EE2510 Introduction to Object-Oriented Programming
Sections 21, Spring 2018
Professor: Joshua D. Carl, Ph.D.

Lab 1: Basic C++ Programming

Goals:
1. Become familiar with functional programming in C/C++
2. Become familiar with Eclipse for developing C++ desktop applications

Grading:
Turn in commented source code files on Blackboard by 8:00 AM the day after your lab period Week 2 (March 16 depending on your section). There is no demo associated with this lab, but you must turn in all source code files implementing the assignment described below, and a text file from the Windows command line that demonstrates the full and correct functionality of the program. Submit your code as one PDF file.

Limits: You may not change the attached `*.h` file, and you may not use global variables to complete this assignment.

Assignment Description:
Create a calculator using C++. You must write a `main` function that presents the different calculation options to the user, allows the user to select an option, and then gets the necessary inputs from the user to perform the specified operation. You must also implement the functions to perform the calculations. The calculation options are:
1. Perform simple math on scalars: add, subtract, multiply, divide, and exponent
2. Average all of the values in a user defined array
3. Find the largest and smallest values in a user defined array
4. Calculate the effective series or parallel resistance of any number of resistors

The function prototypes for each calculation option are attached to this assignment. You may not change the supplied function prototypes. You must implement the functions your `main` function will call and any additional functions that may be required/useful. The functions you must implement are `my_basic_math`, `my_average`, `my_find_large_small`, and `my_series_parallel`. The description of the inputs and outputs of the functions are in the `CalculatorFunctions.h` header file. The functions you create/implement should not output unnecessary information to the Windows command line.

When you turn in your assignment, turn in your commented source code files (be sure to include good comments) for your `main` function and the four `my_*` functions, and a copy of the Windows command line output showing that the functions you wrote work properly. Submit everything as a single PDF document.

```
/*
 * CalculatorFunctions.h
 *
 * Functions that need to be implemented as a part of EE2510 Lab 1
assignment.
 *
 *  Created on: Feb 22, 2017
 *      Author: Joshua D. Carl, PhD
 */

#ifndef CALCULATORFUNCTIONS_H_
#define CALCULATORFUNCTIONS_H_


// Name:          my_basic_math
// Purpose:       Performs basic math operations.
// Return:        0 if successful
//                      not-0 if error
// Parameters:    int - input - operation to perform:
//                                          1 - addition
//                                          2 - subtraction
//                                          3 - multiplication
//                                          4 - division
//                                          5 - exponent
//                      int - input - first operand
//                      int - input - second operand
//                      float* - output - pointer to memory where
//                                          result can be stored
int my_basic_math(int, int, int, float*);


// Name:          my_average
// Purpose:       Averages the values in an array.
// Return:        0 if successful
//                      not-0 if error
// Parameters:    int[] - input - array of data
//                      int - input - length of array
//                      float* - output - pointer to memory where
//                                          result can be stored
int my_average(int[], int, float*);


// Name:          my_find_large_small
// Purpose:       Finds the largest and smallest values in an array.
// Return:        0 if successful
//                      not-0 if error
// Parameters:    int[] - input - array of data
//                      int - input - length of array
//                      int* - output - pointer to memory where largest
//                                          value in array can be stored
//                      int* - output - pointer to memory where smallest
//                                          value in array can be stored
int my_find_large_small(int[], int, int*, int*);


// Name:          my_series_parallel
// Purpose:       Finds the equivalent resistance of a series or parallel
```

```
//                      sequence of resistors.
// Return:        0 if successful
//                      not-0 if error
// Parameters:  int[] - input - array of resistor values
//                      int - input - length of array
//                      int - input - resistor configuration
//                                    1 = series
//                                    2 = parallel
//                      float* - output - pointer to memory where equivalent
resistance
//                                                value can be stored
int my_series_parallel(int[], int, int, float*);


#endif /* CALCULATORFUNCTIONS_H_ */
```