

实验 类的继承，super

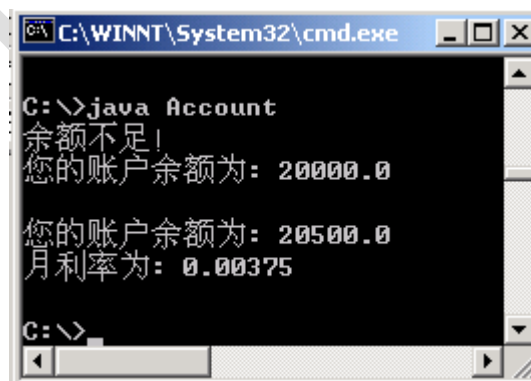
1、写一个名为 `Account` 的类模拟账户。该类的属性和方法如下图所示。该类包括的属性：账号 `id`，余额 `balance`，年利率 `annualInterestRate`；包含的方法：访问器方法（`getter` 和 `setter` 方法），返回月利率的方法 `getMonthlyInterest()`，取款方法 `withdraw()`，存款方法 `deposit()`。

Account
<pre>private int id private double balance private double annualInterestRate</pre>
<pre>public Account (int id, double balance, double annualInterestRate)</pre>
<pre>public int getId() public double getBalance() public double getAnnualInterestRate() public void setId(int id) public void setBalance(double balance) public void setAnnualInterestRate(double annualInterestRate) public double getMonthlyInterest() public void withdraw (double amount) public void deposit (double amount)</pre>

写一个用户程序测试 `Account` 类。在用户程序中，创建一个账号为 1122、余额为 20000、年利率 4.5% 的 `Account` 对象。使用 `withdraw` 方法提款 30000 元，并打印余额。再使用 `withdraw` 方法提款 2500 元，使用 `deposit` 方法存款 3000 元，然后打印余额和月利率。

提示：在提款方法 `withdraw` 中，需要判断用户余额是否能够满足提款数额的要求，如果不能，应给出提示。

运行结果如图所示：



```
C:\WINNT\System32\cmd.exe
C:\>java Account
余额不足!
您的账户余额为: 20000.0
您的账户余额为: 20500.0
月利率为: 0.00375
C:\>
```

2、创建 `Account` 类的一个子类 `CheckAccount` 代表可透支的账户，该账户中定义一个属性 `overdraft` 代表可透支限额。在 `CheckAccount` 类中重写 `withdraw` 方法，其算法如下：

如果（取款金额 \leq 账户余额），

可直接取款

如果（取款金额 $>$ 账户余额），

计算需要透支的额度

判断可透支额 `overdraft` 是否足够支付本次透支需要，如果可以

将账户余额修改为 0，冲减可透支金额

如果不可以

提示用户超过可透支额的限额

要求：写一个用户程序测试 `CheckAccount` 类。在用户程序中，创建一个账号为 1122、余额为 20000、年利率 4.5%，可透支限额为 5000 元的 `CheckAccount` 对象。

使用 `withdraw` 方法提款 5000 元，并打印账户余额和可透支额。

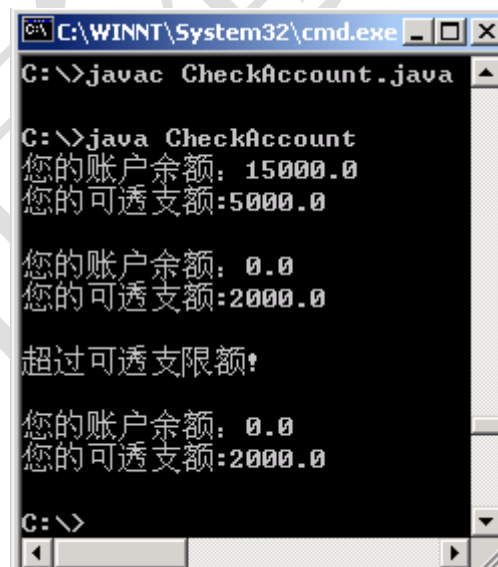
再使用 `withdraw` 方法提款 18000 元，并打印账户余额和可透支额。

再使用 `withdraw` 方法提款 3000 元，并打印账户余额和可透支额。

提示：

- （1）子类 `CheckAccount` 的构造方法需要将父类继承的 3 个属性和子类自己的属性全部初始化。
- （2）父类 `Account` 的属性 `balance` 被设置为 `private`，但在子类 `CheckAccount` 的 `withdraw` 方法中需要修改它的值，因此应修改父类的 `balance` 属性，定义其为 `protected`。

运行结果如下图所示：



```
C:\WINNT\System32\cmd.exe
C:\>javac CheckAccount.java
C:\>java CheckAccount
您的账户余额: 15000.0
您的可透支额:5000.0

您的账户余额: 0.0
您的可透支额:2000.0

超过可透支限额!

您的账户余额: 0.0
您的可透支额:2000.0

C:\>
```