CS5590/490-Python/DeepLearning

LAB ASSIGNMENT-2

By: Syed Jawad Hussain Shah

Student ID: 16117985

Class ID: 46

**Author:** Syed Jawad Hussain Shah

**Objective:**

The objective of this lab task is to get familiar with Python language and its features, like dictionaries, sets, classes and NumPy and concepts like inheritance, multiple inheritance and super calls.

**Features:**

The features of this lab include to write a program to find the books from the dictionary in the range given by user. The second feature of the program is to create a list of contact and then prompt a user to do following: display contact by name, by number, edit a contact, or exit. The third task is to write a program that uses classes to create a management system (Airline management system) that meets all the prerequisites. The fourth feature is about NumPy library. The task is to use NumPy to create a program that randomly generates a vector of size 15 containing integers between 0 and 20 and find the most frequent integer.

**Configuration:**

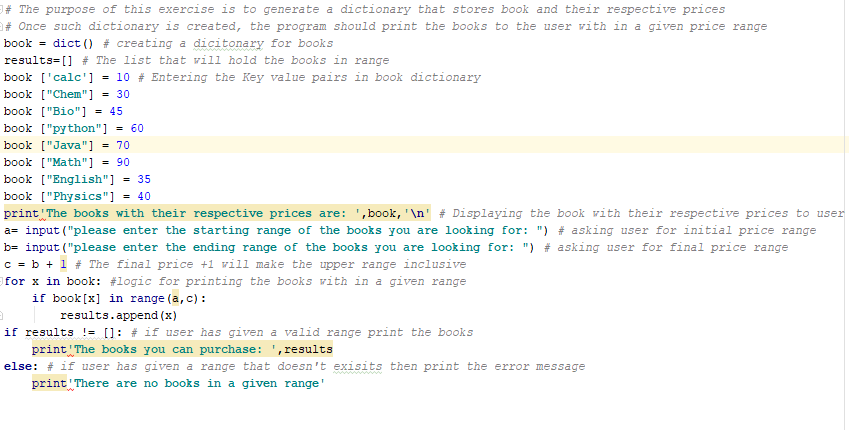
Python 3.6.4

IDE: JetBrains PyCharm community Edition 2017.3.3

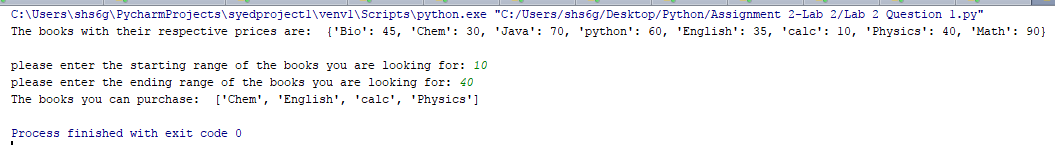
**Input/output (screenshots):**

Q1) Books in a Dictionary:

1. Code:

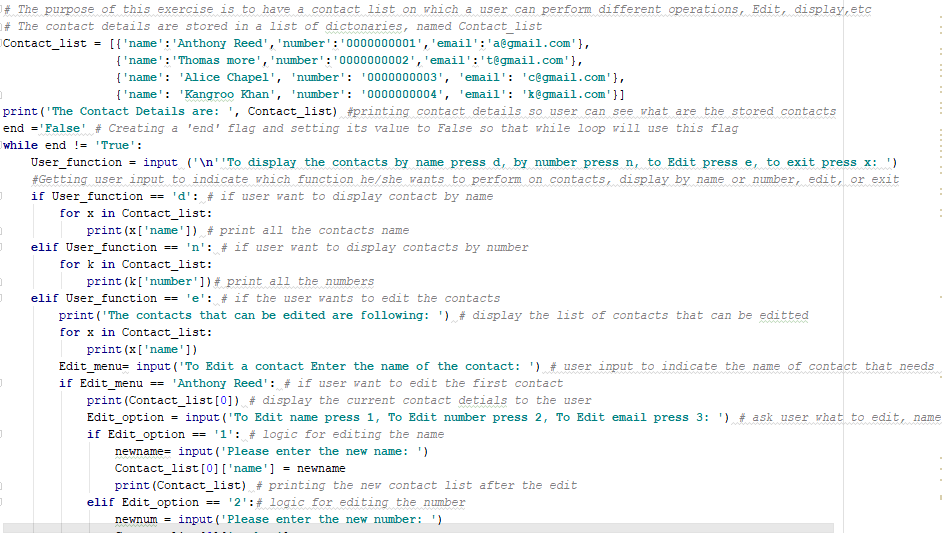


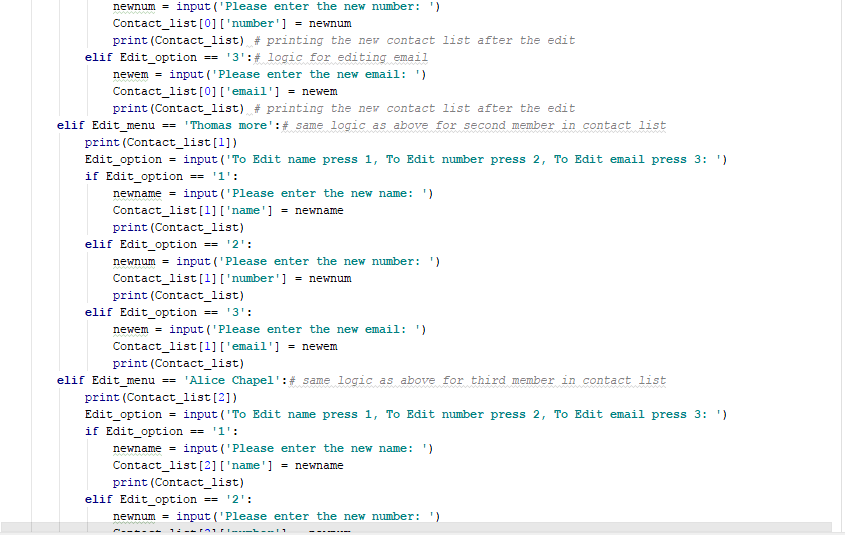
1. Results:

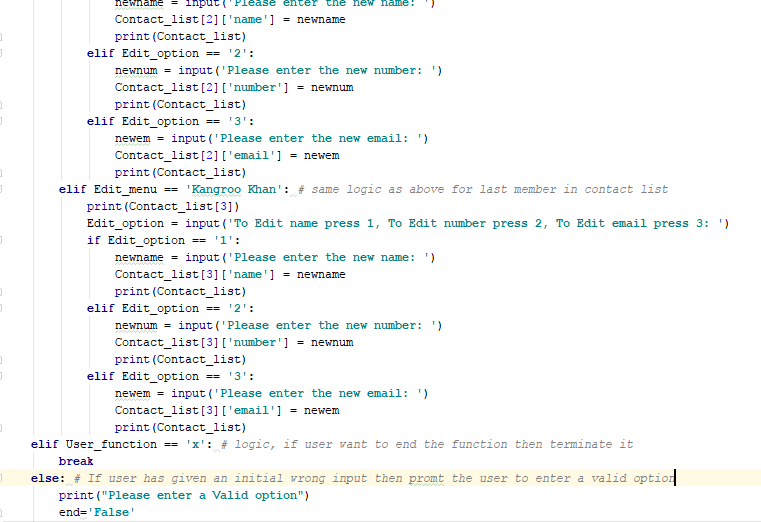


Q2) Contact list/ Edit Functionalities:

1. Code:

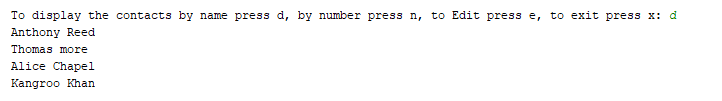




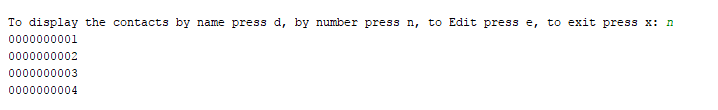


Results:

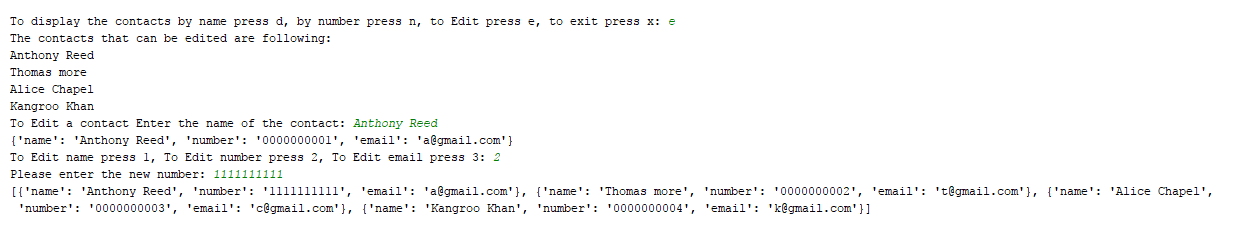
1. Display contacts by name:



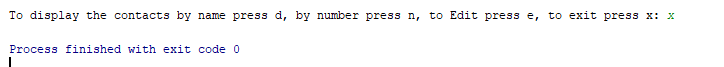
1. Display contacts by number:



1. Edit by name:

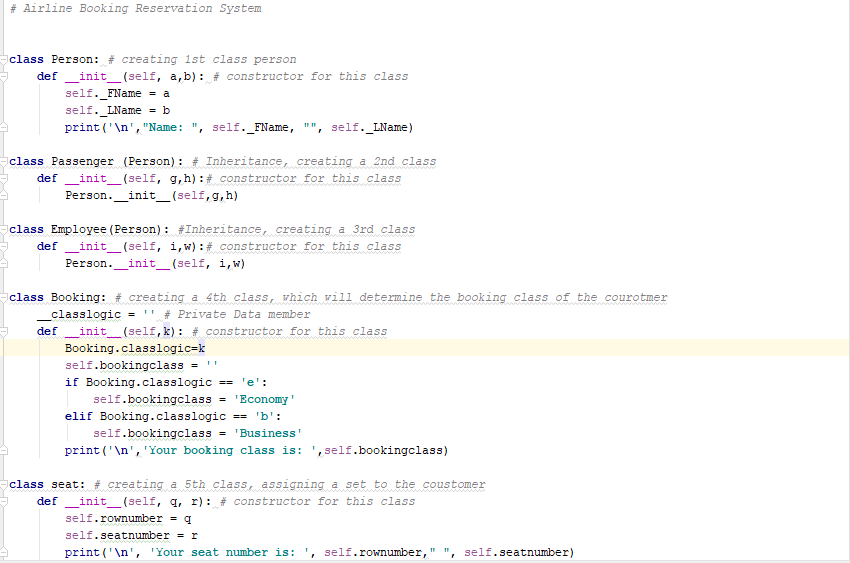


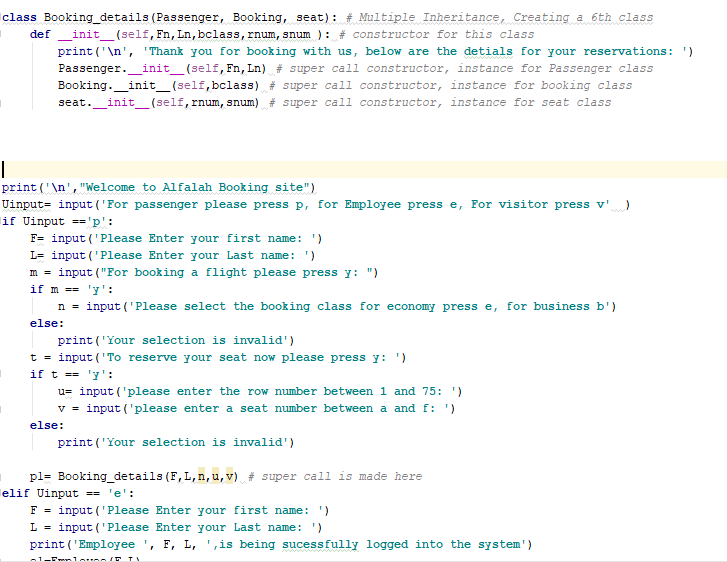
1. Exit:

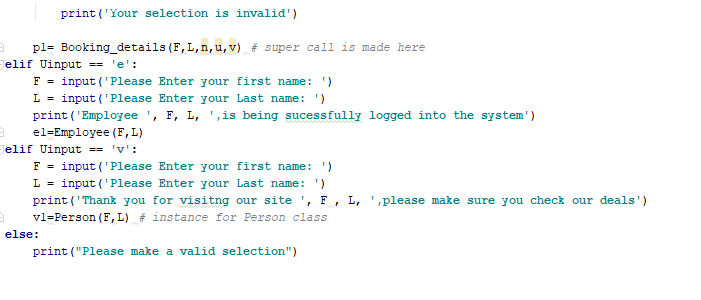


Q3) Airline Management system:

Code:

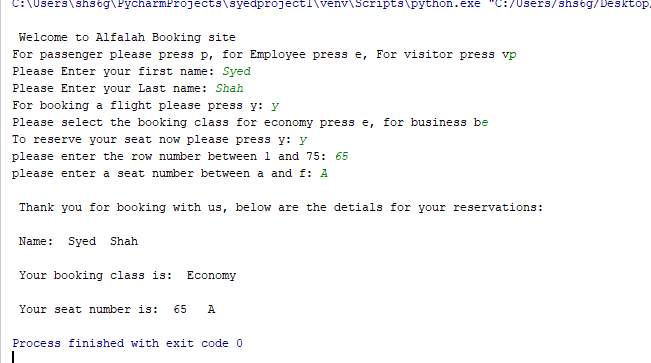




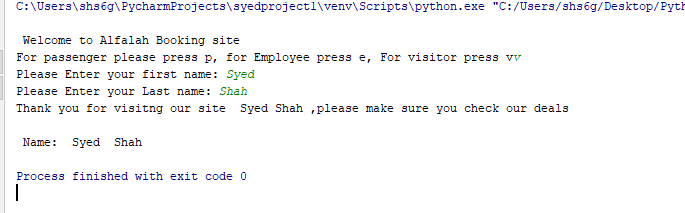


Results:

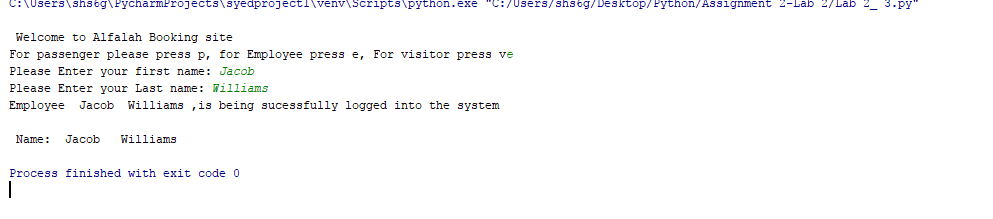
1. Passenger booking a flight:



1. A visitor visiting the airline site:

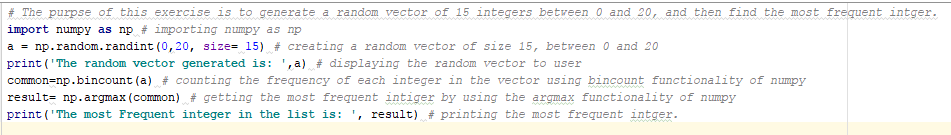


1. An Employee:

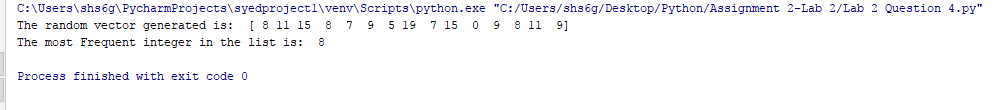


Q4) Generating a random vector of size 15(range 0 to 20) and finding the most frequent integer in it:

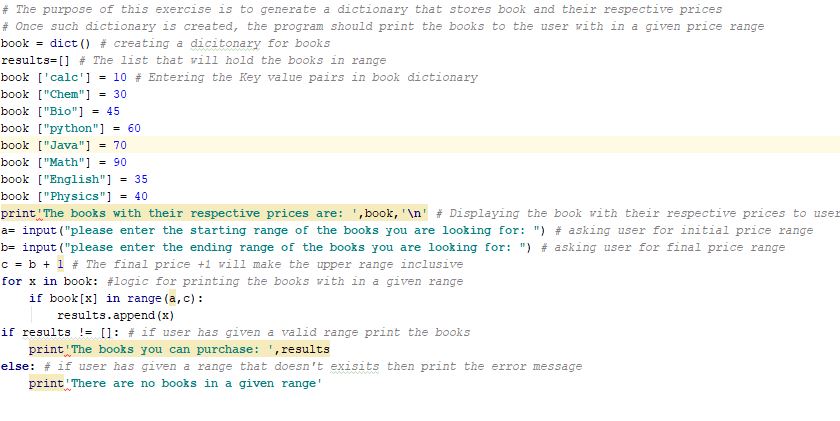
Code:



Results:



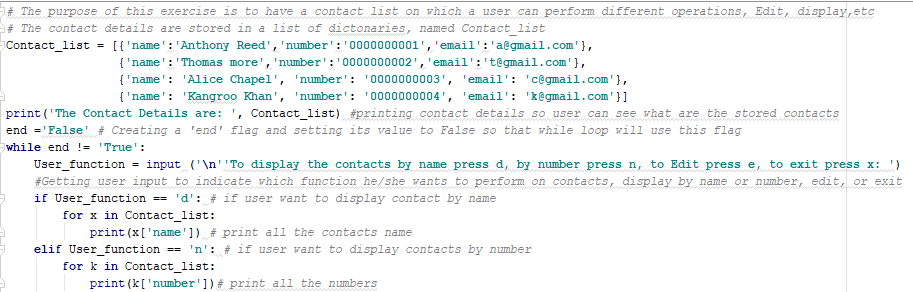
**Explanation of the implementation including code snippet**

Q1) :Books in a Dictionary:

The program starts by creating an empty dictionary named “book”. After that it stores some books and their prices in key, value pair fashion with in that dictionary. Then, it represents the list of books along with their respective prices to the users and ask them to insert a starting and ending price range for which, they intend to buy the books and stores these values in the variable “a” and “b” respectively. Then it creates a variable “c” that adds 1 to the value of variable b, so that to include the upper limit provided by the user with in the range function. Finally, the program uses the for loop and in function of python to print the list of the books that are within the range of the user. If, it doesn’t find any such list the program will display an error message and will terminate.

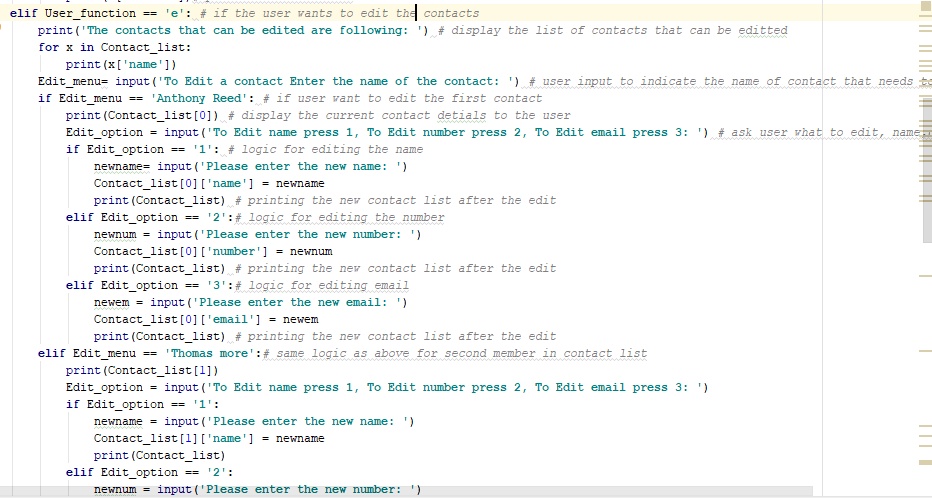
Q2) Contact list/ Edit Functionalities:

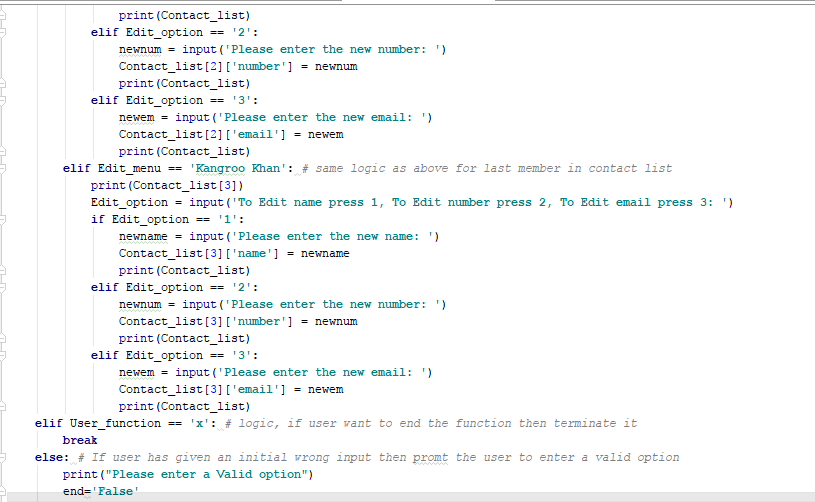
1. Start, display contact by name/ by number functionality:



The program starts by creating a list of dictionaries that stores contact’s name, number and email. This list is called “Contact\_list” and it has four contacts details. The program then prints this information to the user and asks the user if he/she wants to display the contacts by name (by pressing d) or by number (by pressing n) or want to edit the contact (by pressing e) or just want to exit the program (by pressing e). If the user press d the program will display the contact by name. A for loop is use for this functionality. If the user press n the program will display the contact by number. Again, a for loop is use for this functionality

1. Edit mode / exiting the program:



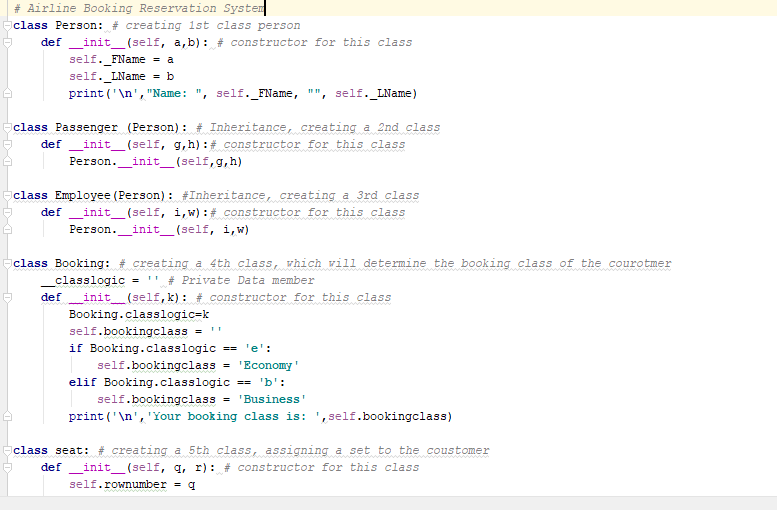


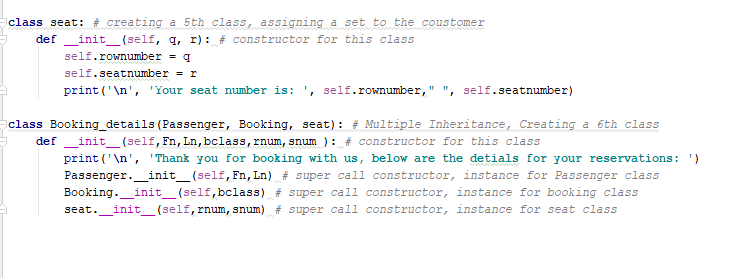
If the users press e the system enters an edit mode and displays the names of the contacts that can be edited and ask the user to provide the name of the contact that he/she intends to edit. Once the user provides the name of the contact the code displays that user information to the user and asks from the user what needs to be edited, name, number, or email. If user presses 1, the system asks for new name, if user enters 2 the program asks for the new number and if user enters 3 the code asks for the new email. Once the edited information is provided the system displays the edited contact list to the user. If/else statements are using to make this logic and it is applied to all the four contacts in the list.

If the user wants to exit the system, he/she can simply press the x in the menu option and the code will break and system will terminate.

Q3) Airline Management system:

Creation of 6 classes and their relation:





The program starts by creating 6 classes. A person class that stores the first name and last name of a person, who could be a passenger, an employee or visitor. At the last it prints the first name and last name of that person.

A passenger class, that is a child class and inherits a parent Person class. This class takes its values from a super class called Booking details and pass these values, first name, and last name, to the parent Person class.

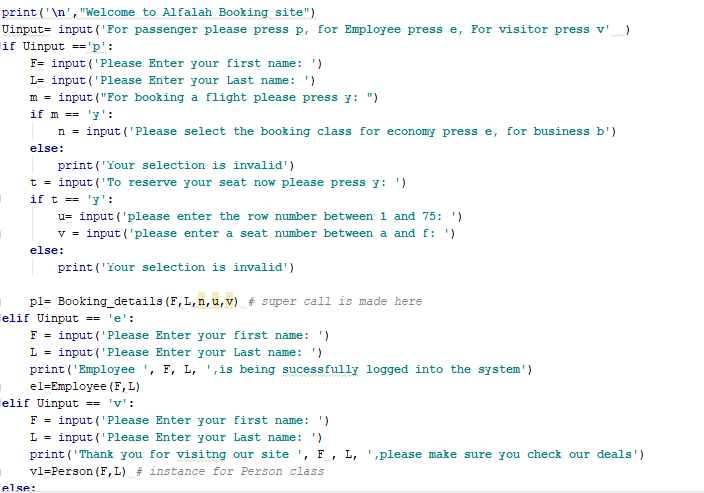
An Employee class, that is a child class and inherits a parent Person class. This class takes its values from a user input and pass these values, first name, and last name, to the parent Person class.

A Booking class, this class takes its values from super class called Booking details and stores this value into a private data member called ‘\_\_classlogic’. This class then analyze the value in this data member. If the value is e, then it determine the class to be economy, and if the value is b then the it establishes that passenger wants to travel in the business class. At the end it prints the traveling class selected by the passenger.

A seat class, this class takes its values from super class called Booking details’. This class stores the seat allocation, row number and seat number selected by the customer and prints this at the end.

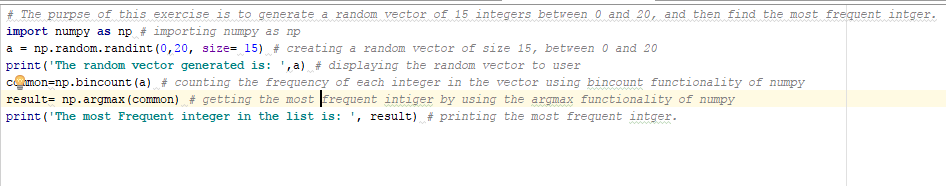
A super class Booking\_details. This class has multiple inheritance and has Passenger, seat and Booking as its parent classes. This class takes 5 values from user input and send two of them to Passenger class (First name and Last name), one to the Booking Class (Traveling class economy or business selected by user), and two to seat class (row number and seat number).

User interface:



In the user interface the code as from user weather he is a passenger or employee or visitor. If user is a passenger, the program will ask for the first and last name and will store it in two variables. Then the program will ask for the booking class and will store the value in a variable. Finally, the program will ask for the seat allocation, row number and seat number and will store these inputs into 2 variables. So far, the program has stored 5 user inputs and at this point the code will make a super call and will send these 5 variables to the super class Booking\_details, which will further pass these values to the 3 of its parent classes, Passenger, Booking, seat. If the user specified himself/herself as a visitor or employee the program will store their first name and last name and will send it to Person or Employee class respectively and will print an acknowledgement message.

Q4) Generating a random vector of size 15(range 0 to 20) and finding the most frequent integer in it:



In this program NumPy functionality is imported and a random vector of integers of size 15(range 0 to 20) is generated and stored in a variable called ‘a’. Then the program print this vector to the user. After that using bincount functionality of NumPy, the code counts the frequency of each integer within the vector and store these values in a variable called common. Lastly the code uses argmax functionality of NumPy and finds the integer that has the highest frequency and stores into a variable called ‘result’, which is printed at the end.

**Code:**

**Q1)**

*# The purpose of this exercise is to generate a dictionary that stores book and their respective prices  
# Once such dictionary is created, the program should print the books to the user with in a given price range*book = dict() *# creating a dicitonary for books*results=[] *# The list that will hold the books in range*book [**'calc'**] = 10 *# Entering the Key value pairs in book dictionary*book [**"Chem"**] = 30  
book [**"Bio"**] = 45  
book [**"python"**] = 60  
book [**"Java"**] = 70  
book [**"Math"**] = 90  
book [**"English"**] = 35  
book [**"Physics"**] = 40  
print**'The books with their respective prices are: '**,book,**'\n'** *# Displaying the book with their respective prices to user*a= input(**"please enter the starting range of the books you are looking for: "**) *# asking user for initial price range*b= input(**"please enter the ending range of the books you are looking for: "**) *# asking user for final price range*c = b + 1 *# The final price +1 will make the upper range inclusive***for** x **in** book: *#logic for printing the books with in a given range* **if** book[x] **in** range(a,c):  
 results.append(x)  
**if** results != []: *# if user has given a valid range print the books* print**'The books you can purchase: '**,results  
**else**: *# if user has given a range that doesn't exisits then print the error message* print**'There are no books in a given range'**

**Q2)**

*# The purpose of this exercise is to have a contact list on which a user can perform different operations, Edit, display,etc  
# The contact details are stored in a list of dictonaries, named Contact\_list*Contact\_list = [{**'name'**:**'Anthony Reed'**,**'number'**:**'0000000001'**,**'email'**:**'a@gmail.com'**},  
 {**'name'**:**'Thomas more'**,**'number'**:**'0000000002'**,**'email'**:**'t@gmail.com'**},  
 {**'name'**: **'Alice Chapel'**, **'number'**: **'0000000003'**, **'email'**: **'c@gmail.com'**},  
 {**'name'**: **'Kangroo Khan'**, **'number'**: **'0000000004'**, **'email'**: **'k@gmail.com'**}]  
print(**'The Contact Details are: '**, Contact\_list) *#printing contact details so user can see what are the stored contacts*end =**'False'** *# Creating a 'end' flag and setting its value to False so that while loop will use this flag***while** end != **'True'**:  
 User\_function = input (**'\n''To display the contacts by name press d, by number press n, to Edit press e, to exit press x: '**)  
 *#Getting user input to indicate which function he/she wants to perform on contacts, display by name or number, edit, or exit* **if** User\_function == **'d'**: *# if user want to display contact by name* **for** x **in** Contact\_list:  
 print(x[**'name'**]) *# print all the contacts name* **elif** User\_function == **'n'**: *# if user want to display contacts by number* **for** k **in** Contact\_list:  
 print(k[**'number'**])*# print all the numbers* **elif** User\_function == **'e'**: *# if the user wants to edit the contacts* print(**'The contacts that can be edited are following: '**) *# display the list of contacts that can be editted* **for** x **in** Contact\_list:  
 print(x[**'name'**])  
 Edit\_menu= input(**'To Edit a contact Enter the name of the contact: '**) *# user input to indicate the name of contact that needs to be edited* **if** Edit\_menu == **'Anthony Reed'**: *# if user want to edit the first contact* print(Contact\_list[0]) *# display the current contact detials to the user* Edit\_option = input(**'To Edit name press 1, To Edit number press 2, To Edit email press 3: '**) *# ask user what to edit, name,number,email* **if** Edit\_option == **'1'**: *# logic for editing the name* newname= input(**'Please enter the new name: '**)  
 Contact\_list[0][**'name'**] = newname  
 print(Contact\_list) *# printing the new contact list after the edit* **elif** Edit\_option == **'2'**:*# logic for editing the number* newnum = input(**'Please enter the new number: '**)  
 Contact\_list[0][**'number'**] = newnum  
 print(Contact\_list) *# printing the new contact list after the edit* **elif** Edit\_option == **'3'**:*# logic for editing email* newem = input(**'Please enter the new email: '**)  
 Contact\_list[0][**'email'**] = newem  
 print(Contact\_list) *# printing the new contact list after the edit* **elif** Edit\_menu == **'Thomas more'**:*# same logic as above for second member in contact list* print(Contact\_list[1])  
 Edit\_option = input(**'To Edit name press 1, To Edit number press 2, To Edit email press 3: '**)  
 **if** Edit\_option == **'1'**:  
 newname = input(**'Please enter the new name: '**)  
 Contact\_list[1][**'name'**] = newname  
 print(Contact\_list)  
 **elif** Edit\_option == **'2'**:  
 newnum = input(**'Please enter the new number: '**)  
 Contact\_list[1][**'number'**] = newnum  
 print(Contact\_list)  
 **elif** Edit\_option == **'3'**:  
 newem = input(**'Please enter the new email: '**)  
 Contact\_list[1][**'email'**] = newem  
 print(Contact\_list)  
 **elif** Edit\_menu == **'Alice Chapel'**:*# same logic as above for third member in contact list* print(Contact\_list[2])  
 Edit\_option = input(**'To Edit name press 1, To Edit number press 2, To Edit email press 3: '**)  
 **if** Edit\_option == **'1'**:  
 newname = input(**'Please enter the new name: '**)  
 Contact\_list[2][**'name'**] = newname  
 print(Contact\_list)  
 **elif** Edit\_option == **'2'**:  
 newnum = input(**'Please enter the new number: '**)  
 Contact\_list[2][**'number'**] = newnum  
 print(Contact\_list)  
 **elif** Edit\_option == **'3'**:  
 newem = input(**'Please enter the new email: '**)  
 Contact\_list[2][**'email'**] = newem  
 print(Contact\_list)  
 **elif** Edit\_menu == **'Kangroo Khan'**: *# same logic as above for last member in contact list* print(Contact\_list[3])  
 Edit\_option = input(**'To Edit name press 1, To Edit number press 2, To Edit email press 3: '**)  
 **if** Edit\_option == **'1'**:  
 newname = input(**'Please enter the new name: '**)  
 Contact\_list[3][**'name'**] = newname  
 print(Contact\_list)  
 **elif** Edit\_option == **'2'**:  
 newnum = input(**'Please enter the new number: '**)  
 Contact\_list[3][**'number'**] = newnum  
 print(Contact\_list)  
 **elif** Edit\_option == **'3'**:  
 newem = input(**'Please enter the new email: '**)  
 Contact\_list[3][**'email'**] = newem  
 print(Contact\_list)  
 **elif** User\_function == **'x'**: *# logic, if user want to end the function then terminate it* **break  
 else**: *# If user has given an initial wrong input then promt the user to enter a valid option* print(**"Please enter a Valid option"**)  
 end=**'False'**

**Q3)**

*# Airline Booking Reservation System***class** Person: *# creating 1st class person* **def** \_\_init\_\_(self, a,b): *# constructor for this class* self.\_FName = a  
 self.\_LName = b  
 print(**'\n'**,**"Name: "**, self.\_FName, **""**, self.\_LName)  
  
**class** Passenger (Person): *# Inheritance, creating a 2nd class* **def** \_\_init\_\_(self, g,h):*# constructor for this class* Person.\_\_init\_\_(self,g,h)  
  
**class** Employee(Person): *#Inheritance, creating a 3rd class* **def** \_\_init\_\_(self, i,w):*# constructor for this class* Person.\_\_init\_\_(self, i,w)  
  
**class** Booking: *# creating a 4th class, which will determine the booking class of the courotmer* \_\_classlogic = **''** *# Private Data member* **def** \_\_init\_\_(self,k): *# constructor for this class* Booking.classlogic=k  
 self.bookingclass = **''  
 if** Booking.classlogic == **'e'**:  
 self.bookingclass = **'Economy'  
 elif** Booking.classlogic == **'b'**:  
 self.bookingclass = **'Business'** print(**'\n'**,**'Your booking class is: '**,self.bookingclass)  
  
**class** seat: *# creating a 5th class, assigning a set to the coustomer* **def** \_\_init\_\_(self, q, r): *# constructor for this class* self.rownumber = q  
 self.seatnumber = r  
 print(**'\n'**, **'Your seat number is: '**, self.rownumber,**" "**, self.seatnumber)  
  
**class** Booking\_details(Passenger, Booking, seat): *# Multiple Inheritance, Creating a 6th class* **def** \_\_init\_\_(self,Fn,Ln,bclass,rnum,snum ): *# constructor for this class* print(**'\n'**, **'Thank you for booking with us, below are the detials for your reservations: '**)  
 Passenger.\_\_init\_\_(self,Fn,Ln) *# super call constructor, instance for Passenger class* Booking.\_\_init\_\_(self,bclass) *# super call constructor, instance for booking class* seat.\_\_init\_\_(self,rnum,snum) *# super call constructor, instance for seat class*print(**'\n'**,**"Welcome to Alfalah Booking site"**)  
Uinput= input(**'For passenger please press p, for Employee press e, For visitor press v'** )  
**if** Uinput ==**'p'**:  
 F= input(**'Please Enter your first name: '**)  
 L= input(**'Please Enter your Last name: '**)  
 m = input(**"For booking a flight please press y: "**)  
 **if** m == **'y'**:  
 n = input(**'Please select the booking class for economy press e, for business b'**)  
 **else**:  
 print(**'Your selection is invalid'**)  
 t = input(**'To reserve your seat now please press y: '**)  
 **if** t == **'y'**:  
 u= input(**'please enter the row number between 1 and 75: '**)  
 v = input(**'please enter a seat number between a and f: '**)  
 **else**:  
 print(**'Your selection is invalid'**)  
  
 p1= Booking\_details(F,L,n,u,v) *# super call is made here***elif** Uinput == **'e'**:  
 F = input(**'Please Enter your first name: '**)  
 L = input(**'Please Enter your Last name: '**)  
 print(**'Employee '**, F, L, **',is being sucessfully logged into the system'**)  
 e1=Employee(F,L)  
**elif** Uinput == **'v'**:  
 F = input(**'Please Enter your first name: '**)  
 L = input(**'Please Enter your Last name: '**)  
 print(**'Thank you for visitng our site '**, F , L, **',please make sure you check our deals'**)  
 v1=Person(F,L) *# instance for Person class***else**:  
 print(**"Please make a valid selection"**)

**Q4)**

*# The purpse of this exercise is to generate a random vector of 15 integers between 0 and 20, and then find the most frequent intger.***import** numpy **as** np *# importing numpy as np*a = np.random.randint(0,20, size= 15) *# creating a random vector of size 15, between 0 and 20*print(**'The random vector generated is: '**,a) *# displaying the random vector to user*common=np.bincount(a) *# counting the frequency of each integer in the vector using bincount functionality of numpy*result= np.argmax(common) *# getting the most frequent intiger by using the argmax functionality of numpy*print(**'The most Frequent integer in the list is: '**, result) *# printing the most frequent intger.*

**Deployment:**

1. Save the folder in your local machine.
2. Install Python 3.6.4 and PyCharm IDE on your machine.
3. Run PyCharm, click on file->open->files location of the saved folder.
4. Select the desired code file with .py extension to execute.
5. Right click on the code screen and then click on run "filename".
6. Give the input and validate the output.

**limitation**

1. In Q1, the program uses only a single dictionary (containing few books) rather than a whole inventory system.
2. The Program in Q1 also doesn’t arrange the books into different categories, instead it stores all of them in a single list.
3. The program, for displaying the contact list and edit features (Q2), lacks the functionality to add a new contact.
4. The program for Q2 (contact details and edit) also doesn’t check for invalid input by user when it is in edit mode. However, it does check for the invalid input by the user outside the edit mode.
5. In the airline management system (Q3), the program lacks the pictorial view of seat arrangements.
6. Q3 program is also not connected to any database, to verify the login credentials of the employee and therefor doesn’t ask the employees (or passengers) to provide the login credentials.
7. The program in Q3 also assumes that there are equal number of rows of seats in business and economy class (75). However, this is not the case and economy has far more seats than business.
8. The program in Q4, finds the most frequent integer in the list, however if there are more than one integer that have the same higher frequency, then the program will only print the first integer with higher frequency.

**References**

<https://stackoverflow.com/questions/26660654/how-do-i-print-the-key-value-pairs-of-a-dictionary-in-python>

<https://www.w3resource.com/python-exercises/numpy/python-numpy-random-exercise-13.php>

<https://www.reddit.com/r/learnpython/comments/30jcju/how_to_find_the_most_common_element_in_a_list/>

<http://cs231n.github.io/python-numpy-tutorial/>

<http://www.pythonforbeginners.com/dictionary/dictionary-manipulation-in-python>

<https://www.codecademy.com/en/forum_questions/505ba3cfc6addb000200e33c>

<https://stackoverflow.com/questions/20585920/how-to-add-multiple-values-to-a-dictionary-key-in-python/20585947>

<https://stackoverflow.com/questions/44916637/python-typeerror-must-be-str-not-int>

<https://www.codecademy.com/en/forum_questions/54fd72c9e39efe17d3003a6a>

<https://developmentality.wordpress.com/2012/03/30/three-ways-of-creating-dictionaries-in-python/>