**Lab 2**

**Author**

by Paul Charles

**Objective**

Utilize numpy, object-oriented programming, and dictionaries within Python

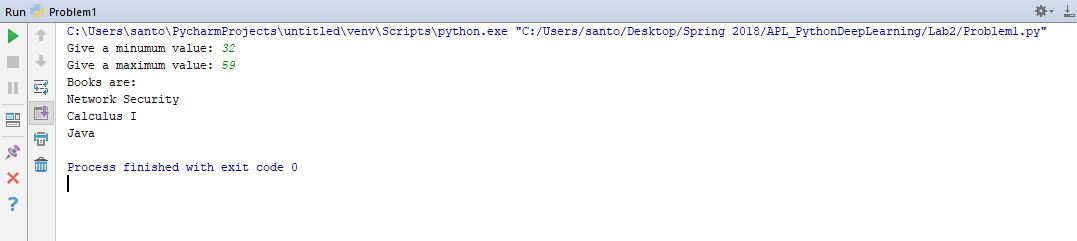
**Features**

* Find books based on price within a price range.
* Take a list and of contacts and change the number, name, and email using dictionaries.
* Create a library system for a school in which the student has to check out a selection of books that are added to the library from the librarian.
* Find the most common number in a number list that is randomly generated

**Configuration**

* Python
* Python Library re
* PyCharm
* Python Library numpy
* Python Library random

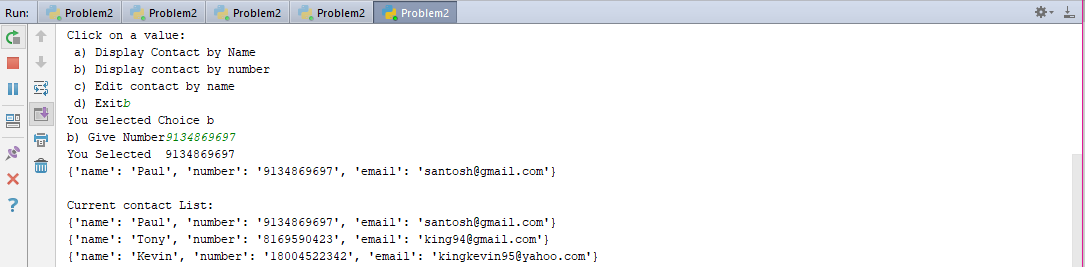
**Demo (Input/Output)**



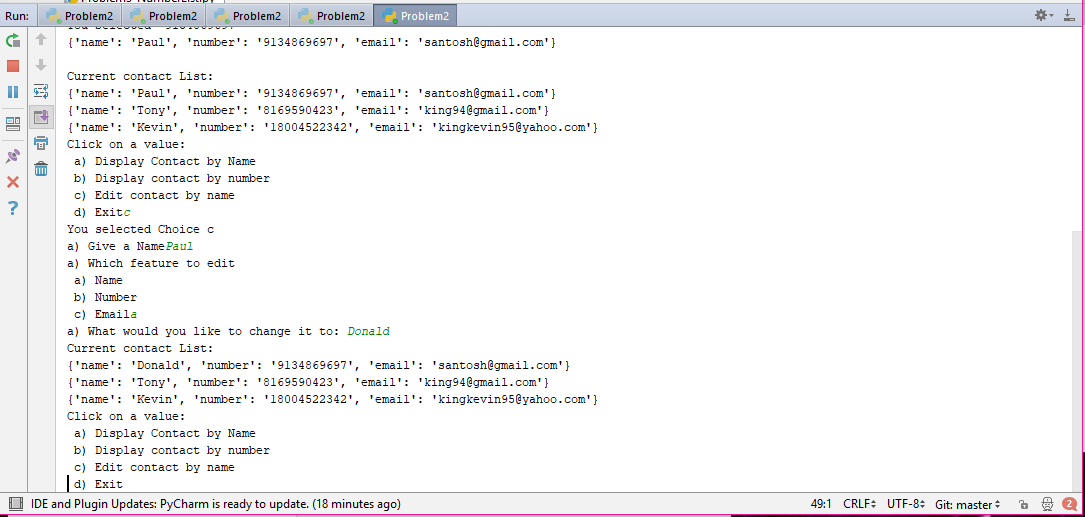
**Problem 1-Price Range** This image shows how the user inputting the value for minimum price and maximum price. Then the output will give the books that are within the price range.



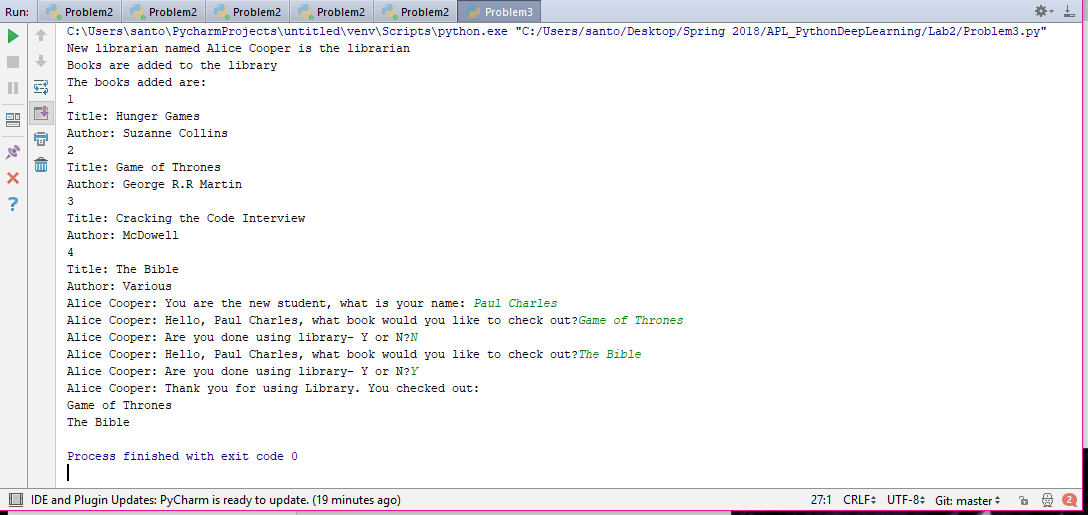
**Problem 2a-User obtains Contact by Name: Input and Output** This shows the user finding the contact based on the Name.



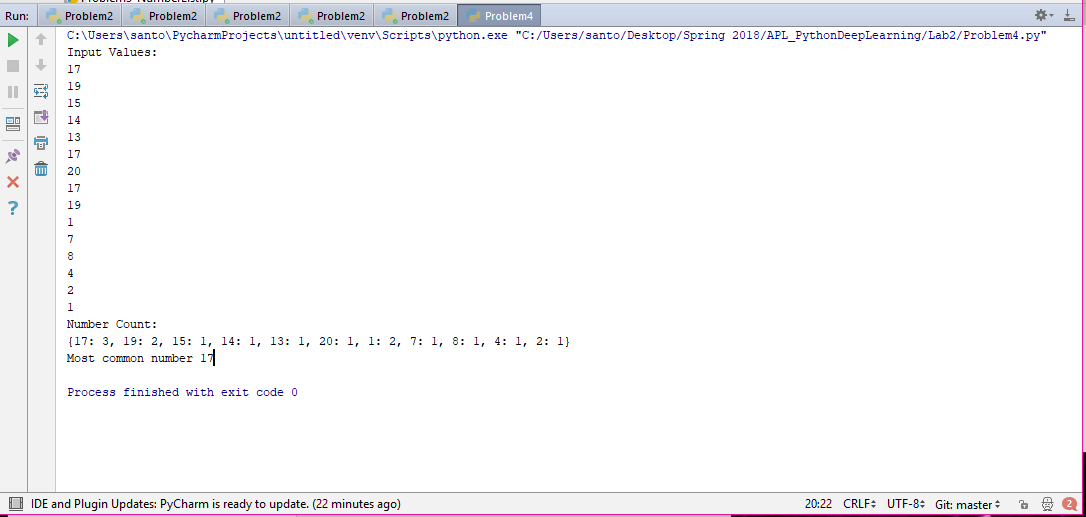
**Problem 2b-User obtains Contact by Number: Input and Output** This shows the user finding the contact based on the Number of the contact.



**Problem 2c-User edits Contact: Input and Output** This shows the user editing the contact based on the Name, can also edit by number and email.



**Problem 3-Library System: Input and Output** This shows the Library which is based on a librarian and a student, which are inherited form a Person class. Then there are books available to check out, the librarian adds the book to the shelf. The books are of two types: Nonfiction and Novel, which are inherited from Book class. The librarian adds the books to the shelf and the student checks them out. The librarian tells the student which book(s) they checked out.



**Problem 4-Display most common number using numpy: Input and Output** This shows a random list being generated as input. A dictionary then sorts the numbers based on frequency, and then displays the most common number.

**Implementation**

import numpy as np a = np.random.rand(15)

b=[] for i in a: b.append(int(round(20\*i)))

#Gives input values print('Input Values: ') for i in b: print(i)

mydict = {} for i in b: if i in mydict: val = mydict.get(i,"none") mydict[i]=val+1 else: mydict[i]=1

**Numpy library** This is a code snippet showing how numpy is used to produce random integers.

#Book class for library system class Book(): def \_\_init\_\_(self): self.name = 'Default Title' self.pages = 100 self.author = 'Default Author' #Book Novel and inherited classes class Novel(Book): def \_\_init\_\_(self): super(Book, self).\_\_init\_\_() self.genre = 'Drama' self.hero = 'Hero' self.villan = 'Villain' self.plot = 'The hero wins'

**Objected oriented programming** This is a code snippet showing how object-oriented programming works.

#Gives input values print('Input Values: ') for i in b: print(i)

mydict = {} for i in b: if i in mydict: val = mydict.get(i,"none") mydict[i]=val+1 else: mydict[i]=1

**Dictionary use** This code snippet shows how the dictionary is used.

**Deployment**

The code was executed in PyCharm to create the code. The code was then added to Github and committed. The code was then pushed to my Github account. The code was in the Source folder, and the images and documentation was in the Documentation folder.

**Limitations**

* Better libraries available
* Harder to utilize classes compared to other OOP languages, like Java-especially for encapsulation

**References**

**StackOverflow:**I referred to StackOverflow for help and references to complete my lab. Particularly for dictionaries, numpy, and class (OOP) usage.