In-class lab

A bank needs a program to help it calculate the charges of a commercial chequing account.

Design a class called **BankCharges**. Include appropriate Javadoc comments for each class element.

Declare these symbolic constants at the top of your class

public static final double FIRST\_CHEQUES\_CHARGE = 0.10;

public static final double SECOND\_CHEQUES\_CHARGE =0.08;

public static final double THIRD\_CHEQUES\_CHARGE = 0.06;

public static final double FOURTH\_CHEQUES\_CHARGE = 0.04;

public static final int BANK\_CHARGE = 10;

public static final int BANK\_LOW\_BALANCE\_CHARGE = 15;

public static final int LOW\_BALANCE\_LIMIT = 400;

public static final int FIRST\_CHEQUES\_LEVEL=20;

public static final int SECOND\_ CHEQUES \_LEVEL = 40;

public static final int THIRD\_ CHEQUES \_LEVEL = 60;

Here are the relevant attributes of a BankCharges object. These must be the **only** fields of your class.

* clientName
* endingBalance
* numberOfCheques

Provide a constructor that does not take any parameters, one which initializes all instance variables to 0, except that the String field clientName should get (“unknown”) instead of null.

The second constructor takes parameters to initialize all the instance variables. The constructor validates the passed parameter making sure that endingBalance and numberOfCheques get positive values otherwise an IllegalArgumentException will be thrown with a message explaining what the problem is. Also make sure that clientName does not get null, if the passed parameter was null then an IllegalArgumentException will be thrown with a message explaining what the problem is.

The constructor signature is:

**public BankCharges(String cName, double cendingBalance, int cNumberOfCheques)**

Provide an accessor (get) and a mutator (set) method for each instance variable. The mutators must validate the numeric parameters and use them only if they were not negative, if the passed parameter was negative an IllegalArgumentException will be thrown with a message explaining what the problem is. The String filed mutator will validate the String parameter and uses it only if it was not null, if the passed parameter was null an IllegalArgumentException will be thrown with a message explaining what the problem is

Provide a method called **calculateBankServiceFees()**. This method will **return** the service fees of the bank calculated as follows:

* The bank charges $ 10 a month for each chequing account plus the following fees:
  + $ 0.10 for each cheque if less than 20 cheques were written
  + $ 0.08 for each cheque if 20 through 39 cheques were written
  + $ 0.06 for each cheque if 40 through 59 cheques were written
  + $ 0.04 for each cheque if 60 or more cheques were written
* The bank also charges $15 if the account balance falls below $400 **(before any cheque fees are applied)**.

The bank service fees should be deducted from the ending balance before returning the service fees.

Demonstrate your completed project to your instructor or TA before leaving the lab and be sure we have checked it off. A suggested solution will be given during the next class and labs that have not been checked off will not receive any points.