## General Instructions

**Due Date:** Sunday, October 2nd, 2022 by 11:59pm (submit via Zybooks)

**Assignment Summary Instructions:**
This assignment has two problems, summarized below. You will use MATLAB as a tool to solve the problem for the given test cases, ensuring that your code is flexible for any additional test cases that might be used to evaluate it.

- Beam Deflection with Piecewise Functions (Application to Civil and Mechanical Engineering)
- Resistor Decoder (Application to Electrical and Computer Engineering)

**zyBooks Submission Instructions:**
After completing this assignment in MATLAB, to receive credit, you must submit your code in Zybooks. The following components must be submitted under the specified chapter of the course Zybooks before the deadline to receive credit.

- Chapter 32.1 MA3 Problem 1: FindDeflection Function
- Chapter 32.2 MA3 Problem 1: Main Script
- Chapter 32.3 MA3 Problem 2: Resist2Color Function
- Chapter 32.4 MA3 Problem 2: Color2Resist Function
- Chapter 32.5 MA3 Problem 2: Main Script

To submit your script, copy and paste your code into the submission window, making sure to remove any housekeeping commands. You may submit to Zybooks as many times as you want before the deadline, without any penalty. The highest score attained before the deadline will be graded. All components are due before the due date. No credit will be given if it is not submitted through the Zybooks platform before the deadline. Credit for each component will be awarded based upon the percentage of successfully completed assessments.

**Explanation of P-Code:**
Under the Additional Resources folder accompanying this prompt, you will find files named **FindDeflection_Solution.p**, **Resist2Color_Solution.p** and **Color2Resist_Solution.p** and. These are working solutions for the functions required to complete each problem. These files are unopenable and the contents can't be read in a text editor, but they can be called like a typical .m function file in MATLAB. If you get stuck and are unable to successfully develop your own functions for each problem, you are encouraged to instead copy the necessary .p file to your working folder and rename it **FindDeflection.p** (Problem 1) or **Resist2Color.p** and **Color2Resist.p**, (Problem 2). Now, when your main script calls the appropriate function, it will automatically call the .p file and you will now have an opportunity to successfully finish developing your main script.

**Proficiency Time:** Times are included with the Background and Task sections. These times are the estimated amount of time it should take you to **redo** an assignment once you are fully proficient in material that it covers. To practice, reread the background in the given Comprehension Time and attempt to complete the problem in the given Proficiency Time.

## Academic Honesty Reminder

**The work you submit for this assignment should be your work alone.** You are encouraged to support one another through collaboration in brainstorming approaches to the problem and troubleshooting. In this capacity, you are permitted to view other students' solutions, however, copying of another student's work is strongly discouraged.

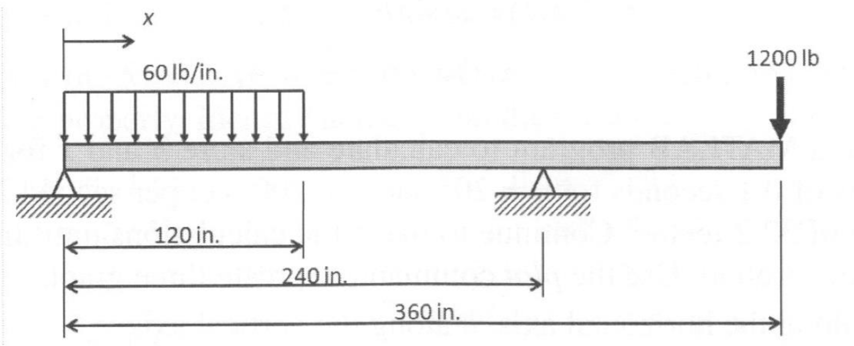
**This assignment will be checked for similarity using a MATLAB code.** The similarity code will check each submission for likeness between other student submissions, past student submissions, the solution manual, and online resources and postings. If your submission is flagged for an unreasonably high level of similarity, it will be reviewed by the ENGI 1331 faculty, and action will be taken by faculty if deemed appropriate.

**NOTE**: Since this is an automated system for all sections, if any of your work is not your own, you will be caught. Changing variable names, adding comments, or spacing will not trick the similarity algorithm.

## Problem #1: Beam Deflection with Piecewise Functions

### Background: Comprehension Time: 10 – 15 min

For the two-support beam shown below, you need to calculate and plot the beam's deflection under the given loads. Deflection is the movement of the beam downward due to external forces applied. This deflection changes based on the physical properties of the beam system. For this beam system, the deflection can be shown as a piecewise function (Equation 1). Another engineer has already derived the equation you need below.




The deflection equation *v(x)* was derived using the free-body-diagram above. This equation can be shown as the piecewise function given in Equation 1. This piecewise function is true for any point *x* inches along the beam, starting from the left support.

$$v(x) = \frac{1}{3.19*10^9}\begin{cases} 800x^3 - 13.68*10^6x - 2.5x^4, & 0 < x \le 120 \\ 800x^3 - 13.68*10^6x - 2.5x^4 + 2.5(x-120)^4, & 120 < x \le 240 \\ 800x^3 - 13.68*10^6x - 2.5x^4 + 2.5(x-120)^4 + 600(x-240)^3, & 240 < x \le 360 \end{cases} \quad \text{Eq. 1}$$

In this form, each portion of the beam follows a slightly different equation, based on where forces are placed on the beam. At each critical point on the beam (where the forces acting on the beam change) a new term is added to the overall equation.

### Tasks: Proficiency Time: 20 – 30 min

<u>TASK 1: (12 – 15 min)</u>

Develop a user-defined function named **FindDeflection** which calculates the deflection of a beam at specific location(s). You will need to apply the deflection equation, Eq. 1, with the piecewise conditions described. This function should work for a single location (scalar), or multiple locations (vector). Your function should take one input, the location(s) along the beam where the deflection will be calculated, and produce one output, the deflection of the beam at the given location(s).

HINT: For most of the beam, Eq. 1 results in a negative answer. This implies the deflection is in the downwards direction.

<u>TASK 2: (3 – 5 min)</u>

In the main script, prompt the user to enter a location along the beam between 0 and 360 [in] (inclusive). Using your function, **FindDeflection**, calculate the deflection of the beam at this single point. Output this value to the command window, along with the position at which the deflection was calculated.

<u>TASK 3: (5 – 10 min)</u>

Although the value of the beam deflection at a given user-inputted location is useful, you would also like to develop the deflection profile from the fixed end of the beam (*x* = 0) to the user-inputted location. Create a vector of location values

from 0 to the user-inputted location. Using your function, **FindDeflection**, calculate the deflection of the beam at all positions in this vector.
Create a plot of the theoretical result for deflection [in] (y axis) at each location on the beam [in] (x axis). Your plot should include the following:

- X-axis label
- Title
- Y-axis label
- Gridlines
- Solid line (theoretical data)

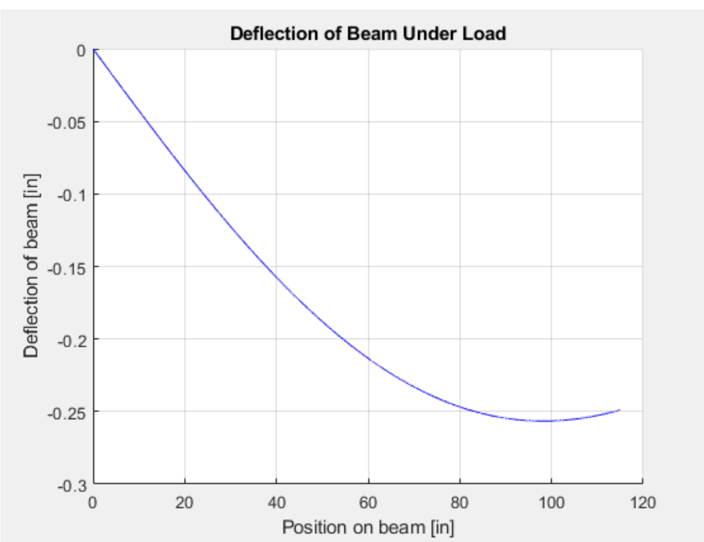
## Sample Output

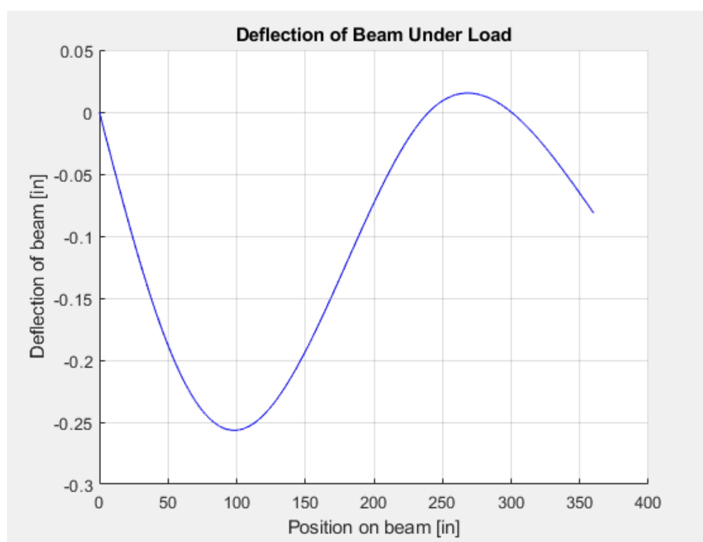
<u>Sample output for given data:</u>

Command Window

```
Please enter a position between 0 and 360 [in]: 115
The deflection of the beam at 115.0 [in] after the leftmost support is -0.249 [in].>>
>>
Please enter a position between 0 and 360 [in]: 360
The deflection of the beam at 360.0 [in] after the leftmost support is -0.081 [in].>>
```

- Additional test case scenarios are described in the Additional Resources folder accompanying this prompt
- Note that in this sample output the code has been run 2 times



## Problem #2: Resistor Decoder

### Background: Comprehension Time: 10 – 15 min

Most resistors are so small that the actual value would be too difficult to read if printed on the resistor. Instead, colored bands denote the value of resistance in ohms. Anyone involved in constructing electronic circuits must become familiar with the color code and with practice, one can tell at a glance what value a specific set of colors means.

When reading a resistor's color bands:

- The first two colored bands, pulled from Table 1, represent the first and second digits of the resistance, respectively
- The third colored band, pulled from Table 2, represents the multiplier applied to the first two digits. For example, if there are three zeros after the first two digits, that would be represented by an orange color as the third colored band. If there are no zeros after the first two digits, that would be represented by a black color as the third colored band. See test cases below for additional examples.

Table 1: Color Codes (**ColorCode**)

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| *Black* | *Brown* | *Red* | *Orange* | *Yellow* | *Green* | *Blue* | *Violet* | *Grey* | *White* |

Table 2: Multipliers (**Multiplier**)

| 1 | 10 | 100 | 1,000 | 10,000 | 100,000 | 1,000,000 |
|---|---|---|---|---|---|---|
| *Black* | *Brown* | *Red* | *Orange* | *Yellow* | *Green* | *Blue* |

The goal of this program is to convert a given resistance into a set of color bands or convert a given set of color bands into a resistance value. The two variables above, **ColorCode** and **Multiplier** are located in **ColorGuide.mat**.

### Tasks: Proficiency Time: 45 min – 1 hr 15 min

<u>TASK 1 (5 – 10 min):</u>
Ask the user whether they would like to convert resistance to color bands, or vice versa. Perform the following data validation on the menu selection: if the user exits out of the menu, produce an error that states that the user did not select a value in the menu and terminate the program.

<u>TASK 2 (15 – 25 min):</u>
Develop a user-defined function named **Resist2Color** to convert a resistance [ohms] to its corresponding color band. This function will take a single input, the resistance in ohms, and return a single output, the corresponding color band, as a 1x3 string vector. Assume that the input has been formatted in the form of a vector, where every digit is a different element in the vector. You can determine the corresponding color band using the information in **ColorCode** and **Multiplier**.

<u>TASK 3 (15 – 25 min):</u>
Develop a user-defined function named **Color2Resist** to convert a color band to its corresponding resistance [ohms]. The single input to this function will be the resistor's color band. Assume that the color band has been formatted in the form of a 1x3 string vector, where every color is a different element in the vector. You may also assume that the color names will always match the case sensitivity shown in Tables 1 and 2. Your function should then output the corresponding resistance [ohms], which can be determined using **ColorCode** and **Multiplier**.

<u>TASK 4 (10 – 15 min):</u>
In your main script, based on the user's choice from Task 1, accomplish the following:

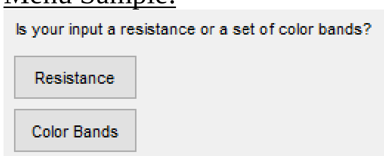
- If the user chose to convert a resistance to a color band, prompt the user to enter a resistance [ohms] as a vector, with each digit being a separate value in the vector. First, check that all digits entered after the first two digits are zeros. If they are not, produce an error message that terminates the program. Then, using **Resist2Color**, convert the resistance entered by the user to its corresponding color band and output the colors to the command window.
- If the user chose to convert a color band to a resistance, prompt the user enter the color band as one string array in a single input statement. First, check that there are only three colors entered. If not, produce an error message that terminates the program. Next, using **Color2Resist**, convert the color band entered by the user to its corresponding resistance [ohms] and output this to the command window.

## Sample Output:

<u>Sample output for given data:</u>

- Additional test case scenarios are described in the Additional Resources folder accompanying this prompt.

<u>Menu Sample:</u>




<u>Resistance to Color Bands (the first case shown is an example of an incorrect input causing an error):</u>

```
Command Window
Converting Resistance to Color Bands

Enter the resistance in ohms as a vector: [1 5 0 0 6]
Error using MA4_cougarnetID (line 14)
Invalid resistance entered. Program terminated.

Converting Resistance to Color Bands

Enter the resistance in ohms as a vector: [1 0 0 0]
fx The color bands for that resistance are: Brown Black Red>>
```

<u>Color Bands to Resistance:</u>

```
Command Window
Converting Color Bands to Resistance

Enter the color bands as a string array of 3 colors: ["Red", "Yellow", "Orange"]
fx The resistance for the given color bands is 24000 Ω.>>
```