

Jenn Wall

05 March, 2022

Foundations of Programming: Python

Assignment 06

Functions

Introduction

In this assignment we covered how to use functions and classes. Additionally, we touched on variable scope and DocString. We then used this knowledge to adapt our CDInventory script to perform using function calls. This allowed us to more cleanly organize our code into sections for data, processing, and presentation.

What Is a Function

A function is like a set of instructions. We create a function to perform a set of actions to be run later in the program. This helps reduce repetitive code within our program by allowing us to only write the processing code once, and then calling it as many times as we'd like throughout the program. The function is set up in the data section of our script, but does not run here. Functions do not run until they are called in the presentation section of our script. Functions cannot be called before they are defined.

Functions can receive parameters, otherwise known as arguments, just like built-in functions like `len()` and `range()`. A parameter is a value received by a function. Parameters work like variables and are placed inside the parentheses of the function header. There is no limit to the number of parameters in a function.

Return Values

Functions also return information through return values. This is done in the return statement. You can also have multiple return values in a function. Return values are captured by the program when a function is called.

Global vs Local Variables

Because functions are encapsulated, meaning whatever is performed in a function is not accessible outside that function, variables set within a function cannot be called or accessed outside of that function. These are called local variables.

The exception to that rule is a global variable. You can set a global variable in a function that will persist outside of the function. Global variables cannot be changed from within other functions, but by shadowing a global variable you can hide it with a local variable of the same name.

Organizing Your Code

Using functions allows us to further organize our code into easily readable sections. Functions let us set up the processing portion of our code without needing to include the presentation portion. We create our presentation operations later on and call our previously defined functions to run the processing code.

Function vs Class

Classes are a group of functions, variables, and constants. This allows for even further organization and grouping within our code. Classes will be covered in more detail later.

Create a Script

For assignment 06 we were tasked with updating our CD Inventory code to include functions and docstrings. I moved all of the processing code from the main body to functions in the processing section.

```
20 class DataProcessor:
21     """Precessing data to add to memory"""
22
23     @staticmethod
24     def add_data(CD_data):
25         """Adding CD data to memory
26
27         Args:
28             CD_data (list): [ID, Title, Artist] entered by user
29
30         Returns:
31             None.
32         """
33         intID = int(CD_data[0])
34         strTitle = str(CD_data[1])
35         strArtist = str(CD_data[2])
36         dicRow = {'ID': intID, 'Title': strTitle, 'Artist': strArtist}
37         lstTbl.append(dicRow)
38
```

Figure 1. Function to add data to memory

```
65
66 @staticmethod
67 def write_file(file_name, table):
68     objFile = open(strFileName, 'w')
69     for row in lstTbl:
70         lstValues = list(row.values())
71         lstValues[0] = str(lstValues[0])
72         objFile.write(','.join(lstValues) + '\n')
73     objFile.close()
74
75 @staticmethod
76 def delete_file(intIDDel):
77     """Deleting specific CD data from memory
78
79     Args:
80         intIDDel (integer): The ID of the CD to be deleted
81
82     Returns:
83         None
84     """
85     intRowNr = -1
86     blnCDRemoved = False
87     for row in lstTbl:
88         intRowNr += 1
89         if row['ID'] == intIDDel:
90             del lstTbl[intRowNr]
91             blnCDRemoved = True
92             break
93     if blnCDRemoved:
94         print('The CD was removed')
95     else:
96         print('Could not find this CD!')
```

Figure 2. Functions to write to file and delete from memory

```

151     @staticmethod
152     def CD_data_add():
153         """Gathers CD data to add to memory
154
155         Args:
156             CD_data (list): ID, Title, Artist
157
158         Returns:
159             None
160
161         """
162         strID = input('Enter ID: ').strip()
163         strTitle = input('What is the CD\'s title? ').strip()
164         strArtist = input('What is the Artist\'s name? ').strip()
165         CD_data = [strID, strTitle, strArtist]
166         return CD_data

```

Figure 3. Function to collect data from user to add to memory

These functions were then called in the main body.

```

195     # 3.3 process add a CD
196     elif strChoice == 'a':
197         # 3.3.1 Ask user for new ID, CD Title and Artist
198         CD_data = IO.CD_data_add()
199         # 3.3.2 Add item to the table
200         DataProcessor.add_data(CD_data)
201         # 3.3.3 Display current inventory after adding data
202         IO.show_inventory(lstTbl)
203         continue # start loop back at top.

```

Figure 4. Calling functions to add data to memory

```

210     # 3.5 process delete a CD
211     elif strChoice == 'd':
212         # 3.5.1 get Userinput for which CD to delete
213         # 3.5.1.1 display Inventory to user
214         IO.show_inventory(lstTbl)
215         # 3.5.1.2 ask user which ID to remove
216         intIDDel = int(input('Which ID would you like to delete? ').strip())
217         # 3.5.2 search thru table and delete CD
218         FileProcessor.delete_file(intIDDel)
219         # 3.5.3 show current inventory after removing data
220         IO.show_inventory(lstTbl)
221         continue # start loop back at top.

```

Figure 5. Calling functions to delete data from memory

<https://github.com/jrwall704/Assignment06>

Running the Code

I then tested the code by running each option and checking the output was expected.

```

Which operation would you like to perform? [l, a, i, d, s or x]: i
===== The Current Inventory: =====
ID  CD Title (by: Artist)

1   Bad (by:Michael Jackson)
2   Thriller (by:Michael Jackson)
3   The Big Wheel (by:Runrig)
=====

```

Figure 6. Running display current inventory option

```

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

Enter ID: 4

What is the CD's title? Elephant

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

1   Bad (by:Michael Jackson)
2   Thriller (by:Michael Jackson)
3   The Big Wheel (by:Runrig)
4   Elephant (by:The White Stripes)
=====

```

Figure 7. Running add data to memory

```

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

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

1   Bad (by:Michael Jackson)
2   Thriller (by:Michael Jackson)
3   The Big Wheel (by:Runrig)
4   Elephant (by:The White Stripes)
=====

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

```

Figure 8. Running save data to file

```

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

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

1   Bad (by:Michael Jackson)
2   Thriller (by:Michael Jackson)
3   The Big Wheel (by:Runrig)
4   Elephant (by:The White Stripes)
=====

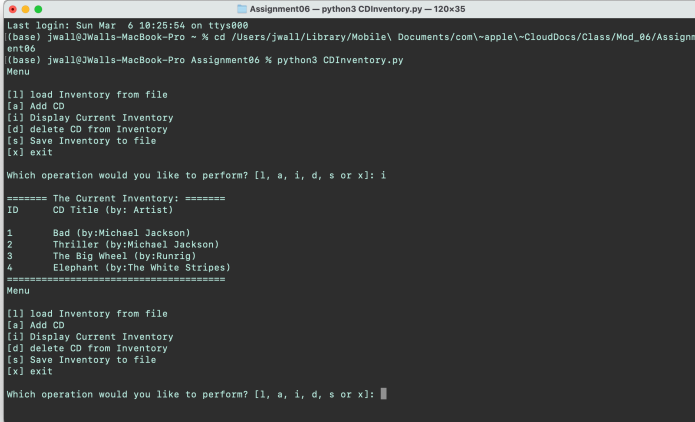
Which ID would you like to delete? 3
The CD was removed
===== The Current Inventory: =====
ID  CD Title (by: Artist)

1   Bad (by:Michael Jackson)
2   Thriller (by:Michael Jackson)
4   Elephant (by:The White Stripes)
=====

```

Figure 9. Running delete data from memory

I also tested my code by running it through terminal.



```
Assignment06 - python3 CDInventory.py - 120x35
Last login: Sun Mar  6 10:26:54 on ttys000
(base) jwall@JWalls-MacBook-Pro ~ % cd /Users/jwall/Library/Mobile Documents/com~apple~CloudDocs/Class/Mod_06/Assignment06
(base) jwall@JWalls-MacBook-Pro Assignment06 % python3 CDInventory.py
Menu

[i] 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       Bad (by:Michael Jackson)
2       Thriller (by:Michael Jackson)
3       The Big Wheel (by:Runrig)
4       Elephant (by:The White Stripes)
=====

Menu

[i] 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]:
```

Figure 10. Running code in terminal

Summary

To recap, assignment 06 introduced us to functions and a little bit about classes. Functions will be a big help in reducing repeated code and keeping out code organized into clear and concise sections. I found the material for this assignment to be very sparse and vague. I ended up having to search the internet for additional sources of information to complete the assignment.