QUESTION 1  Vehicle Drive Control (15 points)

Complete the C program of a differential-drive controller for two motors using two P- and one I-controller as per this diagram:

Note: Assume the "control" subroutine is called 50 times per second in background.

```c
#define Kp = 0.5
#define Ki = 0.2
int desired_speed = 500; /* ticks per sec.*/

void control(void)
{ int L_ticks, R_ticks, L_speed, R_speed, L_err, R_err, LR, RR;

    L_ticks = QuadRead(L_enc);   R_ticks = QuadRead(R_enc);

    MotorDrive(L_mot, LR);    MotorDrive(R_mot, RR);
}
```
QUESTION 2  

SENSOR INTERFACING  

A digital distance sensor sends 4-bit words as digital output (most significant bit first) when triggered by 4 input pulses of 1ms high/low:

![Diagram of sensor input and output]

example output: $1\ 0\ 1\ 1_2 = 11_{\text{dec}}$

Write a C subroutine to read and return data words (integer) from the sensor

**Note:** Data should only be read at the **falling edge** of the input signal

Your program has to put 4 bits together into an integer in the right order

Use pre-defined subroutines `set_sensor(x)` to set the sensor input to 1 or 0, `int read_sensor()` to read a single bit and `wait(1)` to wait for 1ms

```c
int sensor(void)
{
    int result;

    // Code to read sensor data
    return result;
}
```
QUESTION 3  TIMER-INTERRUPTS (10 points)

In an electric car, a digital “fuel gauge” (calculating remaining kilometers that can be driven) should be updated once every second by reading the analog input (battery voltage), calculating and writing the result to the LCD at position [0,0]. Write a C subroutine to implement this.

Note: The formula for converting Volts into km is: \( \text{dist} = 80 \text{km} \times (\text{batt\_volt} - 120)/24 \)

Limit printed values to the range [0..80km]

In \text{main}, complete pre-defined subroutine \text{OSAttachTimer}(x,y) to activate periodic calls to \text{fuel\_gauge} \textbf{once per second}

In subroutine \text{fuel\_gauge}, use pre-defined subroutine \text{int OSGetAD(batt)} to read and convert the current battery voltage level and \text{LCDSetPrintf(..)} to write the remaining kilometers to the screen at position [0,0].

```c
int main()
{
    ... 
    OSAttachTimer(___________, ___________);
    ... 
}

void fuel_gauge(void)
{
    ...
}
```