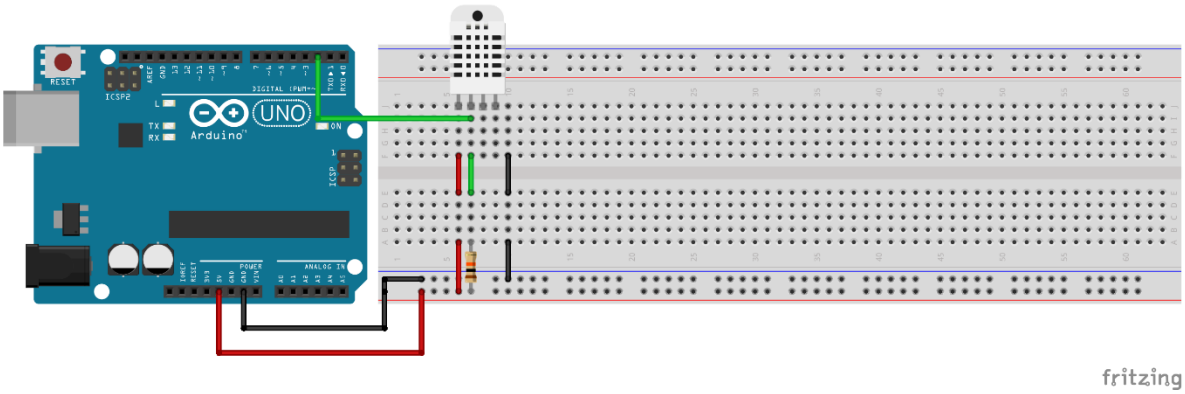
Práctica 1

Para esta práctica se necesitan los siguientes materiales:

- 1 Arduino con cable USB
- 1 Sensor DHT11
- 1 Resistencia 10kΩ
- Cables
- 1 Protoboard

1.- Conectar el sensor al Arduino siguiendo las conexiones mostradas en la siguiente figura:



2.- En el IDE de Arduino, escriba un programa para detectar la temperatura y la humedad de la habitación donde se encuentra e imprimirla en consola. Utilice el siguiente esqueleto de programa:

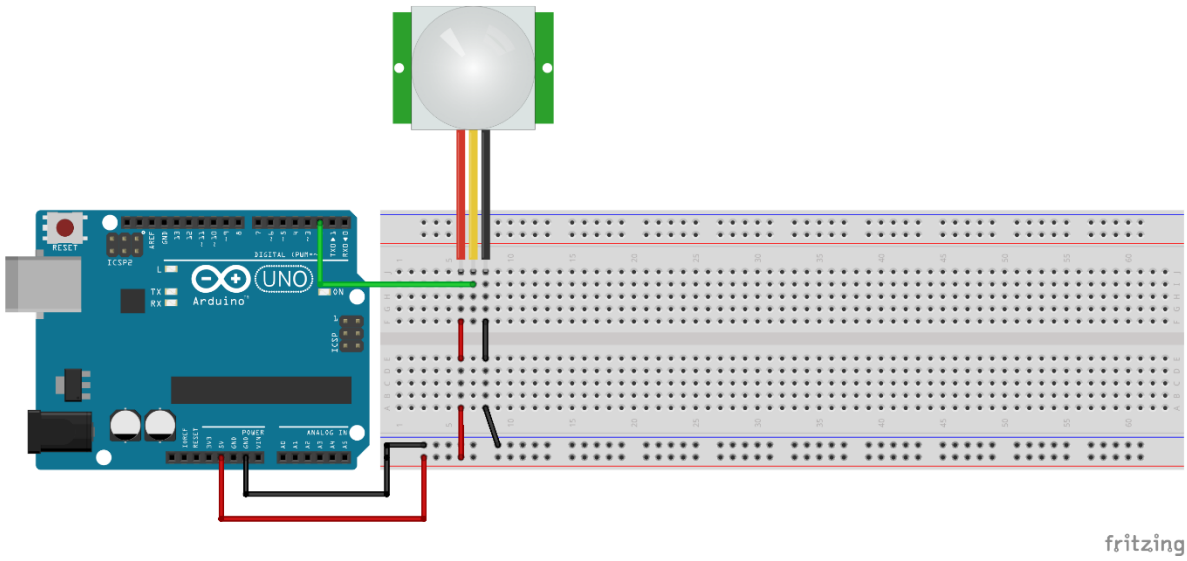
```
#include "DHT.h" //Incluir la librería DHT para el sensor
#define DHTPIN 2 //Definir donde se recibirán los datos del sensor
/**
Definir el tipo de sensor. Este dato puede variar ya que existen variantes de este mismo sensor.
*/
#define DHTTYPE DHT11
DHT sensor (DHTPIN, DHTTYPE) //Crear objeto del sensor

void setup() {
   Serial.begin (9600); //Iniciar consola serial a 9600 bps
   dht.begin();//Inicializar sensor
}

void loop() {
   delay (2000); // Pausa entre mediciones
/**
Funciones a utilizar

Función readHumidity(): Toma lectura de la humedad.
Nota: La lectura de humedad toma alrededor de 250ms.

Función readTemperature(): Toma lectura de la temperatura en grados dependiendo del parámetro
ingresado:
             o   Sin parámetro = lectura en grados Celsius (default).
             o   true = lectura en grados Fahrenheit

isnan(valor): Verifica si el valor analizado es válido.

Función computeHeatIndex(t, h, false): Calcula el índice de calor
Parámetros:
         t = temperatura en grados Celsius o Fahrenheit (si el ultimo parámetro no está presente).
         h = humedad
*/
}
```

Práctica 2

Para esta práctica se necesitan los siguientes materiales:

- 1 Arduino con cable USB
- 1 Sensor PIR
- Cables
- 1 Protoboard

1.- Conectar el sensor al Arduino siguiendo las conexiones mostradas en la siguiente figura:



2.- En el IDE de Arduino, escriba un programa para detectar movimiento con las lecturas del sensor. Utilice el siguiente esqueleto de programa:

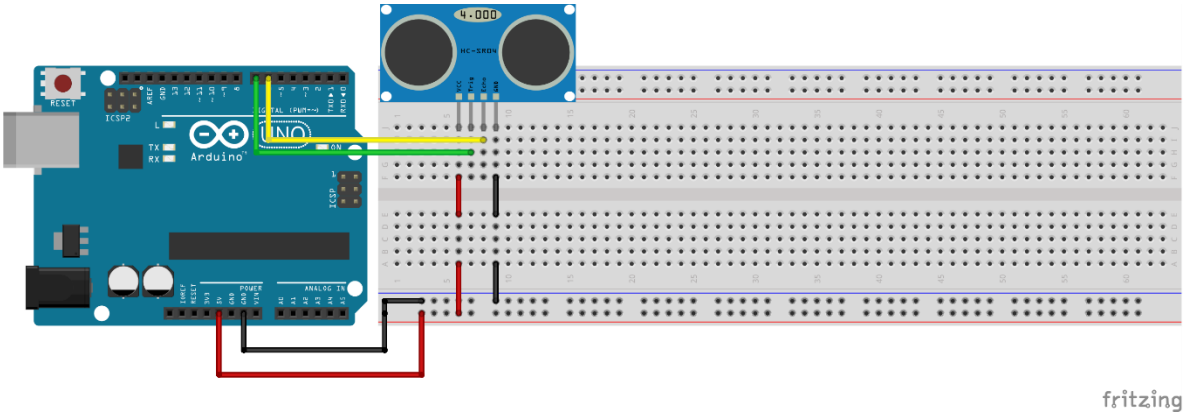
```
#define pirPin 2 //Definir el pin de entrada del sensor
long unsigned int tiempo; //Variable para almacenar el tiempo actual
long unsigned int pausa = 5000; //Pausa de 5 segundos
boolean movimientoDetectado = true; //Bandera para movimiento actual
boolean detectarMovimiento; //Bandera para nuevo movimiento

void setup() {
   Serial.begin(9600);//Inicializar consola a 9600 bps
   pinMode(pirPin, INPUT); //Establecer pin del sensor como entrada
}
void loop() {
/**
Si se detecta un estado HIGH en el pin del sensor
    -   Se verifica la bandera de movimiento; si es verdadera, se invierte su
        valor y se imprime un mensaje en consola con una pausa de 50ms.
    -   Se activa la bandera de nuevo movimiento.

Si se detecta un estado LOW en el pin del sensor
    -   Se verifica la bandera de nuevo movimiento; si es verdadera, se obtiene
        el tiempo actual y se invierte el valor de la bandera.
    -   Se verifica si se cumple la condición siguiente: si no hay movimiento
        detectado y si la diferencia de la lectura del tiempo actual con la
        lectura anterior es menor al valor de la pausa, se imprime un mensaje de
        fin de movimiento y se invierte el valor de la bandera de movimiento
        detectado
*/
}
```

Práctica 3

Para esta práctica se necesitan los siguientes materiales:

- 1 Arduino con cable USB
- 1 Sensor Ultrasónico HC-SR04
- Cables
- 1 Protoboard

1.- Conectar el sensor al Arduino siguiendo las conexiones mostradas en la siguiente figura:



2.- En el IDE de Arduino, escriba un programa para detectar la distancia de un objeto con las lecturas del sensor. Utilice el siguiente esqueleto de programa:

```
const int trigPin = 7;//Se define el pin para el trigger
const int echoPin = 6;//Se define el pin para el eco
long tiempo;//Variable para almacenar el tiempo en microsegundos
float distancia; //Variable para almacenar la distancia del objeto

void setup() {
   pinMode(trigPin, OUTPUT); // Se establece el pin de trigger como
salida
   pinMode(echoPin, INPUT); // Se establece el pin de eco como entrada
   Serial.begin(9600); // Se inicia comunicación serial a 9600 bps

}
void loop() {
/**
    -  Se envía una señal LOW al pin de trigger durante 2 microsegundos.
    -  Se envía una señal HIGH al pin de trigger durante 10 microsegundos, e
       inmediatamente después de envía una señal LOW al mismo pin.
    -  Se lee el pin de eco
    -  Se calcula la distancia con la fórmula: distancia = (tiempo*0.034)/2
    -  Se imprime la distancia en consola
    -  Funciones a utilizar:
          o  delayMicroseconds(x): establece una pausa en microsegundos
          o  pulseIn(pin, estado): lee un pulso o estado en un pin
                ▪  pin: número del pin a leer
                ▪  estado: valor del estado a leer (HIGH o LOW)
*/
}
```

Referencias:

Arduino Language Reference. [2018]. Recuperado de: https://www.arduino.cc/reference/en/#functions