```
//Group 13 Senior Design
//Diabetic Breathalyzer Hardware Code
//Jonathan Brown, Edert Geffrard, Christine Sleppy, Noah Spenser

#include <SoftwareSerial.h>
#include"DHT.h"
#include <math.h>
#define DHTPIN 5
#define DHTTYPE DHT22
#define TGS_Ro 4.62 //TGS_Ro=80/17.31
#define WSP_Ro 180
#define TGS_RL 4.69
#define WSP_RL 4.7
#define TGSBase 50 //Limits that must be met for stability.
#define WSPBase 100

SoftwareSerial BTserial(0, 1); // RX | TX
DHT dht(DHTPIN, DHTTYPE);

float getResTGS(float, float); //sends back single RES value for each
float getResWSP(float, float);

float getAvgTGS(float, float); //gets moving average of RES value for each
float getAvgWSP(float, float);

float getPpmTGS(float, float); //send back PPM value for each
float getPpmWSP(float, float);

void sendValues(float, float); //will send values to BT

float TGS_Array[9]= {0,0,0,0,0,0,0,0,0};
float WSP_Array[9]= {0,0,0,0,0,0,0,0,0};



void setup()
{
    BTserial.begin(9600); //Start serial comm and setup status LED
    pinMode(6, OUTPUT); //Green
    pinMode(7, OUTPUT); //Blue
    pinMode(8, OUTPUT); //Red
    pinMode(9, INPUT);
    dht.begin();

    digitalWrite(7, HIGH); //Programmable Warmup upon initial power up.
    delay(5000);
    digitalWrite(7, LOW);

    }

void loop() {

char PreStabCheck = 0; //Value that is set when button held down.
char PostSet = 0; //Value that is set when button has been released, but post analysis hasn't run.
char PostStabCheck = 0; //Value that is set after button release. Turns on red light instead of running through post analysis.

  //While loop for chekcing button status
  while(digitalRead(9)==HIGH){ //while button pressed
    PostSet = 1;
      if(PreStabCheck == 0){

      digitalWrite(7, HIGH); //Turn on blue LED to show button pressed
      digitalWrite(8, LOW);
      digitalWrite(6, LOW);
      delay(500);
      float  h = dht.readHumidity(); //Get temp and humidity readings
      float  t = dht.readTemperature();
      if (isnan(h) || isnan(t)) {
        BTserial.println("Failed to read"); //Make sure no error in readings
```

```
//Group 13 Senior Design
//Diabetic Breathalyzer Hardware Code
//Jonathan Brown, Edert Geffrard, Christine Sleppy, Noah Spenser

        return;}

      float maintempTGS = getAvgTGS(t,h);
      float maintempWSP = getAvgWSP(t,h);

      if((maintempTGS > TGSBase) && (maintempWSP > WSPBase)){ //Check if levels high enough upon initial button press

      digitalWrite(7, LOW);
      digitalWrite(8, HIGH); //Green light comes on if levels reached
      digitalWrite(6, LOW);
      PostStabCheck = 1;
      PreStabCheck = 1;}} //PreStabCheck set to 1 so light blue light stays off

    PostSet = 1;}

//Post Analysis Loop
  if(PostSet == 1){ //means the button was released, but post analysis loop has not run

      if(PostStabCheck == 0){ //means button was released but stable level not met
      digitalWrite(8, LOW);
      digitalWrite(6, HIGH);
      digitalWrite(7, LOW);
      delay(2000);
      return;} //puts red light on for 2 seconds, returns

    float  h = dht.readHumidity(); //Get temp and humidity readings
    float  t = dht.readTemperature();
      if (isnan(h) || isnan(t)) {
        BTserial.println("Failed to read"); //Make sure no error in readings
        return;}

    for (int j=0; j<14; j++){ //populates Res avg arrays
      digitalWrite(8, LOW);
      digitalWrite(6, LOW);
      digitalWrite(7, LOW);

      delay(500);
      float posttempTGS = getAvgTGS(t,h);
      float posttempWSP = getAvgWSP(t,h);
      digitalWrite(8, LOW);
      digitalWrite(6, LOW);
      digitalWrite(7, HIGH);
      delay(500);}

    sendValues(t,h); //Sends values to BT radio.

    for(int j=0; j<9; j++){ //Wipe clean avg sensor RES from arrays.
      TGS_Array[j]=0;
      WSP_Array[j]=0;}
      PostSet = 0;
      return;}



    digitalWrite(7, LOW); //if Button not pressed LED's off.
    digitalWrite(8, LOW);
    digitalWrite(6, LOW);

}

/////////////////////////////////////////////////////////////////////////////////////////////////

//Sends back TGS resistance.
float getResTGS(float a, float b){

  float scale_tgs = 0.44546-(0.010457*a)-(0.002953*b);
  scale_tgs = pow(10, scale_tgs);
  float TGS = analogRead(A0);
  float tgs_res = ((1023/TGS)-1)*TGS_RL;

  float unscaled_tgs = tgs_res; //Res value without scaling for temp/hum
```

```
//Group 13 Senior Design
//Diabetic Breathalyzer Hardware Code
//Jonathan Brown, Edert Geffrard, Christine Sleppy, Noah Spenser


  tgs_res = (tgs_res/scale_tgs); //Scaled value of resistance.

  return tgs_res;
}

//Sends back WSP resistance.
float getResWSP(float c, float d){

  float scale_wsp = 0.28299-(0.004916*c)-(0.002493*d); //Calculate scaling factor using temp/hum
  scale_wsp = pow(10, scale_wsp);
  float WSP = analogRead(A1);
  float wsp_res = ((1023/WSP)-1)*WSP_RL;

  float unscaled_wsp = wsp_res;

  wsp_res = (wsp_res/scale_wsp);

  return wsp_res;
}

//Sends back averaged TGS resistance.
float getAvgTGS(float y, float z){
  int avgtemp1 =0;
  float sum1 = 0;
  float tgs_avg = 0;
  for (int i=9; i>0; i--){      //Shift out oldest value, put in newest in array
      TGS_Array[i]=TGS_Array[i-1];
      }

      TGS_Array[0]= getResTGS(y,z);

      for(int m=0; m<9; m++){    //sum up each array to get average
        if(TGS_Array[m] != 0){
        sum1 = sum1 + TGS_Array[m];


        avgtemp1 = avgtemp1 + 1;
        }
      }

      tgs_avg= sum1/avgtemp1;

return tgs_avg;
}

//Sends back averaged WSP resistance.
float getAvgWSP(float r, float s){
  int avgtemp2 =0;
  float sum2 = 0;
  float wsp_avg = 0;
    for (int k=9; k>0; k--){      //Same shift for wsp array
       WSP_Array[k]=WSP_Array[k-1];
       }

       WSP_Array[0]= getResWSP(r,s);

      for(int n=0; n<9; n++){
        if(WSP_Array[n] != 0){
        sum2 = sum2 + WSP_Array[n];
        avgtemp2 = avgtemp2 +1;}
      }

      wsp_avg= sum2/avgtemp2;

return wsp_avg;
}
```

```
//Group 13 Senior Design
//Diabetic Breathalyzer Hardware Code
//Jonathan Brown, Edert Geffrard, Christine Sleppy, Noah Spenser


  if(r2>50){
    r2 = r1;}
  BTserial.print(r2);
  BTserial.print("\r\n");

  digitalWrite(7, LOW);
  digitalWrite(8, HIGH);
  digitalWrite(6, LOW);
  delay(2000);

return;
}
```