Q1. Plot the probability density function of a chi-squared distribution. As some of you know, the chi-squared distribution is widely used in categorical data analysis. The equation for the probability density function of the chi-squared distribution is

$$f(x) = \frac{1}{2^{k/2} \Gamma(k/2)} x^{k/2 - 1} e^{-x/2}$$

$\theta$ is an important mathematical constant that shows up all over the place in mathematics and statistics. Like $\pi$, it's just a number. It's equal to 2.71828182846... In Matlab, there is a function `exp(x)` which computes $\theta$ raised to some power $x$ (check the documentation on `exp` in Matlab).

The gamma function $\Gamma(x)$ is also ubiquitous in mathematics and statistics. In Matlab, there is a function `gamma(x)` you can use. Compute $\Gamma(x)$ where $x$ is an integer between 1 and 10 and see if you can figure out what $\Gamma(x)$ is computing when you give it an integer. $\Gamma(x)$ generalizes to all real numbers, not just integers.

In the equation for the chi-squared, $x$ if the value you are determining the likelihood of, $f(x)$ is the computed likelihood, and $k$ is the degrees of freedom. Unlike a distribution like the normal, for the chi-square $x$ is constrained to be greater than or equal to zero (the chi-squared distribution is undefined for negative values of $x$).

Create three plots of the chi-squared distribution for three values of the degrees of freedom, $k=2, 4, and 9$. You can adapt the example we discussed in class for plotting the normal distribution. Make sure your plots reasonably show the complete distribution (obviously, you can’t show the complete distribution in its entirety since it is defined in the range between 0 and infinity, but you can show a reasonable range to see its full shape).

Q2. In Matlab code, convince yourself (and me) that the inverse of a matrix is on the same as the matrix of the inverse $(1/x)$ of all of its elements. First, I want you to show me that they are not equal to one another. Second, I also want you to show me that the former (the inverse of a matrix) and not the latter (the matrix of the inverse of its elements) satisfies the key property for what it means to be an inverse of a matrix.

Note that for some matrices it is impossible to calculate an inverse. Matlab will often signal this by saying that the matrix is “singular” (sometimes called “degenerate” as well). If you find that your matrix is singular (noninvertible) try another matrix until you find one that can be inverted.
Q3. I want you to create a data structure that holds the first name and last name of the subjects in an experiment:

Mary Smith  
Robert Jones  
Frank Tinny  
Larry Jacobson

Demonstrate how you would reference only the first name of all the participants, then demonstrate how you would reference the full name of one of the participants. Do so without hard-coding the solution.

Next use an fprintf statement to print the full name of the 4th subject.

Q4. Learn about structures (struct arrays) in Matlab. I have notes in today's class slides that you can look at. In the slides I also have links to some online sources. You can also read about structures in the Attaway textbook.

Imagine that I am currently holding data for my experiment in a cell array as follows:

```matlab
data{1,1} = 'Frank Jones'; % name
data{1,2} = 23; % age
data{1,3} = [24 26 21 22]; % BDI score each month
data{1,4} = [178 175 181 182]; % weight each month
data{2,1} = 'Bob Mills'; % name
data{2,2} = 26; % age
data{2,3} = [28 31 30 32]; % BDI score each month
data{2,4} = [154 152 156 153]; % weight each month
data{3,1} = 'Jim Russo'; % name
data{3,2} = 32; % age
data{3,3} = [21 22 19 21]; % BDI score each month
data{3,4} = [198 194 195 196]; % weight each month
```

a) Recreate this data using a structure (struct array) instead.

b) Write a single line of Matlab code that calculates the average BDI score for each participant in the cell arrays. It should be general enough so that the same piece of code will work even if `data` contained more than 3 subjects.

c) Write a single line of Matlab code that calculates the average BDI score for each participant in the struct arrays. It should be general enough so that the same piece of code will work even if `data` contained more than 3 subjects.

Unexcused late assignments will be penalized 10% for every 24 hours late, starting from the time class ends, for a maximum of two days, after which they will earn a 0.