CptS 121 - Program Design and Development

Programming Assignment 1: Equation Evaluator

Assigned: Monday, August 31, 2020
Due: Wednesday, September 9, 2020 by midnight

I. Learner Objectives:

At the conclusion of this programming assignment, participants should be able to:

- Analyze a basic set of requirements for a problem and derive logical solutions to them
- Declare variables
- Apply C data types and associated mathematical operators
- Comment a program according to class standards
- Logically order sequential C statements to solve small problems
- Compose a small C language program
- Compile a C program using Microsoft Visual Studio Community 2019
- Execute a program
- Create basic test cases for a program

II. Prerequisites:

Before starting this programming assignment, participants should be able to:

- Access Microsoft Visual Studio 2019 Integrated Development Environment (IDE)
- Summarize topics from Hanly & Koffman Chapters 1 – 2 including:
  - The steps of the software development method
  - C language elements (preprocessor directives, reserved words, and standard identifiers)
  - The standard C data types
  - The general form of a high-level program

III. Overview & Requirements:

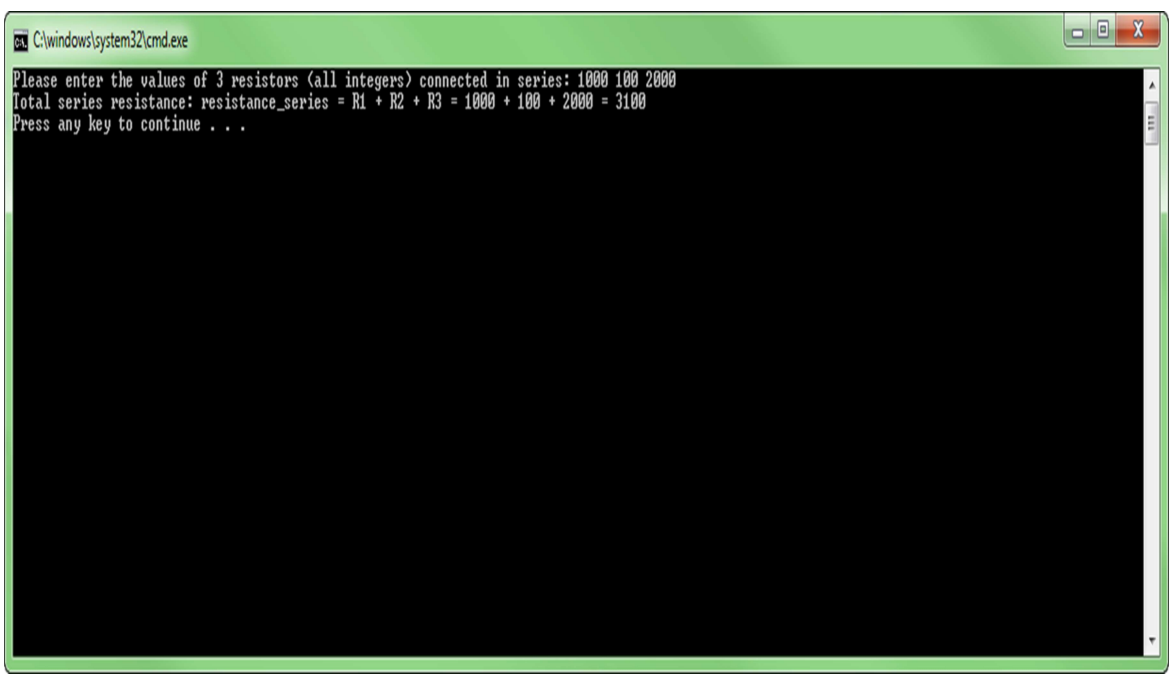
Write a C program that evaluates the equations provided below. The program must prompt the user for inputs to each equation and evaluate them based on the inputs. All equations should be placed into a single .c file. This means you should ***NOT*** have 7 Visual Studio projects or 7 .c files. All variables on the right-hand sides of the equations must be inputted by the user. All variables, except for the *plaintext*_character, *encoded_character*, variable *a*, *shift_int*, *R1*, *R2*, and *R3* are floating-point values. The *plaintext_character*

and *encoded_character* variables are characters, and the *a*, *shift_int*, *R1*, *R2*, and *R3* variables are integers. F should be defined as a constant macro (#defined constants). Error checking is not required for your program. You do ***NOT*** need to check for faulty user input or ***dividing by zero***.

1. Total series resistance: series_resistance = R1 + R2 + R3, for 3 resistors. *R1*, *R2*, and *R3* are integers.
2. Sales tax: total_sales_tax = sales_tax_rate * item_cost (note: it's OK to show the result beyond the hundredths place, we don't know how to show to the hundredths place yet)
3. Volume of a right rectangular pyramid: volume_pyramid = (l * w * h) / 3, where l and *w* are the length and width of the base, respectively, and *h* is the height of the pyramid.
4. Total parallel resistance: parallel_resistance = 1 / (1 / R1 + 1 / R2 + 1 / R3), for 3 resistors. *R1*, *R2*, and *R3* are integers.
5. Character encoding: encoded_character = (plaintext_character - 'a') + 'A' - shift_int; *shift_int* is an integer (note: what happens if plaintext_character is lowercase? What happens with various *shift_int* values? Please use the ASCII table to help you understand how to interpret the encoded character produced.)
6. Distance between two points: distance = square root of $((x_1 - x_2)^2 + (y_1 - y_2)^2)$ (note: you will need to us sqrt ( ) out of <math.h>)
7. General equation: y = y / (3/17) - z + x / (a % 2) + PI (recall: *a* is an integer; the 3 and 17 constants in th equation should be left as integers initially, but explicitly type-casted as floating-point values)

IV. Expected Results:

The following console window illustrates inputs and outputs that are appropriate for your program. Your program must display the results in a similar form as shown in the window. The window shows possible results for the given input tests, for the first two equations only.



Note: you will need to display the results for all of the equations!

V. Submitting Assignments:

1. Using Blackboard Learn https://learn.wsu.edu/webapps/login/ submit your assignment. You will submit your assignment in the ***lab*** Blackboard space. Under the "Content" link navigate to the "Programming Assignment Submissions" folder and upload your solution to the appropriate "Assignment" space. You must upload your solution, through an attachment, as <your last name>_pa1.zip by the due date and time.

2. Your .zip file should contain your one source file (.c file), and project workspace. Delete the two debug folders before you zip the project folder.
3. Your project must build properly. The most points an assignment can receive if it does not build properly is 65 out of 100.

VI. Grading Guidelines:

This assignment is worth 100 points. Your assignment will be evaluated based on a successful compilation and adherence to the program requirements. We will grade according to the following criteria:

- 5 pts for correct declaration of constant macro(s)
- 35 pts for proper prompts and handling of input (5 pts/equation)
- 49 pts for correct calculation of results based on given inputs (7 pts/equation)
- 11 pts for adherence to proper programming style established for the class and comments