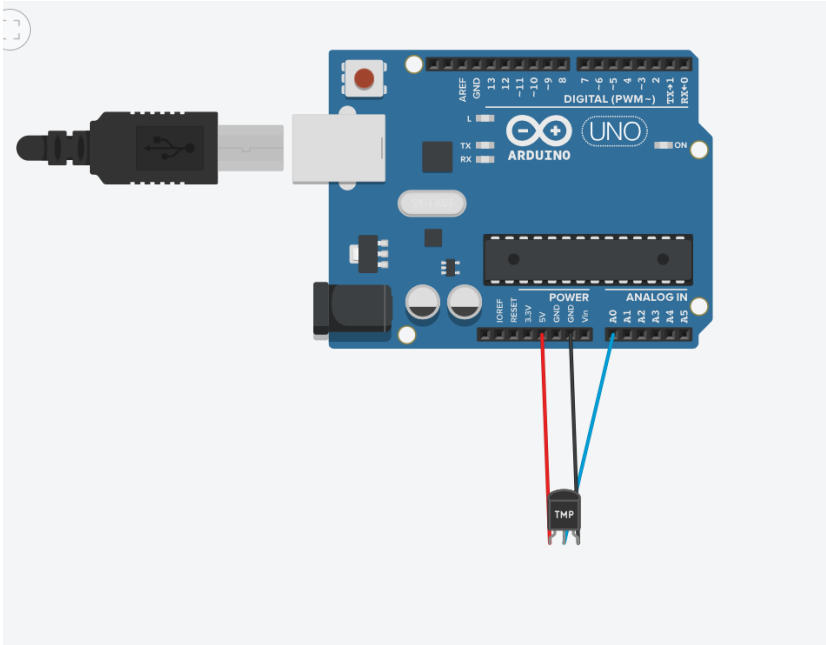


Serial Monitor

```
Temperatura: 24.78 Â°C
Temperatura: 24.78 Â°C
Temperatura: 24.78 Â°C
Temperatura: 24.78 Â°C
Temperatura: 24.78 Â°C
Temperatura: 24.78 Â°C
Temperatura: 24.78 Â°C
Temperatura: 24.78 Â°C
Temperatura: 24.78 Â°C
Temperatura: 24.78 Â°C
Temperatura: 24.78 Â°C
Temperatura: 24.78 Â°C
```

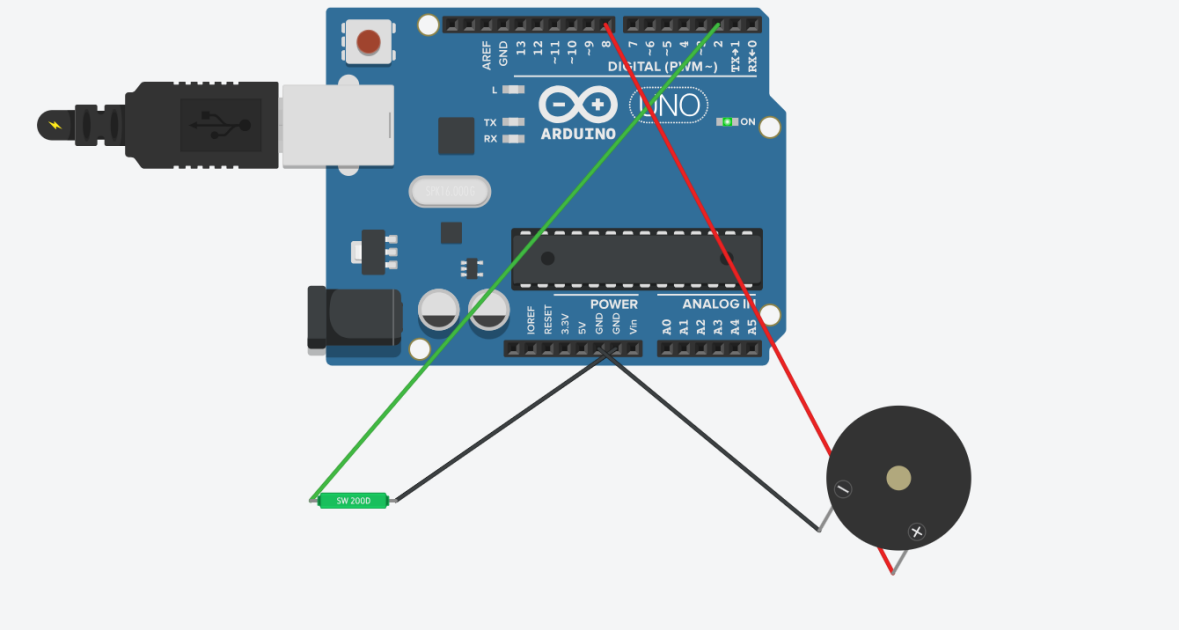
https://www.tinkercad.com/things/jAH5Dcsc9gc-practica-1?sharecode=gJzcIV2X-bqg4SCpdt8SpgO65ZA7fi3hQkUKhFF5vLg

Código

```
// C++ code
//
const int sensorPin = A0;



void setup()
{
  Serial.begin(9600);
}

void loop()
{// Leer el valor analógico del sensor (0-1023)
  int sensorValue = analogRead(sensorPin);

  // Convertir el valor analógico a voltaje (0-5V)
  float voltage = sensorValue * (5.0 / 1023.0);

  // Calcular la temperatura en grados Celsius
  float temperatureC = (voltage - 0.5) / 0.01;

  // Mostrar la temperatura en el monitor serial
  Serial.print("Temperatura: ");
  Serial.print(temperatureC);
  Serial.println(" °C");

  // Esperar un segundo antes de la próxima lectura
  delay(1000);
```

}

Practica 2



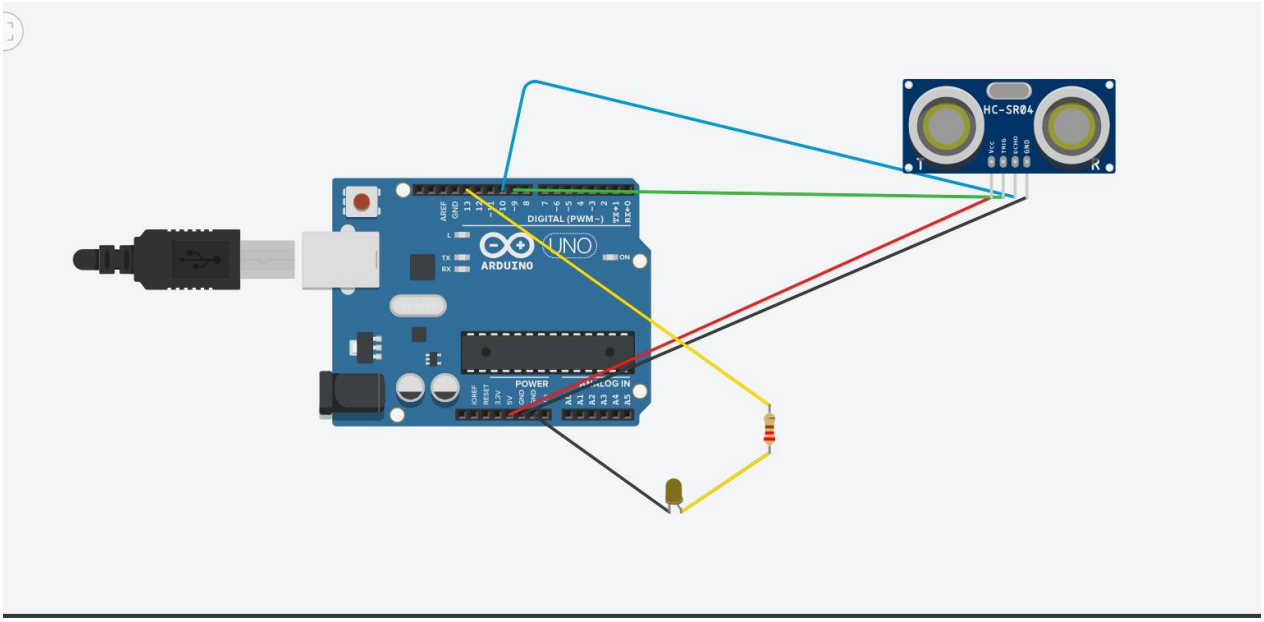
https://www.tinkercad.com/things/bbami4hjOdy-practica-2?sharecode=OjF-xT-GYG-MXXCyYhMeKyq6dOwns8DM7N0sn5agYBs

Código:

```
// Pin donde está conectado el sensor de inclinación
const int tiltPin = 2;

// Pin donde está conectado el buzzer
const int buzzerPin = 8;
```

```
// Pin donde está conectado el LED
const int ledPin = 10;

void setup() {
  // Configura el pin del sensor como entrada
  pinMode(tiltPin, INPUT);

  // Configura el pin del buzzer como salida
  pinMode(buzzerPin, OUTPUT);



}

void loop() {
  // Lee el estado del sensor de inclinación
  int tiltState = digitalRead(tiltPin);

  // Si el tilt sensor detecta inclinación (posición horizontal)
  if (tiltState == HIGH) {
    // Encender el buzzer (emitir sonido continuo)
    tone(buzzerPin, 1000);  // Emitir un tono de 1000 Hz
  } else {
    // Apagar el buzzer
    noTone(buzzerPin);
  }
}
```

Practica 3



Serial Monitor

```
Distancia: 111.89 cm
Distancia: 112.08 cm
Distancia: 112.05 cm
Distancia: 112.08 cm
Distancia: 112.08 cm
Distancia: 112.08 cm
Distancia: 111.89 cm
Distancia: 112.08 cm
Distancia: 112.08 cm
Distancia: 112.08 cm
Distancia: 112.08 cm
Distancia: 112.05 cm
Distancia: 111.86 cm
Distancia: 112.08 cm
```

https://www.tinkercad.com/things/5CqvIZJ4cVh-practica-3?sharecode=GESKiThzROv6AbNfS-IU3PRRf4xNW74bzqIBYS_pV4w

Código:

```
// Definir los pines del sensor de ultrasonido
const int trigPin = 9;    // Pin Trig (salida)
const int echoPin = 10;   // Pin Echo (entrada)

void setup() {
  // Configurar los pines del sensor
  pinMode(trigPin, OUTPUT);  // Trig como salida
  pinMode(echoPin, INPUT);   // Echo como entrada

  // Inicializar comunicación serial para depuración
  Serial.begin(9600);
}

void loop() {
  // Enviar un pulso al pin Trig
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);  // Esperar 2 microsegundos
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10); // Mantener el pulso HIGH durante 10 microsegundos
  digitalWrite(trigPin, LOW);

  // Leer el tiempo de respuesta del pin Echo
  long duration = pulseIn(echoPin, HIGH);

  // Calcular la distancia en cm
  float distance = duration * 0.034 / 2;

  // Mostrar la distancia en el monitor serial
  Serial.print("Distancia: ");
```

```
  Serial.print(distance);
  Serial.println(" cm");

  // Esperar un poco antes de la próxima lectura
  delay(200);
}
```

Reporte Paso a Paso de las Prácticas

Práctica 1: Medición de Temperatura con un Sensor Analógico

Objetivo:Leer la temperatura en grados Celsius utilizando un sensor analógico conectado a un Arduino.

Materiales Utilizados:

Arduino UNO

Sensor de temperatura (LM35 o similar)

Cables de conexión

Computadora con Arduino IDE

Procedimiento:

Conexión del Sensor:

Se conectó la salida del sensor al pin A0 de Arduino.

VCC del sensor a 5V y GND a tierra.

Configuración Inicial (setup):

Se inicializó la comunicación serial con Serial.begin(9600) para monitorear la temperatura en el monitor serial.

Lectura del Sensor (loop):

Se utilizó analogRead(sensorPin) para leer el valor analógico del sensor (0-1023).

Conversión a Voltaje:

Se convirtió el valor leído a voltaje con la fórmula:

```
float voltage = sensorValue * (5.0 / 1023.0);
```

Cálculo de Temperatura:

Se calculó la temperatura en grados Celsius con:

```
float temperatureC = (voltage - 0.5) / 0.01;
```

Visualización de Datos:

La temperatura se mostró en el monitor serial con Serial.print y Serial.println.

Retraso entre Lecturas:

Se añadió un delay(1000) para leer la temperatura cada segundo.

Resultados:Se obtuvo una lectura estable de la temperatura en tiempo real a través del monitor serial.

Práctica 2: Sensor de Inclinación con Alerta Sonora

Objetivo:

Detectar la inclinación de un objeto utilizando un sensor de inclinación y activar una alarma sonora (buzzer).

Materiales Utilizados:

Arduino UNO

Sensor de inclinación

Buzzer

LED (opcional)

Cables de conexión

Procedimiento:

Conexión de Componentes:

Sensor de inclinación al pin digital 2.

Buzzer al pin digital 8.

LED al pin digital 10.

Configuración Inicial (setup):

Se configuraron los pines de entrada y salida usando pinMode().

Lectura del Sensor (loop):

Se leyó el estado del sensor con digitalRead(tiltPin).

Activación del Buzzer:

Si el sensor detectaba inclinación (tiltState == HIGH), se activaba el buzzer con:

```
tone(buzzerPin, 1000);
```

Resultados:

El buzzer se activó al detectar la inclinación, funcionando correctamente como alarma.

Práctica 3: Medición de Distancia con Sensor Ultrasónico

Objetivo:

Medir la distancia de un objeto utilizando un sensor ultrasónico HC-SR04 y mostrar el valor en el monitor serial.

Materiales Utilizados:

Arduino UNO

Sensor ultrasónico HC-SR04

Cables de conexión

Procedimiento:

Conexión del Sensor:

Trig al pin digital 9.

Echo al pin digital 10.

Configuración Inicial (setup):

Se configuraron los pines de Trig y Echo con pinMode().

Se inicializó la comunicación serial.

Generación del Pulso Ultrasónico (loop):

Se envió un pulso de 10 microsegundos a Trig:

```
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
```

Medición del Tiempo de Respuesta:

Se midió el tiempo de respuesta del pin Echo:

Medición del Tiempo de Respuesta:

- Se midió el tiempo de respuesta del pin Echo:

```
long duration = pulseIn(echoPin, HIGH);
```

Cálculo de la Distancia:

- La distancia se calculó con la fórmula:

```
float distance = duration * 0.034 / 2;
```

Visualización de Datos:

- Los resultados se mostraron en el monitor serial.

Intervalo entre Mediciones:

- Se utilizó `delay(200)` para espaciar las lecturas.

Resultados:

Las mediciones de distancia se mostraron en tiempo real, con lecturas precisas y consistentes.