Andrew Yeo
March 4, 2020
Foundations of Programming: Python
Github Repository: https://github.com/andyhyeo/Assignment_06

Module 07

Knowledge Document

Intro

In this week's assignment, we go over working with binary files as a substitute with working with text files. Binary files is a way to store and transfer data that saves space. Binary files are made by use of a python feature called "pickling", whereby the data is "pickled" or "preserved" for later use, much in the similar sense as pickling vegetables. This way of storing data makes it difficult for the user to read in its raw pickled form, so it must be "unpickled" as it were, by the user in order to make it readable by the user.

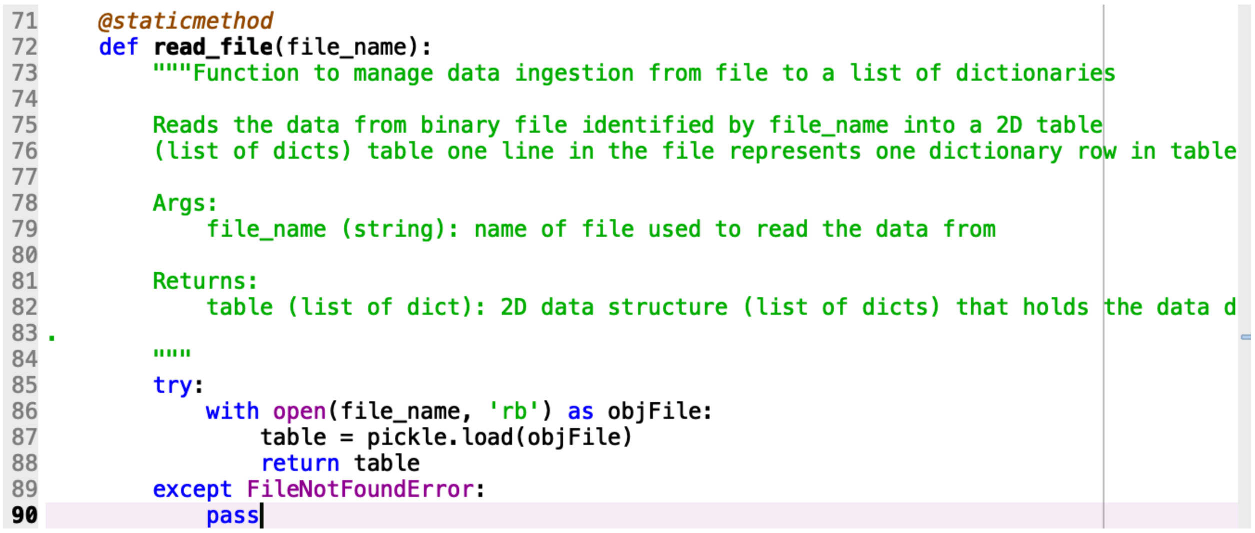
We also go over structured error handling. Structured error handling redirects an error condition into a named object that can be more than just a flag and a subsequent message. It supplies structure by redirecting code to be made associated with supplementary error-handling code paths (catch block).

Details

In this week's assignment we are instructed to go over the CD Inventory script that we have been developing in order to amend the parts to reflect what we learned about the structured error handling and pickling where we can with in our script. We add specifically in areas where there are user interaction, type acting (string to int) or file access operations.

Substituting Pickle and Try-Except in the Read_File Function

Figure 1, shows the substituted pickling in the read_file() function. The original code was a for looped that appended every row that was read into a table that we defined in the script. I was able to successfully retrofit the try-except structure by watching the videos and trying to emulate what the professor did. I had to write the script so that it produced the built-in error that I was trying to call and I finally was able to find the FileNotFound error.

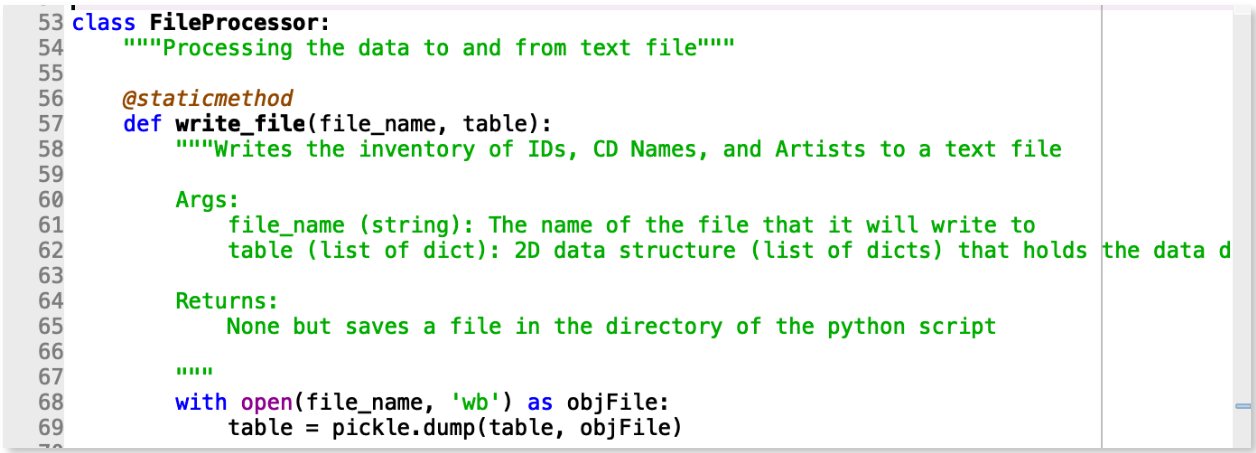


```
    @staticmethod
    def read_file(file_name):
        """Function to manage data ingestion from file to a list of dictionaries

        Reads the data from binary 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

        Returns:
            table (list of dict): 2D data structure (list of dicts) that holds the data d
.
        """
        try:
            with open(file_name, 'rb') as objFile:
                table = pickle.load(objFile)
                return table
        except FileNotFoundError:
            pass
```

Figure 1: Substituting Pickle in the Read File Function

Substituting Pickle in the Write_File Function

Similarly rewriting the code for the write_file () function was done by watching the Dirk's youtube video and trial-and-erroring the code. Note that both in figure 1 and 2, I was able to successfully to call the open() function using the with-as structure.

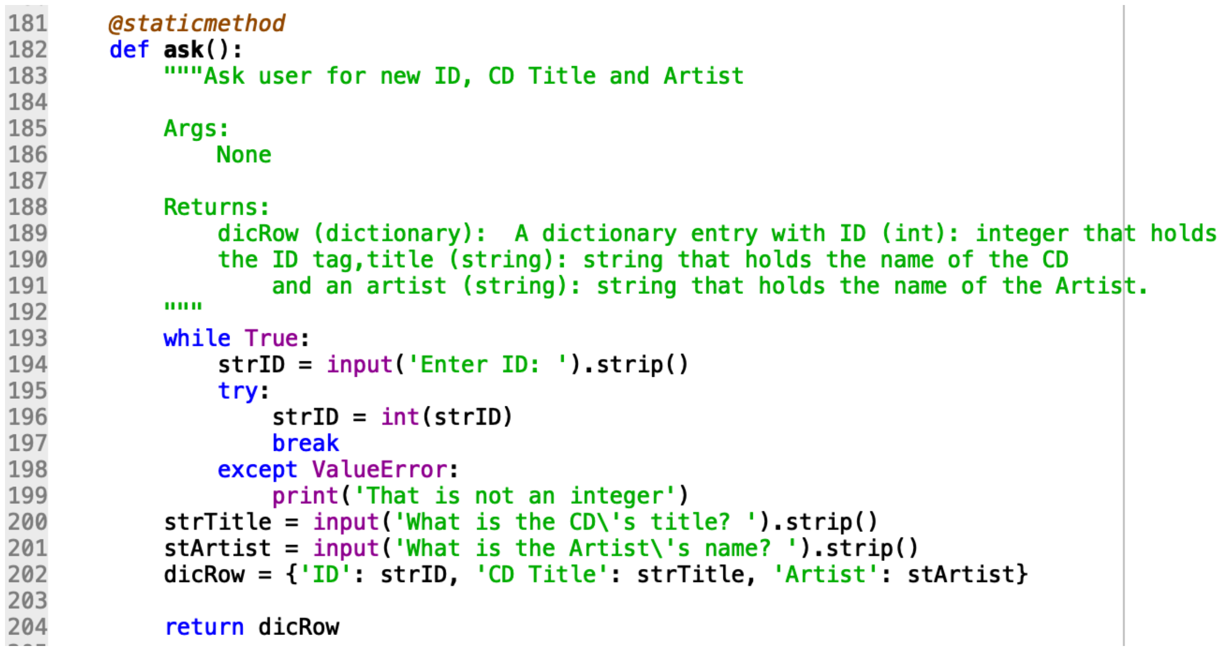


```
class FileProcessor:
    """Processing the data to and from text file"""

    @staticmethod
    def write_file(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 d

        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 2: Substituting Pickle in the Write File Function

Try-Except for ValueError in IO.ask() functionThe IO.add_cd() 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.

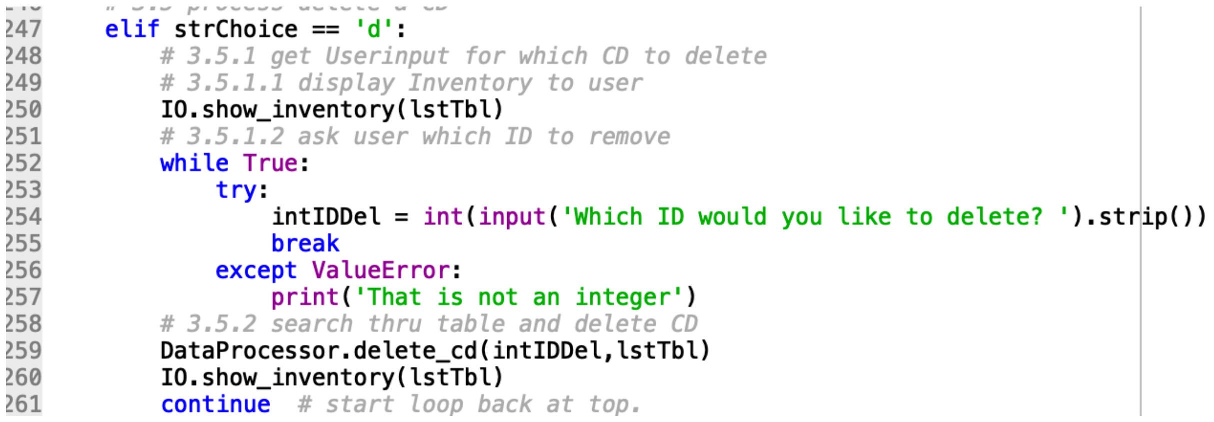


```
    @staticmethod
    def ask():
        """Ask user for new ID, CD Title and Artist

        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 3: Substituting Pickle in the Write File Function

Error Structure Handling to handle not-integers when deleting

Figure 4 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.



```
    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

Summary

We go over pickling and binary data storing methods. We also go over using try-except structures as a way to handle errors in a structured way. Error structure handling is useful in order to prevent the script from unintentionally crashing. We adding pickling and error structure handling to the script. Further investigations include successfully converting the FileProcessor.append() function to add binary rows to the inventory and add structured error handling to the IO.menu_choice() function. Also, a subject of further investigation includes creating custom error classes and not just using the built-in python ones.

References

https://www.youtube.com/watch?v=nfDuGaTmuj8&feature=youtu.be
https://www.youtube.com/watch?v=IZNHBvmc7nA&feature=youtu.be
https://www.youtube.com/watch?v=Wb0XsWqaj4s&feature=youtu.be
https://www.youtube.com/watch?v=Wb0XsWqaj4s&feature=youtu.be
https://www.youtube.com/watch?v=XXc2dJ3r018&feature=youtu.be

Appendix: The Code

Made using Planet B's Syntax Highlighter

```python
#------------------------------------------#
# Title: CDInventory.py
# Desc: Working with classes and functions.
# Change Log: (Andrew Yeo, 02-25-2020, Group features into functions)
# AYeo, 2020-Feb-25, Created File
# Ayeo, 2020-Mar-03, Modified the permanent data store to use binary data.
# Ayeo, 2020-Mar-03, Subbed Try-Except for FileNotFoundError in FileProcessor.read_file() function.
# Ayeo, 2020-Mar-03, Subbed Try-Except for ValueError in IO.ask() function.
# AYeo, 2020-Mar-03, Added Error Structure Handling tto handle not-integers when prompted to delete
# AYeo, 2020-Mar-03, Tried convert to FileProcessor.append() to pickling unsuccessfully
# AYeo, 2020-Mar-03, Could try adding structure handling to IO.menu_choice

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

import pickle

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


# -- PROCESSING -- #
class DataProcessor:
    @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 row in table:
            intRowNr += 1
            if row['ID'] == intIDDel:
                del table[intRowNr]
                blnCDRemoved = True
                break
            if blnCDRemoved:
                print('The CD was removed')
            else:
                print('Could not find this CD!')
        return table


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

    @staticmethod
    def write_file(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)

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

        Reads the data from binary 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

        Returns:
            table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
.
        """
        try:
            with open(file_name, 'rb') as objFile:
                table = pickle.load(objFile)
                return table
        except FileNotFoundError:
            pass


# -- 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:
            choice (string): a lower case sting of the users input out of the choices l, a, i, d, s or x

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


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


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

        Returns:
            None.

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

    @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

##  Attempt to add pickling to append function
#    @staticmethod
#    def add_cd(row,file_name):
#        """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.
#
#        """
#        # TODO Add Pickling
#        with open(file_name, 'ab') as objFile:
#            pickle.dump(row, objFile)


    @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
```

```python
                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


# 1. When program starts, read in the currently saved Inventory
#objFile = open(file_name, 'w') #Commenting out
FileProcessor.read_file(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 from file. otherwise reload will be canceled')
        if strYesNo.lower() == 'yes':
            print('reloading...')
            lstTbl = FileProcessor.read_file(strFileName)
            IO.show_inventory(lstTbl)
        else:
            input('canceling... Inventory data NOT reloaded. Press [ENTER] to continue 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 Ask user for new ID, CD Title and Artist
        dicRow = IO.ask()
        # 3.3.2 Add item to the table
        IO.add_cd(dicRow,lstTbl)
        IO.show_inventory(lstTbl)
        continue  # start loop back at top.
    # 3.4 process display current inventory
    elif strChoice == 'i':
        IO.show_inventory(lstTbl)
        continue  # start loop back at top.
    # 3.5 process delete a CD
    elif strChoice == 'd':
        # 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.
    # 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)
        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_07/Assignment07/CDInventory.py', wdir='/Users/andyyeo/Google Drive/Intro to Python/Mod_07/Assignment07')
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? Brah Tah

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

1       Brah Tah (by:Tah Tah Tah)
======================================
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       Brah Tah (by:Tah Tah Tah)
```

```
======================================
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       Brah Tah (by:Tah Tah Tah)
======================================

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       Brah Tah (by:Tah Tah Tah)
======================================

Which ID would you like to delete? 1
======= 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       Brah Tah (by:Tah Tah Tah)
======================================
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: Tue Feb 25 15:46:07 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:~ andyyeo$ cd desktop/assignment06
(base) Andys-MacBook-Pro:assignment06 andyyeo$ zip -r dir.zip . -x ".*" -x "_MAXOSX"
 (stored 0%)on
  adding: Assignment06 Knowledge Document.pdf (deflated 30%)
  adding: CDInventory.py (deflated 71%)
(base) Andys-MacBook-Pro:assignment06 andyyeo$
  [Restored Mar 4, 2020 at 11:10:28 AM]
## cd /Users/andyyeo/desktop/assignment06 ##
Last login: Tue Feb 25 17:51:19 on ttys000
Restored session: Tue Feb 25 17:51:49 PST 2020

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$ clear



(base) Andys-MacBook-Pro:~ andyyeo$ cd desktop/assignment07
(base) Andys-MacBook-Pro:assignment07 andyyeo$ python CDInventory.py
Menu

[l] load Inventory from file
[a] Add CD
[i] Display Current Inventory
[d] delete CD from Inventory
[s] Save Inventory to file
[x] exit

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

Enter ID: 1
```

```
What is the CD's title? Ska Bangers
What is the Artist's name? Reel Fig Bish
======= The Current Inventory: =======
ID  CD Title (by: Artist)

1   Ska Bangers (by:Reel Fig Bish)
======================================
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   Ska Bangers (by:Reel Fig Bish)
======================================
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   Ska Bangers (by:Reel Fig Bish)
======================================
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   Ska Bangers (by:Reel Fig Bish)
======================================
Which ID would you like to delete? 1
======= 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   Ska Bangers (by:Reel Fig Bish)
======================================
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:assignment07 andyyeo$

```