Assignment Title: Bank Management System

Objective: To assess students' understanding and ability to apply the concepts of encapsulation, inheritance, polymorphism, and formatting in C++ programming to create a bank management system

Encapsulation:

a. Define a class called "Customer" with the following private attributes:

i. firstName (string)

ii. lastName (string)

iii. address (string)

iv. phone (string)

b. Create public member functions that allow users to:

i. Set and get the customer's first name, last name, address, and phone number.

ii. Print out the customer's information in a table-like format.

Inheritance:

a. Define a class called "Account" with the following private attributes:

i. accountNumber (string)

ii. accountType (string)

iii. balance (double)

iv. interestRate (double)

b. Create public member functions that allow users to:

i. Deposit money into the account.

ii. Withdraw money from the account.

iii. Check the account balance.

iv. Set the account type.

v. Calculate and return the interest earned on the account balance.

vi. Print out the account's information in a table-like format.

c. Create a derived class called "CheckingAccount" that inherits from the Account class.

d. Add a private attribute to the CheckingAccount class called "monthlyFee" (double).

e. Add a public member function that returns the monthly fee.

f. Override the printAccountInfo method to include the monthly fee in the table-like format.

g. Create another derived class called "SavingsAccount" that also inherits from the Account class.

h. Add a private attribute to the SavingsAccount class called "minimumBalance" (double).

i. Override the deposit and withdrawal methods from the Account class to check if the withdrawal would take the balance below the minimum balance. If so, deny the withdrawal and print an error message. If not, allow the withdrawal and subtract any applicable monthly fee from the account balance.

j. Add a public member function that returns the minimum balance.

k. Override the printAccountInfo method to include the minimum balance in the table-like format.

Polymorphism:

a. Define a virtual function in the Account class called "applyInterest" that applies the interest rate to the account balance.

b. Override the "applyInterest" function in both the CheckingAccount and SavingsAccount classes to account for the monthly fee and minimum balance, respectively.

c. Create an array of Account pointers that contains both Account, CheckingAccount, and SavingsAccount objects.

d. Loop through the array and call the "applyInterest" function for each object.

Bank Management System:

a. Create a class called "Bank" with the following private attributes:

i. customers (vector of Customer objects)

ii. accounts (vector of Account pointers)

b. Create public member functions that allow users to:

i. Add a new customer to the bank.

ii. Add a new account to the bank.

iii. Print out a list of all customers and their information in a table-like format.

iv. Print out a list of all accounts and their information in a table-like format.

v. Print out a list of all accounts for a specific customer in a table-like format.

vi. Deposit money into an account.

vii. Withdraw money from an account.

viii. Apply interest to all accounts.