

```

1 // fo1/5/bank/BankAccount.java
2 //
3 //
4 // Copyright 2003 Bill Campbell and Ethan Bolker
5
6 /**
7  * A BankAccount object has private fields to keep track
8  * of its current balance, the number of transactions
9  * performed and the Bank in which it is an account, and
10 * and public methods to access those fields appropriately.
11 *
12 * @see Bank
13 * @version 5
14 */
15
16 public abstract class BankAccount
17 {
18     private int balance = 0; // Account balance (whole dollars)
19     private int transactionCount = 0; // Number of transactions performed
20     private Bank issuingBank; // Bank issuing this account
21
22     /**
23      * Construct a BankAccount with the given initial balance and
24      * issuing Bank. Construction counts as this BankAccount's
25      * first transaction.
26      *
27      * @param initialBalance the opening balance.
28      * @param issuingBank the bank that issued this account.
29      */
30
31     public BankAccount( int initialBalance, Bank issuingBank )
32     {
33         this.issuingBank = issuingBank;
34         deposit( initialBalance );
35     }
36
37     /**
38      * Withdraw the given amount, decreasing this BankAccount's
39      * balance and the issuing Bank's balance.
40      * Counts as a transaction.
41      *
42      * @param amount the amount to be withdrawn
43      * @return amount withdrawn
44      */
45
46     public int withdraw( int amount )
47     {
48         incrementBalance( -amount );
49         countTransaction();
50         return amount ;
51     }
52
53     /**
54      * Deposit the given amount, increasing this BankAccount's
55      * balance and the issuing Bank's balance.
56      * Counts as a transaction.

```

```

57     *
58     * @param amount the amount to be deposited
59     * @return amount deposited
60     */
61
62     public int deposit( int amount )
63     {
64         incrementBalance( amount );
65         countTransaction();
66         return amount ;
67     }
68
69     /**
70      * Request for balance. Counts as a transaction.
71      *
72      * @return current account balance.
73      */
74
75     public int requestBalance()
76     {
77         countTransaction();
78         return getBalance() ;
79     }
80
81     /**
82      * Get the current balance.
83      * Does NOT count as a transaction.
84      *
85      * @return current account balance
86      */
87
88     public int getBalance()
89     {
90         return balance;
91     }
92
93     /**
94      * Increment account balance by given amount.
95      * Also increment issuing Bank's balance.
96      * Does NOT count as a transaction.
97      *
98      * @param amount the amount of the increment.
99      */
100
101     public void incrementBalance( int amount )
102     {
103         balance += amount;
104         this.getIssuingBank().incrementBalance( amount );
105     }
106
107     /**
108      * Get the number of transactions performed by this
109      * account. Does NOT count as a transaction.
110      *
111      * @return number of transactions performed.
112     */

```

```
113 public int getTransactionCount()
114 {
115     return transactionCount;
116 }
117
118 /**
119  * Increment by 1 the count of transactions, for this account
120  * and for the issuing Bank.
121  * Does NOT count as a transaction.
122  */
123
124 public void countTransaction()
125 {
126     transactionCount++;
127     this.getIssuingBank().countTransaction();
128 }
129
130 /**
131  * Get the bank that issued this account.
132  * Does NOT count as a transaction.
133  * @return issuing bank.
134  */
135
136 public Bank getIssuingBank()
137 {
138     return issuingBank;
139 }
140
141 /**
142  * Action to take when a new month starts.
143  */
144
145 public abstract void newMonth();
146
147 }
148 }
```