New York University
Computer Science Department
Introduction to Software Development
Dr. Anasse Bari

Homework#7: PhoneBookApplication - *Classes, Objects, Inheritance, Polymorthism, Method Overriding and more on Object- Oriented Paradigm*

Deadline: *See NYUclasses for the deadline, 15% off per day after the deadline (3 days maximum).*

Learning Objectives:

- Practicing the object-oriented programming paradigm
- Practicing abstraction, encapsulation, inheritance, method overriding and polymorphism
- Practicing sorting
- Practicing File/IO in Java

Read the guidelines bellow carefully to avoid receiving a zero grade on the HW:

- **Please use the following naming convention "ExerciseX.java"**, where X is the number of the exercise. Please apply this convention to all coming homework assignments during the semester. Also, please attach the answers and include them into HW's zip file.
- Compile and run the program for each exercise.
- Comment Your Code (at least 5 comments per exercise)
- Make an archive (zip file or compressed file) with all the **java files (the .java files NOT the .class files)** and post it on NYU Classes (StudentName_Homework_X.zip)
- It is your responsibility to make sure that the Zip files has your actual files. You may send the file to yourself by email to double check that is the actual file before you upload on NYU classes.
- **If the graders cannot open the file, you will receive a grade of zero.**
- **If you send the .class files instead of the .java files (source files) you will receive a zero.**
- An act of cheating will be severely addressed with an immediate zero on the homework and a report to the academic advisor and the administration.
- You will automatically lose 50%of the points for an exercise if the program does not compile and run correctly.
- The number of points for each question is being specified.
- **Plagiarized assignment will get a ZERO grade**. You cannot change the variable names of other student's solution and submit it as yours. The program structure of other students must not match yours. Every student must come up with his/her own solution. Any

cheating (e.g. copying from internet without citing sources) is a serious violation of the University student code.

Exercise 1 (30 points)

Getting Started with Objects and Classes

- Define a class **named PhoneBookEntry. Each PhoneBookEntry has an Id (int), first name, last name, email, zip code and a phone number. In addition to the constructors, getters and setters (as covered in class), you will need to define a method that prints information (first name, last name…) of a PhoneBookEntry. The default Id is -1.**
- **Examples of Setters and Getters with signatures like:**
  - **public void setFirstName(String Fname)**
  - **public String getFirstName()**

  You will need to define the setters and getters for all private class members variables

- **You need to create following *constructors (you might need other constructors)* :**
  - **You will need to create a default constructor of the following signature**
    - **public PhoneBookEntry()**
  - **A Constructor that takes as arguments all attributes**
  - **A Constructor that accepts only firstName and phoneNumber**

- Define a method **printBookEntry()** that prints all the data fields (first name, last name, zip code, phone number..). This completes the definition of class **PhoneBookEntry**

- Next, you will need create a class called **PhoneBook** that instantiates three objects of type **PhoneBookEntry**. The attributes of each object are inputs from the user. This class will have the main method.

  (A) Object Attributes are as follows, you will need to create objects with the appropriate constructors accordingly:
  Create a class with the main method name it **PartOne where you instantiate three objects as follows: (the data should be given as parameters to the constructors)**
  PartOne is the class the has the main method. Create the following object using the appropriate constructors (you do not need to use scanner, you can hardcode the inputs)

  First Object: 5,John,Smith,jsmith@gmail.edu,20037,2023334444

  Second Object: James,202344344

  Third Object: Sarah

- Provide the code for the following in the main method of Phonebook – **You must comment your code.**

Demonstrate how you can use getters and setters to change the first name and phone number of attributes of objects in **(A):**

1. Change the phone number of John Smith to 202555555
2. Print the attributes of John Smith using the method PrintBookEntry().
3. Assign the Zipcode of John Smith to James's.

Create a folder for this exercise and include all your classes in the same folder and zip it under the name of HW7Exercise1

Exercise 2 (70points)

An Array of PhoneBookEntries, PhoneBookAdmin and NormalUser

- This part uses the PhoneBookEntry defined in Exercise#1
- The objective of this exercise is to create a phone book application using the object-oriented programming paradigm.

Create a class named **PhoneBookDirectory**. The PhoneBookDirectory has an array of 6 PhoneBookEntry' objects (for simplicity use size 6 so you can test your program).
Define in PhoneBookDirectory and use methods that allow to perform the following:

- **Add an entry to the phonebook**
  Adding an entry to the phone to the array in first empty space in the array.
  The method signature is as follows: (You must use this signature)

  public int addEntry(PhoneBookEntry entry)

  the method returns 1 if the object entry was successfully added or 0 otherwise (in case the array is full)
- Print all phonebook entries
- Search for an entry by Phone Number (Linear Search)
  public int LinearSearchByPhoneNumber(String PhoneNumber)

  return 1 if found 0 if not found
- Search for an entry by id (Using Binary Search)

  public PhoneBookEntry SearchbyIdBinarySearch(int id)

  returns a PhoneBookEntry if found
  return an empty PhoneBookEntry if NOT found
  Revisit the book and the notes on Binary Search.

- Sort Phone Book Entries by id(you can use selection sort for sorting – Do not use the built in .sort function defined in Java) – **You must use the comparable interface. Read Chapter 13th, section 13.6 the comparable interface and sorting comparable objects before doing this part.**

- Edit an entry (allows to edit any attributes (data fields) they want of a specific phone book entry of a given last name and firstname)

  public int Edit(String firstName, String lastName)

  return a 1 if the object was found and edited, returns 0 if the entry of the given userName and lastName do not exist.

- Delete an Entry of a given id, the method should return 1 if the Entry is deleted and 0 if not (not found). Deleting an entry from the array means you set all attributes of the entry's object to be deleted to default values (feel free to define your own default values – add comments about your default values)

  public int DeleteEntry(int id)
  return a 0 if the item of the id is being found and deleted, return a 1 otherwise.

PhoneBookAdmin and NormalUser

- Assume that there will be two types of users of the PhoneBookDirectory

  - PhoneBookAdmin
    has a username, password, emailAddress (string) and a PhoneBookDirectory

  - NormalUser
    has an id(int), username, password and PhoneBookDirectory

Both PhoneBookAdmin and NormalUser should inherit from a class called User that you need to create. The class User should have common attributes from both PhoneBookAdmin and NormalUser. The class User will have a method PrintUserInfo that prints the user name and password of a user object.

The method PrintUserInfo should be **overridden in both PhonBookAdmin and NormalUser that will allow their objects to print ALL attributes.**

A PhoneBookAdmin should be able to do the following:

- Add a phone entry
- Edit a phone entry of a given first name and last name
- Delete a phone entry of a given first name and last name
- Sort the PhoneBookDirectory
- Search using Linear Search
- Search using Binary Search
- Print Admin's info
- Change Password
- Change Username

A NormalUser should be able to do the following:

- Add a phone entry
- Edit a phone entry of a given first name and last name
- Sort the PhoneBookDirectory
- Search using Linear Search
- Print user's info

- Define an interface for the NormalUser that will have the methods to be implemented by NormalUser.

- Define an interface for PhoneBookAdmin that will have the methods to be implemented by PhoneBookAdmin

PhoneBookApplication class (the one that will have the main method)

- Initially, you should create an object of type PhoneBookAdmin and another one of type NormalUser.

- PhoneBookAdmin's object should have a username, password and email address. They should read from a file. Username, password and email address are separated by comma.

  You can choose a Username, password and email and save them into a file named: admin.txt

- NormalUser's object initially has an id, username, password. They should be read from a file (separated by comma).
  You can choose a Username, password and email and save them into a file named: user.txt
  Make sure that the paths are NOT independent. (the files should be located in the same folder as the project)

- Then you ask the user to login and you will need to check if the user is an admin or a normal user, then you display the menu accordingly.

- See the book chapter for File/IO, attend the lecture where we cover File I/O

Make sure all your functionalities mentioned in the menus are working before you submit and comment your code. Feel free to make you own assumptions in terms of the menu, and the way the program should work with the data entry.

Create a folder for this exercise and include all your classes in the same folder and zip it under the name of HW7Exercise2