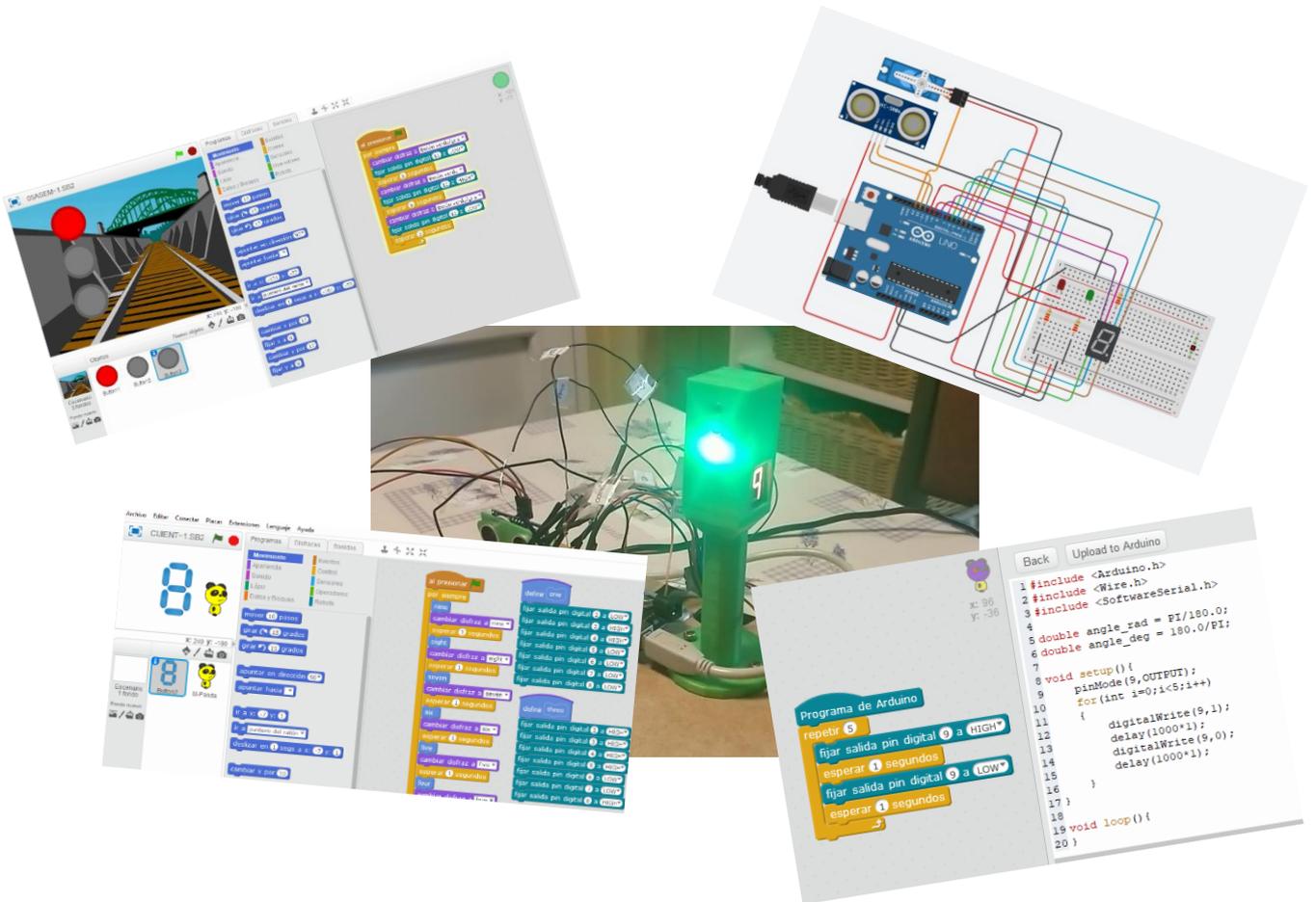


# EJERCICIOS DE MBLOCK PARA APRENDER A PROGRAMAR CON “EL ENTRENADOR DE PROGRAMACIÓN” LA PLACA ARDUINO



DISPLAY							LEDES		ULTRASONIDOS		SERVOMOTOR
a	b	c	d	e	f	g	R	V	TRIGER	ECHO	
2	3	4	5	6	7	8	9	10	11	12	13

**SANTIAGO PÉREZ ANTOLÍN**  
**CEO BOECILLO**



## ÍNDICE

### Contenido

1.- Encender y apagar un LED tres veces. ....	3
2.- Encender y apagar dos LEDs intermitentemente cinco veces. ....	4
3.- Encender y apagar un LED cinco veces intermitentemente con la función FOR. ....	5
4.- Encender y apagar un LED todo el tiempo. Función LOOP. ....	6
5.- Programar el funcionamiento de un SEMÁFORO de tres LEDES. ....	7
6.- Programar la cuenta atrás de un display. ....	9
6.1.- Programación en paralelo: ....	9
6.2.- Programación del display con un solo objeto y disfraces: ....	10
8 A.- Programar el SENSOR DE ULTRASONIDOS: ....	13
9.- Programa un SERVOMOTOR:.....	15
9.A.- Servomotor sube y baja utilizando una variable. ....	15
9.B.- Servomotor suba y baje en función de unas distancias.....	15
9.C.- Programa un SERVOMOTOR para que una barrera de aparcamiento suba y baje. <b>FOR</b> . ....	16
11.- Retos: Barrera de aparcamiento automática.....	16
12.- Entradas analógicas y digitales. ....	19
13.- Control de un servomotor y encendido de 3 LEDs con un potenciómetro. ....	20
14.- Brazo robot. Control servos mediante potenciómetros. ....	21
15.- COCHE FANTÁSTICO LEDS .....	24
16.- MBLOCK Y PYTHON .....	24

## 1.- Encender y apagar un LED tres veces.

### MODULO ARDUINO

```

1 #include <Arduino.h>
2 #include <Wire.h>
3 #include <SoftwareSerial.h>
4
5 double angle_rad = PI/180.0;
6 double angle_deg = 180.0/PI;
7
8 void setup() {
9   pinMode(9,OUTPUT);
10  digitalWrite(9,1);
11  delay(1000*1);
12  digitalWrite(9,0);
13  delay(1000*1);
14  digitalWrite(9,1);
15  delay(1000*1);
16  digitalWrite(9,0);
17  delay(1000*1);
18  digitalWrite(9,1);
19  delay(1000*1);
20  digitalWrite(9,0);
21  delay(1000*1);
22 }
23
24 void loop() {
25 }
    
```

## 2.- Encender y apagar dos LEDs intermitentemente cinco veces.

The image shows a Scratch workspace with two characters: a bear (Bear2) and a panda (M-Panda). The script is as follows:

```

al presionar bandera verde clicada
  fijar salida pin digital 10 a LOW
  cambiar disfraz a PANDA VERDE
  fijar salida pin digital 11 a HIGH
  fijar salida pin digital 9 a LOW
  esperar 1 segundos
  cambiar disfraz a PANDA BLANCO
  fijar salida pin digital 11 a LOW
  fijar salida pin digital 9 a HIGH
  esperar 1 segundos
  cambiar disfraz a PANDA VERDE
  fijar salida pin digital 11 a HIGH
  fijar salida pin digital 9 a LOW
  esperar 1 segundos
  cambiar disfraz a PANDA BLANCO
  fijar salida pin digital 11 a LOW
  fijar salida pin digital 9 a HIGH
  esperar 1 segundos
  cambiar disfraz a PANDA VERDE
  fijar salida pin digital 11 a HIGH
  fijar salida pin digital 9 a LOW
  esperar 1 segundos
  cambiar disfraz a PANDA BLANCO
  fijar salida pin digital 11 a LOW
  fijar salida pin digital 9 a HIGH
  esperar 1 segundos
  fijar salida pin digital 10 a LOW
  cambiar disfraz a PANDA VERDE
  fijar salida pin digital 11 a HIGH
  fijar salida pin digital 9 a LOW
  esperar 1 segundos
  
```

The image shows a Scratch workspace with two characters: a bear (Bear2) and a panda (M-Panda). The script is as follows:

```

al presionar bandera verde clicada
  cambiar disfraz a OSO BLANCO
  esperar 1 segundos
  cambiar disfraz a OSO ROJO
  esperar 1 segundos
  cambiar disfraz a OSO BLANCO
  esperar 1 segundos
  cambiar disfraz a OSO ROJO
  esperar 1 segundos
  cambiar disfraz a OSO BLANCO
  esperar 1 segundos
  cambiar disfraz a OSO ROJO
  esperar 1 segundos
  cambiar disfraz a OSO BLANCO
  esperar 1 segundos
  cambiar disfraz a OSO ROJO
  esperar 1 segundos
  cambiar disfraz a OSO BLANCO
  esperar 1 segundos
  cambiar disfraz a OSO ROJO
  
```

### 3.- Encender y apagar un LED cinco veces intermitentemente con la función FOR.

The screenshot shows the Scratch IDE interface. On the left, a scene titled 'Escenario 2 fondos' features a purple panda character named 'M-Panda' on a city street. The script editor on the right contains the following code:

```

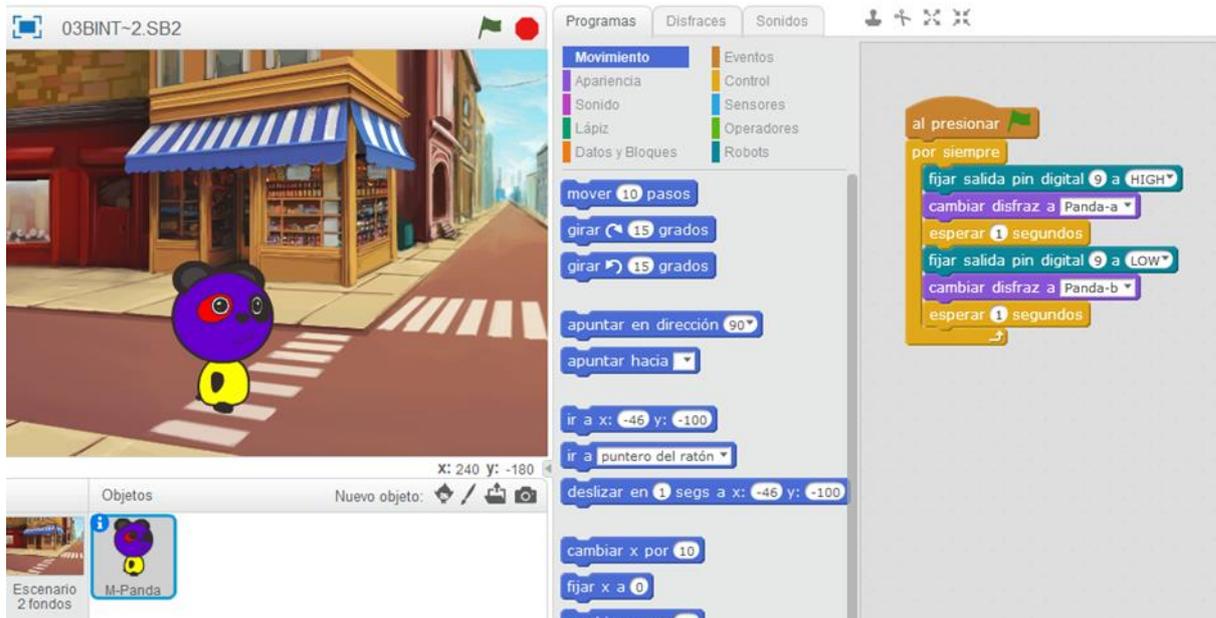
al presionar bandera verde clicada
  repetir (5)
    fijar salida pin digital (9) a HIGH
    mostrar
    esperar (1) segundos
    fijar salida pin digital (9) a LOW
    esconder
    esperar (1) segundos
  
```

The screenshot shows the Arduino IDE interface with the compiled C++ code. The code is as follows:

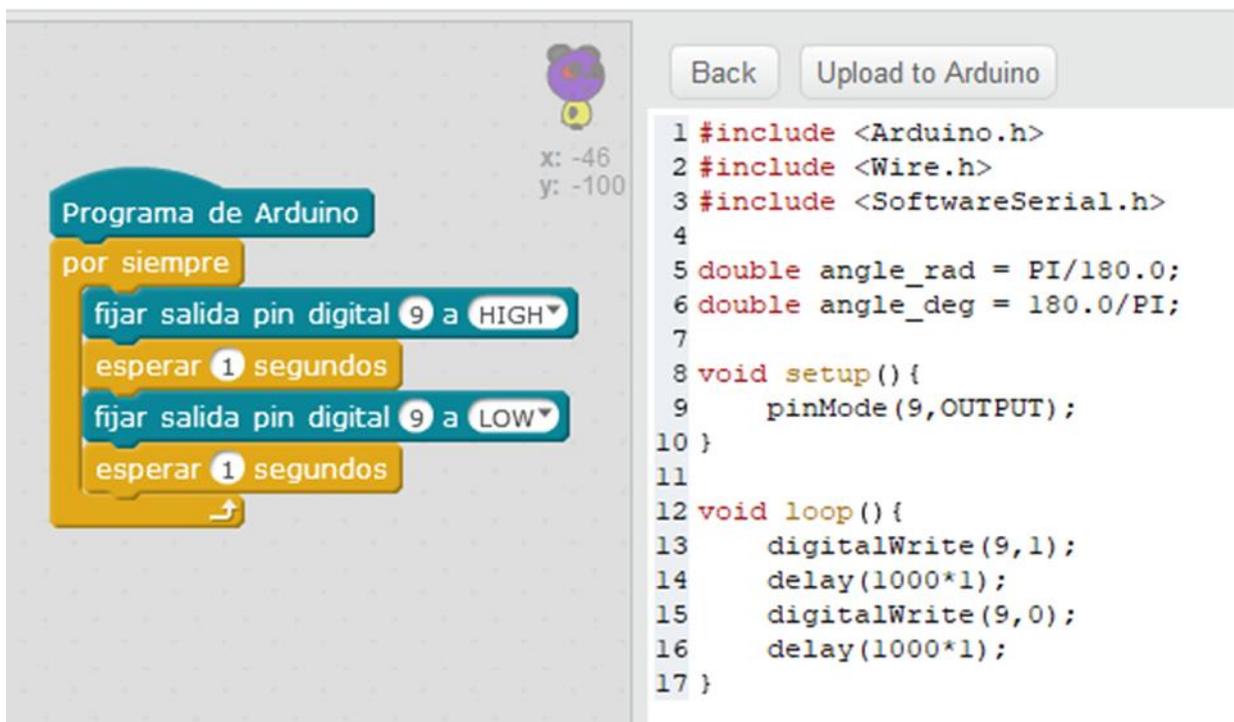
```

1 #include <Arduino.h>
2 #include <Wire.h>
3 #include <SoftwareSerial.h>
4
5 double angle_rad = PI/180.0;
6 double angle_deg = 180.0/PI;
7
8 void setup() {
9   pinMode(9, OUTPUT);
10  for(int i=0; i<5; i++)
11  {
12    digitalWrite(9, 1);
13    delay(1000*1);
14    digitalWrite(9, 0);
15    delay(1000*1);
16  }
17 }
18
19 void loop() {
20 }
  
```

#### 4.- Encender y apagar un LED todo el tiempo. Función LOOP.



The screenshot shows the Scratch IDE interface. On the left, a panda character is on a street scene. The right panel shows a 'por siempre' loop with the following blocks: 'fijar salida pin digital 9 a HIGH', 'cambiar disfraz a Panda-a', 'esperar 1 segundos', 'fijar salida pin digital 9 a LOW', 'cambiar disfraz a Panda-b', and 'esperar 1 segundos'. The bottom panel shows a 'Programa de Arduino' block with a similar loop structure.



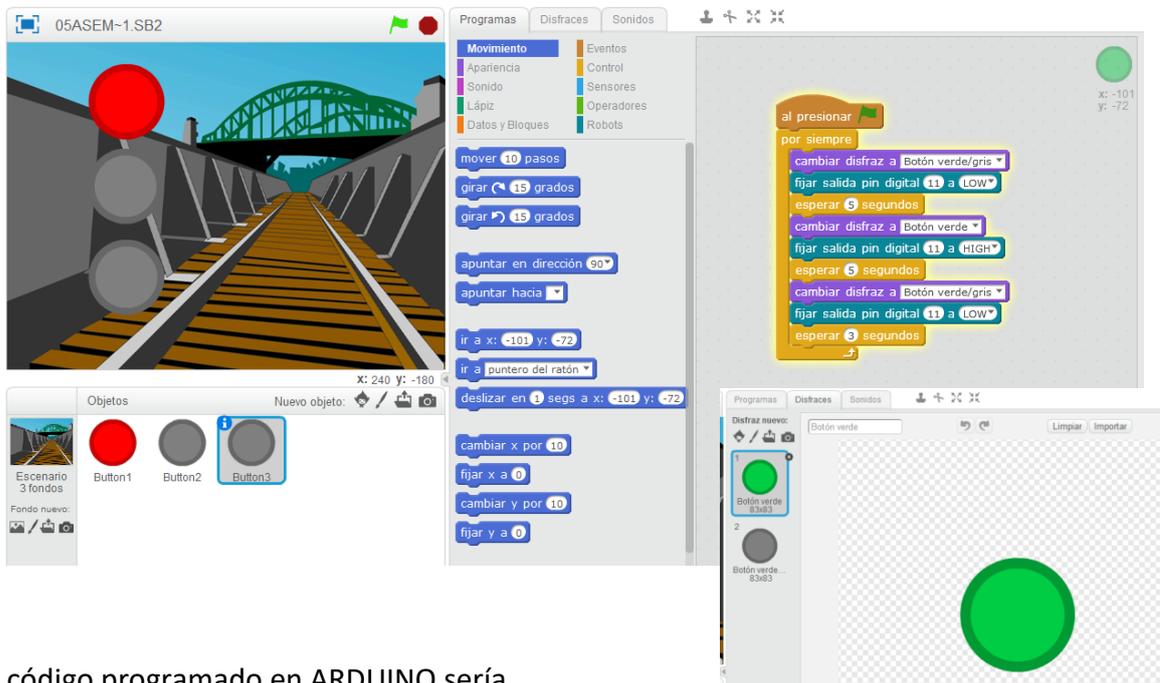
```
1 #include <Arduino.h>
2 #include <Wire.h>
3 #include <SoftwareSerial.h>
4
5 double angle_rad = PI/180.0;
6 double angle_deg = 180.0/PI;
7
8 void setup() {
9     pinMode(9, OUTPUT);
10 }
11
12 void loop() {
13     digitalWrite(9, 1);
14     delay(1000*1);
15     digitalWrite(9, 0);
16     delay(1000*1);
17 }
```

## 5.- Programar el funcionamiento de un SEMÁFORO de tres LEDES.

A.- Creando tres objetos que simularán el funcionamiento de cada LED.

TIEMPO	COCHES		
	ROJO	NARANJA	VERDE
	PIN 9	PIN 10	PIN 11
5 segundos	HIGH	LOW	LOW
5 segundos	LOW	LOW	HIGH
3 segundos	LOW	HIGH	LOW

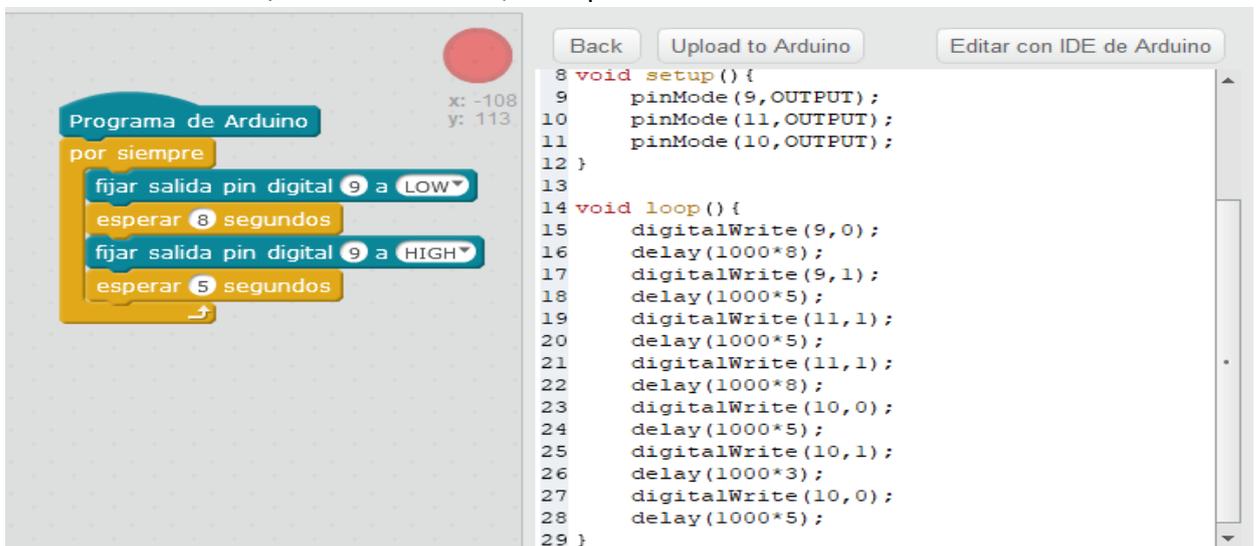
Ayuda: para un objeto la programación sería:



El código programado en ARDUINO sería.

Tendremos que eliminar de los tres objetos los bloques morados(apariencia).

Si vamos a: EDITAR \ MODO ARDUINO, nos aparecerá:

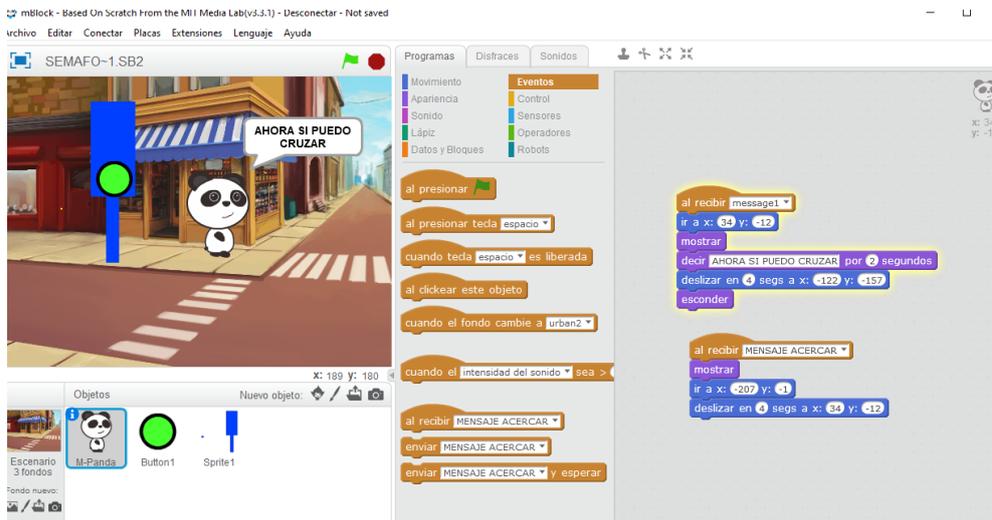


B.- Realiza el programa creando solo un objeto y a partir de él sus disfraces con los tres colores.



RETO: Crea un objeto (PANDA) y un escenario. El panda deberá cruzar la calle cuando el semáforo de peatones esté en verde y deberá esperar cuando esté en rojo. Añadir que haga un comentario.

Ayuda:



RETO: Programa un semáforo de 5 ledes ( 3 coches y 2 peatones).

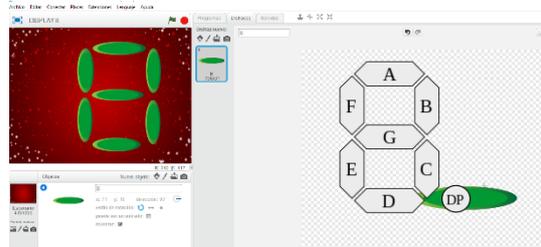
## PROGRAMACIÓN SEGÚN TABLA

TIEMPO	COCHES			PEATONES	
	ROJO	NARANJA	VERDE	ROJO	VERDE
	PIN 9	PIN 10	PIN 11	PIN 12	PIN 13
5 segundos	HIGH	LOW	LOW	LOW	HIGH
5 segundos	LOW	LOW	HIGH	HIGH	LOW
3 segundos	LOW	HIGH	LOW	HIGH	LOW

## 6.- Programar la cuenta atrás de un display.

### 6.1.- Programación en paralelo:

A.- En Mblock diseña un objeto que se semeje a un display de siete segmentos, a continuación, duplica ese objeto seis veces y haz que cada objeto simule un led del display. Nombra los objetos con las letras correspondientes a las del display. También crea un fondo.



B.- Completa la tabla indicando el estado en el que tiene que estar cada led (Objeto) a lo largo de la cuenta atrás. (1 = encendido, 0 = apagado). (1 punto).

		Pin 2	Pin 3	Pin 4	Pin 5	Pin 6	Pin 7	Pin 8	
TIEMPO	VER	a	b	c	d	e	f	g	
0-1 seg	9	1	1	1	0	0	1	1	
1-2	8	1	1	1	1	1	1	1	
2-3	7	1	1	1	0	0	0	0	
3-4	6	1	0	1	1	1	1	1	
4-5	5	1	0	1	1	0	1	1	
5-6	4	1	1	0	0	0	1	1	
6-7	3	1	1	1	1	0	0	1	
7-8	2	1	1	0	1	1	0	1	
8-9	1	0	1	1	0	0	0	0	
9-10	0	1	1	1	1	1	1	0	

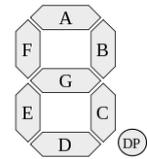
C.- Realiza la programación de los objetos para la cuenta atrás.

RETO: Después de la cuenta crea una animación en la que despegue un cohete de otro escenario, aterrice en otro escenario y salga otro objeto, de la nave, que deberá hacer un comentario.

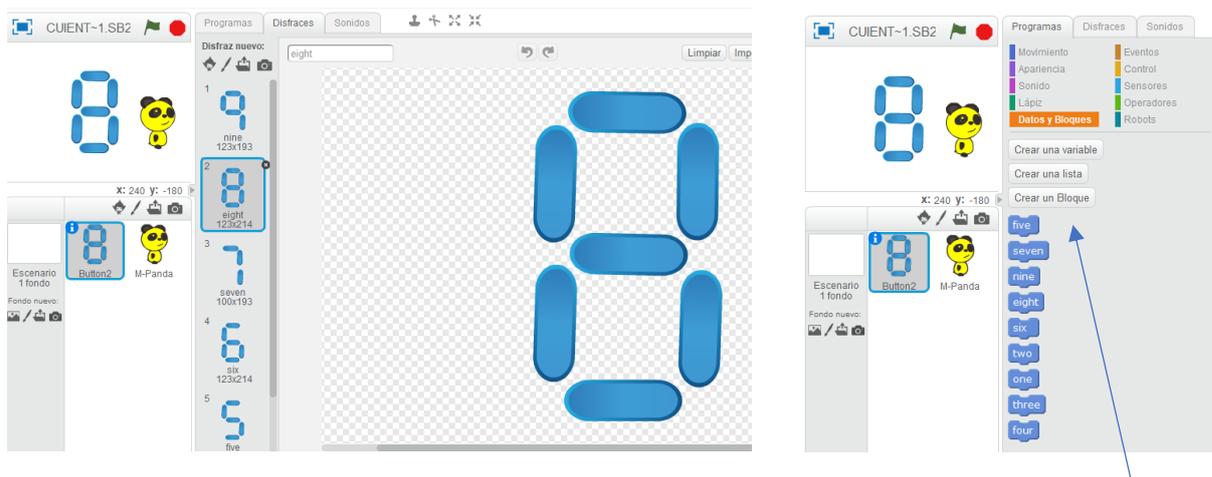
## 6.2.- Programación del display con un solo objeto y disfraces:

A.- Completa la tabla de los pines de conexión.

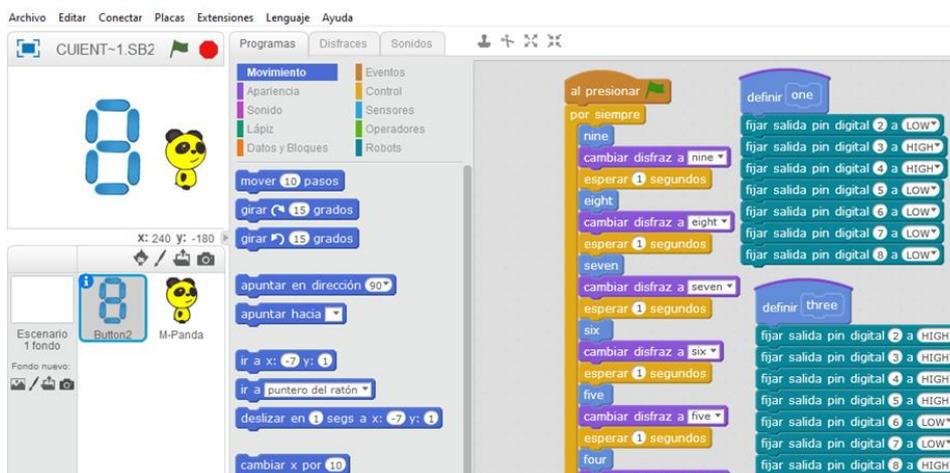
	Pin 2	Pin 3	Pin 4	Pin 5	Pin 6	Pin 7	Pin 8	
VER	a	b	c	d	e	f	g	
9								
8								
7								
6								
5								
4								
3								
2								
1								
0								



B.- Crea los disfraces.



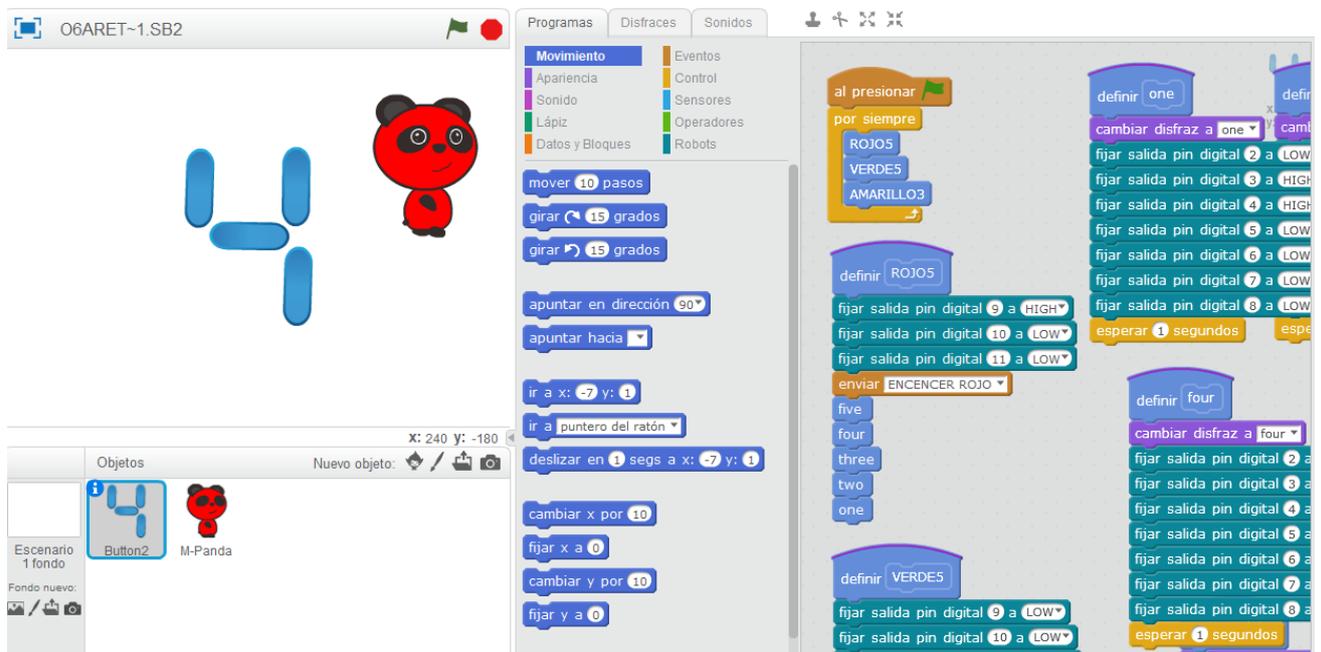
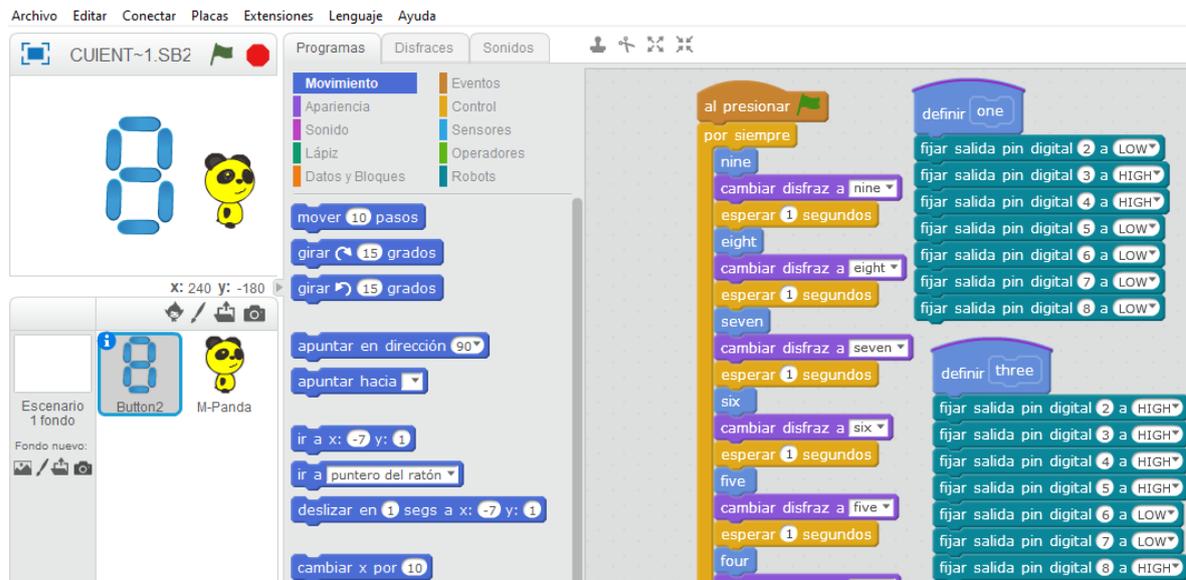
C.- Crea los Bloques de cada número y prográmalo para que funcione a la vez con Arduino.



## 7.- Reto:

Realiza el programa de un semáforo en el que se indique mediante un DISPLAY la cuenta atrás de lo que dura cada color. Para ello una vez que tengas creado los bloques de cada número, crea tres bloques que contengan, cada uno, la programación de los LEDES y la cuenta atrás. Finalmente estos tres bloques los pondrás dentro de un POR SIEMPRE.

SOLUCIÓN:





## 8 A.- Programar el SENSOR DE ULTRASONIDOS:

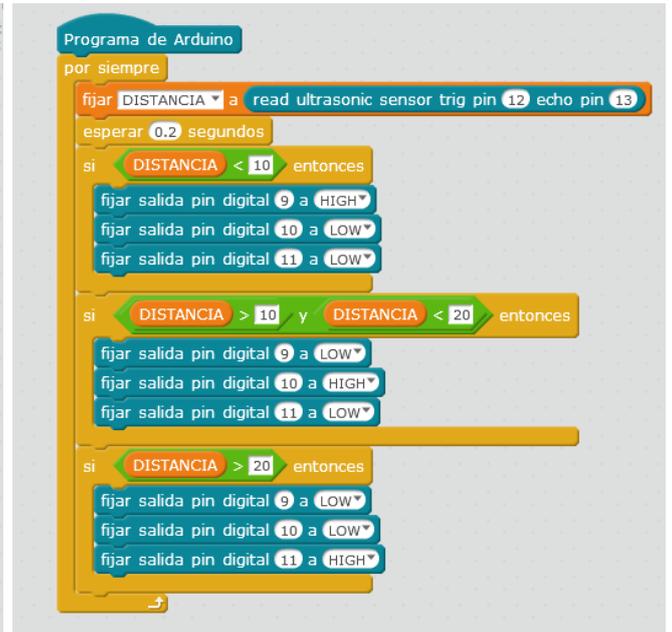
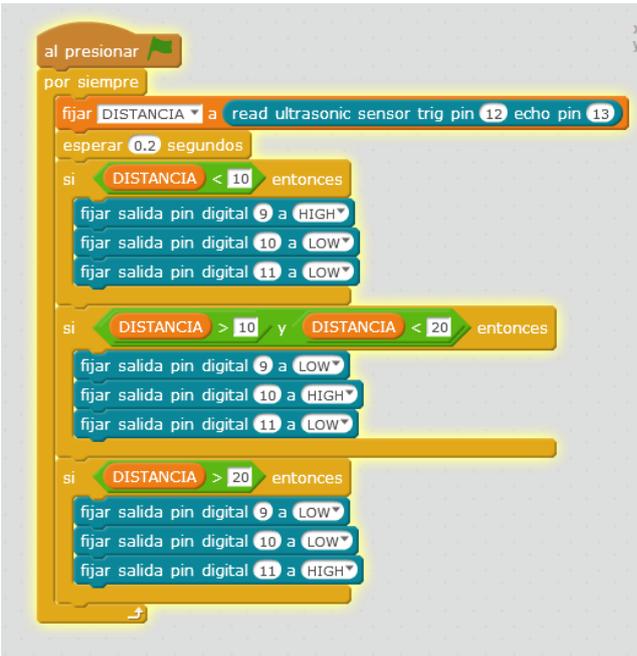
Para que en función de la distancia de un objeto a la barrera cambie el color de dos LEDES del semáforo. **IF...ELSE**. Tendrás que crear una variable (una caja en la que almacenarás los valores de lectura del ultrasonidos a la que llamarás "DISTANCIA")

The image shows two screenshots of the Arduino IDE. The left screenshot displays a block-based code editor with the following logic: 'al presionar' (when button pressed) leads to a 'por siempre' (forever) loop. Inside the loop, it sets 'DISTANCIA' to the value of 'read ultrasonic sensor trig pin 12 echo pin 13', waits for 1 second, and then checks 'if DISTANCIA < 20'. If true, it sets pin 9 to HIGH and pin 10 to LOW. If false, it sets pin 9 to LOW and pin 10 to HIGH. The right screenshot shows the same code in text format, titled 'Programa de Arduino'.

```
1 #include <Arduino.h>
2 #include <Wire.h>
3 #include <SoftwareSerial.h>
4
5 double angle_rad = PI/180.0;
6 double angle_deg = 180.0/PI;
7 double DISTANCIA;
8 float getDistance(int trig,int echo){
9     pinMode(trig,OUTPUT);
10    digitalWrite(trig,LOW);
11    delayMicroseconds(2);
12    digitalWrite(trig,HIGH);
13    delayMicroseconds(10);
14    pinMode(echo,INPUT);
15    return pulseIn(echo,HIGH,30000)/58.0;
16 }
17
18 void setup() {
19     pinMode(9,OUTPUT);
20     pinMode(10,OUTPUT);
21 }
```

```
23 void loop() {
24     DISTANCIA = getDistance(12,13);
25     delay(1000*1);
26     if((DISTANCIA) < (20)){
27         digitalWrite(9,1);
28         digitalWrite(10,0);
29     }else{
30         digitalWrite(9,0);
31         digitalWrite(10,1);
32     }
33 }
```

8 B.- Programar el ULTRASONIDOS para que en función de la distancia de un objeto a la barrera cambie el color de tres LEDES del semáforo. **IF , & y operadores de**



comparación.

```
sketch_mar02a $
#include <Arduino.h>
#include <Wire.h>
#include <SoftwareSerial.h>

double angle_rad = PI/180.0;
double angle_deg = 180.0/PI;
double DISTANCIA;
float getDistance(int trig,int echo) {
    pinMode(trig, OUTPUT);
    digitalWrite(trig, LOW);
    delayMicroseconds(2);
    digitalWrite(trig, HIGH);
    delayMicroseconds(10);
    pinMode(echo, INPUT);
    return pulseIn(echo, HIGH, 30000)/58.0;
}

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

```

```
void loop() {
    DISTANCIA = getDistance(12,13);
    delay(1000*0.2);
    if((DISTANCIA) < (10)){
        digitalWrite(9,1);
        digitalWrite(10,0);
        digitalWrite(11,0);
    }
    if(((DISTANCIA) > (10)) & ((DISTANCIA) < (20))){
        digitalWrite(9,0);
        digitalWrite(10,1);
        digitalWrite(11,0);
    }
    if((DISTANCIA) > (20)){
        digitalWrite(9,0);
        digitalWrite(10,0);
        digitalWrite(11,1);
    }
}

```

## 9.- Programa un SERVOMOTOR:

### 9.A.- Servomotor sube y baja utilizando una variable.



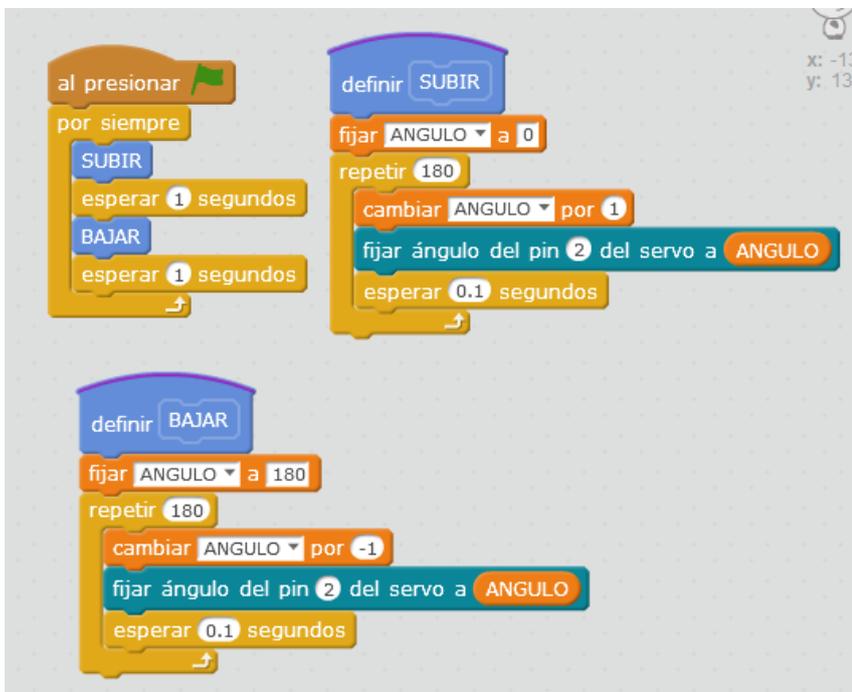
The screenshot shows the Scratch IDE interface. On the left, a variable named 'ANGULO' is defined with a value of 117. The 'Datos y Bloques' category is selected in the left sidebar. The main workspace contains a script starting with 'al presionar' (when green flag clicked), followed by 'fijar ANGULO a 0' (set ANGULO to 0). A 'por siempre' (forever) loop contains two repeating blocks: 'repetir 180' (repeat 180 times) containing 'fijar ANGULO a ANGULO + 1' (set ANGULO to ANGULO + 1) and 'fijar ángulo del pin 13 del servo a ANGULO' (set servo pin 13 angle to ANGULO). This is followed by 'esperar 2 segundos' (wait 2 seconds), 'fijar ANGULO a 180' (set ANGULO to 180), another 'repetir 180' loop containing 'fijar ANGULO a ANGULO - 1' (set ANGULO to ANGULO - 1) and 'fijar ángulo del pin 13 del servo a ANGULO' (set servo pin 13 angle to ANGULO), and finally another 'esperar 2 segundos' block.

### 9.B.- Servomotor suba y baje en función de unas distancias.



The screenshot shows a Scratch script starting with 'al presionar' (when green flag clicked), followed by a 'por siempre' (forever) loop. The first block in the loop is 'fijar DISTANCIA a read ultrasonic sensor trig pin 12 echo pin 13' (set DISTANCIA to read ultrasonic sensor trig pin 12 echo pin 13). This is followed by 'esperar 1 segundos' (wait 1 seconds). A 'si' (if) block checks 'DISTANCIA < 20'. If true, it sets 'ángulo del pin 6 del servo a 180' (set servo pin 6 angle to 180) and waits 1 second. If false, it sets 'ángulo del pin 6 del servo a 0' (set servo pin 6 angle to 0) and waits 1 second.

## 9.C.- Programa un SERVOMOTOR para que una barrera de aparcamiento suba y baje. FOR.



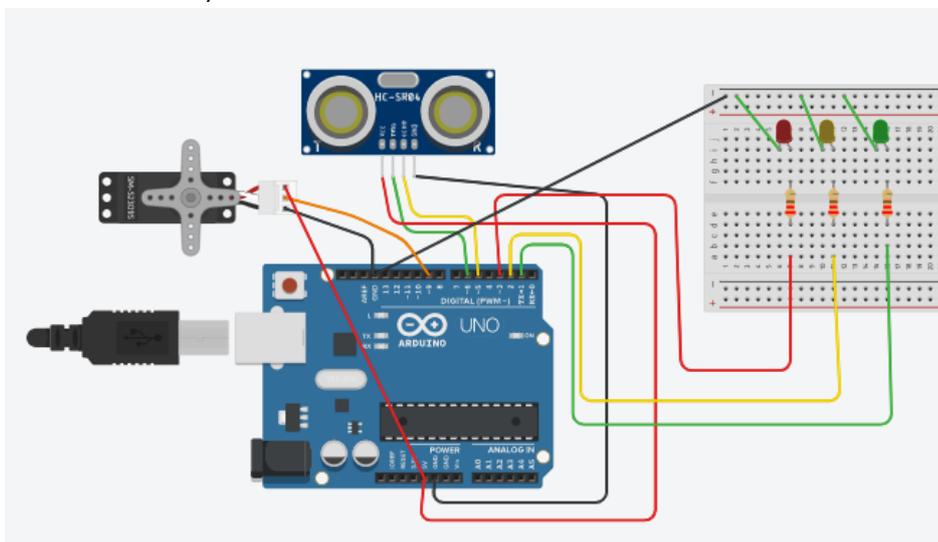
```

5 #include <Servo.h>
6
7 double angle_rad = PI/180.0;
8 double angle_deg = 180.0/PI;
9 void SUBIR();
10 double ANGULO;
11 Servo servo_2;
12 void BAJAR();
13
14 void SUBIR()
15 {
16     ANGULO = 0;
17     for(int i=0;i<180;i++)
18     {
19         ANGULO += 1;
20         servo_2.write(ANGULO);
21         delay(1000*0.1);
22     }
23 }
24
25 void BAJAR()
26 {
27     ANGULO = 180;
28     for(int i=0;i<180;i++)
29     {
30         ANGULO += -1;
31         servo_2.write(ANGULO);
32         delay(1000*0.1);
33     }
34 }
35
36 void setup(){
37     servo_2.attach(2);
38 }
39
40 void loop(){
41     SUBIR();
42     delay(1000*1);
43     BAJAR();
44     delay(1000*1);
45 }

```

## 11.- Retos: Barrera de aparcamiento automática.

RETO A.- Realiza un Sketch en Mblock que simule el acceso a un parking en el que cuando no haya un objeto a menos de 10 cm de sensor de ultrasonidos el semáforo permanecerá en rojo y la barrera bajada y cuando esté a menos de 10 cm , el semáforo pasará a verde, se elevará una barrera y al cabo de 10 segundos se bajará la barrera y cambiará el semáforo a color rojo. **(NO UTILIZAR EL PIN 1).**



RETO B: A partir de la configuración de la tabla programa una barrera de estacionamiento en la que participen coordinados todos los elementos.

DISPLAY							LEDES		ULTRASONIDOS		SERVOMOTOR
a	b	c	d	e	f	g	R	V	TRIGER	ECHO	
2	3	4	5	6	7	8	9	10	11	12	13

AYUDA.

```

al presionar
por siempre
  fijar DISTANCIA a read ultrasonic sensor trig pin 6 echo pin 5
  si DISTANCIA < 10 entonces
    fijar salida pin digital 2 a HIGH
    fijar salida pin digital 3 a LOW
    fijar salida pin digital 4 a LOW
  si DISTANCIA > 10 y DISTANCIA < 20 entonces
    fijar salida pin digital 2 a LOW
    fijar salida pin digital 3 a HIGH
    fijar salida pin digital 4 a LOW
  si DISTANCIA > 20 y DISTANCIA < 50 entonces
    fijar salida pin digital 2 a HIGH
    fijar salida pin digital 3 a HIGH
    fijar salida pin digital 4 a LOW
  esperar 1 segundos
  
```

## SOLUCIÓN RETO B.

The screenshot shows the Arduino IDE interface for a project named 'M-Panda'. The main workspace contains a block-based code for controlling a servo motor and LEDs based on distance measurements. The code is structured as follows:

```

al presionar
  fijar ángulo del pin 9 del servo a 0°
  por siempre
    fijar DISTANCIA a read ultrasonic sensor trig pin 11 echo pin 12
    si DISTANCIA < 10 entonces
      fijar ángulo del pin 13 del servo a 90°
      VERDES
    si no
      fijar ángulo del pin 13 del servo a 0°
      ROJOS
  
```

The code defines several functions for setting servo angles and LED states:

- definir ROJOS:** Sets pins 2, 3, 4, 5, 6, 7, 8, and 9 to HIGH and pins 10, 11, 12 to LOW. It then sends the 'ENCENDER ROJO' command and waits 1 second.
- definir VERDES:** Sets pins 2, 3, 4, 5, 6, 7, 8, and 9 to LOW and pins 10, 11, 12 to HIGH. It then sends the 'ENCENDER VERDE' command and waits 1 second.
- definir one through nine:** Each function sets a specific pin to HIGH and all other pins to LOW. It then sends a corresponding 'ENCENDER' command (e.g., 'ENCENDER one') and waits 1 second.

The bottom panel shows the 'Objetos' palette with a 'Button2' and 'M-Panda' character. The background of the workspace is a red starry field with a panda character and blue limbs.

## 12.- Entradas analógicas y digitales.

11VARI~1.SB2

VALOR 0

IIZQ 0

IDER 0

POTENCIOMETRO 177

LDR 813

Objetos

Nuevo objeto:

Escenario 1 fondo

Fondo nuevo:

M-Panda

Programas Disfraces Sonidos

Movimiento Apariencia Sonido Lápiz Datos y Bloques

Eventos Control Sensores Operadores Robots

Crear una variable

IDER

IIZQ

LDR

POTENCIOMETRO

VALOR

fijar LDR a 0

cambiar LDR por 1

mostrar variable LDR

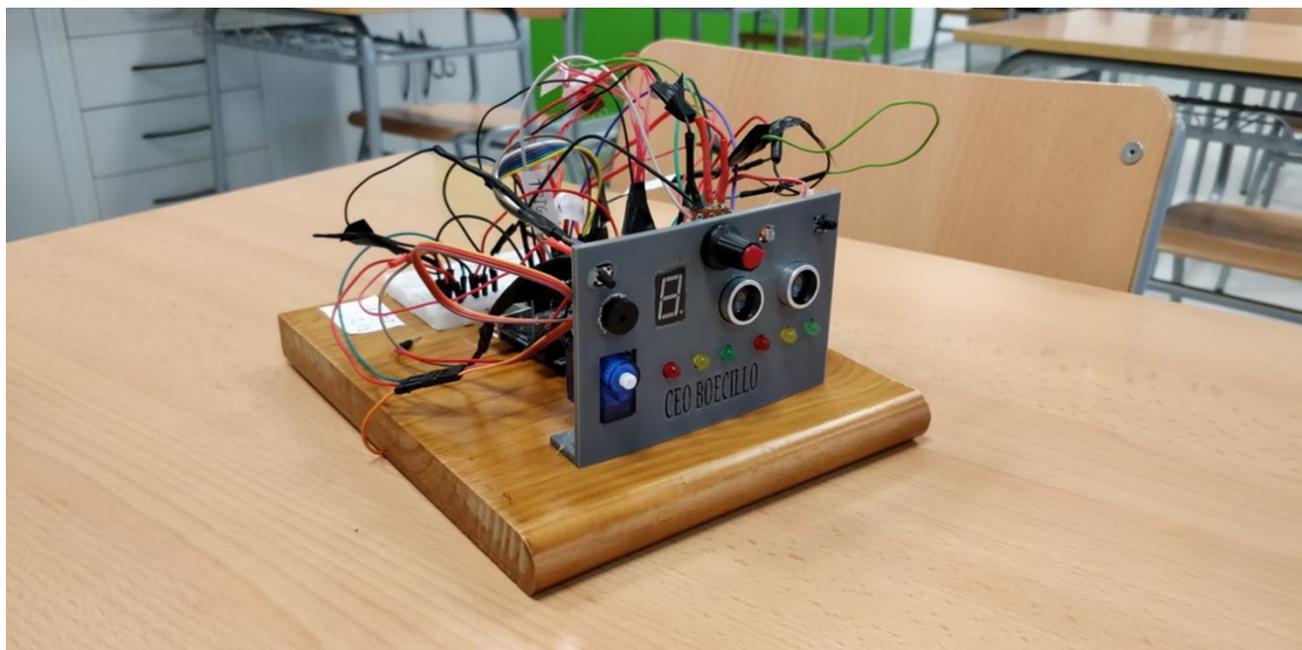
esconder variable LDR

Crear una lista

Crear un Bloque

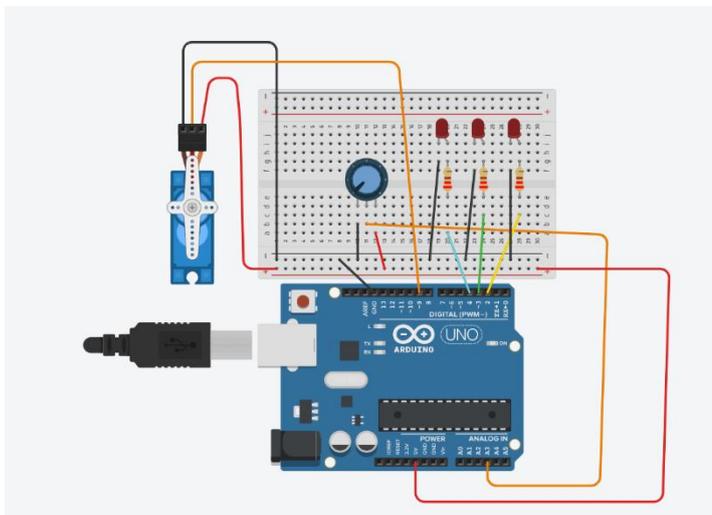
```
al presionar
por siempre
  esperar 1 segundos
  fijar IIZQ a leer pin analógico(A) 0
  fijar IDER a leer pin analógico(A) 1
  fijar LDR a leer pin analógico(A) 2
  fijar POTENCIOMETRO a leer pin analógico(A) 3
  si POTENCIOMETRO > 140 entonces
    fijar salida pin digital a HIGH
  si no
    fijar salida pin digital a LOW
    fijar salida pin digital a LOW
```

A partir de la comprensión de este contenido se pueden plantear ejercicios interactivos con servos, display, leds, etc.



### 13.- Control de un servomotor y encendido de 3 LEDs con un potenciómetro.

#### CONEXIONES



	SERVOM	POTENCIÓMETRO	LEDs		
PIN	9	A3	2	3	4

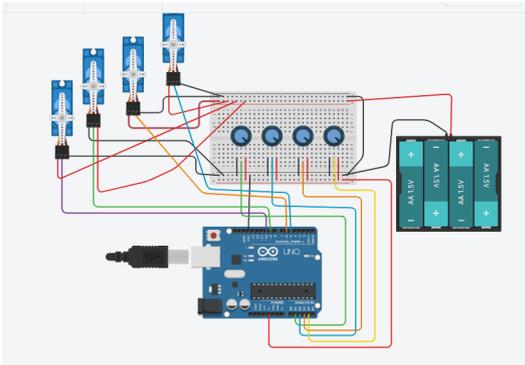
```
POTENCIOMETRO_LED_SERVO $
1 #include <Servo.h>
2 #define PINSERVO 9
3 #define POT A3
4 Servo myservo;
5 int giro = 0;
6 int valor=0;
7 #define LED2 2
8 #define LED3 3
9 #define LED4 4
10 void setup()
11 {
12     Serial.begin(9600);
13     myservo.attach(PINSERVO);
14     pinMode(POT, INPUT);
15     pinMode(LED2, OUTPUT);
16     pinMode(LED3, OUTPUT);
17     pinMode(LED4, OUTPUT);
18 }
19 void loop()
20 {
21     valor=analogRead(POT);
22     giro=map(valor, 0, 1024, 0, 180);
23
24     myservo.write(giro);
25
26     if(valor<300){
27         digitalWrite(LED2,HIGH);
28         digitalWrite(LED3,HIGH);
29         digitalWrite(LED4,HIGH);
30     }
31     if(valor>300&&valor<600){
32         digitalWrite(LED2,HIGH);
33         digitalWrite(3,LOW);
34         digitalWrite(4,LOW);
35     }
36     if(valor>600&& valor<800){
37         digitalWrite(LED2,HIGH);
38         digitalWrite(LED3,HIGH);
39         digitalWrite(LED4,LOW);
40     }
41     if(valor>800){
42         digitalWrite(LED2,HIGH);
43         digitalWrite(LED3,HIGH);
44         digitalWrite(LED4,HIGH);
45     }
46     delay(1000);
47     Serial.print("El valor del poteciometro es= ");
48     Serial.println(valor);
49     Serial.print("malor giro= ");
50     Serial.println(giro);
51 }
```

## 14.- Brazo robot. Control servos mediante potenciómetros.

The image shows a software interface for programming a robot arm. On the left, a panda character is positioned on a stage. To its left are eight sliders: POT1-4 (0-100) and GIRO1-4 (0-180). The main workspace contains a list of movement blocks: mover 10 pasos, girar 15 grados, apuntar en dirección 90, apuntar hacia, ir a x: 13 y: -1, ir a puntero del ratón, deslizar en 1 segs a x: 13 y: -1, cambiar x por 10, and fijar x a 0. On the right, the 'Programa de Arduino' block contains a 'por siempre' loop with the following code:

```
Programa de Arduino
por siempre
  fijar POT1 a leer pin analógico(A) 1
  fijar POT2 a leer pin analógico(A) 2
  fijar POT3 a leer pin analógico(A) 3
  fijar POT4 a leer pin analógico(A) 4
  fijar GIRO1 a POT1 / 4
  fijar GIRO2 a POT2 / 4
  fijar GIRO3 a POT3 / 4
  fijar GIRO4 a POT4 / 4
  fijar ángulo del pin 5 del servo a GIRO1
  fijar ángulo del pin 6 del servo a GIRO2
  fijar ángulo del pin 9 del servo a GIRO3
  fijar ángulo del pin 10 del servo a GIRO4
  esperar 0.1 segundos
```

#### 14.- BRAZO ROBOT CONTROLADO POR CUATRO POTENCIÓMETROS. – MBLOCK.

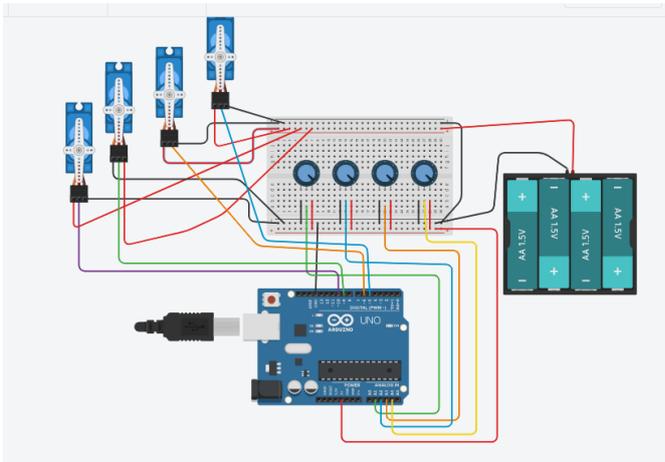


```
1 #include <Arduino.h>
2 #include <Wire.h>
3 #include <SoftwareSerial.h>
4
5 #include <Servo.h>
6
7 double angle_rad = PI/180.0;
8 double angle_deg = 180.0/PI;
9 double POT1;
10 double POT2;
11 double POT3;
12 double POT4;
13 double GIRO1;
14 double GIRO2;
15 double GIRO3;
16 double GIRO4;
17 Servo servo_5;
18 Servo servo_6;
19 Servo servo_9;
20 Servo servo_10;
21
```

```
22 void setup() {
23     pinMode(A0+1, INPUT);
24     pinMode(A0+2, INPUT);
25     pinMode(A0+3, INPUT);
26     pinMode(A0+4, INPUT);
27     servo_5.attach(5);
28     servo_6.attach(6);
29     servo_9.attach(9);
30     servo_10.attach(10);
31 }
```

```
32
33 void loop() {
34     POT1 = analogRead(A0+1);
35     POT2 = analogRead(A0+2);
36     POT3 = analogRead(A0+3);
37     POT4 = analogRead(A0+4);
38     GIRO1 = (POT1) / (5.688);
39     GIRO2 = (POT2) / (5.688);
40     GIRO3 = (POT3) / (5.688);
41     GIRO4 = (POT4) / (5.688);
42     servo_5.write(GIRO1);
43     servo_6.write(GIRO2);
44     servo_9.write(GIRO3);
45     servo_10.write(GIRO4);
46     delay(1000*0.1);
47 }
```

## 14.- BRAZO ROBOT CONTROLADO POR CUATRO POTENCIÓMETROS - ARDUINO



```

28 void setup()
29 {
30     Serial.begin(9600);
31
32     myservo1.attach(PINSERVO1);
33     myservo2.attach(PINSERVO2);
34     myservo3.attach(PINSERVO3);
35     myservo4.attach(PINSERVO4);
36
37     pinMode(POT1, INPUT);
38     pinMode(POT2, INPUT);
39     pinMode(POT3, INPUT);
40     pinMode(POT4, INPUT);
41
42 }
43 void loop()
44 {
45     valor1=analogRead(POT1);
46     giro1=map(valor1,0,1024,0,180);
47
48     valor2=analogRead(POT2);
49     giro2=map(valor2,0,1024,0,180);
50
51     valor3=analogRead(POT3);
52     giro3=map(valor3,0,1024,0,180);
53
54     valor4=analogRead(POT4);
55     giro4=map(valor4,0,1024,0,180);
56
57     myservo1.write(giro1);
58     myservo2.write(giro2);
59     myservo3.write(giro3);
60     myservo4.write(giro4);
61

```

```

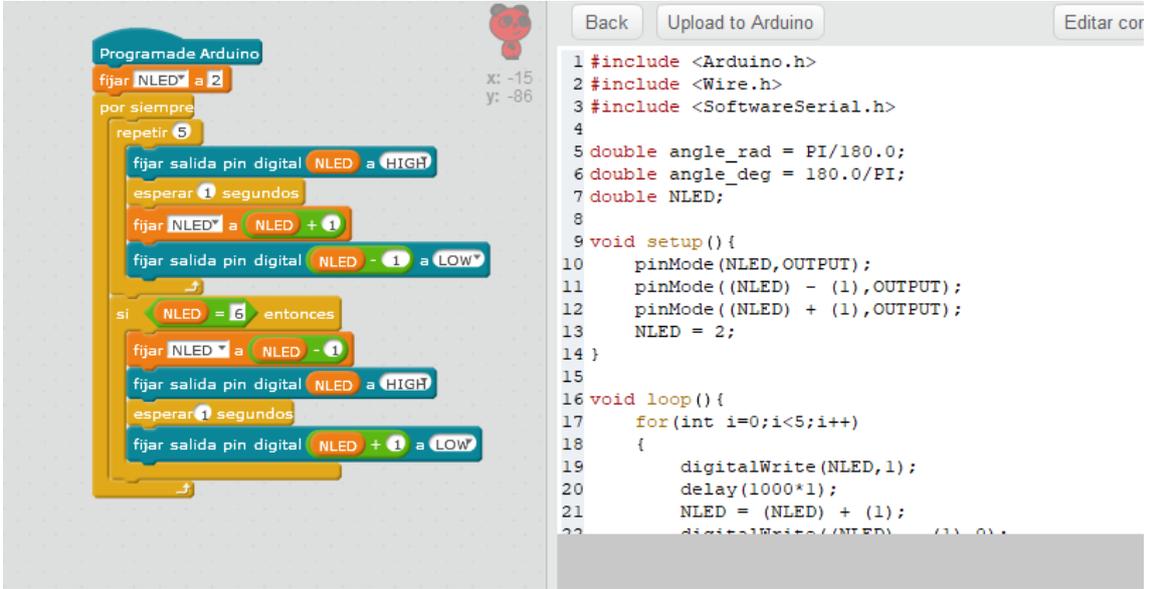
10_BRAZO_ROBOT_SAMUEL_4_SERVOS $
1 #include <Servo.h>
2
3 #define PINSERVO1 5
4 #define POT1 A1
5 Servo myservo1;
6
7 #define PINSERVO2 6
8 #define POT2 A2
9 Servo myservo2;
10
11 #define PINSERVO3 9
12 #define POT3 A3
13 Servo myservo3;
14
15 #define PINSERVO4 10
16 #define POT4 A4
17 Servo myservo4;
18
19 int giro1 = 0;
20 int valor1=0;
21 int giro2 = 0;
22 int valor2=0;
23 int giro3 = 0;
24 int valor3=0;
25 int giro4 = 0;
26 int valor4=0;
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63 delay(1000);
64 Serial.print("El valor del potenciómetro 1 es= ");
65 Serial.println(valor1);
66 Serial.print("Valor giro1= ");
67 Serial.println(giro1);
68
69 Serial.print("El valor del potenciómetro 2 es= ");
70 Serial.println(valor2);
71 Serial.print("Valor giro2= ");
72 Serial.println(giro2);
73
74 Serial.print("El valor del potenciómetro 3 es= ");
75 Serial.println(valor3);
76 Serial.print("Valor giro3= ");
77 Serial.println(giro3);
78
79 Serial.print("El valor del potenciómetro 4 es= ");
80 Serial.println(valor4);
81 Serial.print("Valor giro4= ");
82 Serial.println(giro4);
83 delay(500);
84
85 }

```

## 15.- COCHE FANTÁSTICO LEDS

Programa una secuencia de encendido y apagado de 5 LEDs, de forma que semeje las luces del coche fantástico.

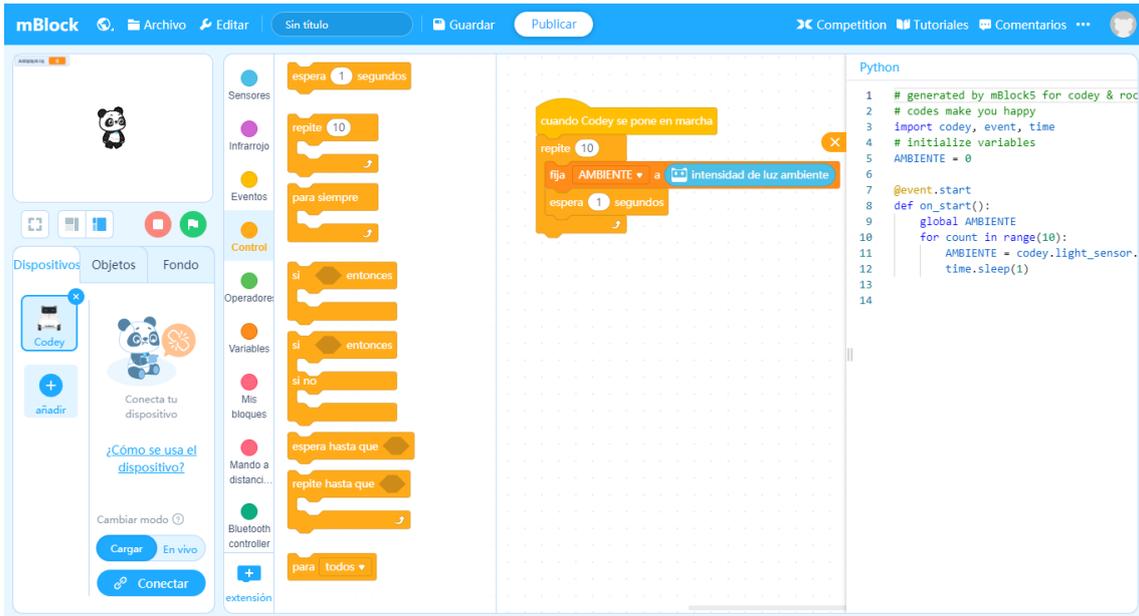
Ayuda:



```
1 #include <Arduino.h>
2 #include <Wire.h>
3 #include <SoftwareSerial.h>
4
5 double angle_rad = PI/180.0;
6 double angle_deg = 180.0/PI;
7 double NLED;
8
9 void setup() {
10   pinMode(NLED,OUTPUT);
11   pinMode((NLED) - (1),OUTPUT);
12   pinMode((NLED) + (1),OUTPUT);
13   NLED = 2;
14 }
15
16 void loop() {
17   for(int i=0;i<5;i++)
18   {
19     digitalWrite(NLED,1);
20     delay(1000*1);
21     NLED = (NLED) + (1);
22     digitalWrite(NLED, (1) - (0));
```

## 16.- MBLOCK Y PYTHON

La nueva versión de Mblock permite el control de más dispositivos.



```
1 # generated by mBlock5 for codey & roc
2 # codes make you happy
3 import codey, event, time
4 # initialize variables
5 AMBIENTE = 0
6
7 @event.start
8 def on_start():
9   global AMBIENTE
10   for count in range(10):
11     AMBIENTE = codey.light_sensor.
12     time.sleep(1)
13
14
```

# RETOS

A.- Programa un aparcamiento automático en el que intervengan: Un sensor de ultrasonidos, tres LEDes y un buzzer.

Funcionamiento: Al acercarse un objeto al sensor de ultrasonidos si está a más de 20 cm no se iluminará ningún LED ni sonará ningún sonido. Si el objeto está entre 20 y 15 cm se iluminará un LED y sonará un pitido cada 2 segundos. Si el objeto está entre 15 y 10 cm se iluminarán dos LEDes y sonará un pitido cada 1 segundos. Si el objeto está entre 10 y 5 cm se iluminarán tres LEDes y sonará un pitido cada 0,5 segundos.

B.- Programa un piano que funcione con pulsadores y un buzzer.

C.- Programa un display controlado por un potenciómetro.

D.- Crea un reto y resuélvelo.