Andrew Yeo
March 11, 2020
Foundations of Programming: Python
Github Repository: https://github.com/andyhyeo/assignment_08

Module 08
Knowledge Document

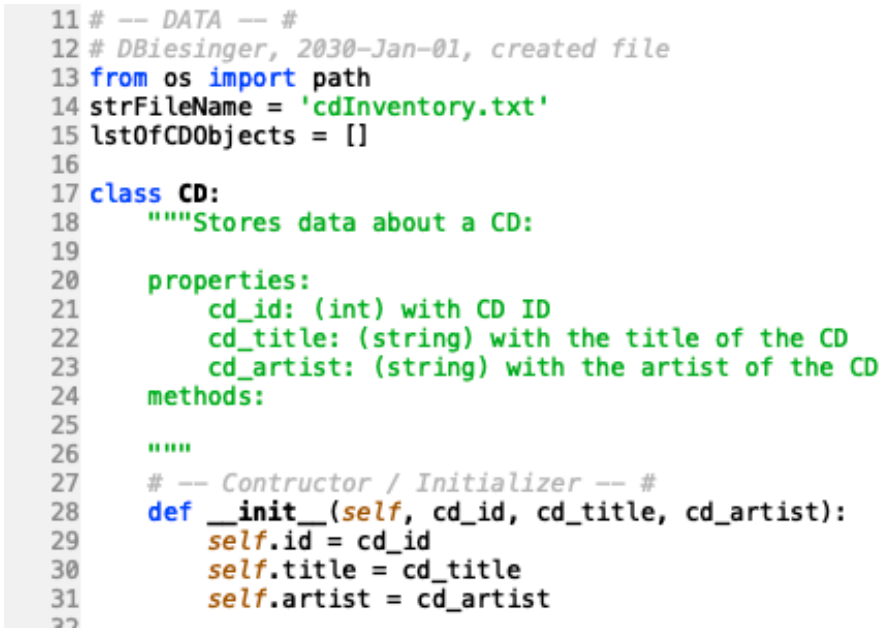
Intro

In this week's assignment, we go over Object Oriented Programming. Object Oriented Programming makes use of classes which are the blueprints for objects. The class maintains then data and the operations of the object. The object is an "instantiation" of the class. And the "features" of the objects are inherited from its class (attributes, properties, and methods)

Details

In this week's assignment we are instructed to build the CD inventory script from scratch. I l largely already used what the script that we have been using in order to focus on the parts that were the topics of this weeks modules: object oriented programming, defining classes, methods, properties, and attributes.

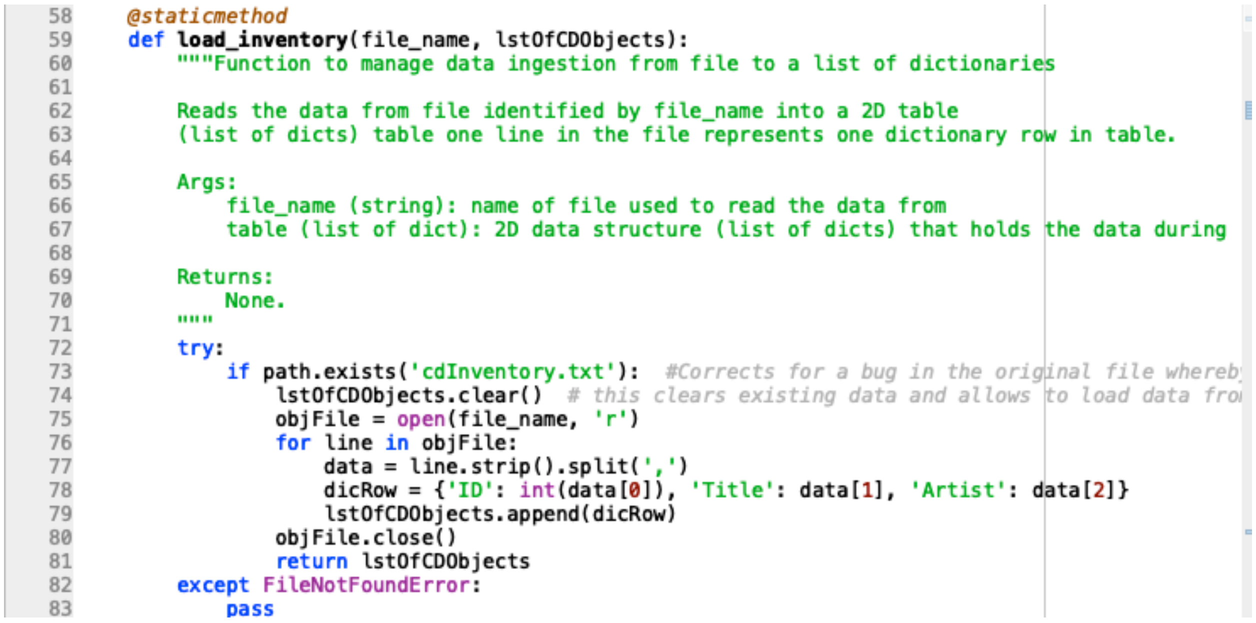
Class CD



```python
# -- DATA -- #
# DBiesinger, 2030-Jan-01, created file
from os import path
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:

    """
    # -- Contructor / Initializer -- #
    def __init__(self, cd_id, cd_title, cd_artist):
        self.id = cd_id
        self.title = cd_title
        self.artist = cd_artist
```

Figure 1: Building the CD class with attributes of id, title, and artist.

The CD class was built simply by adding parameters. Even though I have gotten a little bit of purchase on Object Oriented Programming concepts, I still had to build this largely by emulating the script within the module notes. I tested it within the console to make sure that it probably stored data when called.



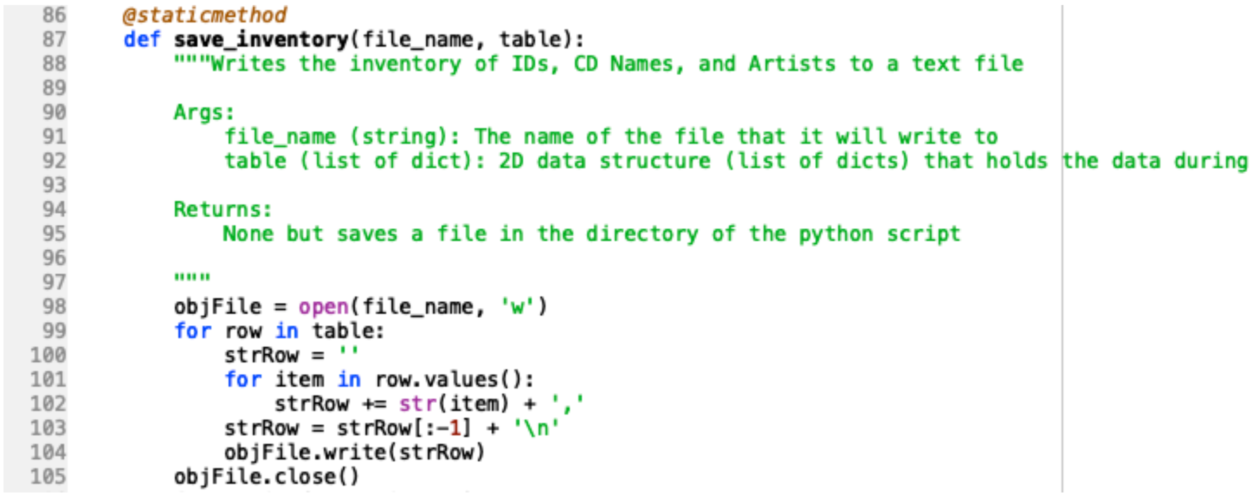
```python
    @staticmethod
    def load_inventory(file_name, lstOfCDObjects):
        """Function to manage data ingestion from file to a list of dictionaries

        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

        Returns:
            None.
        """
        try:
            if path.exists('cdInventory.txt'):  #Corrects for a bug in the original file whereb
                lstOfCDObjects.clear()  # this clears existing data and allows to load data fro
                objFile = open(file_name, 'r')
                for line in objFile:
                    data = line.strip().split(',')
                    dicRow = {'ID': int(data[0]), 'Title': data[1], 'Artist': data[2]}
                    lstOfCDObjects.append(dicRow)
                objFile.close()
                return lstOfCDObjects
        except FileNotFoundError:
            pass
```

Figure 2: Building the FileIO class with the method of load_inventory.

Class File IO with Method load_inventory

Figure 2 shows the part of the script where I built the load_inventory method of within the class FileIO. The module goes over concepts and best practices of writing class such as writing constructors, attributes, and properties but I was not precisely sure how to apply that and if I should have applied it. It was largely a copy and paste from the previous scripts. For simplicity, I opted to use the non-pickling loading strategy. And as shown, I wrote in the structured error handling to continue with the script if the file is not found.

Class File IO with Method save_inventory



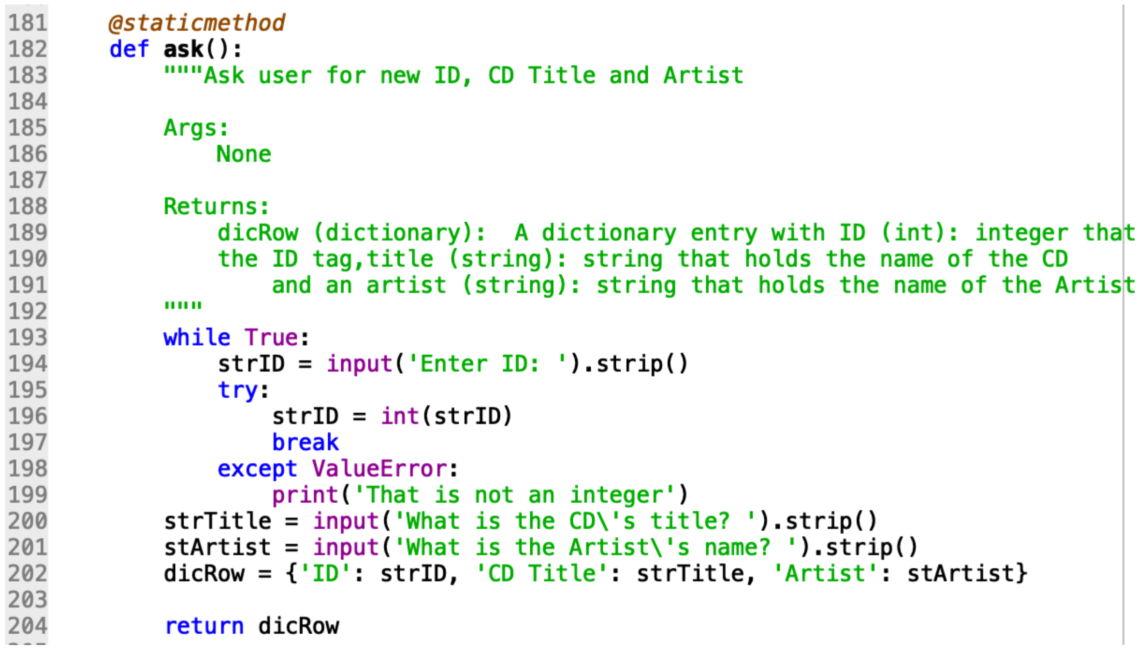
```
    @staticmethod
    def save_inventory(file_name, table):
        """Writes the inventory of IDs, CD Names, and Artists to a text file

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

        Returns:
            None but saves a file in the directory of the python script

        """
        objFile = open(file_name, 'w')
        for row in table:
            strRow = ''
            for item in row.values():
                strRow += str(item) + ','
            strRow = strRow[:-1] + '\n'
            objFile.write(strRow)
        objFile.close()
```

Figure 3: Building the FileIO class with the method of save_inventory.

The load_inventory script was an amalgamation of previous scripts and so to was the the save_inventory.

Try-Except Error Structure Handling for ValueError in IO.ask() function

I also successfully substituted the try-except structure within the IO.ask() function which took a fair amount of trial error. I am learning that I can use all of the Spyder console in order to help me debug a code. For instance, I can run just line by line of the code. I can use the variable explorer as feedback if a variable was defined or changed the way that I wanted to.

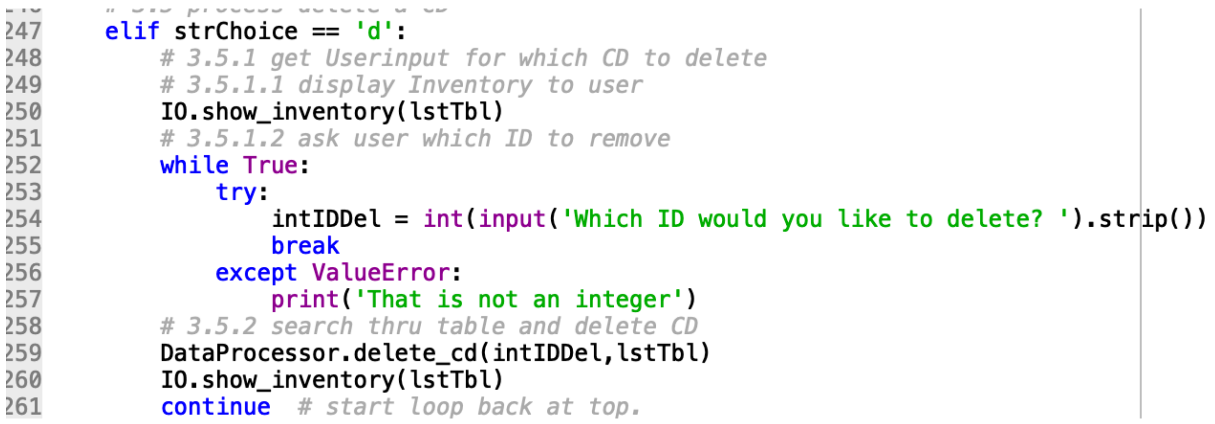


```
    @staticmethod
    def ask():
        """Ask user for new ID, CD Title and Artist

        Args:
            None

        Returns:
            dicRow (dictionary):  A dictionary entry with ID (int): integer that holds
            the ID tag,title (string): string that holds the name of the CD
                and an artist (string): string that holds the name of the Artist.
        """
        while True:
            strID = input('Enter ID: ').strip()
            try:
                strID = int(strID)
                break
            except ValueError:
                print('That is not an integer')
        strTitle = input('What is the CD\'s title? ').strip()
        stArtist = input('What is the Artist\'s name? ').strip()
        dicRow = {'ID': strID, 'CD Title': strTitle, 'Artist': stArtist}

        return dicRow
```

Figure 4: Substituting Structured Error Handling in the Ask Function

Try-Except Error Structure Handling to handle not-integers when deleting

Figure 5 shows the part of the CD Inventory script which prompts the user to that to put in an integer when they have not done so succesfully.

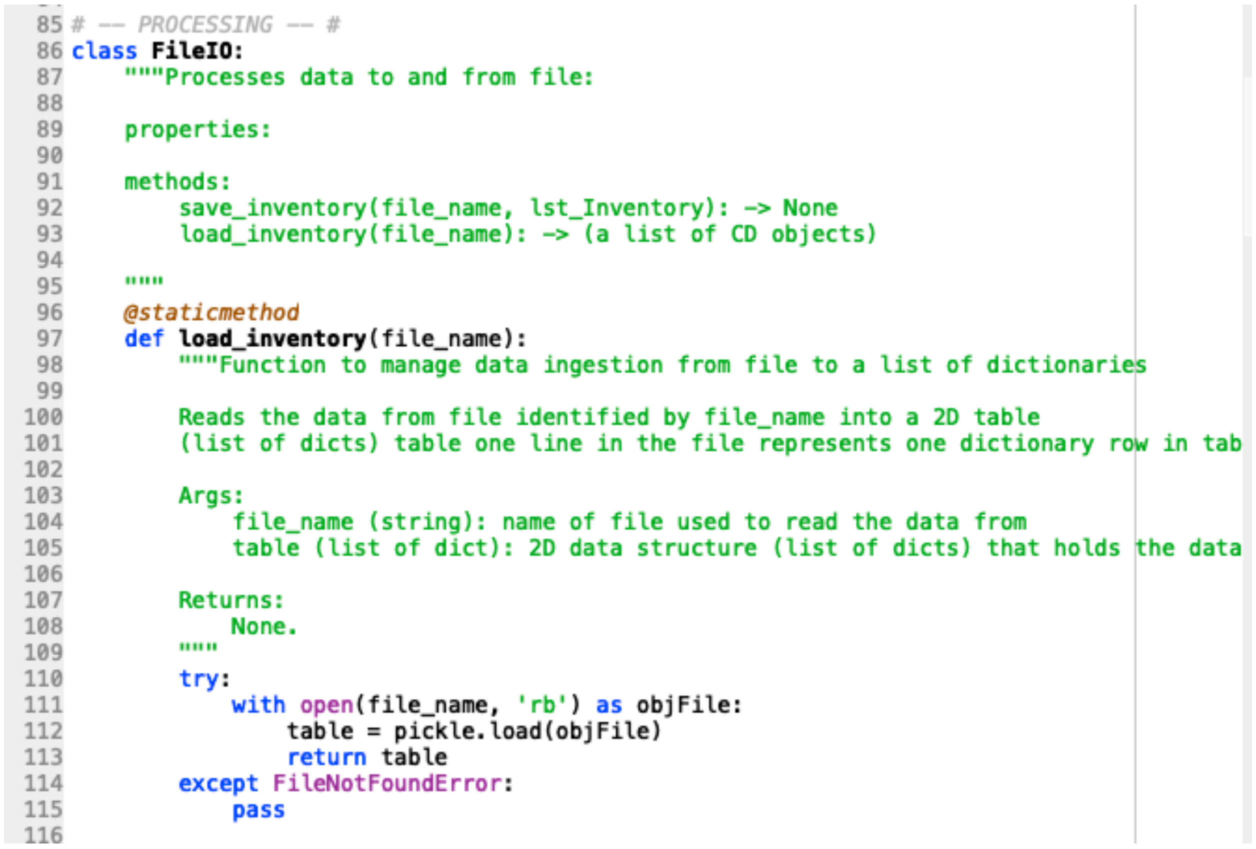


```python
    elif strChoice == 'd':
        # 3.5.1 get Userinput for which CD to delete
        # 3.5.1.1 display Inventory to user
        IO.show_inventory(lstTbl)
        # 3.5.1.2 ask user which ID to remove
        while True:
            try:
                intIDDel = int(input('Which ID would you like to delete? ').strip())
                break
            except ValueError:
                print('That is not an integer')
        # 3.5.2 search thru table and delete CD
        DataProcessor.delete_cd(intIDDel,lstTbl)
        IO.show_inventory(lstTbl)
        continue  # start loop back at top.
```

Figure 4: Error Structure Handling to handle not-integers when prompted to delete

FileIO.load_inventory

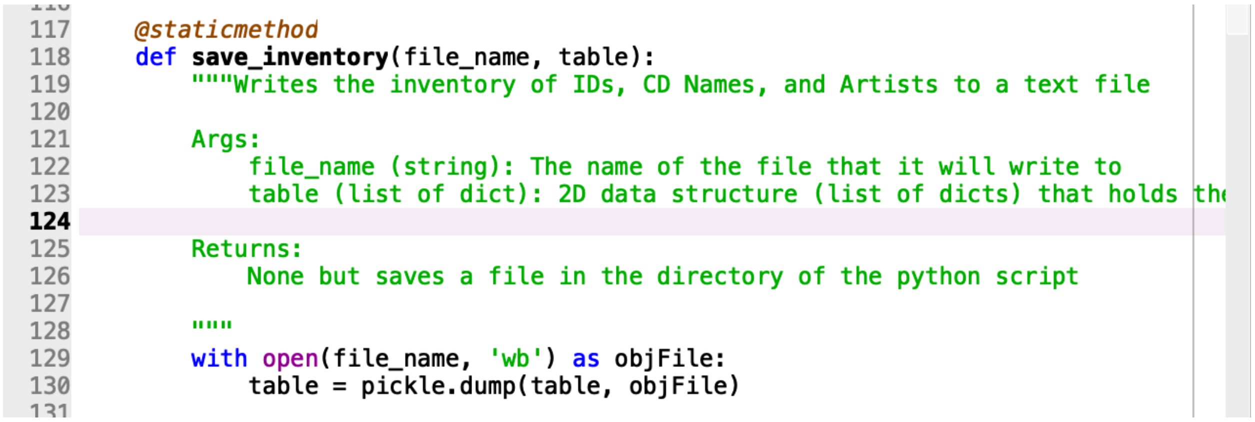
Figure 5: Reading Files into the Script



```python
# -- PROCESSING -- #
class FileIO:
    """Processes data to and from file:

    properties:

    methods:
        save_inventory(file_name, lst_Inventory): -> None
        load_inventory(file_name): -> (a list of CD objects)

    """
    @staticmethod
    def load_inventory(file_name):
        """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 tab

        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

        Returns:
            None.
        """
        try:
            with open(file_name, 'rb') as objFile:
                table = pickle.load(objFile)
                return table
        except FileNotFoundError:
            pass

```

The FileIO class is populated with the load_inventory function which was largely a derivative of the pickle method. After researching online, it looks as if pickling was the best way to manage the reading of files into the script. This is corroborated by the fact that we went over pickling in preceding courses.

FileIO.save_inventory - The File Saving Function



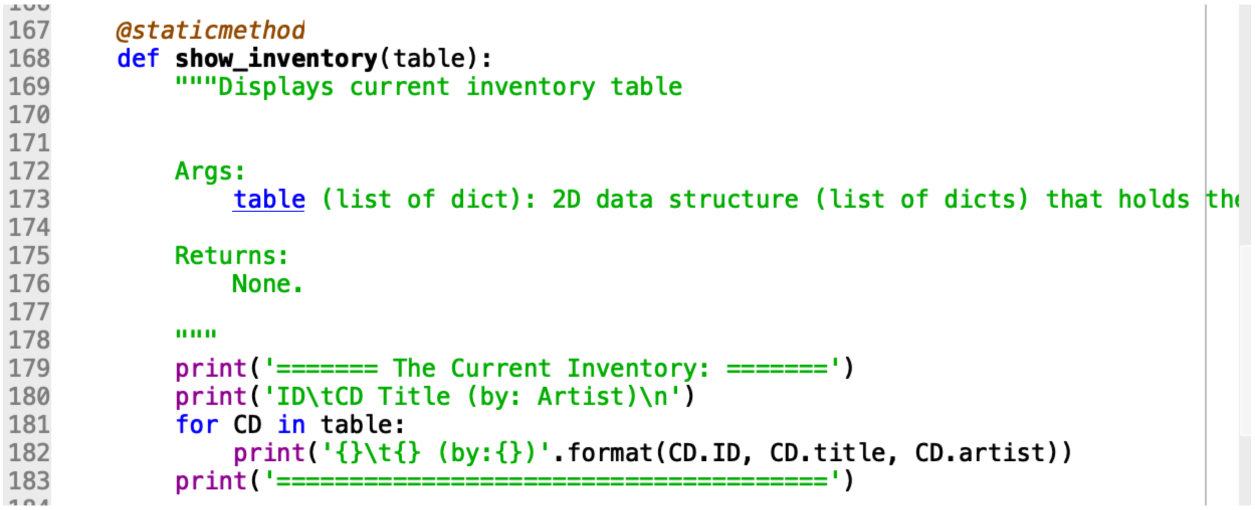
```
    @staticmethod
    def save_inventory(file_name, table):
        """Writes the inventory of IDs, CD Names, and Artists to a text file

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

        Returns:
            None but saves a file in the directory of the python script

        """
        with open(file_name, 'wb') as objFile:
            table = pickle.dump(table, objFile)
```

Figure 6: Writing Files from the Script

The save_inventory method of the FileIO class is largely a derivative from a subsequent scripts pickling method. This seemed to be the most viable solution for handle class and objects.

FileIO.show_inventory - The Inventory Showing Function

Figure 7: Showing Inventory



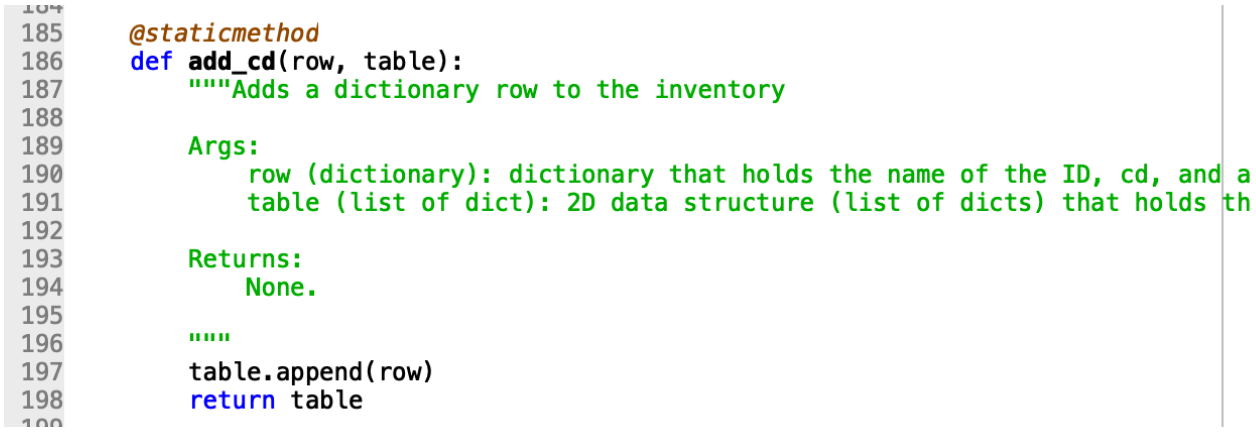
```
    @staticmethod
    def show_inventory(table):
        """Displays current inventory table


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

        Returns:
            None.

        """
        print('======= The Current Inventory: =======')
        print('ID\tCD Title (by: Artist)\n')
        for CD in table:
            print('{}\t{} (by:{})'.format(CD.ID, CD.title, CD.artist))
        print('======================================')
```

Douglas, our class TA, wrote all of us to address a question that many of us must have posed to him - "Should we start from scratch or should we adjust previously written code?" As you can probably tell, I took the latter strategy. This code was largely readjusted to call on the CDs to display.

IO.addCD - The CD adding method

The CD adding method, ask Douglas graciously explained, is a simple method of the table object.

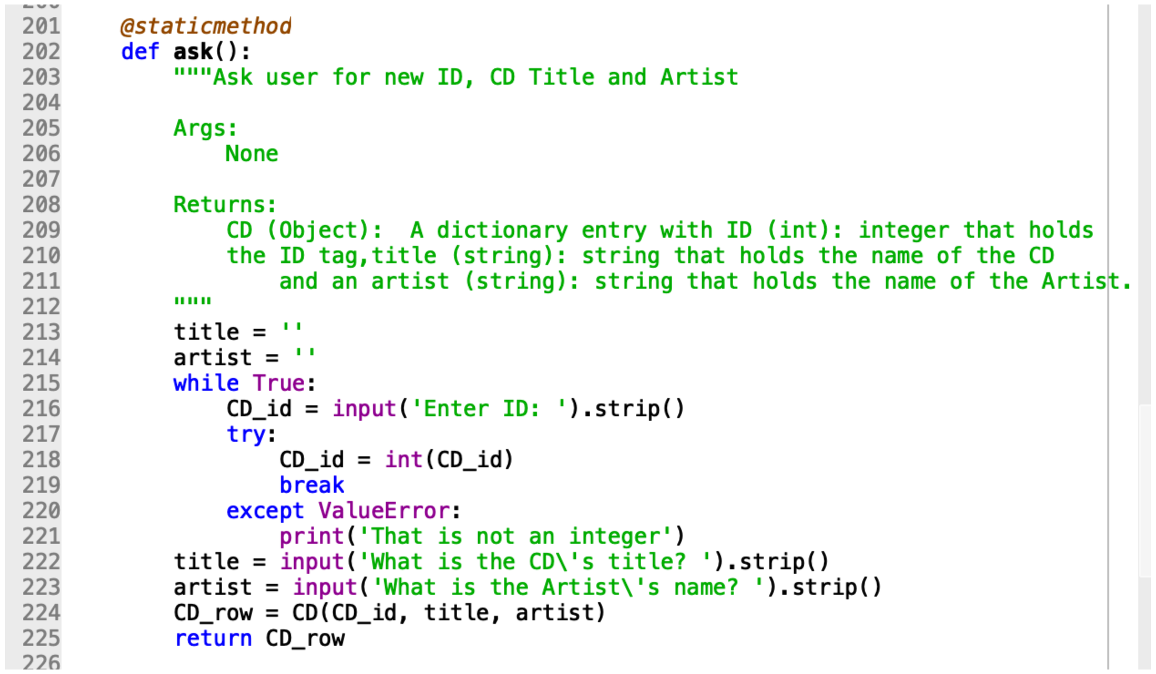
Figure 8: Add CD



```
    @staticmethod
    def add_cd(row, table):
        """Adds a dictionary row to the inventory

        Args:
            row (dictionary): dictionary that holds the name of the ID, cd, and a
            table (list of dict): 2D data structure (list of dicts) that holds th

        Returns:
            None.

        """
        table.append(row)
        return table
```

IO.ask - The CD information entering method

Figure 9 : Add CD



```
    @staticmethod
    def ask():
        """Ask user for new ID, CD Title and Artist

        Args:
            None

        Returns:
            CD (Object):  A dictionary entry with ID (int): integer that holds
            the ID tag,title (string): string that holds the name of the CD
                and an artist (string): string that holds the name of the Artist.
        """
        title = ''
        artist = ''
        while True:
            CD_id = input('Enter ID: ').strip()
            try:
                CD_id = int(CD_id)
                break
            except ValueError:
                print('That is not an integer')
        title = input('What is the CD\'s title? ').strip()
        artist = input('What is the Artist\'s name? ').strip()
        CD_row = CD(CD_id, title, artist)
        return CD_row
```

The CD adding method takes advantage of the CD class by entering information through the CD object.

Summary

As Dirk mentions in his notes and lectures, I do not yet follow appreciate the power of Object Oriented Programming. Dirk mentions that it takes more than just one time using it appreciate its power and beauty. I figure that some of the ways that I can come to appreciate it is to try and building really difficult code without it. Also as Dirk mentions, its usage is probably so standardized and par for the course that it might be difficult to "learn" it again.

Some of the future investigations can include adding the features of pickling to this code in order to increase efficiency of data storage and management. Another feature that can be looked into adding is the feature of preventing duplicate IDs to be stored.

References

https://www.youtube.com/watch?v=lkXfgP-fAkY&feature=youtu.be
https://www.youtube.com/watch?v=KZdFvyCOLUQ&feature=youtu.be
https://www.youtube.com/watch?v=qvRyls8NX-E&feature=youtu.be
https://www.youtube.com/watch?v=QuwU34OT4XA&feature=youtu.be
https://www.youtube.com/watch?v=swAXTwW6xoA&feature=youtu.be
https://www.youtube.com/watch?v=5MM6IaESdQ0&feature=youtu.be
https://realpython.com/python3-object-oriented-programming/
https://www.youtube.com/watch?v=IHaTbJPdB-s&feature=youtu.be

Appendix: The Code

Made using Planet B's Syntax Highlighter

```
#-------------------------------------------#
# Title: Assignmen08.py
# Desc: Assignnment 08 - Working with classes
# Change Log: (Who, When, What)
# AYeo, 2020-Mar-10 9:06 PM,  Added working CD class
# AYeo, 2020-Mar-10 1:52 PM,  Added Main Body from previous script
# AYeo, 2020-Mar-10 2:99 PM,  Added Structured Error Handling
# AYeo, 2020-Mar-15 9:29 PM, Fixed CD class to have property, private methods, setters
# AYeo, 2020-Mar-15 9:29 PM, Edited FileIO to use CD class, HAVE NOT TESTED
# AYeo, 2020-Mar-15 9:29 PM, Edited FileIO to use CD class, Successfully Tested
# AYeo, 2020-Mar-16 5:00 PM, Correct FileIO.load_inventory to use pickle, Successfully Tested
# AYeo, 2020-Mar-16 5:00 PM, Correct FileIO.save_inventory to use pickle, NOT TESTED
# AYeo, 2020-Mar-16 5:00 PM, Correct FileIO.save_inventory to use pickle, Tested


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

# -- DATA -- #
# DBiesinger, 2030-Jan-01, created file
from os import path
import pickle

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:

    """
    # -- Contructor -- #
    def __init__(self, cd_id, cd_title, cd_artist):
        # -- Attributes -- #
        self.__ID = cd_id
        self.__title = cd_title
        self.__artist = cd_artist

    # -- Properties -- #
    @property
    def ID(self):
        return self.__ID

    @property
    def title(self):
        return self.__title.title()

    @property
    def artist(self):
        return self.__artist.title()

    @ID.setter
    def ID(self, value):
        if str(value).isnumeric():
            self.__ID = value
        else:
            raise Exception('The ID must be numeric')

    @title.setter
    def title(self, value):
        if str(value).isnumeric():
            raise Exception('The title must be a string')
        else:
            self.__title = value

    @artist.setter
    def artist(self, value):
        if str(value).isnumeric():
            raise Exception('The artist must be a string')
        else:
            self.__artist = value
```

```python

    # -- Methods -- #
    def __str__(self):
        return self.ID
        return self.title
        return self.artist


# -- PROCESSING -- #
class FileIO:
    """Processes data to and from file:

    properties:

    methods:
        save_inventory(file_name, lst_Inventory): -> None
        load_inventory(file_name): -> (a list of CD objects)

    """
    @staticmethod
    def load_inventory(file_name):
        """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.
        """
        try:
            with open(file_name, 'rb') as objFile:
                table = pickle.load(objFile)
                return table
        except FileNotFoundError:
            pass

    @staticmethod
    def save_inventory(file_name, table):
        """Writes the inventory of IDs, CD Names, and Artists to a text file

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

        Returns:
            None but saves a file in the directory of the python script

        """
        with open(file_name, 'wb') as objFile:
            table = pickle.dump(table, objFile)


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

```python
            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 CD in table:
            print('{}\t{} (by: {})'.format(CD.ID, CD.title, CD.artist))
        print('======================================')

    @staticmethod
    def add_cd(row, table):
        """Adds a dictionary row to the inventory

        Args:
            row (dictionary): dictionary that holds the name of the ID, cd, and artist
            table (list of dict): 2D data structure (list of dicts) that holds the data during runtime

        Returns:
            None.

        """
        table.append(row)
        return table


    @staticmethod
    def ask():
        """Ask user for new ID, CD Title and Artist

        Args:
            None

        Returns:
            CD (Object):  A dictionary entry with ID (int): integer that holds
            the ID tag,title (string): string that holds the name of the CD
                and an artist (string): string that holds the name of the Artist.
        """
        title = ''
        artist = ''
        while True:
            CD_id = input('Enter ID: ').strip()
            try:
                CD_id = int(CD_id)
                break
            except ValueError:
                print('That is not an integer')
        title = input('What is the CD\'s title? ').strip()
        artist = input('What is the Artist\'s name? ').strip()
        CD_row = CD(CD_id, title, artist)
        return CD_row

class DataProcessor:
    @staticmethod
    def delete_cd(intIDDel,table):
        """Deletes a CD row from the table

        Args:
            intIDDel (int): ID which indicate which entry user would like to delete
            table (list of dict): 2D data structure (list of dicts) that holds the data during runtime

        Returns:
              table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
        """
        intRowNr = -1
        blnCDRemoved = False
        for CD in table:
            intRowNr += 1
            if CD.ID == intIDDel:
                del table[intRowNr]
                blnCDRemoved = True
                break
            if blnCDRemoved:
                print('The CD was removed')
            else:
                print('Could not find this CD!')
        return table

# -- Main Body of Script -- #
# Load data from file into a list of CD objects on script start
FileIO.load_inventory(strFileName)

# 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':
        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 fro
m file. otherwise reload will be canceled')
        if strYesNo.lower() == 'yes':
            print('reloading...')
            lstOfCDObjects = FileIO.load_inventory(strFileName)
            IO.show_inventory(lstOfCDObjects)
        else:
            input('canceling... Inventory data NOT reloaded. Pres
s [ENTER] to continue to the menu.')
            IO.show_inventory(lstOfCDObjects)
        continue  # start loop back at top.
    # 3.3 process add a CD
    elif strChoice == 'a':
```

```python
        # 3.3.1 Ask user for new ID, CD Title and Artist
        row = IO.ask()
        # 3.3.2 Add item to the table
        IO.add_cd(row,lstOfCDObjects)
        IO.show_inventory(lstOfCDObjects)
        continue  # start loop back at top.
    # 3.4 process display current inventory
    elif strChoice == 'i':
        IO.show_inventory(lstOfCDObjects)
        continue  # start loop back at top.
    # 3.5 process delete a CD
    elif strChoice == 'd':
        # 3.5.1 get Userinput for which CD to delete
        # 3.5.1.1 display Inventory to user
        IO.show_inventory(lstOfCDObjects)
        # 3.5.1.2 ask user which ID to remove
        while True:
            try:
                intIDDel = int(input('Which ID would you like to delete? ').strip())
                break
            except ValueError:
                print('That is not an integer')
        # 3.5.2 search thru table and delete CD
        DataProcessor.delete_cd(intIDDel,lstOfCDObjects)
        IO.show_inventory(lstOfCDObjects)
        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.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.
    # 3.7 catch-all should not be possible, as user choice gets vetted in IO, but to be save:
    else:
        print('General Error')

```

Appendix: The Spyder Output

```
runfile('/Users/andyyeo/Google Drive/Intro to Python/Mod_08/Assignment08/Assignment_08.py', wdir='/Users/andyyeo/Google Drive/Intro to Python/Mod_08/Assignment08')
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? asdfa

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

1       Asdfa (by:Vasdc)
======================================
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: 2

```

```
What is the CD's title? asc

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

1       Asdfa (by:Vasdc)
2       Asc (by:Casdg)
======================================
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       Asdfa (by:Vasdc)
2       Asc (by:Casdg)
======================================

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]: d

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

```

```
1       Asdfa (by:Vasdc)
2       Asc (by:Casdg)
======================================

Which ID would you like to delete? 1
======= The Current Inventory: =======
ID      CD Title (by: Artist)

2       Asc (by:Casdg)
======================================
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)

2       Asc (by:Casdg)
======================================

Which ID would you like to delete? 2
======= 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]: 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       Asdfa (by:Vasdc)
2       Asc (by:Casdg)
======================================
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       Asdfa (by:Vasdc)
2       Asc (by:Casdg)
======================================
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]: x
```

Appendix: The Terminal Output

```
Last login: Wed Mar 11 14:14:33 on ttys000
^[[A^[[A
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) Andys-MacBook-Pro:~ andyyeo$ cd desktop/assignment08
(base) Andys-MacBook-Pro:assignment08 andyyeo$ zip -r dir.zip . -x ".*" -x "_MAXOSX"
  adding: Assignment08 Knowledge Document .pdf (deflated 29%)
  adding: Assignment_08_Starter.py (deflated 71%)
(base) Andys-MacBook-Pro:assignment08 andyyeo$
  [Restored Mar 16, 2020 at 7:00:13 PM]
Last login: Wed Mar 11 15:00:02 on ttys000

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) Andys-MacBook-Pro:assignment08 andyyeo$ clear






(base) Andys-MacBook-Pro:assignment08 andyyeo$ cd desktop/assignment08
-bash: cd: desktop/assignment08: No such file or directory
(base) Andys-MacBook-Pro:assignment08 andyyeo$ python Assignment_08.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]: a
```

```

Enter ID: 1
What is the CD's title? asdfd
What is the Artist's name? casd
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Asdfd (by:Casd)
======================================
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: 2
What is the CD's title? asdfasdg
What is the Artist's name? asgdg
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Asdfd (by:Casd)
2   Asdfasdg (by:Asgdg)
======================================
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   Asdfd (by:Casd)
```

```
2   Asdfasdg (by:Asgdg)
======================================
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]: d

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

1   Asdfd (by:Casd)
2   Asdfasdg (by:Asgdg)
======================================
Which ID would you like to delete? 1
======= The Current Inventory: =======
ID  CD Title (by: Artist)

2   Asdfasdg (by:Asgdg)
======================================
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)

2   Asdfasdg (by:Asgdg)
======================================
Which ID would you like to delete? 2
======= 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]: 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   Asdfd (by:Casd)
2   Asdfasdg (by:Asgdg)
======================================
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   Asdfd (by:Casd)
2   Asdfasdg (by:Asgdg)
======================================
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]: x

(base) Andys-MacBook-Pro:assignment08 andyyeo$

```