CS 532 Lab #5 Due 11/21/14

The purpose of this lab is to get practice working with databases, using transactions, and supporting multiple database connections.

1. Start with the code found in bankAcctConnPool.zip. Add the following method (and any necessary support methods):

   ```
   public static void doTransferRetry(int fromAcctId, double amt, int toAcctId)
               throws AccountDbFailure, SQLException
   ```

   The method should transfer money from one bank account to another. For example we might remove $100 from one account and add it to another account.

2. Your method should:
   a. **Use a transaction**
      If any part of the operation fails, **no modifications** should be made to the database.

   b. **Use an Optimistic Locking Approach**
      A **dirty check** should be performed when making updates to ensure that no other connections have modified the data while your method is working on it. If the dirty check fails, rollback and redo the entire transaction.

   c. **No duplicate code**
      Some code that you need already exists so it should not be duplicated. In addition, you should create reusable methods that someone else in the future could use to add new functionality (just like you are doing in this lab!). Learning to avoid writing duplicate code and writing reusable methods is good for your professional development. Also, writing reusable code generally means you will have to write smaller, simpler methods that perform one basic task, not five or six different tasks. Thus your code ends up being simpler, easier to read and understand, and reusable at the same time!

   d. **Follow the general code structure of the existing code**
      When you work for a company, you want to follow the general code patterns used by your group. This ensures consistency and helps everyone to understand the code more readily. You can use your own style for the class project, but for this lab, use the same basic patterns/naming conventions.

   e. **Use good method and variable names**
      No one should be using single letter names (except for loop variables). Take time to use meaningful variable names that describe what the variable is used for. Others reading your code should understand what the variable is used for just by looking at the name.

3. Modify the **main()** method so that it will call your **doTransferRetry()** method. The call should be wrapped in a try block and give an appropriate error if something goes wrong. Test this before handing in your solution to make sure it works – for example create a bad SQL statement and see what happens. Is the transaction rolled back? Is a proper error message given? Is the database connection still closed even though the exception is thrown? Use the debugger and step through the execution to follow the program flow.

4. Remove the **depositToAccountTwoConn()** method to clean up the code since you will not be using it for this lab.

5. Make your code as professional as possible so that others will respect your work and see you as a more advanced programmer.

Hand in an exported **Eclipse Project** (.zip file).