UMass

Electrical and Computer Engineering Department

# ECE 264, Object-Oriented Software Development

Lab Assignment 3

**Lab 3 :: 100 points (see Grading Notes for details) ::**
**Wednesday Session (Feb 20): Due Feb 22, 2013 (Friday);**
**Monday Session (Feb 25): Due Feb 27, 2013 (Wednesday)**

## 1. Lab Objectives

This lab was designed to reinforce programming concepts from Chapter 3 of *C++ How to Program, 8th Edition*.

In this lab, you will practice:

- Creating member functions.
- Invoking functions and receiving return values from functions.
- Testing a condition using an **if** statement.
- Outputting variables with stream insertion and the **cout** object.
- Declaring data members.
- Providing ***set*** and ***get*** functions to manipulate a data member's value.
- Declaring member functions with parameters.

## 2. Deliverables

Create "lab3" sub-directory on your M:\ drive. Submit your files to this sub-directory on the M:\ drive. Call your project *lab3_Account and lab3_GradeBook* respectively. You should place all the source files (.h and .cpp) on the"lab3" sub-directory. Failure to meet this specification will reduce your grade, as described in the ECE 264 lab grading handout, which you are strongly encouraged to read before starting the lab.

## 3. Description of the Problem 1 (Project name: *lab3_Account)*

Modify class **Account** (Fig. L 3.1 and Fig. L 3.2) to provide a member function called **debit** that withdraws money from an **Account**. Ensure that the debit amount does not exceed the Account's balance. If it does, the balance should be left unchanged and the function should print a message indicating **"Debit amount exceeded account balance."** Modify class **AccountTest** (Fig. L 3.3) to test member function **debit**.

# ECE 264, Object-Oriented Software Development

Lab Assignment 3

## Sample Output

```
account1 balance: $50.00
account2 balance: $0.00

Enter withdrawal amount for account1: 25

subtracting 25 from account1 balance

account1 balance: $25
account2 balance: $0.00

Enter withdrawal amount for account2: 10

subtracting 10 from account2 balance

Debit amount exceeded account balance.

account1 balance: $25
account2 balance: $0
```

## Program Template

```
1  // Lab 1: Account.h
2  // Definition of Account class.
3
4  class Account
5  {
6  public:
7     Account( int ); // constructor initializes balance
8     void credit( int ); // add an amount to the account balance
9     /* write code to declare member function debit. */
10    int getBalance(); // return the account balance
11 private:
12    int balance; // data member that stores the balance
13 }; // end class Account
```

Fig. L 3.1 | Account.h.

```
1  // Lab 1: Account.cpp
2  // Member-function definitions for class Account.
3  #include <iostream>
4  using namespace std;
5
6  #include "Account.h" // include definition of class Account
7
8  // Account constructor initializes data member balance
9  Account::Account( int initialBalance )
10 {
11    balance = 0; // assume that the balance begins at 0
12
13    // if initialBalance is greater than 0, set this value as the
14    // balance of the Account; otherwise, balance remains 0
```

Fig. L 3.2 | Account.cpp. (Part 1 of 2.)

# ECE 264, Object-Oriented Software Development

Lab Assignment 3

```
15     if ( initialBalance > 0 )
16        balance = initialBalance;
17
18     // if initialBalance is negative, print error message
19     if ( initialBalance < 0 )
20        cout << "Error: Initial balance cannot be negative.\n" << endl;
21  } // end Account constructor
22
23  // credit (add) an amount to the account balance
24  void Account::credit( int amount )
25  {
26     balance = balance + amount; // add amount to balance
27  } // end function credit
28
29  /* write code to define member function debit. */
30
31  // return the account balance
32  int Account::getBalance()
33  {
34     return balance; // gives the value of balance to the calling function
35  } // end function getBalance
```

**Fig. L 3.2** | Account.cpp. (Part 2 of 2.)

```
1   // Lab 1: AccountTest.cpp
2   // Create and manipulate Account objects.
3   #include <iostream>
4   using namespace std;
5
6   // include definition of class Account from Account.h
7   #include "Account.h"
8
9   // function main begins program execution
10  int main()
11  {
12     Account account1( 50 ); // create Account object
13     Account account2( 0 ); // create Account object
14
15     // display initial balance of each object
16     cout << "account1 balance: $" << account1.getBalance() << endl;
17     cout << "account2 balance: $" << account2.getBalance() << endl;
18
19     int withdrawalAmount; // stores withdrawal amount read from user
20
21     cout << "\nEnter withdrawal amount for account1: "; // prompt
22     cin >> withdrawalAmount; // obtain user input
23     cout << "\nsubtracting " << withdrawalAmount
24        << " from account1 balance\n\n";
25     /* write code to withdraw money from account1 */
26
27     // display balances
28     cout << "account1 balance: $" << account1.getBalance() << endl;
29     cout << "account2 balance: $" << account2.getBalance() << endl;
30
```

# ECE 264, Object-Oriented Software Development

Lab Assignment 3

```
31      cout << "\nEnter withdrawal amount for account2: "; // prompt
32      cin >> withdrawalAmount; // obtain user input
33      cout << "\nsubtracting " << withdrawalAmount
34         << " from account2 balance\n\n";
35      /* write code to withdraw money from account2 */
36
37      // display balances
38      cout << "account1 balance: $" << account1.getBalance() << endl;
39      cout << "account2 balance: $" << account2.getBalance() << endl;
40   } // end main
```

**Fig. L 3.3** | AccountTest.cpp. (Part 2 of 2.)

## Problem-Solving Tips

- Declare public member function debit with a return type of void.
- Use a parameter to enable the program to specify the amount the user wishes to withdraw.
- In the body of member function debit, use an if statement to test whether the withdrawal amount is
- more than the balance. Output an appropriate message if the condition is true.
- Use another if statement to test whether the withdrawal amount is less than or equal to the balance.
- Decrement the balance appropriately.
- Be sure to follow the spacing and indentation conventions mentioned in the text.
- If you have any questions as you proceed, ask your lab instructor for help.

## 4. Description Problem 2 (Project name: *lab3_GradeBook)*

Modify class **GradeBook** (Fig. L 3.4 and Fig. L 3.6). Include a second string data member that represents the name of the course's instructor. Provide a ***set*** function to change the instructor's name and a ***get*** function to retrieve it. Modify the constructor to specify *two* parameters—one for the course name and one for the instructor's name. Modify member function **displayMessage** such that it first outputs the welcome message and course name, then outputs "**This course is presented by:** " followed by the instructor's name. Modify the test application (Fig. L 3.6) to demonstrate the class's new capabilities.

### Sample Output

```
Welcome to the grade book for
CS101 Introduction to C++ Programming!
This course is presented by: Sam Smith

Changing instructor name to Judy Jones

Welcome to the grade book for
CS101 Introduction to C++ Programming!
This course is presented by: Judy Jones
```

# ECE 264, Object-Oriented Software Development

Lab Assignment 3

## Program Template

```
1  // Lab 2: GradeBook.h
2  // Definition of GradeBook class that stores an instructor's name.
3  #include <string> // program uses C++ standard string class
4  using namespace std;
5
6  // GradeBook class definition
7  class GradeBook
8  {
9  public:
10    // constructor initializes course name and instructor name
11    GradeBook( string, string );
12    void setCourseName( string ); // function to set the course name
13    string getCourseName(); // function to retrieve the course name
14    /* write code to declare a get function for the instructor's name */
15    /* write code to declare a set function for the instructor's name */
16    void displayMessage(); // display welcome message and instructor name
17 private:
18    string courseName; // course name for this GradeBook
19    string instructorName; // instructor name for this GradeBook
20 }; // end class GradeBook
```

**Fig. L 3.4** | GradeBook.h.

```
1  // Lab 2: GradeBook.cpp
2  // Member-function definitions for class GradeBook.
3  #include <iostream>
4  using namespace std;
5
6  // include definition of class GradeBook from GradeBook.h
7  #include "GradeBook.h"
8
9  // constructor initializes courseName and instructorName
10 // with strings supplied as arguments
11 GradeBook::GradeBook( string course, string instructor )
12 {
13    setCourseName( course ); // initializes courseName
14    setInstructorName( instructor ); // initializes instructorName
15 } // end GradeBook constructor
16
17 // function to set the course name
18 void GradeBook::setCourseName( string name )
19 {
20    courseName = name; // store the course name
21 } // end function setCourseName
22
23 // function to retrieve the course name
24 string GradeBook::getCourseName()
25 {
26    return courseName;
27 } // end function getCourseName
28
29 /* write code to define a get member function for the instructor's name */
30
```

**Fig. L 3.5** | GradeBook.cpp. (Part 1 of 2.)

# ECE 264, Object-Oriented Software Development

Lab Assignment 3

```
31  /* write code to define a set member function for the instructor's name */
32
33  // display a welcome message and the instructor's name
34  void GradeBook::displayMessage()
35  {
36     // display a welcome message containing the course name
37     cout << "Welcome to the grade book for\n" << getCourseName() << "!"
38        << endl;
39     /* write code to output the instructor's name */
40  } // end function displayMessage
```

**Fig. L 3.5** | GradeBook.cpp. (Part 2 of 2.)

```
1  // Lab 2: GradeBookTest.cpp
2  // Test program for modified GradeBook class.
3  #include <iostream>
4  using namespace std;
5
6  // include definition of class GradeBook from GradeBook.h
7  #include "GradeBook.h"
8
9  // function main begins program execution
10 int main()
11 {
12    // create a GradeBook object; pass a course name and instructor name
13    GradeBook gradeBook(
14       "CS101 Introduction to C++ Programming" );
15
16    // display welcome message and instructor's name
17    gradeBook.displayMessage();
18
19    /* write code to change instructor's name and output changes */
20 } // end main
```

**Fig. L 3.6** | GradeBookTest.cpp.

## Problem-Solving Tips

- In class GradeBook, declare a string data member to represent the instructor's name.
- Declare a public *set* function for the instructor's name that does not return a value and takes a string
- as a parameter. In the body of the *set* function, assign the parameter's value to the data member that
- represents the instructor's name.
- Declare a public *get* function that returns a string and takes no parameters. This member function
- should return the instructor's name.
- Modify the constructor to take two string parameters. Assign the parameter that represents the instructor's
- name to the appropriate data member.
- Add an output statement to member function displayMessage to output the value of the data member
- you declared earlier.
- Be sure to follow the spacing and indentation conventions mentioned in the text.
- If you have any questions as you proceed, ask your lab instructor for help.

## 5. Testing Your Program

- For this program, there is no user input so the only way to test your program is to run it and see if it displays all of the information correctly.
- In all of your programs, but especially a program where there isn't any user input, you should focus on making the output easy to read. One of the most difficult things for a user of your program to deal with is poorly formatted output. The easier your output is to read, the easier it is to identify the relevant information that you're producing.