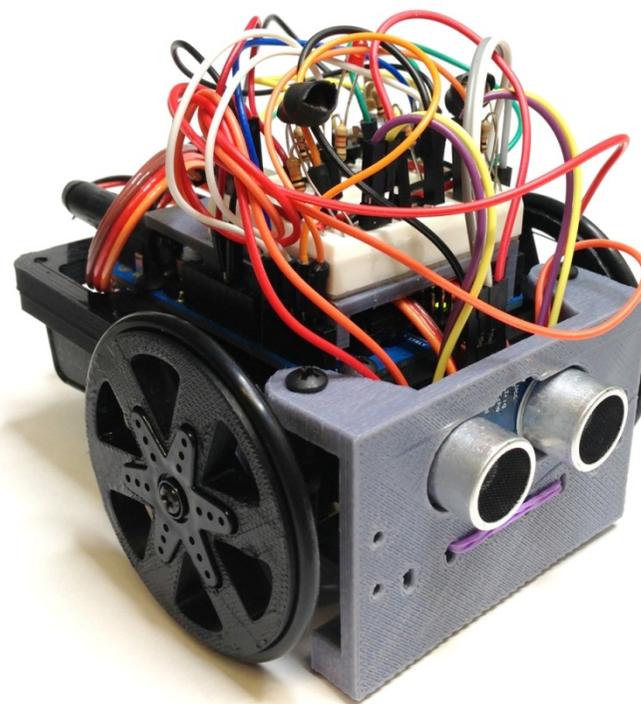


## **GUIA DE MONTAJE Y PROGRAMACIÓN**

SENSORES



# Sistema Basado en Microprocesador



- El **Microprocesador** es el cerebro de un ordenador
- Para funcionar necesita un **Programa** donde se indican las **órdenes** que el procesador debe seguir.
- **Elementos** de un Sistema basado en Microprocesador:
  - CPU (Central Processing Unit)
  - Memoria
  - Periféricos

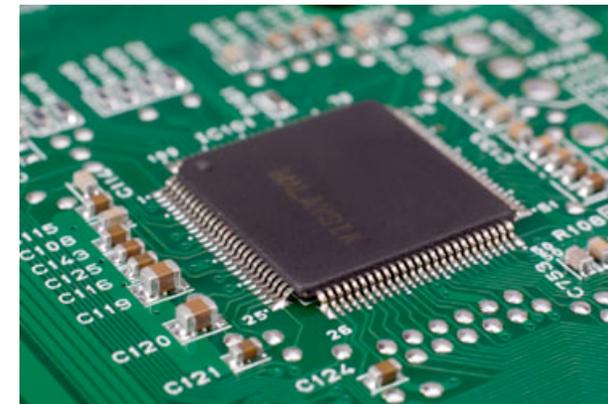


Source: <http://www.callegranvia.com/images/product/375/1b78d5c0b624c9a3c966e0854829dfe6.jpg>

# Programa y compilador



- Un **Programa** se puede escribir en muchos **lenguajes** diferentes.
- El microprocesador realmente entiende sólo órdenes en **código máquina** (**unos y ceros**)



Source: <http://pacotraver.files.wordpress.com/2011/11/interprete.jpg>

[http://2.bp.blogspot.com/\\_Pm8qvnCsVOI/TCwv\\_SAuznI/AAAAAAAAA4/9asQgJGiQMw/s1600/MICRO.jpg](http://2.bp.blogspot.com/_Pm8qvnCsVOI/TCwv_SAuznI/AAAAAAAAA4/9asQgJGiQMw/s1600/MICRO.jpg)

# Programa y compilador

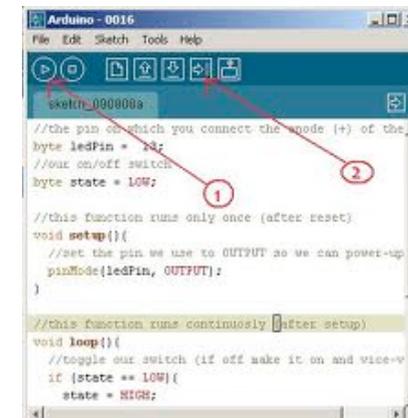
- Pero para nosotros es un poco “complicado” y nos es más fácil utilizar el **inglés** (como lengua internacionalmente más extendida)
- Nosotros vamos a utilizar una versión simplificada del **lenguaje C++** que es un **lenguaje de programación en alto nivel**
  - Las órdenes básicas están en **Inglés**
  - Tiene pocas **reglas gramaticales** pero muy estrictas.
- Un **compilador** pasa las órdenes del lenguaje de alto nivel al código máquina que entiende el procesador



Source: [http://www.microchip.com/stellent/images/mchpsiteimages/c32\\_CoverMainGfx.jpg](http://www.microchip.com/stellent/images/mchpsiteimages/c32_CoverMainGfx.jpg)

# Arduino: buen sistema de iniciación

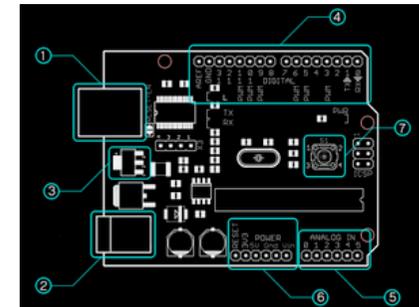
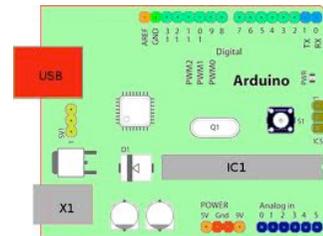
- ❑ Plataforma diseñada como elemento de **iniciación a la programación y a la electrónica.**
- ❑ Muy **sencillo** de utilizar.
- ❑ Dispone de una enorme **comunidad de usuarios.**



# Arduino: buen sistema de iniciación



- Desarrollada totalmente como **open-source**.
  - Los **esquemas** de las tarjetas hardware están disponibles para poderlos replicar.
  - Las fuentes de las **librerías** están disponibles para poder modificarlas y utilizarlas.
  - El **entorno de programación** también es abierto



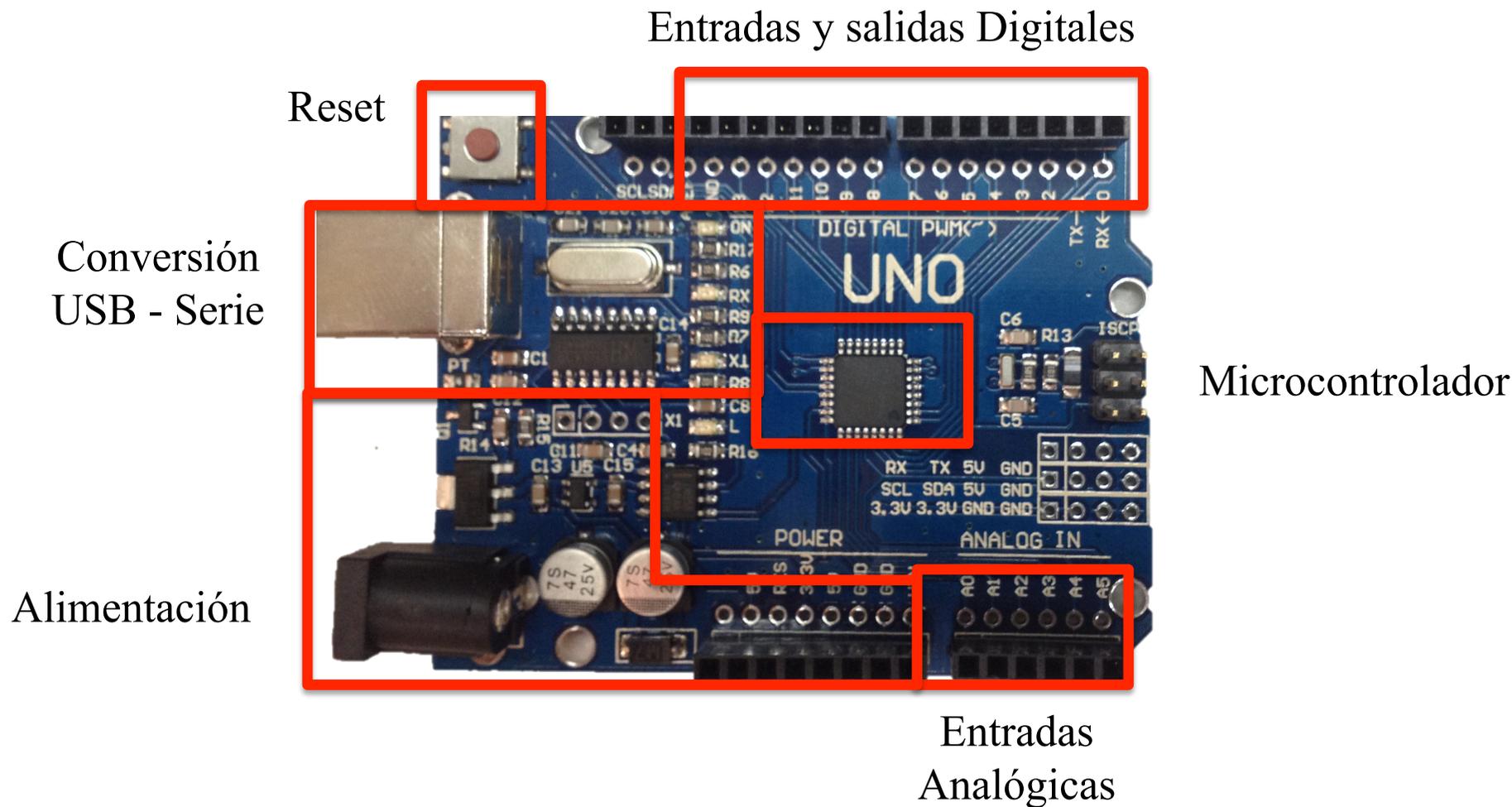
- **Multiplataforma**: Windows, Linux, MAC OS-X



Source: <http://www.ikkaro.com/definicion-arduino> <http://proyectos-arduino-geometriaemocional.blogspot.com.es/2012/03/jugando-con-sonido.html>

# Arduino: buen sistema de iniciación

- Elementos de la tarjeta **Arduino Uno**



# Entradas y Salidas

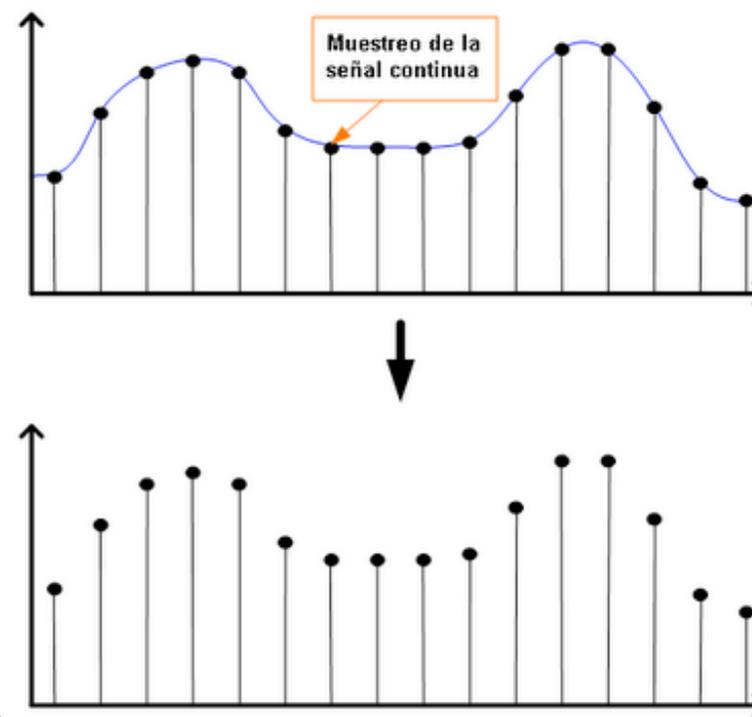
## □ Entradas y Salidas Digital

'1' → 5 voltios → Nivel Alto → HIGH

'0' → 0 voltios → Nivel Bajo → LOW

## □ Entradas Analógicas

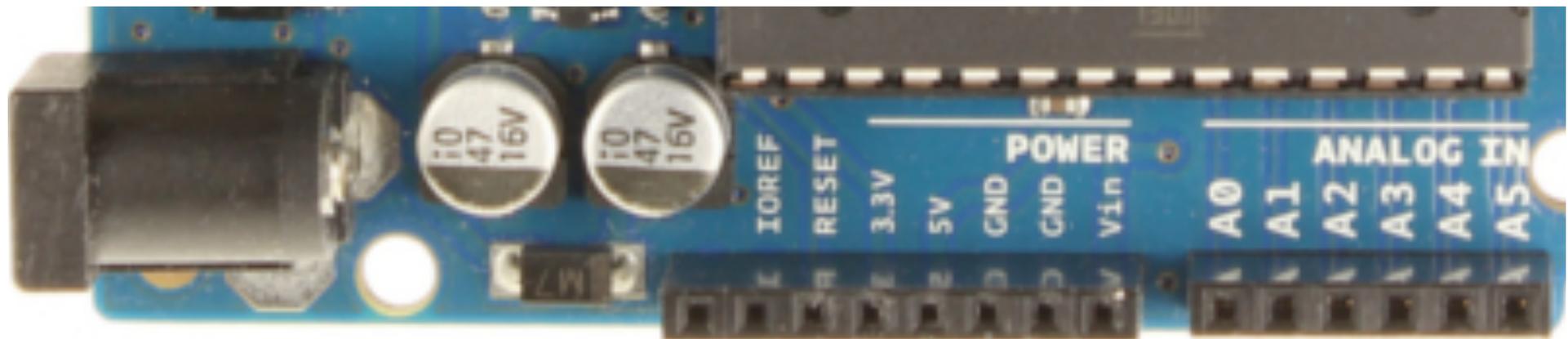
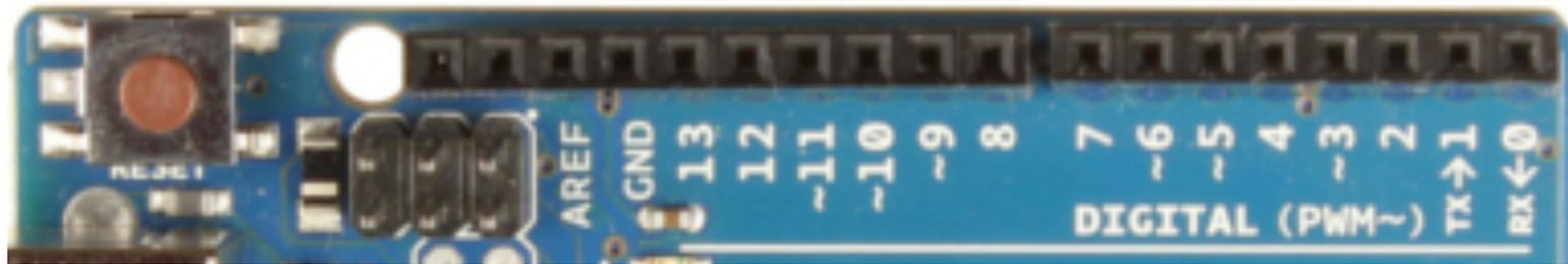
- La tensión varía de 0 voltios a 5 voltios



Source: <http://oirverycontar.files.wordpress.com/2011/09/muestreo.png>

# Arduino: buen sistema de iniciación

- Entradas y salidas de la tarjeta Arduino Uno y compatibles



**GND = 0V**



# Preparando la Tarjeta Arduino

- ❑ Comprueba que las pegatinas están pegadas correctamente



# Instalando el software en el ordenador



## Actividad

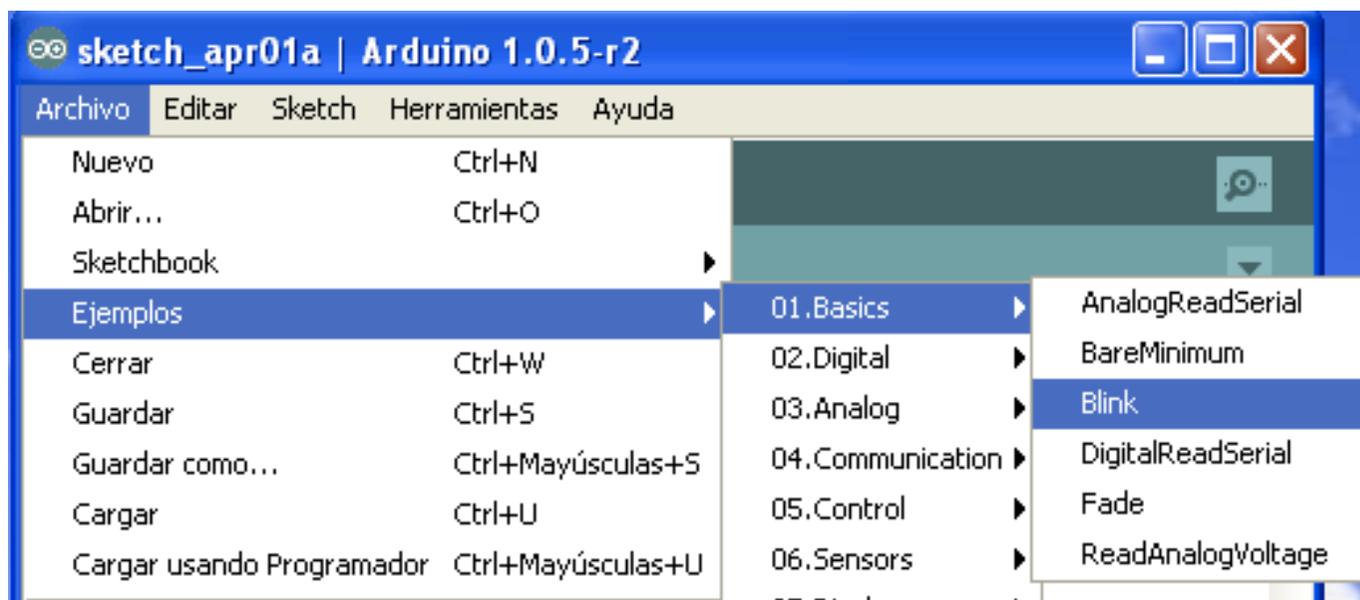
- Si en el ordenador no están instaladas las herramientas de Arduino se deben descargar.
  - Descargar Arduino 1.6.0 de <http://arduino.cc/en/pmwiki.php?n=main/software>
  - Seguir las instrucciones proporcionadas por Arduino:
    - En inglés: <http://arduino.cc/en/Guide/Windows>
    - En castellano: <http://arduino.cc/es/Guide/Windows#>
- Resumen de instrucciones:
  - Descomprimir el fichero descargado en el directorio donde se quiera tener la aplicación
  - Conectar la tarjeta al ordenador con el cable USB
  - Instalar el Driver
    - Para Windows 7 y versiones anteriores.
      - <http://www.wch.cn/downloads/downloadfile.php?id=65>
  - Ejecutar la aplicación Arduino

# Cargando el primer programa

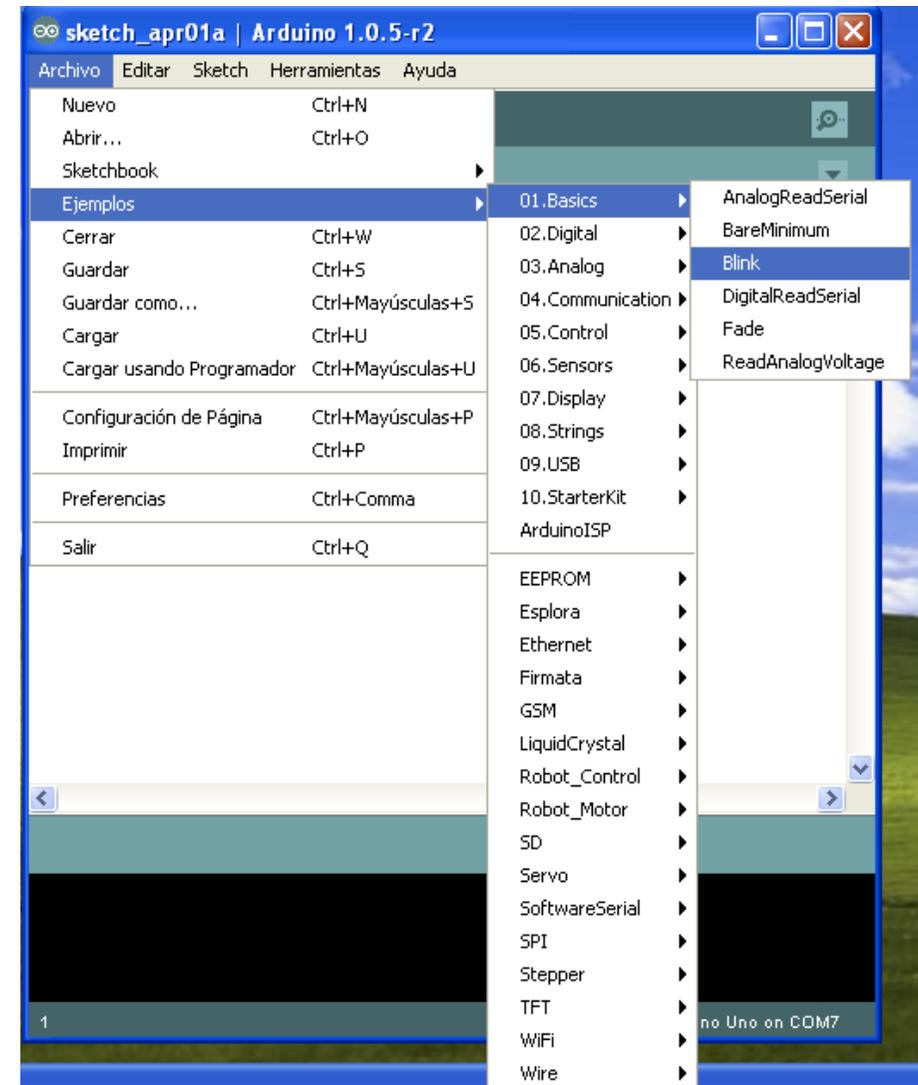


## □ Actividad

- Conecta la tarjeta al ordenador por USB
- Ejecuta el entorno de programación de Arduino
- Carga el ejemplo "Blink" (parpadeo)
  - Archivo → Ejemplos → 0.1 Basics → Blink
- Verifica y descarga el programa
- Comprobar que el LED de la tarjeta parpadea.

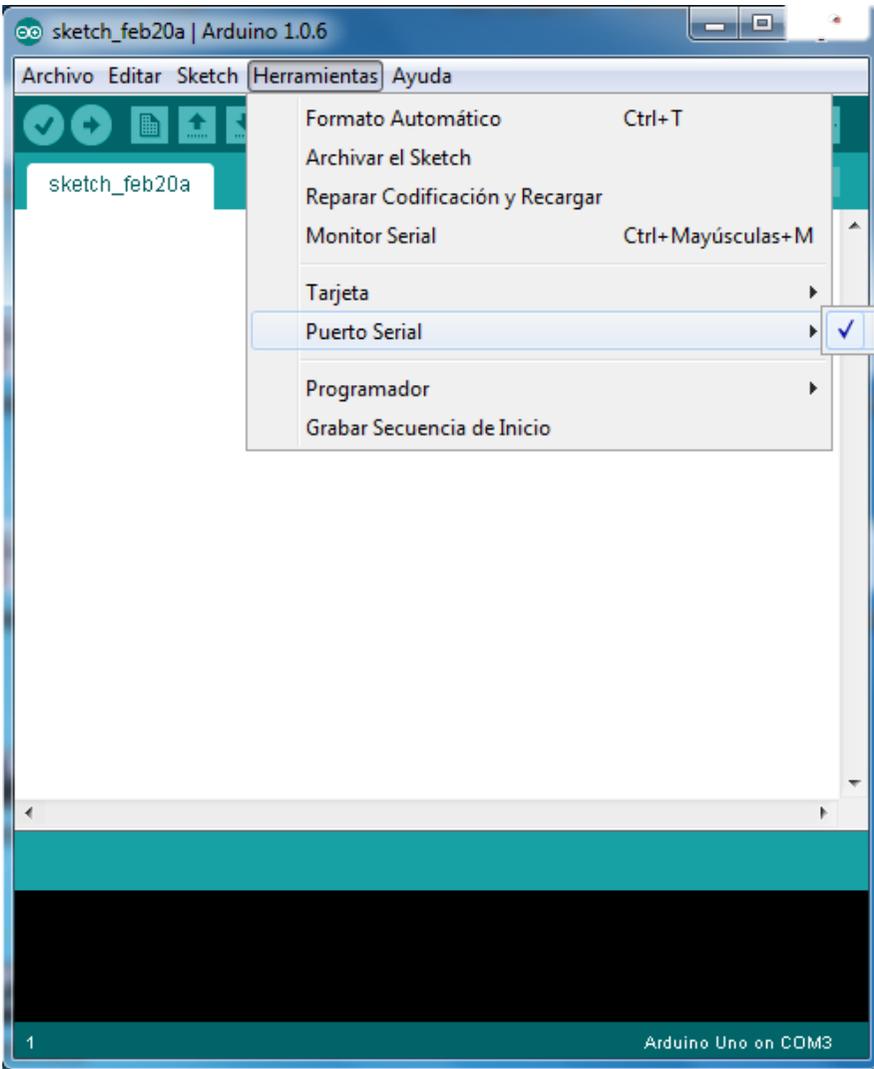


# Cargando el primer programa



-  Verificar que el programa es correcto
-  Cargar el programa verificado en la placa
-  Crear un nuevo programa
-  Abrir un programa existente
-  Guardar las modificaciones del programa

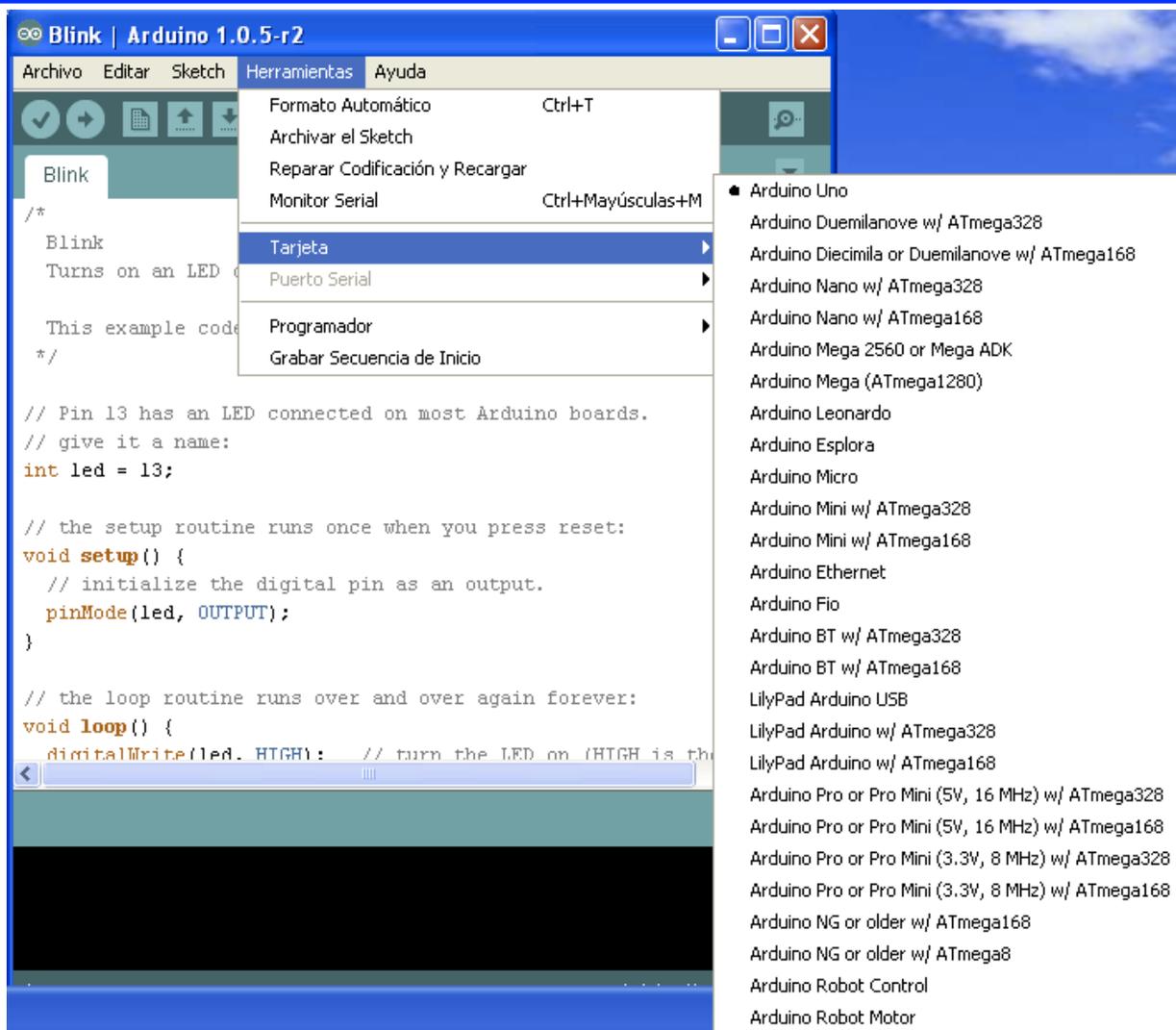
# Cargando el primer programa



The screenshot shows the Arduino IDE window titled 'sketch\_feb20a | Arduino 1.0.6'. The 'Herramientas' menu is open, displaying options such as 'Formato Automático', 'Archivar el Sketch', 'Reparar Codificación y Recargar', 'Monitor Serial', 'Tarjeta', 'Puerto Serial', 'Programador', and 'Grabar Secuencia de Inicio'. The 'Puerto Serial' option is selected, and a sub-menu is visible showing 'COM3' as the chosen port. The status bar at the bottom indicates '1' and 'Arduino Uno on COM3'.

**Verifica que el Puerto Serial coincide con el puerto que está conectada la tarjeta.**

# Cargando el primer programa



# Aprendiendo a programar

- Estructura básica de un programa de Arduino
  - `setup()`
    - Se ejecuta sólo una vez al principio (al encenderlo o al cargarlo)
    - Se utiliza para configurar las entradas y salidas, para inicializar variables, ...
  - `loop()`
    - Se ejecuta continuamente sin fin

```
void setup() {  
    // put your setup code here, to run once:  
}  
  
void loop() {  
    // put your main code here, to run repeatedly:  
}
```

Fíjate en la sintaxis (llaves y paréntesis)

# Aprendiendo a programar



## □ Ejemplo “Blink” (“Parpadeo”)

```
// Pin 13 has an LED connected on most Arduino boards.  
// give it a name:  
int led = 13;  
  
// the setup routine runs once when you press reset:  
void setup() {  
    // initialize the digital pin as an output.  
    pinMode(led, OUTPUT);  
}  
  
// the loop routine runs over and over again forever:  
void loop() {  
    digitalWrite(led, HIGH);    // turn the LED on (HIGH is the voltage level)  
    delay(1000);                // wait for a second  
    digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW  
    delay(1000);                // wait for a second  
}
```

# Aprendiendo a programar

- Ejemplo "Blink" ("Parpadeo")
  - Se identifican claramente el "setup()" y el "loop()"

```
// Pin 13 has an LED connected on most Arduino boards.  
// give it a name:  
int led = 13;
```

```
// the setup routine runs once when you press reset:  
void setup() {  
  // initialize the digital pin as an output.  
  pinMode(led, OUTPUT);  
}
```

```
// the loop routine runs over and over again forever:  
void loop() {  
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)  
  delay(1000);             // wait for a second  
  digitalWrite(led, LOW);  // turn the LED off by making the voltage LOW  
  delay(1000);             // wait for a second  
}
```

# Aprendiendo a programar

- Ejemplo "Blink" ("Parpadeo")
  - Se identifican claramente el "setup()" y el "loop()"

```
// Pin 13 has an LED connected  
// give it a name:  
int led = 13;
```

```
// the setup routine runs once  
void setup() {  
  // initialize the digital pin  
  pinMode(led, OUTPUT);  
}
```

```
// the loop routine runs over and over again forever:  
void loop() {  
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)  
  delay(1000); // wait for a second  
  digitalWrite(led, LOW); // turn the LED off by making the voltage LOW  
  delay(1000); // wait for a second  
}
```

## Esto es una variable:

- Es un elemento del programa que almacena datos (es un contenedor de datos)
- Los datos almacenados pueden cambiar
- Hay variables de diferentes tipos
- En este caso se está creando una variable llamada "led" de tipo "int" a la que se asigna el número 13

# Aprendiendo a programar

## □ Ejemplo "Blink" ("Parpadeo")

- Se identifican claramente el "setup()" y el "loop()"

```
// Pin 13 has an LED connected  
// give it a name:  
int led = 13;
```

```
// the setup routine runs once  
void setup() {  
  // initialize the digital pin  
  pinMode(led, OUTPUT);  
}
```

```
// the loop routine runs over and over again forever:  
void loop() {  
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)  
  delay(1000); // wait for a second  
  digitalWrite(led, LOW); // turn the LED off by making the voltage LOW  
  delay(1000); // wait for a second  
}
```

### Esta es la orden para configurar un pin digital

- **pinMode** es una función con dos parámetros
  - El primer parámetro es el número de la entrada o salida digital a configurar
  - El segundo parámetro es:
    - OUTPUT → Para configurarlo como salida
    - INPUT → Para configurarlo como entrada
- Esta función ejecuta el pin 13 como salida.

# Aprendiendo a programar

## □ Ejemplo "Blink" ("Parpadeo")

- Se identifican claramente el "setup()" y el "loop()"

```
// Pin 13 has an LED connected on most Arduino boards.  
// give it a name:  
int led = 13;
```

```
// the setup routine runs once when you press reset:  
void setup() {  
  // initialize the digital pin  
  pinMode(led, OUTPUT);  
}
```

```
// the loop routine runs over  
void loop() {  
  digitalWrite(led, HIGH);  
  delay(1000);  
  digitalWrite(led, LOW);  
  delay(1000);  
}
```

### Esta orden cambia el valor de una salida digital

- **digitalWrite** esta función tiene dos parámetros:
  - El primer parámetro es el número de la salida digital que se quiere modificar
  - El segundo parámetro es:
    - HIGH → Para ponerla a nivel alto ('1' ó 5V)
    - LOW → Para ponerla a nivel bajo ('0' ó 0V)
- Estas funciones actúan sobre la salida digital 13 poniéndola primero a nivel alto (5V) y luego a nivel bajo (0V).

# Aprendiendo a programar



- Ejemplo "Blink" ("Parpadeo")
  - Se identifican claramente el "setup()" y el "loop()"

```
// Pin 13 has an LED connected on most Arduino boards.  
// give it a name:  
int led = 13;
```

```
// the setup routine runs once when you press reset:  
void setup() {  
  // initialize the digital pin as an output.  
  pinMode(led, OUTPUT);  
}
```

```
// the loop routine runs over  
void loop() {  
  digitalWrite(led, HIGH);  
  delay(1000);  
  digitalWrite(led, LOW);  
  delay(1000);  
}
```

## Esta orden introduce una pequeña espera

- **Delay** esta función tiene un parámetro:
  - Como parámetro se introduce el tiempo de espera en milisegundos
    - 1000 → 1000ms = 1 segundo



## □ Ejercicios:

1. Modifica el programa para que el LED parpadee más rápido
  - Escribe y verifica el programa y cárgalo en la placa
2. Modifica el programa para que repita una secuencia de dos parpadeos rápidos consecutivos seguidos de un tiempo de apagado
  - Escribe y verifica el programa y cárgalo en la placa

# Aprendiendo a programar

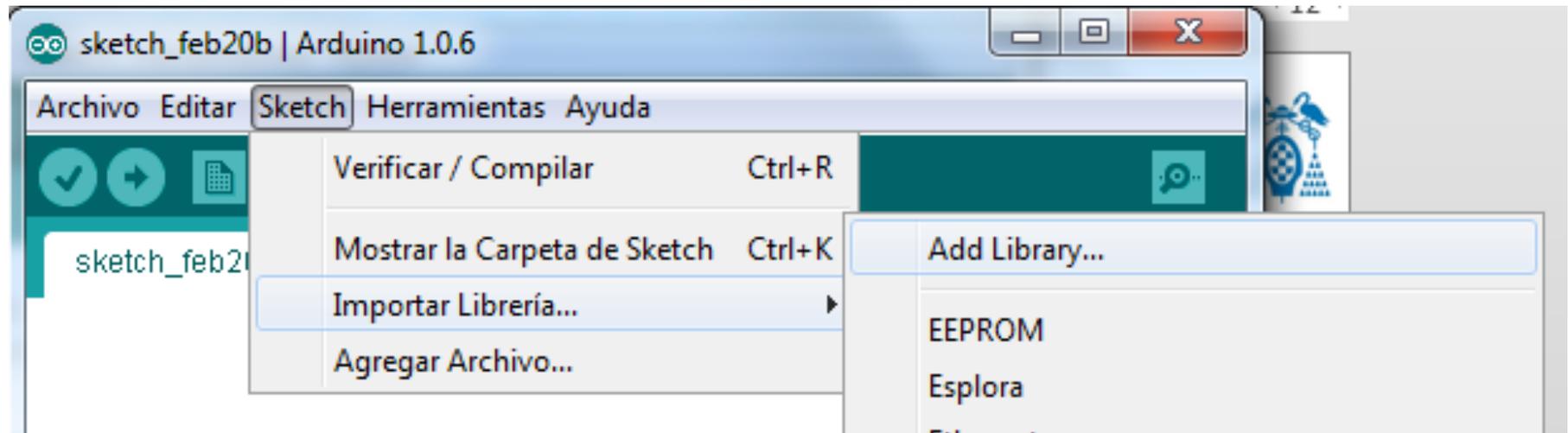
---

- Importar librerías (Bibliotecas de funciones):
  - Una librería es un “conjunto de comandos”.
  - Para poder usar una librería primero debemos importarla
  - Nosotros deberemos importar 2 librerías
  
- 1) `DistanceSRF04`:
  - Contiene los comandos para controlar el sensor de distancia SRF04.
  
- 2) `motor_lib`:
  - Contiene los comandos para controlar los motores y los ejemplos de este taller de tubot.

# Aprendiendo a programar



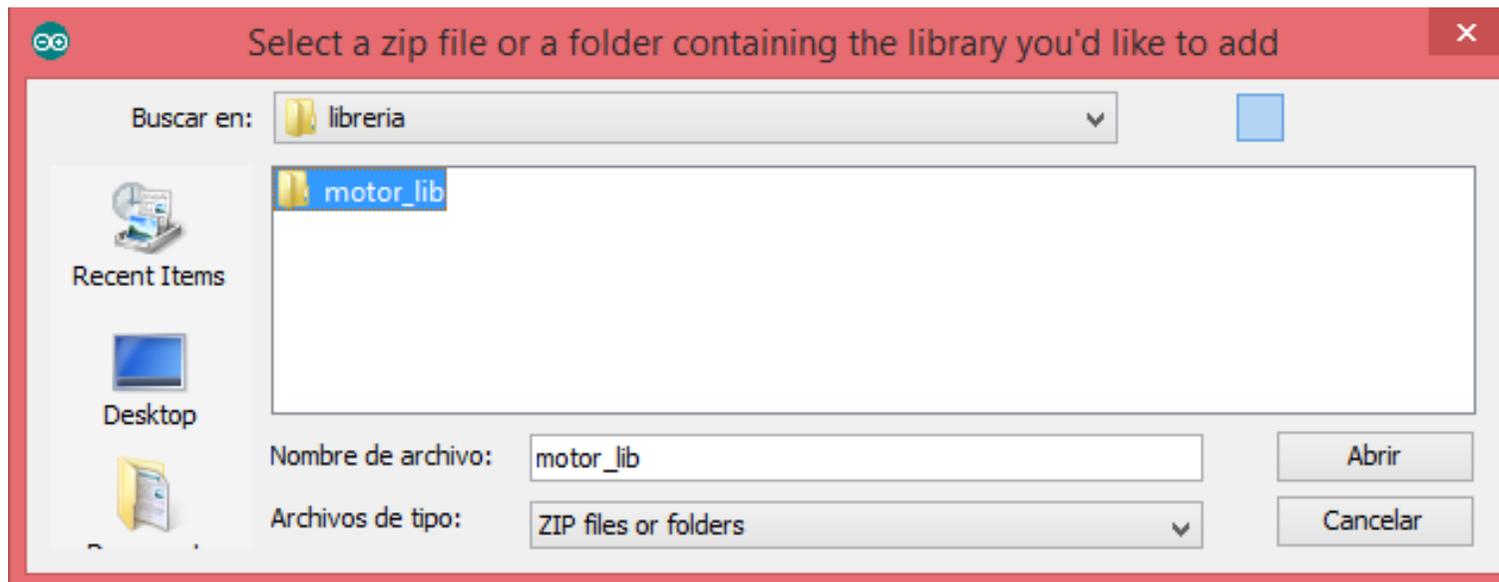
- Importar librerías:
  - Para importar una librería nos vamos al menú de Arduino y seleccionamos **Sketch > Importar Librería > Add Library**



# Aprendiendo a programar



- Importar librerías:
  - Aparecerá una ventana donde debemos **seleccionar el archivo comprimido .zip** donde se encuentra la librería
  - Seleccionamos la carpeta con la librería, y le damos a **“abrir”**

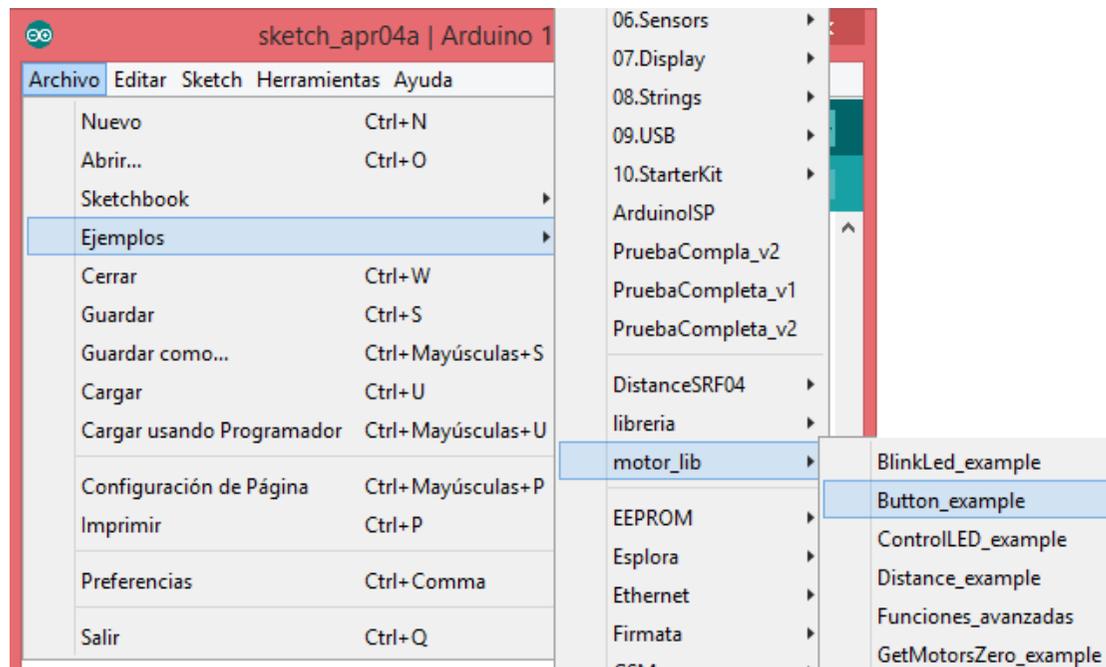


# Aprendiendo a programar



## □ Importar librerías:

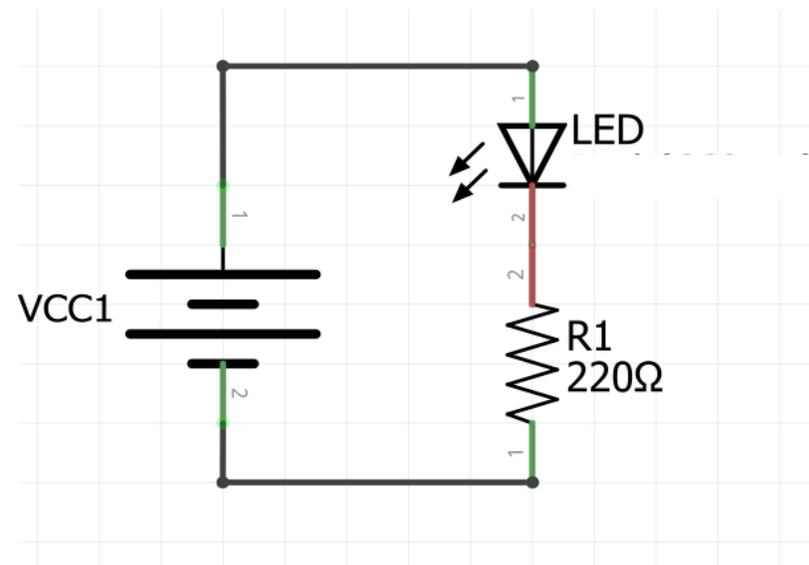
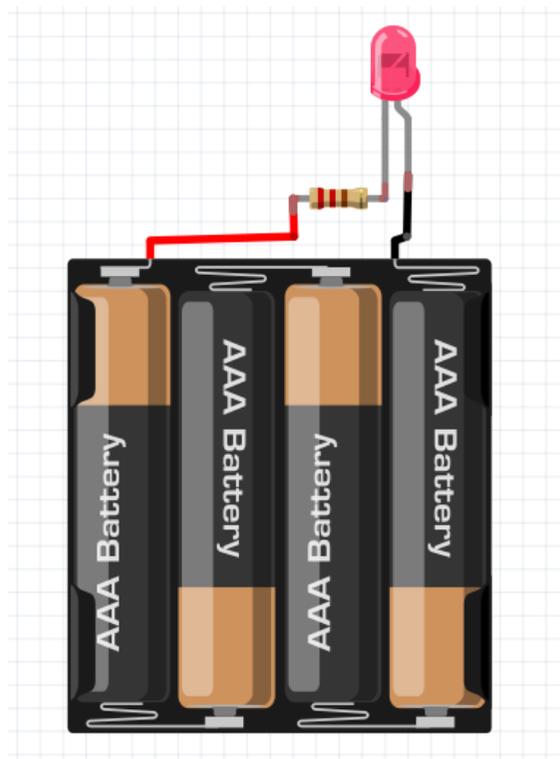
- Una vez hecho esto ya tenemos la librería importada y la podemos utilizar incluyendo en nuestro proyecto el archivo:
  - `#include <Nombre_librería.h>`
- Cada librería incluye además una serie de ejemplos a los que podemos acceder a través del menú de Arduino.



# Entendiendo el funcionamiento de un LED



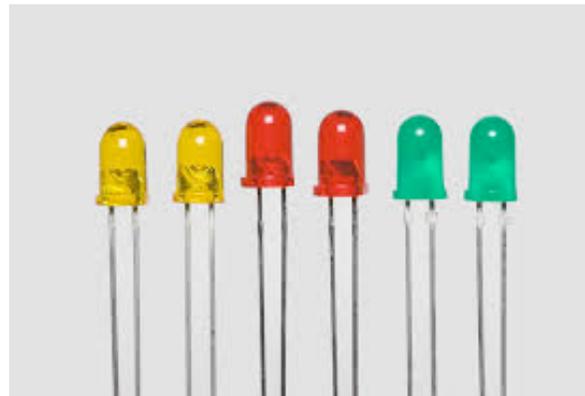
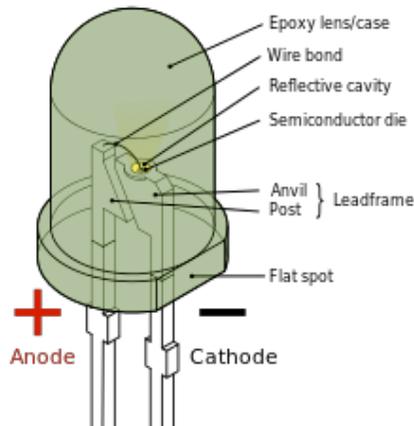
- El circuito eléctrico
  - Cuando se cierra un circuito eléctrico, circula una corriente del polo positivo al negativo de la batería.



# Entendiendo el funcionamiento de un LED



- Un LED (Light Emitting Diode – Diodo emisor de luz)
  - Dispositivo que deja pasar la corriente en un sentido (en el otro no)
  - Cuanta más corriente pasa más luce.



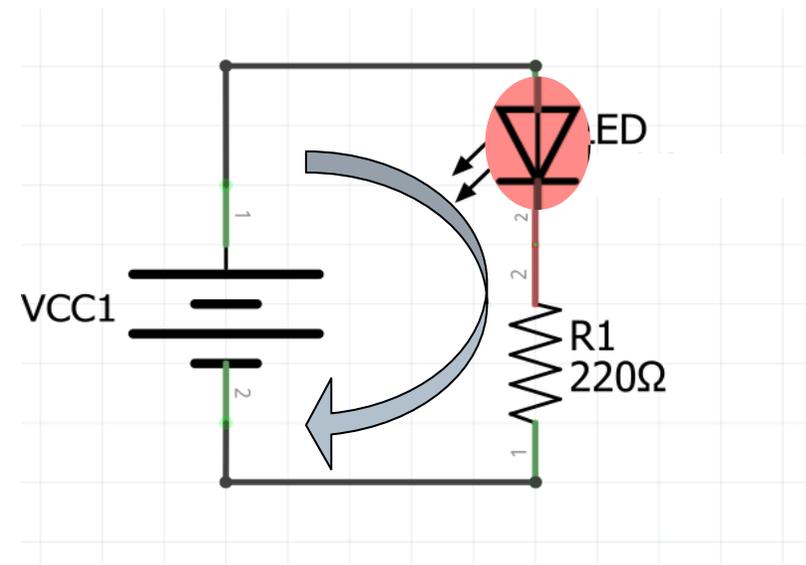
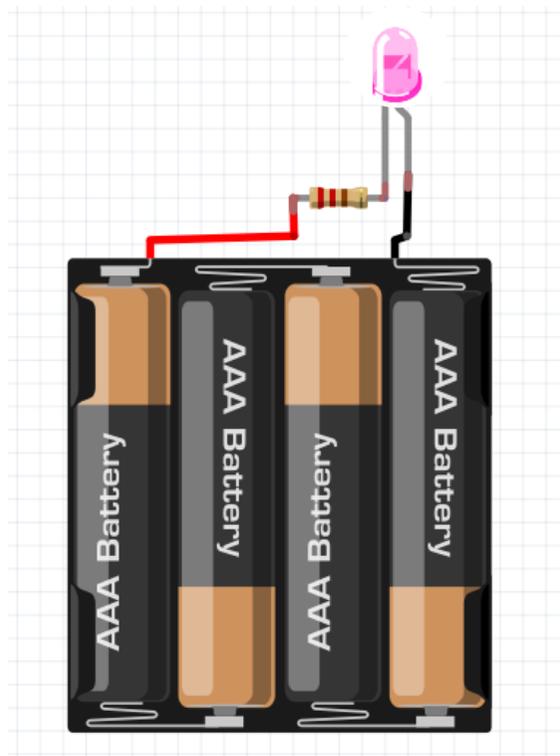
Source: [http://en.wikipedia.org/wiki/Light-emitting\\_diode](http://en.wikipedia.org/wiki/Light-emitting_diode)  
<http://earth911.com/news/2013/02/12/cfl-led-choose-light-bulb-options/>  
<http://www.ecotownstore.com/led/flood/transportation.html>

# Entendiendo el funcionamiento de un LED



## □ Circuito de un LED

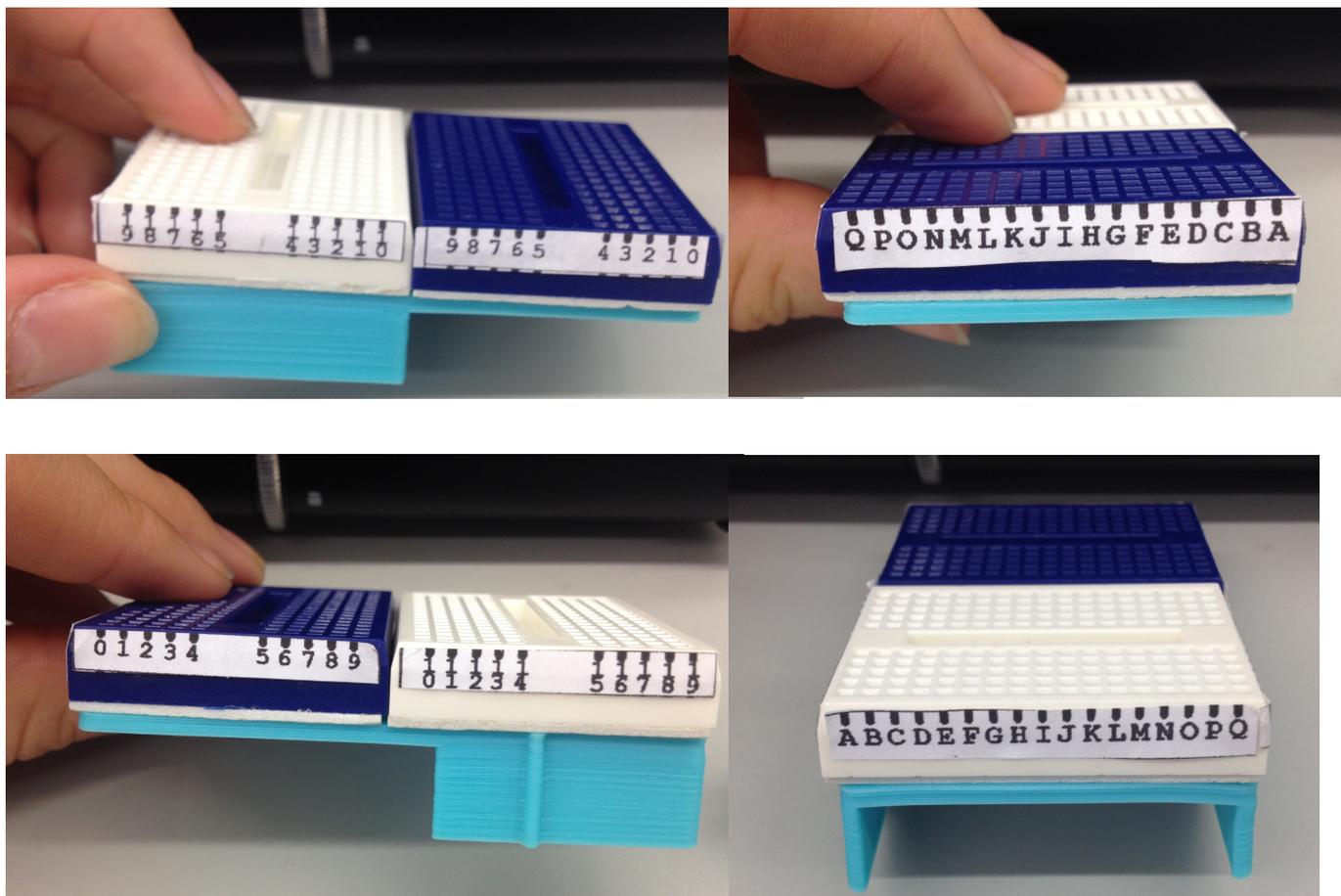
- El LED tiene polaridad. Sólo luce cuando pasa corriente por él, y sólo pasa corriente en un sentido.
- Se suele poner en serie con una resistencia para limitar la corriente máxima que pasa por él.



# Preparación de la ProtoBoard



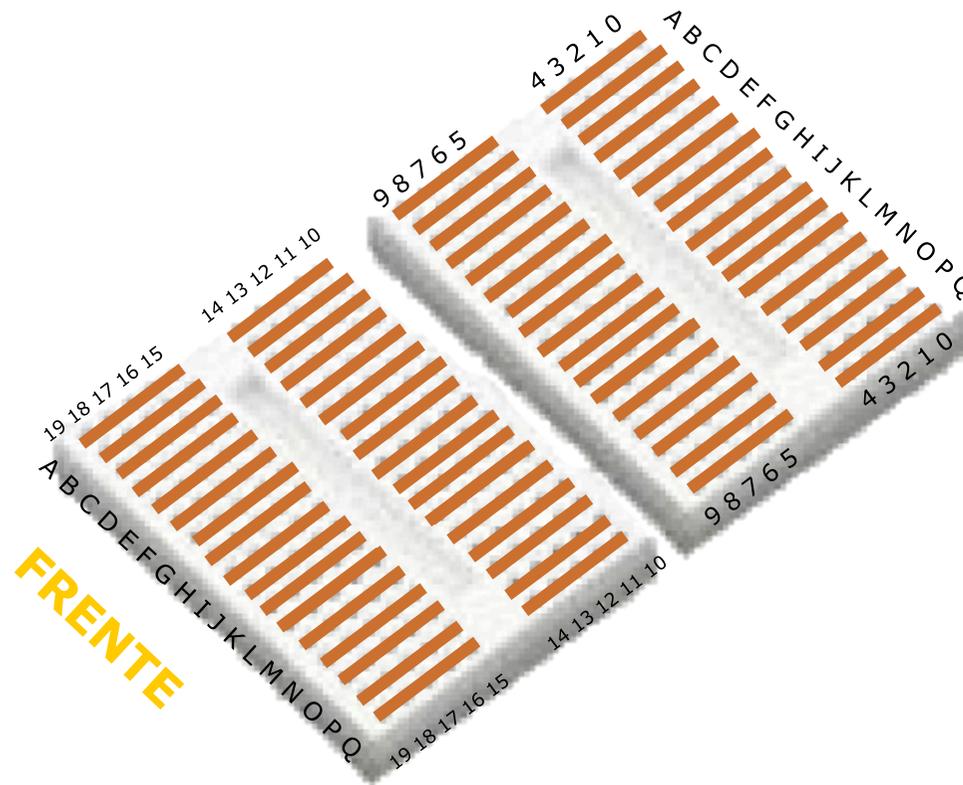
- Las pegatinas están para orientarnos con las conexiones.



# Entendiendo el funcionamiento de un LED



- Placa de prototipado rápido (BreadBoard o ProtoBoard)
  - Utilizada para hacer conexiones eléctricas sencillas.

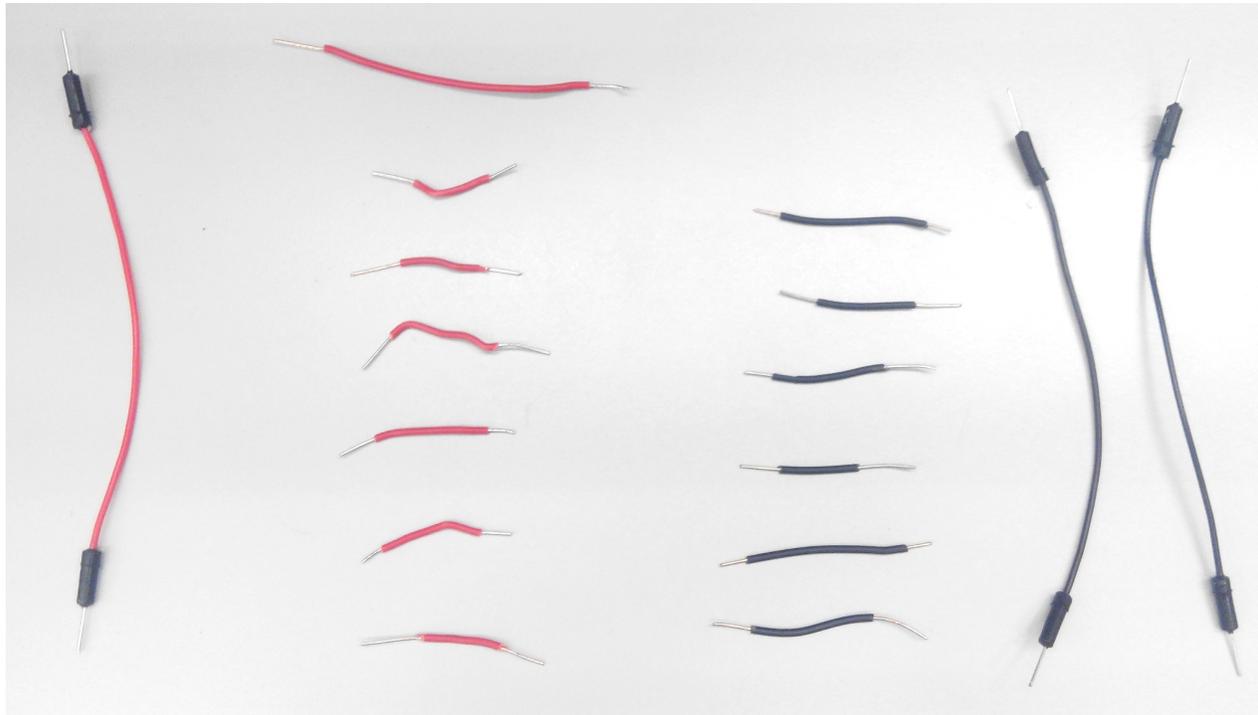


- Las filas de 5 cuadros de los laterales están conectadas internamente entre sí

# Materiales para la preparación de la ProtoBoard



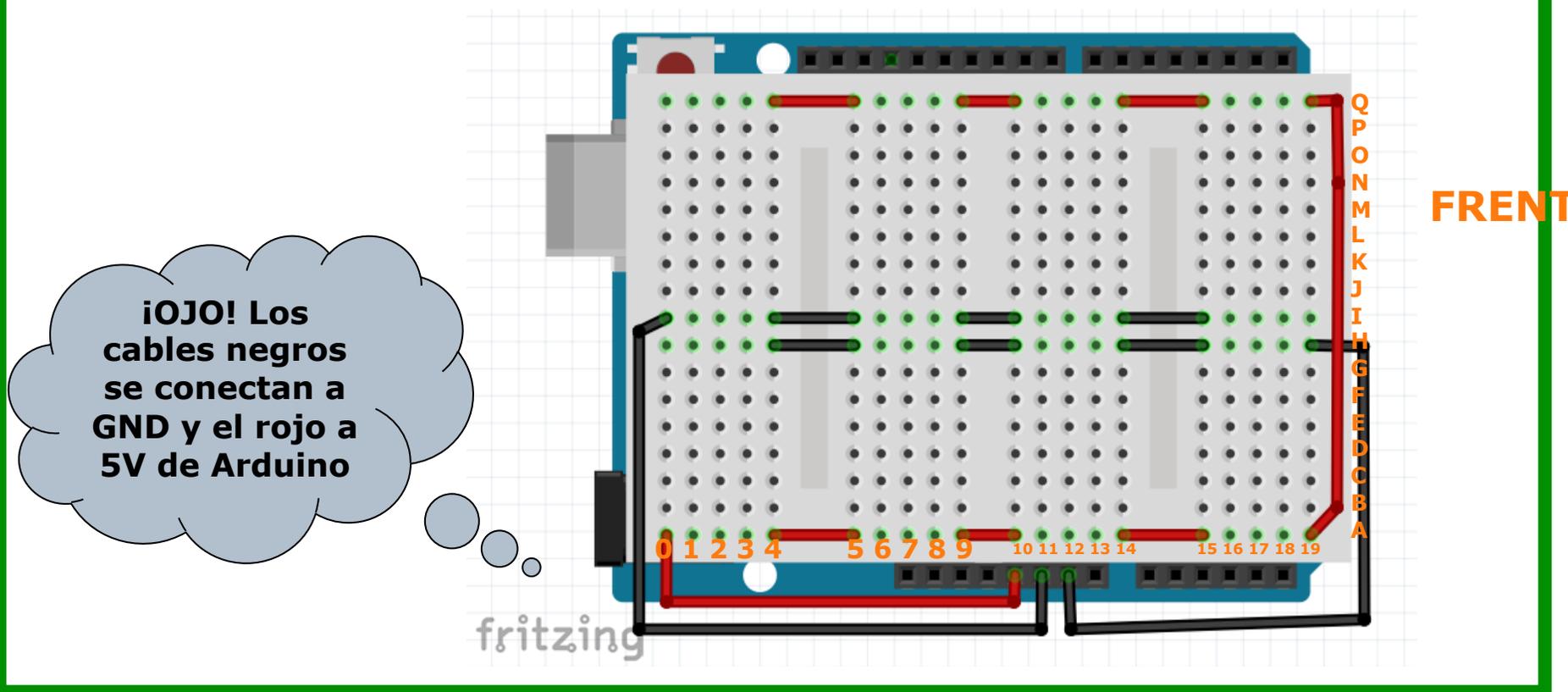
## □ Materiales



# Preparación de la ProtoBoard

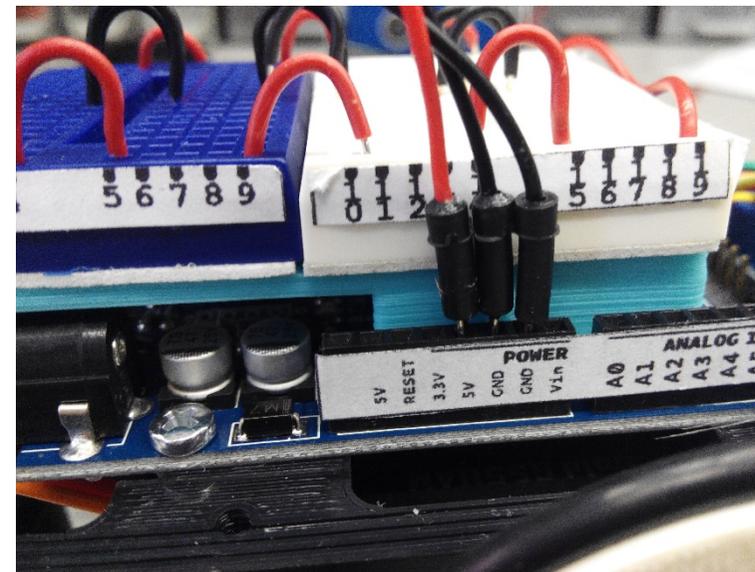
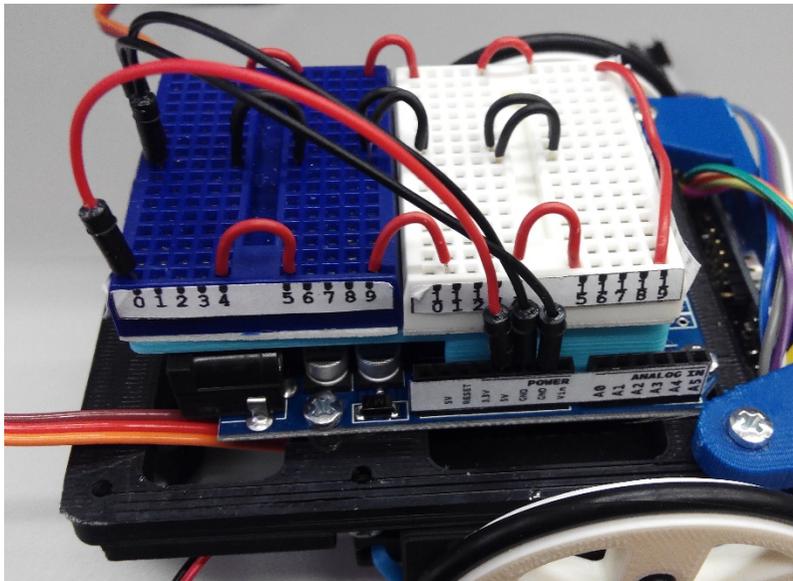


- Para ir montando progresivamente la electrónica, es necesario partir de unas conexiones iniciales.
  - Realiza las conexiones que se indican en la figura.



# Preparación de la ProtoBoard

- Para ir montando progresivamente la electrónica, es necesario partir de unas conexiones iniciales.
  - Realiza las conexiones que se indican en la figura.



# Conexión de un LED

## □ Materiales



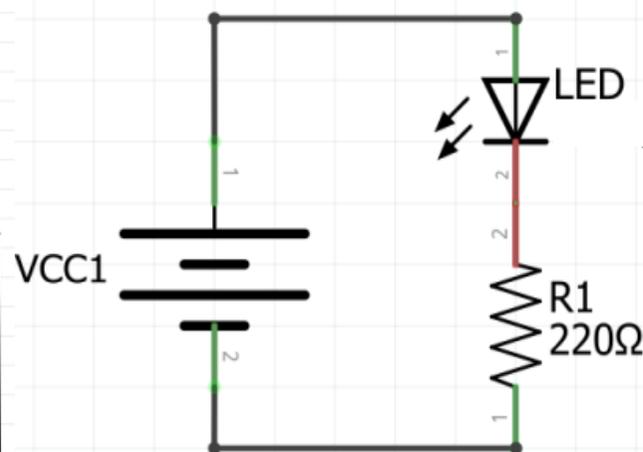
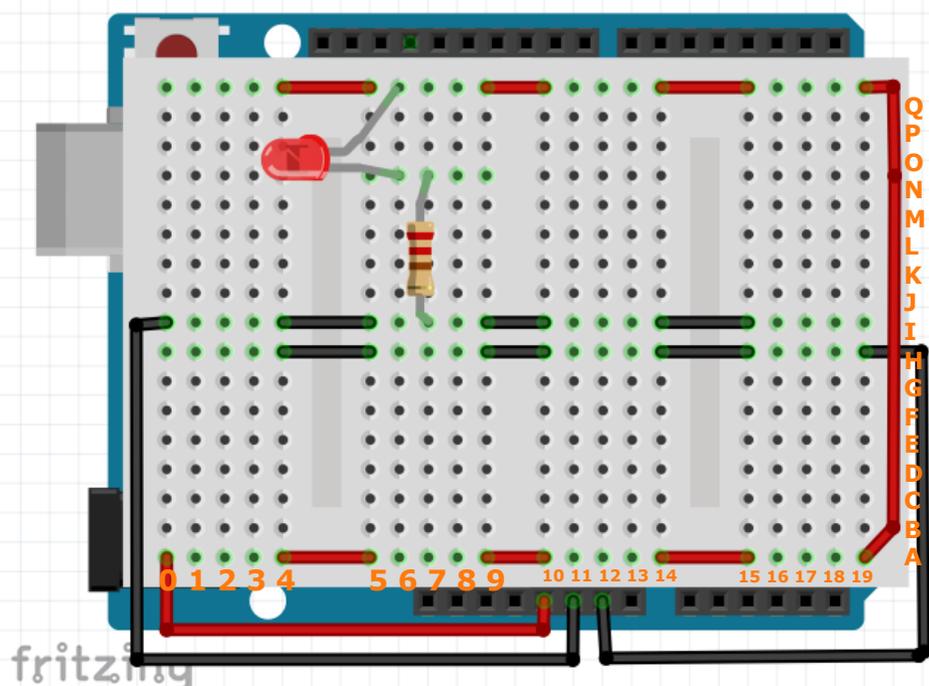
Resistencia de  $330\Omega$

**Rojo** – **Rojo** – **Marrón** – **Oro**

# Conexión de un LED

## □ Pasos:

- Paso 1: Conectar la resistencia de  $220\Omega$  entre I - 7 y N - 7.
- Paso 2: Conectar el LED entre N - 6 y Q - 6
- Paso 3: Conectar el cable USB al ordenador

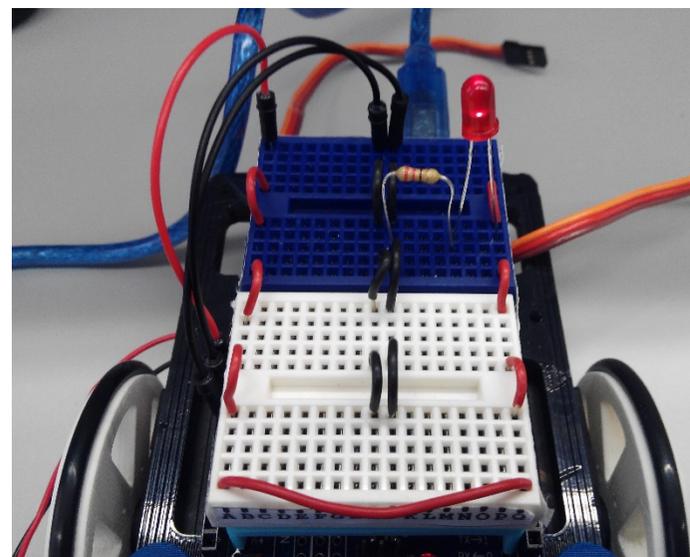
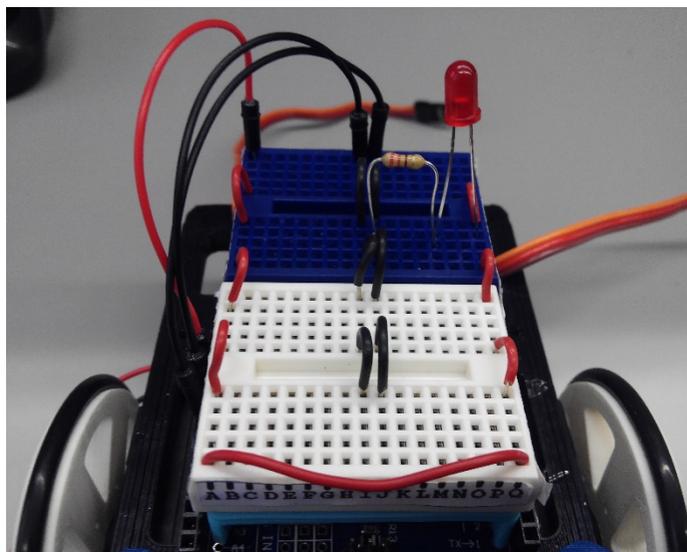


# Conexión de un LED



## □ Resultado:

- El LED debe encenderse
- Si no se enciende, gira el LED intercambiando sus patas (posiblemente esté al revés)

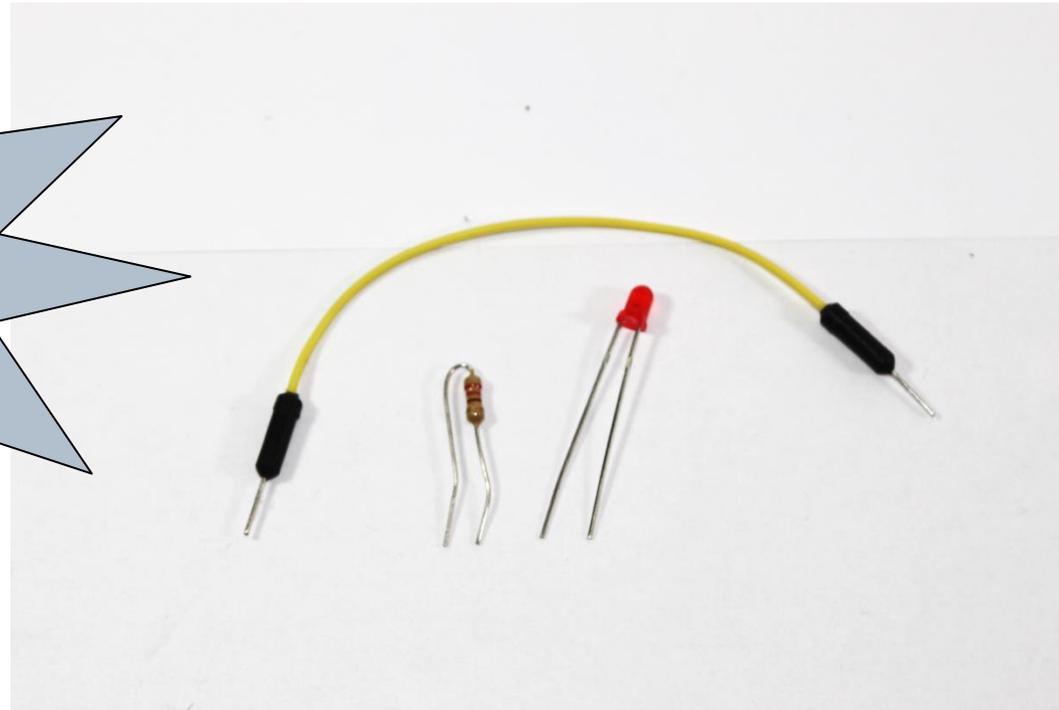


# Conexión de un LED a un PIN de Arduino



## □ Materiales

**Desconecta el  
Arduino a la hora  
de hacer el montaje  
y desmontaje de los  
componentes.**



Resistencia de  $330\Omega$

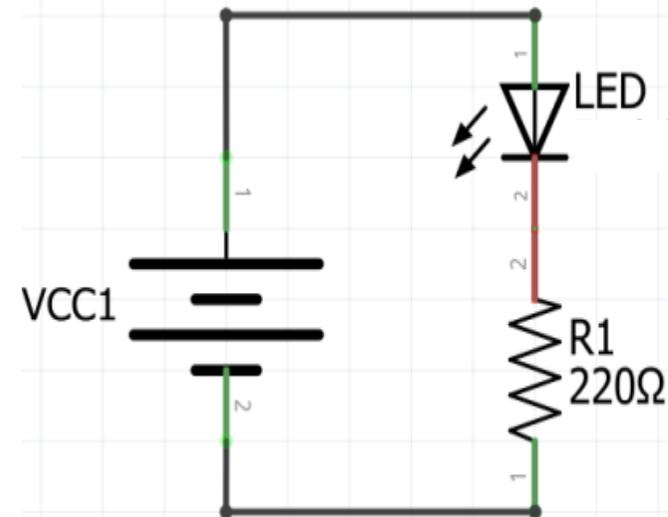
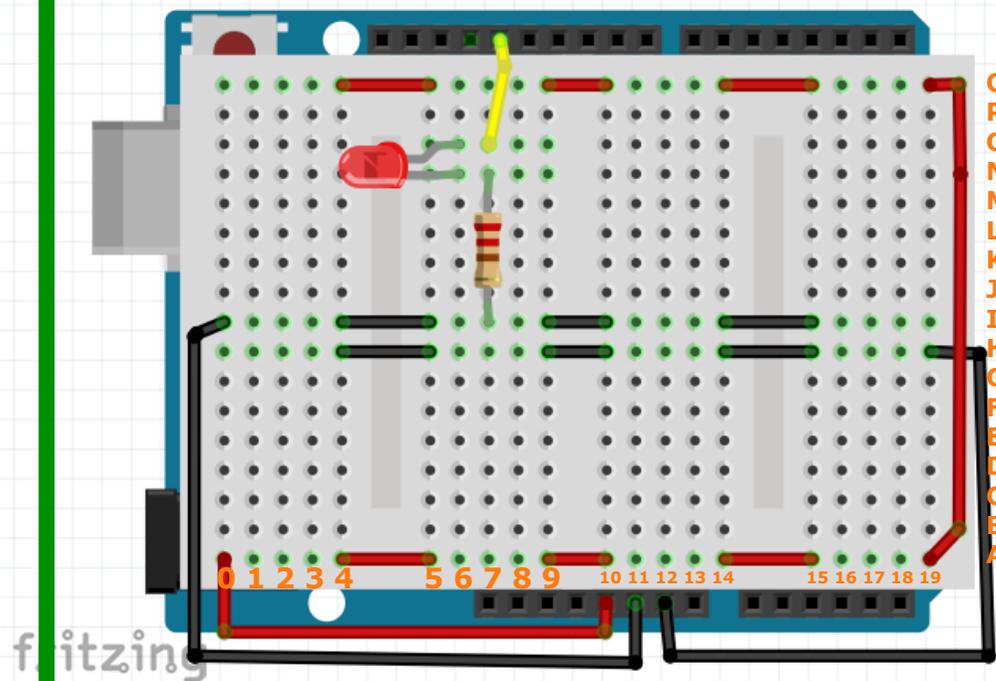
**Rojo - Rojo - Marrón - Oro**

# Conexión de un LED al PIN 13



## □ Pasos:

- Paso 1: Conectar el LED al PIN 13 en vez de a 5V
- Paso 2: Cargar el programa "BlinkLed\_example"
- Resultado: El LED debe parpadear

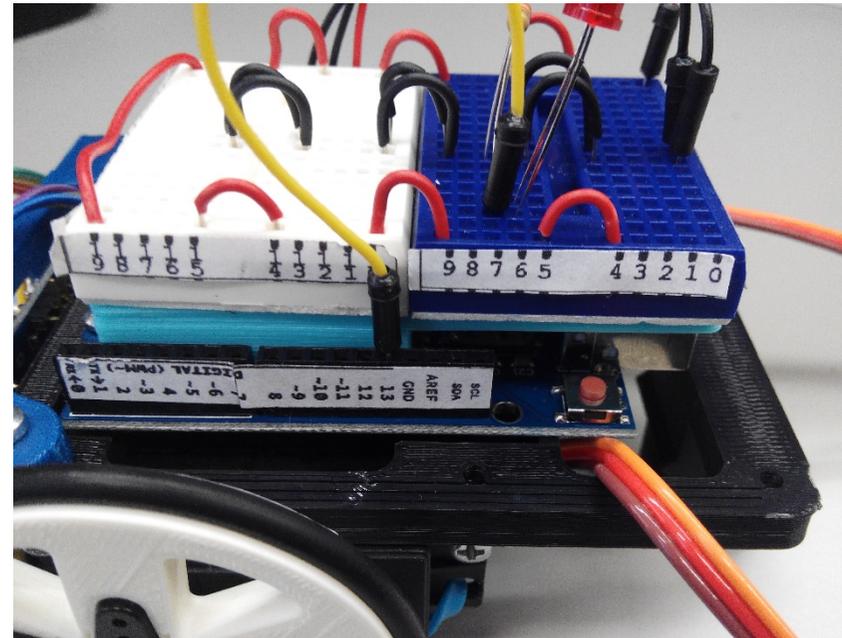
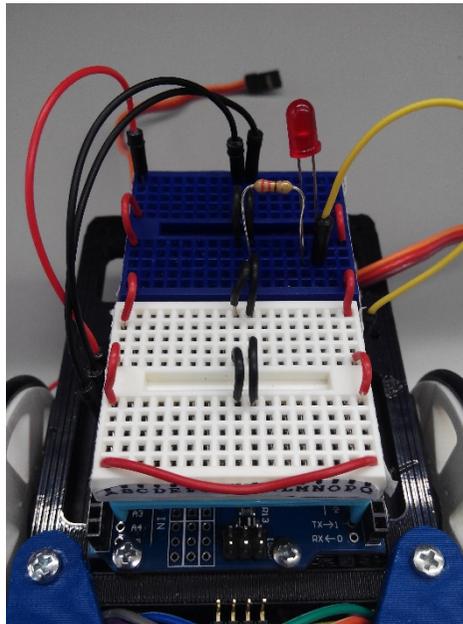


# Conexión de un LED al PIN 13



## □ Pasos:

- Paso 1: Conectar el LED al PIN 13 en vez de a 5V
- Paso 2: Cargar el programa "BlinkLed\_example"
- Resultado: El LED debe parpadear



# Aprendiendo a programar

- Ejemplo: Parpadeo LED externo (BlinkLed\_example).
  - Podemos hacer parpadear un LED conectado a cualquier pin digital.
  - Mismos ejemplo que el de Arduino pero lo podemos modificar.

```
int pinLed = 13;

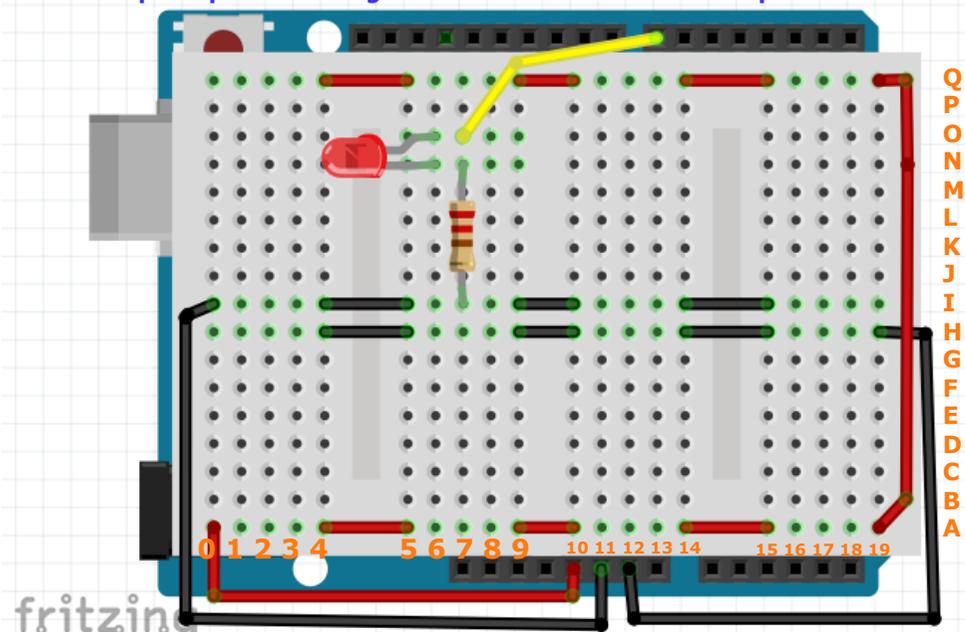
void setup(){
    // Establecemos el pin del LED como salida
    pinMode(pinLed, OUTPUT);
}

void loop(){
    delay(500);
    digitalWrite(pinLed, HIGH);
    delay(500);
    digitalWrite(pinLed, LOW);
}
```

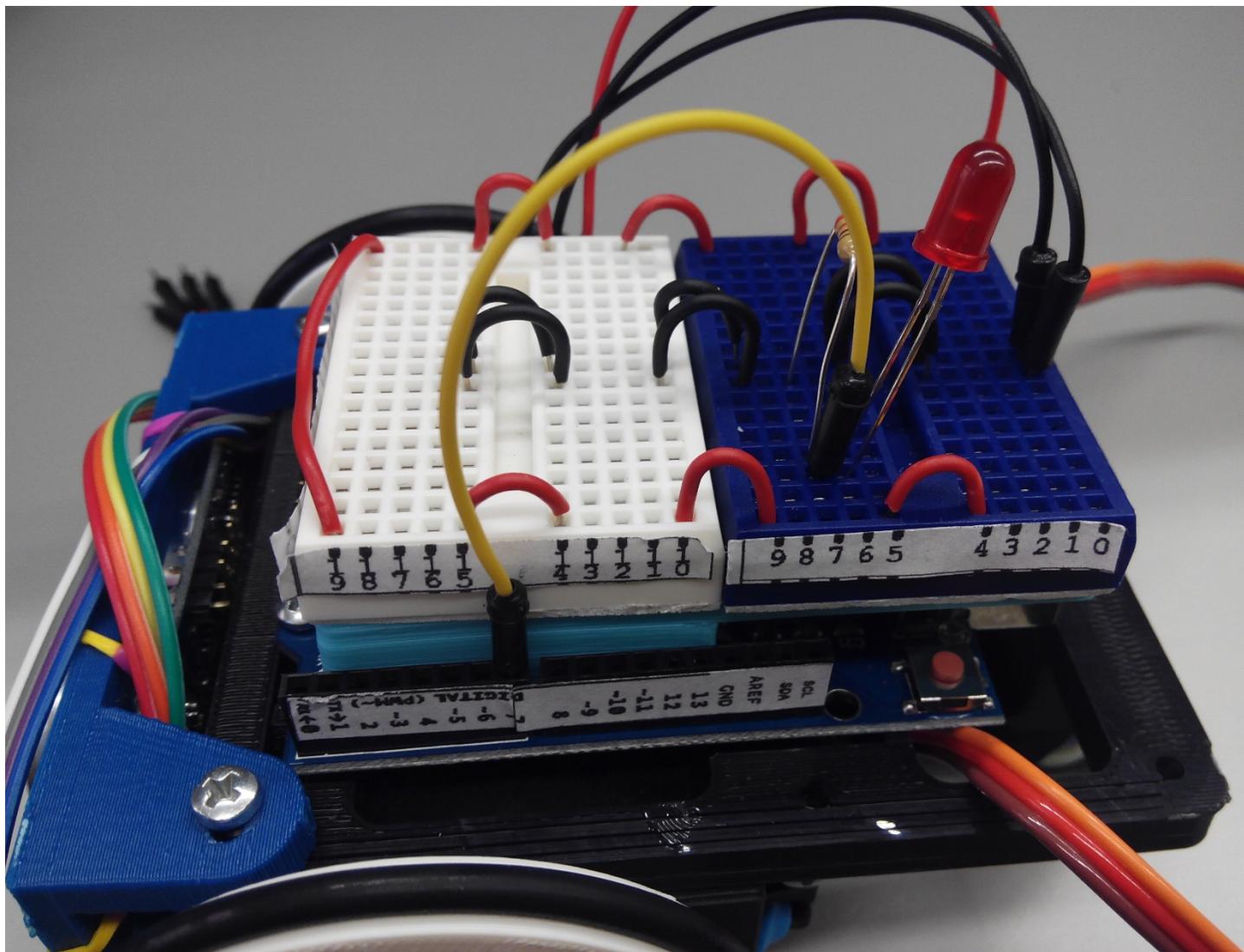
# Conexión de un LED a otro PIN



- ❑ Conecta el LED y la resistencia como se indica en la figura
  - LED entre 0 - 6 y N - 6.
  - Resistencia entre los puntos I - 7 y N -7.
  - Une N - 7 con el PIN 7 del Arduino UNO.
- ❑ Modifica el programa para que parpadee con el pin 7.
- ❑ Resultado:
  - El LED debe parpadear junto con el de la placa.



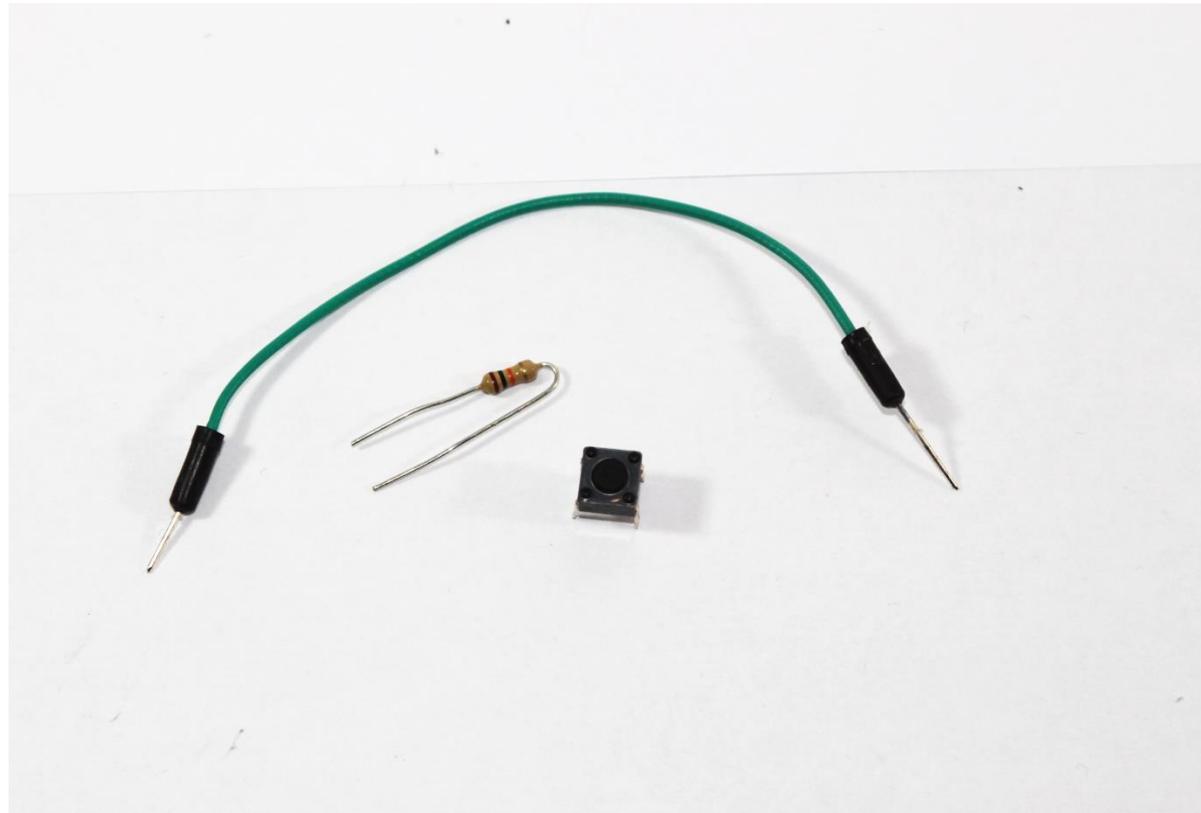
# Conexión de un LED a otro PIN



# Añadimos un pulsador



## □ Materiales



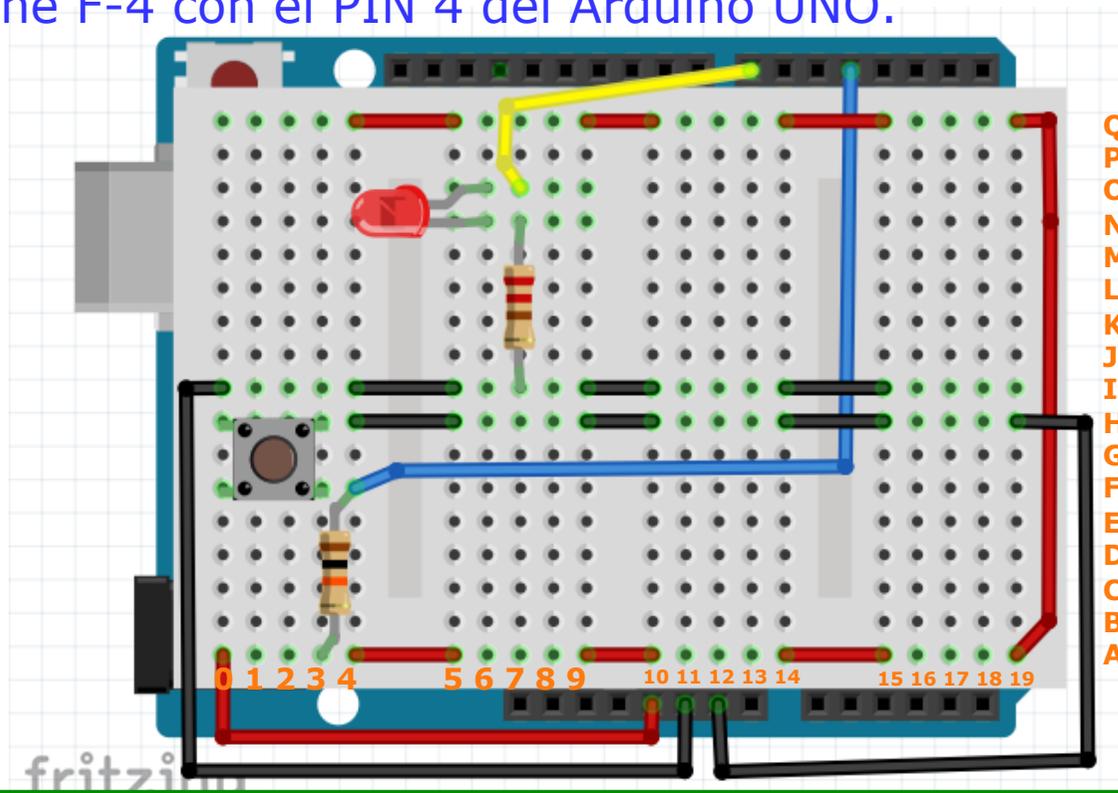
Resistencia de  $10k\Omega$

**Marrón** – **Negro** – **Naranja** – **Oro**

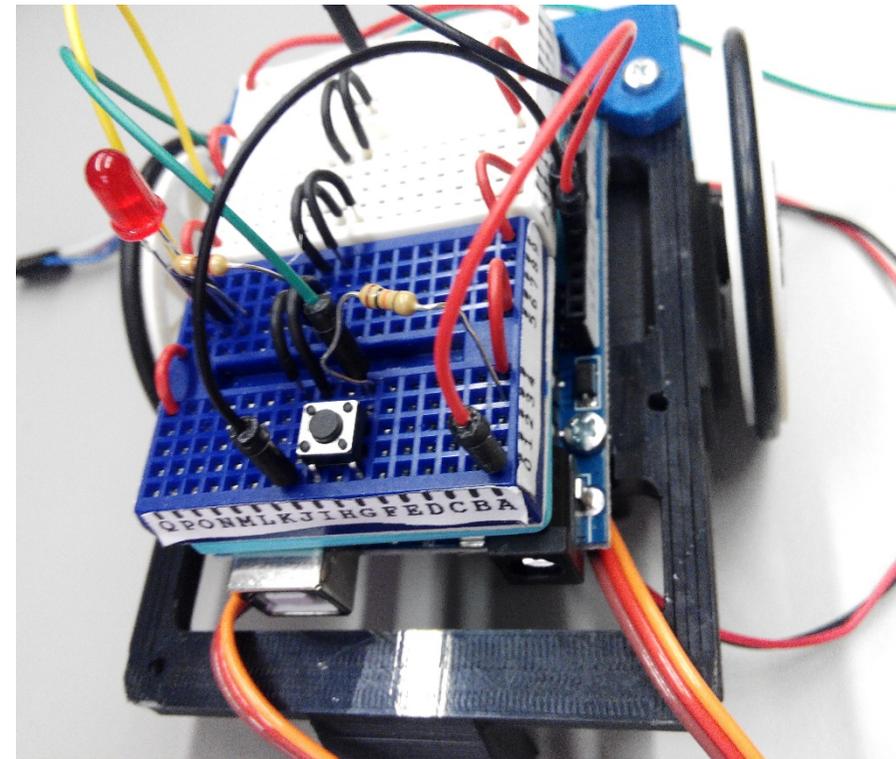
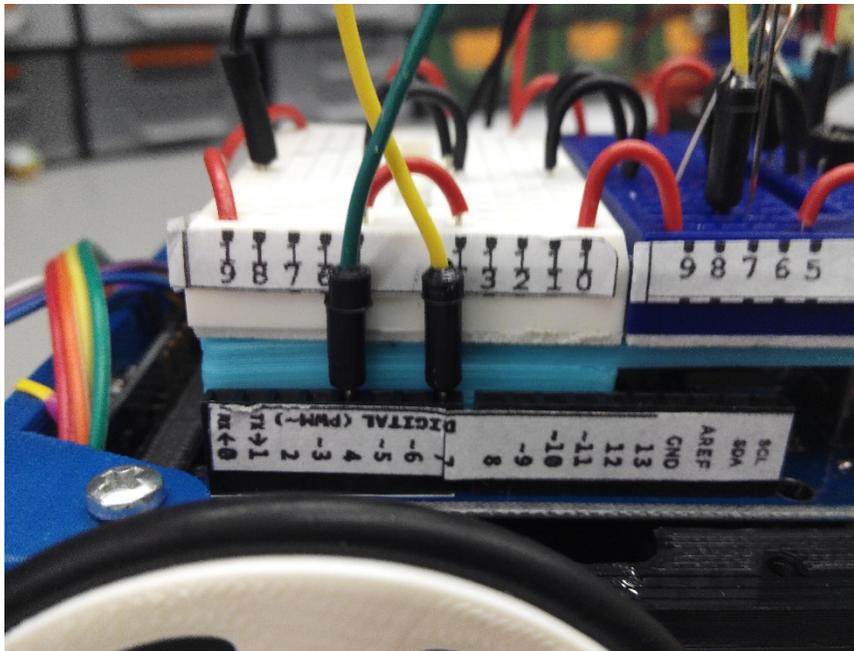
## Añadimos un pulsador



- Conecta el pulsador y la resistencia como se indica en la figura.
  - Pulsador entre H-0 y F-0.
  - Resistencia entre F-4 y A-3.
  - Une F-4 con el PIN 4 del Arduino UNO.



# Añadimos un pulsador



# Añadimos un pulsador

- Encender LED mediante el pulsador.
  - Explicación de los condicionales:
    - IF\_ELSE

```
5 int pinButton = 4;
6 int led = 13;
7 int button; // Variable donde se guarda el estado del botón
8 void setup(){
9     //Configuración de la comunicación serie.
10
11     //Establecemos el pin del botón como entrada.
12     pinMode(pinButton, INPUT);
13     pinMode(led, OUTPUT);
14
15 }
16
17 void loop(){
18     delay(200);
19     // Leemos el valor digital del botón (0 o 1).
20     button = digitalRead(pinButton);
21     if(button==0)
22         digitalWrite(led, HIGH);
23     else
24         digitalWrite(led, LOW);
25
26 }
```

# Aprendiendo a programar

- Ejemplo: Estado de botón externo (Button\_example).
  - En este ejemplo leeremos el estado digital de un botón conectado al pin 4 de la placa Arduino y enviaremos dicho valor por la comunicación serie.

```
int pinButton =      4;
int button; // Variable donde se guarda el estado del botón
void setup(){
    //Configuración de la comunicación serie.
    Serial.begin(19200);
    //Establecemos el pin del botón como entrada.
    pinMode(pinButton, INPUT);

}

void loop(){
    delay(200);
    // Leemos el valor digital del botón (0 o 1).
    button = digitalRead(pinButton);
    // Imprimimos por pantalla el estado del botón.
    Serial.print("Valor boton: ");
    Serial.println(button);
}
```

# Aprendiendo a programar

- Ejemplo: Estado de botón externo (Button\_example).
  - En este ejemplo leeremos el estado digital de un botón conectado al pin 4 de la placa Arduino y enviaremos dicho valor por la comunicación serie.

```
int pinButton = 4;
int button; // Variable donde se guarda el estado del botón
void setup(){
    //Configuración de la comunicación serie.
    Serial.begin(19200);
    //Establecemos el pin del botón como entrada digital.
    pinMode(pinButton, INPUT);
}

void loop(){
    delay(200);
    // Leemos el valor digital del botón
    button = digitalRead(pinButton);
    // Imprimimos por pantalla el estado
    Serial.print("Valor boton: ");
    Serial.println(button);
}
```

Establecemos el pin como entrada digital.

- **digitalRead** Esta función lee el estado digital del pin dado como parámetro.
  - Devuelve el estado del pin:
    - 1 → Sí está a nivel alto ('HIGH' ó 5V)
    - 0 → Sí está a nivel bajo ('LOW' ó 0V)

# Aprendiendo a programar

- Ejemplo: Estado de botón externo (Button\_example).
  - En este ejemplo leeremos el estado digital de un botón conectado al pin 4 de la placa Arduino y enviaremos dicho valor por la comunicación serie.

```
int pinButton = 4;
int button; // Variable donde se guarda el estado del botón
void setup(){
  //Configuración de la comunicación serie
  Serial.begin(19200);
  //Establecemos el pin del botón como entrada.
  pinMode(pinButton, INPUT);
}
```

**Serial.begin(tasa):** Esta función configura la comunicación serie con una velocidad de "tasa" bits por segundo

```
void loop(){
  delay(200);
  // Leemos el valor digital del botón
  button = digitalRead(pinButton);
  // Imprimimos por pantalla el estado del botón
  Serial.print("Valor boton: ");
  Serial.println(button);
}
```

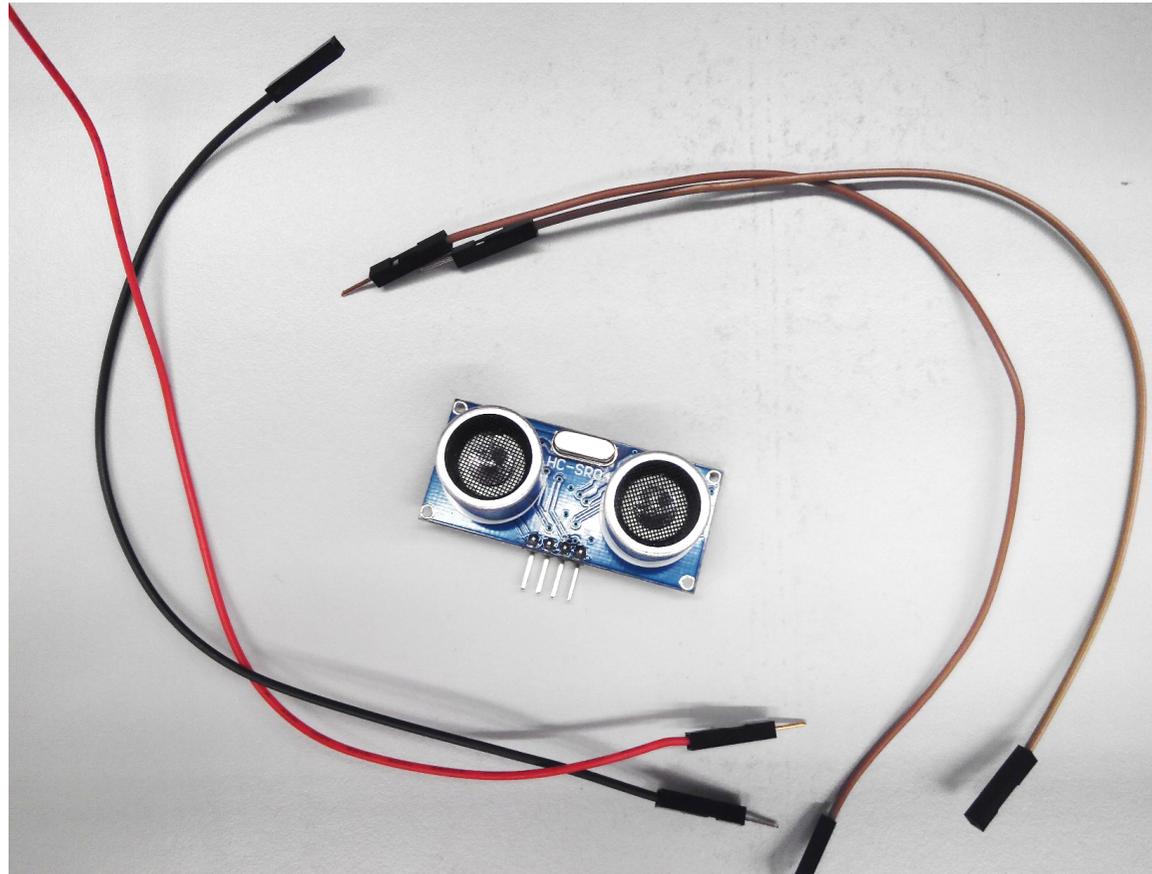
**Serial.print(mensaje):** Esta función envía por la comunicación serie el contenido del parámetro "mensaje".

- "mensaje" puede ser un texto o una variable.
  - Texto: Palabras entre comillas dobles "".
  - Variable: Envía el contenido de la variable

# Jugamos con el sensor de Ultrasonidos



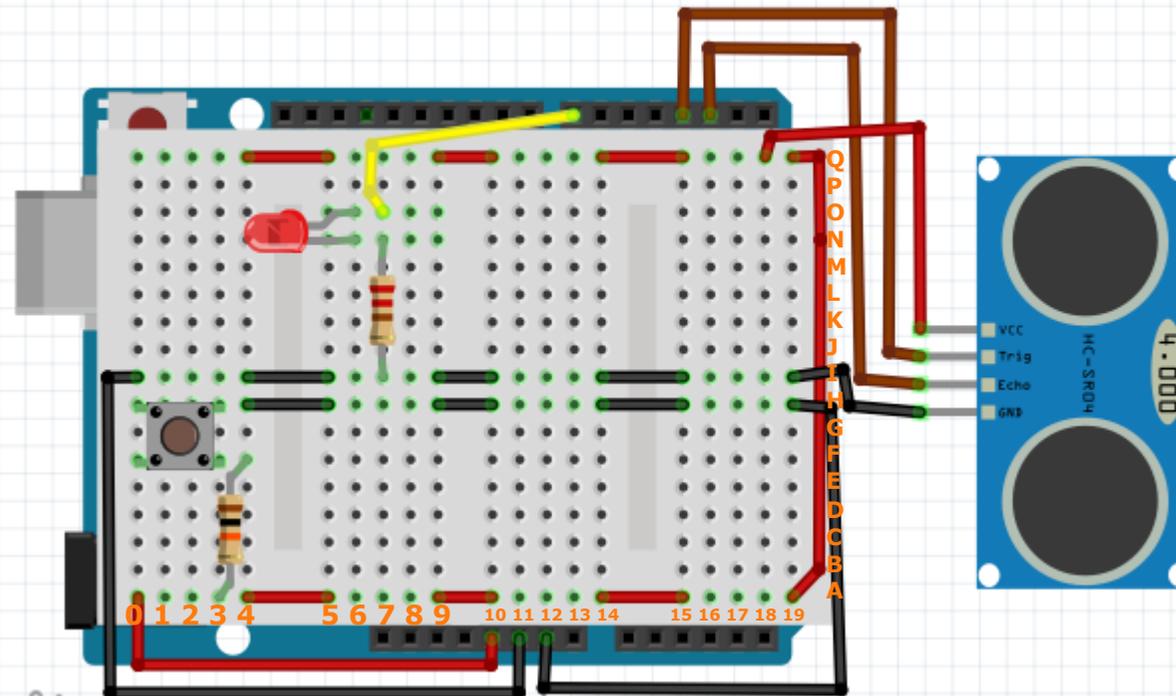
## □ Materiales



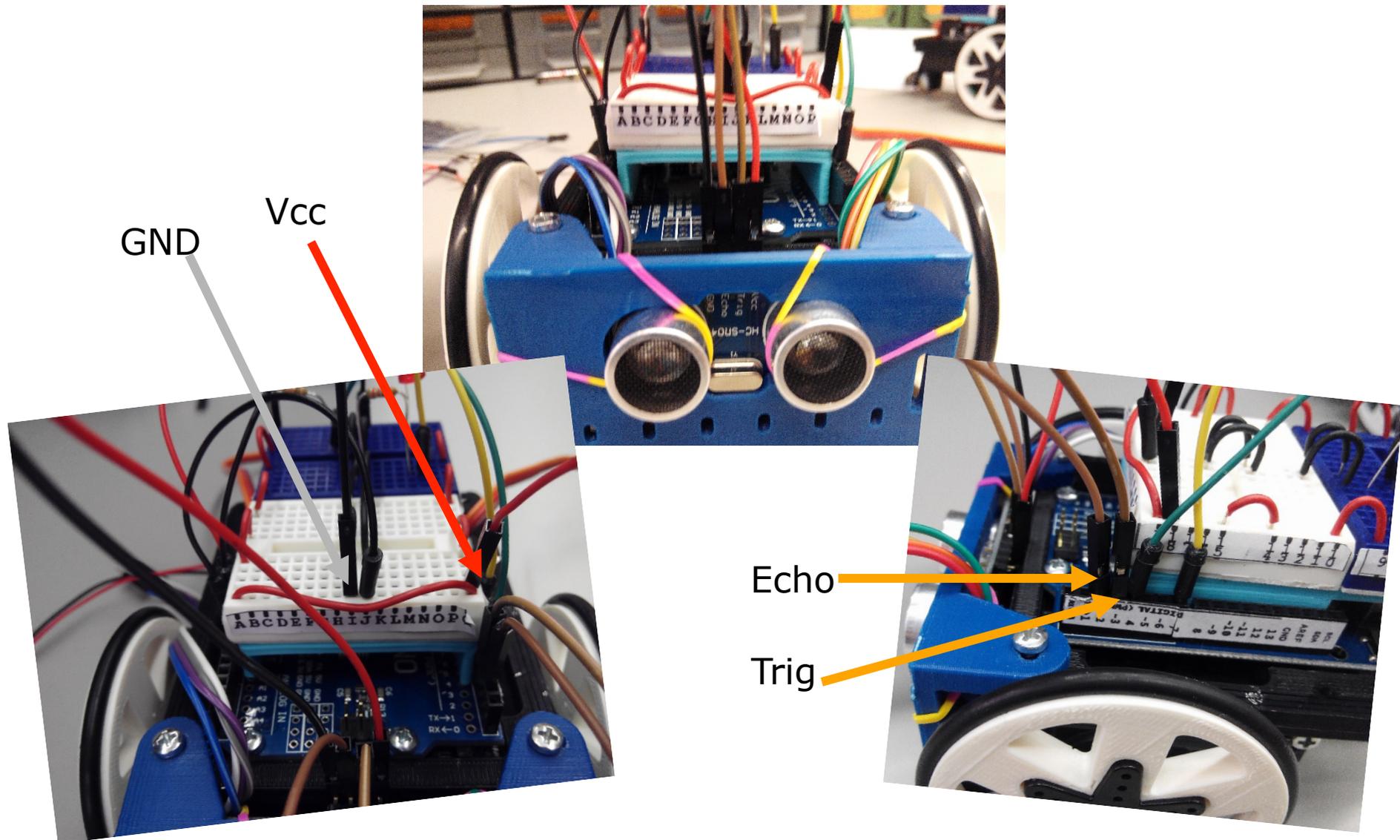
# Jugamos con el sensor de Ultrasonidos



- ❑ Conecta el sensor de ultrasonidos como se indica en la figura.
  - Alimentamos el sensor.
    - ❑ Conectamos Vcc con Q-18 y GND con I-19.
  - Une el TRIG y el ECHO
    - ❑ Conectamos TRIG al PIN 3 y el ECHO al PIN 2.



# Jugamos con el sensor de Ultrasonidos



## □ Ejemplo: Sensor de Distancia (Distance\_example)

### ■ En este ejemplo leeremos la distancia medida por el SRF04

```
#include <DistanceSRF04.h> // Incluimos librería del SRF04
// Pines del SRF04
int pinEco = 2;
int pinTrigger = 3;
// Variables para el control del sensor
DistanceSRF04 Dist;
int distance;

void setup(){
  Serial.begin(19200);
  // Configuramos los pines del sensor.
  Dist.begin(pinEco,pinTrigger);
}

void loop(){
  delay(200);
  // Guardamos la distancia del sensor en la variable
  distance = Dist.getDistanceCentimeter();
  // Enviamos el valor de la distancia por la comunicación serie
  Serial.print("Distancia ");
  Serial.print(distance);
  Serial.println("cm");
}
```

# Aprendiendo a programar

## □ Ejemplo: Sensor de Distancia (Distance\_example)

■ En este ejemplo leeremos la distancia medida por el SRF04

```
#include <DistanceSRF04.h> // Incluimos librería del SRF04
// Pines del SRF04
```

```
int pinEco = 2;
```

```
int pinTrigger = 3;
```

```
// Variables para el control del sensor
```

```
DistanceSRF04 Dist;
```

```
int distance;
```

Variable para utilizar el sensor

Variable donde guardamos la distancia medida por el sensor

```
void setup(){
```

```
  Serial.begin(19200);
```

```
  // Configuramos los pines del sensor.
```

```
  Dist.begin(pinEco,pinTrigger);
```

```
}
```

Configuramos el sensor SRF04 conectándolo a los pines

```
void loop(){
```

```
  delay(200);
```

```
  // Guardamos la distancia del sensor en la variable
```

```
  distance = Dist.getDistanceCentimeter();
```

```
  // Enviamos el valor de la distancia por la comunicación serie
```

```
  Serial.print("Distancia ");
```

```
  Serial.print(distance);
```

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

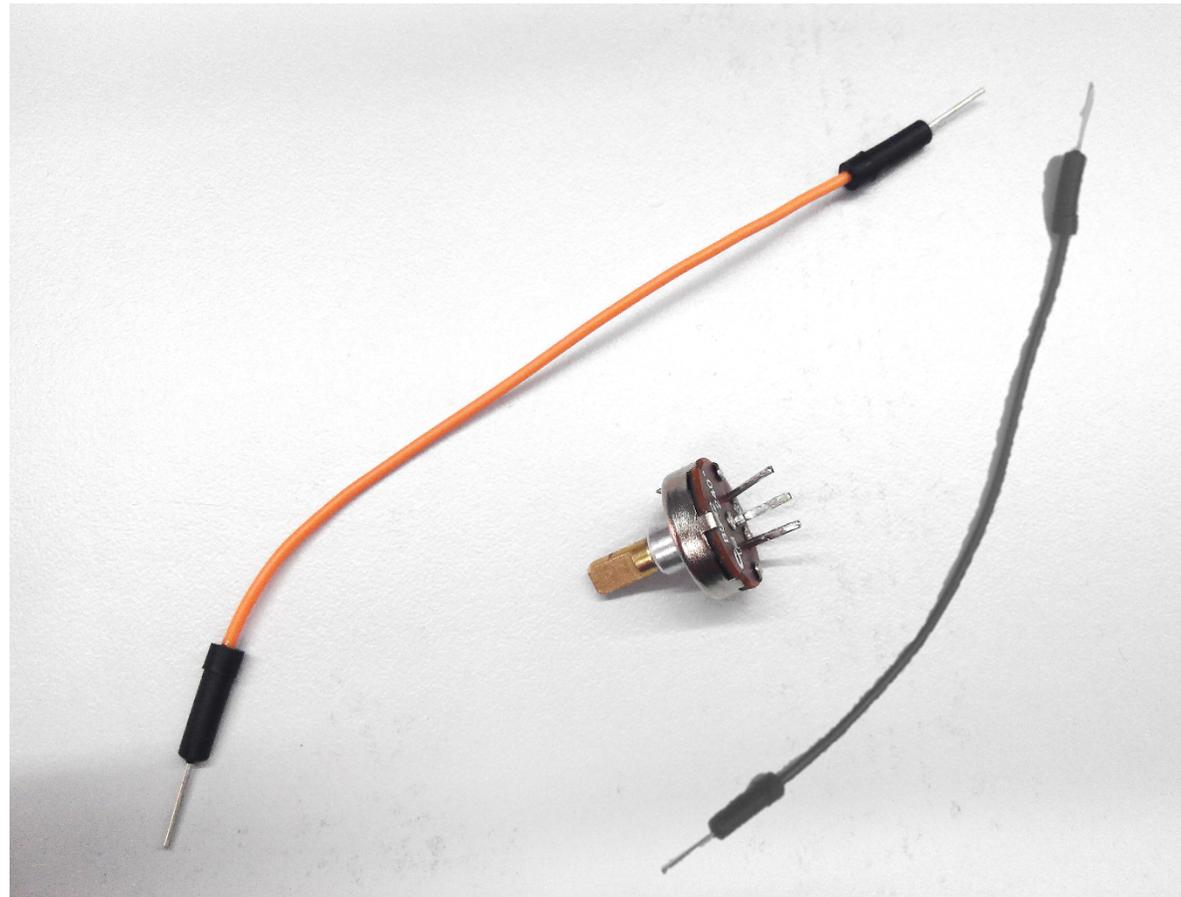
```
}
```

Obtenemos la distancia medida por el sensor

# Conexión del potenciómetro



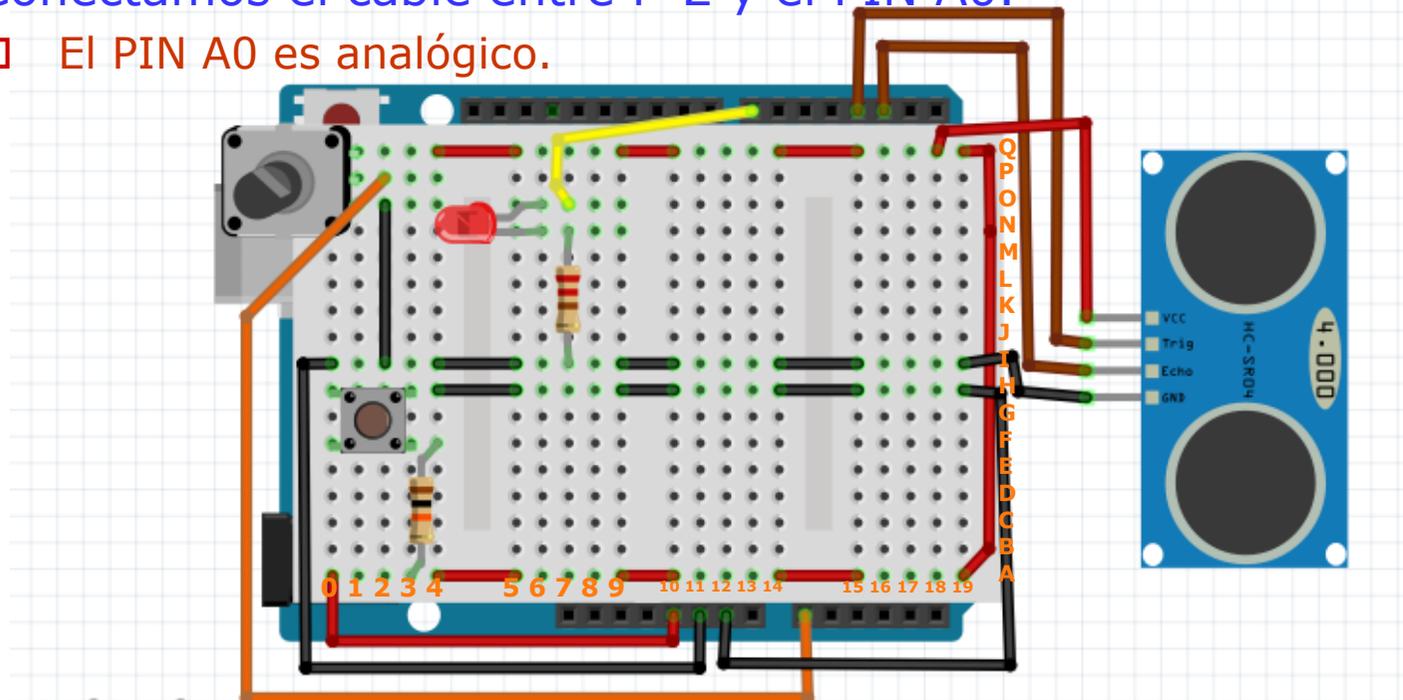
## □ Materiales



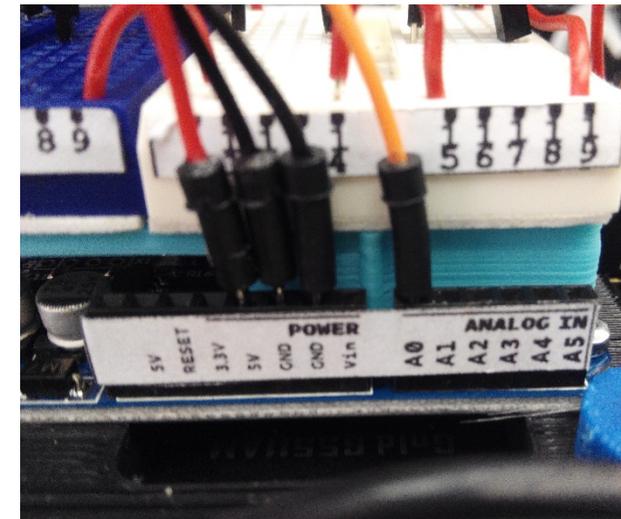
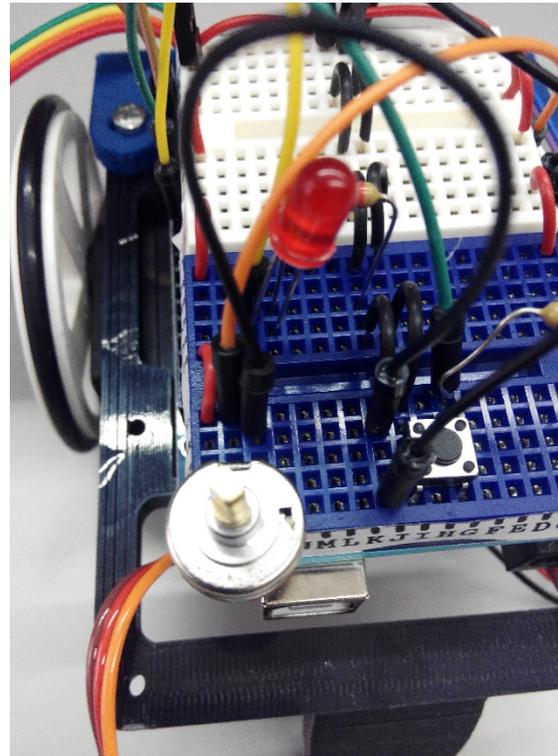
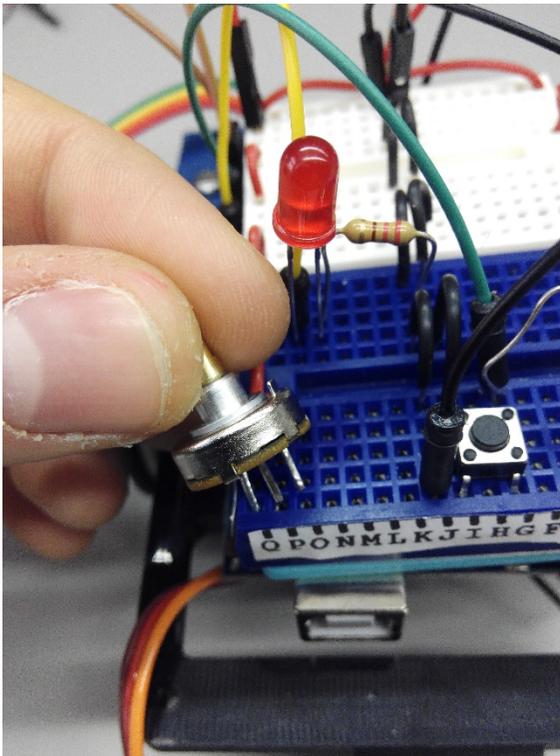
# Conexión del potenciómetro



- Conecta el potenciómetro como se indica en la figura.
  - Potenciómetro.
    - Las tres patillas que tienen, las colocamos en la protoboard en los pines O-1, P-1 y Q-1.
    - Realiza la interconexión de 0V con el cable negro entre O-2 y I-2.
  - Conectamos el cable entre P-2 y el PIN A0.
    - El PIN A0 es analógico.



# Conexión del potenciómetro



# Aprendiendo a programar

## □ Ejemplo: Potenciómetro

- En este ejemplo leeremos la tensión que cae en el potenciómetro.

```
int pinPotenciometer = A0;
```

```
int potenciometer;
```

```
void setup(){
```

```
    Serial.begin(19200); // Configuramos comunicacion serie
```

```
}
```

```
void loop(){
```

```
{
```

```
    float calculo;
```

```
    potenciometer = analogRead(pinPotenciometer);
```

```
    calculo = (float)potenciometer*(5.0/1023);
```

```
    Serial.println("Potenciometro ");
```

```
    Serial.print(calculo);
```

```
    Serial.println("\n");
```

```
    delay(100);
```

```
}
```

Esta función devuelve un valor entre 0 y 1023.

Cuando la entrada tiene 0V -> 0  
Cuando tiene 5V -> 1023

Esta función realiza la conversión a voltios.

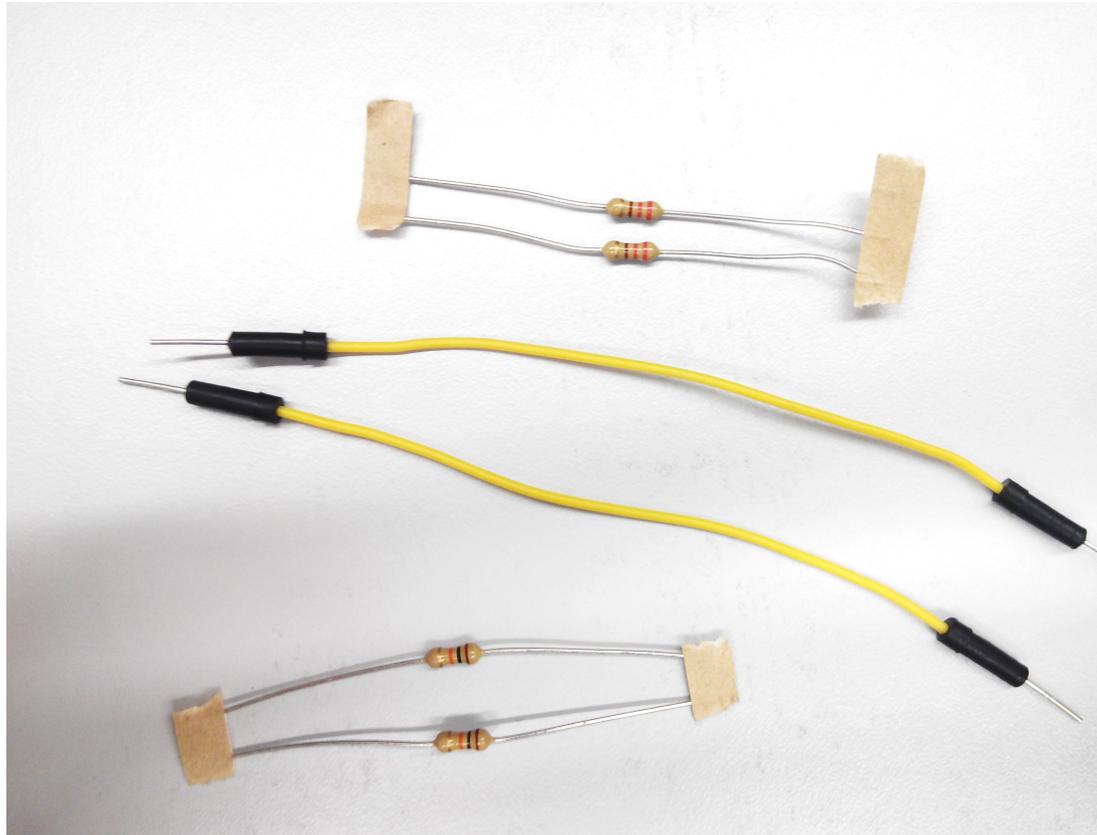
La regla de tres es la siguiente:

1023	-----	5V
Potenciometer	-----	Calculo

**Calculo=potenciometer\*(5/1023)**

# Conexión de los CNY70

## □ Materiales



2 resistencias de  $220\Omega$

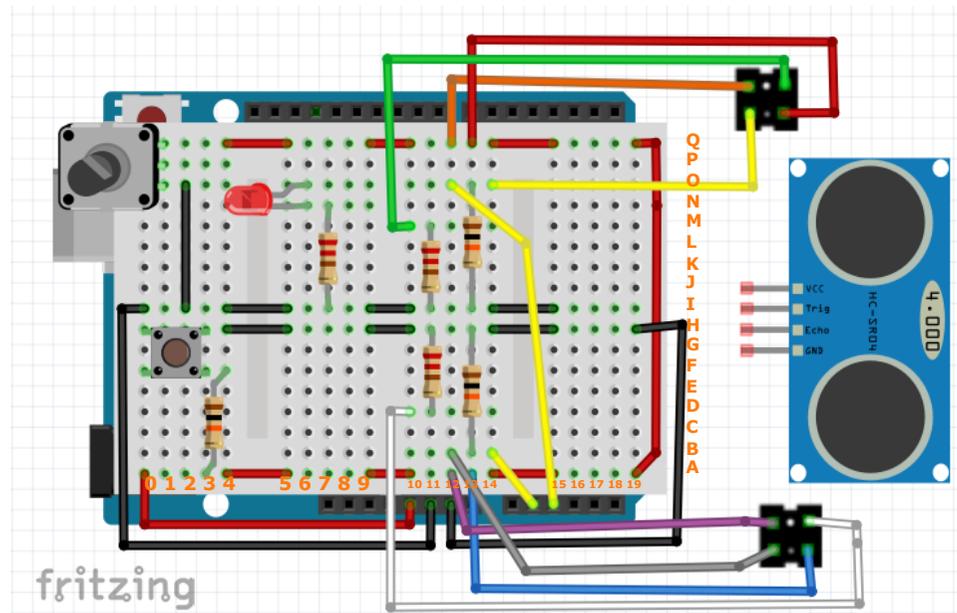
**Rojo** – **Rojo** – **Marrón** – **Oro**

2 resistencias de  $10k\Omega$

**Marrón** – **Negro** – **Naranja** – **Oro**

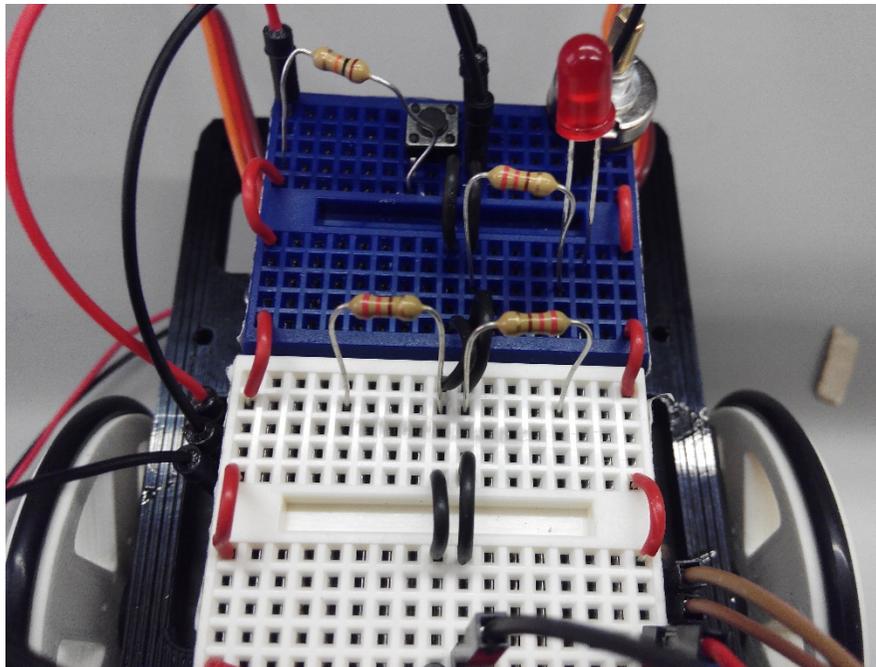
# Conexión de los CNY70

- Conectamos los cables que suben de los sensores CNY70 como se indica en la figura.
  - Los cables de alimentación los conectamos a la columna A.
  - El cable del emisor lo conectamos con la resistencia de  $220\Omega$ .
  - El cable del receptor lo conectamos con la resistencia de  $10k\Omega$ .
  - Lo unimos a los pines analógicos A1 y A2.

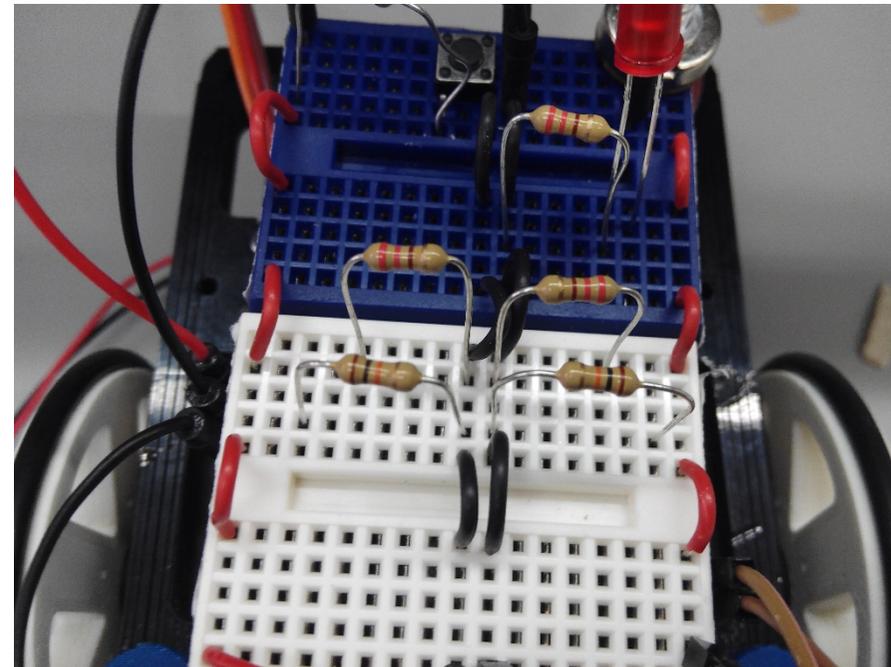


**En las siguientes diapositivas, hay más detalles del montaje**

# Conexión de los CNY70

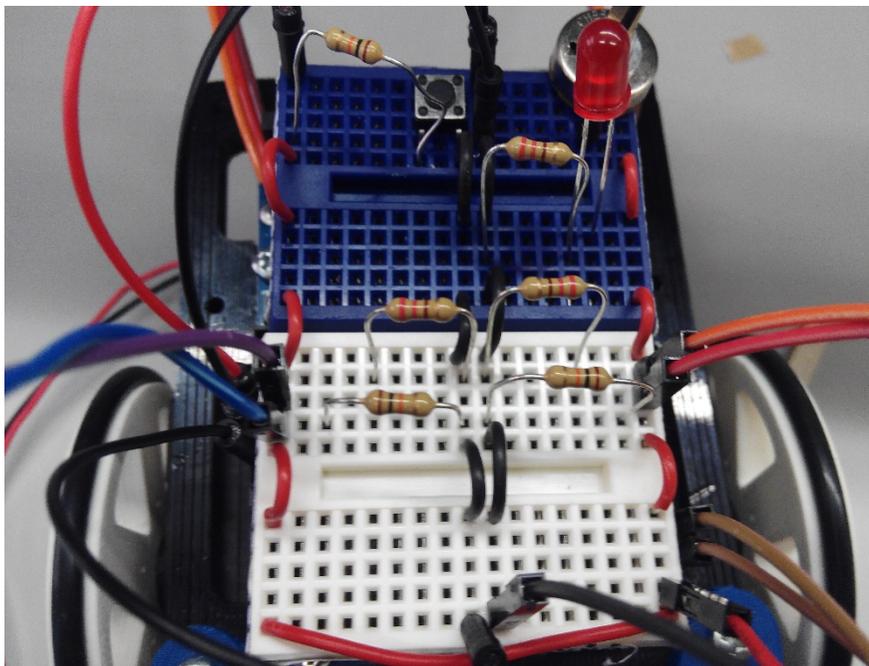


Poner las resistencias de  $220\Omega$

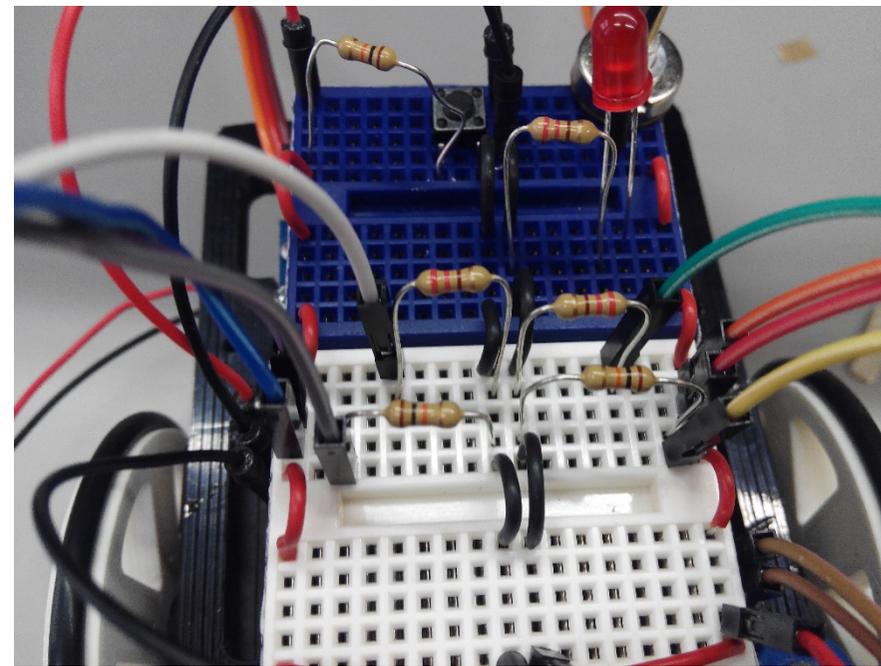


Poner las resistencias de  $10k\Omega$

# Conexión de los CNY70

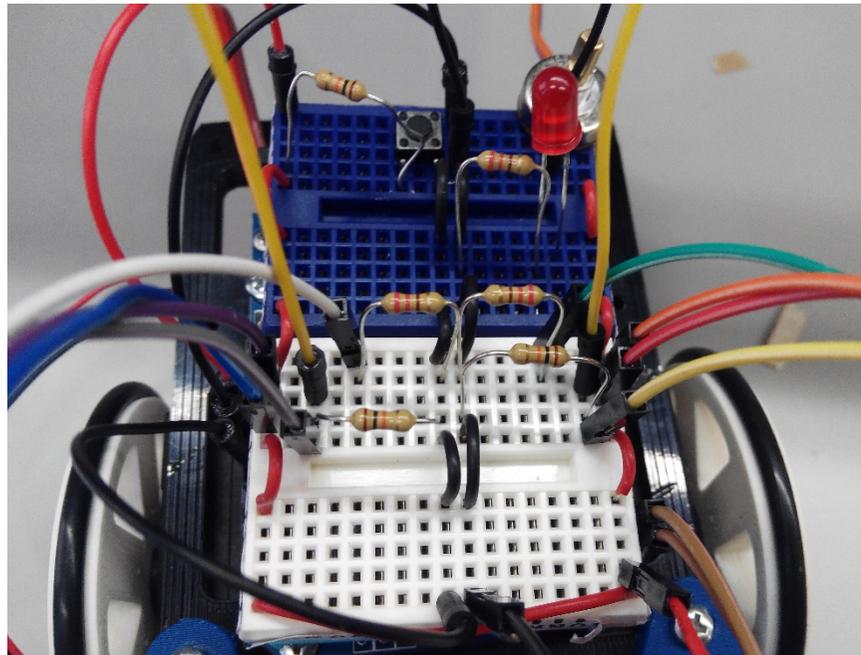


Conecta los cables **Azul**, **Morado**, **Rojo** y **Naranja** a 5V.

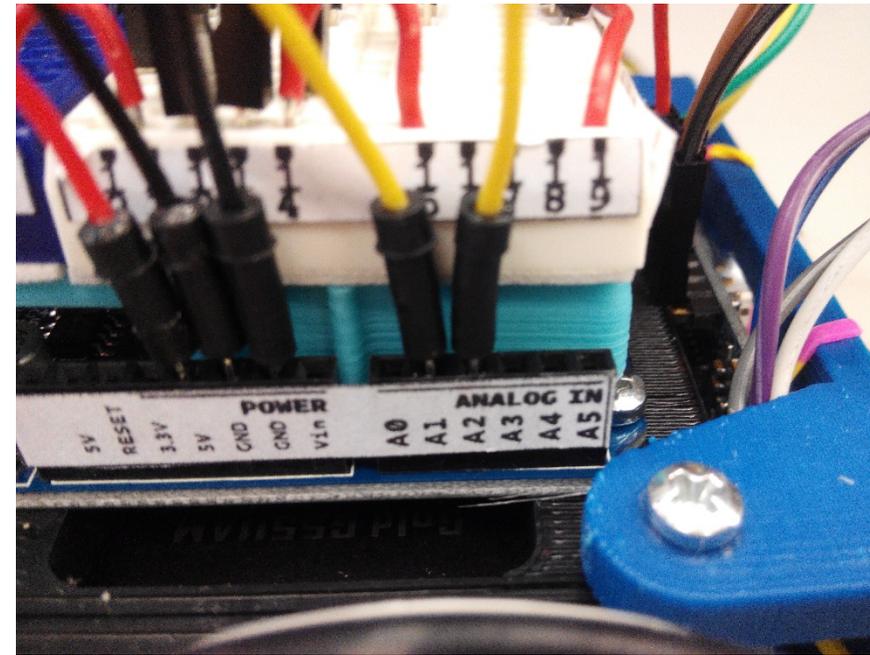


Conecta los cables **Blanco** y **Verde** junto con las resistencias de  $220\Omega$  y los cables **Gris** y **Amarillo** junto con las de  $10k\Omega$ .

# Conexión de los CNY70



Conecta los cables amarillos macho- macho en la misma fila de letras que los cables amarillo y gris de los CNY70.



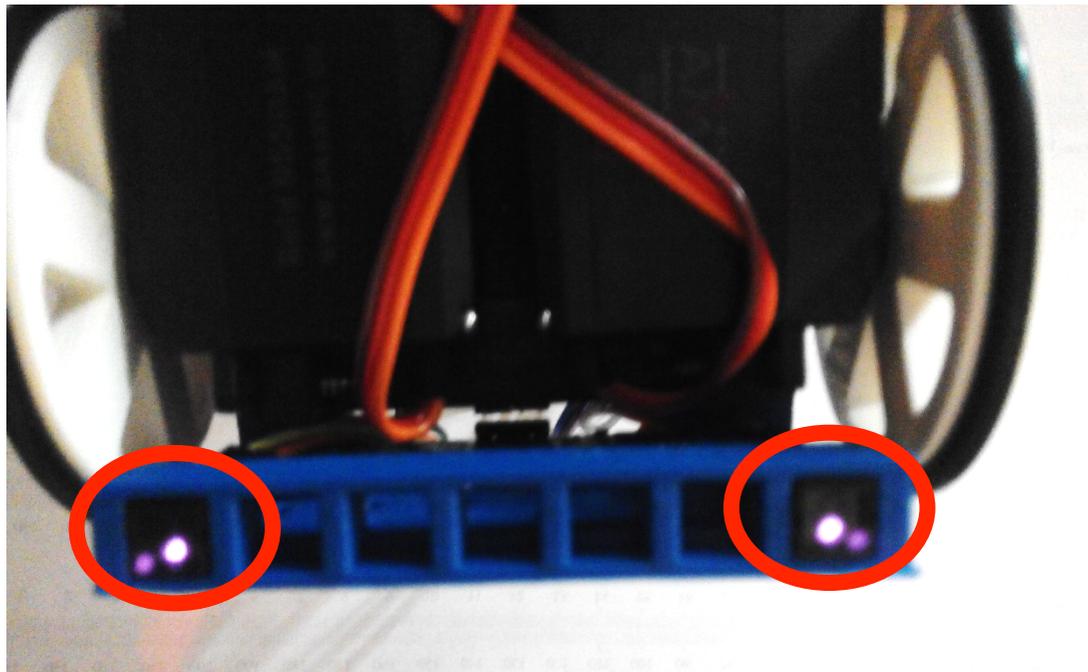
Por último, conecta el cable de la derecha en el A1 y el de la izquierda en el A2. Si te equivocas puedes cambiarlo cuando lo programes.

# Conexión de los CNY70



## □ ¿Funciona?

- Para ver si funciona vamos a usar la cámara del móvil. Nuestro ojo no puede ver los infrarrojos, pero si la cámara no tiene un filtro se puede ver como en la foto.



# Aprendiendo a programar

- Ejemplo: Medida infrarrojos (Infrared\_example)
  - Medida de infrarrojos.

```
int pinInfraredRight = A1;
int pinInfraredLeft = A2;
// Variables donde se guarda el valor de los sensores
int infraredRight;
int infraredLeft;

void setup(){
  Serial.begin(19200); // Configuramos comunicacion serie
}
void loop(){
  delay(200);
  // Leemos valor de los sensores.
  infraredRight = analogRead(pinInfraredRight);
  infraredLeft = analogRead(pinInfraredLeft);

  // Imprimimos el valor de los sensores
  Serial.println("SensorIzd      SensorDer  ");
  Serial.print(infraredLeft);
  Serial.print(" |      ");
  Serial.println(infraredRight);
}
```

# Aprendiendo a programar

- Ejemplo: Medida infrarrojos (Infrared\_example)
  - Medida de infrarrojos.

```
int pinInfraredRight = A1;
int pinInfraredLeft = A2;
// Variables donde se guarda el valor de
int infraredRight;
int infraredLeft;

void setup(){
  Serial.begin(19200); // Configuramos comunicacion serie
}
void loop(){
  delay(200);
  // Leemos valor de los sensores.
  infraredRight = analogRead(pinInfraredRight);
  infraredLeft = analogRead(pinInfraredLeft);

  // Imprimimos el valor de los sensores
  Serial.println("SensorIzd      SensorDer  ");
  Serial.print(infraredLeft);
  Serial.print(" |      ");
  Serial.println(infraredRight);
}
```

## Pines Analógicos:

Pueden tomar cualquier valor analógico de 0V a 5V

## analogRead(pin):

Esta función lee el voltaje que hay en el **pin** analógico dado como parámetro.

# Conectamos los LDR



## □ Materiales



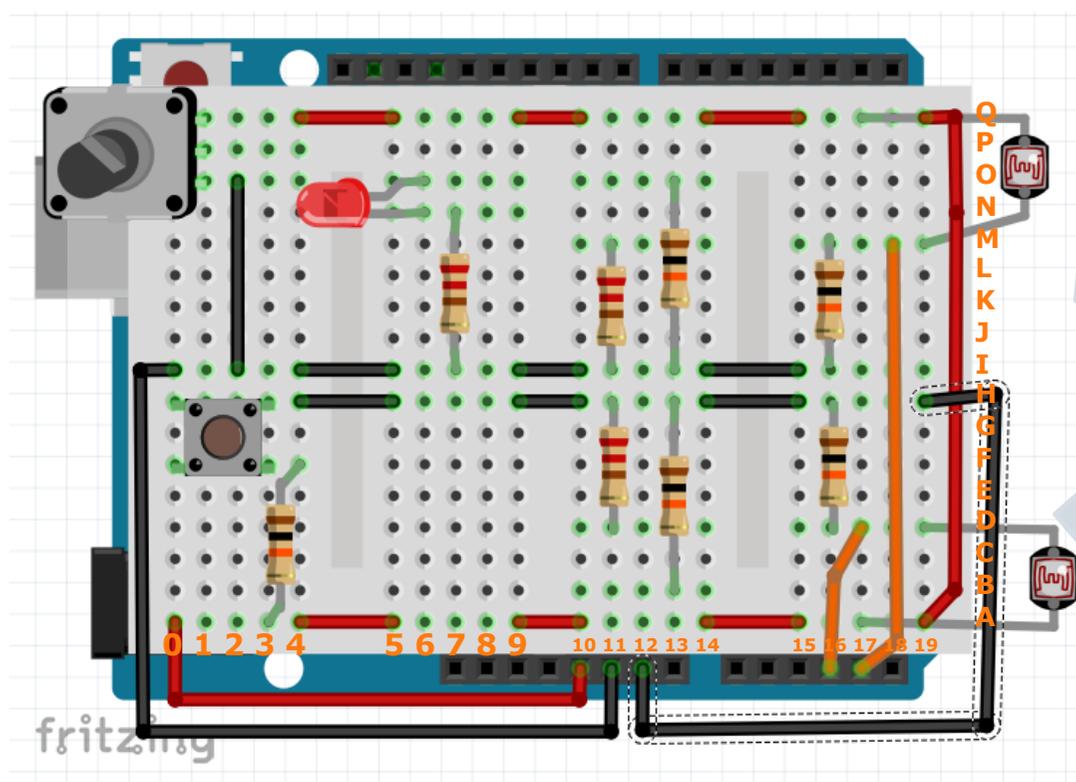
2 resistencias de  $10k\Omega$

**Marrón** - **Negro** - **Naranja** - **Oro**

## Conectamos los LDR

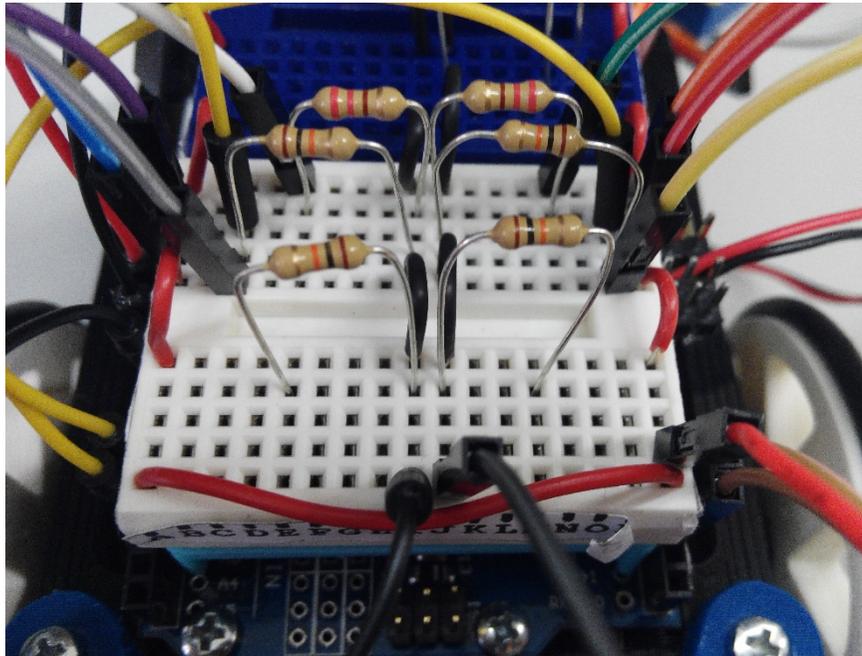


- Conecta las patillas del LDR como se indica en la figura.
- Uno de ellos entre Q-17 y M-17, el otro A-17 y D-17.
- Conectamos las resistencias entre M-16 y I-16 ; H-16 y D-16.
- Une los puntos M-18 y D-18 con los pines analógicos A3 y A4.

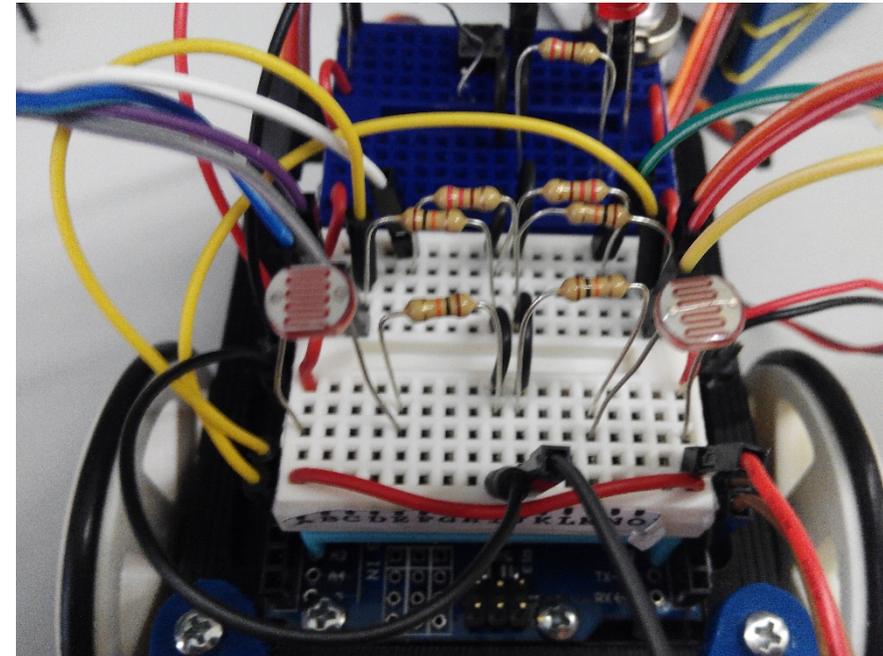


**En las siguientes diapositivas, hay más detalles del montaje**

# Conectamos los LDR

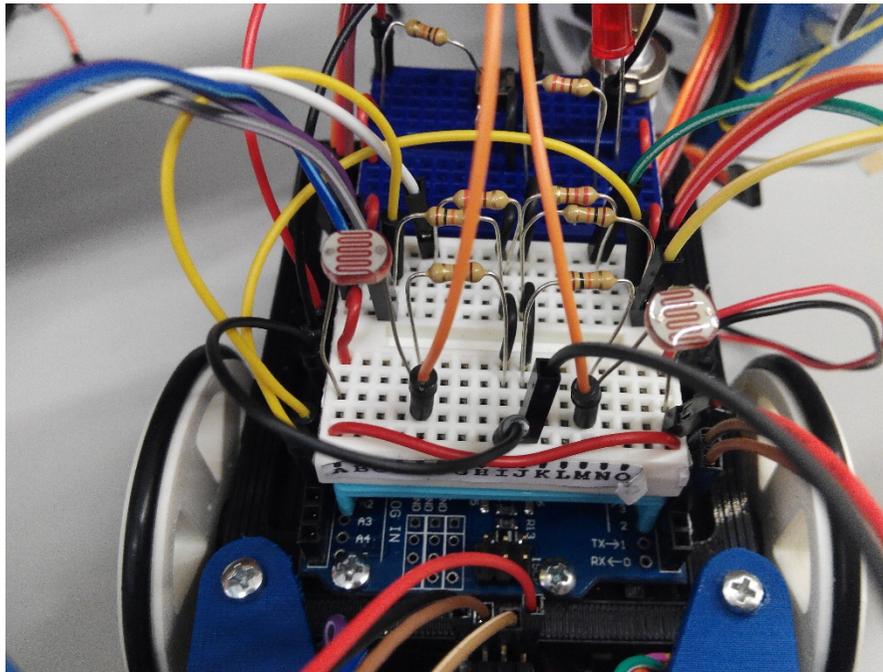


Conectamos las resistencias de 10k, entre M-16 y I-16 ; H-16 y D-16.

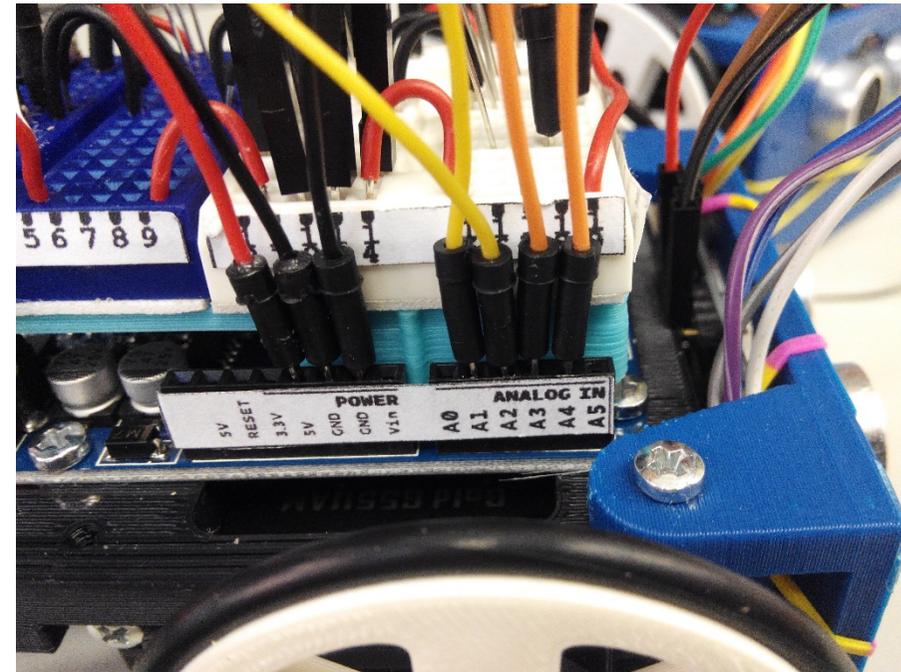


Conectamos los LDR entre Q-17 y M-17, el otro A-17 y D-17.

# Conectamos los LDR



Conectamos los cables naranjas macho-macho en la unión entre la resistencia y el LDR. En los puntos, M-18 y D-18.



Conectamos la otra punta de los cables a la entrada analógica A3 al LDR derecho y A4 al LDR izquierdo.



- Ejemplo: Medida infrarrojos (LDR\_example)
  - Medida de LDR.

```
// Pines de los sensores
int pinLDRRight =    A3;
int pinLDRLeft  =    A4;
// Variables donde se guarda el valor de los sensores
int LDRRight;
int LDRLeft;

void setup(){
  Serial.begin(19200); // Configuramos comunicacion serie
}
void loop(){
  delay(200);
  // Leemos el valor de los sensores
  LDRRight = analogRead(pinLDRRight);
  LDRLeft  = analogRead(pinLDRLeft);

  // Imprimimos el valor de los sensores
  Serial.println("LDR Izd      LDR Der ");
  Serial.print(LDRLeft);
  Serial.print(" | ");
  Serial.println(LDRRight);
}
```