BTE 324 – R

Hyun Kim

Midterm Exam – Banking System (ATM)

For the midterm, I will construct a banking system using Java. The banking system will be designed to be a database including people with their **name, SSN, date of birth, checking balance, savings balance and account number** as their attributes along with an **auto-incremented ID**. The system will primarily have features of an **ATM**, in which the user would be able to access their account using their last name and last 4 digits of their account number. The user would then be able to check their balance, deposit or withdraw money into either their checking or savings with the receipt of the transaction printed.

As for the program specifics, an interface called **Person** will be made and have the getter methods for a person’s name, SSN and date of birth as the interface methods. A class called **PersonImpl** will then be made to implement the **Person** interface and contain the basic attributes of their name, SSN and date of birth with respective getters (as specified in the **Person** interface), setters and constructors. Then the class **PersonImpl** will then be inherited (extended) by another class called **Client** with additional attributes of a checking balance, savings balance and account number with getters, setters and constructors for the additional attributes.

As for the client/person data, a random function will likely be used to create 100 datasets for this program. Then the file containing the dataset will be loaded into the program in an **ArrayList** and sorted by date of birth. Through a search function, the user then can log in by providing their last name and last 4 digits of their account number, as stated above. The program will then output the transaction receipt depending on the user’s choice (check balance, deposit, withdraw) on both the console and an output file named with the date of the transaction and the last name of the person. If the user deposits or withdraws money, the user’s balance will be updated using the setter functions. Also, the program will not allow a user to withdraw money if it will result in a negative balance; for example, if a user has $300, but attempts to withdraw $500.

Recursion will not be used, unless necessary. Functions (also the run() function) will be used for the program to have a more organized and shorter main.