In place of a signature, please include a commented statement in your code affirming your recognition of the Academic Honesty Policy for the Exam 1. You will replace the blank with your name and UHID as an acknowledgement in your starting file as acknowledgement of this policy.

```
% I,<INSERT FULL NAME> (<INSERT UHID>) acknowledge that the Exam 1 for ENGI 1331,
% is to be completed by myself with no collaboration with anyone.
% I have read the ENGI 1331 Position on Academic Honesty and agree
% to abide by its provisions while taking this exam.
% I acknowledge that my submission will be run through a similarity code.
% Any student with unacceptable levels of commonality with peers or
% other sources will be brought up for an academic honesty violation.')
```

## INSTRUCTIONS:

This virtual computer exam will be given on Saturday, October 3, 2020.

- You must be logged in with video on to your virtual classroom during the entire exam. If you lose connection during the exam, re-enter the virtual classroom and proceed with the exam.
- Computer with internet access and video is required.

## General Rules

- You should **suppress** all output to the Command Window except if specific formatted output is requested.
- You must use any variable names specified. **If a variable name is not specified, you may create your own name for the variable.**
- **Do Not Hard Code for a specific case – your code must be flexible based on the instructions provided.**

## Exam Timeline

***Programming in MATLAB (only portion):*** This exam should take you approximately 1 hour to complete and 15 minutes for planning and 5 minutes to upload your code. Total time is 1 hours and 20 minutes.

## Finishing the exam

You will **have 1 hours to complete all tasks after the 15 minute downloading/planning period and 5 minutes to upload code**. After time is called, you will be expected to close out of MATLAB and zip your exam folder.

- During the exam, **NO COLLABORATION**, or you will be dismissed with a **ZERO**.

After the exam is completed and the file collected, any procedure (such as opening the file) which alters the date / time stamp of the file will void any allowances for mis-saved files – in other words, **after the exam is over DO NOT OPEN the file again!**

## Saving Exam File

You will be expected to have one script file associated with your exam submission and any starting files or exported files. Save your script file in the exam folder on your desktop as **Exam1_cougarnet.m**. All other functions requested should be the named as instructed with correct cougarnet. You must submit a .ZIP file named **Exam1_cougarnet.zip** that contains the script necessary to run your code.

# MATLAB Programming (100 points) – Approx. 60 minutes + 15 minutes for planning

## Background

***Background:** A civil engineer measures how much a bridge sags (falls vertically). This is known as deflection. Deflection was measured at identically sized concrete bridges in various Gulf Coast cities. Several measurements are taken at locations on the bridge. The starting position (at one end) is 0 [m] and the final position (at the other end) is 1500 [m].*

***The table below represents a sample data that is NOT the actual data.***

**City**

| Lake Charles | Mobile | Beaumont |
|---|---|---|

**Location [m]**

| |
|---|
| 0 |
| 300 |
| 800 |
| 1500 |

**Deflection [mm]**

| | | |
|---|---|---|
| 0.1 | 0.4 | 0.2 |
| 0.9 | 2.2 | 1.3 |
| 1 | 1.8 | 2.3 |
| 1.3 | 0.9 | 1.1 |

NOTE: The number of rows in **Deflection** is equal to the number of locations provided in **Location** and the number of columns in **Deflection** is equal to the number of cities given in **City**. See the tables above for a visual representation. All data provided is within **Exam1_data.mat**.

**NOTE (avoid hardcoding):** Your functions and script should produce different results if the data for **Location, City,** or **Deflection** changes.

## Tasks:

## Task 1 (20 min) – 30 pts

### Function (11 pts)

Create a function named **Stats_cougarnet** (replace with your courgarnet username) that, given a matrix, first determines the average values for each row and appends it as a new column, and then determines the maximum values for each column and appends is as a new row to the bottom of the matrix (see the sample output for Task 1 - Exported File).

**Function Inputs:**

1. Matrix (any size)

**Function Outputs:**

1. Updated Matrix (new last row and last column)

The function header should be formatted similarly to the following:

```
function [out1,out2,out3,out4] = Stats_cougarnet(in1)
```

Remember you are free to use whatever variable names you want, but they must be listed in the same order as given in the input/output lists provided above.

### Main Script (19 pts)

Load in **Exam1_data.mat**. Prompt the user to enter a new city and a new vector of deflection values (length of vector equals number of rows in **Deflection**). Update the **City** string vector with the new city appended to the end, and the **Deflection** matrix with the new vector of deflection values on the last column. Use this updated **Deflection** matrix to find the element with the highest overall deflection, its associated location and city and output the results to the command window as seen in the sample output.

Using your function **Stats_cougarnet**, update the deflection matrix. Save the updated matrix to **Exam1_Task1.csv**.

## Task 2 (30 min) – 40 pts

### Function (18 pts)

Create a function named **Critical_cougarnet** (replace with your courgarnet username) that, given a deflection vector and a critical value determines the number of values above that critical value, the average of all values above that critical value, a row vector of the two lowest deflection values and a row vector of the two locations [m] from the location string array associated with the lowest deflection values.

**Function Inputs:**

1. A location string array
2. A deflection vector
3. A critical value (scalar)

**Function Outputs:**

1. Number of values above the critical value
2. Average of all values above the critical value
3. Row vector containing the two lowest deflection values
4. Row vector containing the two locations [m] from the location string array of the lowest deflection values

The function header should be formatted similarly to the following:

```
function [out1,out2,out3out4] = Critical_cougarnet(in1,in2,in3)
```

Remember you are free to use whatever variable names you want, but they must be listed in the same order as given in the input/output lists provided above.

### Main Script (22 pts)

Prompt the user to enter a critical value for the bridges. The critical deflection is the sag at which the bridge could fail (collapse). Any deflection beyond the critical deflection indicates a potential failure point.

Prompt the user to select a City.

For the selected City (column vector):

1. Use your function, **Critical_cougarnet**, to determine the number of locations above the critical deflection, the average of deflection values above the critical value, the two lowest deflections values and their associated indices. Output the number of locations above the critical deflection and the average deflection of those values to the command window (see sample output). Save the results of two lowest deflections and the associated bridge locations as a .mat named **Exam1_Task2.mat**. See sample output for task 2.

2. Create a plot for the selected city with the location on the x-axis and the deflection on the y-axis with the deflections above the critical deflection in red and deflections below or equal the critical deflection in black. The plot formatting should include
   - X and y axis labels
   - Grid lines
   - Plot shown as single symbols (not a line)
   - All deflections above the critical deflection in red symbols.
   - All deflections below the critical deflection in black symbols.

## Sample Output:

Sample output for given data:

- Expected exported files are given below. Note the different file types of the exported files.

**Sample Output to Command Window**

```
Command Window
Enter a new city: Houston
Enter a new vector of deflections: [0.1;0.5;1.0;0.4;1.5]
The overall maximum deflection of 2.38 occurs in the city of Pensacola at 1500 [m] on the bridge.
Enter a critical deflection value:  0.6
fx There are 2 deflections above the critical with an average of 0.81>>
```

**Sample Output For Task 1 (Exported File)**

Exam1_Task1.csv

| | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| | Exam1Task1 | | | | | | | |
| | VarName1 | VarName2 | VarName3 | VarName4 | VarName5 | VarName6 | VarName7 | VarName8 |
| | Number | Number | Number | Number | Number | Number | Number | Number |
| 1 | 1.875 | 1.5 | 0.5 | 2.25 | 1.625 | 1.75 | 0.1 | 1.3714 |
| 2 | 0.25 | 2.25 | 1 | 0.25 | 0.25 | 1.5 | 0.5 | 0.85714 |
| 3 | 2.25 | 0.875 | 0.25 | 0.625 | 0.25 | 1 | 1 | 0.89286 |
| 4 | 0.125 | 2.125 | 0.375 | 1.375 | 2 | 2 | 0.4 | 1.2 |
| 5 | 2.125 | 0.375 | 0.625 | 2 | 0.75 | 2.375 | 1.5 | 1.3929 |
| 6 | 2.25 | 2.25 | 1 | 2.25 | 2 | 2.375 | 1.5 | 1.3929 |

**Sample Output For Task 2 (Exported File and Plot) for Selection: Lake Charles**

**Exam1_Task3.mat**

Workspace

| Name | Value |
|---|---|
| defLOW | [0.2500,0.3750] |
| locLOW | [1085,1028] |