INFO-C 210 (Problem Solving and Programming I)
Homework#3 – 50 points
No late submission **Group Assignment**

Solve the following programming problems: Keep in mind that these exercises require careful reading of the related module sections in Canvas and the related chapters. Make sure to document your code very well and follow proper programming practices.

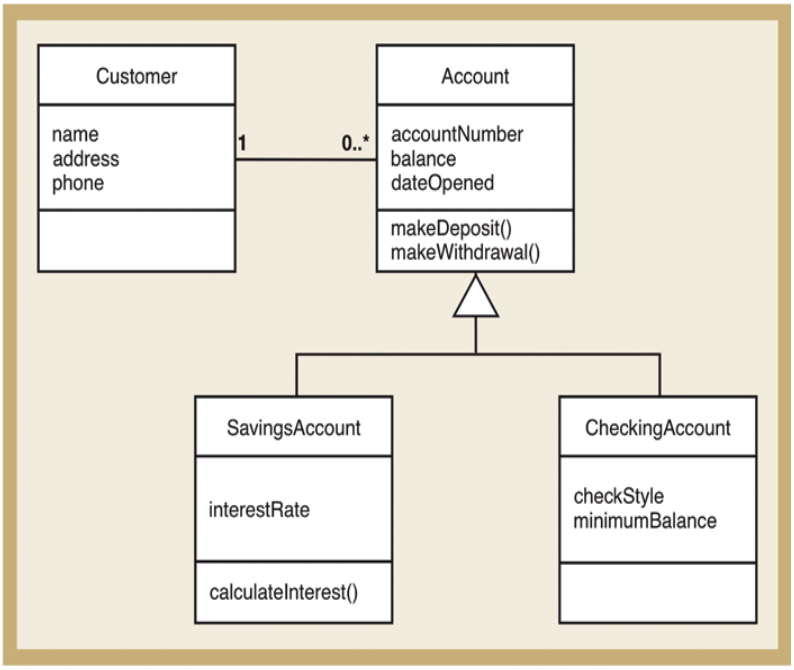
Grading rubric for this assignment is provided

Problem 1 (Objects, Aggregation, and Inheritance, Polymorphism)

Consider the following problem specification:

A bank system needs to store information about bank Accounts and Customers. The bank supports two different types of accounts (Checking and Savings). All bank accounts have account Number, balance, and date opened. Two Operations are defined for all accounts, *makeDeposit()* and *makeWithdrawal().* Checking accounts have additional attribute for check style and minimum balance. Saving accounts have additional attribute for interest rate and an operation for *calculateInterest().* All customers have a name, an address, and a phone number. In addition, a customer can have as many accounts as he needs.

The following is a class diagram for the Bank system described above:



The above specifications have been expanded with new requirements as follows:

There are two special types of customers (Personal and Commercial). Commercial customers have additional attributes for credit rating, contact person, and contact person phone. Personal customers have attributes for home phone and work phone. Moreover, expand the model to show that the bank has multiple branches, and each account is serviced by one branch. Naturally, each branch has many accounts.

The task:

1- Create a new Class Diagram for the complete system and submit this diagram.
2- Implement all classes in your diagram using the Java programming language. Implement the relationships as shown in your diagram.
3- Create a simple test program. The name of this test program is ***Bank.java***, it should make use of the above classes. *Bank.java* should declare an ArrayList to hold all kinds of bank accounts. The test program should utilize the system capabilities; The following are sample operations that demonstrate the systems capabilities:
    a. Create a Checking account for a commercial customer in Chicago's branch and add it to the array list
    b. Create a separate method to display the customer information and account balance. Call the method on behalf of the customer you created in the previous step.
    c. Deposit a $100 into the account you created in 'a', and then display the new info.
    d. Create a Savings account for an individual customer in some branch with initial balance of $100 and interest rate of 10% and add it to the array list.
    e. Display the savings account information
    f. Make a $100 deposit to the savings account, calculate the interest, then display the information
    g. Implement other operations of your choice!

Problem 2 (Objects, Aggregation, and Inheritance, Polymorphism)

Consider the following problem specification:

The system stores information about two things: Cars and CarOwners. A car has attributes for make, model, year and vin number. The owner has attributes for name and address. Assume that a car must be owned by one owner, and an owner can own many cars, but an owner might not own any cars (perhaps the owner just sold them all, but we still want a record of that owner in the system). The system should account for two different kinds of cars, Sports cars and Sedan cars. A sport car has additional attributes for race stats, represented by integer value. A sedan car has attributes for number of doors and trunk size, also represented by integer values.

(Note: Before you begin, you may find the example about Employees in Section 4.11 useful!)

1- Design and draw a UML class diagram, including classes, class attributes, and class relationships (composition, aggregation, inheritance, etc.)

2- Implement all classes in your diagram using the Java programming language. Implement the relationships as shown in your diagram. Include a setter and getter methods for each attribute in each class. Also, make sure to use the *toString()* method whenever appropriate to translate the object's state into text. Add other methods as necessary.

3- Create a test program named ***UseCars*** that should make use of the above classes and displays some results. In this test application do the following:

   a. Create an ArrayList that stores references of any type of cars
   b. Create a method that request information about a new car owner. The method will collect this information and returns a reference of a newly instantiated owner.
   c. Create another method that requests information about a sport car. The method will instantiate a sports car and adds it to the array list. Create another method that does exactly the same but for sedan cars.
   d. Create a method that would find and display the information about a given care based on a provided VIN number.
   e. In your main method or any method that acts as a main method. Call the methods that you just created. In other words, create a few car owners, create a few cars of each kind. Search for a particular car and display its information, and finally display all the cars that are currently stored in the list.

Questions about This Assignment:

- For any questions about this assignment, please post your questions in this Module's Open Discussion forum in Canvas. Please check this forum regularly and feel free to respond to other student's questions (I highly encourage and appreciate active participation in discussion forums). You are not allowed to include your entire code in the forum, if you feel that you must show your progress in a particular question, you can include your file(s) and submit that to me directly.

Submission Guidelines (Read Carefully)

- This is a group assignment
- Include all your solution files (only the ***.java** files, no .class files) in a Zip archive. Submit only the Zip file. If you do not hand in all *.java files, you will receive no credit for the submission. Also, include the UML diagrams (if any) in this Zip archive
- Submit your assignment via the "Submit Assignment" link. Don't email it to me directly
- Each exercises/projects/cases must be done in accordance to coding standards, as discussed in class. You may not just put something together' and exclaim; "but it works!" That's not acceptable!