Lab 04: Temperature Conversion Functions

CSci 515 Spring 2015

2015-01-23

Dates:

Due: In Lab, Wednesday February 11, by 4:10 pm (lab end time)

Objectives

- Write functions in C that take 1 or more parameters and return a result.
- Learn about breaking up a larger problem into smaller sub-parts.
- Create simple functions that use an input parameter.
- Learn how to return results using the return values from a function.
- Practice with input and output.

Description

In this lab we will write two simple functions. The functions will take a single parameter as input, and return a single value as their result. You are to write functions that convert temperatures on the Fahrenheit scale into the Celsius scale, and vice verse.

The formula for converting a temperature from Fahrenheit to Celsius is given by:

$$C = (F - 32) \cdot \frac{5}{9}$$

And to convert in the other direction, from Celsius to Fahrenheit:

$$F = C \cdot \frac{9}{5} + 32$$

Perform the following tasks:

1. Write a function named `fahrenheitToCelsius()`. The function takes a single float value as input, and returns a single float value as its result. The input of course represents a temperature in degrees Fahrenheit, and your program should use the above formula to convert the value to a temperature on the Celsius scale.

2. Write a function named `celsiusToFahrenheit()`. The function takes a single float values as input, and returns a single float value as its result. As you can guess, this function performs the other operation, to convert from the Celsius scale back to the Fahrenheit scale.

3. In your `main()` function, prompt the user for a temperature on the Fahrenheit scale. Use the first function to convert this value to the Celsius scale and display this to the user. Then prompt the user for a temperature in degrees Centigrade, and use your second function to convert it and display your results.

Your program output should look exactly like the following when I run your program:

```
Enter a value in degrees Fahrenheit, and I will convert it to the Celsius scale: 6
6 degrees Fahrenheit is equal to -14.4444 degress Celsius

Enter a value in degrees Celsius, and I will convert it to the Fahrenheit scale: 22
22 degrees Celsius is equal to 71.6 degress Fahrenheit
```

NOTE: Now that our programs have more functions than just the `main()` function, the use of the function headers becomes meaningful and required. Make sure that all of your functions (`main`, `fahrenheitToCelsius`, `celsiusToFahrenheit`) have function headers before them that document the purpose of the functions, and the input values and return value of the function.

Lab Submission

An eCollege dropbox has been created for this lab. You should upload your version of the lab by the end of lab time to the eCollege dropbox named

`Lab 04 Temperature Conversion`. Work submitted by the end of lab will be considered, but after the lab ends you may no longer submit work, so make sure you submit your best effort by the lab end time in order to receive credit.

Requirements and Grading Rubrics

Program Execution, Output and Functional Requirements

1. Your program must compile, run and produce some sort of output to be graded. 0 if not satisfied.

2. 40+ pts. Your program must have the 2 required named functions, that accept the required input parameters and return the required values (if any).

3. 30+ pts. Your functions must convert temperatures correctly, according to the formula given for you for converting between the two scales.

4. 30+ pts. You should prompt the user for values in your `main()` function, and display the results. The interaction with your program should be as shown in the example output above.

Program Style

Your programs must conform to the style and formatting guidelines given for this course. The following is a list of the guidelines that are required for the lab to be submitted this week.

1. The file header for the file with your name and program information and the function header for your main function must be present, and filled out correctly.

2. A function header must be present for all functions you define. You must document the purpose, input parameters and return values of all functions.

3. You must indent your code correctly and have no embedded tabs in your source code. (Don't forget about the Visual Studio Format Selection command).

4. You must not have any statements that are hacks in order to keep your terminal from closing when your program exits.

5. You must have a single space before and after each binary operator.

6. You must have a single blank line after the end of your declaration of variables at the top of a function, before the first code statement.

7. You must have a single blank space after , and ; operators used as a separator in lists of variables, parameters or other control structures.

8. You must have opening { and closing } for control statement blocks on their own line, indented correctly for the level of the control statement block.

Failure to conform to any of these formatting and programming practice guidelines for this lab will result in at least 1/3 of the points (33) for the assignment being removed. Failure to follow other class/textbook programming guidelines may result in a loss of points, especially for those programming practices given in our Deitel textbook that have been in our required reading so far.