General Instructions

Due Date:
Saturday, February 11 at 11:00pm (submit via blackboard)

Assignment Summary Instructions:
This assignment has three problems summarized below. You will use MATLAB as tool to solve the problems for the given test cases provided and that it is flexible for any additional test case that might be used to evaluate your code.

- Problem #1: Mower controls
- Problem #2: Decoding Resistors
- Problem #3: Springs in Design Packaging Systems

Submission Instructions:
You will submit a **.zip file** that will contain the following files:

1. **.m file – Matlab file (separate sections using %% for each problem)**. Your final script should be in the zipped folder and named as **MA#_USERNAME.m (for example, MA3_dwburles.m).**
2. .pdf file - scan of your algorithm sheet (use the template) for Problem #1
3. .pdf file - scan of your algorithm sheet (use the template) for Problem #2
4. .pdf file - scan of your algorithm sheet (use the template) for Problem #3

Submit your **.zip file** to blackboard using the Mastery Assignment 1 link. Your final submission should be a zipped folder named as **MA#_USERNAME.zip (for example, MA3_dwburles.zip).** The file must be completely submit before 11pm on February 11 for credit. **No late work is accepted and will result in a zero on the assignment**. It is your responsibility to make sure the file is completely submitted prior to the deadline. You are provided 2 upload opportunities in case your first upload is incomplete or a mistake.

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. **However, all support should be only verbal in nature**. Sharing files and showing other students your file is considered physical and visual help and is considered an academic honesty violation.

Some examples of academic misconduct in ENGI 1331 include but are not limited to the following actions:

1. Picking up and using or discarding another student's written or computer output;
2. Using the computer account of another student;
3. Representing as one's own the work of another on assignments, quizzes, and projects;
4. Giving another student a copy of one's work on an assignment before the due date.
5. Copying work from online resources (Chegg, google forums, etc.)
6. Posting work to online resources where other students can view your work

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 a high level of similarity, the ENGI 1331 faculty will review the files, and then the guilty parties will be turned in for an academic honesty violation 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 and will result in a violation.

Problem #1

Background: *For the protection of both the operator of a zero-turn radius mower and the mower itself, several safety interlocks must be implemented. These interlocks are shown in the table below with the state associated in each.*

Interlock Description	*States*
Brake Switch	*On or Off*
Operator Seat Switch	*Seated or Not-seated*
Blade Power Switch	*Turning or Off*
Left Guide Bar Neutral Switch	*Neutral or Forward or Back*
Right Guide Bar Neutral Switch	*Neutral or Forward or Back*
Ignition Switch	*Run or Off*
Motor Power Interlock	*Enabled or Disabled*

For the motor to be enabled (thus capable of running), all of the following conditions must be true:

- *The ignition switch must be set to "Run".*
- *If the operator is not properly seated, both guide levers must be in the locked neutral position.*
- *If either guide lever is not in the locked neutral position, the brake must be off.*
- *If the blades are powered, the operator must be properly seated.*

The goal of this program is to determine whether the motor should be enabled or not based on menu selections for all of the interlock descriptions above it. If any selection is not chosen (user exits out of menu), the program should show an error and the program should terminate before showing results.

Algorithm Requirements:

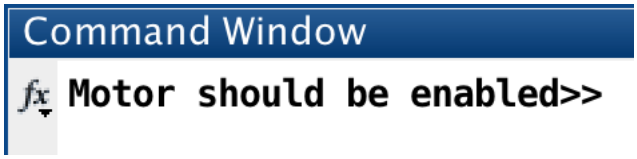
- Fill in the algorithm template provided on blackboard. It can be typed or by hand.
- Save or scan the file to a .pdf file and include in your final .zip submission

Coding Requirements:

- Include required documentation and header for the problem.
- Include menu selections for all interlock descriptions except Motor Power Interlock.
- The output of your program should look like the sample output displayed below with the same spacing, indentations, and number of significant digits.

Sample Output to Command Window based on the following input case:

Brake Switch	Operator Seat Switch	Blade Power Switch	Left Guide Bar Neutral Switch	Right Guide Bar Neutral Switch	Ignition Switch
On	Seated	Off	Neutral	Neutral	Off



Other test cases:

Test Case	*Inputs*						*Output*
	Brake Switch	Operator Seat Switch	Blade Power Switch	Left Guide Bar Neutral Switch	Right Guide Bar Neutral Switch	Ignition Switch	*Motor*
Test Case 1	On	Seated	Off	Forward	Neutral	Off	Not Enabled
Test Case 2	On	Seated	Off	Forward	Neutral	**BLANK**	Missed a Selection

Problem #2

Background: *Most resistors are so small that the actual value would be 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. For the novice, however, trying to read color codes can be a bit challenging.*

When reading a resistor color band:

- *the first digit is represented by the first color in the color band pulled from Table 1*
- *the second digit is represented by the second color in the color band pulled from Table 1*
- *the remaining digits represent the multiplier used on 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 color band. If there are no zeros after the first two digits, that would be represented by a black color as the third color 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: Multiplier (**Multiplier**)*

1	10	100	1000	10000	100000	1000000
Black	Brown	Red	Orange	Yellow	Green	Blue

The goal of this program is to first determine if the user will be entering a resistance or a color band.

1. *If the user chooses resistance, the user will enter a resistance in ohms as vector with each digit being a separate value in the vector and the program should display (in text) the color bands in the order they will appear on the resistor.*
   - Check that every digit entered after the first two digits are zero. If not, your program should show an error message and terminate.
2. *If the user chooses color band, the user will enter be asked to enter the color band, as one cell array in a single input statement, and the program should display the resistance in ohms.*
   - Check that there are only three colors entered. If not, your program should show an error message and terminate.
3. If no option is chosen, the program should show an error message and terminate.

Algorithm Requirements:

- Fill in the algorithm template provided on blackboard. It can be typed or by hand.
- Save or scan the file to a .pdf file and include in your final .zip submission

Coding Requirements:

- Include required documentation and header for the problem.
- Load the .mat file provided and reference those tables in determining those arrays or resistance
- The output of your program should look like the test case displayed below with the same spacing, indentations, and number of significant digits.

Sample Output to Command Window based two conversion options:

```
Command Window
  Converting Resistance to Color Band

  Enter the resistance in ohms as a vector:  >> [1 0 0 0]
fx The correct color band is: Brown Black Red>>
```

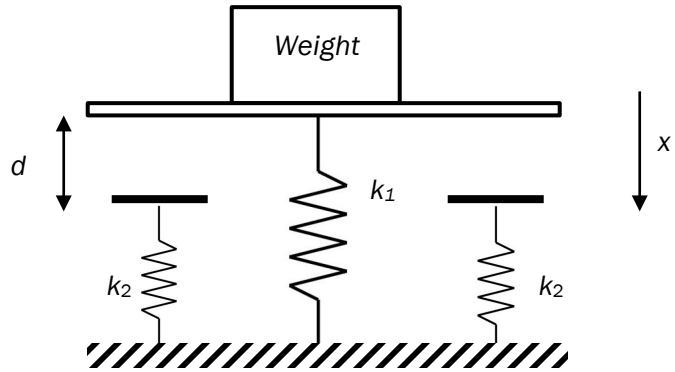
```
Command Window
  Converting Color Band to Resistance

  Enter the color band as a cell array (3 colors): >> { 'Black', 'Green', 'Orange'}
fx The restistance for the given color band is 5000.>>
```

Other Test Cases

Test Case	Choice	Input	Output
Test Case 1	Resistance	[5 6]	Green Blue Black
Test Case 2	Resistance	[5 6 5]	Error: Program terminates
Test Case 3	Color Band	{ 'Brown', 'Orange', 'Black'}	13
Test Case 4	Color Band	{'Orange', 'Blue'}	Error: Program terminates
Test Case 5	No Selection	Program Terminated	Program Terminated

Problem #3

Background: *The figure below shows a mass-spring model of the type used to design package systems and vehicle suspensions, for example. The springs exert a force that is proportional to their compression, and the proportionality constant is the spring constant k. The two side springs provide additional resistance if the weight is too heavy for the center spring. When the weight is gently placed, it moves through a distance, x, before coming to rest at a position, d.*



From statics, the weight force must balance the spring force at this new position resulting in the following equations

$$W = k_1 x \qquad \text{if } x < d$$
$$W = k_1 x + 2k_2(x-d) \qquad \text{if } x \geq d$$

The goal of the program is to determine the distance it moves, x, based on the user entered values for W, K_1, K_2, and d. Allow the user to enter a W value as a single value or a vector of many different weights.

- *If the user only enters a single W value, your program should produce a single x value.*
- *If the user enters a vector of W values, your program should calculate a vector of x values (one for each weight). From this vector,*
  - *Determine the maximum weight entered and the associated distance it moves through.*
  - *Plot weight (x-axis) versus distance moved through, x, on a basic plot. Include at minimum the title and axis labels.*

Algorithm Requirements:

- Fill in the algorithm template provided on blackboard. It can be typed or by hand.
- Save or scan the file to a .pdf file and include in your final .zip submission

Coding Requirements:

- Include required documentation and header for the problem.
- The output of your program should look like the test case displayed below with the same spacing, indentations, and number of significant digits.

Sample Output to Command Window (2 case – one for a single value of W and one for a vector for W):

Case 1:

```
Command Window
  Enter the weight [N]:  >> 500
  Enter the k_1 value [N/m]:  10000
  Enter the k_2 value [N/m]:  15000
  Enter the rest position, d [m]:  1
fx The weight will pass through a distance of 0.0500 m.>>
```

Case 2:

```
Command Window
  Enter the weight [N]:  >> [100 500 2000 400 5000 950 1500 7000]
  Enter the k_1 value [N/m]:  1000
  Enter the k_2 value [N/m]:  1000
  Enter the rest position, d [m]:  .7
fx The max weight given (7000) will pass through a distance of 2.80 m.>>
```