SOLUTION BY SUSHAMA KUMARI ([sushamawork@gmail.com](mailto:sushamawork@gmail.com))

Following are the design level changes and code-quality changes I propose for the shared code.

Also, I have proposed design for one of the mentioned Additional Features at the end.

1. Separate class preferred for ‘interest calculation’
   1. Interest rates may vary often depending on market conditions.
   2. Add a Class named: InterestOnAccount
      1. Method ‘interestEarned’ from class ‘Account’ can be moved to class ‘InterestOnAccount’
      2. Inside the switch statement in the method ‘interestEarned’ there should be a separate case for ‘CHECKING’ instead of adding it to ‘default’ case
      3. A ‘default’ case can be added that performs nothing
2. Configurable interest rates
   1. Interest rates should not be hard coded as they may vary
   2. In a properties/configuration file final variables can be defined for interest rates
3. Unique Id for every Customer instance:
   1. A new member variable can be added to the class Customer: customerId
      1. Multiple customers can have similar name name but customerId should be unique for each instance of class Customer
      2. This involves code changes in the 2 classes: Customer, Bank
         1. Constructor in class ‘Customer’ should :
            1. generate a unique customerId. It can be accomplished by incrementing the highest available customerId by 1
            2. call the method ‘addCustomer’ of the class Bank
4. ‘default’ case should be included in switch statement inside method ‘statementForAccount’ in class Customer

This ‘default’ case should perform nothing.

1. Better way of representing ‘withdraw’ operations in the method ‘withdraw’ of class Account
   1. In the else block add the following instead:

transactions.add(new Transaction((-1)\*amount)));

1. Additional Feature:
   1. Customer can transfer between their accounts
      1. New method to be added to class Account called : transferToAccount(Account account, double amountToBeTransferred)

This methods accepts the account to which the transfer is required. It deducts the specified amount from the caller account and adds it to the accountId specified as input parameter.