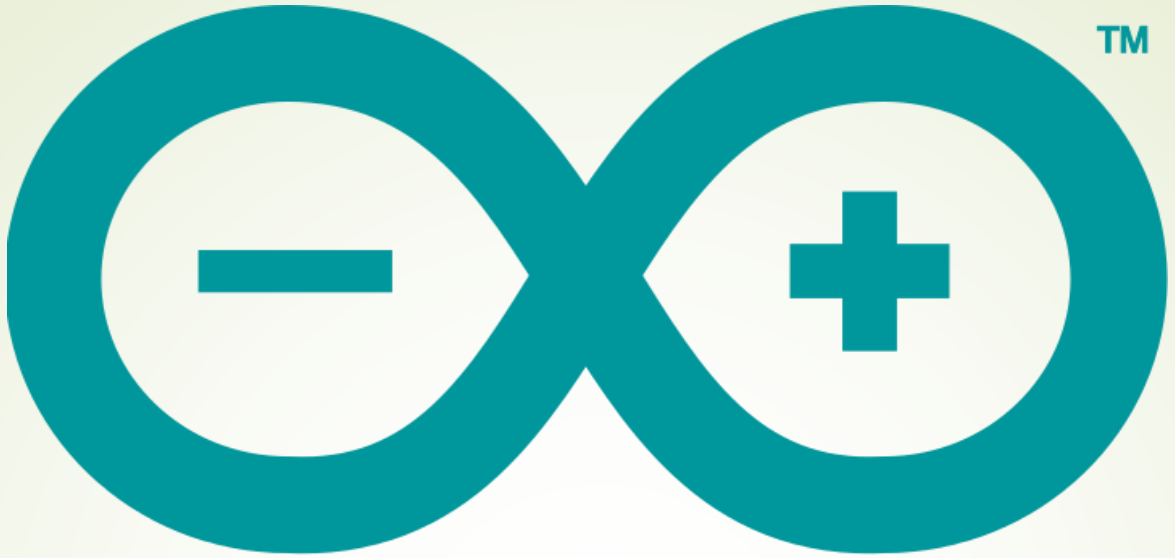
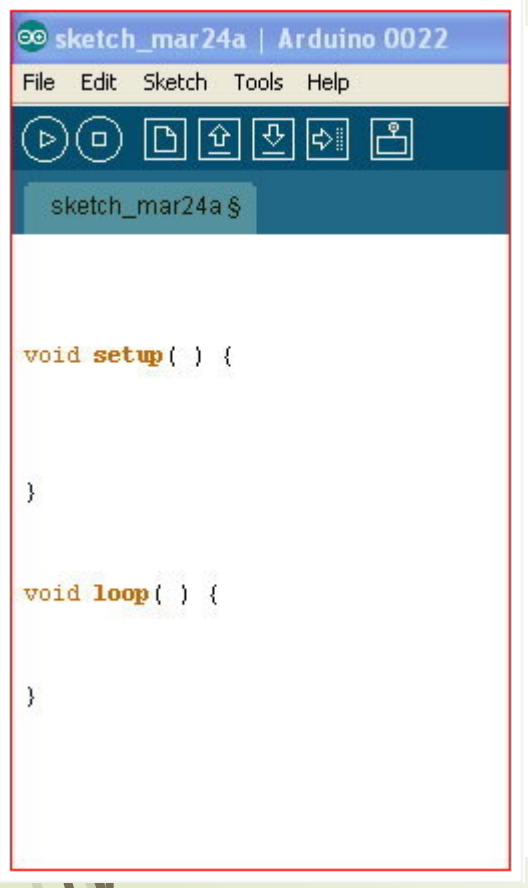


Alvaro Acosta Agón

Magister en Educación

ESTRUCTURA DE PROGRAMACIÓN




```
/* Encender un Led
Esta programación permitirá encender un Led durante un segundo y luego
apagarse
*/

int Led = 13;                    //LED en el pin digital 13

void setup()                     //ejecutar una vez
{
  pinMode (Led, OUTPUT);         //asigna al pin 13 como salida
}

void loop()
{
  digitalWrite(Led, HIGH);       //activa el LED
  delay (1000);                  //espera 1 segundo
  digitalWrite(Led, LOW);        //desactiva el LED
  delay(1000);                   //espera 1 segundo
}
```

Bloque de Comentarios (opcional)

Declarar la(s) variable(s)

Función Setup

Función Loop

Comentarios /*...*/

Los bloques de comentarios o comentarios multilíneas son áreas de texto ignoradas por el programa y se usan para describir códigos o comentarios que ayudan a otras personas a entender parte del programa. Inician con /* y terminan con */ y pueden abarcar varias líneas.

Comentarios de Línea //

Empieza con // y termina con la siguiente línea de código. Son ignorados por el programa y no ocupan espacio en memoria.

Variables

Es una forma de llamar y almacenar un valor numérico para usarse después por el programa. Una variable necesita ser declarada y opcionalmente, asignada al valor que necesita para ser almacenada.

Funciones

Es un bloque de código que tiene un nombre y un grupo de declaraciones que se ejecutan cuando se llama a la función. Se pueden usar funciones integradas como *void setup(), se ejecuta un sola vez;* y *void loop()* se ejecutan de manera infinita.

Punto y coma;

Debe usarse al final de cada declaración y separa los elementos del programa. También se usa para separar los elementos en un bucle ***for***.
No utilizarlo al final de una declaración producirá un error de compilación.

Llaves { }

Las llaves define el comienzo y final del bloque de función y bloques de declaraciones como ***void loop()*** y sentencias ***for*** e ***if***. Las llaves deben estar balanceadas, a una llave de apertura **{** debe seguirle una llave de cierre **}**.

Tabulaciones

Las tabulaciones de las instrucciones contenidas en el *void setup()* o *void loop()*, no son estrictamente necesarias, se utilizan como una manera ordenada, clara y cómoda de escribir el código para el programa.

Variables

Variables Globales: cuando se declaran al inicio del sketch

Variables Locales: cuando se declaran en el interior de alguna sección del *void setup()* o *void loop()*.

TIPO DE DATOS

byte

Byte almacena un valor numérico de 8 bits sin puntos decimales. (0 a 255)

```
byte electrónica = 180;      //declara “electrónica” como un tipo byte
```

int

Enteros son los tipos de datos primario para almacenamiento de números sin puntos decimales y almacena un valor numérico de 16 bits con un rango de – 32.768 a 32.767

```
byte electrónica = 500;      //declara “electrónica” como un tipo entero
```

long

Enteros Largo son los tipos de datos extendido sin puntos decimales, almacenados en un valor de 32 bits con un rango de –3.4028235E –38 a 3.4028235E +38

```
byte electrónica = 9000;      //declara “electrónica” como un tipo entero largo
```

float

Punto flotante u números que tienen un punto decimal. Almacenan un valor de 32 bits con un rango de – 2.146.483.618 a 2.147.483.617

```
byte electrónica = 3.14;      //declara “electrónica” como un tipo flotante
```

arrays

Existen array de variables de tipo “boolean”, de tipo “int”, de tipo “float”, etc. Un array es una colección de valores que son accedidos con un índice numérico. Cualquier valor en el array debe llamarse escribiendo el nombre del array y el índice numérico del valor. Los arrays indexados a cero, con el primer valor en el array comenzando con el índice número 0. Un array necesita ser declarado y opcionalmente asignarle valores antes de que puedan ser usados.

```
int myArray[ ] = {value0, value1, value2, …;

int myArray[6];                 //declara un array de enteros con 6 posiciones
myArray[4] = 10;                //asigna a la cuarta posición del índice el valor 10
int myArray[8] = {2,5,6,7};     //declara un array de 6 elementos e inicializa algunos de ellos
```

Asignaciones Compuestas

```
x++;          //lo mismo que x = x+1
x– –;         //lo mismo que x = x–1
x += y;       //lo mismo que x = x+y
x –= y;       //lo mismo que x = x–y
x *= y;       //lo mismo que x = x*y
x /= y;       //lo mismo que x = x/y
```

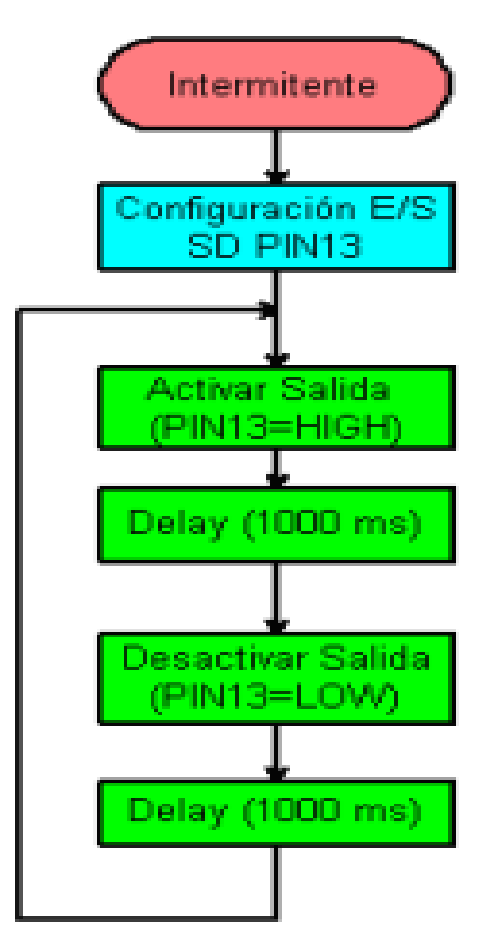
Operadores de Comparación

```
x == y;       //x es igual a y
x != y;       //x no es igual a y
x < y;        //x es menor que y
x > y;        //x es mayor que y
x <= y;       //x es menor o igual que y
x >= y;       //x es mayor o igual que y
```

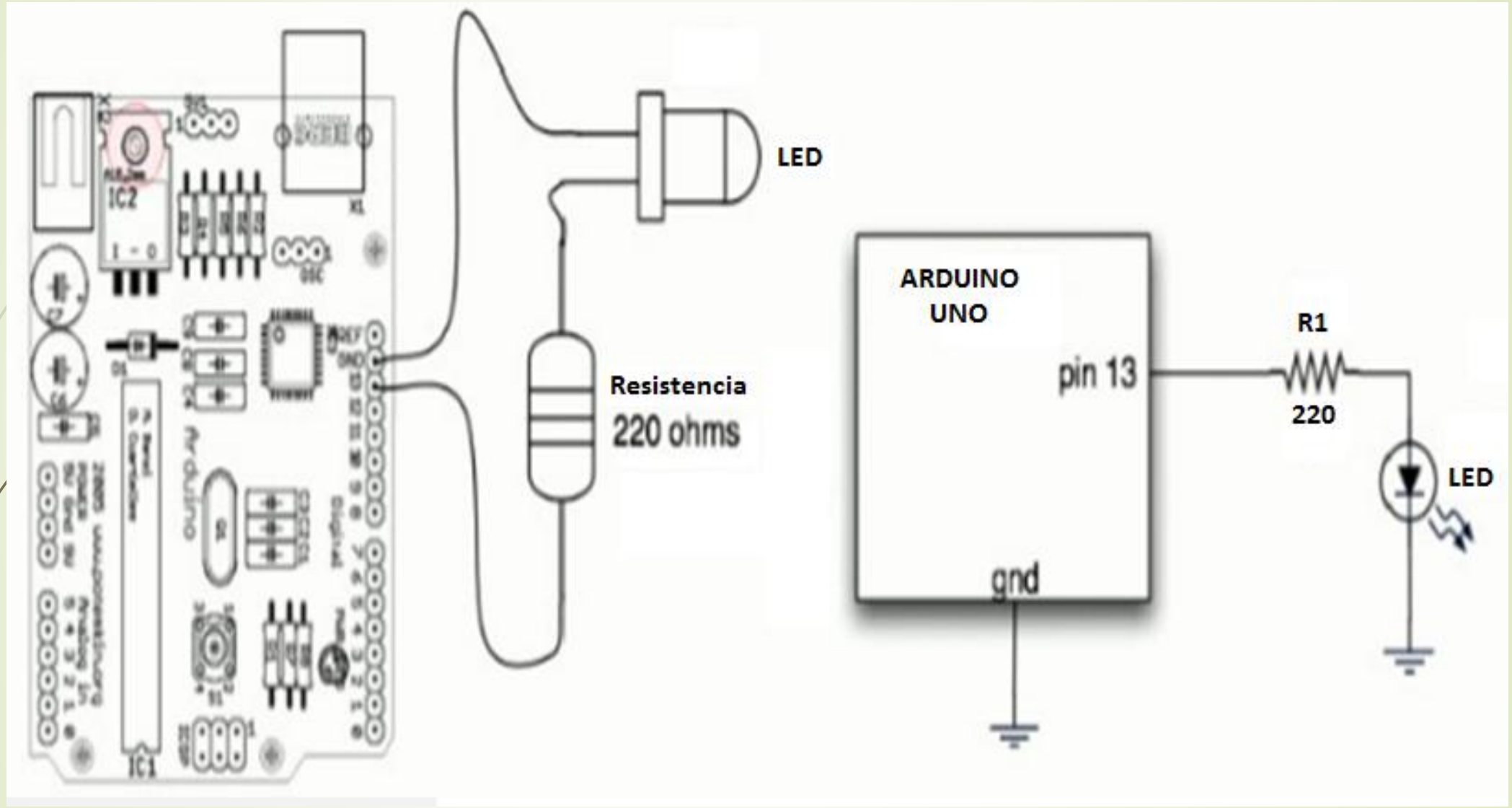
Operadores Lógicos

```
//AND Lógico
if(x>0 && x<5)        //verdadero solo si las dos expresiones son ciertas

//OR Lógico
if(x>0 || y>0)        //verdadero si al menos una expresión es cierta

//NOT Lógico
if(!(x>0))            //verdadero sólo si la expresión es falsa
```

ENCENDER Y APAGAR UN LED DE MANERA INFINITA




```
/* Encender un Led
Esta programación permitirá encender un Led durante un segundo y luego
apagarse
*/


int Led = 13;                          //LED en el pin digital 13

void setup()                           //ejecutar una vez
{
  pinMode (Led, OUTPUT);               //asigna al pin 13 como salida
}

void loop()
{
  digitalWrite(Led, HIGH);             //activa el LED
  delay (1000);                        //espera 1 segundo
  digitalWrite(Led, LOW);              //desactiva el LED
  delay(1000);                         //espera 1 segundo
}
```


LED
Resistencia
220 ohms
ARDUINO
UNO
pin 13
gnd
R1
220
LED

Enciende y Apaga indefinidamente (declarando la variable Led)

```
int Led = 13;                          Declara que el pin 13 corresponde a Led

void setup() {                         //inicia la configuración
pinMode (Led, OUTPUT);                 //configura a Led como pin de salida
}

void loop() {                          // bucle principal del programa
digitalWrite(Led, HIGH);               //envía 5V (alto) al pin Led
delay (500);                           //espera 500 ms
digitalWrite(Led, LOW);                //envía 0V (bajo) al pin Led
delay (400);                           //espera 400 ms
}                                      //termina la configuración
```

Enciende y Apaga una sola vez

```
void setup() {                         //inicia la configuración
pinMode (13, OUTPUT);                  //configura el pin 13 como de salida
digitalWrite (13, HIGH);               //envía 5V (alto) al pin 13
delay (1000);                          //espera 1000 ms
digitalWrite(13, LOW);                 //envía 0V (bajo) al pin 13
delay (1000);                          //espera 1000 ms
}

void loop() {                          //bucle principal del programa
}                                      //termina la configuración
```

Enciende y Apaga indefinidamente

```
void setup() {                         //inicia la configuración
pinMode (13, OUTPUT);                  //configura el pin 13 como de salida
}

void loop() {                          // bucle principal del programa
digitalWrite(13, HIGH);                //envía 5V (alto) al pin 13
delay (1000);                          //espera 1000 ms
digitalWrite(13, LOW);                 //envía 0V (bajo) al pin 13
delay (1000);                          //espera 1000 ms
}                                      //termina la configuración
```

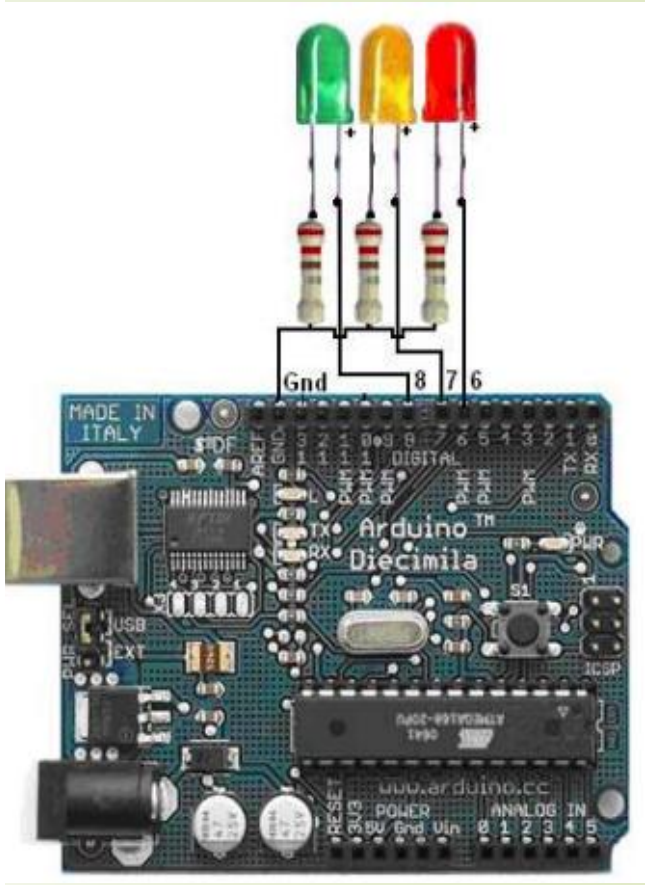
Secuencia de Led, declarando el tiempo de transición	
int t=200;	//declara una variable como entero y da valor 200 ms

void setup() {
pinMode (5,OUTPUT);
}
	//inicia la configuración
	//configura el pin 5 como de salida

void loop() {
digitalWrite (5,HIGH);
delay(t);
digitalWrite (5,LOW);
delay(t);
}
	// bucle principal del programa
	//envía 5V al pin 5 (salida)
	//espera 200 ms
	//envía 0V al pin 5
	//espera 200 ms
	//termina configuración

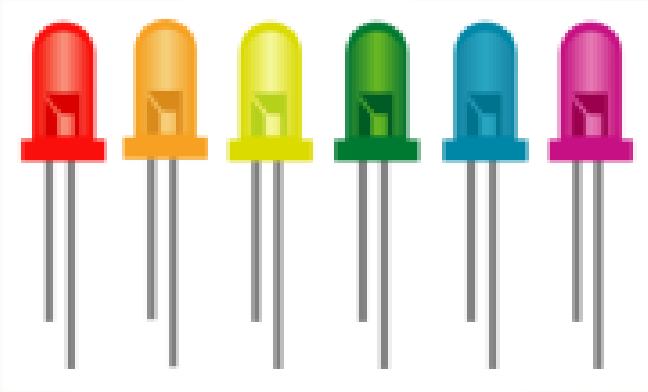
Enciende y Apaga tres Led's

```
int Led1 = 6;                          //Define las salidas de los Led's
int Led2 = 7;
int Led3 = 8;

void setup() {                         //inicia la configuración
pinMode (Led1, OUTPUT);                //configura los Led's como salida
pinMode (Led2, OUTPUT);
pinMode (Led3, OUTPUT);
digitalWrite(Led1, LOW);               // Apaga los LEDs
digitalWrite(Led2, LOW);
digitalWrite(Led3, LOW);
}

void loop() {                          // bucle principal del programa

digitalWrite (Led1, HIGH);             // Apaga y enciende los leds cada 200 ms
delay (200);

digitalWrite(Led1, LOW);
digitalWrite(Led2, HIGH);
delay (200);

digitalWrite(Led2, LOW);
digitalWrite(Led3, HIGH);
delay (200);

digitalWrite(Led3, LOW);               //termina la configuración
}
```

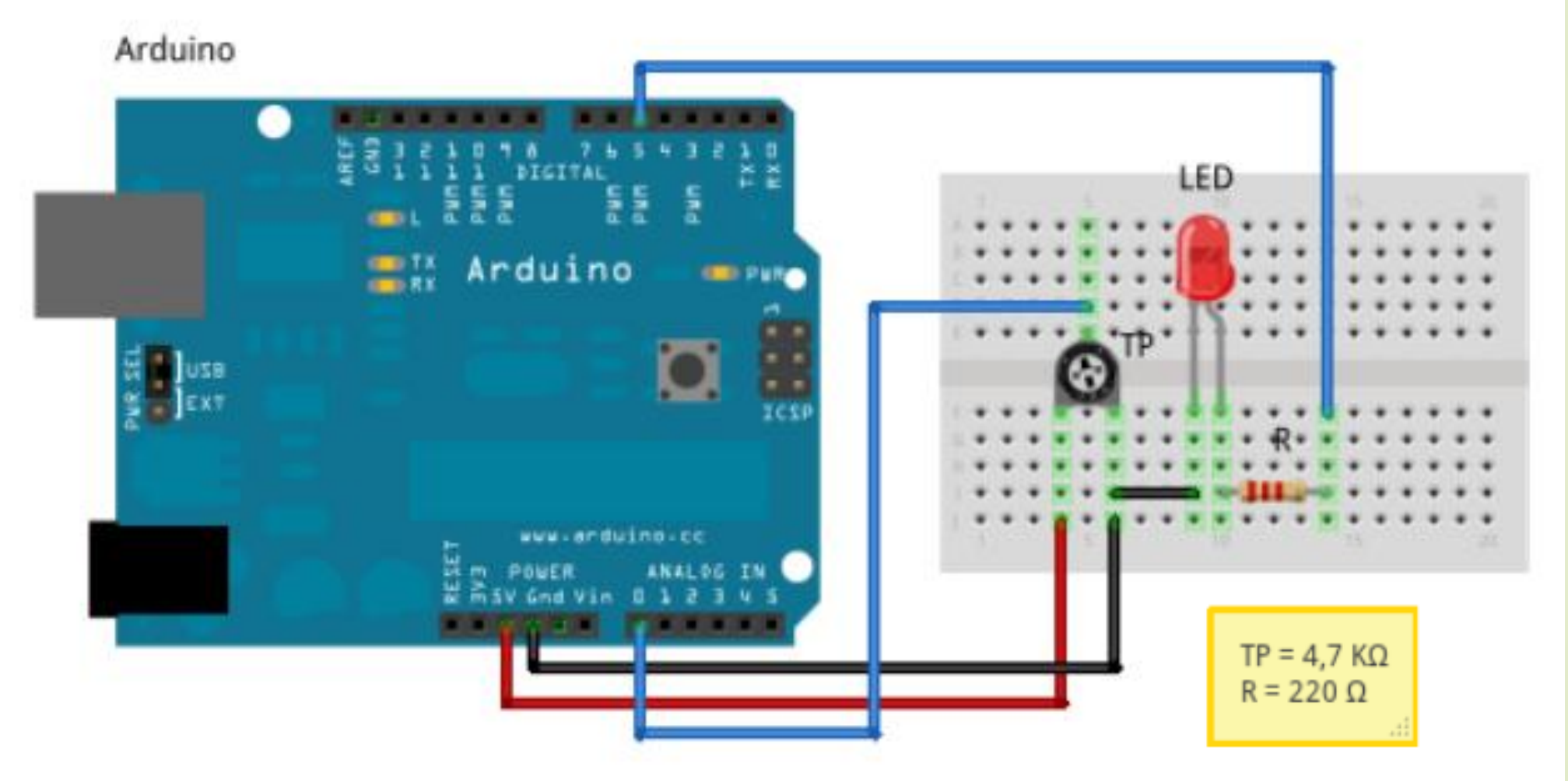


RETO 1: SECUENCIAS DE LED'S

Realice una secuencia de encendido y apagado de seis Led's, de manera cíclica, configuración el barrido de encendido y tiempo de transición.

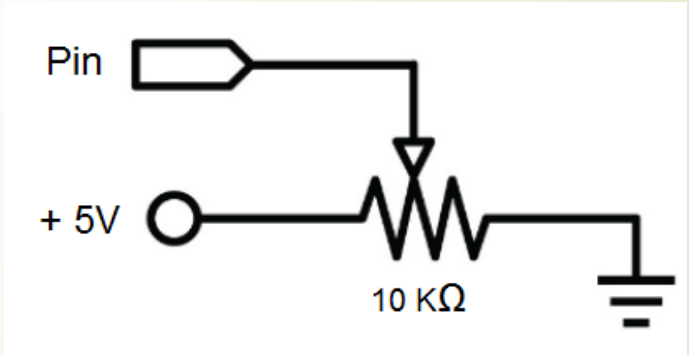


CONTROLAR LA LUZ DEL LED CON UN POTENCIÓMETRO



Usando el potenciómetro y uno de los pines de conversor A/D de Arduino es posible leer valores de 0 a 1024.

```
int potPin = 0;                 //pin de entrada para el potenciómetro
Int ledPin = 13;                //Led de salida

void setup()
{
 pinMode (ledPin, OUTPUT);      //ledPin como salida
}
void loop()
{
 digitalWrite(ledPin, HIGH);    //activa ledPin
 delay(analogRead(potPin);      //pausa el programa
 digitalWrite(ledPin, LOW);     //activa ledPin
 delay(analogRead(potPin);      //pausa el programa
```

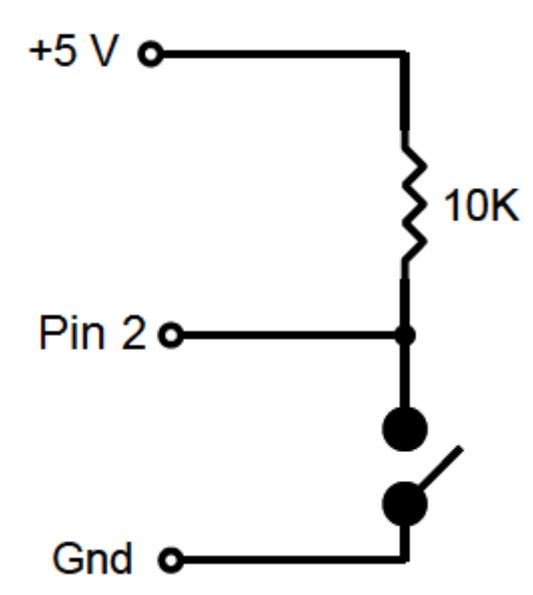


MONITOR SERIAL

```
*/Incremento de un número a través de una variable */
int mivariable=555;
void setup()
{
Serial.begin(9600);
}
void loop()
{
Serial.println(mivariable);
mivariable=mivariable+1;
}
```

```
*/lectura del potenciómetro*/
int analogPin=0;
int val;

void setup()
{
Serial.begin(9600);
Serial.println(“lectura del potenciómetro”);
}
void loop()
{
val=analogRead(analogPin);
Serial.println(val);
delay(400);
}
```

Para este ejercicio, el Arduino leerá el estado de un interruptor pulsador normalmente abierto y muestra los resultados en el PC con la orden Serial.println ().



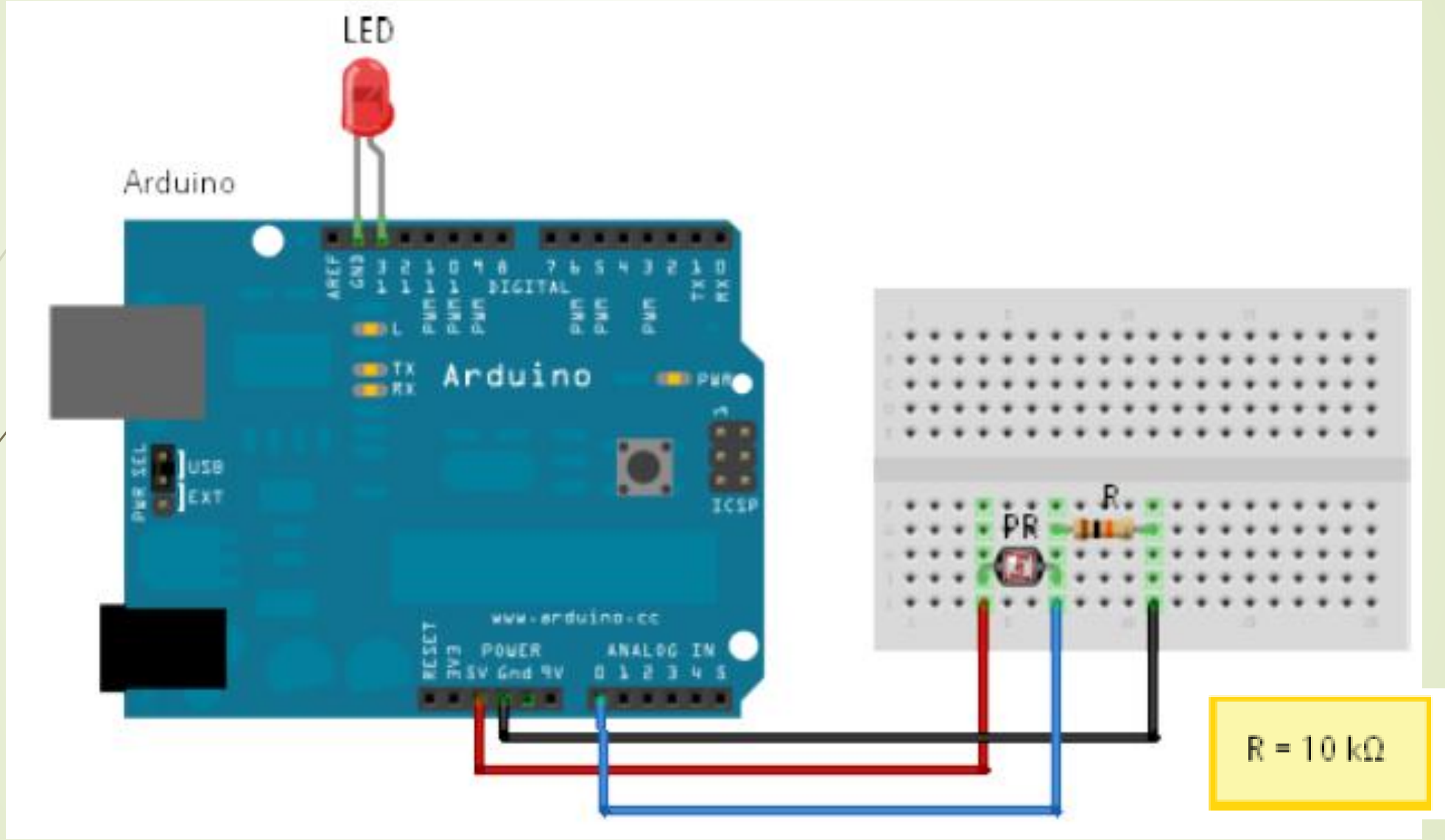

```
void setup()
{
Serial.begin(9600);
}
void loop()
{
Serial.println(digitalRead(2));
delay(250);
}
```

Abra la ventana del monitor de serie.
Cuando el interruptor está abierto, debería ver un tren de 1 de la pantalla.
Cuando está cerrado, el cambio será de 1 a 0.

RETO 2: ENCENDIDO DE LED EN FUNCIÓN DE LA LUZ

Realice el encendido de un led de acuerdo al funcionamiento de una fotocelda. Cuando la luz se encuentre entre 0 y 512 el led debe colocarse en el nivel de potencia máxima (255), si la luz se encuentra entre valores 512 y 1024 el debe lucir al nivel de potencia baja (64).

ENCENDER UN LED UTILIZANDO UNA FOTOCELDA



Estructuras de control

Son instrucciones que permiten tomar decisiones y hacer diversas repeticiones de acuerdo a unos parámetros, dentro de las más importantes se destacan:

If
Switch/case
For
While

Condicionales

If (Si)

```
 if (entrada < 500)
{
        // acción A
} else
{
        // acción B
 }
```

Switch/case (Casos)

```
switch (var) {
  case 1:
    // acción A
    break;
  case 2:
    // acción B
    break;
  default:
    // acción C
}
```

Ciclos

For (por)

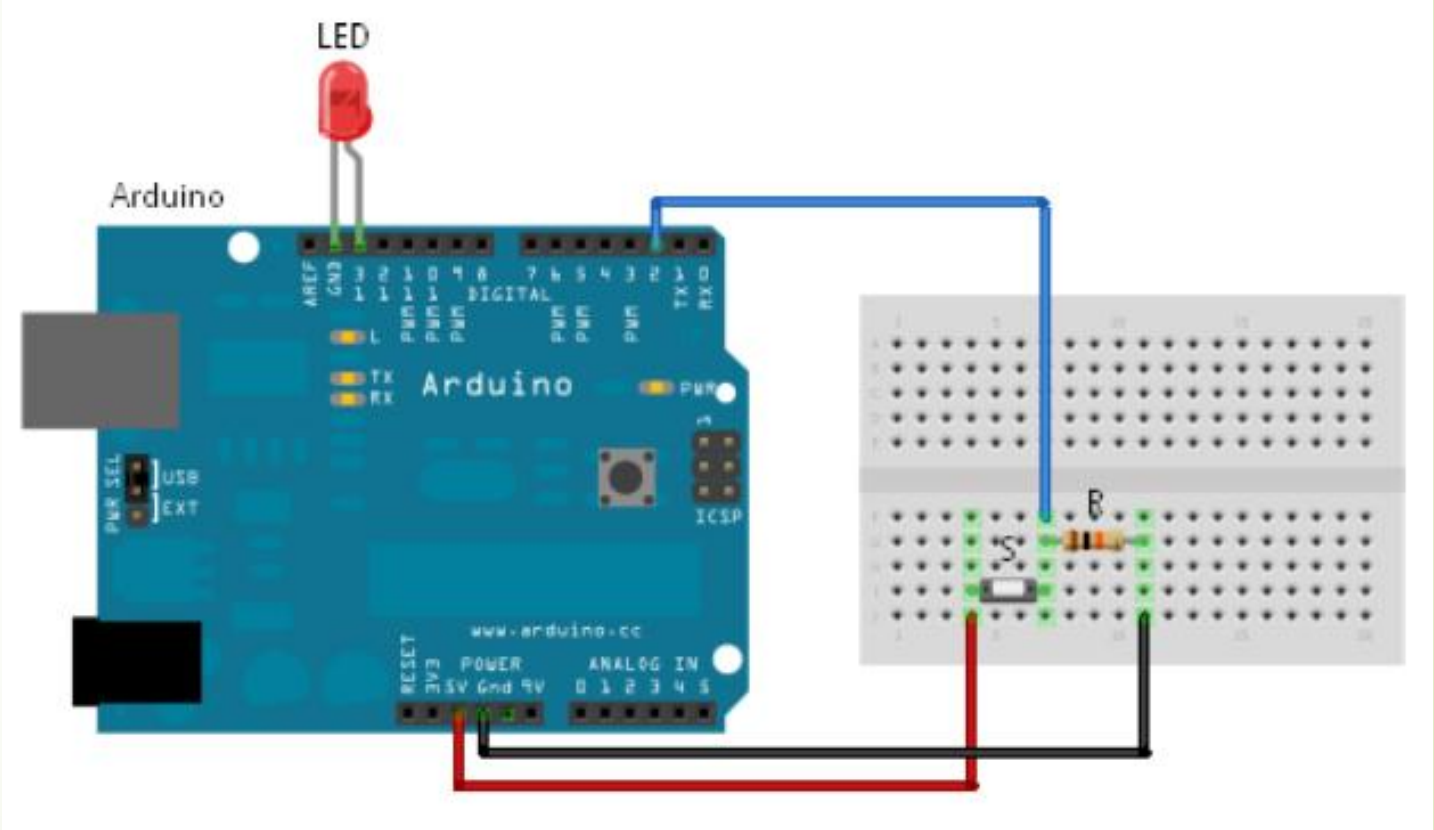
```
for( int a=0; a>10; a++ )
{
        // acción a repetir
}
```

While (mientras)

```
while ( var < 200) {
   // acción a repetir
   var++;
}
```

ENCENDER UN LED UTILIZANDO UN PULSADOR

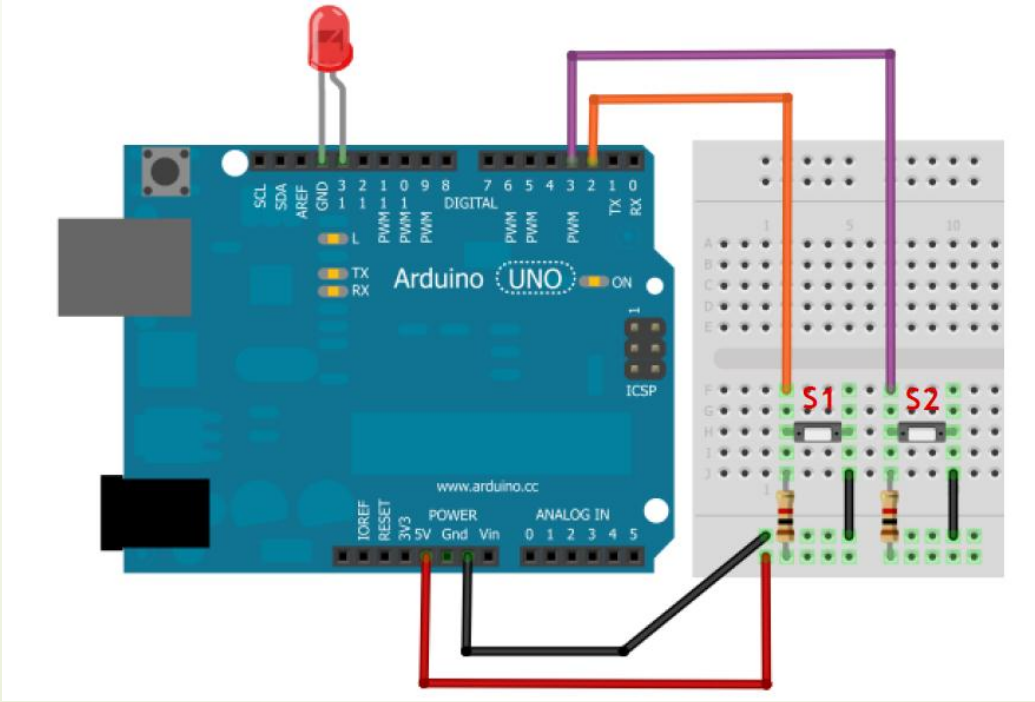
```
*/Encender un led con un pulsador*/
int pulsador=2;
int led=13;

void setup()
{
pinMode (pulsador, INPUT);
pinMode (led, OUTPUT);
}

void loop()
{
if(digitalRead(pulsador)==HIGH);
digitalWrite(led,HIGH);
else
digitalWrite(led,LOW);
}
```



RETO 3: ENCENDIDO DE UN LED AL PULSAR DOS BOTONES

Control de una máquina: cuando se pulsen simultáneamente dos botones, (uno con cada mano), la máquina se acciona.

Dos pulsadores (S1 y S2) representan los botones, y un led simula la operación de la máquina.



LECTURA SERIAL DE UNA ENTRADA

Leer una entrada digital y mostrar por la pantalla del computador (monitor serial) el estado del pulsador cuando es oprimido.

```
*/Visualizar en el monitor si está pulsado o no*/
int pulsador=2;

void setup()
{
pinMode (pulsador, INPUT);
Serial.begin(9600);
}

void loop()
{
int estado=digitalRead(pulsador);
{
  if (estado==1);
  Serial.println(“pulsado”);
}
else
{
 Serial.println(“no pulsado”);
}
delay(100);
}
```

AUMENTAR Y DISMINUIR LA INTENSIDAD DEL LED

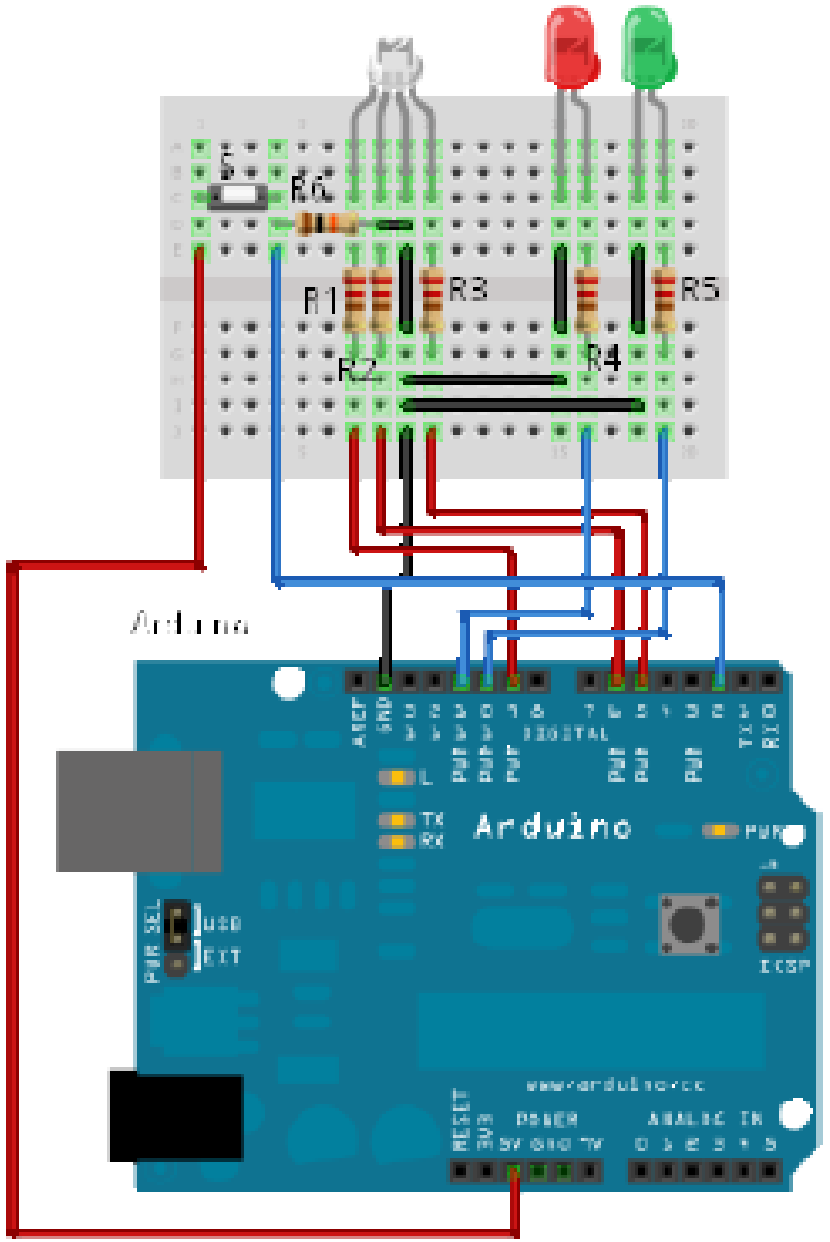
```
int led = 9;

void setup() {
  pinMode(led, OUTPUT);
}

void loop() {
  for(int a=0; a<225; a++)
  {analogWrite(led,a);
    delay(20);

  }

  for(int a=255; a>1; a--)
  {analogWrite(led,a);
    delay(20);
  }
}
```

RETO 4: ENCENDIDO DE LEDs DE ACUERDO AL CONTROL DEL POTENCIÓMETRO

Realizar un programa que al leer una entrada análoga de un potenciómetro (0-1023), se pueda ajustar al valor de PMW (0-255) para que encienda un led diferente si cumple lo siguiente:

Si el menor o igual de 255 enciende led1
Si el menor o igual de 512 enciende led2
Si el mayor a 512 enciende led 3

UTILIZANDO LEDs RGB



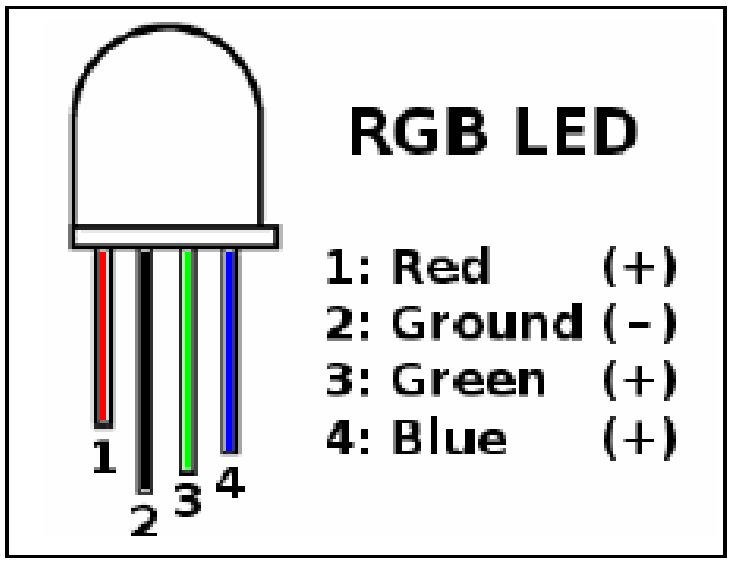
R1 = R2 = R3 = R4 = R5 = 220 Ω
R6 = 10 kΩ

LEDs RGB

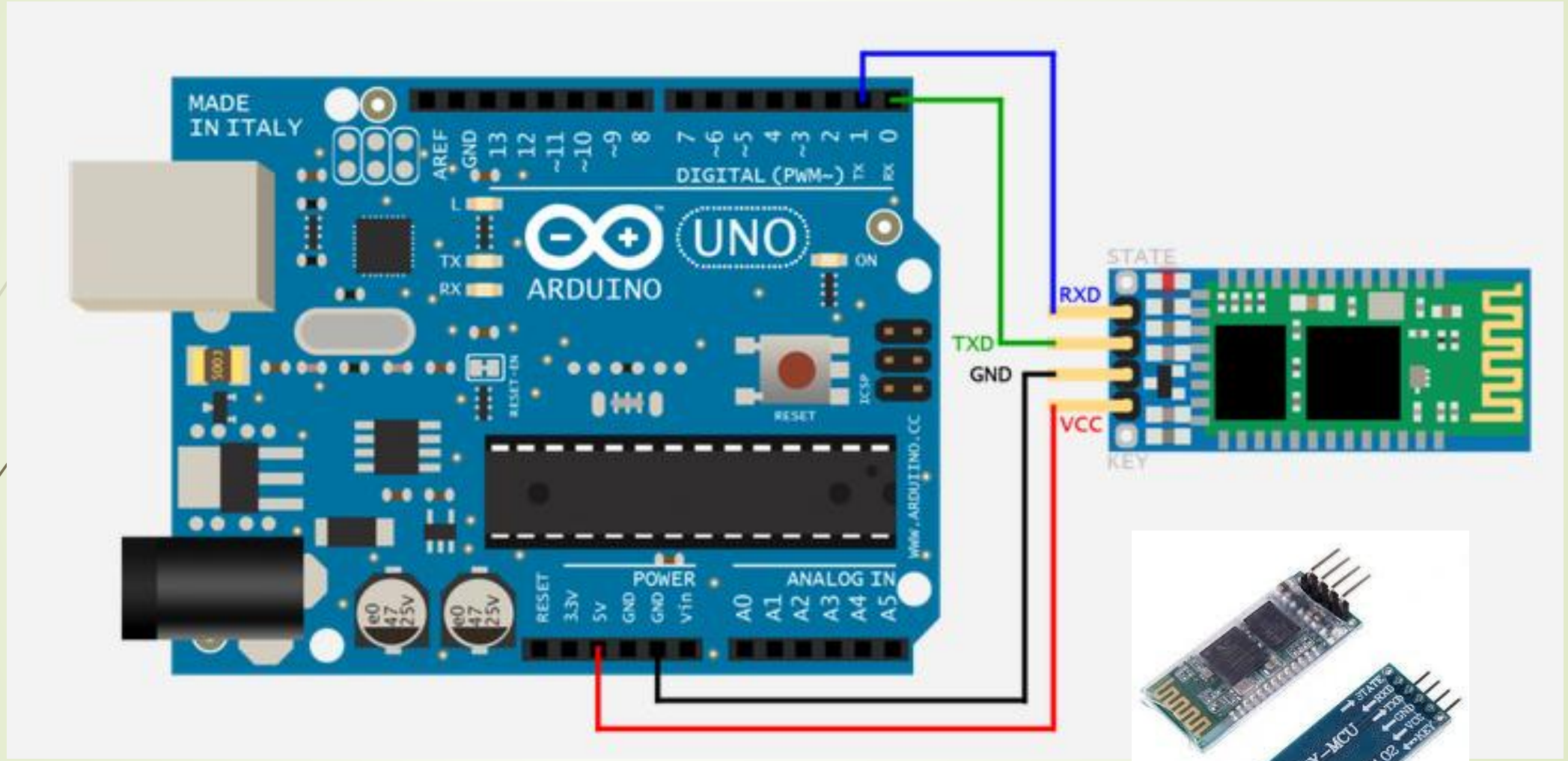
Los **LEDs RGB** proporcionan un espectro completo de colores (en contraste con las normales, que dan un solo color), resultando en una luz compuesto de tres colores primarios. Dependiendo de la intensidad de cada color, se podrá obtener una gran cantidad de diferentes tonos.

Es muy importantes a saber los controles de color cada pin del LED:

El terminal más largo del led RGB debe ser conectado a tierra



Conexión Bluetooth



http://www.youtube.com/watch?v=fokloKBNCOE#t=17

pinMode ()
Comando que declara un pin como entrada o salida.

digitalWrite
Este comando establece un pin I/O alto (+ 5V) o bajo (0 V)

delay
Detiene el programa durante un número especificado de milisegundos.

Este comando , que va en la función de configuración () , se utiliza para establecer la dirección de un pin I / O digital. Ajuste el perno de salida si el pasador está impulsando y LED, motor u otro dispositivo. Ajuste el perno de ENTRADA si el pin está leyendo un interruptor u otro sensor . Al encender o reiniciar , todo pasadores defecto insumos.

Instructables (http://www.instructables.com)
Makezine (http://blog.makezine.com/arduino)
MakeProjects (http://makeprojects.com/Topic/Arduino)
HackADay (http://hackaday.com/category/arduino-hacks)
HackNMod (http://hacknmod.com)
Dangerous Prototypes (http://dangerousprototypes.com)
Electronics Lab (http://www.electronics-lab.com/blog)
BricoGeek (http://www.bricogeek.com)
Embedds (http://www.embedds.com)
Sección de proyectos Jameco (http://www.jameco.com/Jameco/PressRoom/diy.html)
Sección de proyectos Arduino (http://arduino.cc/playground/Projects/ArduinoUsers)
Proyectos del HLT (http://hlt.media.mit.edu/?cat=20 , http://hlt.media.mit.edu/?p=1283 y ?p=1314)