```
109         // determine how to proceed based on the input value
110         switch ( input )
111         {
112            case 1: // if the user chose a withdrawal amount
113            case 2: // (i.e., chose option 1, 2, 3, 4 or 5), return the
114            case 3: // corresponding amount from amounts array
115            case 4:
116            case 5:
117               userChoice = amounts[ input ]; // save user's choice
118               break;
119            case CANCELED: // the user chose to cancel
120               userChoice = CANCELED; // save user's choice
121               break;
122            default: // the user did not enter a value from 1-6
123               screen.displayMessageLine(
124                  "\nIvalid selection. Try again." );
125         } // end switch
126      } // end while
127
128      return userChoice; // return withdrawal amount or CANCELED
129   } // end function displayMenuOfAmounts
```

Fig. 25.33 | `Withdrawal` class member-function definitions. (Part 6 of 6.)

26.4.11 Class `Deposit`

```
1  // Deposit.h
2  // Deposit class definition. Represents a deposit transaction.
3  #ifndef DEPOSIT_H
4  #define DEPOSIT_H
5
6  #include "Transaction.h" // Transaction class definition
7  class Keypad; // forward declaration of class Keypad
8  class DepositSlot; // forward declaration of class DepositSlot
9
10 class Deposit : public Transaction
11 {
12 public:
13    Deposit( int, Screen &, BankDatabase &, Keypad &, DepositSlot & );
14    virtual void execute(); // perform the transaction
15 private:
16    double amount; // amount to deposit
17    Keypad &keypad; // reference to ATM's keypad
18    DepositSlot &depositSlot; // reference to ATM's deposit slot
19    double promptForDepositAmount() const; // get deposit amount from user
20 }; // end class Deposit
21
22 #endif // DEPOSIT_H
```

Fig. 25.34 | `Deposit` class definition.

```
1  // Deposit.cpp
2  // Member-function definitions for class Deposit.
3  #include "Deposit.h" // Deposit class definition
4  #include "Screen.h" // Screen class definition
5  #include "BankDatabase.h" // BankDatabase class definition
6  #include "Keypad.h" // Keypad class definition
7  #include "DepositSlot.h" // DepositSlot class definition
8
9  static const int CANCELED = 0; // constant representing cancel option
10
11 // Deposit constructor initializes class's data members
12 Deposit::Deposit( int userAccountNumber, Screen &atmScreen,
13    BankDatabase &atmBankDatabase, Keypad &atmKeypad,
14    DepositSlot &atmDepositSlot )
15    : Transaction( userAccountNumber, atmScreen, atmBankDatabase ),
16      keypad( atmKeypad ), depositSlot( atmDepositSlot )
17 {
18    // empty body
19 } // end Deposit constructor
20
```

Fig. 25.35 | `Deposit` class member-function definitions. (Part 1 of 4.)

```
21  // performs transaction; overrides Transaction's pure virtual function
22  void Deposit::execute()
23  {
24     BankDatabase &bankDatabase = getBankDatabase(); // get reference
25     Screen &screen = getScreen(); // get reference
26
27     amount = promptForDepositAmount(); // get deposit amount from user
28
29     // check whether user entered a deposit amount or canceled
30     if ( amount != CANCELED )
31     {
32        // request deposit envelope containing specified amount
33        screen.displayMessage(
34           "\nPlease insert a deposit envelope containing " );
35        screen.displayDollarAmount( amount );
36        screen.displayMessageLine( " in the deposit slot." );
37
38        // receive deposit envelope
39        bool envelopeReceived = depositSlot.isEnvelopeReceived();
40
```

Fig. 25.35 | `Deposit` class member-function definitions. (Part 2 of 4.)

```
41         // check whether deposit envelope was received
42         if ( envelopeReceived )
43         {
44            screen.displayMessageLine( "\nYour envelope has been received."
45               "\nNOTE: The money deposited will not be available until we"
46               "\nverify the amount of any enclosed cash, and any enclosed "
47               "checks clear." );
48
49            // credit account to reflect the deposit
50            bankDatabase.credit( getAccountNumber(), amount );
51         } // end if
52         else // deposit envelope not received
53         {
54            screen.displayMessageLine( "\nYou did not insert an "
55               "envelope, so the ATM has canceled your transaction." );
56         } // end else
57      } // end if
58      else // user canceled instead of entering amount
59      {
60         screen.displayMessageLine( "\nCanceling transaction..." );
61      } // end else
62   } // end function execute
63
```

Fig. 25.35 | `Deposit` class member-function definitions. (Part 3 of 4.)

```
64  // prompt user to enter a deposit amount in cents
65  double Deposit::promptForDepositAmount() const
66  {
67     Screen &screen = getScreen(); // get reference to screen
68
69     // display the prompt and receive input
70     screen.displayMessage( "\nPlease enter a deposit amount in "
71        "CENTS (or 0 to cancel): " );
72     int input = keypad.getInput(); // receive input of deposit amount
73
74     // check whether the user canceled or entered a valid amount
75     if ( input == CANCELED )
76        return CANCELED;
77     else
78     {
79        return static_cast< double >( input ) / 100; // return dollar amount
80     } // end else
81  } // end function promptForDepositAmount
```

Fig. 25.35 | `Deposit` class member-function definitions. (Part 4 of 4.)

26.4.12 Test Program `ATMCaseStudy.cpp`

```
1  // ATMCaseStudy.cpp
2  // Driver program for the ATM case study.
3  #include "ATM.h" // ATM class definition
4
5  // main function creates and runs the ATM
6  int main()
7  {
8     ATM atm; // create an ATM object
9     atm.run(); // tell the ATM to start
10 } // end main
```

Fig. 25.36 | ATMCaseStudy.cpp starts the ATM system.

Answers to Self-Review Exercises

```
1  // Fig. 26.37: 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     bool validatePIN( int ); // is user-specified PIN correct?
10    double getAvailableBalance(); // returns available balance
11    double getTotalBalance(); // returns total balance
12    void credit( double ); // adds an amount to the Account
13    void debit( double ); // subtracts an amount from the Account
14 private:
15    int accountNumber; // account number
16    int pin; // PIN for authentication
17    double availableBalance; // funds available for withdrawal
18    double totalBalance; // funds available + funds waiting to clear
19 }; // end class Account
20
21 #endif // ACCOUNT_H
```

Fig. 25.37 | Account class header file based on Fig. 26.1 and Fig. 26.2.

```
1  // Fig. 36.38: Transaction.h
2  // Transaction abstract base class definition.
3  #ifndef TRANSACTION_H
4  #define TRANSACTION_H
5
6  class Screen; // forward declaration of class Screen
7  class BankDatabase; // forward declaration of class BankDatabase
8
9  class Transaction
10 {
11 public:
12    int getAccountNumber(); // return account number
13    Screen &getScreen(); // return reference to screen
14    BankDatabase &getBankDatabase(); // return reference to bank database
15
16    // pure virtual function to perform the transaction
17    virtual void execute() = 0; // overridden in derived classes
18 private:
19    int accountNumber; // indicates account involved
20    Screen &screen; // reference to the screen of the ATM
21    BankDatabase &bankDatabase; // reference to the account info database
22 }; // end class Transaction
23
24 #endif // TRANSACTION_H
```

Fig. 25.38 | `Transaction` class header file based on Fig. 26.10 and Fig. 26.11.