficiently and effectively organize our code using *Classes*.

Assignment 06

For assignment 6, we were given a Starter Python code that contained an implementation of Assignment 05. Several tasks or TO DOs were marked for us to address to completion in this code. We were also tasked to organize and rearrange code as we saw fit. So, this assignment saw us being able to implement whatever we learnt under Improving Scripts in the previous learning module (05). For example, the Separation of Concerns.

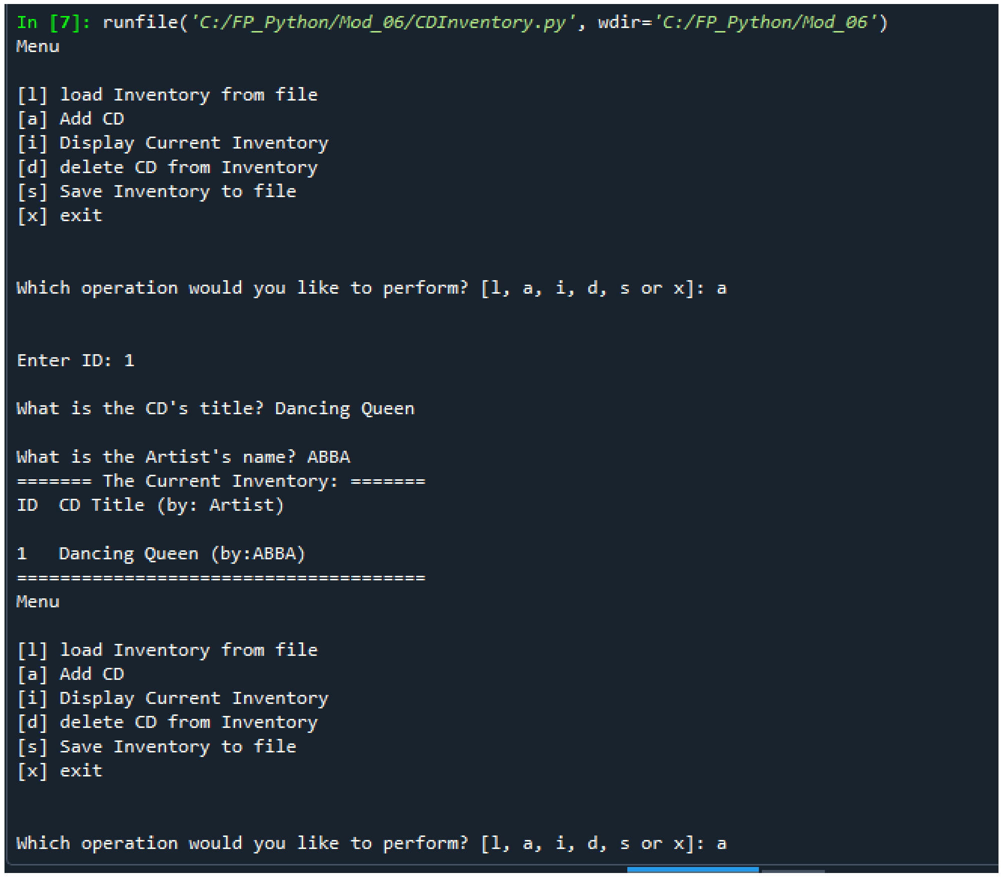
I applied the various learning modules of this section of the course to write and successfully execute a Python script that hones the CD Inventory program to offer a comprehensive menu that a user can choose from: to add, delete, save, display, load or exit. Snips of code execution are listed under the **Code Execution** section below – in both **Spyder** and on the **Terminal**. The code is detailed in the **Appendix** section.

Steps taken to address Assignment 06 –

(1) Executed the starter script that was given as part of the assignment to play-around and understand what was happening.
(2) Started editing from the main body of the script:
  - tested loading for leading / trailing white space > treats it as equivalent to 'no', so added strip() to line#127
  - tested random string instead of 'yes' to see that it follows the 'no' fork of the script
  - upon the first execution of file > chose Load option > empty list displayed, as expected
  - line #173: actually, any key will return the control back to the main menu
(3) Separation of Concerns: read through the code and made an outline of it, before separating and / or re-writing the code to fall under logical code block for easier reading and comprehension
  - added a method to clear in-memory list and called it as the first line of code execution / main body
(4) Test cases: the snips of code execution correspond to the remaining testing done to ensure that the code is executing as expected

Code Execution in Spyder

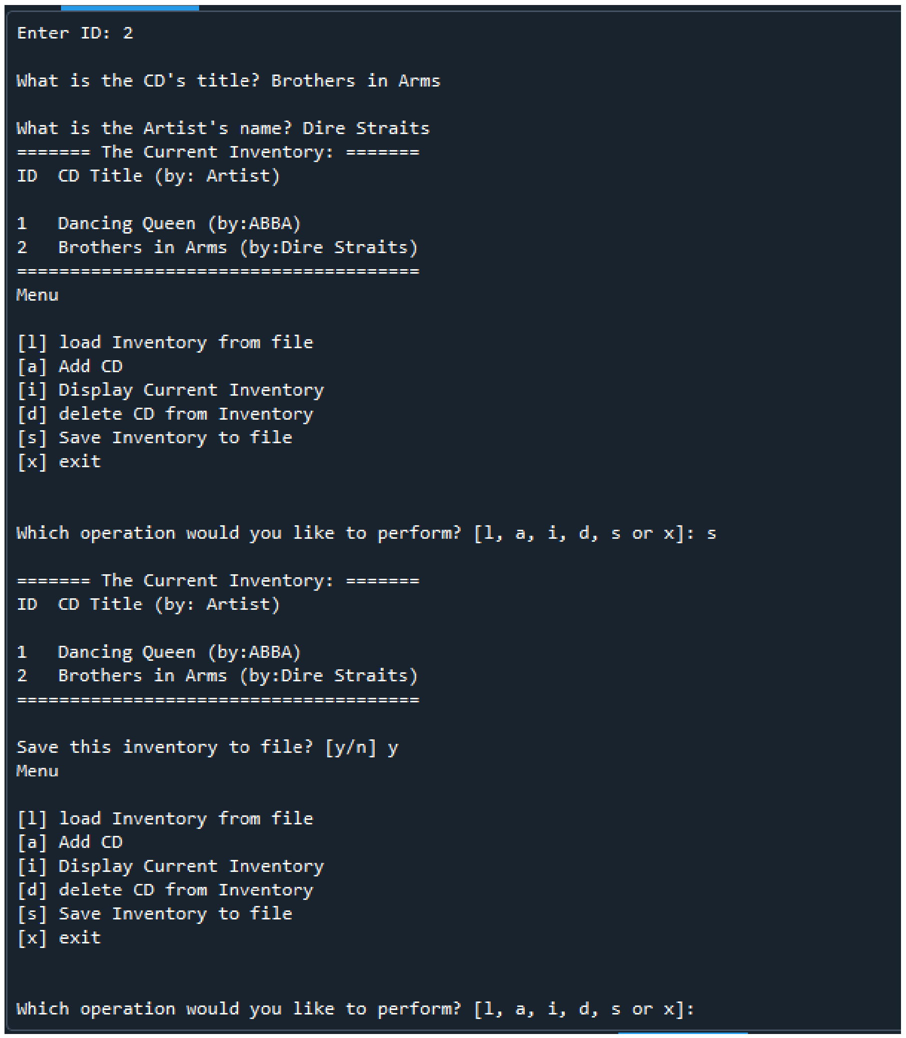
1. Adding entries to the CD Inventory



```
In [7]: runfile('C:/FP_Python/Mod_06/CDInventory.py', wdir='C:/FP_Python/Mod_06')
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]: a


Enter ID: 1

What is the CD's title? Dancing Queen

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

1   Dancing Queen (by:ABBA)
======================================
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]: a
```

Figure 1: Adding an entry to the CD Inventory

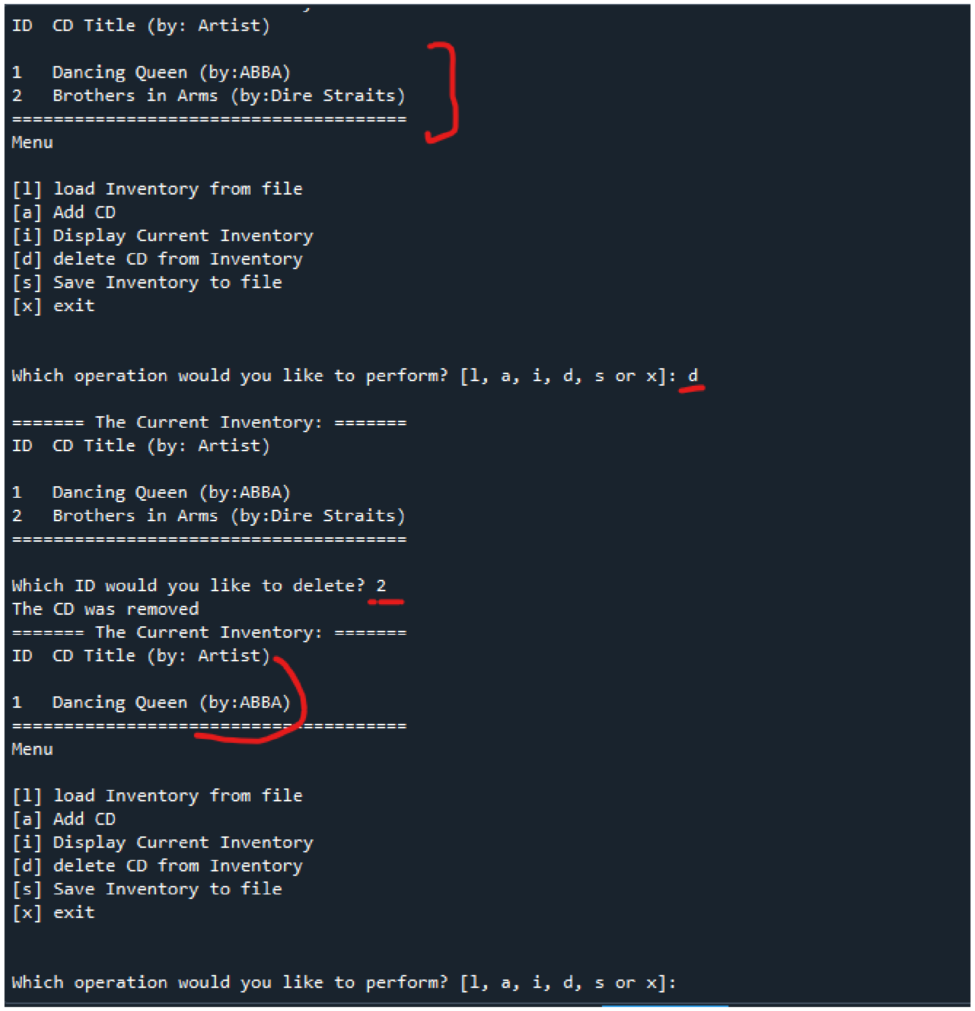
2. Saving entries to file



```
Enter ID: 2

What is the CD's title? Brothers in Arms

What is the Artist's name? Dire Straits
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Dancing Queen (by:ABBA)
2   Brothers in Arms (by:Dire Straits)
======================================
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]: s

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

1   Dancing Queen (by:ABBA)
2   Brothers in Arms (by:Dire Straits)
======================================

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

Figure 2: Saving entries to file

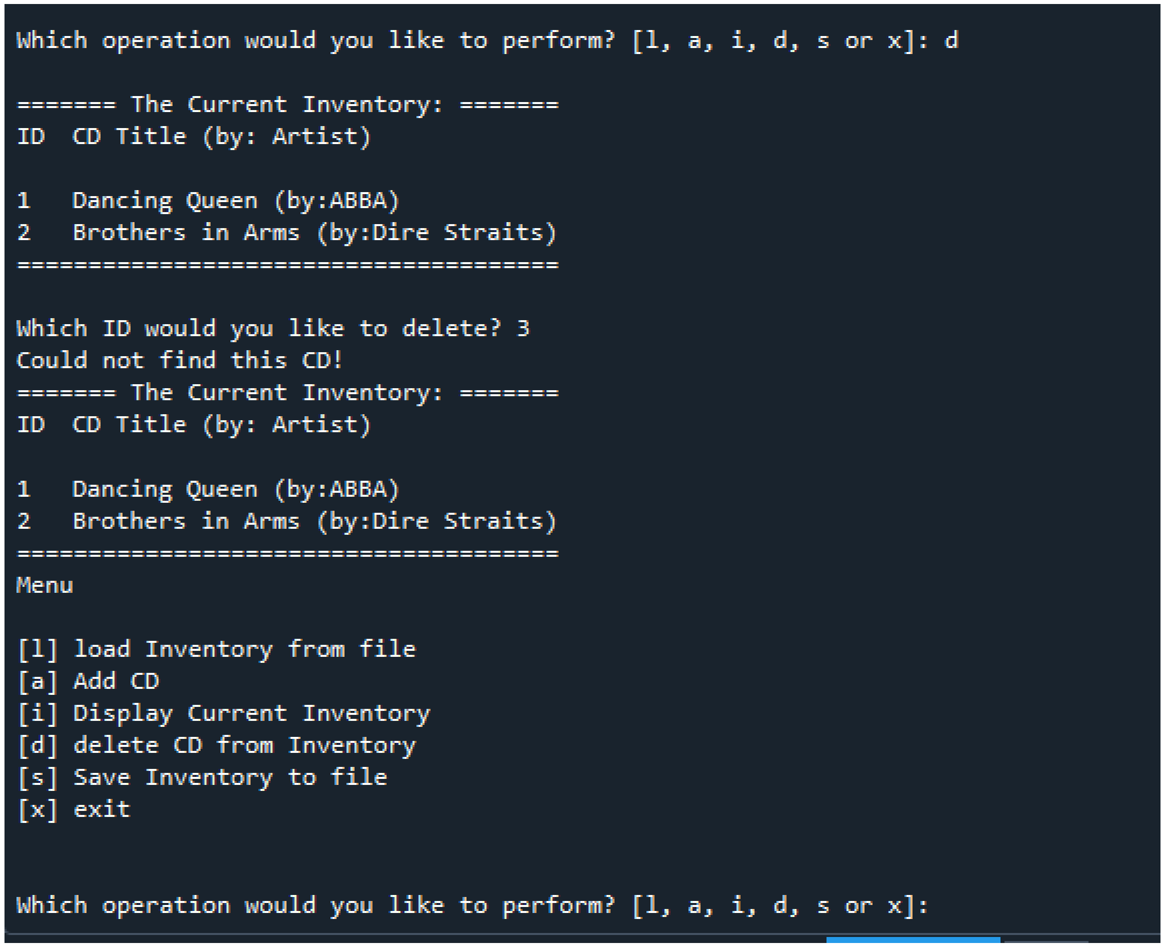
3. Deleting an entry in the list and displaying the edited inventory



```
ID  CD Title (by: Artist)

1   Dancing Queen (by:ABBA)
2   Brothers in Arms (by:Dire Straits)
======================================
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]: d

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

1   Dancing Queen (by:ABBA)
2   Brothers in Arms (by:Dire Straits)
======================================

Which ID would you like to delete? 2
The CD was removed
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Dancing Queen (by:ABBA)
======================================
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]:
```

Figure 3: Deleting from the list based on CD's ID

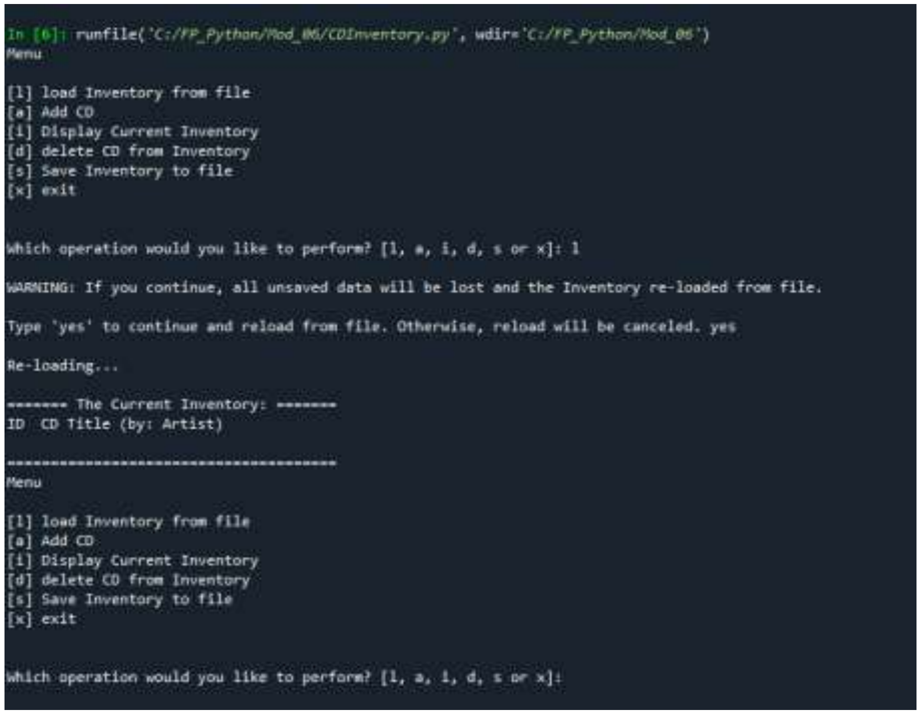
4. Attempting to delete an entry that does not exist



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

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

1   Dancing Queen (by:ABBA)
2   Brothers in Arms (by:Dire Straits)
======================================

Which ID would you like to delete? 3
Could not find this CD!
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Dancing Queen (by:ABBA)
2   Brothers in Arms (by:Dire Straits)
======================================
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]:
```

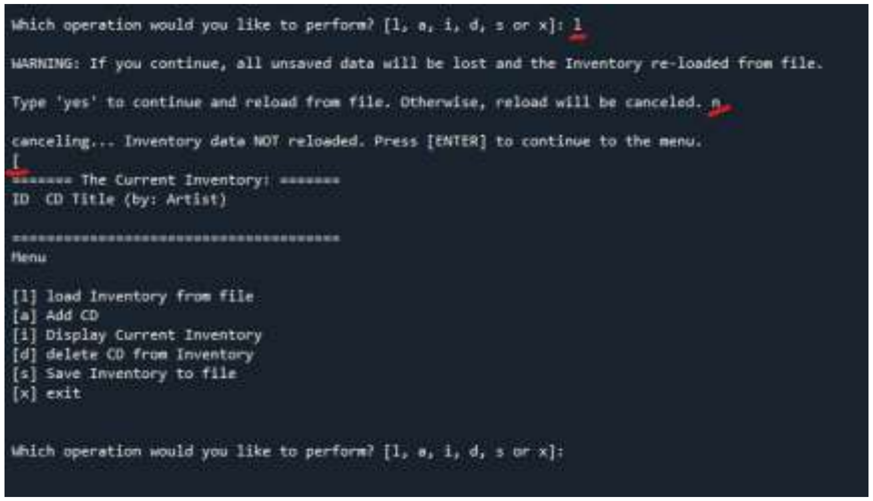
Figure 4: Deleting a non-existent entry based on ID

5. Loading data



```
In [6]: runfile('C:/FP_Python/Mod_06/CDInventory.py', wdir='C:/FP_Python/Mod_06')
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

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

Type 'yes' to continue and reload from file. Otherwise, reload will be canceled. yes

Re-loading...

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

======================================
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]:
```

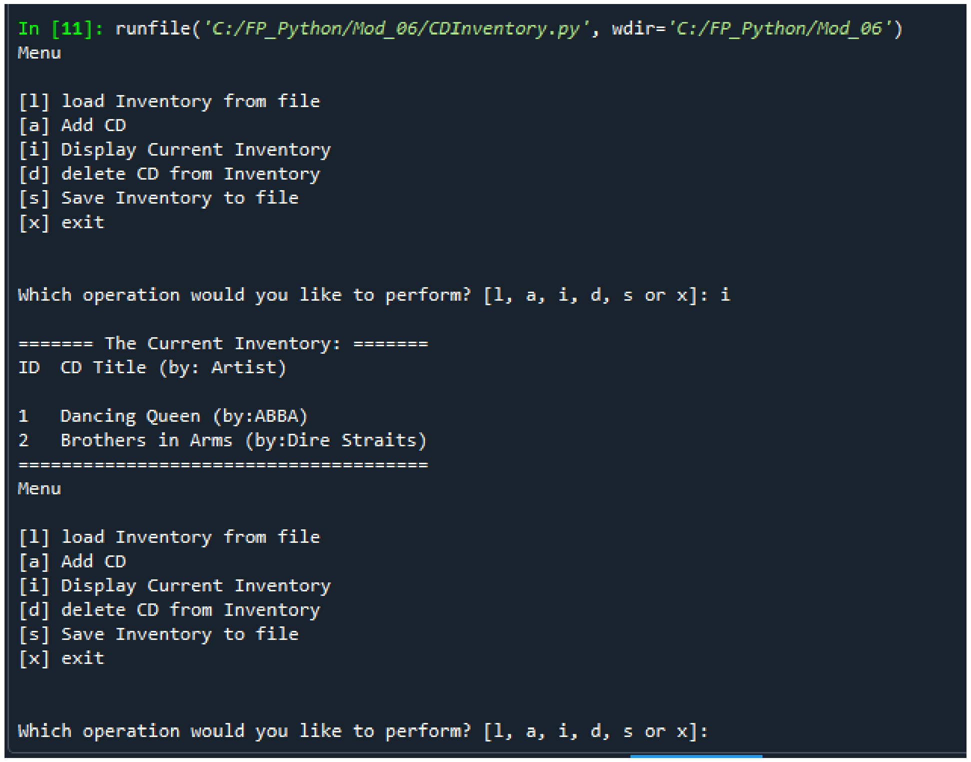
Figure 5: Loading data, after confirming the action as 'yes'



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

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

Type 'yes' to continue and reload from file. Otherwise, reload will be canceled. n

canceling... Inventory data NOT reloaded. Press [ENTER] to continue to the menu.

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

======================================
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]:
```

Figure 6: Opting out of loading the inventory

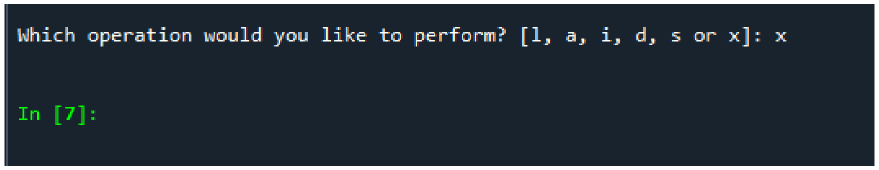
6. Displaying the inventory



```
In [11]: runfile('C:/FP_Python/Mod_06/CDInventory.py', wdir='C:/FP_Python/Mod_06')
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]: i

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

1   Dancing Queen (by:ABBA)
2   Brothers in Arms (by:Dire Straits)
======================================
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]:
```

Figure 7: Display inventory

7. Exiting the program

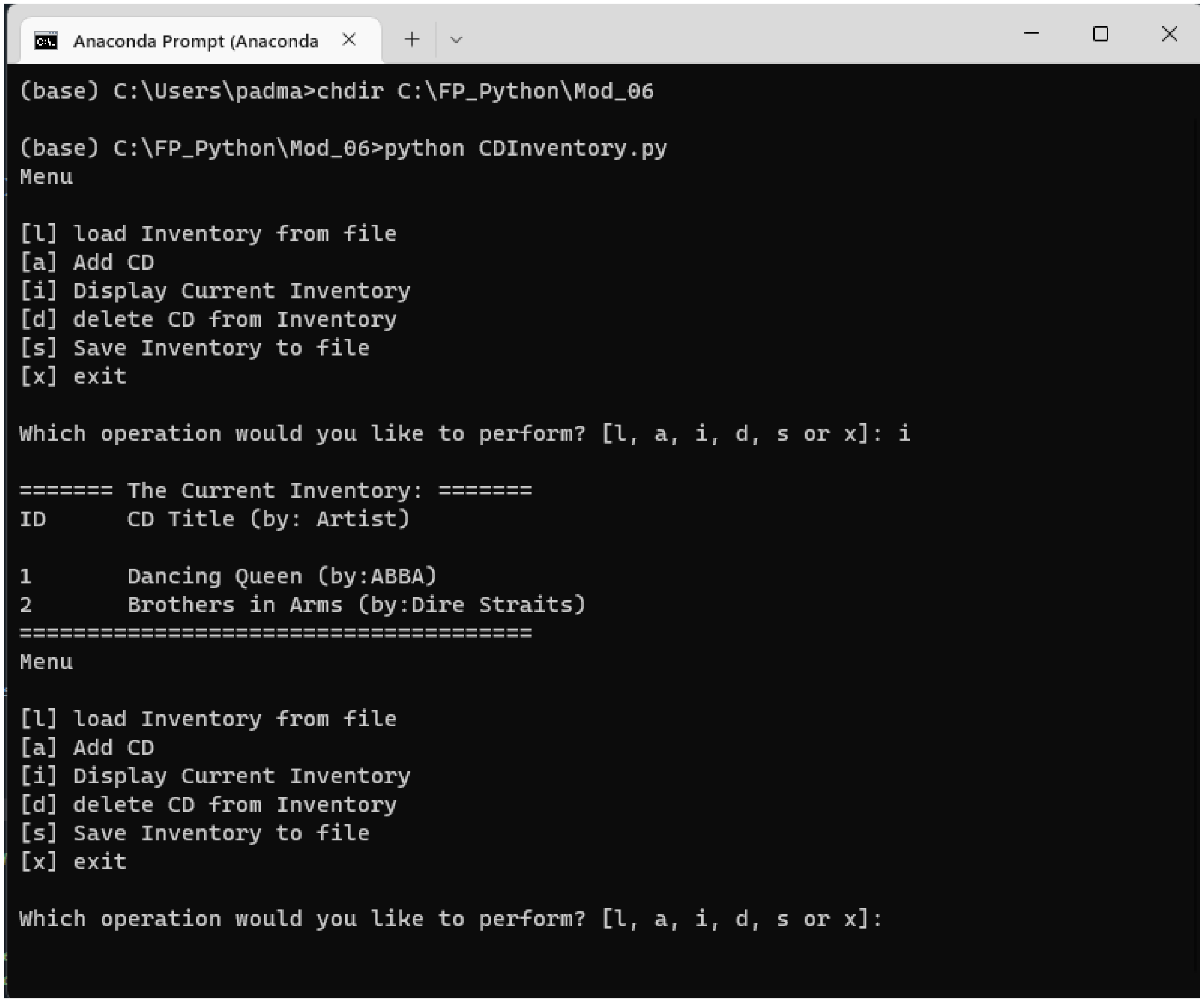


```
Which operation would you like to perform? [l, a, i, d, s or x]: x


In [7]:
```

Figure 8: Exit program

Execution on Terminal

The following image captures a successful execution of Assignment 06’s script on the Terminal.



```
(base) C:\Users\padma>chdir C:\FP_Python\Mod_06

(base) C:\FP_Python\Mod_06>python CDInventory.py
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]: i

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

1       Dancing Queen (by:ABBA)
2       Brothers in Arms (by:Dire Straits)
======================================
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]:
```

Figure 9: Execution of code on Terminal

GitHub

The following is the link to my GitHub repository for this assignment:

https://github.com/PadmaBham/Assignment_06

Appendix

1. My **Python script** for the assignment is as follows:

```
#------------------------------------------#
# Title: Assignment06_Starter.py
# Desc: Working with classes and functions.
# Change Log: (Who, When, What)
# DBiesinger, 2030-Jan-01, Created File
# PBhamidipati, 2022-Nov-19, Edited file to address TODOs as part of Assignment 06
# PBhamidipati, 2022-Nov-20, Edited file to add DocStrings, corrected code after testing it and debugging
#------------------------------------------#

# -- DATA -- #
strChoice = '' # User input
lstTbl = []  # list of lists to hold data
dicRow = {}  # list of data row
strFileName = 'CDInventory.txt'  # data storage file
objFile = None  # file object


# -- PROCESSING -- #
class DataProcessor:
    """ Manipulating data; contains functions that are typically called from IO functions"""
    # TODONE add functions for processing here

    #   METHOD TO CLEAR IN-MEMORY LIST / VARIABLE
    def list_clear():
        """Clears the in-memory list of lists ahead of new data load and is called only as required

        Args:
```

```
        None.

    Returns:
        None.
    """
    lstTbl.clear() # this clears existing data and allows to load data from file

# METHOD FOR ADD CD
@staticmethod
def data_add(getID, getTitle, getArtist):
    """Adds new entry at the bottom of the CD inventory

    Args:
        getID (string): the unique ID of a CD entry that's received as a string
            but converted to integer within the method
        getTitle (string): the title of the CD being added
        getArtist (string): the name of the artist of the CD

    Returns:
        None.
    """
    intID = int(getID)
    dicRow = {'ID': intID, 'Title': getTitle, 'Artist': getArtist}
    lstTbl.append(dicRow)

# METHOD FOR DELETE
@staticmethod
def delete_row(intIDDel):
    """Deletes an entry from the inventory if the corresponding ID is matched with the specified ID;
    otherwise, due info given to user

    Args:
        intIDDel (integer): this is the CD's ID that the user specifies for the entry's deletion from inventory

    Returns:
        None.
    """
    intRowNr = -1
    blnCDRemoved = False
    for row in lstTbl:
            intRowNr += 1
            if row['ID'] == intIDDel:
                del lstTbl[intRowNr]
                blnCDRemoved = True
                break
        if blnCDRemoved:
            print('The CD was removed')
        else:
            print('Could not find this CD!')


class FileProcessor:
    """Processing the data to and from text file"""

    @staticmethod
    def read_file(file_name, table):
        """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
            table (list of dict): 2D data structure (list of dicts) that holds the data during runtime

        Returns:
            None.
        """
        objFile = open(file_name, 'r')
        for line in objFile:
            data = line.strip().split(',')
            dicRow = {'ID': int(data[0]), 'Title': data[1], 'Artist': data[2]}
            table.append(dicRow)
        objFile.close()

    @staticmethod
    def write_file(file_name, table):
        """Function that writes data from a list of dictionaries to a file

        Reads the data from a 2D table and writes into a file identified by file_name
        (list of dicts) table one line in the file represents one dictionary row in table.

        Args:
            file_name (string): name of file used to write the data to
            table (list of dict): 2D data structure (list of dicts) that holds the data during runtime

        Returns:
            None.
        """
        # TODONE Add code here
        # METHOD TO SAVE
        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()


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

class IO:
    """Handling Input / Output"""

    @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('[d] delete CD from Inventory\n[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, d, s or x

        """
        choice = ' '
        while choice not in ['l', 'a', 'i', 'd', 's', 'x']:
            choice = input('Which operation would you like to perform? [l, a, i, d, s or x]: ').lower().strip()
        print()  # Add extra space for layout
        return choice

    @staticmethod
    def show_inventory(table):
        """Displays current inventory table


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

        Returns:
            None.

        """
        print('======= The Current Inventory: =======')
        print('ID\tCD Title (by: Artist)\n')
        for row in table:
            print('{}\t{} (by:{})'.format(*row.values()))
        print('======================================')

    # TODONE add I/O functions as needed
    # ASK USER FOR INPUT; call DataProcessor.
    @staticmethod
    def ask_input_addCD():
        """Gets user input for the CD's details that are to be added to the inventory

        Args:
            None.

        Returns:
            None.

        """
        strID = input('Enter ID: ').strip()
        strTitle = input('What is the CD\'s title? ').strip()
        strArtist = input('What is the Artist\'s name? ').strip()

        DataProcessor.data_add(strID, strTitle, strArtist)
        IO.show_inventory(lstTbl)

    @staticmethod
    def delete_data():
        """Gets CD's ID from user for deletion of the entry from the inventory;
        then calls delete func. from Data Processing section;
        displays the latest inventory after deletion

        Args:
            None.

        Returns:
            None.

        """
        # 3.5.1 get Userinput for which CD to delete
        # 3.5.1.1 display Inventory to user
        IO.show_inventory(lstTbl)
        # 3.5.1.2 ask user which ID to remove
        intIDDel = int(input('Which ID would you like to delete? ').strip())

        # 3.5.2 search thru table and delete CD
        # TODO move processing code into function
        DataProcessor.delete_row(intIDDel)
        IO.show_inventory(lstTbl)

    @staticmethod
    def save_data_list():
        """Displays the latest inventory from the in-memory list and confirms that the list is to be saved
```

```
            to the file. If confirmed, the list is written to the file, otherwuse it informs that the list
            was NOT saved and asks the user to go back to the menu list

        Args:
            lstTbl (list): is the in-memory list variable that is used to store the list of CDs and their details

        Returns:
            None.

        """
        IO.show_inventory(lstTbl)
        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
            # TODO move processing code into function
            FileProcessor.write_file(strFileName, lstTbl)
        else:
            input('The inventory was NOT saved to file. Press [ENTER] to return to the menu.')

    @staticmethod
    def load_from_file():
        """Ascertains from the user if the CD inventory is to be loaded from the file or the in-memory list variable,
            displaying suitable warning messages as to what would happen for each of their choices, and proceeds to
            load or cancel loading as may be the user's choice

        Args:
            None.

        Returns:
            None.

        """
        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('\nRe-loading...\n')
            DataProcessor.list_clear()
            FileProcessor.read_file(strFileName, lstTbl)
            IO.show_inventory(lstTbl)
        else:
            input('canceling... Inventory data NOT reloaded. Press [ENTER] to continue to the menu.\n')
            IO.show_inventory(lstTbl)


# 1. When program starts, read in the currently saved Inventory
DataProcessor.list_clear() # this clears existing data and allows to load data from file
FileProcessor.read_file(strFileName, lstTbl)

# 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 load inventory
    if strChoice == 'l':
        IO.load_from_file() # call method to load data from file to in-memory list
        continue  # start loop back at top.


    # 3.3 process add a CD
    elif strChoice == 'a':
        # 3.3.1 Ask user for new ID, CD Title and Artist
        # TODO move IO code into function
        IO.ask_input_addCD()
        continue  # start loop back at top.


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

```
# 3.5 process delete a CD
elif strChoice == 'd':
    IO.delete_data()
    continue  # start loop back at top.


# 3.6 process save inventory to file
elif strChoice == 's':
    # 3.6.1 Display current inventory and ask user for confirmation to save
    IO.save_data_list()
    continue  # start loop back at top.


# 3.7 catch-all should not be possible, as user choice gets vetted in IO, but to be save:
else:
    print('General Error')
```

2. Reference material to learning this module has been all the reading documentation and videos demonstrations that were provided in this course.

3. Search strings for this week and URLs referenced:
   - **Abs()** - to use to find the difference of two numbers for Lab 06-A