Name: Shaan Yadav
NetID: ay140
Honor Code: *I have adhered to the Duke Community Standard in completing this assignment.*

Deliverable (1):

Parts: 2 resistors, wires as needed, red LED, green LED, A1324 Hall effect sensor, CX Bot

```
// parts - two resistors, a red LED, a green LED, an A1324 Hall effect sensor, and wires as needed.

const int Hall_In = 0;
const float VCC = 5.0;
const float Hall_sensitivity = 0.005; // 5 mV/G

#define RED 6
#define GRN 2

void setup() {
 pinMode(RED, OUTPUT);
 pinMode(GRN, OUTPUT);
 Serial.begin(9600);
}

void loop() {
 float Hall_Reading = analogRead(Hall_In);
 float Hall_Voltage = Hall_Reading * 5.0 / 1023.0;
 float Hall_Gauss = (Hall_Voltage - (VCC/2)) / 0.005;

 if (Hall_Gauss > 200) {
   digitalWrite(RED, HIGH);
   digitalWrite(GRN, LOW);
 } else if (Hall_Gauss < -200) {
   digitalWrite(RED, LOW);
   digitalWrite(GRN, HIGH);
 } else {
   digitalWrite(RED, LOW);
   digitalWrite(GRN, LOW);
 }



 Serial.print("Analog reading = ");
 Serial.print(Hall_Reading);
 Serial.print(" ");
 Serial.print("Analog voltage = ");
 Serial.print(Hall_Voltage);
 Serial.print(" ");
 Serial.print("Hall_Gauss = ");
 Serial.println(Hall_Gauss);
 delay(100);
}
```

Deliverable (2):

Parts: Color Sensor (TCS34725), CX bot

```
// Based on colorview.ino from Adafruit_TCS34725 library
// Simplified and modified to work with CX-Bot

#include "Adafruit_TCS34725.h"
#define redpin 45
#define greenpin 46
#define bluepin 44


// Initialize color sensor
Adafruit_TCS34725 tcs = Adafruit_TCS34725(TCS34725_INTEGRATIONTIME_50MS,
TCS34725_GAIN_4X);

void setup() {

 pinMode(redpin, OUTPUT);
 pinMode(greenpin, OUTPUT);
 pinMode(bluepin, OUTPUT);

 Serial.begin(9600);
 Serial.println("Sensor Test!");

 if (tcs.begin()) {
   Serial.println("Found sensor");
 } else {
   Serial.println("No TCS34725 found ... check your connections");
   while (1); // halt!
 }
}

void loop() {
 float red, green, blue;

 delay(60);  // takes 50ms to read
 tcs.getRGB(&red, &green, &blue);

 analogWrite(redpin, 255 - int(red));
 analogWrite(greenpin, 255 - int(green));
 analogWrite(bluepin, 255 - int(blue));
 delay(100);

 Serial.print("R:\t"); Serial.print(int(red));
 Serial.print("\tG:\t"); Serial.print(int(green));
 Serial.print("\tB:\t"); Serial.print(int(blue));
 Serial.print("\n");
}
```

Deliverable (3):

Parts: Infrared Thermal Sensor (MLX90614), Multi-Character Liquid Crystal Display (27977), 7 wires, CX bot

```
#include <SoftwareSerial.h>

#define TxPin 14

SoftwareSerial mySerial = SoftwareSerial(255, TxPin);

#define num 17

#include <Wire.h> // I2C library, required for MLX90614
#include <SparkFunMLX90614.h> //Click here to get the library:
http://librarymanager/All#Qwiic_IR_Thermometer by SparkFun

IRTherm therm; // Create an IRTherm object to interact with throughout

int durs[num]  = {211, 211, 211, 210, 210, 211, 211, 211, 211, 211, 211, 211, 210,
210, 211, 211, 212};
```

```
int octs[num]  = {216, 216, 216, 216, 216, 216, 215, 216, 216, 216, 216, 216, 216,
216, 216, 215, 216};

int notes[num] = {220, 220, 220, 220, 220, 220, 224, 232, 220, 220, 220, 220, 220,
220, 220, 224, 232};

void setup() {
 Serial.begin(9600);
 mySerial.begin(9600);
 delay(100);
 mySerial.write(12); // clear
 delay(10);
 mySerial.write(22); // no cursor no blink
 delay(10);
 mySerial.write(17); // backlight
 delay(10);
 // mySerial.print("Super Mario");
 // mySerial.write(13);
 // mySerial.print("Brothers!");


 Wire.begin(); //Joing I2C bus
  if (therm.begin() == false){ // Initialize thermal IR sensor
   Serial.println("Qwiic IR thermometer did not acknowledge! Freezing!");
   while(1);
 }
 Serial.println("Qwiic IR Thermometer did acknowledge.");
  therm.setUnit(TEMP_F); // Set the library's units to Farenheit (sic)
 // Alternatively, TEMP_F can be replaced with TEMP_C for Celsius or
 // TEMP_K for Kelvin.
  pinMode(LED_BUILTIN, OUTPUT); // LED pin as output

 // for(long k=0; k<num; k++){

 //   mySerial.write(durs[k]); mySerial.write(octs[k]); mySerial.write(notes[k]);
 //   int len = 214 - durs[k];
 //   float del = 2000 / pow(2, len);
 //   delay(int(del*1.1));
 // }


}
```

```
void loop() {

 digitalWrite(LED_BUILTIN, HIGH);

 // Call therm.read() to read object and ambient temperatures from the sensor.
 if (therm.read()) // On success, read() will return 1, on fail 0.
 {
   // Use the object() and ambient() functions to grab the object and ambient
 // temperatures.
 // They'll be floats, calculated out to the unit you set with setUnit().
   mySerial.print("Object: " + String(therm.object(), 2));
   mySerial.println("F");
   // mySerial.print("Ambient: " + String(therm.ambient(), 2));
   // mySerial.println("F");
   mySerial.println();
 }

 if (therm.object() < 50) {
   for(long k=0; k<num; k++){
     mySerial.write(durs[k]); mySerial.write(octs[k]); mySerial.write(notes[k]);
     int len = 214 - durs[k];
     float del = 2000 / pow(2, len);
     delay(int(del*1.1));
   }
 }

 digitalWrite(LED_BUILTIN, LOW);
 delay(10000);

}
```

Deliverable (4):

Parts: Multi-Character Liquid Crystal Display (27977), Radio-Frequency Identification Reader (ID-12LA), 6 wires, CX Bot

```

// Based on https://www.instructables.com/Reading-RFID-Tags-with-an-Arduino/
// Expanded by Michael R. Gustafson II to store code

#include <SoftwareSerial.h>

```

```
#define TxPin 14

SoftwareSerial mySerial = SoftwareSerial(255, TxPin);

char val = 0; // variable to store the data from the serial port
int len = 12;

void setup() {
 Serial.begin(9600); // connect to the serial port for the monitor
 Serial1.begin(9600); // connect to the serial port for the RFID reader
 mySerial.begin(9600);

 mySerial.write(12);
 mySerial.write(22);
 mySerial.write(17);
}

void loop () {
 char rfidData[len+1] = {};
 int get_more = 1;
 int i = 0;
  while(get_more == 1){
   if(Serial1.available() > 0) {
   val = Serial1.read();
      // Handle unprintable characters
     switch(val) {
       case 0x2: break;               // start of transmission - do not save
       case 0x3: get_more = 0; break; // end of transmission - done with code
       case 0xA: break;               // line feed - do not save
       case 0xD: break;               // carriage return - do not save
       default:  rfidData[i]=val; i+=1;  break; // actual character
     }
   }
 }
 Serial.println(rfidData);
 mySerial.println(rfidData);

 delay(1500);

 mySerial.write(12);
}
```