Jerry Hsieh

Mar-3-2022

Foundations of Programming: Python

Assignment 06

CD Inventory (Functions)

# Introduction

This is a simple script of how to modify the sample script as required to add “function” in the script. There are a lot of “TODOs” that need to be done. Use “Function” to do things for data adding/deleting/processing.

I studied the Module\_06 materials and tried to understand how “functions” works and make the scripts. According to the sample script of Assignment06\_Starter, we can do some modifications on this script to achieve the goal.

# Explanation of the script

We would like to use Functions, which are a way of grouping statements and making this group available via a programmer defined name. We have 3 groups of classes: 1) Data Processor (Figure 1), 2) File Processor (Figure 2), and 3) Input and Output (IO) (Figure 3). In Data Processor, we can add files or delete data. In File Processor, we can read data from or save the data to text file. In IO, we can show the menu, ask user to select what to do, or ask user to input CD data.

Text

Description automatically generated

Figure - Data Processor

Graphical user interface, text, chat or text message, website

Description automatically generated

Figure - File Processor

Text

Description automatically generated

Figure - IO

Here we explain some scripts in detail.

* For “add\_file” function, First I defined “CDInput” function to return 3 values every time we call this function (Figure 4). I stored these values to D1, D2, and D3 (Figure 5), then use “add\_file” function to process the data. (Figure 6)

Text

Description automatically generated

Figure – CDInput function for data input

Text

Description automatically generated

Figure - Data in D1,D2,and D3 when user chooses ‘a’

Text

Description automatically generated

Figure - 'add\_file' function

* We can call “delete\_file” function when user input ’d’ (Figure 7). Use “del” to delete the row with ID value which matched our input (Figure 8). Now, we have a new 2D-table for the data we input. Then, return this new 2D-table.

Text

Description automatically generated

Figure - Delete data when user input 'd'

Text

Description automatically generated

Figure -- 'delete\_file' function

* For “write\_file” function, we can save the stored data to the text file. Every time we call this function (Figure 9), it will open the text file and write the data to it. (Figure 10)

A picture containing text

Description automatically generated

Figure - Save data when user input 's'

Text

Description automatically generated

Figure - 'write\_file' function

* If we want to load the data from text file. We can call “read\_file” function. Use “for-loop” to read the text file row by row, then save them into dictionaries. “Read\_file” function aim to achieve the goal of reading (Figure 11 & 12).

Text

Description automatically generated

Figure - Load data when user input 'l' and for-loop

Text

Description automatically generated

Figure – for-loop and 'read\_file' function

* Use “show\_inventory” function to display the data which stored in memory (Figure 13)

Text

Description automatically generated

Figure - 'show inventory' function

# Save your script

File path of my script is C:\\_FDP\Assignment06\CDInventory.py (Figure 14)

**A screenshot of a computer

Description automatically generated with medium confidence**

Figure - File path

# Run your script/Verify correct functioning

I used Spyder to run this script first. I planned to enter 2 CD datasets (Figure 15). First, I input ‘a’ for data input then input ‘i’ to display. CD data stored perfectly, and we can see the display result. (Figure 16)



Figure - 2 datasets

Text

Description automatically generated with medium confidence

Figure - Display the data

Enter ‘s’ to save data to txt file and open txt file to confirm. (Figure 17 & 18)

A screenshot of a computer

Description automatically generated with medium confidence

Figure - Input 's'

Text

Description automatically generated

Figure - Data stored in txt file

Load data from the txt file. (Figure 19)

Text

Description automatically generated

Figure - Load data from text file

Delete an entry. I entered “d” to select delete function and entered “5”, which means dataset with ID==5 is what we want to delete. (Figure 20)

Graphical user interface, text

Description automatically generated

Figure - Delete an entry

Use ‘i’ to display the result again. The row with ID==5 has been deleted.

Text

Description automatically generated

Figure - Display the result after deleting an entry

# Run your script (Terminal Window)

I used Anaconda Prompt to run the scripts again with different data input. I added the data as follows. (Figure 22 & 23)

Text

Description automatically generated

Figure - Add data (1)

Text

Description automatically generated

Figure - Add data (2)

Display the data (Figure 24):

Text

Description automatically generated

Figure - Display the data (Anaconda Prompt)

Save the data to txt file (Figure 25) and text file result (Figure 26):

Text

Description automatically generated

Figure - Save the data (Anaconda Prompt)

Text

Description automatically generated

Figure - Result in text file (Anaconda Prompt)

Load the data from txt file (Figure 27):

Text

Description automatically generated

Figure - Load result from text file (Anaconda Prompt)

Delete CD ID number “1” then display it: (Figure 28)

Text

Description automatically generated

Figure - Delete and the result (Anaconda Prompt)

The scripts work perfectly!

# GitHub link for Assignment

[chhsieh0630/Assignment\_06: IT FDN 110 B Wi 22: Foundations Of Programming: Python (github.com)](https://github.com/chhsieh0630/Assignment_06)

# Summary

In this assignment, I learned how to add functions to the scripts. The difficulties are telling the differences of variables. The variable used in main body could be different from which used in functions. We put our data into the function to process, then return a value. We use main body to call the functions only, so it makes the main script body clean and tidy. This is a very good practice for me in the future to handle data such as this.