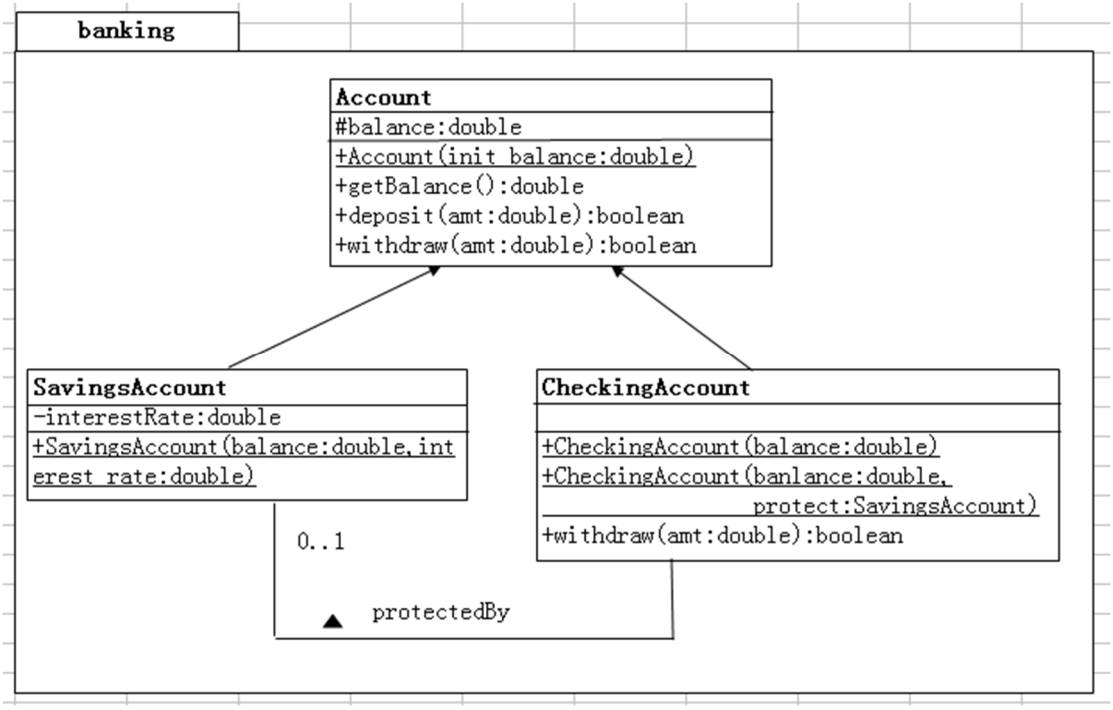
实现更为复杂的透支保护机制

注释-这是练习 1 的选择练习。它包括了更为复杂的透支保护机制模型。它使用客户储蓄。它使用客户储蓄账户完成透支保护。本练习必须在完成上述连个练习后进行



1．仿照练习 1“Account 类的两个子类”一节实现 SavingsAccount 类。

2．SavingsAccount 类必须扩展 Account 类。

3．该类必须包含一个类型为 double 的 interestRate 属性

4．该类必须包括一个带有两个参数(balance 和 interest_rate)的公有构造器。该构造器必须通过调用 super (balance) 来将 balance 参数传递给父类构造器

5．仿照练习 1“Account 类的两个子类”一节实现 CheckingAccount 类。

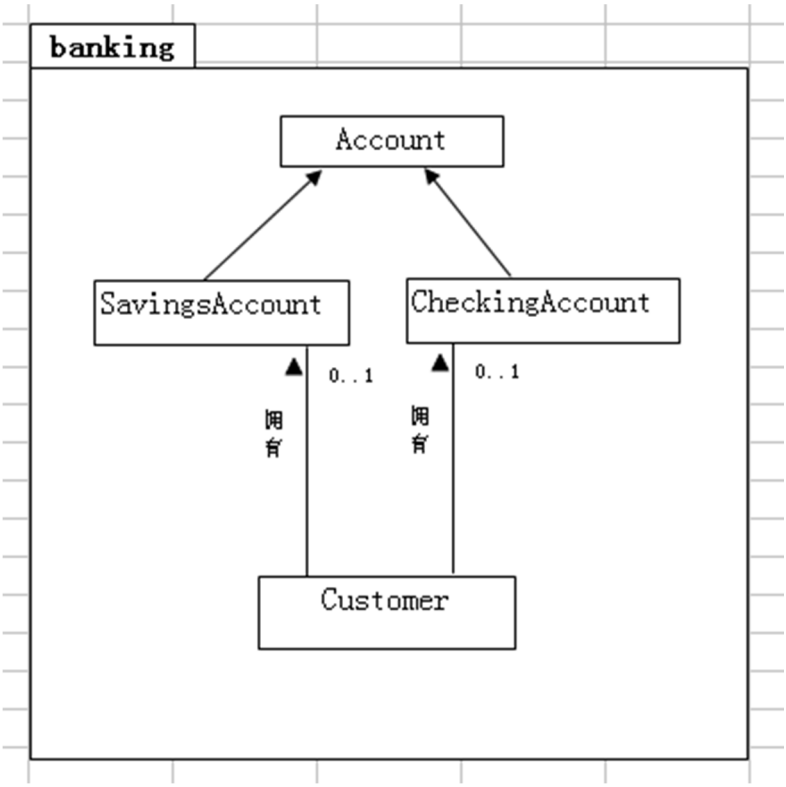
6．CheckingAccount 类必须扩展 Account 类

7．该类必须包括一个关联属性，称作 protectedBy，该属性的类型为 SavingsAccount，缺省值必须是 null。除此之外该类没有其他的数据属性

8．该类必须包括一个带有参数（balance）的公有构造器，该构造器必须通过调用 super（balance）将 balance 参数传递到父类构造器

9．该类必须包括另外一个带有两个参数的（balance 和 protect）的公有构造器。该构造器必须通过调用 super（balance）将 balance 参数传递给父类构造器。

10． CheckingAccount 类必须覆盖 withdraw 方法。该类必须执行下面的检查：如果当前余额足够弥补取款 amount，则正常进行。如果不够弥补但是存在透支保护则尝试用储蓄账户来弥补这个差值（balance-amount）。如果后一个交易失败，则整个交易一定失败，但余额未受影响。



修改客户使其拥有两个账户

修改 Customer 类使其拥有两个银行账户：一个储蓄账户和一个支票账户：两个都是可选的。

11．　原来的 Customer 类包含一个称作 account 的关联属性,可用该属性控制一个 Account 对象。重写这个类，使其包含两个关联属性：savingsAccount 和 checkingAccount，这两个属性的缺省值是 null

12．　包含两个访问方法：getSavings 和 getChecking，这两个方法分别返回储蓄和支票总数。

13．　包含两个相反的方法：SetSavings 和 setChecking，这两个方法分别为 savings 和 checking account 这两个关联属性赋值。

14．　编译执行 TestBanking 程序。输出应为：

Customer [Simms, Jane] has a checking balance of 200.0 and a savings balance of

500.0

Checking Acct [Jane Simms] : withdraw 150.00 succeeds? true

Checking Acct [Jane Simms] : deposit 22.50 succeeds? true

Checking Acct [Jane Simms] : withdraw 147.62 succeeds? true

Customer [Simms, Jane] has a checking balance of 0.0 and a savings balance of 42

4.88

Customer [Bryant, Owen] has a checking balance of 200.0

Checking Acct [Owen Bryant] : withdraw 100.00 succeeds? true

Checking Acct [Owen Bryant] : deposit 25.00 succeeds? true

Checking Acct [Owen Bryant] : withdraw 175.00 succeeds? false

Customer [Bryant, Owen] has a checking balance of 125.0