
	Computación	Docente: Diego Quisi Peralta
	Programación Aplicada	Período Lectivo: Septiembre 2020 – Febrero 2021

		FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES	
CARRERA: COMPUTACIÓN/INGENIERÍA DE SISTEMAS		ASIGNATURA: PROGRAMACIÓN APLICADA	
NRO. PROYECTO:	1.1	TÍTULO PROYECTO: Practica de Arduino Desarrollo e implementación de un entorno de simulación Online para placas Arduino y electronica basica.	
OBJETIVO: Reforzar los conocimientos adquiridos en clase sobre Arduino.			
INSTRUCCIONES:		1. Revisar el contenido teórico y practico del tema.	
		2. Profundizar los conocimientos revisando los libros guías, los enlaces contenidos en los objetos de aprendizaje Java y la documentación disponible en fuentes académicas en línea de Arduino.	
		3. Crear una cuenta dentro de la herramienta Online https://www.tinkercad.com para simular circuitos electricos.	
		4. Revisar los siguientes videos que le ayudaran para realizar la tarea: - https://www.youtube.com/watch?v=r25dG32IWSU (Video de Electrónica Básica) - https://www.youtube.com/watch?v=hZmSG-IALAM (Video de Arduino Básico)	
		5. Revisar el ejemplo subido al AVAC del prender un led dentro del simulador TinkerCad y cargar en la herramienta para ver la simulación (PrenderApagarLed.brd). 6. Subir el informe de la practica en formato PDF y los archivos al GitPersonal. Fecha de Entrega: 31 de Enero 2021	
ACTIVIDADES POR DESARROLLAR			

1. Investigue, diseñe y desarrolle e implemente tres sistemas de simulación electrónica de Arduino dentro de la herramienta online Thincad.

DEFINICIÓN DEL PROBLEMA:

TinkerCad es un software gratuito para el diseño 3D desarrollado por Autodesk. En su apartado circuits ofrece un simulador online de Arduino bastante completo y facil de utilizar.

TinkerCad ofrece bastantes componentes para armar nuestros esquemas y circuitos, y muchos de ellos se pueden configurar (como por ejemplo las resistencias y los diodos) y manipular en tiempo real (potenciómetros, botones, etc.).

La programación en TinkerCad se puede realizar en modo código y en modo bloques, y también tenemos disponible una pantalla dividida donde vemos los dos modos simultáneamente.

Al compilar el código, si hay algún error es marcado por el depurador. Una vez que tenemos el código arduino listo y la simulación funciona sin errores podemos descargar el archivo .ino para subirlo a nuestro arduino. El software nos provee de un monitor serie con plotter serial incluido similar al IDE de Arduino. TinkerCad es una plataforma ideal para quienes están aprendiendo Arduino y programación. Es muy intuitiva y de fácil manejo, gratuita y online.

En base a ello se propone resolver tres problemas electrónicos:


1. Generar un autofantastico que se prenda y se apague desde un pulsante.
2. Generar una lampara de ciudad, es decir que se prenda cuando es noche y se apague cuando ya exista luz para esto deben utilizar un LDR y un LED.
3. Finalmente, controlar un servomotor con un potenciómetro el grado de giro.

RESULTADO(S) OBTENIDO(S):

- Interpreta de forma correcta la programación en Arduino.
- - Identifica correctamente qué herramientas de electronicas se pueden aplicar.

CONCLUSIONES:

- Los estudiantes implementan soluciones de hardware en sistemas.
- Los estudiantes estan en la capacidad de implementar sistemas electronicos en Arduino.

	Computación	Docente: Diego Quisi Peralta
	Programacion Aplicada	Período Lectivo: Septiembre 2020 – Febrero 2021

RECOMENDACIONES:

- Revisar la información proporcionada por el docente previo a la práctica.
- Haber asistido a las sesiones de clase.
- **Consultar con el docente las dudas que puedan surgir al momento de realizar la práctica.**

BIBLIOGRAFIA:

[1]: <https://www.ups.edu.ec/evento?calendarBookingId=98892>

Docente / Técnico Docente: Ing. Diego Quisi Peralta Msc.

Firma: _____

CARRERA: Ing computation

ASIGNATURA: Programacion aplicada.

NRO. PRÁCTICA:

6

TÍTULO PRÁCTICA: Arduino




OBJETIVO ALCANZADO:

- Desarrollo de códigos en una plataforma virtual que su funcionamiento es similar en un aspecto real
- Programación en Arduino en implantación de circuitos.
- Combinación de componentes electrónicos con códigos

ACTIVIDADES DESARROLLADAS

1. Generar un autofantástico que se prenda y se apague desde un pulsante

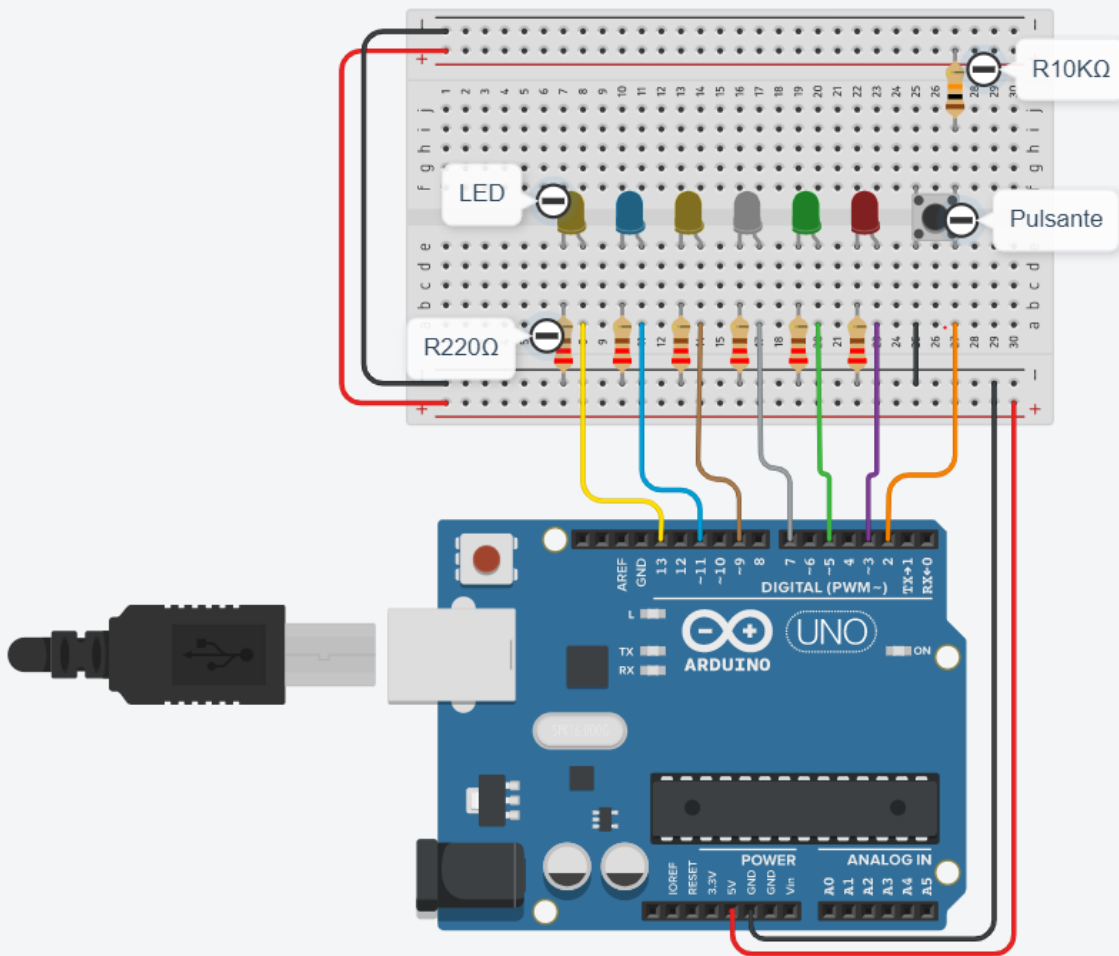
Para el desarrollo del problema planteado se necesita una lista detallada a continuación además de una placa de prueba.


Brilliant Leelo-Inari
Se han guardado todos los cambios.



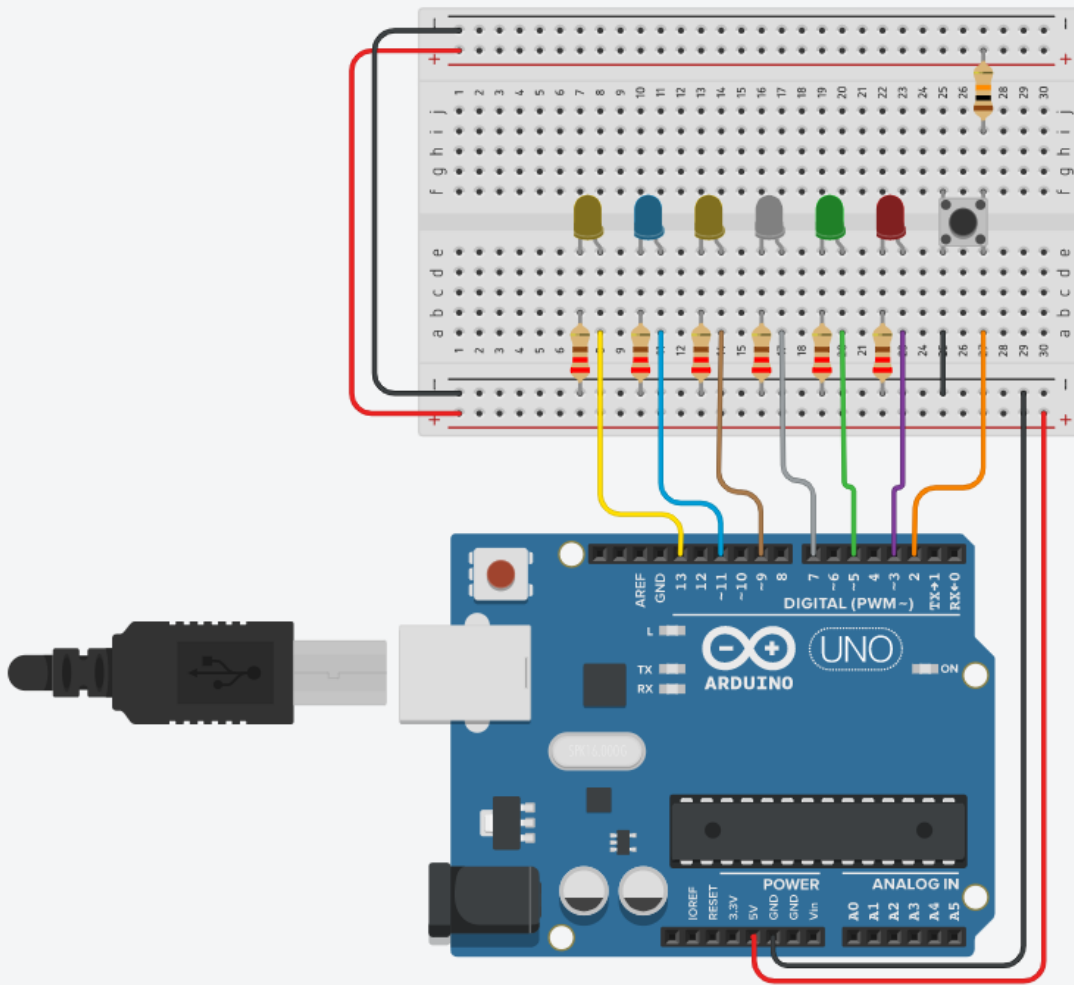
Lista de componentes Descargar CSV

Nombre	Cantidad	Componente
U1	1	Arduino Uno R3
D1 D3	2	Amarillo LED
D2	1	Azul LED
D4	1	Blanco LED
D5	1	Verde LED
D6	1	Rojo LED
R1 R2 R3 R4 R5 R6	6	220 Ω Resistencia
S1	1	Pulsador
R7	1	10 k Ω Resistencia

A continuación, se ilustrar el nombre de cada componente en el circuito.



Ahora procedemos a armar el circuito con los componentes ya mencionados con anterioridad pero para ello es necesario que ya debe de estar conectado los 5V a (+) y GND o tierra a (-) y claro también las entradas de positivo con positivo y negativo con negativo que en este caso rigiéndonos a las normas de los circuitos el positivo será de color rojo y el negativo de color negro como se ilustra a continuación.



después de armar el circuito se procede ala codificación.

Se han guardado todos los cambios.

Código ▶ Iniciar simulación Exportar Compartir

1 (Arduino Uno R3)

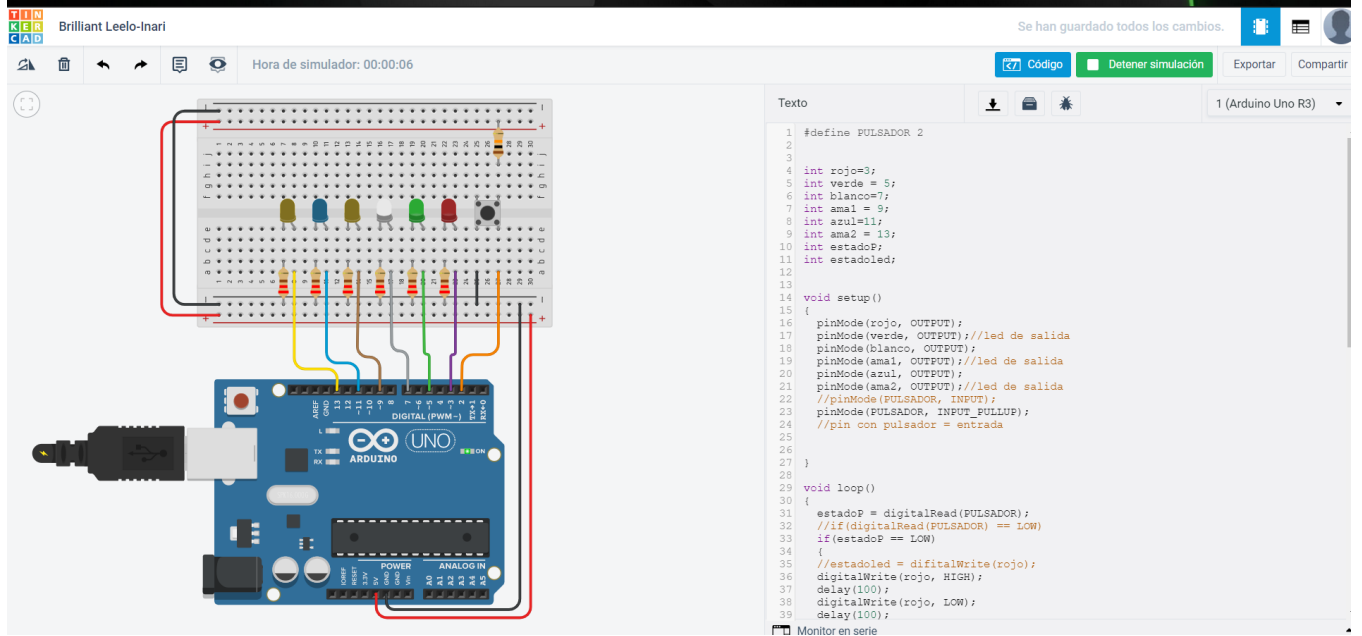
```

1 #define PULSADOR 2
2
3
4 int rojo=3;
5 int verde = 5;
6 int blanco=7;
7 int ama1 = 9;
8 int azul=11;
9 int ama2 = 13;
10 int estadoP;
11 int estadoled;
12
13
14
15
16 void setup()
17 {
18   pinMode(rojo, OUTPUT);
19   pinMode(verde, OUTPUT); //led de salida
20   pinMode(blanco, OUTPUT);
21   pinMode(ama1, OUTPUT); //led de salida
22   pinMode(ama2, OUTPUT); //led de salida
23   //pinMode(PULSADOR, INPUT);
24   pinMode(PULSADOR, INPUT_PULLUP);
25   //pin con pulsador = entrada
26 }
27
28
29 void loop()
30 {
31   estadoP = digitalRead(PULSADOR);
32   //if(digitalRead(PULSADOR) == LOW)
33   if(estadoP == LOW)
34   {
35     //estadoled = digitalWrite(rojo);
36     digitalWrite(rojo, HIGH);
37     delay(100);
38     digitalWrite(rojo, LOW);
39     delay(100);
40   }
41 }

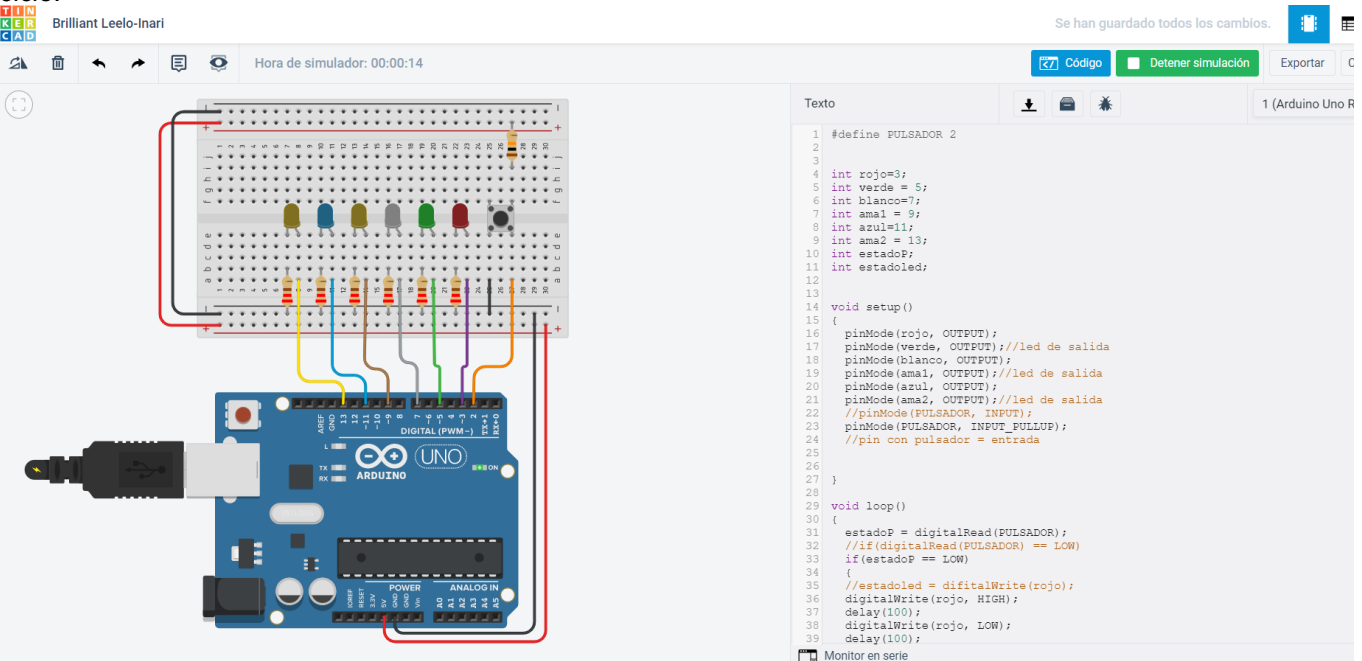
```

Monitor en serie

Después podemos iniciar la simulación



Se presiona el pulsante y así se puede apreciar que el led de color blanco está encendido debido a que en la codificación se dice que cada led deberá encenderse cada determinado tiempo, deberá encenderse y posteriormente se apagará y así un ciclo finito hasta que se presione otra vez el pulsante para que termine dicho ciclo.



Codificación.
#define PULSADOR 2

int rojo=3;
int verde = 5;
int blanco=7;
int ama1 = 9;

```
int azul=11;

int ama2 = 13;

int estadoP;

int estadoled;


void setup()
{
    pinMode(rojo, OUTPUT);
    pinMode(verde, OUTPUT);//led de salida
    pinMode( blanco, OUTPUT);
    pinMode(ama1, OUTPUT);//led de salida
    pinMode(azul, OUTPUT);
    pinMode(ama2, OUTPUT);//led de salida
    //pinMode(PULSADOR, INPUT);
    pinMode(PULSADOR, INPUT_PULLUP);
    //pin con pulsador = entrada

}

void loop()
{
    estadoP = digitalRead(PULSADOR);
    //if(digitalRead(PULSADOR) == LOW)
    if(estadoP == LOW)
    {
        //estadoled = digitalWrite(rojo);
        digitalWrite(rojo, HIGH);
        delay(100);
        digitalWrite(rojo, LOW);
        delay(100);
    }
}
```



```
//estadoled = digitalWrite(verde);  
digitalWrite(verde, HIGH);  
delay(100);  
digitalWrite(verde, LOW);  
delay(100);  
  
//estadoled = digitalWrite(amarillo);  
digitalWrite(amarillo, HIGH);  
delay(100);  
digitalWrite(amarillo, LOW);  
delay(100);  
  
//estadoled = digitalWrite(azul);  
digitalWrite(azul, HIGH);  
delay(100);  
digitalWrite(azul, LOW);  
delay(100);  
  
//estadoled = digitalWrite(rojo);  
digitalWrite(rojo, HIGH);  
delay(100);  
digitalWrite(rojo, LOW);  
delay(100);  
  
}  
//if(digitalRead(PULSADOR) == HIGH)
```

```

if(estadoP == HIGH)
{
    digitalWrite(verde, LOW);
    digitalWrite(rojo, LOW);
    digitalWrite(blanco, LOW);
    digitalWrite(azul, LOW);
    digitalWrite(ama1, LOW);
    digitalWrite(ama2, LOW);

}

}

```

2. Generar una lampara de ciudad, es decir que se prenda cuando es noche y se apague cuando ya exista luz para esto deben utilizar un LDR y un LED.

Para el desarrollo del siguiente circuito procedemos a ver las lista de materiales a usar en este caso los materiales usados son una placa de pruebas y un Arduino además de componentes detallados a continuación.


Lampara

Se han guardado todos los cambios.

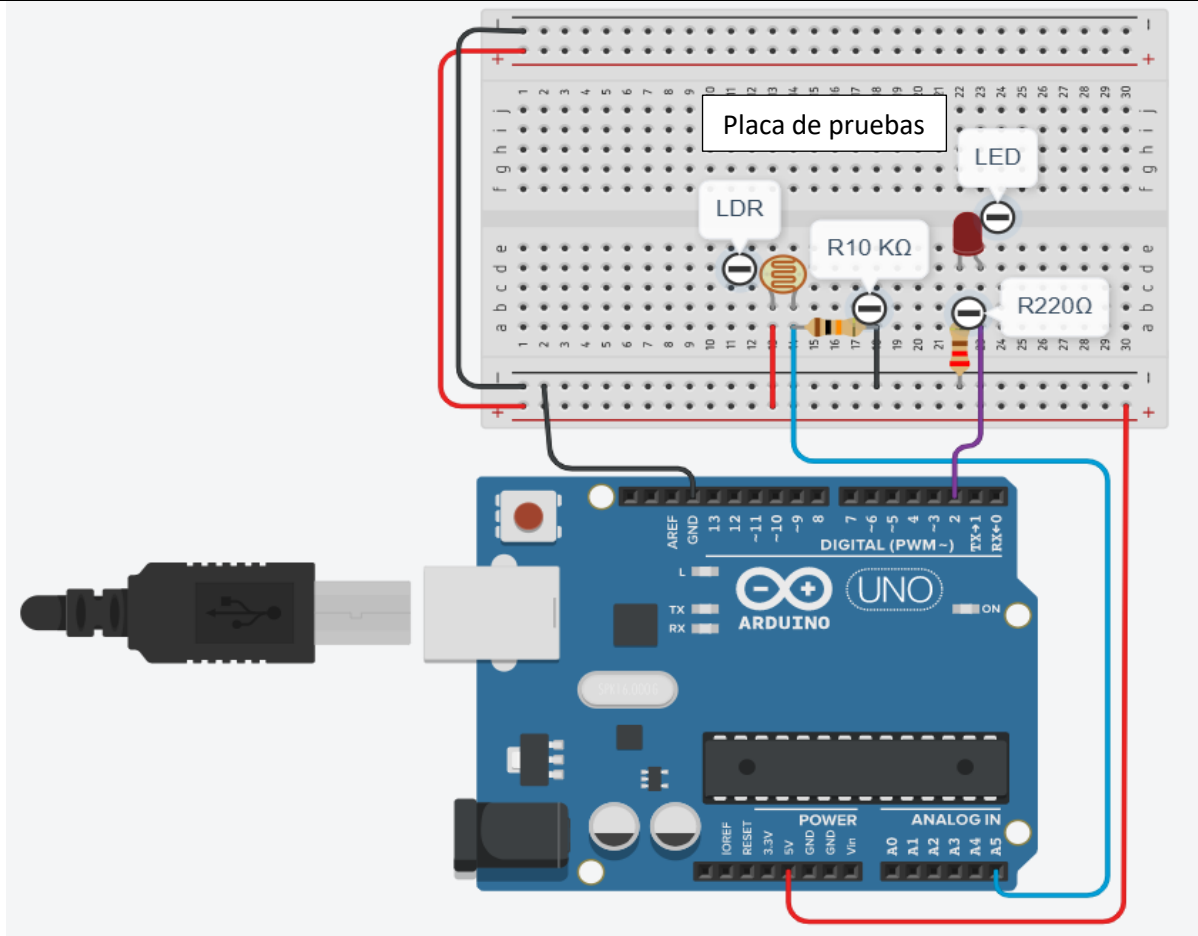


Lista de componentes

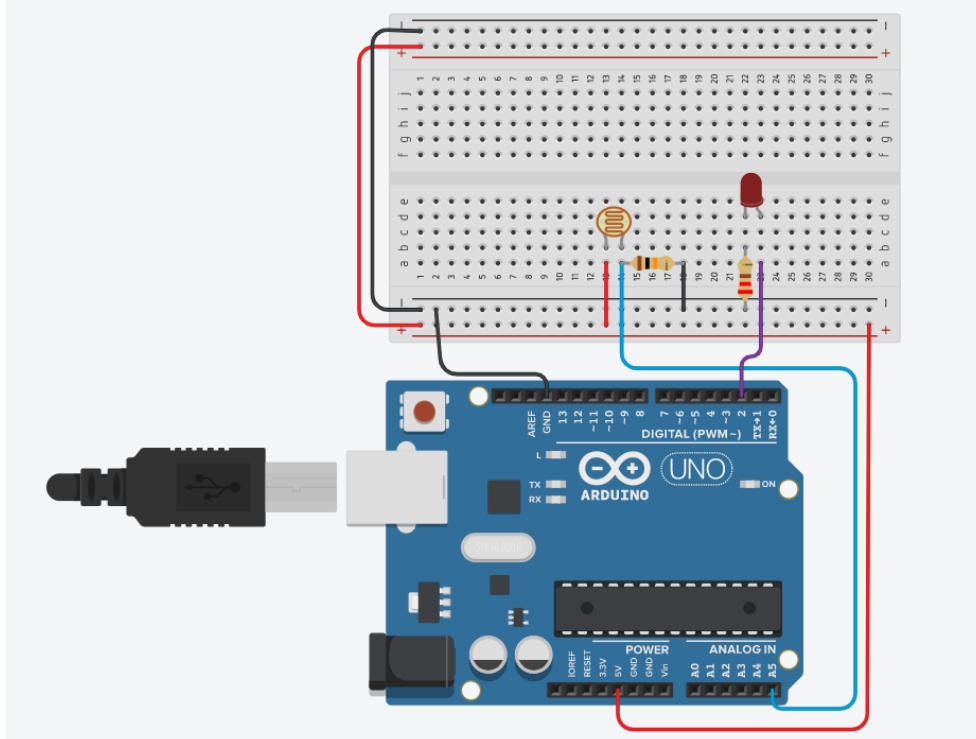

Descargar CSV

Nombre	Cantidad	Componente
U1	1	Arduino Uno R3
R1	1	Fotorresistencia
D1	1	Rojo LED
R2	1	10 kΩ Resistencia
R3	1	220 Ω Resistencia

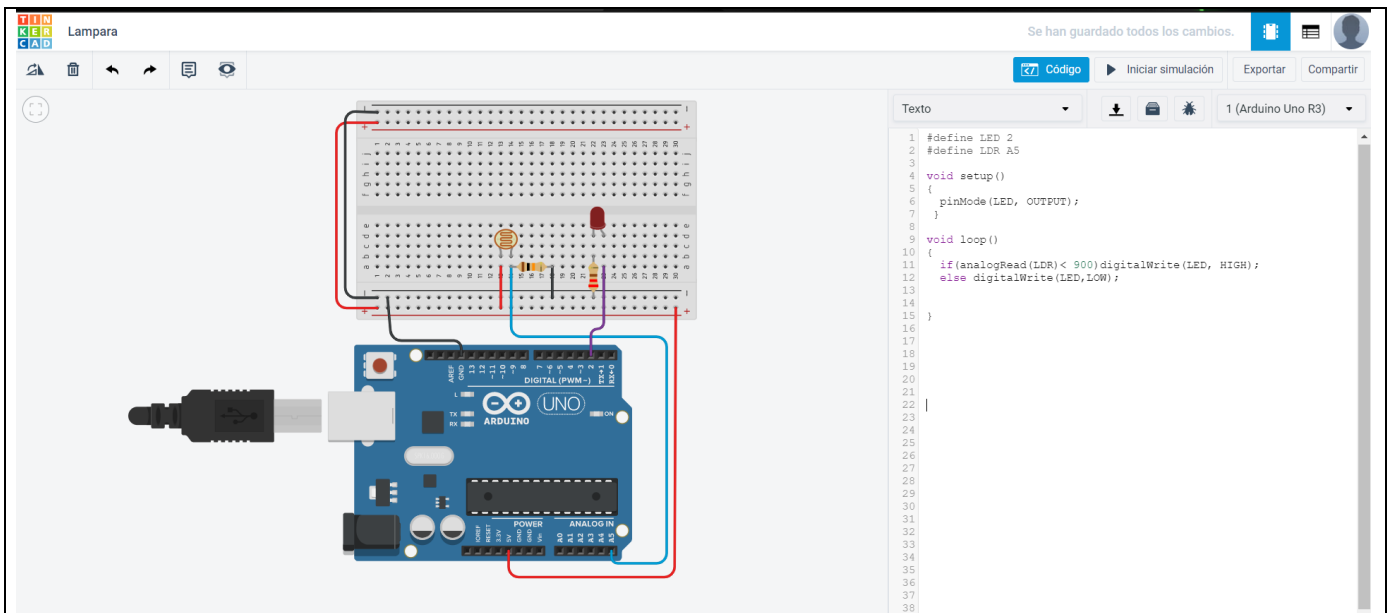
A continuación se procede a ilustrar el nombre de cada componente en el circuito.



Para el siguiente proceso se procedió a armar el circuito con los componentes ya antes descritos.

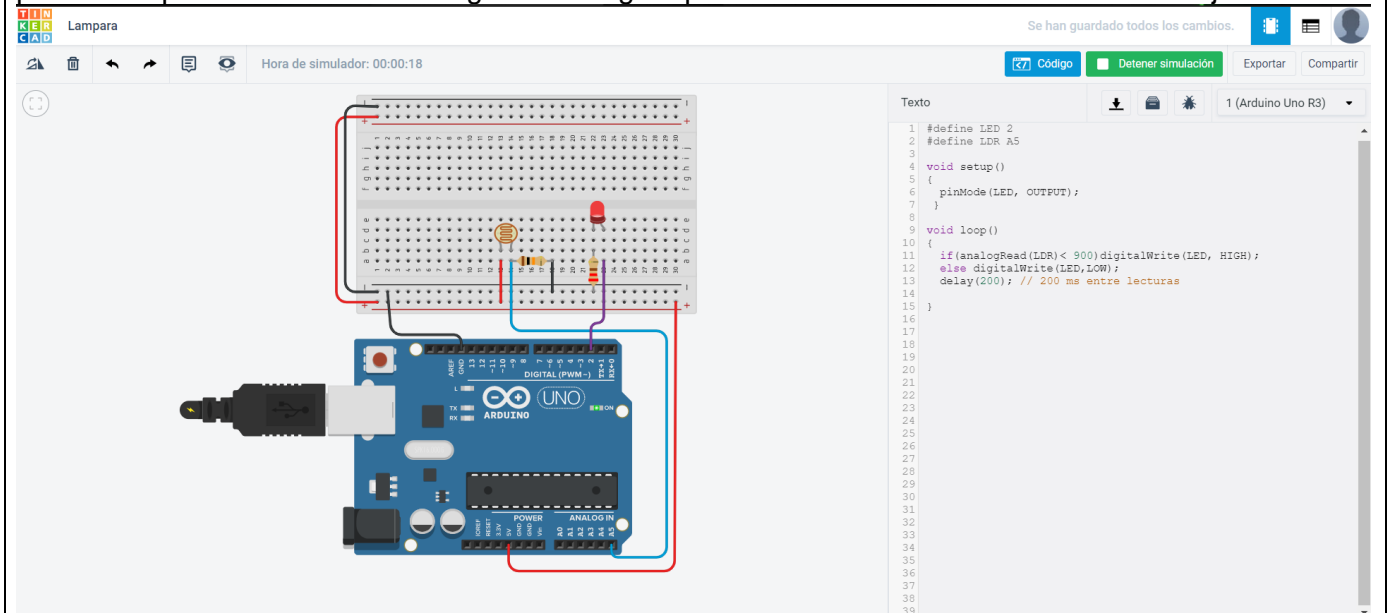


Ya terminado el circuito se procede a la codificación, para ello definimos los puertos a usar ya sean analógicos y digitales en este caso se usa el puerto analógico A5 y el digital 2

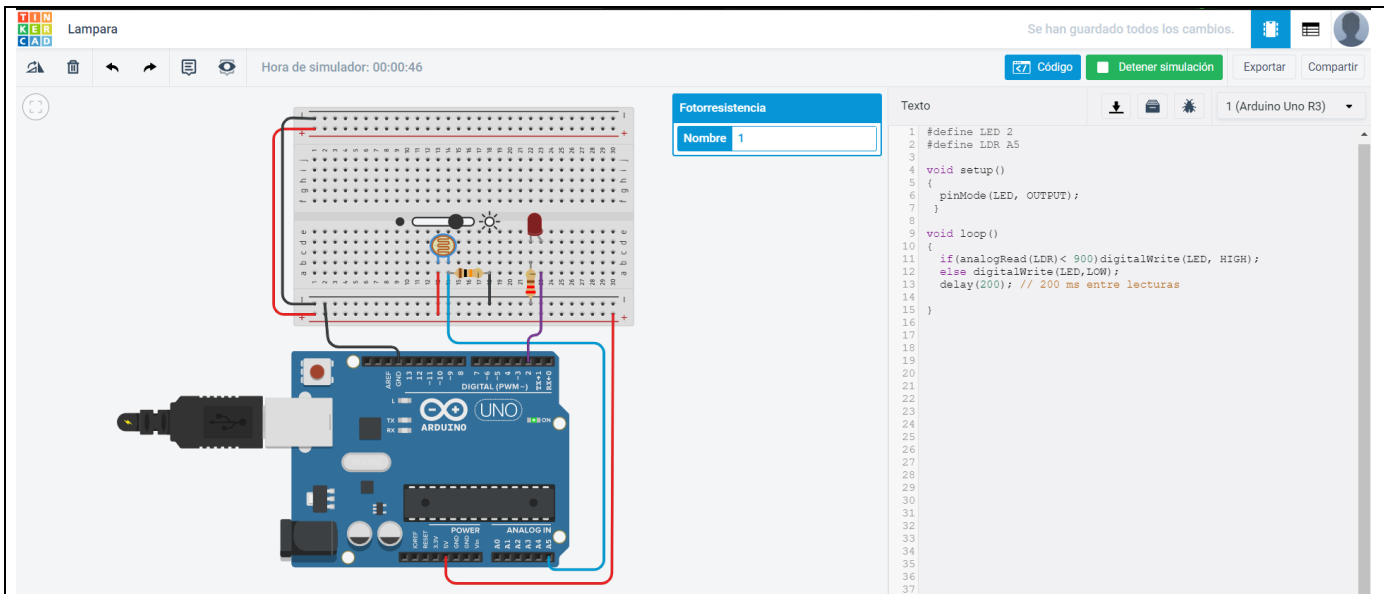


Iniciar simulación.

En el momento que se inicia la simulación consideramos que ya estamos entre las horas 19:00 a 6:00 por eso se puede observar en la siguiente imagen que tenemos encendido el led de color rojo.



Pero de lo contrario si alteramos el LDR de tal manera de bajar la luminosidad que esta sobre el, es decir entre las horas 6:00 a 19:00, obteniendo el siguiente resultado.



Codificación y explicación del código.

//definimos el led en el puerto 2 digital

```
#define LED 2
```

//definimos el led A5 en el puerto analogico

```
#define LDR A5
```

/*el codigo de setup se ejecuta solo una vez al encenderse el dispositivo o iniciar la simulacion y empezar el programa

```
*/
```

```
void setup()
```

```
{
```

```
  pinMode(LED, OUTPUT); //led de salida
```

```
}
```

/*El codigo del loop se ejecuta infinitamente hasta que se apaga la tarjeta o se detiene la simulacion

```
*/
```

```
void loop()
```

```
{
```

```
  /* en el siguiente codigo primero leemos el valor
```

```
  que esta dando el LDR que debe oscilar entre los 0 y 1023
```

```
  que es lo que se obtiene en una entrada analogica y
```

de cumplirlo decimos que el led se encienda y de caso

contrario lo apagamos

*/

```
if(analogRead(LDR)< 900)digitalWrite(LED, HIGH);
```


```
else digitalWrite(LED,LOW);
```

```
delay(200); // 200 ms entre lecturas
```



```
}
```

3. Finalmente, controlar un servomotor con un potenciómetro el grado de giro.

El desarrollo del problema planteado se necesitó de una placa de pruebas y un Arduino además de componentes detallados en la siguiente tabla.

 Servomotor

Se han guardado todos los cambios.

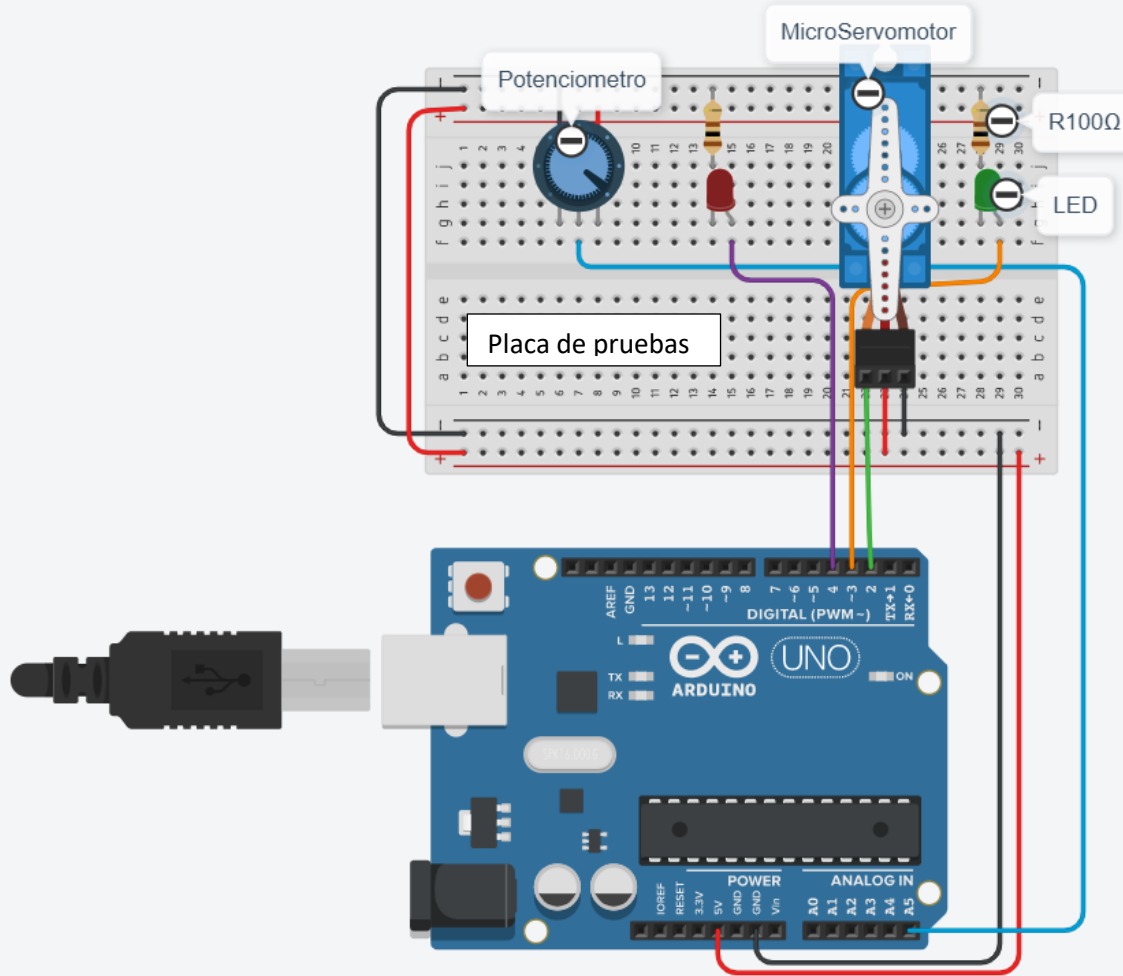


Lista de componentes

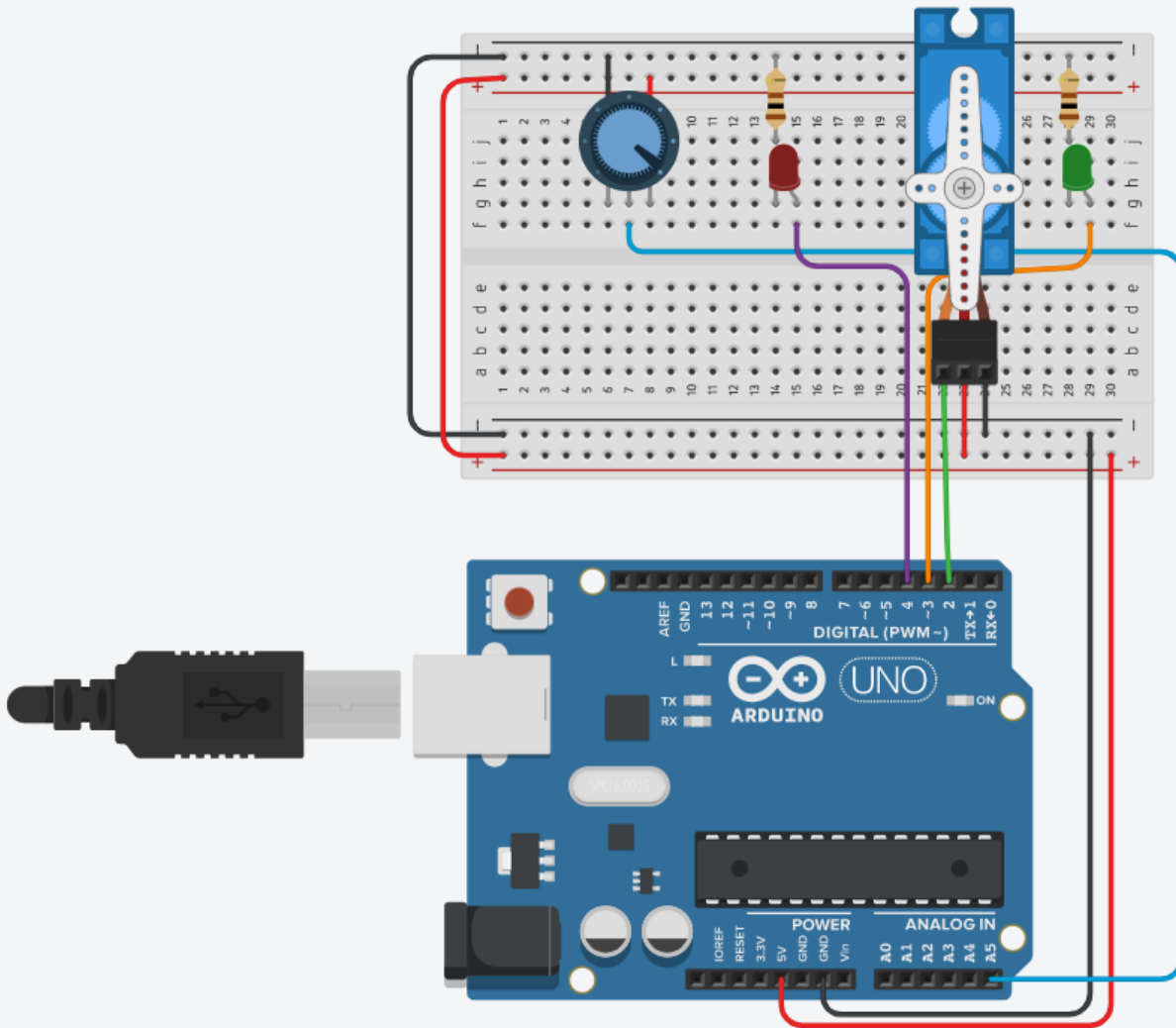
Descargar CSV

Nombre	Cantidad	Componente
U1	1	Arduino Uno R3
Rpot1	1	100 kΩ Potenciómetro
SERV01	1	Posicional Microservomotor
D1	1	Rojo LED
D2	1	Verde LED
R1 R2	2	100 Ω Resistencia

A continuación se procede a ilustrar el nombre de cada componente en el circuito.



Ahora procedemos a armar el circuito con los componentes ya mencionados con anterioridad pero para ello es necesario que ya debe de estar conectado los 5V a (+) y GND o tierra a (-) y claro también las entradas de positivo con positivo y negativo con negativo que en este caso rigiéndonos a las normas de los circuitos el positivo será de color rojo y el negativo de color negro como se ilustra a continuación.



Después de haber finalizado la conexión de todos los componentes se procede a ir a la codificación

Servomotor

Se han guardado todos los cambios.

Código

Iniciar simulación

Exportar

Compartir

1 (Arduino Uno R3)

```

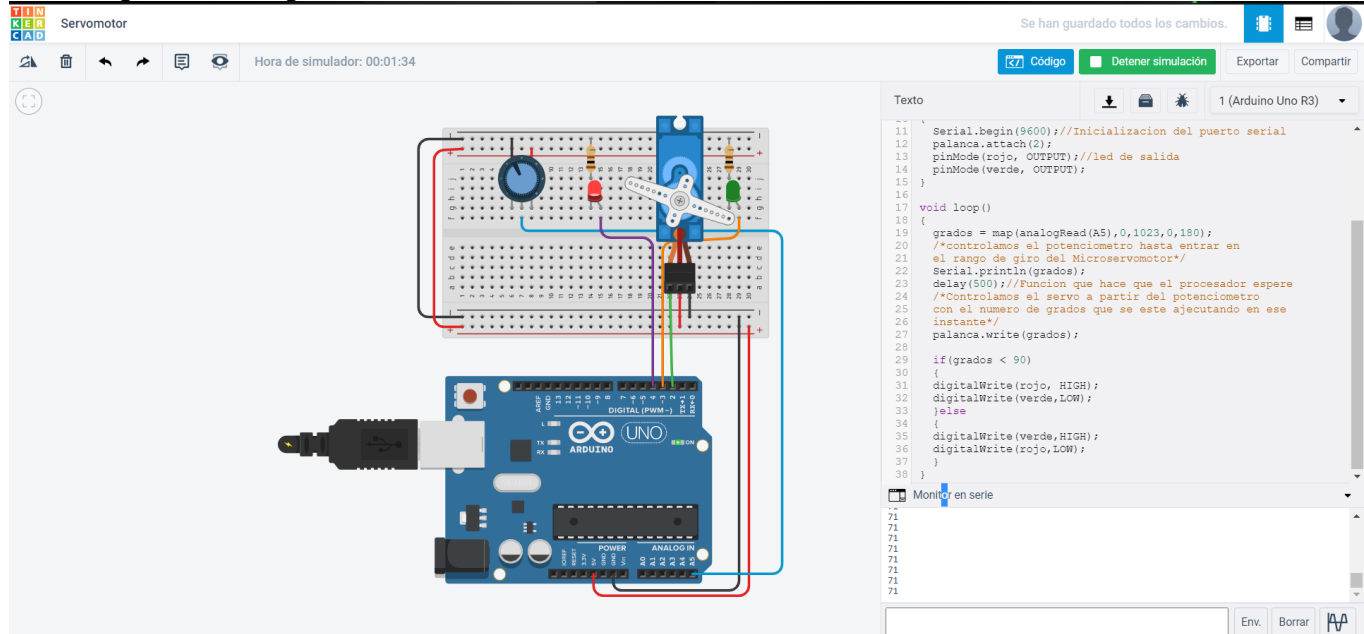
1 #include <Servo.h>
2 // incluimos la libreria del servo
3 Servo palanca;
4
5 int grados;//va a ser la lectura analoga A5
6 int rojo=4;
7 int verde = 3;
8
9 void setup()
10 {
11   Serial.begin(9600);//Inicializacion del puerto serial
12   palanca.attach(2);
13   pinMode(rojo, OUTPUT);//led de salida
14   pinMode(verde, OUTPUT);
15 }
16
17 void loop()
18 {
19   grados = map(analogRead(A5),0,1023,0,180);
20   /*controlamos el potenciómetro hasta entrar en
21   el rango de giro del Microservomotor*/
22   Serial.println(grados);
23   delay(500);//funcion que hace que el procesador espere
24   /*Controlamos el servo a partir del potenciómetro
25   con el numero de grados que se este ejecutando en ese
26   instante*/
27   palanca.write(grados);
28
29   if(grados < 90)
30   {
31     digitalWrite(rojo, HIGH);
32     digitalWrite(verde,LOW);
33   }else
34   {
35     digitalWrite(verde,HIGH);
36     digitalWrite(rojo,LOW);
37   }
38 }

```

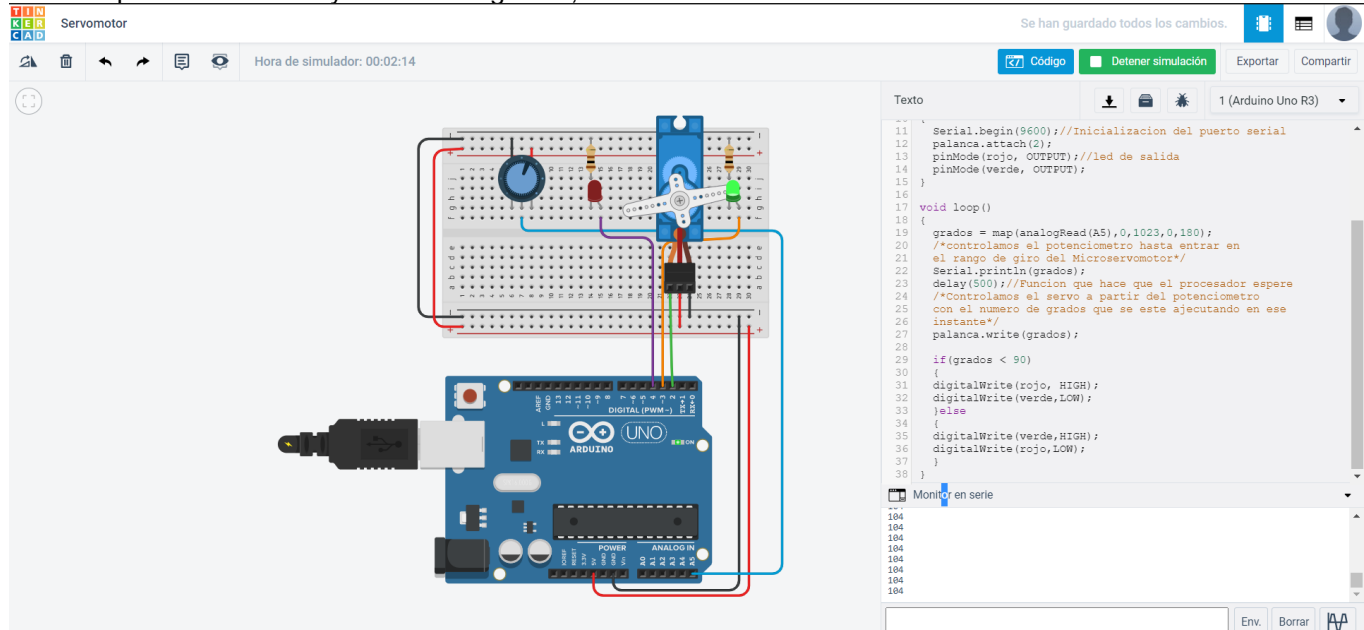
Monitor en serie

Iniciar simulación.

Procedemos a iniciar la simulación considerando que para que sea más evidente el control del servomotor con el potenciómetro se agrego dos leds, de esta manera se puede observar que cuando el potenciómetro toma valores menores a los 90 y el servomotor gira hacia el led rojo el mismo se encenderá y de caso contrario si el potenciómetro toma valores mayores a los 90 grados el servo girara al lado contrario indicando el led de color verde este procederá a encenderse como se muestra en las siguientes imágenes.



Valor de potenciómetro mayor a los 90 grados, enciende el led verde.



Codificación.

```
#include <Servo.h>
```

```
// incluimos la libreria del servo
```

```
Servo palanca;
```

```
int grados; //va a ser la lectura analogica A5
```

```


int rojo=4;

int verde = 3;


void setup()
{
  Serial.begin(9600); //Inicializacion del puerto serial
  palanca.attach(2);
  pinMode(rojo, OUTPUT); //led de salida
  pinMode(verde, OUTPUT);
}

void loop()
{
  grados = map(analogRead(A5),0,1023,0,180);
  /*controlamos el potenciómetro hasta entrar en
  el rango de giro del Microservomotor*/
  Serial.println(grados);
  delay(500); //Funcion que hace que el procesador espere
  /*Controlamos el servo a partir del potenciómetro
  con el numero de grados que se este ejecutando en ese
  instante*/
  palanca.write(grados);
  /*si el grado de giro del potenciómetro es mayor a los 90 grados
  y el servomotor gira hacia la posicion del led de color rojo
  este se encendera y de caso contrario si el numero de grados es
  mayor a los 90 se encendera el led de color verde apagando el led
  de color rojo
  */
  if(grados < 90)
  {
    digitalWrite(rojo, HIGH);
    digitalWrite(verde, LOW);
  }
}

```

	Computación	Docente: Diego Quisi Peralta
	Programación Aplicada	Período Lectivo: Septiembre 2020 – Febrero 2021

```

}else
{
digitalWrite(verde,HIGH);
digitalWrite(rojo,LOW);
}
}

```


CONCLUSIONES:

- Los estudiantes implementan soluciones de hardware en sistemas.
- Los estudiantes están en la capacidad de implementar sistemas electrónicos en Arduino.

RECOMENDACIONES:

- Revisar la información proporcionada por el docente previo a la práctica.
- Haber asistido a las sesiones de clase.
- **Consultar con el docente las dudas que puedan surgir al momento de realizar la práctica.**

Nombre de estudiante: John Farez



Firma de estudiante: