ACTUADORES
BUZZER
Clase 20

Suscríbete

ROBOTICS SPACE NV

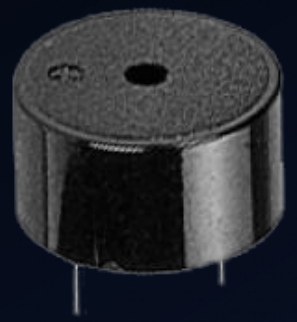
TUTOR:NAGIB LUIS VALLEJOS M.

BUZZER

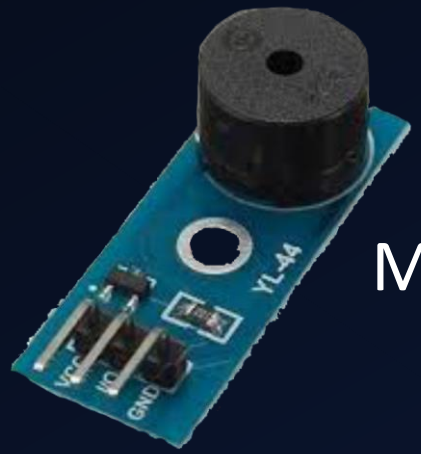
Es un actuador que produce un sonido, cuando se le aplica una corriente.

Se encuentran en diferentes dispositivos como alarmas, timbres, entre otros.

Existe dos tipos de Buzzer: Activo y pasivo y como tal, en el mercado se puede encontrar un Buzzer suelto o ya encapsulado en un módulo para su manejo



Buzzer suelto



Módulo buzzer

TUTOR:NAGIB LUIS VALLEJOS M.

BUZZER ACTIVO VS PASIVO

El Buzzer activo es aquel que genera una sola frecuencia predeterminada al conectar a la corriente directa.

Un Buzzer pasivo es aquel en el cual se puede generar diferentes frecuencias.

Los Buzzer activos son usados básicamente en las máquinas de hospitales o alarmas, en cambio los Buzzer pasivos nos permiten generar melodías gracias a su manejo de diferentes frecuencias.

TUTOR:NAGIB LUIS VALLEJOS M.

CARACTERÍSTICAS

- Voltaje de entrada:3V – 12V
- Frecuencia de trabajo:2 a 5 KHz
- Consumo de corriente: 32mA
- Salida de sonido: >=85dB
- Impedancia: 16Ω
- Rango de temperatura: -20ºC – 45ºC

TUTOR:NAGIB LUIS VALLEJOS M.

FUNCIÓN TONE() Y NOTONE()

La función **tone,** nos permite poder generar un sonido en una frecuencia determinada, su sintaxis es la siguiente:

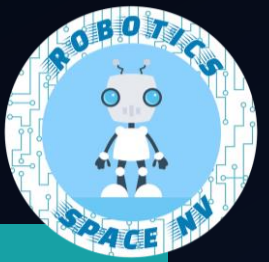
tone(pin,frecuencia);

La función **notone**, nos permite dejar al Buzzer en silencio, maneja la siguiente sintaxis:

noTone(pin);

TUTOR:NAGIB LUIS VALLEJOS M.

FRECUENCIAS DE NOTAS MUSICALES



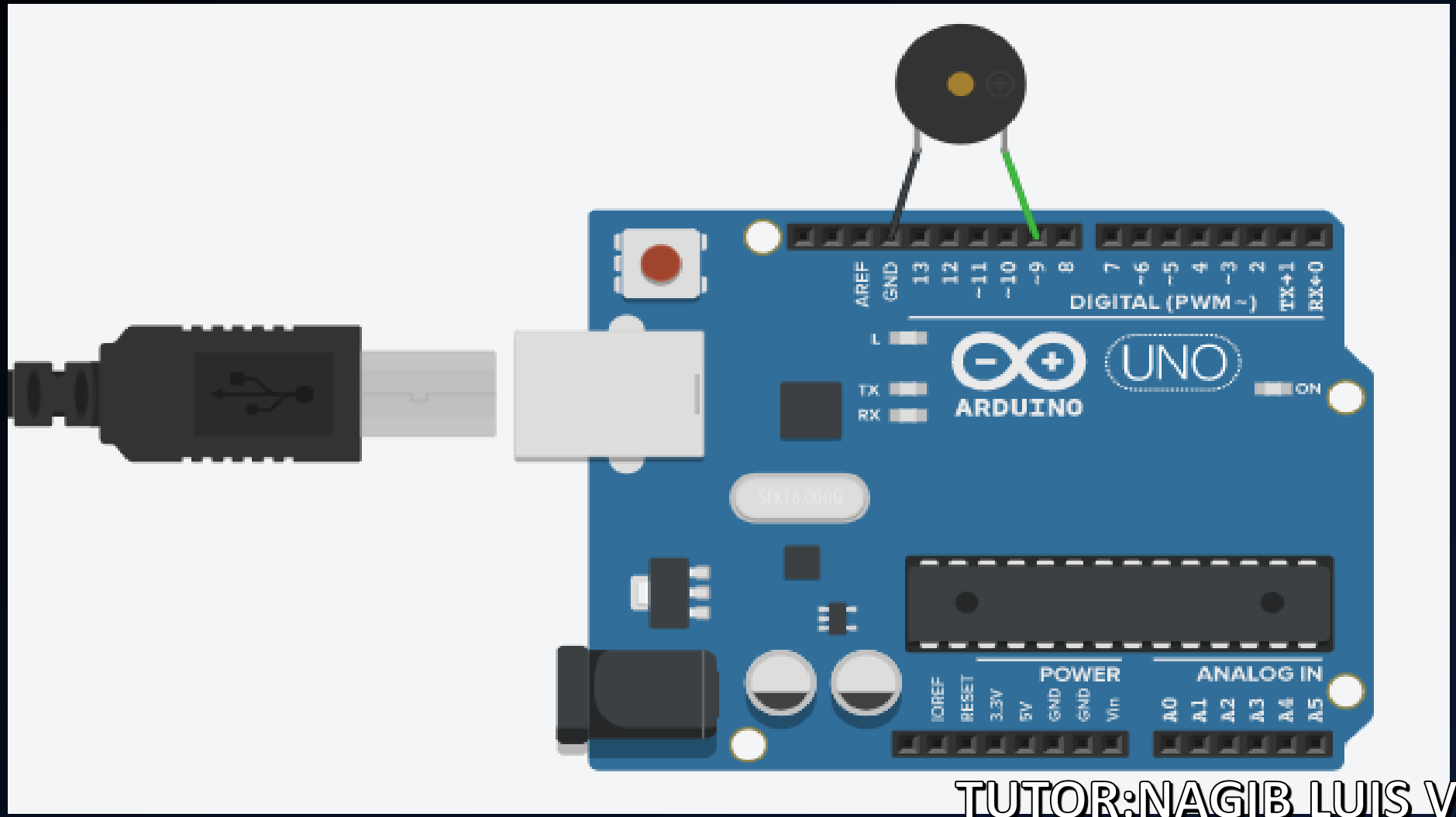
S3-EJ1 | pitches.h

```
/************************
 * Notas musicales
 ************************
#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
```

TUTOR:NAGIB LUIS VALLEJOS M.

EJEMPLO 1 – CIRCUTO

Generar las siguientes notas musicales a razón de un segundo: DO, RE, MI, FA, SOL, LA, SI



TUTOR:NAGIB LUIS VALLEJOS M.

EJEMPLO 1 – SOLUCIÓN

Generar las siguientes notas musicales a razón de un segundo: DO, RE, MI, FA, SOL, LA, SI

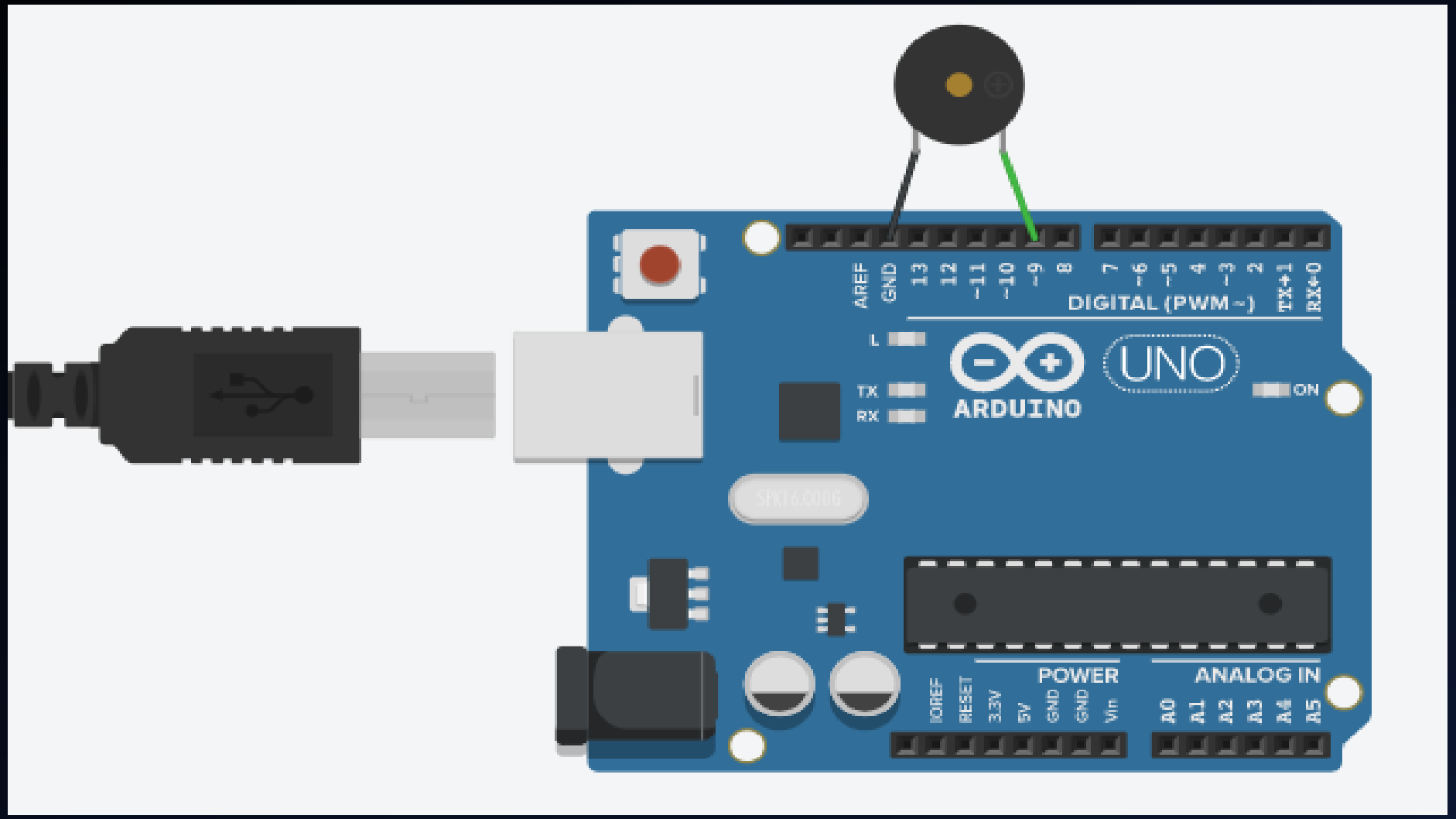
S20-E1

```
int buz=9;
void setup(){
  pinMode(buz,OUTPUT);
}
void loop(){
  tone(buz,262); //DO
  delay(1000);
  tone(buz,294); //RE
  delay(1000);
  tone(buz,330); //MI
  delay(1000);
  tone(buz,349); //FA
  delay(1000);
  tone(buz,392); //SOL
  delay(1000);
  tone(buz,440); //LA
  delay(1000);
  tone(buz,494); //SI
  delay(1000);
}
```

TUTOR:NAGIB LUIS VALLEJOS M.

EJEMPLO 2 – CIRCUTO

Generar la melodía anterior dando un silencio de 2 segundos entre cada nota



TUTOR:NAGIB LUIS VALLEJOS M.

EJEMPLO 2 – SOLUCIÓN

Generar la melodía anterior dando un silencio de 2 segundos entre cada nota

S20-E2

```
int buz=9;
void setup(){
  pinMode(buz,OUTPUT);
}
void loop(){
  tone(buz,262); //DO
  delay(1000);
  noTone(buz);
  delay(2000);
  tone(buz,294); //RE
  delay(1000);
  noTone(buz);
  delay(2000);
  tone(buz,330); //MI
  delay(1000);
  noTone(buz);
  delay(2000);
  tone(buz,349); //FA
  delay(1000);
  noTone(buz);
  delay(2000);
  tone(buz,392); //SOL
  delay(1000);
  noTone(buz);
  delay(2000);
  tone(buz,440); //LA
  delay(1000);
  noTone(buz);
  delay(2000);
  tone(buz,494); //SI
  delay(1000);
  noTone(buz);
  delay(2000);
}
```

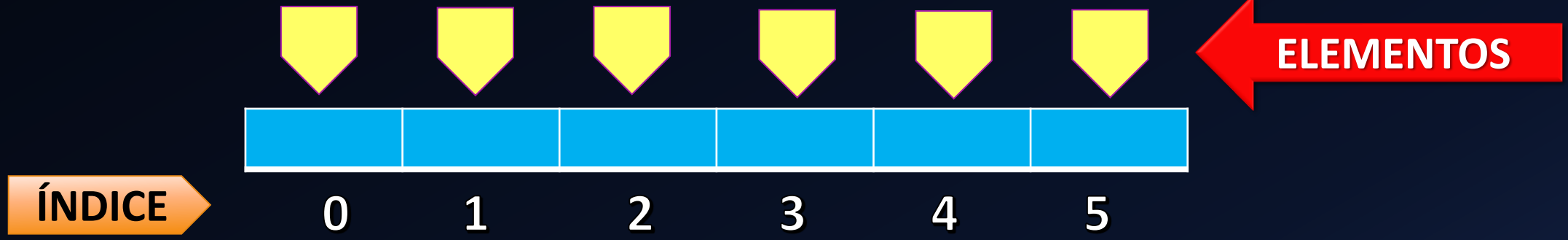
TUTOR:NAGIB LUIS VALLEJOS M.

VECTOR (ARREGLO UNIDIMENSIONAL)

Es una colección o grupo de datos agrupados, donde cada dato tiene una posición especifica y siempre son de un solo tipo de elemento.

Existen vectores (arreglos) numéricos, de cadenas, caracteres, etc.

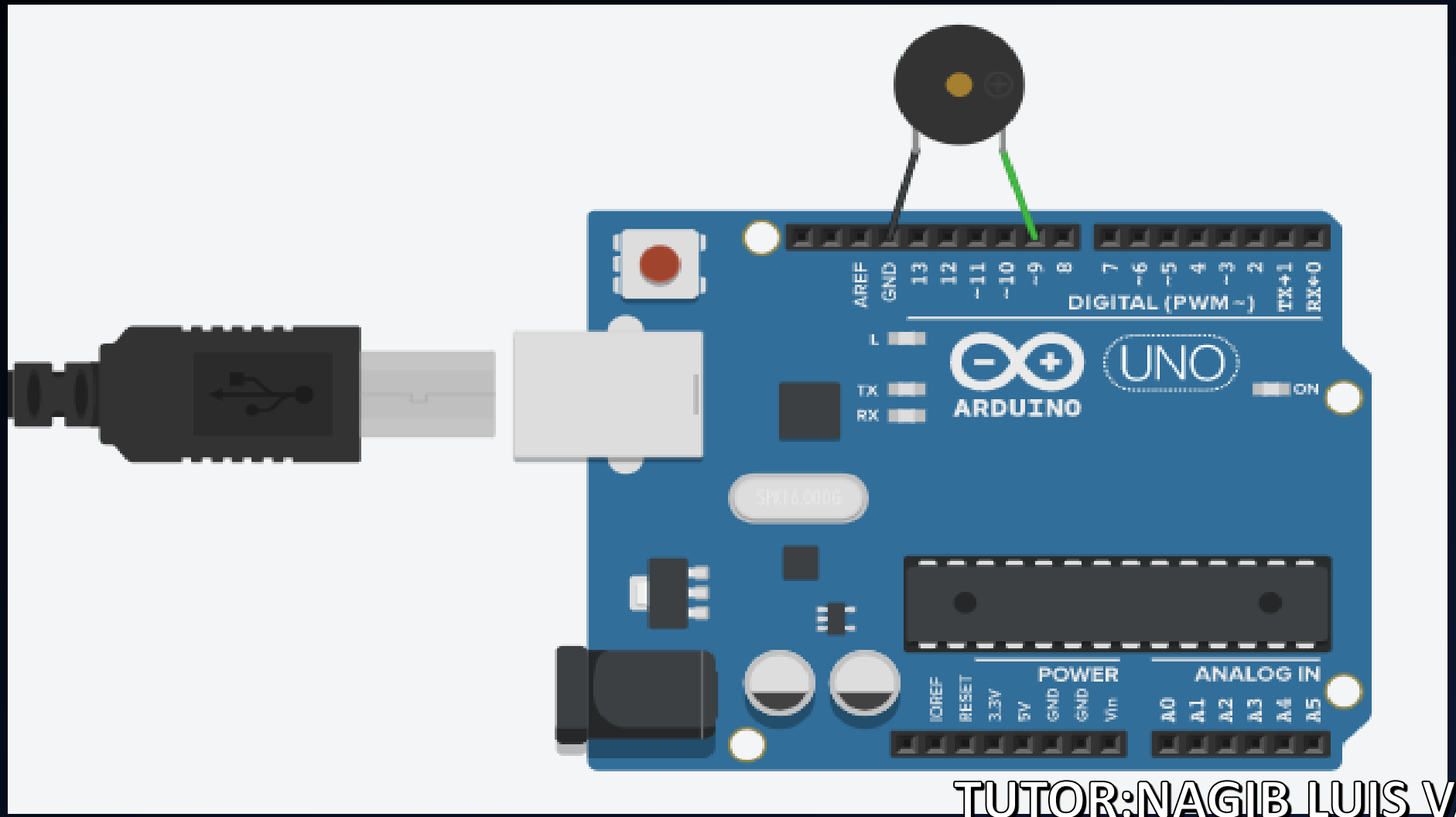
Vector numérico:	5	78	52	1	0	98
Vector de cadenas:	Hola	Robótica	Arduino	Azul	Lápiz	led
Vector de caracteres:	A	s	f	x	m	L



TUTOR:NAGIB LUIS VALLEJOS M.

EJEMPLO 3 – CIRCUTO

Generar la melodía anterior dando un silencio de 0,1 segundo entre cada nota, aplicando vectores



TUTOR:NAGIB LUIS VALLEJOS M.

EJEMPLO 3 – SOLUCIÓN

Generar la melodía anterior dando un silencio de 0,1 segundo entre cada nota, aplicando vectores

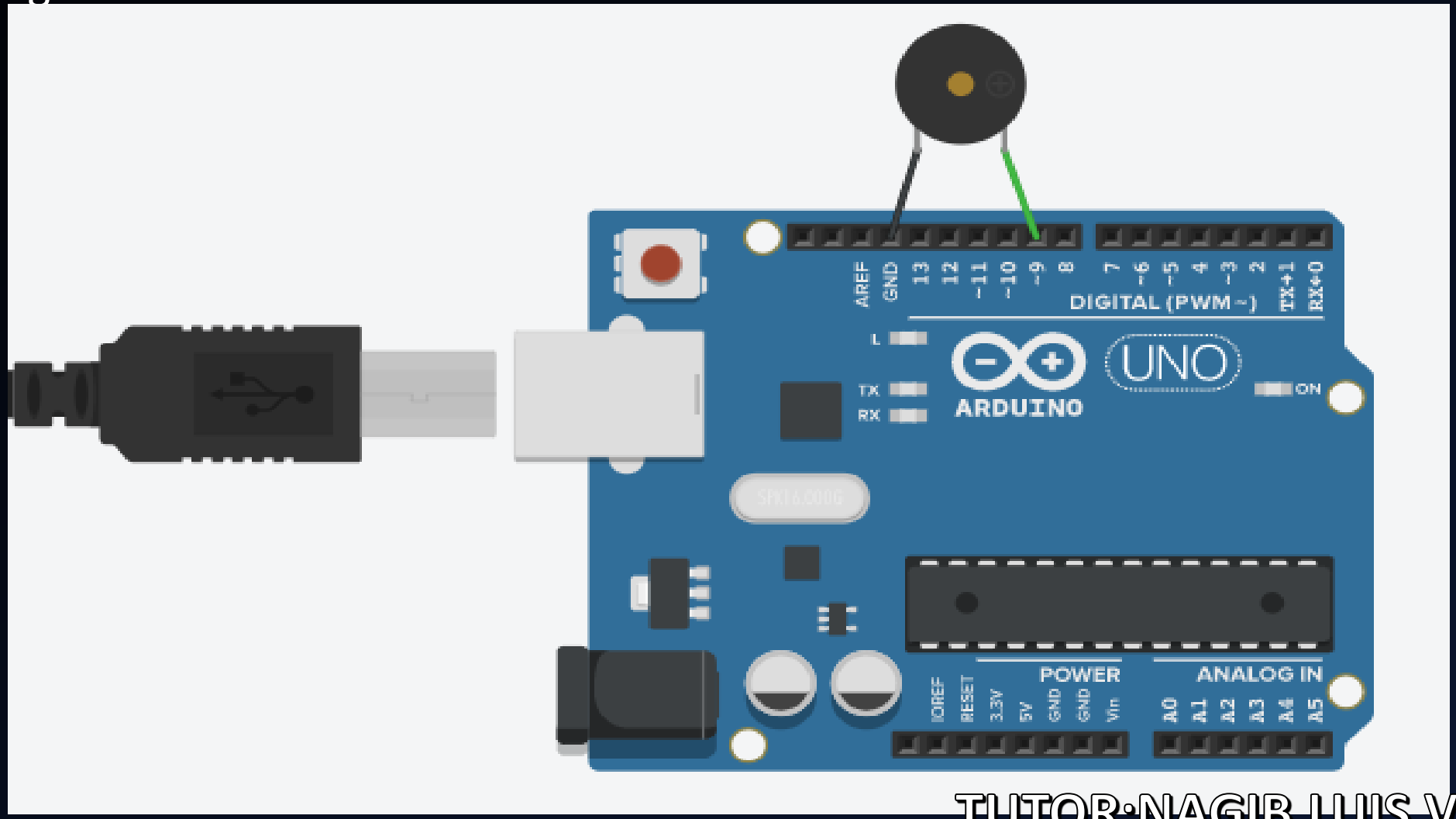
S20-E3

```
1  int buz=9;
2  int notas []={262,294,330,349,392,440,494}; //Vector de notas
3  void setup(){
4    pinMode(buz, OUTPUT);
5  }
6  void loop(){
7    for(int i=0;i<7;i++){
8      tone(buz,notas[i]);
9      delay(1000);
10     noTone(buz);
11     delay(100);
12   }
13 }
```

TUTOR:NAGIB LUIS VALLEJOS M.

EJEMPLO 4 – CIRCUTO

Generar una melodía que contenga 10 notas musicales y que no dure más de 2 segundos



TUTOR:NAGIB LUIS VALLEJOS M.

EJEMPLO 4 – SOLUCIÓN

Generar una melodía que contenga 10 notas musicales y que no dure más de 2 segundos

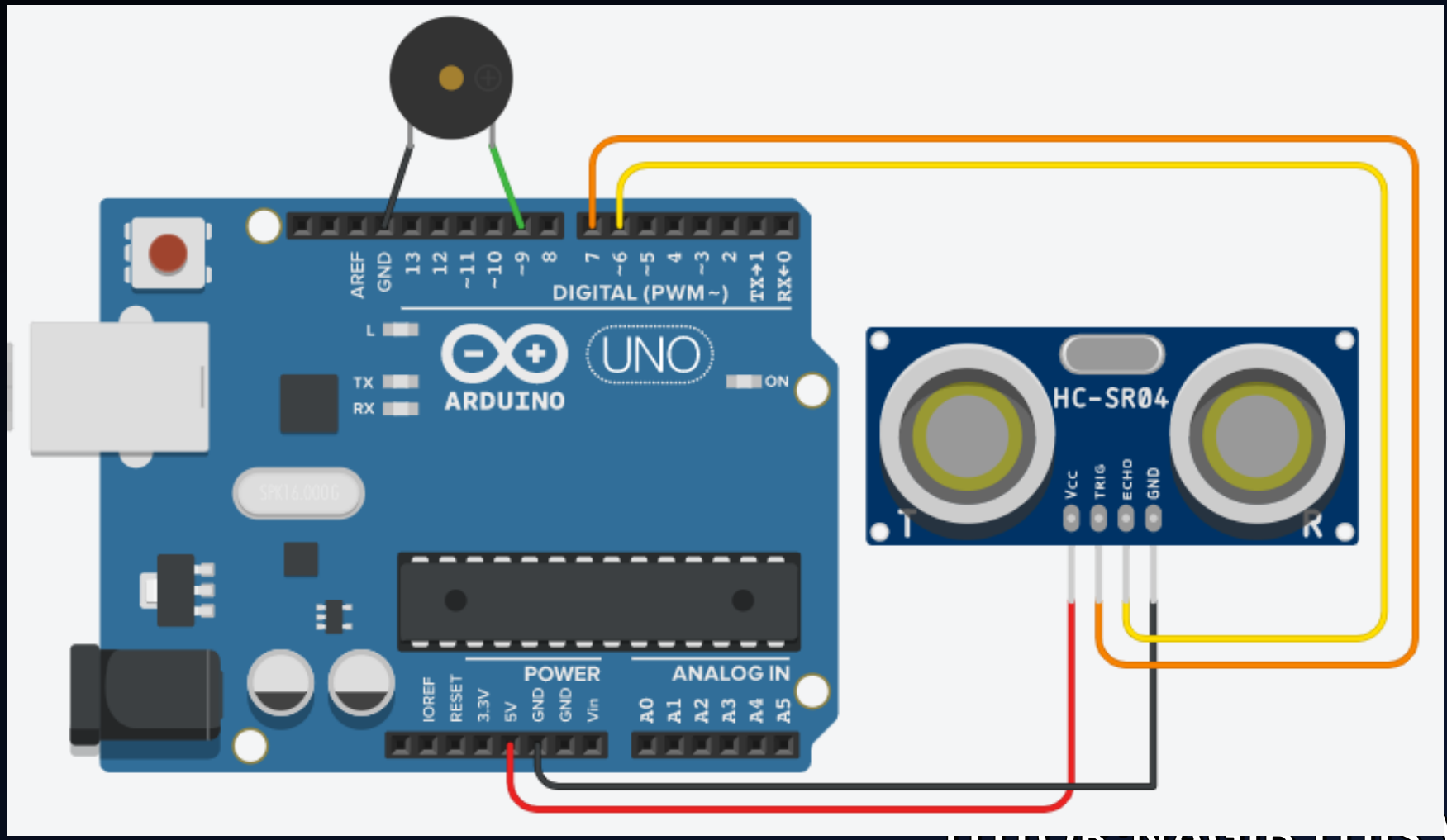
S20-E4

```
1  int buz=9;
2  int notas[]={262,294,330,349,392,440,494,440,392,349};
3  void setup(){
4    pinMode(buz,OUTPUT);
5  }
6  void loop(){
7    for(int i=0;i<10;i++){
8      tone(buz,notas[i]);
9      delay(150);
10     noTone(buz);
11     delay(50);
12   }
13 }
```

TUTOR:NAGIB LUIS VALLEJOS M.

EJEMPLO 5 – CIRCUTO

Generar una melodía la cual debe sonar cuando se detecte un obstáculo <15, la melodía no debe durar mas de 2 seg



TUTOR:NAGIB LUIS VALLEJOS M.

EJEMPLO 5 – SOLUCIÓN

Generar una melodía la cual debe sonar cuando se detecte un obstáculo <15, la melodía no debe durar mas de 2 seg

S20-E5

```
1  int buz=9,trig=7,echo=6;
2  long duracion, distancia;
3  int notas[]={262,294,330,349,392,440,494,330,294,440};
4  void setup(){
5    pinMode(buz,OUTPUT);
6    pinMode(trig,OUTPUT);
7    pinMode(echo,INPUT);
8  }
9
10 void loop(){
11   digitalWrite(trig,0);
12   delayMicroseconds(2);
13   digitalWrite(trig,1);
```

TUTOR:NAGIB LUIS VALLEJOS M.

EJEMPLO 5 – SOLUCIÓN

Generar una melodía la cual debe sonar cuando se detecte un obstáculo <15, la melodía no debe durar mas de 2 seg

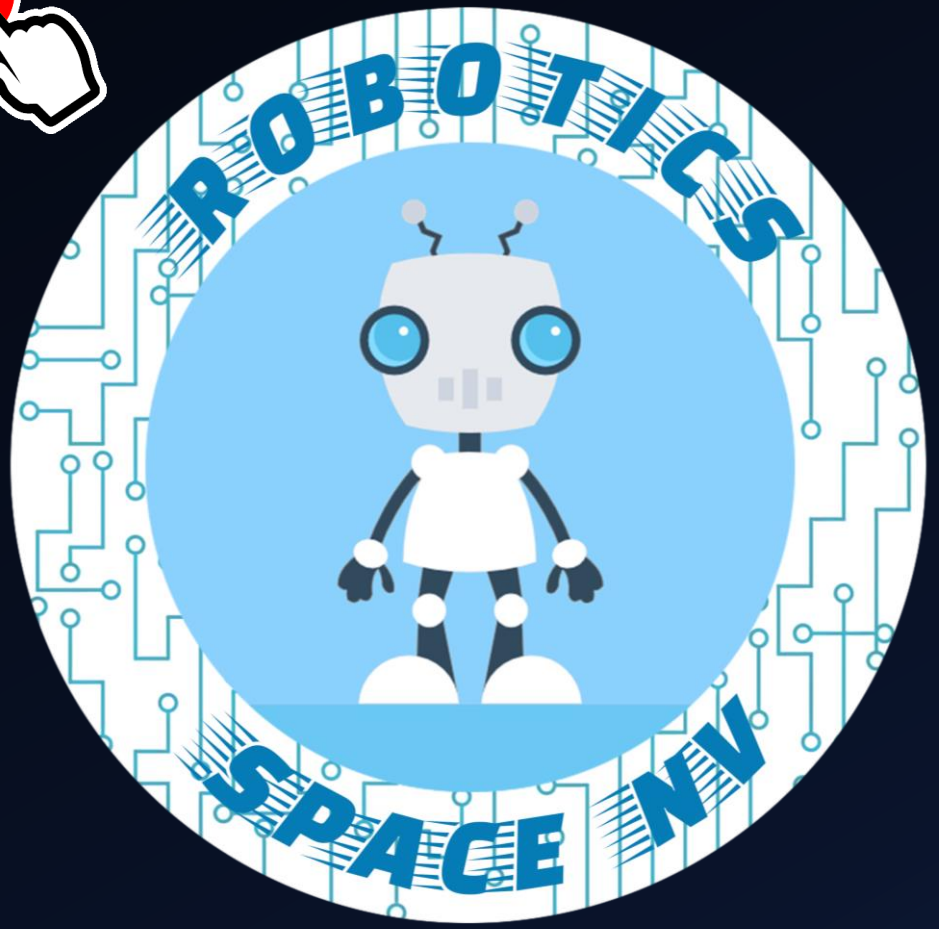
S20-E5

```
14    delayMicroseconds(10);
15    digitalWrite(trig,0);
16    duracion=pulseIn(echo,1);
17    distancia=duracion/58;   //(duracion/2)/29
18    if(distancia<15){
19      for(int c=0;c<10;c++){
20      tone(buz,notas[c]);
21      delay(150);
22      noTone(buz);
23      delay(50);
24      }
25    }
26  }
```

TUTOR:NAGIB LUIS VALLEJOS M.

CONTACTOS

Suscríbete



(+591) 63096640

robotics.space.nv@gmail.com

fb.me/RoboticsSpaceNV

@NagibVallejos

Robotics Space NV

https://github.com/nagibvalejos/Robotics-Space-NV

TUTOR:NAGIB LUIS VALLEJOS M.