```
1  // DepositSlot.h
2  // DepositSlot class definition. Represents the ATM's deposit slot.
3  #ifndef DEPOSIT_SLOT_H
4  #define DEPOSIT_SLOT_H
5
6  class DepositSlot
7  {
8  public:
9     bool isEnvelopeReceived() const; // tells whether envelope was received
10 }; // end class DepositSlot
11
12 #endif // DEPOSIT_SLOT_H
```

Fig. 25.22 | `DepositSlot` class definition.

```
1  // DepositSlot.cpp
2  // Member-function definition for class DepositSlot.
3  #include "DepositSlot.h" // DepositSlot class definiton
4
5  // indicates whether envelope was received (always returns true,
6  // because this is only a software simulation of a real deposit slot)
7  bool DepositSlot::isEnvelopeReceived() const
8  {
9     return true; // deposit envelope was received
10 } // end function isEnvelopeReceived
```

Fig. 25.23 | `DepositSlot` class member-function definition.

26.4.6 Class Account

```
1  // Account.h
2  // Account class definition. Represents a bank account.
3  #ifndef ACCOUNT_H
4  #define ACCOUNT_H
5
6  class Account
7  {
8  public:
9     Account( int, int, double, double ); // constructor sets attributes
10    bool validatePIN( int ) const; // is user-specified PIN correct?
11    double getAvailableBalance() const; // returns available balance
12    double getTotalBalance() const; // returns total balance
13    void credit( double ); // adds an amount to the Account balance
14    void debit( double ); // subtracts an amount from the Account balance
15    int getAccountNumber() const; // returns account number
16 private:
17    int accountNumber; // account number
18    int pin; // PIN for authentication
19    double availableBalance; // funds available for withdrawal
20    double totalBalance; // funds available + funds waiting to clear
21 }; // end class Account
22
23 #endif // ACCOUNT_H
```

Fig. 25.24 | Account class definition.

```
1  // Account.cpp
2  // Member-function definitions for class Account.
3  #include "Account.h" // Account class definition
4
5  // Account constructor initializes attributes
6  Account::Account( int theAccountNumber, int thePIN,
7     double theAvailableBalance, double theTotalBalance )
8     : accountNumber( theAccountNumber ),
9       pin( thePIN ),
10      availableBalance( theAvailableBalance ),
11      totalBalance( theTotalBalance )
12 {
13    // empty body
14 } // end Account constructor
15
16 // determines whether a user-specified PIN matches PIN in Account
17 bool Account::validatePIN( int userPIN ) const
18 {
19    if ( userPIN == pin )
20       return true;
21    else
22       return false;
23 } // end function validatePIN
24
```

Fig. 25.25 | Account class member-function definitions. (Part 1 of 3.)

```
25  // returns available balance
26  double Account::getAvailableBalance() const
27  {
28     return availableBalance;
29  } // end function getAvailableBalance
30
31  // returns the total balance
32  double Account::getTotalBalance() const
33  {
34     return totalBalance;
35  } // end function getTotalBalance
36
37  // credits an amount to the account
38  void Account::credit( double amount )
39  {
40     totalBalance += amount; // add to total balance
41  } // end function credit
42
```

Fig. 25.25 | `Account` class member-function definitions. (Part 2 of 3.)

```
43  // debits an amount from the account
44  void Account::debit( double amount )
45  {
46     availableBalance -= amount; // subtract from available balance
47     totalBalance -= amount; // subtract from total balance
48  } // end function debit
49
50  // returns account number
51  int Account::getAccountNumber() const
52  {
53     return accountNumber;
54  } // end function getAccountNumber
```

Fig. 25.25 | `Account` class member-function definitions. (Part 3 of 3.)

26.4.7 Class `BankDatabase`

```
1  // BankDatabase.h
2  // BankDatabase class definition. Represents the bank's database.
3  #ifndef BANK_DATABASE_H
4  #define BANK_DATABASE_H
5
6  #include <vector> // class uses vector to store Account objects
7  using namespace std;
8
9  #include "Account.h" // Account class definition
10
```

Fig. 25.26 | BankDatabase class definition. (Part 1 of 2.)

```
11  class BankDatabase
12  {
13  public:
14     BankDatabase(); // constructor initializes accounts
15
16     // determine whether account number and PIN match those of an Account
17     bool authenticateUser( int, int ); // returns true if Account authentic
18
19     double getAvailableBalance( int ); // get an available balance
20     double getTotalBalance( int ); // get an Account's total balance
21     void credit( int, double ); // add amount to Account balance
22     void debit( int, double ); // subtract amount from Account balance
23  private:
24     vector< Account > accounts; // vector of the bank's Accounts
25
26     // private utility function
27     Account * getAccount( int ); // get pointer to Account object
28  }; // end class BankDatabase
29
30  #endif // BANK_DATABASE_H
```

Fig. 25.26 | BankDatabase class definition. (Part 2 of 2.)

```
1  // BankDatabase.cpp
2  // Member-function definitions for class BankDatabase.
3  #include "BankDatabase.h" // BankDatabase class definition
4
5  // BankDatabase default constructor initializes accounts
6  BankDatabase::BankDatabase()
7  {
8     // create two Account objects for testing
9     Account account1( 12345, 54321, 1000.0, 1200.0 );
10    Account account2( 98765, 56789, 200.0, 200.0 );
11
12    // add the Account objects to the vector accounts
13    accounts.push_back( account1 ); // add account1 to end of vector
14    accounts.push_back( account2 ); // add account2 to end of vector
15 } // end BankDatabase default constructor
16
```

Fig. 25.27 | BankDatabase class member-function definitions. (Part 1 of 4.)

```
17  // retrieve Account object containing specified account number
18  Account * BankDatabase::getAccount( int accountNumber )
19  {
20     // loop through accounts searching for matching account number
21     for ( size_t i = 0; i < accounts.size(); i++ )
22     {
23        // return current account if match found
24        if ( accounts[ i ].getAccountNumber() == accountNumber )
25           return &accounts[ i ];
26     } // end for
27
28     return NULL; // if no matching account was found, return NULL
29  } // end function getAccount
30
31  // determine whether user-specified account number and PIN match
32  // those of an account in the database
33  bool BankDatabase::authenticateUser( int userAccountNumber,
34     int userPIN )
35  {
36     // attempt to retrieve the account with the account number
37     Account * const userAccountPtr = getAccount( userAccountNumber );
38
```

Fig. 25.27 | BankDatabase class member-function definitions. (Part 2 of 4.)

```
39      // if account exists, return result of Account function validatePIN
40      if ( userAccountPtr != NULL )
41         return userAccountPtr->validatePIN( userPIN );
42      else
43         return false; // account number not found, so return false
44   } // end function authenticateUser
45
46   // return available balance of Account with specified account number
47   double BankDatabase::getAvailableBalance( int userAccountNumber )
48   {
49      Account * const userAccountPtr = getAccount( userAccountNumber );
50      return userAccountPtr->getAvailableBalance();
51   } // end function getAvailableBalance
52
53   // return total balance of Account with specified account number
54   double BankDatabase::getTotalBalance( int userAccountNumber )
55   {
56      Account * const userAccountPtr = getAccount( userAccountNumber );
57      return userAccountPtr->getTotalBalance();
58   } // end function getTotalBalance
59
```

Fig. 25.27 | BankDatabase class member-function definitions. (Part 3 of 4.)

```
60  // credit an amount to Account with specified account number
61  void BankDatabase::credit( int userAccountNumber, double amount )
62  {
63     Account * const userAccountPtr = getAccount( userAccountNumber );
64     userAccountPtr->credit( amount );
65  } // end function credit
66
67  // debit an amount from Account with specified account number
68  void BankDatabase::debit( int userAccountNumber, double amount )
69  {
70     Account * const userAccountPtr = getAccount( userAccountNumber );
71     userAccountPtr->debit( amount );
72  } // end function debit
```

Fig. 25.27 | BankDatabase class member-function definitions. (Part 4 of 4.)

26.4.8 Class `Transaction`