ICE for Week 8

Problem 1

Write a script that will prompt the user to enter a word, and then print the first character in the word. For example, the output might look like this:

```
>> Enter a word: howdy
The word howdy starts with the letter ‘h’
```

Problem 2

Write a function that will receive a name and department as separate strings and will create and return a code consisting of the first two letters of the name and the last two letters of the department. The code should be uppercase letters. For example,

```
>> namedept('Robert','Mechanical')
ans =
ROAL
```

Problem 3

Write a script that will create x and y vectors. Then, it will ask the user for a color ('red', 'blue', or 'green') and for a plot style ('o', '*'). It will then create a string *pstr* that contains the color and plot style, so that the call to the **plot** function would be **plot(x,y,pstr)**.
For example, if the user enters 'blue' and '*', the variable pstr would contain 'b*'.

Problem 4

Create the following two variables:

```
>> var1 = 123;
>> var2 = '123';
```

Then, add 1 to each of the variables. What is the difference?

Problem 5

Words in a sentence variable (just a string variable) called *mysent* are separated by /'s instead of blank spaces. For example, mysent might have this value:

```
'This/is/not/quite/right'
```

Write a function *slashtoblank* that will receive a string in this form and will return a string in which the words are separated by blank spaces. This should be general and work regardless of the value of the argument. No loops are allowed in this function; the built-in string function(s) must be used.

```
>> mysent = 'This/is/not/quite/right';
>> newsent = slashtoblank(mysent)
newsent =
This is not quite right
```

Problem 6

Using the functions **char** and **double**, you can shift words. For example, you can convert from lowercase to uppercase by subtracting 32 from the character codes:

```
>> orig = 'ape';
>> new = char(double(orig)-32)
new =
APE
>> char(double(new)+32)
ans =
ape
```

We've encrypted a string by altering the character codes. Figure out the original string. Try adding and subtracting different values (do this in a loop) until you decipher it:

```
Jmkyvih$mx$syx$}ixC
```

Problem 7

Two variables store strings that consist of a letter of the alphabet, a blank space, and a number (in the form ‘r 14.3’). Write a script that would initialize two such variables. Then, use string manipulating functions to extract the numbers from the strings and add them together.

Problem 8

Write a script that will first initialize a string variable that will store x and y coordinates of a point in the form ‘x 3.1 y 6.4’. Then, use string manipulating functions to extract the coordinates and plot them.

Modify the script in the previous example to be more general: the string could store the coordinates in any order; for example, it could store ‘y 6.4 x 3.1’.

Problem 9

Create a cell array that stores phrases, for example,

```
exclaimcell = {'Bravo', 'Fantastic job'};
```

Pick a random phrase to print.

Problem 10

Create three cell array variables that store people's names, verbs, and nouns. For example,

```
names = {'Harry', 'Xavier', 'Sue'};
verbs = {'loves', 'eats'};
nouns = {'baseballs', 'rocks', 'sushi'};
```

Write a script that will initialize these cell arrays, and then print sentences using one random element from each cell array, for example, 'Xavier eats sushi'.

Problem 11

Create a 2 × 2 cell array by using the **cell** function to preallocate and then put values in the individual elements. Then, insert a row in the middle so that the cell array is now 3 × 2. Hint: Extend the cell array by adding another row and copying row 2 to row 3, and then modify row 2.

Problem 12

Write the code in MATLAB that would create the following data structure, and put the following values into the variable:

	experiments					
	num	code	weights		height	
			1	2	feet	inches
1	33	'x'	200.34	202.45	5	6
2	11	't'	111.45	111.11	7	2

The variable is called *experiments*, which is a vector of structs. Each struct has four fields: *num*, *code*, *weights*, and *height*. The field *num* is an integer, *code* is a character, *weights* is a vector with two values (both of which are double values), and *height* is a struct with fields *feet* and *inches* (both of which are integers). Write the statements that would accomplish this, so that typing the following expressions in MATLAB would give the results shown:

```
>> experiments
experiments =
1x2 struct array with fields:
  num
  code
  weights
  height

>> experiments(2)

  ans =
        num: 11
       code: 't'
    weights: [111.4500 111.1100]
     height: [1x1 struct]
  >> experiments(1).height
  ans =
       feet: 5
     inches: 6
```

Problem 13

A script stores information on potential subjects for an experiment in a vector of structures called *subjects*. The following show an example of what the contents might be:

```
>> subjects
subjects =
1x3 struct array with fields:

  name
  sub_id
  height
  weight

>> subjects(1)
ans =
name: 'Joey'
  sub_id: 111
  height: 6.7000
  weight: 222.2000
```

For this particular experiment, the only subjects who are eligible are those whose height or weight is lower than the average height or weight of all subjects. The script will print the names of those who are eligible. Create a vector with sample data in a script, and then write the code to accomplish this. Don't assume that the length of the vector is known; the code should be general.