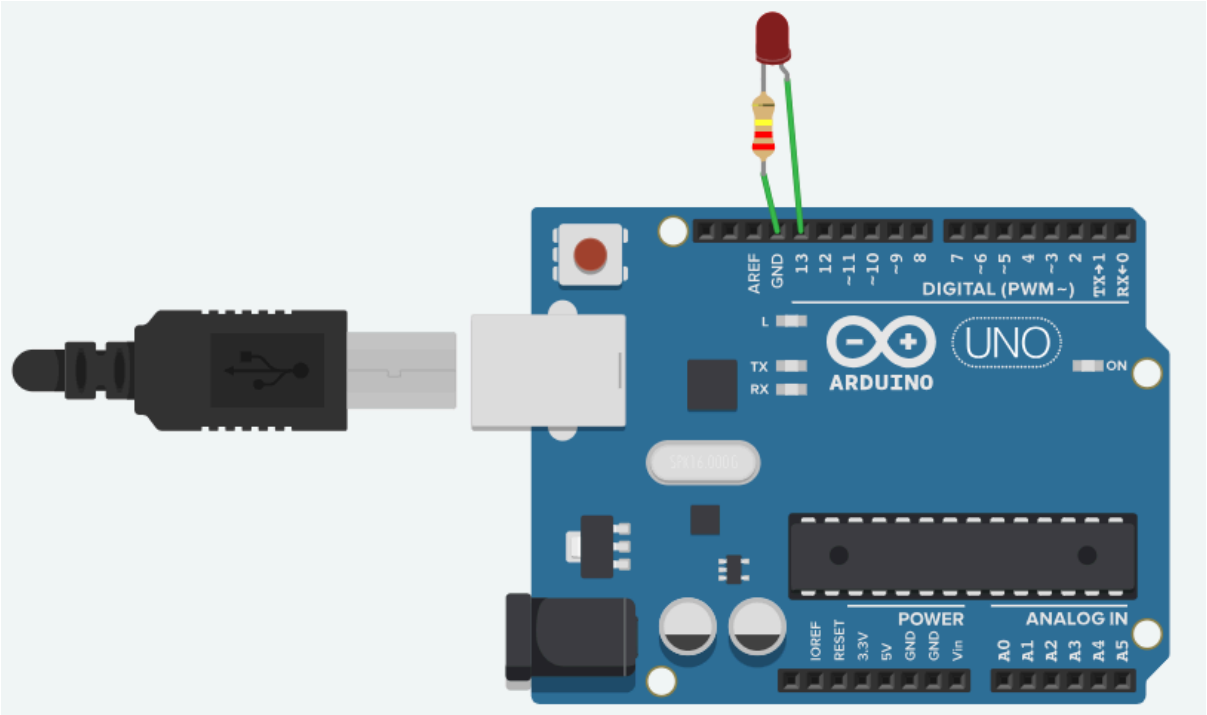
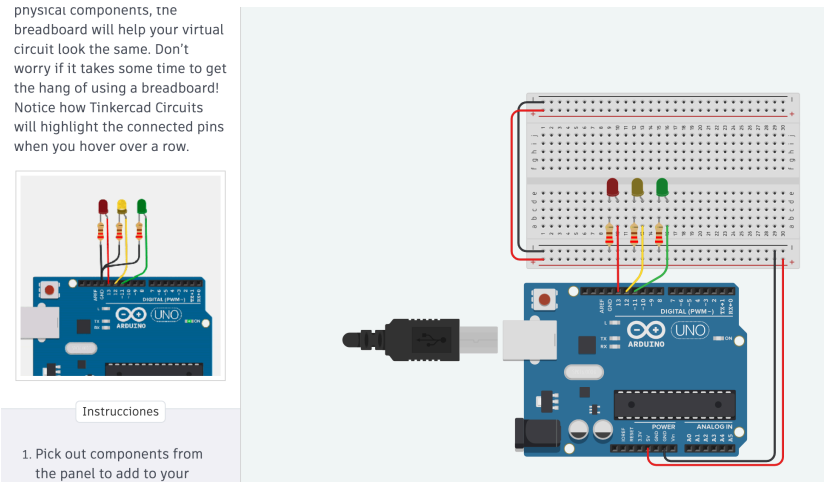
Blink an LED with Digital Output→ Dice básicamente que si tienes una placa arduino, un led y una resistencia que tienes que unir una de las patas del led (ánodo o cátodo) a la resistencia, luego el ánodo (la +, q es más larga) va conectada al 13 (corriente) , mientras que el cátodo (-) va conectada a tierra.



Multiple LED's & Motherboards → Los dos circuitos de abajo son equivalentes. Algo importante cada led tiene que tener una resistencia. El circuito de la derecha parece un poco más complicado pero lo explico. El rojo olvidalo no sirve para nada, ahora imagina que de los pins (el 13 por ej) sale la electricidad, luego entra por por el ánodo y pasa al cátodo (encendiendo el led), pasa por la resistencia y la resistencia está conectada al cable negro que es la tierra.

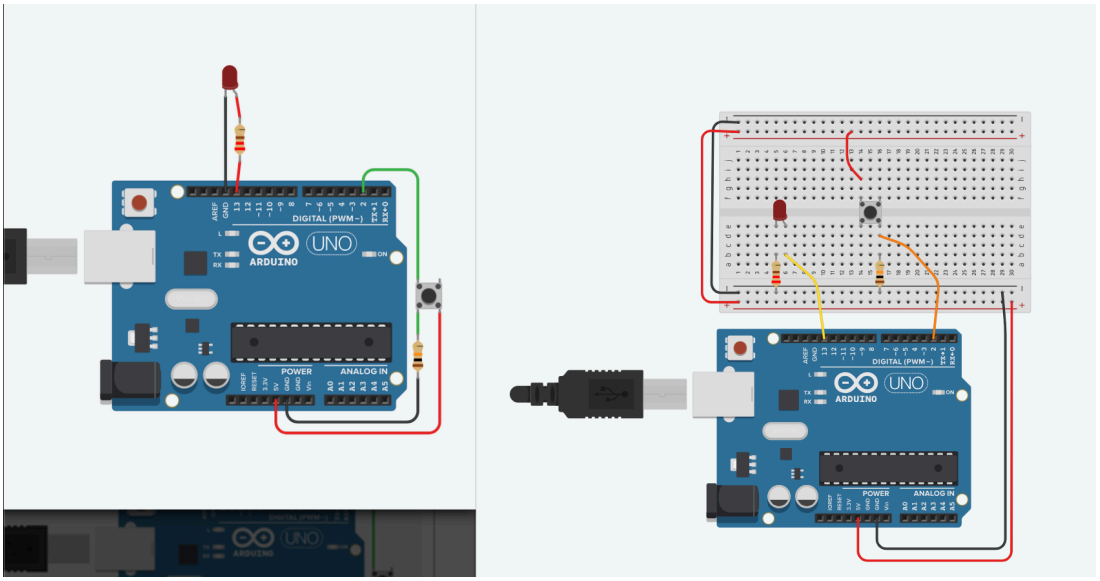


Ahora, para programar en C++ esto (lo que hace el programa es que se enciende un led, espera, se apaga, espera, se enciende otro y así sucesivamente):

```
// C++ code
//
int animationSpeed = 0;

void setup()
{
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(11, OUTPUT);
}

void loop()
{
  animationSpeed = 400;
  digitalWrite(LED_BUILTIN, HIGH);
  delay(animationSpeed); // Wait for animationSpeed millisec
  digitalWrite(LED_BUILTIN, LOW);
  delay(animationSpeed); // Wait for animationSpeed millisec
  digitalWrite(12, HIGH);
  delay(animationSpeed); // Wait for animationSpeed millisec
  digitalWrite(12, LOW);
  delay(animationSpeed); // Wait for animationSpeed millisec
  digitalWrite(11, HIGH);
  delay(animationSpeed); // Wait for animationSpeed millisec
  digitalWrite(11, LOW);
  delay(animationSpeed); // Wait for animationSpeed millisec
}
```

Primero se define la variable animationSpeed como int, pq es un entero. Luego tenemos el void setup(), aquí tenemos que poner los OUTPUTS, por eso se usa la función pinMode() y aquí pones primero el pin y luego el output que queremos dar, como es un led pues ponemos HIGH (on) o LOW (off). Observa que para usar la variable animationSpeed se usa la función delay().

Digital Input / Analog Input→ PUSHBUTTON (digital input) →Lo de siempre los dos circuitos de abajo son equivalentes, ahora expliquemos lo que pasa en el de la derecha. Primero mira el cable naranja (que sale del pin 2), está conectado a una resistencia muy grande y cuando el botón no está pulsado debe pasa por ahí (no tiene de otra), por lo que va directo a tierra. Cuando pulsamos el botón le estamos dando la opción de pasar por una resistencia más pequeña (la del led) y por lo tanto va por ahí, pilla los 5 v del arduino cruzando el pulsador y el cable rojo pasa por el otro cable rojo (el que está al lado del negro), vuelve al arduino y va al 13, que está unido al led y luego pasa por la resistencia que va otra vez a tierra cerrando el circuito.



Ahora miramos el código, primero inicializamos la variable buttonState como 0 (también es int, dado que será un número), luego vemos el void setup() y utilizamos pinMode para dar info o recibir info de los pins, del pin 2 leemos info (si se ha pulsado o no el botón, viendo si hay 5V o no),por eso es un input, mientras que el pin 13 (o LED_BUILTIN) se encarga de activar el led por eso es un output.
En el loop, primero se lee con digitalRead(pin) el estado del pin (5V ES HIGH, GROUND ES LOW) y lo almacena en buttonState. Lo demás es un if que compara (con ==) si es High o Low y enciende o apaga el led en consecuencia

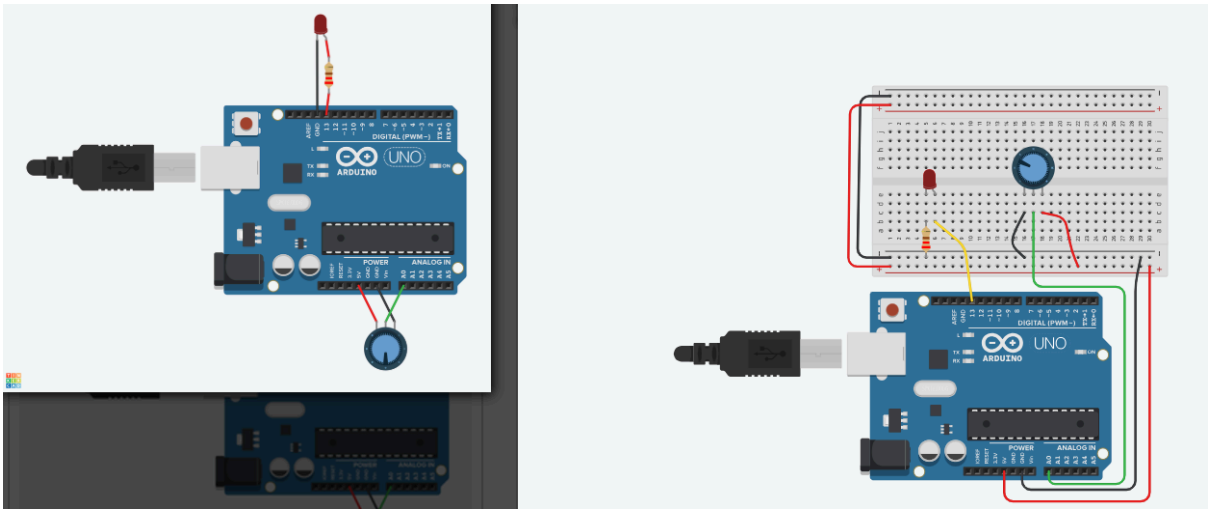
```
// C++ code
//
int buttonState = 0;

void setup()
{
  pinMode(2, INPUT);
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop()
{
  // read the state of the pushbu
  buttonState = digitalRead(2);
  // check if pushbutton is pressed. if it is, the
  // button state is HIGH
  if (buttonState == HIGH) {
    digitalWrite(LED_BUILTIN, HIGH);
  } else {
    digitalWrite(LED_BUILTIN, LOW);
  }
  delay(10); // Delay a little bit to improve simulation performan
}
```

check if pushbutton is pressed. if it is, the butt

// SIRVE PARA PONER COMENTARIOS

Digital Input / Analog Input→ POTENTIOMETER (analog input) → Un potenciometro es una resistencia que se puede controlar (lo define como a type of rotating variable resistor). Como siempre la imagen de la izquierda es equivalente al circuito del de la derecha.

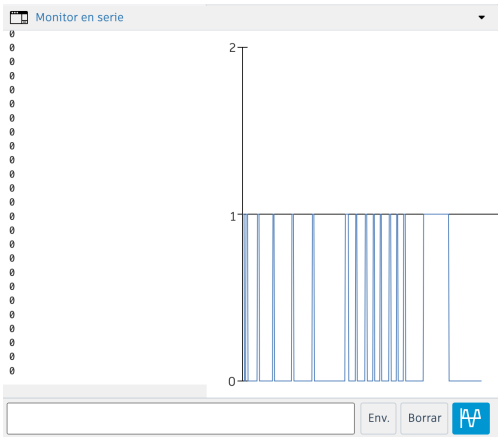
Creo que se entiende mejor con el código de abajo, peor bueno, primero le metes una señal analógica al cable verde, este va al potenciometro y según como esté dará una resistencia u otra, puede apagar el circuito cable negro o mantenerlo activo cable rojo.

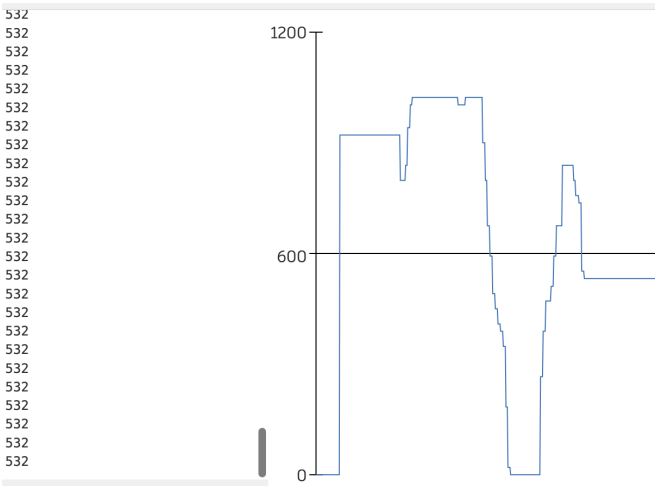


El código primero defines como int (pq la variable va a ser un número), sensorValue. Luego ne el setup(), lees A0 como input y 13 como output igual que antes. En el loop utilizas analogRead(pin) para leer el estado de A0 (valor entre 1 y 1023) luego editas el led con digital Write igual que los ejers anterior y pones un delay con el valor del pin A0 (el del potenciometro.

```
// C++ code
//
int sensorValue = 0;

void setup()
{
  pinMode(A0, INPUT);
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop()
{
  // read the value from the sensor
  sensorValue = analogRead(A0);
  // turn the LED on
  digitalWrite(LED_BUILTIN, HIGH);
  // pause the program for <sensorValue> millseconds
  delay(sensorValue); // Wait for sensorValue millisecond(s)
  // turn the LED off
  digitalWrite(LED_BUILTIN, LOW);
  // pause the program for <sensorValue> millseconds
  delay(sensorValue); // Wait for sensorValue millisecond(s)
}
```

Digital Input / Analog Input→ USING THE SERIAL MONITOR→ Te enseña a usar un monitor de serie para representar lo que pasa en el circuito (no sé mucho más). En la izquierda tienes un gráfica de un botón y en la derecha una de un potenciometrro

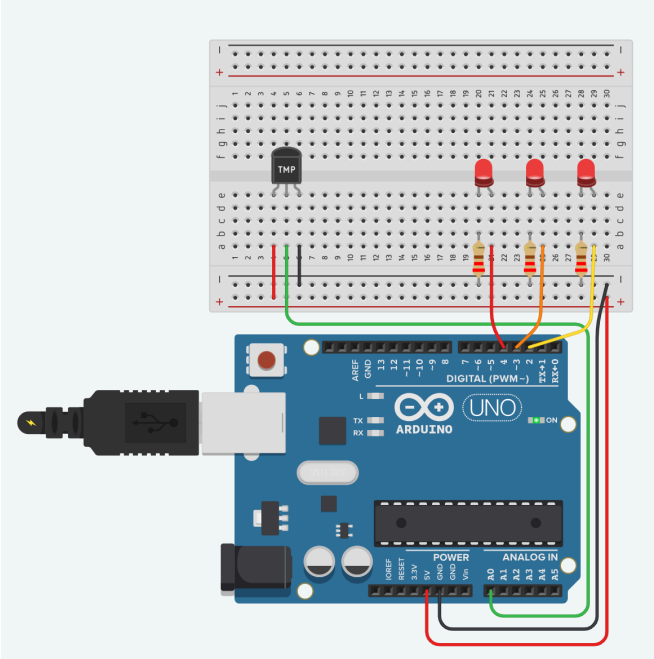


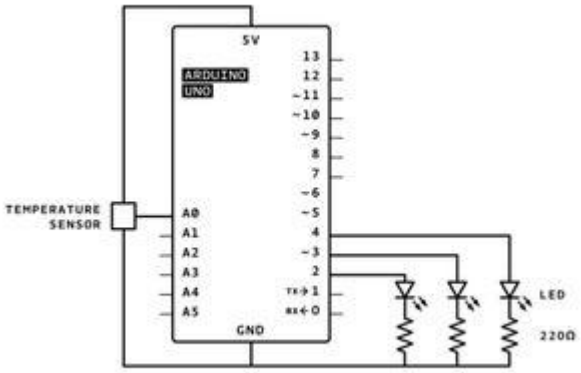



```
// C++ code
//
/*
  AnalogReadSerial
  Reads an analog input (potentiometer) on pin 0,
  prints the result to the serial monitor.

  OPEN THE SERIAL MONITOR TO VIEW THE OUTPUT FROM
  THE POTENTIOMETER >>

  Attach the center pin of a potentiometer to pin
  A0, and the outside pins to +5V and ground.

  This example code is in the public domain.
*/

int sensorValue = 0;

void setup()
{
  pinMode(A0, INPUT);
  Serial.begin(9600);
}

void loop()
{
  // read the input on analog pin 0:
  sensorValue = analogRead(A0);
  // print out the value you read:
  Serial.println(sensorValue);
  delay(10); // Delay a little bit to improve simulation performanc
}
```

Serial.begin() abre un puente en serie y fija un valor (en baudios) para la velocidad de transmisión. Luego serial.println manda esos valores leidos al monitor en serie y este luego los plotea

Temperature Sensor (analog input) → Vale forma final de entenderlo lo juro, mira lo de la temperatura (lo negro), es un sensor, esto lo que hace es sacar un output en V, el verde está en analog y lo cambia para que los cables rojo, naranja y amarillo pillen ya ese valor, los cables. ES TIPO entra por el rojo, sale por el negro y sale info por el verde, luego entra por el otro rojo/naranja/amarillo enciende el led y pasa por la resistencia y al ground de nuevo





Si el calor no es mucho, se enciende solo el led de la derecha

Pasando al código:
Defines tres variables como int
SETUP()
LEES A0, pones un serial.begin para empezar a leer cada vez, luego pones un output en los led
LOOP()
Pones la temp para activar los led
Usas map para una vez leido el A0, lo escalas: **map(valor, mínimo original, máximo original, mínimo a escalar, máximo a escalar)**

Pasas los valores al monitor de serie y luego comparas con if y actuas en consecuencia mirando la temp y comparando con el valor baselineTemp que hemos usado lo antes y activando o desactivando los leds.

```
// C++ code
//
int baselineTemp = 0;

int celsius = 0;

int fahrenheit = 0;

void setup()
{
  pinMode(A0, INPUT);
  Serial.begin(9600);
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
}

void loop()
{
  // set threshold temperature to activate LEDs
  baselineTemp = 40;
  // measure temperature in Celsius
  celsius = map(((analogRead(A0) - 20) * 3.04), 0, 1023, -40, 125
  // convert to Fahrenheit
  fahrenheit = ((celsius * 9) / 5 + 32);
  Serial.print(celsius);
  Serial.print(" C, ");
  Serial.print(fahrenheit);
  Serial.println(" F");
  if (celsius < baselineTemp) {
    digitalWrite(2, LOW);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
  }
  if (celsius >= baselineTemp && celsius < baselineTemp + 10) {
    digitalWrite(2, HIGH);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
  }
  if (celsius >= baselineTemp + 10 && celsius < baselineTemp + 20
    digitalWrite(2, HIGH);
    digitalWrite(3, HIGH);
    digitalWrite(4, LOW);
  }
  if (celsius >= baselineTemp + 20 && celsius < baselineTemp + 30
    digitalWrite(2, HIGH);
    digitalWrite(3, HIGH);
    digitalWrite(4, HIGH);
  }
  if (celsius >= baselineTemp + 30) {
    digitalWrite(2, HIGH);
    digitalWrite(3, HIGH);
    digitalWrite(4, HIGH);
  }
  delay(1000); // Wait for 1000 millisecond(s)
}
```