Deadline: 6/6/2007, 8:30am

Object-Oriented Programming Homework-6

1. Create an inheritance hierarchy that a bank might use to represent customers' bank account. 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 type of account also exists. Saving accounts, for example, earn interest on the money they hold.

   Create an inheritance hierarchy containing base class Account and derived classes SavingsAccount that inherits from class Account. Base class Account should include one data member of type double to represent the account balance. The class should provide a constructor that receives an initial balance and uses it to initialize the data member. The constructor should validate the initial balance to ensure that it is greater than or equal to 0.0. If not, the balance should be set to 0.0 and the constructor should display an error message, indicating that the initial balance was invalid. The class should provide three member functions. Member function credit should add an amount to the current balance. Member function 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 function should print the message "Debit amount exceeded account balance." Member function getBalance should return the current balance.

   Derived class SavingsAccount should inherit the functionality of an Account, but also include a data member of type double indicating the interest rate (percentage) assigned to the Account. SavingsAccount's constructor should receive the initial balance, as well as an initial value for the SavingsAccount's interest rate. SavingsAccount should provide a public member function calculateInterest that returns a double indicating the amount of interest earned by an account. Member function calculateInterest should determine this amount by multiplying the interest rate by the account balance. Notice that SavingsAccount should inherit member functions credit and debit as is without redefining them. After defining the classes in this hierarchy, write a program that creates objects of each class and tests their member functions. Add interest to the SavingsAccount object by first invoking its calculateInterest function, then passing the returned interest amount to the object's credit function.

2. Develop a polymorphic banking program using the Account hierarchy created in the previous question. Create a vector of Account pointers to SavingsAccount. For each Account in the vector, allow the user to specify an account of money to withdraw from the Account using member function debit and an amount of money to deposit into the Account using member function credit. As you process each Account, determine its type. For SavingsAccount, calculate the amount of interest owed to the Account using member function calculateInterest, then add the interest to the account balance using member function credit. After processing an Account, print the updated account balance obtained by invoking base-class member function getBalance. [Hint: To achieve polymorphic behavior in the Account hierarchy, each class definition must declare the debit and credit member functions as virtual functions.]