Christine Buffalow
2021-Feb-19
Foundations of Programming
Assignment 6

# Creating a CD Inventory, version 3.0

## Introduction

For this assignment, we were given a Python file with an example solution from our second CD Inventory project. For version 3.0, our task was to update the file in various locations so that it used functions instead. The first section of this document goes through all of the added functions and the second section of this document goes through the changes made to the main body of the script.

## Opening Remarks

For the most part, the variable names within functions were changed to avoid shadowing a global variable, even if it did not appear to have any effect on the functioning of the program during my testing. I also added `Input` statements after each section that ask the user to press 'Enter' before returning to the menu as I felt this created a better experience for the user. During my initial testing, I found that I often didn't notice `print` statements that occurred towards the end of a section because I was distracted by the menu popping up immediately after they were displayed. The `input` statement forced me to acknowledge what was on the screen before returning to the main menu. Personally, I found it to be a better experience when I had the ability to control the flow of the program as a user.

As with the previous assignment, CD names were generated using Cool Generator's Album Name Generator[1] and the artist names were generated using Random Name Generator[2] to keep things fun and interesting. No reference to actual album names or people is intended.

## DataProcessor.process_cd_data

The `DataProcessor.process_cd_data` function takes 3 arguments: `cd_id` (integer), `cd_title` (string), and `cd_artist` (string). These three arguments are then combined into a single dictionary called `cd_info` using the keys 'ID', 'Title', and 'Artist', respectively. The function then returns the dictionary `cd_info`. This can all be seen in Listing 1.

```python
    @staticmethod
    def process_cd_data(cd_id, cd_title, cd_artist):
        """Function that takes user input about CD and formats data into a dictionary.

        Keys are "ID", 'Title', and 'Artist' for each dictionary and the values are assigned
        from cd_id, cd_title, and cd_artist, respectively.

        Args:
            cd_id (int): ID of CD, generated by program
            cd_title (str): name of CD, provided by user
            cd_artist (str): name of artist, provided by user

        Returns:
            cd_info (dictionary): cd_id, cd_title, and cd_artist contained within a single dictionary

        """
        cd_info = {'ID': cd_id, 'Title': cd_title, 'Artist': cd_artist}
```

[1] https://www.coolgenerator.com/album-name-generator, accessed 2021-Feb-20.
[2] http://random-name-generator.info/, accessed 2021-Feb-20.

```python
        return cd_info
```

*Listing 1 - DataProcessor.process_cd_data function*

## DataProcessor.process_delete

The `process_delete` function takes 2 arguments: `cd_to_delete` (integer) and `table` (2D list of dictionaries). The `cd_to_delete` argument states the ID number of the CD the user selected to delete. The list `table` is the current CD inventory.

To begin, the variable `row_number` is set to '-1' and the `CD_found` flag is set to 'False'. The program then cycles through each row (a.k.a. dictionary) of `table`, counts the row/dictionary (`'row_number += 1'`), and determines if the key:value pair in that row/dictionary with the key of "ID" has the desired ID number for its value. If it does, `CD_found` is set to 'True' and the `'for'` loop is stopped with a `'break'` statement. If not, the `'for'` loop continues until it finds the desired ID number or it reaches the end of the list `table`.

In the end, the function returns the `row_number` and `CD_found` variables. If the CD was found, `row_number` will be the index of the row it was found on (because the `'for'` loop is stopped with a `'break'` once the CD is found and the `row_number` does not count any more rows after this) and `CD_found` will be 'True'. If the CD was not found, `row_number` will be the index of the last row in the table and `CD_found` will still be 'False'. This information is then used within the main body of the text to alter the current inventory and print the appropriate statement. The code is seen in Listing 2.

```python
@staticmethod
    def process_delete(cd_to_delete, table):
        """Function that processes which row index needs to be deleted based upon input from user.

        For each dictionary, the function determines if the key:value pair in that row/dictionary
        with the key of "ID" has the desired ID number for its value.  If yes, row_number stops counting
        at that row and CD_found is set to 'True'.

        Args:
            cd_to_delete (int): ID number of CD that user wants to delete
            table (list of dicts): 2D data structure (list of dicts) that holds data during runtime

        Returns:
            row_number (int): counter of rows (stops on row if desired CD found, otherwise returns index of last row)
            CD_found (Boolean): flag that states if desired CD found (True = yes, False = no)
        """
        row_number = -1
        CD_found = False
        for row in table:
            row_number += 1
            if row['ID'] == cd_to_delete:
                CD_found = True
                break

        return row_number, CD_found
```

*Listing 2 - DataProcessor.process_delete function*

## FileProcessor.read_file

The `FileProcessor.read_file` function was supplied with `Assignment06_Starter.py`, but I added the `os.path.exist` check to avoid an error if `CDInventory.txt` does not exist. This can be seen in Listing 3.

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

        Function checks to make sure specified txt file exists.  If yes, continues by reading
        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.
        """
               if os.path.exists(file_name):
                   objFile = None  # file object
                   table.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(',')
                       cd_info = {'ID': int(data[0]), 'Title': data[1], 'Artist': data[2]}
                       table.append(cd_info)
                   objFile.close()
               else: pass
```

*Listing 3 - DataProcessor.read_file function*

## FileProcessor.calculate_cd_id

The code for the `FileProcessor.calculate_cd_id` function was copied from my previous assignment.  It is based upon the fact that the text file, `CDInventory.txt`, has all the information about one CD on its own line. Within each line, each individual piece of data is separated by a comma, and the information is organized such that the first piece of data in the line is always the ID number.  In more detail:

(1) checking to see if `CDInventory.txt` exists (and simply setting `cd_id` = '0' if it does not exist);
(2) reading `CDInventory.txt` into current memory as one giant string in the variable `objFile_content`,
(3) separating `objFile_content` into separate strings based on the location of the new line character (`'\n'`) and assigning the results to `objFile_rows`;
(4) subtracting 2 from the length of the `objFile_rows` list to get the index of the final row of data in `objFile_rows` and assigning this integer to `indexFinalRow`; and
(5) taking the final row of data in `objFile_rows` (based on the index found in step 4), separating this string into separate strings based on the location of the comma (','), and assigning the string in index position '0' to `cd_id`.

In essence, the function finds the last row of data in `CDInventory.txt` and reports the ID number used in that row. This becomes the start point in which the rest of the ID numbers are based upon.  I am still not concerned about gaps in the id numbers as long as (1) there are no duplicate IDs and (2) the ID numbers are always increasing.  The code for this can be seen in Listing 4.

```
           @staticmethod
           def calculate_cd_id(file_name):
               """Function to determine last ID used in CDInventory.txt so that next ID number can be assigned

               The function finds the last row of data in CDInventory.txt and reports
               the ID number used in that row.  This becomes the start point in which the rest of the ID numbers
```

```
        are based upon.

        Args:
            file_name (str): name of file used to read data from

        Returns:
            cd_id (int): id number of last used CD_ID in the CDInventory.txt file
        """
        if os.path.exists(file_name):
            objFile = None  # file object
            objFile = open(file_name, 'r')
            objFile_content = objFile.read() #returns all content of file as a giant string
            objFile.close()
            objFile_rows = objFile_content.split(sep= '\n') #separates data into separate items based on location of \n
            indexFinalRow = len(objFile_rows) -2
            cd_id = int(objFile_rows[indexFinalRow].split(sep= ',')[0])
        else:
            cd_id = 0
        return cd_id
```

*Listing 4 - FileProcessor.calculate_cd_id function*

## DataProcessor.write_file

The `DataProcessor.write_file` function was obtained by moving the appropriate code from the main body of `Assignment06_Starter.py` into this function. It has two arguments (`file_name` and `table`) and does not return anything. Since I did not write this code from scratch, I felt it was important to take the time to perform the necessary research in order to fully understand it. From my research, Python has a `list()` function that creates a list from its arguments[3] and the `join()` method takes all the items in an iterable object and combines them into one string using the specified character between them (a comma in this case)[4]. Based on this knowledge, the `DataProcessor.write_file` function works by opening the text file with 'write' access (where the text file is specified in the argument using `file_name`). The function then goes through each row (a.k.a. dictionary) of the list of dictionaries (specified in the argument using `table`) and:

(1) assigns the values from this dictionary into `lstValues`;
(2) converts the item in the first index to a string (it's currently an integer – the ID number);
(3) joins these three values together into one string with a comma between and a new line character at the end; and
(4) writes this final string out to the specified txt file.

Once this is complete, the file is closed. The code can be seen in Listing 5.

```
    @staticmethod
    def write_file(file_name, table):
        """Function to overwrite data from current runtime into a text file.

        For each row, the data is converted into a string with a comma separating
        each piece of data and then a new line is started.  The data is then saved
        out to the designated txt file.

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

[3] https://www.w3schools.com/python/ref_func_list.asp, accessed 2021-Feb-20
[4] https://www.w3schools.com/python/ref_string_join.asp, accessed 2021-Feb-20

```python

        Returns:
            None.
        """
        objFile = None  # file object
        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()
```

*Listing 5 - DataProcessor.write_file function*

## IO.print_menu, IO.menu_choice, IO.show_inventory

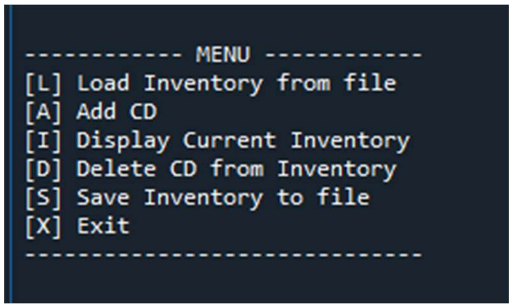
These three functions were supplied with `Assignment06_Starter.py`, but I made some adjustments to the formatting to match (or expand upon) what I did in the previous assignment.

For the `IO.print_menu` and `IO.menu_choice` functions, I made changes to both the functionality and formatting. To begin, I again changed the "menu letters" to upper-case to improve readability (upper-case letters tend to be easier to distinguish). I also added uppercase letters to the menu tuple so that the code works with either upper- or lower-case letters. In addition, I made sure that the first letter of every menu item was capitalized.

Following this, I played around with some print statements and centered 'MENU' using the '-' character to form a string 30 characters long. I also created a string with the '-' character at the bottom of the menu that was 30 characters long to create some separation between the menu and everything else. The code for `IO.print_menu` can be seen in the Appendix CDInventory.py, the code for `IO.menu_choice` can be seen in Listing 6, and the result can be seen in Figure 1.

```python
    @staticmethod
    def menu_choice():
        """Gets user input for menu selection.


        Args:
            None.

        Returns:
            choice (string): an upper 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', '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
```

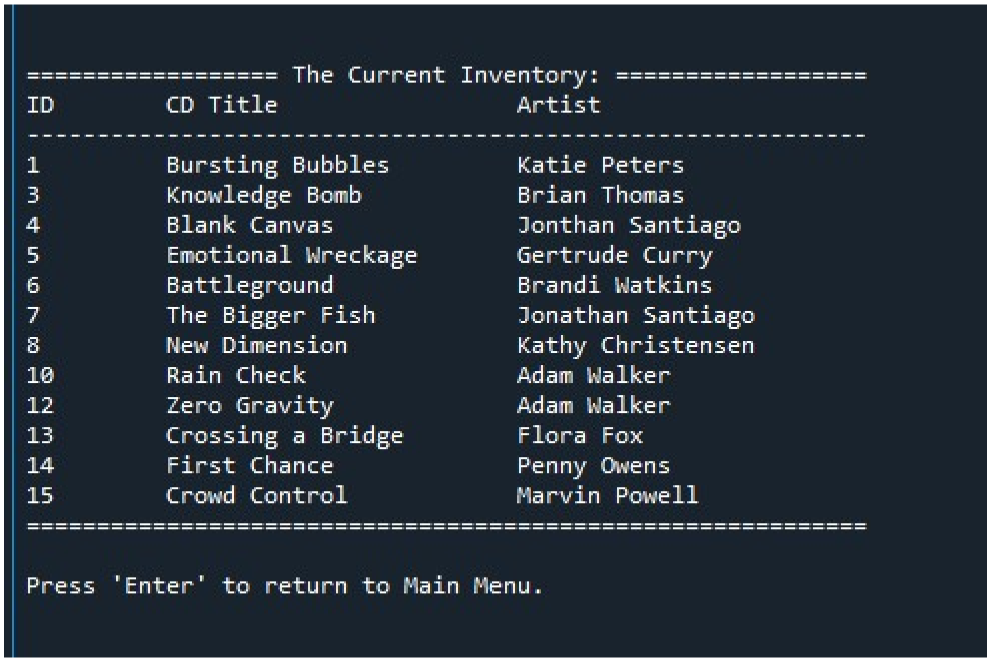
*Listing 6 - IO.menu_choice function*



```
------------ 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
------------------------------
```

*Figure 1 – Menu of CDInventory.py in Spyder*

For `IO.show_inventory`, I just made some adjustments to the formatting. Instead of using `'print('{}\t{} (by:{})'.format(*row.values()))'`, I specified the lengths of each string so that it forms a table with three set columns sizes with all the text starting on the left: `'print('{:<10}{:<25}{:<25}'.format(*row.values()))'`. The same was done for the "header" so it would all line up in a highly organized fashion and be easy to read. I also updated the decorative lines above and below the inventory so that they would also be exactly 60 characters long to match the rest of the table (60 = 10 + 25 + 25) and I used the `center()` method to put the title in the center of this decorative string. The code can be seen in Appendix CDInventory.py and the result can be seen in Figure 2.



```
================== The Current Inventory: ==================
ID        CD Title                 Artist
------------------------------------------------------------
1         Bursting Bubbles         Katie Peters
3         Knowledge Bomb           Brian Thomas
4         Blank Canvas             Jonthan Santiago
5         Emotional Wreckage       Gertrude Curry
6         Battleground             Brandi Watkins
7         The Bigger Fish          Jonathan Santiago
8         New Dimension            Kathy Christensen
10        Rain Check               Adam Walker
12        Zero Gravity             Adam Walker
13        Crossing a Bridge        Flora Fox
14        First Chance             Penny Owens
15        Crowd Control            Marvin Powell
============================================================

Press 'Enter' to return to Main Menu.
```

*Figure 2 - View of Current Inventory in CDInventory.py in the Spyder*

## IO.get_cd_data

The function `IO.get_cd_data` was also created by moving the appropriate code from the main body of `Assignment06_Starter.py` into this function. There are no arguments, and it returns two variables: `cd_title` and `cd_artist`. To avoid shadowing global variables, I updated the variables inside the function to new names. I also removed the input for obtaining the ID number of the CD since the code in the main body generates this number automatically. The code for the function can be seen in Listing 7.

```
        @staticmethod
    def get_cd_data():
        """Collects information about CD from user.

        Asks user to input the name of the CD and the artist of the CD.

        Args:
            None

        Returns:
            cd_title (str): name of CD, provided by user
            cd_artist (str): name of artist, provided by user

        """
        cd_title = input('What is the CD\'s title? ').strip()
        cd_artist = input('What is the Artist\'s name? ').strip()
        return(cd_title, cd_artist)
```

*Listing 7 - IO.get_cd_data function*

## Main Program – Section 1 – 3.2

Section 1 (with initial program actions) was provided with `Assignment06_Starter.py`, but I added a function call for the `FileProcessor.calculate_cd_id` function I created. It's designed to read the data file immediately before anything can be added or deleted from it. After the initial read, `intID` simply counts upwards any time a CD is added in section 3.3 (this works since I am not concerned about gaps created between ID numbers when CDs are deleted). I also added a program header with the program's title from the previous assignment. The code for this can be seen in Listing 8.

```
    # 1. When program starts, read in the currently saved inventory, print program header
    FileProcessor.read_file(strFileName, lstTbl)
    intID = FileProcessor.calculate_cd_id(strFileName)
    print('\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n')
    print('The Magic CD Inventory'.center(62))
    print('\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')
```

*Listing 8 - Section 1 of Main Script*

Section 2 – 3.2 were also provided with `Assignment06_Starter.py`, but no changes were made other than a few minor grammatical/spelling updates. The code can be seen in Appendix Listing CDInventory.py.

## Main Program – Section 3.3 – 3.4

The code in Section 3.3 was mostly replaced by functions, either directly or indirectly. The initial value of `intID` was generated in section 1 using the `FileProcessor.calculate_cd_id` function with `strFileName` as its argument. After this initial function call, this function is not accessed again since it is outside the infinite `'while'` loop. Any time the user enters Section 3 to add a cd, the value of 1 is simply added to the value of `intID` to generate the next ID number.

The user inputs for the CD title and Artist are now collected from the user using the function `IO.get_cd_data()` and the values are assigned to `strTitle` and `strArtist`, respectively. This data (along with `intID`) is processed into a single row, or dictionary, of data along using the function call `'DataProcessor.process_cd_data (intID, strTitle, strArtist)'` and the returning dictionary is assigned to `dicRow`. The dictionary `dicRow` is then appended to `lstTbl` within the main body of the program. I did this to avoid altering a "global" entity within a function. The inventory is then displayed using `'IO.show_inventory(lstTbl)'`. The code for this can be seen in Listing 9.

```
        # 3.3 process add a CD
        elif strChoice == 'a':
```

```python
            # 3.3.1 Generate ID and Ask user for CD Title and Artist
            intID += 1
            strTitle, strArtist = IO.get_cd_data()
            # 3.3.2 Add item to the table
            dicRow = DataProcessor.process_cd_data(intID, strTitle, strArtist)
            lstTbl.append(dicRow)
            IO.show_inventory(lstTbl)
            input('CD Added. Press \'Enter\' to return to Main Menu. ')
            continue  # start loop back at top.
```

*Listing 9 - Section 3.3 of Main Script*

Section 3.4 was provided with Assignment06_Starter.py, but no changes were made.  The code can be seen in Appendix Listing CDInventory.py.

## Main Program – Section 3.5

For Section 3.5, the actual process of finding the appropriate row to delete was transferred into the function `DataProcessor.process_delete()`. This function takes the ID number the user input and the current inventory as arguments, and it returns the index of the dictionary/row with the desired CD (if found, otherwise it's simply the index of the final dictionary) and whether or not the CD was found.  These two return values are assigned to `intRowNr` and `blnCDRemoved`, respectively.  If `blnCDRemoved` is 'True' (i.e. the desired CD was found in the current inventory), the dictionary/row is now deleted from the current inventory table based on the index provided in `intRowNr` and a confirmation statement is printed.  As with the 'adding a CD' section, I decided it would be best to only change `lstTbl` within the main body of the code rather than within the function since it is a global entity.  If `blnCDRemoved` is 'False' (i.e. the desired CD was <u>not</u> found in the current inventory), the program simply spits out a `print` statement informing the user that the CD was not found.  This is all controlled with `'if/else'` blocks.  After these blocks, the inventory is displayed (this means that the current inventory will be displayed regardless of whether or not the CD was found). The code for this can be seen in listing 10.

```python
        # 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(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
            intRowNr, blnCDRemoved = DataProcessor.process_delete(intIDDel, lstTbl)
            if blnCDRemoved:
                del lstTbl[intRowNr]
                input('CD #' + str(intIDDel) + ' deleted. Press \'Enter\' to view updated inventory.')
            else:
                input('CD #' + str(intIDDel) + ' not found.  Press \'Enter\' view inventory.')
            IO.show_inventory(lstTbl)
            input('Press \'Enter\' to return to Main Menu. ')
            continue  # start loop back at top.
```

*Listing 10 - Section 3.5 of Main Script*

## Main Program – Section 3.6 – End

For section 3.6, the process of saving the data to the file was transferred from the main body into the function `FileProcessor.write_file()`. This function has two arguments: `strFileName` and `lstTbl`.  Nothing is returned from the function.  The change did not require a lot of work.  The code in the main body was simply replaced with a function call to `FileProcessor.write_file()`.  As in many of the other sections, I added a confirmatory statement within an `'input'` function that confirmed that the file had been saved.  The code for this can be seen in listing 11.

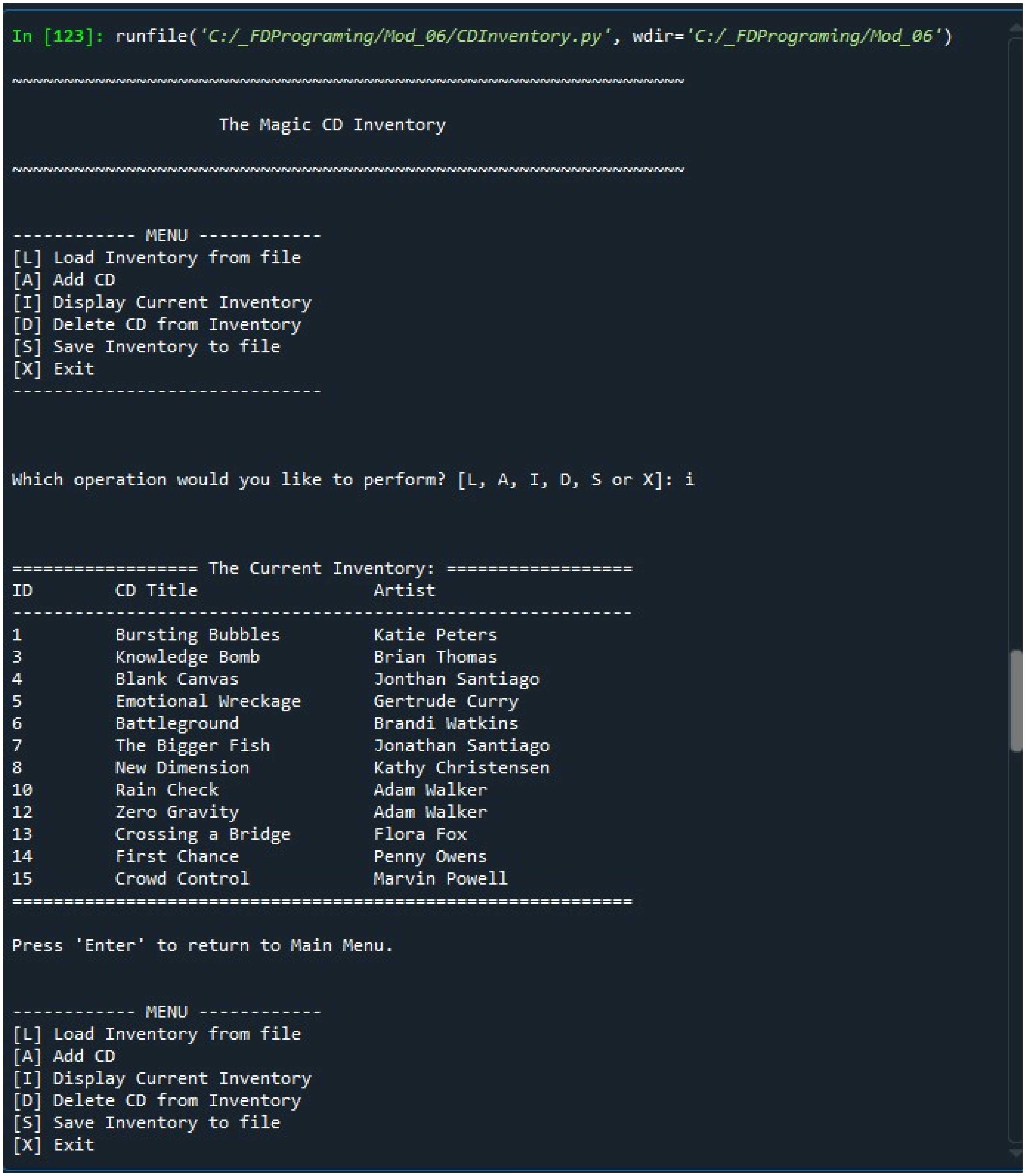
```
            # 3.6 process save inventory to file
            elif strChoice == 's':
                # 3.6.1 Display current inventory and ask user for confirmation to save
                IO.show_inventory(lstTbl)
                strYesNo = input('Save this inventory to file? [y/n] ').strip().lower()
                # 3.6.2 Process choice
                if strYesNo == 'y':
                    # 3.6.2.1 save data
                    FileProcessor.write_file(strFileName, lstTbl)
                    input('The inventory was saved to file. Press \'Enter\' to return to Main Menu. ')

                else:
                    input('The inventory was NOT saved to file. Press [ENTER] to return to Main Menu. ')
                continue  # start loop back at top.
```

*Listing 11 - Section 3.5 of Main Section*

Section 3.7 was provided with Assignment06_Starter.py, but no changes were made.  The code can be seen in Appendix Listing CDInventory.py.

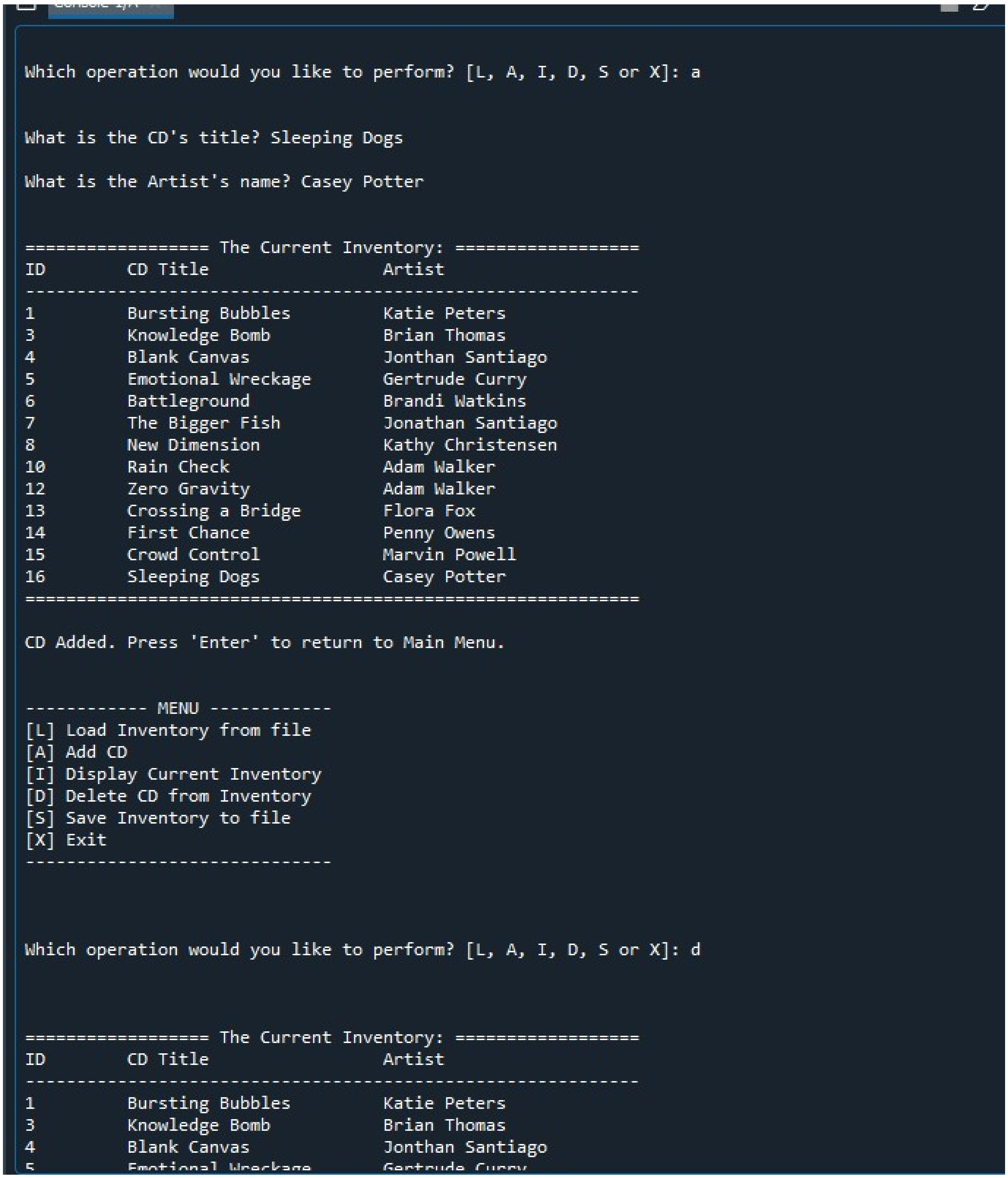
## Running Script in Spyder

The program was run successfully in Spyder, as shown in Figures 3 – 6.



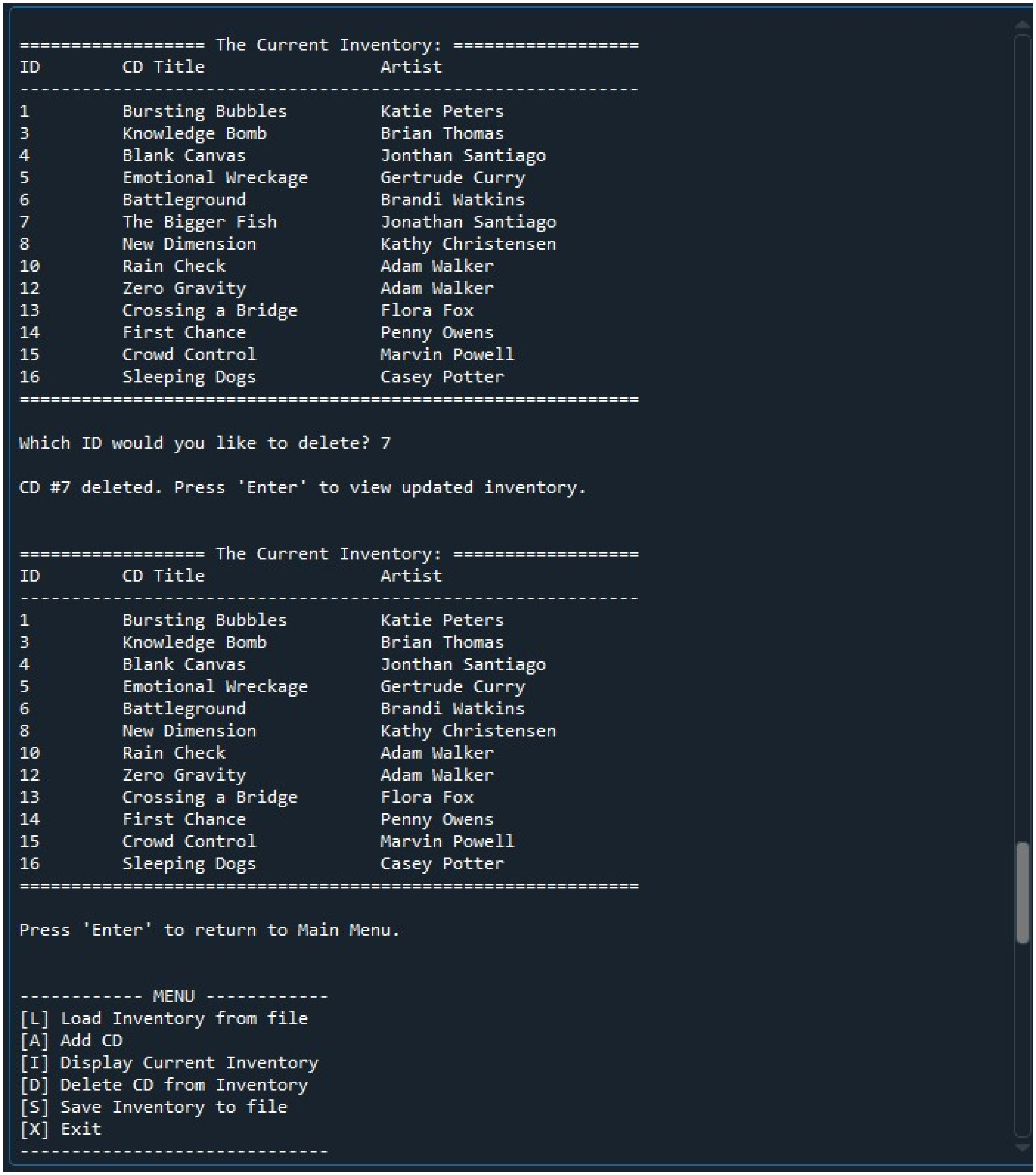
```
In [123]: runfile('C:/_FDPrograming/Mod_06/CDInventory.py', wdir='C:/_FDPrograming/Mod_06')

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

                    The Magic CD Inventory

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


------------ 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                 Artist
------------------------------------------------------------
1         Bursting Bubbles         Katie Peters
3         Knowledge Bomb           Brian Thomas
4         Blank Canvas             Jonthan Santiago
5         Emotional Wreckage       Gertrude Curry
6         Battleground             Brandi Watkins
7         The Bigger Fish          Jonathan Santiago
8         New Dimension            Kathy Christensen
10        Rain Check               Adam Walker
12        Zero Gravity             Adam Walker
13        Crossing a Bridge        Flora Fox
14        First Chance             Penny Owens
15        Crowd Control            Marvin Powell
============================================================

Press 'Enter' to return to Main Menu.


------------ 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
```

*Figure 3 - Running CDInventory.py in Spyder*



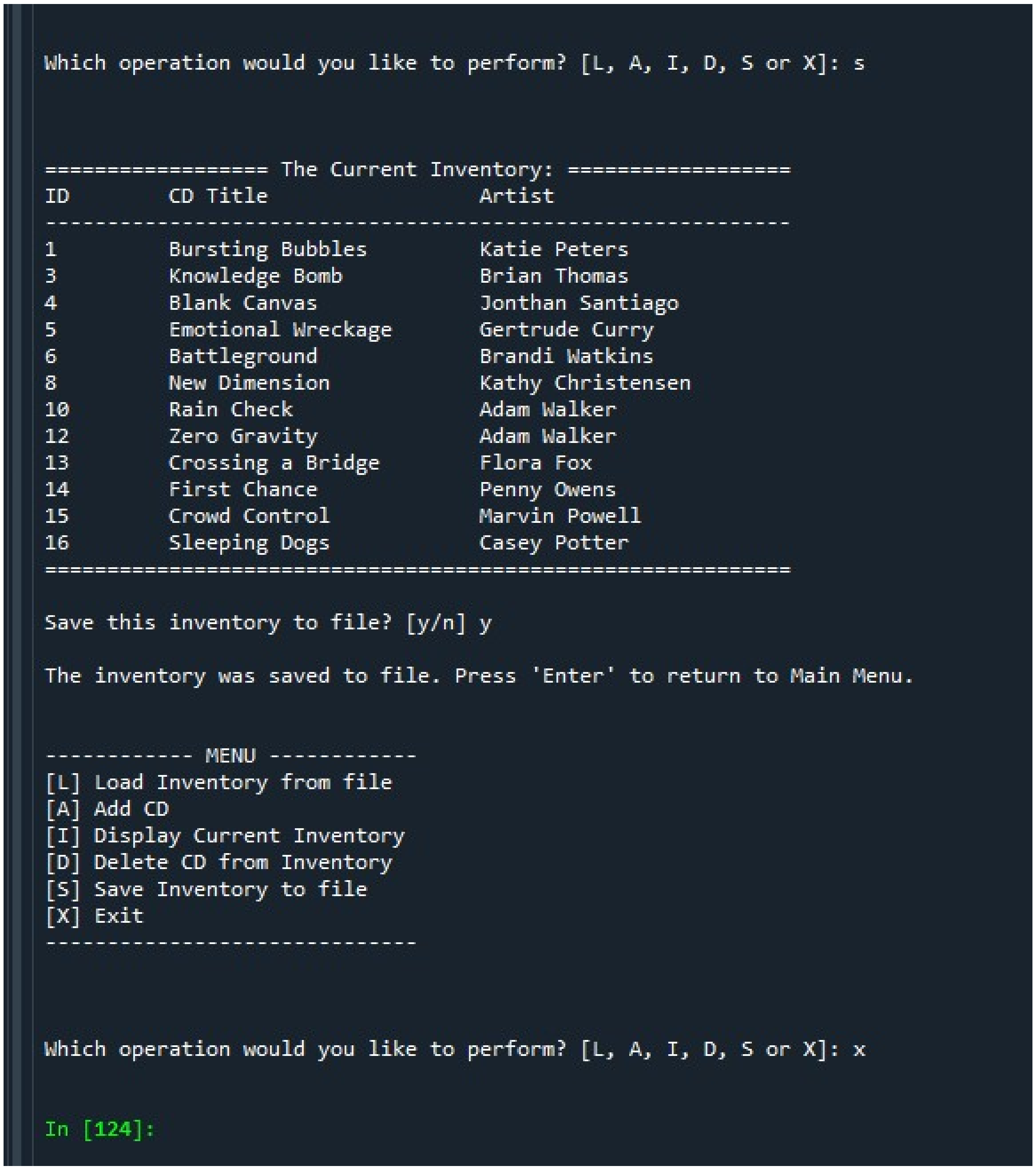
```
Which operation would you like to perform? [L, A, I, D, S or X]: a


What is the CD's title? Sleeping Dogs

What is the Artist's name? Casey Potter


================== The Current Inventory: ==================
ID        CD Title                 Artist
------------------------------------------------------------
1         Bursting Bubbles         Katie Peters
3         Knowledge Bomb           Brian Thomas
4         Blank Canvas             Jonthan Santiago
5         Emotional Wreckage       Gertrude Curry
6         Battleground             Brandi Watkins
7         The Bigger Fish          Jonathan Santiago
8         New Dimension            Kathy Christensen
10        Rain Check               Adam Walker
12        Zero Gravity             Adam Walker
13        Crossing a Bridge        Flora Fox
14        First Chance             Penny Owens
15        Crowd Control            Marvin Powell
16        Sleeping Dogs            Casey Potter
============================================================

CD Added. Press 'Enter' to return to Main Menu.


------------ 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                 Artist
------------------------------------------------------------
1         Bursting Bubbles         Katie Peters
3         Knowledge Bomb           Brian Thomas
4         Blank Canvas             Jonthan Santiago
```

*Figure 4 - Running CDInventory.py in Spyder*



```
================== The Current Inventory: ==================
ID        CD Title                Artist
------------------------------------------------------------
1         Bursting Bubbles        Katie Peters
3         Knowledge Bomb          Brian Thomas
4         Blank Canvas            Jonthan Santiago
5         Emotional Wreckage      Gertrude Curry
6         Battleground            Brandi Watkins
7         The Bigger Fish         Jonathan Santiago
8         New Dimension           Kathy Christensen
10        Rain Check              Adam Walker
12        Zero Gravity            Adam Walker
13        Crossing a Bridge       Flora Fox
14        First Chance            Penny Owens
15        Crowd Control           Marvin Powell
16        Sleeping Dogs           Casey Potter
============================================================

Which ID would you like to delete? 7

CD #7 deleted. Press 'Enter' to view updated inventory.


================== The Current Inventory: ==================
ID        CD Title                Artist
------------------------------------------------------------
1         Bursting Bubbles        Katie Peters
3         Knowledge Bomb          Brian Thomas
4         Blank Canvas            Jonthan Santiago
5         Emotional Wreckage      Gertrude Curry
6         Battleground            Brandi Watkins
8         New Dimension           Kathy Christensen
10        Rain Check              Adam Walker
12        Zero Gravity            Adam Walker
13        Crossing a Bridge       Flora Fox
14        First Chance            Penny Owens
15        Crowd Control           Marvin Powell
16        Sleeping Dogs           Casey Potter
============================================================

Press 'Enter' to return to Main Menu.


------------ 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
------------------------------
```

*Figure 5 - Running CDInventory.py in Spyder*

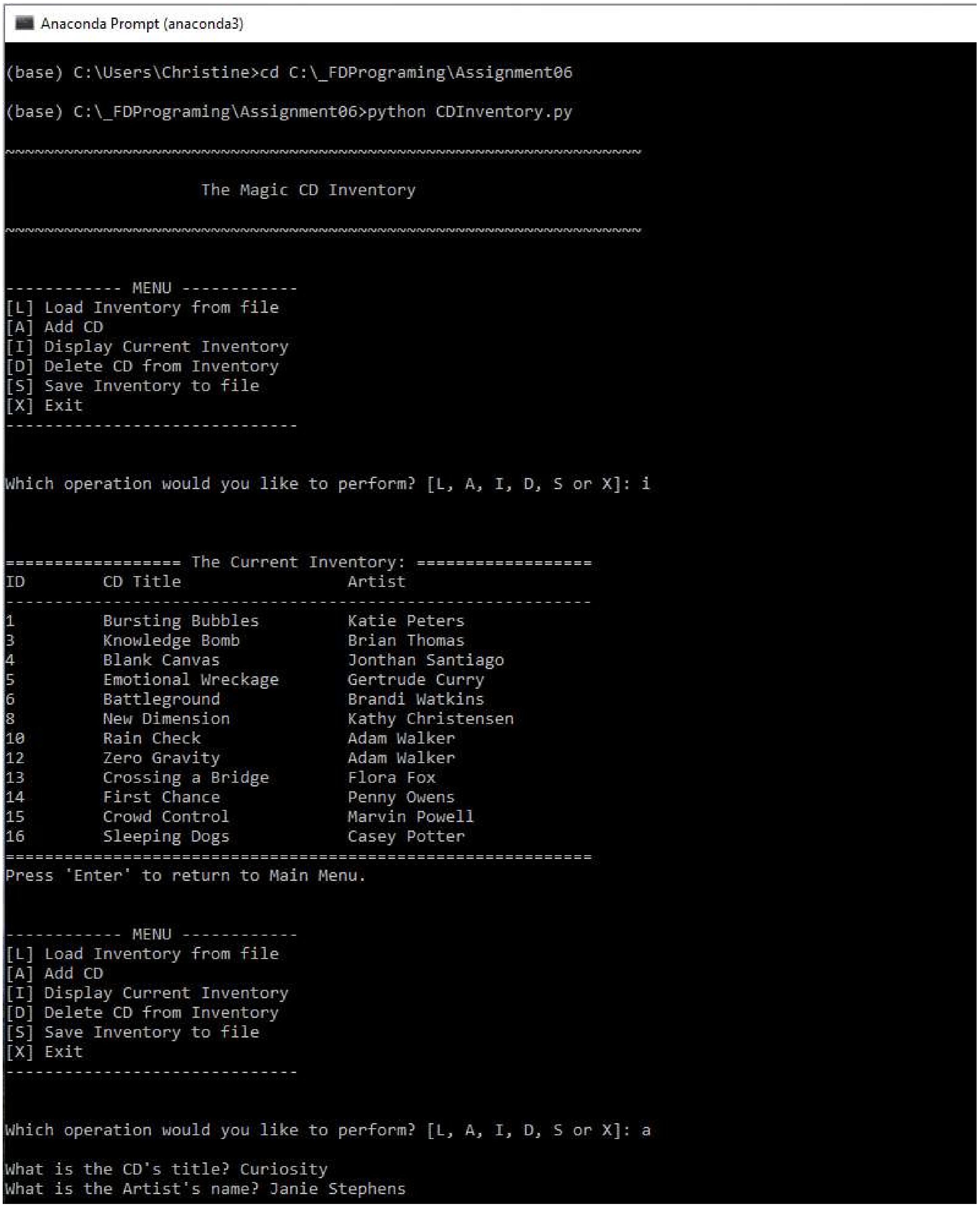


```
Which operation would you like to perform? [L, A, I, D, S or X]: s



================== The Current Inventory: ==================
ID        CD Title                 Artist
------------------------------------------------------------
1         Bursting Bubbles         Katie Peters
3         Knowledge Bomb           Brian Thomas
4         Blank Canvas             Jonthan Santiago
5         Emotional Wreckage       Gertrude Curry
6         Battleground             Brandi Watkins
8         New Dimension            Kathy Christensen
10        Rain Check               Adam Walker
12        Zero Gravity             Adam Walker
13        Crossing a Bridge        Flora Fox
14        First Chance             Penny Owens
15        Crowd Control            Marvin Powell
16        Sleeping Dogs            Casey Potter
============================================================

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

The inventory was saved to file. Press 'Enter' to return to Main Menu.


------------ 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


In [124]:
```

*Figure 6 - Running CDInventory.py in Spyder*

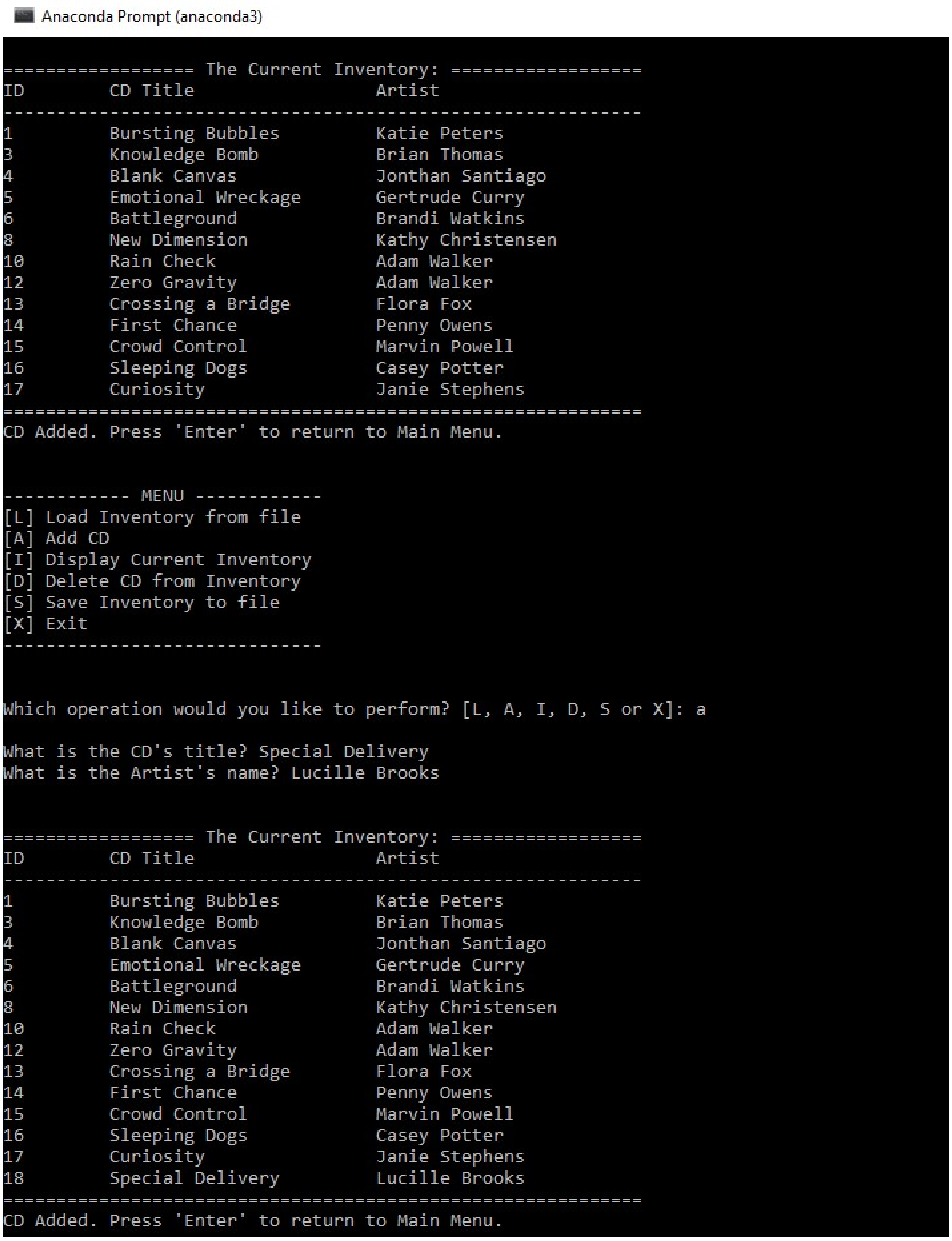
## Running Script in Anaconda Prompt

The program was run successfully in Anaconda Prompt, as shown in Figures 7-10.



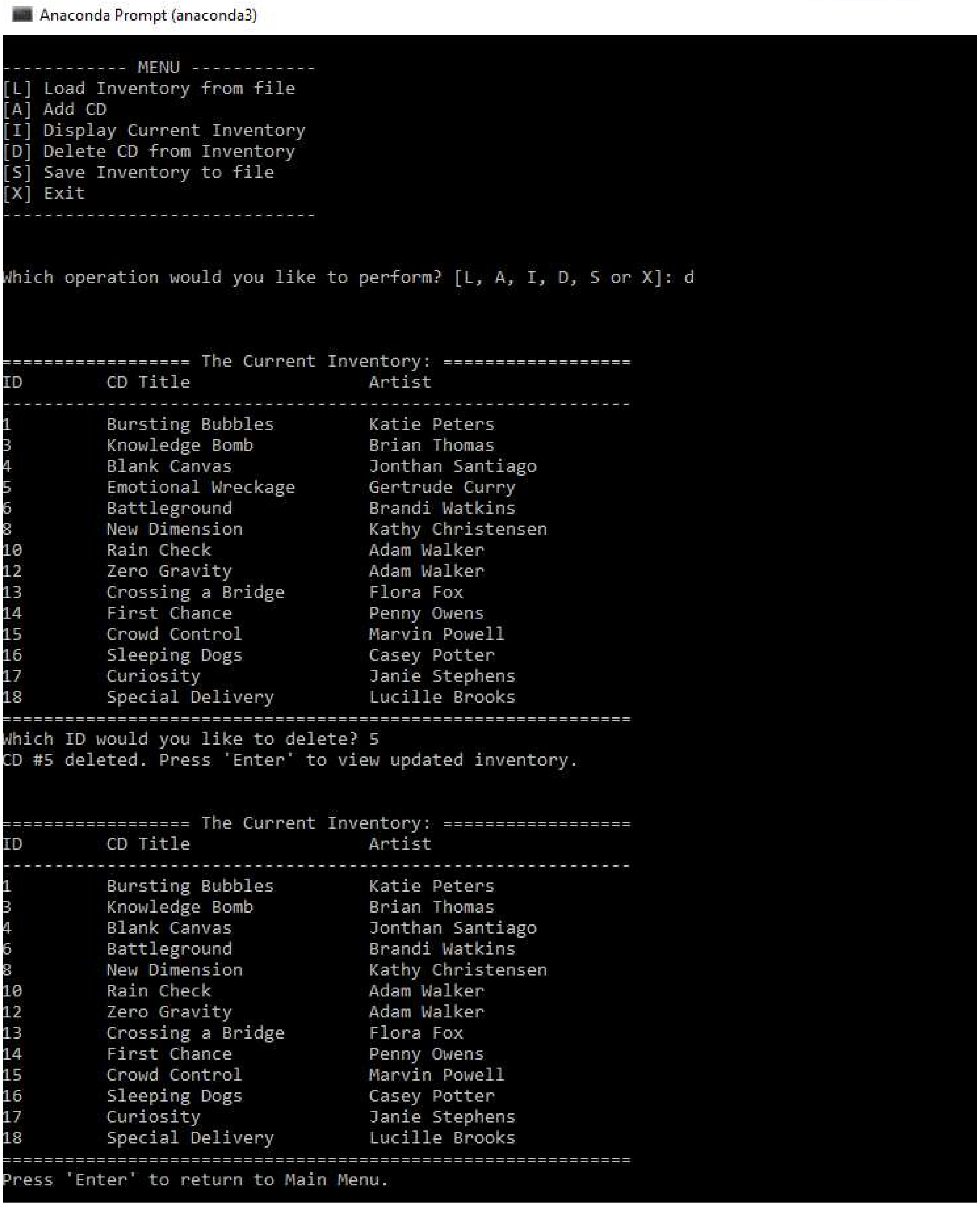
```
(base) C:\Users\Christine>cd C:\_FDPrograming\Assignment06

(base) C:\_FDPrograming\Assignment06>python CDInventory.py

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

                    The Magic CD Inventory

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


------------ 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                Artist
------------------------------------------------------------
1         Bursting Bubbles        Katie Peters
3         Knowledge Bomb          Brian Thomas
4         Blank Canvas            Jonthan Santiago
5         Emotional Wreckage      Gertrude Curry
6         Battleground            Brandi Watkins
8         New Dimension           Kathy Christensen
10        Rain Check              Adam Walker
12        Zero Gravity            Adam Walker
13        Crossing a Bridge       Flora Fox
14        First Chance            Penny Owens
15        Crowd Control           Marvin Powell
16        Sleeping Dogs           Casey Potter
============================================================
Press 'Enter' to return to Main Menu.


------------ 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

What is the CD's title? Curiosity
What is the Artist's name? Janie Stephens
```

*Figure 7 - Running CDInventory.py in Anaconda Prompt*


Anaconda Prompt (anaconda3)

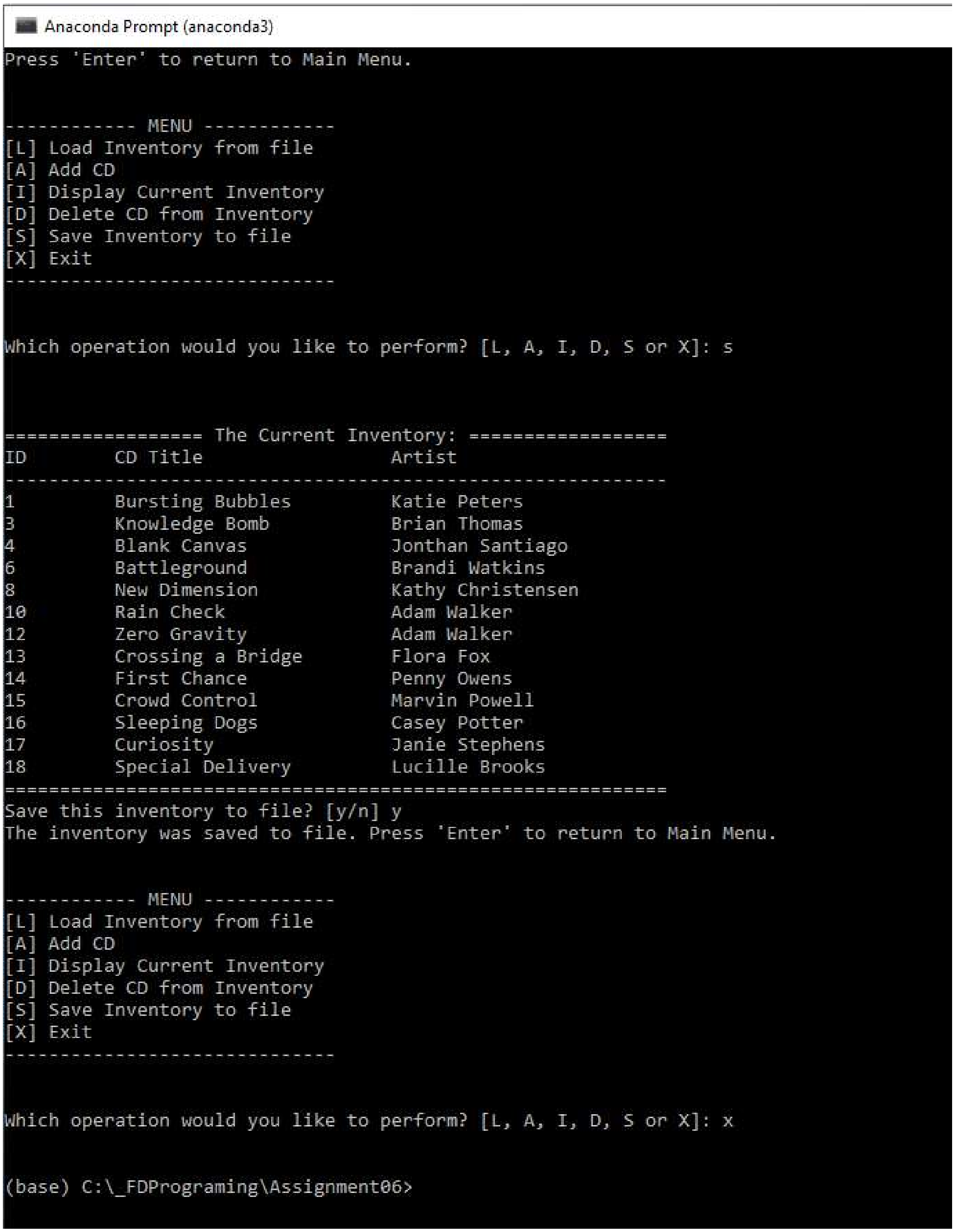
```
================== The Current Inventory: ==================
ID        CD Title                 Artist
------------------------------------------------------------
1         Bursting Bubbles         Katie Peters
3         Knowledge Bomb           Brian Thomas
4         Blank Canvas             Jonthan Santiago
5         Emotional Wreckage       Gertrude Curry
6         Battleground             Brandi Watkins
8         New Dimension            Kathy Christensen
10        Rain Check               Adam Walker
12        Zero Gravity             Adam Walker
13        Crossing a Bridge        Flora Fox
14        First Chance             Penny Owens
15        Crowd Control            Marvin Powell
16        Sleeping Dogs            Casey Potter
17        Curiosity                Janie Stephens
============================================================
CD Added. Press 'Enter' to return to Main Menu.


------------ 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

What is the CD's title? Special Delivery
What is the Artist's name? Lucille Brooks


================== The Current Inventory: ==================
ID        CD Title                 Artist
------------------------------------------------------------
1         Bursting Bubbles         Katie Peters
3         Knowledge Bomb           Brian Thomas
4         Blank Canvas             Jonthan Santiago
5         Emotional Wreckage       Gertrude Curry
6         Battleground             Brandi Watkins
8         New Dimension            Kathy Christensen
10        Rain Check               Adam Walker
12        Zero Gravity             Adam Walker
13        Crossing a Bridge        Flora Fox
14        First Chance             Penny Owens
15        Crowd Control            Marvin Powell
16        Sleeping Dogs            Casey Potter
17        Curiosity                Janie Stephens
18        Special Delivery         Lucille Brooks
============================================================
CD Added. Press 'Enter' to return to Main Menu.
```

*Figure 8 - Running CDInventory.py in Anaconda Prompt*



```
------------ 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                 Artist
------------------------------------------------------------
1         Bursting Bubbles         Katie Peters
3         Knowledge Bomb           Brian Thomas
4         Blank Canvas             Jonthan Santiago
5         Emotional Wreckage       Gertrude Curry
6         Battleground             Brandi Watkins
8         New Dimension            Kathy Christensen
10        Rain Check               Adam Walker
12        Zero Gravity             Adam Walker
13        Crossing a Bridge        Flora Fox
14        First Chance             Penny Owens
15        Crowd Control            Marvin Powell
16        Sleeping Dogs            Casey Potter
17        Curiosity                Janie Stephens
18        Special Delivery         Lucille Brooks
============================================================
Which ID would you like to delete? 5
CD #5 deleted. Press 'Enter' to view updated inventory.


================== The Current Inventory: ==================
ID        CD Title                 Artist
------------------------------------------------------------
1         Bursting Bubbles         Katie Peters
3         Knowledge Bomb           Brian Thomas
4         Blank Canvas             Jonthan Santiago
6         Battleground             Brandi Watkins
8         New Dimension            Kathy Christensen
10        Rain Check               Adam Walker
12        Zero Gravity             Adam Walker
13        Crossing a Bridge        Flora Fox
14        First Chance             Penny Owens
15        Crowd Control            Marvin Powell
16        Sleeping Dogs            Casey Potter
17        Curiosity                Janie Stephens
18        Special Delivery         Lucille Brooks
============================================================
Press 'Enter' to return to Main Menu.
```

*Figure 9 - Running CDInventory.py in Anaconda Prompt*

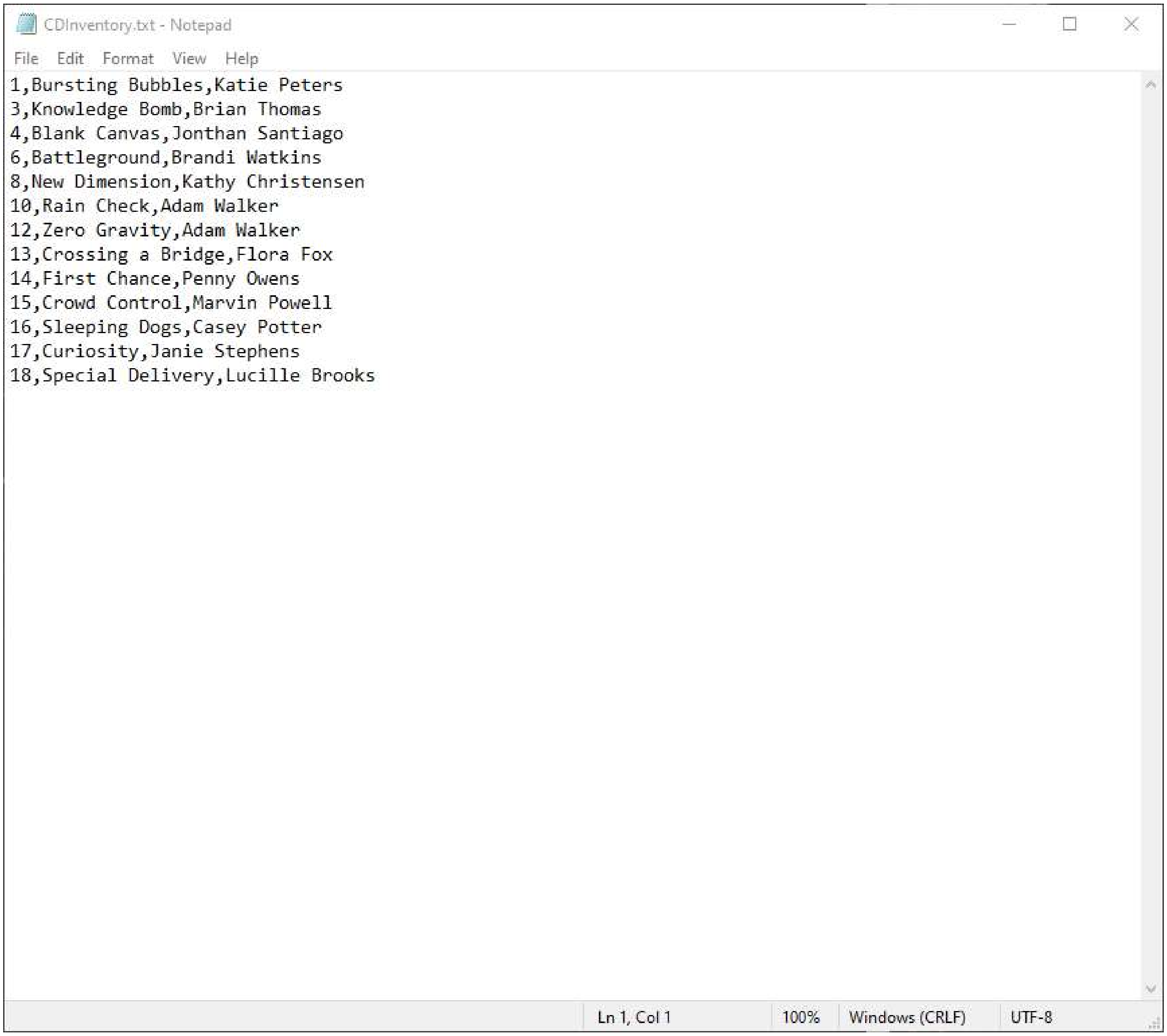

Anaconda Prompt (anaconda3)

```
Press 'Enter' to return to Main Menu.


------------ 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                 Artist
------------------------------------------------------------
1         Bursting Bubbles         Katie Peters
3         Knowledge Bomb           Brian Thomas
4         Blank Canvas             Jonthan Santiago
6         Battleground             Brandi Watkins
8         New Dimension            Kathy Christensen
10        Rain Check               Adam Walker
12        Zero Gravity             Adam Walker
13        Crossing a Bridge        Flora Fox
14        First Chance             Penny Owens
15        Crowd Control            Marvin Powell
16        Sleeping Dogs            Casey Potter
17        Curiosity                Janie Stephens
18        Special Delivery         Lucille Brooks
============================================================
Save this inventory to file? [y/n] y
The inventory was saved to file. Press 'Enter' to return to Main Menu.


------------ 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) C:\_FDPrograming\Assignment06>
```

*Figure 10 - Running CDInventory.py in Anaconda Prompt*

## Text File

The text file from the two runs of the script above is shown in Figure 11. Note that although there are gaps between numbers (as to be expected based on how I wrote my script), the numbers are all unique and increasing as they go down the list.



```
1,Bursting Bubbles,Katie Peters
3,Knowledge Bomb,Brian Thomas
4,Blank Canvas,Jonthan Santiago
6,Battleground,Brandi Watkins
8,New Dimension,Kathy Christensen
10,Rain Check,Adam Walker
12,Zero Gravity,Adam Walker
13,Crossing a Bridge,Flora Fox
14,First Chance,Penny Owens
15,Crowd Control,Marvin Powell
16,Sleeping Dogs,Casey Potter
17,Curiosity,Janie Stephens
18,Special Delivery,Lucille Brooks
```

*Figure 11 - CDInventory.txt data file*

## Summary

Overall, this coding project was not particularly difficult. The most challenging aspect was determining if I was using "best practices" when it came to issues involving shadow variables and changing global entities within a function. It was a challenging question to answer since the program did not generate errors either way! It was nice to hear other students confirm similar thinking patterns during Office Hours.

## Appendix

### Listing CDInventory.py

```python
#------------------------------------------#
# Title: CDInventory.py
# Desc: Working with classes and functions.
# Change Log: (Who, When, What)
# DBiesinger, 2030-Jan-01, Created File
# CBuffalow, 2021-Feb-19, Added functions (process_cd_data, process_delete, write_file,
#    get_cd_data), added uppercase functionality to menu for better readability,
#    adjusted formatting, added program header
# Cbuffalow, 2021-Feb-20, Added calculcate_cd_id function, adjusted
#    process_cd_data & get_cd_data functions, revised comments
#------------------------------------------#

# -- MODULES -- #
import os.path

# -- DATA -- #
#Global Variables
strChoice = '' # user input (string)
strYesNo = '' # user input for yes/no question (string)
indID = -1 # ID number of CD (integer)
strTitle = '' # user input (string)
strArtist = '' # user input (string)
lstTbl = []  # table to hold data (list of dictionaries)
dicRow = {}  # row of data (dictionary)
strFileName = 'CDInventory.txt'  # data storage file
intIDDel = '' # user selected ID to delete (integer)
intRowNr = '' # index of row to be deleted (integer)
blnCDRemoved = "False" # flag if desired CD found (Boolean)


# -- PROCESSING -- #
class DataProcessor:
    """Processing data within program's current runtime."""

    @staticmethod
    def process_cd_data(cd_id, cd_title, cd_artist):
        """Function that takes user input about CD and formats data into a dictionary.

        Keys are "ID", 'Title', and 'Artist' for each dictionary and the values are assigned
        from cd_id, cd_title, and cd_artist, respectively.

        Args:
            cd_id (int): ID of CD, generated by program
            cd_title (str): name of CD, provided by user
            cd_artist (str): name of artist, provided by user

        Returns:
            cd_info (dictionary): cd_id, cd_title, and cd_artist contained within a single dictionary

        """
        cd_info = {'ID': cd_id, 'Title': cd_title, 'Artist': cd_artist}
        return cd_info



    @staticmethod
    def process_delete(cd_to_delete, table):
        """Function that processes which row index needs to be deleted based upon input from user.

        For each dictionary, the function determines if the key:value pair in that row/dictionary
        with the key of "ID" has the desired ID number for its value.  If yes, row_number stops counting
        at that row and CD_found is set to 'True'.

        Args:
            cd_to_delete (int): ID number of CD that user wants to delete
            table (list of dicts): 2D data structure (list of dicts) that holds data during runtime

        Returns:
            row_number (int): counter of rows (stops on row if desired CD found, otherwise returns index of last row)
            CD_found (Boolean): flag that states if desired CD found (True = yes, False = no)
        """
        row_number = -1
        CD_found = False
        for row in table:
            row_number += 1
            if row['ID'] == cd_to_delete:
                CD_found = True
                break
        return row_number, CD_found



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

        Function checks to make sure specified txt file exists.  If yes, continues by reading
        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.
        """
              if os.path.exists(file_name):
                  objFile = None  # file object
                  table.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(',')
                      cd_info = {'ID': int(data[0]), 'Title': data[1], 'Artist': data[2]}
                      table.append(cd_info)
                  objFile.close()
              else: pass



          @staticmethod
          def calculate_cd_id(file_name):
              """Function to determine last ID used in CDInventory.txt so that next ID number can be assigned

              The function finds the last row of data in CDInventory.txt and reports
              the ID number used in that row.  This becomes the start point in which the rest of the ID numbers
              are based upon.

              Args:
                  file_name (str): name of file used to read data from

              Returns:
                  cd_id (int): id number of last used CD_ID in the CDInventory.txt file
              """
              if os.path.exists(file_name):
                  objFile = None  # file object
                  objFile = open(file_name, 'r')
                  objFile_content = objFile.read() #returns all content of file as a giant string
                  objFile.close()
                  objFile_rows = objFile_content.split(sep= '\n') #separates data into separate items based on location of \n
                  indexFinalRow = len(objFile_rows) -2
                  cd_id = int(objFile_rows[indexFinalRow].split(sep= ',')[0])
              else:
                  cd_id = 0
              return cd_id



          @staticmethod
          def write_file(file_name, table):
              """Function to overwrite data from current runtime into a text file.

              For each row, the data is converted into a string with a comma separating
              each piece of data and then a new line is started.  The data is then saved
              out to the designated txt file.

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

              Returns:
                  None.
              """
              objFile = None  # file object
              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) -- #

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

          @staticmethod
          def print_menu():
              """Displays a menu of choices to the user.


              Args:
                  None.

              Returns:
                  None.
              """
              print('\n')
              print(' MENU '.center(30,'-'))
              print('[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')
              print('-'*30)
              print('\n')



          @staticmethod
          def menu_choice():
              """Gets user input for menu selection.


              Args:
                  None.

              Returns:
                  choice (string): an upper 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', '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('\n')
              print(' The Current Inventory: '.center(60,'='))
              print('{:<10}{:<25}{:<25}'.format('ID', 'CD Title', 'Artist'))
              print('-'*60)
              for row in table:
                  print('{:<10}{:<25}{:<25}'.format(*row.values()))
              print('='*60)



          @staticmethod
          def get_cd_data():
              """Collects information about CD from user.

              Asks user to input the name of the CD and the artist of the CD.

              Args:
                  None

              Returns:
                  cd_title (str): name of CD, provided by user
                  cd_artist (str): name of artist, provided by user

              """
              cd_title = input('What is the CD\'s title? ').strip()
              cd_artist = input('What is the Artist\'s name? ').strip()
              return(cd_title, cd_artist)




      # 1. When program starts, read in the currently saved inventory, print program header
      FileProcessor.read_file(strFileName, lstTbl)
      intID = FileProcessor.calculate_cd_id(strFileName)
      print('\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n')
      print('The Magic CD Inventory'.center(62))
      print('\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')

      # 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 will be reloaded from file.')
              strYesNo = input('Type \'yes\' to continue and reload from file - otherwise reload will be cancelled. ')
              if strYesNo.lower() == 'yes':
                  print('Reloading...')
                  FileProcessor.read_file(strFileName, lstTbl)
                  IO.show_inventory(lstTbl)
                  input('Inventory Loaded. Press \'Enter\' to return to Main Menu. ')
              else:
                  input('Cancelling... Inventory data NOT reloaded. Press [ENTER] to return to the menu. ')
                  IO.show_inventory(lstTbl)
              continue  # start loop back at top.
          # 3.3 process add a CD
          elif strChoice == 'a':
              # 3.3.1 Generate ID and Ask user for CD Title and Artist
              intID += 1
              strTitle, strArtist = IO.get_cd_data()
              # 3.3.2 Add item to the table
              dicRow = DataProcessor.process_cd_data(intID, strTitle, strArtist)
              lstTbl.append(dicRow)
              IO.show_inventory(lstTbl)
              input('CD Added. Press \'Enter\' to return to Main Menu. ')
              continue  # start loop back at top.
          # 3.4 process display current inventory
          elif strChoice == 'i':
              IO.show_inventory(lstTbl)
              input('Press \'Enter\' to return to Main Menu. ')
              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(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
```

```
                intRowNr, blnCDRemoved = DataProcessor.process_delete(intIDDel, lstTbl)
                if blnCDRemoved:
                    del lstTbl[intRowNr]
                    input('CD #' + str(intIDDel) + ' deleted. Press \'Enter\' to view updated inventory.')
                else:
                    input('CD #' + str(intIDDel) + ' not found.  Press \'Enter\' view inventory.')
                IO.show_inventory(lstTbl)
                input('Press \'Enter\' to return to Main Menu. ')
                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(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
                    FileProcessor.write_file(strFileName, lstTbl)
                    input('The inventory was saved to file. Press \'Enter\' to return to Main Menu. ')

                else:
                    input('The inventory was NOT saved to file. Press [ENTER] to return to Main 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 safe:
            else:
                print('General Error')
```