CS 135

Design Assignment 2 (DA2-09/20)

Programming Assignment 3 (PA3-09/24)

As specified in your syllabus, you must turn your assignments in by 6:00 pm on the due date specified. If it is turned in late, but prior to 12:00 midnight the day it is due, credit will be reduced by 50% of the earned score. Any laboratories turned in more than 6 hours late will not earn any credit.

Objectives:

1) You will use a set of standardized functions to implement console I/O operations in a formatted command line system
2) You will use global constants to assist with program clarity
3) You will use a systematic development process to create a program
4) You will create simple functions as part of applying program modularity
5) You will implement mathematical operations, including mixed-mode mathematical operations, in order to solve problems
6) You will use math library functions as needed to calculate values

Special Instructions:

Using the Six Step Programming Process to Create Your Programs

1) For this assignment, you will begin using the complete Six Step Programming Process. You have been working with the programming process without function development which stops at step 4. Now you will develop your own functions and complete the fifth step which involves developing function design specifications for the supporting functions, and the sixth step which involves implementing the code for those functions.

2) Also for this assignment, you will need to download and use the **new** formatted command line header (**`formatted_cmdline_io_v08.h`**). This has been updated for use in this program and will be used from now on in the course. Your program name will be **`relevance.cpp`**.

Tasks:

*Calculating Capacitance for RL Relevance**

1) Back story. Any wiring in electrical systems may generate magnetic fields and in doing so, they store a small amount of energy. This effectively makes the wires themselves look like an electronic component called an ***inductor***. Now if you have thousands of miles of wire, as is found in an electrical grid, you have the equivalent of a large-scale inductor. If you have other components in the system that act like ***capacitors*** by storing electrical charges, you may have conditions supporting relevance which is a naturally-occurring frequency response related to these components, much like resonance, resistance, reactance, or reluctance*.

2) In some cases relevance is a good thing. In other cases, it is a bad thing. Either way, Engineers may need to know the capacitance that would lead to a system's relevance, and in many cases they need to know if their electrical system is within a certain range of relevance. Your program will calculate the ***capacitance*** it would take to create relevance at a given frequency as well as the capacitance for 50% below the frequency up to 50% above the frequency.

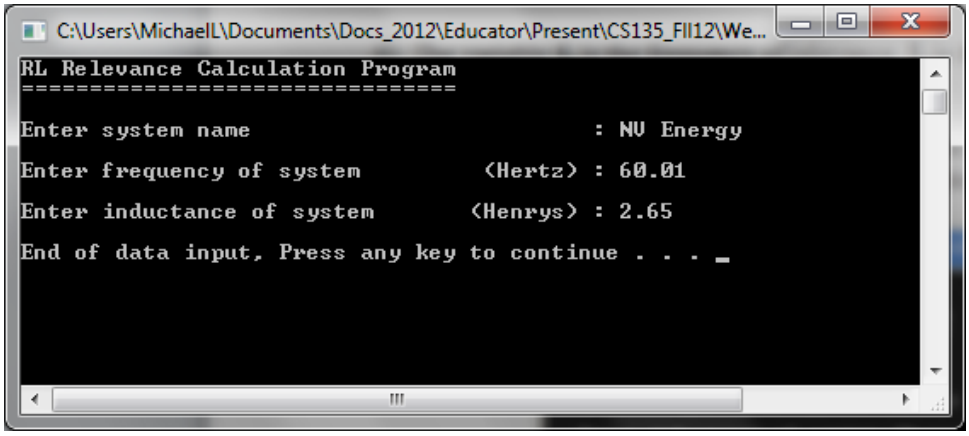
3) The equation you will need to use for this process is shown here.

$$C = \left| \frac{1}{\cos(4\pi^2 f_r^2 L)} \right|$$

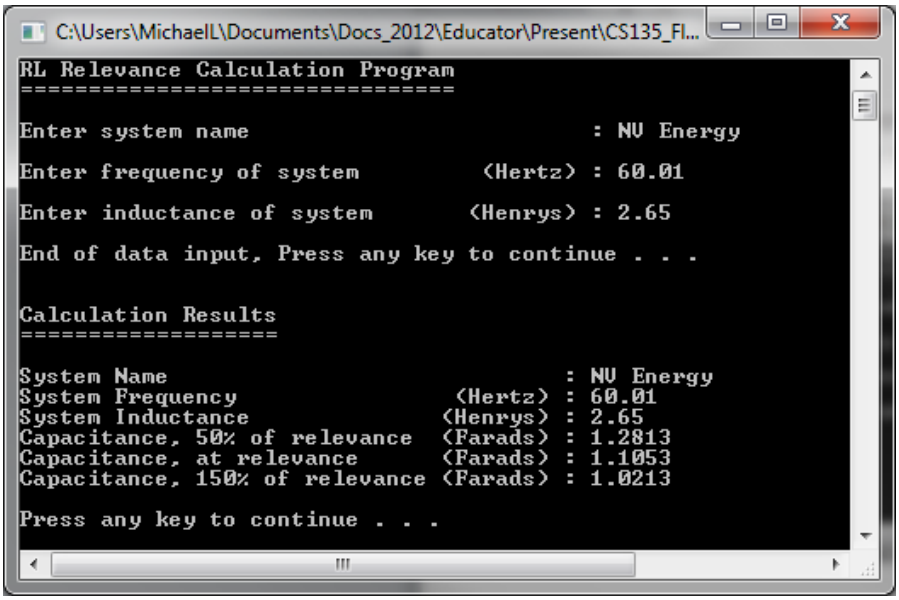
4) The variable $\mathbf{f_r}$ is the frequency of relevance, **L** is the inductance of a system in Henrys, and **C** is the capacitance of a system in Farads.

*Note: unlike resonance, resistance, reactance, reluctance, and other Electrical Engineering terms, relevance is completely made up so we could invent our own equation for you to use.

5) Your program will accept as input the name of the system, the inductance of the system, and the operating frequency of the system. An example is shown here.



6) Once the user has entered the appropriate data, and then pressed a key, the following formatted results are displayed. Note the differences in precision between the input data validation display and the calculated values. You must also use the same justification of the various displayed quantities as is shown here.



7) This will be your first program with functions, but these are all very simple functions. You are ***required*** to create the following functions. You may create others, but be judicious and effective.

   a) **`printTitle`** - prints the initial program title and the underline

b) **`calcCapAtRelevanceFreq`** - conducts the capacitance calculation at the specified frequency (make sure you break the equation into at least three smaller expressions rather than creating one monolithic math expression)

c) **`displayResultHeader`** – displays the title and underline for the result output

d) **`printResultString`** – displays the "System Name" reference and the system name that was entered by the user

e) **`printResultData`** - prints the two data items for one line of results at their respective locations on one line; this is the function that will accept the reference name (e.g., "System Frequency", "System Inductance", etc, the related value (e.g., 60.01 and 2.65, respectively for the data shown in the screen shot above), and the precision required for that particular line of display. For example, the precision is 2 for the frequency and inductance, but it is 4 for the three displayed capacitances. This ***one*** function must adapt to these differing circumstances.

8) While the mathematical calculations are not difficult for this assignment, you are ***required*** to develop easily readable and maintainable code for this, so you should plan to break the calculations down into two or more modular components. Note also that since you will be using some advanced mathematical functions, you will need the math library header in your program.

9) If you have any questions for following the six step process, review the pertinent sections of the online reference and the online code examples and videos, or check with the CS 135 Instruction Team.

10) You will develop a Design Assignment which includes the step 1 through 5 source code files by Thursday at 6:00 pm, and then you must develop the program code for next Monday at 6:00 pm. Further explanation of the required components and uploading process are provided near the end of this document.

11) You must also acquire three screen shots of the input and results of your program operation. The first screen shot should show the input used in the example shown previously in this document. The other two should show examples of other data input used. Again remember to annotate your screen shot document.

Turning in your Design Assignment:

Information:

Week: 4
Laboratory: 4
Design Assignment: 2
Due Date: 09/20, 6:00 pm

To turn in:

The first five steps of the Six Step Programming Process, including:

1. relevance_s1.cpp
2. relevance_s2.cpp
3. relevance_s3.cpp
4. relevance_s4.cpp
5. relevance_s5.cpp

Upload these as separate files.

For information on how to turn in Design Assignments, refer to the "How to Turn in Design Assignments" in the "General Course Information" folder

Turning in your Programming Assignment:

Information:

Week: 4
Laboratory: 4
Programming Assignment: 3
Due Date: 09/24, 6:00 pm

To turn in:

1. The Word file containing the following:
    a. There should be at least three (3) screenshots for the programs as specified in item #11 previously in this document
    b. Remember to clearly annotate every displayed result

2. The executable file:
    a. relevance.exe (relevance_s6.exe is acceptable)

3. The source code file:
    a. relevance_s6.cpp

These files must be compressed and uploaded as one zip file. To do this, select all of the required files, right click on them, and select “Send To”, then select “Compressed (zipped) Folder”.

Once the folder is created, it will be placed in the same folder in which you are working. Change the name of the zipped folder to “LastnameFirstname_PAX” (where ‘X’ is the number of the Programming Assignment) as shown in the following example: “LeveringtonMichael_PA3” (no quotes). After you have renamed the zipped folder, double click on it to verify that it has all the files it is supposed to have.

For information on how to turn in Programming Assignments, refer to the "How to Turn in Programming Assignments" in the "General Course Information" folder