Lab 9 Sensors and Conditionals
ENGR 1204: Electrical and Computer Engineering Module

Theory

- **Variables in the C programming language**
    - A variable in a computer program is a name for memory location used to store some value used by the program. A variable is created in C by specifying a data type, a name, and optionally an initial value.
        - types include
            - integer → int
            - floating point → float
        - The name of the variable can be any continuous string of letters and numbers (no spaces, no punctuation marks other than the underscore character (as in hi_there), cannot start with a number, may have any mixture of capital/small letters.
    - *Examples*
        - `int rightFloor; // rightFloor has type integer`
        - `int Lft = 0; // Lft is integer, initial value of 0.`
        - `float pi = 3.14159;  // pi is a floating point number`

- **Console window for Arduino**
    - Human programmers (like yourself) often need to know the values of variables in order to know if the program is running correctly or not. One way to do this is to print text strings and number values to the display. Microcontrollers like the Arduino have no display, but they can print to a console window on a PC that is connected to the Arduino by USB cable. Several lines of code can set this console up and display the values of variables as code is running.
        - In the `setup()` function, the following statement initializes the console window on PC to print text and numbers. The 9600 is a standard transmission rate in baud, which is essentially the number of bits per second transmitted.

          ```
          Serial.begin(9600); // initialize console window on PC
          ```

        - In the `loop()` function, these statements print a variable value to the Arduino console window on the PC.

          ```
          Serial.println(Lft); // Write out the value of variable Lft
          ```

- **Sensors**
    - Sensors convert a phenomenon of nature (e.g. light, temperature, humidity, force, pressure, sound, acceleration) into an electrical signal, such as a voltage. Typically, the output of sensors is tiny and noisy, so electrical circuits are necessary to do signal conditioning, i.e. amplify and filter.
    - The front side corners of your car have optical sensors pointed down toward the floor. These can be used to detect color of the floor. The sensor outputs infrared light from an LED down to the floor, and any light reflected back is detected by

an optical sensor. The sensor module includes the necessary amplification and filtering circuits.
    - The sensor is powered by Ground and 5 V supply. The signal conditioning maps the detected color range to an analog voltage between Ground (0 V) and 5 V. Voltage values can be observed using an oscilloscope or a digital multimeter (DMM). The lab today will use the DMM.

- **Analog to Digital Converter**
    - For the software to know what the analog voltage from the sensor is, the voltage needs to be digitized (converted to a number). The microcontroller chip of the Arduino can do analog-to- digital conversion (ADC). The analog inputs on the Arduino are pins A0 – A5.
    - A software function controls the ADC and returns a number that can be stored in a variable. This example reads the analog voltage on pin A0, returns a number, and the number is stored in a variable of type integer named rightFloor.

```
int rightFloor = analogRead(A0); // convert from pin A0
```

- **Flow Charts**
    - Flow charts are diagrams used to show the steps and logical decisions for some control or computing procedure. The following way to draw a flowchart will be followed for these labs:
        - The word **start** and **end** (or words with the same meaning) are placed inside of a **circle**. Microcontroller programs typically have no end, since stop running only when power is turned off.
        - **Statements** of some step(s) to do are placed inside of **rectangles**.
        - **Conditions** (questions) are placed inside of **diamonds**. Two corners of the diamonds are chosen as the two possible output pathways and are labeled True (yes) or False (no). The condition is a comparison such as “equal to”, “greater than”, or “less than”. Based on the result of the condition, only one of the two branches (the “True” path or the “False” path) will be followed by the computer.
        - **Arrows** connect all boxes.
        - Loops in the processing are shown by **arrows that go back** up to a higher location in the flowchart.

- **Conditional Branches in C**
    - Logic developed in flow charts can be used to write a computer program.
    - Flowcharts provide a visual model of your logic.

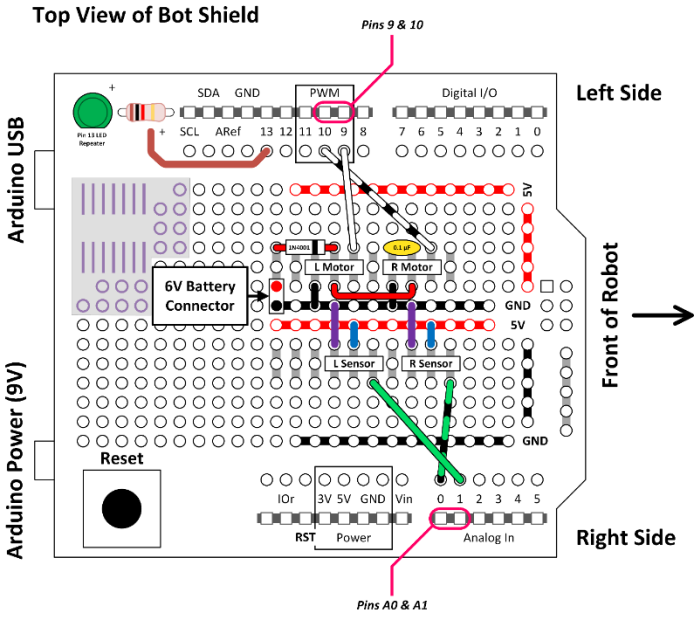
In-Lab Work

Materials Required per Team:

- Sumobot Kit (may try to use same # as prior lab)
  - Sumobot with USB cable (1)
  - Bot Shield (1) – already attached
  - Bag of wires (1)
- Banana-to-alligator clip leads (1 pair)

Procedure

- Start another document for Lab 9 (on paper or word document). Name Lab9_XYZ (but replace XYZ with your name or initials).
- Copy the Lab 8 Arduino code by using File → Save As, and place in a new Lab9 folder, and name the new file as Lab9_SensorConditional_XYZ, but replace XYZ with your name or initials.
- In code text editor, modify 1st line comment in Lab 9 code, so that indicates a title for Lab 9 and your name.
- Determine typical sensor values for black or white, as follows.
  - The wires for the **right front floor sensor** are connected to Arduino pin **A0** for analog input to Analog-to-Digital-Conversion. The Arduino shield board also provides Ground and 5V to the sensor. These are the signals on the 3 wire cable between the sensor and the botShield.
    - sensor Purple wire ↔ Ground
    - sensor Blue wire ↔ 5 V supply
    - sensor Green wire (the sensor output) ↔ Arduino pin A0.
  - Use cable and wires to connect the black lead from the benchtop instrument, Digital Multimeter (DMM), to any labelled GND on the Arduino. Connect the red lead of the DMM to pin A0 of the Arduino. Set the DMM to read DC voltage.
    - Ask instructor if not sure how to connect the Arduino to the DMM.



Modify code as follows:

- In the `setup()` function of the code:
  - add a statement to initialize the console window on PC to print text and numbers. See Theory section above for example code.
  - Leave the code line that does the following:
    - makes the PWM pins to be in OUTPUT mode
    - sets the period for PWM
  - Move out (comment out now, and later move to setup()):
    - sets the PWM pulse-width, duty cycle
- In the `loop()` function of the code:
  - Add a code line to read the analog value from the sensor. The analog voltage on pin A0 is converted by the Arduino chip to a digital integer value. Store the new digitized value in an integer variable named `rightFloor`, or a similar name. See Theory section above for example.
  - Add statements to print the read value (`rightFloor`) to the Arduino console window on the PC. See Theory section above for example code.
- Run the program while holding the right sensor of the car over the white and black paper, and observe the resulting value displayed on the console window of PC.
  - Note, the wheels do not need to be turning for this step, so do not connect motor power yet.
  - Observe in the console window on the PC values from the sensor for when the sensor was over white and black floor colors. Fill in the chart for about five typical values observed. Put this table in your report.

Typical Values Observed from Floor Sensor			
Over Black Color		**Over White Color**	
Analog Voltage from DMM	**Number value in Console**	**Analog Voltage from DMM**	**Number value in Console**
0.258 V	50	4.911 V	991
0.259 V	50	4.933 V	993
0.260 V	51	4.929 V	992
0.2606 V	50	4.919 V	987
0.2607 V	51	4.915 V	983

Modify code as follows:

- Above the setup() function, add a global variable (global variable is one that can be seen and used by all functions) of type integer to hold a threshold value. Store into this threshold value some number between the typical number values for Black and for White. This threshold will determine what inputs are considered either Black or White (for this code, the world will only be Black or White, no gray or other colors!). Replace the '?' in below code example with your chosen number for threshold:

```
int thresh = ?; // threshold to distinguish Black and White
```

- In the `loop()` function after the statements to read the sensor and print sensor values to the console, add code for a conditional branch. The logic of the branch should be as follows:
  - if the color is Black make the car move forward (both wheels forward),
  - otherwise make the car go backwards.
  - Recall that the a line from setup() in Lab 8 set the pulse-width of the PWM signal, and thus determined if wheel going forward or backwards. That line of code may now be useful here in loop().

- Test this function with the colored paper: (Black → Forward; White → Backward).

- **Draw a flowchart** for your program code. Insert this flow chart into your report.

Demo the function (Black → Forward; White → Backward) to instructor.

Submit the Report (*(each person turns in their own report)*)

Save your report as a PDF file from MSWord.
Report should include

- Title section at top (name, lab, class, date)
- Introduction
  - Short summary of what you did, procedure, and results
- Procedure and Result
  - Present your calculations, flowchart, code (paste in the Arduino software code), and table
  - Clearly explain your method and discuss results
- Conclusion
  - Short summary of your report. Did you achieve your goal? Discuss problems that you faced and what you did to solve it?
    - Submit Report to Brightspace for Lab 9.