Shem Thuo   SCT212-0529/2022

BIT 2203 Advanced Programming

Assignment 1; Quiz 2

1. **Modify the** `WithdrawalTransaction` **Class:** Implement the `reverse()` method to restore the bank account balance to its original amount.

   java

```
import java.util.Calendar;

public class WithdrawalTransaction extends BaseTransaction {
    private boolean reversed = false;

    public WithdrawalTransaction(double amount, Calendar date, String
transactionID) {
        super(amount, date, transactionID);
    }

    @Override
    public void apply(BankAccount ba) {
        ba.withdraw(amount);
        System.out.println("Withdrew: " + amount);
    }

    public boolean reverse(BankAccount ba) {
        if (reversed) {
            System.out.println("Transaction already reversed.");
            return false;
        }
        try {
            ba.deposit(amount);
            reversed = true;
            System.out.println("Withdrawal reversed: " + amount);
            return true;
        } catch (Exception e) {
            System.out.println("Failed to reverse withdrawal: " +
e.getMessage());
            return false;
        }
    }
}
```

2. **Ensure Deposits are Irreversible:** No need for a reverse method in `DepositTransaction`. The `apply` method will remain as is.

   java

```
import java.util.Calendar;

public class DepositTransaction extends BaseTransaction {
    public DepositTransaction(double amount, Calendar date, String
transactionID) {
        super(amount, date, transactionID);
    }

    @Override
    public void apply(BankAccount ba) {
        ba.deposit(amount);
        System.out.println("Deposited: " + amount);
    }
}
```

3. **Update BankAccount Class:** Ensure the `BankAccount` class has methods to handle deposits and withdrawals, as well as maintaining balance integrity.

   java

```
public class BankAccount {
    private double balance;

    public BankAccount(double initialBalance) {
        this.balance = initialBalance;
    }

    public void deposit(double amount) {
        balance += amount;
    }

    public void withdraw(double amount) throws
InsufficientFundsException {
        if (amount > balance) {
            throw new InsufficientFundsException("Insufficient funds
for withdrawal");
        }
        balance -= amount;
    }

    public double getBalance() {
        return balance;
    }
}
```

4. **Client Code:** Demonstrate the functionality of both deposit and withdrawal transactions, including the reversal of a withdrawal.

   java

```
import java.util.Calendar;

public class Main {
    public static void main(String[] args) {
        BankAccount account = new BankAccount(1000);
        Calendar date = Calendar.getInstance();
        DepositTransaction deposit = new DepositTransaction(200, date,
"TXN001");
        WithdrawalTransaction withdrawal = new
WithdrawalTransaction(150, date, "TXN002");

        deposit.apply(account);
        System.out.println("Balance after deposit: " +
account.getBalance());

        withdrawal.apply(account);
        System.out.println("Balance after withdrawal: " +
account.getBalance());

        boolean reversed = withdrawal.reverse(account);
        System.out.println("Withdrawal reversed: " + reversed);
        System.out.println("Final Balance: " + account.getBalance());
    }
}
```