Guidelines:

1. Work in your assigned teams to complete all tasks in lab today
2. You may turn in the assignment tomorrow if you need more time for the reflection questions
3. Each student will turn in their own work
4. The objective is to test a speed controller for your DC motor

Project Description:

At your station you will see an unloaded DC motor controlled by output from a microprocessor. Motor speed will also be measured by the microprocessor and plotted in Matlab. There is also a belt to add a load, but leave this off until you need it.

Tasks:

1. Install and setup the Matlab script for this project
    a. Download the *Project 5 Matlab Files* zipped folder from D2L, and unzip it to your desktop
    b. Set the correct com port in the *serial_start* file
        i. Start the Device Manager
        ii. Expand out the Ports tab
        iii. Look for which com port the USB Serial Device is on (you can unplug and plug it in if you are not sure which is correct)
        iv. Open the *serial_start.m* file and make sure the correct com port is used in the serial command on line 11
    c. Open the *Project5_openloop* and *Project5_PIControl* files
2. Explore the open loop control response of your DC motor
    a. I have set your voltage source near 10 V, but you need the exact value. In the *Project5_openloop* code, set the *V_max* variable to the voltage displayed on your multi-meter.
    b. By varying the *V_in* value in the code each time you run it (NOTE: Do not adjust the external supply voltage!), find out which input voltage will result in a 300 rad/s output speed on your motor. You will find that this is not as predictable as you would expect, and it will be difficult to get an accurate number. Once you have tried around five values, go ahead and just pick a number that you think is pretty close. What is that input voltage value?

    c. Add the belt between your two motors to add an unexpected load. By what percentage is the output speed reduced when you do this? (Take the belt back off when you are done with this.)

3. Explore proportional control of the motor
    a. In the *Project5_PIControl* code, set the *V_max* variable to the voltage displayed on your multi-meter.
    b. Leave the *Ki* variable at zero for now, but play around with the *Kp* variable to explore what is possible with proportional controller
        i. Find the maximum *Kp* value before the control signal exceeds the allowed 10 V. You can tell that you have exceeded this when the green line in the bottom graph has flat portions that don't follow the proportional control line. Your number should be close to the maximum *K* value you found in Project 4 Task 4f. Put the maximum *Kp* in the bottom left element of the table below.
        ii. Find the minimum *Kp* value that will still cause continuous motion of the DC motor (it doesn't stall intermittently or just vibrate). Put this value in the top left element of the table below.
        iii. Select two other *Kp* values spaced between the minimum and maximum values to fill out the rest of the first column.
    c. Record metrics for how the controlled system responds based on the response graphs when you run the control code.

Gain (Kp)	Settled Speed	Settled Error (e_ss)	Settling Time

    d. Reflection question: Does error go down as control gain increases as our theory predicted?

    e. Reflection question: Can proportional gain alone reduce the error sufficiently?

    f. Reflection question: What do you think is causing the oscillating behavior in the output speed?

4. Test your designed PI controller
    a. Write down your *Kp* and *Ki* values from Project 4, Task 6b-vi.

    b. Program the values from Task 4a into the *Project4_PIControl* code, and run this controller. You can tweak the parameters and run it again to find a better response if you want.
    c. What is the steady-state error of this controller response?

    d. What is the settling time of this controller response?

    e. Reflection question: In what ways does this response surprise you in comparison with your simulated step response from Project 4, Task 5d?

    f. Add the belt in to test the controller's response to an unexpected load. Run the code with the same control parameters.
    g. Did the steady-state error increase? If so, by how much?

    h. Reflection question: Would you describe the PI control approach as a successful for this application? Why or why not?

    i. Reflection question: Do you feel that this series of hands on project has been more or less beneficial than classroom lectures? Why?

    j. Optional reflection question: What changes would you recommend for future versions of this series of projects?