COM203 Test 3. (25 Marks)

Create an inheritance hierarchy that a bank might use to represent customer's bank accounts. All customers at this bank can deposit money into their accounts (i.e. credit) and withdraw money from their accounts (i.e debit). 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 (i.e. debit or credit)

Create an inheritance hierarchy containing base class **Account** and derived classes **Savings_Account** and **Checking_Account** that inherit from class **Account**. Base class **Account** should include one data member of type float to represent 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 (methods). 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 "Insufficient funds!". Member function **get_balance** should return the current balance.

Derived class **Saving_Account** should inherit the functionality of an **Account**, but also include a data member of type float indicating the interest rate (percentage) assigned to the Account. **Savings_Account**'s constructor should receive the initial balance, as well as an initial value for the **Saving_Account's** interest rate. **Savings_Account** should provide a public member function **calculate_interest** that returns a float value indicating the amount of interest earned by an account. Member function **calculate_interest** should determine this amount by multiplying the interest rate by the account balance

Derived class **Checking_Account** should inherit from base class Account and include an additional data member of type float that represents the fee charged per transaction. **Checking_Account's** constructor should receive the initial balance as well as a parameter indicating a fee amount. Class **Checking_Account** should redefine member functions credit and debit so that they subtract the fee from the account balance whenever either transaction is performed successfully.

Checking_Account's versions of these functions should invoke the base class Account's version to perform the updates to an account balance. **Checking_Account's** debit function 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 function so that it returns a boll 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 a program that creates objects of each class and tests their member functions. Add interest to the **Savings_Account** object by first invoking it's **calculate_interest** function, then passing the returned interest to the object's **credit** function