Due: Monday 15 February 2010

Topics: Class hierarchy design, object-oriented programming, using generic types.

Collaboration: This homework assignment is to be completed individually.

Submission: Follow the instructions for submitting programs via P-Web and handing in a printed copy.

In this assignment, we will use a bank’s accounts as a case study for some object-oriented design. In short, you are to write a bank class that maintains a variety of accounts, and prints monthly statements for each of them. That is, based on some criteria we define below, print the name associated with the account and its balance.

Our bank has three types of accounts: standard, interest-bearing, and platinum.

- Standard accounts bear no interest and have no fees.
- An interest-bearing account (obviously) garners interest. However, clients open their accounts at different times and thus may have different interest rates on their accounts.
- A platinum account also bears interest, typically at a higher rate than a regular interest-bearing account. However, it only garners interest when the balance is above a certain threshold. Current bank policy has this threshold at $10,000, but this will likely change with further swings in the market. Platinum accounts with balances below that threshold garner no interest. In addition, there is a monthly fee for this account—currently $5 but also subject to change.

Since we have a national bank, all accounts must also be associated with a particular U.S. state.

Design a class hierarchy for accounts with appropriate fields and methods so that your design exhibits good encapsulation and cohesion, with a minimum of repeated code.

Design a bank class that uses an ArrayList to keep track of its accounts. Include methods to:

- add an account
- add interest to all accounts that are interest bearing,
- calculate the total value of its assets, and
- print a statement (account-holder’s name, and new balance) for each account—don’t worry about handling rounding or only displaying two fractional digits.

Be sure to test your classes and include your tests. In addition to your unit tests, create a method in a separate driver class that creates a bank with several accounts of each type, adding monthly interest, displaying total assets and printing statements for the accounts over several iterations.

Note: You should be aware of Java’s enhanced for loop for syntactical ease in processing (see Weiss 2.4.6).