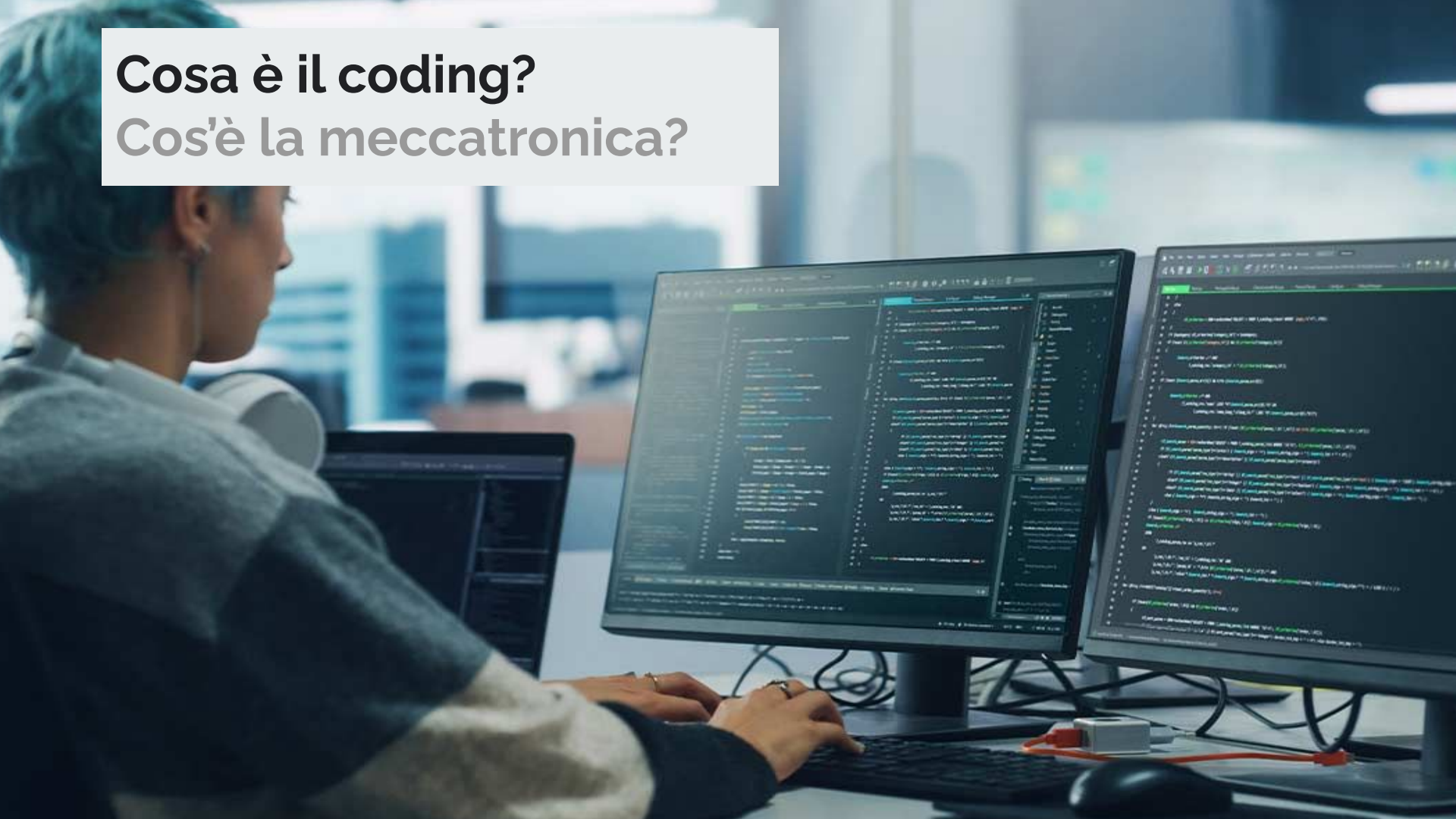




Coding e meccatronica

Corso pratico di meccatronica con Arduino
A cura di Giulio Pons

Cosa è il coding?
Cos'è la meccatronica?



Cosa è il coding?
Cos'è la meccatronica?





Esploriamo il kit



Cosa è Arduino

È un piccolo computer.

Open source.

connessione USB
al computer

orologio

regolatore 5V

alimentazione

pin digitali
per collegare cose

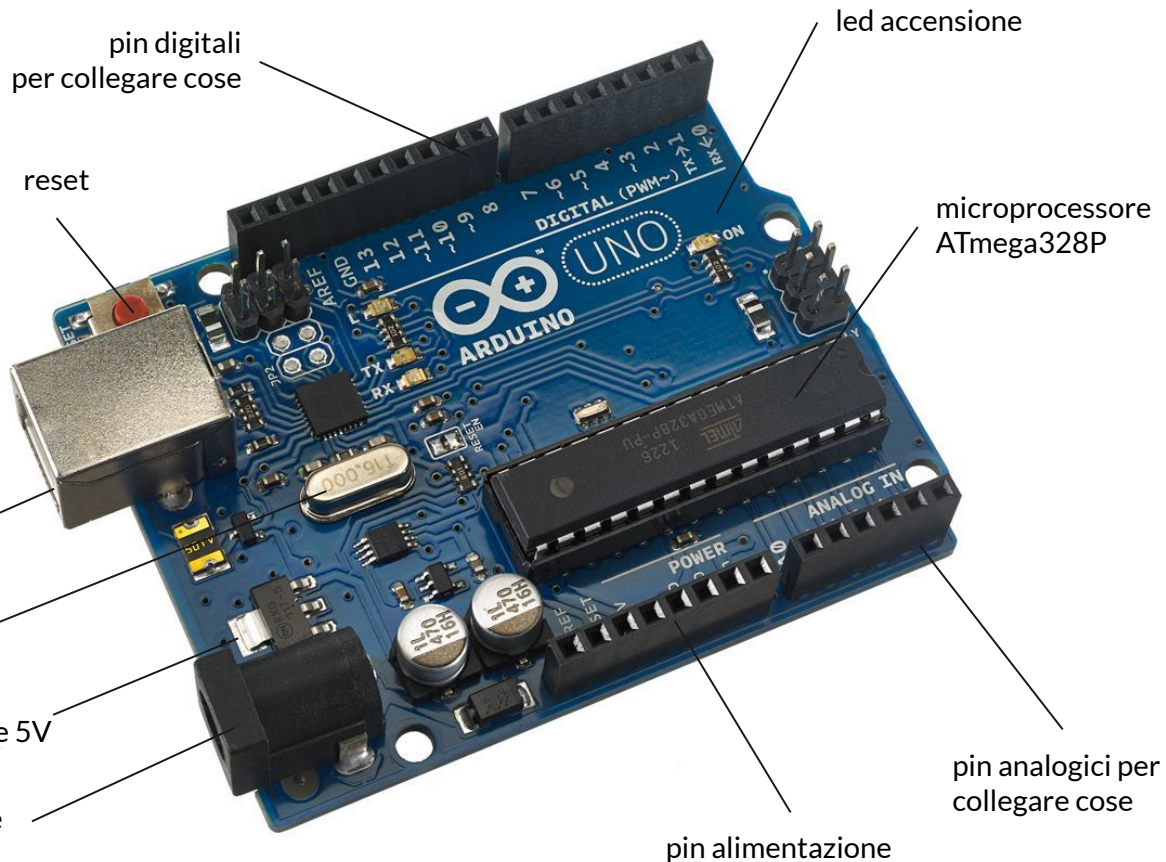
reset

led accensione

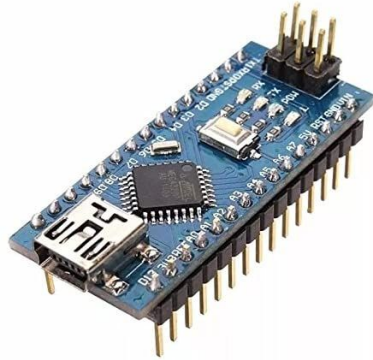
microprocessore
ATmega328P

pin alimentazione

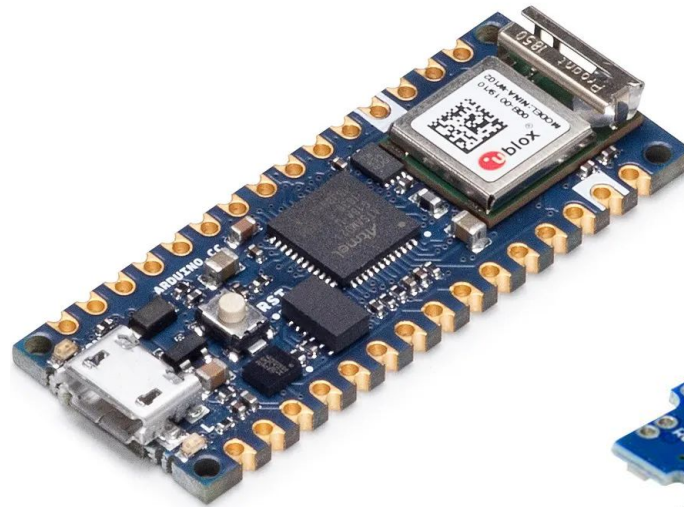
pin analogici per
collegare cose



Altri modelli



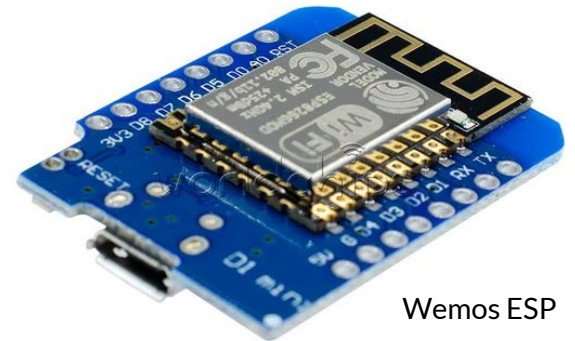
Arduino Nano



Arduino IOT BLE



ESP8266-01



Wemos ESP

Sensori

Possiamo collegare dei sensori per vedere/sentire quello che succede intorno.



luce



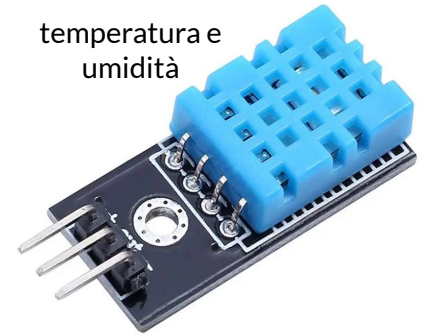
campo magnetico



distanza



infrarossi



temperatura e
umidità



prossimità

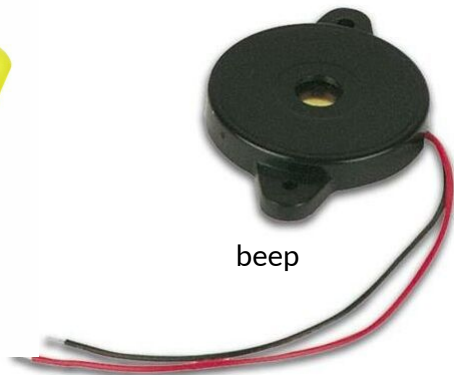


Attuatori

Possiamo collegare dei convegni che “fanno cose” per agire sul mondo intorno.



luce



beep



serratura

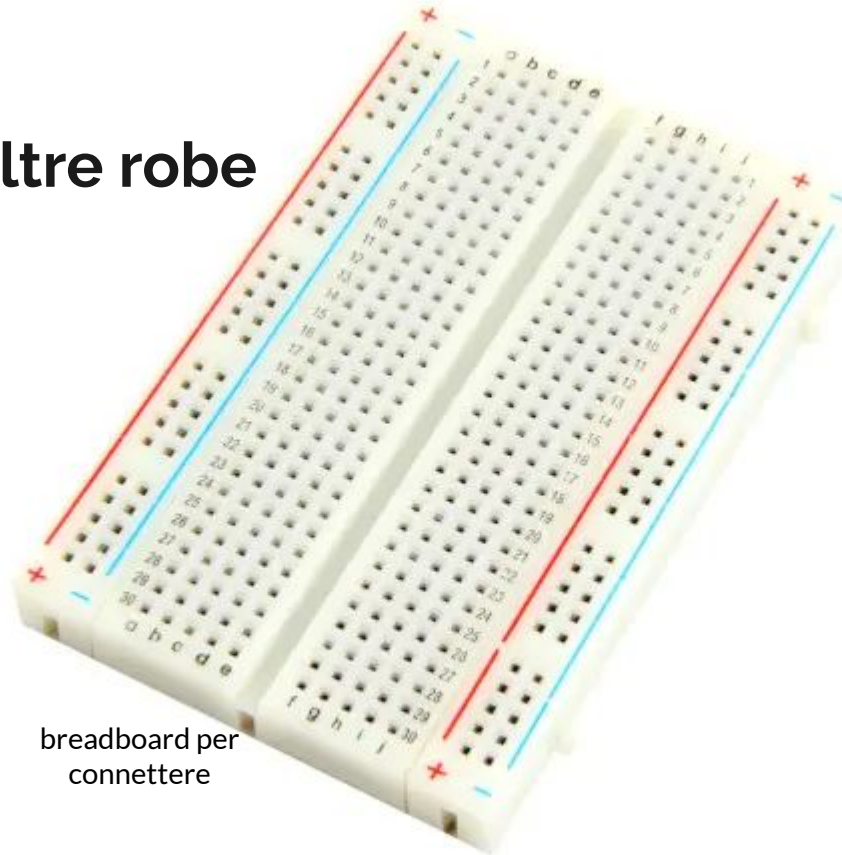


relè

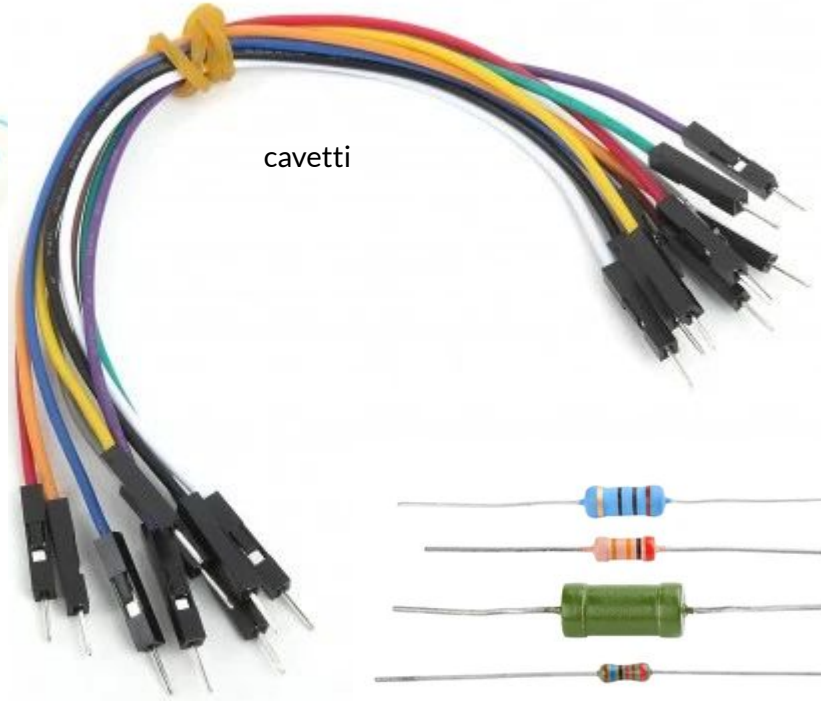


motore

Altre robe



breadboard per
connettere



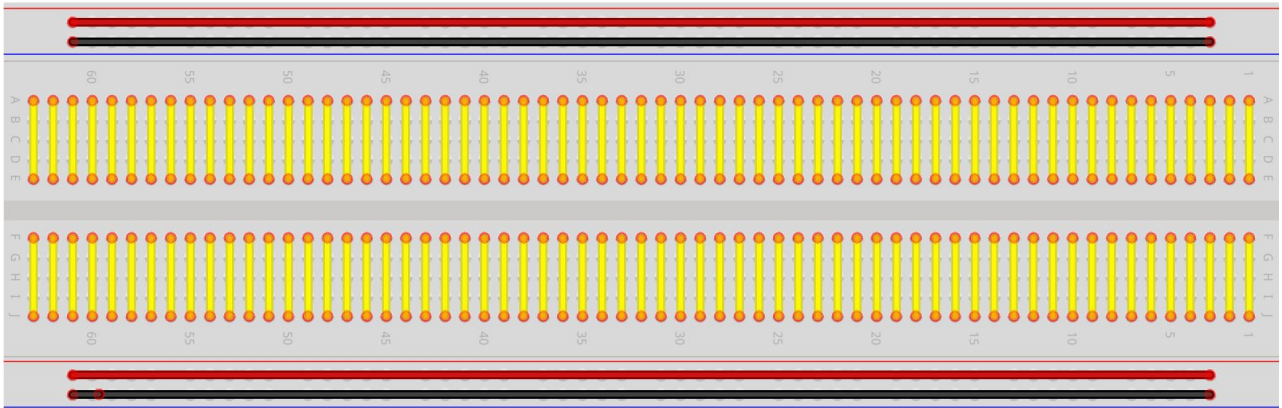
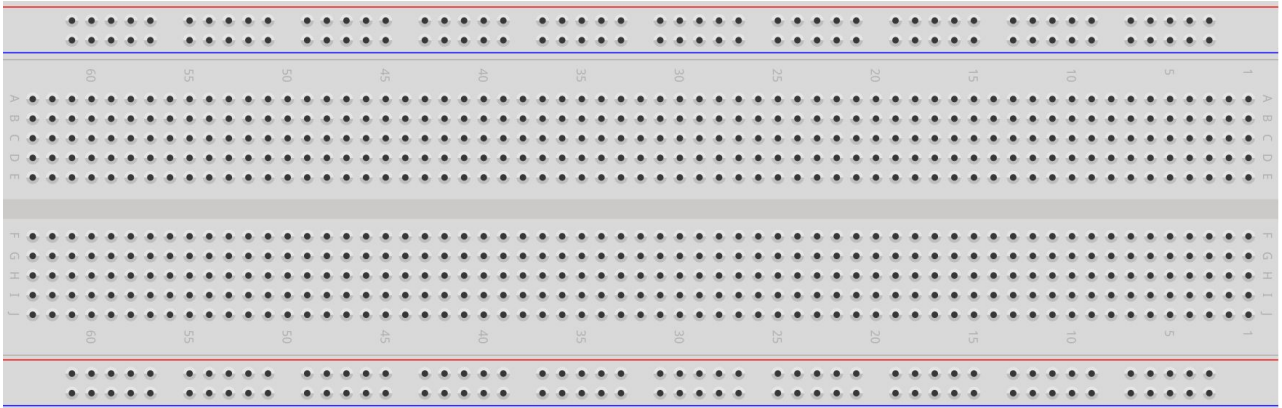
cavetti



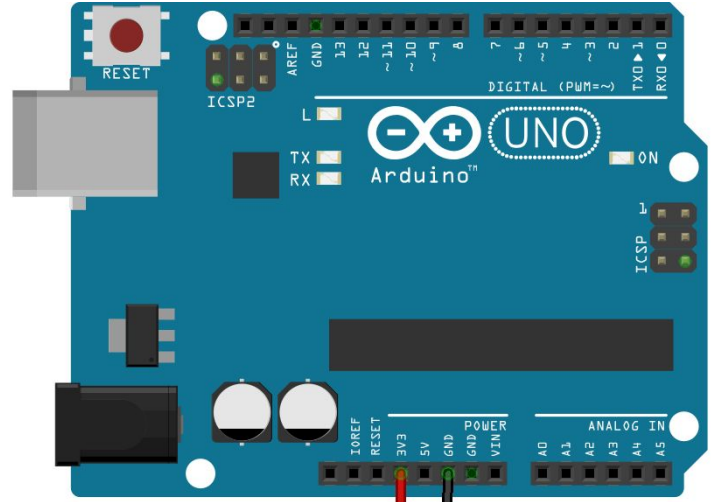
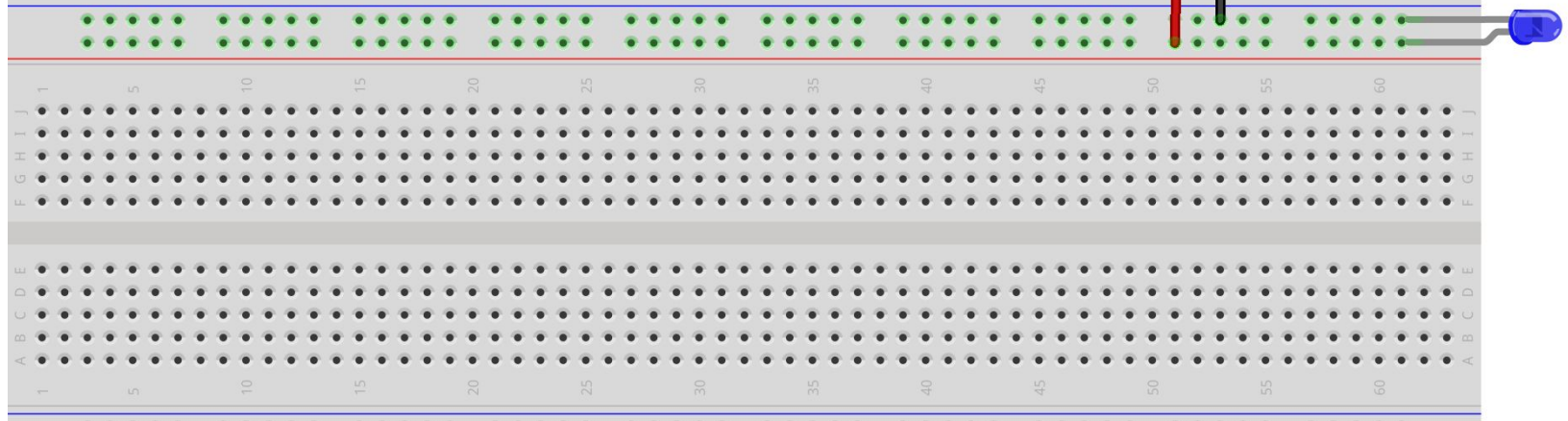
resistenze



Breadboard

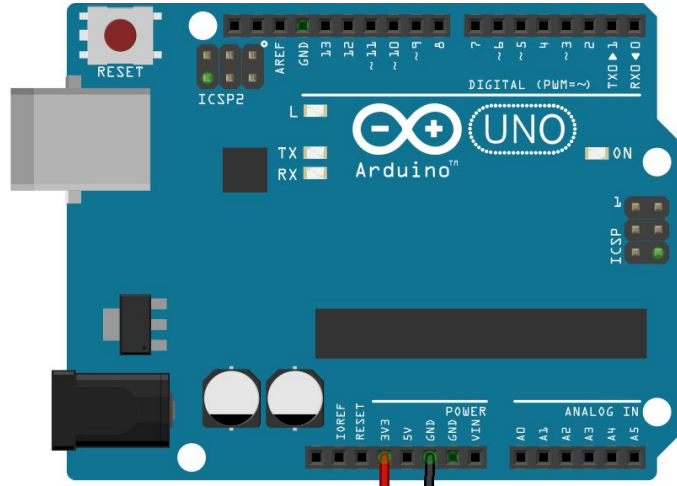
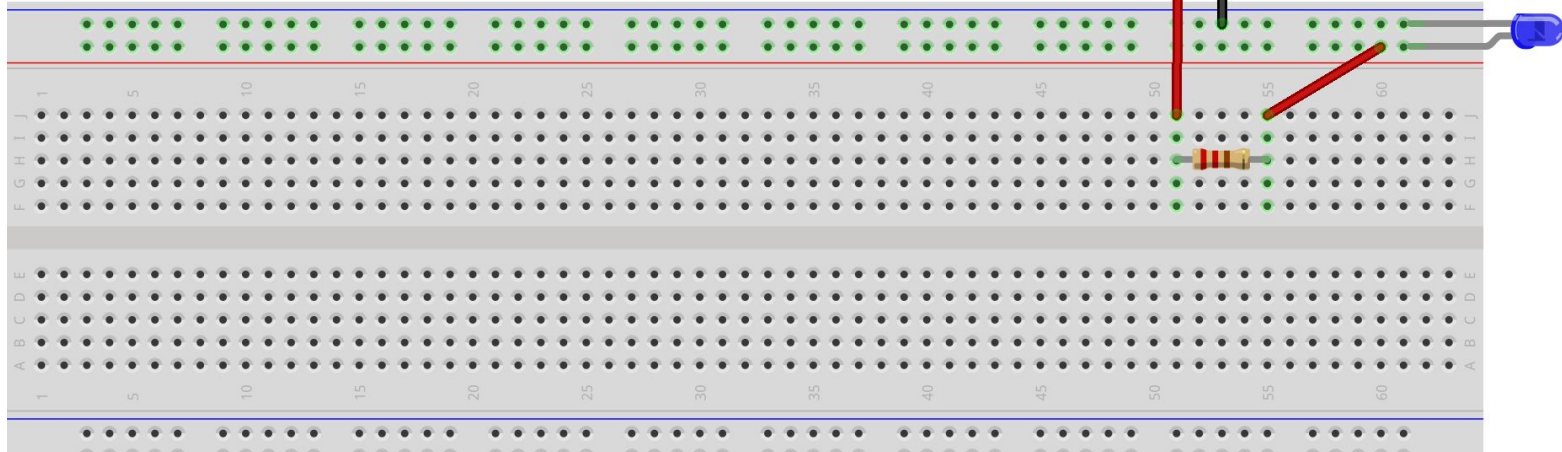


Accendiamo una luce



Accendiamo una luce

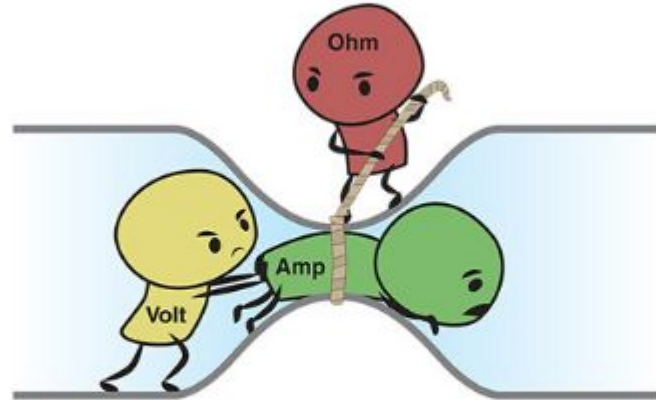
Aggiungiamo una resistenza



Resistenza

La resistenza rallenta il passaggio
della corrente elettrica

$$V = I * R$$





Riprendere circuito elettrico

Riprendere cosa è un circuito elettrico.

Riprendere il concetto di resistenza, introdurre il concetto di corrente elettrica.

Legge di Ohm.

I programmi

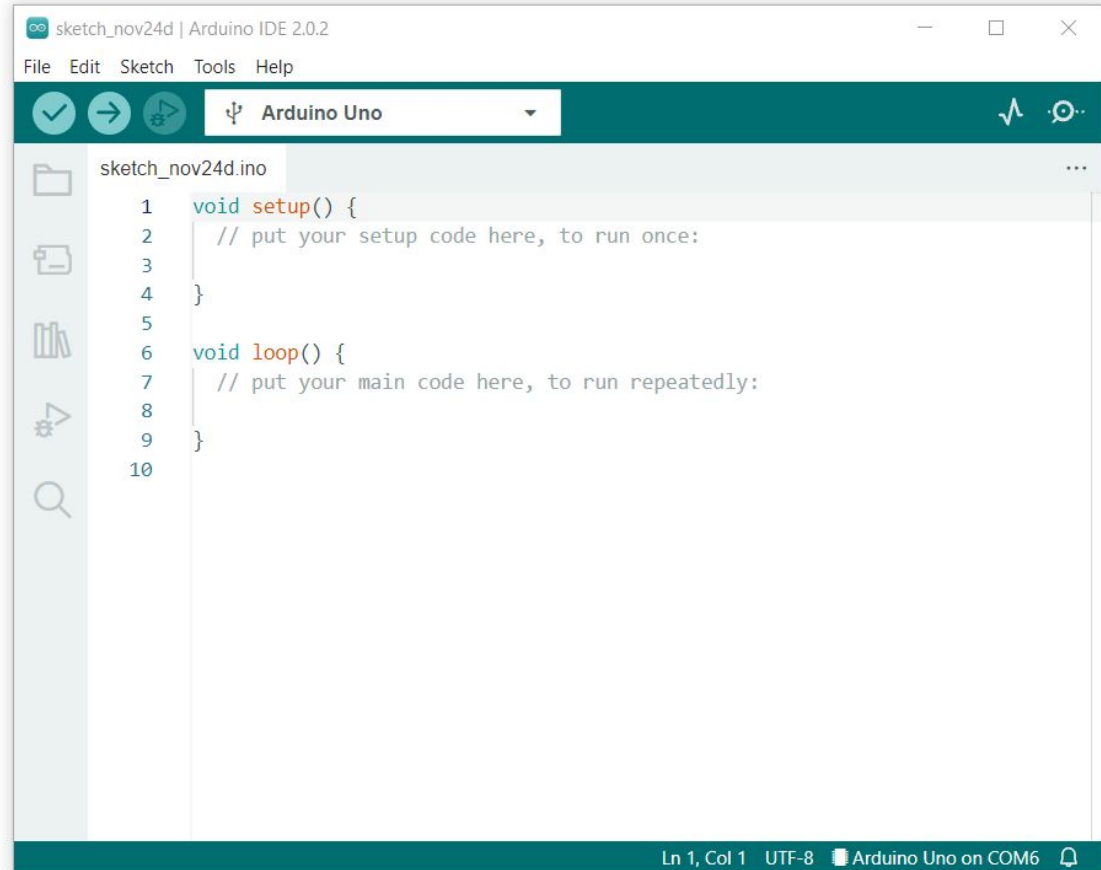
Come tutti i computer Arduino funziona con i programmi.

I programmi si scrivono dentro la Arduino IDE.

Abbiamo tutti installato la IDE?

WEB EDITOR <https://create.arduino.cc/editor>

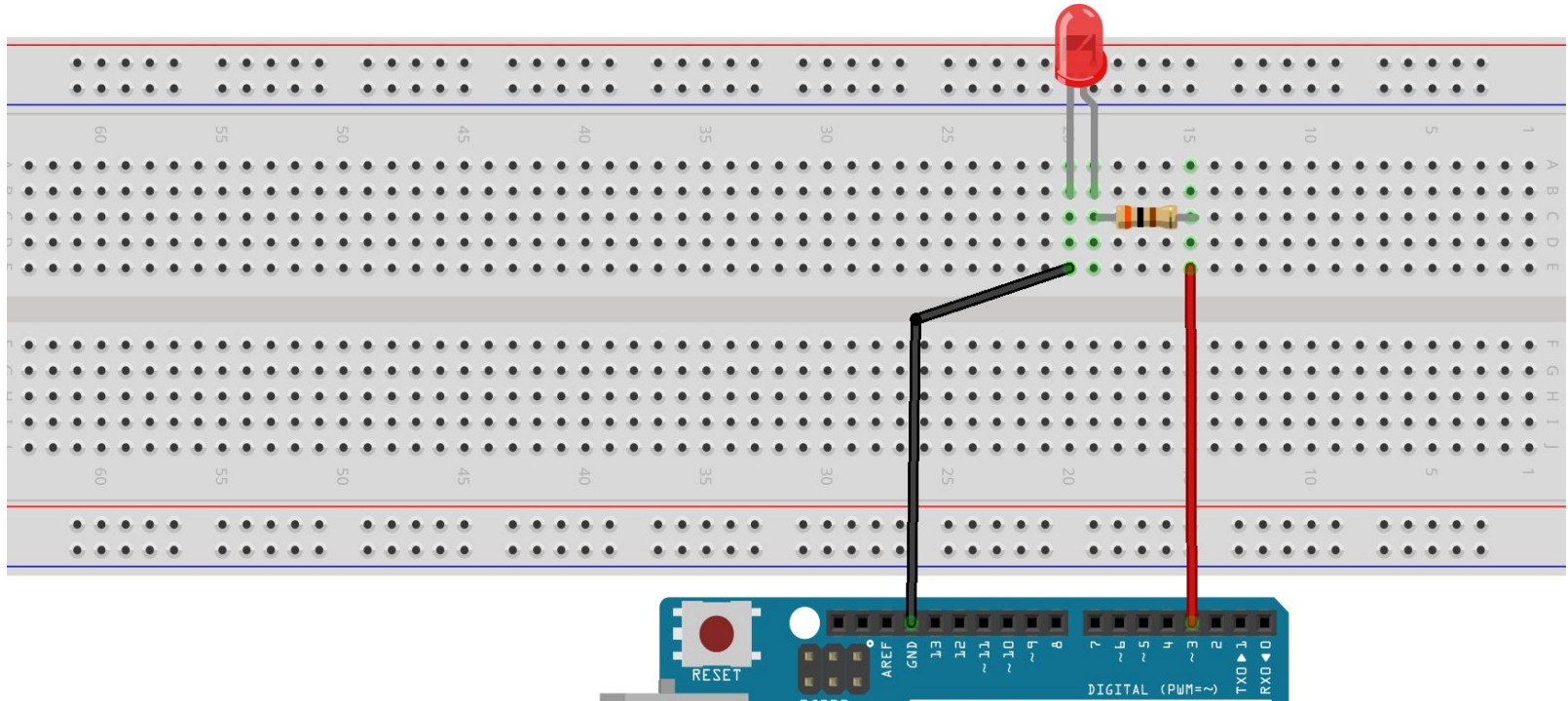
DOWNLOAD <https://docs.arduino.cc/software/ide-v2>

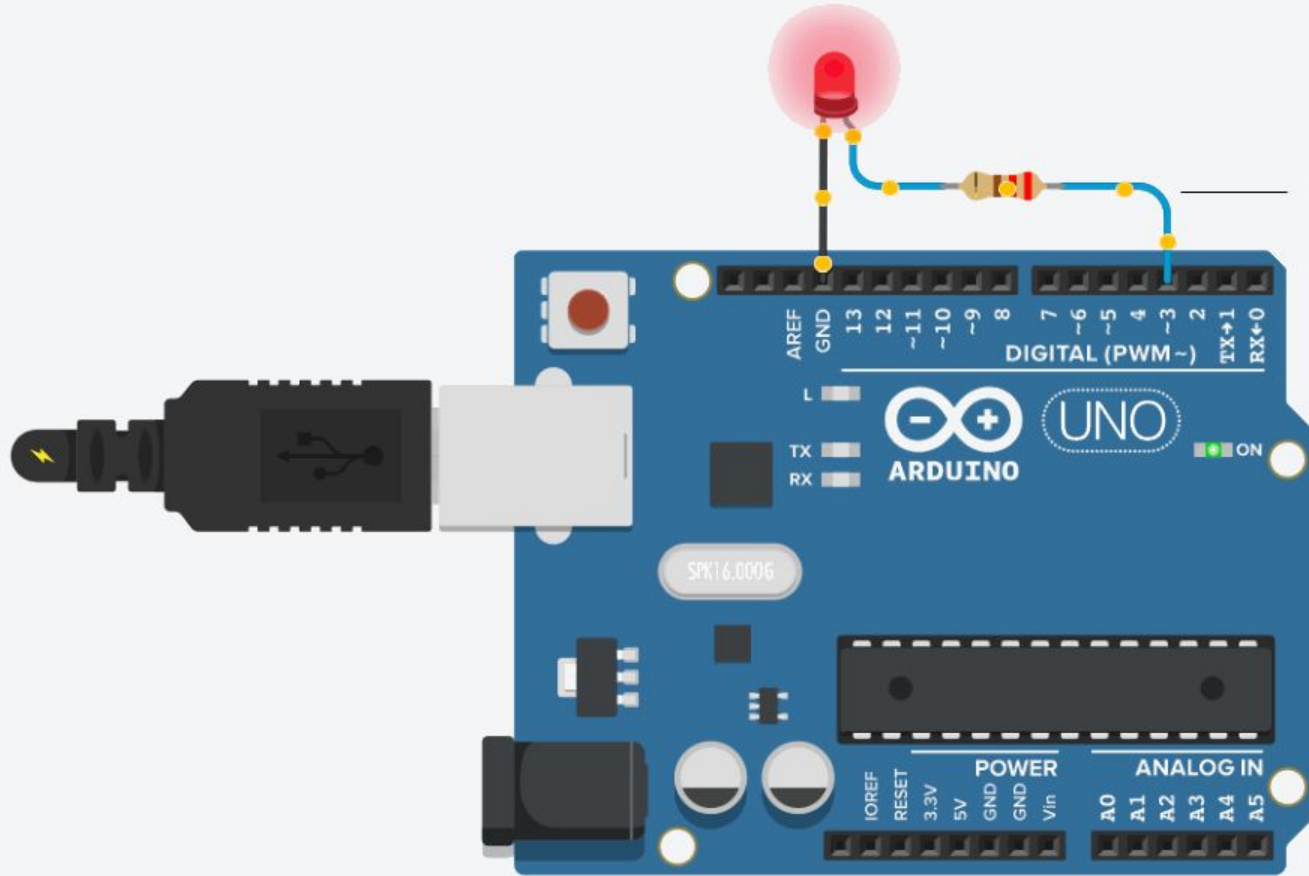


```
sketch_nov24d.ino
1 void setup() {
2   // put your setup code here, to run once:
3
4 }
5
6 void loop() {
7   // put your main code here, to run repeatedly:
8
9 }
10
```

Ln 1, Col 1 UTF-8 Arduino Uno on COM6

Si inizia con le basi





Pin 3 is HIGH (5V)
Pin 3 is set to 5V via
`digitalWrite(3, HIGH);`



Accendi

Variabili e comandi.

Spiegazione codice.

Upload.

Esecuzione.

```
1  int pin = 3;
2
3  void setup() {
4      delay(1000);
5      pinMode(pin, OUTPUT);
6  }
7
8  void loop() {
9      digitalWrite(pin, HIGH);
10 }
```



Accendi / spegni

Variabili e comandi.

Spiegazione codice.

Upload.

Esecuzione.

```
1  int pin = 3;
2
3  void setup() {
4      pinMode(pin,OUTPUT);
5  }
6
7  void loop() {
8      digitalWrite(pin,HIGH);
9      delay(1000);
10     digitalWrite(pin,LOW);
11     delay(1000);
12 }
```



Nei programmi si usano variabili e comandi

```
int i = 10;
```

Variabili: memorie (scatole) con nome e tipo che contengono dei valori.

```
delay(1000);
```

Comandi: istruzioni che dicono al computer cosa fare (con parametri o no).

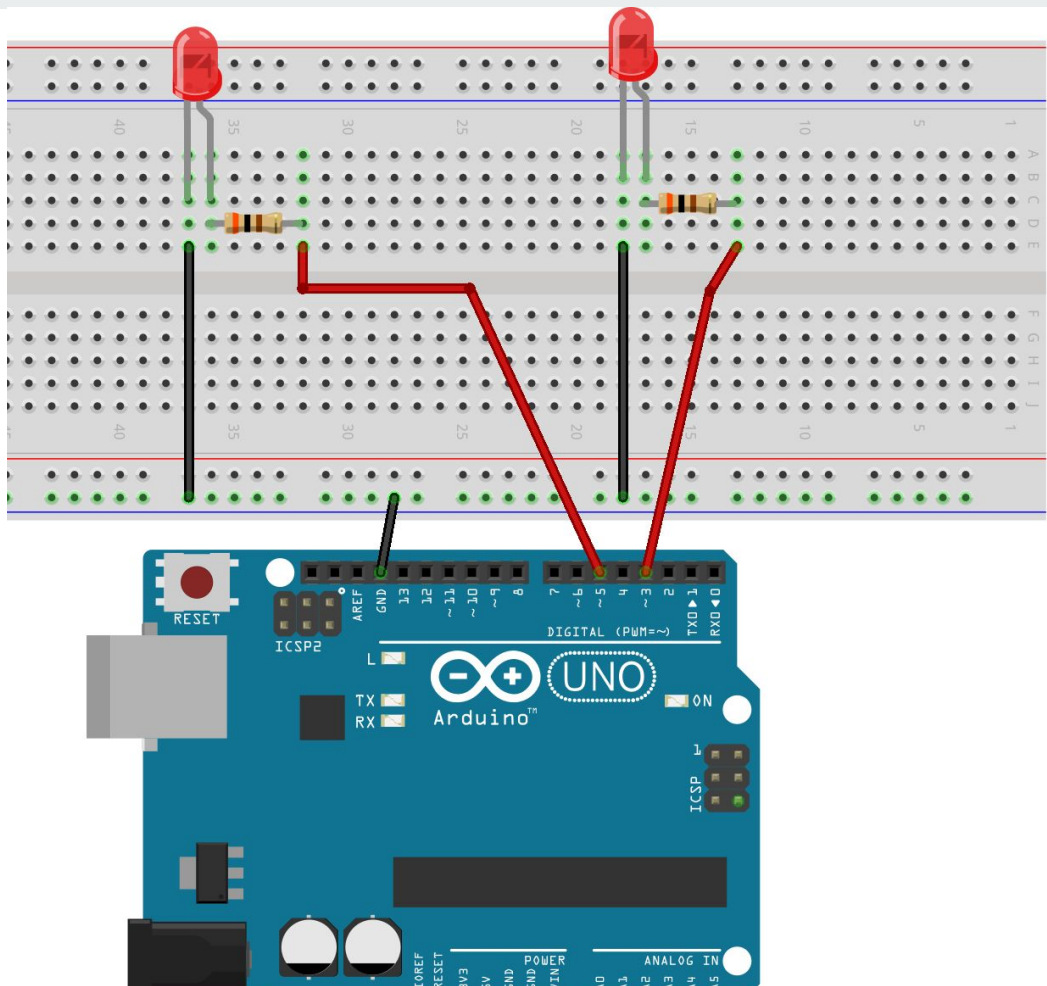
```
digitalWrite(3,HIGH);  
pinMode(3,OUTPUT);
```

Accendiamo due luci

PIN 3

PIN 5

GND



Che differenza c'è?

```
1  int pin = 3;
2  int pin2 = 5;
3  void setup() {
4      pinMode(pin,OUTPUT);
5      pinMode(pin2,OUTPUT);
6  }
7
8  void loop() {
9      digitalWrite(pin,HIGH);
10     delay(500);
11     digitalWrite(pin,LOW);
12     delay(500);
13
14     digitalWrite(pin2,HIGH);
15     delay(500);
16     digitalWrite(pin2,LOW);
17     delay(500);
```

```
1  int pin = 3;
2  int pin2 = 5;
3  void setup() {
4      pinMode(pin,OUTPUT);
5      pinMode(pin2,OUTPUT);
6  }
7
8  void loop() {
9      digitalWrite(pin,HIGH);
10     digitalWrite(pin2,LOW);
11     delay(500);
12
13     digitalWrite(pin,LOW);
14     digitalWrite(pin2,HIGH);
15     delay(500);
16 }
```



Valori in uscita con pinMode OUTPUT

Non solo HIGH e LOW

Valori in entrata con pinMode INPUT

```
1  int pin = 3;
2  void setup() {
3      pinMode(pin,OUTPUT);
4  }
5
6  int i=0;
7  void loop() {
8      analogWrite(pin,i);
9      delay(5);
10     i=i+1;
11     if(i==255) {
12         i=0;
13     }
14 }
```



Pipistrelli!

Sensore HC-SR04





Misurare la distanza

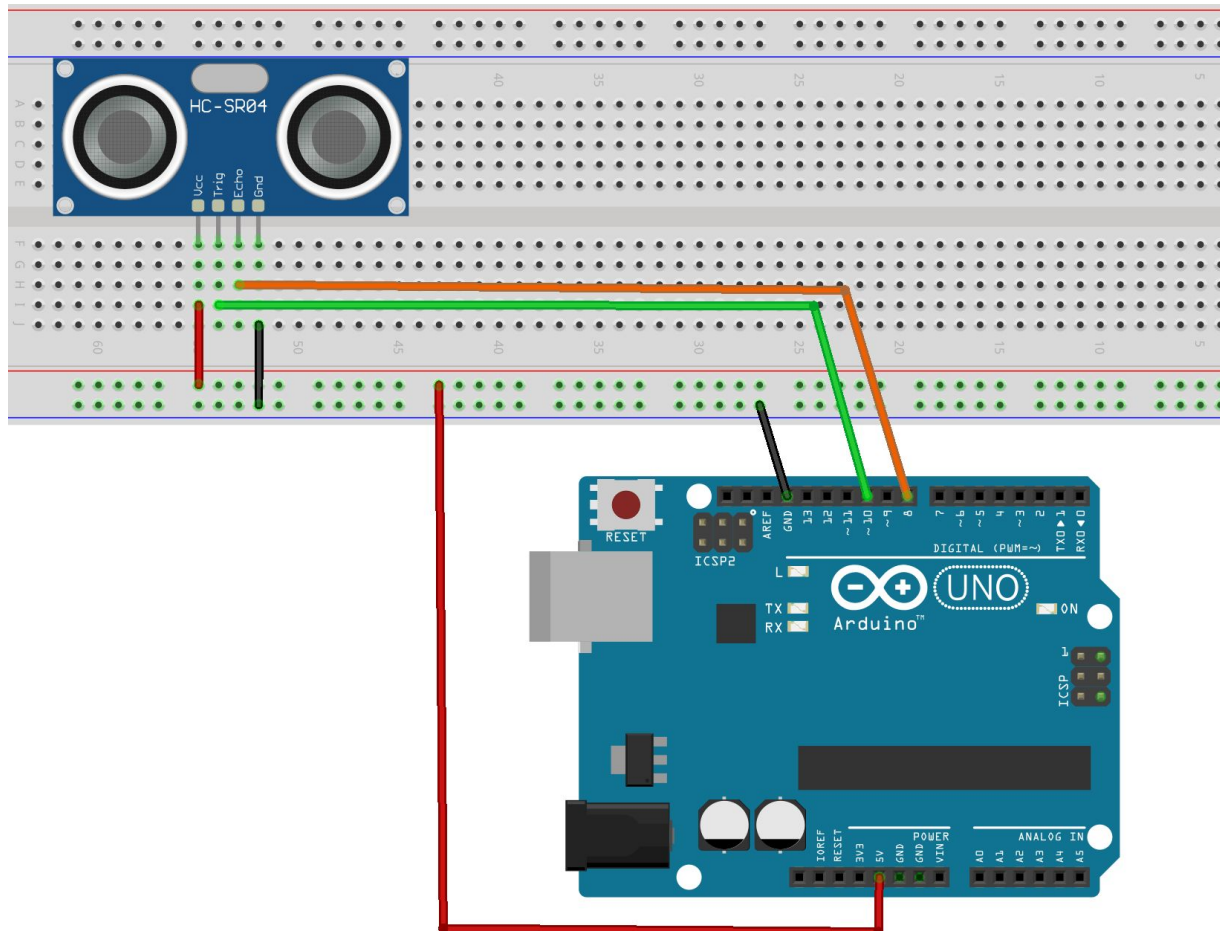
Sensore HC-SR04

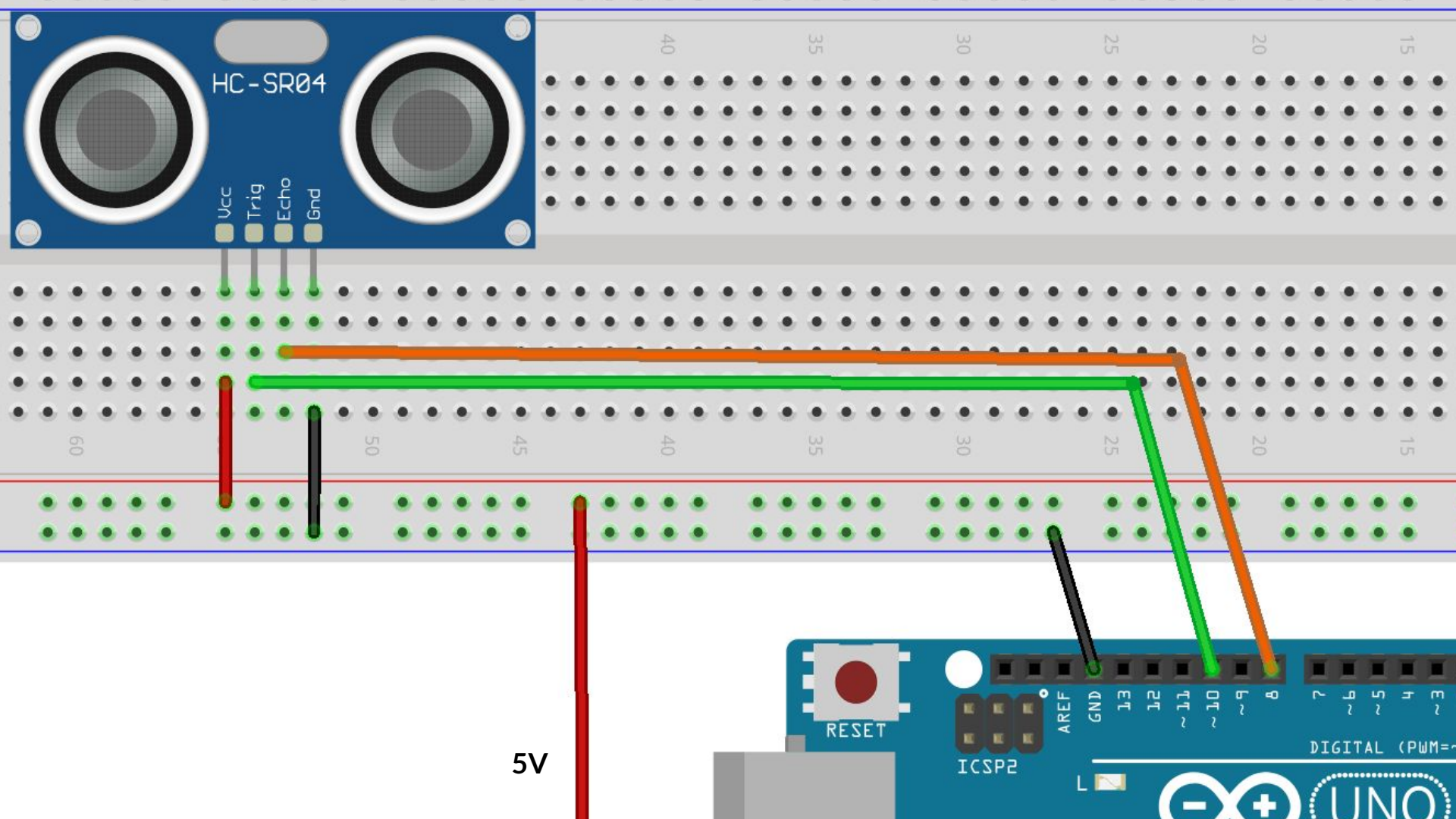




Collegamento

Sensore HC-SR04







Codice

Codice per leggere la distanza col sensore ad ultrasuoni HC-SR04.

Serial Monitor.

Upload.

Esecuzione.

`trigger` = grilletto

`echo` = eco

```
1  int triggerPin = 10;
2  int echoPin = 8;
3
4  void setup() {
5      pinMode(echoPin, INPUT);
6      pinMode(triggerPin, OUTPUT);
7      Serial.begin(9600);
8  }
9
10 long durata=0;
11 long distanza =0;
12
13 void loop() {
14
15     digitalWrite(triggerPin, HIGH);
16     delayMicroseconds(10);
17     digitalWrite(triggerPin, LOW);
18
19     durata = pulseIn(echoPin, HIGH, 200000);
20
21     distanza = durata*0.0172;
22     Serial.print("distanza= ");
23     Serial.println(distanza);
24     delay(50);
25 }
```



Riepilogo funzioni

```
delay(1000);  
digitalWrite(3,HIGH); analogWrite(3,123);  
pinMode(3,OUTPUT); pinMode(3,INPUT);
```

```
pulseIn(8,HIGH,200000);  
delayMicroseconds(10);  
Serial.print("ciao");  
Serial.println("ciao");
```

Comandi (o funzioni): istruzioni che dicono al computer cosa fare (con parametri o no).

Riepilogo variabili

```
int i = 10;
```

```
long durata = 1000000;
```

```
int pin = 3;
```

```
i = i + 1;
```

Variabili: memorie (scatole) con nome e tipo che contengono dei valori.



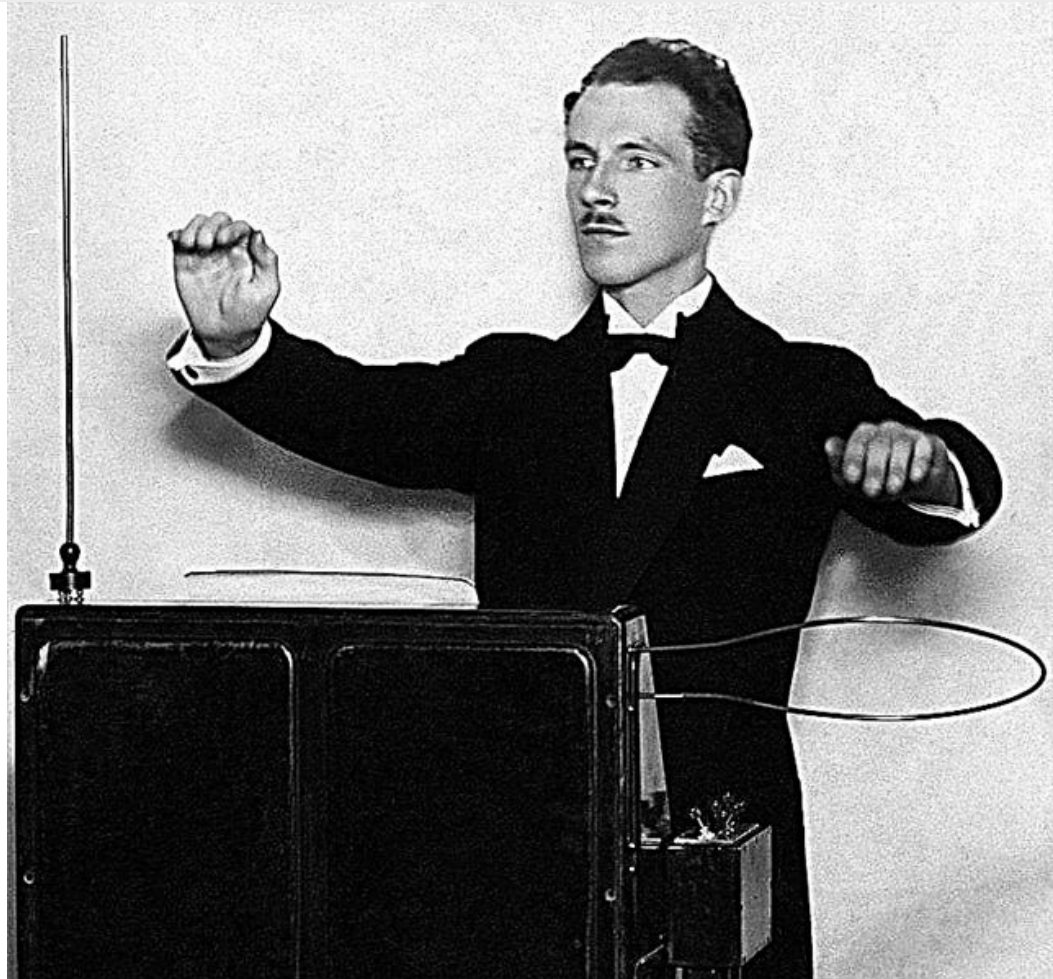
Theremin



Sensore

Attuatore

Suoni modulati



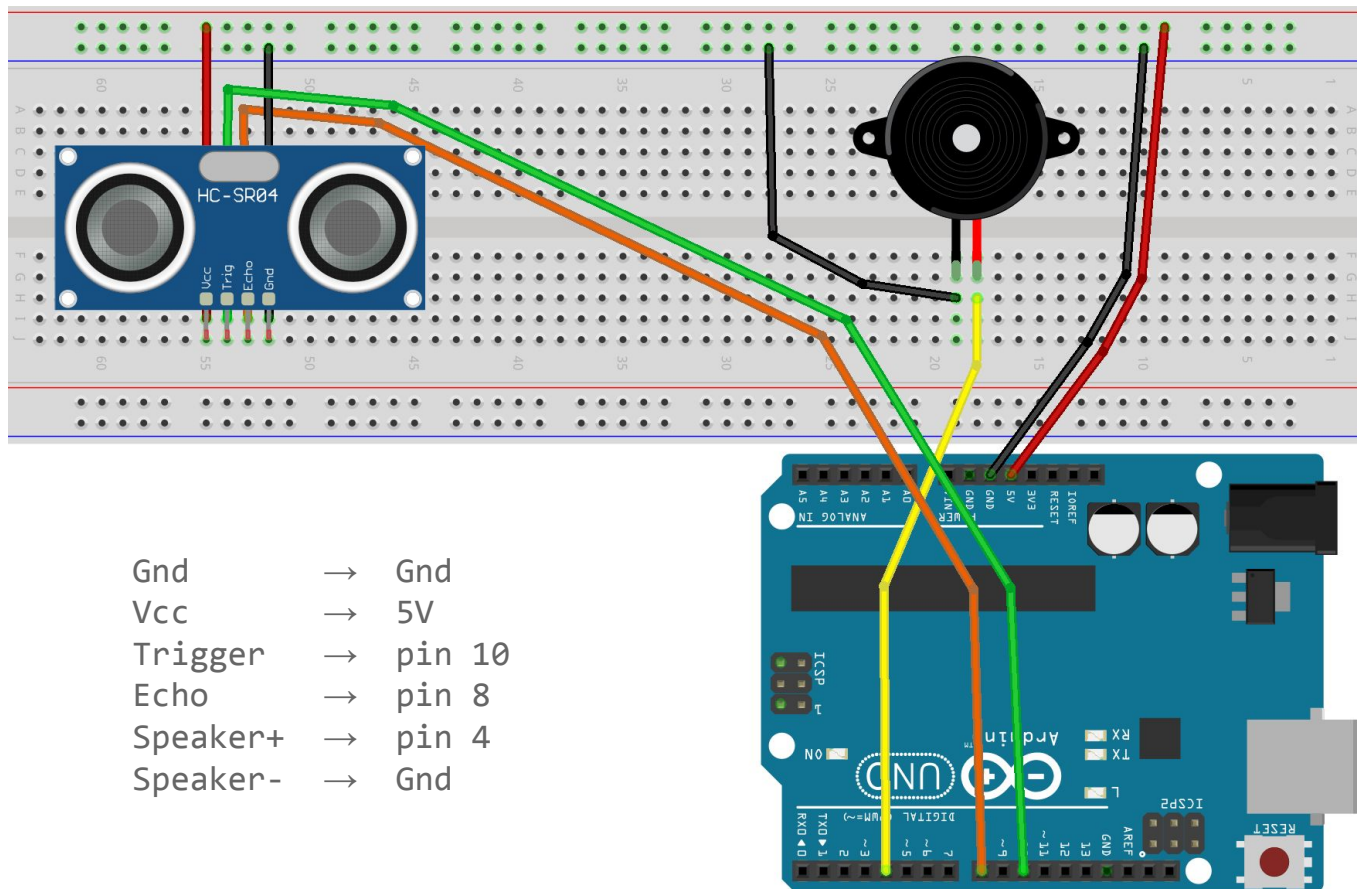


Theremin

Sensore

Attuatore

Suoni modulati!

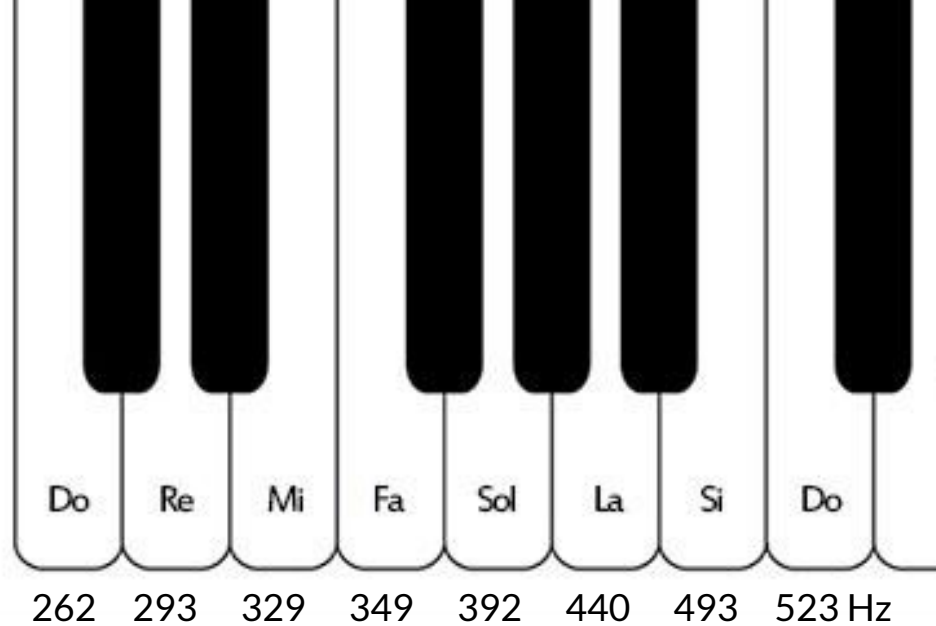
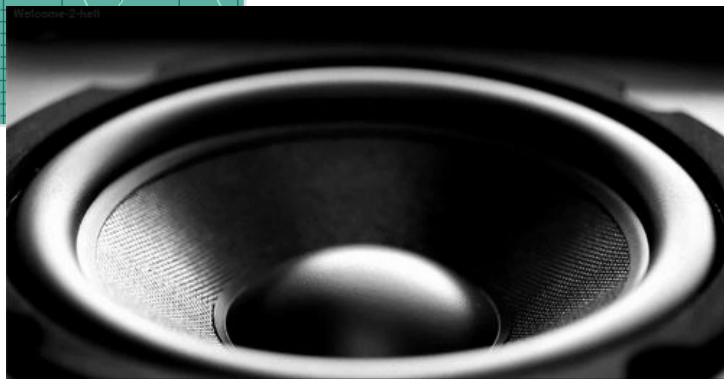
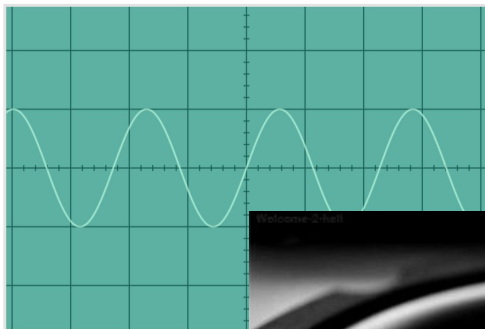


- Gnd → Gnd
- Vcc → 5V
- Trigger → pin 10
- Echo → pin 8
- Speaker+ → pin 4
- Speaker- → Gnd



Note musicali

tone(pin, freq);



Aggiungiamo il piezo speaker



questo va meglio

```
1  int triggerPin = 10;
2  int echoPin = 8;
3  int pinBuzzer= 4;
4  long durata=0;
5  long distanza =0;
6
7  void setup() {
8      pinMode(echoPin, INPUT);
9      pinMode(triggerPin, OUTPUT);
10     pinMode(pinBuzzer, OUTPUT);
11 }
12
13 void loop() {
14
15     digitalWrite(triggerPin, HIGH);
16     delayMicroseconds(10);
17     digitalWrite(triggerPin, LOW);
18
19     durata = pulseIn(echoPin, HIGH, 200000);
20     distanza = durata*0.1175;
21
22     if (distanza>40 && distanza<300) {
23         long nota = map(distanza,40,300,262,523);
24         tone(pinBuzzer, nota);
25     } else {
26         noTone(pinBuzzer);
27     }
28     delay(50);
29 }
```



Funzioni

```
delay(1000);  
digitalWrite(3,HIGH);  
pinMode(3,OUTPUT);
```

```
pulseIn(8,HIGH,200000);  
delayMicroseconds(10);  
Serial.print("ciao");  
Serial.println("ciao");
```

```
tone(4,650);  
noTone(4);  
map( valore, min, max, nuovomin, nuovomax);
```

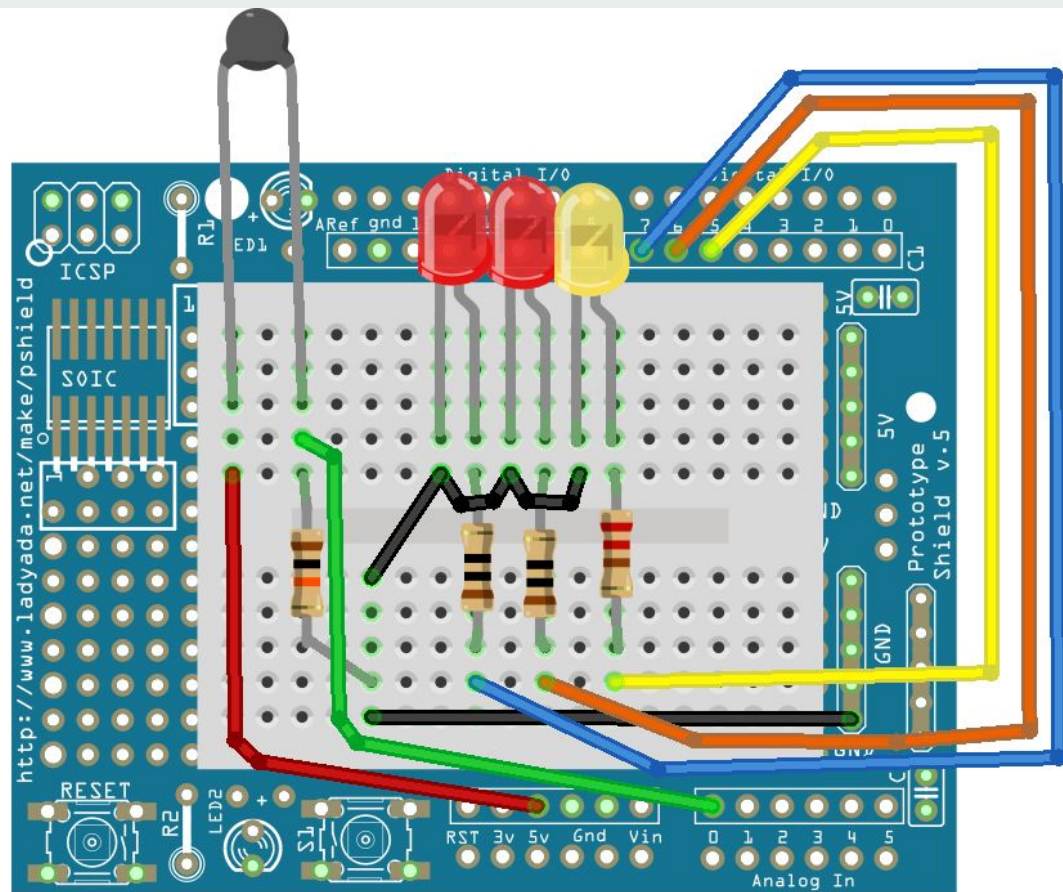
Parentesi graffe e if

```
if(distanza>3000) {  
    noTone(pinBuzzer);  
}
```

```
if(i == 255) i=0;
```

Luce e sensore temperatura

R10k

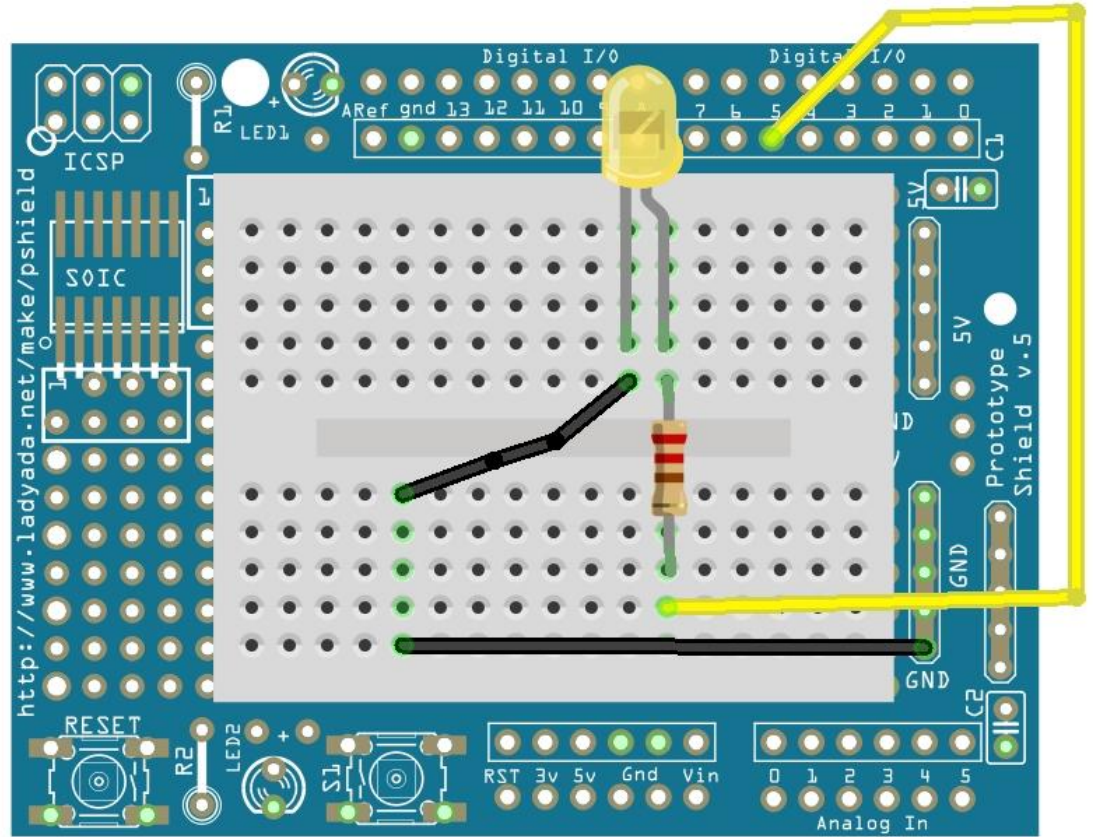


R10

R10

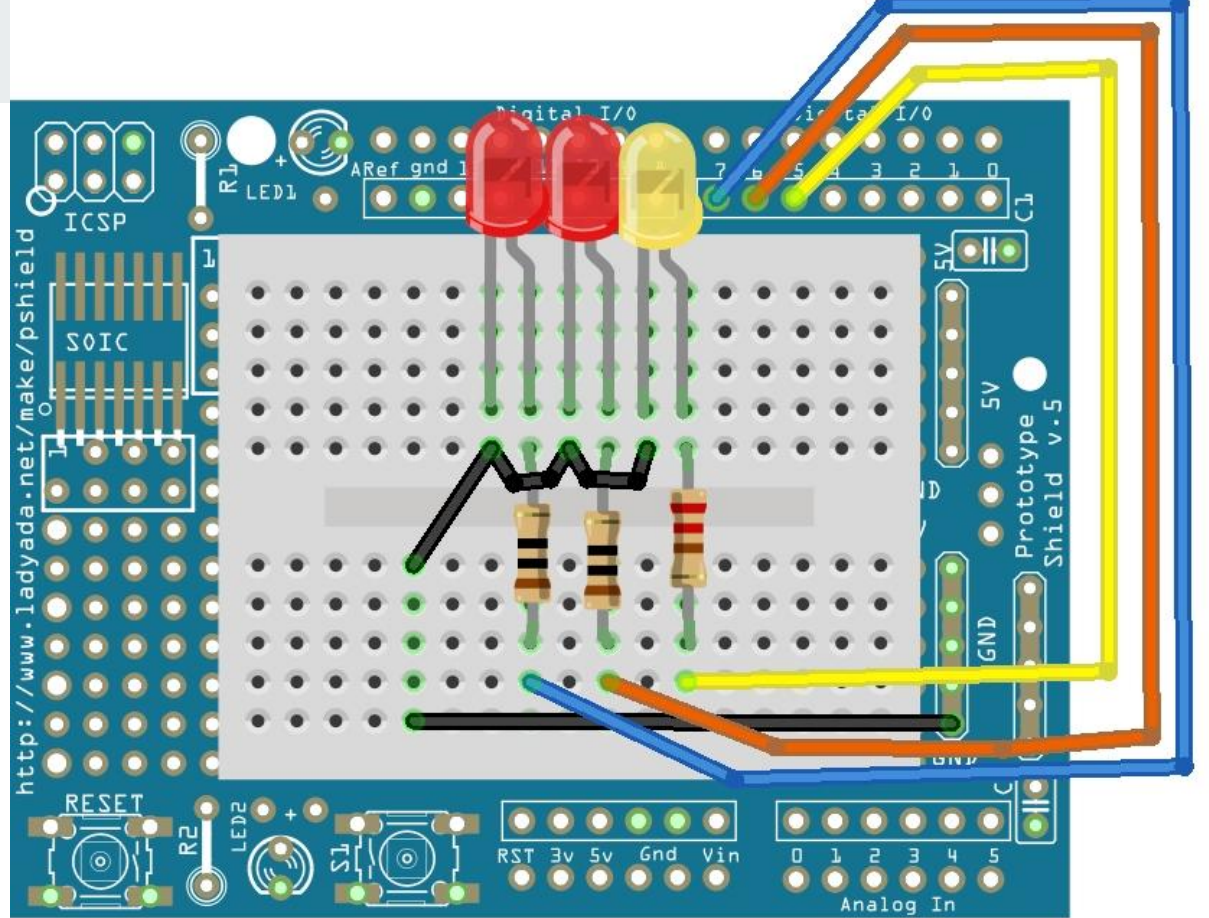
R100

Un led
poi due
...



R100

Poi tre



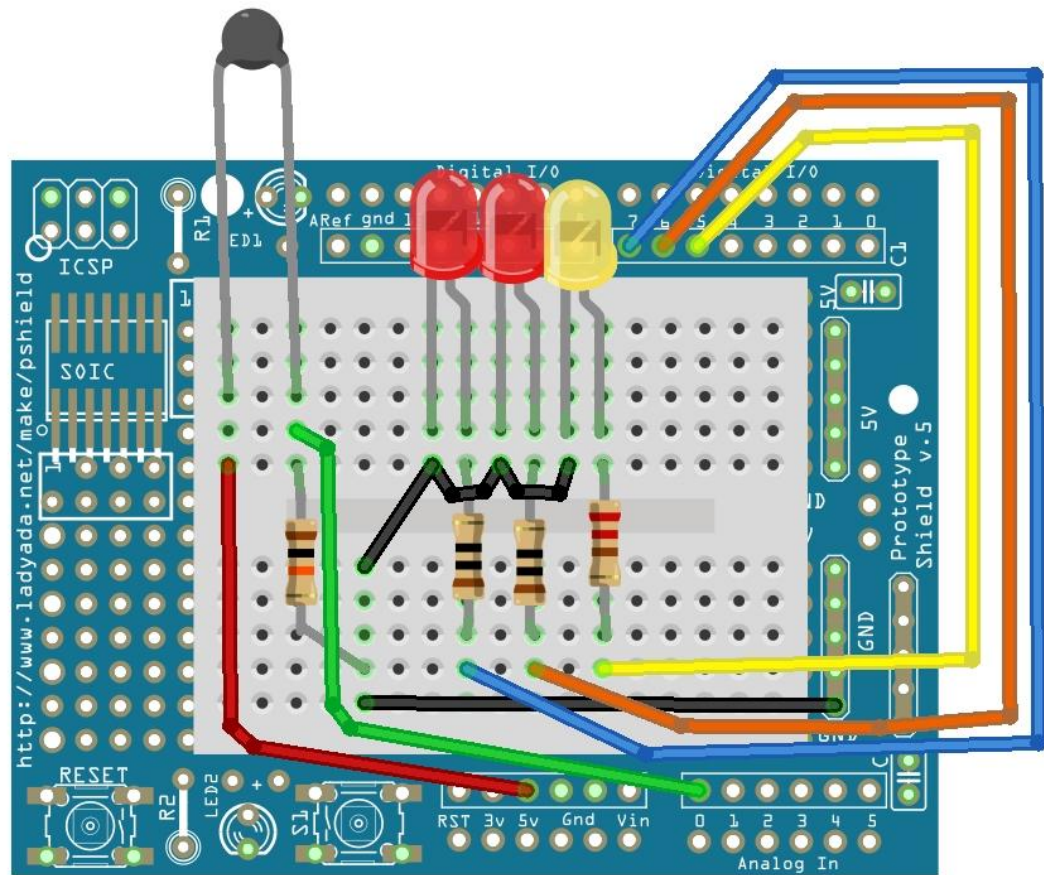
R10

R10

R100

Poi il sensore
di temperatura

R10k

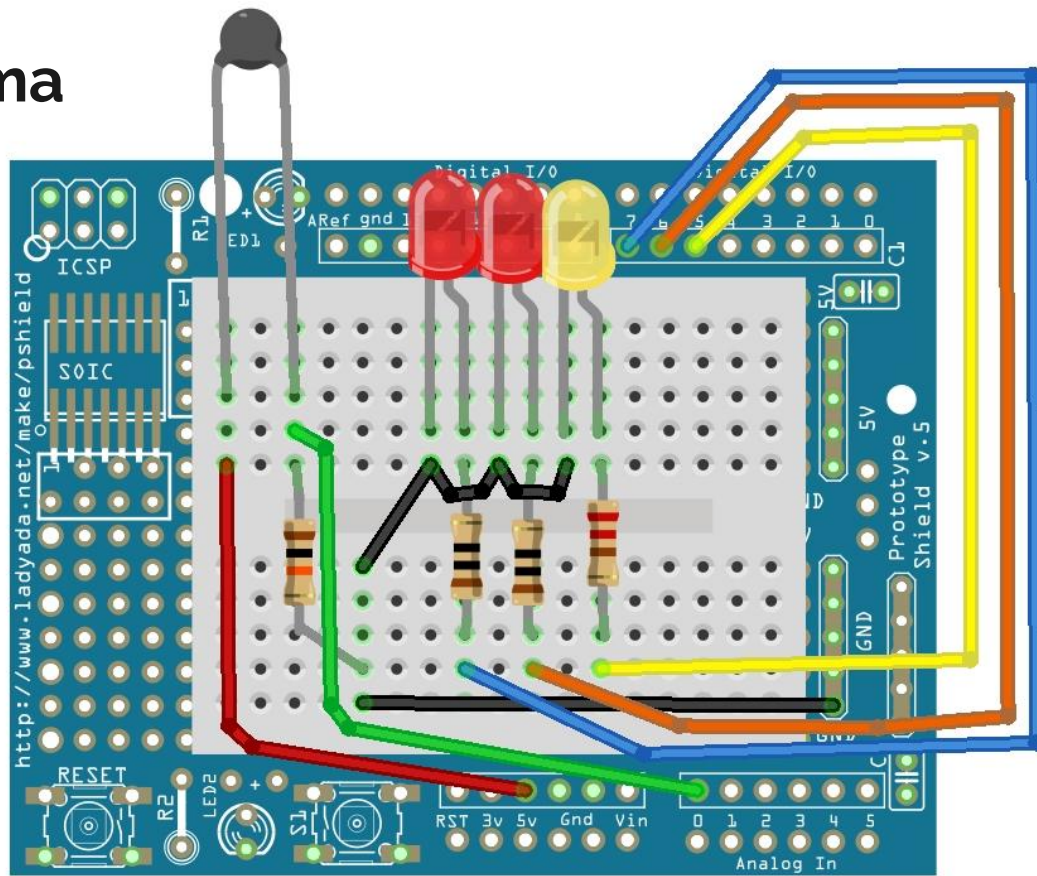


R10

R10

R100

Fiamma



R10k

R10

R10

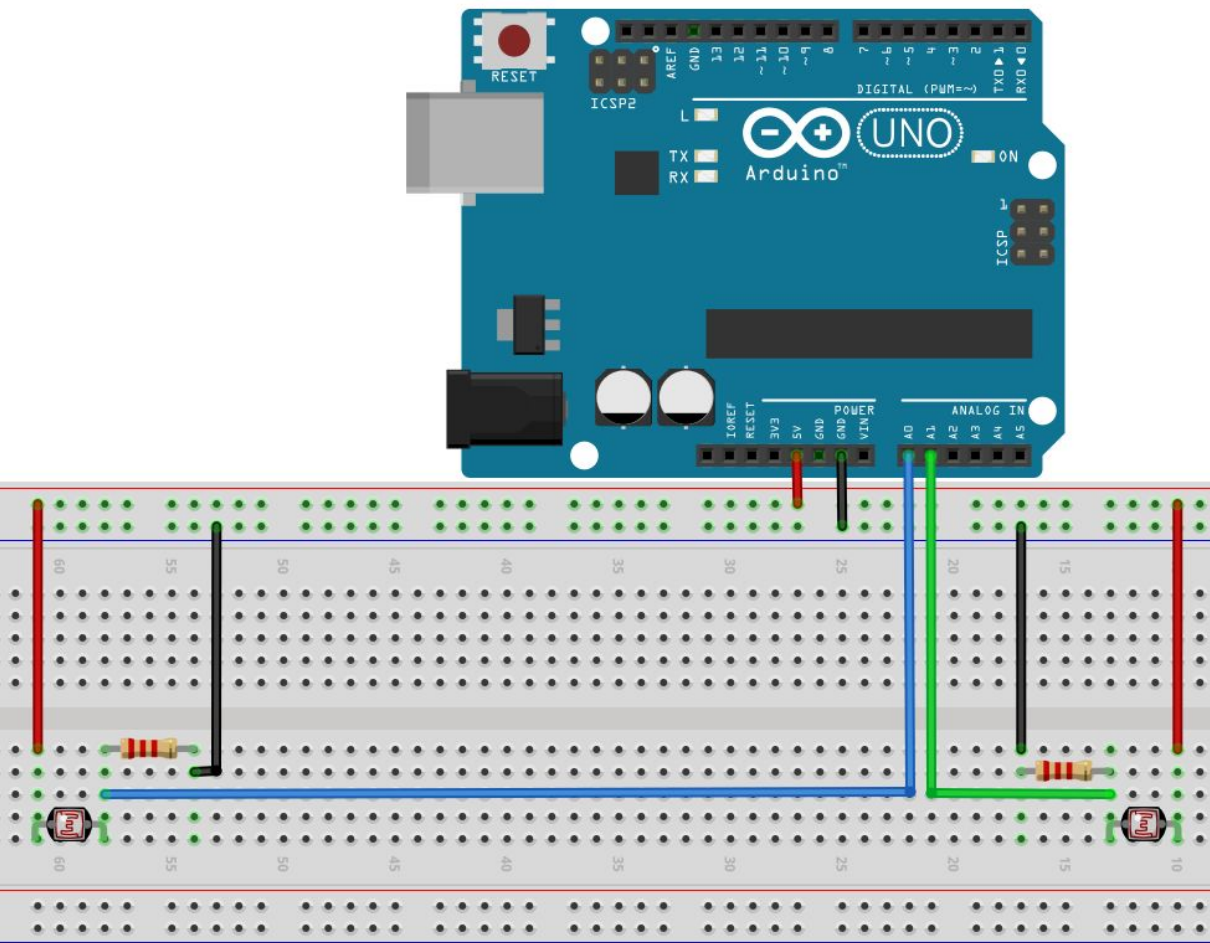
R100

```
int i;  
int led1=5;  
int led2=6;  
int led3=7;  
int tempPin = A0;  
int t=0;  
void setup() {  
  pinMode(led1, OUTPUT);  
  pinMode(led2, OUTPUT);  
  pinMode(led3, OUTPUT);  
  pinMode(tempPin, INPUT);  
  Serial.begin(9600);  
}  
void loop() {  
  tempPin = analogRead(tempPin);  
  Serial.println(t);  
  if(t> 500) {  
    i = random(0,255);  
    analogWrite(led1, i);  
    i = random(0,255);  
    analogWrite(led2, i);  
    i = random(0,255);  
    analogWrite(led3, i);  
    delay(50);  
  } else {  
    analogWrite(led1, 0);  
    analogWrite(led2, 0);  
    analogWrite(led3, 0);  
  }  
}
```

Girasole!

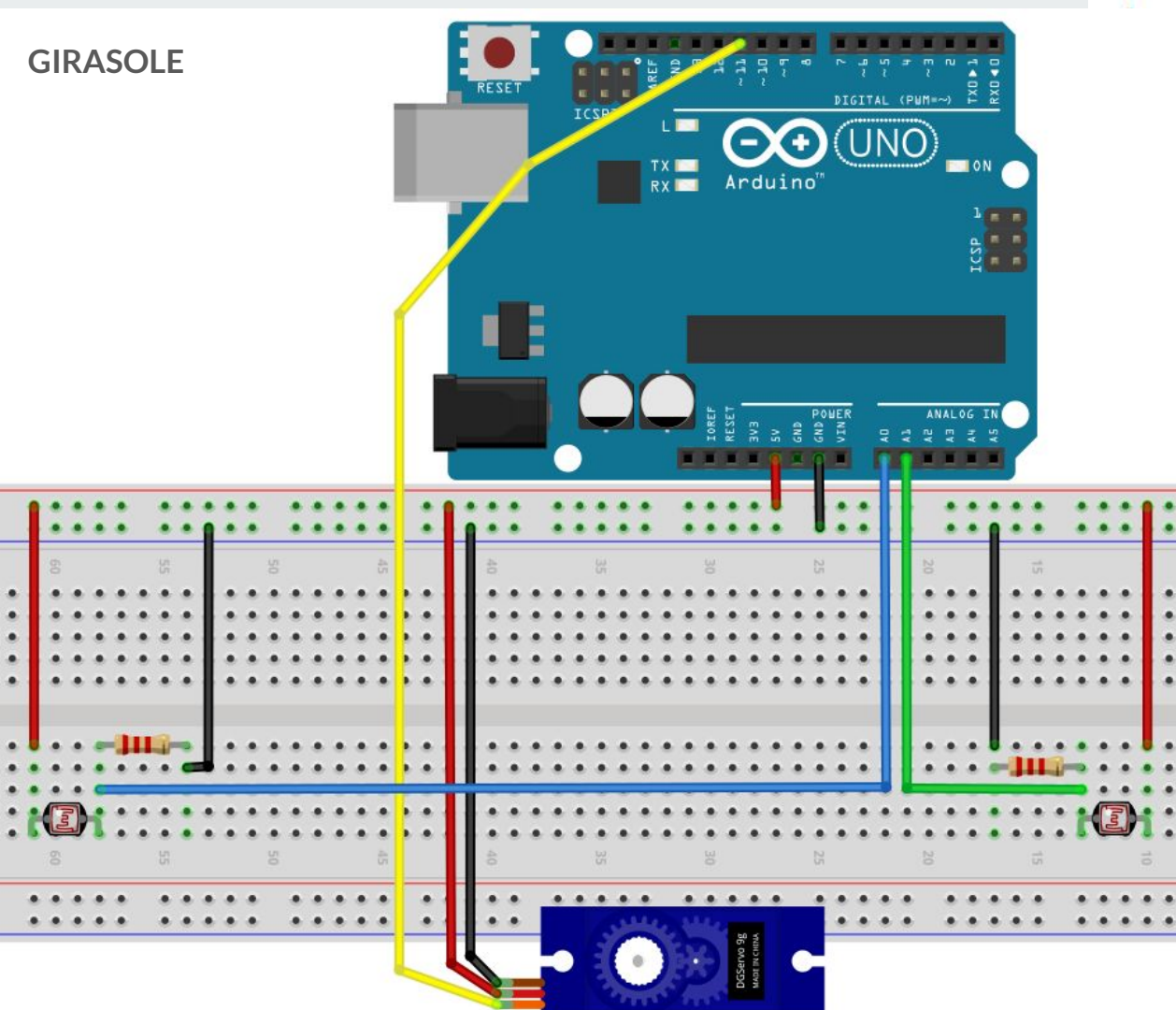
Inseguitore solare con due
photoresistor:





```
1 int sensore1=A0;
2 int sensore2=A1;
3
4 void setup() {
5     pinMode(sensore1,INPUT);
6     pinMode(sensore2,INPUT);
7     Serial.begin(9600);
8 }
9
10 void loop() {
11     int v1 = analogRead(sensore1);
12     int v2 = analogRead(sensore2);
13     Serial.print(v1);
14     Serial.print(",");
15     Serial.println(v2);
16     delay(100);
17 }
```

GIRASOLE



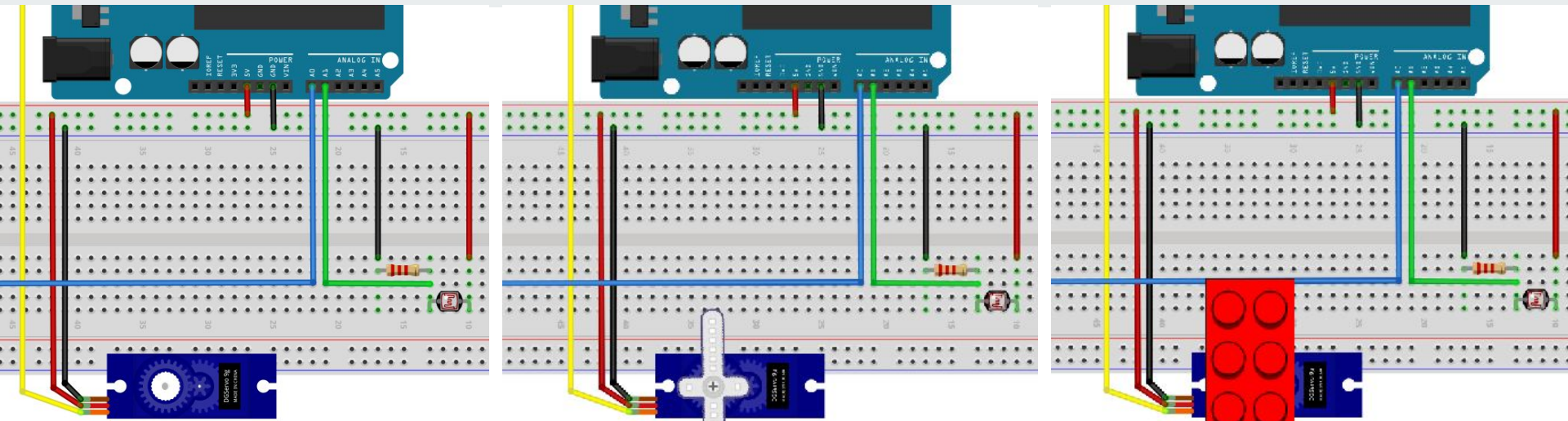
```
#include <Servo.h>
```

```
int sensore1=A0;  
int sensore2=A1;  
Servo girasole;
```

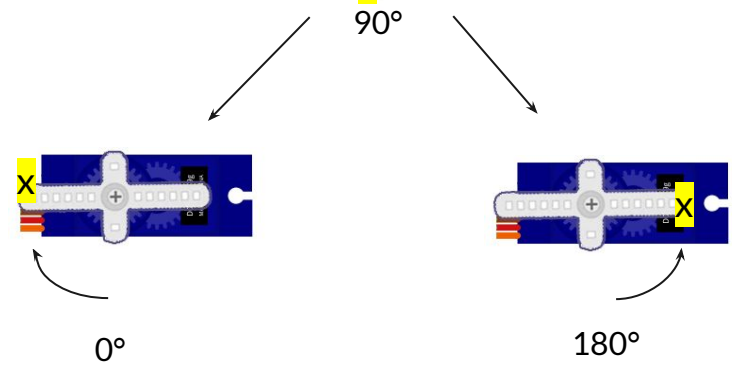
```
void setup() {  
  pinMode(sensore1,INPUT);  
  pinMode(sensore2,INPUT);  
  girasole.attach(11);  
  girasole.write(90);  
}
```

```
void loop() {  
  int v1 = analogRead(sensore1);  
  int v2 = analogRead(sensore2);  
  
  int diff = v1 - v2;  
  if (diff > 300) diff = 300;  
  if (diff < -300) diff = -300;  
  
  int angolo;  
  angolo = map(diff, -300,300,0,180);  
  girasole.write(angolo);  
  delay(500);  
}
```

CALIBRARE MOTORE GIRASOLE



```
1 #include <Servo.h>
2
3 int sensore1=A0;
4 int sensore2=A1;
5 Servo girasole;
6
7 void setup() {
8   pinMode(sensore1,INPUT);
9   pinMode(sensore2,INPUT);
10  girasole.attach(11);
11  girasole.write(90);
12 }
13
14 void loop() {
15
16 }
```





Funzioni

```
delay(1000);  
digitalWrite(3,HIGH);  
pinMode(3,OUTPUT);
```

```
pulseIn(8,HIGH,200000);  
delayMicroseconds(10);  
Serial.print("ciao");  
Serial.println("ciao");
```

```
tone(4,650);  
noTone(4);  
map( valore, min, max, nuovomin, nuovomax);
```

Parentesi graffe e if

```
if(distanza>3000) {  
    noTone(pinBuzzer);  
}
```

```
if(i == 255) i=0;
```

```
if(i == 255) {  
    i=0;  
}
```



Librerie

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

```
Servo girasole;
```

```
girasole.write(90);
```

LED

- + → PIN ARDUINO
- → GND

Con resistenza per non bruciare il led.

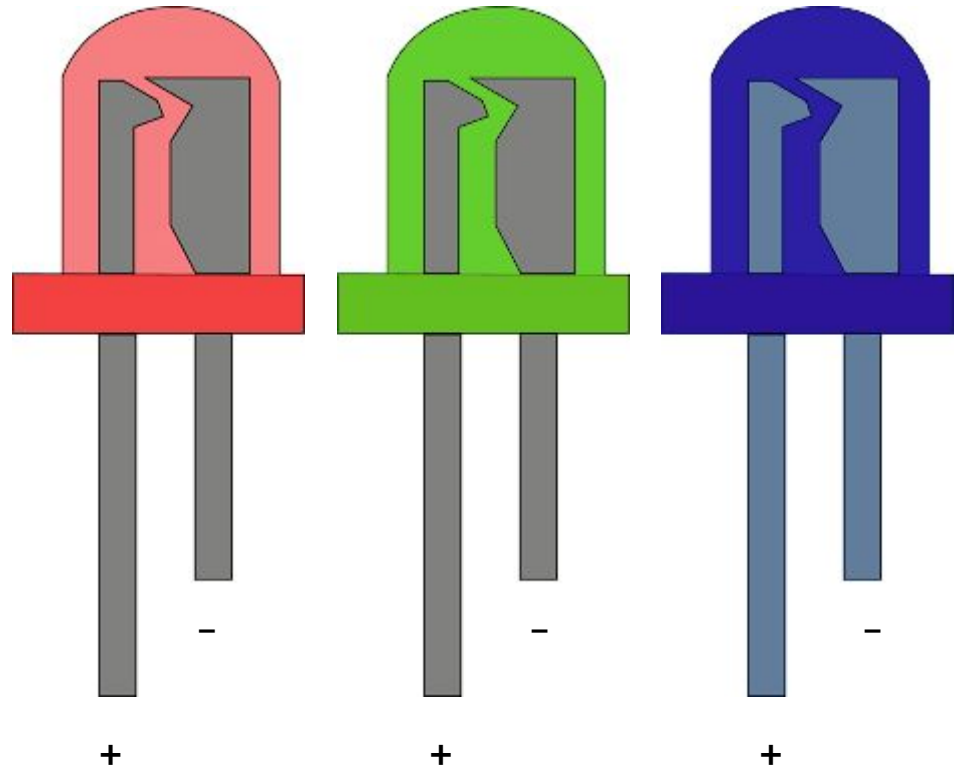
ROSSO ha bisogno di resistenza → ~ 200 ohm

BLU → ~ 200 ohm

VERDE → ~ 200 ohm

BIANCO → ~ 100 ohm

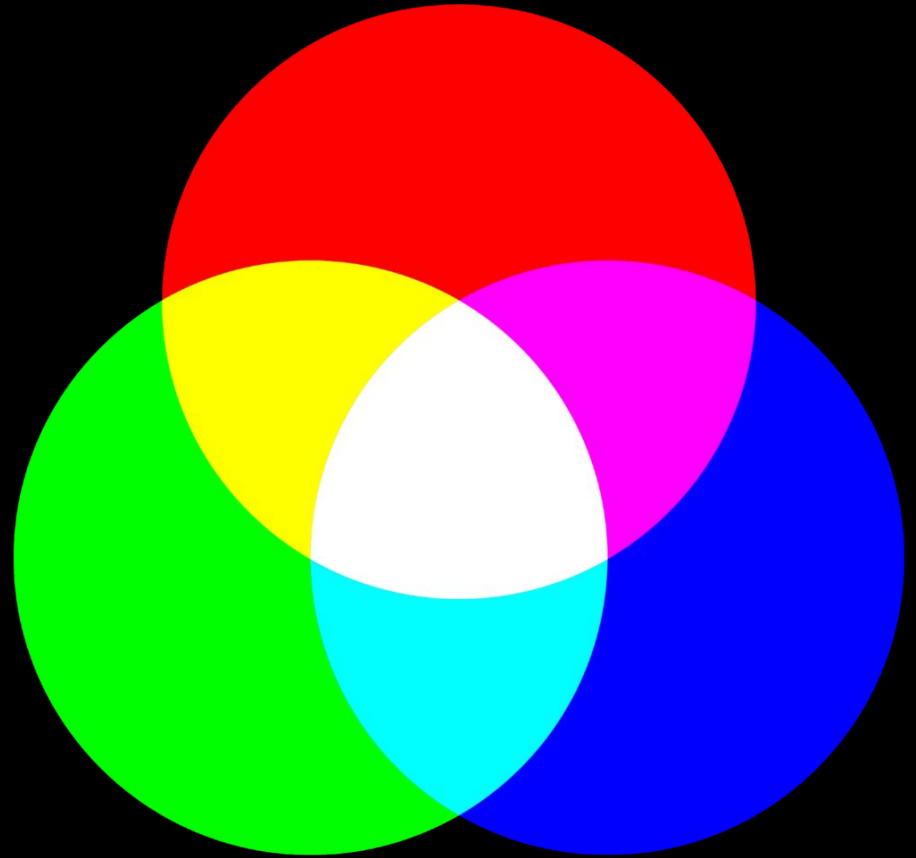
GIALLO → 200

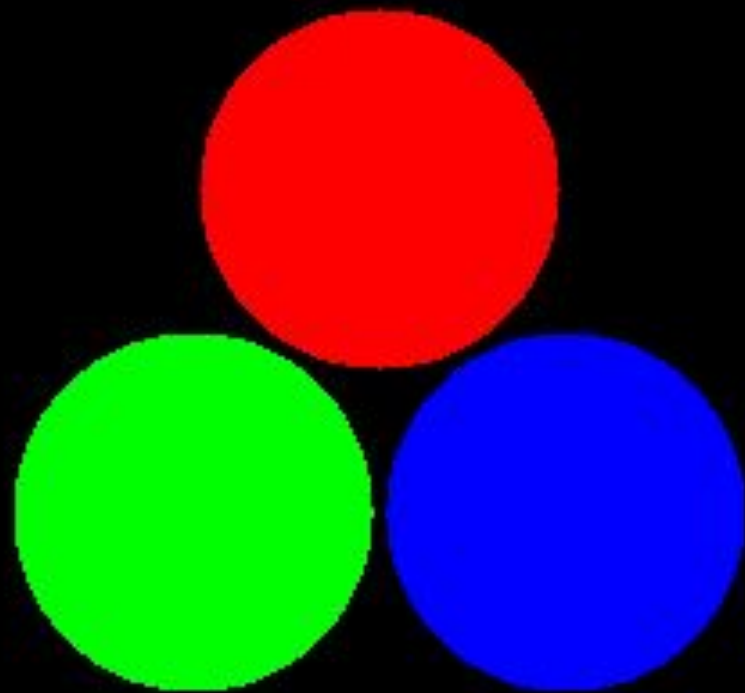




Colori con i LED

La luce si combina diversamente
dai colori stampati









https://www.youtube.com/watch?v=Ra9X_UPfibw

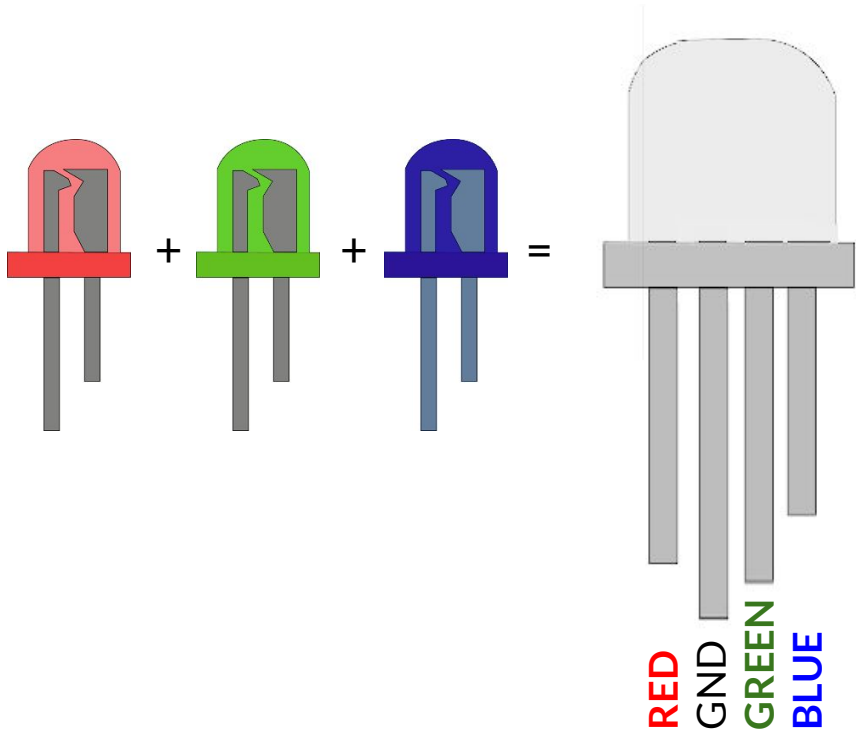
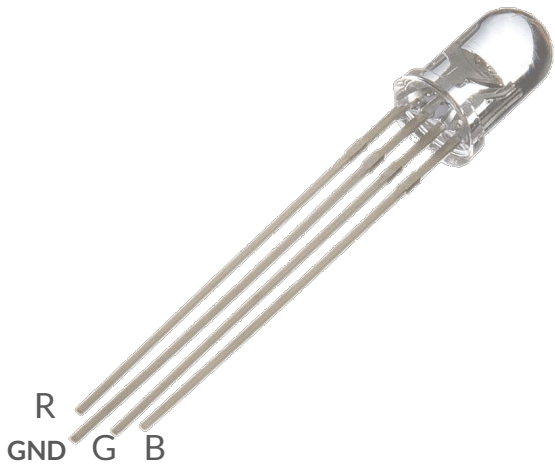
<https://www.youtube.com/watch?v=d-7Xa7Sdbfc>

<https://www.youtube.com/watch?v=JmlHsKxuCF8>





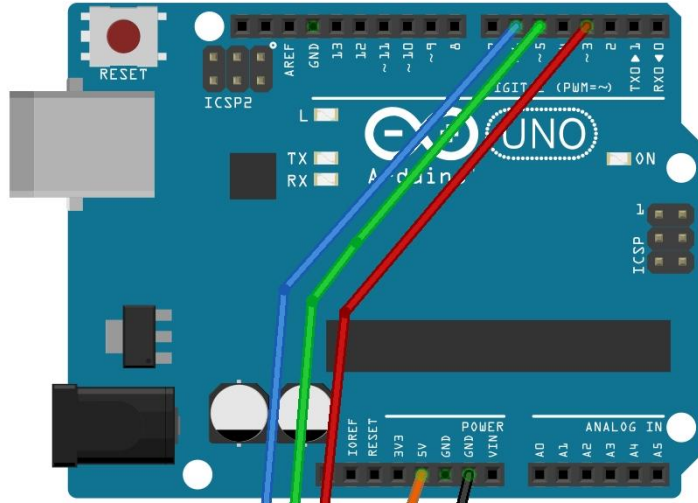
LED RGB: 3 led in 1



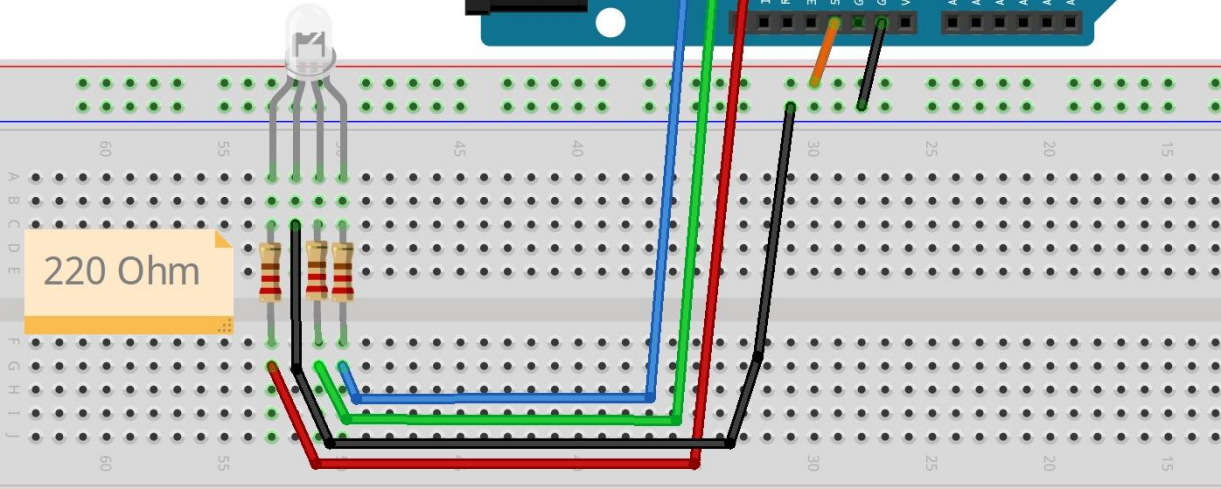
TEST

RGB LED:

- 3 ---> RED
- 5 ---> GREEN
- 6 ---> BLUE



```
1 int redPin = 3;
2 int greenPin = 5;
3 int bluePin = 6;
4
5 void setup() {
6   pinMode(redPin,OUTPUT);
7   pinMode(greenPin,OUTPUT);
8   pinMode(bluePin,OUTPUT);
9   analogWrite(redPin, 255 );
0   analogWrite(greenPin, 0 );
1   analogWrite(bluePin, 0 );
2 }
3
4 void loop() {
5
6 }
```





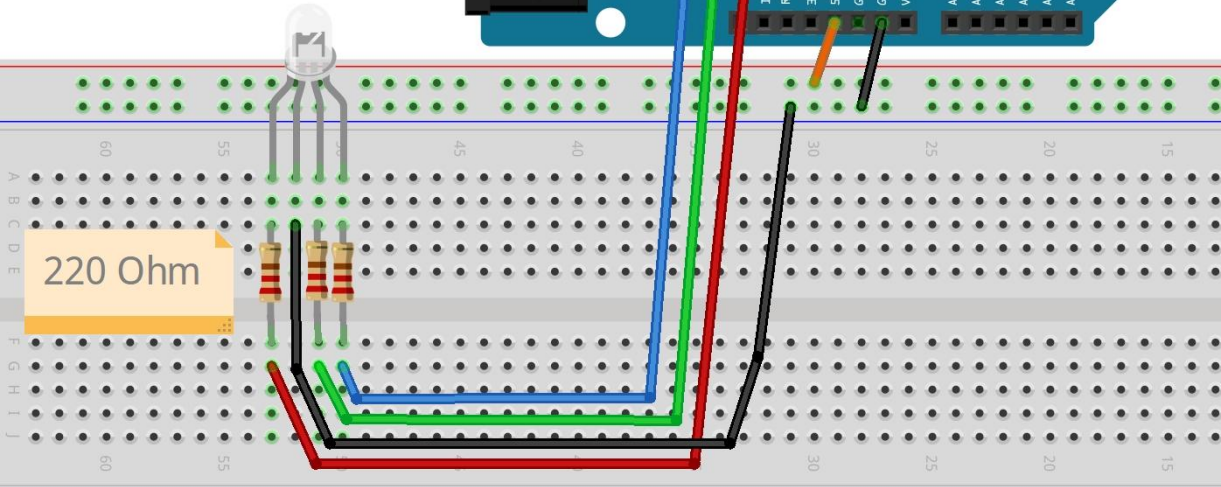
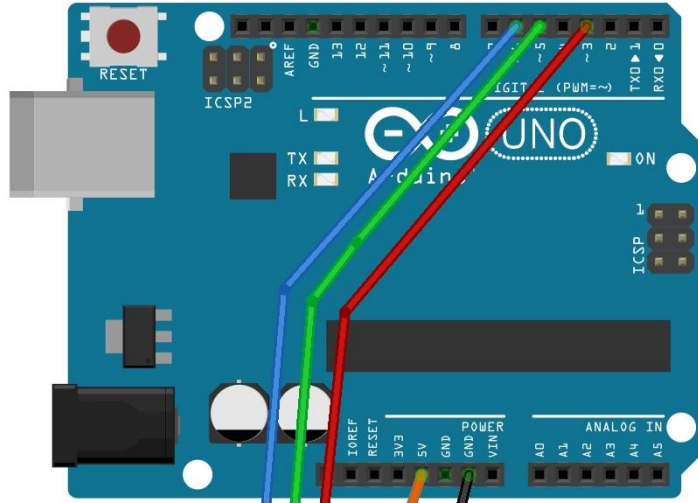
color quiz

<https://prones94.github.io/RGB-Game/>

CASUALI

RGB LED:

- 3 ---> RED
- 5 ---> GREEN
- 6 ---> BLUE



```
1 int redPin = 3;
2 int greenPin = 5;
3 int bluePin = 6;
4 int r=0;
5 int b=0;
6 int g=0;
7
8 void setup() {
9   pinMode(redPin,OUTPUT);
10  pinMode(greenPin,OUTPUT);
11  pinMode(bluePin,OUTPUT);
12  randomSeed(A0);
13 }
14
15 void loop() {
16   r=random(0,255);
17   b=random(0,255);
18   g=random(0,255);
19   analogWrite(redPin,r);
20   analogWrite(greenPin,b);
21   analogWrite(bluePin,g);
22   delay(1000);
23 }
```

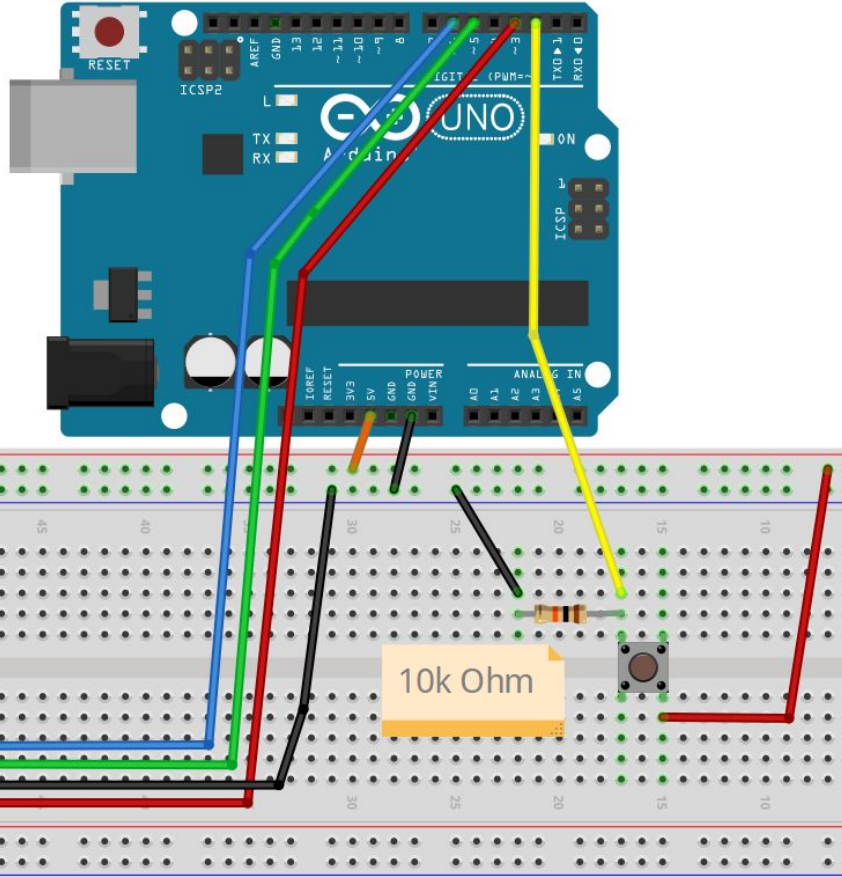

CASUALI + BOTONE

RGB LED:

3 ---> RED

5 ---> GREEN

6 ---> BLUE



```
1 int redPin = 3;
2 int greenPin = 5;
3 int bluePin = 6;
4 int r=0;
5 int b=0;
6 int g=0;
7
8 void setup() {
9   pinMode(2,INPUT);
10  pinMode(redPin,OUTPUT);
11  pinMode(greenPin,OUTPUT);
12  pinMode(bluePin,OUTPUT);
13  randomSeed(A0);
14 }
15
16 void loop() {
17   int i = digitalRead(2);
18   if(i==1) {
19     r=random(0,255);
20     b=random(0,255);
21     g=random(0,255);
22     analogWrite(redPin,r);
23     analogWrite(greenPin,b);
24     analogWrite(bluePin,g);
25     delay(200);
26   }
27 }
```

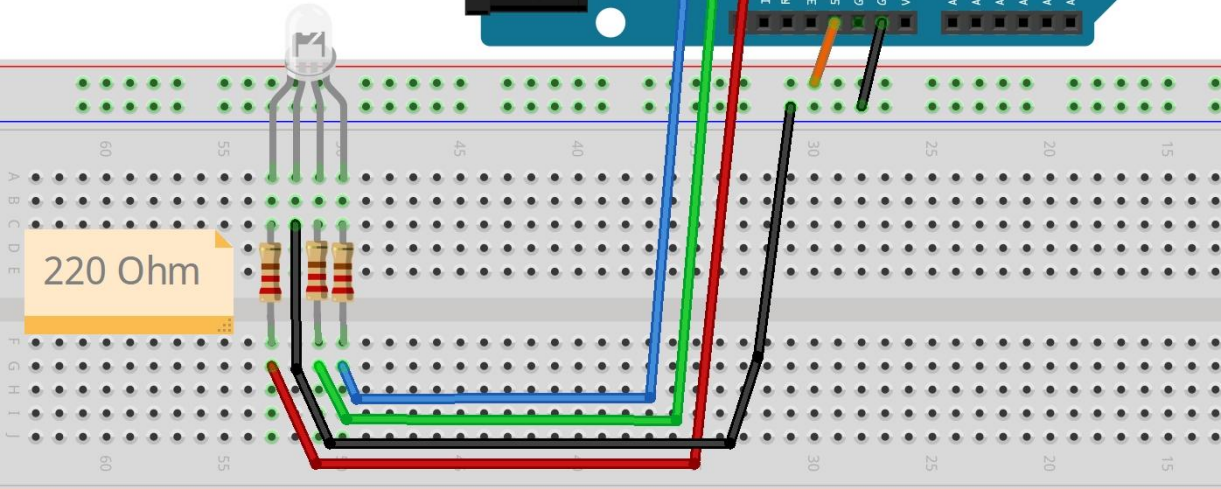
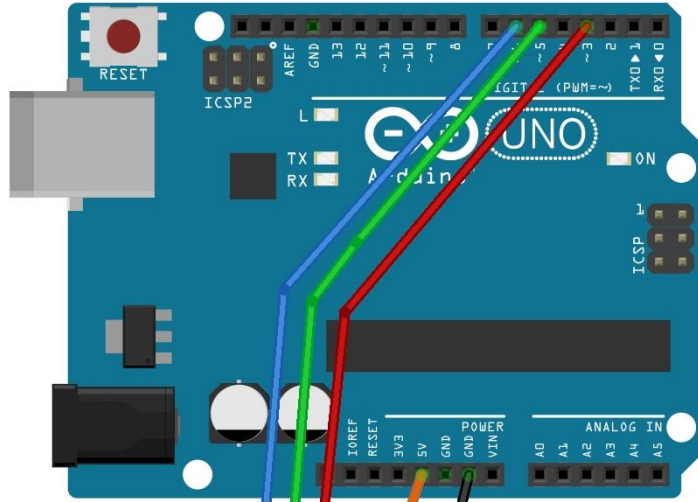
GRADIENTE

RGB LED:

3 ---> RED

5 ---> GREEN

6 ---> BLUE



220 Ohm

```
1 int redPin = 3;
2 int greenPin = 5;
3 int bluePin = 6;
4 int r=0;
5 int b=0;
6 int g=0;
7 int i=1;
8
9 void setup() {
10   pinMode(redPin,OUTPUT);
11   pinMode(greenPin,OUTPUT);
12   pinMode(bluePin,OUTPUT);
13 }
14
15 void loop() {
16   if(i==1) {
17     if(r>0) r=r+i;
18     if(r==0 && b>0) b=b+i;
19     if(b==0 && g>0) g=g+i;
20     if(g==0 && b==0 && r==0) i=1;
21   }
22   if(i==1) {
23     if(r<255) r=r+i;
24     if(r==255 && b<255) b=b+i;
25     if(b==255 && g<255) g=g+i;
26     if(g==255 && b==255 && r==255) i=-1;
27   }
28   analogWrite(redPin,r);
29   analogWrite(greenPin,g);
30   analogWrite(bluePin,b);
31   delay(5);
32 }
```

LUCE + SUONO

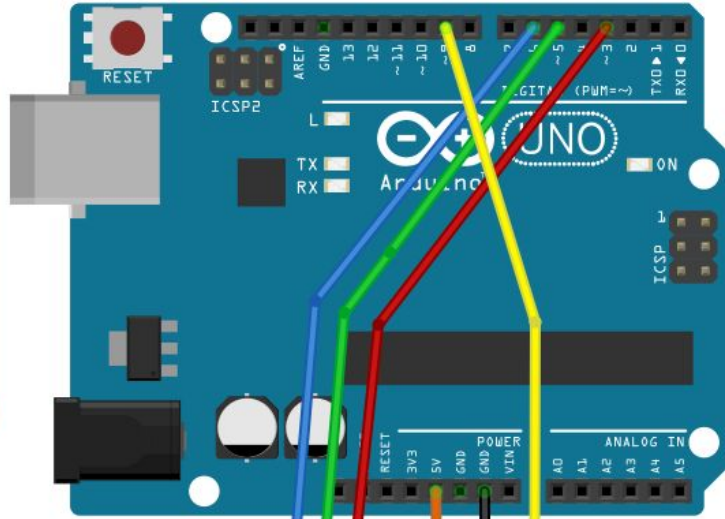
RGB LED:

3 ---> RED

5 ---> GREEN

6 ---> BLUE

9 ---> BUZZER



220 Ohm

```
1  int redPin = 3;
2  int greenPin = 5;
3  int bluePin = 6;
4  int r=0;
5  int b=0;
6  int g=0;
7  int i=1;
8
9  void setup() {
10     pinMode(9,OUTPUT);
11     pinMode(redPin,OUTPUT);
12     pinMode(greenPin,OUTPUT);
13     pinMode(bluePin,OUTPUT);
14 }
15
16 void loop() {
17     if(i==0) {
18         if(r>0) r=r+i;
19         if(r==0 && b>0) b=b+i;
20         if(b==0 && g>0) g=g+i;
21         if(g==0 && b==0 && r==0) i=1;
22     }
23     if(i==1) {
24         if(r<255) r=r+i;
25         if(r==255 && b<255) b=b+i;
26         if(b==255 && g<255) g=g+i;
27         if(g==255 && b==255 && r==255) i=-1;
28     }
29     analogWrite(redPin,r);
30     analogWrite(greenPin,g);
31     analogWrite(bluePin,b);
32     tone(r+g+b);
33     delay(5);
34 }
```



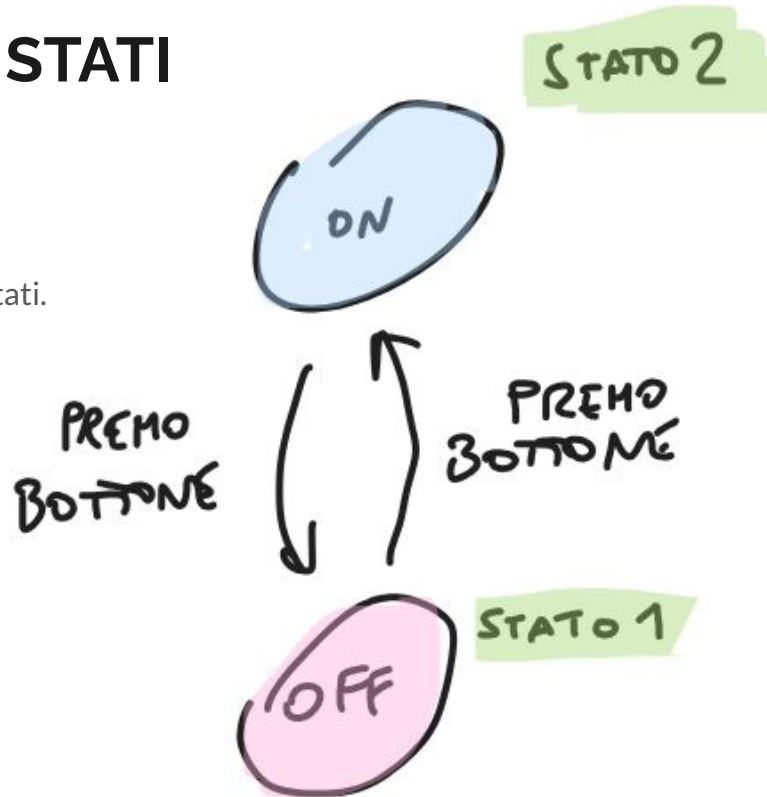
ripasso quiz

https://docs.google.com/spreadsheets/d/1WqZ4jzpSdrx7hdwji59sL8VXTzMxGEH3dEyxk_9HXTc/edit#gid=924062683

MACCHINA A STATI

Esempio macchina a stati:

Lampada accesa e spenta: 2 stati.



uno “STATO” è una condizione o una situazione possibile di un sistema.

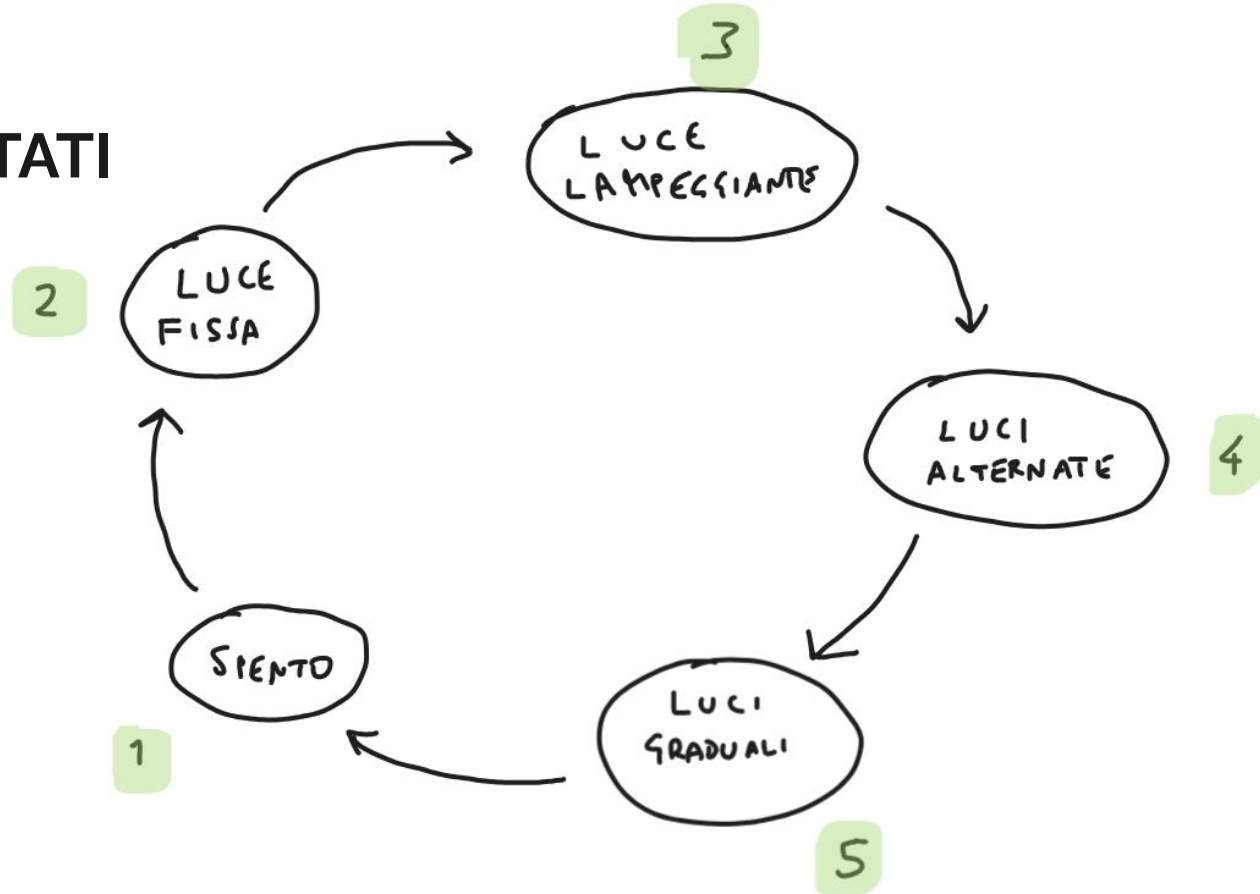
E’ definito attraverso alcune variabili del sistema scelte da chi scrive il codice.

può essere rappresentato con un diagramma in cui ogni “palla” è uno stato

MACCHINA A STATI

Esempio macchina a 5 stati:

Lucine di Natale con bottone che cambia le luci

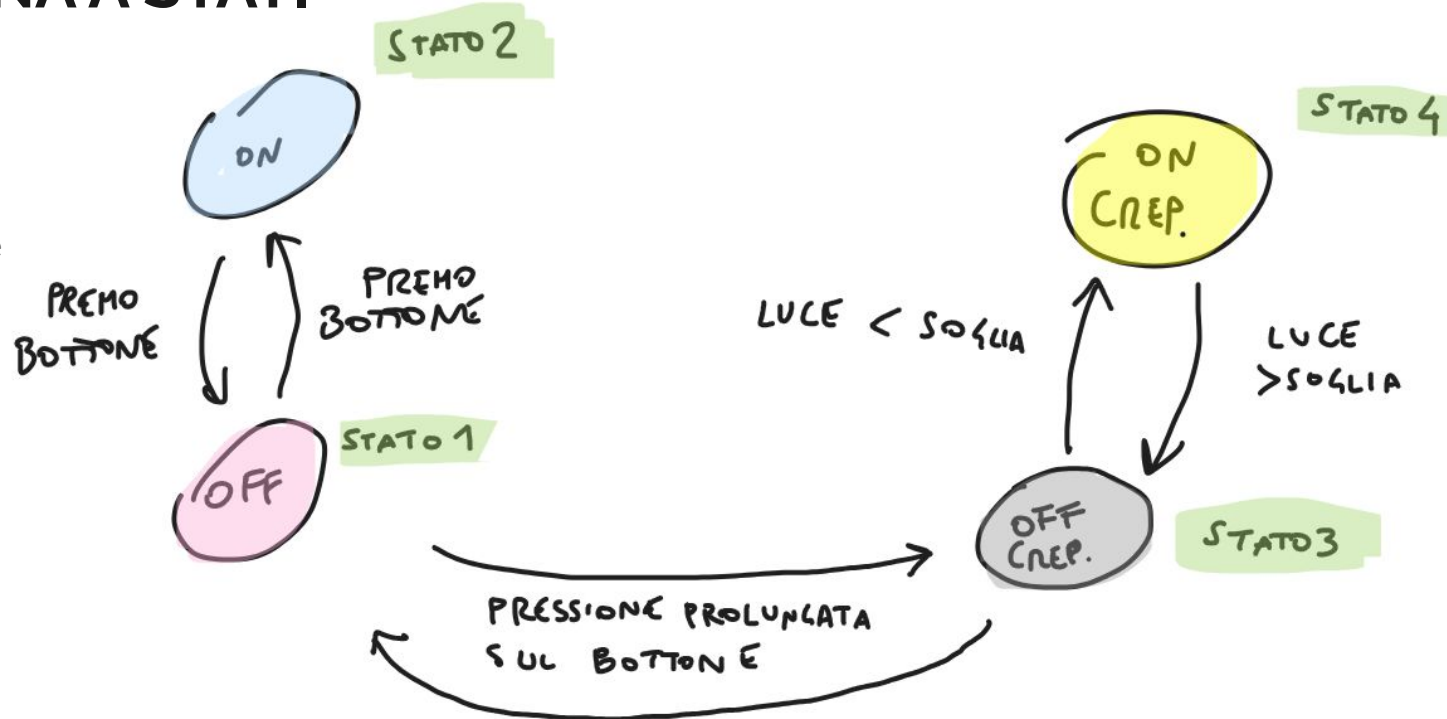


“diagrammi di stato” o “pallogrammi” :-)

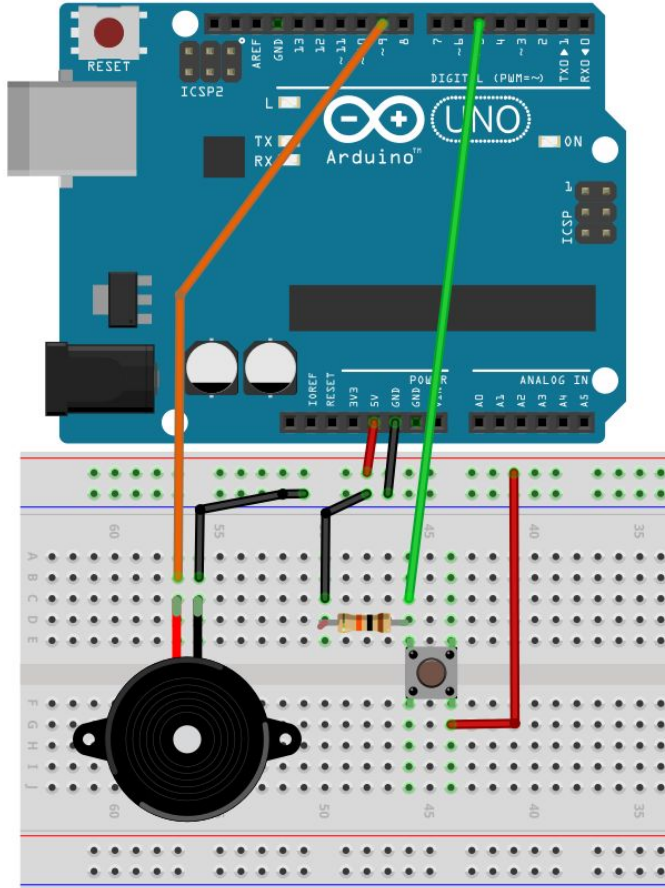
MACCHINA A STATI

Esempio macchina a stati:

Luce crepuscolare che si accende al buio oppure che è usata con interruttore, 4 stati.



BOTTONE



```
1 int pinBottone = 5;
2 int pinBuzzer = 9;
3
4 void setup() {
5     pinMode ( pinBottone, INPUT);
6     pinMode ( pinBuzzer, OUTPUT);
7     Serial.begin(9600);
8 }
9
10 void loop() {
11     int i = digitalRead( pinBottone );
12     if( i == 1 ) {
13         tone( pinBuzzer, 600);
14         delay(150);
15         Serial.println("STATO: premuto");
16     } else {
17         noTone( pinBuzzer);
18         Serial.println("STATO: non premuto");
19     }
20 }
```



IF

```
int i = 3;  
if ( i == 3) {  
    digitalWrite(pinLed, HIGH);  
} else {  
    digitalWrite(pinLed, LOW);  
}
```



IF

```
int i = 2;  
if ( i > 3 ) {  
    digitalWrite(pinLed, HIGH);  
} else {  
    digitalWrite(pinLed, LOW);  
}
```

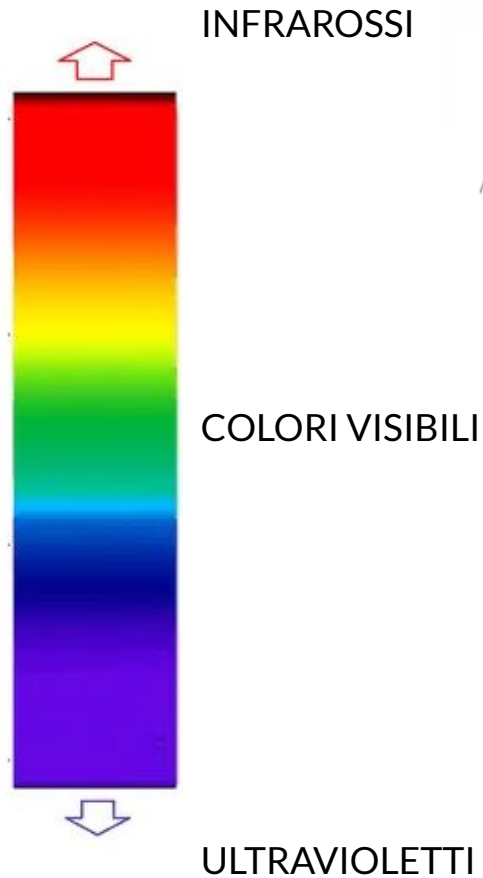
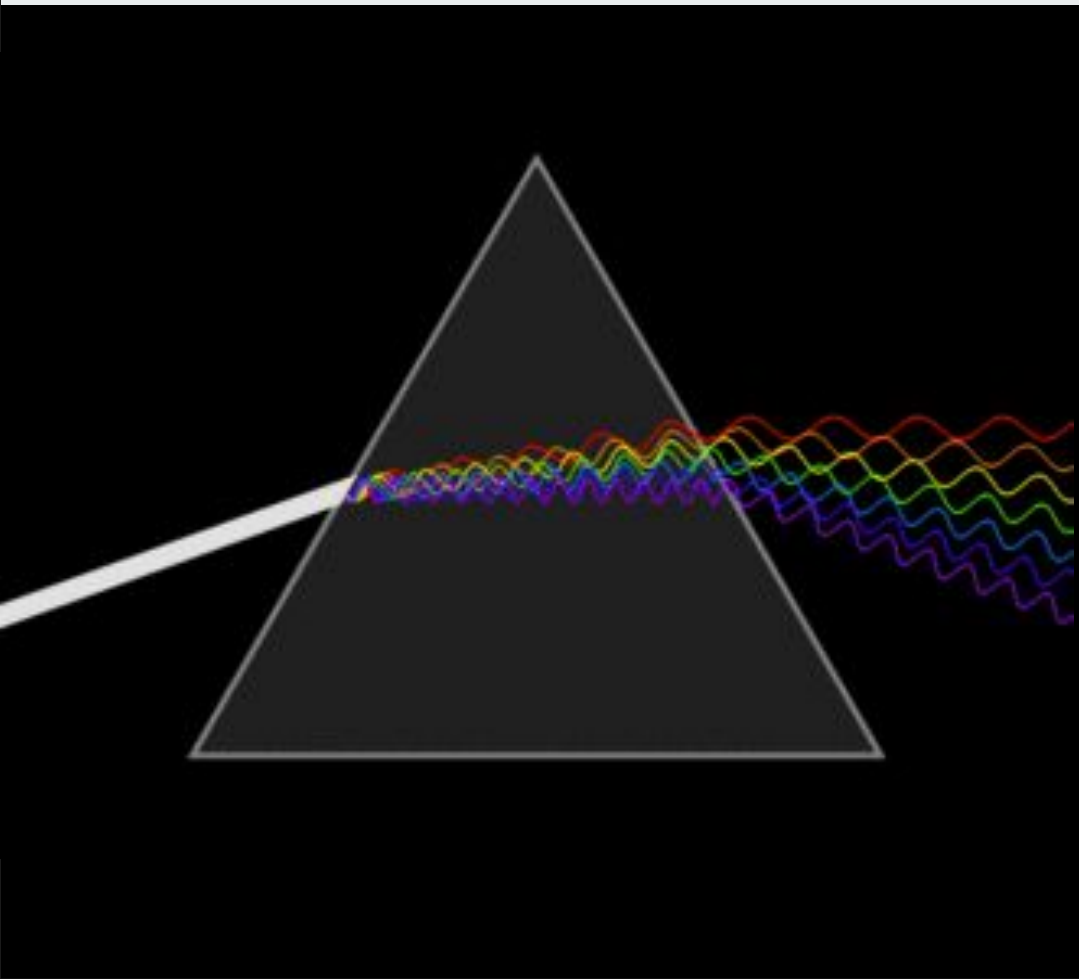
IF

```
int i = 2;
int k = 3;
if ( i == 3 && k==1) {
    digitalWrite(pinLed, HIGH);
} else {
    digitalWrite(pinLed, LOW);
}
```

IF

```
int i = 2;
int k = 3;
if ( i == 3 || k==1) {
    digitalWrite(pinLed, HIGH);
} else {
    digitalWrite(pinLed, LOW);
}
```

Il telecomando spara raggi infrarossi. Il sensore li riceve.



LUCE RGB E TELECOMANDO INFRAROSSO

INFRAROSSO:

GND --> G

5V ---> R

7 ---> Y

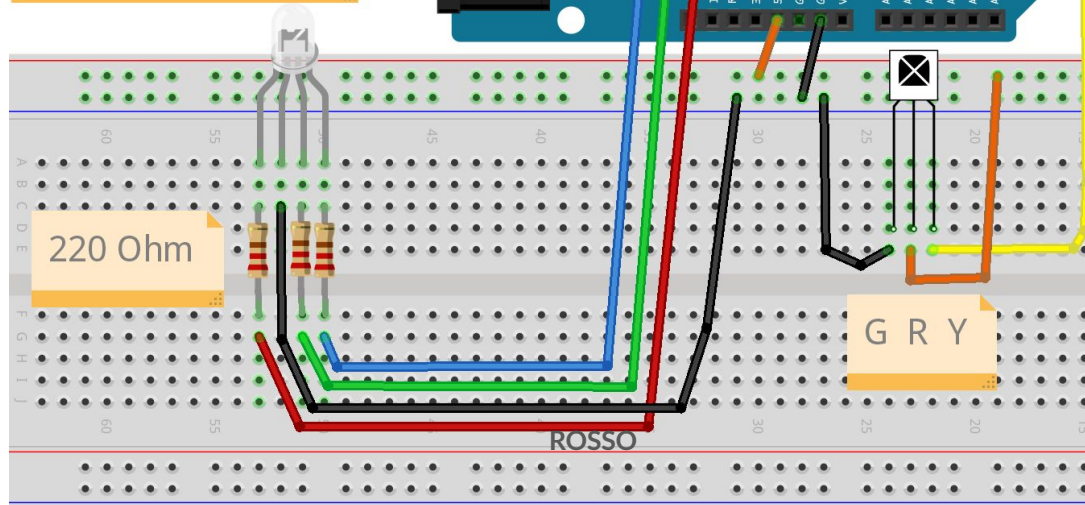
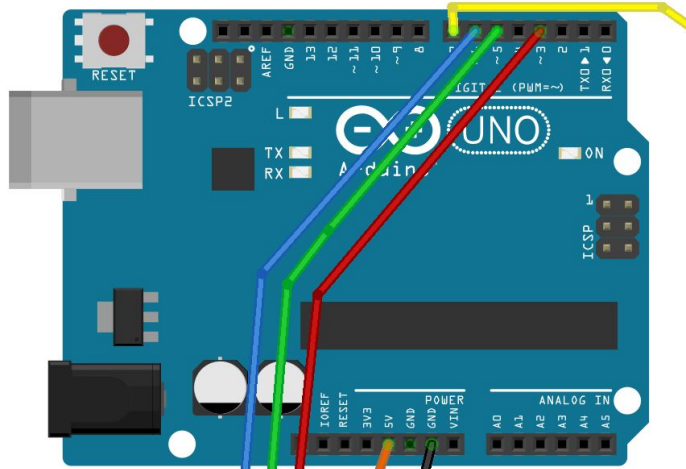
RGB LED:

3 ---> RED

5 ---> GREEN

6 ---> BLUE

YSENS
BLU
VERDE
ROSSO
7 6 5 3



```
1 #include <IRremote.h>
2 IRrecv ricevitore(7);
3
4 int redPin = 3;
5 int greenPin = 5;
6 int bluePin = 6;
7 int tasto = NULL;
8
9 void setup(){
10   pinMode(redPin,OUTPUT);
11   pinMode(greenPin,OUTPUT);
12   pinMode(bluePin,OUTPUT);
13   Serial.begin(9600);
14   ricevitore.start();
15   analogWrite(redPin,0);
16   analogWrite(greenPin,0);
17   analogWrite(bluePin,0);
18 }
19 void loop(){
20   if (ricevitore.decode() == true){
21     Serial.println(ricevitore.decodedIRData.command);
22     if(ricevitore.decodedIRData.command == 12) tasto=1;
23     if(ricevitore.decodedIRData.command == 24) tasto=2;
24     ricevitore.resume();
25   }
26   if(tasto==1) {
27     analogWrite(redPin,255);
28     analogWrite(greenPin,0);
29     analogWrite(bluePin,0);
30   }
31   if(tasto==2) {
32     analogWrite(redPin,0);
33     analogWrite(greenPin,255);
34     analogWrite(bluePin,0);
35   }
36 }
```



Luce Diffrazione





Funzioni

```
delay(1000);  
digitalWrite(3,HIGH);  
pinMode(3,OUTPUT);
```

```
randomSeed(A0);
```

```
random(0,255);
```

Parentesi graffe e if

```
if(tasto==1) {  
    analogWrite(redPin,255);  
    analogWrite(greenPin,0);  
    analogWrite(bluePin,0);  
}
```

```
if(i == 255) i=0;
```

```
if(i == 255) {  
    i=0;  
}
```



Librerie

```
#include <IRremote.h>

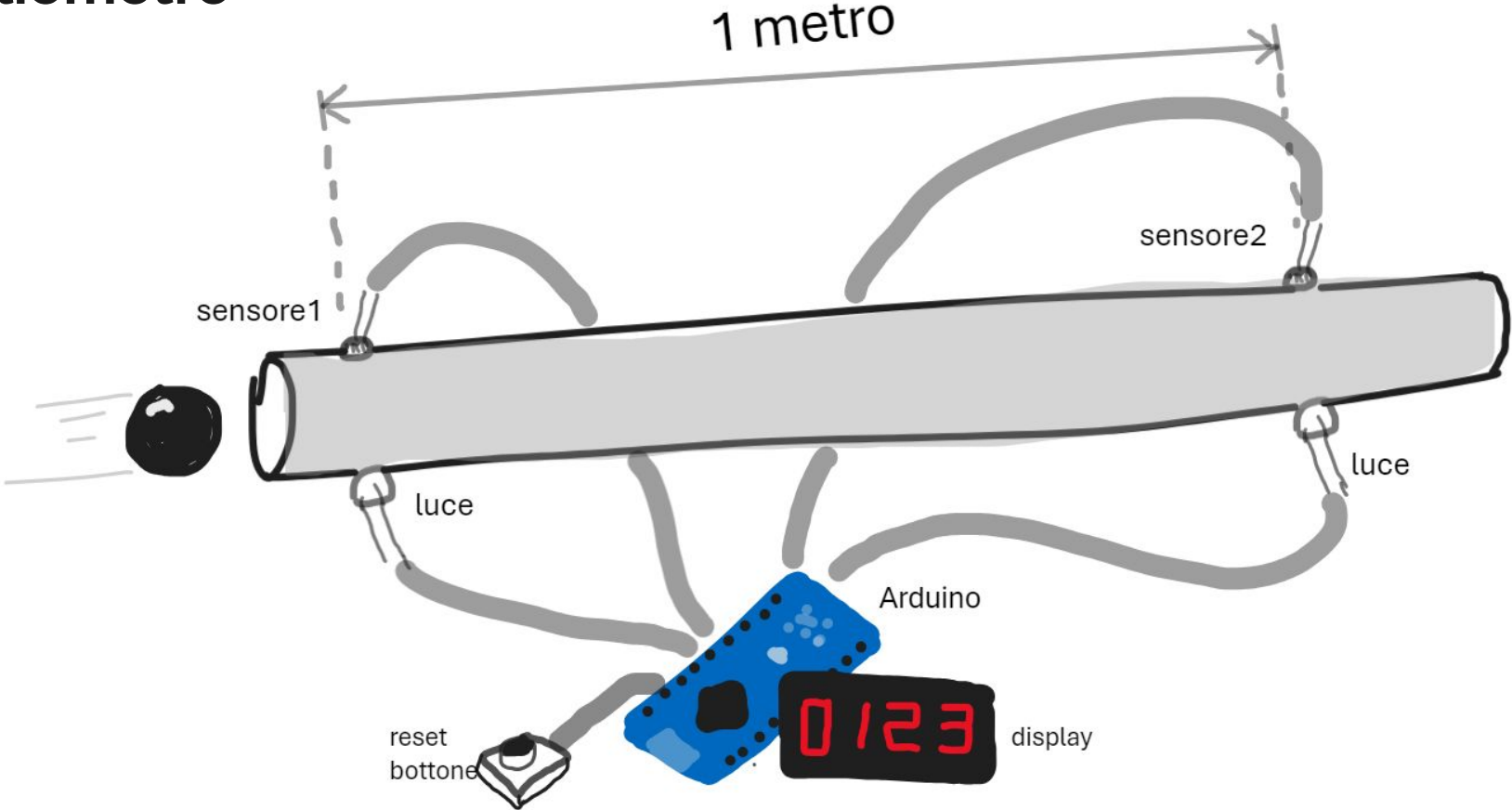
IRrecv ricevitore(7);

if(ricevitore.decode() == true) {

    if(ricevitore.decodedIRData.command == 12) tast=1;
    ricevitore.resume();

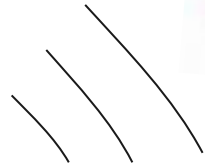
}
```

Bigliometro

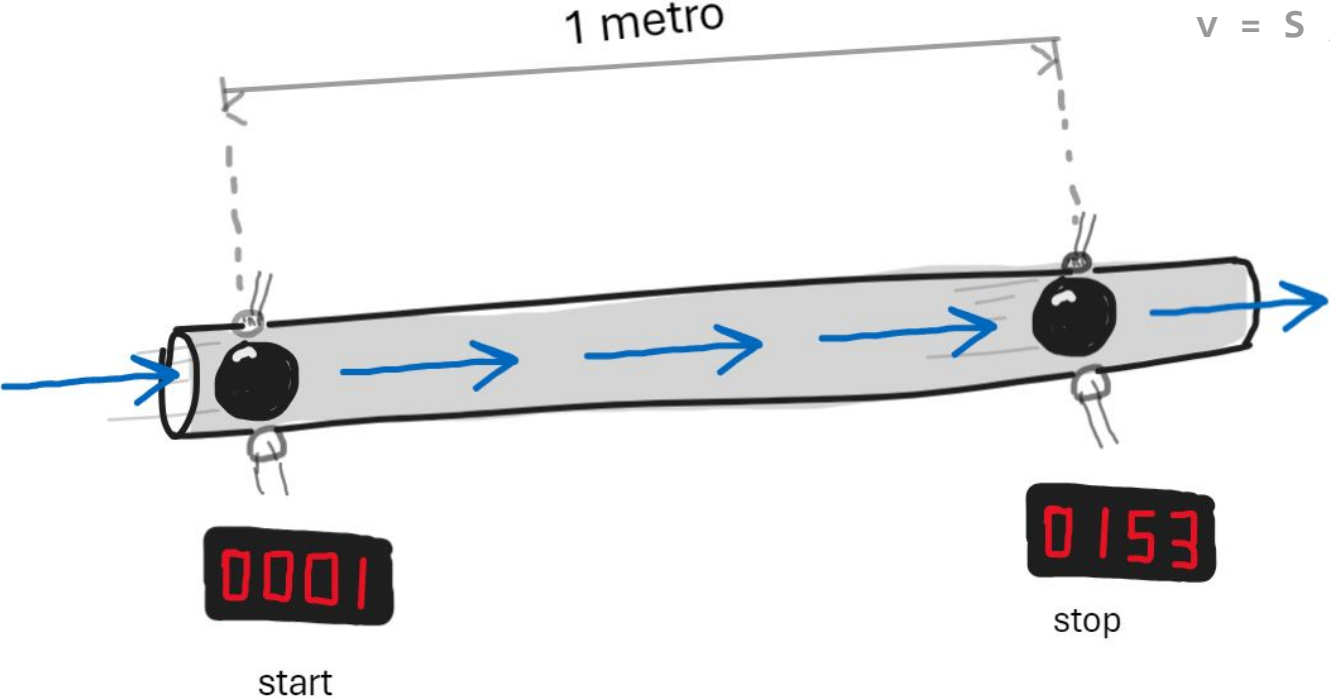




Bigliometro



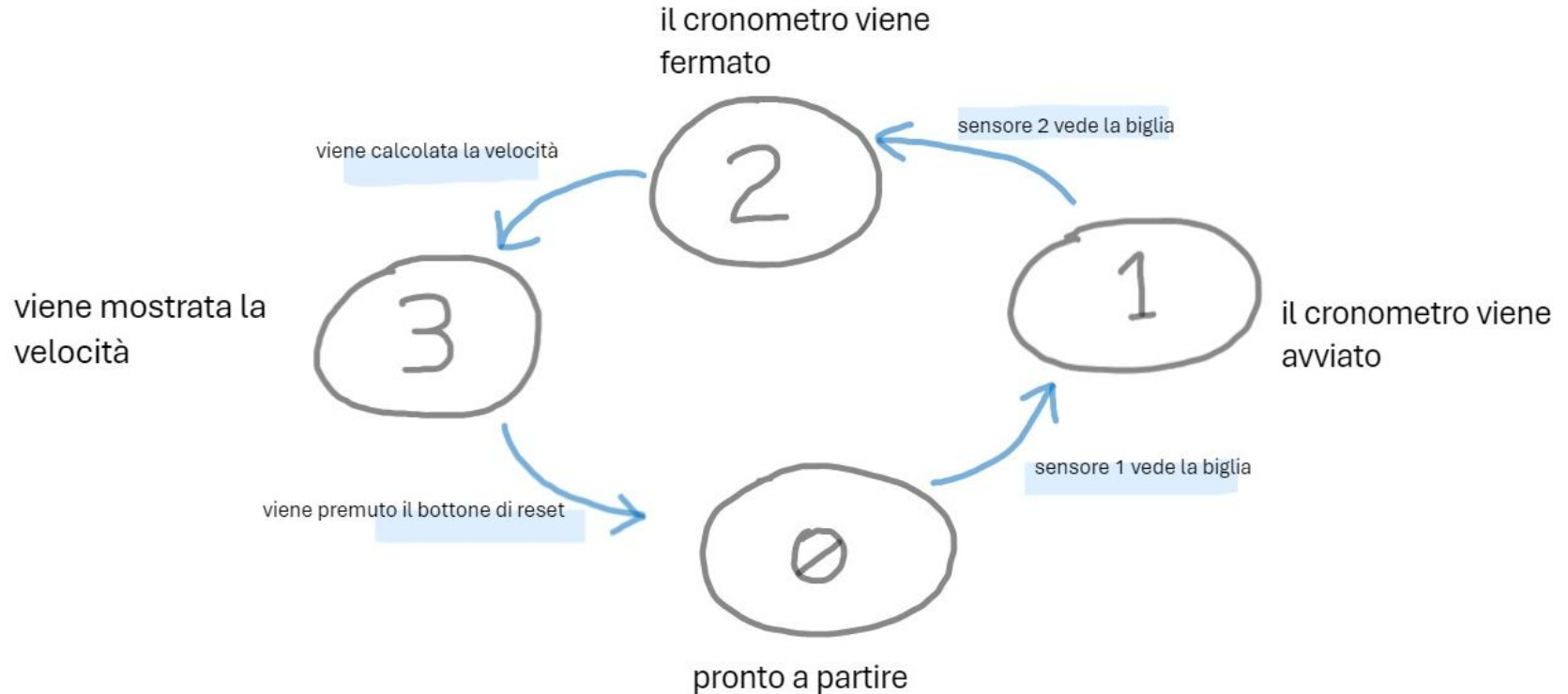
Bigliometro



$$S = v * t$$

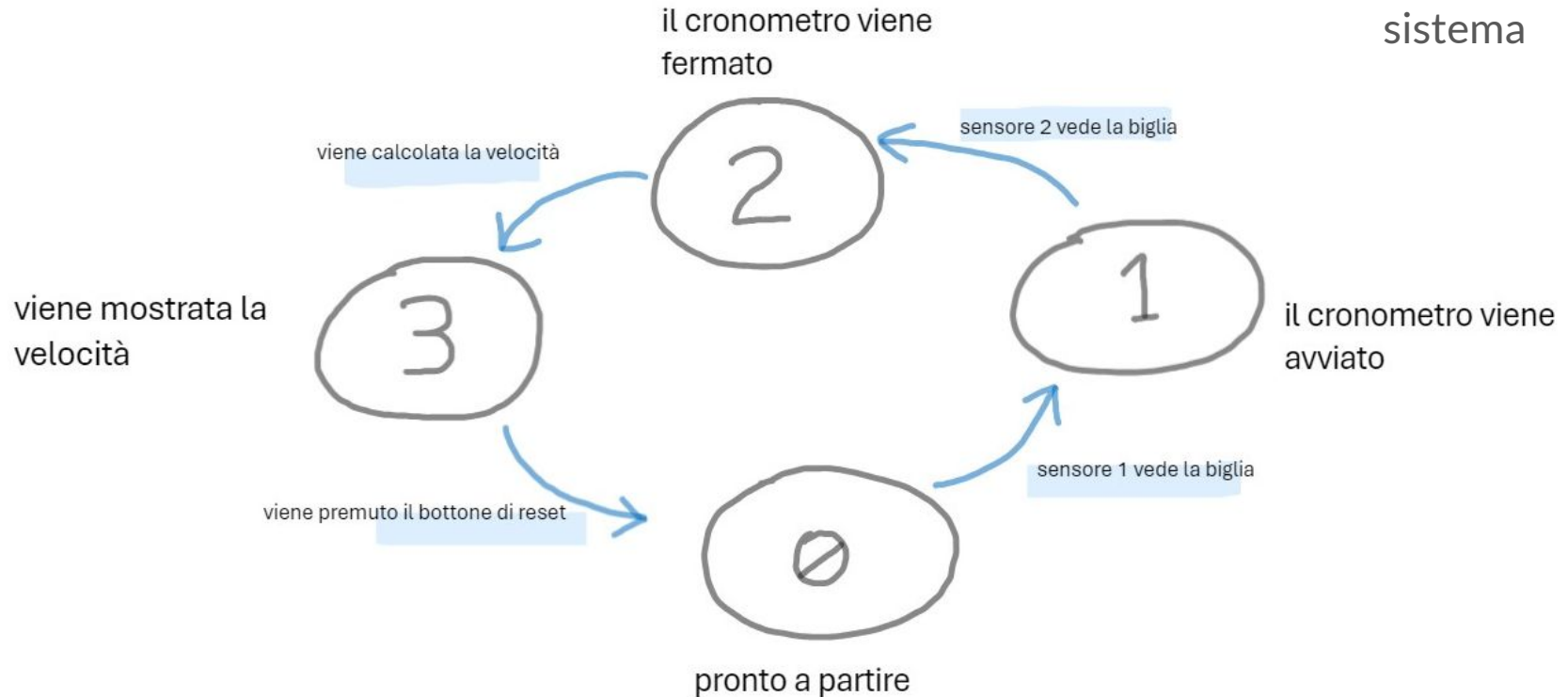
$$v = S / t$$

Bigliometro



Bigliometro

uno "STATO" è
una situazione
possibile del
sistema

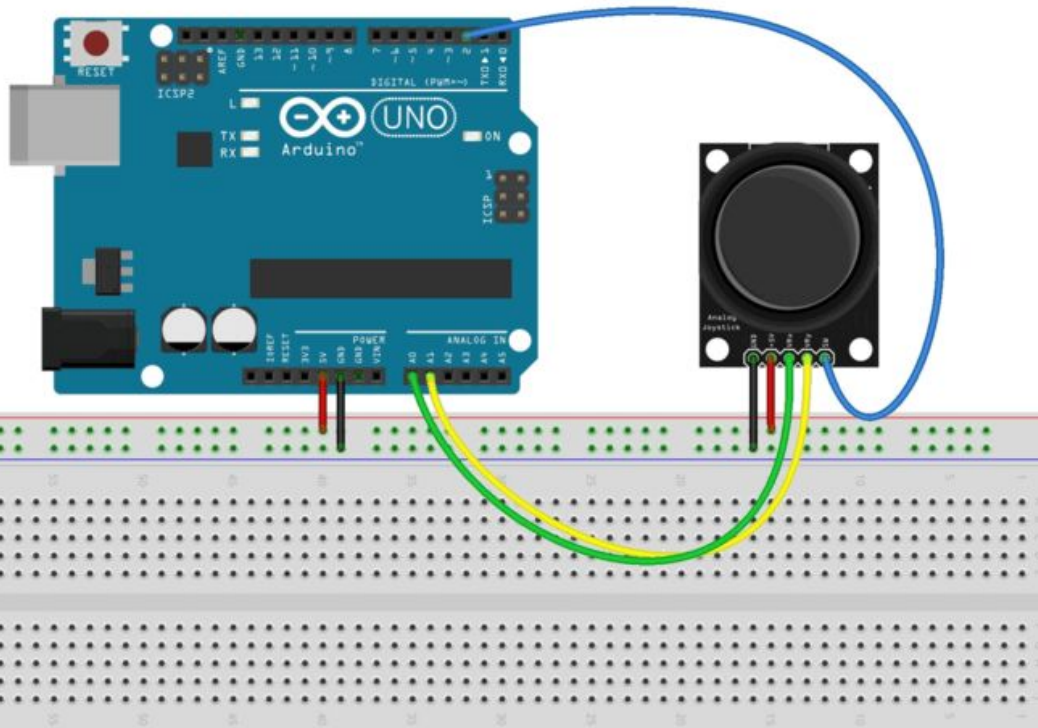




Bigliometro → garetta

https://docs.google.com/spreadsheets/d/1D5Weefi7e3npQ45E2G_JJ5Qj01LifkL9hvj7iC5Cg2E/edit#gid=0

Joystick



```
1  int sx_dx = A0;
2  int su_giu = A1;
3  int bottone = 2;
4
5  void setup(){
6      Serial.begin(9600);
7      pinMode(bottone, INPUT_PULLUP);
8  }
9
10 void loop(){
11     int x = analogRead(sx_dx);
12     int y = analogRead(su_giu);
13     int button = !digitalRead(bottone);
14
15     Serial.print("X:");
16     Serial.print(x);
17     Serial.print(",Y:");
18     Serial.print(y);
19     Serial.print(",B:");
20     Serial.println(button);
21
22     delay(100);
23 }
```

joystick_0.ino

```
1 int sx_dx = A0;
2 int su_giu = A1;
3 int bottone = 2;
4
5 void setup(){
6   Serial.begin(9600);
7   pinMode(bottone, INPUT_PULLUP);
8 }
9
10 void loop(){
11   int x = analogRead(sx_dx);
12   int y = analogRead(su_giu);
13   int button = !digitalRead(bottone);
14
15   Serial.print("X:");
16   Serial.print(x);
17   Serial.print(",Y:");
18   Serial.print(y);
19   Serial.print(",B:");
20   Serial.println(button);
21
22   delay(100);
```

Output Serial Monitor x

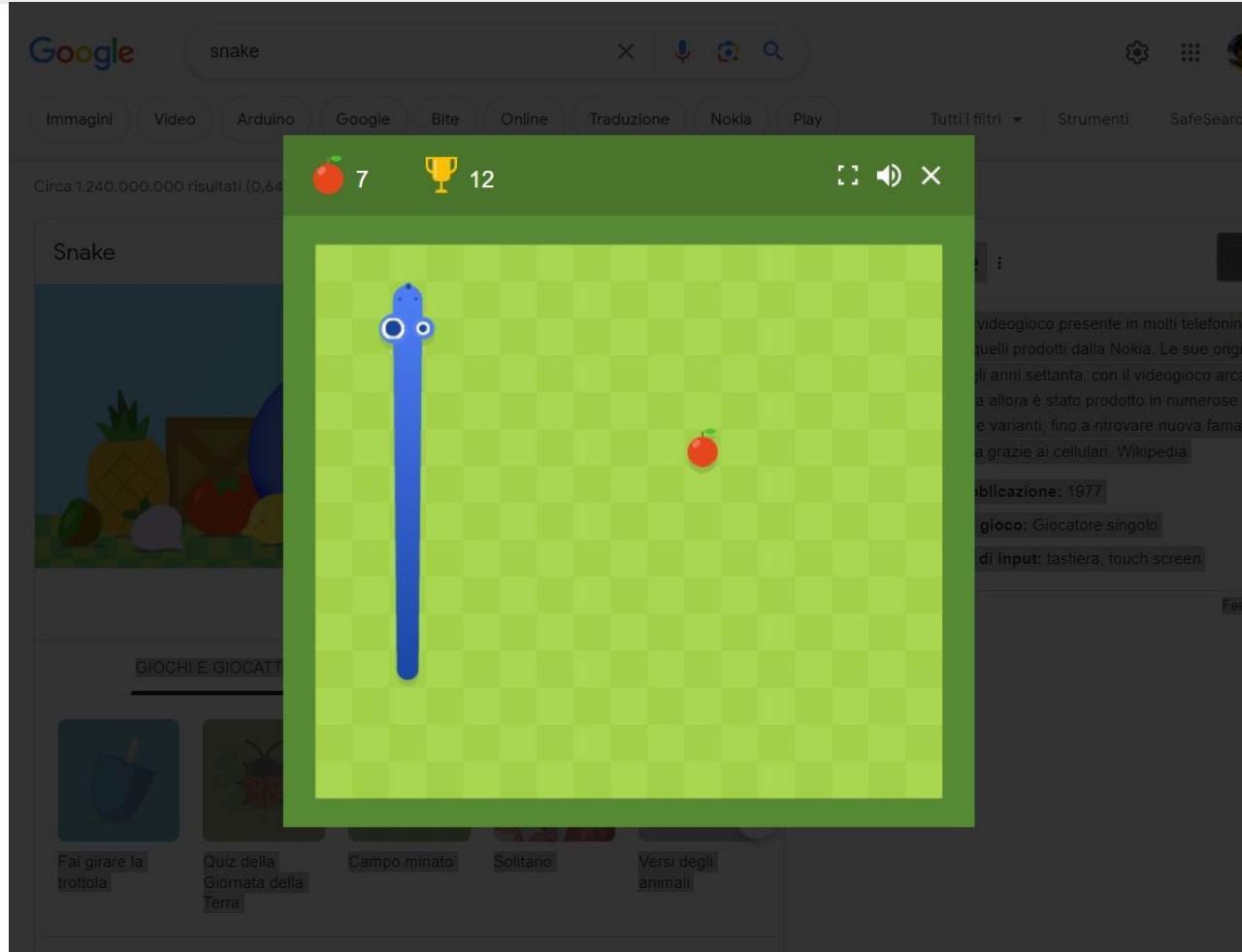
Message (Enter to send message to 'Arduino Uno' on

```
23:16:38.914 -> X:519,Y:511,B:0
23:16:38.999 -> X:519,Y:512,B:0
23:16:39.102 -> X:519,Y:512,B:0
23:16:39.243 -> X:519,Y:512,B:0
23:16:39.306 -> X:519,Y:512,B:0
23:16:39.447 -> X:519,Y:512,B:0
23:16:39.526 -> X:519,Y:511,B:0
23:16:39.601 -> X:519,Y:512,B:0
```

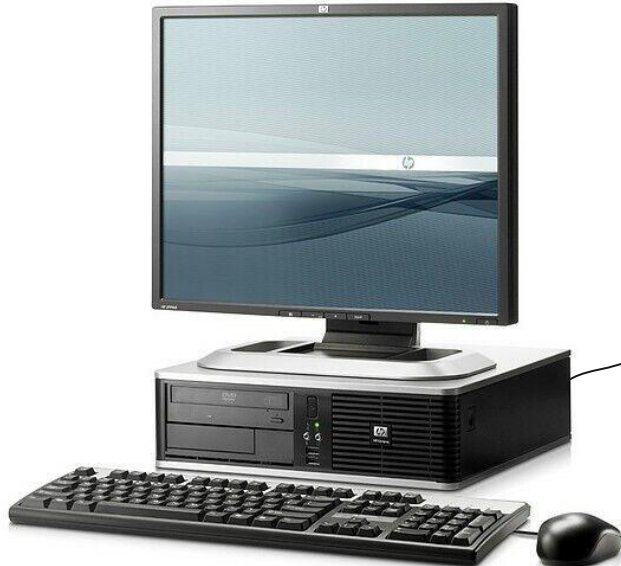
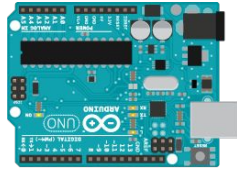


Joystick

Arduino non è visto dal computer come una tastiera o un mouse o un joystick, quindi non possiamo usarlo direttamente come un gamepad.



Joystick



Configure

Port: COM3 Baud: 9600

OK

Cancel

Start Up

Run at Start Up

Automatically Log In as:

User: _____

Password: _____

Extended Character Interpretation

ASCII Extended e.g. DeltaTalker

ANSI Extended

Instructions

Select the port you wish to use then set the Baud to match your communications device.

Check 'Run at Start Up' if you want this program to run automatically each time you start your computer or log in.

Check 'Automatically Log In' and enter a user name and password to automatically log into your system. (This may not be available on your computer.)

Select either ASCII or ANSI. Most newer devices will use ANSI.

Select OK to use the new values or Cancel to quit without making any changes.

AAC Keys è un programma che riceve comandi sulla porta seriale e li traduce in sequenze di tasti premuti sulla tastiera.

Ci permette di simulare una tastiera con Arduino!

<https://aacinstitute.org/aac-keys/>

E' un programma gratuito, realizzato per aiutare le persone affette da disabilità che hanno bisogno di sistemi e interfacce personalizzate per usare il computer.

AAC Keys - Comandi speciali

Basic Mouse Commands

Single Click	<esc>,click.
Double Click	<esc>,dblclick.
Right Click	<esc>,click,right.

Emulator Command	Mouse Cursor Movement You See on the Screen
<esc>,move,+5,0.	Moves the mouse cursor 5 pixels to the right.
<esc>,move,-5,0.	Moves the mouse cursor 5 pixels to the left
<esc>,move,0,+5.	Moves the mouse cursor 5 pixels down
<esc>,move,0,-5.	Moves the mouse cursor 5 pixels up
<esc>,move,+10,+10.	Moves the mouse cursor 10 pixels diagonally downward and right
<esc>,move,-10,+10.	Moves the mouse cursor 10 pixels diagonally downward and left
<esc>,move,-10,-10.	Moves the mouse cursor 10 pixels diagonally upward and left
<esc>,move,+10,-10.	Moves the mouse cursor 10 pixels diagonally upward and right
<esc>,moureset.	Resets the mouse and sends it to the upper left corner of the screen
<esc>,goto,+100,+150.	Moves the mouse to a specific location on the screen.

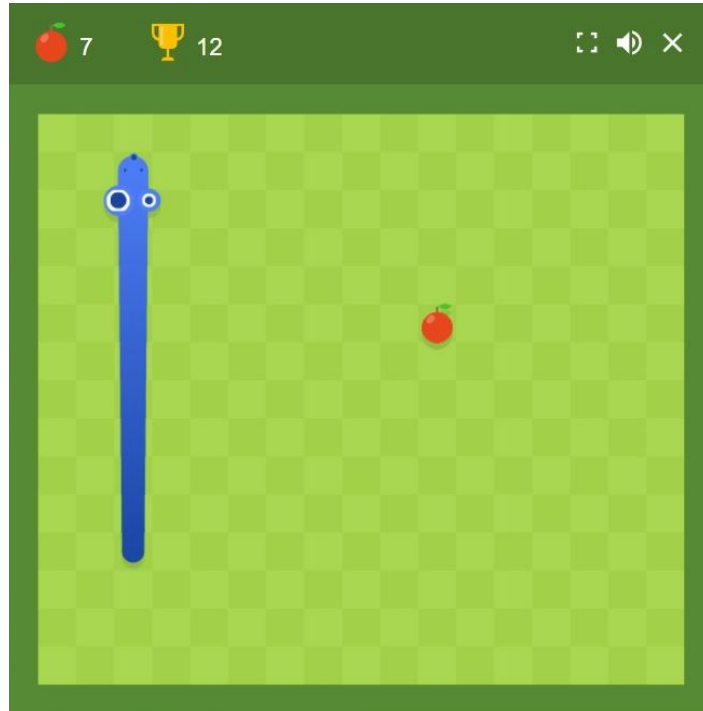
Keyboard commands:

Return	<esc>,return.
Backspace	<esc>,backspace.
Tab	<esc>,tab.
Escape/Cancel	<esc>,esc.
Print (IBM) Print (Mac)	<esc>,hold,ctrl.p <esc>,hold,command,p.
Save (IBM)	<esc>,hold,ctrl.s <esc>,hold,command.s
Left Arrow key	<esc>,left.
Right Arrow key	<esc>,right.
Up Arrow key	<esc>,up.
Down Arrow key	<esc>,down.
Page Up	<esc>,pageup.
Page Down	<esc>,pagedown.

Joystick con AAC Keys

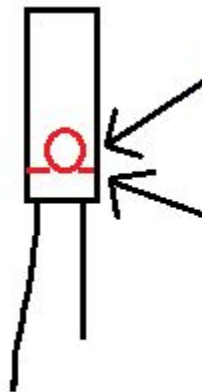
1. caricare il programma su Arduino
2. avviare AAC
3. Andare su Google e cercare "Snake"
4. Giocare

(Se si vuole caricare ancora un altro programma bisogna disattivare AAC)



```
1 int sx_dx = A0;
2 int su_giu = A1;
3 int bottone = 2;
4
5 void setup(){
6   Serial.begin(9600);
7   pinMode(bottone, INPUT_PULLUP);
8 }
9
10 void loop(){
11   int x = analogRead(sx_dx);
12   int y = analogRead(su_giu);
13   int button = !digitalRead(bottone);
14
15   if (x==0) {
16     Serial.print((char)27);
17     Serial.print("right.");
18   }
19   if (x==1023) {
20     Serial.print((char)27);
21     Serial.print("left.");
22   }
23   if (y==0) {
24     Serial.print((char)27);
25     Serial.print("up.");
26   }
27   if (y==1023) {
28     Serial.print((char)27);
29     Serial.print("down.");
30   }
31   if (button==1) {
32     Serial.print(" ");
33   }
34   delay(100);
35 }
```

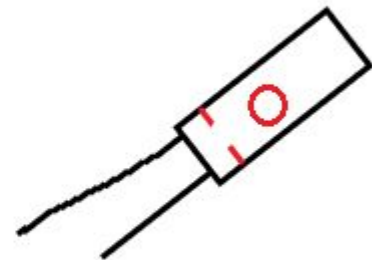
Tilt sensor / Sensore di inclinazione / Come funziona



una piccola sfera, metallica si muove all'interno del cilindro e quando tocca le parti conduttive si chiude il circuito

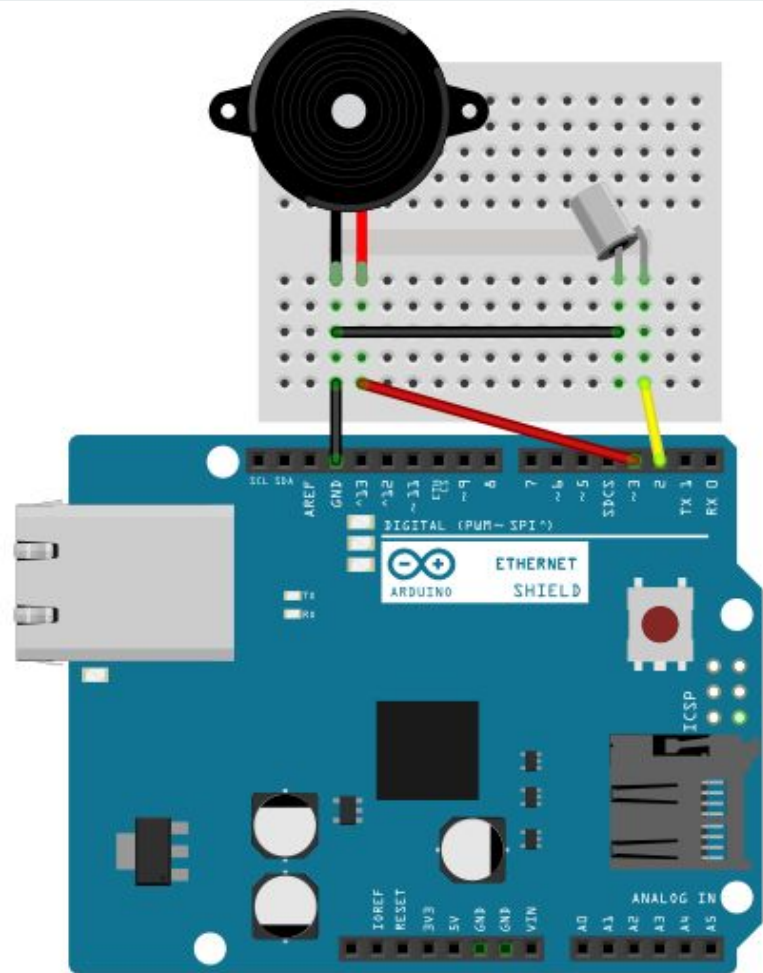
conduttori metallici

circuito chiuso, la palla completa il circuito e fa da interruttore



circuito aperto, la palla è staccata dai conduttori

Tilt switch

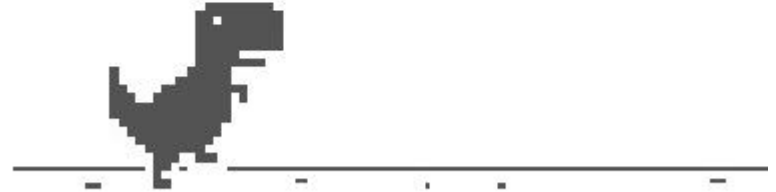


```
1 int oldStato = 0;
2 int stato = 0;
3
4 void suonaSu(){
5     for(int i=0;i<1000;i++){
6         tone(3,i);delay(1);
7     }
8     noTone(3);
9 }
10
11 void suonaGiu(){
12     for(int i=1000;i>0;i--){
13         tone(3,i);delay(1);
14     }
15     noTone(3);
16 }
17
18 void setup() {
19     pinMode(2,INPUT_PULLUP);
20     pinMode(3,OUTPUT);
21 }
22
23 void loop() {
24     stato = digitalRead(2);
25     if(stato == HIGH && oldStato==LOW) {
26         suonaSu();    // STATO su
27     }
28     if(stato == LOW && oldStato==HIGH) {
29         suonaGiu();    // STATO giu
30     }
31     oldStato = stato;
32     delay(100);
33 }
```




chrome://dino/

Giochiamo a dino usando Arduino come interfaccia per comandare i salti di dino



Configure

Port: [COM3] Baud: [9600] [OK] [Cancel]

Start Up:

Run at Start Up

Automatically Log In as:

User: _____

Password: _____

Extended Character Interpretation:

ASCII Extended e.g. DeltaTalker

ANSI Extended

Instructions:

Select the port you wish to use then set the Baud to match your communications device.

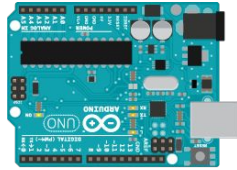
Check 'Run at Start Up' if you want this program to run automatically each time you start your computer or log in.

Check 'Automatically Log In' and enter a user name and password to automatically log into your system. (This may not be available on your computer.)

Select either ASCII or ANSI. Most newer devices will use ANSI.

Select OK to use the new values or Cancel to quit without making any changes.

AAC Keys



Configure

Port: COM3 Baud: 9600 OK
Cancel

Start Up

Run at Start Up

Automatically Log In as:

User: _____

Password: _____

Extended Character Interpretation

ASCII Extended e.g. DeltaTalker

ANSI Extended

Instructions

Select the port you wish to use then set the Baud to match your communications device.

Check 'Run at Start Up' if you want this program to run automatically each time you start your computer or log in.

Check 'Automatically Log In' and enter a user name and password to automatically log into your system. (This may not be available on your computer.)

Select either ASCII or ANSI.
Most newer devices will use ANSI.

Select OK to use the new values or Cancel to quit without making any changes.

Comunica con Arduino attraverso le porte COM (cioè le porte USB del computer).

Riceve i comandi da Arduino e li traduce in pressione dei tasti che possono essere ricevuti dai programmi del computer (se la finestra ha il focus).

La porta COM può essere utilizzata da un sistema alla volta: se vuoi utilizzare AAC per giocare allora devi togliere il monitor di Arduino; se vuoi caricare un nuovo sketch su Arduino devi uscire da AAC.



chrome://dino/

con un bottone, comandare i salti è
come comandare la pressione di un
tasto della tastiera



CORSA

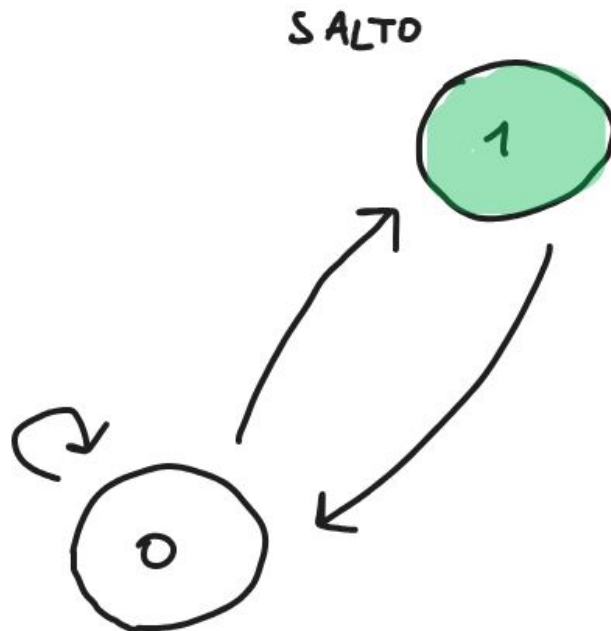
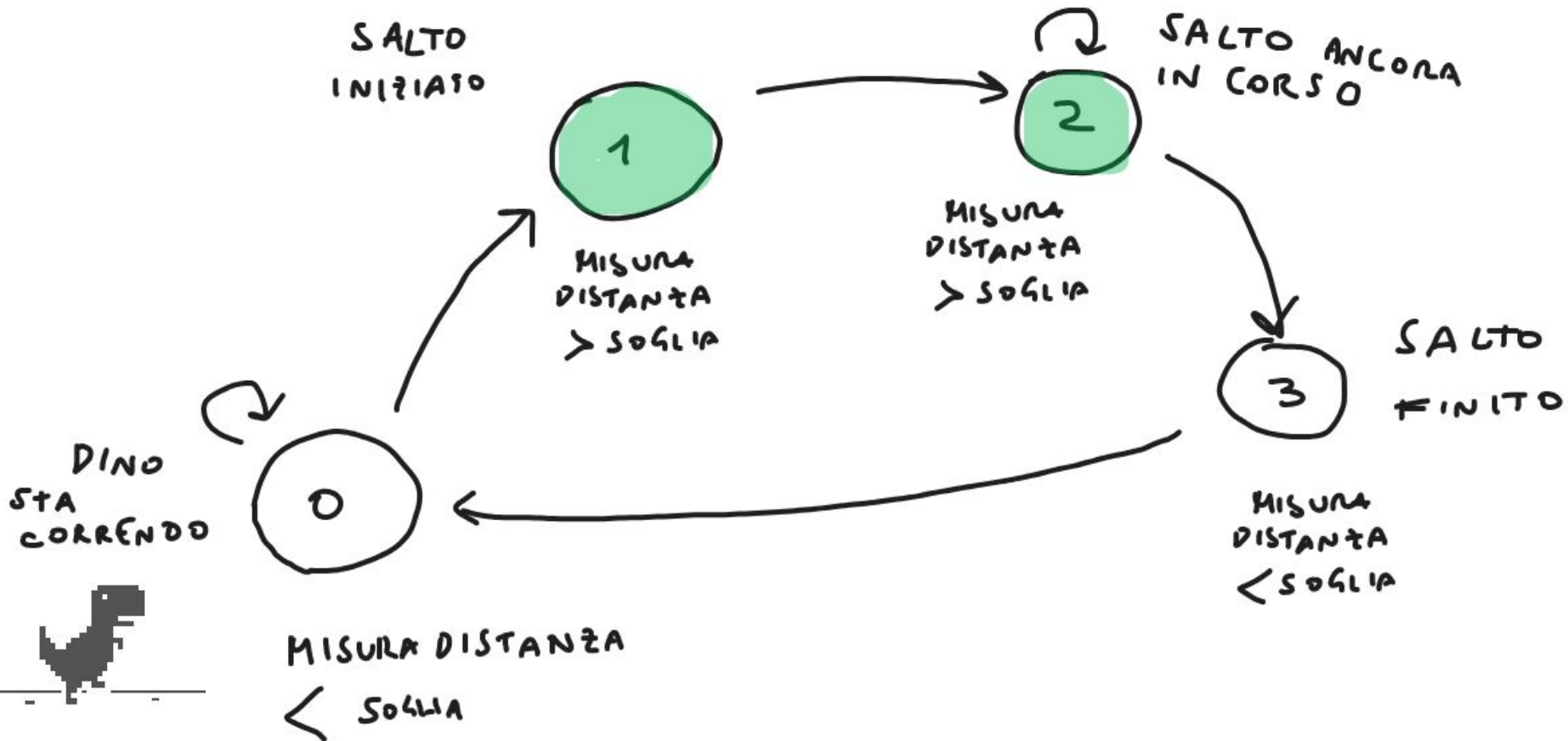


Diagramma degli stati per il salto di dino con un bottone

Diagramma degli stati per il salto di dino con sensore ultrasuoni



Il programma

```
1 int triggerPin = 9;
2 int echoPin = 7;
3 int soglia = 50;
4 int ledPin = 2;
5
6 void setup() {
7     pinMode(echoPin, INPUT);
8     pinMode(triggerPin, OUTPUT);
9     pinMode(ledPin, OUTPUT);
10    Serial.begin(9600);
11 }
12
13 unsigned long durata=0;
14 unsigned long distanza=0;
15
16 int old_stato = 0;
17 int new_stato = 0;
18
```

```
19 void loop() {
20     digitalWrite(triggerPin, HIGH);
21     delayMicroseconds(10);
22     digitalWrite(triggerPin, LOW);
23
24     durata = pulseIn(echoPin, HIGH, 50000 );
25
26     distanza = durata*0.0172; // cm
27
28     delay(50 - (durata / 1000));
29
30     if(distanza > soglia && old_stato==0) {
31         new_stato = 1;
32         c++;
33         Serial.print(" ");
34         digitalWrite(ledPin, HIGH);
35         delay(50);
36     } else if( distanza > soglia && old_stato==1) {
37         new_stato = 1;
38     } else if(distanza <= soglia && old_stato==1) {
39         new_stato = 2;
40     } else if(distanza <= soglia && old_stato==3) {
41         new_stato = 0;
42         digitalWrite(ledPin, LOW);
43     } else if(distanza <= soglia && old_stato==2) {
44         new_stato = 3;
45     } else if(distanza <= soglia && old_stato==0) {
46         new_stato = 0;
47     }
48     old_stato = new_stato;
49 }
```

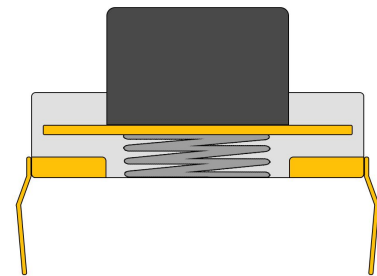
Debounce (togliere i segnali di “rimbalzo”)

E' una tecnica per filtrare quei segnali extra indesiderati e ottenere solo quello che ci serve. È come usare un filtro per il caffè: rimuoviamo la polvere del caffè per ottenere solo il caffè che vogliamo bere.

Ad esempio quando se premendo un bottone la molla dentro al bottone si piega un po' lateralmente e tocca e non tocca velocemente il contatto possiamo ottenere una sequenza di 0 e 1 invece che un passaggio netto da 0 a 1.

Con il debounce scriviamo un codice che, quando riceve un segnale dall'Arduino, aspetta un breve periodo di tempo prima di fare qualcosa. Durante questo tempo, se riceve altri segnali, li ignora.

In pratica, questo significa che quando usiamo un sensore in un progetto Arduino, utilizziamo questa tecnica per assicurarci che il nostro programma risponda correttamente.



Problema analogo con un pulsante:

Immaginate di avere un pulsante collegato ad Arduino. **Ogni volta che lo premiamo, può mandare segnali veloci e ripetuti anche se lo teniamo premuto solo una volta.** Questo può creare problemi nei nostri programmi, ad esempio se premo un pulsante per accendere/spegnere una luce.



Cesso automatico

da un'idea di Giovanni!



Cosa utilizzare

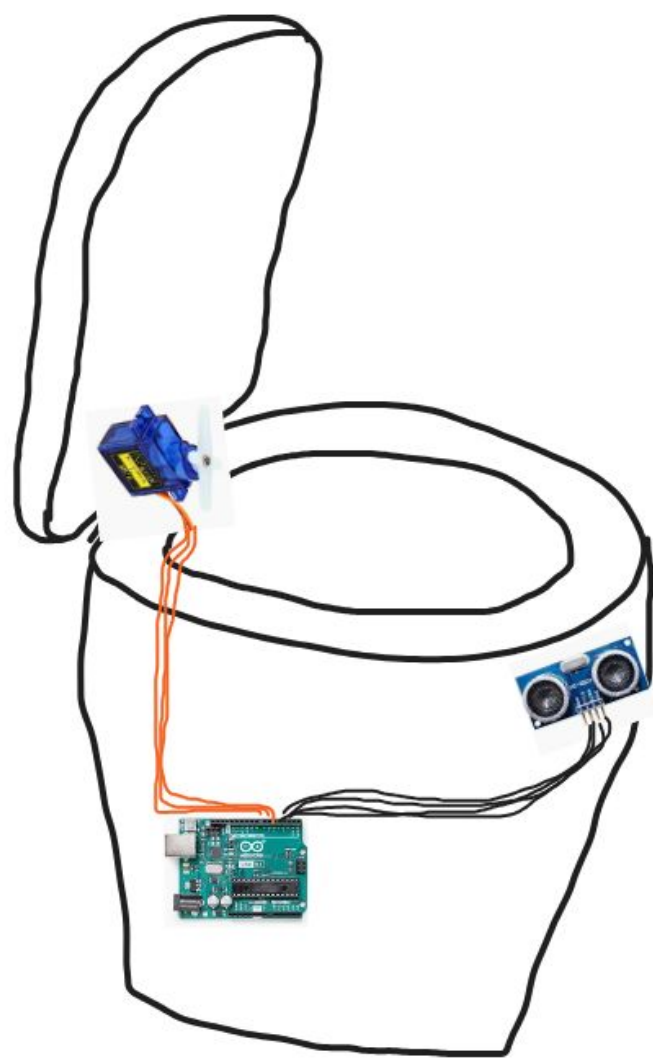
Servo motore

Sensore ultrasuoni

Arduino

Cavetti

Fare il prototipo collegando i pezzi all'Arduino



Scatola automatica

Costruiamo una scatola che si apre
e si chiude da sola
(come l'idea di Giovanni ma "safe")



Scatola automatica

Cosa serve:

Servo motore

Sensore ultrasuoni

Arduino

Cavetti



Facciamo il prototipo
collegando i pezzi
all'Arduino



Il programma deve:

- prima di tutto importare la libreria per il servo motore e definire le variabili
- nel setup, definire come si comportano i pin (il motore e il trigger degli ultrasuoni OUTPUT, e echo INPUT)
- misurare la distanza con gli ultrasuoni
- se la distanza è minore di 15 cm
 - muovere il motore per aprire
- altrimenti
 - muovere il motore per chiudere

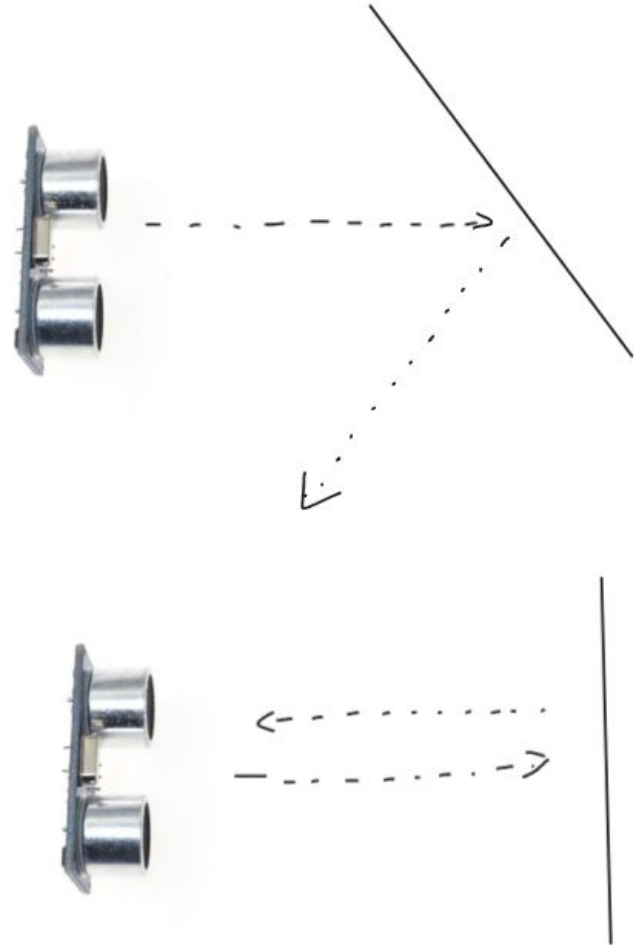
Miglioramenti

Il sensore ad ultrasuoni è “rumoroso”, le misure sono influenzate da rimbalzi del segnale contro altri ostacoli, oppure dall’inclinazione dell’ostacolo.

Possiamo fare la media tra più misurazioni per evitare che una misura errata causi comportamenti imprevisti del coperchio (tipo che si chiude prima del previsto).

Possiamo controllare lo stato precedente e aprire solo se cambia lo stato, per evitare di sentire il motore che continua a lavorare.

Possiamo muovere il coperchio più lentamente usando un ciclo for che aumenta/diminuisce l’angolo gradualmente .



Assemblaggio

Ricavare due alloggiamenti, per il sensore e per il motore.

Il motore deve essere messo in modo che l'angolo 90 sia chiuso e 0 aperto.

Taglierino e colla a caldo.



FINE



PDF delle lezioni:

<https://www.barattalo.it/scuola.pdf>

Slide online:

<https://www.barattalo.it/scuola>