Engineering 6
Winter 2012
Homework 7 (all topics except GUI and classes)
Issue: Tuesday, February 14th at 10am.
Due: Tuesday, February 21st at 10am.

General Instructions:

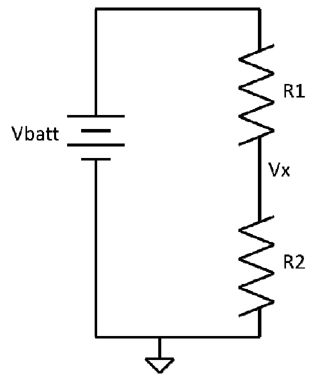
- **Submission Guideline: Zip all files into a single .zip file. Name it HW7_XXX.zip, where XXX are your initials.**
    - **See submission checklist below.**
- For each problem, save the test script as test_HW7P**#**_XXX.m where # after P = the problem number, and XXX are your initials.
- Use comments to designate each task of the problem within your script and to explain your steps. **Consult the MatLab Code Style Guide posted on SmartSite.**
- Use the functions 'disp()' **or 'fprintf'** to display answers for each task.
- Do not leave semi-colons off of lines of code as a means to print out answers.
- Check the honor pledge. Late submissions are not accepted.

Submission checklist: (A single .zip file)

- **Problem 1: One MatLab script, containing all 3 tasks.**
- **Problem 2: One MatLab script, containing both tasks.**
- **Problem 3: One MatLab script. One function.**
- **Problem 4: One MatLab script.**

Problem 1: Resistor Divider

A resistor divider circuit is shown below. R1 and R2 are resistors, V_{batt} is the voltage of the battery, and V_x is the voltage of the node in between R1 and R2.



The voltage, V_x changes depending on the values of R1 and R2. If R1 is equal to R2, then V_x is simply half of V_{batt} (energy supplied by the battery is being evenly dissipated through the two resistors). As R1 and R2 vary, V_x also varies. The trends are described in the table below.

R1	**R2**	V_x
Fixed	Increasing	Increasing (more energy dissipated by R2 than R1)
Fixed	Decreasing	Decreasing (more energy dissipated by R1 than R2)
Increasing	Fixed	Decreasing (more energy dissipated by R1 than R2)
Decreasing	Fixed	Increasing (more energy dissipated by R2 than R1)

The relationships described table above can be accurately predicted by the following formula:

$$V_x = V_{batt} * \left(\frac{R2}{(R1 + R2)}\right)$$

Task 1 (2 pts.): If R1 is 1 kOhm and R2 is 1 kOhm, calculate V_x using V_{batt}= 9 volts.

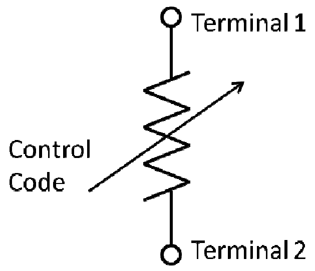
Task 2 (2 pts.): If R1 is 10 kOhm and R2 is 2 kOhm, calculate V_x using V_{batt}= 9 volts.

Task 3 (2 pts.): Let R1 be a vector of resistances that ranges from 1 kOhm to 100 kOhm at steps of 1 kOhm, and R2 is also an 100 element vector with resistances that ranges from 50 kOhms to 100 kOhms at equally spaced increments (hint: linspace()).

Calculate V_x using the 100 element vectors, R1 and R2. Assume V_{batt} is 9 volts. Write code to plot V_x, R1, and R2 on the same plot, using red, blue, and green, respectively.

Problem 2: Variable Resistor

A variable resistor changes the resistance across its terminals (terminals 1 and 2 shown below) based on the control code set by the user.



The control code for a particular variable resistor can range from 0 to 255, at increments of 1, while the resistance across the terminals varies from 60 to 10000 Ohms. The formula that converts from control code to a resistance value is as follows:

$$Resistance \ = Control\ code * \left(\frac{10000 - 60}{256}\right) + 60$$

Task 4 (2 pts.): Write code (without using any loops) to compute the resistances for a sequence of control codes that ranges from 0 to 255, at increments of 1.

Task 5 (2 pts.): In the resistor divider circuit shown previously in Problem 1, R2 is actually a variable resistor with control code ranging from 0 to 255, at increments of 1 (and follows the formula above). If R1 is a resistor with 5 kOhms of resistance, and V_{batt} is 3 Volts,

1) Write code to calculate V_x for all control codes (0 to 255).
2) Write code to plot V_x (y-axis) versus control code (x-axis).

Problem 3: Converting Binary Numbers to Decimal Numbers

Numbers we use every day are known as decimal system, or base 10 (meaning the weight of successive digits is 10) system. Additional numerical representations exist that use other bases. For example, binary number representation is a base 2 system. Binary number representation is typically used inside computers to represent numerical data.

Below is a table that illustrates the difference between binary and decimal systems. The top table shows how to convert decimal number 2307 is constructed using weights. The bottom table shows how binary number 1011 is represented, and how its numerical value is calculated.

Note: For both cases, the numerical value is calculated by taking the sum of the element-wise multiplication between the entries in the **Digit** row, and the **Weight** row.

Base 10 (Decimal) representation of 2307				
Digit	2	3	0	7
Weight	1000 (=10^3)	100 (=10^2)	10 (=10^1)	1 (=10^0)

Numerical value = 1000*2 + 100*3 + 10*0 + 1*7 = 2000 + 300 + 0 + 7 = 2307

Base 2 (Binary) representation of 4-bit binary number, 1011 (decimal value: 11)				
Digit	1	0	1	1
Weight	8 (=2^3)	4 (=2^2)	2 (=2^1)	1 (=2^0)

Numerical value = 8*1 + 4*0 + 2*1 + 1*1 = 8 + 0 + 2 + 1 = 11

Again, as you can see, converting a binary number to a decimal number involves element-wise multiplication of the weight and digit vectors, then summing the resulting vector.

Task 6 (1 pt.): For a "Base X" (X is 2 for binary system. X is 10 for decimal system) system, the weight is simply a vector X^k,where the exponent, k is a linearly decreasing vector, from N to 0, where N is the number of bits-1. Example, for an 8-bit binary number, N will be 8-1=7.

Create a vector called **weight** that contains the weight for an 8-bit binary number.

Task 7 (1 pt): Write a function (name it "binary2dec") that converts a binary (base 2) number into a decimal (base 10) number. Your function will take one input argument, and return one output. The input argument will be the binary number, and it will be a 1D vector. For example, the binary number, 1111111 will be entered into your program as [1 1 1 1 1 1 1 1].

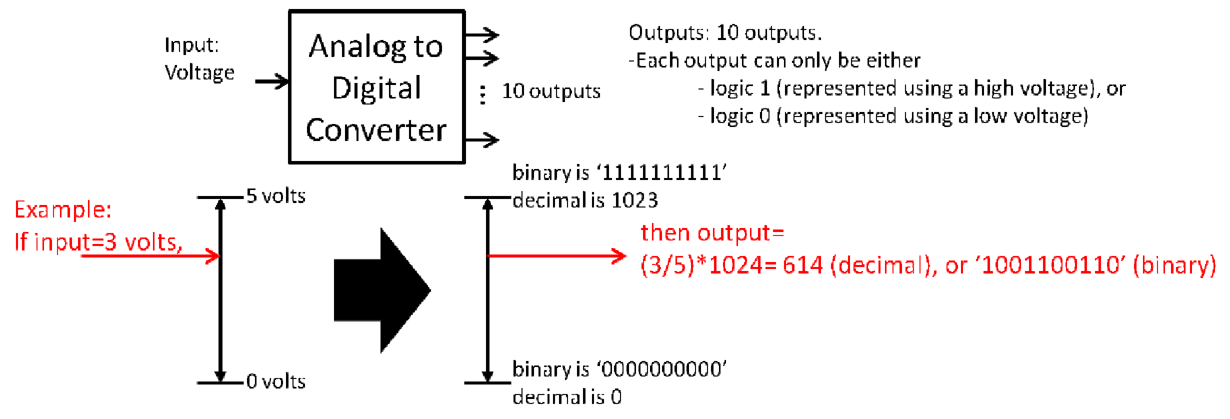
Test your function using the following 4 binary numbers.

	Binary Number (most significant bit… least significant bit)								**Equivalent Decimal Value**
No. 1	1	1	1	1	1	1	1	1	**255**
No. 2	0	0	0	0	0	0	0	0	**0**
No. 3	1	0	1	0	1	0	1	0	**170**
No. 4	0	1	0	1	0	1	0	1	**85**

Problem 4: Analog to Digital Converter

An Analog to Digital Converter (ADC) converts a (analog) voltage signal to a (digital) binary number. The conversion from voltage to (digital) binary numbers depends on the ADC's input voltage range and the ADC's resolution.

For example, an ADC with an input voltage range from 0 to 5 volts, with 10-bits of resolution is shown in figure below. This ADC converts 0 volts at its input to a 10 bit binary number with all zeros at its output, and 5 volts at its input to a 10 bit binary number with all ones at its output. An example illustrating the conversion process is shown in red in the figure. 3 volts at the ADC input is converted to '1001100110' at the ADC's output, which is 614 in decimal.


Input:
Voltage
Analog to Digital Converter
10 outputs
Outputs: 10 outputs.
-Each output can only be either
- logic 1 (represented using a high voltage), or
- logic 0 (represented using a low voltage)
Example:
If input=3 volts,
5 volts
0 volts
binary is '1111111111'
decimal is 1023
then output=
(3/5)*1024= 614 (decimal), or '1001100110' (binary)
binary is '0000000000'
decimal is 0

The above mapping relationship can be written as the formula below.

$$Voltage\ (analog) = Decimal\ number * 5 / 1024$$

Task 8 (2 pts.): Use the function you write in Task 7 to convert the <u>first two</u> binary numbers from the table below. Then write code (without using any loops) to compute the equivalent voltage for those two numbers using formula above. (FUNCTION)

	Binary Number (most significant bit… least significant bit)								**Equivalent Decimal Value**
No. 1	0	0	1	0	1	1	0	1	**45**
No. 2	0	1	0	1	1	1	0	0	**92**
No. 3	1	1	0	0	0	0	0	1	**193**
No. 4	1	1	1	0	1	1	1	1	**239**
No. 5	1	1	1	0	0	1	0	1	**229**
No. 6	0	1	1	1	1	1	0	0	**124**
No. 7	0	0	0	1	1	1	0	1	**29**
No. 8	0	0	1	0	1	0	1	1	**43**
No. 9	0	1	0	1	1	1	0	1	**93**
No. 10	1	1	0	1	1	0	1	1	**219**
No. 11	1	0	1	0	1	1	1	1	**175**
No. 12	0	1	0	1	0	1	0	0	**84**

Task 9 (2 pts.): First enter the 12 binary numbers into a 2-dimensional matrix. Then write code (use a loop) to compute the analog voltages for the sequence of 12 binary numbers shown in table above. Plot the sequence of 12 numbers.