```
 /*
SiyuanZan 12/10/2020
light project for digital_fabrication

Sinfade_Reference from Tome Igoe light and Interaction Class
  https://github.com/tigoe/LightProjects/blob/master/FadeCurves/SineFade/
SineFade.ino



SoundSensor pin A0
DMX485 pin 3
*/

#include <DmxSimple.h>
const int period =5 ;
unsigned long previousMillisSound = 0;
unsigned long previousMillisSound2 = 0;
unsigned long previousMillislight = 0;
unsigned long previousMillislight2 = 0;

unsigned long timeStart = 0;

int brightness = 255;
int direction = 1;
byte levelTable[256];
unsigned long TimeTotal = 2550*4;



bool ifRotate = false;
bool value_above_threshold=false;
int value_input;
int pre_value_input;
int value_output = 0;
int threshold=120;
int data_pointer;
int sum = 0;
int unit = 10;
int data_filter[10] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
int sensorValue;
int a = 0;

void setup() {

  Serial.begin(9600);
  fillLevelTable();
```

```
  DmxSimple.usePin(3);


  DmxSimple.maxChannel(18);
  DmxSimple.write(3, 240 );
  DmxSimple.write(12, 240 );
  DmxSimple.write(2, 125);
  DmxSimple.write(11, 125);

  for (int i = 4; i <= 7; i++) {
    DmxSimple.write(i, 0);
    DmxSimple.write(i + 9, 0);
  }

}

void loop() {
  //
  //
  value_input = analogRead(A3);
  unsigned long currentMillislight = millis();


    int value_input2;
   pre_value_input = a;
  if (data_pointer == NULL) {
    data_pointer = 0;
  }
  data_filter[data_pointer] = value_input;
  data_pointer++;
  if (data_pointer == unit)data_pointer = 0;
  sum = 0;
  for (int i = 0; i < unit; i++) {
    sum += data_filter[0];
  }
  value_output = sum / unit;
  //Serial.println(value_output);


  if ( millis() - previousMillisSound >= 100) {
    previousMillisSound =millis() ;
    a = value_output;

    ////a=int(random(50,200));
  }

 if(a>threshold){
```

```
value_above_threshold=true;
 DmxSimple.write(3, 230 );
      DmxSimple.write(12, 230 );
   delay(300);
}else{
value_above_threshold=false;

      DmxSimple.write(3, 255 );
     DmxSimple.write(12, 255 );
}

if(value_above_threshold){
  a=255;
}else{
  a=0;
}
 Serial.println(value_output);









  if ((millis() - timeStart) % TimeTotal < period * 255) {
    if (currentMillislight - previousMillislight2 >= period) {
      previousMillislight2 = currentMillislight;
      if ((brightness >= 255) || (brightness <= 0)) {

        direction = -1 * direction;
      }
      brightness += direction;
      DmxSimple.write(4,12+levelTable[brightness]*18/255);
      DmxSimple.write(5,11+levelTable[brightness]*81/255);
      DmxSimple.write(6,130-levelTable[brightness]*21/255);
//      DmxSimple.write(7, 0);
      DmxSimple.write(13, 200-levelTable[brightness]*8/255);
      DmxSimple.write(14, 16+levelTable[brightness]*72/255);
      DmxSimple.write(15, 138-levelTable[brightness]*40/255);
//      DmxSimple.write(16, 0);

    }

  }

  if ((millis() - timeStart) % TimeTotal >= period * 510 && (millis() - timeStart) %
TimeTotal < period * 510*2) {
```

```
    if (millis() - previousMillislight2 >= period) {
      previousMillislight2 = millis();
      if ((brightness >= 255) || (brightness <= 0)) {

        direction = -1 * direction;
      }
      brightness += direction;

      DmxSimple.write(4, 30+levelTable[brightness]*3/255);
      DmxSimple.write(5, 92+levelTable[brightness]*73/255);
      DmxSimple.write(6, 130+levelTable[brightness]*14/255);
//      DmxSimple.write(7, 0);
      DmxSimple.write(13,235+levelTable[brightness]*3/255);
      DmxSimple.write(14,88+levelTable[brightness]*46/255 );
      DmxSimple.write(15,98-levelTable[brightness]*18/255);
//      DmxSimple.write(16, 0);

//
//      //DmxSimple.write(5, levelTable[brightness]);
//      //DmxSimple.write(15,levelTable[brightness]);
    }
  }

    if ((millis() - timeStart) % TimeTotal >= period * 510*2 && (millis() - timeStart) %
TimeTotal < period * 510*3) {

    if (millis() - previousMillislight2 >= period) {
      previousMillislight2 = millis();
      if ((brightness >= 255) || (brightness <= 0)) {

        direction = -1 * direction;
      }
      brightness += direction;

      DmxSimple.write(4, 33-levelTable[brightness]*6/255);
      DmxSimple.write(5, 165+levelTable[brightness]*77/255);
      DmxSimple.write(6, 144-levelTable[brightness]*10/255);
//      DmxSimple.write(7, 0);
      DmxSimple.write(13, 238+levelTable[brightness]*5/255);
      DmxSimple.write(14, 134+levelTable[brightness]*94/255);
      DmxSimple.write(15, 80-levelTable[brightness]*69/255);
//      DmxSimple.write(16, 0);

    }
  }
```

```
    if ((millis() - timeStart) % TimeTotal >= period * 510*3 && (millis() - timeStart) %
TimeTotal < period * 510*4) {

    if (millis() - previousMillislight2 >= period) {
      previousMillislight2 = millis();
      if ((brightness >= 255) || (brightness <= 0)) {

        direction = -1 * direction;
      }
      brightness += direction;

      DmxSimple.write(4, 24+levelTable[brightness]*67/255);
      DmxSimple.write(5, 242-levelTable[brightness]*14/255);
      DmxSimple.write(6, 130+levelTable[brightness]*13/255);
//      DmxSimple.write(7, 0);
      DmxSimple.write(13, 243-levelTable[brightness]*41/255);
      DmxSimple.write(14, 228+levelTable[brightness]*12/255);
      DmxSimple.write(15, 11+levelTable[brightness]*144/255);

    }
  }


}
void fillLevelTable() {
  // set the range of values:
  float maxValue = 255;

  // iterate over the array and calculate the right value for it:
  for (int l = 0; l <= maxValue; l++) {

    // map input to a 0-179 range:
    float angle = map(l, 0, maxValue, 0, 179);
    //here it all is in one line:
    float lightLevel = (sin((angle * PI / 180) + PI / 2) + 1) * 127.5;
    levelTable[l] = lightLevel;
  }
}
```