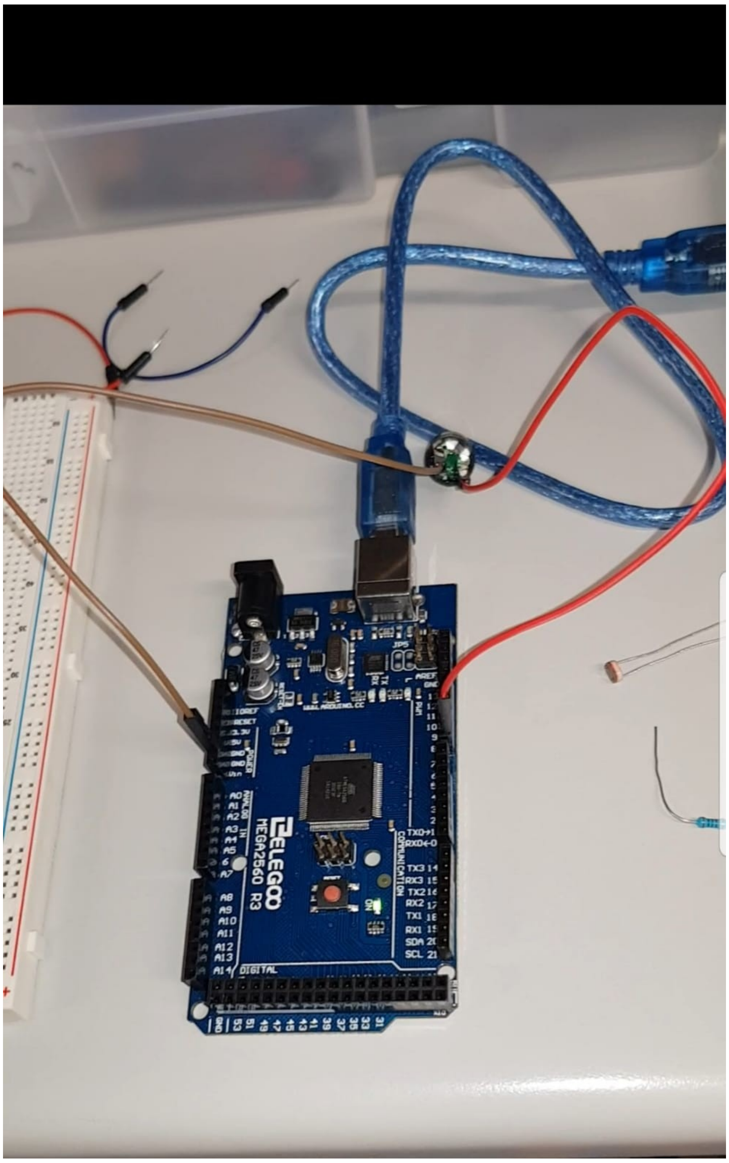
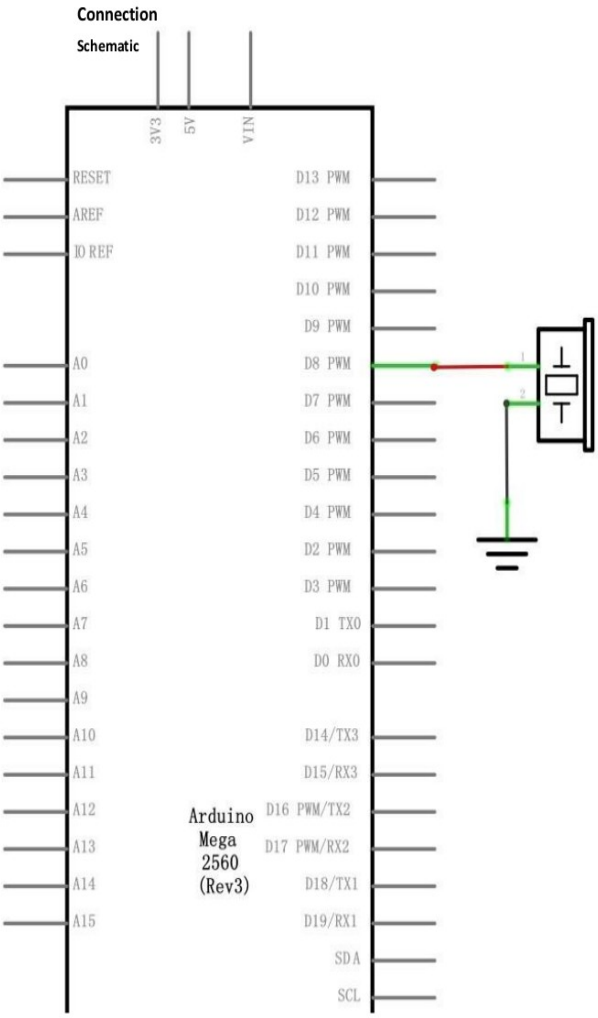
Práctica Arduino: José Santos Salvador

1.- Zumbador pasivo

Con el zumbador pasivo y tras seguir el esquema del circuito monté esta parte de la práctica. Sabiendo que notas tenemos y buscando la canción original cambie el código para que sonase una canción. El código se puede buscar ya escrito en plataformas como github.





Realicé dos canciones diferentes, con sus dos códigos diferentes
https://drive.google.com/open?id=1P25OvBtMYIwk01h6BTZ7fbZwkGBAlODD
https://drive.google.com/open?id=1BvFeubjEFYRxklmeo90bh9rpLISQ3x26

Primer código:

```
 #include "pitches.h"

// notes in the melody:
int melody[] = {
  NOTE_C4, NOTE_A4, NOTE_G4, NOTE_F4, NOTE_G4,  NOTE_E4,
  NOTE_C4, NOTE_A4, NOTE_G4, NOTE_F4, NOTE_E4,  NOTE_F4,
  NOTE_C4, NOTE_A4, NOTE_G4, NOTE_F4, NOTE_G4,  NOTE_CS4,
  NOTE_C4, NOTE_A4, NOTE_G4, NOTE_F4, NOTE_CS4, NOTE_C4,
  NOTE_C4, NOTE_A4, NOTE_G4, NOTE_F4, NOTE_DS4, NOTE_CS4,
  NOTE_A3, NOTE_G3, NOTE_G4, NOTE_F4, NOTE_FS4, NOTE_F4,
  NOTE_G4, NOTE_F4, NOTE_G4, NOTE_CS4, NOTE_F4, NOTE_G4,
  NOTE_F4, NOTE_CS4, NOTE_C4
};

// note durations: 4 = quarter note, 8 = eighth note, etc.:
int noteDurations[] = {
                    3, 4, 4, 6, 6, 3,
                        4, 6, 6, 4, 4, 3,
                        2, 6, 6, 4, 6, 6,
                        3, 4, 4, 6, 6, 3,
                        3, 4, 4, 6, 6, 3,
                        4, 6, 6, 4, 4, 3,
                        4, 6, 6, 4, 6, 6,
                        3, 4, 4
                      };

void setup() {
  // iterate over the notes of the melody:
  for (int thisNote = 0; thisNote < 45; thisNote++) {

    // to calculate the note duration, take one second
    // divided by the note type.
    //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
    int noteDuration = 1000/noteDurations[thisNote];
    tone(8, melody[thisNote],noteDuration);

    // to distinguish the notes, set a minimum time between them.
    // the note's duration + 30% seems to work well:
    int pauseBetweenNotes = noteDuration * 1.30;
    delay(pauseBetweenNotes);
    // stop the tone playing:
    noTone(8);
  }
}

void loop() {
  // no need to repeat the melody.
}
```

__Segundo código:__

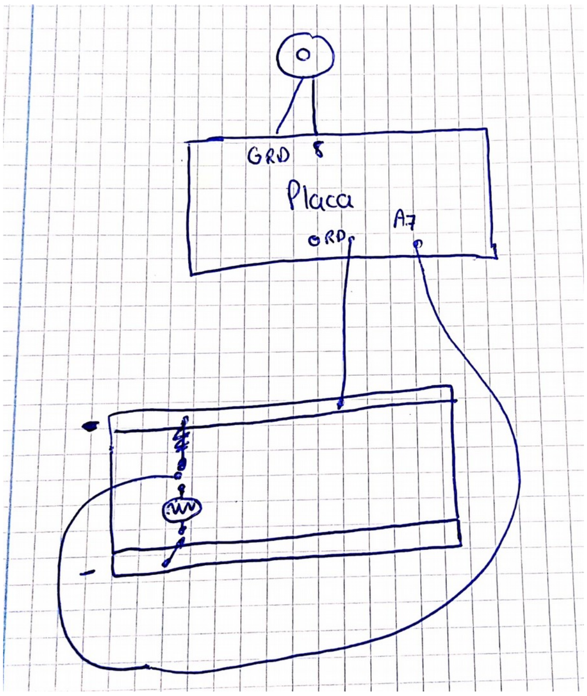
```
#define NOTE_B0  31
#define NOTE_C1  33
#define NOTE_CS1 35
#define NOTE_D1  37
#define NOTE_DS1 39
#define NOTE_E1  41
#define NOTE_F1  44
#define NOTE_FS1 46
#define NOTE_G1  49
#define NOTE_GS1 52
#define NOTE_A1  55
#define NOTE_AS1 58
#define NOTE_B1  62
#define NOTE_C2  65
#define NOTE_CS2 69
#define NOTE_D2  73
#define NOTE_DS2 78
#define NOTE_E2  82
#define NOTE_F2  87
```

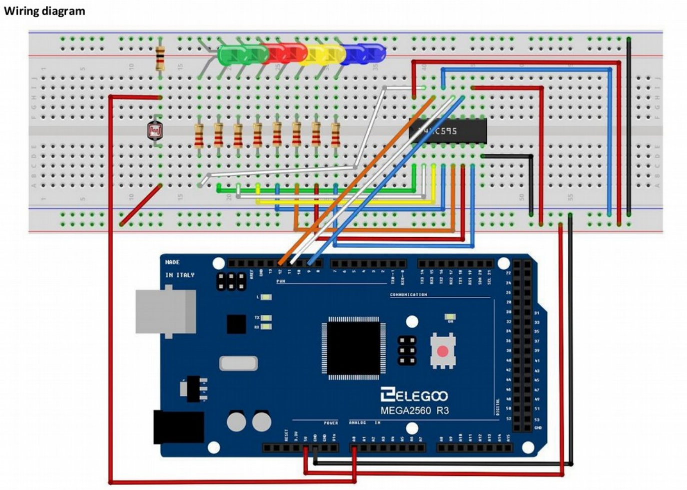
```
#define NOTE_FS2 93
#define NOTE_G2  98
#define NOTE_GS2 104
#define NOTE_A2  110
#define NOTE_AS2 117
#define NOTE_B2  123
#define NOTE_C3  131
#define NOTE_CS3 139
#define NOTE_D3  147
#define NOTE_DS3 156
#define NOTE_E3  165
#define NOTE_F3  175
#define NOTE_FS3 185
#define NOTE_G3  196
#define NOTE_GS3 208
#define NOTE_A3  220
#define NOTE_AS3 233
#define NOTE_B3  247
#define NOTE_C4  262
#define NOTE_CS4 277
#define NOTE_D4  294
#define NOTE_DS4 311
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_FS4 370
#define NOTE_G4  392
#define NOTE_GS4 415
#define NOTE_A4  440
#define NOTE_AS4 466
#define NOTE_B4  494
#define NOTE_C5  523
#define NOTE_CS5 554
#define NOTE_D5  587
#define NOTE_DS5 622
#define NOTE_E5  659
#define NOTE_F5  698
#define NOTE_FS5 740
#define NOTE_G5  784
#define NOTE_GS5 831
#define NOTE_A5  880
#define NOTE_AS5 932
#define NOTE_B5  988
#define NOTE_C6  1047
#define NOTE_CS6 1109
#define NOTE_D6  1175
#define NOTE_DS6 1245
#define NOTE_E6  1319
#define NOTE_F6  1397
#define NOTE_FS6 1480
#define NOTE_G6  1568
#define NOTE_GS6 1661
#define NOTE_A6  1760
#define NOTE_AS6 1865
#define NOTE_B6  1976
#define NOTE_C7  2093
#define NOTE_CS7 2217
#define NOTE_D7  2349
#define NOTE_DS7 2489
#define NOTE_E7  2637
#define NOTE_F7  2794
#define NOTE_FS7 2960
#define NOTE_G7  3136
#define NOTE_GS7 3322
```

```
#define NOTE_A7  3520
#define NOTE_AS7 3729
#define NOTE_B7  3951
#define NOTE_C8  4186
#define NOTE_CS8 4435
#define NOTE_D8  4699
#define NOTE_DS8 4978
#define REST 0

int tempo=88;

// change this to whichever pin you want to use
int buzzer = 11;

// notes of the moledy followed by the duration.
// a 4 means a quarter note, 8 an eighteenth , 16 sixteenth, so on
// !!negative numbers are used to represent dotted notes,
// so -4 means a dotted quarter note, that is, a quarter plus an eighteenth!!
int melody[] = {

  //Based on the arrangement at https://www.flutetunes.com/tunes.php?id=169

  NOTE_AS4,-2,  NOTE_F4,8,  NOTE_F4,8,  NOTE_AS4,8,//1
  NOTE_GS4,16,  NOTE_FS4,16,  NOTE_GS4,-2,
  NOTE_AS4,-2,  NOTE_FS4,8,  NOTE_FS4,8,  NOTE_AS4,8,
  NOTE_A4,16,  NOTE_G4,16,  NOTE_A4,-2,
  REST,1,

  NOTE_AS4,4,  NOTE_F4,-4,  NOTE_AS4,8,  NOTE_AS4,16,  NOTE_C5,16, NOTE_D5,16, NOTE_DS5,16,//7
  NOTE_F5,2,  NOTE_F5,8,  NOTE_F5,8,  NOTE_F5,8,  NOTE_FS5,16, NOTE_GS5,16,
  NOTE_AS5,-2,  NOTE_AS5,8,  NOTE_AS5,8,  NOTE_GS5,8,  NOTE_FS5,16,
  NOTE_GS5,-8,  NOTE_FS5,16,  NOTE_F5,2,  NOTE_F5,4,

  NOTE_DS5,-8, NOTE_F5,16, NOTE_FS5,2, NOTE_F5,8, NOTE_DS5,8, //11
  NOTE_CS5,-8, NOTE_DS5,16, NOTE_F5,2, NOTE_DS5,8, NOTE_CS5,8,
  NOTE_C5,-8, NOTE_D5,16, NOTE_E5,2, NOTE_G5,8,
  NOTE_F5,16, NOTE_F4,16, NOTE_F4,16,
NOTE_F4,16,NOTE_F4,16,NOTE_F4,16,NOTE_F4,16,NOTE_F4,16,NOTE_F4,8, NOTE_F4,16,NOTE_F4,8,

  NOTE_AS4,4,  NOTE_F4,-4,  NOTE_AS4,8,  NOTE_AS4,16,  NOTE_C5,16, NOTE_D5,16, NOTE_DS5,16,//15
  NOTE_F5,2,  NOTE_F5,8,  NOTE_F5,8,  NOTE_F5,8,  NOTE_FS5,16, NOTE_GS5,16,
  NOTE_AS5,-2, NOTE_CS6,4,
  NOTE_C6,4, NOTE_A5,2, NOTE_F5,4,
  NOTE_FS5,-2, NOTE_AS5,4,
  NOTE_A5,4, NOTE_F5,2, NOTE_F5,4,

  NOTE_FS5,-2, NOTE_AS5,4,
  NOTE_A5,4, NOTE_F5,2, NOTE_D5,4,
  NOTE_DS5,-2, NOTE_FS5,4,
  NOTE_F5,4, NOTE_CS5,2, NOTE_AS4,4,
  NOTE_C5,-8, NOTE_D5,16, NOTE_E5,2, NOTE_G5,8,
  NOTE_F5,16, NOTE_F4,16, NOTE_F4,16,
NOTE_F4,16,NOTE_F4,16,NOTE_F4,16,NOTE_F4,16,NOTE_F4,16,NOTE_F4,8, NOTE_F4,16,NOTE_F4,8

};

// sizeof gives the number of bytes, each int value is composed of two bytes (16 bits)
// there are two values per note (pitch and duration), so for each note there are four bytes
int notes=sizeof(melody)/sizeof(melody[0])/2;

// this calculates the duration of a whole note in ms (60s/tempo)*4 beats
int wholenote = (60000 * 4) / tempo;
int divider = 0, noteDuration = 0;

void setup() {
  // iterate over the notes of the melody.
  // Remember, the array is twice the number of notes (notes + durations)
  for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2) {

    // calculates the duration of each note
    divider = melody[thisNote + 1];
    if (divider > 0) {
      // regular note, just proceed
      noteDuration = (wholenote) / divider;
    } else if (divider < 0) {
      // dotted notes are represented with negative durations!!
      noteDuration = (wholenote) / abs(divider);
      noteDuration *= 1.5; // increases the duration in half for dotted notes
    }

    // we only play the note for 90% of the duration, leaving 10% as a pause
    tone(buzzer, melody[thisNote], noteDuration*0.9);

    // Wait for the specief duration before playing the next note.
    delay(noteDuration);

    // stop the waveform generation before the next note.
    noTone(buzzer);
  }

}

void loop() {
  // no need to repeat the melody.
}
```

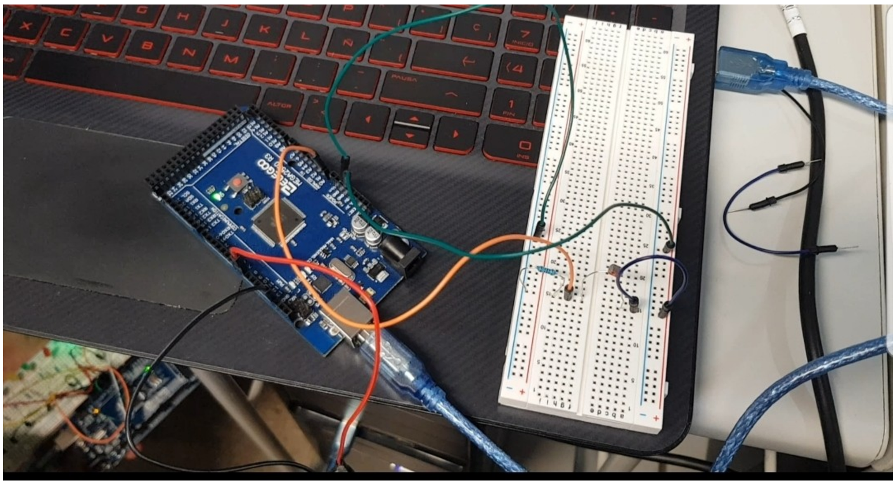
2.- Theremin de luz sin LEDS

Manteniendo el circuito del zumbador pasivo le añadí lo necesario para crear el theremin de luz sin leds siguiendo los siguientes esquemas.





(En este esquema ignoramos los leds)

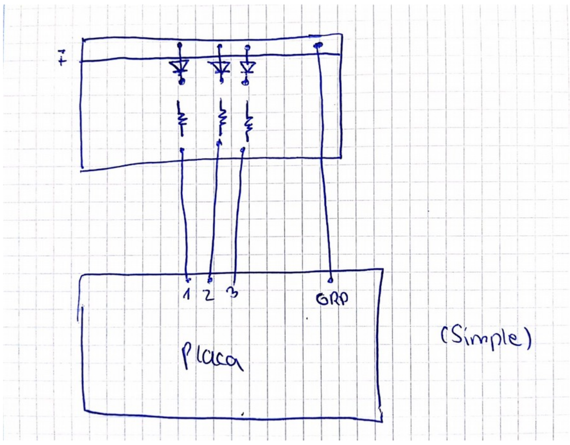


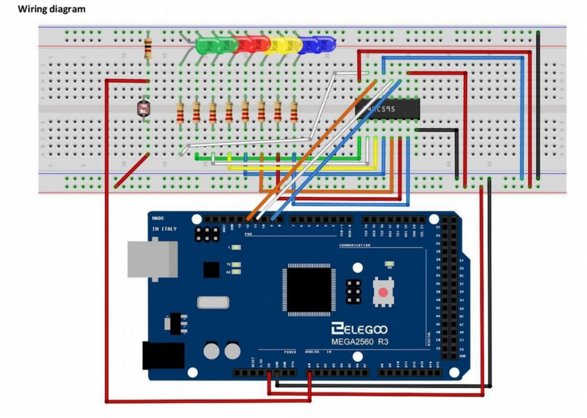
video sobre el circuito:
https://drive.google.com/open?id=1BgTYlBzRLegk6RL1NKjLZOzfdmZV1KY7

código:

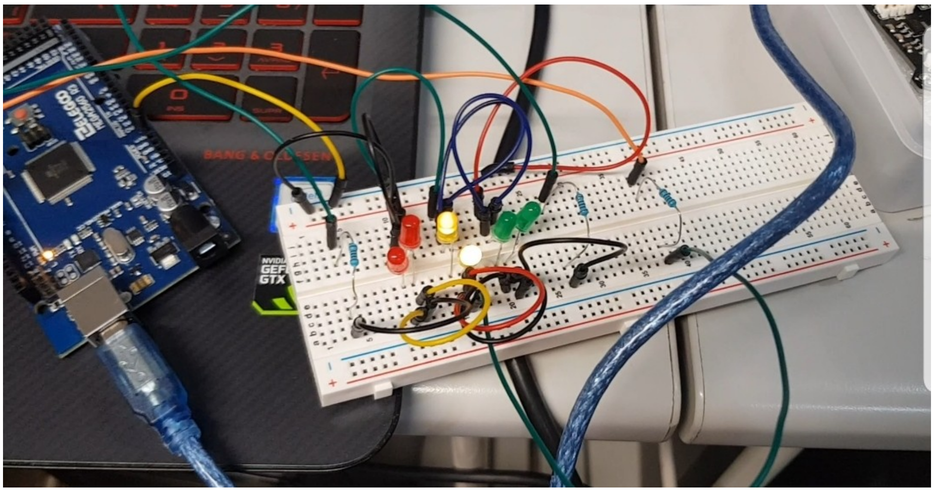
```
/*
  Arduino Starter Kit example
  Project 6 - Light Theremin

  This sketch is written to accompany Project 6 in the Arduino Starter Kit

  Parts required:
  - photoresistor
  - 10 kilohm resistor
  - piezo

  created 13 Sep 2012
  by Scott Fitzgerald

  http://www.arduino.cc/starterKit

  This example code is part of the public domain.
*/

// variable to hold sensor value
int sensorValue;
// variable to calibrate low value
int sensorLow = 1023;
// variable to calibrate high value
int sensorHigh = 0;
// LED pin
const int ledPin = 13;

void setup() {
  // Make the LED pin an output and turn it on
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, HIGH);

  // calibrate for the first five seconds after program runs
  while (millis() < 5000) {
    // record the maximum sensor value
    sensorValue = analogRead(A0);
    if (sensorValue > sensorHigh) {
      sensorHigh = sensorValue;
    }
    // record the minimum sensor value
    if (sensorValue < sensorLow) {
      sensorLow = sensorValue;
    }
  }
  // turn the LED off, signaling the end of the calibration period
  digitalWrite(ledPin, LOW);
}

void loop() {
  //read the input from A0 and store it in a variable
  sensorValue = analogRead(A0);

  // map the sensor values to a wide range of pitches
  int pitch = map(sensorValue, sensorLow, sensorHigh, 50, 4000);

  // play the tone for 20 ms on pin 8
  tone(8, pitch, 20);

  // wait for a moment
  delay(10);
}
```

3.- Semáforo de LEDS





(solo la parte de los leds)



video: https://drive.google.com/open?id=1Bmvu_yvwFl6u40EXC7P7ZbQ00UBWjg6L

codigo:

```
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(12, OUTPUT);
  pinMode(10, OUTPUT);
  pinMode(8, 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(3000);                       // wait for a second
  digitalWrite(12, LOW);    // turn the LED off by making the voltage LOW
  for(int i=0; i<5;i++)
  {
    digitalWrite(10, HIGH);
    delay(1000);
    digitalWrite(10, LOW);
  }
  digitalWrite(8, HIGH);
  delay(3000);                       // wait for a second
  digitalWrite(8, LOW);
}
```