Chanchal Bajoria

Making Interactive ART (CVA-0051-1)

Project Report

Self-watering plant and Music-reactive LED balance lamp

- **Concept**

A) Self-watering plant

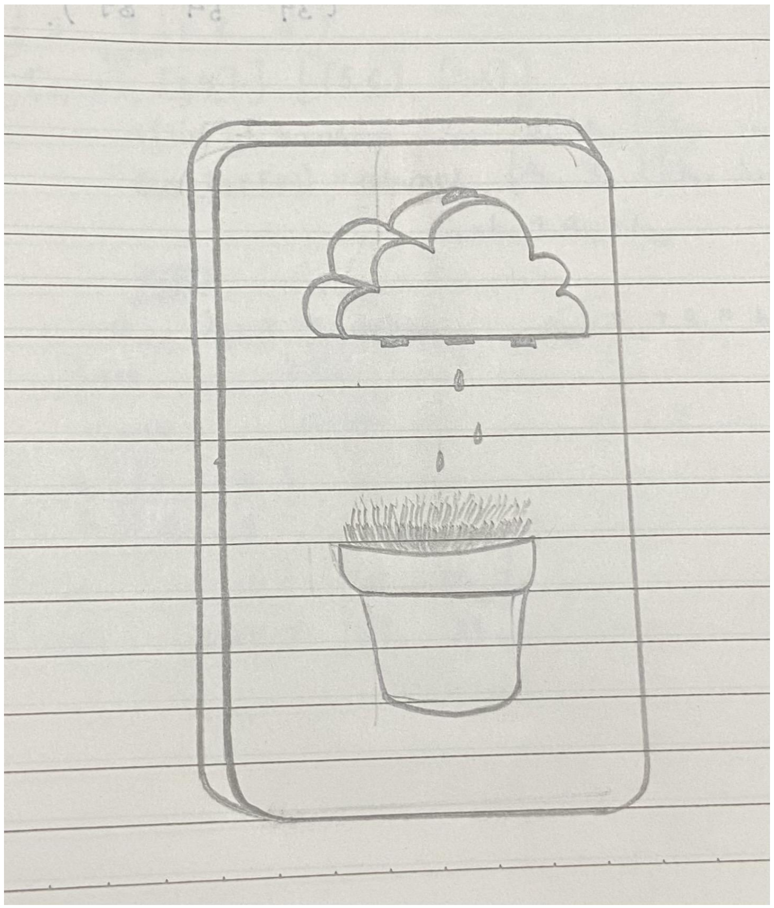
The idea of the self-watering plant was a result of Eryn and my hunt for a live plant for our shared room. However, knowing how busy college students are, we did not want to risk the plant dying due to lack of watering. Thus, we thought of making a self-watering plant. We got Bhavye and Aniruddh on our team and with their help, we discussed and developed several ideas and designs to make this happen. We looked for ways to accomplish this and decided on using a moisture sensor and programming an arduino to draw water from a water pump when the moisture levels went below the optimum. Regarding the design, we settled on a terrarium and an attached cloud with sprinklers to give the effect of rain.

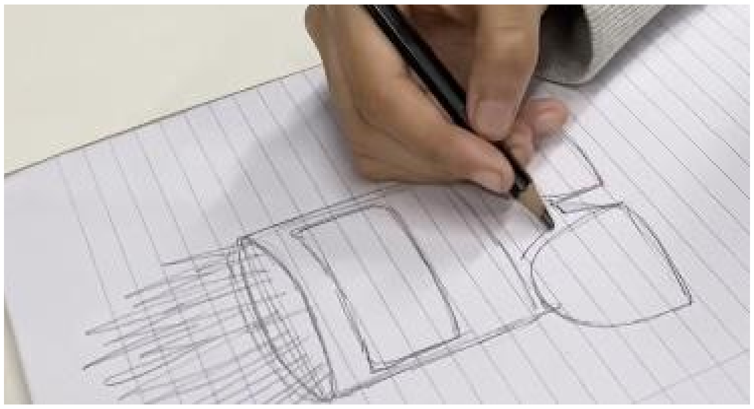
B) LED music-reactive balance lamp

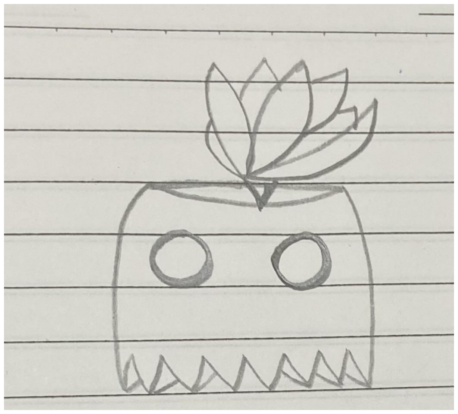
The LED music-reactive balance lamp was Soham's individual project idea. When he suggested this to us, it really intrigued the rest of us and we decided to make both projects fall under the same team. Soham's idea involved an LED lamp with balance magnets as a switch and the LED pixels would react and respond to changes in amplitude of the sound. We achieved this using an Arduino.

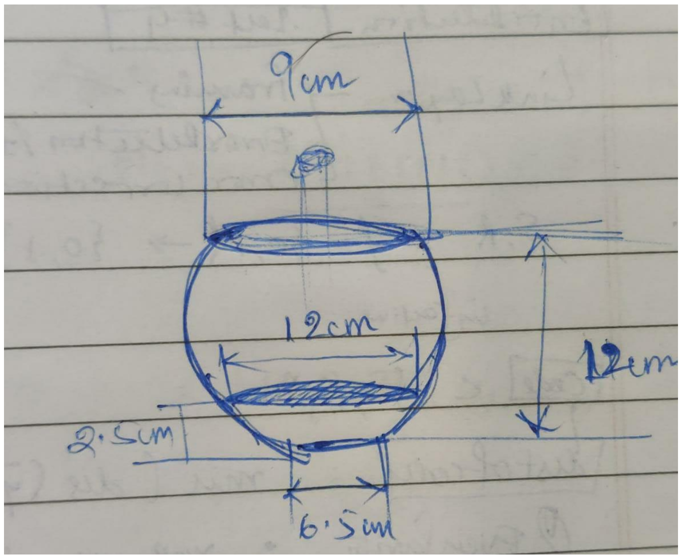
- **Sketch**

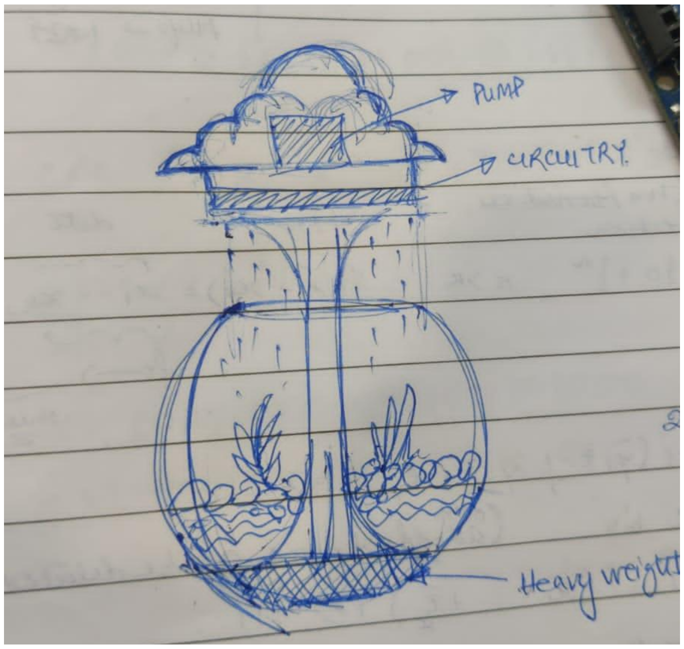
A) Self-watering plant



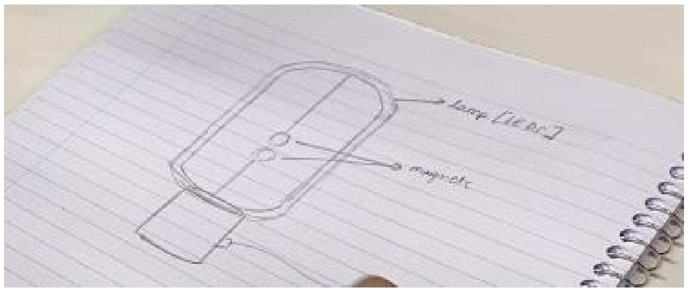


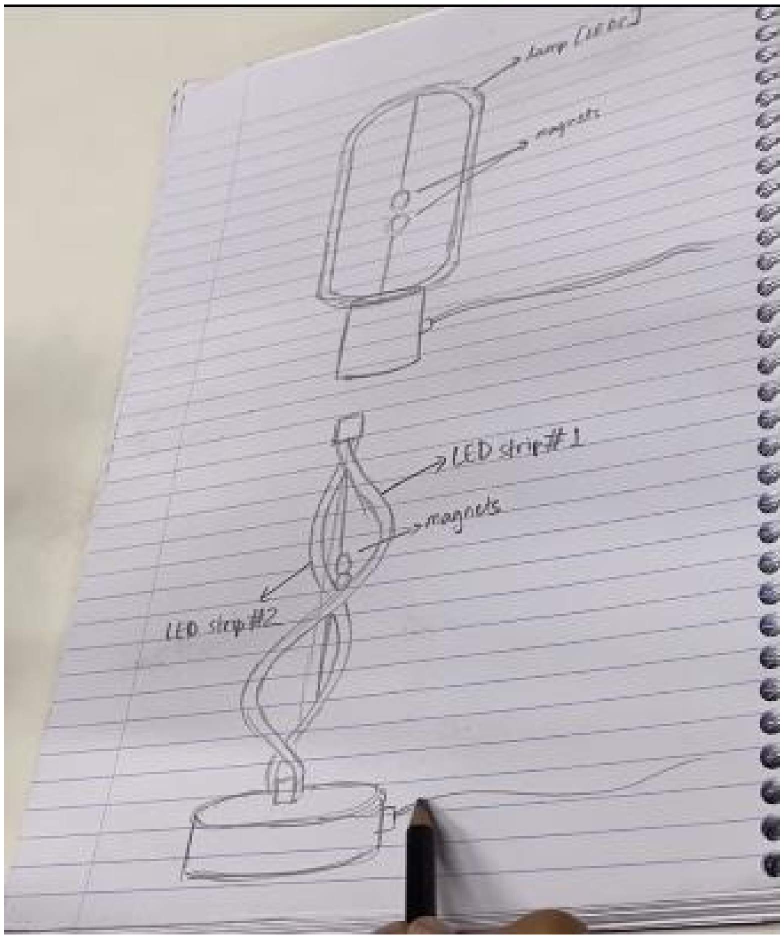






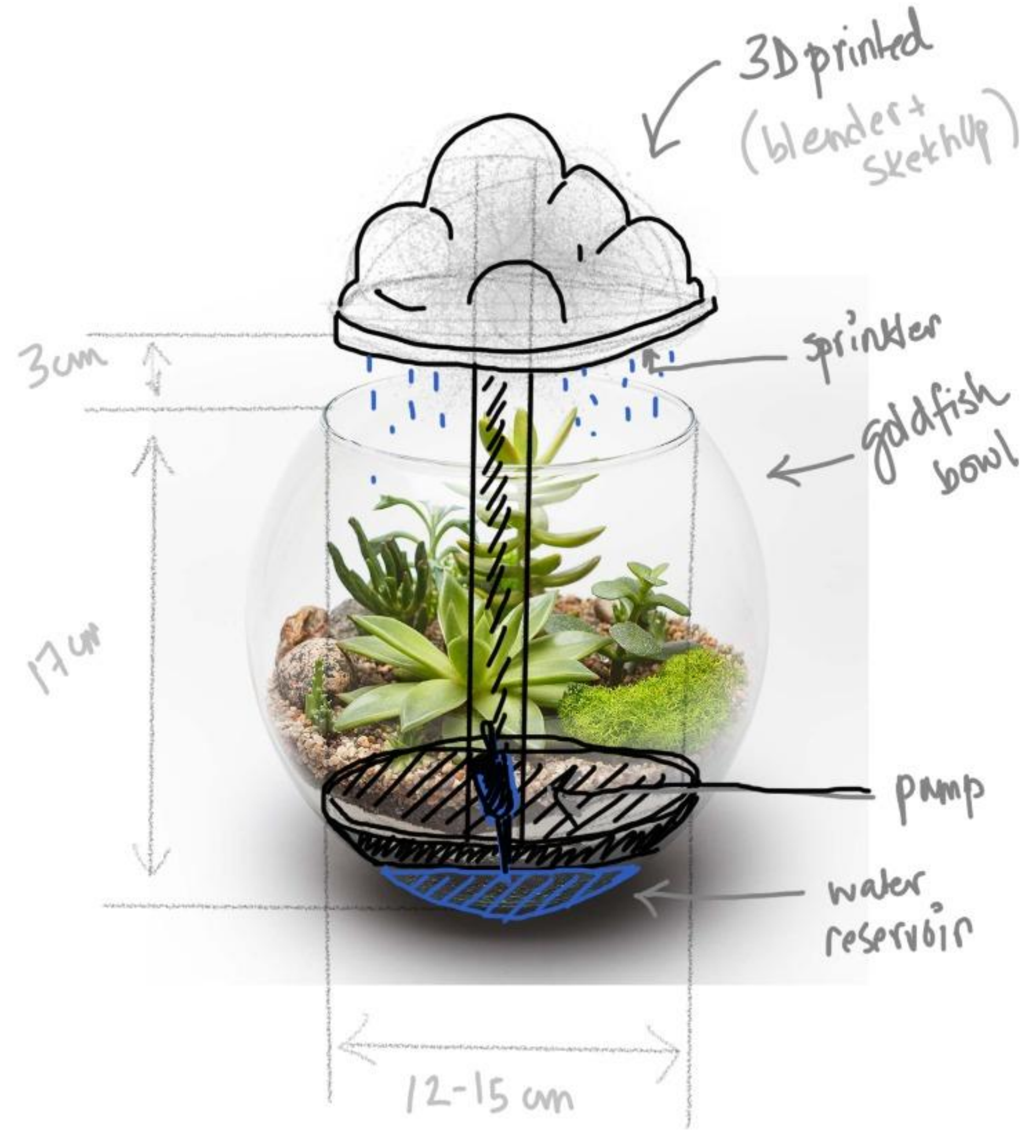
B) LED music-reactive balance lamp



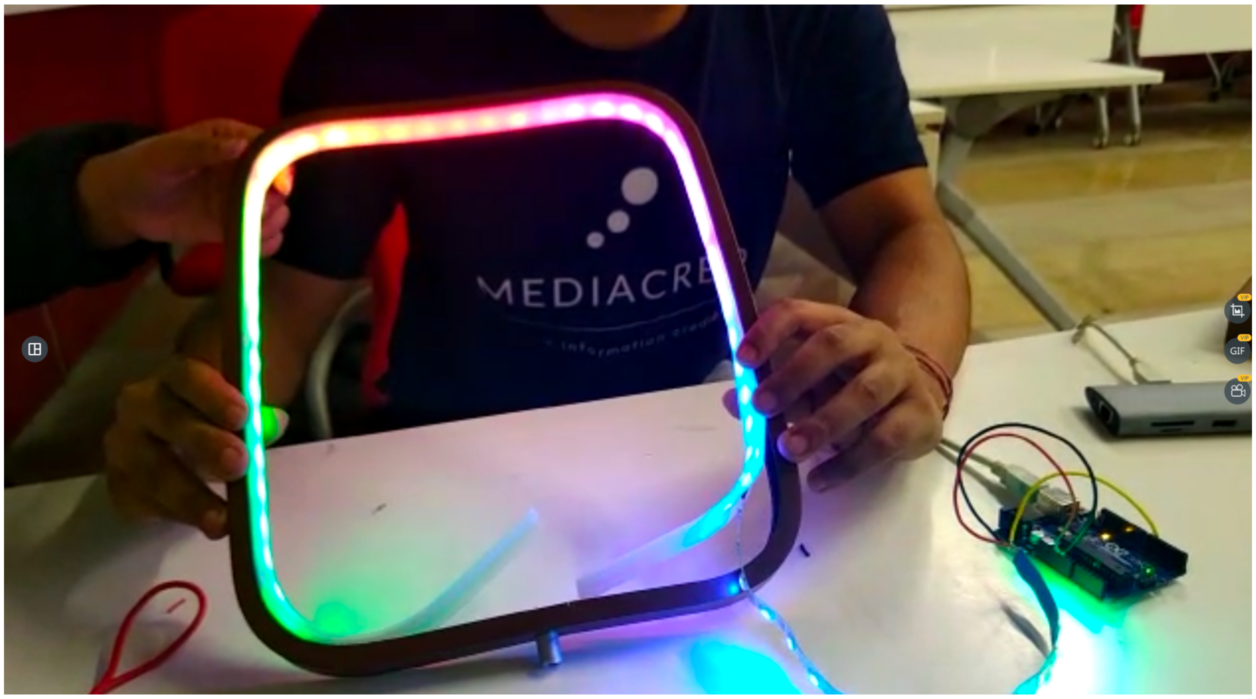

lamp [LED]
magnets
LED strip#1
magnets
LED strip#2

- **Design**

A) Self-watering plant



- **Circuit Design with Prototype**



Also see attached videos in the email.

- **Code**

A) Self-watering plant

1) pot circuit:

```
/*

  Blink


  Turns an LED on for one second, then off for one second,
repeatedly.


  Most Arduinos have an on-board LED you can control. On the
UNO, MEGA and ZERO

  it is attached to digital pin 13, on MKR1000 on pin 6.
LED_BUILTIN is set to

  the correct LED pin independent of which board is used.

  If you want to know what pin the on-board LED is connected to
on your Arduino

  model, check the Technical Specs of your board at:

  https://www.arduino.cc/en/Main/Products


  modified 8 May 2014

  by Scott Fitzgerald

  modified 2 Sep 2016

  by Arturo Guadalupi

  modified 8 Sep 2016

  by Colby Newman


  This example code is in the public domain.


  https://www.arduino.cc/en/Tutorial/BuiltInExamples/Blink
```

```
*/


// the setup function runs once when you press reset or power
the board

void setup() {

  // initialize digital pin LED_BUILTIN as an output.

  pinMode(12, OUTPUT);

}


// the loop function runs over and over again forever

void loop() {

  digitalWrite(12, HIGH);   // turn the LED on (HIGH is the
voltage level)

  delay(5000);                       // wait for a second

  digitalWrite(12, LOW);    // turn the LED off by making the
voltage LOW

  delay(5000);                       // wait for a second

}
```

2) moisture sensor code:

```
const int AirValue = 620;   //you need to replace this value with
Value_1

const int WaterValue = 310;  //you need to replace this value with
Value_2

int soilMoistureValue = 0;

int soilmoisturepercent=0;

void setup() {
  Serial.begin(9600); // open serial port, set the baud rate to 9600 
bps

}

void loop() {

soilMoistureValue = analogRead(A0);  //put Sensor insert into soil

Serial.println(soilMoistureValue);

soilmoisturepercent = map(soilMoistureValue, AirValue, WaterValue, 0, 
100);

if(soilmoisturepercent >= 100)

{

  Serial.println("100 %");

}

else if(soilmoisturepercent <=0)

{

  Serial.println("0 %");

}

else if(soilmoisturepercent >0 && soilmoisturepercent < 100)

{

  Serial.print(soilmoisturepercent);

  Serial.println("%");


}

  delay(250);

}
```

3) moisture sensor testing code:

```
int sensor_pin = A0;

int output_value ;
```

```
void setup() {
   Serial.begin(9600);
   Serial.println("Reading From the Sensor ...");
   delay(2000);
   }

void loop() {
   output_value= analogRead(sensor_pin);
   Serial.print(output_value);
   Serial.println("\n");
//   output_value = map(output_value,550,0,0,100);
//   Serial.print("Moisture : ");
//   Serial.print(output_value);
//   Serial.println("%");
//   delay(1000);
}
```

B) LED music-reactive balance lamp

1) LED code:

```
#include <Adafruit_NeoPixel.h>
#ifdef _AVR_
  #include <avr/power.h>
#endif

#define PIN 6
```

```
// Parameter 1 = number of pixels in strip

// Parameter 2 = Arduino pin number (most are valid)

// Parameter 3 = pixel type flags, add together as needed:

//   NEO_KHZ800  800 KHz bitstream (most NeoPixel products w/WS2812
LEDs)

//   NEO_KHZ400  400 KHz (classic 'v1' (not v2) FLORA pixels, WS2811
drivers)

//   NEO_GRB     Pixels are wired for GRB bitstream (most NeoPixel
products)

//   NEO_RGB     Pixels are wired for RGB bitstream (v1 FLORA pixels,
not v2)

//   NEO_RGBW    Pixels are wired for RGBW bitstream (NeoPixel RGBW
products)

Adafruit_NeoPixel strip = Adafruit_NeoPixel(60, PIN, NEO_GRB +
NEO_KHZ800);


// IMPORTANT: To reduce NeoPixel burnout risk, add 1000 uF capacitor
across

// pixel power leads, add 300 - 500 Ohm resistor on first pixel's
data input

// and minimize distance between Arduino and first pixel.  Avoid
connecting

// on a live circuit...if you must, connect GND first.


void setup() {

  // This is for Trinket 5V 16MHz, you can remove these three lines
if you are not using a Trinket

  #if defined (_AVR_ATtiny85_)

    if (F_CPU == 16000000) clock_prescale_set(clock_div_1);

  #endif

  // End of trinket special code
  strip.begin();
  strip.setBrightness(50);
  strip.show(); // Initialize all pixels to 'off'
  Serial.begin(9600);
}

int i = 0;

void loop() {

  // Some example procedures showing how to display to the pixels:
//  colorWipe(strip.Color(200, 0, 0), 50); // Red
//  colorWipe(strip.Color(0, 10, 0), 50); // Green
//  colorWipe(strip.Color(0, 0, 200), 50); // Blue
  if (i % 2 == 0) {
    colorWipe(strip.Color(200, 0, 0), 50); // Red
    colorWipe(strip.Color(200, 200, 200), 50); // Green
    colorWipe(strip.Color(0, 0, 200), 50);
  } else {
    colorWipe(strip.Color(10, 0, 0), 50); // Red
    colorWipe(strip.Color(10, 10, 10), 50); // Green
    colorWipe(strip.Color(0, 0, 10), 50);
  }

  i++;
//colorWipe(strip.Color(0, 0, 0, 255), 50); // White RGBW
```

```
  // Send a theater pixel chase in...
//  theaterChase(strip.Color(127, 127, 127), 50); // White
//  theaterChase(strip.Color(127, 0, 0), 50); // Red
//  theaterChase(strip.Color(0, 0, 127), 50); // Blue
//
//  rainbow(20);
//  rainbowCycle(20);
//  theaterChaseRainbow(50);
}

// Fill the dots one after the other with a color
void colorWipe(uint32_t c, uint8_t wait) {
  for(uint16_t i=0; i<strip.numPixels(); i++) {
    strip.setPixelColor(i, c);
    strip.show();
    delay(wait);
  }
}

void rainbow(uint8_t wait) {
  uint16_t i, j;

  for(j=0; j<256; j++) {
    for(i=0; i<strip.numPixels(); i++) {
      strip.setPixelColor(i, Wheel((i+j) & 255));
    }
    strip.show();
    delay(wait);

  }

}


// Slightly different, this makes the rainbow equally distributed
throughout

void rainbowCycle(uint8_t wait) {

  uint16_t i, j;


  for(j=0; j<256*5; j++) { // 5 cycles of all colors on wheel

    for(i=0; i< strip.numPixels(); i++) {

      strip.setPixelColor(i, Wheel(((i * 256 / strip.numPixels()) +
j) & 255));

    }

    strip.show();

    delay(wait);

  }

}


//Theatre-style crawling lights.

void theaterChase(uint32_t c, uint8_t wait) {

  for (int j=0; j<10; j++) {  //do 10 cycles of chasing

    for (int q=0; q < 3; q++) {

      for (uint16_t i=0; i < strip.numPixels(); i=i+3) {

        strip.setPixelColor(i+q, c);    //turn every third pixel on

      }

      strip.show();
      delay(wait);


      for (uint16_t i=0; i < strip.numPixels(); i=i+3) {

        strip.setPixelColor(i+q, 0);        //turn every third pixel
off

      }

    }

  }

}


//Theatre-style crawling lights with rainbow effect

void theaterChaseRainbow(uint8_t wait) {

  for (int j=0; j < 256; j++) {     // cycle all 256 colors in the
wheel

    for (int q=0; q < 3; q++) {

      for (uint16_t i=0; i < strip.numPixels(); i=i+3) {

strip.setPixelColor(i+q, Wheel( (i+j) % 255));    //turn every third
pixel on

      }

      strip.show();


      delay(wait);


      for (uint16_t i=0; i < strip.numPixels(); i=i+3) {

        strip.setPixelColor(i+q, 0);        //turn every third pixel
off

      }

    }

  }
```

```
}

// Input a value 0 to 255 to get a color value.
// The colours are a transition r - g - b - back to r.
uint32_t Wheel(byte WheelPos) {
  WheelPos = 255 - WheelPos;
  if(WheelPos < 85) {
    return strip.Color(255 - WheelPos * 3, 0, WheelPos * 3);
  }
  if(WheelPos < 170) {
    WheelPos -= 85;
    return strip.Color(0, WheelPos * 3, 255 - WheelPos * 3);
  }
  WheelPos -= 170;
  return strip.Color(WheelPos * 3, 255 - WheelPos * 3, 0);
}
```

2) sound sensor code:

```
void setup() {
Serial.begin(9600); // setup serial
}
void loop() {
Serial.println(analogRead(A0));
delay(100);
}
```