HW 03 - Course Catalog Analytics and Visualization

In this homework you will work with course catalog data from a university. As long as the listing contains several hundred courses, you are free to select any university you like,.

Question 1. - Download the Data

Find the html pages holding the course descriptions. Use `curl` to download the course catalog data. You can start with a subset of the data to simplify your work. However, make sure you test on the complete dataset and that your submission contains the full dataset.

```
abel@osx:~$ curl http://student.mit.edu/catalog/m1a.html \
> http://student.mit.edu/catalog/m1b.html \
> http://student.mit.edu/catalog/m1c.html > catalog.html
```

Figure 1 - Sample curl command for MIT course catalog

1.1. - Submit your curl script. Name your file `curl.txt`
1.2. - Submit your raw data. Name your file `raw.html`

Question 2. - Remove Whitespace

Your will read the data you downloaded as a string. In order to do that you need to remove line breaks and whitespaces. You can use the NPM package `html-minifier`.

```
abel@osx:~$ html-minifier catalog.html --collapse-whitespace
 --minify-js --minify-css -o no_whitespace.html
```

Figure 2 - HTML Minifier

2.1. - Submit your html-minifier script. Name your file `whitespace_remove.txt`
2.2. - Submit your file without whitespace. Name your file `no_whitespace.html`

Question 3. - Additional Data Cleaning

You will most likely need to do additional data cleaning step(s).

3.1. - Submit the list of additional tools/packages your used.
3.2. - Submit a screen shot of every command line action you performed.
3.2. - Submit a file containing the additional command line scripts you performed. Name the file `additional_cleaning.txt`
3.3. - Submit your final file, the ouput after all your data cleaning steps. Name your file `scrubbed.html`

Question 4. - Load data into a JavaScript string

Create a file called `data.js`. Assign your scrubbed html to a JavaScript variable. Name the variable `data`.

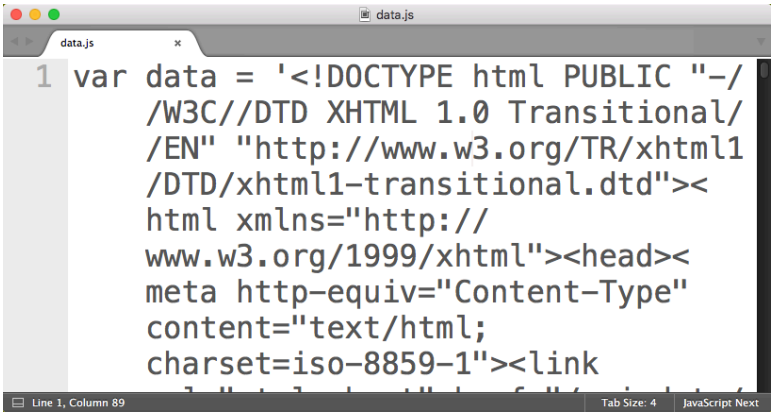

Figure 4 - mitcourses JavaScript variable

4.1. - Submit your file. Name your file `data.js`

Question 5. - Load data into HTML document, get courses

Load your scrubbed HTML into the DOM. Use the DOM structure to get all the courses. Match the function signatures provided below.

```
// pass in html to add to page
// return element containing new HTML
function addHtmlToPage(htmlString){

     // --------------------------
     //        YOUR CODE
     // --------------------------

}



// pass in html element containing data
// return nodelist of courses
function getCourseNodeList(tag){

     // --------------------------
     //        YOUR CODE
     // --------------------------

}


// pass in nodelist of courses
// return array of courses
function nodeListToArray(nodeList){

     // --------------------------
     //        YOUR CODE
     // --------------------------

}
```

5.1. - Add your functions to a file called `courses.html`

Question 6. - Get titles, clean titles, make word arrays

Match the function signature provided below.

```
// pass in array of courses
// return course titles
function getTitles(list){

     // --------------------------
     //        YOUR CODE
     // --------------------------

}
```

The title cleaning function demonstrated in class was very simplistic. Improve it by filtering out words like "and", "the", "a", etc. What other improvements can you make?

```
// pass in course titles
// return words
// filter out punctuation/numbers, make words array
function scrubTitles(titles){

     // --------------------------
     //        YOUR CODE
     // --------------------------

}


// pass in words array
// return flat words array
// flatten the 2D words array
function flattenArray(words){

     // --------------------------
     //        YOUR CODE
     // --------------------------

}
```

6.1. - Add your functions to the file you created in question 5, `courses.html`

Question 7. - Calculate word scores

Match the function signature provided below.

```
// pass in the flat words array
// return word scores
// count the word frequency
function scores(wordsFlat){

     // --------------------------
     //        YOUR CODE
     // --------------------------

}
```

7.1. - Add your function to the file you created in question 5, `courses.html`

Question 8. - (OPTIONAL) Improve Graphing Logic

The graph logic is based on the scores you calculated for words. You can find the code in the graphing JavaScript file.

```
for (var word in scores) {
  nodes.push({radius: radius(scores[word]),
    color: color(word.length), word: word,
    score: scores[word]});
}
```

Can you improve the graph?

8.1. - Submit your changes in a new file. Name your file `new_graph.js`