Christopher Manzuk

14 Mar 2022

Foundations of Programming Python

Assignment 7

Lesson Seven – Binary Files & Error Handling

Introduction

Lesson Seven discusses structured error handling, and details on text file read/writes, and binary file handling with *pickle* and *shelve*.

CD Inventory.py

The task is to upgrade a version of our CD Inventory program to a class/function structure. Most main procedure code was moved into functions within pre-defined classes, and full documentation ('docstrings') was added to each function. An operating code from Assignment 5 was supplied. The final script itself, with trimmed documentation, is shown below:

"CDInventory.py"

```
# -- DATA -- #
strChoice = '' # User input
lstTbl = []  # list of dicts to hold data
dicRow = {}  # dict of data row
strFileName = 'CDInventory'  # data storage filename root
strFileNameTxt = strFileName+'.txt'  # data storage text
strFileNameBin = strFileName+'.bin'  # data storage binary
objFile = None  # file object

# -- PROCESSING -- #

class DataProcessor:

  @staticmethod
  def addCD_LineItem (newRow, table):          # imput parameters as lists
    """Function to append new row of data to existing table
    Makes needed data type conversion, creates dict row from the input list,
        appends to existing table
        """

        # ASG7 - Add Error trap

        try:
            dicRow = {'ID': newRow[0], 'Title': newRow[1], 'Artist': newRow[2]}
        except:
            print("\n\tCheck Data Types\n")

        table.append(dicRow)
        return table


    @staticmethod
    def delCD_LineItem (IDselect, table):    # input parameters  int, list
        """Function to delete a selected row of data from existing table

        Takes user's ID selection, converts to list index number, identifies
        the corresponding row in the table, and deletes it.
        """

        intRowNr = -1
        blnCDRemoved = False

        for row in table:
            intRowNr += 1
            if row['ID'] == IDselect:
                del table[intRowNr]
                blnCDRemoved = True
                break

        if blnCDRemoved:
            print('The CD was removed\n')
        else:
            print('Could not find this CD!')
        return table


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

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

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

        Initial read at startup is an existing text file; Later saves and reads
        accomplished as binary. Selected by passing flag in function call from Main
        """
        table.clear()  # this clears existing data to allow clean data load from file

    # ASG7 - Add Error trap
    # ASG7 - convert to reading binary >>> added bin capability rather than replace

        try:
            if binFlag == True                      # execute read for binary file
                objFile = open(file_name , 'rb')
                with open(strFileNameBin , 'rb')  as objFile:
                    table = pickle.load(objFile)
                objFile.close()
                print("\n\tLoaded " + file_name+" from binary\n")
            else:                                                   # execute read for text file
                objFile = open(file_name, 'r')
                for line in objFile:
                    data = line.strip().split(',')
                    dicRow = {'ID': int(data[0]), 'Title': data[1], 'Artist': data[2]}
                    table.append(dicRow)
                objFile.close()
                print("\n\tLoaded " + file_name+" as text")

        except TypeError:
            print("ERROR - Binary or Text file unclear - returning to main menu")
        except IOError:                  # covers both text & bin files
            print("ERROR - File doesn't exist, or points to wrong directory")
        return table


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

        Writes the data from a 2D table (list of dicts) into a BINARY file
        identified by file_name; one line in the file represents one dictionary
        row in table.
        """

    # ASG7 - convert to writing binary
    # ASG7 - Add Error trap

        print("Saving to " + file_name + " as BINARY")
        objFile = open(file_name, 'wb')
        pickle.dump(table, objFile)
        objFile.close()
        print("Saved")


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


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

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

    print("""
        \n\nMenu\n\n
        [l] Load Inventory from file\n
        [i] Display Current Inventory\n
        [a] Add CD\n
        [d] Delete CD from Inventory\n
        [s] Save Inventory to binary file\n
        [x] eXit\n
          """)


@staticmethod
def menu_choice():
    """Gets user input for Main Menu selection
    """

    choice = ' '

  # ASG7 - Add Error trap

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

    return choice


@staticmethod
def InputCD_LineItem():
    """Function to collect new line item data

    Takes input strings and returns a list of the strings
    """

    # ASG7 - Add Error trap

    try:
        intID = int(input('Enter ID: ').strip())        # moved int conversion from function
        strTitle = input('What is the CD\'s title? ').strip()
        stArtist = input('What is the Artist\'s name? ').strip()
        print()
    except (TypeError, ValueError):
        print("\n\tNumbers for IDs, strings for CD/Artist")

    return[intID,strTitle,stArtist]     # return a List
    @staticmethod
    def show_inventory(table):
        """Displays current inventory table
        """

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


## M A I N
# 1. When program starts, read in the currently saved Inventory
FileProcessor.read_file(strFileNameTxt, lstTbl, False)


# 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 \'y\' to continue and reload from file. otherwise reload will be canceled :')

        if strYesNo.lower() == 'y':
            print('\nreloading...\n')
            lstTbl=FileProcessor.read_file(strFileNameBin, lstTbl, True)
        else:
            input('\ncanceling... Inventory data NOT reloaded. Press [ENTER] to continue to the menu.\n')

        print("\n\t\t Updated Table - for verification\n")
        IO.show_inventory(lstTbl)      # trying this - more efficient

        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
    lstNewRow=[]
    lstNewRow = IO.InputCD_LineItem()

    # 3.3.2 Add item to the table
    lstTbl = DataProcessor.addCD_LineItem(lstNewRow,lstTbl)
    print()
    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
    # ASG7 - Add Error trap
    try:
       intIDdel = int(input('\nWhich ID would you like to delete? \n').strip() )
    except (TypeError, ValueError):
       print("\n\tERROR - Numbers for IDs - back to main menu")
       continue       # on error skip function call, restart menu

    # 3.5.2 search thru table and delete CD
    lstTbl = DataProcessor.delCD_LineItem (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('\nSave this inventory to BINARY file? [y/n] \n').strip().lower()

    # 3.6.2 Process choice
    if strYesNo == 'y':
            # 3.6.2.1 save data
            print("standby...")
            FileProcessor.write_file(strFileNameBin, lstTbl)

        elif strYesNo == 'n':
            input('\nThe inventory was NOT saved to file. Press [ENTER] to return to the menu.\n')
        else:
           input('\nChoose either y or n. Returning to Main Menu\n')

        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('\nGeneral Error')
```

==============================

Adding error handling was modestly challenging, if only in deciding how much to do. It would be easy to go overboard trying to completely lock down every possible error at the suggested points. The try/except statements were easy enough, but determining what the program flow from exceptions should be led to several false starts.

A decision was also made to maintain the capability to read a text file, since the program reads the file at startup, and the intial available copy is text. There was no need to do thst for the write function, as after the initial read, everything would be saved and opened as binary.

An attempt was made to cue a text vs binary choice for the Read function based on a TypeError trap. The idea was to make the function call assuming the file to read is saved as text, and if binary, a TypeError would be triggered, and the 'binary read' code block would be under the exception. This got a bit complex, needing to add and subtract .xxx file extensions a few times, among other complications. The whole idea was jettisoned in favor of adding a 'bin/txt' flag to the function call, which simplified thingds considerably.

There was a maddening difficulty reading in the binary file. After getting past a slew of errors from basic syntax errors, the read would appear to work, but the table (via 'Display Inventory') would be empty. It was eventually realized that the 'Read File' function call was made by itself, without being assigned to the table variable, ie: was: "FileProcessor.read_file(strFileNameBin, lstTbl, True)" >> now: "lstTbl=FileProcessor.read_file(strFileNameBin, lstTbl, True)."

CD Inventory – IDE Run

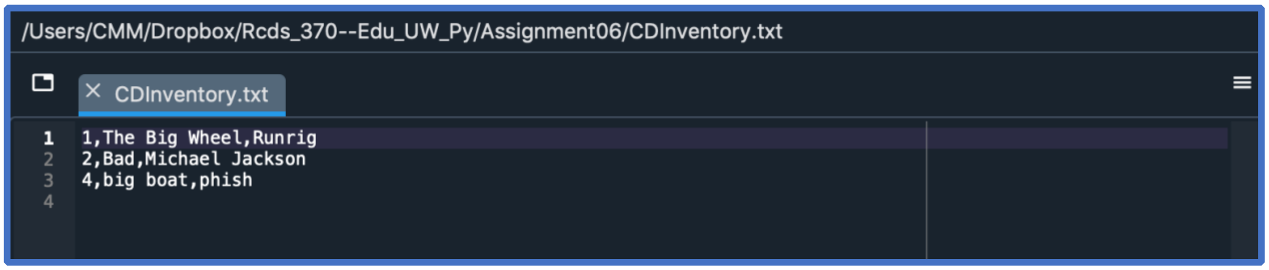
Running the code in the spyder IDE worked without incident, as shown in the console text stream below. Red text indicates comments describing which functions are being tested or demonstrated. Examples of the output file are shown in Figs 1 ("CDInventory.txt") & 2 ("CDInventory.bin").

Console Output:

```
Python 3.9.7 (default, Sep 16 2021, 08:50:36)
Type "copyright", "credits" or "license" for more information.
IPython 7.29.0 -- An enhanced Interactive Python.

In [1]: runcell(0, '/Users/CMM/Dropbox/Rcds_370--
Edu_UW_Py/Assignment07/ CDInventory.py')

   Loaded CDInventory.txt as text

Menu
         [l] Load Inventory from file
         [i] Display Current Inventory
         [a] Add CD
         [d] Delete CD from Inventory
         [s] Save Inventory to binary file
         [x] eXit
Which operation would you like to perform? [l, a, i, d, s or x]: w

 ## tested error trap with out of range selection

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

 ## tested Display Function w/added error trap code

======= The Current Inventory: =======
ID CDTitle(by:Artist)
1  The Big Wheel (by: Runrig)
2  Bad (by: Michael Jackson)
4 big boat (by: phish) ======================================

Menu
         [l] Load Inventory from file
         [i] Display Current Inventory
         [a] Add CD
         [d] Delete CD from Inventory
         [s] Save Inventory to binary file
         [x] eXit
Which operation would you like to perform? [l, a, i, d, s or x]: a

 ## tested Add Function w/added error trap code

Enter ID: 3
What is the CD's title? fuego
What is the Artist's name? phish

======= The Current Inventory: =======
ID CDTitle(by:Artist)
1  The Big Wheel (by: Runrig)
2  Bad (by: Michael Jackson)
4 big boat (by: phish)
3 fuego (by: phish) ======================================

Menu
[l] Load Inventory from file
[i] Display Current Inventory
[a] Add CD
[d] Delete CD from Inventory
[s] Save Inventory to binary file
[x] eXit
Which operation would you like to perform? [l, a, i, d, s or x]: s

 ## tested Save Function using binary format

======= The Current Inventory: =======
ID CDTitle(by:Artist)
1  The Big Wheel (by: Runrig)
2  Bad (by: Michael Jackson)
4 big boat (by: phish)
3 fuego (by: phish) ======================================
Save this inventory to BINARY file? [y/n]  W

 ## tested error trap with out of range selection

Choose either y or n. Returning to Main Menu


Menu
         [l] Load Inventory from file
         [i] Display Current Inventory
         [a] Add CD
         [d] Delete CD from Inventory
```

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

======= The Current Inventory: =======
ID CDTitle(by:Artist)
1  The Big Wheel (by: Runrig)
2  Bad (by: Michael Jackson)
4 big boat (by: phish)
3 fuego (by: phish) ======================================
Save this inventory to BINARY file? [y/n]  N

  ## tested cancel selection

The inventory was NOT saved to file. Press [ENTER] to return to the
menu.

Menu
[l] Load Inventory from file
[i] Display Current Inventory
[a] Add CD
[d] Delete CD from Inventory
[s] Save Inventory to binary file
[x] eXit
Which operation would you like to perform? [l, a, i, d, s or x]: s

======= The Current Inventory: =======
ID CDTitle(by:Artist)
1  The Big Wheel (by: Runrig)
2  Bad (by: Michael Jackson)
4 big boat (by: phish)
3 fuego (by: phish) ======================================

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

  ## tested <string>.lower() using cap "Y"
  ## table now with 4 entries

standby...
Saving to CDInventory.bin as BINARY
Saved
Menu
          [l] Load Inventory from file
          [i] Display Current Inventory
          [a] Add CD
          [d] Delete CD from Inventory
          [s] Save Inventory to binary file
          [x] eXit
Which operation would you like to perform? [l, a, i, d, s or x]: d

======= The Current Inventory: =======
ID CDTitle(by:Artist)
1  The Big Wheel (by: Runrig)
2  Bad (by: Michael Jackson)
4 big boat (by: phish)
3 fuego (by: phish) ======================================
Which ID would you like to delete?  R

  ## tested error trap with out of range selection

   ERROR - Numbers for IDs - back to main menu

Menu
          [l] Load Inventory from file
          [i] Display Current Inventory
          [a] Add CD
          [d] Delete CD from Inventory
          [s] Save Inventory to binary file
          [x] eXit
Which operation would you like to perform? [l, a, i, d, s or x]: d

======= The Current Inventory: =======
ID CDTitle(by:Artist)
1  The Big Wheel (by: Runrig)
2  Bad (by: Michael Jackson)
4 big boat (by: phish)
3 fuego (by: phish) ======================================
Which ID would you like to delete?  4

The CD was removed

======= The Current Inventory: =======
ID CDTitle(by:Artist)
1  The Big Wheel (by: Runrig)
2  Bad (by: Michael Jackson)
3  fuego (by: phish)
======================================

  ## table now with 3 entries

Menu
          [l] Load Inventory from file
          [i] Display Current Inventory
          [a] Add CD
          [d] Delete CD from Inventory
          [s] Save Inventory to binary 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 'y' to continue and reload from file. otherwise reload will be canceled
:y

 ## tested Read Function using binary format

reloading...

   Loaded CDInventory.bin from binary

       Updated Table - for verification
======= The Current Inventory: =======
ID CDTitle(by:Artist)
1  The Big Wheel (by: Runrig)
2  Bad (by: Michael Jackson)
4 big boat (by: phish)
3 fuego (by: phish) ======================================

 ## table of 3 entries now overwritten with 4 entries from saved state

Menu
         [l] Load Inventory from file
         [i] Display Current Inventory
         [a] Add CD
         [d] Delete CD from Inventory
         [s] Save Inventory to binary file
         [x] eXit
Which operation would you like to perform? [l, a, i, d, s or x]: x

In [2]:

 ## tested 'save' function with an actual save
```

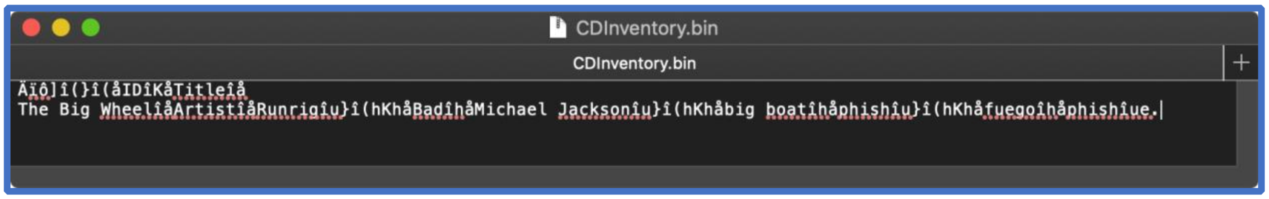


/Users/CMM/Dropbox/Rcds_370--Edu_UW_Py/Assignment06/CDInventory.txt

CDInventory.txt

```
1,The Big Wheel,Runrig
2,Bad,Michael Jackson
4,big boat,phish

```

Figure 1 - Output File “CDInventory.txt” in Initial State



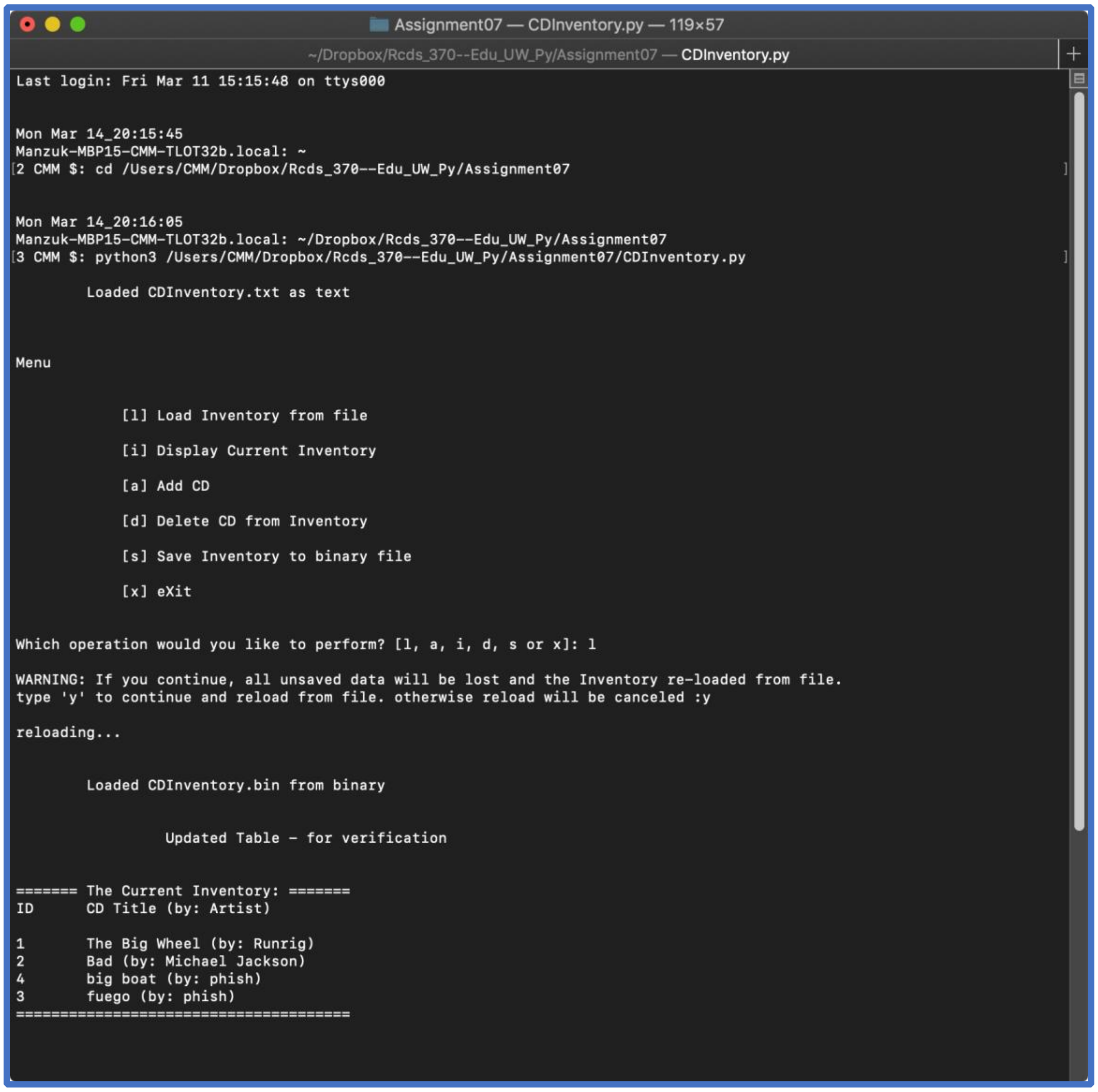
CDInventory.bin

```
Äïô]î(}î(åIDîKåTitleîå
The Big WheelîåArtistîåRunrigîu}î(hKhåBadîhåMichael Jacksonîu}î(hKhåbig boatîhåphishîu}î(hKhåfuegoîhåphishîue.
```

Figure 2 - Output File “CDInventory.bin” in Final State
(opened in text editor)

CD Inventory – Terminal Run

Using ‘python3,’ the script runs as advertised. The Read Function was demonstrated, as it had been the most complex to implement.



Assignment07 — CDInventory.py — 119×57
~/Dropbox/Rcds_370--Edu_UW_Py/Assignment07 — CDInventory.py

```
Last login: Fri Mar 11 15:15:48 on ttys000


Mon Mar 14_20:15:45
Manzuk-MBP15-CMM-TLOT32b.local: ~
[2 CMM $: cd /Users/CMM/Dropbox/Rcds_370--Edu_UW_Py/Assignment07


Mon Mar 14_20:16:05
Manzuk-MBP15-CMM-TLOT32b.local: ~/Dropbox/Rcds_370--Edu_UW_Py/Assignment07
[3 CMM $: python3 /Users/CMM/Dropbox/Rcds_370--Edu_UW_Py/Assignment07/CDInventory.py

        Loaded CDInventory.txt as text



Menu


            [l] Load Inventory from file

            [i] Display Current Inventory

            [a] Add CD

            [d] Delete CD from Inventory

            [s] Save Inventory to binary 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 'y' to continue and reload from file. otherwise reload will be canceled :y

reloading...


        Loaded CDInventory.bin from binary


                 Updated Table - for verification


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

1       The Big Wheel (by: Runrig)
2       Bad (by: Michael Jackson)
4       big boat (by: phish)
3       fuego (by: phish)
======================================
```

Figure 3 - CDInventory.py run on Terminal Command Line

Summary

Assignment 7 went deeper into file usage and added error handling, which led to more complex syntax errors to sort through. After choosing some excessively complex strategies, the code ran successfully in both the IDE and Mac Terminal.