Testing Report for Project 1

For project 1, we are asked to create four classes that extends Object: the BankAccount class, the CreditCardAccount class, the Account class, and the Date class. This testing report will test the validity of each method presented in each of these classes along with the testing codes in the interaction plane.

**Testing the class BankAccount**

*Testing the deposit method*

> BankAccount a=new BankAccount(0.15,10000,10.0,3.25,13.5)

> a.deposit(12000.0)

> a.getCurrentBalance()

12000.0

This is a direct copy of the interaction pane. As presented, we created a instance a with the reference type of BankAccount with 5 inputs, corresponds to the interest rate, the minimum balance, the overdraft fee, the ATM fee, and the bounced check. We call the deposit method with 12000.0 dollars added into the account, and we call the getCurrentBalance method will return the current balance stored in the account.

*Testing the withdraw method*

BankAccount a=new BankAccount(0.15,10000,10.0,3.25,13.5)

> a.deposit(10000)

> a.setWithdrawLimit(1)

> a.setWithdrawFee(5)

> a.withdraw(100)

true

> a.withdraw(400)

true

> a.getCurrentBalance()

9495.0

> a.setWithdrawLimit(0)

> a.withdraw(1)

true

> a.withdraw(1)

true

> a.withdraw(1)

true

> a.withdraw(1)

true

> a.withdraw(1)

true

> a.getCurrentBalance()

9490.0

> a.withdraw(1000000)

false

>

As presented above, we created an account with the balance of 10000. Then we employ the setWithdrawLimit and assign it to 1, and the withdraw fee is 5 if the number of monthly withdraws exceeds the limit.

In the three occasions, the withdraw method will extract money from balance as long as the withdraw money is less than the balance. When the monthly withdraw exceeds the withdraw limit, a 5 dollar fee is charged as shown in 9495 dollars. If the monthly withdraw limit is 0, we can withdraw multiple times.

*Testing the withdrawDraft method*

BankAccount a=new BankAccount(0.15,10000,10.0,3.25,13.5)

> a.deposit(1000.0)

> a.setWithdrawLimit(30)

> a.withdrawDraft(100000.0)

false

> a.getCurrentBalance()

986.5

If the user want to withdraw more money than what is remaining in the current balance，then the bank will charge with a bounced check fee of 13.5. As presented in the interaction pane, the current balance is deducted by 13.5 once we try to extract more.

*Testing the withdrawATM method*

BankAccount a=new BankAccount(0.15,10000,10.0,3.25,13.5)

> a.deposit(1000)

> a.withdrawATM(100)

true

> a.getCurrentBalance()

890.0

> a.withdrawATM(100000)

false

> a.getCurrentBalance()

890.0

>

For the withdrawATM method, the user will successfully extract money form the bank account if the balance have the grater amount than the sum of the ATM fee and the withdraw amount. As presented, the method return false when user tend to withdraw more than current balance, whereas ATM fee is applied along with the amount of withdraw in the current balance.

*Testing the incrementDay method and the incrementMonth method*

> BankAccount a=new BankAccount(0.15,10000,10.0,3.25,13.5)

> a.deposit(1000)

> a.incrementDay()

> a.getCurrentBalance()

996.75

> a.incrementMonth()

> a.getCurrentBalance()

996.75

> a.deposit(100000)

> a.incrementDay()

> a.getCurrentBalance()

100996.75

> a.incrementMonth()

> a.getCurrentBalance()

101038.25551369863

>

Since the two methods are void, there will be no output. On the other hand, the project instruction didn’t mention creating accessory to the instance fields, so we can call the current balance to see the result of this method.

In the first condition, the current balance1000 is less than the minimum balance 10000, so a 3.25 over draft fee is charged for the account. While the increment month have no change. If we add 100000 more into the account, the balance is going to be greater than the minimum balance, so no overdraft fee charged, and the balance remained the same by calling incrementDay method. But for the incrementMonth method, our interest is added.

**Testing the class CreditCardAccount**

*Testing the charge method*

CreditCardAccount a=new CreditCardAccount(100000,0.15,5000,100)

> a.charge(12000.0)

true

> a.getCurrentBalance()

12000.0

> a.charge(1000000.0)

false

This is a direct copy of the interaction pane. As presented, we created a instance a with the reference type of CreditCardAccount with 4 inputs, corresponds to the credit limit, the interest rate, the minimum payment, and the late payment penalty. We call the charge method with 12000.0 dollars added into the account, and we call the getCurrentBalance method will return the current balance stored in the account.

Our credit limit is set to 100000.0 when we created the instance a. Then,12000 is successfully added into the current balance with the return of true. The charge method return false when the sum of our charged input(1000000) and the current balance(12000.0) is more than the limit balance(100000).

*Testing the payment method*

CreditCardAccount a=new CreditCardAccount(100000,0.15,5000,100)

> a.charge(10000)

true

> a.payment(100)

> a.getCurrentBalance()

9900.0

> a.getMonthlyPayment()

100.0

>

To test the method payment, we store 10000 into the current balance of the credit card account. Then we called the payment method to extract 100 dollars. When we call the getCurrentBalance() and getMonthlyPayment() method, we can see that the 100 successfully withdraw from the current balance and stored into the monthly payment.

*Testing the incrementDay and incrementMonth method*

CreditCardAccount a=new CreditCardAccount(100000,0.15,5000,100)

> a.charge(0)

true

> a.incrementDay()

> a.incrementMonth()

> a.getCurrentBalance()

0.0

> a.getMonthlyPayment()

0.0

> a.charge(1000)

true

> a.getTotalPayment()

0.0

> a.payment(500)

> a.getTotalPayment()

500.0

> a.getMonthlyPayment()

0.0

> a.getCurrentBalance()

500.0

> a.incrementDay()

> a.getCurrentBalance()

500.0

> a.incrementMonth()

> a.getCurrentBalance()

500.2054794520548

> a.getTotalPayment()

0.0

> a.getMonthlyPayment()

500.2054794520548

> a.payment(10000)

> a.getCurrentBalance()

-9499.794520547945

> a.incrementDay()

> a.incrementMonth()

> a.getMonthlyPayment()

-9499.794520547945

>

This is a copy of the interaction pane. Since the method is void, there will be no output. On the other hand, the project instruction didn’t mention creating accessory to the private instance fields, so we can call the current balance to see the result of this method. The interest is added to the balance after calling the increment month method when the current balance is not payed. The total payment is what we paid and the monthly payment is what we need to pay.

I tested in three circumstances. The first circumstance is when the current balance is 0, and there will be no interest added into current balance and the monthly payment is zero. The second circumstance is when our current balance is 1000 and our total payment is 500, so there will be interest added after calling the incrementDay and incrementMonth method. As a result, our monthly payment is the sum of balance and interest the the total amount payed is 0. The last situation is when our total payment is greater than the account balance. There will be no interest added, our monthly payment method is called as it show a negative value which is the extra amount we paid.