Laboratory Worksheet #11
PD Control and Type Casting Exercise

Exercise 1: Type Casting Calculations in the Motor Control Algorithm

In Laboratory 6 of the Course Material, download the `worksheet_11.c` file. Looking through the file, you can see six (plus a 16-bit version) different expressions that represent the same equation but have different typecasting of the variables in the equation. One of the expressions is a two byte form of temp_motorpw and is included as a further example of potential problems. For reference, the different expressions are shown below.

```
error = desired - actual;

/* 1st control algorithm equation */
        temp_motorpw = pw_neut + kp*(error) + kd*(error - prev_error);
/* 2nd control algorithm */
        temp_motorpw = pw_neut + (signed long)kp*(error) + kd*(error - prev_error);
/* 3rd control algorithm equation */
        temp_motorpw = (signed long)(pw_neut + kp*(error) + kd*(error - prev_error));
/* 4th control algorithm equation */
        temp_motorpw = pw_neut + kp*(signed int)(error) + kd*(signed int)(error - prev_error);
/* 5th control algorithm equation */
        temp_motorpw = (signed long)pw_neut + (signed long)(kp*(error)) + (signed long)(kd*(error -
                        prev_error));
/* 6th control algorithm equation */
        temp_motorpw = (signed long)pw_neut + (signed long)kp*(signed long)(error) + (signed
                        long)kd*(signed long)(error - prev_error);

prev_error = error;
```

temp_motorpw is the calculated pulsewidth to be implemented
pw_neut is the pulsewidth when the blimp is heading in the desired direction
k_p is the proportional gain constant of the control algorithm (use a value of 30 for the calculations)
k_d is the derivative gain constant of the control algorithm (use a value of 30 for the calculations)
desired is the reading from the input desired heading
actual is the latest heading measurement
prev_error is the previous calculation of the difference between desired and actual headings

The following four cases represent different physical conditions for the blimp. Based on the numbers provided, calculate the resulting value of temp_motorpw (use a calculator). Run the worksheet10.c code after editing the appropriate variables and compare your calculation with results from the different algorithms. Indicate which algorithms provide the expected answer. Remember, negative results are possible. Use a proportional gain of Kp = 30 and a derivative gain of Kd = 30. Record which typecasting algorithms are consistent with your calculation.

Case 1: center_motorpw=2765, prev_error=-1760, desired=1800, actual=3500 (blimp is turned too far to the right)
temp_motorpw (calculated) = -46435
Algorithms that provide a correct result: 2 & 6

Case 2: center_motorpw=2765, prev_error=1760, desired=3500, actual=1800 (blimp is turned too far to the left)
temp_motorpw (calculated) = 51965
Algorithms that provide a correct result: 1, 4, & 6

Case 3: center_motorpw=2765, prev_error=-250, desired=50, actual=250 (blimp is turned too far to the right)
temp_motorpw (calculated) = -1735
Algorithms that provide a correct result: 2 & 6

Case 4: center_motorpw=2765, prev_error=20, desired=3500, actual=1800 (rudder fan is at full power, but blimp is turning slower than desired)
temp_motorpw (calculated) = 104165
Algorithms that provide a correct result: 2, 5, & 6

Exercise 2: Code execution

Based on your observations, **implement one of the typecasting algorithms in your code** where the pulsewidth is determined for the steering servo. You will need to work with long variable typecasting. Refer to the gondola_info sheet for suggestions on refining your code. Download your code to the microcontroller on a gondola. Set a desired heading of 135° (SE) and a proportional gain constant of 12. Fill in the first two columns in the table below, correcting your error is necessary so that it is bounded $-180^{o} < \text{error} < 180^{o}$.

Run your code and manually position the gondola at the heading directions indicated in the table and fill in the table using output from your code. You will need to print both the 'raw' (before limit correction) calculated pulsewidth and the 'corrected' (after limit correction) pulsewidth. As indicated in the table, record both calculated temp_motorpw before you check for pulsewidth limits and the actual pulsewidth after limits are enforced. Again, remember that the 'raw' pulsewidth can be a negative number. Note, since you are holding the gondola stationary, the differential gain term is zero (previous_error-current_error = 0).

Heading	Heading Error	Manually calculated temp_motorpw	Program calculated temp_motorpw (before limit correction)	Final temp_motorpw (after limit correction)
0.0°	135	4385	4385	3502
45.0°	90	3845	3845	3502
90.0°	45	3305	3305	3305
135.0°	0	2765	2765	2765
180.0°	-45	2225	2225	2225
225.0°	-90	1685	1685	2028
270.0°	-135	1145	1145	2028
315.0°	-179	617	617	2028

When complete, insert Worksheet 11 in your laboratory notebook. Worksheets are required when the notebooks are graded. Perform any necessary calculations on the left page of the notebook where the worksheet is placed. Keep individual copies of the worksheet for your own records.