Week 4 Tutorial 2

The purpose of this tutorial is to teach you how to import workspace variables from a binary .mat file. We will also cover the use of the `rand()` function and create some fake resistor values and store them in our Excel file used previously.

```
% Always clear workspace variables before a new tutorial or program.
clear
clc
```

Edit the code below and update the variable named **`name`** with your name for this tutorial in the code below.

```
name="";
fprintf("Output for Tutorial_04_2 run by %s.\n", name)
```

Input

Importing From .mat

Refer to the lecture slides, book, or the help doc if you don't remember. Load the workspace variables that were stored when running Tutorial_04_1.

```
filename='Tutorial_04_1.mat'

% Load stored workspace variables


% Display some info to the user
fprintf('The following variables have been retrieved from %s\n\n', filename)
whos

fprintf('Variable values:\n\n')
fprintf('resistorsSize = %g\n', resistorsSize)
fprintf('resistorsCount = %i\n', resistorsCount)
fprintf('resistorsSum = %g\n', resistorsSum)
fprintf('resistorsMean = %g\n', resistorsMean)
fprintf('resistorsStd = %g\n', resistorsStd)
```

Create a random matrix

Let's say we want to create a matrix of equal size as the matrix previously loaded from the Excel file. But in this new matrix, we're going to fill it with random values and store that data on a separate sheet in the Excel file.

Furthermore, let's say we want those values to be with roughly the same mean and std dev as the previous set of resistor values. We discussed the `rand()` function briefly, there is also the `randn()` function which creates a matrix of random values with a normal distribution, a mean of 0 and a std dev of 1. Because this is a normal distribution (bell curve), the random values will have a higher chance of being closer to the mean than farther.

The formula for adjusting the distribution of `randn()` using a specified mean and std dev is *mean + stdev*randn(m,n).*

With that knowledge, create a new **resistors** variable containing random values with roughly the same mean and std dev as Tutorial_04_1's resistor values.

```
resistors=resistorsMean + resistorsStd*randn(resistorsSize)
```

Get Meta Data

Just as was done in Tutorial_04_1, compute the following statistical values on this newly generated matrix of random values.

```
newResistorsSize=
newResistorsCount=
newResistorsSum=
newResistorsMean=
newResistorsStd=
```

Output

To the command window

Let's output our data to the command window

```
fprintf('The new batch of resistors have the following attributes\n\n')
fprintf('The number of rows = %i\n', newResistorsSize(1))
fprintf('The number of columns = %i\n', newResistorsSize(2))
fprintf('The number of resistor values = %i\n', newResistorsCount)
fprintf('The sum of all resistor values = %g ohms\n', newResistorsSum)
fprintf('The mean of all resistor values = %g ohms\n', newResistorsMean)
fprintf('The std dev of all resistor values = %g ohms\n', newResistorsStd)
```

To the Excel spreadsheet

Now store the data in the original Excel spreadsheet using `writematrix`

Let's put the values in as follows.

- Store the array of resistors in cells A4:D28 of sheet 'NewResistors'
- resistorsCount in cell H4 of sheet 'NewResistors'
- resistorsSum in cell H5 of sheet 'NewResistors'
- resistorsMean in cell H6 of sheet 'NewResistors'
- resistorsStd in cell H7 of sheet 'NewResistors'

I'll do the first one, it's in the format `writematrix(value, filename, ... sheet and range params)`. We should use variables so we don't have to repeat things like the filename, and sheet name

```
filename='Tutorial_04_1_Data.xlsx';
sheetName='newResistors';



% Notify the user that data is written to the Excel file
```

```
fprintf('These statistical values have also been written to Tutorial_04_1_Data.xlsx \n')
```

Pro Tip: You can see the benefit of storing the filename and sheetname in a variable, you can copy and paste your `writematrix` commands from Tutorial_04_2 and just change the names of the variables, the filename and sheetname are stored in variables so you don't need to replace those values 8 different times. So, use variables

Compare the Results

After you've run both scripts, compare Excel sheets 'OldResistors' and 'NewResistors' and see how closely they compare even though the 'NewResistors' were randomly generated values.

Example output

If you were to run your tutorial (enter **Tutorial_04_02** into the command window) your output should appear as follows. Note that your values will be slightly different but should be relatively close to the ones shown here.

```
Output for Tutorial_04_2 run by Geoff Berl.

filename =

    'Tutorial_04_1.mat'

The following variables have been retrieved from Tutorial_04_1.mat

  Name                Size              Bytes  Class     Attributes

  filename            1x17                 34  char
  name                1x1                 148  string
  resistorsCount      1x1                   8  double
  resistorsMean       1x1                   8  double
  resistorsSize       1x2                  16  double
  resistorsStd        1x1                   8  double
  resistorsSum        1x1                   8  double

Variable values:

resistorsSize = 25
resistorsSize = 4
resistorsCount = 100
resistorsSum = 7595.01
resistorsMean = 75.9501
resistorsStd = 16.6077
```

```
resistors =

   84.8795   93.1340   61.6068   52.6616
  106.4067   88.0220   77.2349   52.3277
   38.4359   70.9106   55.7864   84.0579
   90.2688   80.8306   57.4574   73.0043
   81.2441   62.8751   75.8363   72.6941
   54.2324   90.7043  101.4036   99.5216
   68.7491   56.8999   63.1677   80.7926
   81.6403   58.1986   82.1178   79.2353
  135.3790   62.5062   72.2037  102.3181
  121.9441   27.0523   94.5068   62.5898
   53.5316   99.8383   57.8633   87.5194
  126.3532   81.3508   76.4908   89.8190
   87.9974   63.4125   85.1263   71.9026
   74.9029   98.7076   94.2287   79.5319
   87.8203   47.5258  101.5959   56.5881
   72.5461   74.2521   77.3772   56.8853
   73.8883   71.9402   51.1782   77.6918
  100.6905   81.2514   63.6222   87.9451
   99.3509   81.1460   58.3197  118.8891
   99.4864   61.5864  114.9858   64.8746
   87.1021   75.4510   65.7264   79.0612
   55.8965   73.2118   88.3739   74.5801
   87.8618   86.3749   72.7545   43.8470
  103.0245   94.1067   90.7079   68.6599
   84.0695   94.3726   63.2477   46.1446


newResistorsSize =

    25     4


newResistorsCount =

   100


newResistorsSum =

   7.7994e+03


newResistorsMean =

   77.9943
```

```
newResistorsMean =

   77.9943


newResistorsStd =

   19.3048

The new batch of resistors have the following attributes

The number of rows = 25
The number of columns = 4
The number of resistor values = 100
The sum of all resistor values = 7799.43 ohms
The mean of all resistor values = 77.9943 ohms
The std dev of all resistor values = 19.3048 ohms
These statistical values have also been written to Tutorial_04_1_Data.xlsx
```

Additional Notes:

-