Ljubica Trpcheva Smileski

November 20, 2022

Foundations of Programming, Python

Assignment 06

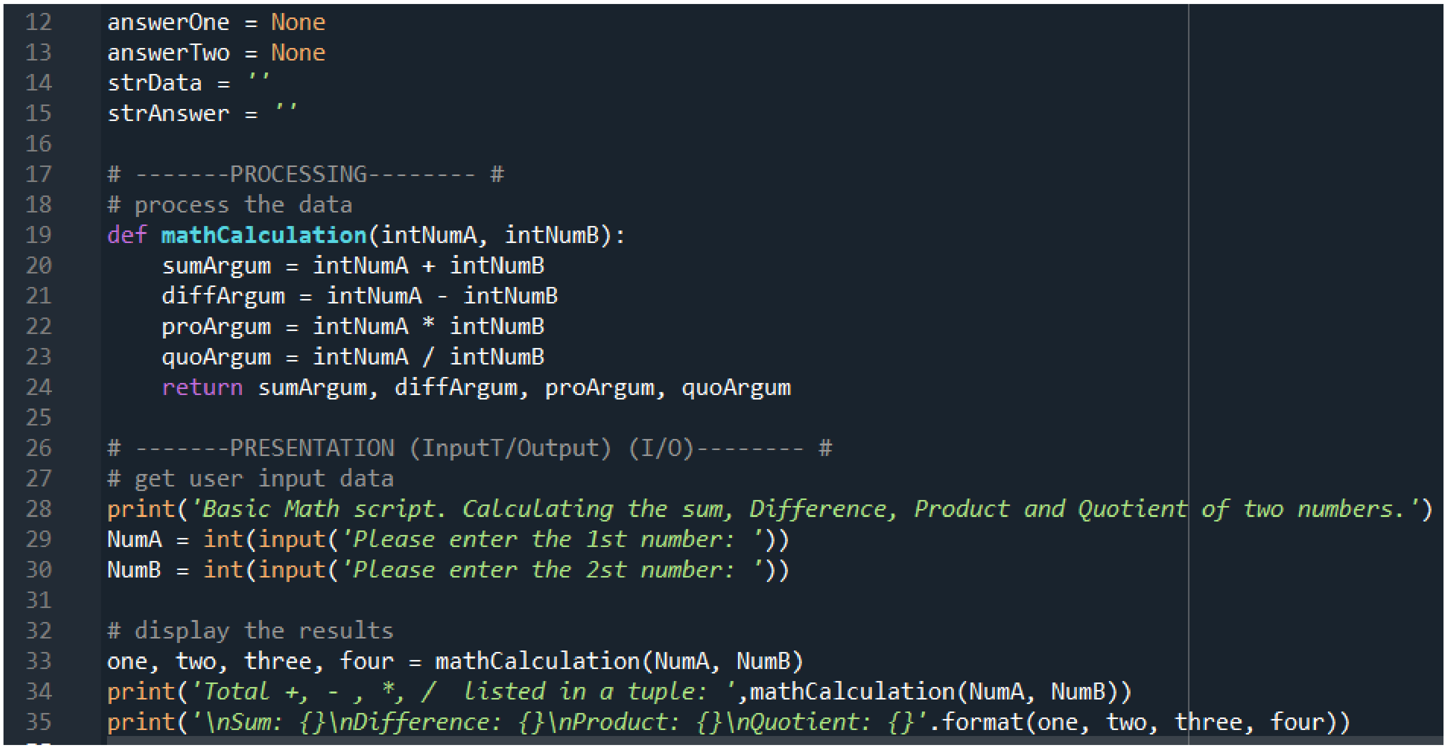
Working with functions on CD Inventory program

Introduction

In module 06 we got an introduction to functions in Python. The difference between parameter and arguments. Adding docstring to each function, so programmers can understand what it does without having to read the details of the implementation. So, at the end to get familiar with classes, which are code templates for creating objects. An object is a class that allows us to use variables and methods from inside the class.

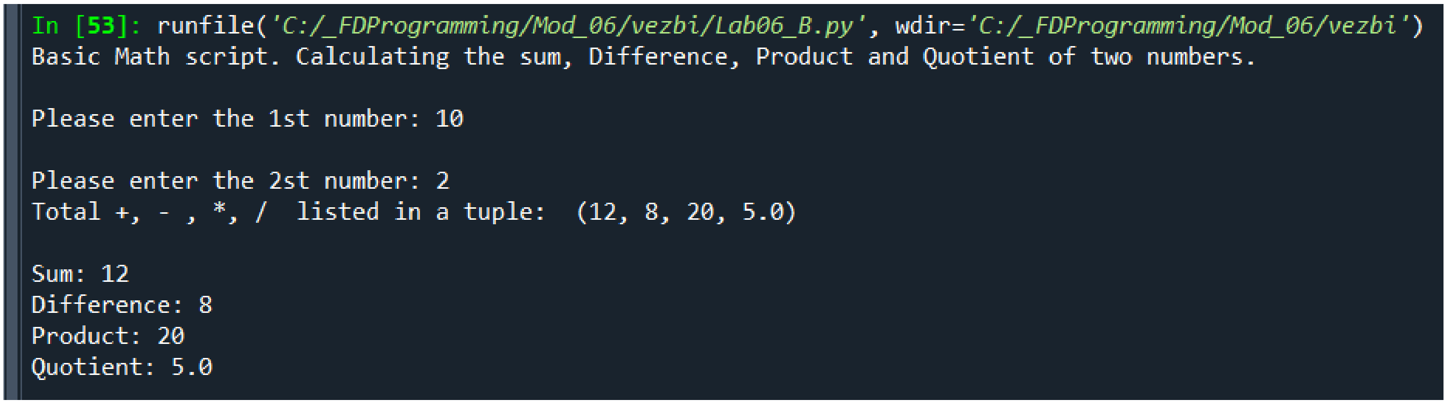
Working with functions

The function mathCalculation(intNumA, intNumB) that I create in Lab6_B returns multiple values. This function is calculating sum, difference, products and quotient of two numbers, and return those results in tuple (line 24).



```
12  answerOne = None
13  answerTwo = None
14  strData = ''
15  strAnswer = ''
16
17  # -------PROCESSING-------- #
18  # process the data
19  def mathCalculation(intNumA, intNumB):
20      sumArgum = intNumA + intNumB
21      diffArgum = intNumA - intNumB
22      proArgum = intNumA * intNumB
23      quoArgum = intNumA / intNumB
24      return sumArgum, diffArgum, proArgum, quoArgum
25
26  # -------PRESENTATION (InputT/Output) (I/O)-------- #
27  # get user input data
28  print('Basic Math script. Calculating the sum, Difference, Product and Quotient of two numbers.')
29  NumA = int(input('Please enter the 1st number: '))
30  NumB = int(input('Please enter the 2st number: '))
31
32  # display the results
33  one, two, three, four = mathCalculation(NumA, NumB)
34  print('Total +, - , *, /  listed in a tuple: ',mathCalculation(NumA, NumB))
35  print('\nSum: {}\nDifference: {}\nProduct: {}\nQuotient: {}'.format(one, two, three, four))
```

First, we ask the user to input data, NumA to enter the first number, and NumB to enter the second number. Then we call the function where we receive the result into tuple and unpack the tuple (line 33). So, at the end to be able to print the result one (Sum:), two (Difference:), three(Product:) and four (Quotient:).

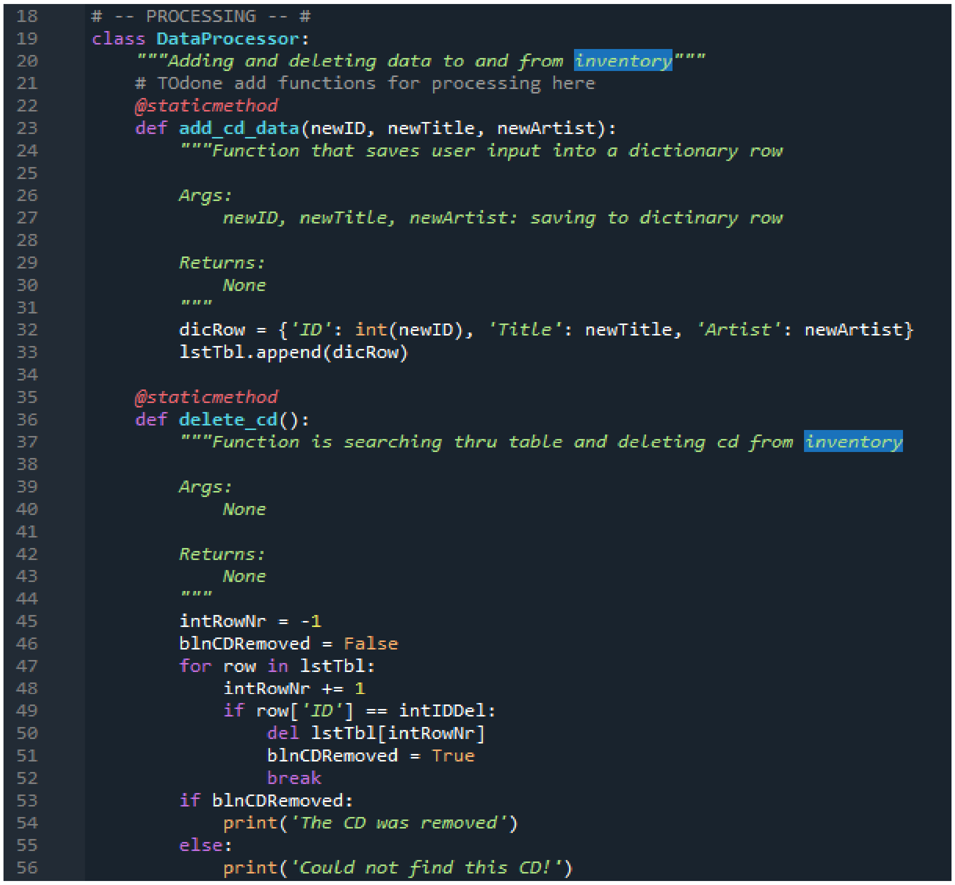


```
In [53]: runfile('C:/_FDProgramming/Mod_06/vezbi/Lab06_B.py', wdir='C:/_FDProgramming/Mod_06/vezbi')
Basic Math script. Calculating the sum, Difference, Product and Quotient of two numbers.

Please enter the 1st number: 10

Please enter the 2st number: 2
Total +, - , *, /  listed in a tuple:  (12, 8, 20, 5.0)

Sum: 12
Difference: 8
Product: 20
Quotient: 5.0
```

Working with functions and running the CD Inventory program in Spyder IDE

This week's Assignment6 was modifying the given data structure while using functions. First, I spent some time reviewing the code, to make sure I understood what the code was doing, then to be able to modify it. The given starter code was a CD inventory program that contains a menu with these options: adding cd, displaying current inventory, deleting cd from inventory, saving inventory to a .txt file, loading/reading inventory from a .txt file and exit. I run the given code in Spyder to make sure the menu is working properly and there are no errors. I was very careful not to delete any of the previous programmer's notes. There were given TODOs code that I started working on.

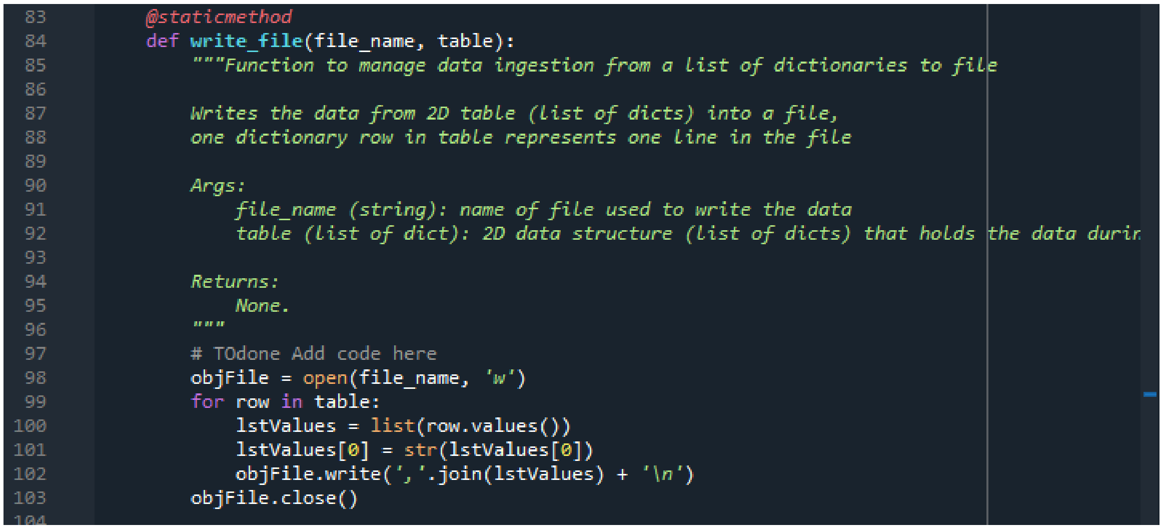
I created a couple functions add_cd_data() and delete_cd() into the class DataProcessor. Those functions are adding and deleting data to and from the inventory. Figure 1



```
# -- PROCESSING -- #
class DataProcessor:
    """Adding and deleting data to and from inventory"""
    # TOdone add functions for processing here
    @staticmethod
    def add_cd_data(newID, newTitle, newArtist):
        """Function that saves user input into a dictionary row

        Args:
            newID, newTitle, newArtist: saving to dictionary row

        Returns:
            None
        """
        dicRow = {'ID': int(newID), 'Title': newTitle, 'Artist': newArtist}
        lstTbl.append(dicRow)

    @staticmethod
    def delete_cd():
        """Function is searching thru table and deleting cd from inventory

        Args:
            None

        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!')
```

Figure 1 – Function that Add and delete data to and from inventory

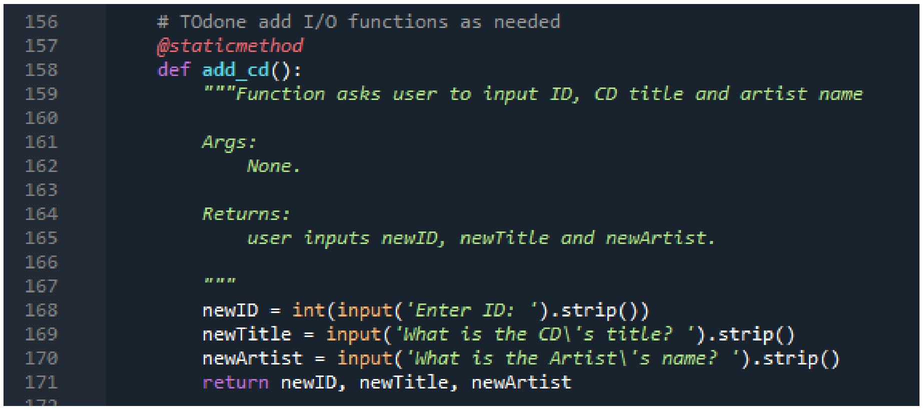
Next I created write_file() function, where it saves the data into the text file. The function is in class FileProcessor: where we use this class for processing the data to and from text file. Figure 2



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

        Writes the data from 2D table (list of dicts) into a file,
        one dictionary row in table represents one line in the file

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

        Returns:
            None.
        """
        # TOdone Add code here
        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()
```

Figure 2 – Function that writes data to a text file

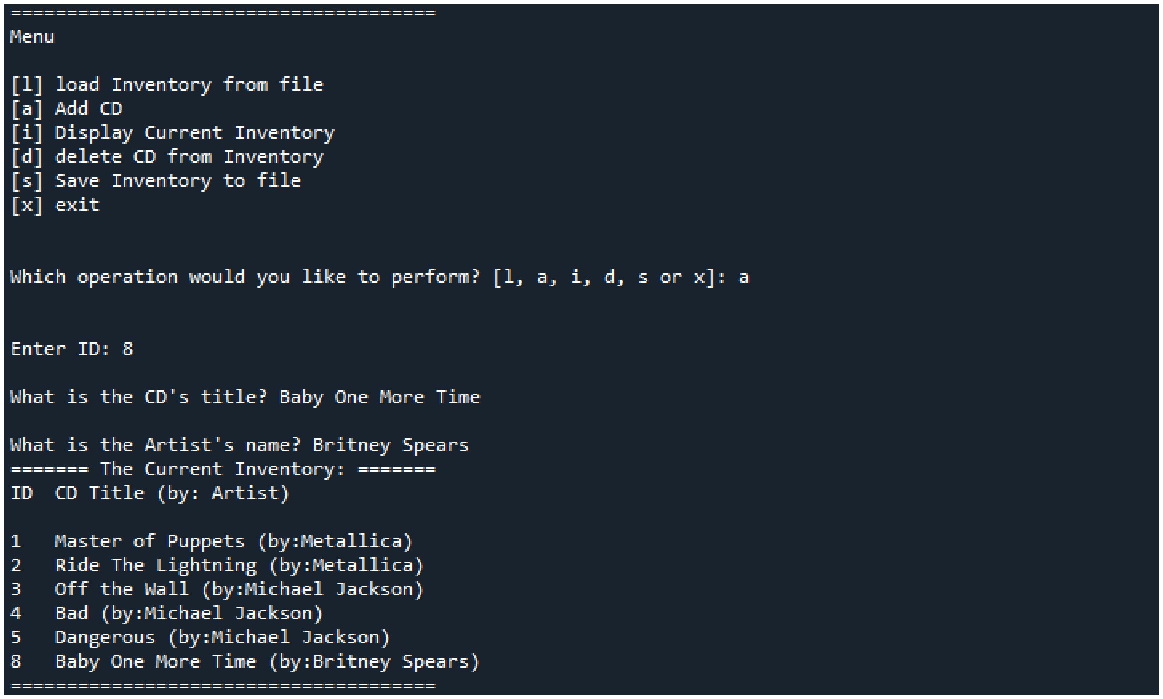
Next function add_cd() asks the user to input data, and return the values. It is located in class IO: that handles input and output data.



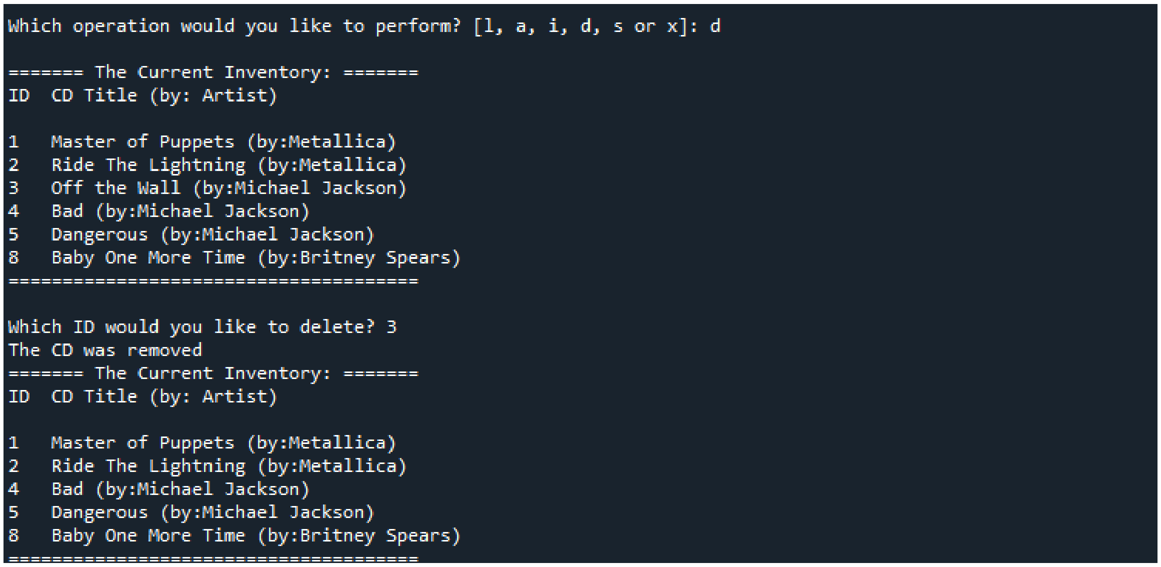
```
    # TOdone add I/O functions as needed
    @staticmethod
    def add_cd():
        """Function asks user to input ID, CD title and artist name

        Args:
            None.

        Returns:
            user inputs newID, newTitle and newArtist.

        """
        newID = int(input('Enter ID: ').strip())
        newTitle = input('What is the CD\'s title? ').strip()
        newArtist = input('What is the Artist\'s name? ').strip()
        return newID, newTitle, newArtist
```

Figure 3 – Function that asks user to input data

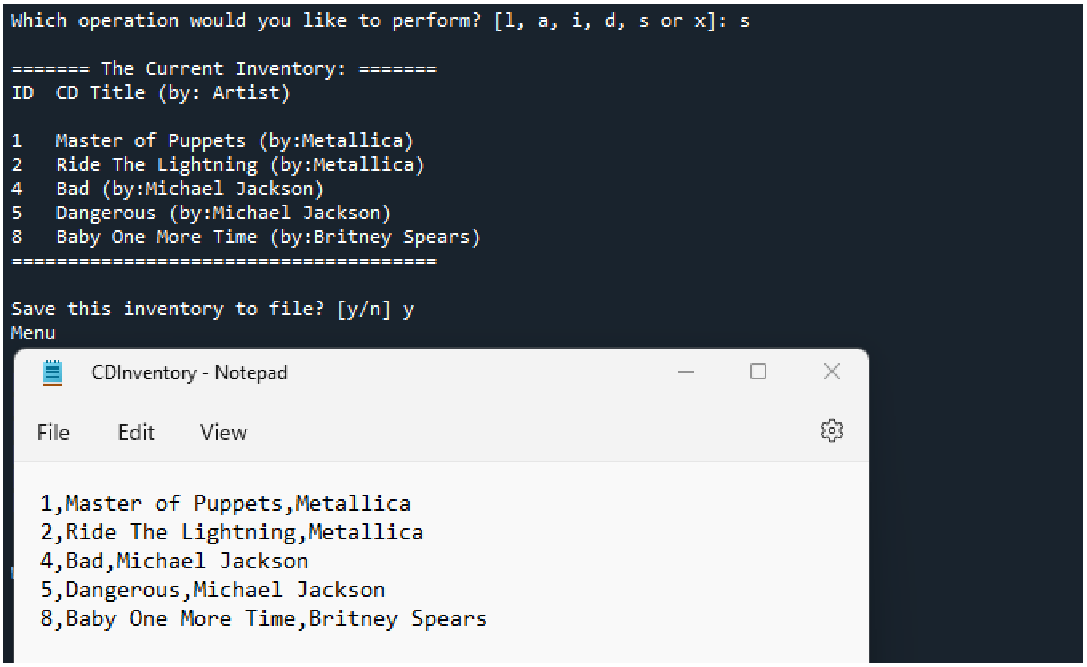
I call each function in the main menu, so this is the result after executing the program in Spyder:



```
======================================
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: 8

What is the CD's title? Baby One More Time

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

1   Master of Puppets (by:Metallica)
2   Ride The Lightning (by:Metallica)
3   Off the Wall (by:Michael Jackson)
4   Bad (by:Michael Jackson)
5   Dangerous (by:Michael Jackson)
8   Baby One More Time (by:Britney Spears)
======================================
```

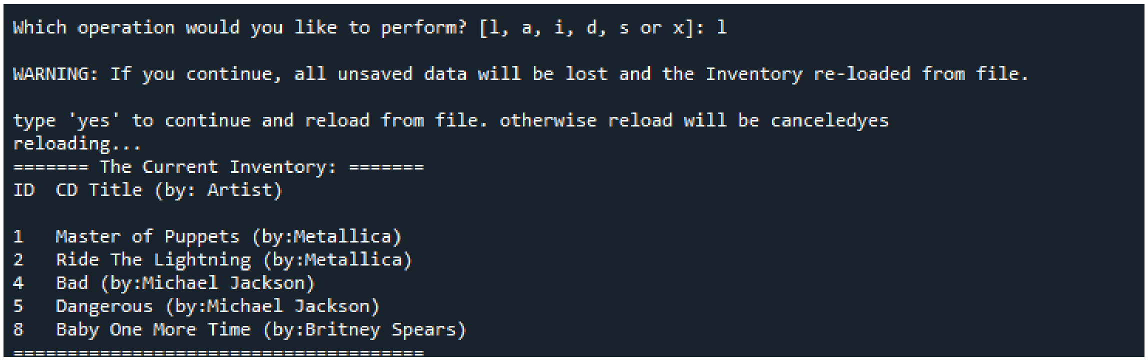


```
Which operation would you like to perform? [l, a, i, d, s or x]: d

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

1   Master of Puppets (by:Metallica)
2   Ride The Lightning (by:Metallica)
3   Off the Wall (by:Michael Jackson)
4   Bad (by:Michael Jackson)
5   Dangerous (by:Michael Jackson)
8   Baby One More Time (by:Britney Spears)
======================================

Which ID would you like to delete? 3
The CD was removed
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Master of Puppets (by:Metallica)
2   Ride The Lightning (by:Metallica)
4   Bad (by:Michael Jackson)
5   Dangerous (by:Michael Jackson)
8   Baby One More Time (by:Britney Spears)
======================================
```

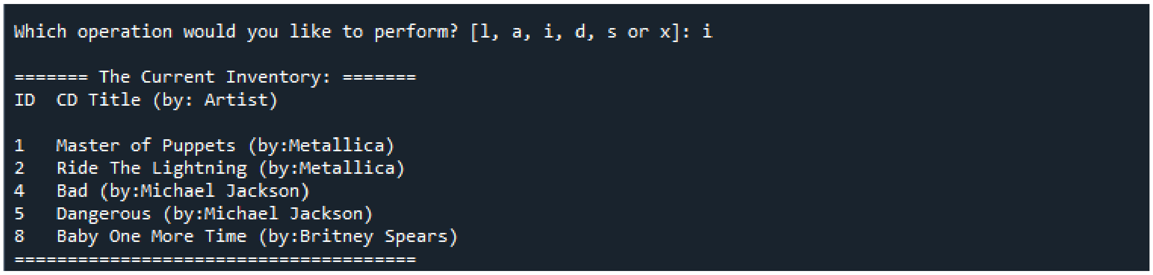


```
Which operation would you like to perform? [l, a, i, d, s or x]: s

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

1   Master of Puppets (by:Metallica)
2   Ride The Lightning (by:Metallica)
4   Bad (by:Michael Jackson)
5   Dangerous (by:Michael Jackson)
8   Baby One More Time (by:Britney Spears)
======================================

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

```
1,Master of Puppets,Metallica
2,Ride The Lightning,Metallica
4,Bad,Michael Jackson
5,Dangerous,Michael Jackson
8,Baby One More Time,Britney Spears
```



```
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 canceledyes
reloading...
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Master of Puppets (by:Metallica)
2   Ride The Lightning (by:Metallica)
4   Bad (by:Michael Jackson)
5   Dangerous (by:Michael Jackson)
8   Baby One More Time (by:Britney Spears)
======================================
```



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

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

1   Master of Puppets (by:Metallica)
2   Ride The Lightning (by:Metallica)
4   Bad (by:Michael Jackson)
5   Dangerous (by:Michael Jackson)
8   Baby One More Time (by:Britney Spears)
======================================
```

Summary

After finishing this module, I found that functions can be called anywhere in a Python program, including calling functions within other functions. They do provide a couple benefits like allowing the same piece of code to run multiple times and functions break programs into smaller components.

Appendix

CDInventory.py using classes and functions

Link to my github.com repository for Assignment 06 https://github.com/Smileski/Assignment_06