UMass

Electrical and Computer Engineering Department

ECE 264, Object-Oriented Software Development

Lab Assignment 4

Lab 4 :: 100 points (see Grading Notes for details) ::
Wednesday Session (Feb 27): Due March 1, 2013 (Friday);
Monday Session (March 4): Due March 6, 2013 (Wednesday);

1. Lab Objectives

This lab was designed to reinforce programming concepts from Chapter 3, 10 of *C++ How To Program, 8th Edition*. In this lab, you will practice:

- Creating a class definition.
- Declaring data members.
- Defining a constructor.
- Defining *set* and *get* functions.
- Writing a test application to demonstrate the capabilities of another class.
- Using classes to create a data type SimpleCalculator capable of performing arithmetic operations.

The follow-up questions and activities also will give you practice:

- Using constructors to specify initial values for data members of a programmer-defined class.

2. Deliverables

Create "lab4" sub-directory on your M:\ drive. Submit your file to this sub-directory on the M:\ drive. Call your project *lab4_ Employee and lab4_Calculator* respectively. You should place all the source files (.h and .cpp) on the"lab4" 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

Description of the Problem

Create a class called Employee that includes three pieces of information as data members—a first name (type string), a last name (type string) and a monthly salary (type int). [*Note:* you can use numbers that contain decimal points (e.g., 2.75)—called floating-point values—to represent dollar amounts.] Your class should have a constructor that initializes the three data members. Provide a *set* and a *get* function for each data member. If the monthly salary is not positive, set it to 0. Write a test program that demonstrates class Employee's capabilities. Create two Employee objects and display each object's yearly salary. Then give each Employee a 10 percent raise and display each Employee's yearly salary again.

Sample Output

ECE 264, Object-Oriented Software Development

Lab Assignment 4

```
Employee 1: Bob Jones; Yearly Salary: 34500
Employee 2: Susan Baker; Yearly Salary: 37800

Increasing employee salaries by 10%
Employee 1: Bob Jones; Yearly Salary: 37944
Employee 2: Susan Baker; Yearly Salary: 41580
```

Program Template

```
1  // Lab 3: Employee.h
2  // Employee class definition.
3
4  #include <string> // program uses C++ standard string class
5  using namespace std;
6
7  // Employee class definition
8  class Employee
9  {
10 public:
11    /* Declare a constructor that has one parameter for each data member */
12    /* Declare a set method for the employee's first name */
13    /* Declare a get method for the employee's first name */
14    /* Declare a set method for the employee's last name */
15    /* Declare a get method for the employee's last name */
16    /* Declare a set method for the employee's monthly salary */
17    /* Declare a get method for the employee's monthly salary */
18 private:
19    /* Declare a string data member for the employee's first name */
20    /* Declare a string data member for the employee's last name */
21    /* Declare an int data member for the employee's monthly salary */
22 }; // end class Employee
```

Fig. L 3.7 | Employee.h.

ECE 264, Object-Oriented Software Development

Lab Assignment 4

```
1  // Lab 3: Employee.cpp
2  // Employee class member-function definitions.
3  #include <iostream>
4  using namespace std;
5
6  #include "Employee.h" // Employee class definition
7
8  /* Define the constructor. Assign each parameter value to the appropriate data
9     member. Write code that validates the value of salary to ensure that it is
10    not negative. */
11
12 /* Define a set function for the first name data member. */
13
14 /* Define a get function for the first name data member. */
15
16 /* Define a set function for the last name data member. */
17
18 /* Define a get function for the last name data member. */
19
20 /* Define a set function for the monthly salary data member. Write code
21    that validates the salary to ensure that it is not negative. */
22
23 /* Define a get function for the monthly salary data member. */
```

Fig. L 3.8 | Employee.cpp.

```
1  // Lab 3: EmployeeTest.cpp
2  // Create and manipulate two Employee objects.
3  #include <iostream>
4  using namespace std;
5
6  #include "Employee.h" // include definition of class Employee
7
8  // function main begins program execution
9  int main()
10 {
11    /* Create two Employee objects and assign them to Employee variables. */
12
13    /* Output the first name, last name and salary for each Employee. */
14
15    /* Give each Employee a 10% raise. */
16
17    /* Output the first name, last name and salary of each Employee again. */
18 } // end main
```

Fig. L 3.9 | EmployeeTest.cpp .

ECE 264, Object-Oriented Software Development

Lab Assignment 4

Problem-Solving Tips

1. Class Employee should declare three data members.
2. The constructor must declare three parameters, one for each data member. The value for the salary should be validated to ensure it is not negative.
3. Declare a public *set* and *get* functions for each data member. The *set* functions should not return values and should each specify a parameter of a type that matches the corresponding data member (string for first name and last name, int for the salary). The *get* functions should receive no parameters and should specify a return type that matches the corresponding data member.
4. When you call the constructor from the main function, you must pass it three arguments that match the parameters declared by the constructor.
5. Giving each employee a raise will require a call to the *get* function for the salary to obtain the current salary and a call to the *set* function for the salary to specify the new salary.
6. Be sure to follow the spacing and indentation conventions mentioned in the text.
7. If you have any questions as you proceed, ask your lab instructor for help.

4. Description of the Problem 2

Write a SimpleCalculator class that has public methods for adding, subtracting, multiplying and dividing two doubles. A sample call is as follows:

```
double answer = sc.add( a, b );
```

Object sc is of type SimpleCalculator. Member function add returns the result of adding its two arguments.

Sample Output

```
The value of a is: 10
The value of b is: 20

Adding a and b yields 30
Subtracting b from a yields -10
Multiplying a by b yields 200
Dividing a by b yields 0.5
```

ECE 264, Object-Oriented Software Development

Lab Assignment 4

Template

```
1  // Lab Exercise 1: SimpleCalculator.h
2
3  // class SimpleCalculator definition
4  class SimpleCalculator
5  {
6  public:
7     /* Write prototype for add member function */
8     double subtract( double, double ) const;
9     double multiply( double, double ) const;
10    /* Write prototype for divide member function */
11
12 }; // end class SimpleCalculator
```

Fig. L 10.1 | Contents of SimpleCalculator.h.

```
1  // Lab Exercise 1: SimpleCalculator.cpp
2
3  #include "SimpleCalculator.h"
4
5  /* Write definition for add member function */
6
7  // function subtract definition
8  double SimpleCalculator::subtract( double a, double b )
9  {
10    return a - b;
11
12 } // end function subtract
13
14 // function multiply definition
15 double SimpleCalculator::multiply( double a, double b )
16 {
17    return a * b;
18
19 } // end function multiply
20
21 /* Write definition for divide member function */
22
```

Fig. L 10.2 | Contents of SimpleCalculator.cpp.

```
1  // Lab Exercise 1: CalcTest.cpp
2
3  #include <iostream>
4  using namespace std;
5
6  #include "SimpleCalculator.h"
7
8  int main()
9  {
10    double a = 10.0;
11    double b = 20.0;
12
```

Fig. L 10.3 | Contents of CalcTest.cpp. (Part 1 of 2.)

ECE 264, Object-Oriented Software Development

Lab Assignment 4

```
13     /* Instantiate an object of type SimpleCalculator */
14     cout << "The value of a is: " << a << "\n"
15          << "The value of b is: " << b << "\n\n";
16
17     /* Write a line that adds a and b through your SimpleCalculator
18        object; assign the result to a variable named addition */
19     cout << "Adding a and b yields " << addition << "\n";
20
21     double subtraction = sc.subtract( a, b );
22     cout << "Subtracting b from a yields" << subtraction << "\n";
23
24     double multiplication = sc.multiply( a, b );
25     cout << "Multiplying a by b yields " << multiplication << "\n";
26
27     /* Write a line that divides a and b through the
28        SimpleCalculator object; assign the result to a
29        variable named division */
30     cout << "Dividing a by b yields " << division << endl;
31  } // end main
```

Fig. L 10.3 | Contents of `CalcTest.cpp`. (Part 2 of 2.)

Problem-Solving Tip

1. All of SimpleCalculator's member functions should have return type double.

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.