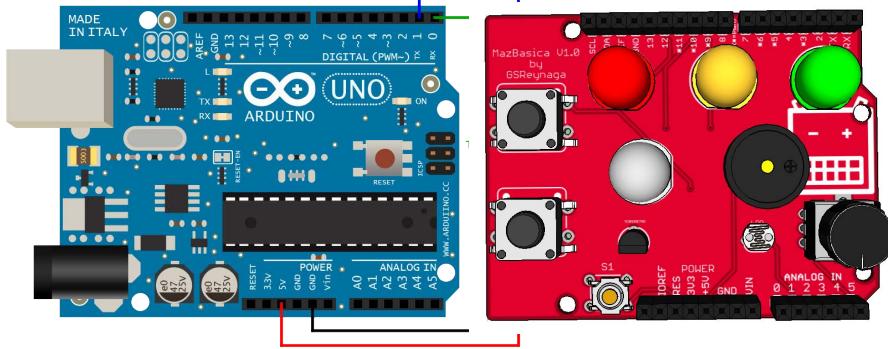




Tarjeta Entrenadora

"EduMazShield"

100% OpenHardware



¿Qué aprendo?

- Activar una salida digital
- Encender un LED en ON/OFF
- Temporizar una señal de salida
- Sintaxis de un programa en Arduino

Conocimientos previos

- Señal digital
- Función digitalWrite()
- Polaridad de un LED
- Conexión de la placa Arduino al computador

Materiales

1



Arduino UNO

1



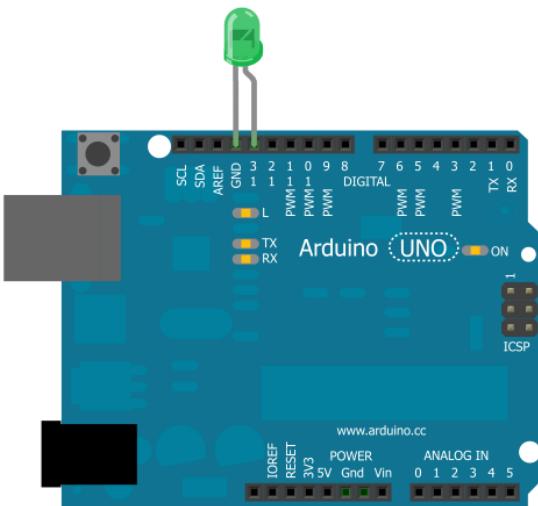
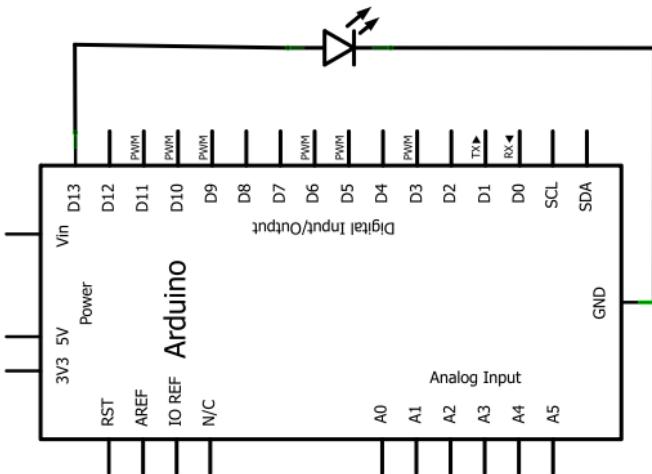
LED Verde

1

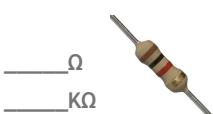


Cable USB Tipo AB





1— ¿Cuál es el valor de esta resistencia?



2— ¿Qué hace esta función?

`digitalRead()`

3— Completa

$$I = \frac{?}{R}$$

C
Ó
D
I
G
O

```
/*
-----  

  Hola Mundo  

-----  

  Enciende un LED por un segundo y lo apaga por el mismo tiempo  

*/
```

D
E

```
//-----  

//Función principal  

//-----
```

P
R
O
G
R
A
M
A
C
I
O
N

```
void setup()      // Se ejecuta cada vez que el Arduino se inicia
{
    pinMode(13,OUTPUT); // Inicializa el pin 13 como una salida
}

//-----  

//Función cíclica  

//-----
```

```
void loop()      // Esta función se mantiene ejecutando
{
    digitalWrite(13,HIGH); // Enciende el LED
    delay(1000);          // Temporiza un segundo (1s = 1000ms)
    digitalWrite(13,LOW); // Apaga el LED
    delay(1000);          // Temporiza un segundo (1s = 1000ms)
}
```

// Fin del programa

T
I
P
S

1- El `//` en programación se utiliza para hacer comentarios, es muy útil para que puedas explicar algo acerca de la sintaxis de una línea de código. Un ejemplo de su uso:

`digitalWrite(13,LOW); // Apaga el LED`

2- Las señales digitales (Encendido o apagado) están muy presentes en todos los sistemas, y muchos sensores trabajan sobre este principio, te invitamos a conocer algunos:



Sensor PIR

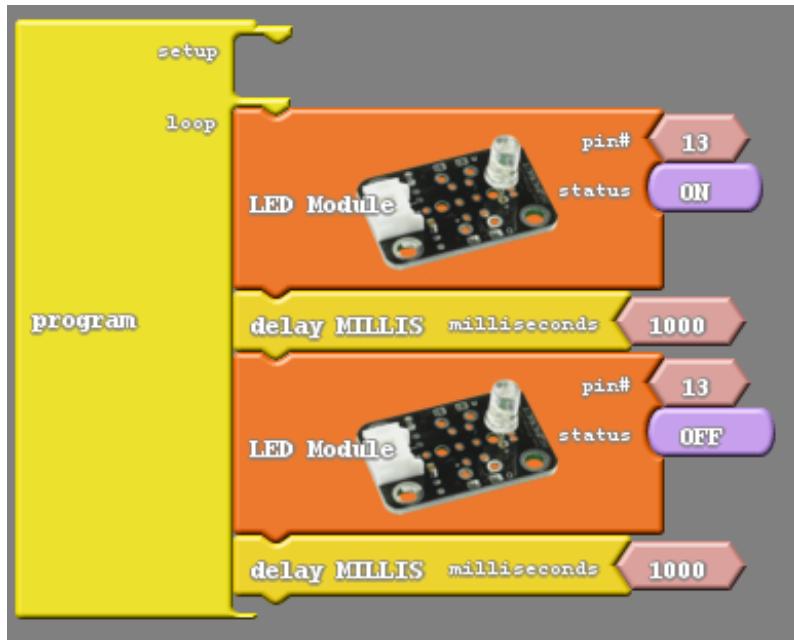
Detecta movimiento



Sensor óptico SHARP

Detecta la presencia de algún objeto en un rango de 5cm

```
/*
-----  
Hola Mundo  
-----  
Enciende un LED por un segundo y lo apaga por el mismo tiempo  
*/
```



Encender un LED con un pulsador

¿Qué aprendo?

- Cablear un circuito
- Condicional If/else
- Estado de un pulsador
- Leer una entrada digital y escribir una salida digital

Conocimientos previos

- Señal digital
- Función digitalWrite() y digitalRead()
- Divisor de voltaje
- Condicional y operadores de comparación

Materiales

1



Arduino UNO

1



LED Amarillo

1



Protoboard

1



Cable USB Tipo AB

1



Pulsador

1



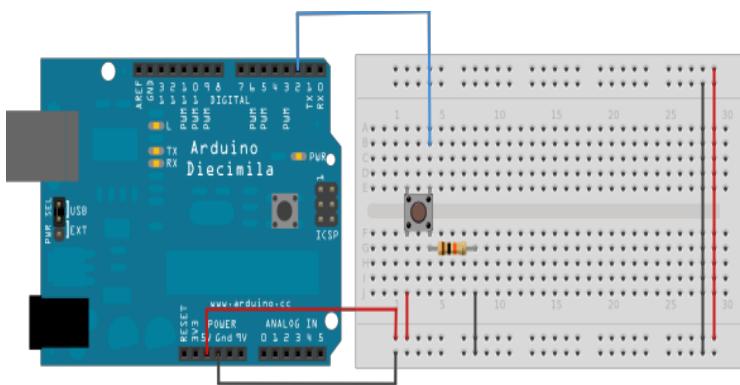
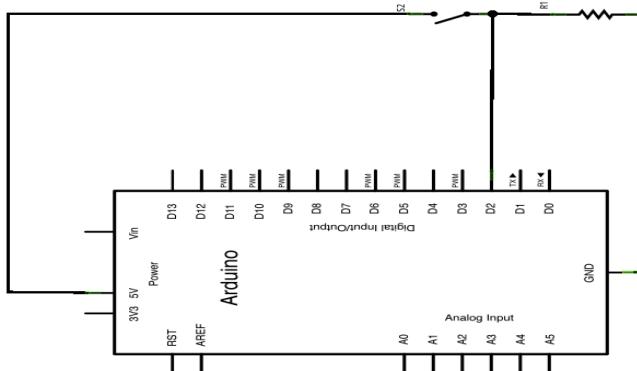
Resistencia 1K

4

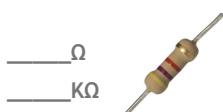


Conectores MM





1— ¿Cuál es el valor de esta resistencia?



2— ¿Qué hace esta función?

`digitalWrite()`

3— Un ejemplo de un lenguaje de alto nivel

C
Ó
D
I
G
O

D
E

P
R
O
G
R
A
M
A
C
I
Ó
N

```
/*
-----  

  Encender LED con un pulsador  

-----  

  Oprimir un pulsador y mientras este se mantenga accionado  

    un LED se enciende  

*/  

//-----  

//Declara puertos de entradas y salidas  

//-----  

int pulsador=2;           //Pin donde se encuentra el pulsador, entrada  

int led=13;                //Pin donde se encuentra el LED, salida  

//-----  

//Funcion principal  

//-----  

void setup() // Se ejecuta cada vez que el Arduino se inicia  

{  

  pinMode(pulsador, INPUT); //Configurar el pulsador como una entrada  

  pinMode(led,OUTPUT);     //Configurar el LED como una salida  

}  

//-----  

//Funcion ciclica  

//-----  

void loop() // Esta funcion se mantiene ejecutando  

{  

  //Condicional para saber estado del pulsador  

  if (digitalRead(pulsador)==HIGH)  

  {  

    //Pulsador oprimido  

    digitalWrite(led,HIGH); //Enciende el LED  

  }  

  else  

  {  

    //Pulsador NO oprimido  

    digitalWrite(led,LOW); //Apaga el LED  

  }  

}  

//Fin programa
```

T
I
P
S

1- Cuando estés programando en el Software de Arduino, muchas cosas de las que escribes son palabras reservadas por el lenguaje, todas las palabras reservadas las puedes encontrar en la sección **5S**, al escribirlas éstas se colocan en un color diferente, este es un truco para saber que esta bien, por ejemplo:

```
void loop() {  

  digitalWrite(13, HIGH);  

  delay(1000);  

  digitalWrite(13, LOW);  

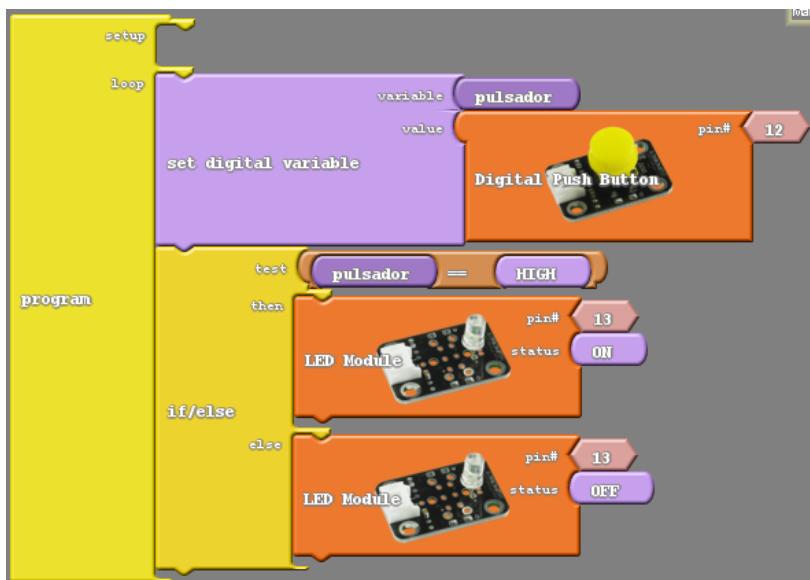
  delay(1000);  

}
```

2- Todas la instrucciones de programación para Arduino, se encuentran totalmente documentadas con claros ejemplos de cómo se utilizan, te invitamos a que visites: arduino.cc/en/Reference/HomePage



```
/*
-----  
Encender LED con un pulsador  
-----  
Oprimir un pulsador y mientras este se mantenga accionado  
un LED se enciende  
*/
```



Lectura serial de una entrada digital

¿Qué aprendo?

- Manejar una entrada digital
- Ver datos por la pantalla del computador
- Consola serial
- Leer una entrada digital y escribir por consola serial

Conocimientos previos

- Señal digital
- Función `digitalRead()` y `Serial.println()`
- Opción de Consola serial, ver **6F** (paso 3)

Materiales

1



Arduino UNO

1



Pulsador

1



Protoboard

1



Cable USB Tipo AB

1



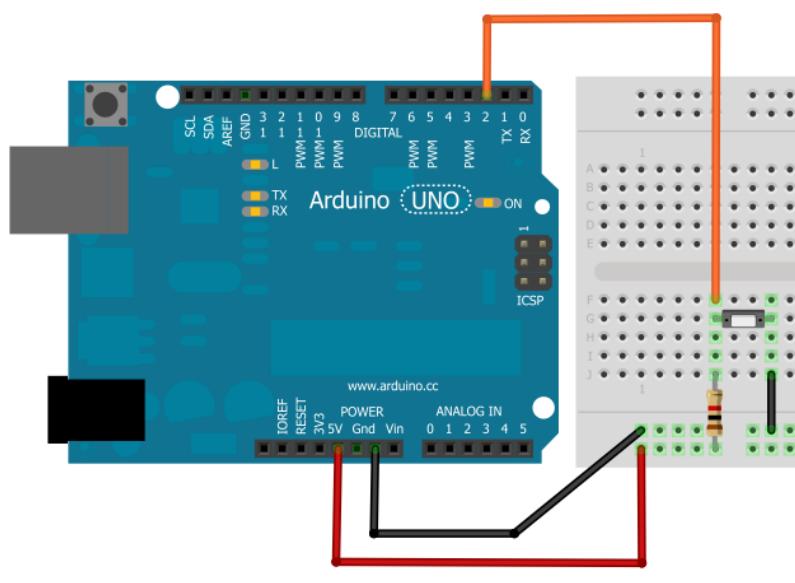
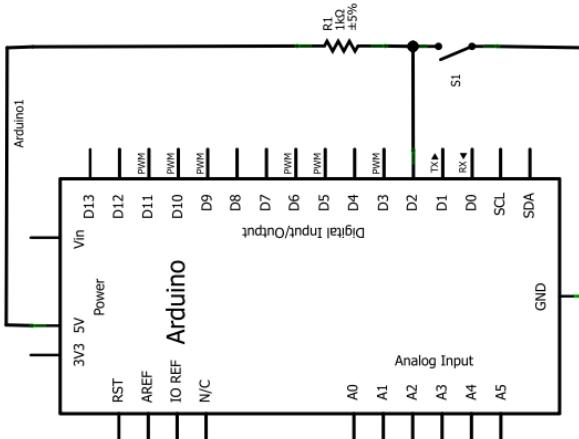
Resistencia 1K

4

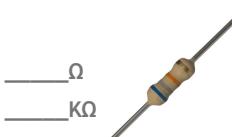


Conectores MM





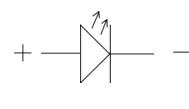
1— ¿Cuál es el valor de esta resistencia?



2— ¿Qué hace esta función?

`Serial.println()`

3— Este símbolo a que corresponde



```

C /*-----*/
Ó Lectura serial de una entrada digital
D -----  

I Leer una entrada digital y mostrar por la pantalla del
G computador (consola serial) el estado del pulsador
O cuando es oprimido
D */
E
P //-----  

R //Declara puertos de entradas y salidas  

A //-----  

P int boton=2; //Pin donde se encuentra el pulsador, entrada  

R //-----  

A //Funcion principal  

P //-----  

R void setup() // Se ejecuta cada vez que el Arduino se inicia  

O {  

    //Configuración  

    pinMode(boton,INPUT); //Configurar el boton como una entrada  

    Serial.begin(9600); //Inicia comunicación serial  

} //-----  

G //Funcion ciclica  

R //-----  

A void loop() // Esta funcion se mantiene ejecutando  

M { // cuando este energizado el Arduino  

    //Guardar en una variable entera el valor del boton 0 ó 1  

    int estado = digitalRead(boton);  

    //Condicional para saber estado del pulsador  

    C if (estado==0)  

        I {  

            // Pulsado  

            Serial.println("Pulsado"); //Imprime en la consola serial  

        } else  

            O {  

                // No esta pulsado  

                Serial.println("NO Pulsado"); //Imprime en la consola serial  

            }  

            delay(100); //Retardo para la visualización de datos en la consola  

        }  

    //Fin programa
  
```

TIPS

1- La codificación binaria es muy importante para transmitir datos entre dispositivos, son las largas cadenas de 0 y 1, por ejemplo 00011101010101 esto podría ser un mensaje que contiene información referente a una clave personal para acceder a un edificio. Los números en base 10 se pueden representar como valores binarios:

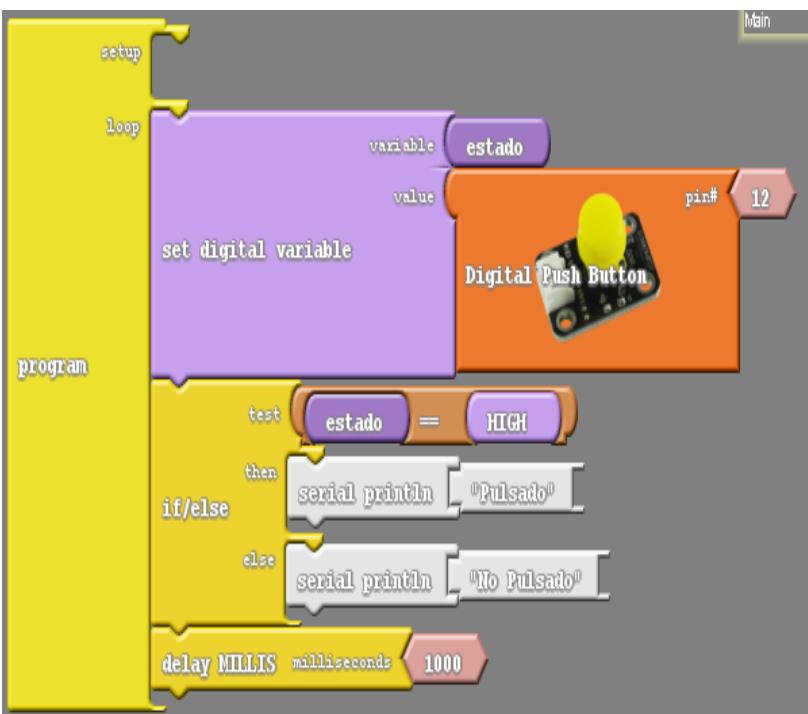
2- Para recordar



- Para leer una señal digital usa: **digitalRead(numeroPin);**
- Para escribir una señal digital usa: **digitalWrite(numeroPin, valor);**
- Una salida o entrada digital siempre es **HIGH** o **LOW**

| | |
|---|-----|
| 0 | 000 |
| 1 | 001 |
| 2 | 010 |
| 3 | 011 |
| 4 | 100 |
| 5 | 101 |
| 6 | 110 |
| 7 | 111 |

```
/*
Lectura serial de una entrada digital
-----
Leer una entrada digital y mostrar por la pantalla del
computador (consola serial) el estado del pulsador
cuando es oprimido
*/
```



¿Qué aprendo?

- Manejar una entrada analógica
- Ver datos por la pantalla del computador
- Múltiples estados de un potenciómetro
- Leer una entrada analógica

Conocimientos previos

- Señal analoga
- Función analogRead() y Serial.println()
- Opción de Consola serial, ver **6F** (paso 3)

Materiales

1



Arduino UNO

1



Potenciómetro 10K

1



Protoboard

1



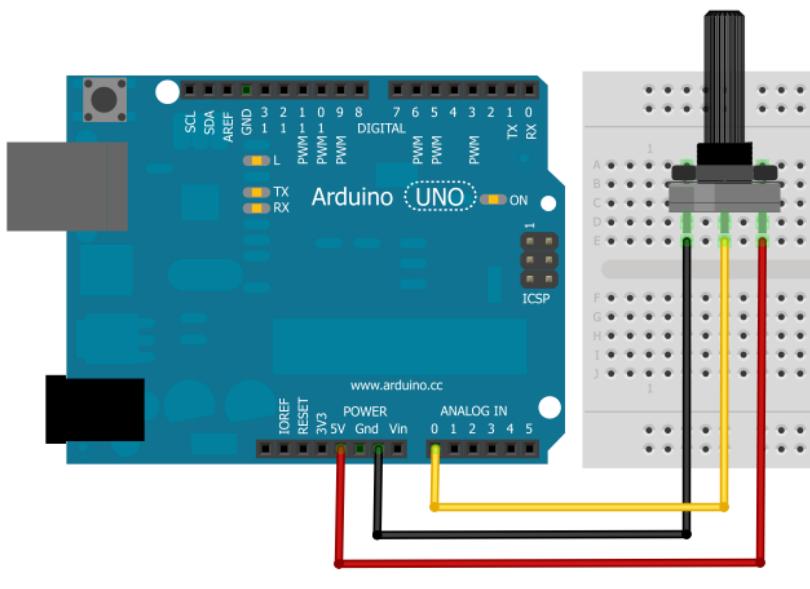
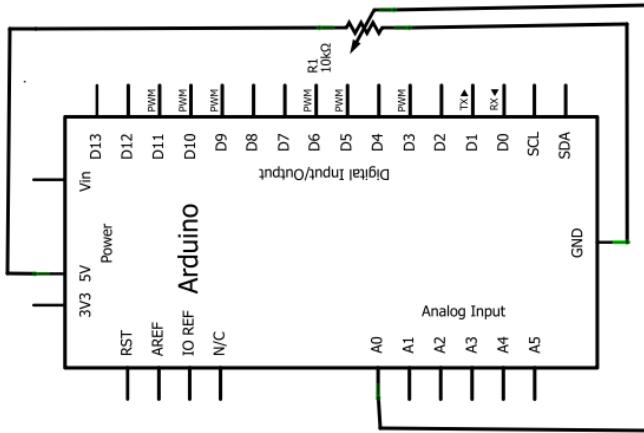
Cable USB Tipo AB

3



Conectores MM





1— ¿Cuál es el valor de esta resistencia?

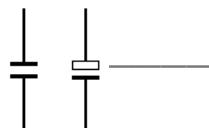


2— ¿Qué hace esta función?

`analogRead()`



3— Este símbolo a que corresponde



```
/*
-----  
Lectura serial de entrada analoga  
-----  
  
Leer una entrada analoga y mostrar por la pantalla del  
computador (consola serial) el valor luego de girar  
el potenciómetro  
  
*/  
  
//-----  
//Funcion principal  
//-----  
void setup() // Se ejecuta cada vez que el Arduino se inicia  
{  
    Serial.begin(9600); //Inicia comunicación serial  
}  
  
//-----  
//Funcion ciclica  
//-----  
void loop() // Esta funcion se mantiene ejecutando  
{  
    // cuando este energizado el Arduino  
  
    //Guardar en una variable entera el valor del potenciómetro 0 a 1024  
    int valor= analogRead(A0);  
  
    //Imprime en la consola serial el valor de la variable  
    Serial.println(valor);  
  
    //Retardo para la visualización de datos en la consola  
    delay(100);  
}  
  
//Fin programa
```

1- Te invitamos a que conozcas algunos tipos de potenciómetros



SoftPot
Sistema touch



Trimmer
Alta precisión



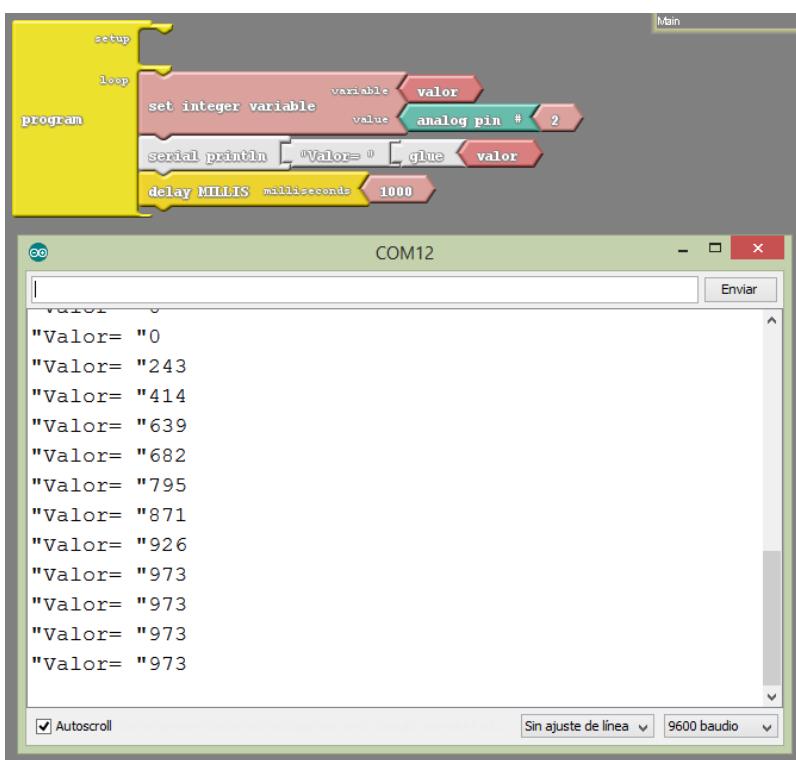
Encoder RGB
Giro continuo

2- Para recordar



- Para leer una señal analoga usa: `analogRead(numeroPin);`
- Para escribir una señal analoga de PWM usa: `analogWrite(numeroPin, valor);`
- Una entrada analoga va de **0** o **1023**
- Una salida analoga de PWM va de **0** o **255**

```
/*
-----  
Lectura serial de entrada analoga  
-----  
  
Leer una entrada analoga y mostrar por la pantalla del  
computador (consola serial) el valor luego de girar  
el potenciómetro
```



The screenshot shows the Arduino IDE interface. On the left, there's a vertical bar with the text "CÓDIGO DE PROGRAMACIÓN". The main area displays the following code:

```
/*
-----  
Lectura serial de entrada analoga  
-----  
  
Leer una entrada analoga y mostrar por la pantalla del  
computador (consola serial) el valor luego de girar  
el potenciómetro
```

The code consists of a setup block and a loop block. The setup block contains an empty function. The loop block contains the following code:

```
set integer variable valor
value analog pin # 2
serial println "Valor= " valor
delay millis milliseconds 1000
```

Below the code, the串行监视器 (Serial Monitor) window is open, connected to COM12. It shows the following output:

```
Valor =  
"Valor= "0  
"Valor= "243  
"Valor= "414  
"Valor= "639  
"Valor= "682  
"Valor= "795  
"Valor= "871  
"Valor= "926  
"Valor= "973  
"Valor= "973  
"Valor= "973  
"Valor= "973
```

The monitor also has settings at the bottom: Autoscroll, Sin ajuste de línea, and 9600 baudio.



¿Qué aprendo?

- Entrada por consola (teclado)
- Variables booleanas
- Estado de un LED
- Escritura serial digital

Conocimientos previos

- Señal digital
- Función digitalWrite() y Serial.read()
- Configuración de una comunicación serial
- Polaridad de un LED

Materiales

1



Arduino UNO

1



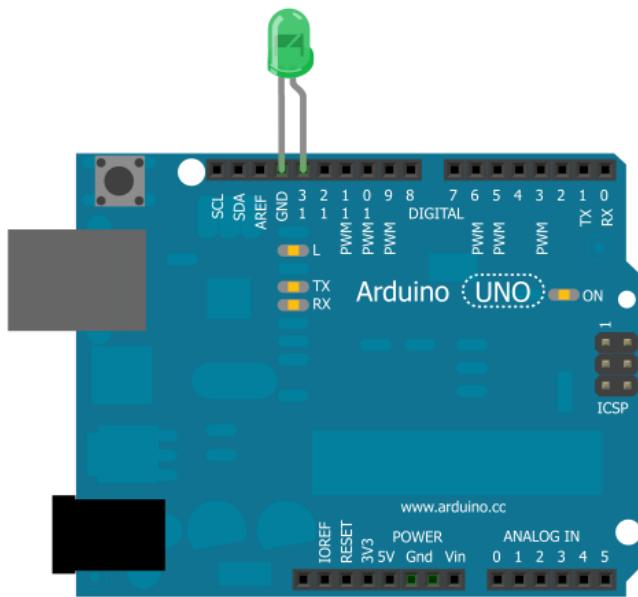
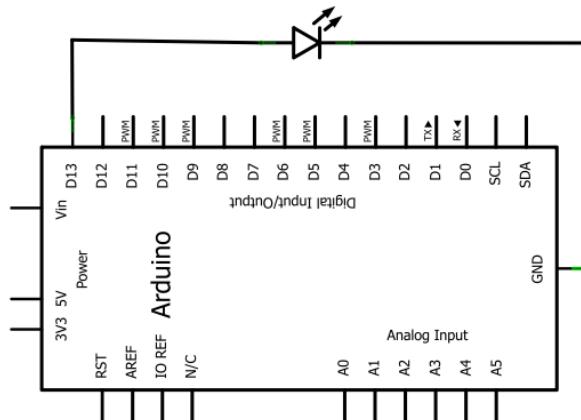
LED de cualquier color.

1

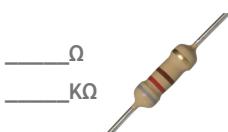


Cable USB Tipo AB





1— ¿Cuál es el valor de esta resistencia?



2— ¿Qué hace esta función?

`Serial.read()`

3— Nombra 4 tipos de variables

a. _____

b. _____

c. _____

d. _____

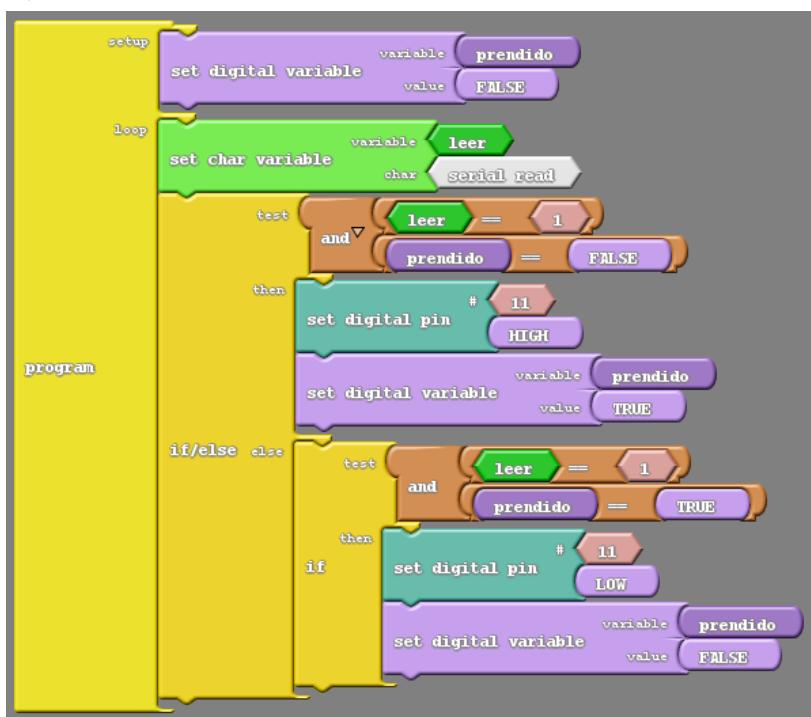


```
/*
-----  
Escritura serial  
-----  
  
Consiste en escribir por la pantalla del computador (consola serial)  
una letra predeterminada, la primera vez que se escriba está  
un LED se enciende, si se vuelve a escribir por segunda vez  
el LED se apaga.  
  
*/  
  
//-----  
//Declara puertos de entradas y salidas y variables  
//-----  
E int led = 13; //Pin donde se encuentra el LED, salida  
char leer; //Variable donde se almacena la letra  
boolean prendido=false; //Estado LED la primera vez, apagado  
  
//-----  
//Funcion principal  
//-----  
R void setup() // Se ejecuta cada vez que el Arduino se inicia  
{  
    Serial.begin(9600); //Inicia comunicación serial  
    pinMode(led, OUTPUT); //Configurar el LED como una salida  
}  
  
//-----  
//Funcion ciclica  
//-----  
M void loop() // Esta funcion se mantiene ejecutando  
{  
    // Guardar en una variable el valor de la consola serial  
    leer=Serial.read();  
  
    // Si es el numero '1' y además el LED está apagado  
    if ( (leer=='1') && (prendido==false) )  
    {  
        digitalWrite(led,HIGH); // Enciende el LED  
        prendido=true; // Actualiza el estado del LED  
    } // Si es la letra 'a' y además el LED está encendido  
    else if ( (leer=='a') && (prendido==true) )  
    {  
        digitalWrite(led,LOW); // Apaga el LED  
        prendido=false; // Actualiza el estado del LED  
    }  
}  
  
//Fin programa
```



```
/*
Escritura serial
Consiste en escribir por la pantalla del computador (consola serial)
una letra predeterminada, la primera vez que se escriba está
un LED se enciende, si se vuelve a escribir por segunda vez
el LED se apaga.
```

```
*/
```



¿Qué aprendo?

- Encender un LED de manera proporcional
- Apropiar el concepto de PWM
- Escribir una salida analógica
- If/else con operadores lógicos

Conocimientos previos

- PWM
- Función analogWrite()
- Polaridad de un LED
- Incrementar y manipular variables

Materiales

1



Arduino UNO

1



LED Amarillo

1



Protoboard

1



Cable USB Tipo AB

1



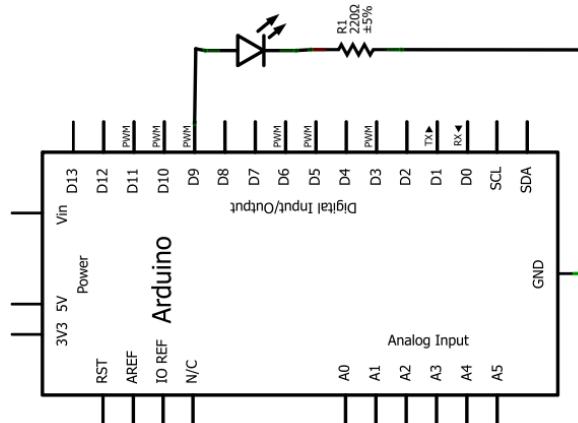
Resistencia 220Ω

2

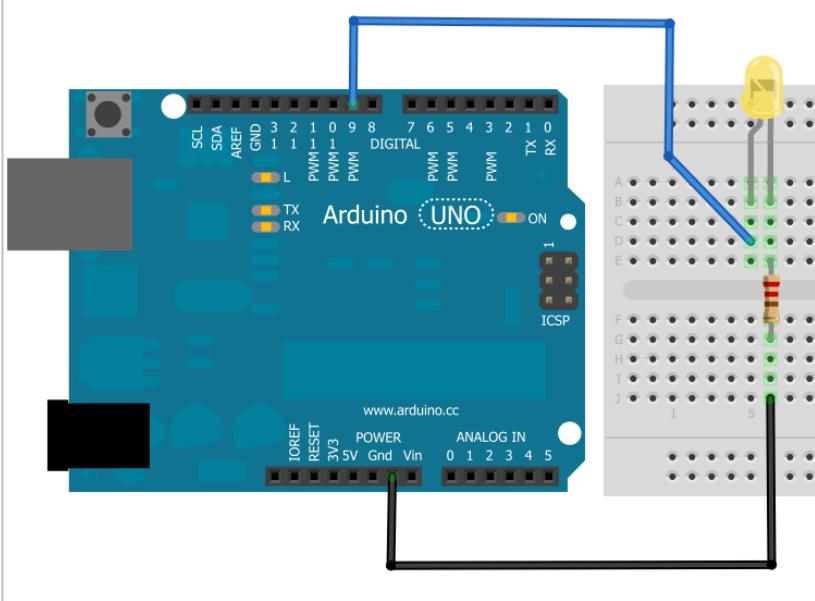


Conectores MM

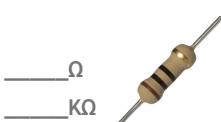




E S Q U E M A



1— ¿Cuál es el valor de esta resistencia?



2– ¿Qué hace está función?

analogWrite()

3— A que corresponden estos operadores

&&

1

1

!= _____



```

/*
----- Enciende/Apaga un LED de forma proporcional -----
----- Programa que enciende proporcionalmente un LED cuando
----- llega a su máximo punto de brillo comienza a apagarse
----- proporcionalmente.

*/
//----- Declara puertos de entradas y salidas y variables -----
//----- Pin donde se encuentra el LED, salida
int led = 9; //Pin donde se encuentra el LED, salida

//----- Funcion principal -----
//----- Se ejecuta cada vez que el Arduino se inicia
void setup () // Esta funcion se mantiene ejecutando
{
}

//----- Funcion ciclica -----
//----- Esta funcion se mantiene ejecutando
void loop () // cuando este energizado el Arduino

// Brillo va de minimo al maximo en incrementos de 5
for (int brillo = 0 ; brillo <= 255; brillo += 5) {
    // asigna el valor (rango es de 0 a 255):
    analogWrite(led, brillo);
    // espera 30 millisegundos para ver el efecto
    delay(30);
}
// Brillo va del maximo al minimo en incrementos de 5
for (int brillo = 255 ; brillo >= 0; brillo -= 5) {
    // asigna el valor (rango es de 0 a 255):
    analogWrite(led, brillo);
    // espera 30 millisegundos para ver el efecto
    delay(30);
}

//Fin programa

```

- 1- Estos elementos reciben señales de PWM y sirven para:



Bomba de agua

Variar la velocidad de bombeo

Micromotor

Variar la velocidad de giro

www.pololu.com



LED

Variar la intensidad de luz



Servomotor

Variar la posición en grados

```
/*
----- Enciende/Apaga un LED de forma proporcional -----
Programa que enciende proporcionalmente un LED cuando
llega a su máximo punto de brillo comienza a apagarse
proporcionalmente.

*/

```

```
program
  setup
    set integer variable [brillo v] to [0]
  loop
    repeat (10)
      repeat (1)
        commands
          LED with brightness [pin # 3 v] [level brillo]
          delay MILLIS [milliseconds] [10]
      end
      variable [brillo] by [1]
    end
    if [brillo v] = [255]
      repeat (1)
        commands
          LED with brightness [pin # 3 v] [level brillo]
          delay MILLIS [milliseconds] [10]
      end
    end
  end
end
```



¿Qué aprendo?

- Salida digital
- Control ON/OFF
- Comparación
- Condicional a partir de un valor entero de una entrada analógica

Conocimientos previos

- If/else
- Función digitalWrite() y analogRead()
- Valor de una entrada analógica
- Condicional y operadores de comparación

Materiales

1



Arduino UNO

1



LED

1



Protoboard

1



Cable USB Tipo AB

1



Potenciómetro 10K



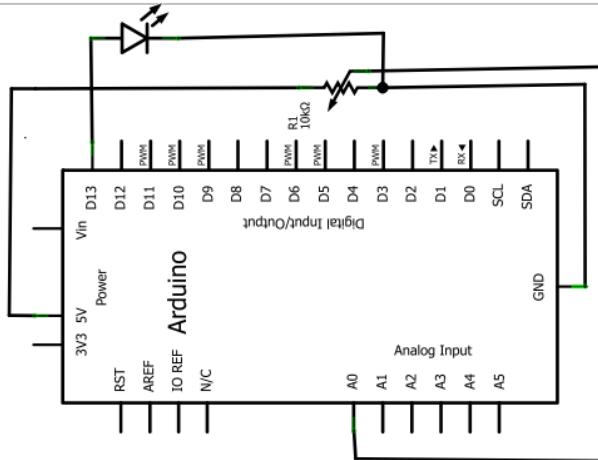
Resistencia 220Ω

3

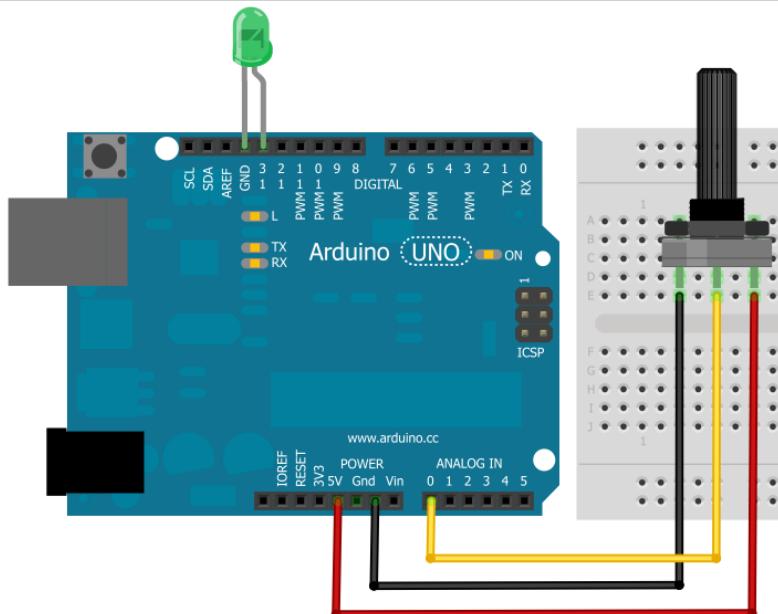


Conectores MM

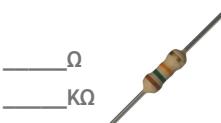




E S Q U E M A



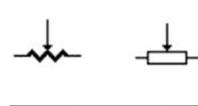
1— ¿Cuál es el valor de esta resistencia?



2— ¿Para que sirve el operador?



3— Este símbolo a que corresponde



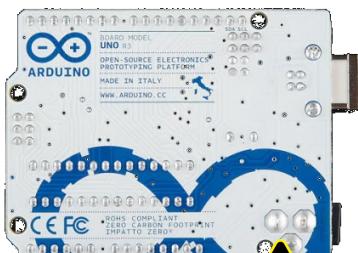
/*

Control ON/OFF con potenciómetro

Programa que enciende un LED cuando el valor de la entrada analógica comandada por el potenciómetro esta en cierto valor, cuando este valor cambia el LED se apaga, es un sistema con control ON/OFF

*/

```
-----  
//Funcion principal  
-----  
void setup() // Se ejecuta cada vez que el Arduino se inicia  
{  
    Serial.begin(9600); //Inicia comunicación serial  
    pinMode(13,OUTPUT); //Configurar el pin 13 como una salida  
}  
-----  
//Funcion ciclica  
-----  
void loop() // Esta funcion se mantiene ejecutando  
{  
    //Guardar en una variable el valor de la lectura analógica  
    int valor = analogRead(A0);  
    Serial.println(valor); //Imprime el valor por la consola  
    //Si el valor es mayor o igual a 500  
    if (valor >= 500)  
    {  
        digitalWrite(13,HIGH); //Enciende el LED en el pin 13  
    }  
    //Si el valor es menor a 500  
    else  
    {  
        digitalWrite(13,LOW); //Apaga el LED en el pin 13  
    }  
    delay(100); //Retardo de 100ms para ver los datos de la consola  
}  
//Fin programa
```



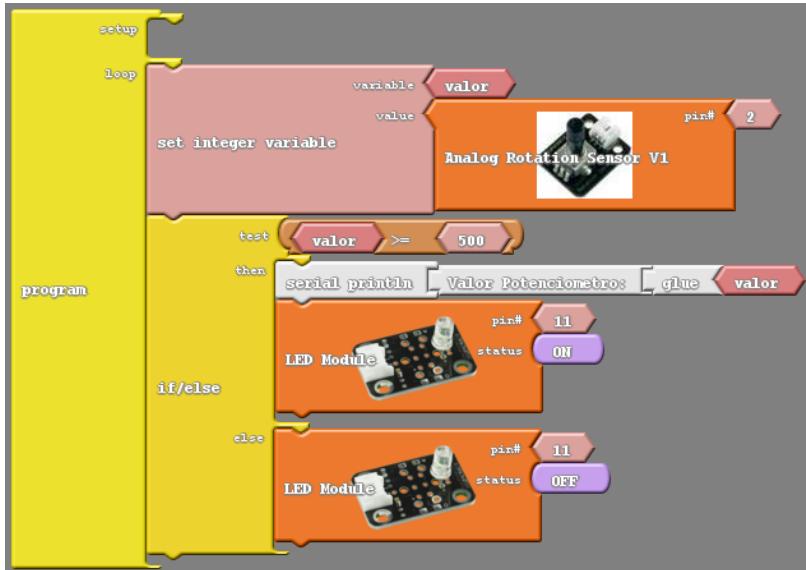
1- Debes tener mucha precaución al momento en que tu placa Arduino este energizada, si miras la palca por la parte inferior esta tiene todos sus puntos de soldadura al aire, si tienes la placa alimentada y la colocas encima de una superficie metálica, por ejemplo una mesa, es posible que la placa se dañe ya que queda en corto. Si vas a trabajar la placa Arduino te recomendamos trabajar sobre superficies de materiales aislantes como los son los sintéticos, madera o vidrio. Este es un consejo para que cuides tu placa.

/*

Control ON/OFF con potenciómetro

Programa que enciende un LED cuando el valor de la entrada analógica comandada por el potenciómetro está en cierto valor, cuando este valor cambia el LED se apaga, es un sistema con control ON/OFF

*/



¿Qué aprendo?

- Escritura por PWM en un LED
- Leer una entrada análoga por medio de una fotocelda
- Trabajar con una variable
- Ajustar una entrada análoga a una salida análoga

Conocimientos previos

- Señal análoga
- Función `analogWrite()` y `analogRead()`
- PWM
- Imprimir datos por consola serial

Materiales

Arduino UNO



LED



Protoboard

1



Cable USB Tipo AB

1



Fotocelda

1

1



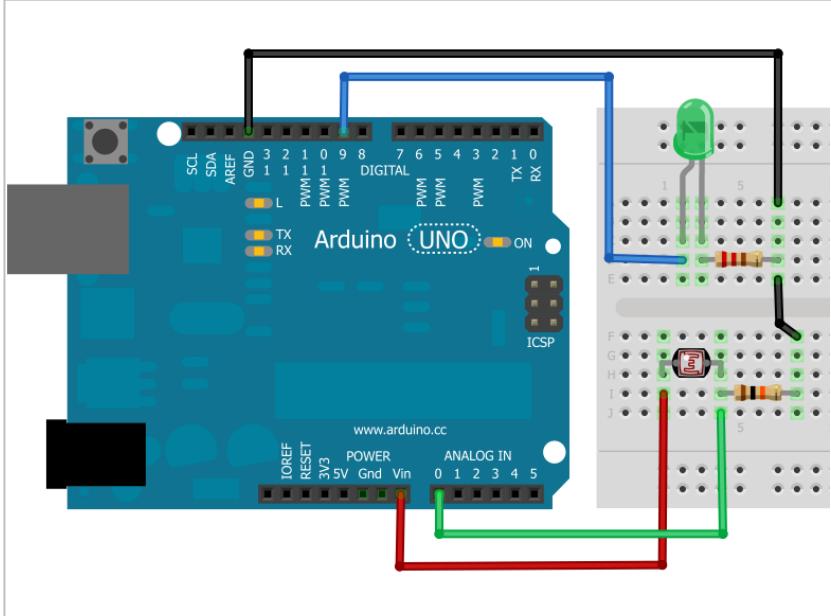
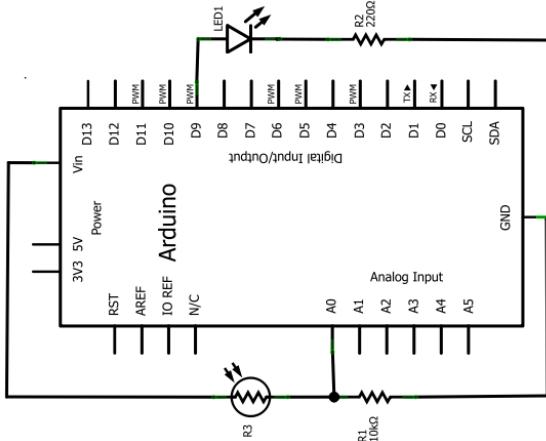
Resistencia 1K

5

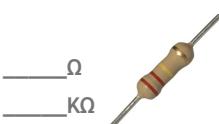


Conectores MM



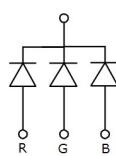


1— ¿Cuál es el valor de esta resistencia?



2— La letra A se guarda en una variable de tipo:

3— Este símbolo a que corresponde



```
/*
-----  
Control de intensidad de un LED  
-----  
  
Programa que enciende un LED de manera proporcional de  
acuerdo a la cantidad de luz que incide en una  
fotocelda.  
  
*/  
//-----  
//Funcion principal  
//-----  
void setup() // Se ejecuta cada vez que el Arduino se inicia  
{  
    Serial.begin(9600); //Inicia comunicación serial  
    pinMode(9,OUTPUT); //Configurar el pin 9 como una salida de PWM  
}  
  
//-----  
//Funcion ciclica  
//-----  
void loop() // Esta funcion se mantiene ejecutando  
{  
    //Guardar en una variable el valor de la lectura analoga de la  
    // fotocelda  
    int foto = analogRead(A0);  
  
    //Verifica el valor máximo y realizar una conversión  
    int conversion = 780 - foto;  
  
    //Condicional para establecer un valor absoluto  
    if (conversion < 0)  
        conversion = conversion * -1; //Multiplicar por -1 porque es negativo  
  
    //Imprimir datos del valor de la lectura analoga de la fotocelda  
    Serial.print("Foto : ");  
    Serial.print(foto);  
    Serial.println("");  
  
    //Imprimir datos del valor de la conversión  
    Serial.print("Conv : ");  
    Serial.print(conversion);  
    Serial.println("");  
  
    //Escritura analoga de PWM en el LED de acuerdo a la conversión  
    analogWrite(9, conversion);  
  
    delay(100); //Retardo para datos en la consola  
}  
  
//Fin programa
```



Calculadora

Cargar batería y apagado automático



Cámara digital

Verifica si hay necesidad de foto con flash



Medidor de luz

Mide una cantidad de luz y lo da en lux (lx)



Alumbrado público

Enciende la luz cuando llega la oscuridad



```
/*
Control de intensidad de un LED
-----  
Programa que enciende un LED de manera proporcional de acuerdo a la cantidad de luz que incide en una fotocelda.
```

```
program
  setup
    loop
      set integer variable [foto v] to [analog pin #0]
      if (foto > 700) then
        set digital pin [11 v] to [LOW]
      else
        set digital pin [11 v] to [HIGH]
      end
      serial println [message] [glue foto]
    end
  end
```



ARDUINO
MAZATLÁN

¿Qué aprendo?

- Incrementar una variables
- Condicional If/else anidado
- Anti-rebote de un pulsador
- Leer una entrada digital y escribir una salida digital a determinada condición

Conocimientos previos

- Señal digital
- Función digitalWrite() y digitalRead()
- Imprimir datos por consola
- Declarar variables enteras

Materiales

1



Arduino UNO

1



LED

1



Protoboard

1



Cable USB Tipo AB

1



Pulsador



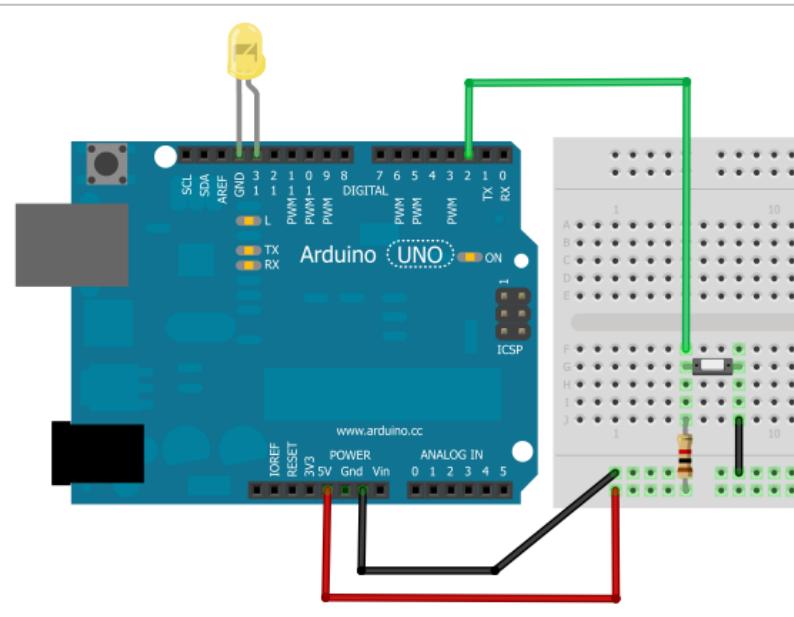
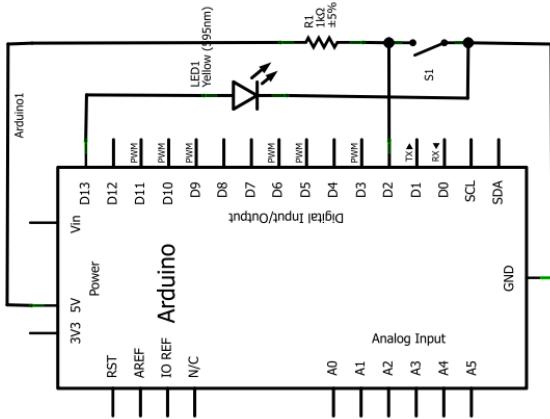
Resistencia 1K

4



Conectores MM





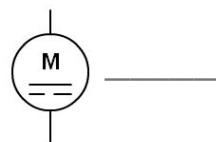
1— ¿Cuál es el valor de esta resistencia?



2— ¿Cuáles son los estados de una variable booleana?

- a. _____
- b. _____

3— Este símbolo a que corresponde



```

C /*-----  
Ó Contador de pulsos  
D -----  
I Programa que muestra por pantalla (consola serial) el número  
G de veces que el pulsador ha sido presionado, se realiza un  
P proceso que de acuerdo al número de pulsaciones se enciende  
R un LED,  
O Cosas de Mecatrónica y Tienda de Robótica  
D */  
E //-----  
P //Declara puertos de entradas y salidas y variables  
R //-----  
A int conta = 0; //Variable para guardar el conteo de los pulsos  
M //-----  
C //Funcion principal  
A //-----  
R void setup() // Se ejecuta cada vez que el Arduino se inicia  
O {  
P Serial.begin(9600); //Inicia comunicación serial  
R pinMode(2,INPUT); //Configura el pin 2 como una entrada, pulsador  
A pinMode(13,OUTPUT); //Configura el pin 13 como una salida, LED  
M }  
C //-----  
A //Funcion ciclica  
C //-----  
R void loop() // Esta funcion se mantiene ejecutando  
O { // cuando este energizado el Arduino  
A // Si el pulsador esta oprimido  
C if ( digitalRead(2) == HIGH )  
A { // Si el pulsador no esta oprimido, flanco de bajada  
C if ( digitalRead(2) == LOW )  
A { conta++; //Incrementa el contador  
C Serial.println(conta); //Imprime el valor por consola  
A delay (100); // Retardo  
C }  
A // Si el valor del contador es 5  
C if (conta==5)  
A { digitalWrite(13,HIGH); //Enciende el LED  
C }  
A // Si el valor del contador es 8  
C if (conta==8)  
A { digitalWrite(13,LOW); // Apaga el LED  
C }  
} //Fin programa

```

T
I
P
S

- 1- Arduino tiene una gran comunidad de aprendizaje y para compartir diversas preguntas, en el foro oficial puedes encontrar diversos temas con gran cantidad de respuestas:

Main Site | Blog | PlayGround | Forum | Links | Stats | Help | Sign In or Register | Search

Welcome, Guest! Please [log in or register](#).
April 06, 2013, 06:28:33 PM
are you looking for our old forums?

Using Arduino

| Topic | Posts | Replies |
|--|-------|---------|
| Installation & Troubleshooting For problems with Arduino itself, NOT your project. Last post by Nik Connon on Today at 07:59:55 PM | 13938 | 2691 |
| Program Coding Advice on general approaches or feasibility. Last post by Matt on Today at 09:04:44 PM | 45192 | 6106 |
| Power Understanding the language, error messages, etc. Last post by What's wrong with me... by joseantonio on Today at 09:21:19 PM | 66648 | 7901 |
| General Electronics Resistors, capacitors, breadboards, soldering, etc. Last post by Measuring Voltage... by Ruggerito on Today at 09:24:11 PM | 21950 | 2604 |
| Microcontroller Standalone or alternative microcontrollers, in-system programming, bootloaders, etc. Last post by Using the 1284p/164... by Crossroads on Today at 09:27:31 PM | 10988 | 1227 |

arduino.cc/forum

```

/*
----- Contador de pulsos -----
Programa que muestra por pantalla (consola serial) el número
de veces que el pulsador ha sido presionado, se realiza un
proceso que de acuerdo al número de pulsaciones se enciende
un LED,
*/

```

```

Main
setup [set integer variable (conta) to (0)]
loop
  test [digital pin # (12) = (HIGH)]
    then
      test [digital pin # (12) = (LOW)]
        then
          if [if (conta) = (5)]
            then
              if [if (conta) = (8)]
                then
                  [LED Module (pin# (11) status (ON))]
                else
                  [LED Module (pin# (11) status (OFF))]
              end
              [set integer variable (conta) to (0)]
            end
          end
        end
      end
    end
  end
end

```

Pulsos:1
Pulsos:2
Pulsos:3
Pulsos:4
Pulsos:5
Pulsos:6
Pulsos:7
Pulsos:8

¿Qué aprendo?

- Manejar el código de colores RGB
- Uso de una función
- PWM a tres salidas
- Manipular una variable

Conocimientos previos

- Señal digital
- Función analogWrite()
- LED de anodo común
- Estructura de un programa en Arduino

Materiales

1



Arduino UNO

1



LED RGB

1



Protoboard

1



Cable USB Tipo AB

3



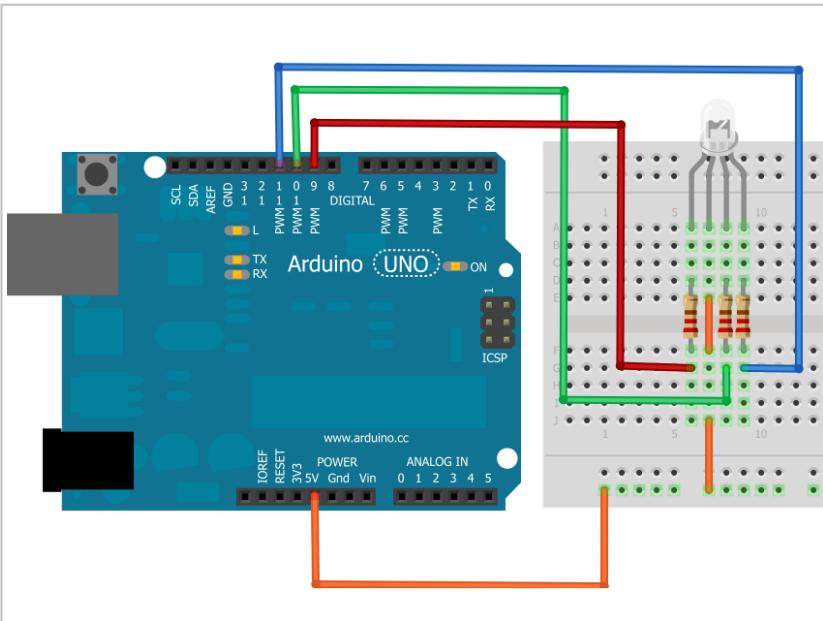
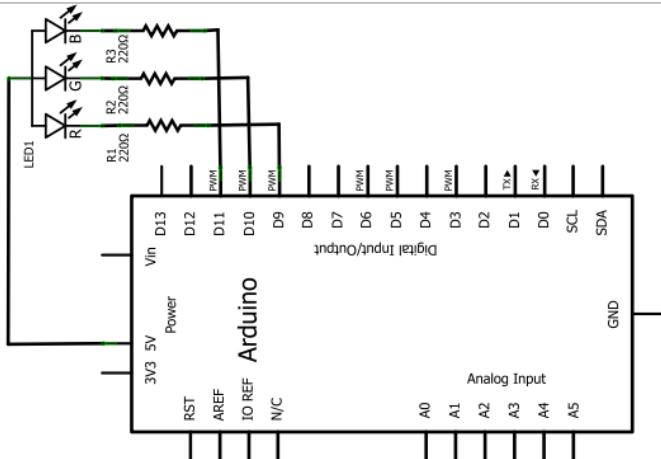
Resistencia 220

5



Conectores MM



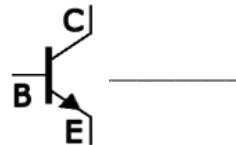


1— ¿Cuál es el valor de esta resistencia?



2— La palabra “pulso” se debe guardar en una variable de tipo :

3— Este símbolo a que corresponde



```

C      /*-----*
Ó      LED RGB - Tabla de Colores-----*
D      Programa que hace uso de una función llamada
I      color para generar diversas tonalidades en
G      un LED RGB
      */
D
E
P
R
O
G
R
A
M
A
C
I
Ó
N
      int ledRojo = 9;
      int ledVerde = 5;
      int ledAzul = 6;

      void Rojo();
      void Blanco();
      void Magenta();
      void Azul();
      void Verde();
      void Naranja();

      void setup()
      {
          pinMode( ledRojo , OUTPUT);
          pinMode( ledVerde , OUTPUT);
          pinMode( ledAzul , OUTPUT);
      }

      void loop()
      {
          Rojo();
          delay( 1000 );
          Verde();
          delay( 1000 );
          Azul();
          delay( 1000 );
          Blanco();
          delay( 1000 );
          Magenta();
          delay( 1000 );
          Naranja();
          delay( 1000 );
      }

      void Verde()
      {
          analogWrite(ledRojo , 255);
          analogWrite(ledVerde , 0);
          analogWrite(ledAzul , 255);
      }

      void Magenta()
      {
          analogWrite(ledRojo , 0);
          analogWrite(ledVerde , 255);
          analogWrite(ledAzul , 0);
      }

      void Blanco()
      {
          analogWrite(ledRojo , 0);
          analogWrite(ledVerde , 0);
          analogWrite(ledAzul , 0);
      }

      void Rojo()
      {
          analogWrite(ledRojo , 0);
          analogWrite(ledVerde , 255);
          analogWrite(ledAzul , 255);
      }

      void Azul()
      {
          analogWrite(ledRojo , 255);
          analogWrite(ledVerde , 255);
          analogWrite(ledAzul , 0);
      }

      void Naranja()
      {
          analogWrite(ledRojo , 0);
          analogWrite(ledVerde , 127);
          analogWrite(ledAzul , 255);
      }

```

T
I
P
S

- 1- Estos dos dispositivos pueden generar hasta un billón de colores distintos



MegaBrite



ShiftBrite



C
Ó
D
I
G
O

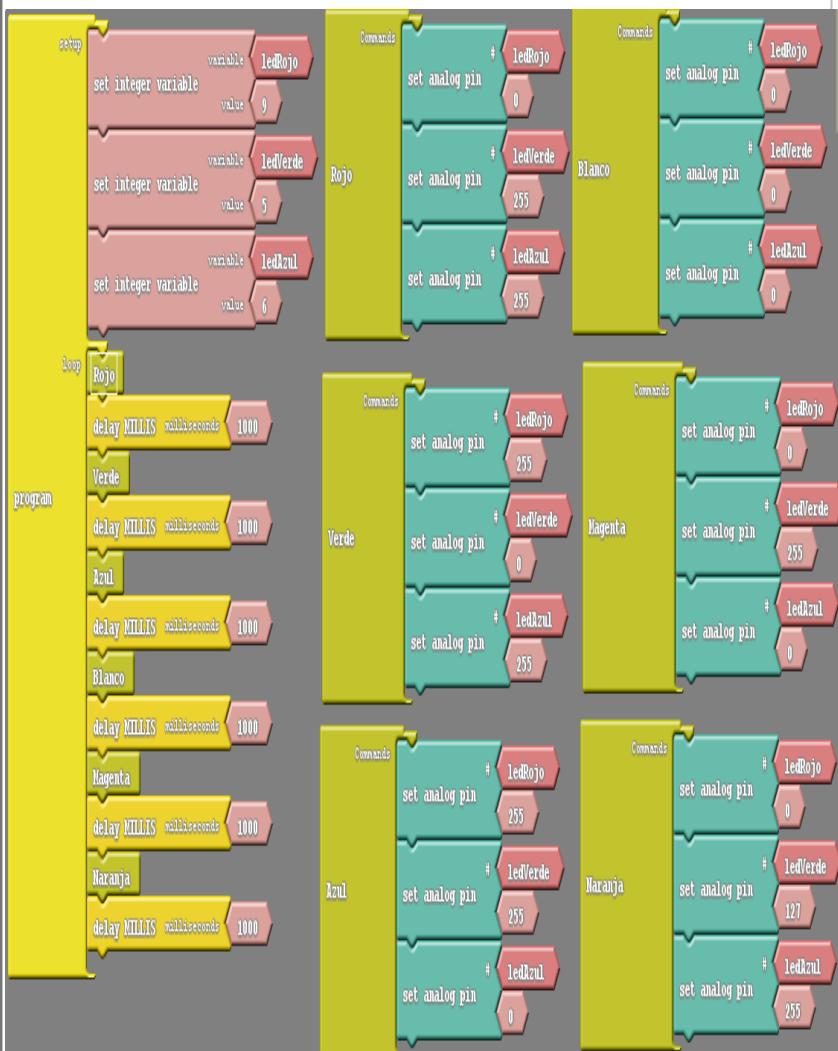
D
E

P
R
O
G
R
A
M
A
C
I
ÓN

```
/*  
LED RGB - Tabla de Colores
```

Programa que hace uso de una función llamada color para generar diversas tonalidades en un LED RGB

```
*/
```



T11 Control ON/OFF de un motor

¿Qué aprendo?

- Control ON/OFF
- Condicional If/else
- Conectar un motor DC por transistor
- Condicionales a partir del estado del pulsador

Conocimientos previos

- Señal digital
- Función digitalWrite() y digitalRead()
- Divisor de voltaje
- Reconocer un transistor y un motor

Materiales

1



Arduino UNO

1



Transistor NPN

1



Protoboard

1



Cable USB Tipo AB

1



Pulsador

1



Resistencia 1K

2

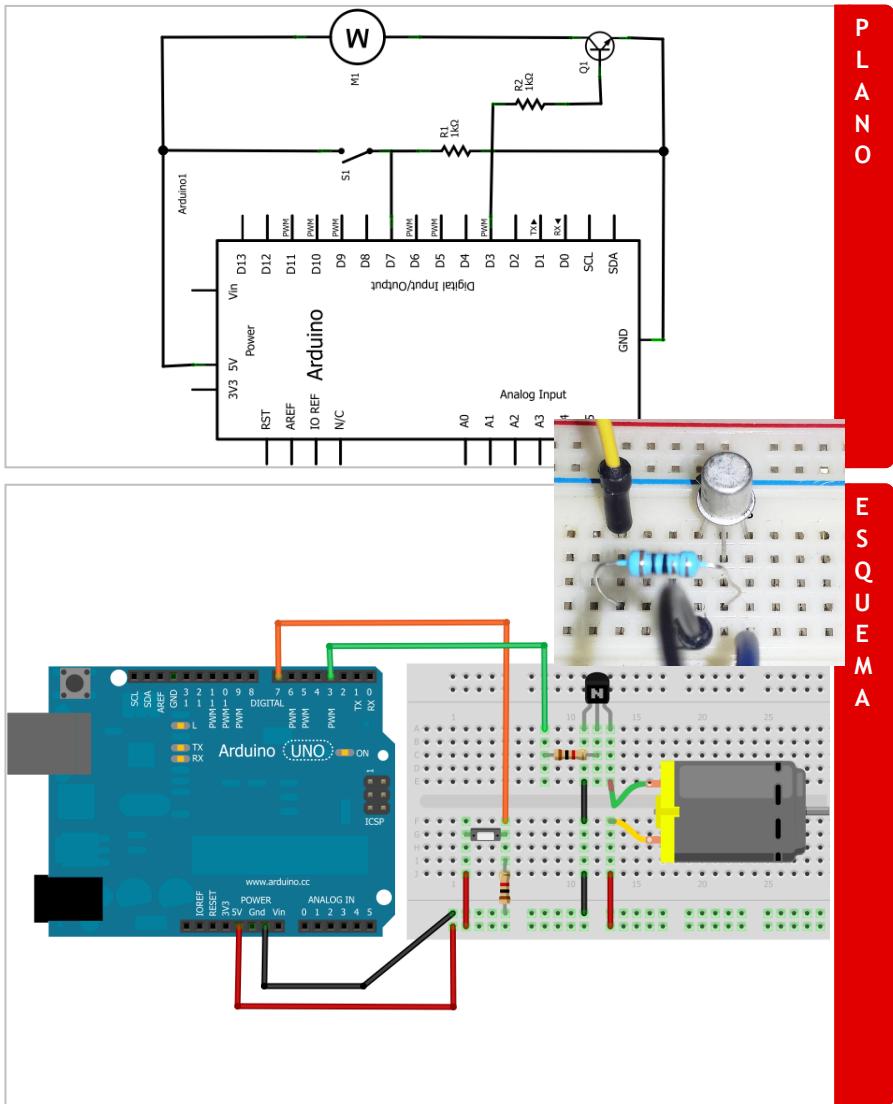


8

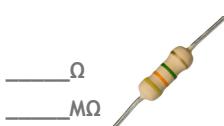


Conectores MM



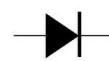


1— ¿Cuál es el valor de esta resistencia?



2— Si quiero guardar el número π que tipo de variable debo usar?

3— Este símbolo a que corresponde



```
/*
-----Control ON/OFF de un motor-----
Programa que hace uso de un motor y un pulsador,
mientras se mantenga pulsado, el motor debe
estar encendido (ON) de lo contrario debe estar
apagado (OFF)

Cosas de Mecatrónica y Tienda de Robótica
*/

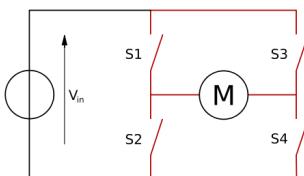
//-----Declarar puertos de entradas y salidas y variables-----
//-----Funcion principal-----
void setup() // Se ejecuta cada vez que el Arduino se inicia
{
    pinMode(pulsador,INPUT); //El pulsador como una entrada
    pinMode(motor,OUTPUT); //El motor como una salida
}

//-----Funcion ciclica-----
void loop() // Esta funcion se mantiene ejecutando
// cuando este energizado el Arduino

    // Si el pulsador se encuentra oprimido
    if(digitalRead(pulsador) == HIGH){
        digitalWrite(motor,HIGH); //Enciende el motor
    }else{ //si el pulsador no esta oprimido
        digitalWrite(motor,LOW); //Apaga el motor
    }
}

// Fin programa
```

- 1- Un Puente H es un circuito electrónico que permite a un motor DC girar en ambos sentidos, *avance* y *retroceso*. Son ampliamente usados en robótica y como convertidores de potencia.



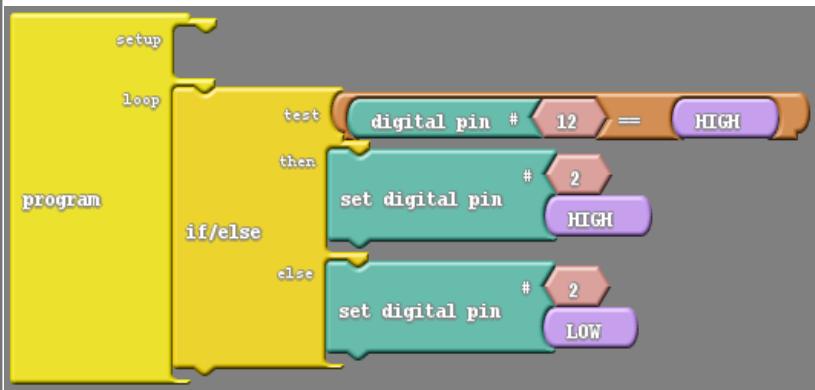
Los puentes H están disponibles como circuitos integrados, pero también pueden construirse a partir de componentes discretos. El término "puente H" proviene de la típica representación gráfica del circuito. Un puente H se construye con 4 interruptores (mecánicos o mediante transistores). Cuando los interruptores S1 y S4 están cerrados y S2 y S3 abiertos se aplica voltaje positivo en el motor, haciéndolo girar en un sentido. Abriendo los interruptores S1 y S4 y cerrando S2 y S3, el voltaje se invierte, permitiendo el giro en sentido inverso del motor. Con la nomenclatura que estamos usando, los interruptores S1 y S2 nunca podrán estar cerrados al mismo tiempo, porque esto cortocircuitaría la fuente de tensión. Lo mismo sucede con S3 y S4.



Este circuito es fundamental para controlar motores DC en aplicaciones de robótica y automoción. Permite controlar tanto la velocidad como la dirección del motor, lo que es esencial para la operación de muchos tipos de vehículos y robots. Los puentes H se utilizan ampliamente en la industria por su eficiencia y simplicidad de diseño.



```
/*
-----  
Control ON/OFF de un motor  
-----  
  
Programa que hace uso de un motor y un pulsador,  
mientras se mantenga pulsado, el motor debe  
estar encendido (ON) de lo contrario debe estar  
apagado (OFF)  
  
Cosas de Mecatrónica y Tienda de Robótica  
*/
```



T12 Control por PWM de un motor

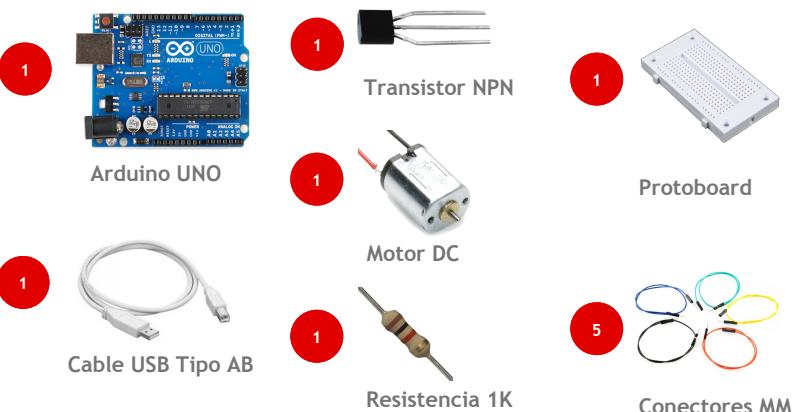
¿Qué aprendo?

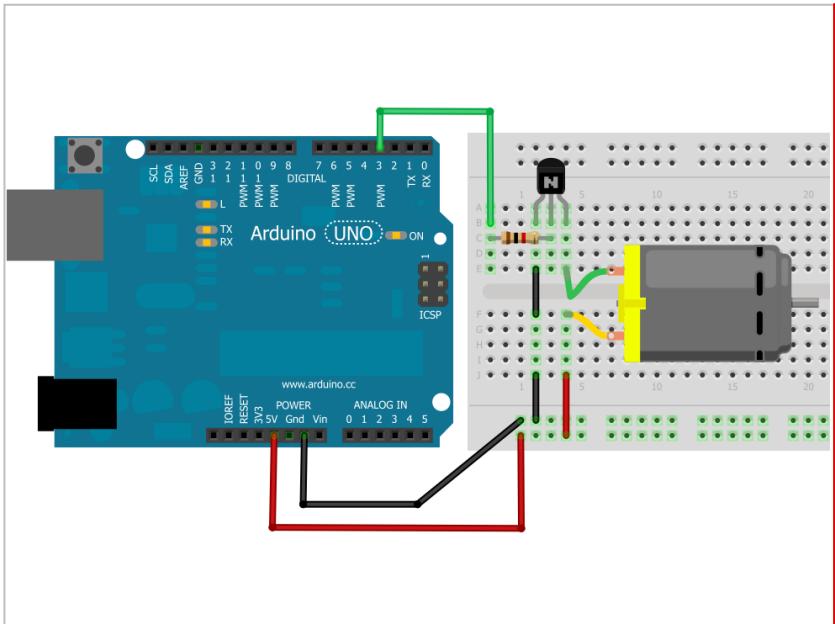
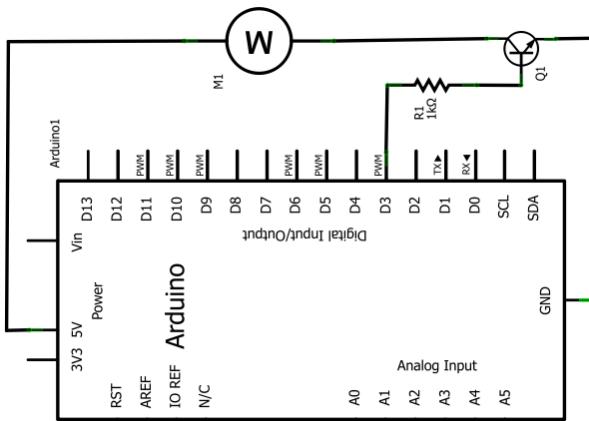
- Leer datos de la Consola Serial
- Manejo de la función map()
- Variar el PWM para producir 5 velocidades distintas
- Etapa de potencia para un motor a través de transistor

Conocimientos previos

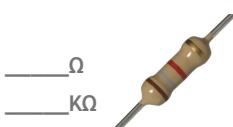
- Señal análoga
- Función analogWrite() y Serial.print()
- PWM
- Condicional y operadores de comparación

Materiales





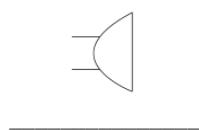
1— ¿Cuál es el valor de esta resistencia?



2— ¿Cómo se escriben los siguientes números en binario?

- 7 = _____
5 = _____
2 = _____

3— Este símbolo a que corresponde



```

/*
Control por PWM de un motor
-----

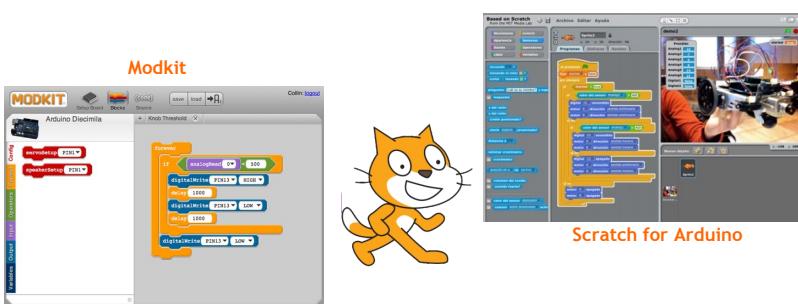

Programa que hace uso de un motor y la Consola serial de Arduino, tiene la posibilidad de configurar al motor 5 velocidades distintas, desde el teclado del PC puedes enviarle la velocidad deseada. Las 5 velocidades se configuran con 5 PWM distintos.

*/
//-----
//Declara puertos de entradas y salidas y variables
//-
int motor=3;      //Declara Pin del motor
//-----
//Funcion principal
//-
void setup() // Se ejecuta cada vez que el Arduino se inicia
{
  Serial.begin(9600); //Inicia la comunicacion serial Arduino-PC
}
//-----
//Funcion ciclica
//-
void loop() // Esta funcion se mantiene ejecutando
{           // cuando este energizado el Arduino
  // Si hay algun valor en la Consola Serial
  if (Serial.available())
    //Variable donde se guarda el caracter enviado desde teclado
    char a = Serial.read();
    // Si el caracter ingresado esta entre 0 y 5
    if (a>='0' && a<='5')
      //Variable para escalar el valor ingresado a rango de PWM
      int velocidad = map(a,'0','5',0,255);
      //Escritura de PWM al motor
      analogWrite(motor,velocidad);
      //Mensaje para el usuario
      Serial.print("El motor esta girando a la velocidad ");
      Serial.println(a);
    }else{ // Si el caracter ingresado NO esta entre 0 y 5
      //Mensaje para el usuario
      Serial.print("Velocidad invalida");
      Serial.println(a);
    }
  }
}

//Fin programa

```

1- Arduino también se puede programar en lenguajes gráficos, por ejemplo:



```

setup
loop
if then
  test serial data available > 0
  if then
    variable velocidad
    value serial read
  end
  test velocidad >= 0 and velocidad <= 5
  then
    set analog pin #5
    value velocidad
    from 0
    map 5 to 0
    to 255
    serial println El motor esta girando a la velocidad: glue velocidad
  end
  else
    serial println Velocidad invalida: glue velocidad
  end
end
delay MILLIS milliseconds 1000

```

Main



ARDUINO
MAZATLÁN

T13 Generar tonos con un buzzer

¿Qué aprendo?

- Manejo de variables de tipo entera
- Usar funciones especiales de Arduino
- Generar diversos tonos
- Producir salidas de frecuencia

Conocimientos previos

- Señal digital y análoga
- Función map() y analogRead()
- Enviar parámetros a las funciones
- Retardos a través de delay()

Materiales

1



Arduino UNO

1



Buzzer

1



Protoboard

1



Cable USB Tipo AB

1



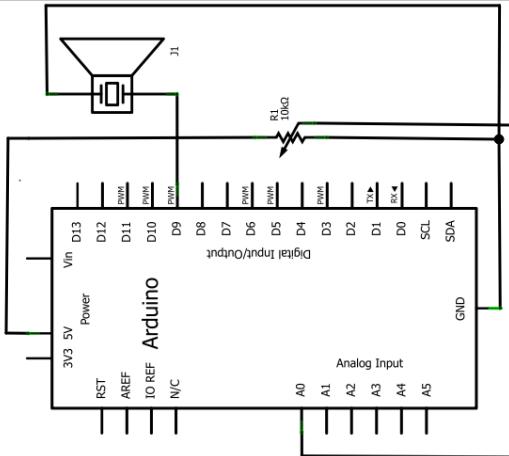
Potenciómetro 10K

6

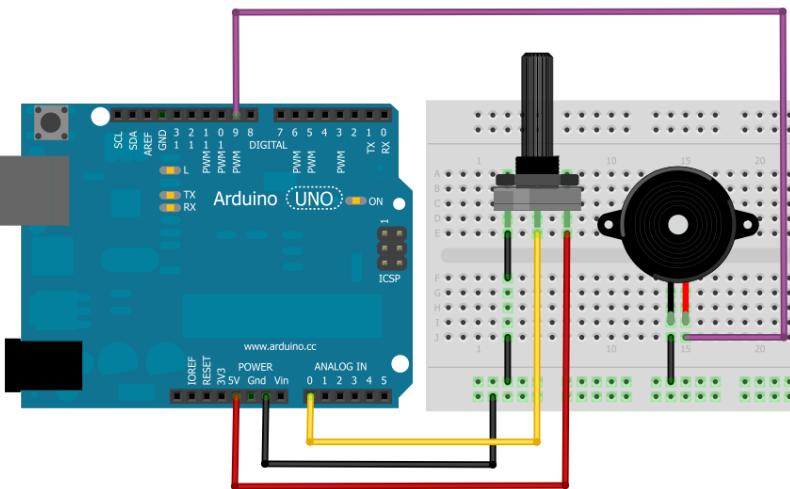


Conectores MM



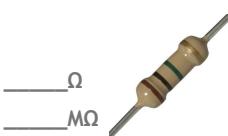


E S Q U E M A

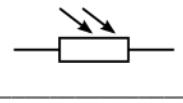


PREGUNTA

1– ¿Cuál es el valor de esta resistencia?



2- El switch...case
es un tipo de



3— Este símbolo a qué corresponde

```
/*
-----  
Generar tonos con un buzzer  
-----
```

Programa que hace uso de un buzzer (chicharra) y un potenciómetro, la idea es generar diversos tonos en el buzzer a partir del estado análogo del potenciómetro. Además se hace uso de la función tone que es muy útil para generar diversas melodías

```
//-----  
//Declara puertos de entradas y salidas y variables  
//-----  
int buzzer = 9; //Declara Pin del buzzer  
int tono = 0; //Declara Pin del potenciómetro  
  
//-----  
//Funcion principal  
//-----  
void setup() // Se ejecuta cada vez que el Arduino se inicia  
{  
    // No se configuran parámetros inciales, pero se debe  
    // colocar el encabezado de la función setup()  
}  
  
//-----  
//Funcion ciclica  
//-----  
void loop() // Esta función se mantiene ejecutando  
{  
    //Variable entera donde se almacena el valor del potenciómetro  
    int sensor = analogRead(tono);  
  
    //Variable donde se escala la frecuencia de 100 a 5000Hz  
    int frecuencia = map(sensor,0,1023,100,5000);  
  
    //Variable entera para guardar el tiempo deseado en ms  
    int duracion = 250;  
  
    //Función tone(), que recibe:  
    // 1ra posición: Pin del elemento sonoro  
    // 2da posición: Frecuencia deseada en Hz  
    // 3ra posición: Duración del tono  
    tone(buzzer, frecuencia, duracion);  
  
    //Retardo  
    delay(100);  
}  
  
//Fin programa
```

1- Si tienes dificultades para aprender los colores de las resistencias una ayudita online no te caería nada mal ;) en reflexiona.biz puedes hacer esto:

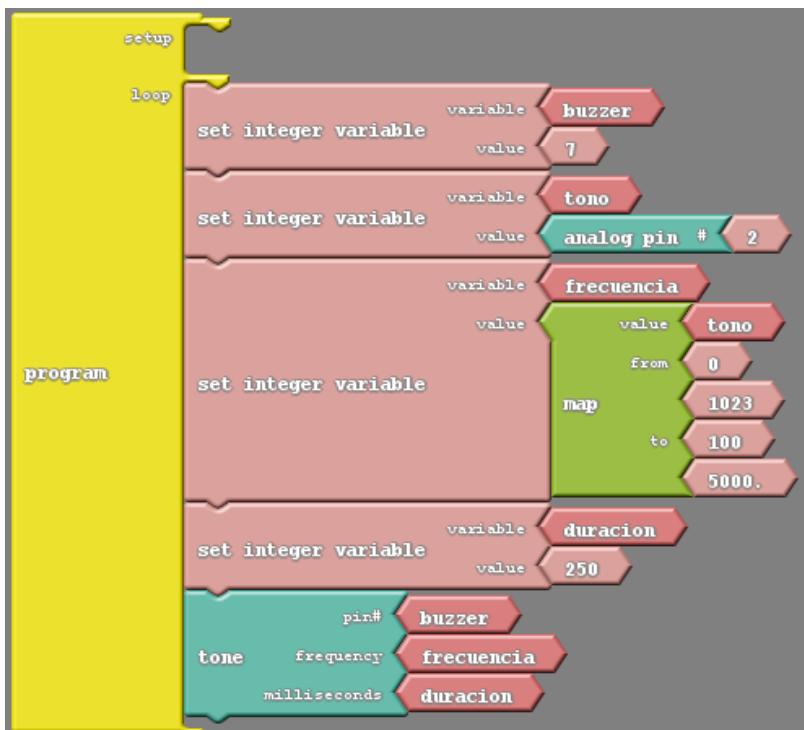


2- Descargar

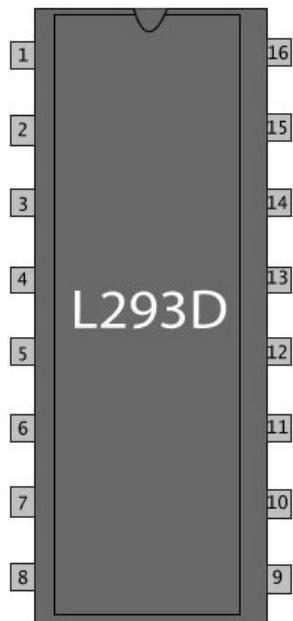


```
/*
-----  
Generar tonos con un buzzer  
-----
```

Programa que hace uso de un buzzer (chicharra) y un potenciómetro, la idea es generar diversos tonos en el buzzer a partir del estado análogo del potenciómetro. Además se hace uso de la función tone que es muy útil para generar diversas melodías

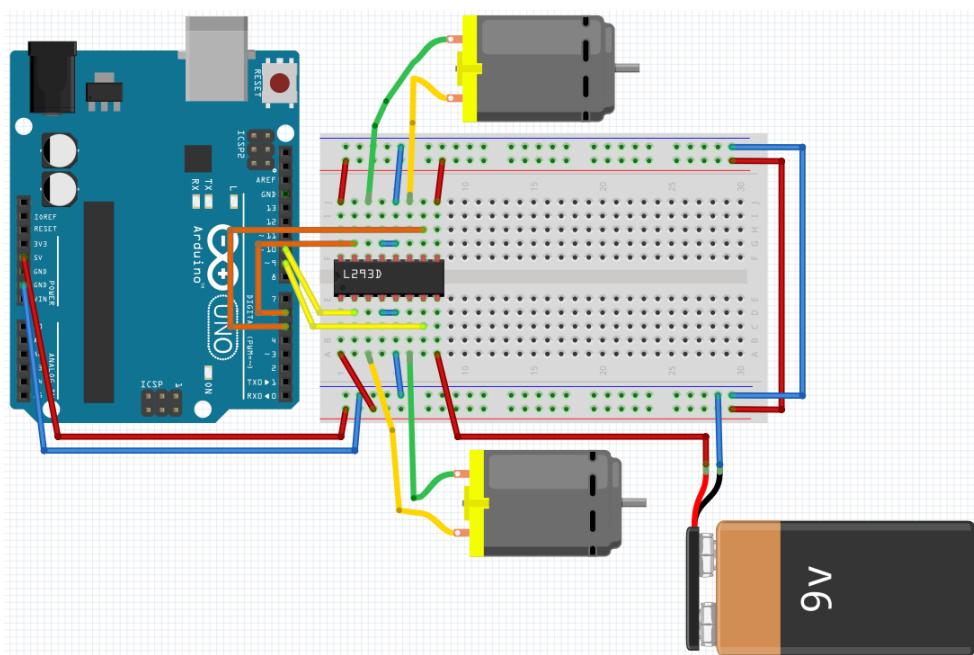


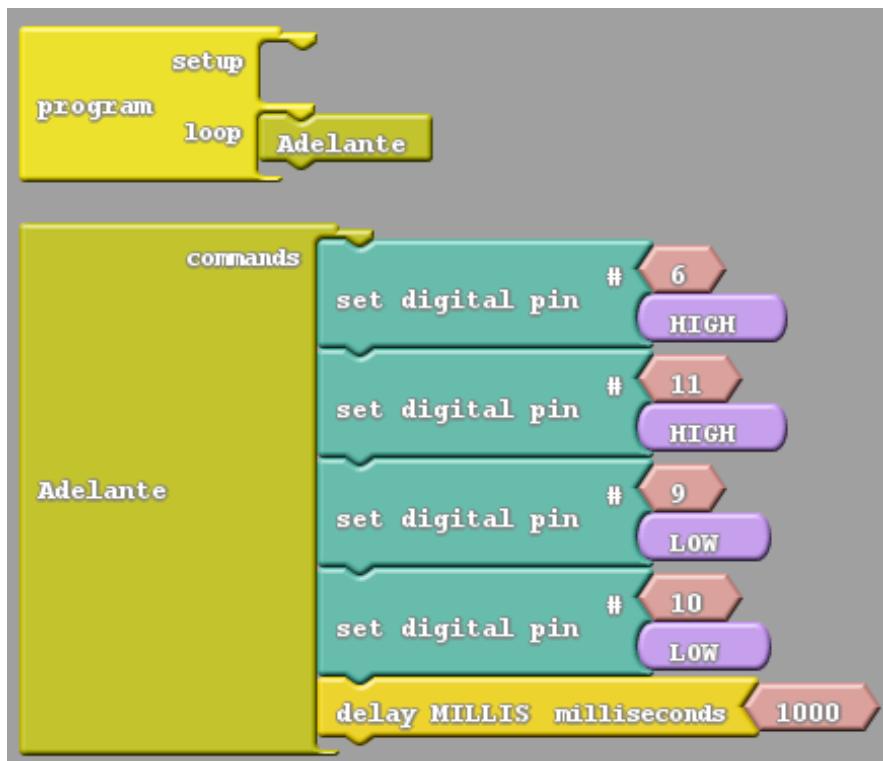
| ARDUINO | L293D(Puente H) |
|---------|-----------------|
| 5 | 10 |
| 6 | 15 |
| 9 | 7 |
| 10 | 2 |
| 5V | 1, 9, 16 |
| GND | 4, 5, 12, 13 |



El motor 1 (Derecho) se conecta a los pines 3 y 6 del Puente H El motor 2 (Izquierdo) se conecta a los pines 11 y 14 del Puente H

La fuente de alimentacion de los Motores se conecta a tierra y el positivo al pin 16 del puente H.



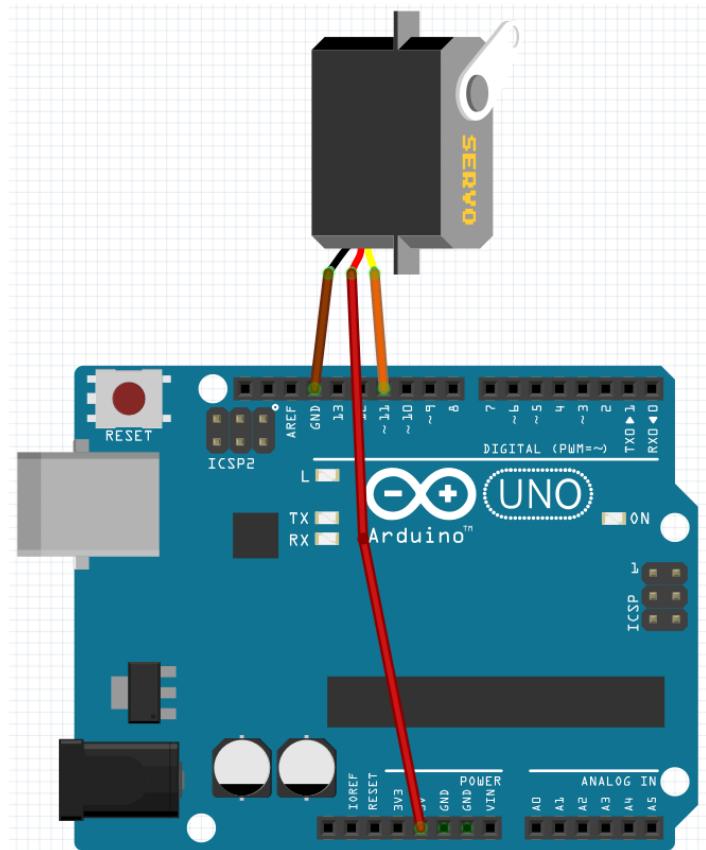


Reto 1: Agrega la programación necesaria, para que el programa haga el siguiente ciclo siguiente:

- 1.- Avance por 1000Mils y Pare.
 - 2.- Gire el Motor derecho por 1000Mils hacia adelante, Pare y luego de reversa y Pare.
 - 2.- Gire el Motor izquierdo por 1000Mils hacia adelante, Pare y luego de reversa y Pare.
 - 4.- De reversa por 1000Mils y Pare.

Reto 2: Modifique el programa para que acepte comandos por la terminal y haga lo siguiente:

a=Avanzar, b=Izquierda, c= Parar, d=Derecha, e=Reversa

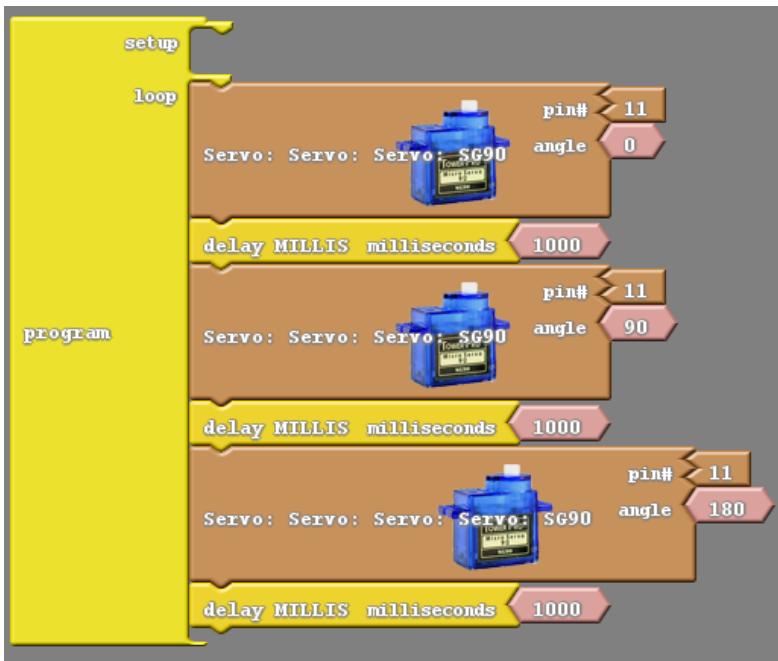


Nota:

Cafe = GND

Rojo= 5V

Anaranjado = pin# PWM

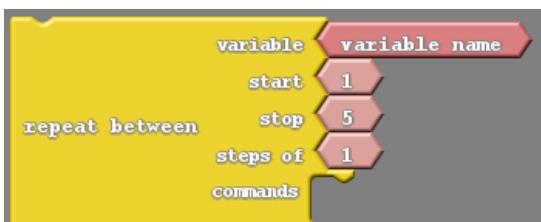


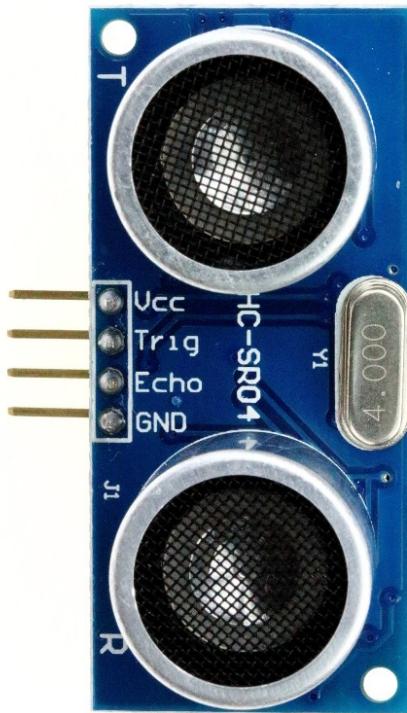
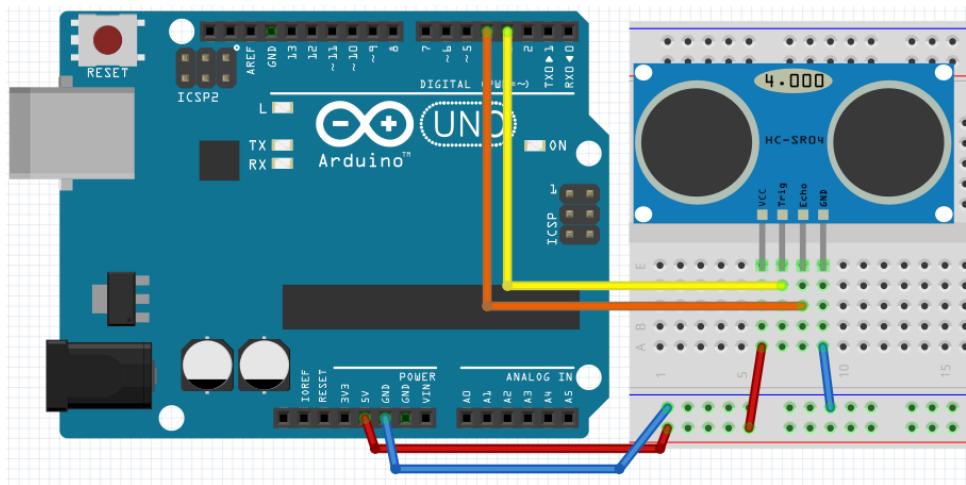
Reto:

Modifica el programa para que:

1.- El servo motor gire de 0 a 180 grados en periodos de 45 grados de ida y vuelta.

2.- Usando el bloque repeat bet een el servo gire de 0 a 180 grados de ida y vuelta, un grado a la vez.





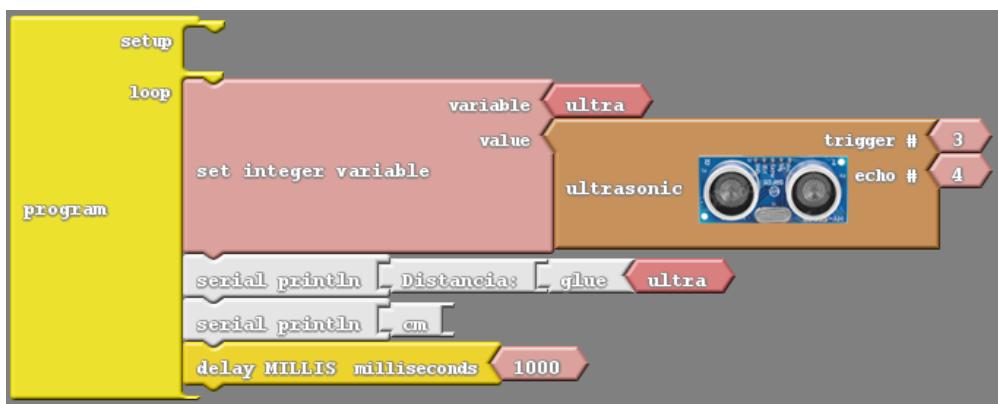
Nota:

VCC = 5v

Trig = pin 3

Echo = pin 4

GND = GND



Reto:

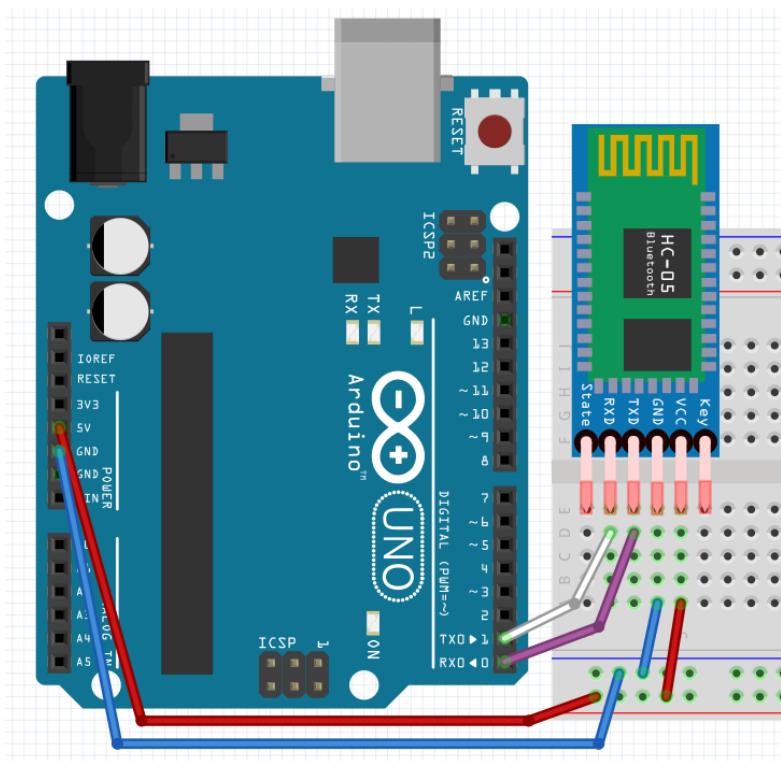
Modifica el programa para que:

1.- Encienda un led:

Verde a los 15cm.

Amarillo a los 10cm.

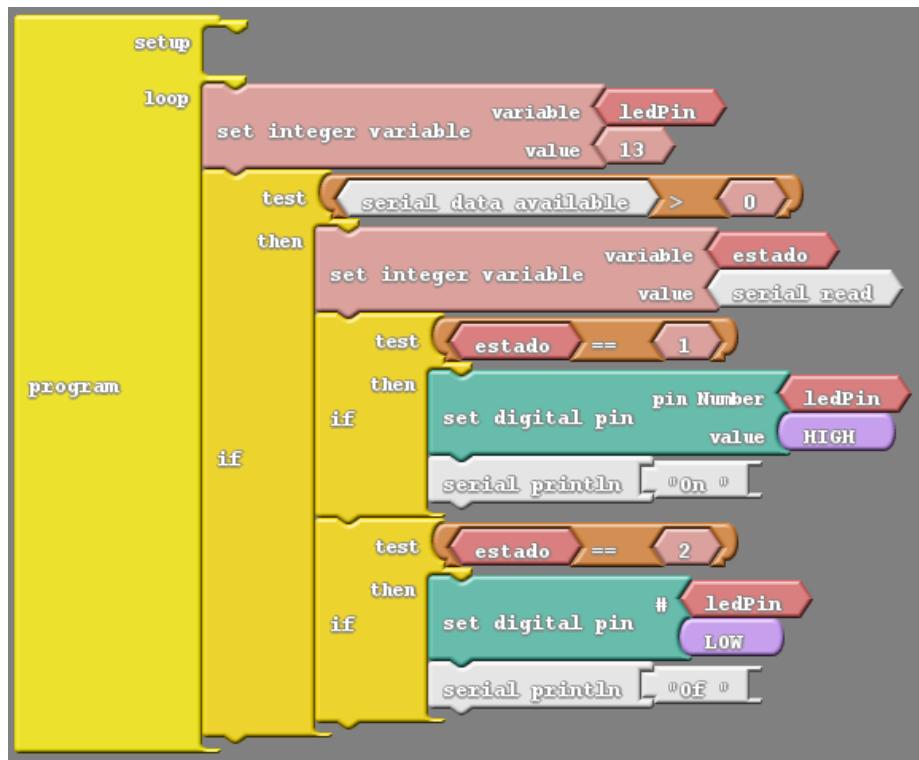
Rojo a los 5cm y ademas suene el buzzer.



Nota:

| | |
|-----------|---------|
| Sensor | Arduino |
| Bluetooth | R |
| R | |
| VCC | 5v |
| GND | GND |

Recuerde desconectar el sensor Bluetooth de los Pines RX-TX, cada vez que desee subir un nuevo programa a la tarjeta Arduino.



Reto:

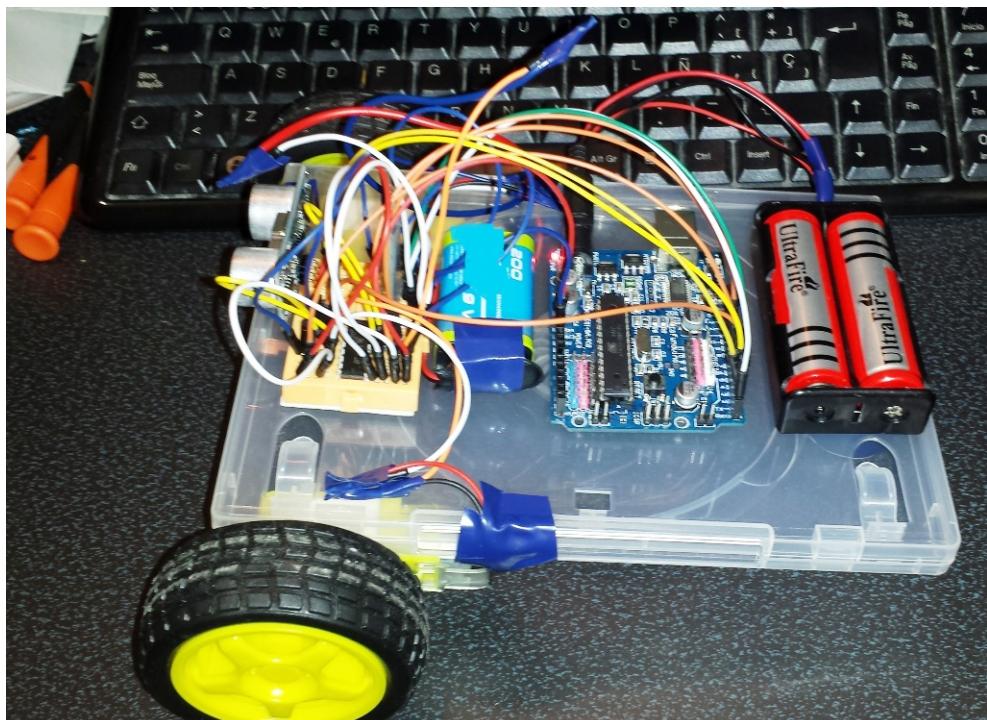
Modifique el programa para que:

- 1.- Usando el puente H Encienda 2 motores a un velocidad de 145 de 255.

Recuerde desconectar el sensor Bluetooth de los Pines RX-TX, cada vez que desee subir un nuevo programa a la tarjeta Arduino.

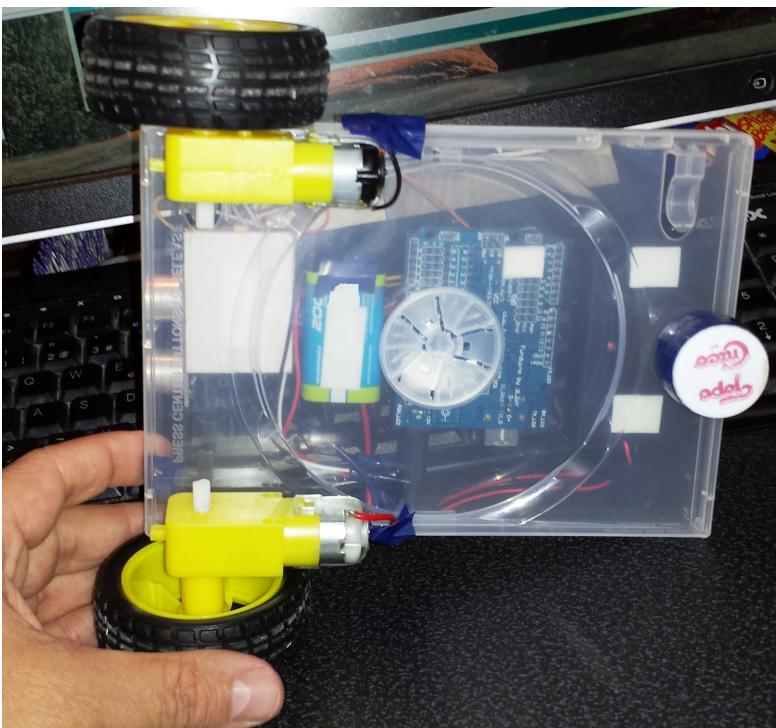
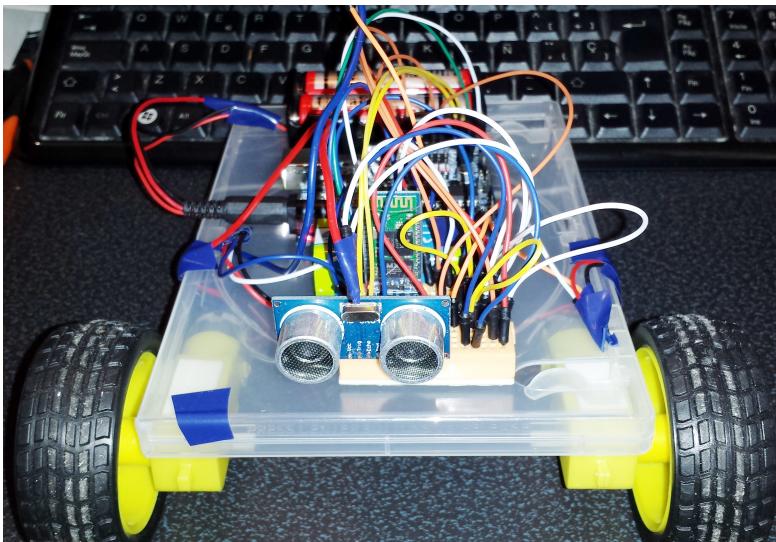
Proyecto integrador.

Arme el circuito del Carro Robot detector de obstaculos con Control Remoto por Bluetooth, usando las imágenes de ejemplo:



PI Proyecto Integrador

Proyecto integrador.



Proyecto integrador.

Una vez armado el circuito del carro robot, se deberá crear el programa que lo haga funcionar, para lo cual se deberá considerar lo siguiente:

a) Declarar las variables necesarias para:

velocidad, estado, Motor izqA (avanzar), Motor izqB (Reversa)
Motor derA (avanzar), Motor derB (Reversa).

estados, a(Avanzar), b(Izquierda), c(Parar), d(Derecha),
e(Reversa).

f(Encendido "Sensar espacio"), g(Apagado).

b) el estado "f", deberá iniciar el sensor ultrasonico y verificar:

1.- Si la distancia es $\leq 15 \text{ & } \geq 2$

 Enciende un Led Rojo
 para los motores por 200 Milis
 Da reversa por 500 Milis
 Gira durante 1100 Milis
 Apaga el Led

Si no hay Obstaculos
 Enciende un Led Verde
 Se desplazara hacia adelante

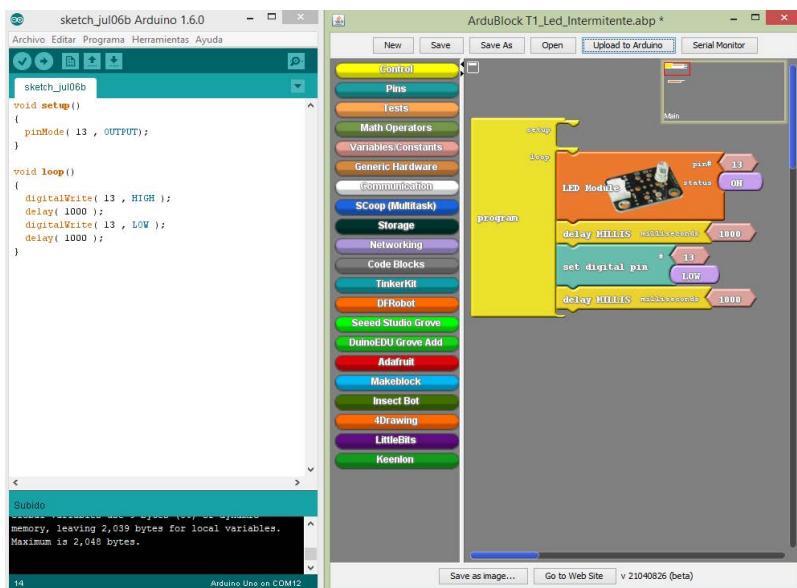
Reto:

Modifica el programa para que:

Introducción a Ardublock

¿Qué es Ardublock?

Es una herramienta de programación visual por bloques para Arduino, funciona como un rompecabezas de bloques funcionales de distintos colores y genera de forma fácil y sencilla el código en el entorno de programación de Arduino (Arduino IDE).



Ardublock es una herramienta que acerca el mundo de la programación de Arduino a estudiantes y aficionados de una forma fácil y sencilla.

Hola, queremos conocer tus opiniones referente a este material, son de gran ayuda con el ánimo de seguir mejorando los contenidos y haciendo éstos más claros.

Estamos atentos a recibir todo tipo de comentarios que nos sirvan de retroalimentación y nos fortalezcan más.

Escríbenos a:

gustavo.reynaga@gmail.com

Muchas gracias por tus aportes :D



Este manual fue adaptado de la obra "Guía Básica de Arduino", elaborado por:

La Tienda de Robótica y el Equipo de Cosas de Mecatrónica.

<http://www.tiendaderobotica.com/>

<http://www.cosasdemelectronica.com/>

