```
/*

Project Title: Ice Cube
David Oppenheim
Final Project
DIGF 6B02, Creative Computation
OCAD
Filename: IceCube_Heater_TalkingTo_v1_8_3_Wking
Last modified on Monday, November 28, 2011

Attribution: This code borrows from the following: Jim Ruxton for audio level code
and to get Processing talking to Arudino,
http://www.sundh.com/blog/2011/05/get-processing-and-arduino-to-talk/

Explanation:

Starts sketch by turning Transistor (heater) ON and Red LED on;
Turns OFF transistor (heater) + Red LED off and colours square blue (and turn blue
LED on)  after audio level has been above a certain threshold level for ___ seconds;
Turns transistor and Red LED back on again (and colours square red) if audio level
falls back below threshold for ____ seconds

STATUS: WORKING

Wiring: Red LED (+) to 330 Ohm resistor and pin 11, (-) to common ground;
Blue LED (+) to 330 Ohm resistor and pin 10, (-) to common ground;
Transistor (TIP120) base is connected to 1K resistor and to pin 5; Emitter of TIP120
is connected to ground of power supply AND to ground of Arduino; 5 ohm resistor:
one side connected to collector of TIP120, one side is connected to power supply;

*/

// LED & Transistor

int ledPinRed = 11; // set pin 11 as Red LED
int ledPinBlue = 10; // set pin 10 as Blue LED
int transistorPin = 5;    // Transistor (Base) connected to 1K resistor, connected to
digital pin (PWM) 5

// USER TALKING VARIABLES

boolean talking = false;
boolean firstTalk = false;
boolean firstNotTalk = false;

int startTalk = 0;
```

```
int notTalking = 0;
int talkedEnough = 0;
int timeTalking = 0;
int timeNotTalking = 0;

int talkThreshold = 12000;
int notTalkingAfterTalkingThreshold = 20000;

// PROCESSING TALKING TO ARDUINO AND MEASURING SOUND LEVELS

import processing.serial.*;
import cc.arduino.*;
import ddf.minim.*;
Arduino arduino;
int scaledAudioInput;
Minim minim;
AudioInput in;
float sum;
float averageAudioInput;


void setup() {

minim = new Minim(this);
minim.debugOn();
// arduino = new Arduino(this, Arduino.list()[0]);
// get a line in from Minim, default bit depth is 16
in = minim.getLineIn(Minim.STEREO,100000); // last variable is size of buffer (was
original (in versions up to 1_8, 800)

// Code TO GET PROCESSING to talk to Arduino re: LED

arduino = new Arduino(this, Arduino.list()[0], 57600);

// Set pinModes

arduino.pinMode(ledPinRed, Arduino.OUTPUT);
arduino.pinMode(ledPinBlue, Arduino.OUTPUT);
arduino.pinMode(transistorPin, Arduino.OUTPUT);

// DRAW RED SQUARE AND TURN RED LED AND HEATER ON

  // Draw red square
  size(1440,900);
  background(255,0,0);
```

```
// Turn Red LED on

  arduino.digitalWrite(ledPinRed, Arduino.HIGH); //...set the ledPin (Red LED) to
HIGH;

// Turn Transistor (heater) on

  arduino.digitalWrite(transistorPin, Arduino.HIGH);

 }


void draw() {


// AUDIO LEVEL CODE
   sum=0;
   for(int i = 0; i < in.bufferSize() - 1; i++) {
   line(i, 150 + in.right.get(i)*200, i+1, 150 + in.right.get(i+1)*200);
    sum=sum+abs(in.right.get(i));
    }
// gets average value of all samples
     averageAudioInput= sum/in.bufferSize();

 // scale this higher and clamp no higher than 255, also turn to integer which
anaologWrite requires
     scaledAudioInput=int(constrain( averageAudioInput*2555,0,255));

    println("Audio input..." + scaledAudioInput);


// AUDIO LEVEL FOR TIME TALKING

if(scaledAudioInput > 2) {
   talking = true;
  }


// AUDIO LEVEL FOR TIME NOT TALKING

if(scaledAudioInput < 2) {
  talking = false;
  firstTalk = false;
  }
```

```
// START TIMER FOR TALKING

if((talking) && firstTalk == false) {
  firstTalk = true;
  startTalk = millis();
  println("Starting to talk at..." + startTalk/1000 + "seconds");
  }

// START TIMER FOR NOT TALKING

if((!talking) && firstNotTalk == false) {
  firstNotTalk = true;
  notTalking = millis();
  println("Not talking at..." + notTalking/1000 + "seconds");
  }


// TIME TALKING CALCULATION

if (talking) {

  timeTalking = millis() - startTalk;

// Has x seconds of talking passed? (x is set in global variables (talkThreshold)

  if (timeTalking > talkThreshold) {
    // Change square colour to blue
    background(5,9,106);

    // Turn off red LED and turn on blue LED
    arduino.analogWrite(ledPinRed, Arduino.LOW);
    arduino.analogWrite(ledPinBlue, Arduino.HIGH);

    // Turn Transistor (heater) OFF

    arduino.digitalWrite(transistorPin, Arduino.LOW);

    println("Talked enough" + millis()/1000 + "seconds");
    talkedEnough = millis();

  }
}

// TIME NOT TALKING AFTER TALKING CALCULATION
```

```
if(!talking) {

  timeNotTalking = millis() - talkedEnough;

// Has y seconds of NOT talking passed? (y is set in global variables
(notTalkingThreshold)

  if (timeNotTalking > notTalkingAfterTalkingThreshold) {

    // Change square color to red
    background(255,0,0);

    // Turn Red LED on, Blue LED off
    arduino.analogWrite(ledPinRed, Arduino.HIGH);
    arduino.analogWrite(ledPinBlue, Arduino.LOW);

    // Turn Transistor (heater) on
    arduino.digitalWrite(transistorPin, Arduino.HIGH);

    println ("Not talking enough at" + millis()/1000 + "seconds");
   }
   }
  }
```