

Andrew Yeo

March 4, 2020

Foundations of Programming: Python

Github Repository: [https://github.com/andyhyeo/Assignment\\_06](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 loop 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 `FileNotFoundError` error.

```
71 @staticmethod
72 def read_file(file_name):
73     """Function to manage data ingestion from file to a list of dictionaries
74
75     Reads the data from binary file identified by file_name into a 2D table
76     (list of dicts) table one line in the file represents one dictionary row in table
77
78     Args:
79         file_name (string): name of file used to read the data from
80
81     Returns:
82         table (list of dict): 2D data structure (list of dicts) that holds the data d
83
84     """
85     try:
86         with open(file_name, 'rb') as objFile:
87             table = pickle.load(objFile)
88             return table
89     except FileNotFoundError:
90         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 call the `open()` function using the with-as structure.

```
53 class FileProcessor:
54     """Processing the data to and from text file"""
55
56     @staticmethod
57     def write_file(file_name, table):
58         """Writes the inventory of IDs, CD Names, and Artists to a text file
59
60     Args:
61         file_name (string): The name of the file that it will write to
62         table (list of dict): 2D data structure (list of dicts) that holds the data d
63
64     Returns:
65         None but saves a file in the directory of the python script
66
67     """
68     with open(file_name, 'wb') as objFile:
69         table = pickle.dump(table, objFile)
```

Figure 2: Substituting Pickle in the Write File Function

## Try-Except for ValueError in IO.ask() function The 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.

```
181 @staticmethod
182 def ask():
183     """Ask user for new ID, CD Title and Artist
184
185     Args:
186         None
187
188     Returns:
189         dicRow (dictionary): A dictionary entry with ID (int): integer that holds
190                             the ID tag,title (string): string that holds the name of the CD
191                             and an artist (string): string that holds the name of the Artist.
192     """
193     while True:
194         strID = input('Enter ID: ').strip()
195         try:
196             strID = int(strID)
197             break
198         except ValueError:
199             print('That is not an integer')
200     strTitle = input('What is the CD\'s title? ').strip()
201     stArtist = input('What is the Artist\'s name? ').strip()
202     dicRow = {'ID': strID, 'CD Title': strTitle, 'Artist': stArtist}
203
204     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 successfully.

```
247 elif strChoice == 'd':
248     # 3.5.1 get Userinput for which CD to delete
249     # 3.5.1.1 display Inventory to user
250     IO.show_inventory(lstTbl)
251     # 3.5.1.2 ask user which ID to remove
252     while True:
253         try:
254             intIDDel = int(input('Which ID would you like to delete? ').strip())
255             break
256         except ValueError:
257             print('That is not an integer')
258     # 3.5.2 search thru table and delete CD
259     DataProcessor.delete_cd(intIDDel,lstTbl)
260     IO.show_inventory(lstTbl)
261     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](#)

```
1. #-----#
2. # Title: CDInventory.py
3. # Desc: Working with classes and functions.
4. # Change Log: (Andrew Yeo, 02-25-2020, Group features into functions)
5. # AYeo, 2020-Feb-25, Created File
6. # Ayeo, 2020-
   Mar-03, Modified the permanent data store to use binary data.
7. # Ayeo, 2020-Mar-03, Subbed Try-
   Except for FileNotFoundError in FileProcessor.read_file() function.
8. # Ayeo, 2020-Mar-03, Subbed Try-
   Except for ValueError in IO.ask() function.
9. # AYeo, 2020-
   Mar-03, Added Error Structure Handling to handle not-
   integers when prompted to delete
10. # AYeo, 2020-
   Mar-03, Tried convert to FileProcessor.append() to pickling unsuccessfully
11. # AYeo, 2020-
   Mar-03, Could try adding structure handling to IO.menu_choice
12.
13. #-----#
14.
15. import pickle
16.
17. # -- DATA -- #
18. strChoice = '' # User input
19. lstTbl = [] # list of lists to hold data
20. dicRow = {} # list of data row
21. strFileName = 'CDInventory.txt' # data storage file
22. objFile = None # file object
23.
24.
25. # -- PROCESSING -- #
26. class DataProcessor:
27.     @staticmethod
28.     def delete_cd(intIDDel, table):
29.         """Deletes a CD row from the table
30.
31.         Args:
```

```

32.         intIDDel (int): ID which indicate which entry user wo
uld like to delete
33.         table (list of dict): 2D data structure (list of dict
s) that holds the data during runtime
34.
35.         Returns:
36.         table (list of dict): 2D data structure (list of di
cts) that holds the data during runtime
37.         """
38.         intRowNr = -1
39.         blnCDRemoved = False
40.         for row in table:
41.             intRowNr += 1
42.             if row['ID'] == intIDDel:
43.                 del table[intRowNr]
44.                 blnCDRemoved = True
45.                 break
46.             if blnCDRemoved:
47.                 print('The CD was removed')
48.             else:
49.                 print('Could not find this CD!')
50.         return table
51.
52.
53. class FileProcessor:
54.     """Processing the data to and from text file"""
55.
56.     @staticmethod
57.     def write_file(file_name, table):
58.         """Writes the inventory of IDs, CD Names, and Artists to
a text file
59.
60.         Args:
61.             file_name (string): The name of the file that it will
write to
62.             table (list of dict): 2D data structure (list of dict
s) that holds the data during runtime
63.
64.         Returns:
65.         None but saves a file in the directory of the python
script
66.
67.         """
68.         with open(file_name, 'wb') as objFile:
69.             table = pickle.dump(table, objFile)
70.

```

```

71.     @staticmethod
72.     def read_file(file_name):
73.         """Function to manage data ingestion from file to a list
of dictionaries
74.
75.         Reads the data from binary file identified by file_name i
nto a 2D table
76.         (list of dicts) table one line in the file represents one
dictionary row in table.
77.
78.         Args:
79.             file_name (string): name of file used to read the dat
a from
80.
81.         Returns:
82.             table (list of dict): 2D data structure (list of dict
s) that holds the data during runtime
83.
84.         """
85.         try:
86.             with open(file_name, 'rb') as objFile:
87.                 table = pickle.load(objFile)
88.             return table
89.         except FileNotFoundError:
90.             pass
91.
92.
93. # -- PRESENTATION (Input/Output) -- #
94.
95. class IO:
96.     """Handling Input / Output"""
97.
98.     @staticmethod
99.     def print_menu():
100.         """Displays a menu of choices to the user
101.
102.         Args:
103.             None.
104.
105.         Returns:
106.             None.
107.         """
108.
109.         print('Menu\n\n[l] load Inventory from file\n[a] Add CD\n
[i] Display Current Inventory')

```

```

110.         print('[d] delete CD from Inventory\n[s] Save Inventory t
o file\n[x] exit\n')
111.
112.     @staticmethod
113.     def menu_choice():
114.         """Gets user input for menu selection
115.
116.         Args:
117.             None.
118.
119.         Returns:
120.             choice (string): a lower case sting of the users inpu
t out of the choices l, a, i, d, s or x
121.
122.         """
123.         choice = ' '
124.         while choice not in ['l', 'a', 'i', 'd', 's', 'x']:
125.             choice = input('Which operation would you like to per
form? [l, a, i, d, s or x]: ').lower().strip()
126.         print() # Add extra space for layout
127.         return choice
128.
129.
130.     @staticmethod
131.     def show_inventory(table):
132.         """Displays current inventory table
133.
134.
135.         Args:
136.             table (list of dict): 2D data structure (list of dict
s) that holds the data during runtime.
137.
138.         Returns:
139.             None.
140.
141.         """
142.         print('==== The Current Inventory: =====')
143.         print('ID\tCD Title (by: Artist)\n')
144.         for row in table:
145.             print('{}\t{} (by:{}'.format(*row.values()))
146.         print('=====')
147.
148.     @staticmethod
149.     def add_cd(row, table):
150.         """Adds a dictionary row to the inventory
151.

```



```

152.         Args:
153.             row (dictionary): dictionary that holds the name of the ID, cd, and artist
154.             table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
155.
156.         Returns:
157.             None.
158.
159.         """
160.         table.append(row)
161.         return table
162.
163. ## Attempt to add pickling to append function
164. # @staticmethod
165. # def add_cd(row, file_name):
166. #     """Adds a dictionary row to the inventory
167. #
168. #     Args:
169. #         row (dictionary): dictionary that holds the name of the ID, cd, and artist
170. #         table (list of dict): 2D data structure (list of dicts) that holds the data during runtime
171. #
172. #     Returns:
173. #         None.
174. #
175. #     """
176. #     # TODO Add Pickling
177. #     with open(file_name, 'ab') as objFile:
178. #         pickle.dump(row, objFile)
179.
180.
181. @staticmethod
182. def ask():
183.     """Ask user for new ID, CD Title and Artist
184.
185.     Args:
186.         None
187.
188.     Returns:
189.         dicRow (dictionary): A dictionary entry with ID (int): integer that holds
190.         the ID tag, title (string): string that holds the name of the CD

```

```

191.         and an artist (string): string that holds the nam
    e of the Artist.
192.         """
193.         while True:
194.             strID = input('Enter ID: ').strip()
195.             try:
196.                 strID = int(strID)
197.                 break
198.             except ValueError:
199.                 print('That is not an integer')
200.             strTitle = input('What is the CD\'s title? ').strip()
201.             stArtist = input('What is the Artist\'s name? ').strip()

202.         dicRow = {'ID': strID, 'CD Title': strTitle, 'Artist': st
    Artist}
203.
204.         return dicRow
205.
206.
207. # 1. When program starts, read in the currently saved Inventory
208. # objFile = open(file_name, 'w') #Commenting out
209. FileProcessor.read_file(strFileName)
210.
211.
212. # 2. start main loop
213. while True:
214.     # 2.1 Display Menu to user and get choice
215.     IO.print_menu()
216.     strChoice = IO.menu_choice()
217.
218.     # 3. Process menu selection
219.     # 3.1 process exit first
220.     if strChoice == 'x':
221.         break
222.     # 3.2 process load inventory
223.     if strChoice == 'l':
224.         print('WARNING: If you continue, all unsaved data will be
    lost and the Inventory re-loaded from file.')
225.         strYesNo = input('type \'yes\' to continue and reload fro
    m file. otherwise reload will be canceled')
226.         if strYesNo.lower() == 'yes':
227.             print('reloading...')
228.             lstTbl = FileProcessor.read_file(strFileName)
229.             IO.show_inventory(lstTbl)
230.         else:

```

```

231.         input('canceling... Inventory data NOT reloaded. Pres
s [ENTER] to continue to the menu.')
232.         IO.show_inventory(lstTbl)
233.         continue # start loop back at top.
234.     # 3.3 process add a CD
235.     elif strChoice == 'a':
236.         # 3.3.1 Ask user for new ID, CD Title and Artist
237.         dicRow = IO.ask()
238.         # 3.3.2 Add item to the table
239.         IO.add_cd(dicRow, lstTbl)
240.         IO.show_inventory(lstTbl)
241.         continue # start loop back at top.
242.     # 3.4 process display current inventory
243.     elif strChoice == 'i':
244.         IO.show_inventory(lstTbl)
245.         continue # start loop back at top.
246.     # 3.5 process delete a CD
247.     elif strChoice == 'd':
248.         # 3.5.1 get Userinput for which CD to delete
249.         # 3.5.1.1 display Inventory to user
250.         IO.show_inventory(lstTbl)
251.         # 3.5.1.2 ask user which ID to remove
252.         while True:
253.             try:
254.                 intIDDel = int(input('Which ID would you like to
delete? ').strip())
255.                 break
256.             except ValueError:
257.                 print('That is not an integer')
258.         # 3.5.2 search thru table and delete CD
259.         DataProcessor.delete_cd(intIDDel, lstTbl)
260.         IO.show_inventory(lstTbl)
261.         continue # start loop back at top.
262.     # 3.6 process save inventory to file
263.     elif strChoice == 's':
264.         # 3.6.1 Display current inventory and ask user for confir
mation to save
265.         IO.show_inventory(lstTbl)
266.         strYesNo = input('Save this inventory to file? [y/
n] ').strip().lower()
267.         # 3.6.2 Process choice
268.         if strYesNo == 'y':
269.             # 3.6.2.1 save data
270.
271.             FileProcessor.write_file(strFileName, lstTbl)
272.         else:

```

```
273.         input('The inventory was NOT saved to file. Press [EN
TER] to return to the menu.')
274.         continue # start loop back at top.
275.     # 3.7 catch-
all should not be possible, as user choice gets vetted in IO, but
to be save:
276.     else:
277.         print('General Error')
```

## Appendix: The Spyder Output

```
1. runfile('/Users/andyeyo/Google Drive/Intro to Python/Mod_07/
   Assignment07/CDInventory.py', wdir='/Users/andyeyo/Google Drive/
   Intro to Python/Mod_07/Assignment07')
2. Menu
3.
4. [l] load Inventory from file
5. [a] Add CD
6. [i] Display Current Inventory
7. [d] delete CD from Inventory
8. [s] Save Inventory to file
9. [x] exit
10.
11.
12. Which operation would you like to perform? [l, a, i, d, s or x]:
    a
13.
14.
15. Enter ID: 1
16.
17. What is the CD's title? Brah Tah
18.
19. What is the Artist's name? Tah Tah Tah
20. ===== The Current Inventory: =====
21. ID      CD Title (by: Artist)
22.
23. 1      Brah Tah (by:Tah Tah Tah)
24. =====
25. Menu
26.
27. [l] load Inventory from file
28. [a] Add CD
29. [i] Display Current Inventory
30. [d] delete CD from Inventory
31. [s] Save Inventory to file
32. [x] exit
33.
34.
35. Which operation would you like to perform? [l, a, i, d, s or x]:
    i
36.
37. ===== The Current Inventory: =====
38. ID      CD Title (by: Artist)
39.
40. 1      Brah Tah (by:Tah Tah Tah)
```

```

41. =====
42. Menu
43.
44. [l] load Inventory from file
45. [a] Add CD
46. [i] Display Current Inventory
47. [d] delete CD from Inventory
48. [s] Save Inventory to file
49. [x] exit
50.
51.
52. Which operation would you like to perform? [l, a, i, d, s or x]:
    s
53.
54. ===== The Current Inventory: =====
55. ID      CD Title (by: Artist)
56.
57. 1      Brah Tah (by:Tah Tah Tah)
58. =====
59.
60. Save this inventory to file? [y/n] y
61. Menu
62.
63. [l] load Inventory from file
64. [a] Add CD
65. [i] Display Current Inventory
66. [d] delete CD from Inventory
67. [s] Save Inventory to file
68. [x] exit
69.
70.
71. Which operation would you like to perform? [l, a, i, d, s or x]:
    d
72.
73. ===== The Current Inventory: =====
74. ID      CD Title (by: Artist)
75.
76. 1      Brah Tah (by:Tah Tah Tah)
77. =====
78.
79. Which ID would you like to delete? 1
80. ===== The Current Inventory: =====
81. ID      CD Title (by: Artist)
82.
83. =====
84. Menu

```

```

85.
86. [l] load Inventory from file
87. [a] Add CD
88. [i] Display Current Inventory
89. [d] delete CD from Inventory
90. [s] Save Inventory to file
91. [x] exit
92.
93.
94. Which operation would you like to perform? [l, a, i, d, s or x]:
    l
95.
96. WARNING: If you continue, all unsaved data will be lost and the I
    nventory re-loaded from file.
97.
98. type 'yes' to continue and reload from file. otherwise reload wil
    l be canceledyes
99. reloading...
100.===== The Current Inventory: =====
101.ID          CD Title (by: Artist)
102.
103.1          Brah Tah (by:Tah Tah Tah)
104.=====
105.Menu
106.
107.[l] load Inventory from file
108.[a] Add CD
109.[i] Display Current Inventory
110.[d] delete CD from Inventory
111.[s] Save Inventory to file
112.[x] exit
113.
114.
115.Which operation would you like to perform? [l, a, i, d, s or x]:
    x

```

## Appendix: The Terminal Output

```
1. Last login: Tue Feb 25 15:46:07 on ttys000
2.
3. The default interactive shell is now zsh.
4. To update your account to use zsh, please run `chsh -s /bin/
  zsh`.
5. For more details, please visit https://support.apple.com/kb/
  HT208050.
6. (base) Andys-MacBook-Pro:~ andyyeo$ cd desktop/assignment06
7. (base) Andys-MacBook-Pro:assignment06 andyyeo$ zip -r dir.zip . -
  x ".*" -x "_MAXOSX"
8. (stored 0%)on
9. adding: Assignment06 Knowledge Document.pdf (deflated 30%)
10. adding: CDInventory.py (deflated 71%)
11. (base) Andys-MacBook-Pro:assignment06 andyyeo$
12. [Restored Mar 4, 2020 at 11:10:28 AM]
13. ## cd /Users/andyyeo/desktop/assignment06 ##
14. Last login: Tue Feb 25 17:51:19 on ttys000
15. Restored session: Tue Feb 25 17:51:49 PST 2020
16.
17. The default interactive shell is now zsh.
18. To update your account to use zsh, please run `chsh -s /bin/
  zsh`.
19. For more details, please visit https://support.apple.com/kb/
  HT208050.
20. (base) Andys-MacBook-Pro:~ andyyeo$ clear
21.
22.
23.
24. (base) Andys-MacBook-Pro:~ andyyeo$ cd desktop/assignment07
25. (base) Andys-MacBook-
  Pro:assignment07 andyyeo$ python CDInventory.py
26. Menu
27.
28. [l] load Inventory from file
29. [a] Add CD
30. [i] Display Current Inventory
31. [d] delete CD from Inventory
32. [s] Save Inventory to file
33. [x] exit
34.
35. Which operation would you like to perform? [l, a, i, d, s or x]:
  a
36.
37. Enter ID: 1
```



```

38. What is the CD's title? Ska Bangers
39. What is the Artist's name? Reel Fig Bish
40. ===== The Current Inventory: =====
41. ID  CD Title (by: Artist)
42. 
43. 1    Ska Bangers (by:Reel Fig Bish)
44. =====
45. Menu
46. 
47. [l] load Inventory from file
48. [a] Add CD
49. [i] Display Current Inventory
50. [d] delete CD from Inventory
51. [s] Save Inventory to file
52. [x] exit
53. 
54. Which operation would you like to perform? [l, a, i, d, s or x]:
    i
55. 
56. ===== The Current Inventory: =====
57. ID  CD Title (by: Artist)
58. 
59. 1    Ska Bangers (by:Reel Fig Bish)
60. =====
61. Menu
62. 
63. [l] load Inventory from file
64. [a] Add CD
65. [i] Display Current Inventory
66. [d] delete CD from Inventory
67. [s] Save Inventory to file
68. [x] exit
69. 
70. Which operation would you like to perform? [l, a, i, d, s or x]:
    s
71. 
72. ===== The Current Inventory: =====
73. ID  CD Title (by: Artist)
74. 
75. 1    Ska Bangers (by:Reel Fig Bish)
76. =====
77. Save this inventory to file? [y/n] y
78. Menu
79. 
80. [l] load Inventory from file
81. [a] Add CD

```

```

82. [i] Display Current Inventory
83. [d] delete CD from Inventory
84. [s] Save Inventory to file
85. [x] exit
86.
87. Which operation would you like to perform? [l, a, i, d, s or x]:
    d
88.
89. ===== The Current Inventory: =====
90. ID   CD Title (by: Artist)
91.
92. 1    Ska Bangers (by:Reel Fig Bish)
93. =====
94. Which ID would you like to delete? 1
95. ===== The Current Inventory: =====
96. ID   CD Title (by: Artist)
97.
98. =====
99. Menu
100.
101.[l] load Inventory from file
102.[a] Add CD
103.[i] Display Current Inventory
104.[d] delete CD from Inventory
105.[s] Save Inventory to file
106.[x] exit
107.
108.Which operation would you like to perform? [l, a, i, d, s or x]:
    l
109.
110.WARNING: If you continue, all unsaved data will be lost and the I
    nventory re-loaded from file.
111.type 'yes' to continue and reload from file. otherwise reload wil
    l be canceledyes
112.reloading...
113.===== The Current Inventory: =====
114.ID   CD Title (by: Artist)
115.
116.1    Ska Bangers (by:Reel Fig Bish)
117.=====
118.Menu
119.
120.[l] load Inventory from file
121.[a] Add CD
122.[i] Display Current Inventory
123.[d] delete CD from Inventory

```

```
124.[s] Save Inventory to file
125.[x] exit
126.
127.Which operation would you like to perform? [l, a, i, d, s or x]:
    x
128.
129.(base) Andys-MacBook-Pro:assignment07 andyyeo$
130.
```