CS 121 Lab 7

Programmer Defined Functions Paul Cao

In this lab, we will define our own function and use it to solve a graphics problem. The code is due before midnight on 10/26/2010 for CS121A and 10/28/2010 for CS121B.

Pre-lab

Read the following explanations. You don't have turn in anything for this prelab.

Programmer-Defined Functions

A function is a subprogram that is included in a C++ program to perform a particular task such as obtaining data, carrying out some calculations, displaying some information or messages, and displaying the output data. In C++ there are many predefined functions that are written to simplify the computations. In addition, we can write our own functions to perform some computations. These type of functions are referred to as programmer-defined functions. A programmer-defined function may have three parts: 1) Function definition, 2) Function Call, and/or 3) Function declaration.

The following program, which computes the total cost of purchases made in a store uses a function to compute the cost plus 7.5% sales tax of purchases made.

```
// P1.cpp - This program computes the total cost of purchases made,
// including 7.5% sales tax, on n items at a cost of p each.

#include <iostream>
using namespace std;
double totalCost( int n, double p); // (1) Function declaration
int main( )
{
    double price, bill;
    int number;

    cout << "Enter the number of items purchased: ";
    cin >> number;
    cout << "Enter the price per item $";
    cin >> price;

    bill = totalCost( number, price); // (2) Function call

    // The following three lines are used for formatting purposes. Since a precision of 2
    // is set, then all numbers will be displayed with two decimal points. We work with $
    // this seems to be the most appropriate way to display the numbers..
     cout.setf( ios::fixed);
     cout.setf( ios::showpoint);
     cout.precision( 2);

     cout << number << " items at " << "$" << price << " each.\n"
     << "Final bill, including tax, is $" << bill << endl;

     return 0;
}

// (3) Function definition

double totalCost( int n, double p)   // Function header
{
     // Function body begins here
     const double TAX_RATE = 0.075; // 7.5% sales tax, const is to make sure this value stays unchanged
```

```
    double subtotal;

    subtotal = p * n;
    return (subtotal + subtotal*TAX_RATE);
}// Function body ends here
```

As you can see, we have used a function called totalCost to compute the cost + 5% tax. Note that this function has computed the total cost and has returned a single value at return (subtotal + subtotal*TAX_RATE); The returned value is of type double.

Remarks:
When you work with functions, there are at least 4 things that you must remember. Violation of any of these will result in syntax or logical errors.

1. A function must have a name, in the above program the function name was totalCost. Don't name any variable that have the same name as the function.
2. A function must have a return type, the type for the above function is double.
3. A function must have correct argument definitions. This means that the arguments, if any, must have type consistency and correct ordering at the: a) function declaration, b) function call, and c) function definition.
4. A function must have the correct return type. In our example, the function was of type double and it returns a value of type double.

In-lab activity

Display an hourglass shape on the screen

Write a program that uses one programmer-defined functions, `drawHourglass`, to display an hourglass shape on the screen. `drawHourglass` receives two parameters: the height of the hourglass (an integer), and a brush character to be used for drawing the triangle (a char). The function doesn't return anything (void). You may assume that the width of the hourglass is the same as the height. Refer to the sample output for details about the shape.

In your main function, you should ask the user for those parameters (basically the height and the brush char) and then call the `drawHourglass` function. In your `drawHourglass` function, you should first output the upper edge of the hourglass. Then draw a cross on the screen. In the end, you should output the lower edge of the hourglass. Your program should be able to repeat this process as long as the user wants to. You can assume that the height of the hourglass is greater than 3.

I have written a function called `drawCross` and it is available to you upon request. It accepts the height and the brush as input and will draw a cross on the screen. You can use this function. **But you may receive a 100% extra credit for this lab if you write your own code to generate the cross. Please let me know if you want to use this function via email. I will email you the drawCross code once I get your request.**

Sample Output

```
This code will display an hourglass shape on the screen.
Please input the height of the hourglass (an integer greater than 3): 7
Please input a paint character: *
*******
 *   *
  * *
   *
  * *
 *   *
*******
Do you want to draw again? y
This code will display an hourglass shape on the screen.
Please input the height of the hourglass (an integer greater than 3): 6
Please input a paint character: &
&&&&&&
 &  &
  &&
  &&
 &  &
&&&&&&
Do you want to draw again? n
```