Account Inheritance Hierarchy

Create an inheritance hierarchy that a bank might use to represent customer’s bank accounts. All customers at this bank can deposit(i.e., credit) money into their accounts and withdraw (i.e., debit) money from their accounts. More specific types of accounts also exist. Savings accounts, for instance, earn interest on the money they hold. Checking accounts, on the other hand, charge a fee per transaction.

Create class Account and classes CheckingAccount and SavingsAccount which descend from Account. Account should include one private instance variable of type decimal to represent the account balance. The class should provide a constructor that receives an initial balance and uses it to initialize the instance variable. Be sure to validate the initial balance to ensure that it’s greater than or equal to 0.0. The class should provide two public methods. Method credit should add an amount to the current balance. Method debit should withdraw money from the Account and ensure that the debit amount does not exceed the Account’s balance. If it does, the balance should be left unchanged, and the method should display the message “Debit amount exceeded account balance” The class should also provide a get method for the Balance that returns the current balance

SavingsAccount >> descends from Account. A constant static (class) variable should be used for the rate of interest earned on all accounts. SavingsAccount’s constructor should receive the initial balance. SavingsAccount should provide a public method CalculateInterest that returns a decimal indicating the amount of interest earned by an account. This is determined by multiplying the interest rate by the account balance. [note: SavingsAccount should inherit methods Credit and Debit without redefining them]

CheckingAccount should inherit from Account and include a decimal static (class) variable that represents the fee charged per transaction. CheckingAccount’s constructor should receive the initial balance. Class CheckingAccount should redefine methods Credit and Debit so that they subtract the fee from the account balance whenever either transaction is performed successfully. CheckingAccount’s version of these methods should invoke the Account version to perform the updates to an account balance. CheckingAccount’s debit method should charge a fee only if money is actually withdrawn (i.e., the debit amount does not exceed the account balance. [HINT: Define Account’s debit method so that it returns a bool indicating whether money was withdrawn. Then use the return value to determine whether a fee should be charged. ]

After defining the classes in this hierarchy write an app that creates objects of each class and test their methods. Add interest to the SavingsAccount object by first invoking its CalculateInterest method, then passing the returned interest amount to the object’s Credit method.