Puja Das

02/14/21

FDNProgramming 110

Assignment05

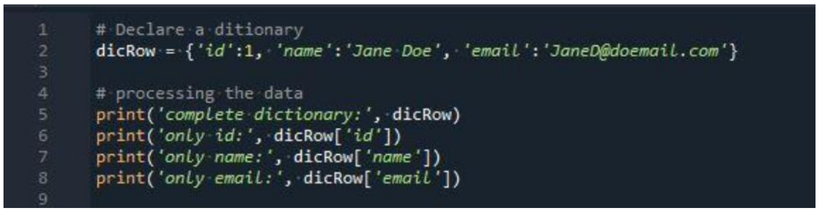
Creating a CD Inventory Program using a Python Script - Dictionaries

Introduction

This module is a continuation of last week's module, except using dictionaries instead of lists.

Difference between Dictionaries and Lists

At first glance, dictionaries and lists may seem like they are interchangeable, however they are not. The simplest difference is that lists are ordered and dictionaries are unordered. With lists, we use indices to access the information, but in dictionaries, key is used. The full syntax is {key: value}. In the example below, you can see that first a dictionary is declared, with each of the row titles and their values listed side by side with a colon. Thus, you can even specify what you want to get printed as lines 5-8 display.

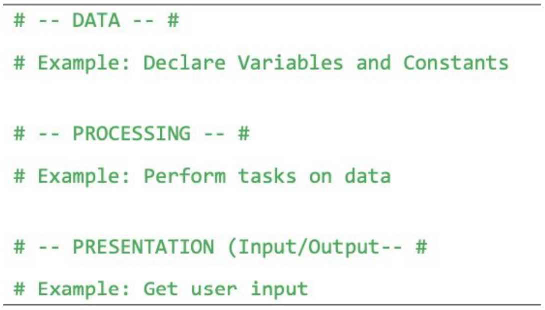


```
1   # Declare a ditionary
2   dicRow = {'id':1, 'name':'Jane Doe', 'email':'JaneD@doemail.com'}
3
4   # processing the data
5   print('complete dictionary:', dicRow)
6   print('only id:', dicRow['id'])
7   print('only name:', dicRow['name'])
8   print('only email:', dicRow['email'])
9
```

Figure 1 Example of a dictionary

Separation of Concerns (SoC)

This is a programming pattern that is a principle that is often exhibited. It can also be known as Graphical User Interface (GUI). This is a design principle for separating a computer program into smaller and more distinct sections that pertain to a different concern. A concern is a set of information that affects the code of a computer program. Most of these computer programs can be divided into three sections: Data, Processing and Presentation, aka, input and output. A basic outline of that is provided below.

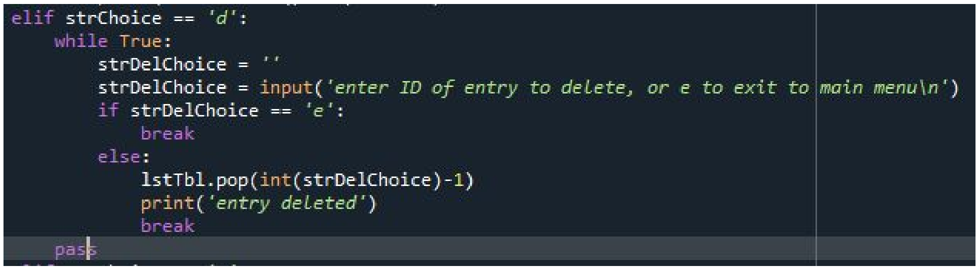


```
# -- DATA -- #
# Example: Declare Variables and Constants

# -- PROCESSING -- #
# Example: Perform tasks on data

# -- PRESENTATION (Input/Output-- #
# Example: Get user input
```

Figure 2 SoC example

How to Create the Script

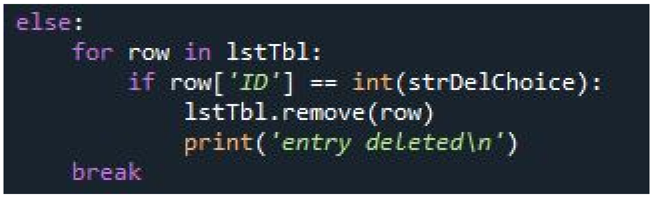
Looking at Lab B and comparing it to Lab A was really helpful in understanding the major differences in code between lists and dictionaries, especially being able to compare them side by side. The basic idea was the same, of course. At the forefront there is a While True loop, and a menu of options to choose from which were indicated by letters, l, a, i, d, s, and x. I added the dictionary statements where I needed to add data and where I was loading existing data. I found the most challenging part of this script creation to be thinking about how to navigate the delete entry function. I first came up with the following solution:



```
elif strChoice == 'd':
    while True:
        strDelChoice = ''
        strDelChoice = input('enter ID of entry to delete, or e to exit to main menu\n')
        if strDelChoice == 'e':
            break
        else:
            lstTbl.pop(int(strDelChoice)-1)
            print('entry deleted')
            break
    pass
```

Figure 3 First attempt at assignment, one-time working delete code

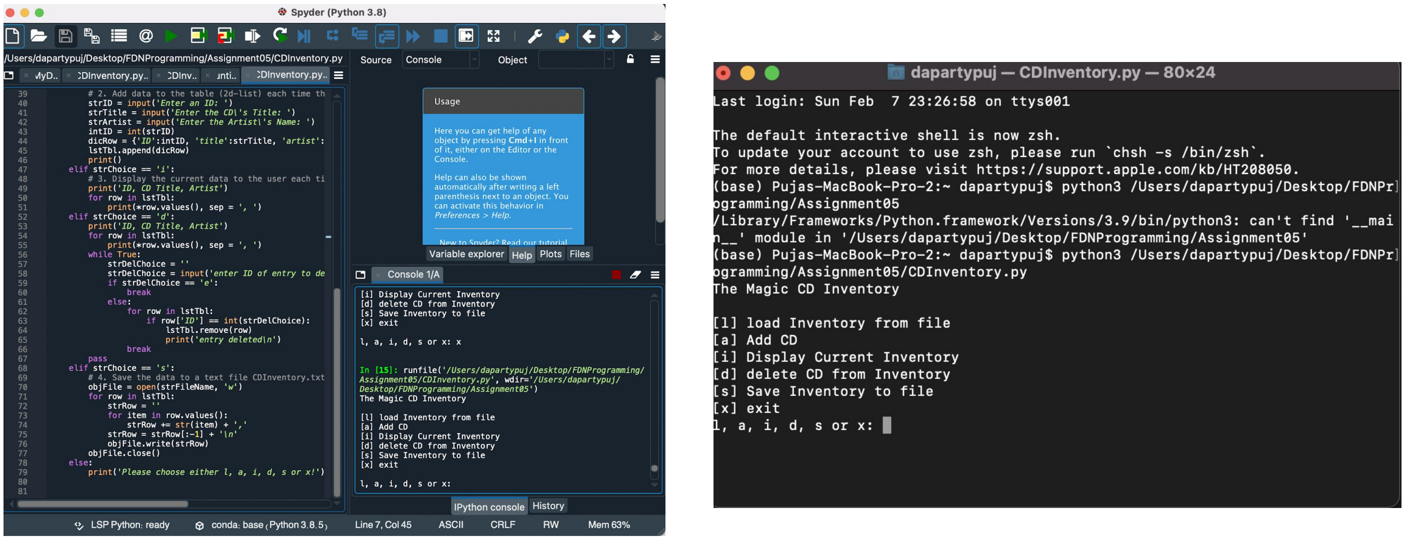
While this works for the first time you use it, it does not work for further times as there is then a difference between the internal order of the list and the external ID that the human reads. I thus had to find a way for the script to ask “If ID = X, then delete row in which ID = X”, which I accomplished with a for loop:



```
else:
    for row in lstTbl:
        if row['ID'] == int(strDelChoice):
            lstTbl.remove(row)
            print('entry deleted\n')
    break
```

Figure 4 Second attempt at delete code, now with a loop that searches for ID number

I also added code that recreates the current table for the user so they can remind themselves which row to exactly delete.



```
            # 2. Add data to the table (2d-list) each time th
            strID = input('Enter an ID: ')
            strTitle = input('Enter the CD\'s Title: ')
            strArtist = input('Enter the Artist\'s Name: ')
            intID = int(strID)
            dicRow = {'ID':intID, 'title':strTitle, 'artist':
            lstTbl.append(dicRow)
            print()
        elif strChoice == 'i':
            # 3. Display the current data to the user each ti
            print('ID, CD Title, Artist')
            for row in lstTbl:
                print(*row.values(), sep = ', ')
        elif strChoice == 'd':
            print('ID, CD Title, Artist')
            for row in lstTbl:
                print(*row.values(), sep = ', ')
            while True:
                strDelChoice = ''
                strDelChoice = input('enter ID of entry to de
                if strDelChoice == 'e':
                    break
                else:
                    for row in lstTbl:
                        if row['ID'] == int(strDelChoice):
                            lstTbl.remove(row)
                            print('entry deleted\n')
                    break
            pass
        elif strChoice == 's':
            # 4. Save the data to a text file CDInventory.txt
            objFile = open(strFileName, 'w')
            for row in lstTbl:
                strRow = ''
                for item in row.values():
                    strRow += str(item) + ','
                strRow = strRow[:-1] + '\n'
                objFile.write(strRow)
            objFile.close()
        else:
            print('Please choose either l, a, i, d, s or x!')
```

Here you can get help of any object by pressing **Cmd+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help*.

New to Spyder? Read our tutorial

```
[i] Display Current Inventory
[d] delete CD from Inventory
[s] Save Inventory to file
[x] exit

l, a, i, d, s or x: x


In [15]: runfile('/Users/dapartypuj/Desktop/FDNProgramming/
Assignment05/CDInventory.py', wdir='/Users/dapartypuj/
Desktop/FDNProgramming/Assignment05')
The Magic CD Inventory

[l] load Inventory from file
[a] Add CD
[i] Display Current Inventory
[d] delete CD from Inventory
[s] Save Inventory to file
[x] exit

l, a, i, d, s or x:
```

```
Last login: Sun Feb  7 23:26:58 on ttys001

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) Pujas-MacBook-Pro-2:~ dapartypuj$ python3 /Users/dapartypuj/Desktop/FDNPr
ogramming/Assignment05
/Library/Frameworks/Python.framework/Versions/3.9/bin/python3: can't find '__mai
n__' module in '/Users/dapartypuj/Desktop/FDNProgramming/Assignment05'
(base) Pujas-MacBook-Pro-2:~ dapartypuj$ python3 /Users/dapartypuj/Desktop/FDNPr
ogramming/Assignment05/CDInventory.py
The Magic CD Inventory

[l] load Inventory from file
[a] Add CD
[i] Display Current Inventory
[d] delete CD from Inventory
[s] Save Inventory to file
[x] exit
l, a, i, d, s or x: 
```

Figure 5 Left Script running in python Right Script running in terminal

Summary

Overall, this code was a much longer one than previous coding assignments. However, understanding the basic principles of lists and dictionaries, and being able to differentiate between the two was fundamental to completing this code successfully. Additionally, in more complex and longer programs, it will become important to organize code with SoCs.

References

Dawson, Michael. *Python Programming for the Absolute Beginner: Michael Dawson*. Course Technology Cengage Learning, 2010.

Module 05 notes and video