Programming Exercise 12-2

In this lab, you will refine your work in Lab 4-3 (concerning Employee) so that it incorporates best practices for exception handling.

In solving Lab 4-3 up to this point, we have handled the situation in which a user attempts to withdraw more than the current balance in the following way: The request for withdrawal is made by the Main class to the appropriate Employee object; this object then requests a withdrawal by referencing the appropriate Account object. If the withdrawal amount is too high, the Account object does not make the withdrawal and simply returns "false". The Employee object then returns this boolean to Main. Main then tests the boolean, and if false, informs the user that the withdrawal amount is too high.

Implement a better exception-handling strategy by doing the following: Create a class OverdrawnAccountException, a subclass of Exception. Have the Account class throw an exception of this type whenever a withdrawal amount is too high (the makeWithdrawal method should now have *no* return value). When one of these exceptions is thrown, Employee objects should pass it along and the Main should handle it with an appropriate message to the user.

Include in your exception class a one-argument constructor that can be used to pass error messages to a handler class. The error message that should be passed in by most Accounts is

Withdrawal amount exceeds balance

However, for Retirement Accounts, the following more descriptive message should be used:

After computing penalties, your withdrawal amount exceeds your balance.