

# Speedy Gonzales

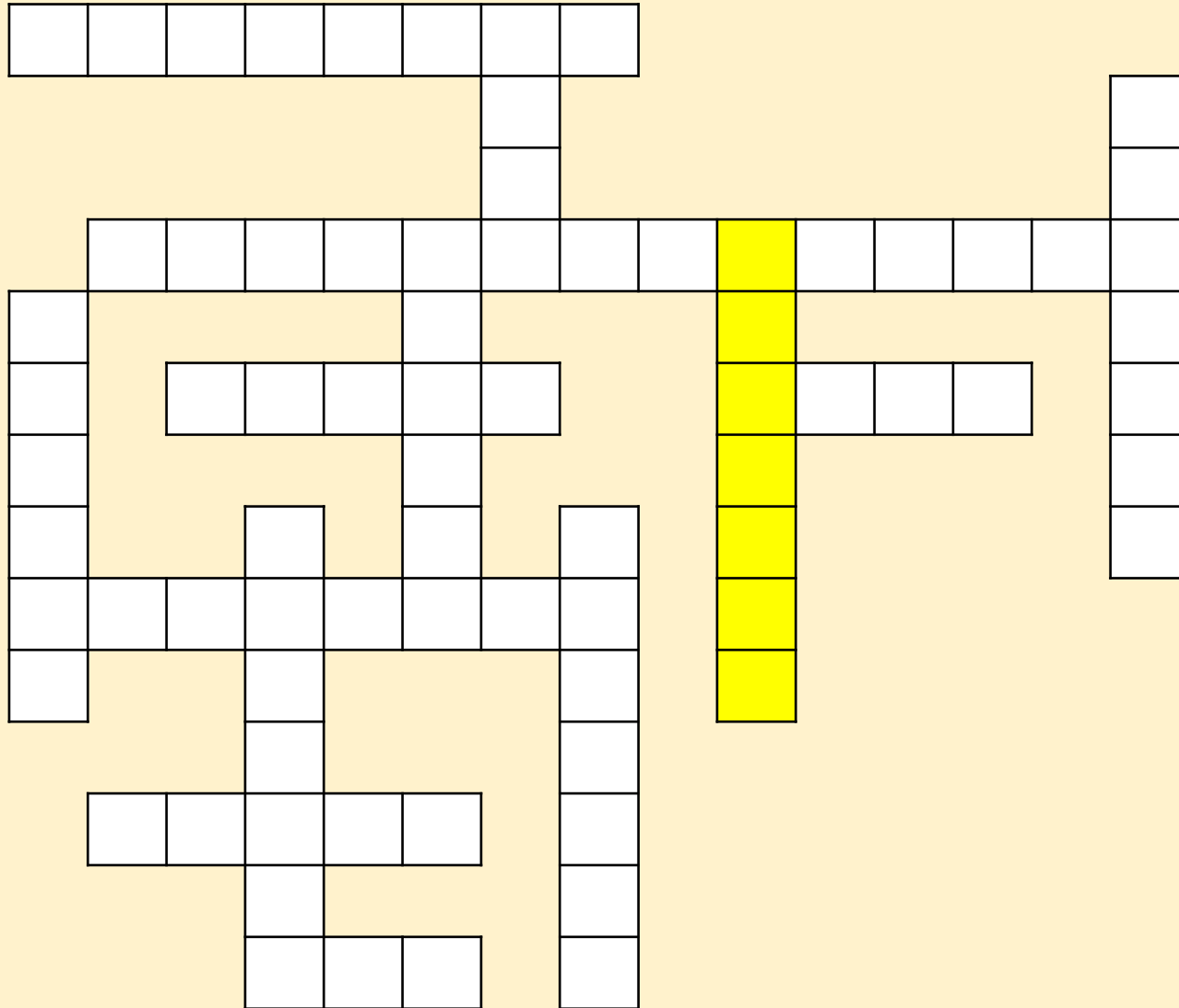
*Speedy Gonzales è un personaggio dei cartoni appartenente alla famiglia dei Looney Toones.*

*Il topolino messicano è famoso per la sua velocità, in pratica un personaggio estremamente piccolo ma anche molto veloce.*



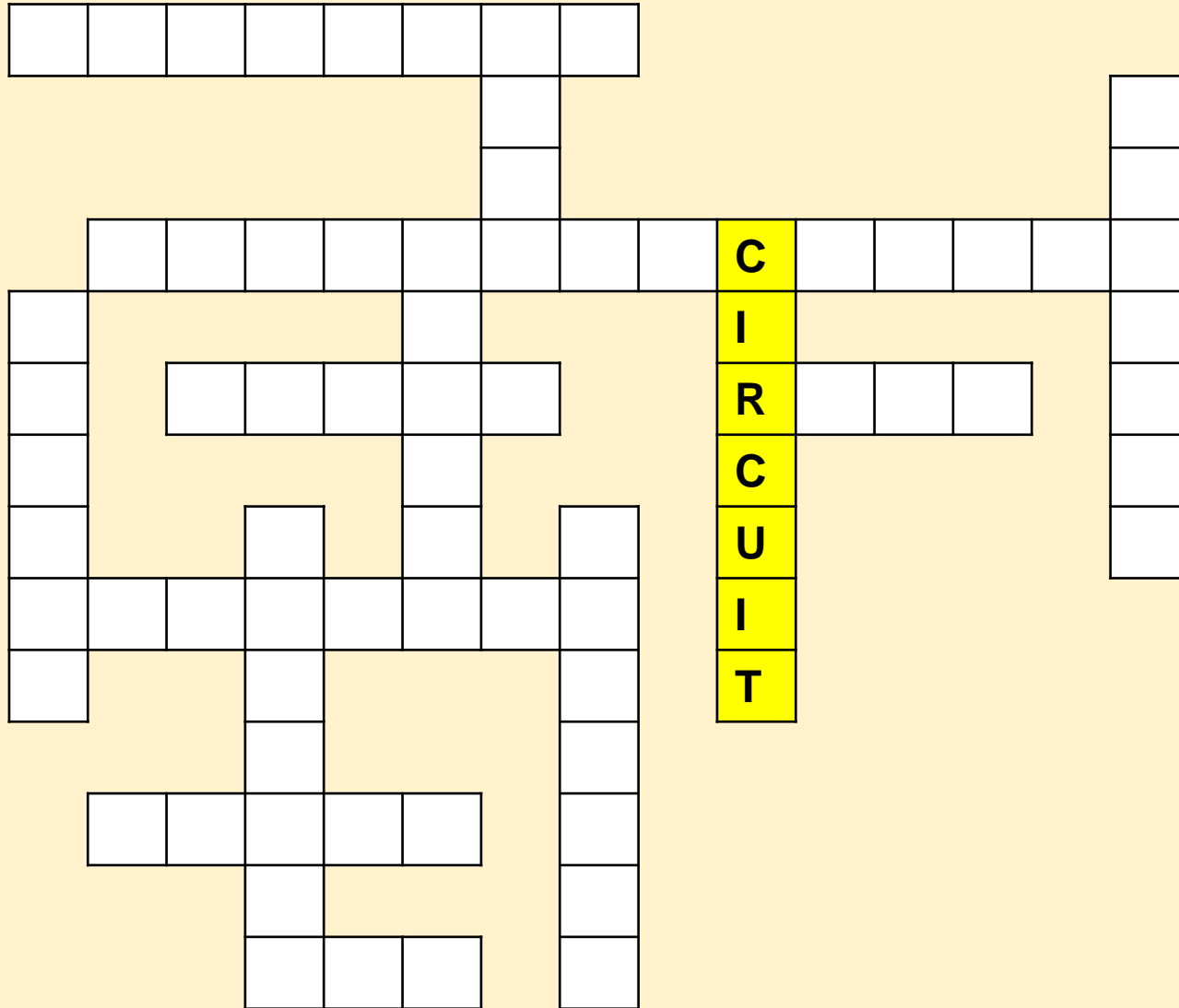
*Essendo messicano, Speedy Gonzales parlava spagnolo, ma noi abbiamo qualcosa di piccolo e veloce come lui che parla solo inglese. Pertanto d'ora in avanti parleremo anche in inglese cercando di risolvere un **cruciverba** che ci aiuterà a scoprire cosa possiamo fare con il nostro Speedy Gonzales.*

# A SET OF ELECTRONIC COMPONENT CONNECTED TO EACH OTHER



- *CIRCUIT*
- *CURRENT*
- *MICROPROCESSOR*
- *MEMORY*
- *BJT*
- *WRITE*
- *TYPE*
- *INPUT*
- *READ*
- *DIR*
- *LOOP*
- *SETUP*
- *LED*
- *RESISTOR*
- *DIGITAL*
- *VOLTAGE*
- *VARIABLE*
- *OUTPUT*
- *BATTERY*
- *ARDUINO*
- *POWER*
- *TOP*

# A SET OF ELECTRONIC COMPONENT CONNECTED TO EACH OTHER

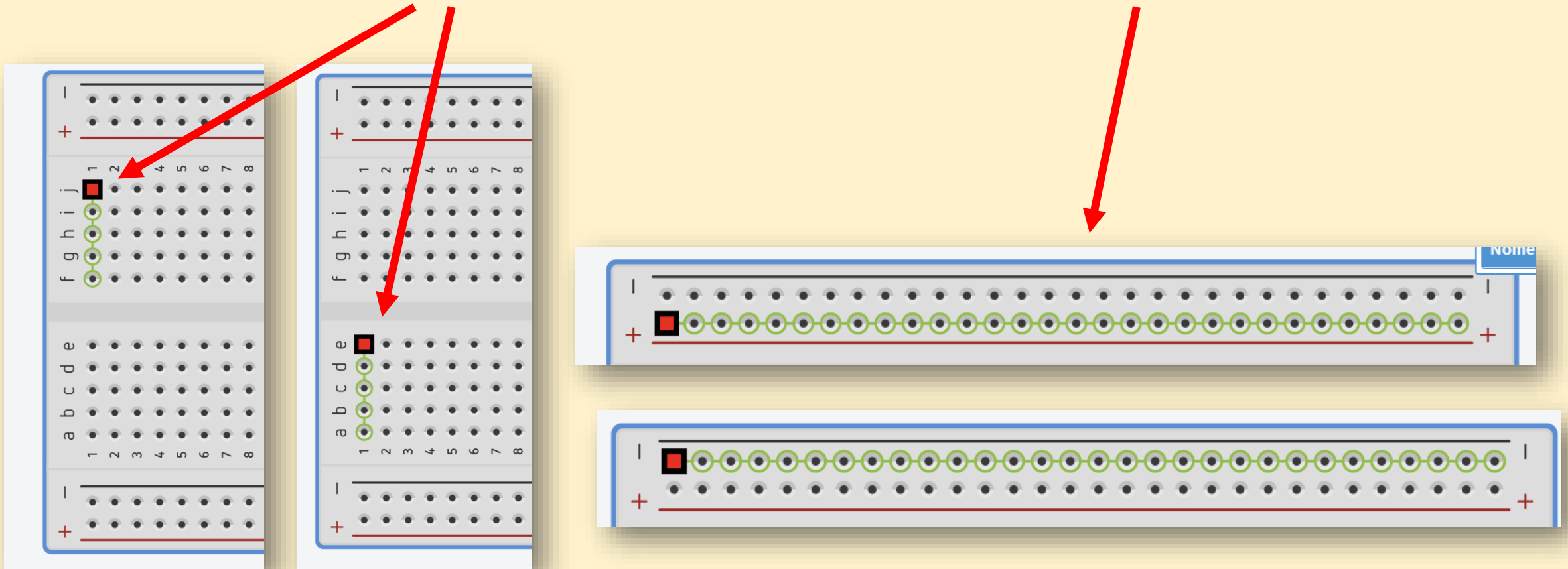


- **CIRCUIT**
- CURRENT
- MICROPROCESSOR
- MEMORY
- BJT
- WRITE
- TYPE
- INPUT
- READ
- DIR
- LOOP
- SETUP
- LED
- RESISTOR
- DIGITAL
- VOLTAGE
- VARIABLE
- OUTPUT
- BATTERY
- ARDUINO
- POWER
- TOP

# BREAD BOARD

La bread board, è un oggetto composto da una griglia di fori metallizzati al loro interno, e collegati tra di loro in una maniera predefinita e viene utilizzata per realizzare dei CIRCUITI ELETTRONICI.

I fori al centro sono collegati in verticale, i fori con i simboli + e - sono collegati tutti in orizzontale tra di loro.



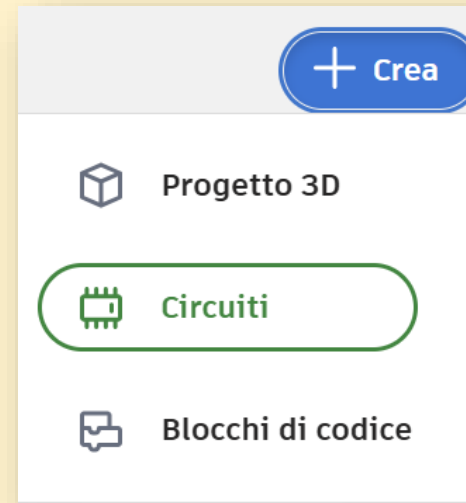
# TINKERCAD

Per aiutarci utilizzeremo il sito internet [www.tinkercad.com](http://www.tinkercad.com)

Tramite questo sito possiamo realizzare schemi, scrivere programmi e provarne il funzionamento.

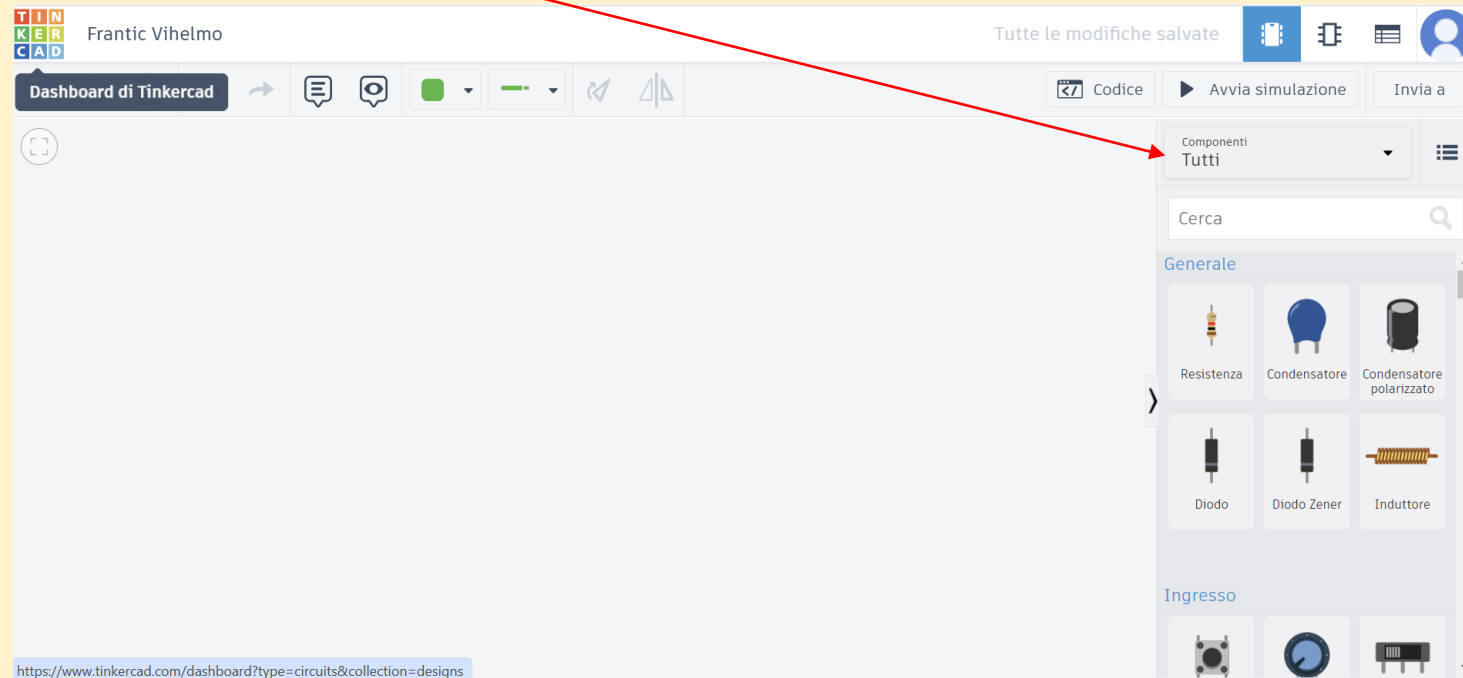
Potete accedere con qualsiasi account google, e successivamente avrete il vostro laboratorio virtuale.

Una volta entrati nel sito, con il pulsante CREA Circuiti possiamo realizzare il nostro primo progetto.



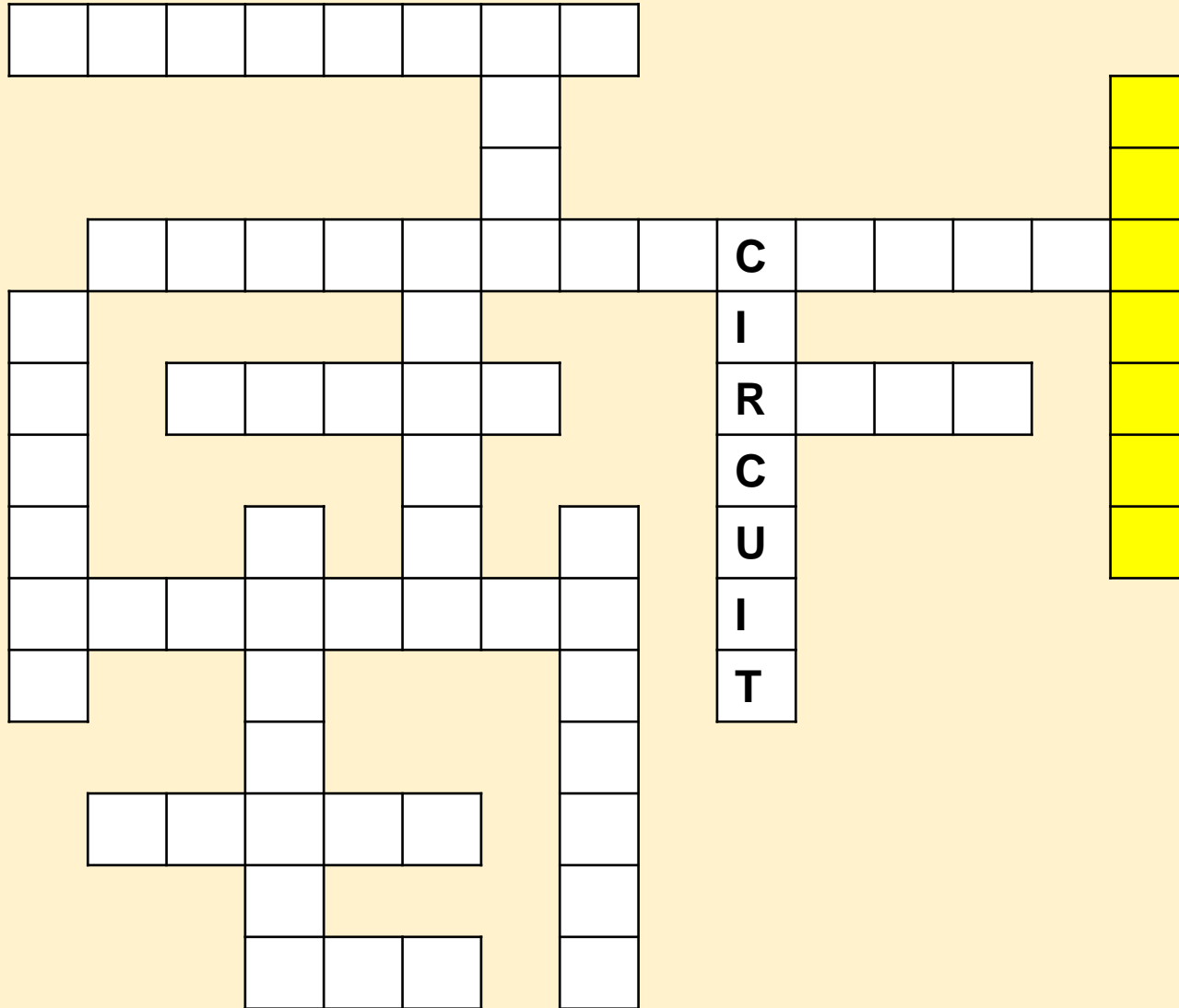
# TINKERCAD

*Scegliamo subito di avere a disposizione tutti i componenti*



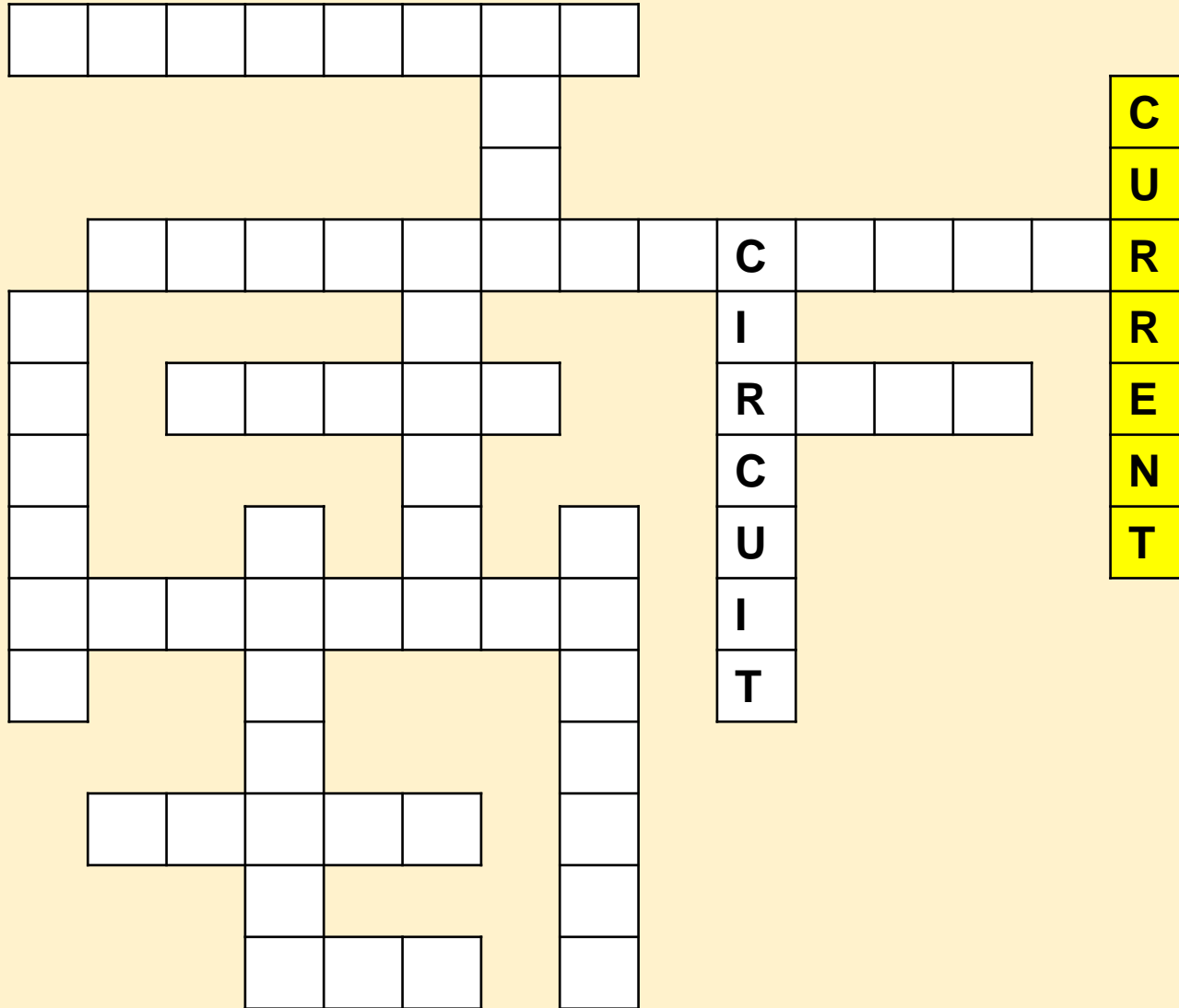
*Ora proseguiamo con il nostro cruciverba.*

# A FLOW OF ELECTRONS



- *CIRCUIT*
- *CURRENT*
- *MICROPROCESSOR*
- *MEMORY*
- *BJT*
- *WRITE*
- *TYPE*
- *INPUT*
- *READ*
- *DIR*
- *LOOP*
- *SETUP*
- *LED*
- *RESISTOR*
- *DIGITAL*
- *VOLTAGE*
- *VARIABLE*
- *OUTPUT*
- *BATTERY*
- *ARDUINO*
- *POWER*
- *TOP*

# A FLOW OF ELECTRONS

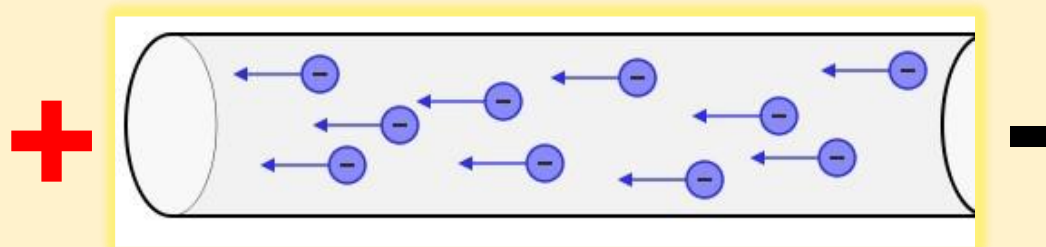


- CIRCUIT
- **CURRENT**
- MICROPROCESSOR
- MEMORY
- BJT
- WRITE
- TYPE
- INPUT
- READ
- DIR
- LOOP
- SETUP
- LED
- RESISTOR
- DIGITAL
- VOLTAGE
- VARIABLE
- OUTPUT
- BATTERY
- ARDUINO
- POWER
- TOP



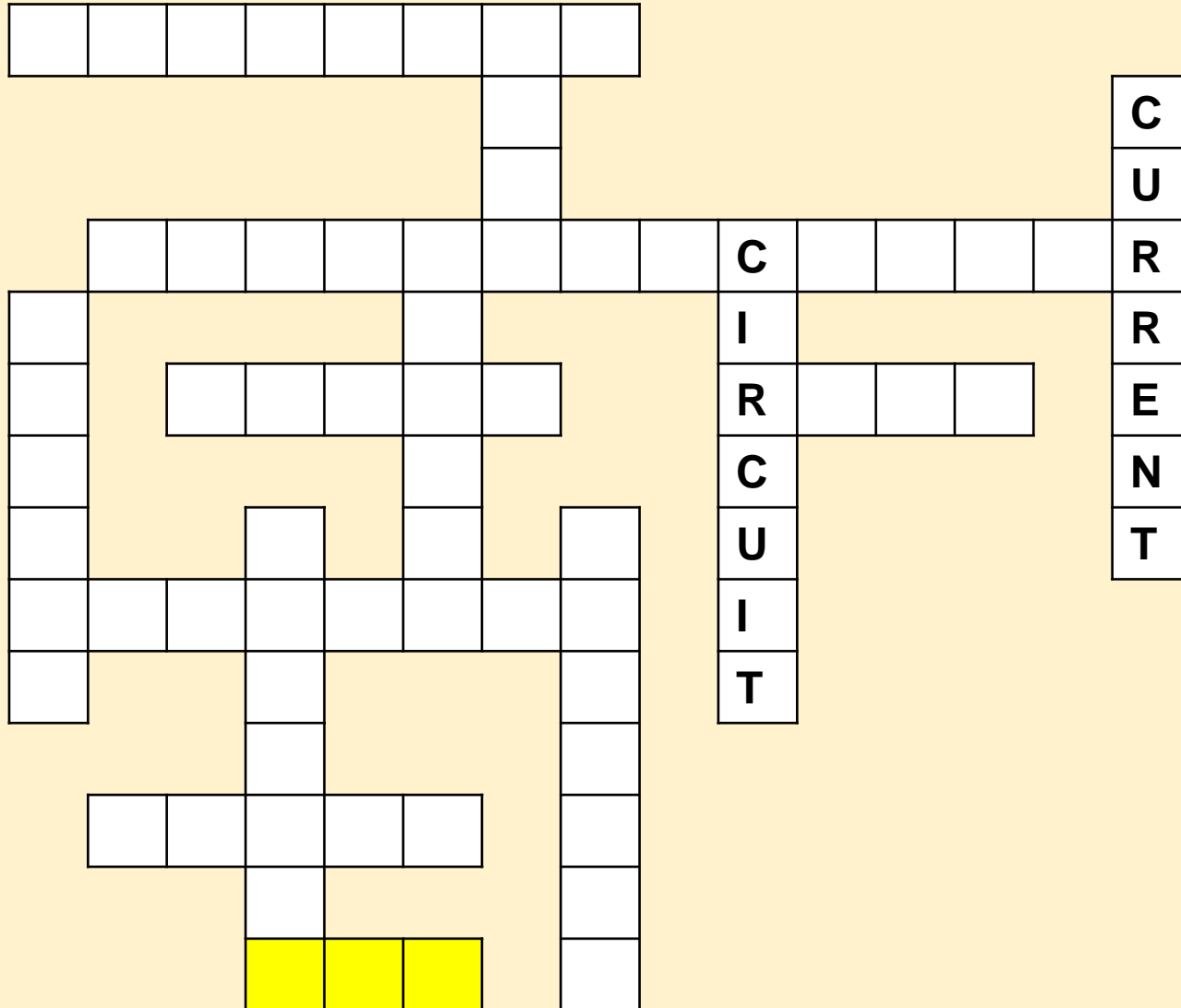
# CORRENTE ELETTRICA

La *CORRENTE ELETTRICA* è un flusso di elettroni (cariche elettriche negative) che si muovono in maniera ordinata in un conduttore.



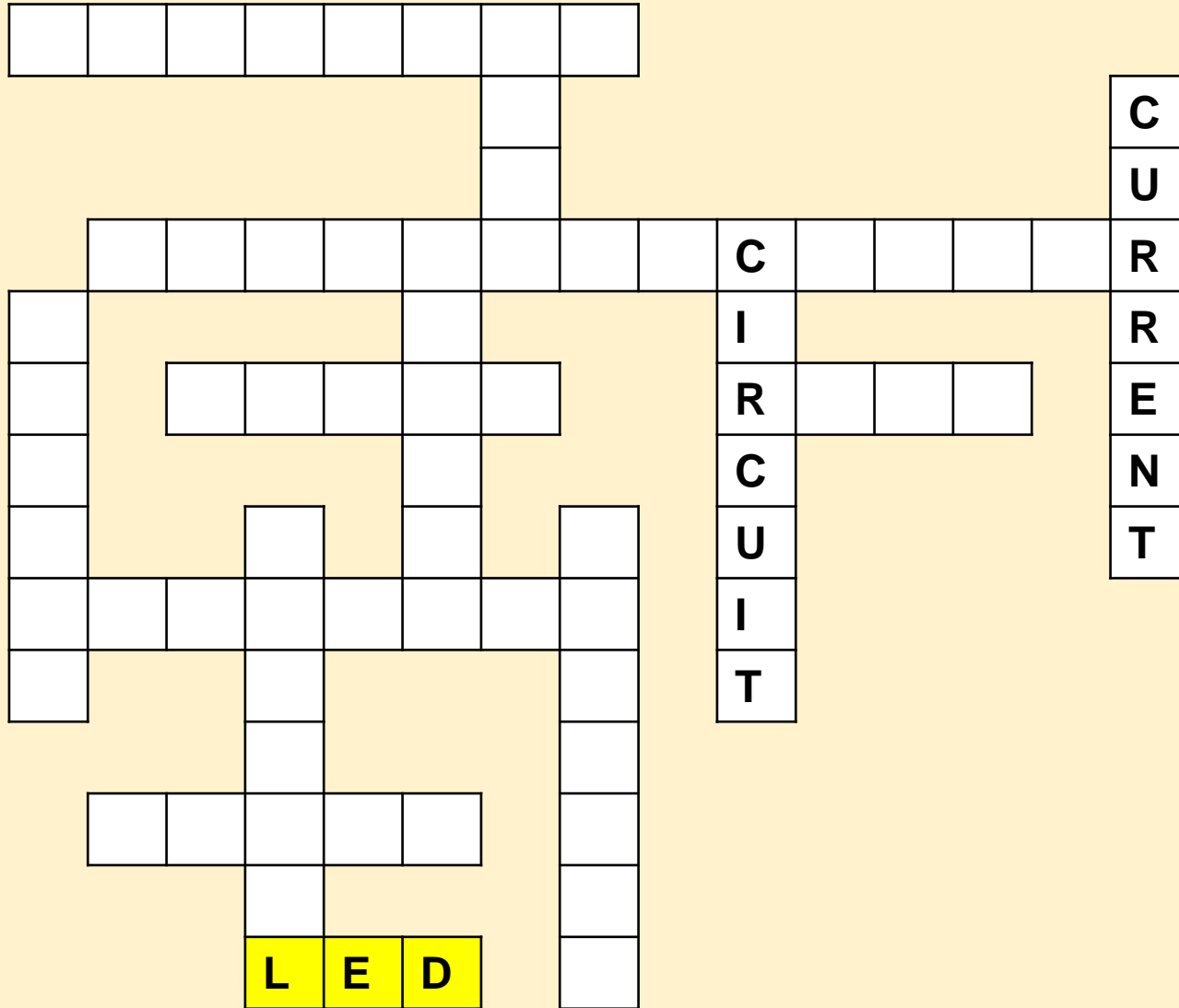
*Gli elettroni si spostano solo se sono attratti da un polo con un potenziale elettrico positivo, ai due capi del conduttore per avere una corrente dovremo avere una differenza di potenziale anche detta **TENSIONE**.*

# ELECTRONIC DEVICE THAT EMITS LIGHT



- *CIRCUIT*
- *CURRENT*
- *MICROPROCESSOR*
- *MEMORY*
- *BJT*
- *WRITE*
- *TYPE*
- *INPUT*
- *READ*
- *DIR*
- *LOOP*
- *SETUP*
- *LED*
- *RESISTOR*
- *DIGITAL*
- *VOLTAGE*
- *VARIABLE*
- *OUTPUT*
- *BATTERY*
- *ARDUINO*
- *POWER*
- *TOP*

# ELECTRONIC DEVICE THAT EMITS LIGHT

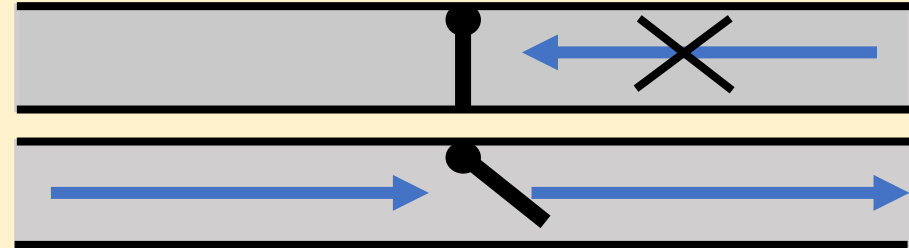


- CIRCUIT
- CURRENT
- MICROPROCESSOR
- MEMORY
- BJT
- WRITE
- TYPE
- INPUT
- READ
- DIR
- LOOP
- SETUP
- **LED**
- RESISTOR
- DIGITAL
- VOLTAGE
- VARIABLE
- OUTPUT
- BATTERY
- ARDUINO
- POWER
- TOP

# DIODO

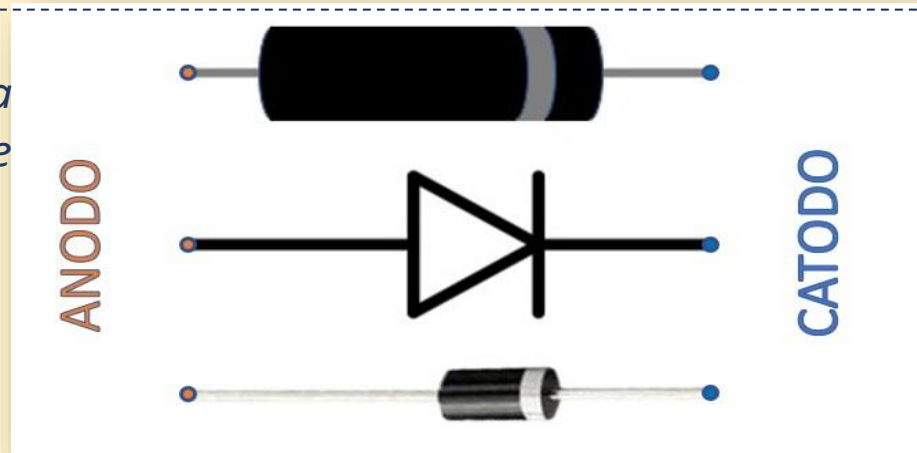
Il **DIODO** è un componente che consente il passaggio di corrente solo in una direzione, per analogia in un circuito idraulico il DIODO può essere paragonato ad una **VALVOLA**.

E consideriamo un tubo in cui scorre un liquido e al suo interno inseriamo una valvola che si apre solo in una direzione, avremo che il fluido scorrerà solo in una direzione e solo dopo aver superato la forza necessaria per aprire la valvola.



Il diodo si comporta allo stesso modo, farà scorrere la corrente solo se entra dall'**ANODO** ed esce dal **CATODO**, e se la tensione applicata supera un valore di circa 0,7Volt.

In caso contrario non fa passare corrente.



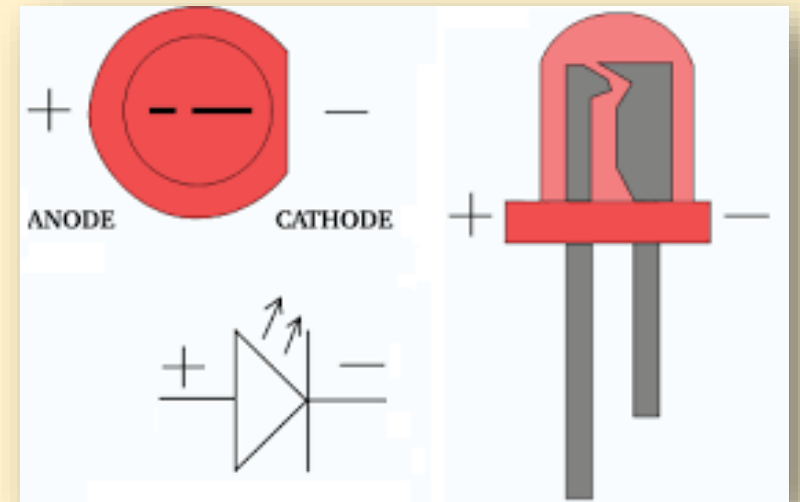
# DIODO LED

*Il DIODO LED (Light Emitting Diode) si comporta esattamente come un normale DIODO, anche se la tensione da superare per farlo condurre è di circa 1,3 Volt.*

*Un po' come se per aprire lo sportello della valvola vista prima servisse una forza maggiore.*

*Inoltre il DIODO LED quando viene percorso da corrente EMETTE UNA RADIAZIONE LUMINOSA COLORATA.*

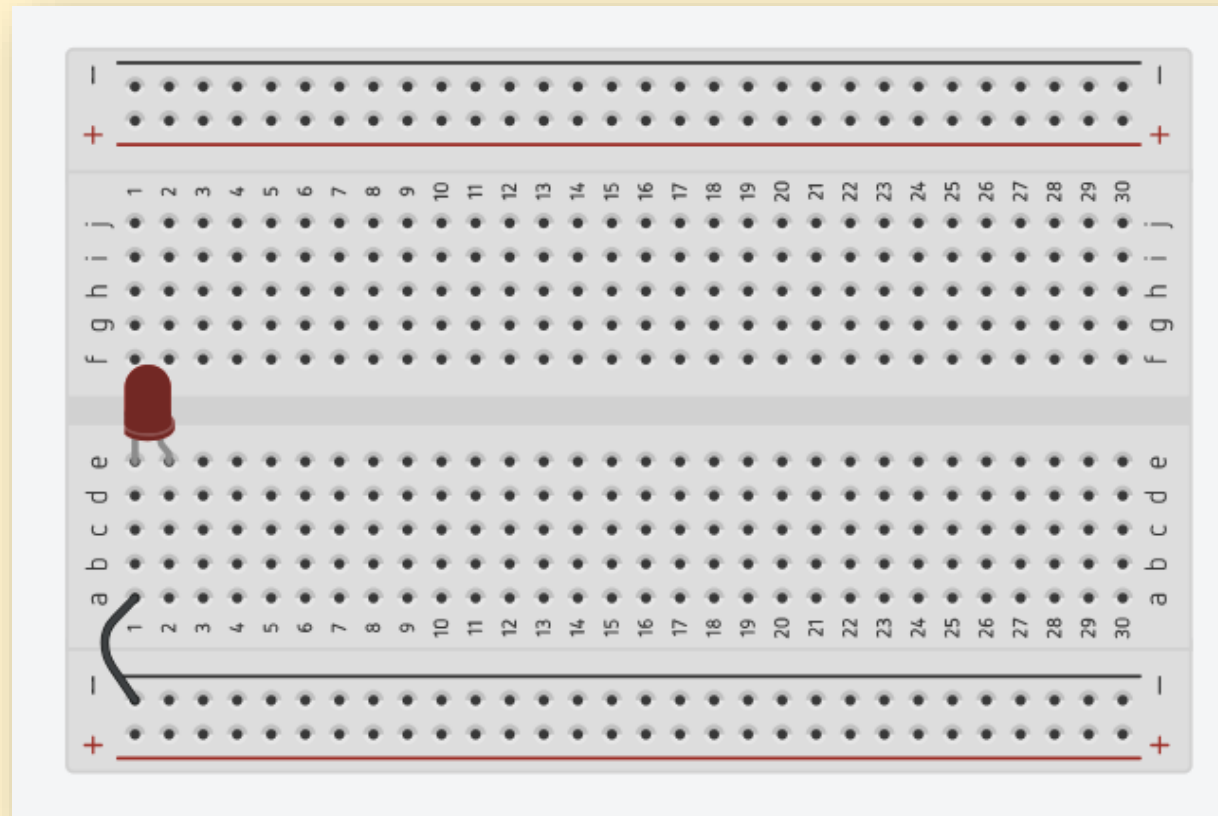
*L'ANODO (dove entra la corrente) è il terminale più lungo. Il CATODO (dove esce la corrente) è il terminale più corto o visto dall'alto quello con una parte piatta.*



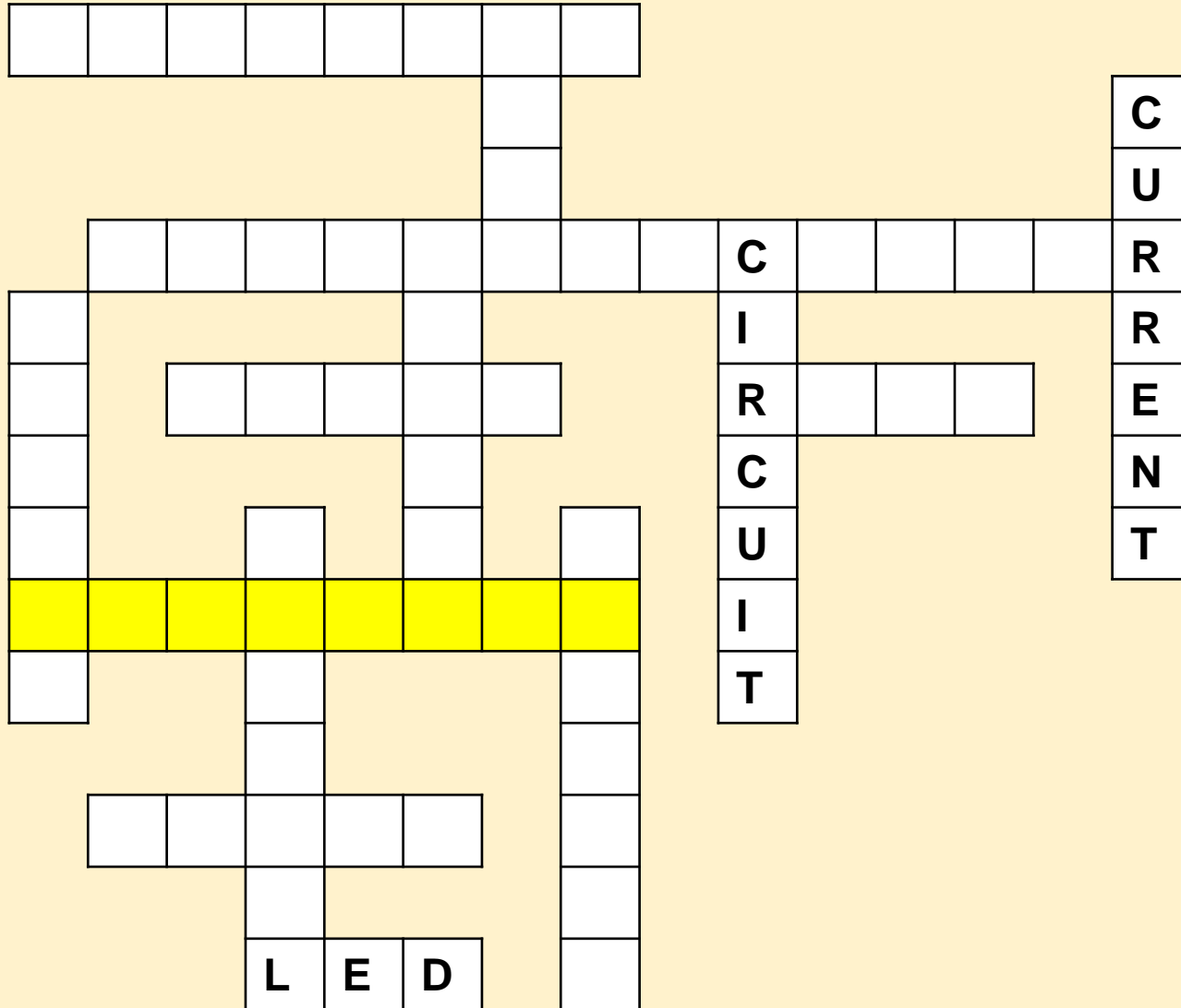
# DIODO LED

Disponiamo il LED sulla BREAD BOARD, come nell'immagine sotto, facciamo attenzione a mettere **il terminale più lungo a destra**.

A sinistra invece collegheremo il filetto nero alla linea in basso identificata con il segno - .



# IT OPPOSES ELECTRIC CURRENT



- *CIRCUIT*
- *CURRENT*
- *MICROPROCESSOR*
- *MEMORY*
- *BJT*
- *WRITE*
- *TYPE*
- *INPUT*
- *READ*
- *DIR*
- *LOOP*
- *SETUP*
- *LED*
- *RESISTOR*
- *DIGITAL*
- *VOLTAGE*
- *VARIABLE*
- *OUTPUT*
- *BATTERY*
- *ARDUINO*
- *POWER*
- *TOP*

# IT OPPOSES ELECTRIC CURRENT



- CIRCUIT
- CURRENT
- MICROPROCESSOR
- MEMORY
- BJT
- WRITE
- TYPE
- INPUT
- READ
- DIR
- LOOP
- SETUP
- LED
- **RESISTOR**
- DIGITAL
- VOLTAGE
- VARIABLE
- OUTPUT
- BATTERY
- ARDUINO
- POWER
- TOP



# RESISTENZA

La **RESISTENZA** è un componente elettronico che si oppone al passaggio della corrente, riducendone il valore. Viene collegata in **SERIE** al **DIODO LED** per regolare la corrente che scorre su di esso.



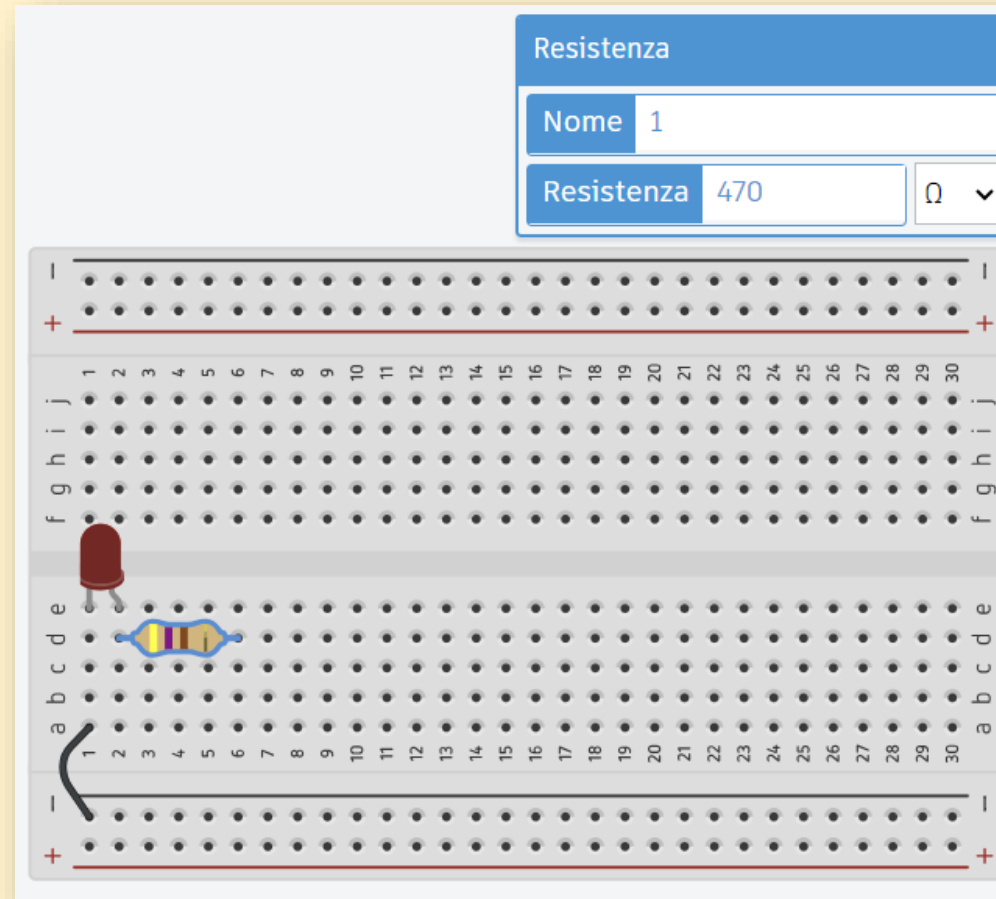
Se la usiamo per controllare la corrente che scorre in un **DIODO LED**, il valore di **RESISTENZA** può andare da 220 Ohm ad un massimo di 1000 Ohm (1kOhm).

Rosso-rosso-marrone	=	220 Ohm	
Arancio-arancio-marrone	=	330 Ohm	
Giallo-viola-marrone	=	470 Ohm	
Verde-blu-marrone	=	560 Ohm	
Blu-grigio-marrone	=	680 Ohm	
Grigio-rosso-marrone	=	820 Ohm	
Marrone-nero-rosso	=	1000 Ohm	

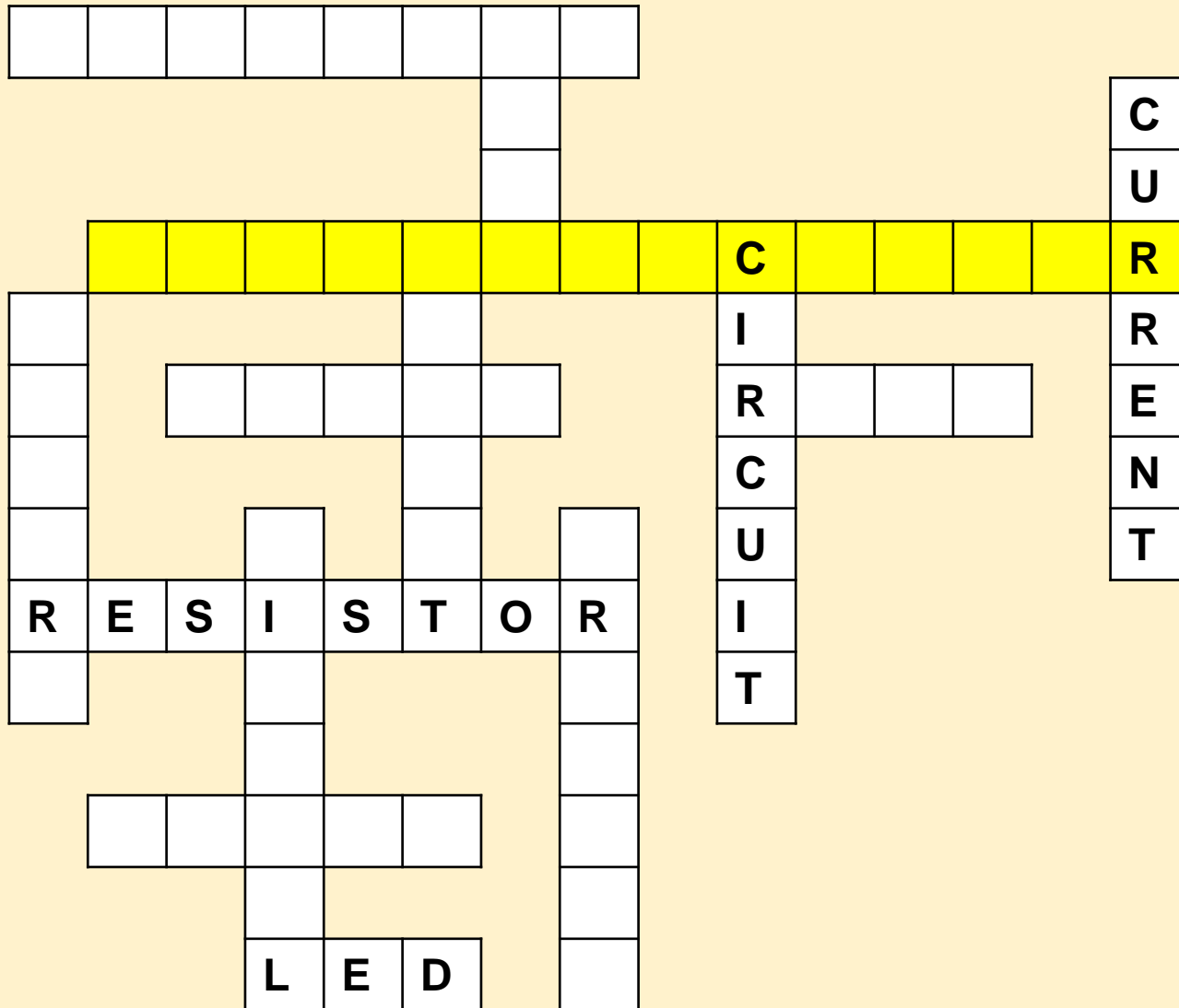
Ad ogni colore corrisponde un numero, i primi due colori rappresentano le prime due cifre, il terzo colore il numero di zeri.

# RESISTENZA

*Inseriamo una resistenza del valore di 470 Ohm come nella figura.*

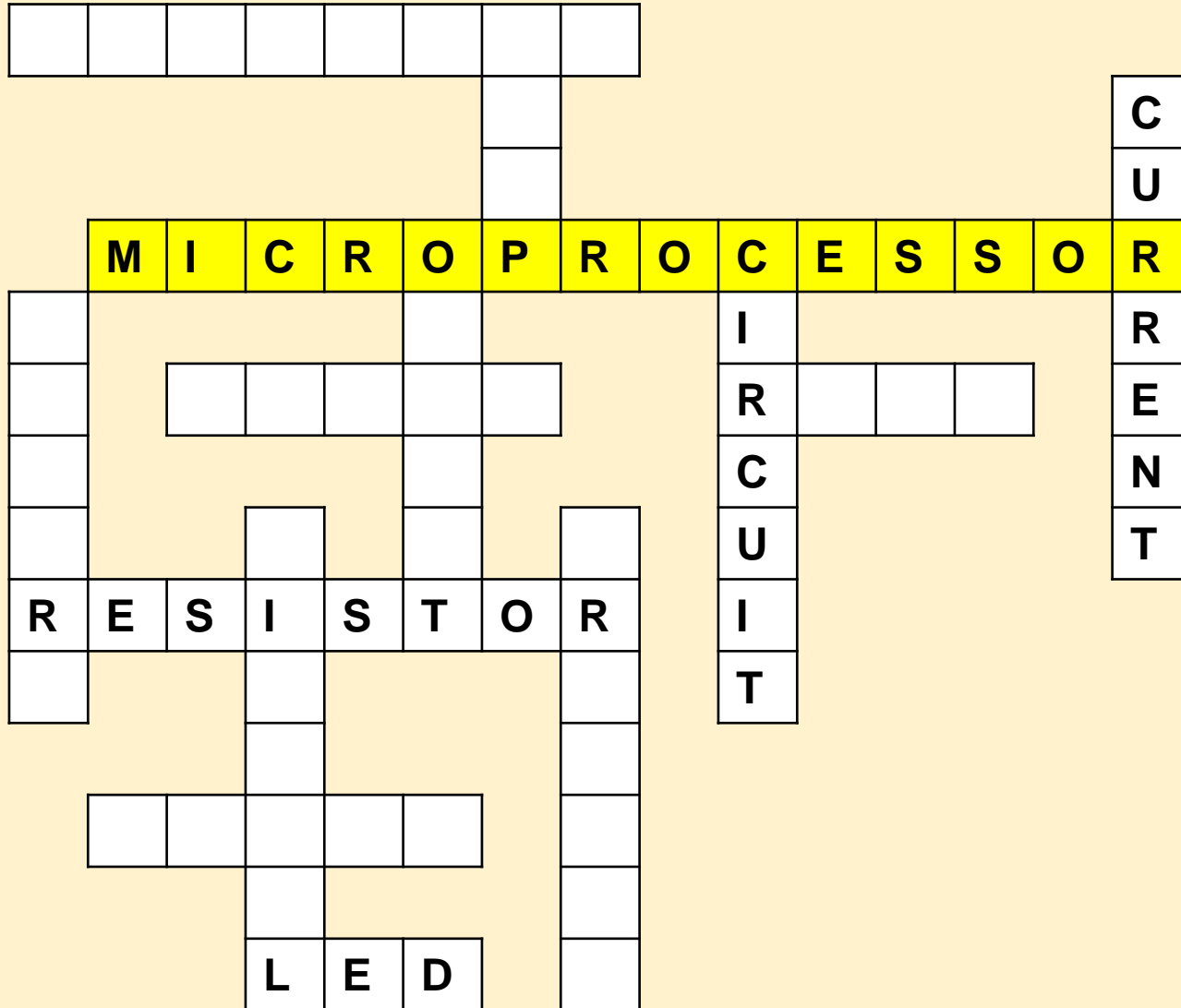


# THE BRAIN OF THE PERSONAL COMPUTER



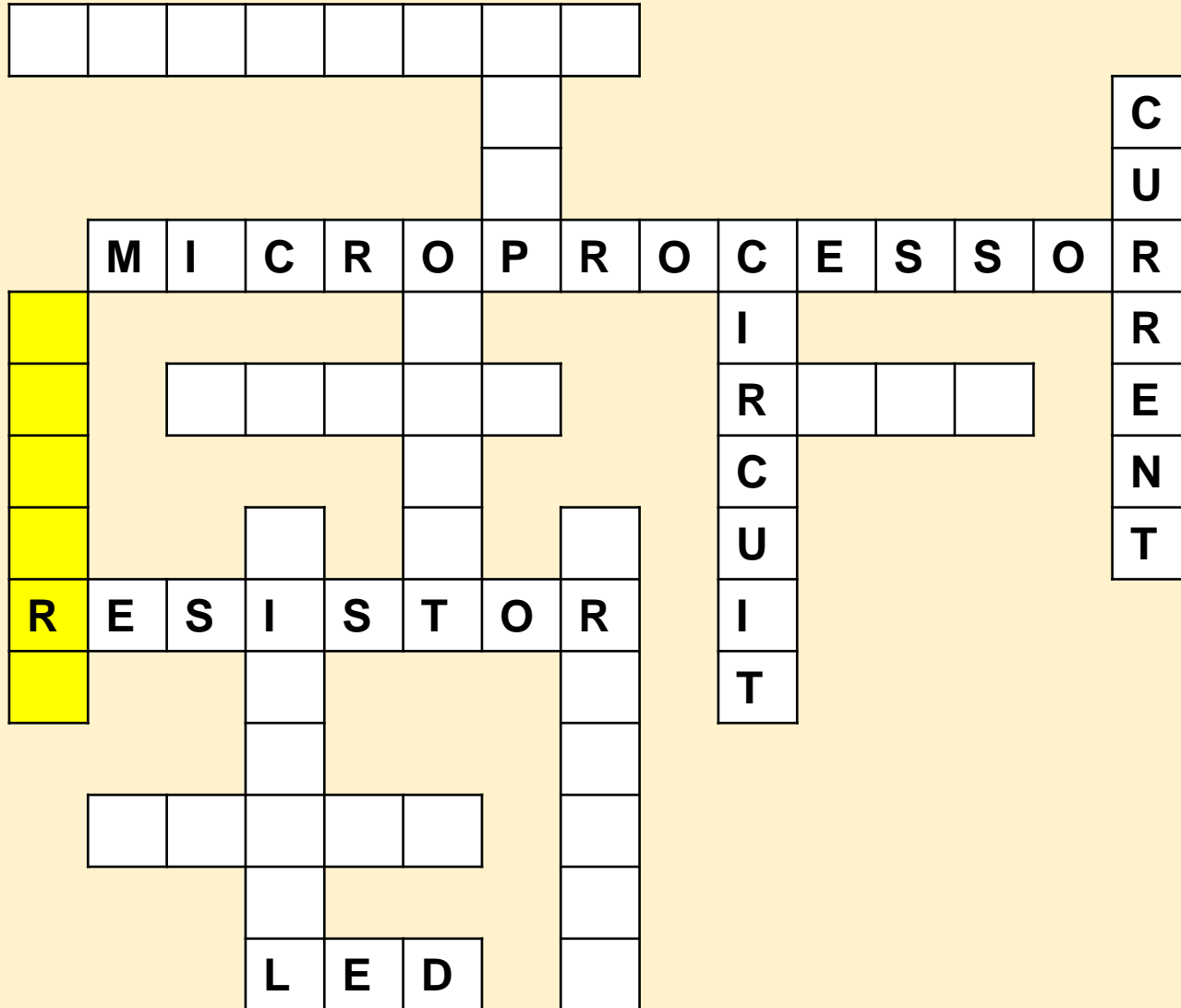
- *CIRCUIT*
- *CURRENT*
- *MICROPROCESSOR*
- *MEMORY*
- *BJT*
- *WRITE*
- *TYPE*
- *INPUT*
- *READ*
- *DIR*
- *LOOP*
- *SETUP*
- *LED*
- *RESISTOR*
- *DIGITAL*
- *VOLTAGE*
- *VARIABLE*
- *OUTPUT*
- *BATTERY*
- *ARDUINO*
- *POWER*
- *TOP*

# THE BRAIN OF THE PERSONAL COMPUTER



- CIRCUIT
- CURRENT
- **MICROPROCESSOR**
- MEMORY
- BJT
- WRITE
- TYPE
- INPUT
- READ
- DIR
- LOOP
- SETUP
- LED
- RESISTOR
- DIGITAL
- VOLTAGE
- VARIABLE
- OUTPUT
- BATTERY
- ARDUINO
- POWER
- TOP

# ELECTRONIC DEVICE CAPABLE OF STORING DATA



- CIRCUIT
- CURRENT
- MICROPROCESSOR
- MEMORY
- BJT
- WRITE
- TYPE
- INPUT
- READ
- DIR
- LOOP
- SETUP
- LED
- RESISTOR
- DIGITAL
- VOLTAGE
- VARIABLE
- OUTPUT
- BATTERY
- ARDUINO
- POWER
- TOP

# ELECTRONIC DEVICE CAPABLE OF STORING DATA

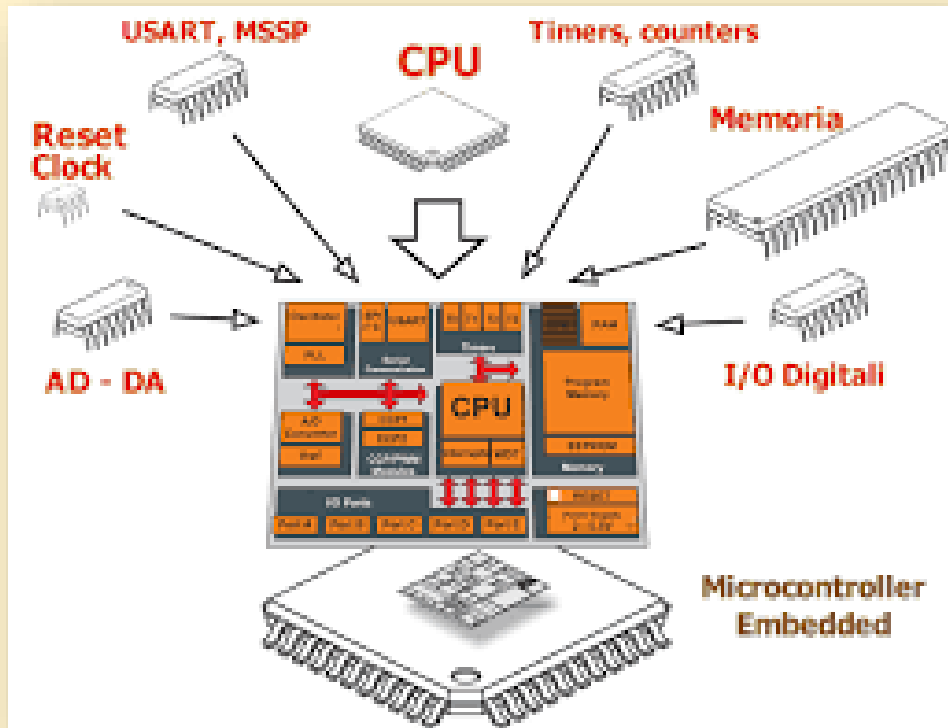


- CIRCUIT
- CURRENT
- MICROPROCESSOR
- **MEMORY**
- BJT
- WRITE
- TYPE
- INPUT
- READ
- DIR
- LOOP
- SETUP
- LED
- RESISTOR
- DIGITAL
- VOLTAGE
- VARIABLE
- OUTPUT
- BATTERY
- ARDUINO
- POWER
- TOP

# MICROCONTROLLORE

*Il MICROCONTROLLORE, è un componente elettronico composto da un PROCESSORE (CPU) memoria RAM, memoria FLASH e periferiche di ingresso ed uscita.*

*E' come se fosse un PC integrato in un unico componente.*



*La memoria RAM è quella memoria che contiene i dati, ed allo spegnimento perde il suo contenuto.*

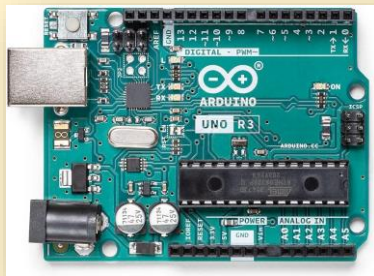
*La memoria FLASH è quella che contiene il programma ed allo spegnimento mantiene il suo contenuto.*

# MICROCONTROLLORE

*Il Microcontrollore trova facilmente utilizzo in applicazioni EMBEDDED, cioè applicazioni specifiche per il controllo di singoli dispositivi (es. router, antifurto, centralina telefonica, centralina automobile ecc...).*

*Il Microprocessore invece opera spesso all'interno di sistemi di elaborazione complessi come i Personal Computer, insieme ad altri processori e componenti, con memorie di massa differenti (Hard disk, SSD) e quasi sempre con un Sistema Operativo (Windows 10, Linux, IOS ecc...).*

**SCHEDA ARDUINO UNO**



**SONO DUE MONDI  
COMPLETAMENTE DIVERSI CON  
APPLICAZIONI DIFFERENTI, PER  
QUESTO **NON COMPARABILI****



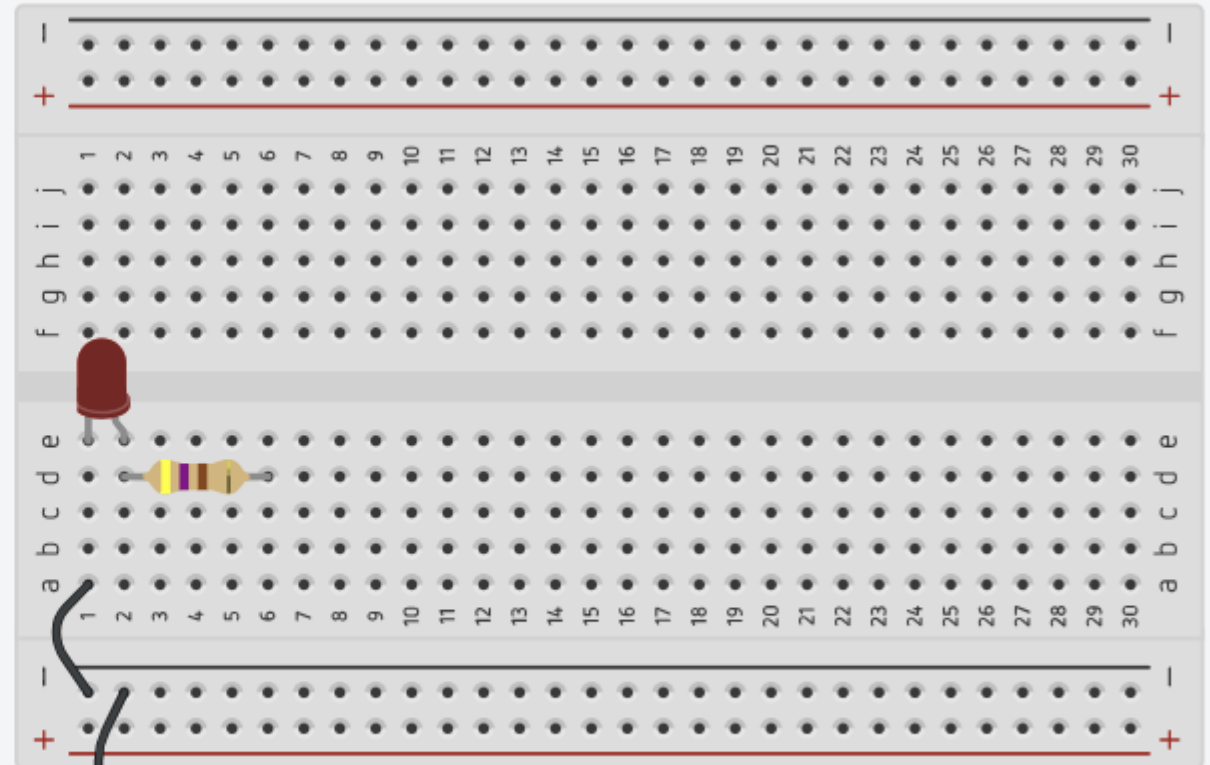
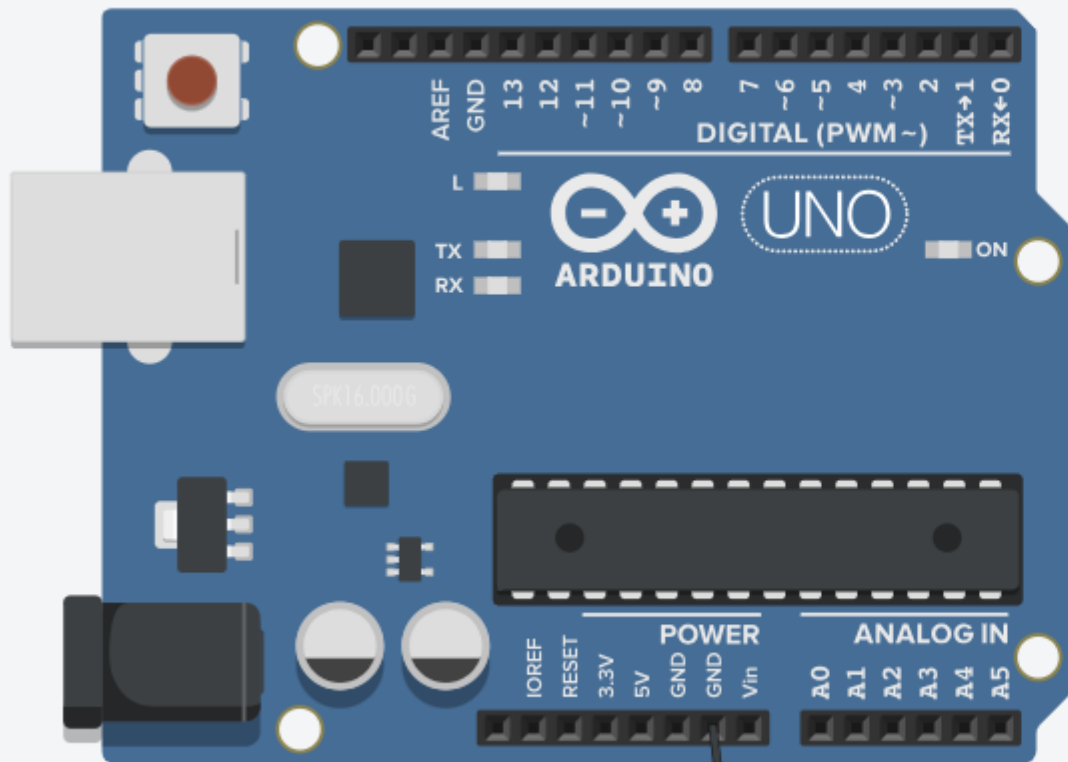
**SCHEDA MADRE PER PC**





# MICROCONTROLLORE

*Inseriamo la scheda a MICROCONTROLLORE e colleghiamo il filo nero sul terminale GND come nell'immagine sotto.*



# ANOTHER WAY TO SAY «EXIT»

												C U R E N T				
													M I C R O P R O C E S S O R			
M		M	I	C	O	P	R	O	C	E	S			S	O	R
E									I					R		
M									R					E		
O									C				N			
R	E	S	I	S	T	O	R						T			
Y									U							
									I							
									T							
				L	E	D										

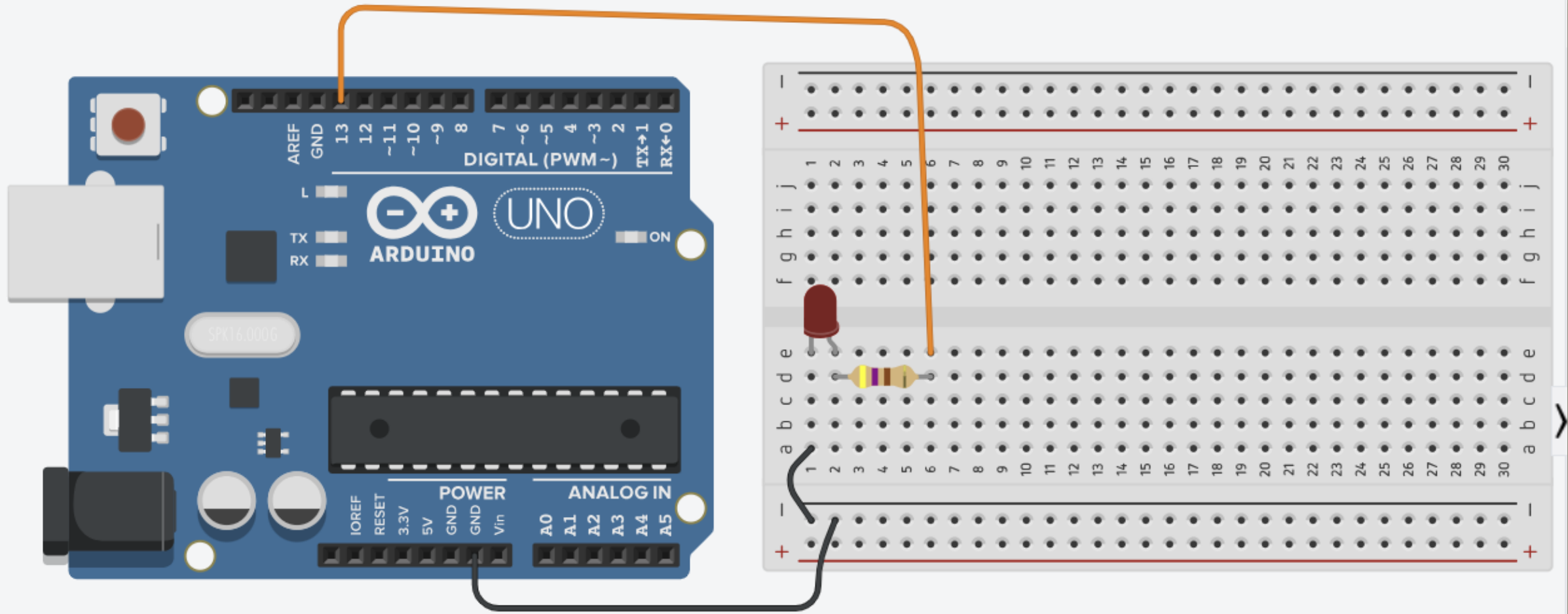
- CIRCUIT
- CURRENT
- MICROPROCESSOR
- MEMORY
- BJT
- WRITE
- TYPE
- INPUT
- READ
- DIR
- LOOP
- SETUP
- LED
- RESISTOR
- DIGITAL
- VOLTAGE
- VARIABLE
- OUTPUT
- BATTERY
- ARDUINO
- POWER
- TOP

# ANOTHER WAY TO SAY «EXIT»



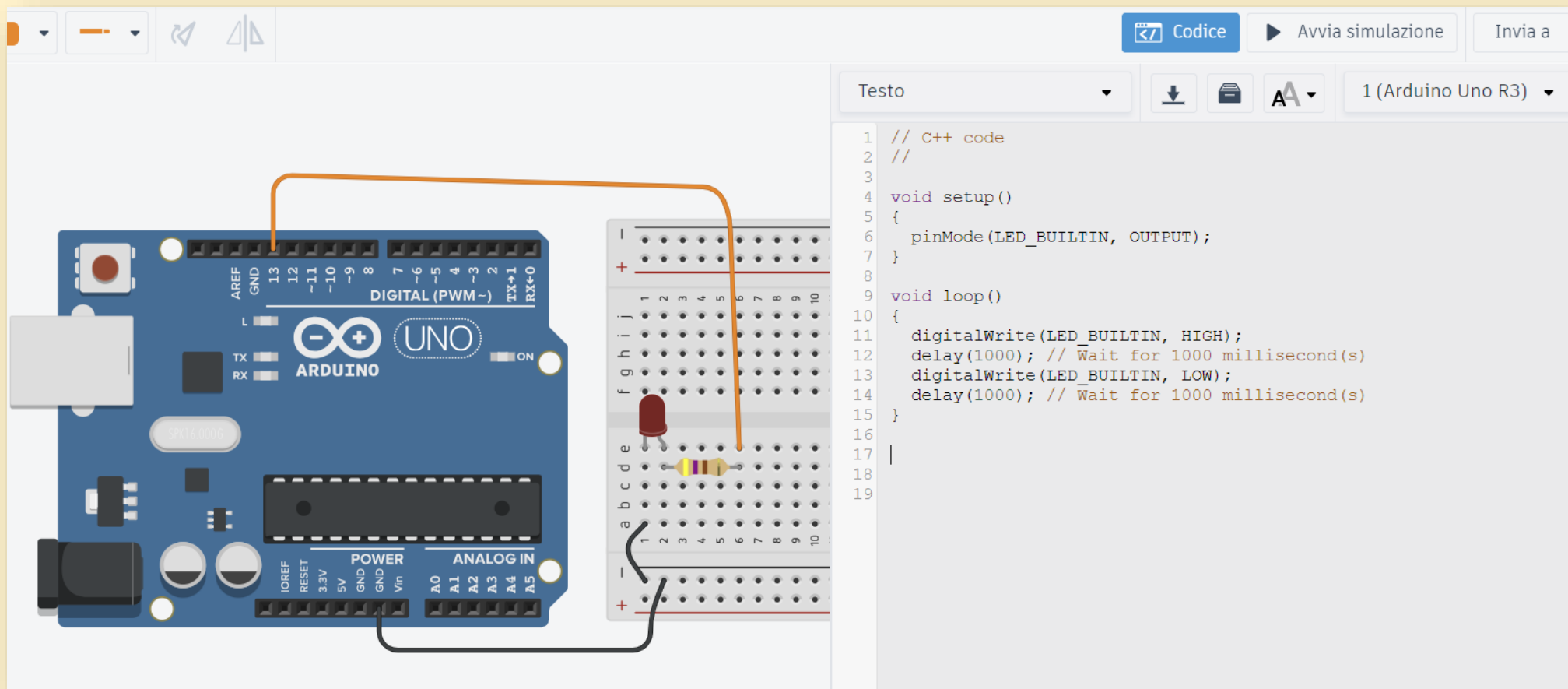
- CIRCUIT
- CURRENT
- MICROPROCESSOR
- MEMORY
- BJT
- WRITE
- TYPE
- INPUT
- READ
- DIR
- LOOP
- SETUP
- LED
- RESISTOR
- DIGITAL
- VOLTAGE
- VARIABLE
- **OUTPUT**
- BATTERY
- ARDUINO
- POWER
- TOP

# TERMINALE DI OUTPUT



# PROGRAMMAZIONE

*Ora procediamo con la scrittura di un semplice programma, visto che il Microcontrollore contiene al suo interno un processore come nei normali PC.*



The image shows a screenshot of the Arduino IDE interface. On the left, there is a 3D rendering of an Arduino Uno R3 board connected to a breadboard. A red LED is connected to digital pin 13, and a resistor is connected between pins 13 and 12. The breadboard also shows a 10k pull-down resistor connected to pins 12 and 13. On the right, the code editor displays the following C++ code:

```
1 // C++ code
2 //
3
4 void setup()
5 {
6   pinMode(LED_BUILTIN, OUTPUT);
7 }
8
9 void loop()
10 {
11   digitalWrite(LED_BUILTIN, HIGH);
12   delay(1000); // Wait for 1000 millisecond(s)
13   digitalWrite(LED_BUILTIN, LOW);
14   delay(1000); // Wait for 1000 millisecond(s)
15 }
16 |
17
18
19
```

# IT'S NOT COSTANT



- CIRCUIT
- CURRENT
- MICROPROCESSOR
- MEMORY
- BJT
- WRITE
- TYPE
- INPUT
- READ
- DIR
- LOOP
- SETUP
- LED
- RESISTOR
- DIGITAL
- VOLTAGE
- VARIABLE
- OUTPUT
- BATTERY
- ARDUINO
- POWER
- TOP

# IT'S NOT COSTANT



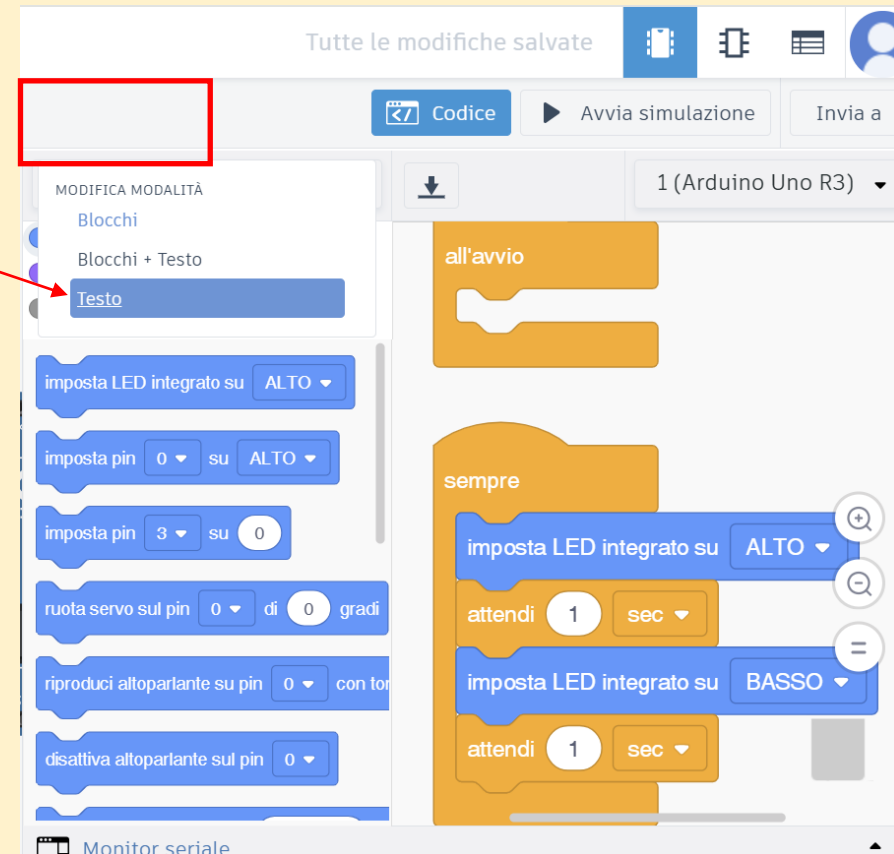
- CIRCUIT
- CURRENT
- MICROPROCESSOR
- MEMORY
- BJT
- WRITE
- TYPE
- INPUT
- READ
- DIR
- LOOP
- SETUP
- LED
- RESISTOR
- DIGITAL
- VOLTAGE
- **VARIABLE**
- OUTPUT
- BATTERY
- ARDUINO
- POWER
- TOP

# PROGRAMMAZIONE

Le **VARIABILI** vengono utilizzate quando si scrive un programma, vediamo come farlo.

Per scrivere un programma su Tinkercad occorre andare sul pulsante **CODICE** e scegliere **TESTO**.

Vi verrà chiesto di confermare la scelta e successivamente potremo scrivere il programma.





# PROGRAMMAZIONE

Aggiungiamo all'inizio la riga `int contatore=0;`

in questo modo abbiamo creato una **VARIABILE** che potrà contenere qualsiasi numero intero, all'inizio gli daremo il valore 0.

```
1 // C++ code
2 //
3
4 int contatore=0;
5
6 void setup()
7 {
8   pinMode(LED_BUILTIN, OUTPUT);
9 }
10
11 void loop()
12 {
13   digitalWrite(LED_BUILTIN, HIGH);
14   delay(1000); // Wait for 1000 millisecond(s)
15   digitalWrite(LED_BUILTIN, LOW);
16   delay(1000); // Wait for 1000 millisecond(s)
17 }
```

# IT'S DONE WHEN YOU TURN ON THE TV FOR THE FIRST TIME



- CIRCUIT
- CURRENT
- MICROPROCESSOR
- MEMORY
- BJT
- WRITE
- TYPE
- INPUT
- READ
- DIR
- LOOP
- SETUP
- LED
- RESISTOR
- DIGITAL
- VOLTAGE
- VARIABLE
- OUTPUT
- BATTERY
- ARDUINO
- POWER
- TOP

# IT'S DONE WHEN YOU TURN ON THE TV FOR THE FIRST TIME



- CIRCUIT
- CURRENT
- MICROPROCESSOR
- MEMORY
- BJT
- WRITE
- TYPE
- INPUT
- READ
- DIR
- LOOP
- **SETUP**
- LED
- RESISTOR
- DIGITAL
- VOLTAGE
- VARIABLE
- OUTPUT
- BATTERY
- ARDUINO
- POWER
- TOP

# PROGRAMMAZIONE

*Dentro alle parentesi graffe del SETUP, vengono messe le istruzioni che verranno eseguite solo all'accensione. Modifichiamo l'istruzione presente con quella che vedere in figura. In questo modo diremo che il terminale 13 (dove è connesso il LED) è un uscita (OUTPUT).*

```
1 // C++ code
2 //
3
4 int contatore=0;
5
6 void setup()
7 {
8   pinMode(13, OUTPUT);
9 }
10
11 void loop()
12 {
13   digitalWrite(LED_BUILTIN, HIGH);
14   delay(1000); // Wait for 1000 millisecond(s)
15   digitalWrite(LED_BUILTIN, LOW);
16   delay(1000); // Wait for 1000 millisecond(s)
17 }
```

# IT REPEATS ITSELF ENDESSLY



- CIRCUIT
- CURRENT
- MICROPROCESSOR
- MEMORY
- BJT
- WRITE
- TYPE
- INPUT
- READ
- DIR
- LOOP
- SETUP
- LED
- RESISTOR
- DIGITAL
- VOLTAGE
- VARIABLE
- OUTPUT
- BATTERY
- ARDUINO
- POWER
- TOP

# IT REPEATS ITSELF ENDESSLY



- CIRCUIT
- CURRENT
- MICROPROCESSOR
- MEMORY
- BJT
- WRITE
- TYPE
- INPUT
- READ
- DIR
- **LOOP**
- SETUP
- LED
- RESISTOR
- DIGITAL
- VOLTAGE
- VARIABLE
- OUTPUT
- BATTERY
- ARDUINO
- POWER
- TOP

# PROGRAMMAZIONE

*Tutte le istruzioni che sono scritte tra le parentesi graffe del LOOP, vengono eseguite ciclicamente all'infinito. Qui dobbiamo scrivere il programma che dovrà essere eseguito dalla scheda.*

```
1 // C++ code
2 //
3
4 int contatore=0;
5
6 void setup()
7 {
8   pinMode(13, OUTPUT);
9 }
10
11 void loop()
12 {
13   digitalWrite(LED_BUILTIN, HIGH);
14   delay(1000); // Wait for 1000 millisecond(s)
15   digitalWrite(LED_BUILTIN, LOW);
16   delay(1000); // Wait for 1000 millisecond(s)
17 }
18
```





# IT'S NOT ANALOG



- CIRCUIT
- CURRENT
- MICROPROCESSOR
- MEMORY
- BJT
- WRITE
- TYPE
- INPUT
- READ
- DIR
- LOOP
- SETUP
- LED
- RESISTOR
- DIGITAL
- VOLTAGE
- VARIABLE
- OUTPUT
- BATTERY
- ARDUINO
- POWER
- TOP







# THE VERB THAT INDICATE THE USE OF THE PEN



- CIRCUIT
- CURRENT
- MICROPROCESSOR
- MEMORY
- BJT
- **WRITE**
- TYPE
- INPUT
- READ
- DIR
- LOOP
- SETUP
- LED
- RESISTOR
- DIGITAL
- VOLTAGE
- VARIABLE
- OUTPUT
- BATTERY
- ARDUINO
- POWER
- TOP

# PROGRAMMAZIONE

*Modifichiamo le due istruzioni DIGITALWRITE che troviamo nel LOOP come in figura.*

*Con questa istruzione diciamo alla scheda di scrivere sul terminale 13 (dove è collegato il LED) un livello HIGH e successivamente un livello LOW.*

*Quando il terminale si trova a livello HIGH dal terminale 13 uscirà corrente ed il LED si accenderà.*

*Quando il terminale si trova a livello LOW dal terminale 13 non uscirà corrente ed il LED si spegnerà.*

```
1 // C++ code
2 //
3
4 int contatore=0;
5
6 void setup()
7 {
8   pinMode(13, OUTPUT);
9 }
10
11 void loop()
12 {
13   digitalWrite(13, HIGH);
14   delay(1000); // Wait for 1000 millisecond(s)
15   digitalWrite(13, LOW);
16   delay(1000); // wait for 1000 millisecond(s)
17 }
18
```

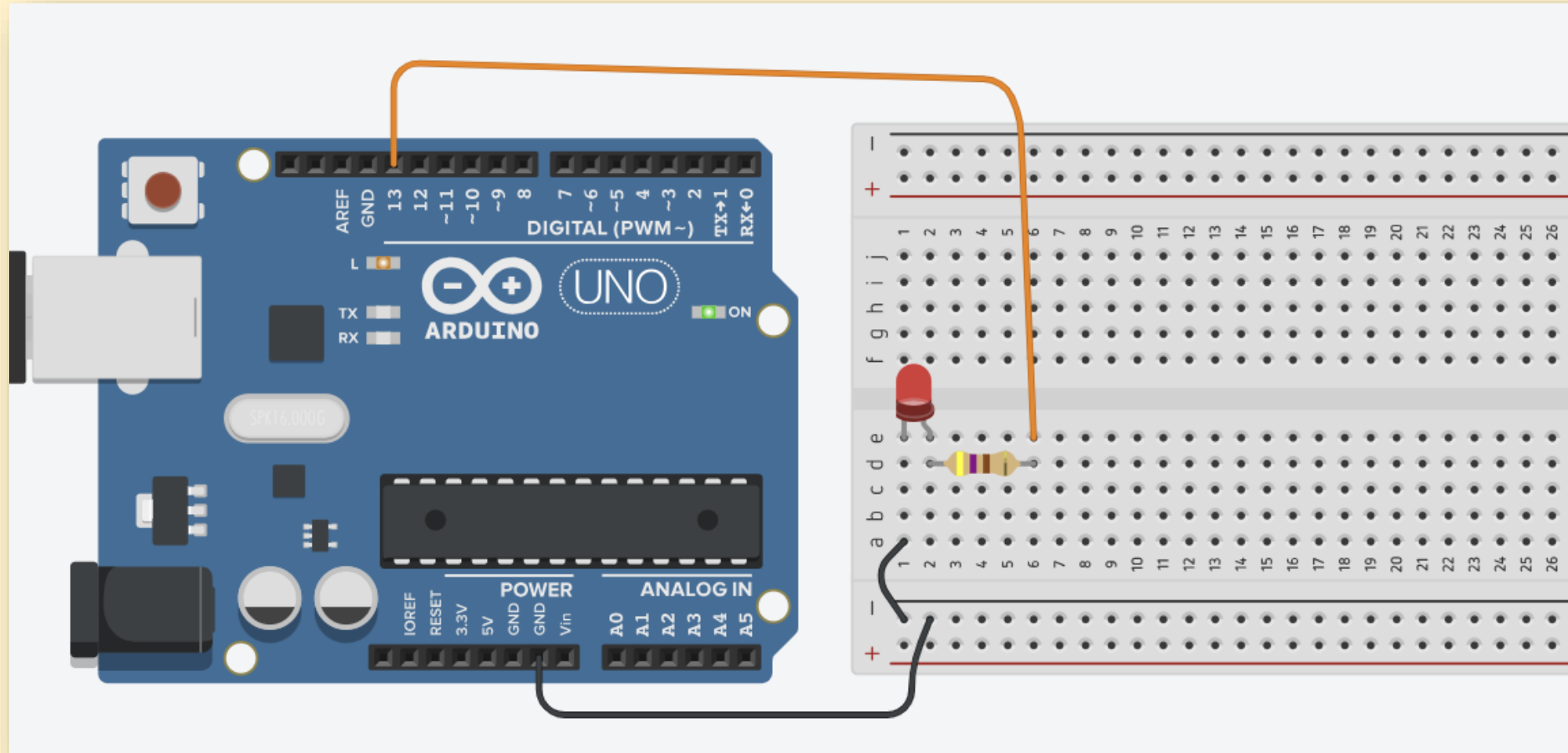
# PROGRAMMAZIONE

*Tra le due istruzioni DIGITALWRITE c'è un'istruzione che si chiama DELAY che serve a bloccare il programma per il tempo indicato tra le parentesi, con delay(1000) il programma si ferma per 1000 millesimi di secondo, cioè un secondo. In questo modo vedremo il LED accendersi e spegnersi.*

```
1 // C++ code
2 //
3
4 int contatore=0;
5
6 void setup()
7 {
8   pinMode(13, OUTPUT);
9 }
10
11 void loop()
12 {
13   digitalWrite(13, HIGH);
14   delay(1000); // Wait for 1000 millisecond(s)
15   digitalWrite(13, LOW);
16   delay(1000); // Wait for 1000 millisecond(s)
17 }
18
```

# PROGRAMMAZIONE

*Per provare il circuito ed il programma scritto premere sul pulsante «Avvia simulazione» in alto a destra.*



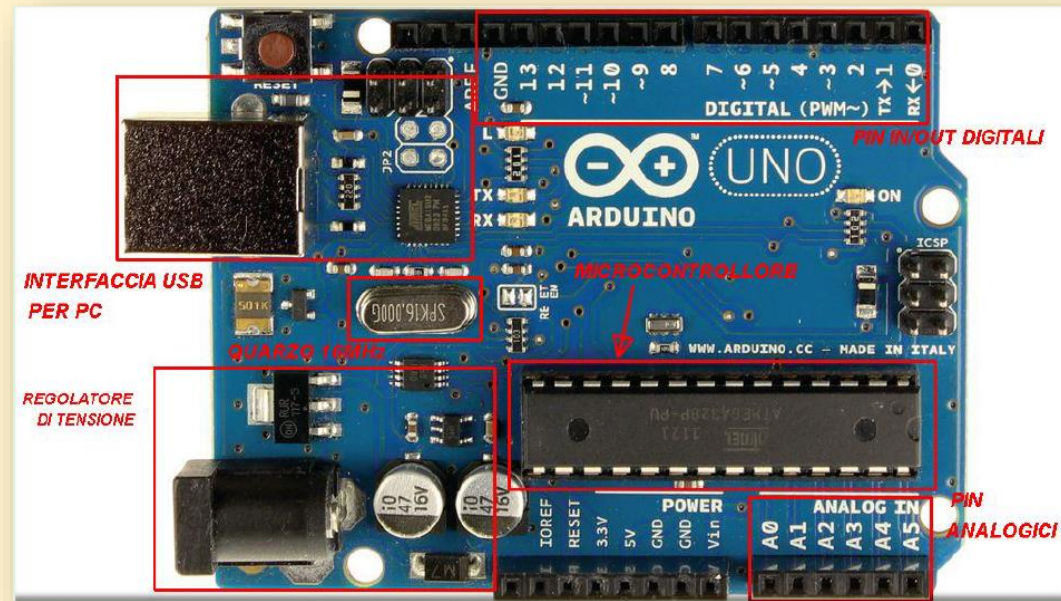






# ARDUINO

La scheda programmabile utilizzata si chiama ARDUINO UNO e fa parte del progetto ARDUINO nato in Italia nel 2005. Il progetto ARDUINO prevede un insieme di schede elettroniche programmabili, come quella utilizzata, ed un software per la loro programmazione scaricabile al sito ufficiale di ARDUINO.

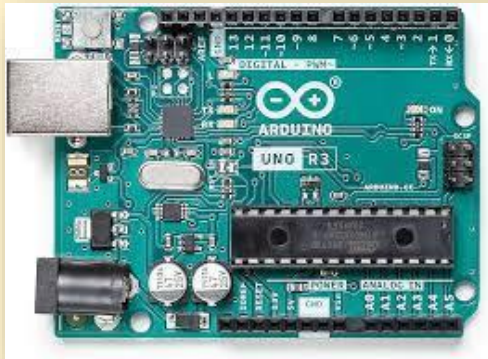


Nella sezione TUTORIAL di questo sito troverete molte istruzioni sull'utilizzo del software e della scheda.  
[www.danielepostacchini.it](http://www.danielepostacchini.it)

# Arduino UNO=Speedy Gonzales

*Perché abbiamo iniziato parlando di Speedy Gonzales? Semplice perché una scheda programmabile come Arduino uno è in grado di eseguire circa **8 milioni di istruzioni al secondo (8 MIPS)**.*

*Anche se non è la scheda programmabile più veloce sul mercato, è sicuramente molto conosciuta e poco costosa. Insomma come se fosse un piccolo ma veloce computer utile a far funzionare qualsiasi meccanismo.*



=



*Ma non facciamoci ingannare dalla velocità, perché come avete visto, la scheda è in grado di eseguire solo operazioni semplici ed è in grado di capire solo i numeri binari cioè 0 ed 1.*

*A differenza del topolino, sicuramente molto furbo, la scheda ARDUINO UNO, come qualsiasi altro computer, è veloce ma estremamente stupida, è **sempre il programmatore a metterci le idee ed il cervello.***

# PROGRAMMIAMO ARDUINO

*Possiamo ora programmare la vera scheda senza lavorare con la simulazione come fatto fino ad ora.*

*Per farlo dobbiamo installare il software scaricabile al seguente link: <https://drive.google.com/file/d/1NXjyVWlfy2EyqqC7XkOyGPPiagJFnMKC/view>*

*Successivamente scriviamo il seguente programma, facendo attenzione alle lettere maiuscole e minuscole.*

```
File Modifica Sketch Strumenti Aiuto
sketch_nov22a $
int contatore=0;

void setup() {
  pinMode(13,OUTPUT);
}

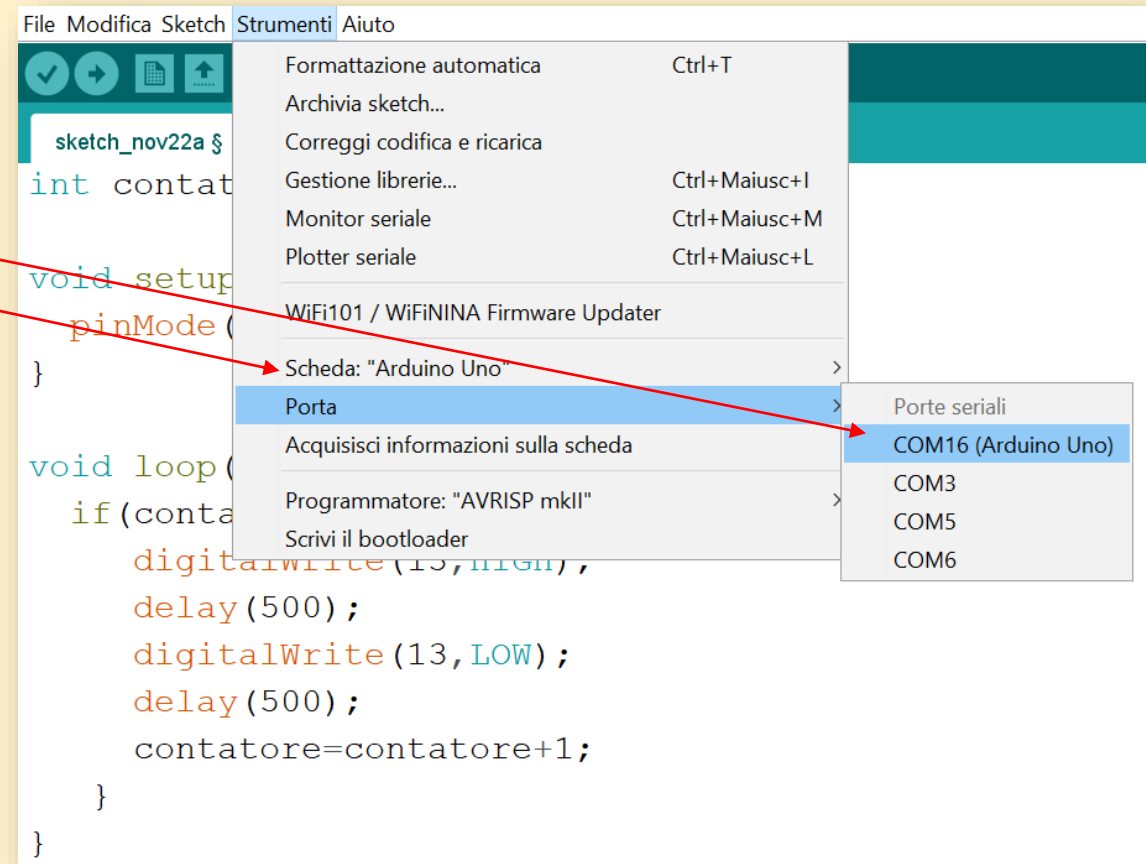
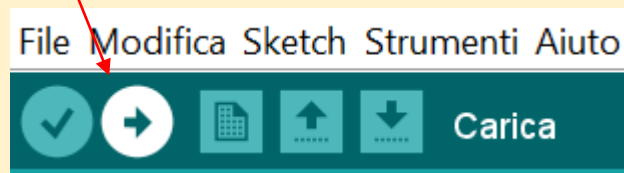
void loop() {
  if(contatore<5){
    digitalWrite(13,HIGH);
    delay(500);
    digitalWrite(13,LOW);
    delay(500);
    contatore=contatore+1;
  }
}
```

# PROGRAMMIAMO ARDUINO

Al termine dobbiamo caricare il programma sulla scheda Arduino per fare questo occorre seguire le seguenti indicazioni:

1) Collegare la scheda Arduino al PC tramite il cavo USB in dotazione, ed impostare sulla voce **STRUMENTI** la **scheda** e la **porta**

2) Successivamente cliccando sul secondo pulsante in alto con la freccia si carica il programma sulla scheda.



# PROGRAMMIAMO ARDUINO

Se andiamo ad analizzare il programma troviamo delle differenze rispetto a quello scritto in precedenza. In particolare vediamo che in questo programma c'è un **ciclo if**.

Il ciclo if è una struttura che consente di eseguire delle istruzioni se una determinata condizione risulta vera.

In questo caso la condizione è quella legata al valore della variabile **contatore**. Se questa ha un valore inferiore a 5 allora si esegue il lampeggio del led come fatto in precedenza, in caso contrario non si esegue nulla.

Ed ora proviamo autonomamente a realizzare un circuito con due Led in modo che venga effettuato un lampeggio alternato tra i due per 10 volte. **Buon lavoro!!!**

```
File Modifica Sketch Strumenti Aiuto
sketch_nov22a $
int contatore=0;

void setup() {
  pinMode(13,OUTPUT);
}

void loop() {
  if(contatore<5){
    digitalWrite(13,HIGH);
    delay(500);
    digitalWrite(13,LOW);
    delay(500);
    contatore=contatore+1;
  }
}
```