Tiago Rodrigues

19/11/2022

Foundations of Programming, Python

Assignment 06

# Introduction

# In this module I will talk about the script requested for this assignment that is a “copy” of the last assignment but work with classes and functions.

# Assignment06

In this assignment I’ve used the previous script on assignment05 and added classes and functions in my code. I use the same scheme in the menu in the previous assignment, I show to the user a list of commands like “add”, “display”, “save”, “read”, “load”, “delete” and “exit”. Each of these functions are inside a function and each of these functions are inside the respective class.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149  150  151  152  153  154  155  156  157  158  159  160  161  162  163  164  165  166  167  168  169  170  171  172  173  174  175  176  177  178  179  180  181  182  183  184  185  186  187  188  189  190  191  192  193  194  195  196  197  198  199  200  201  202  203  204  205  206  207  208  209  210  211  212  213  214  215  216 | *#------------------------------------------#*  *# Title: CDInventory.py*  *# Desc: Working with classes and functions.*  *# Change Log: (Who, When, What)*  *# DBiesinger, 2030-Jan-01, Created File*  *# Tiago Rodrigues, 2022-Nov-17. Updated File*  *#------------------------------------------#*  *# -- 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 -- #*  *# add another class to the delete Process*  **class** **DeleteProcessor**:  @staticmethod  **def** del\_inventory():  *# 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 = -1  blnCDRemoved = **False**  **for** row **in** lstTbl:  intRowNr += 1  **if** row['ID'] == intIDDel:  **del** lstTbl[intRowNr]  blnCDRemoved = **True**  **break**  **if** blnCDRemoved:  print('The CD was removed')  **else**:  print('Could not find this CD!')    **class** **DataProcessor**:  *# TODone add functions for processing here*  **def** add\_file(file\_name, table):  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*  *# TODone move processing code into function*  objFile = open(strFileName, 'w')  **for** row **in** lstTbl:  lstValues = list(row.values())  lstValues[0] = str(lstValues[0])  objFile.write(','.join(lstValues) + '**\n**')  objFile.close()  **else**:  input('The inventory was NOT saved to file. Press [ENTER] to return to the menu.')    **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*  *Reads the data from file identified by file\_name into a 2D table*  *(list of dicts) table one line in the file represents one dictionary row in table.*  *Args:*  *file\_name (string): name of file used to read the data from*  *table (list of dict): 2D data structure (list of dicts) that holds the data during runtime*  *Returns:*  *None.*  *"""*  **try**:  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(',')  dicRow = {'ID': int(data[0]), 'Title': data[1], 'Artist': data[2]}  table.append(dicRow)  objFile.close()  **except**:  print('The file **{}** doesn**\'**t exist'.format(strFileName))    @staticmethod  **def** write\_file(file\_name, table):  lstRow = []    **for** line **in** table:  lstRow += str(line) + ','  lstRow = lstRow[:-1] + '**\n**'  objFile = open(file\_name, 'w')  objFile.write(lstRow)  objFile.close()  **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**\t**CD Title (by: Artist)**\n**')  **for** row **in** table:  print('**{}\t{}** (by:**{}**)'.format(\*row.values()))  print('======================================')  *# TODone add I/O functions as needed*  @staticmethod  **def** add\_iventory():  *# 3.3.1 Ask user for new ID, CD Title and Artist*  strID = input('Enter ID: ').strip()  strTitle = input('What is the CD**\'**s title? ').strip()  stArtist = input('What is the Artist**\'**s name? ').strip()    *# 3.3.2 Add item to the table*  intID = int(strID)  dicRow = {'ID': intID, 'Title': strTitle, 'Artist': stArtist}  lstTbl.append(dicRow)  IO.show\_inventory(lstTbl)      *# 1. When program starts, read in the currently saved Inventory*  **try**:  FileProcessor.read\_file(strFileName, lstTbl)  **except**:  print('The file **{}** doesn**\'**t exist'.format(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...')  FileProcessor.read\_file(strFileName, lstTbl)  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*  *# TODone move IO code into function*  *# 3.3.2 Add item to the table*  *# TODone move processing code into function*  IO.add\_iventory()    **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)  DeleteProcessor.del\_inventory()  *# 3.5.1.2 ask user which ID to remove*  *# 3.5.2 search thru table and delete CD*  *# TODone move processing code into function*  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*  *# start loop back at top.*  DataProcessor.add\_file(strFileName, lstTbl)  *# 3.7 catch-all should not be possible, as user choice gets vetted in IO, but to be save:*  **else**:  print('General Error') |

## Using Spyder terminal

Here I’ve added some data into the list.

Uma imagem com texto

Descrição gerada automaticamente

Ilustração - Data added inside the list

Display the data inside the list.

Uma imagem com texto

Descrição gerada automaticamente

Ilustração 2 - Display current data in list

Save data inside the list into .txt File.

Uma imagem com texto

Descrição gerada automaticamente

Ilustração - Save data inside the file

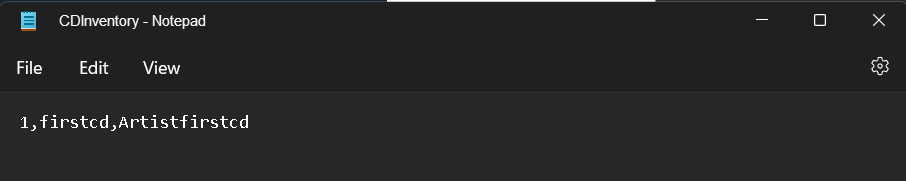


Ilustração - .txt file

Loading the data inside the file

Uma imagem com texto

Descrição gerada automaticamente

Ilustração - Loading data

Removing the select id from the list

Uma imagem com texto

Descrição gerada automaticamente

Ilustração - remove cd

## Using computer terminal

Adding more data to the list

Uma imagem com texto

Descrição gerada automaticamente

Ilustração - Data added inside the list

Display de data inside list

Uma imagem com texto

Descrição gerada automaticamente

Ilustração - Display current data in list

Save the data inside the .txt file.

Uma imagem com texto

Descrição gerada automaticamente

Ilustração - Save data inside the file



Ilustração - .txt file

Loading all the data inside the .txt file.

Uma imagem com texto

Descrição gerada automaticamente

Ilustração 11 - Loading data

Removing the cd.

Uma imagem com texto

Descrição gerada automaticamente

Ilustração - remove cd

# Summary

In this assignment we’ve introduced classes and functions, this is an easier way to reutilize or use the code in different places.

Here is the GitHub url:

<https://github.com/Ehz10/Assignment_06.git>