Software and Embedded System Lab 2 (ELEE08022)

Writing Your Own Functions in C language

1. Write a **`main()`** function of a program to read in three numbers and print out the *minimum* of three values. Type in this program and add the function **`find_min()`** which will return the smallest of its three arguments.

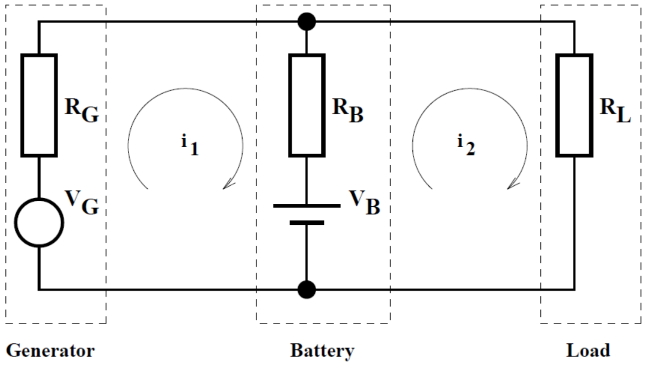
2. Write a function **`find_min()`** which takes two arguments, the first being an array of floating point numbers and the second an integer value representing the number of elements in the array. Your function should return the value of the *smallest* element in the array. Write a **`main()`** function to take 10 floating numbers, then pass two arguments to **`find_min()`**, finally print the minimum element out.

3. Develop (in your daybook) a function **`is_triangle()`** which takes three **`integer`** arguments representing the sides of a triangle and returns a value of **1** (*true*) if the triangle is valid or **0** (*false*) if it is not. Note that for a triangle to be valid the *sum* of *any two* sides must always be *greater than* the *third* side. Now write a **`main()`** function to test the operation of your **`is_triangle`** function, enter your program into the computer, compile and test it.

4. Create a random number generator function and incorporate it in a C program that produces a series of 10 random numbers. There are many mathematical and engineering problems in which probability must be considered or statistical sampling techniques must be used.

   One method of generating random numbers is the power residue method. In one version of this method, a suitable *n* digit "seed" number, where *n* is an even number, for example 6, is multiplied by the value ($10^{n/2}$ - 3). Using the lowest *n* digits of the residue produces a new seed. Continuing this procedure produces a series of random numbers, with each new number used as the seed for the next number. If the original seed has 6 digits and is not divisible by either 2 or 5, this procedure yields $5x10^{(n-2)}$ random numbers before a sequence of numbers repeated itself. Write a function **`rand()`** to generate one random number and call it 10 times in **`main()`** to generate 10 random numbers and print them out.

5. A simplified equivalent circuit for charging a vehicle battery is given below. The generator (normally an alternator) is driven by the engine and supplies the power to recharge the battery. The voltage produced by the generator (V_G) is a (complex) function of engine speed and the battery voltage (V_B) depends on the state of charge of the battery. The value of the load resistance (R_L) depends on what equipment (lights, radio etc) is operating in addition to the ignition circuit - we can assume *RL* is constant at any given time. The internal resistances of the generator and battery are R_G and R_B respectively



Using Kirchoff's Laws with two loop currents i_1 and i_2 gives:

$$-V_G + R_G\, i_1 + R_B\,(i_1 - i_2) + V_B = 0$$
$$-V_B + R_B\,(i_2 - i_1) + R_L\, i_2 = 0$$

Rearranging these to be similar to the standard form for simultaneous equations used above gives:

$$(R_G + R_B)\, i_1 - R_B\, i_2 = V_G - V_B$$
$$-R_B\, i_1 + (R_B + R_L)\, i_2 = V_B$$

If we want to determine whether the battery is being charged or is discharging its energy, we simply need to determine sign of the net current flowing from the battery ($i_2 - i_1$). A positive value means the battery is discharging and a negative value means that the battery is charging.

Set the value of V_B as 12.0 V, the value of R_G as 0.15 Ω, the value of R_B as 0.1 Ω and the value of R_L as 1.0 Ω. Write a C program to print out the values of i_1 and i_2 at different values of V_G between the starting value 12.0 V and final value 15.0 V in 0.1V interval. Your program also should output a message to tell if battery is charging or discharging at each value of V_G.

Hint: to find unknowns x_1 and x_2 in two linear equations,

$$a_{11}\, x_1 + a_{12}\, x_2 = c_1$$

$$a_{21}\, x_1 + a_{22}\, x_2 = c_2$$

use Cramer's Rule, which gives solutions by ratio of determinants:

$$x_1 = \frac{\begin{vmatrix} c_1 & a_{12} \\ c_2 & a_{22} \end{vmatrix}}{\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}} \qquad \text{and} \qquad x_2 = \frac{\begin{vmatrix} a_{11} & c_1 \\ a_{21} & c_2 \end{vmatrix}}{\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}}$$

Write a function to calculate determinant and reuse it in the for loop of **`main()`**.