ENG EK 125 - Worksheet 8B

Name: Ande Chen Section: C1

1) In chemistry, the pH of an aqueous solution is a measure of its acidity. A solution with a pH of 7 is said to be *neutral*, a solution with a pH greater than 7 is *basic*, and a solution with a pH less than 7 is *acidic*. Create a vector of structures with various solutions and their pH values. Write a function that will determine acidity. Add another field to every structure for this.

```
solutions(1) = struct('name', 'water', 'ph', 7);
solutions(2) = struct('name', 'bleach', 'ph', 13);
solutions(3) = struct('name', 'coffee', 'ph', 5);

for i = 1:length(solutions)
    solutions(i).acidity = detacidity(solutions(i).ph);
end

disp(solutions(1));
disp(solutions(2));
disp(solutions(3));
```

```
function out = detacidity(in)
if in == 7
    out = 'neutral';
elseif in > 7
    out = 'basic';
else
    out = 'acidic';
end
```

```
>> WS8B_1
       name: 'water'
         ph: 7
    acidity: 'neutral'

       name: 'bleach'
         ph: 13
    acidity: 'basic'

       name: 'coffee'
         ph: 5
    acidity: 'acidic'

>>
```

2) A team of engineers is designing a bridge to span the Podunk River. As part of the design process, the local flooding data must be analyzed. The following information on each storm that has been recorded in the last 40 years is stored in a file: a code for the location of the source of the data, the amount of rainfall (in inches), and the duration of the storm (in hours), in that order. For example, the file might look like this:

```
321    2.4    1.5
111    3.3    12.1
           etc.
```

Create a data file. Write the first part of the program: design a data structure to store the storm data from the file, and also the intensity of each storm. The intensity is the rainfall amount divided by the duration. Write a function to read the data from the file (use **load**), copy from the matrix into a vector of structs, and then calculate the intensities. Write another function to print all of the information in a neatly organized table.
Add a function to the program to calculate the average intensity of the storms. Add a function to the program to print all of the information given on the most intense storm. Use a subfunction for this function that will return the index of the most intense storm.

```
function outstruct = readData()
    load floodData.dat
    [r c] = size(floodData);

    for i = 1:r
        code = floodData(i, 1);
        rainfall = floodData(i, 2);      %inches
        duration = floodData(i, 3);      %hours
        intensity = rainfall/duration;  %inches per hour
        outstruct(i) = struct('code', code, 'rainfall', rainfall, ...
            'duration', duration, 'intensity', intensity);
    end
end
```

```
function out = maketable(instruct)
    out = struct2table(instruct);
end
```

```
function avg = avgintensity(instruct)
    total = 0;
    for i = 1:length(instruct)
        total = total + instruct(i).intensity;
    end
    avg = total/length(instruct);
end
```

```
function mostintense(instruct)
    index = findindex(instruct);
    info = instruct(index);
    disp(info);
end
```

```
function index = findindex(instruct)
    index = 0;
        for i = 1:length(instruct)
            if instruct(i).intensity > index
                index = i;
            end
        end
end
```

```
floodData = readData();
floodTable = maketable(floodData);
floodTable
avg = avgintensity(floodData);
avg
mostintense(floodData)
```

```
>> WS8B_2

floodTable =

  3×4 table

    code    rainfall    duration    intensity
    ____    ________    ________    _________

    321        2.4          1.5          1.6
    111        3.3         12.1      0.27273
    324        7.6         2.34       3.2479


avg =

    1.7069

        code: 324
    rainfall: 7.6000
    duration: 2.3400
   intensity: 3.2479

>>
```

3) Investigate the built-in function **cell2struct** that converts a cell array into a vector of structs.

```
apples = {'red', 0.25, 0.20; 'green', 0.23, 0.20};
f = {'color', 'weight', 'price'};
astruct = cell2struct(apples, b, 2);
disp(astruct);
```

```
>> WS8B_3
  2×1 struct array with fields:

    color
    weight
    price
```

4) Create a vector of structures *experiments* that stores information on subjects used in an experiment. Each struct has four fields: *num*, *name*, *weights*, and *height*. The field *num* is an integer, *name* is a string, *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). The following is an example of what the format might look like.

experiments

	num	name	weights		height	
			1	2	feet	inches
1	33	Joe	200.34	202.45	5	6
2	11	Sally	111.45	111.11	7	2

Write a function *printhts* that will receive a vector in this format and will print the name and height of each subject in inches (1 foot = 12 inches). This function calls another function *howhigh* that receives a height struct and returns the total height in inches. This function could also be called separately.

```
1 -    experiments(1) = struct('num', 1, 'name', 'Alvin', 'weights',...
2          [200.34 202.45], 'height', struct('feet', 5, 'inches', 6));
3 -    experiments(2) = struct('num', 23, 'name', 'Sally', 'weights',...
4          [111.54 111.11], 'height', struct('feet', 7, 'inches', 2));
5
6 -    [a b] = printhts(experiments);
```

```
1      function [name inches] = printhts(invec)
2 -        for i = 1:length(invec)
3 -            name = invec(i).name;
4 -            inches = howhigh(invec);
5 -            fprintf('%s %d\n', name, inches);
6 -        end
7 -    end
```

```
1      function height = howhigh(invec)
2 -        for i = 1:length(invec)
3 -            height = invec(i).height.feet * 12 + invec(i).height.inches;
4 -        end
5 -    end
```

```
>> WS8B_4
Alvin 86
Sally 86
>>
```