What did we talk about last class?
Do you have any questions about the reading?
■ You make these with {} or the cell function. They don't have to be rectangular and they can hold any data. Each cell can hold a different type of data.
■ We can index into cell arrays with {} as well. If you index a cell array with () you get a cell containing data. If you use {} you get the data that was in the cell.
■ Multiple elements can't normally be pulled out with content addressing unless you put them into multiple variables with a comma separated list.
■ Anything that produces multiple cells will be turned into a comma separated list. This can be difficult to get your brain around.
You can use the dot notation to put fields into a variable to make a structure. Unlike normal imperative languages, the format of the structure isn't predefined.

Matlab deals with arrays of structures just like numeric arrays.

The struct function can build arrays of structures from existing cell arrays.

You can pull out all the values of certain fields with 'dynamic addressing'.
Like most other languages, Matlab does give you the ability to use strings, though that isn't a real strength.

A Matlab string is simply a row array of characters.

A downside of this is that an array with multiple strings must have all the strings be the same length. The char function can help with that.

You can also convert from numbers to strings and back with str2num and num2str.

Matlab also has fprintf and sprintf functions that work much like the C functions.

Similarly, sscanf will pull numbers out of strings.

eval and evalc let you process a string like it were a Matlab function.
We are now less than a week from assignment #3. You should probably look it over so that you can ask questions next class.