Homework M6:

Matlab Primer [1] page numbers:
Conditional Control, pp. 5-2 to 5-5
Loop Control, pp. 5-5 to 5-8
Scripts and Functions, pp. 5-10 to 5-13

1. Write a Matlab® function called `standard_R.m` that prints out the value and color code of the closest standard value resistor to a given input value. Round to the next higher resistance if the value is halfway between standard values. An example of the function's use is as follows:

```
>> R = standard_R(115)   % Find closest standard value.
R =
   120
```

The following table shows what value each color represents.

Color	black	brown	red	orange	yellow	green	blue	purple	gray	white
Value	0	1	2	3	4	5	6	7	8	9

The formula for the color code is to turn the first two colors into a two-digit number multiplied by 10 raised to the power represented by the color of the third band.

2. Write a Matlab® function called `deg_rad.m` that has as input argument a number representing degrees and outputs the input value converted to radians. An example of its use:

```
>> deg_rad(30)   % 30 degrees to be converted to radians (which is pi/6 = 0.5236)
ans =
   0.5236
```

3. Write a Matlab® function called `trim_sig.m` that makes two arrays the same length by shortening the longer one. An example of its use:

```
>> sig1 = [2, 4, 5];   % Horizontal array, 3 elements
>> sig2 = [1, 3];      % Horizontal array, 2 elements
>> [sig1_trim, sig2_trim] = trim_sig(sig1,sig2)
sig1_trim =
     2     4
sig2_trim =
     1     3
```

4. Write a Matlab® function called `delay_sig.m` that has two input arguments: an array name, and the number of samples by which to delay the array in the first argument. Zeros are inserted at the front of the delayed signal, and the end of the delayed signal is cut off so the resulting delayed array has the same length as it had before being delayed. An example of `delay_sig`'s use:

```
>> sig = [2, 4, 5, 1, 0, 3];   % Horizontal array, 6 elements in this case.
>> delay_sig(sig,2)    % Delay "sig" by 2 samples
ans =
     0     0     2     4     5     1
```

Call your `trim_sig` function from problem 3 from inside your function as part of your solution.

5. Write a Matlab® function called `iphasor.m` that has two arguments: a complex number representing a phasor, and a value representing frequency omega. Its output is a string that states what the inverse phasor is. An example of the use of `iphasor`:
   ```
   >> iphasor(3 + 4i, 100)   % Phasor is 3+4i, frequency is 100 rad/sec
   ans =
        5 cos(100t + 53.13 deg)
   ```
   Note that `5 = sqrt(real(3+4i)^2 + imag(3+4i)^2)` and
   `53.13 = atan2(imag(3+4i),real(3+4i))*180/pi`

   Note that the output is a string you will build by concatenating strings. For example, if we wanted to print out `5 cos(100t)`, where `A = 5` and `omega = 100`, we could use the following code:
   ```
        out_str = [num2str(A), 'cos(', num2str(omega),'t)'];
   ```

6. Use for loops to write a Matlab® function called `my_transpose.m` that takes the transpose of its single argument. An example of the use of `my_transpose`:
   ```
   >> A = [1, 2; 3, 5; 0, 1]; % A = 3 x 2 matrix with 3 rows: 1 2, then 3 5, then 0 1.
   >> my_transpose(A)
   ans =
        1     3     0
        2     5     1
   ```

Use the following code in problems 7-10:
```
function [x1, x2] = quad_eqn(a,b,c)
% Function to compute roots of quadratic equation a*x^2 + b*x + c = 0.
% Use: [x1, x2] = quad_eqn(a,b,c)

D = sqrt(b^2 - 4 * a * c);   % Discriminant
x1 = (-b + D)/(2 * a);
x2 = (-b - D)/(2 * a);

return
```

7. If you type `help quad_eqn` at the command prompt, what will be printed out?

8. Modify the code for `quad_eqn.m` so that it will work if `a`, `b`, and `c` are arrays. (In this case, `x1` and `x2` will be arrays, by the way.) The `a`, `b`, and `c` arrays must all be the same size. If this is not the case, print an error message and return to the calling program.

9. What will happen if the user enters the arguments to `quad_eqn` in reverse order: `c`, `b`, `a`? Consider the specific case of `a = 1, b = 2, c = -15`.

10. If the name of the function file is changed to `quadratic_eqn.m`, what changes should be made to the code for the function? (List all changes that *must* be made, and all changes that *should* be made, and clearly distinguish between the two.)

REF: [1] The Mathworks, Inc, *Matlab® Primer,* Natick, MA: The Mathworks, Inc, 2012.