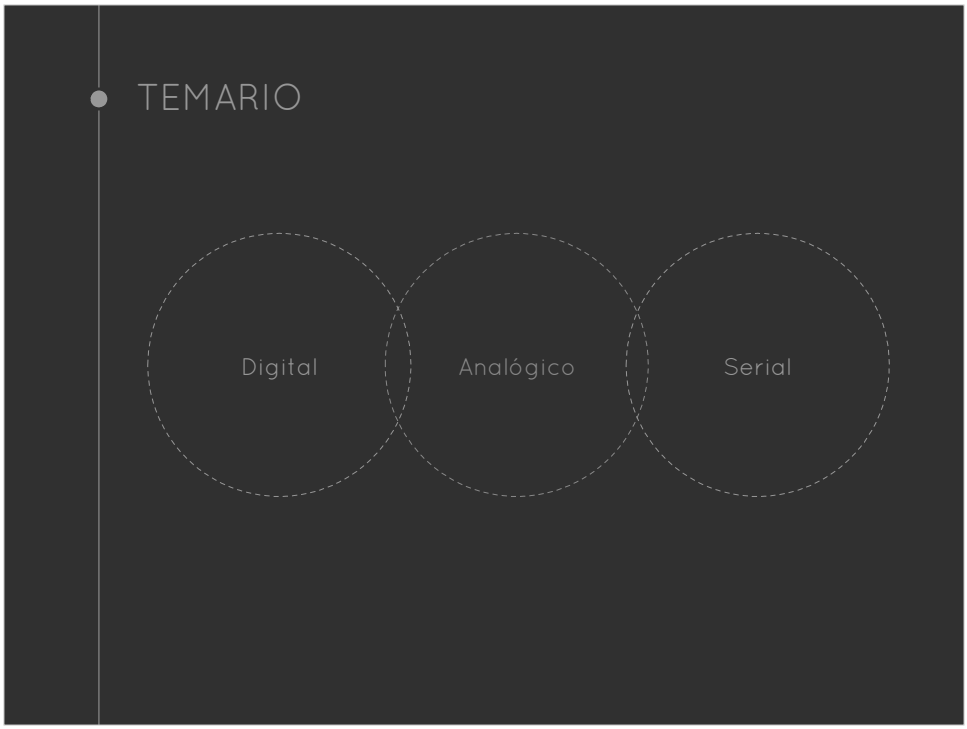
Hola a todos.
Vamos a comenzar con este taller sobre arduino.

Quienes somos.

Yo soy fulano de tal y ella mengana de tal.

Somos alumnos del ante último año de la Tecnicatura en informática profesional y personal.



Este taller se va a centrar en estos tres temas:

* Entrada y salida digital

* Entrada y salida analógica

* Comunicación serial

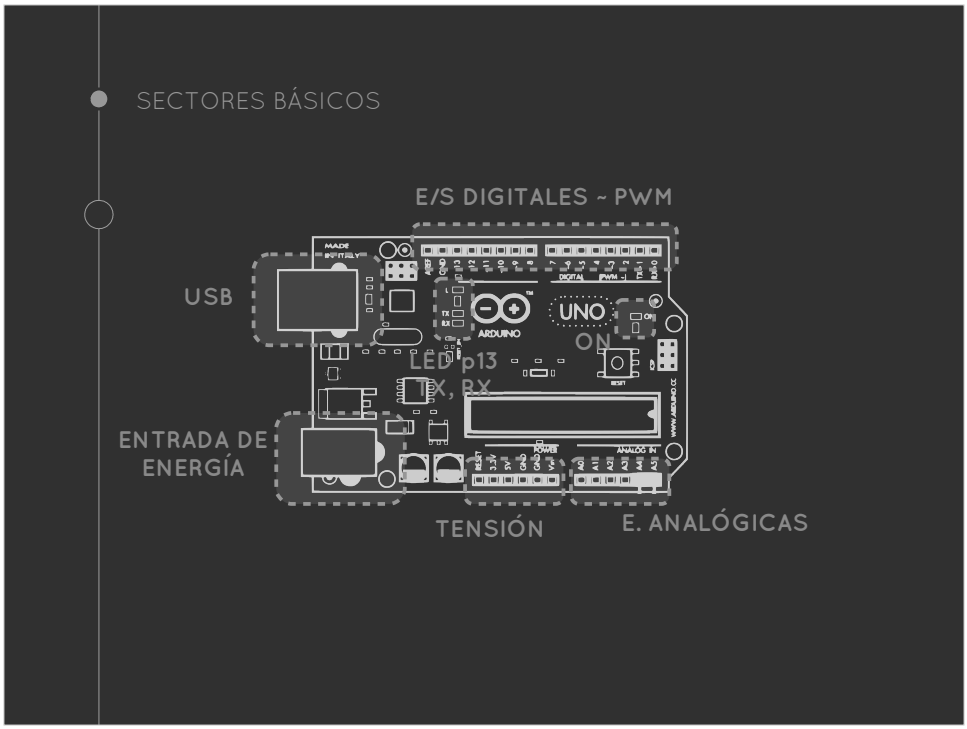
Tiene un formato de taller porque vamos a realizar varias prácticas.

Para poder empezar a trabajar con Arduino, antes, debemos saber con qué vamos a encontrarnos.

De acuerdo a su página oficial ARDUINO es esto:
* Una placa con microcontrolador
* Y un entorno de desarrollo

Sigue la filosofía del hardware libre, que promueve los valores de compartir, cooperar y colaborar.

Un arduino consta de las siguientes partes o sectores básicos.



* Un conector USB. Que permite la carga de programas y la comunicación serial entre computadora y arduino, además de proveer la alimentación de poder para el funcionamientodel mismo.
* También tiene un conector para una fuente externa de energía; esto, para el caso de querer que funcione el dispositivo de forma autónoma (STAND ALONE).
* Los pines de tensión (5V y 3,3V) y GND (tierra/GORUND).
* Pines de entrada analógica.
* Pines entrada/salida digitales y PWM o salidas analógicas.
* LEDs de transmisión de datos (TX), de recepción de datos (RX) y el LED conectado al pin 13.
* LED de encendido (ON).

Como digimos ARDUINO no es sólo un microcontrolador, sino también un entorno de desarrollo.

El IDE se construyó en base a PROCESSING y utilizando como lenguaje un derivado de WIRING.

La interfaz del IDE se ve de la siguiente forma.
La barra de menú, la barra de herramientas, la barra de pestañas, el área de código y la salida del compilador que es donde se van a mostrar los mensajes emitidos por el compilador. (los de errores también)

En la barra de herramientas nos vamos a encontrar con los siguientes botones y el del monitor serial que ese, lo va a explicar el profesor al tratar el tema de comunicación serial.

* Verificar: va a compilar el programa.
* Cargar: lo va a caragr en el arduino.
* Nuevo: va a abrir un documento nuevo.
* Abrir y guardar un archivo.

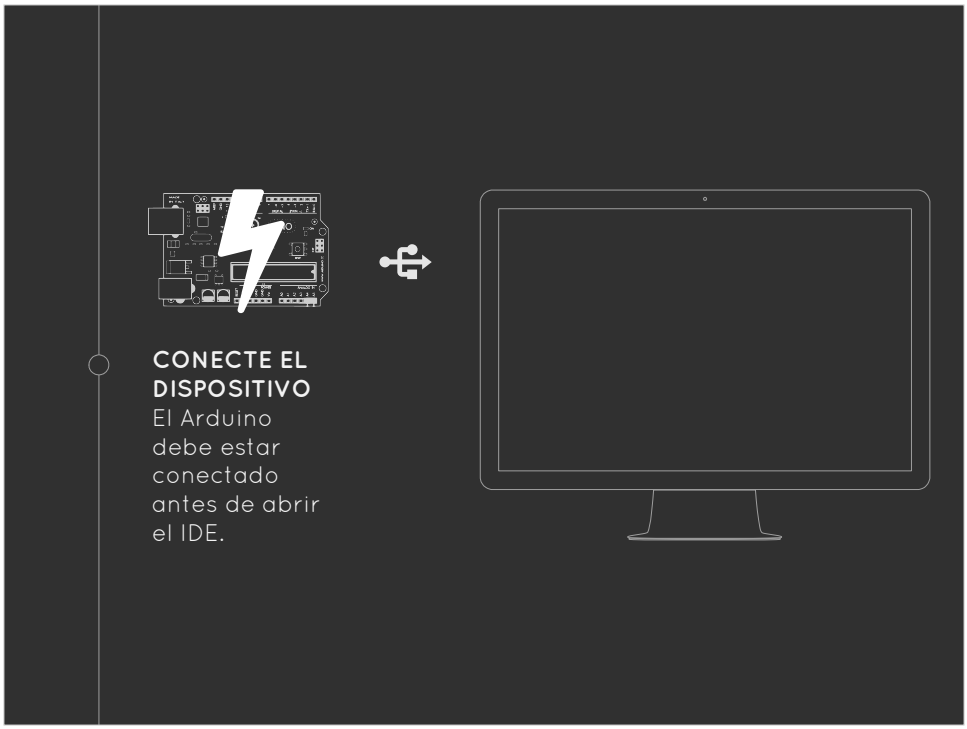
Cuando decimos digital, nos referimos a un conjunto ***discreto*** que sólo tiene dos elementos: 1 y 0, Verdadero y Falso, Alto y bajo, Encendido y Apagado, etc.

Vamos a comenzar a trabajar. Deben sacar un papelito de la caja/bolsa y lo guardan, no lo rompan porque se lo vamos a pedir.

(Luego de que todos sacan un papelito incluso los profesores del IFD)

Todos los que tienen el mismo personaje se agrupan en una mesa.
(las mesas y los personajes los vamos a designar nosotros).

Continuamos...

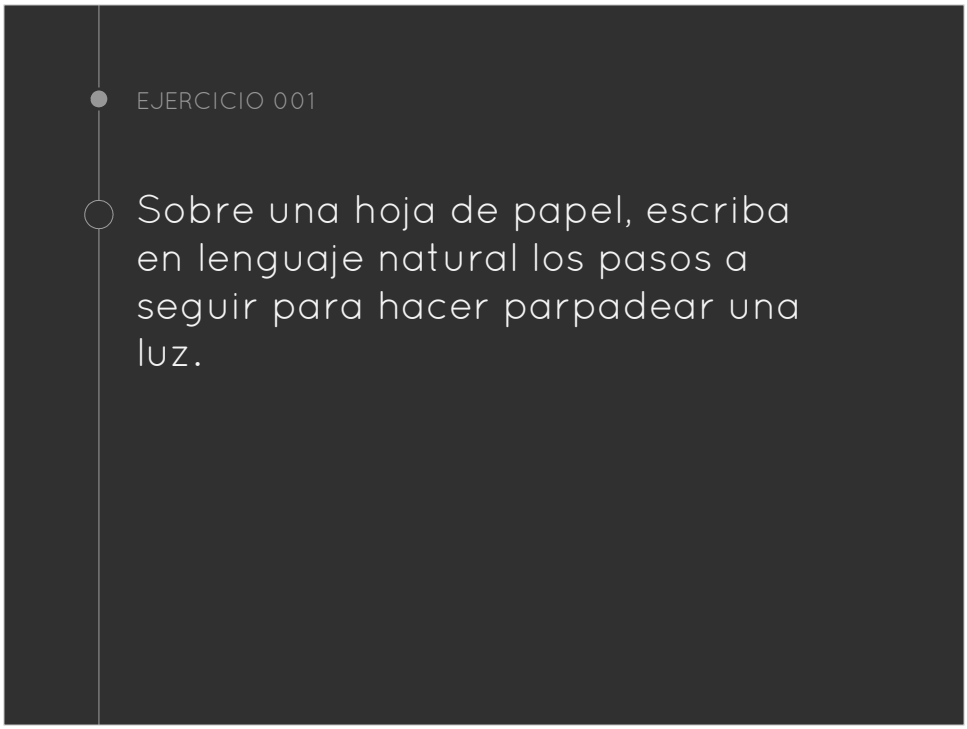


Así que para comenzar con la práctica vamos a conectar el arduno a la computadora, pero hay que tener cuidado con la electricidad estática.

Por lo tanto, antes de tocar el arduino hay que conectarse a la pulsera antiestática o en su defecto descargarse tocando durante un par de segundos la pared o el piso con la mano desnuda.

El arduino debe conectarse antes de correr el IDE o el programa que lo utilice para que este último pueda detectarlo.

Ahora vamos a hacer el clásico hola mundo, pero n arduino, al no tener una salida de texto, e lugar de mostrar el mensaje “hola mundo” vamos a hacer parpadear un LED.

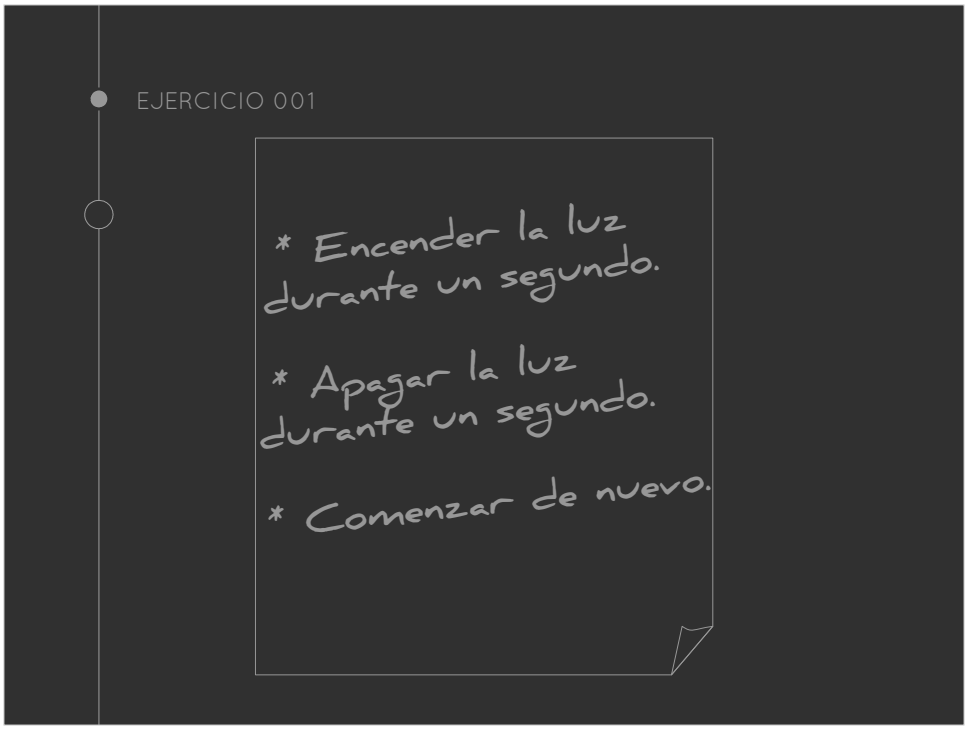


Este es el primer ejercicio.
Entonces, en español plano, vamos a hacer lo que nos pide en una hojita de papel.

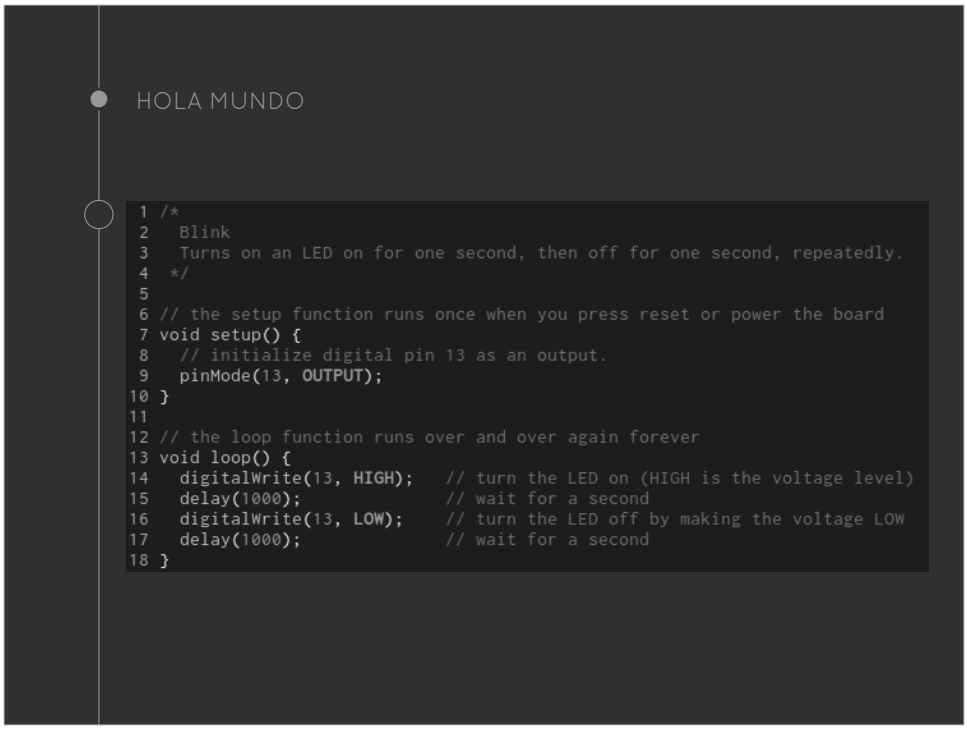
Van a tener 2 minutitos para hacerlo porque es bien fácil el ejercicio.

(Después delos 2 minutos) preguntamos a los asistentes como les quedo el ejercicio y hacemos que dos o tres de ellos nos den sus respuestas).

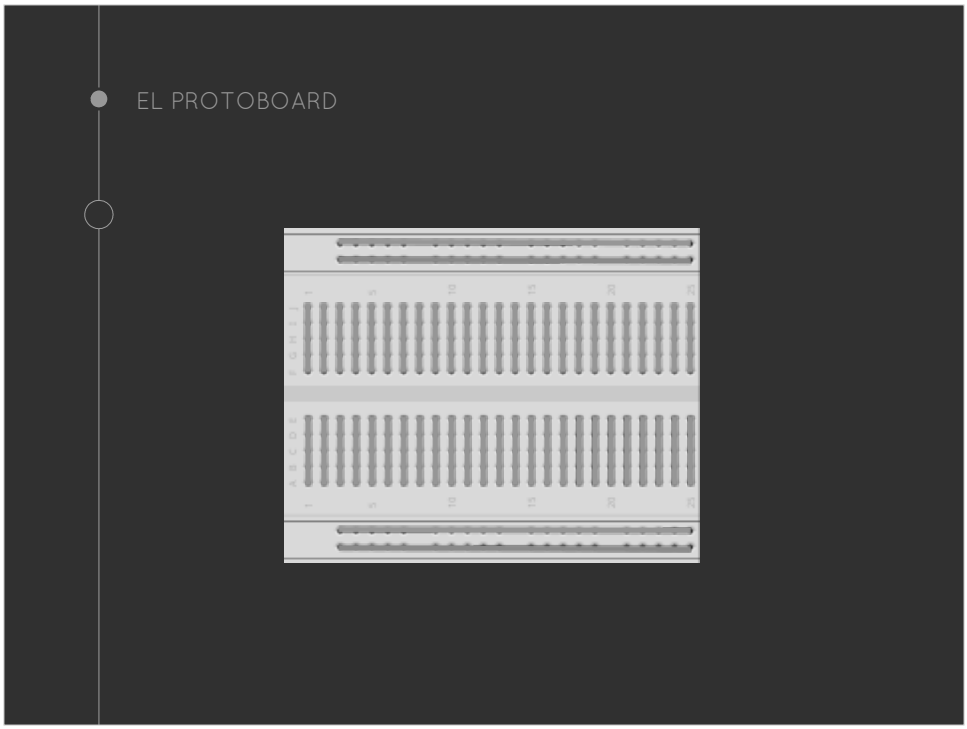
Ok. debería de quedarnos parecido a esto.



Encendemos la luz durante un tiempo, lo apagamos durante un tiempo y repetimos lo mismo.

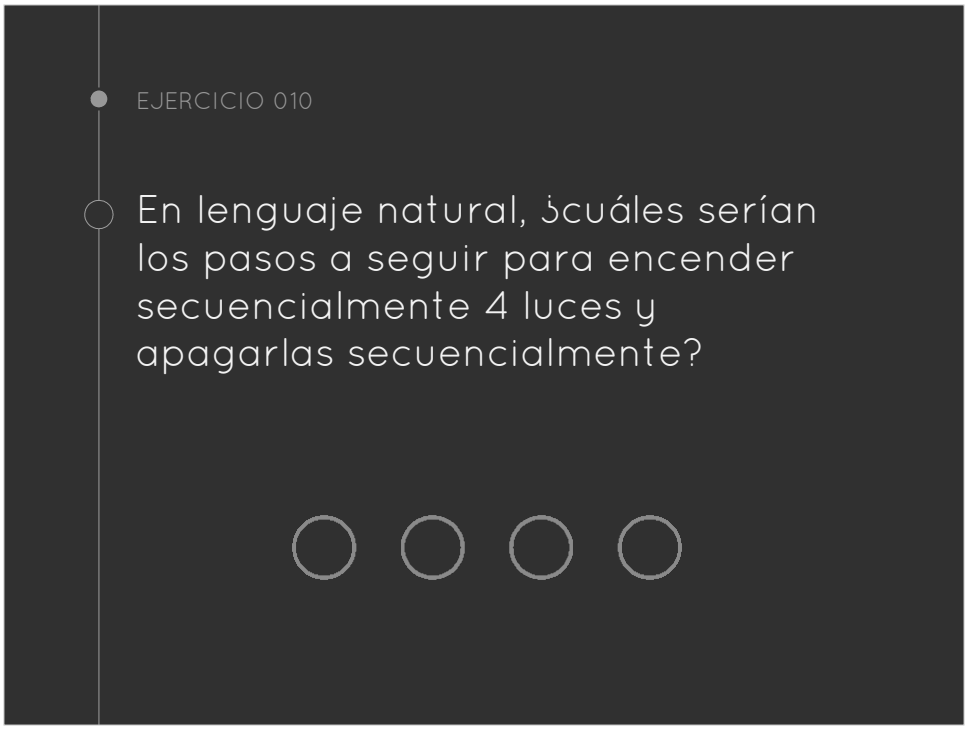


Esto traducido al lenguaje de programación del arduino quedaría así:
Lo que está entre /* y */ es un comentario de varias líneas.
Lo que está a continuación de la doble barra es un comentario de una sola línea.
Luego vamos a encontrarnos con dos funciones: setup y loop
Setup() que quiere decir configuración, sólo se ejecuta una vez al encenderse el arduino. Dentro de esta podemos configurar los pines del arduino como entrada o salida. En este caso configuramos el pin 13 como salida.
Loop() que significa bucle o rulo es una función que se va a ejecutar inmediatamente depués de setup() y se va repetir indefinidamente. Dentro de esta vamos a colocar nuestro programa, todo lo que queremos que nuestro arduino haga.
Primero realizamos una escritura digital en el pin 13. Al colocarlo en HIGH (ALTO) el pin va a quedar con 5V.
Le decimos al programa que espere 1 segundo (1000 milisegundos) y depues continúe su ejecución.
Ahora indicamos que el pin 13 debe pasarse a un nivel LOW (BAJO) y esperar el mismo tiempo.
No vamos a escribir este código porque ya viene como ejemplo en el IDE. Por lo tanto debemos hacer clic en el menú archivo, ejemplos, basics, blink.
Recuerden que el arduina debe estar conectado antes de iniciar el IDE.
Una vez abierto, COMPILAMOS el código y lo cargamos al arduino.



Para realizar el segundo ejercicio debemos conocer al protoboard (también conocido como breadboard).

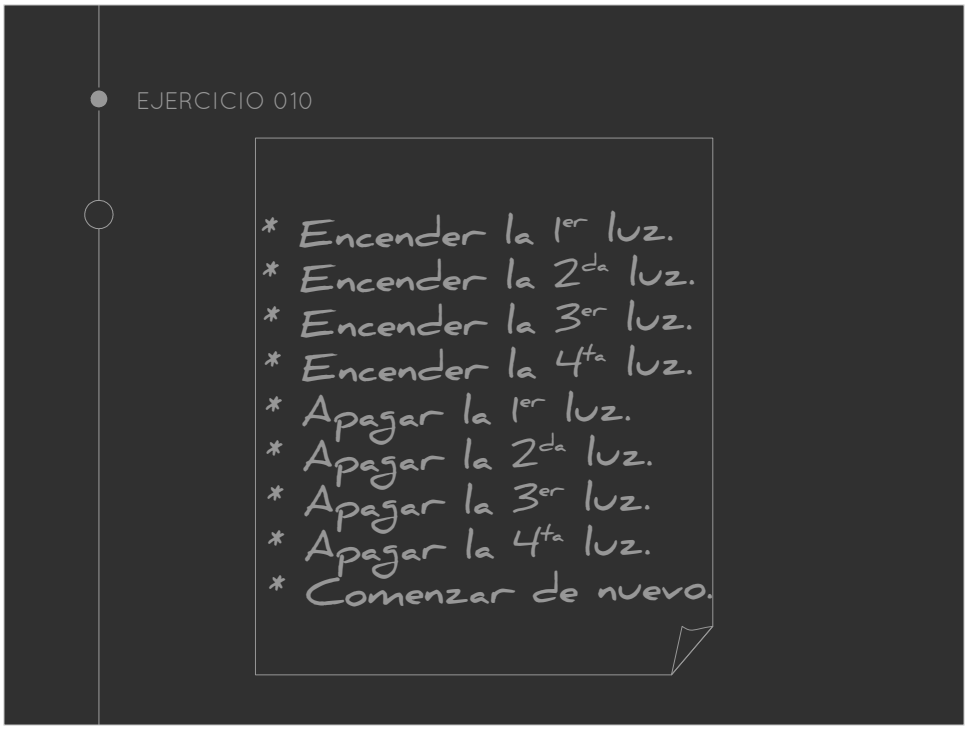
Cada agujerito es un terminal de conexión; e internamente proveen una conexión mecánica y eléctrica entre terminales, de la siguiente forma.



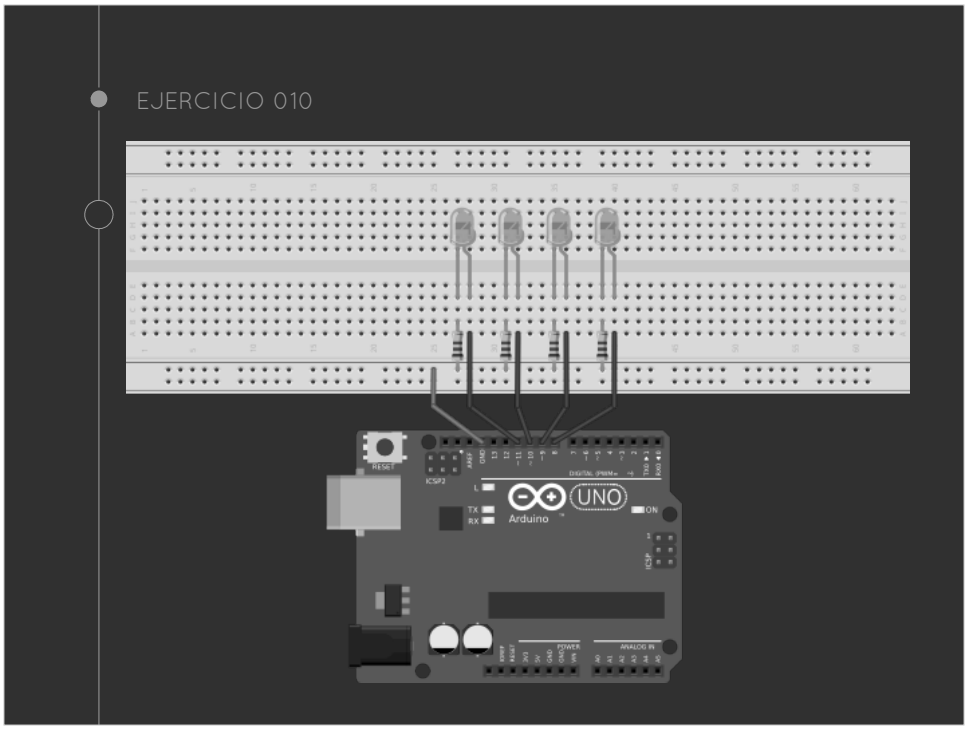
Otra vez, en castellano, en una hojita de papel, vamos a describir el proceso necesario para encender y apagar las luces de la manera que se muestra en la diapositiva.

Van a tener 2 minutitos para realizarlo.

(Despues de los 2 minutos se les pregunta a 2 o tres asistentes cómo les quedó el ejercicio.)



Debe haber quedado parecido a esto.



Primero realizamos las conexiones en el protoboard y después al arduino como indica el esquema.

Las resistencias tienen cuatro colores, deben usar las que tienen el color marrón al lado del dorado.

El diodo LED tiene dos terminales polarizadas. Si observan en el encapsulado van a notar que una es más finita que la otra. Esa terminal finita es el ánodo y en el diagrama está del lado derecho.

EJERCICIO 010

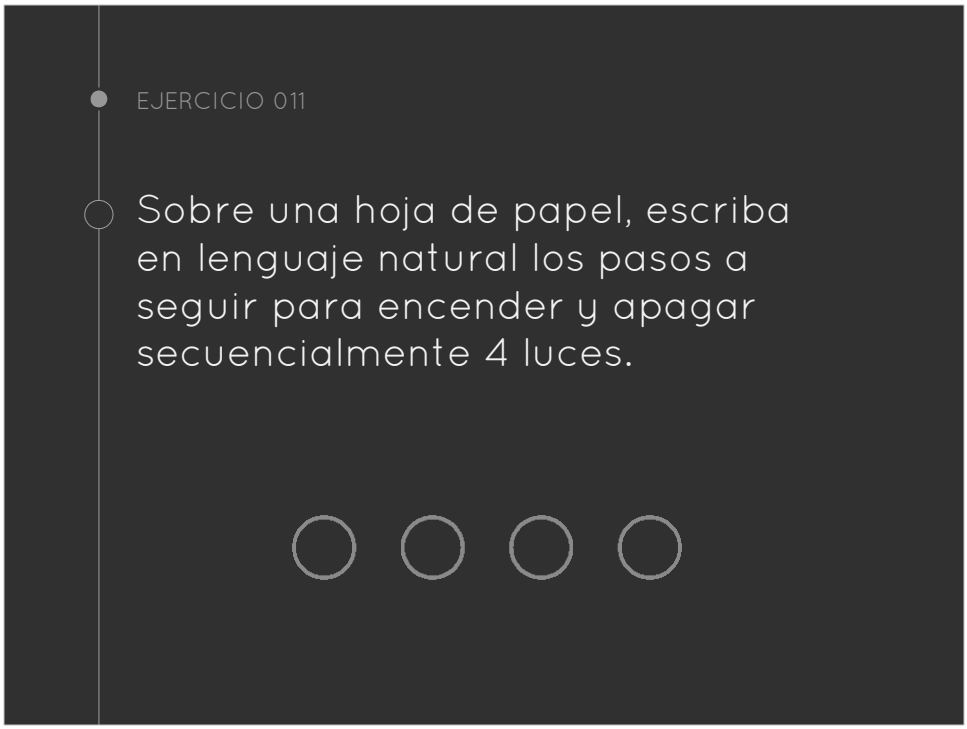
```
void setup() {
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);
}

void loop() {
  digitalWrite(8, HIGH);
  delay(500);
  digitalWrite(9, HIGH);
  delay(500);
  digitalWrite(10, HIGH);
  delay(500);
  digitalWrite(11, HIGH);
  delay(500);
  digitalWrite(8, LOW);
  delay(500);
  digitalWrite(9, LOW);
  delay(500);
  digitalWrite(10, LOW);
  delay(500);
  digitalWrite(11, LOW);
  delay(500);
}
```

Y más o menos ya sabemos como debería quedar el código.

Configuramos los pines y codificamos el encendido y apagado de los LEDs.

Pines altos y bajos y los tiempos de espera.

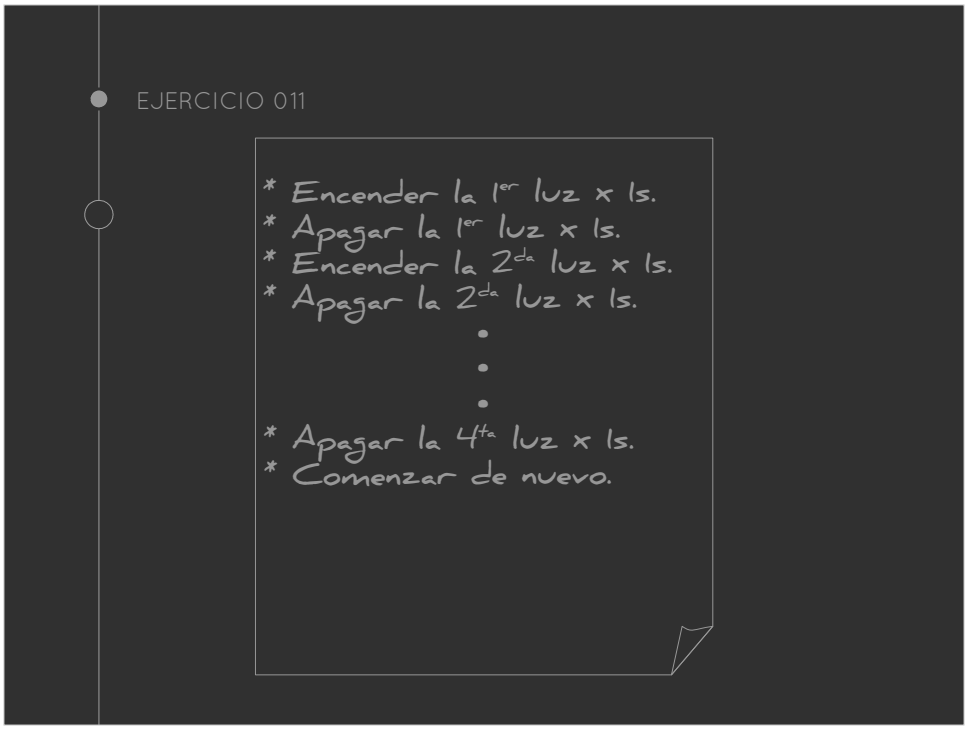
Una vez que terminan lo compilan y lo cargan al arduino.



Este ejercicio es parecido al anterior, pero con una variante en el encendido y apagado.
Ahora se encienden y apagan secuencialmente, como lo muestra la figura.

(Preguntar cómo quedó el código a 2 alumnos)

Debió quedarnos así ó parecido.



¿Todo bien hasta acá?
Muy bien seguimos.

EJERCICIO 011

```
1 void setup() {
2   pinMode(8, OUTPUT);
3   pinMode(9, OUTPUT);
4   pinMode(10, OUTPUT);
5   pinMode(11, OUTPUT);
6 }
7
8 void loop() {
9   digitalWrite(8, HIGH);
10  delay(500);
11  digitalWrite(8, LOW);
12  delay(500);
13  digitalWrite(9, HIGH);
14  delay(500);
15  digitalWrite(9, LOW);
16  delay(500);
17  digitalWrite(10, HIGH);
18  delay(500);
19  digitalWrite(10, LOW);
20  delay(500);
21  digitalWrite(11, HIGH);
22  delay(500);
23  digitalWrite(11, LOW);
24  delay(500);
25 }
```

Veamos qué efecto produce esto.

(dejamos que lo prueben)

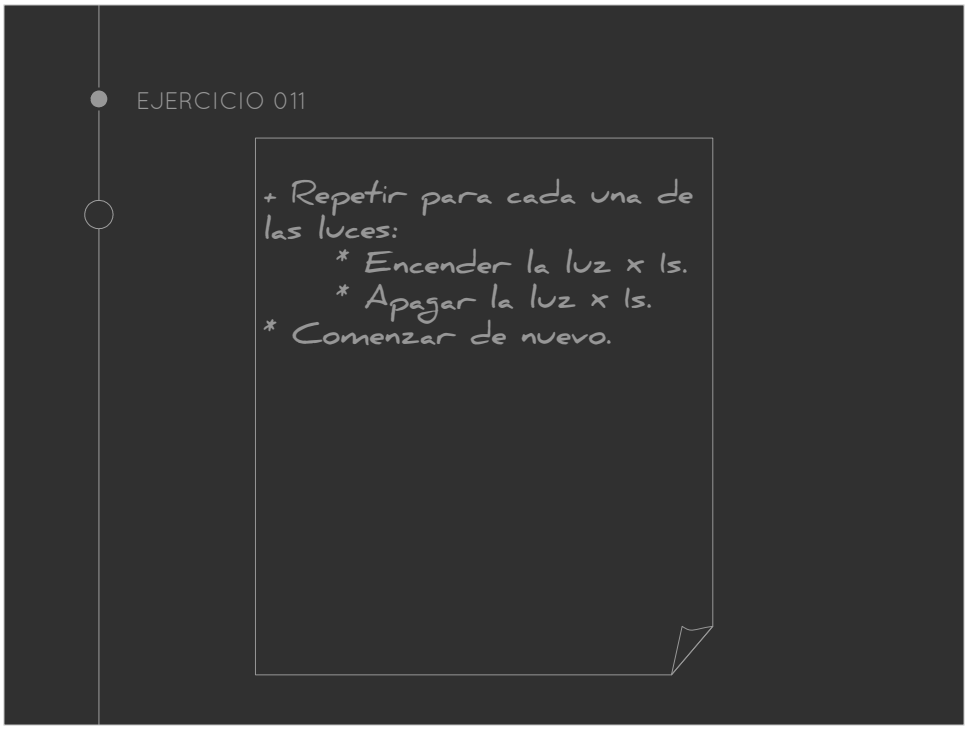
Vamos a hacer una modificación al código anterior.
Quitamos el tiempo de espera, el delay luego del apagado unicamente; quedando el código de la siguiente manera.

EJERCICIO 011

```
1 void setup() {
2   pinMode(8, OUTPUT);
3   pinMode(9, OUTPUT);
4   pinMode(10, OUTPUT);
5   pinMode(11, OUTPUT);
6 }
7
8 void loop() {
9   digitalWrite(8, HIGH);
10  delay(500);
11  digitalWrite(8, LOW);
12  digitalWrite(9, HIGH);
13  delay(500);
14  digitalWrite(9, LOW);
15  digitalWrite(10, HIGH);
16  delay(500);
17  digitalWrite(10, LOW);
18  digitalWrite(11, HIGH);
19  delay(500);
20  digitalWrite(11, LOW);
21 }
```

¿Notaron el cambio?

Y si quisieramos mejorar el código, podemos usar estructuras repetitivas y reducir un poco el código.



En castellano quedaría así.

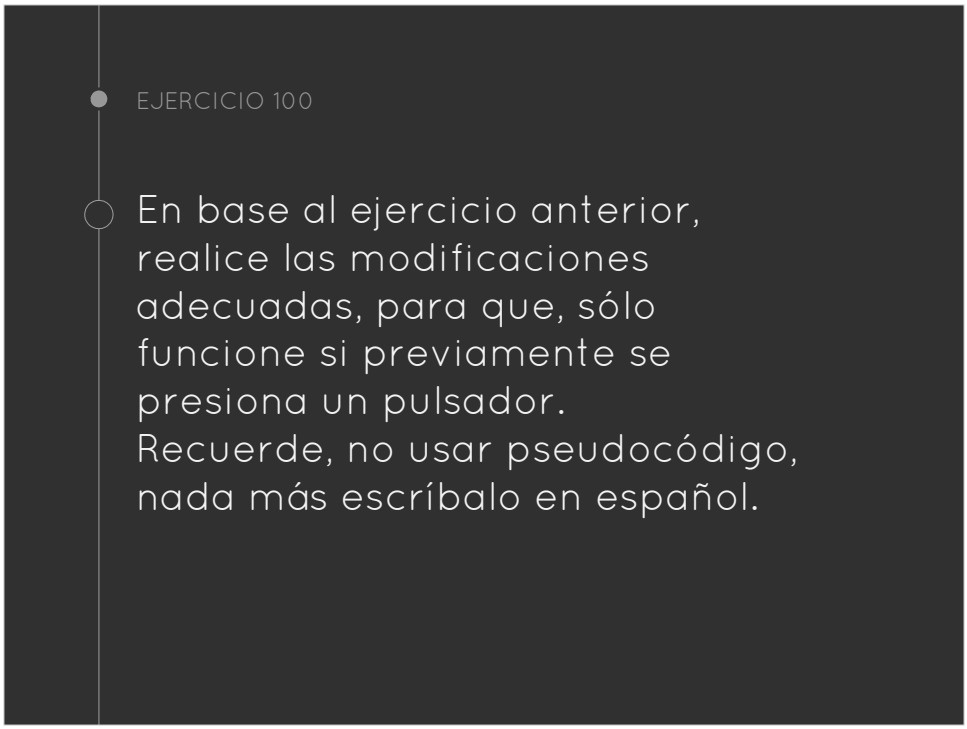
Y el código debería quedar así.

EJERCICIO 011

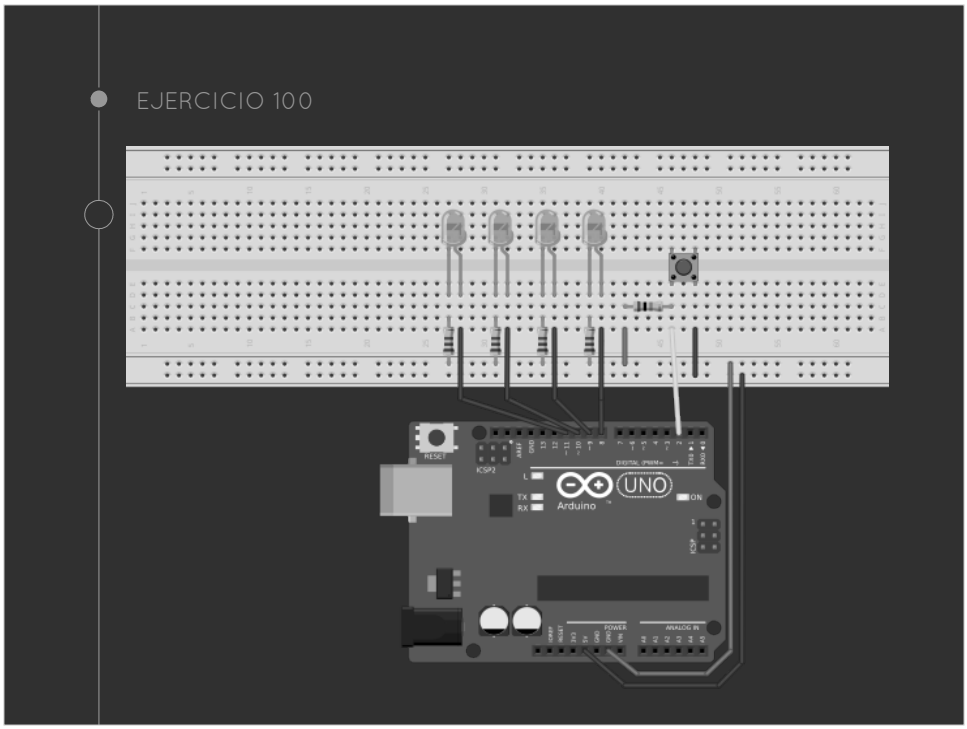
```
void setup() {
    pinMode(8, OUTPUT);
    pinMode(9, OUTPUT);
    pinMode(10, OUTPUT);
    pinMode(11, OUTPUT);
}

void loop() {
    for(int pin = 8; pin <= 11; pin++) {
        digitalWrite(pin, HIGH);
        delay(500);
        digitalWrite(pin, LOW);
    }
}
```

La estructura es el “*para*”. Defino internamente al “*para*” la variabe *pin* y la inicializo en 8 que es el pin donde comienza mi conexión, mientras sea menor o igual a 11 (que es el ultimo pin conectado), ejecuto el código e incremento en 1 la variable *pin*.

Vamos a modificar el código para dejarlo así como lo vemos acá.

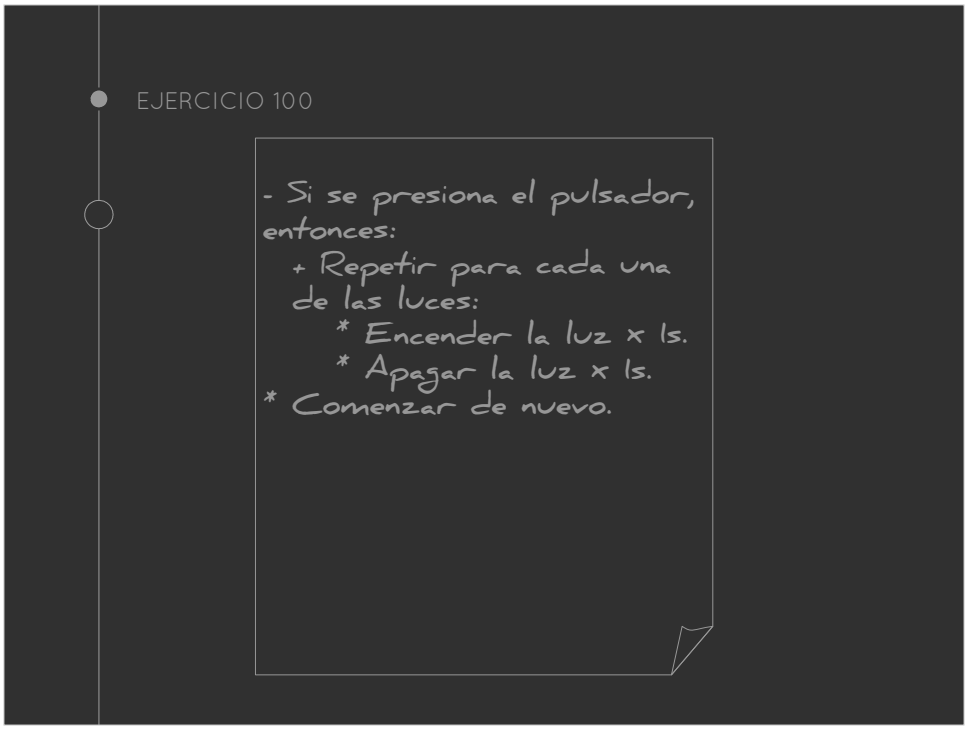


Ahora veremos el tratamiento de entradas digitales.
Mientras realizan el ejercicio, alguien del grupo puede ir realizando las conexiones correspondientes.
(Leer el problema y pasar la diapositiva)



Muy bien.Debemos usar el pulsador y la resistencia que quedaba (la que tiene junto al dorado el color anaranjado).

(Un par de minutos y mostramos como quedaría el ejercicio)



Agregamos una pregunta y en base a la respuesta, que será vardadera ó falsa, se ejecutará ó nó el código.

EJERCICIO 100

```
void setup() {
    pinMode(2, INPUT);
    pinMode(8, OUTPUT);
    pinMode(9, OUTPUT);
    pinMode(10, OUTPUT);
    pinMode(11, OUTPUT);
}

void loop() {
    if(digitalRead(2) == HIGH) {
        for(int pin = 8; pin <= 11; pin++) {
            digitalWrite(pin, HIGH);
            delay(500);
            digitalWrite(pin, LOW);
        }
    }
}
```

Esta es la pregunta: SI (IF) la lectura digital en el pin 2 es HIGH (ALTA) ejecuto el código.
La condición que se debe cumplir va entre paréntesis.

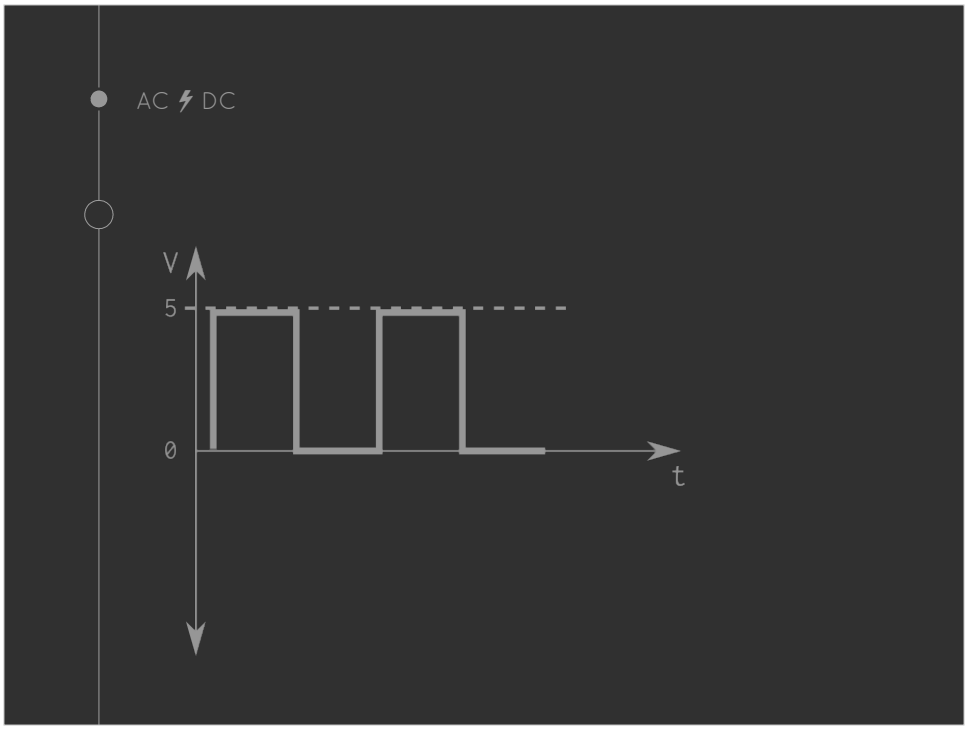
Ahora bien, como HIGH es sinonimo de TRUE (VERDADERO) podemos simplemente preguntar por la lectura digital en el pin 2, quedando de lasiguiente forma.

EJERCICIO 100

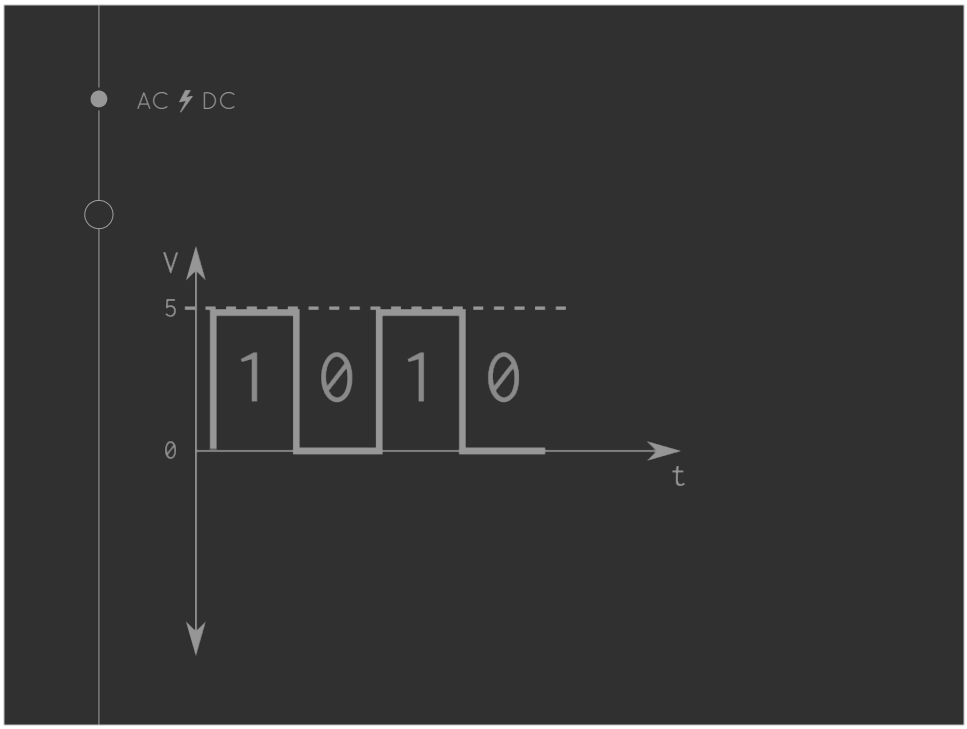
```
void setup() {
    pinMode(2, INPUT);
    pinMode(8, OUTPUT);
    pinMode(9, OUTPUT);
    pinMode(10, OUTPUT);
    pinMode(11, OUTPUT);
}

void loop() {
    if(digitalRead(2)) {
        for(int pin = 8; pin <= 11; pin++) {
            digitalWrite(pin, HIGH);
            delay(500);
            digitalWrite(pin, LOW);
        }
    }
}
```

¿Notaron la diferencia?
Hicimos una modificación en la condición.

Ahora terminamos lo que serían las entradas y salidas digitales y entramos en las analógicas.
¿De qué se trata esto de analógico y en que se diferencia de lo digital? Veamos.



Acá tenemos representada una señal digital en función del tiempo.
Habrá una tensión de 5V en el pico máximo y en los mínimos 0V. Lo que se interpreta como...



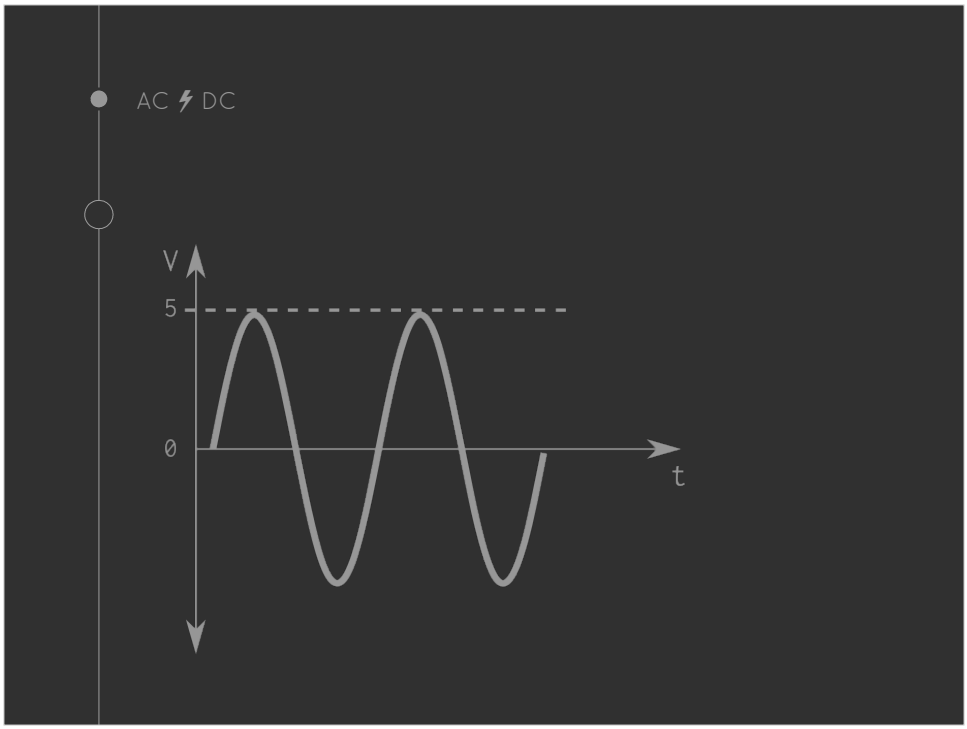
...unos y ceros respectivamente.

Si entre un pin y GND existe una diferencia de potencial de 5V se dice que hay un 1.

Y si entre estos no hay diferencia o sea 0V decimos que hay un 0.

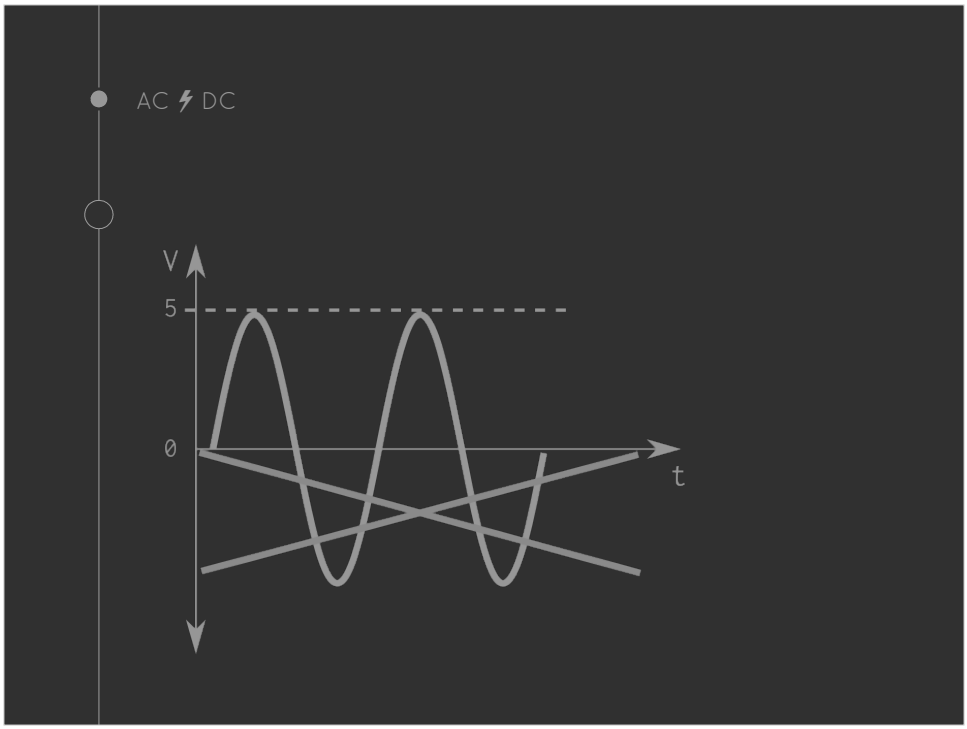
Esto quiere decir que si hablamos de señales digitales sólo tendremos uno de dos valores, 0 o 1.

Sin embargo, cuando hablamos de señales analógicas...

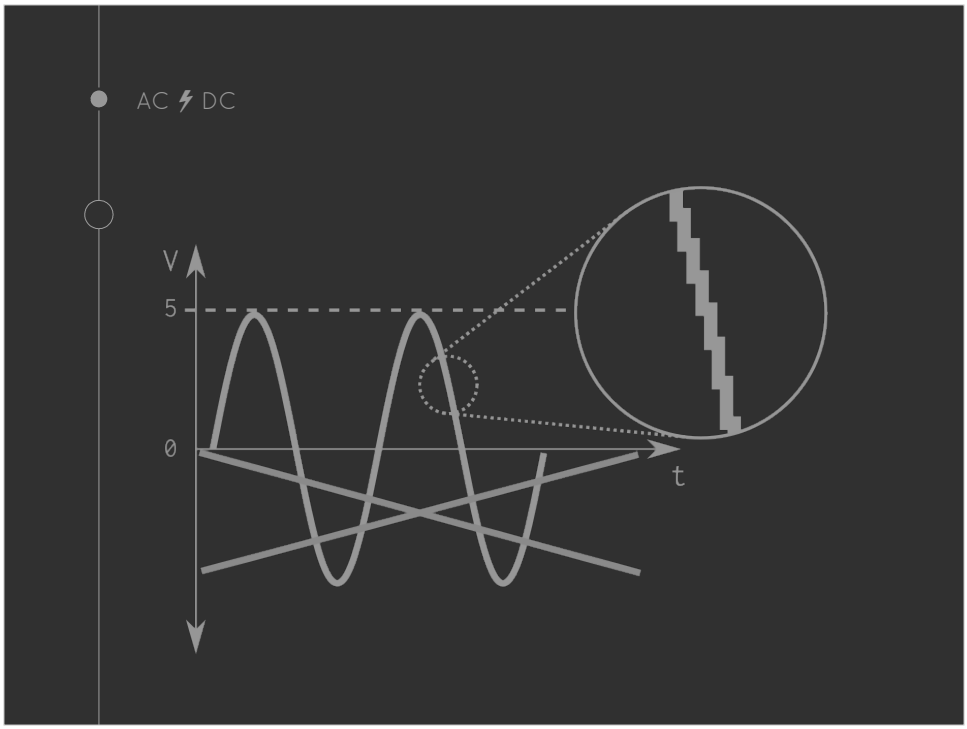


...nos encontramos con que entre 0V y 5V hay infinitos valores intermedios.

Para que el arduino pueda comprender estas señales (recordar que es un artefacto digital) debemos tener algunas consideraciones.
Vamos a prescindir de la parte negativa...



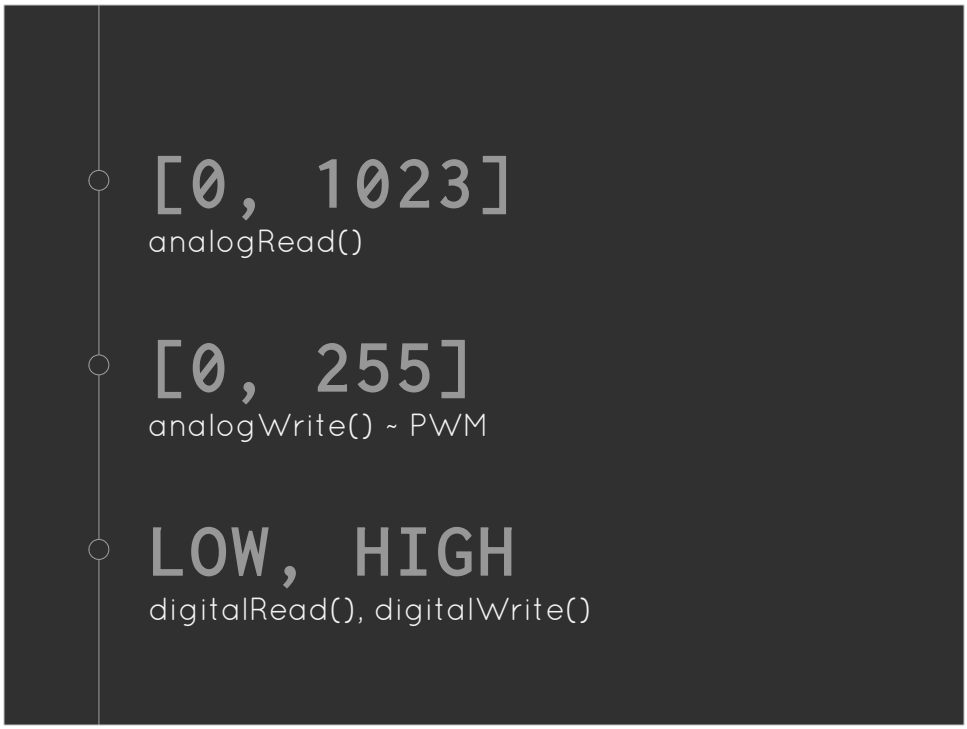
...Y lo que nos queda debe ser tratado como un conjunto discreto (finito, que tiene fin)



Lo que quiere decir que el arduino para las entradas analógicas dividirá los 5V en 1024 partes iguales. Traduciendo, al recibir una señal, esta debe estar entre 0 y 5V y el arduino la va a interpretar como un número entre 0 y 1023.

Por ejemplo, si colocamosuna señal de 5V, el arduino lo interpreta como 1023. Si la señal es de 2,5V (la mitad de 5V), arduino lo entenderá como 511 que es la mitad de 1023.

Y para las salidas analógicas tomaría valores entre 0 y 255 (256 valores). Por ejemplo, si yo quisiera emitir una señal de 5V debería indicarle al arduino que coloque en el pin un valor de 255; si quisiera colocar 2,5V debería indicarle un valor de 127.

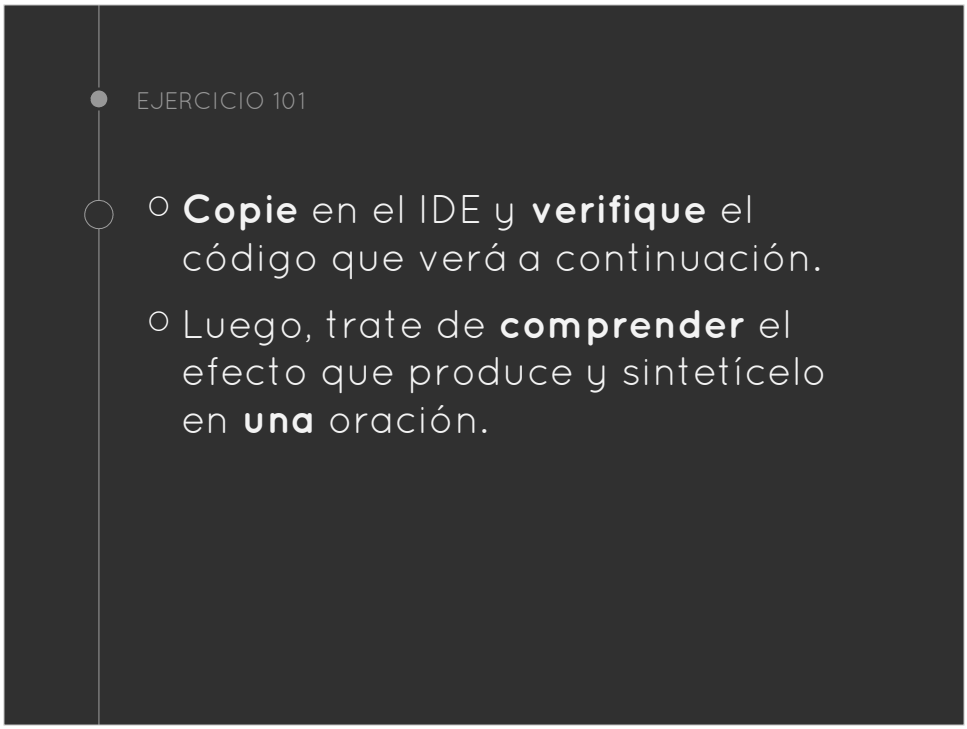


Resumiendo:

Para la entrada analógica, que por cierto no hace falta configurar estos pines, podemos leer valores de entre 0 y 1023, que equivalen avalores comprendidos entre 0V y 5V. La función usada en analogRead() y dentro del paréntesis el número del pin.

Para salida analógica, debemos indicar valores comprendidos entre 0 y 255, obteniendo como salida en el pin valores de entre 0V y 5V. La función para escritura analógica es digitalWrite() y entre parentes lleva dos parámetros, el primero es el pin y el segundo el valor que debe tener.

Y para la entrada y salida digital, que usábamos las funciones digitalRead() y digitalWrite() teníamos solo los valores LOW y HIGH.



Y llegamos al primer ejercicio del entrada/salida analógica.

(Leer el ejercicio y pasar a la siguiente diapositiva)

EJERCICIO 101

```
#define LED 3  // Define una constante llamada LED que equivale al número 3.
               // Esta mantendrá su valor constante y no se podrá alterar
               // durante la ejecución.

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

void loop() {
    for(int v = 1; v <= 255; v++) {
        analogWrite(LED, v);
        delay(20);
    }
    for(int v = 254; v <= 0; v--) {
        analogWrite(LED, v);
        delay(20);
    }
}
```

Van a tener 5 minutitos para copiarlo y analizar qué hace.

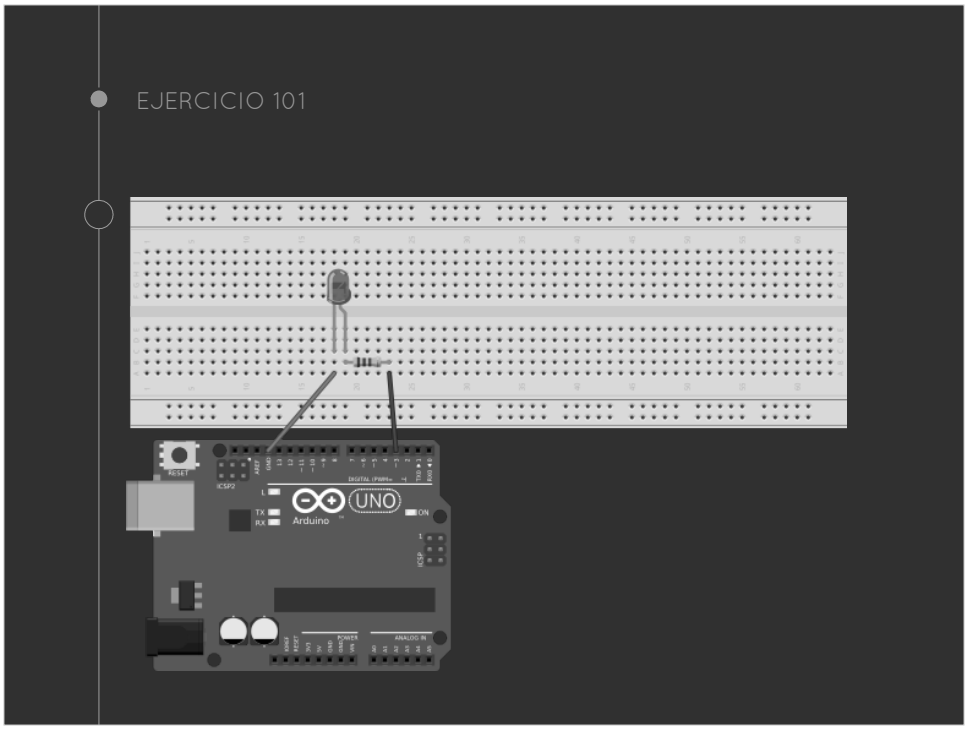
(Transcurrido el tiempo)

¿Qué s lo que pienzan que hace este código?

(soicitan a 2 asistentes que contesten)

Este código hace que un LED partiendo de 0 aumente gradualmente su brillo hasta alcanzar un valor máximo y luego comienza a brillar gradualmente con menor intensidad hasta apagarse y repite el proceso.

EJERCICIO 101



El circuito para probar el código es el siguiente.

(Una vez que lo hayan probado pasar a la siguiente diapositiva)

EJERCICIO 110

- Modifique el código anterior para dejarlo de la siguiente manera.
- Trate de describir en una oración el efecto producido.

(Leer el ejercicio y luego mostrar el código)

EJERCICIO 110

```
#define LED 3
#define POT 4

int lectura_POT;

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

void loop() {
    lectura_POT = analogRead(POT);
    analogWrite(LED, lectura_POT / 4);
    delay(50);
}
```

(este y de ahora en adelante lo explica el profesor)

EJERCICIO 110