Banking Application Version 1.4

- This assignment is SOLO, but for discussion you can form as a group (2 to 3 members). This means you may not look at the other student's code or let another student's look at your code.

Hope you had a good understanding of the core elements and basic Object Oriented Concepts such as Encapsulation (classes, attributes, operations on the attributes and objects of the classes that you create), Inheritance, Polymorphism and Abstraction.

Note: This assignment is designed to reinforce your understanding of how to design the software at a medium scale using object oriented programming concepts such as **encapsulation, inheritance, polymorphism, abstraction and to clearly understand the aggregation, composition, associations and collections API.**

Banking Application version 1.4: Write a java program to implement Banking Application based on the following requirements.

Till now we discussed and implemented the different accounts and with different concepts. Now this is the time for us to use the collections API.

1. Create multiple accounts for customers. The bank wants to find out about their esteemed customers who have higher amounts in their accounts. The bank also wants to find out the accounts which are non-performing (no debit and credit operations are performed with the accounts).
   Things to ponder:
   a. How do you sort the accounts based on the amounts?
   b. Where are the different accounts of the customers that have been maintained in your application (Which class is holding all the accounts)?
   c. What interfaces and which methods of those interfaces should be implemented to the class?
   d. Do you need to implement all the methods of these interfaces that you choose?
2. Write two or more test cases that have both input and output and check whether all your test cases are passed or not. You can read the main method given in the Account class (Banking application version 1.0) on how the test cases are read and how the outputs are formatted and printed on the console.
3. Write the main method in a separate .java file (Example: AccountTest.java) on your own to test your code. Reach out to your mentor If you need further assistance on how to write the test cases and validate your code against the test cases.

Note:

- Based on the above information, gather the ideas from your team and discuss with your mentor in the afternoon session. Once done you can start writing code for the above requirements.
- Please reach out to your mentor/course in-charge for further questions/clarifications.

- Once done, create a zip file of your code and submit on the Autolab. Wait for the mentors to give feedback on your deliverable. You can request your mentors for the feedback if it is getting delayed.