Copyright 2003 Bill Campbell and Ethan Bolker

Lines marked "///" flag places where changes will be needed.

/**
 * A Bank object simulates the behavior of a simple bank/ATM.
 * It contains a Terminal object and a collection of
 * BankAccount objects.
 *
 * Its public method visit opens this Bank for business,
 * prompting the customer for input.
 *
 * To create a Bank and open it for business issue the command
 * <code>java Bank</code>.
 *
 * @see BankAccount
 * @version 4
 */

public class Bank
{
private String bankName;           // the name of this Bank
private Terminal atm;              // for talking with the customer
private int balance = 0;           // total cash on hand
private int transactionCount = 0;  // number of Bank transactions done

private BankAccount[] accountList; /// collection of BankAccounts
/// omit next line when accountList is dynamic
private final static int NUM_ACCOUNTS = 3;

private static final String BANKER_COMMANDS =
"Banker commands: exit, open, customer, report, help."
private static final String CUSTOMER_TRANSACTIONS =
"Customer transactions: deposit, withdraw, transfer, balance, quit, help."

public Bank( String bankName, Terminal atm )
{
this.atm = atm;
this.bankName = bankName;
// initialize collection:
accountList = new BankAccount[NUM_ACCOUNTS]; ///

accountList[0] = new BankAccount(  0, this);
accountList[1] = new BankAccount(100, this);
accountList[2] = new BankAccount(200, this);
}

public void visit()
{
instructUser();

String command;
while (!(command = atm.readWord("banker command: ")).equals("exit")) {

if (command.startsWith("h")) {
help( BANKER_COMMANDS );
}
else if (command.startsWith("o")) {
openNewAccount();
}
else if (command.startsWith("r")) {
report();
}
else if (command.startsWith("c")) {
BankAccount acct = whichAccount();
if ( acct != null )
processTransactionsForAccount( acct );
}
else {
// Unrecognized Request
atm.println("unknown command: "+ command);
}
}
report();
atm.println("Goodbye from "+ bankName);
}

// Open a new bank account,
// prompting the user for information.

accountList[0] = new BankAccount(0, 100, this);
accountList[1] = new BankAccount(0, 200, this);
accountList[2] = new BankAccount(0, 300, this);

}
private void openNewAccount()
{
    if (accountList is a dynamic collection
        remove the next two lines, uncomment and complete
        the code between /* and */
        atm.println("bankName + " is accepting no new customers
    return;

    /*
    // prompt for initial deposit
    int startup = atm.readInt("Initial deposit: ");

    // create newAccount
    BankAccount newAccount = new BankAccount(startup, this);

    // and add it to accountList
    ???

    // inform user
    atm.println("opened new account "+ ??? /// name or number
    + " with $" + newAccount.getBalance());
    */
}

// Prompt the customer for transaction to process.
// Then send an appropriate message to the account.

private void processTransactionsForAccount( BankAccount acct )
{
    help( CUSTOMER_TRANSACTIONS );

    String transaction;
    while (!(transaction =
        atm.readWord("transaction: ")).equals("quit")) {
        if ( transaction.startsWith( "h" ) ) {
            help( CUSTOMER_TRANSACTIONS );
        } else if ( transaction.startsWith( "d" ) ) {
            int amount = atm.readInt("amount: ");
            atm.println("deposited "+ acct.deposit(amount));
        } else if ( transaction.startsWith( "w" ) ) {
            int amount = atm.readInt("amount: ");
            atm.println("withdrew "+ acct.withdraw(amount));
        } else if (transaction.startsWith("t")) {
            atm.print("to ");
            BankAccount toacct = whichAccount();
            if (toacct != null) {
                int amount = atm.readInt("amount to transfer: ");
                atm.println("transfered "+
toacct.deposit(acct.withdraw(amount)));
            }
        } else if (transaction.startsWith("b")) {
            atm.println("current balance "+ acct.requestBalance());
        } else {
            atm.println("sorry, unknown transaction");
        }
    }
    atm.println();
}

private BankAccount whichAccount()
{
    if (accountName or account number
        (whichever is appropriate)
            int accountNumber = atm.readInt("account number: ");

            // look up account in accountList
            if (accountNumber >= 0 && accountNumber < NUM_ACCOUNTS ) {
                return accountList[accountNumber];
            } else {
                atm.println("not a valid account");
                return null;
            }
}

// Report bank activity.
// For each BankAccount, print the customer id (name or number),
// account balance and the number of transactions.
// Then print Bank totals.

private void report()
{
    atm.println(\n        \"Summaries of individual accounts: \n        \" + accountList[0].getBalance() +

    atm.println(\n        \"Bank totals");
    atm.println(\"open accounts: "+ getNumberOfAccounts());
    atm.println(\"cash on hand: "+ getBalance());
    atm.println(\"transactions: "+ getTransactionCount());
    atm.println();
}

// Welcome the user to the bank and instruct her on
// how to use the bank. Then prompt her to make a transaction

app.print("Welcome to the bank");
app.print("How can I help you?");
private void instructUser()
{
atm.println( "Welcome to " + bankName );
atm.println( "Open some accounts and work with them." );
help( BANKER_COMMANDS );
}

private void help( String helpString )
{
atm.println( helpString );
atm.println();
}

/**
* Increment bank balance by given amount.
*
* @param amount the amount increment.
*/
public void incrementBalance(int amount)
{
balance += amount;
}

/**
* Increment by one the count of transactions,
* for this bank.
*/
public void countTransaction()
{
transactionCount++; 
}

/**
* Get the current bank balance.
*
* @return current bank balance.
*/
public int getBalance()
{
return balance;
}

/**
* Get the current number of open accounts.
*
* @return number of open accounts.
*/
public int getNumberOfAccounts()
{
return NUM_ACCOUNTS; /// needs changing ...
}

public static void main( String[] args )
{
// parse the command line arguments for the echo

boolean echo    = false;        // default does not echo
String bankName = "River Bank"; // default bank name

for (int i = 0; i < args.length; i++ )
{
if (args[i].equals("-e"))
{
echo = true;
}
else
{
bankName = args[i];
}
}

Bank aBank = new Bank( bankName, new Terminal(echo) );
aBank.visit();
}
public class BankAccount {
    private int balance = 0;  // Account balance (whole dollars)
    private int transactionCount = 0;  // Number of transactions performed.
    private Bank issuingBank;  // Bank issuing this account

    BankAccount( int initialBalance, Bank issuingBank ) {
        this.issuingBank = issuingBank;
        deposit( initialBalance );
    }

    public int withdraw( int amount ) {
        incrementBalance( -amount );
        countTransaction();
        return amount;
    }

    public int deposit(int amount) {
        incrementBalance( amount);
        countTransaction();
        return amount;
    }

    public int requestBalance() {
        countTransaction();
        return getBalance();
    }

    public int getBalance() {
        return balance;
    }

    public void incrementBalance( int amount ) {
        balance += amount;
        this.getIssuingBank().incrementBalance( amount );
    }

    public int getNumberTransactions() {
        countTransaction();
        return transactionCount;
    }

    public void countTransaction() {
        this.issuingBank.createNewTransaction();
        ++this.transactionCount;
    }

    public Bank getIssuingBank() {
        countTransaction();
        return issuingBank;
    }

    void getIssuingBank() {
        countTransaction();
        issuingBank = null;
    }

    public void deposit(int amount, Bank bank) {
        countTransaction();
        bank.deposit(amount);
    }

    public void withdraw(int amount, Bank bank) {
        countTransaction();
        bank.withdraw(amount);
    }

    public void balance(Bank bank) {
        countTransaction();
        bank.balance();
    }

    public void transactions(Bank bank) {
        countTransaction();
        bank.transactions();
    }

    public static void main(String[] args) {
        BankAccount account = new BankAccount(0, new Bank(12345));
        account.deposit(100);
        System.out.println(account.getBalance());  // 100
        account.withdraw(50);
        System.out.println(account.getBalance());  // 50
    }
}

BankAccount java
public int getTransactionCount()
{
    return transactionCount;
}

/**
 * Increment by 1 the count of transactions, for this account
 * and for the issuing Bank.
 */

public void countTransaction()
{
    transactionCount++;
    this.getIssuingBank().countTransaction();
}

/**
 * Get the bank that issued this account.
 *
 * @return issuing bank.
 */

public Bank getIssuingBank()
{
    return issuingBank;
}
1 open 1000
2 open 2000
3 help
4 report
5 deposit
6 balance
7 withdraw
8 customer balance
9 customer deposit
10 customer balance
11 customer deposit
12 customer withdraw
13 customer balance
14 transfer
15 quit
16 customer transfer
17 customer quit
18 customer transfer
19 customer quit
20 transfer
21 quit
22 exit
23 quit
24 exit
Welcome to River Bank

Open some accounts and work with them.

Banker commands: exit, open, customer, report, help.

banker command: open
Initial deposit: 1000
opened new account 0 with $1000

banker command: open
Initial deposit: 2000
opened new account 1 with $2000

banker command: help
Banker commands: exit, open, customer, report, help.

banker command: report

Summaries of individual accounts:

<table>
<thead>
<tr>
<th>account</th>
<th>balance</th>
<th>transaction count</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$1000</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>$2000</td>
<td>1</td>
</tr>
</tbody>
</table>

Bank totals
open accounts: 2
cash on hand: $3000
transactions: 2

banker command: open
Initial deposit: 3000
opened new account 2 with $3000

banker command: customer
account number: 0
Customer transactions: deposit, withdraw, transfer, balance, quit

transaction: balance
current balance 1000

transaction: deposit
amount: 9999
deposited 9999

transaction: balance
current balance 10999

transaction: quit

banker command: customer
account number: 1
Customer transactions: deposit, withdraw, transfer, balance, quit

transaction: transfer
to account number: 9
not a valid account

transaction: transfer
to account number: 2
amount to transfer: 45
transferred 45

transaction: quit

banker command: exit

Goodbye from River Bank
Welcome to River Bank

Open some accounts and work with them.

Banker commands: exit, open, customer, report, help.

banker command: open
Account name: groucho
Initial deposit: 1000
opened new account groucho with $1000

banker command: customer
account name: harpo
not a valid account

banker command: open
Account name: harpo
Initial deposit: 2000
opened new account harpo with $2000

banker command: help
Banker commands: exit, open, customer, report, help.

banker command: report

Summaries of individual accounts:

<table>
<thead>
<tr>
<th>Account</th>
<th>Balance</th>
<th>Transaction Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>groucho</td>
<td>$1000</td>
<td>1</td>
</tr>
<tr>
<td>harpo</td>
<td>$2000</td>
<td>1</td>
</tr>
</tbody>
</table>

Bank totals
open accounts: 2
cash on hand: $3000
transactions: 2

banker command: open
Account name: chico
Initial deposit: 3000
opened new account chico with $3000

banker command: customer
account name: groucho
Customer transactions: deposit, withdraw, transfer, balance, quit, help.

transaction: balance
current balance 1000

transaction: deposit
amount: 9999
deposited 9999

transaction: balance
current balance 10999

transaction: quit

banker command: customer
account name: harpo
Customer transactions: deposit, withdraw, transfer, balance, quit, help.

transaction: transfer
to account name: chico
amount to transfer: 45
transferred 45

transaction: quit

banker command: exit

Summaries of individual accounts:

<table>
<thead>
<tr>
<th>Account</th>
<th>Balance</th>
<th>Transaction Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>chico</td>
<td>$3045</td>
<td>2</td>
</tr>
<tr>
<td>groucho</td>
<td>$10999</td>
<td>4</td>
</tr>
<tr>
<td>harpo</td>
<td>$19552</td>
<td>2</td>
</tr>
</tbody>
</table>

Bank totals
open accounts: 3
cash on hand: $15999
transactions: 8

Goodbye from River Bank