Assignment 2: Inheritance & Polymorphism

Due Date: See Slate\ Dropbox.
Date: June 9th, 2015
Type: **Individual Assignment**
Weight: **5%**

Summary

Extend the operation of the ATM application developed in Assignment 1 to practice inheritance and polymorphism by adding two new classes and associated functionality. As part of development, follow an UML "thinking map" and verify all your assumptions line-by-line using the NetBeans debugger. You are advised to continue with your solutions for Assignment 1. However, if you choose to, you can elect to start with the Assignment 1 solutions provided in SLATE.

Submission checklist

1. NetBeans Project Folder(s) containing the project and its associated files (source files, resource files, data files, documentation etc. (as required))

Part I (50%) Extend the Account class with two new classes: *ChequingAccount* and *SavingsAccount* as shown in Appendix 1. DO NOT duplicate field variables in the two new classes as they will inherit the code from the Account base class. Ensure the visibility is set accordingly to allow the new classes access to the field variables defined in the Account base class. Override the appropriate methods to implement the requirements below:

1. Chequing Account:
    1. The chequing account has an overdraft limit of 500$. That means that users are allowed to withdraw up to 500$ more than they have in the account.
    2. Annual interest rate of **maximum 1%**.
2. Savings account:
    3. Cannot be overdrawn
    4. Annual interest rate of **minimum 3%**
    5. For every dollar deposited the bank automatically deposits half a dollar (e.g. the account is setup such that an employer/parent automatically contributes as well)

Part II. (50%) Allow the user to create accounts and add them to the Bank. The list of accounts shall be empty to start with and the user shall be able to fill it with accounts through a new "Create Account" menu option.

1. Instead of prompting for an account ID, the application shall display a main menu with the following options:
    a. Create Account (shows the account creation menu)
    b. Select Account. This option works as before by prompting for an account ID and allowing the user to withdraw, deposit and checking account balance. These operations should work unchanged through polymorphism.
    c. Exit (exits the application)

2. The Account creation feature shall allow the user to specify:
    a. Account ID – shall be verified to be valid (non-negative integer) and not conflicting with an existing account ID
    b. Initial Balance
    c. Annual Interest Rate
    d. Type of account: *chequing* or *savings.* Based on the type selected by the user create either a ChequingAccount or a SavingsAccount

Part III – BONUS (10%) In the account management menu add an option to print the list of transactions performed on the account since the application has started. Thus the account management menu shall have the following options. Extend the object-oriented design to accommodate this new functionality.

1. Check balance
2. Withdraw
3. Deposit
4. **Display Transactions**
5. Exit

Each transaction shall be display on a separate line with the following information

1. Transaction type (deposit/withdrawal)
2. Amount
3. Date

NOTE: The bonus functionality is not shown on the thinking map. To implement this bonus functionality the thinking map needs to be updated.

Appendix 1: Thinking Maps

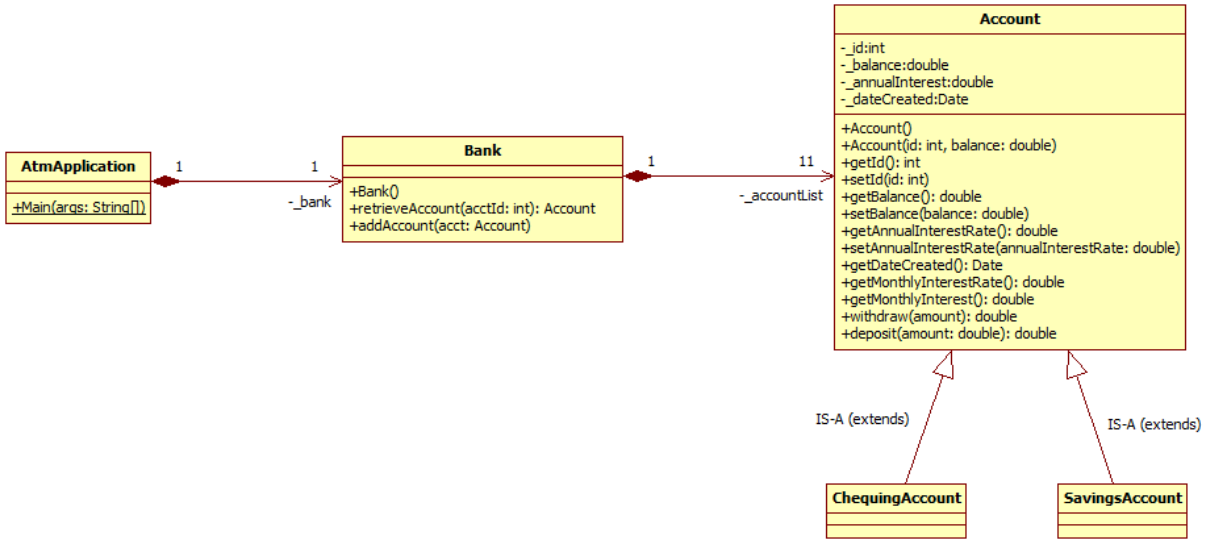

Figure 1: ATM Application Thinking Map

Notes:

1. The **professionalism of your submission**, clarity of written **communication** is extremely important. The ability to communicate your knowledge is as important as the knowledge itself.

    a. Up to 40% of the mark for a program can be deducted due to poor presentation / communication: quality of names according to our *naming and coding conventions (10%)*, *indentation (15%)* and *comments (15%)*.

2. **All assignment shall be submitted by the deadline.** Late submissions will be penalized with 10% per day for up to 3 calendar days after which the assignment cannot be submitted anymore. **An email must be sent** should you choose to submit a late assignment. If no such emails are received the solution will be posted. **Assignments are not accepted after the solutions have been posted.**

3. This assignment shall be **completed individually**. Remember that completing the assignment by yourself will ensure your success on the midterm and final exam. See the Academic Honesty at Sheridan.

4. Submission is done in electronic format **using DropBox.com. You will have to have accepted a DropBox share request to create a virtual drop-box folder for your assignment submissions. DO NOT email your submission.**