University of HUDDERSFIELD

CFS2160: Programming Stream

Tutorial/Practical 8

Using Java Classes

Introduction

This week we will develop the Bank Account class from the lecture[1].

Purpose

Today we will develop the `BankAccount` class discussed in the lecture to protect the values in its instance variables, mainly the balance. We will then test it by developing a second class that creates and uses a single `BankAccount` object.

You saw all the code needed to do all this in the lecture this week.

If in doubt - Ask!

Activities

1. Make a new folder (package) in IntelliJ and create the lecture `BankAccount` class. There is a skeleton for your code on GitHub in the usual repo.
   You *will* need to change the package at the top of this class to match your own directory structure. This is important. If you do not do this, IntelliJ will be confused about which `BankAccount` class you are using later on.
   You need to create the instance variables, constructor, getters and setters, and so on. Most of this code you saw in the lecture.

2. Refactor the `deposit` method so that it protects the value of the balance (that is, it rejects deposits of zero or less). It should return a Boolean to indicate whether or not the deposit succeeded. Do the same with the `withdraw` method (which should also not allow withdrawals of zero or less) and also `addInterest` (which should not allow the interest rate to be zero or negative, and which should not apply interest to any negative balance).

3. Refactor the `withdraw` method so that the balance is not allowed to go below zero *unless* the customer has an overdraft[2].

---

[1] By the end of the module there will have been an awful lot of bank accounts. Sorry.
[2] You may well need one of "or", which is `||` or "and", which is `&&`.

4. In the same package (folder), implement a `BankAccountDemo` class. This class should create a BankAccount object, and should include examples of the methods in the class being used. This new class will just have a single `main` method.

5. (For those who want a challenge.)
   Add a third constructor to the BankAccount class. This should take a single parameter, which is the name of the customer. It should allocate a random account number - there is some code below that will help. The balance should be set to zero, and it should be assumed that the customer does not have an overdraft.

Possibly Useful Code[3]

```
package week08;

import java.util.Random;

public class BankAccountRandom {

    private String accountNumber;

    public BankAccountRandom () {
        this.accountNumber = generateAccountNumber ();
    }

    private String generateAccountNumber ()
    {
        String acc = "";
        Random rg = new Random ();

        for (int i = 0; i < 9; i ++) {
            acc += rg.nextInt (10);
        }

        return acc;
    }
}
```

[3] Yes, that method is private. It's private because we only want it to be used within this class.