```
#include <Tone.h> //Library to help with tone
#include <math.h> //Library to perform math operations like e^x

//Sensor Pins
#define SLEP 7 //Left Sensor Echo Pin
#define SLTP 6 //Left Sensor Trigger Pin
#define SREP 4 // Right Sensor Echo Pin
#define SRTP 5 //Right Sensor Trigger Pin

//Button Pins
#define LBUT 10 //Left Function Switch Button
#define LFOOT 3 //Left Foot Pedal
#define RBUT 11 //Right Function Switch Sensor
#define RFOOT 2 //Right Foot Pedal

//Speaker Defintions
Tone ls; // Left Speaker Object
Tone rs; //Right Speaker Object
#define ls_pin 8
#define rs_pin 12

// define variables
long duration; // variable for the duration of sound wave travel
double d1, d2;
int f1 = 120, f2 = 120;
double dmax = 35;
int Lmode = 0;
int Rmode = 0;


int getData(int mode, double d,double dmax, double qmin, double
qmax)
    {
     if (d>dmax ||d <3) //If an object is not close enough or too
close, it will not play the speaker.
        {
          return 0;
```

```
        }
    switch (mode)
    {
      case 0:
        return abs(linearData(d,dmax, qmin, qmax));
        break;

      case 1:
        return abs(logData(d,dmax, qmin, qmax));
        break;

      case 2:
        return abs(expData(d,dmax, qmin, qmax));
        break;

      default:
        return 0;
      }
    }

int linearData(double d,double dmax, double qmin, double qmax)
    {
      return ((qmax-qmin)/(dmax-3)) * (d-3)+qmin;
    }

int logData(double d,double dmax, double qmin, double qmax)
    {
      return ((qmax-qmin)/log10(dmax-2))*log10(d-2)+qmin;
    }

int expData(double d,double dmax, double qmin, double qmax)
    {
      return qmin*exp((d-3)/((dmax-3)/log((qmax/qmin)))) ;
    }

double getDistance(int tpin, int epin) //Finds distance from the
sensor, does not work for very small distances
```

```
    {
    digitalWrite(tpin, LOW);
    delayMicroseconds(2);
    // Sets the trigPin HIGH (ACTIVE) for 10 microseconds
    digitalWrite(tpin, HIGH);
    delayMicroseconds(10);
    digitalWrite(tpin, LOW);
    duration = pulseIn(epin, HIGH);
    return duration * 0.034 / 2;
    }

void play_tone(Tone x, int f)
{
  return f<=0 ? x.stop() : x.play(f);
}

void TwoSpeaker()
    {
    if(digitalRead(LBUT) == HIGH) //Cycles through modes on button
    {
      Lmode = Lmode++ >1 ? 0: Lmode++;
      }

    if(digitalRead(RBUT) == HIGH) //Cycles through modes on button
    {
      Rmode = Rmode++ >1 ? 0: Rmode++;
      }

    if(digitalRead(LFOOT) == LOW)//If High, I have pressed the
pedal and do not want to update the tone
    {
      d1 = getDistance(SLTP,SLEP);
      f1 = d1 <2000 ? getData(Lmode,d1, dmax, 120,5000) : f1;
      }

    if(digitalRead(RFOOT) == LOW) //If High, I have pressed the
pedal and do not want to update the tone
```

```
    {
      d2 = getDistance(SRTP,SREP);
      f2 = d2 < 2000 ? getData(Rmode,d2, dmax, 120,5000) : f2;
      }
    play_tone(ls,f1);
    play_tone(rs,f2);
    }

void setup() {
  //Sensor Pin Setup
  pinMode(SLTP, OUTPUT);
  pinMode(SLEP, INPUT);
  pinMode(SRTP, OUTPUT);
  pinMode(SREP, INPUT);

  pinMode(LBUT, INPUT);
  pinMode(LFOOT, INPUT);
  pinMode(RBUT, INPUT);
  pinMode(RFOOT, INPUT);

  //Speaker Setup
  ls.begin(ls_pin);
  rs.begin(rs_pin);
  //Serial.begin(9600);

}

void loop()
{
  TwoSpeaker();
  //delay(50);
    }
```