



**PORTO
DESIGN
FACTORY**

öBot CERN
2018

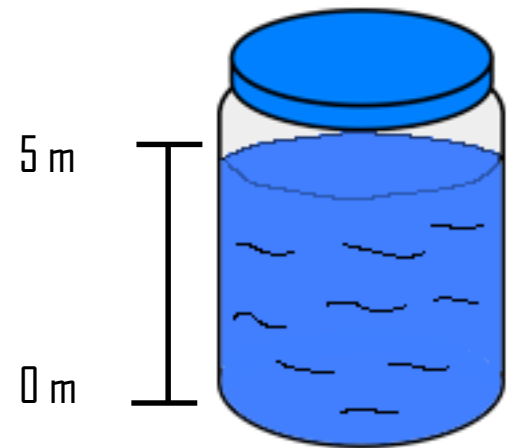
What is Voltage and Amperes??



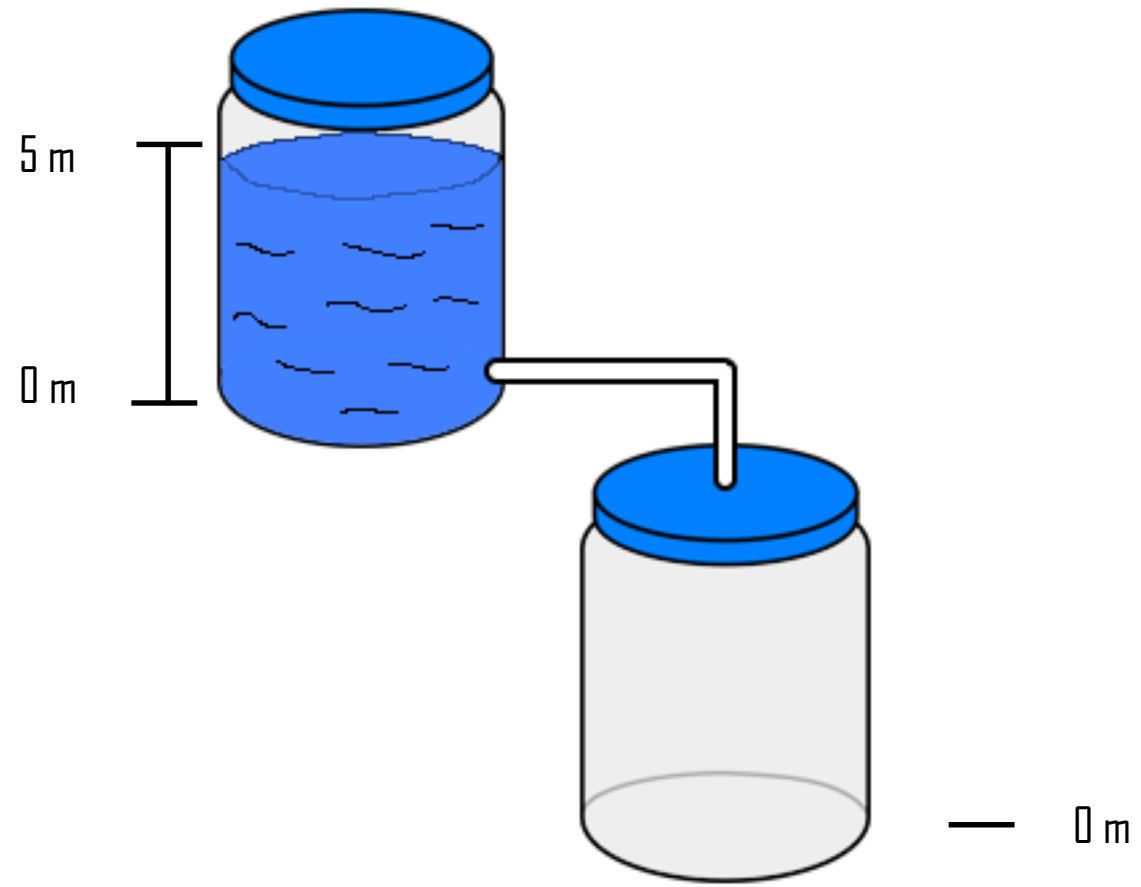
Container 1



Container 1

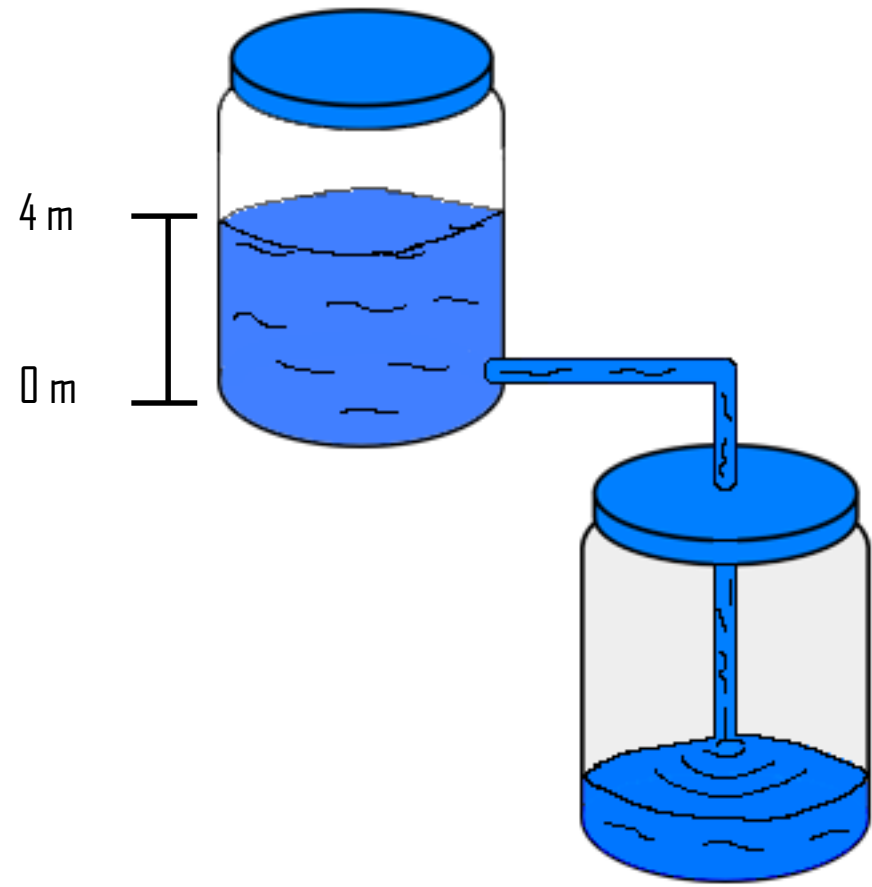


Container 1



Container 2

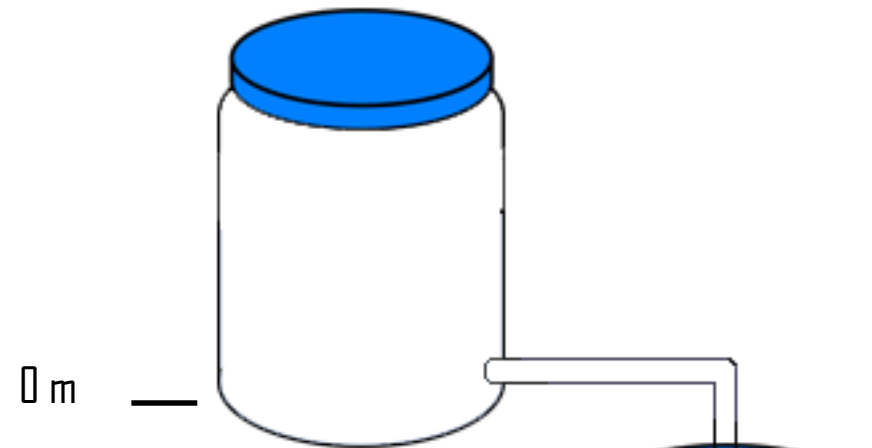
Container 1



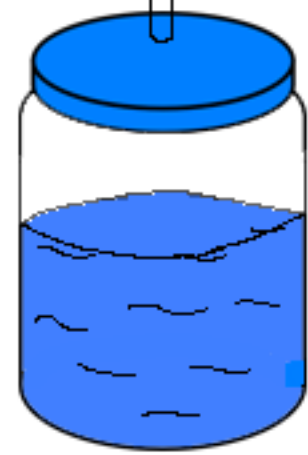
Container 2

H
1 m
0 m

Container 1



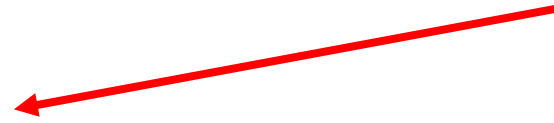
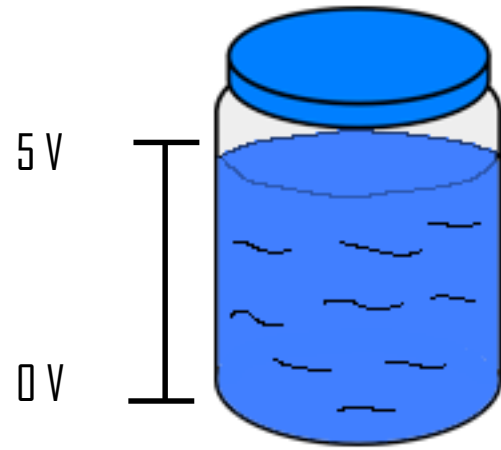
0 m

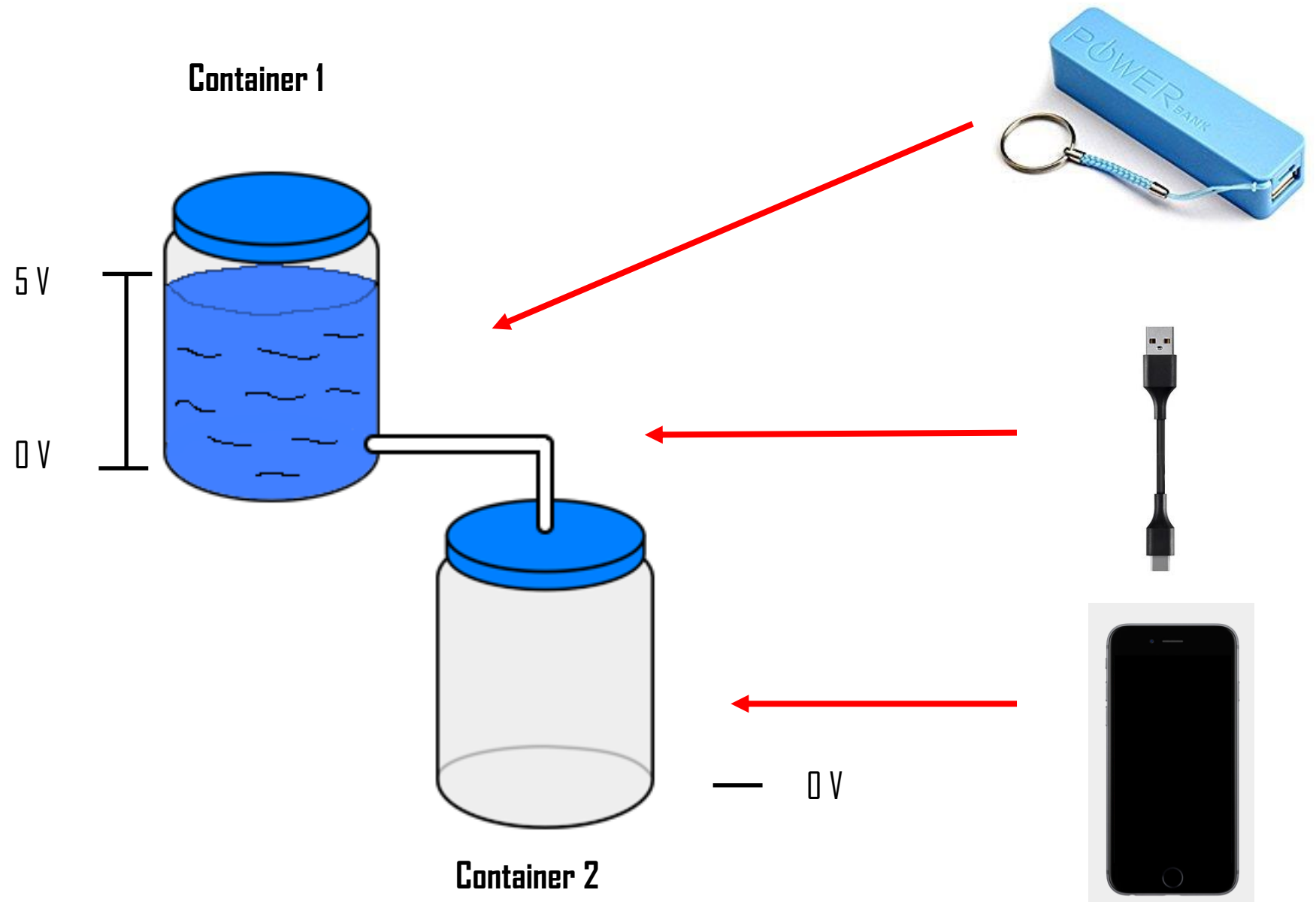


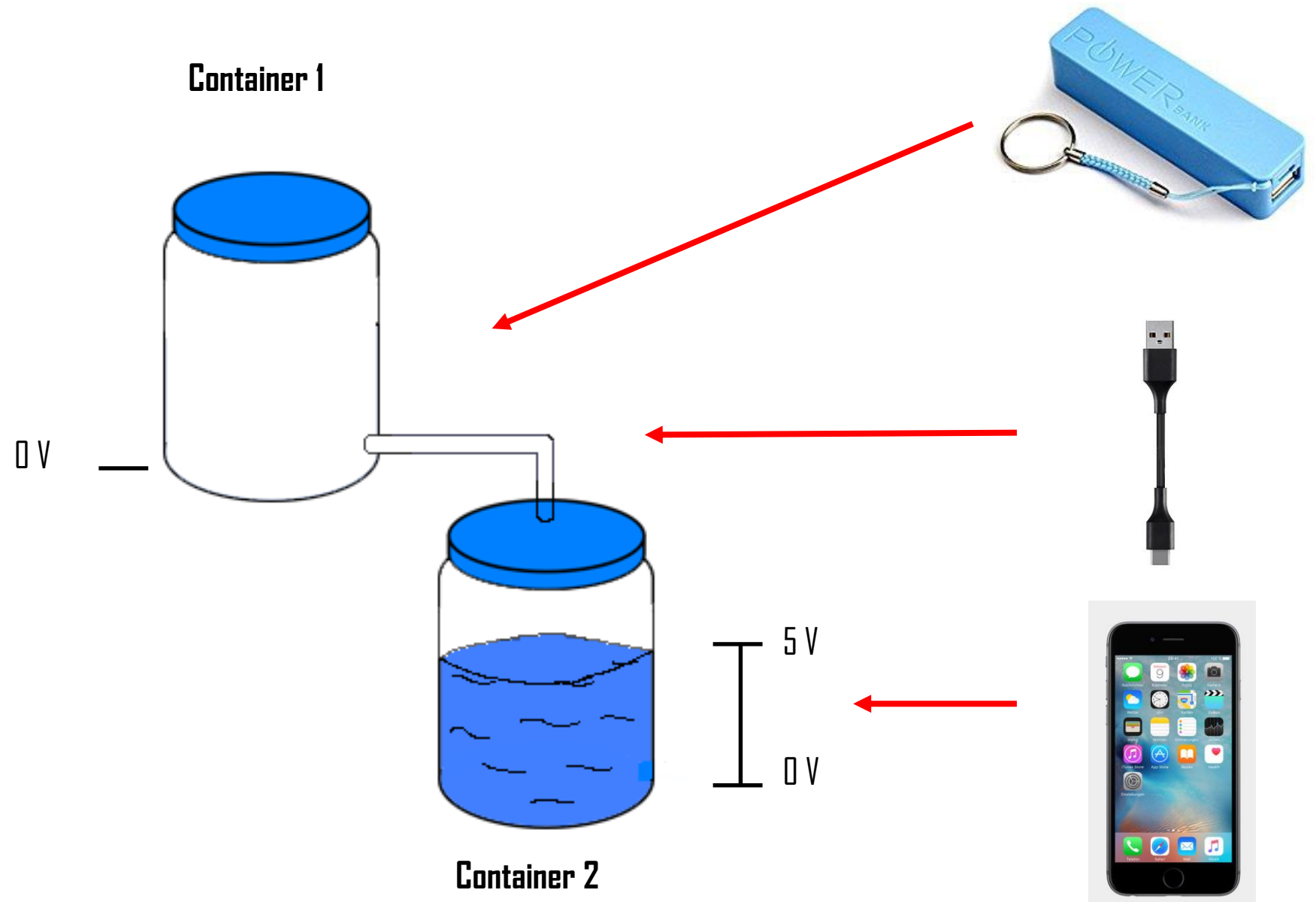
Container 2

5 m
0 m

Container 1

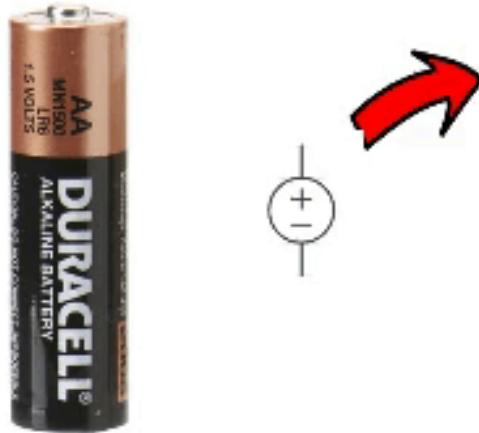






How to turn on an LED?

Current flows always from the positive pin to the negative



Components

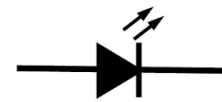
Power supply



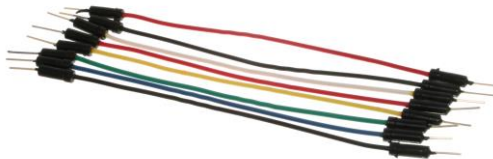
Resistance



LED



Jumpers



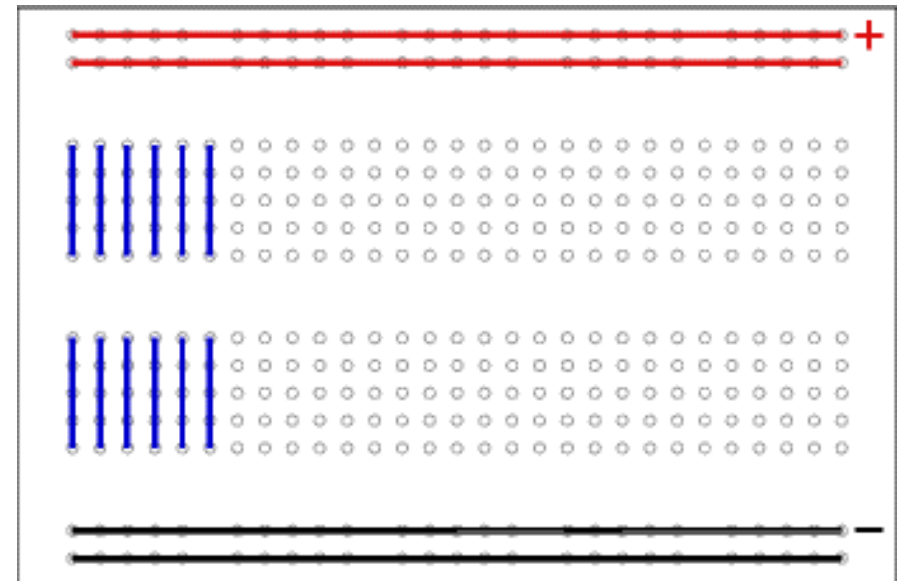
Breadboard

Perfect for prototyping

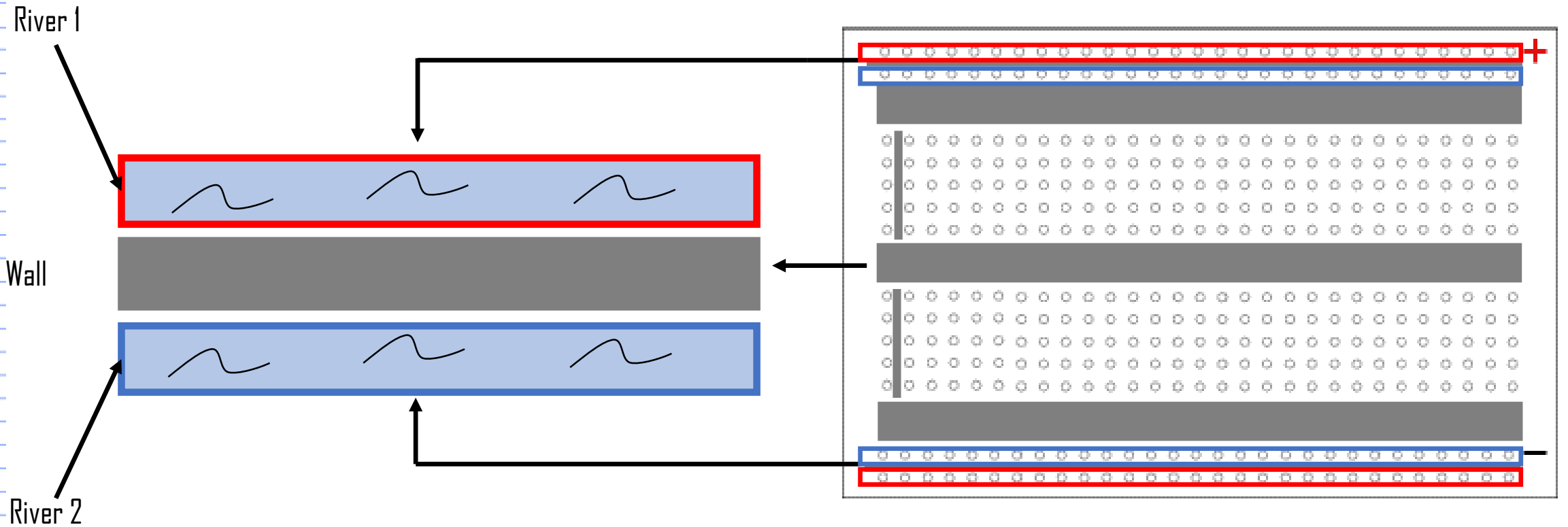
Fast connection, no need to solder every connection

Reusable

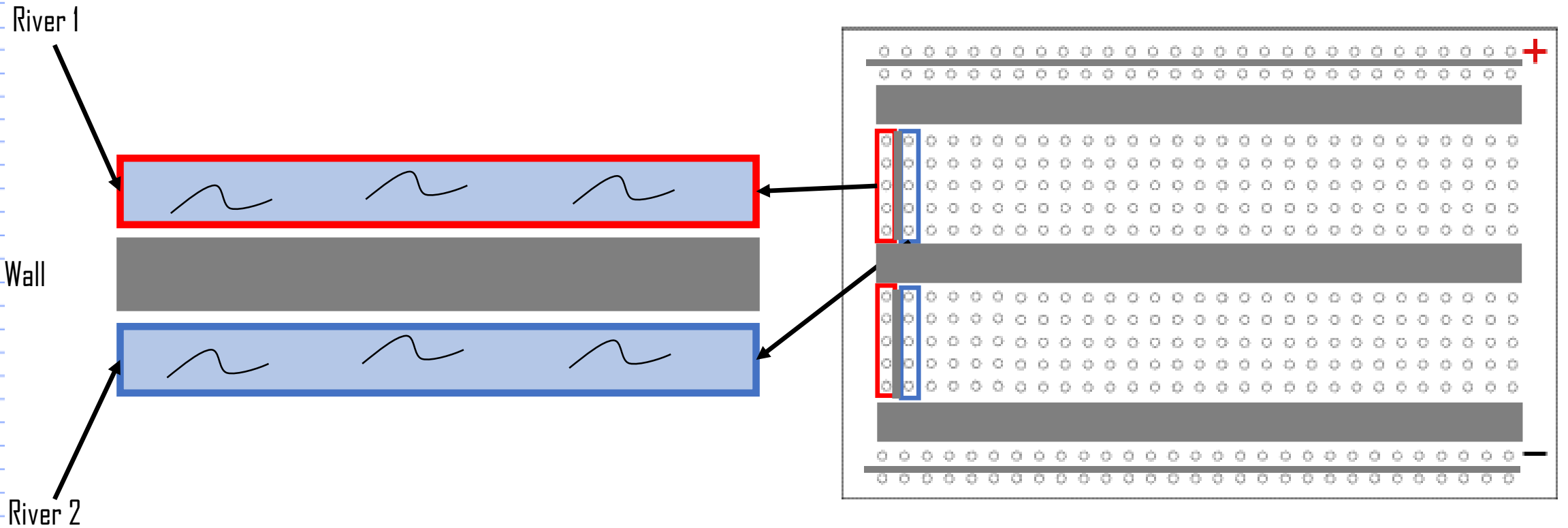
The lines that are shown in the image
are connected



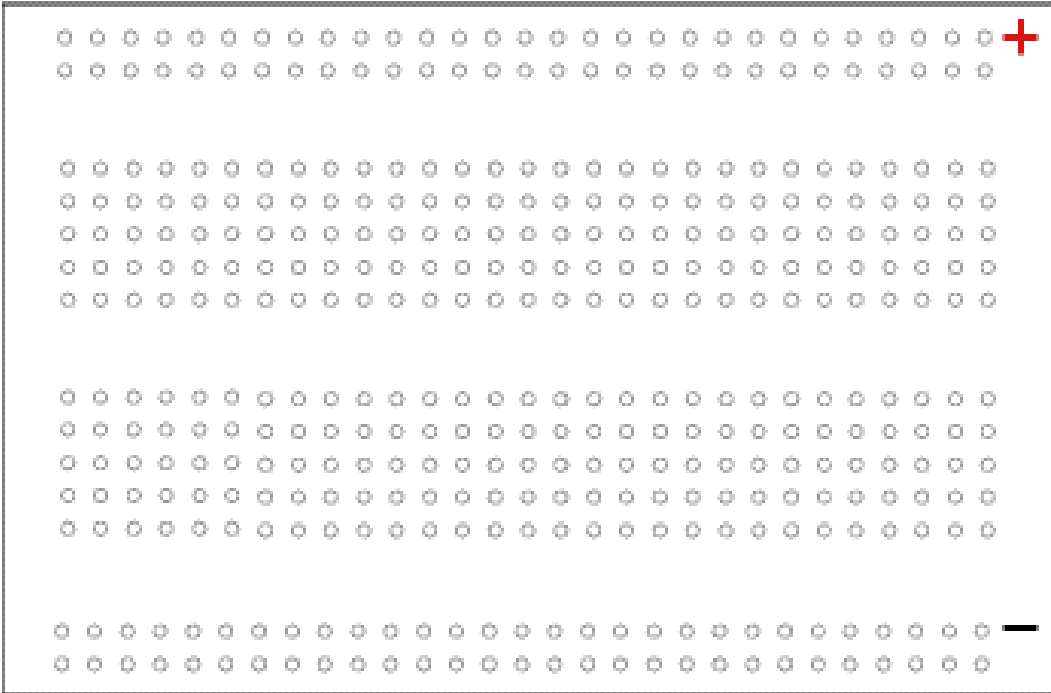
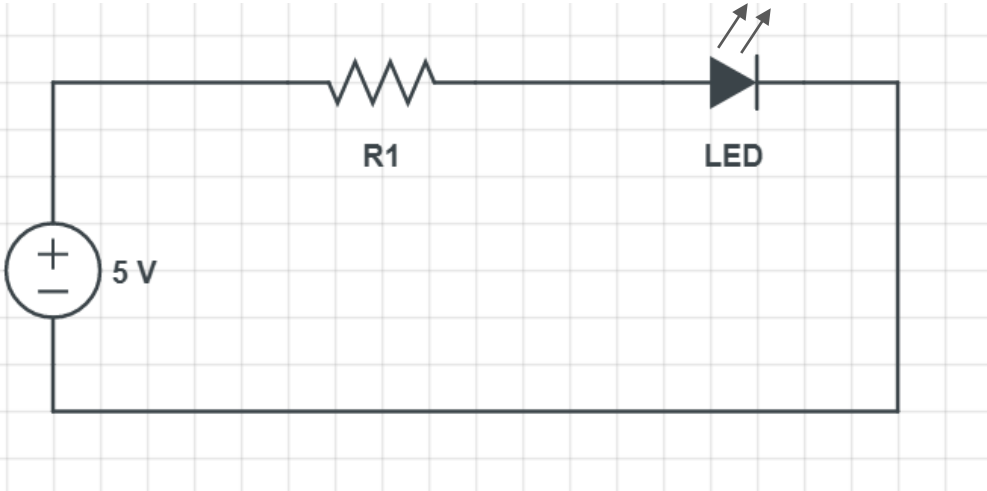
Breadboard



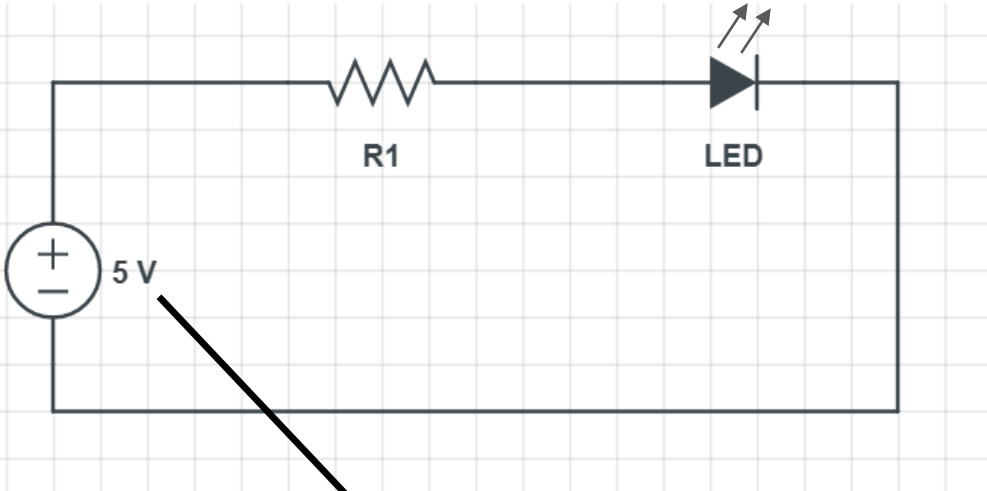
Breadboard



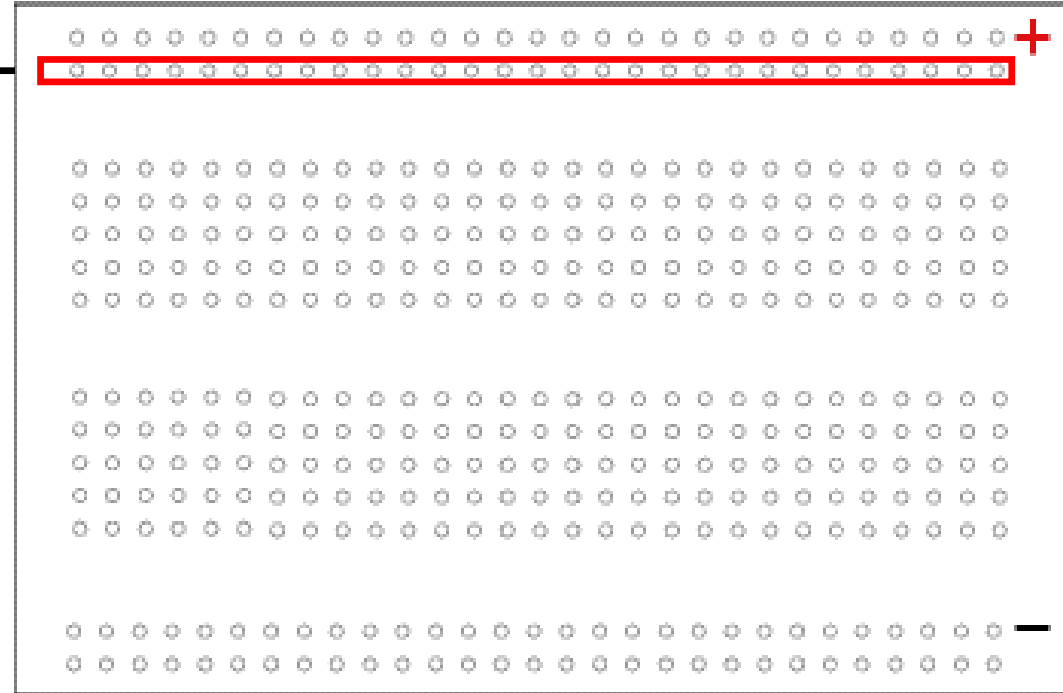
Assemble our first circuit



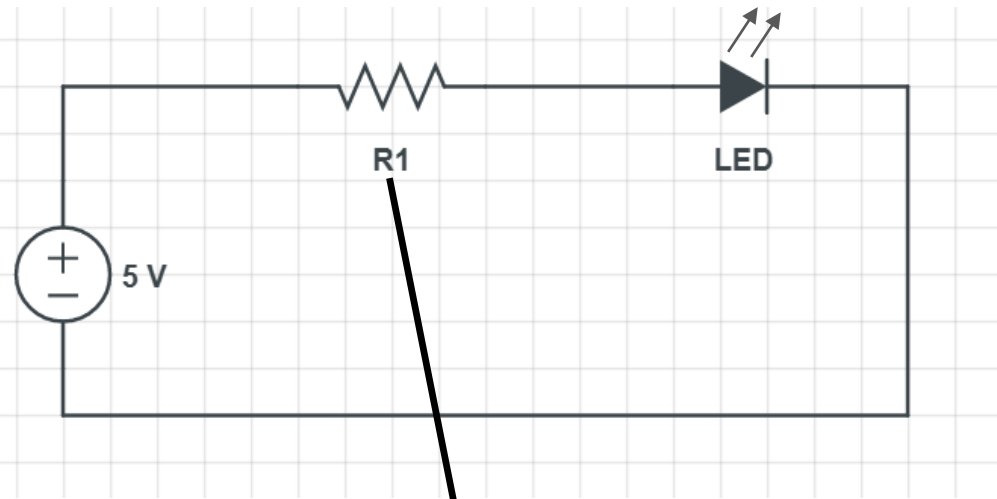
Assemble our first circuit



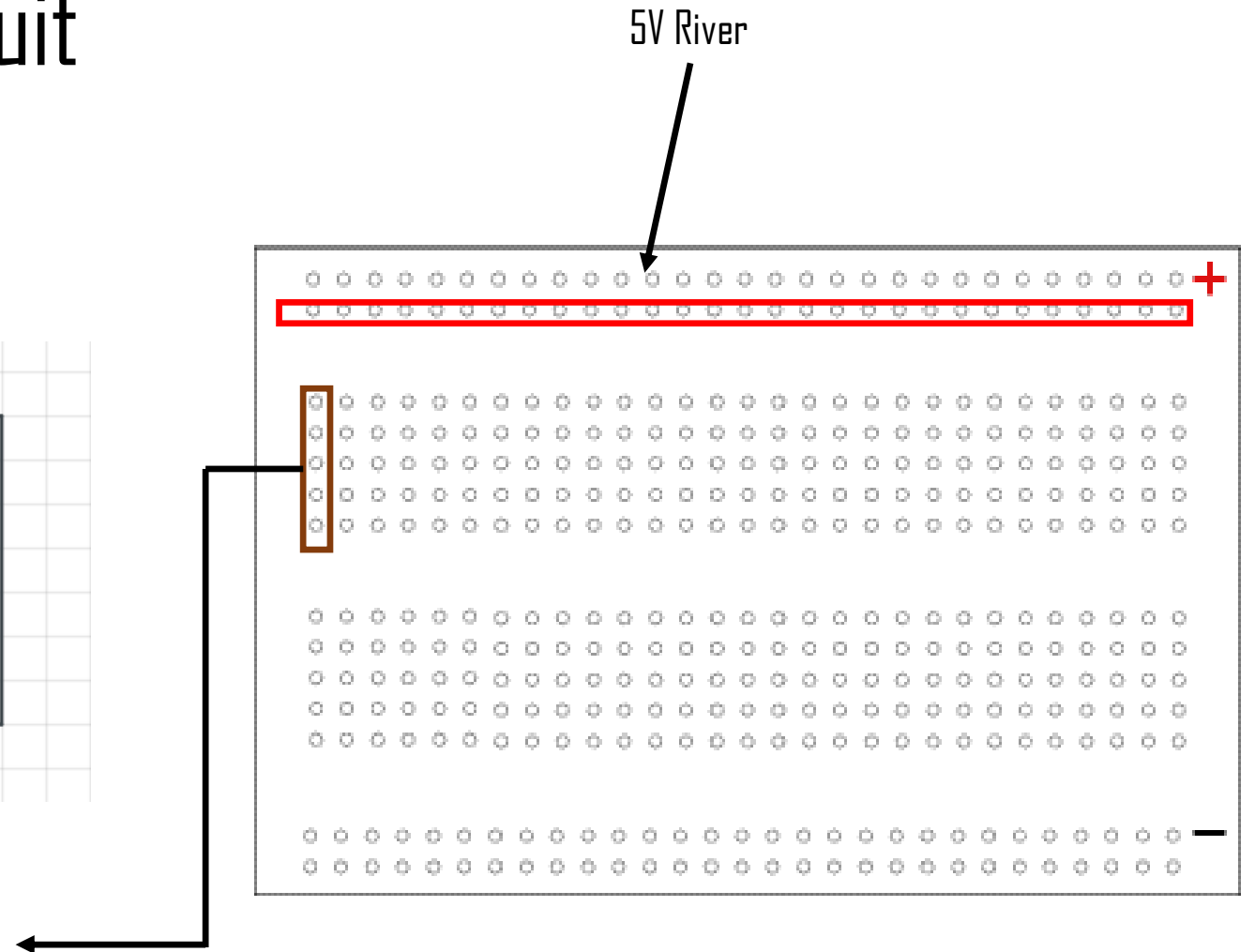
5V River



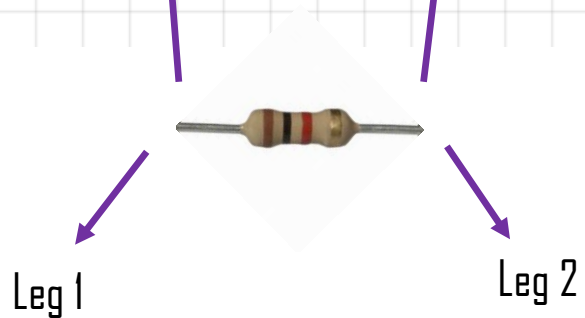
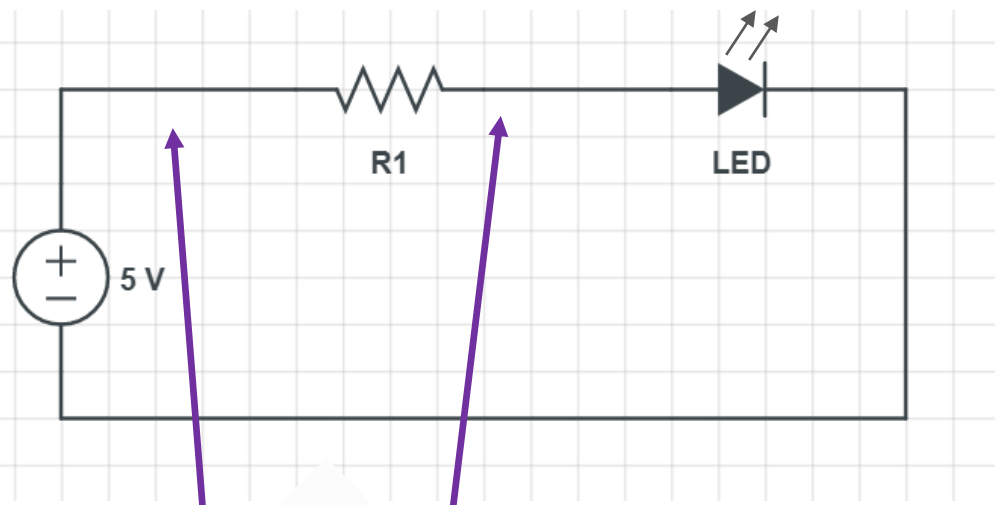
Assemble our first circuit



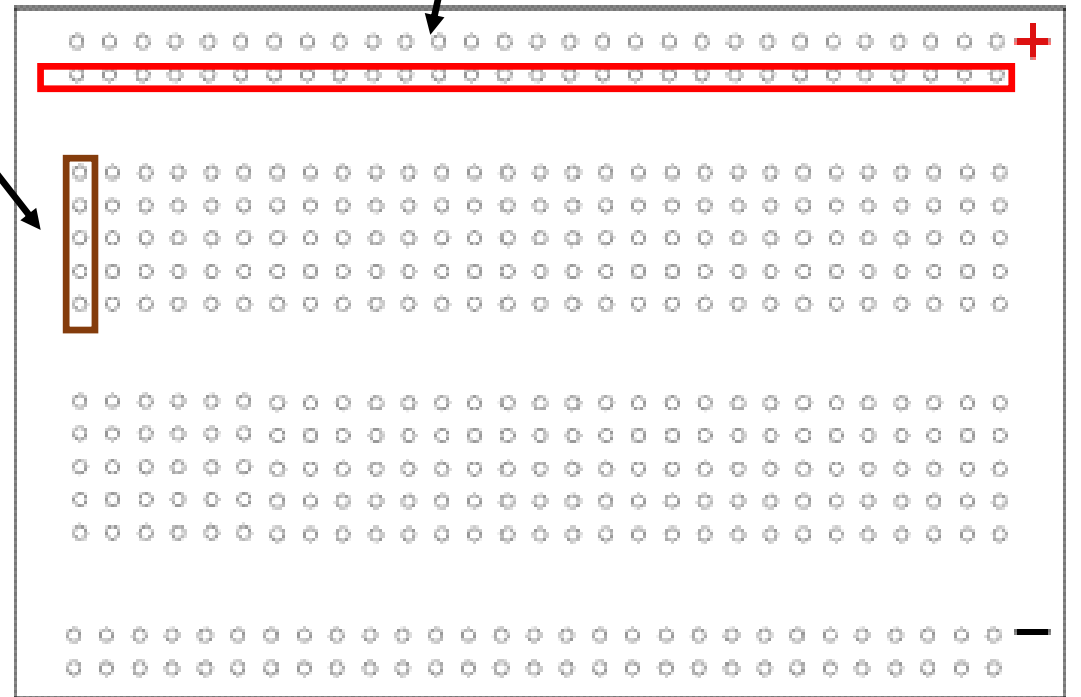
Resistance River



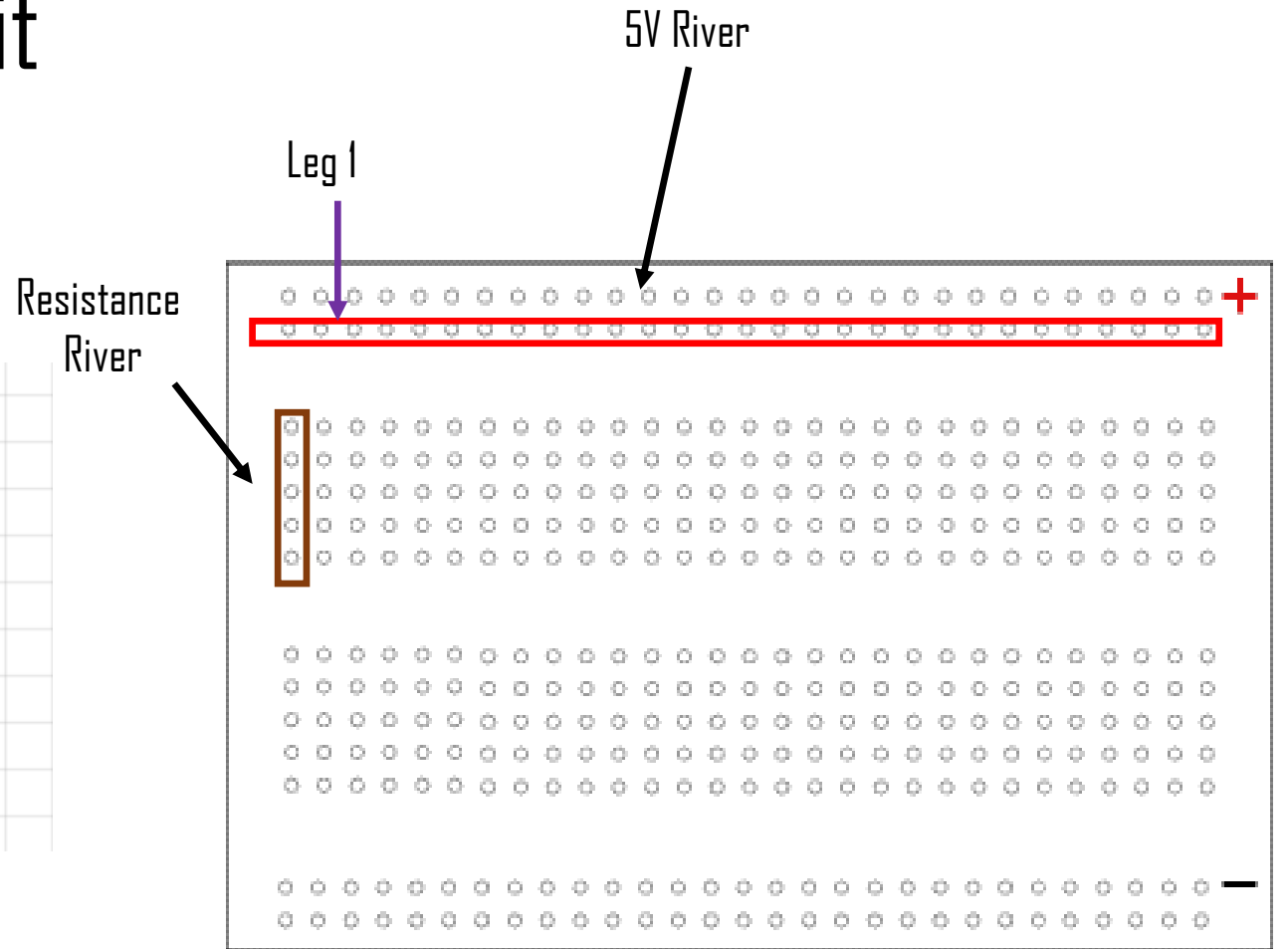
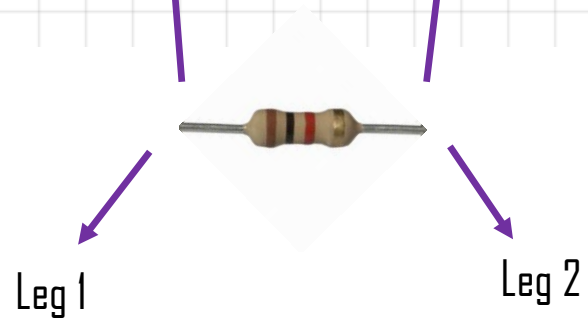
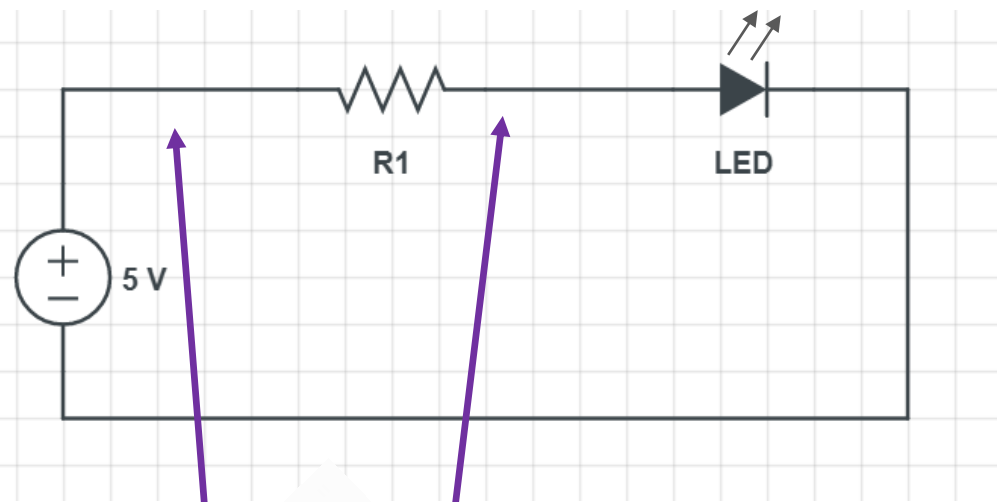
Assemble our first circuit



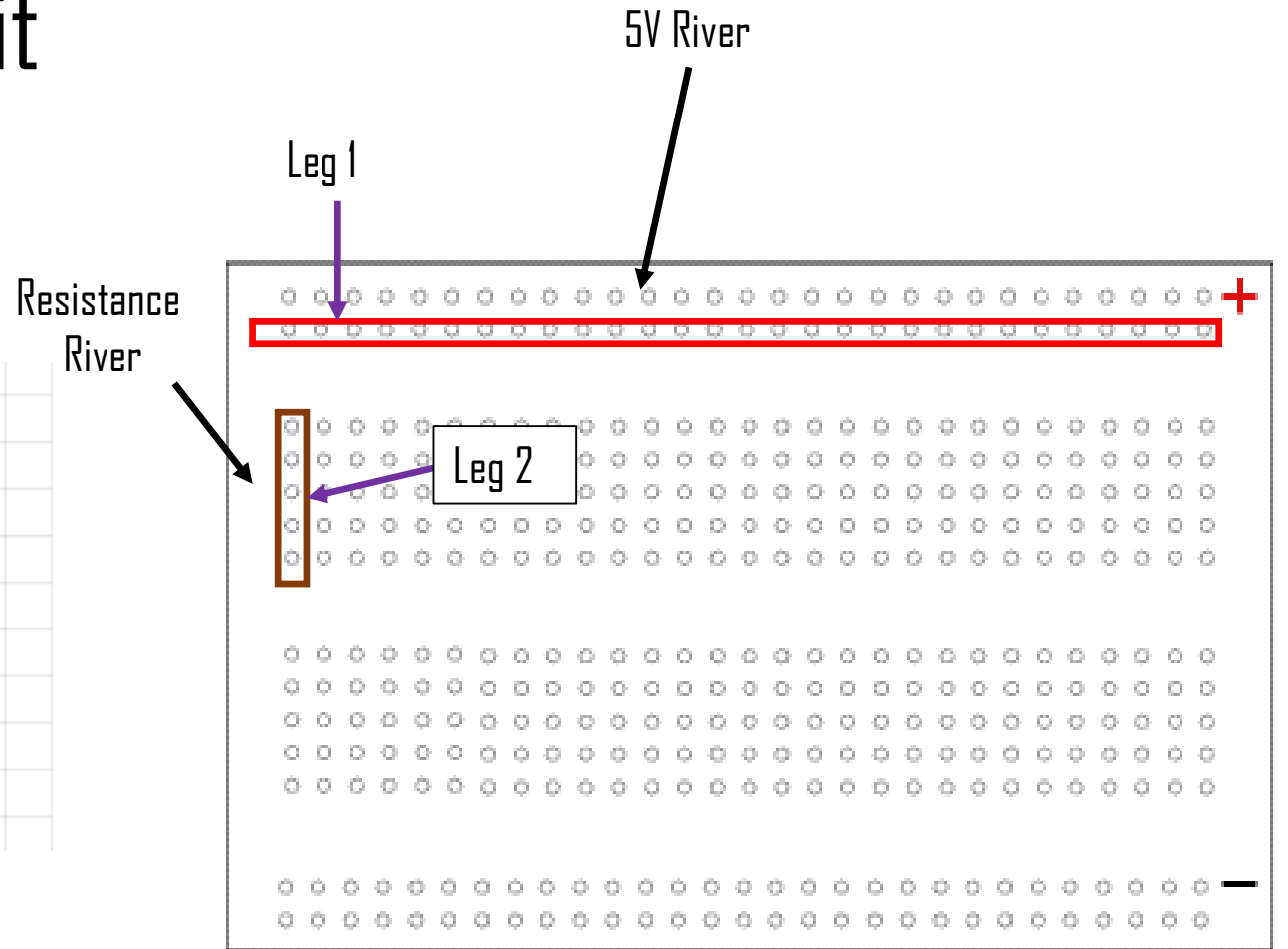
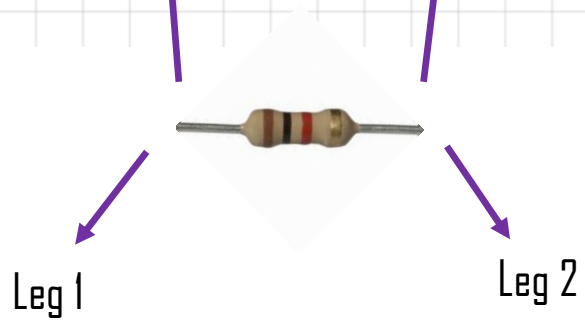
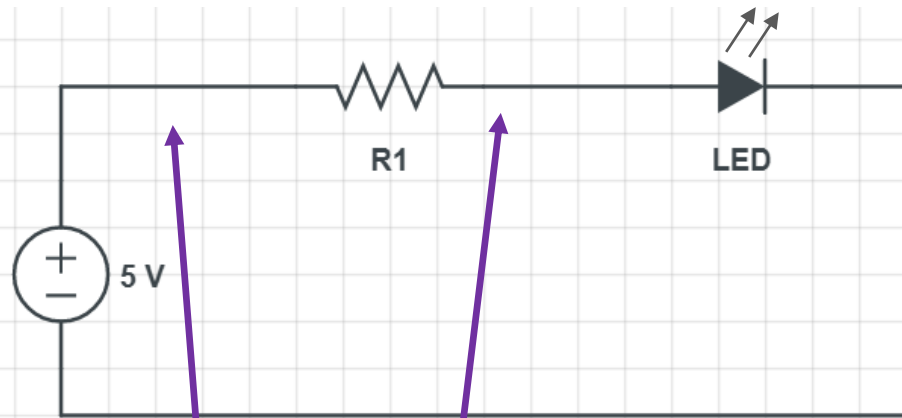
Resistance
River



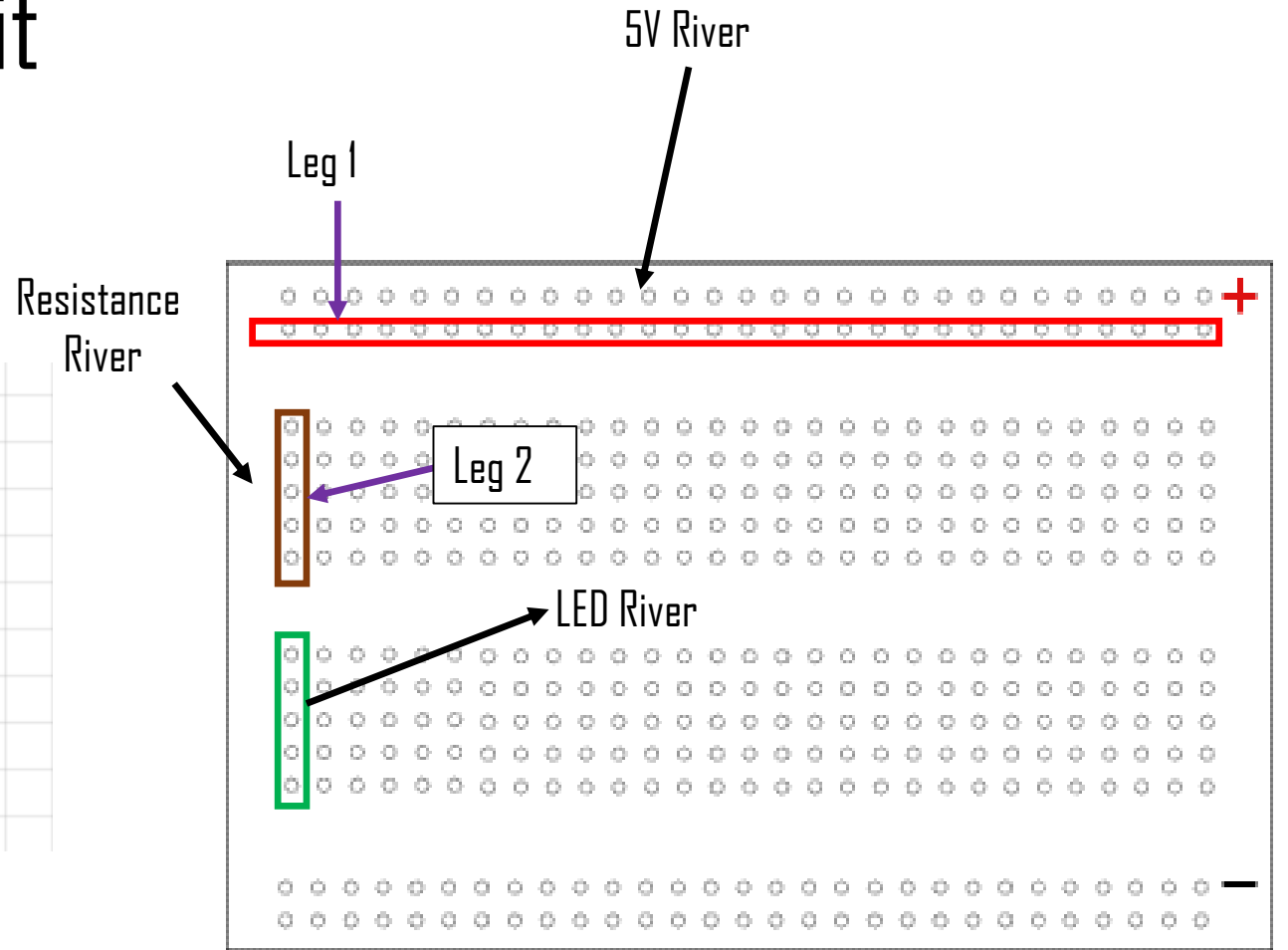
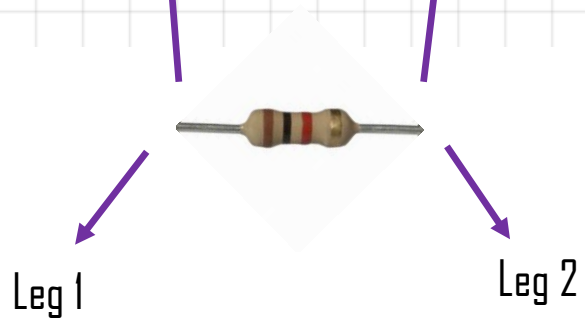
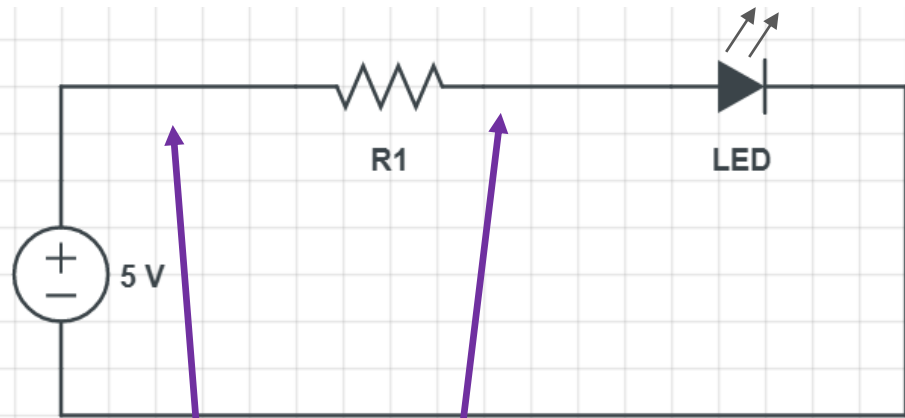
Assemble our first circuit



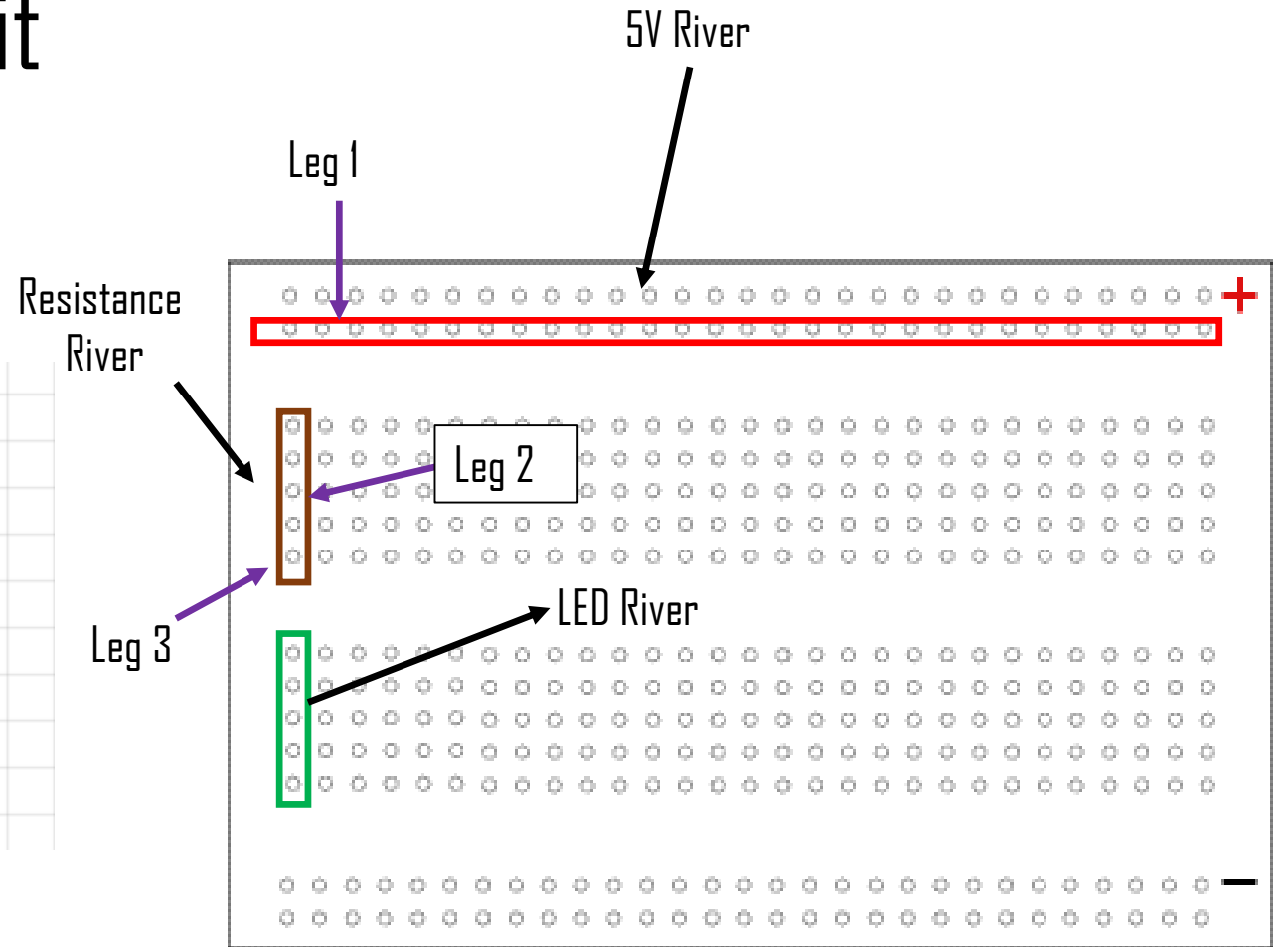
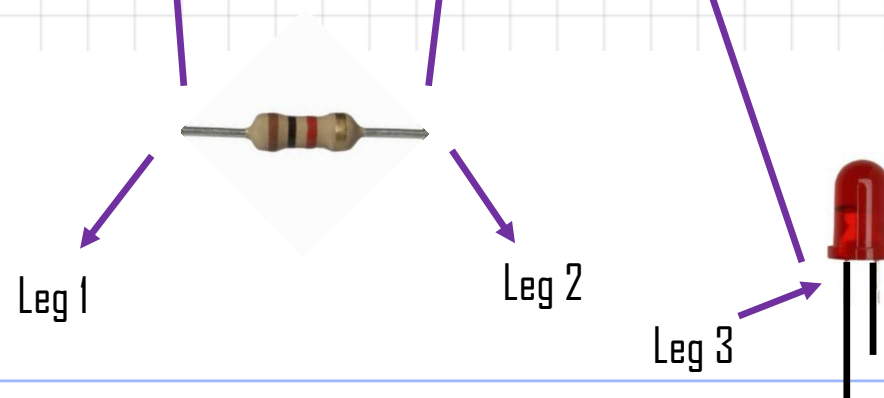
Assemble our first circuit



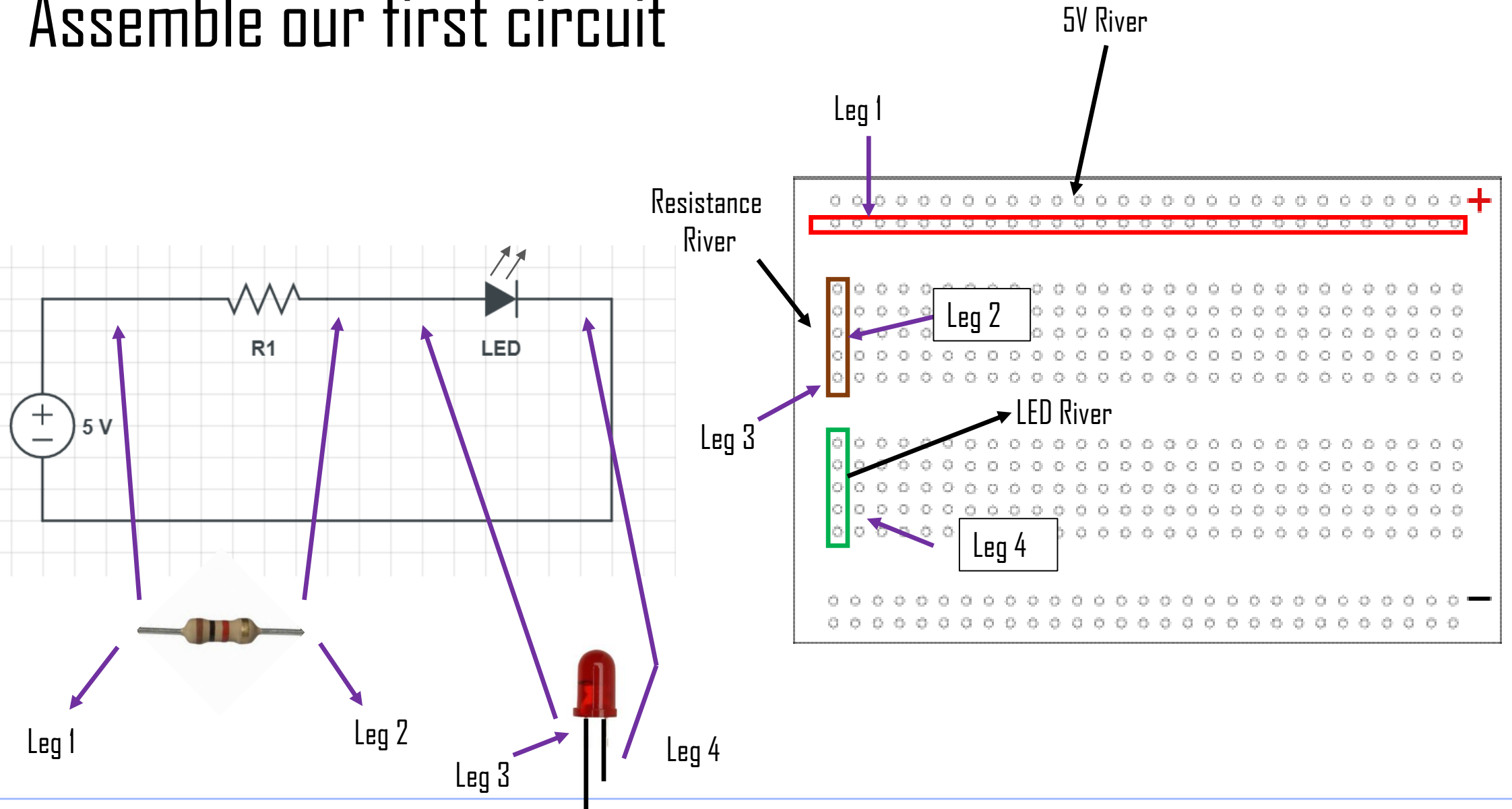
Assemble our first circuit



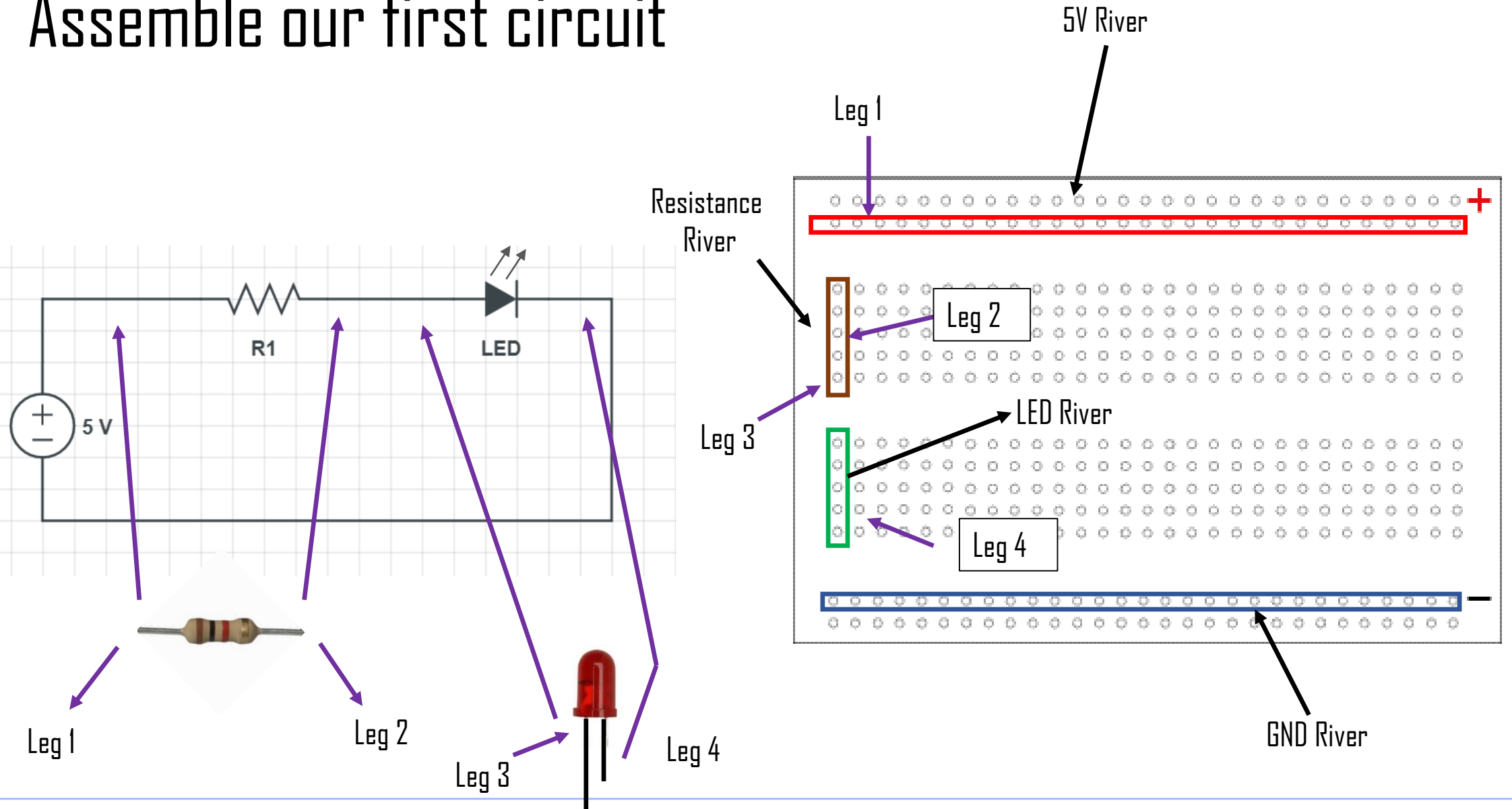
Assemble our first circuit



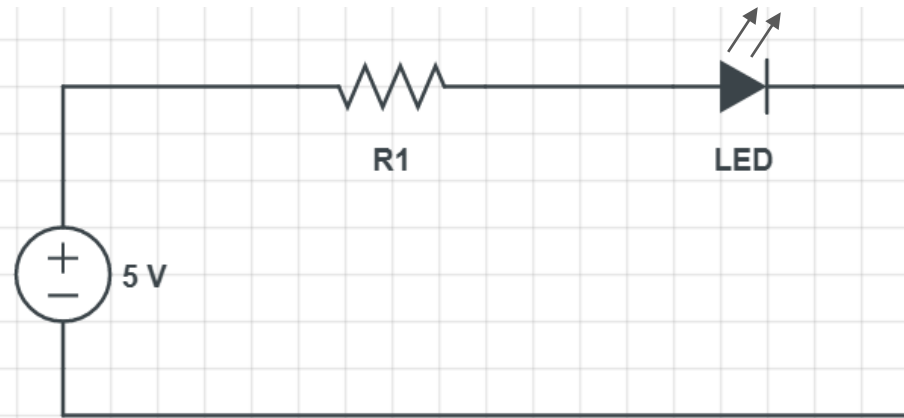
Assemble our first circuit



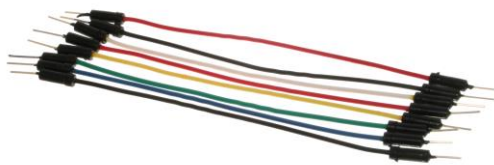
Assemble our first circuit



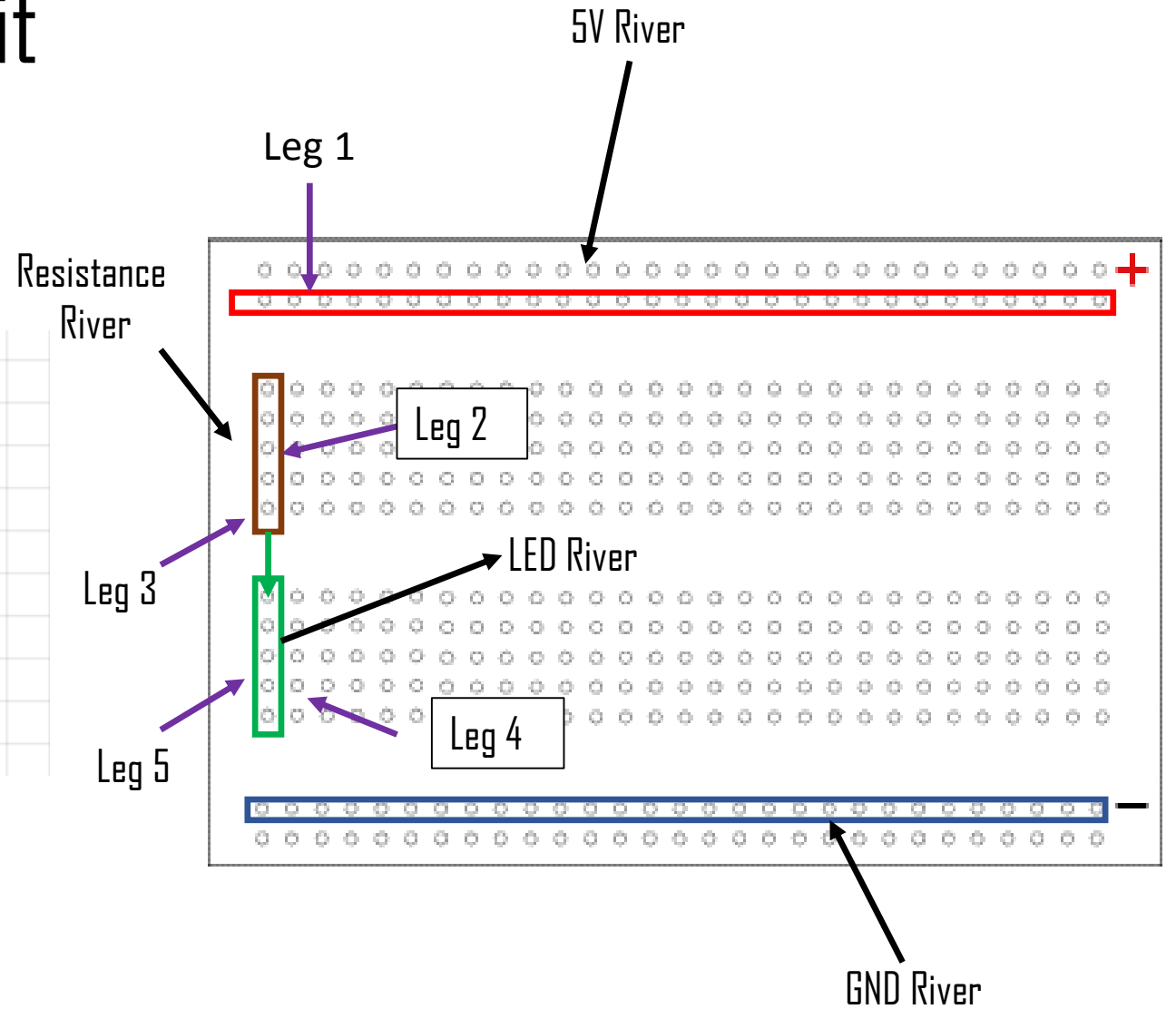
Assemble our first circuit



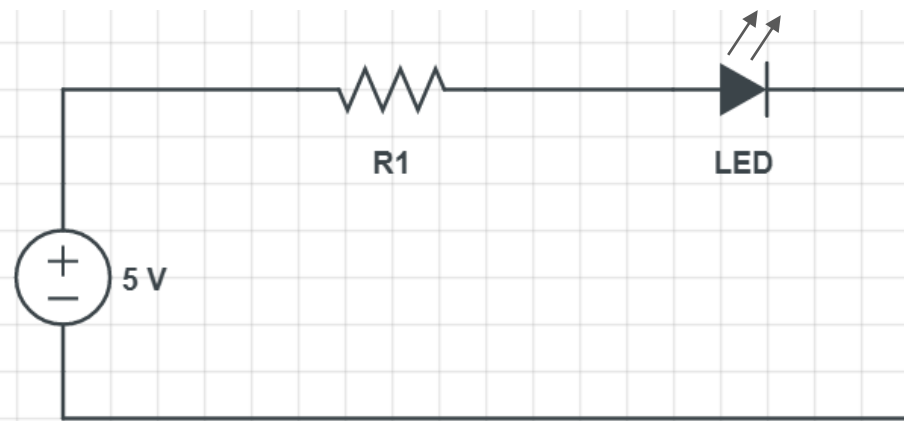
Leg 6



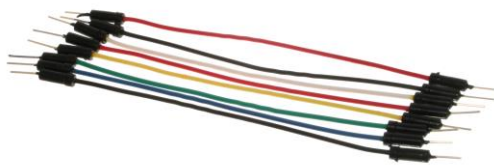
Leg 5



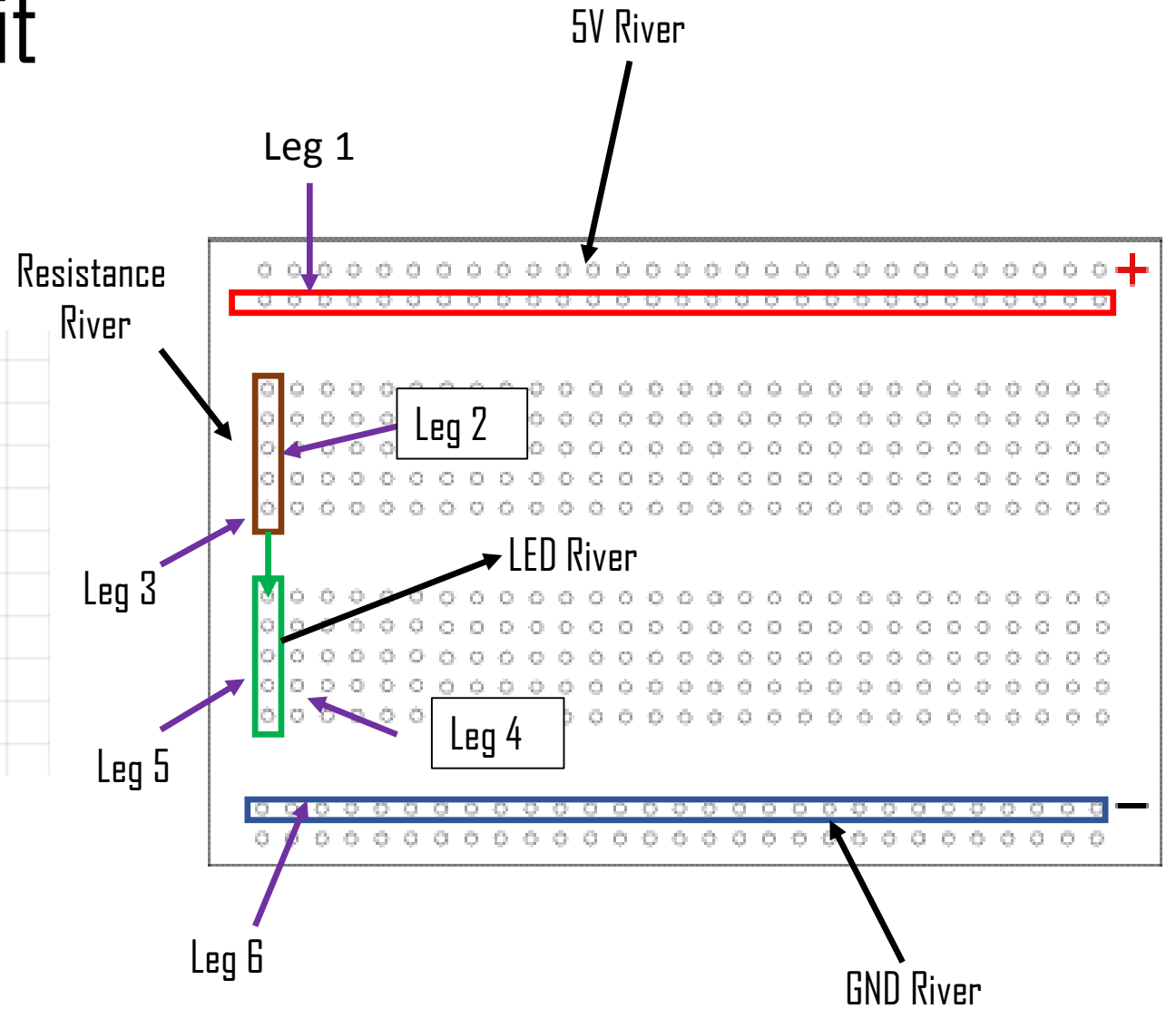
Assemble our first circuit



Leg 6



Leg 5

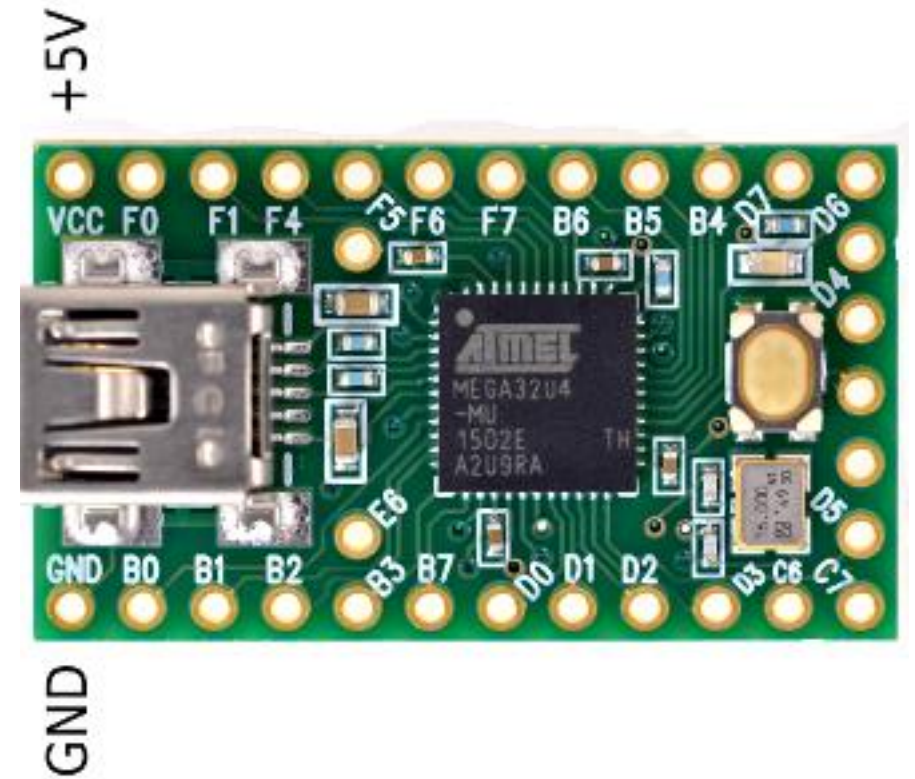


Where is the power supply?

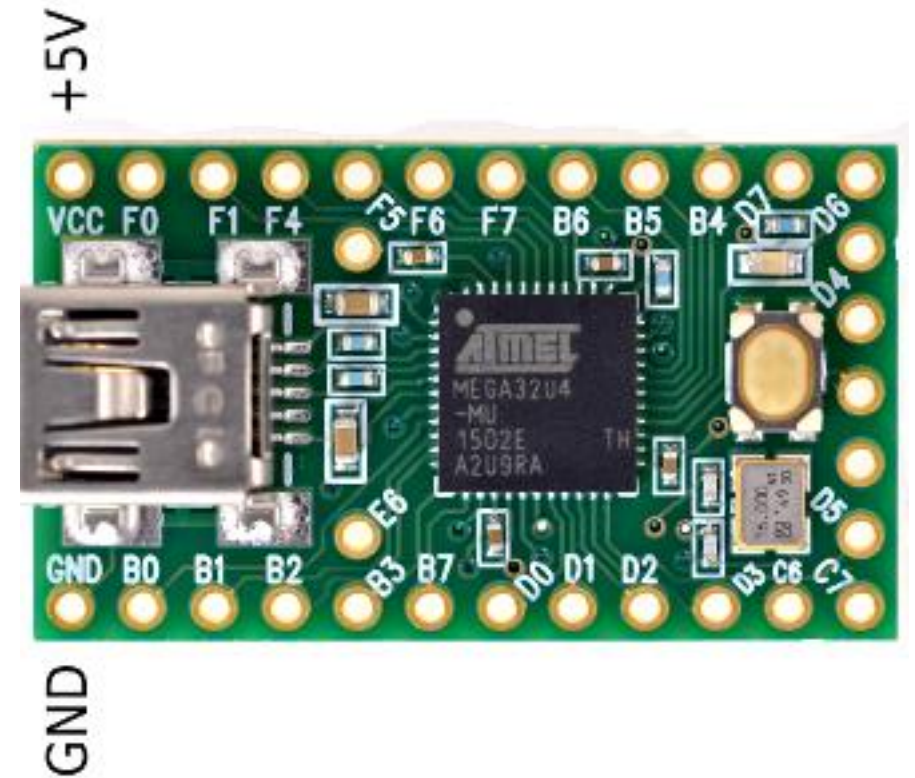
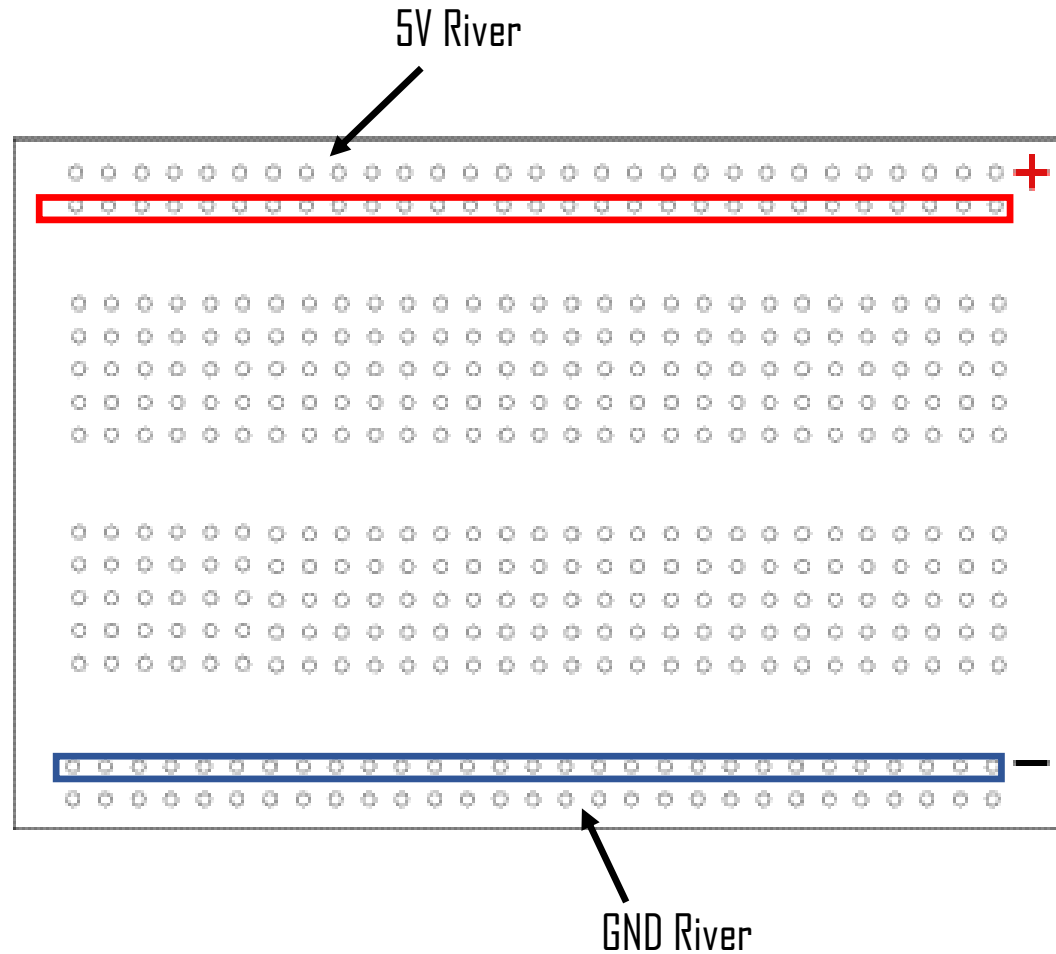
To power up the teensy just connect it to your laptop using USB port

Very important:

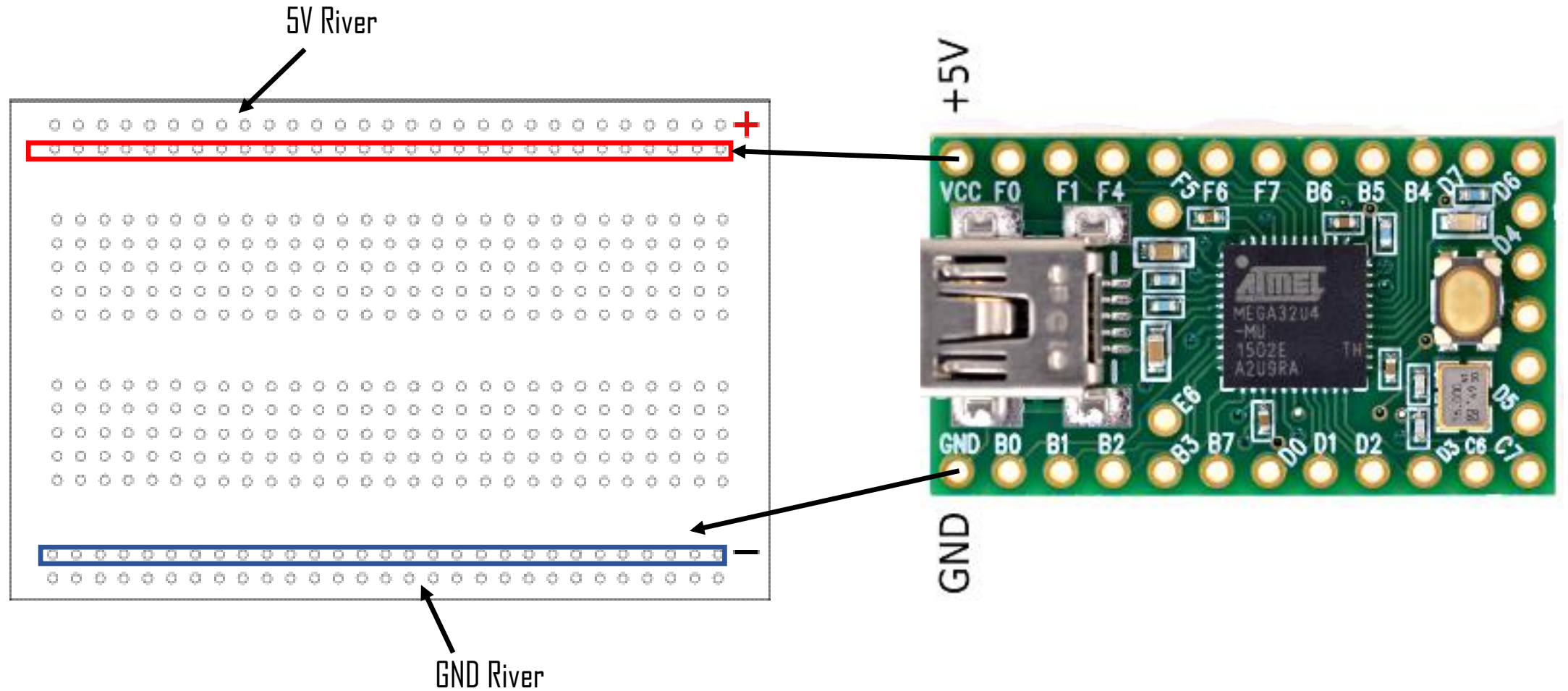
DOUBLE CHECK your circuit before connecting the teensy to your laptop. Invalid connection may damage the microcontroller or your USB port.



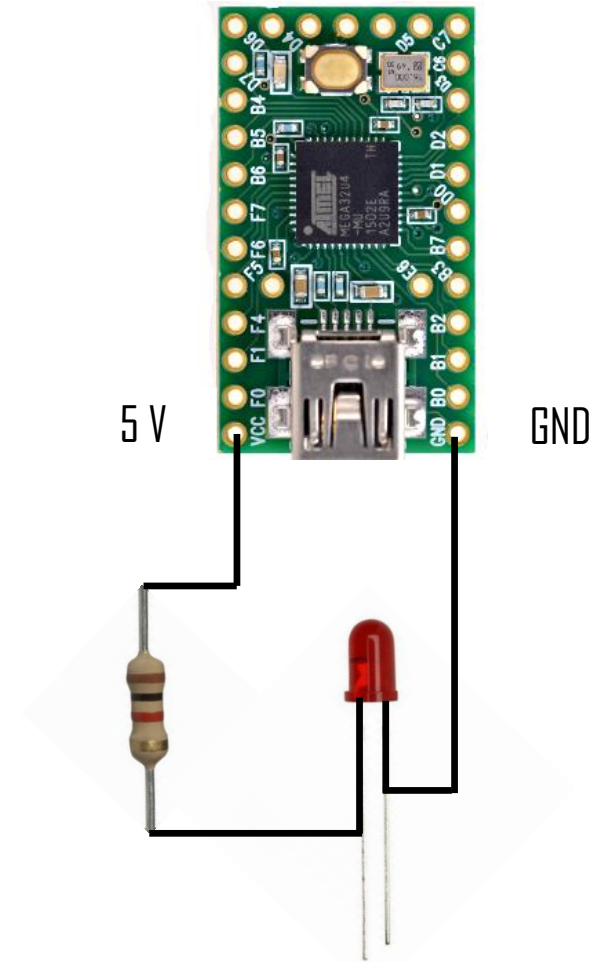
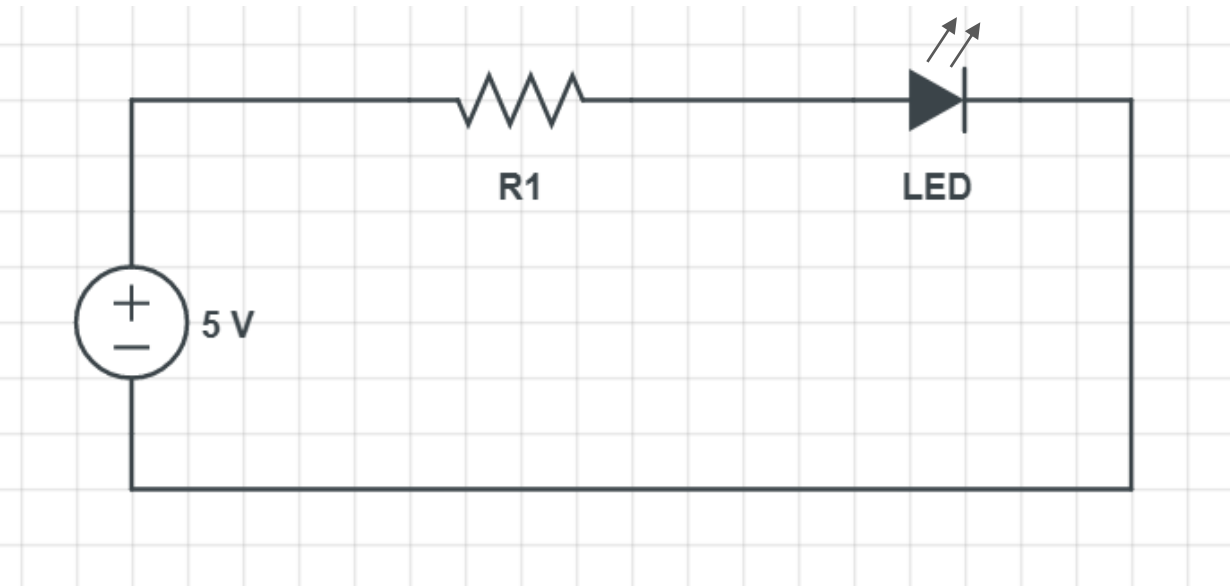
Where is the power supply?



Where is the power supply?

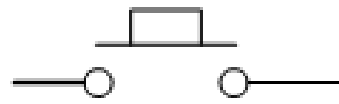
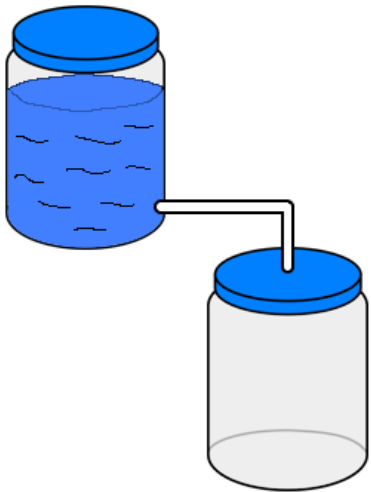
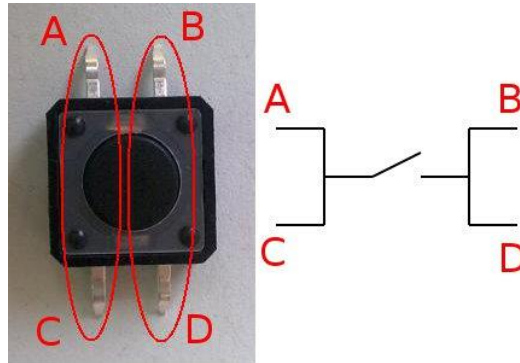


Challenge n°1

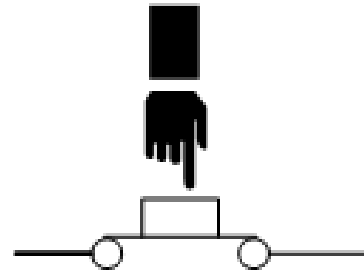


You have 15 min!!

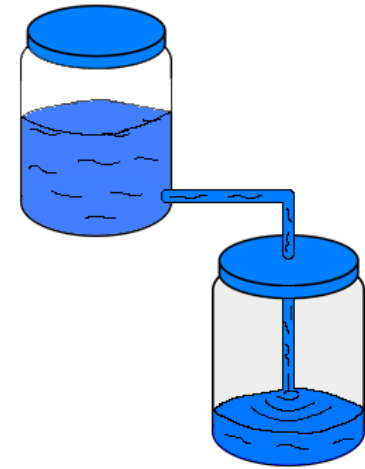
Button



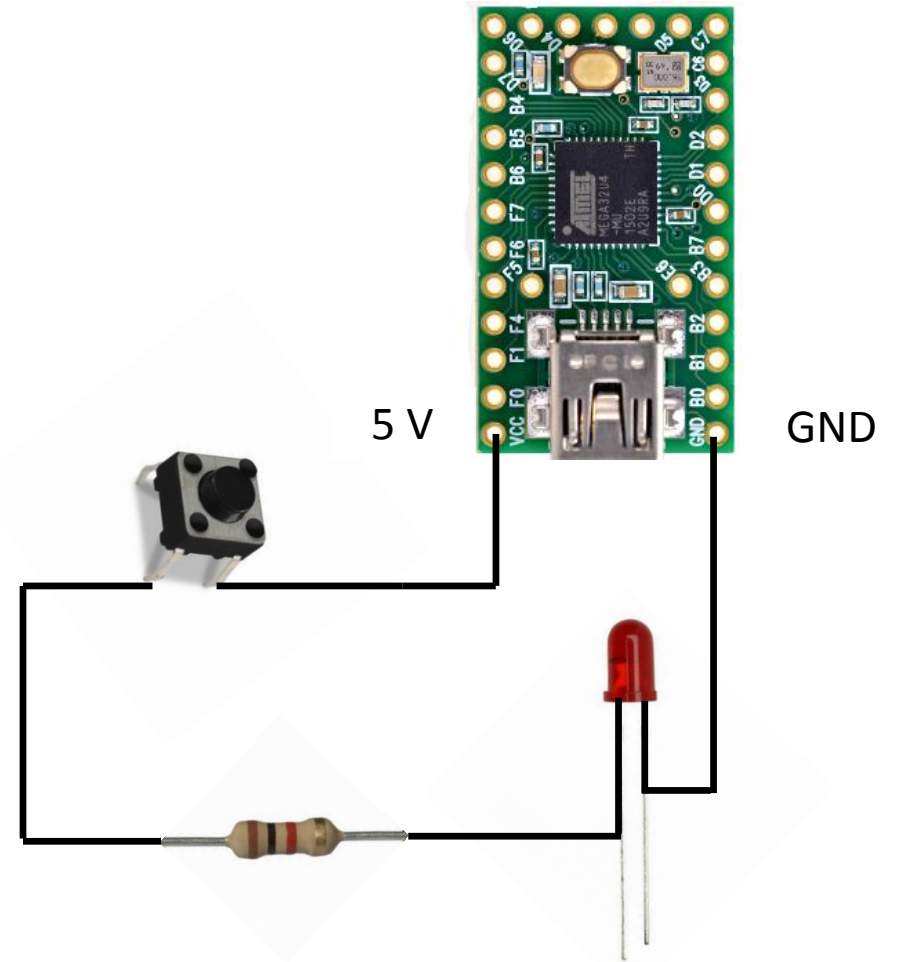
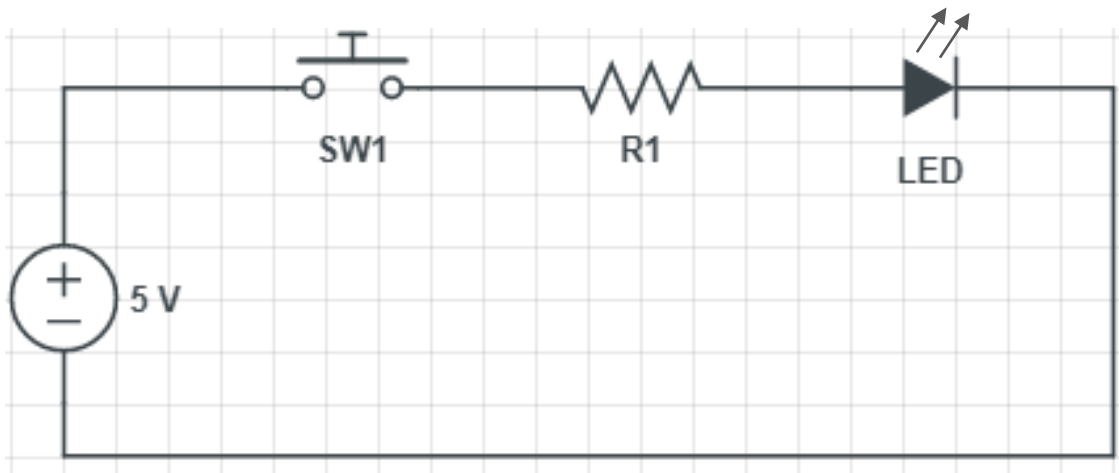
Normally Open
When not pressed



Closed
When pressed



Challenge n°2



You have 5 min!!

What is a microcontroller?

Small computer

Able to do tasks like turn on/off a LED, rotate a motor, read values from different sensors, etc...

It's the brain of the project



What is a microcontroller?

Small computer

Able to do tasks like turn on/off a LED, rotate a motor, read values from different sensors, etc...

It's the brain of the project



What is a microcontroller?

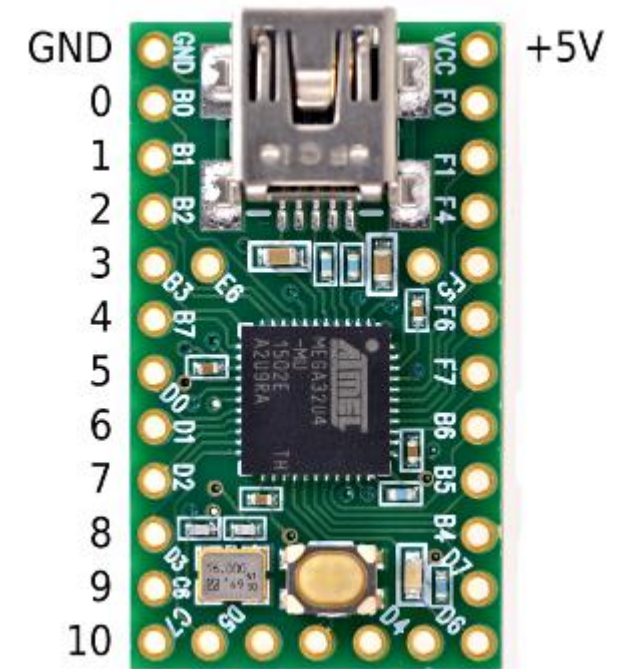
The teensy 2.0 has 11 digital pins, and they can be used to send or read signals

They only have 2 states:

- 1 (High state) 
- 0 (Low state) 

Like the push button, when it's activated the current will flow from the pin to the component that is connected to it

Digital
Pins




Install the software

Go to this link : <https://www.arduino.cc/en/main/software>

HOME BUY SOFTWARE PRODUCTS LEARNING COMMUNITY SUPPORT

SEARCH CART SIGN IN

Download the Arduino IDE



ARDUINO 1.8.5

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any Arduino board. Refer to the [Getting Started](#) page for installation instructions.

Windows Installer
Windows ZIP file for non admin install

Windows app Requires Win 8.1 or 10
Get

Mac OS X 10.7 Lion or newer

Linux 32 bits
Linux 64 bits
Linux ARM

[Release Notes](#)
[Source Code](#)
[Checksums \(sha512\)](#)

Windows

Mac

Linux

HOURLY BUILDS

BETA BUILDS

A aguardar por qstripe.com...

Install the software

ARDUINO

HOME BUY SOFTWARE PRODUCTS LEARNING COMMUNITY SUPPORT

Consider supporting the Arduino Software by contributing to its development. (US tax payers, please note this contribution is not tax deductible). Learn more on how your contribution will be used.

SINCE MARCH 2015, THE ARDUINO IDE HAS BEEN DOWNLOADED **20,519,966** TIMES. (IMPRESSIVE!) NO LONGER JUST FOR ARDUINO AND GENUINO BOARDS, HUNDREDS OF COMPANIES AROUND THE WORLD ARE USING THE IDE TO PROGRAM THEIR DEVICES, INCLUDING COMPATIBLES, CLONES, AND EVEN COUNTERFEITS. HELP ACCELERATE ITS DEVELOPMENT WITH A SMALL CONTRIBUTION! REMEMBER: OPEN SOURCE IS LOVE!

\$3 \$5 \$10 \$25 \$50 OTHER

JUST DOWNLOAD CONTRIBUTE & DOWNLOAD

Download it

Install teensy driver

Go to this link : https://www.pjrc.com/teensy/td_download.html

PJRC Electronic Projects
Components Available Worldwide

[Home](#) [Products](#) [Teensy](#) [Blog](#) [Forum](#)

Shopping Cart • Checkout • Shipping Cost • Download Website

You are here: [Teensy](#) ► [Teensyduino](#) ► [Download+Install](#)

PJRC Store

- [Teensy 3.6](#) \$29.25
- [Teensy 3.5](#) \$24.25
- [Teensy 3.2](#) \$19.80
- [Teensy LC](#) \$11.65
- [Teensy 2.0](#) \$16.00
- [Teensy++ 2.0](#) \$24.00

Teensy

- [Main Page](#)
- [Hardware](#)
- [Getting Started](#)
- [Tutorial](#)
- [How-To Tips](#)
- [Code Library](#)
- [Projects](#)
- [Teensyduino](#)
 - [Main](#)
 - [Download+Install](#)
 - [Basic Usage](#)
 - [Digital I/O](#)
 - [PWM & Tone](#)
 - [Timing](#)
 - [USB Serial](#)
 - [USB Keyboard](#)
 - [USB Mouse](#)
 - [USB Joystick](#)
 - [USB MIDI](#)
 - [USB Flight Sim](#)
 - [Serial](#)

Download Teensyduino, Version 1.40

Teensyduino is a software add-on for the [Teensyduino](#) hardware.

Teensyduino Files:

- [Macintosh OS-X Installer](#)
- [Linux Installer \(X86 32 bit\)](#)
- [Linux Installer \(X86 64 bit\)](#)
- [Linux Installer \(ARM / Raspberry Pi\)](#)
- [Windows XP / 7 / 8 / 10 Installer](#)

Other Files:

- [Linux udev rules](#)
- [Windows Serial Installer](#)

Teensyduino 1.41 is [currently in beta testing](#).

Teensyduino 1.40 supports Arduino versions 1.0.6 and 1.6.5-r5 and 1.8.1 and 1.8.2 and 1.8.3 and 1.8.4 and 1.8.5. Future versions of Teensyduino will drop support for Arduino 1.8.2 and 1.8.3. On Linux, PJRC tests X86 on Ubuntu and ARM on Raspbian. Other distros may work, but are not supported.

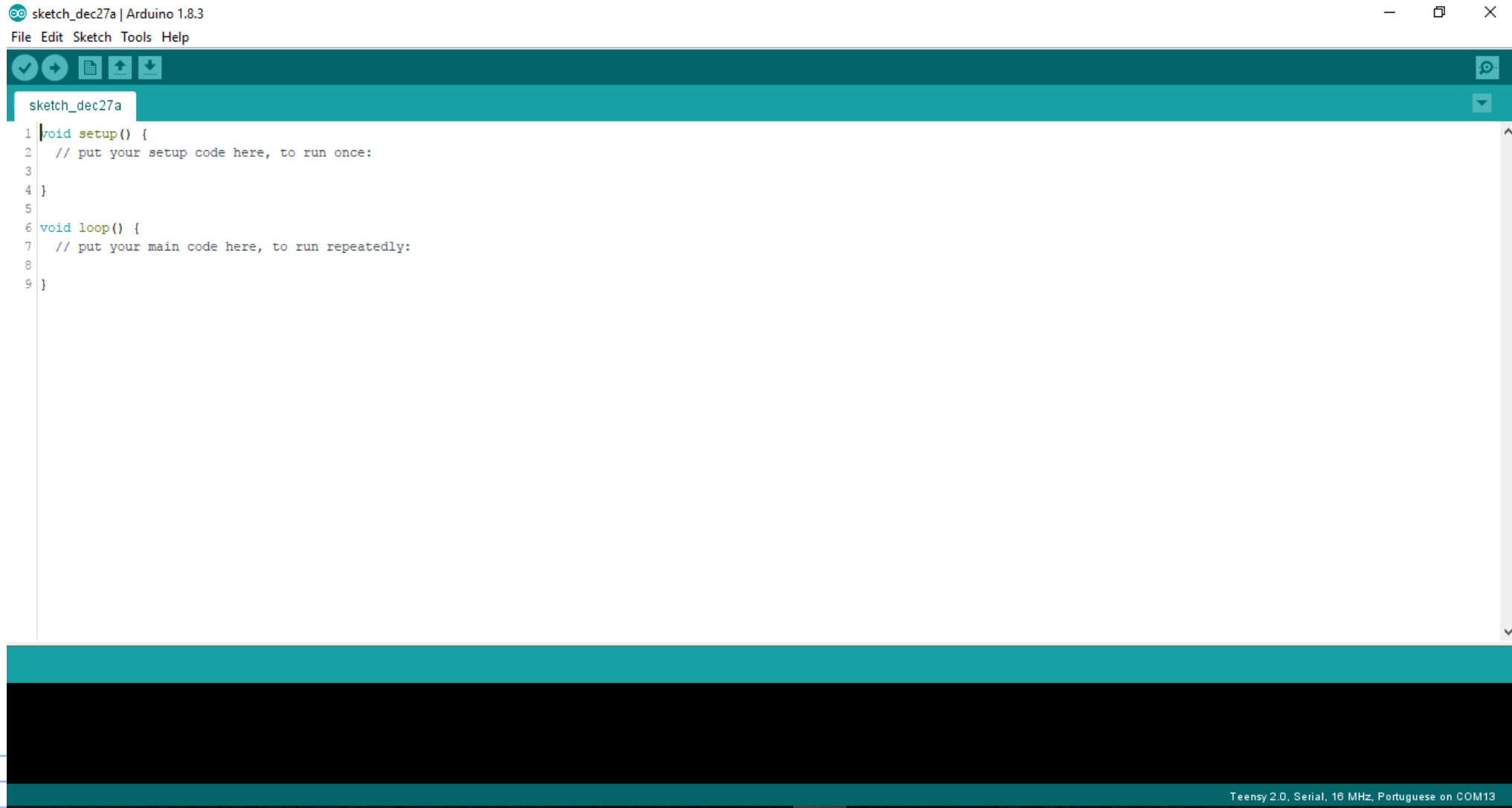
Install Step 1: Download & Extract Arduino

First, you must download the [Arduino Software](#). Remember the location where you extracted the files.

Install Step 2: (Linux only) Install udev Rules

The download files are available on the [Teensyduino Main Page](#).

Arduino IDE



The image shows a screenshot of the Arduino IDE interface. The window title is "sketch_dec27a | Arduino 1.8.3". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". The toolbar contains icons for a checkmark, a right arrow, a grid, an upload arrow, a download arrow, and a gear. The sketch editor shows the following code:

```
sketch_dec27a
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 }
```

At the bottom of the IDE, there is a status bar with the text "Teensy 2.0, Serial, 16 MHz, Portuguese on COM13".

Arduino IDE

The screenshot shows the Arduino IDE interface with the **Tools** menu open. The menu items are:

- Auto Format (Ctrl+T)
- Archive Sketch
- Fix Encoding & Reload
- Serial Monitor (Ctrl+Shift+M)
- Serial Plotter (Ctrl+Shift+L)
- WiFi101 Firmware Updater
- Board: "Teensy 2.0"** (highlighted)
- USB Type: "Serial"
- CPU Speed: "16 MHz"
- Keyboard Layout: "Portuguese"
- Port
- Get Board Info
- Programmer: "AVRISP mkII"
- Burn Bootloader

The **Boards Manager** is also open, showing a list of boards. The **Teensy 2.0** board is selected and highlighted with a red arrow and the number 3.

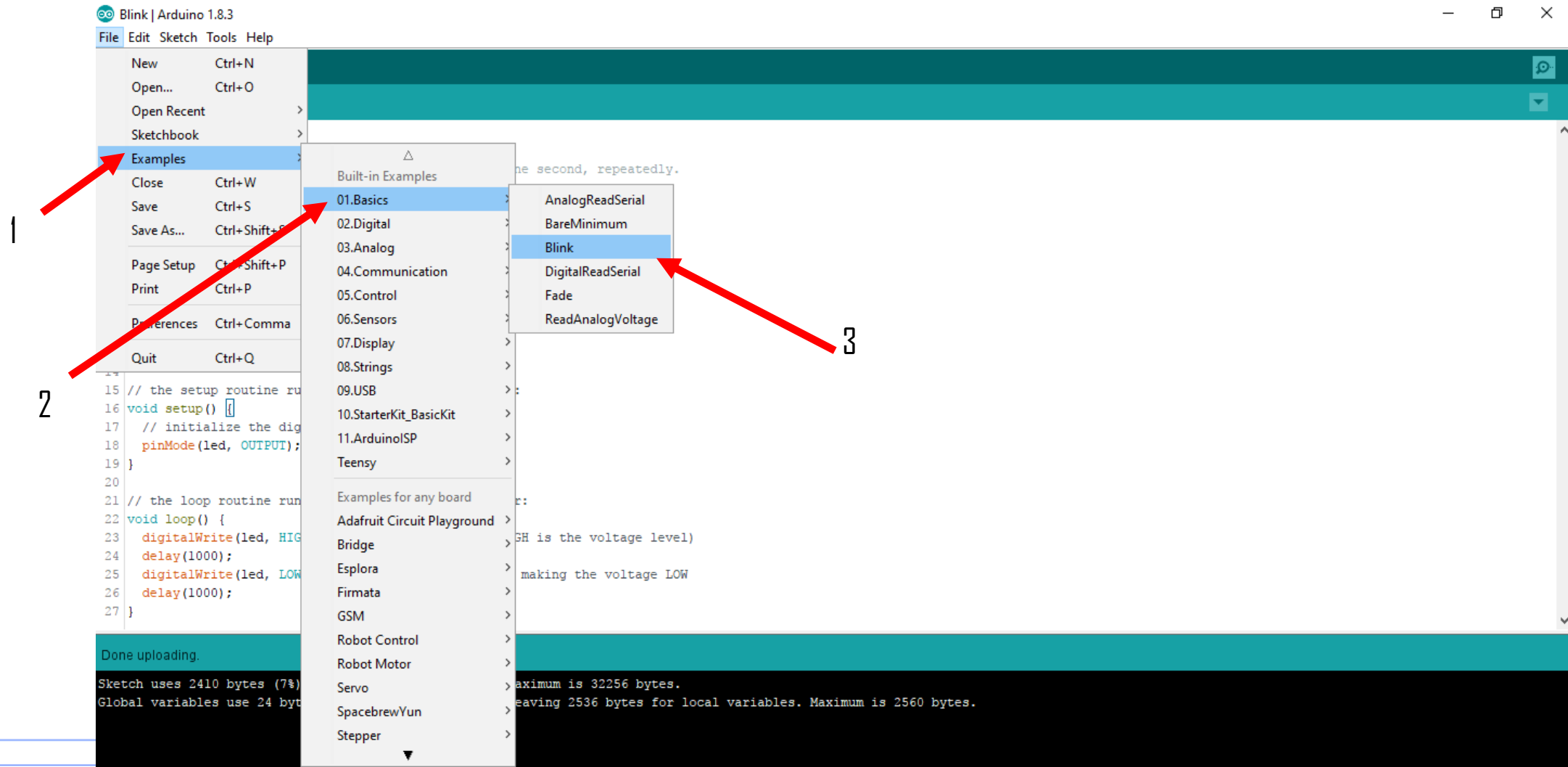
Red arrows and numbers indicate the steps:

- 1: Arrow pointing to the **Tools** menu.
- 2: Arrow pointing to the **Board: "Teensy 2.0"** option in the Tools menu.
- 3: Arrow pointing to the **Teensy 2.0** option in the Boards Manager.

At the bottom of the IDE, there is a notification: "Updates available for some of your [boards](#) and [libraries](#)".

The status bar at the bottom right shows: "Teensy 2.0, Serial, 16 MHz, Portuguese on COM13".

Arduino IDE



Arduino IDE



The screenshot shows the Arduino IDE interface with a sketch named "Blink". The code is as follows:

```
1 /*
2  Blink
3  Turns on an LED on for one second, then off for one second, repeatedly.
4
5  This example code is in the public domain.
6  */
7
8  // Pin 13 has an LED connected on most Arduino boards.
9  // Pin 11 has the LED on Teensy 2.0
10 // Pin 6 has the LED on Teensy++ 2.0
11 // Pin 13 has the LED on Teensy 3.0
12 // give it a name:
13 int led = 13;
14
15 // the setup routine runs once when you press reset:
16 void setup() {
17   // initialize the digital pin as an output.
18   pinMode(led, OUTPUT);
19 }
20
21 // the loop routine runs over and over again forever:
22 void loop() {
23   digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
24   delay(1000);             // wait for a second
25   digitalWrite(led, LOW);  // turn the LED off by making the voltage LOW
26   delay(1000);             // wait for a second
27 }
```

A red arrow points from the text "Change 13 to 11" to the value "13" in the line `int led = 13;`.

13

Teensy 2.0, Serial, 16 MHz, Portuguese on COM5

Arduino IDE



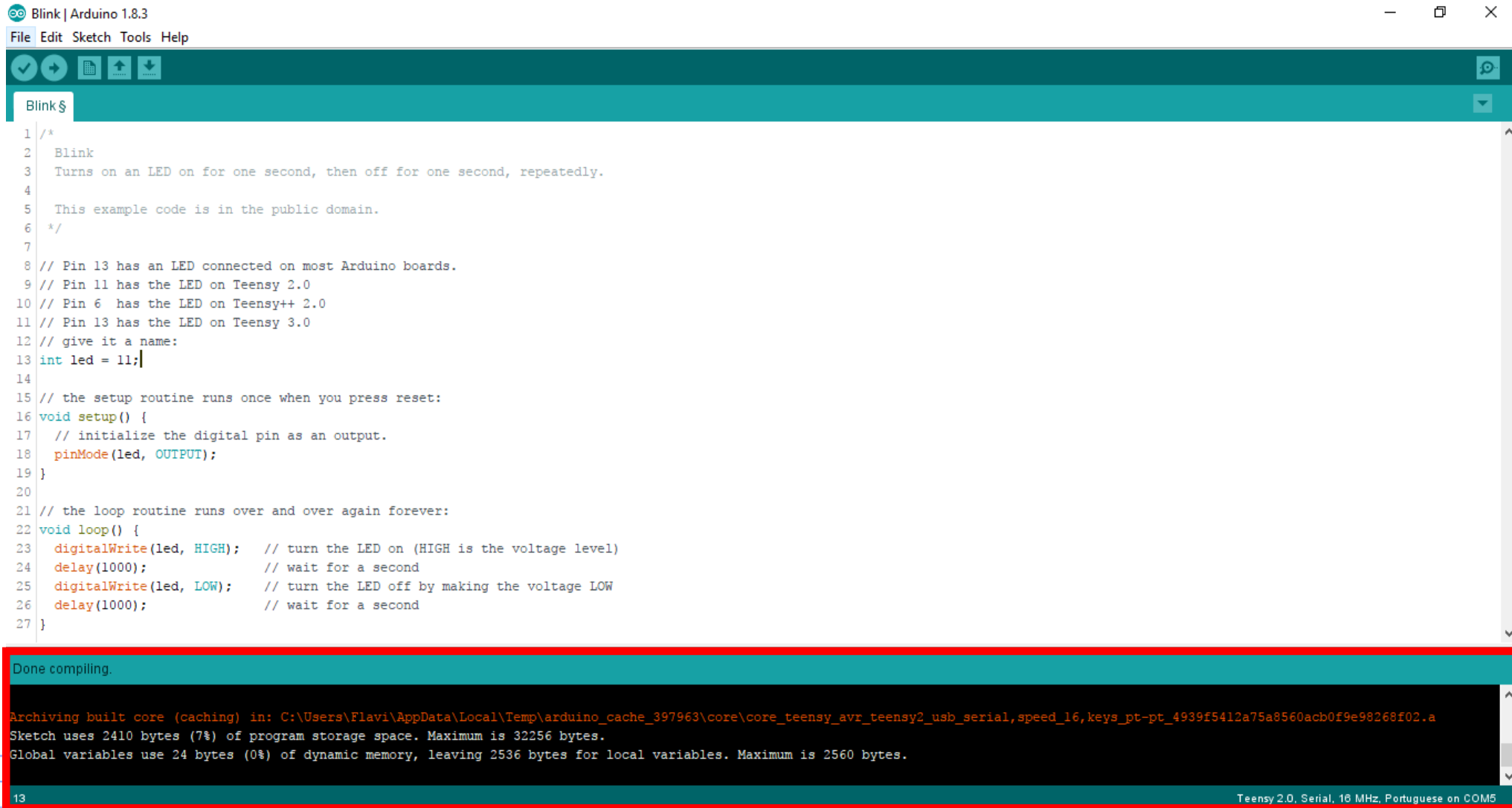
Click on Verify

```
Blink | Arduino 1.8.3
File Edit Sketch Tools Help
Blink
1 /*
2  Blink
3  Turns on an LED on for one second, then off for one second, repeatedly.
4
5  This example code is in the public domain.
6  */
7
8  // Pin 13 has an LED connected on most Arduino boards.
9  // Pin 11 has the LED on Teensy 2.0
10 // Pin 6 has the LED on Teensy++ 2.0
11 // Pin 13 has the LED on Teensy 3.0
12 // give it a name:
13 int led = 13;
14
15 // the setup routine runs once when you press reset:
16 void setup() {
17   // initialize the digital pin as an output.
18   pinMode(led, OUTPUT);
19 }
20
21 // the loop routine runs over and over again forever:
22 void loop() {
23   digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
24   delay(1000); // wait for a second
25   digitalWrite(led, LOW); // turn the LED off by making the voltage LOW
26   delay(1000); // wait for a second
27 }
```

13

Teensy 2.0, Serial, 16 MHz, Portuguese on COM5

Arduino IDE



The screenshot displays the Arduino IDE interface. The main window shows a sketch named "Blink" with the following code:


```
1 /*
2  Blink
3  Turns on an LED on for one second, then off for one second, repeatedly.
4
5  This example code is in the public domain.
6  */
7
8  // Pin 13 has an LED connected on most Arduino boards.
9  // Pin 11 has the LED on Teensy 2.0
10 // Pin 6 has the LED on Teensy++ 2.0
11 // Pin 13 has the LED on Teensy 3.0
12 // give it a name:
13 int led = 11;
14
15 // the setup routine runs once when you press reset:
16 void setup() {
17   // initialize the digital pin as an output.
18   pinMode(led, OUTPUT);
19 }
20
21 // the loop routine runs over and over again forever:
22 void loop() {
23   digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
24   delay(1000);             // wait for a second
25   digitalWrite(led, LOW);  // turn the LED off by making the voltage LOW
26   delay(1000);             // wait for a second
27 }
```

Below the code editor, the "Serial Monitor" window is open, showing the compilation output:

```
Done compiling.
Archiving built core (caching) in: C:\Users\Flavi\AppData\Local\Temp\arduino_cache_397963\core\core_teensy_avr_teensy2_usb_serial,speed_16,keys_pt-pt_4939f5412a75a8560acb0f9e98268f02.a
Sketch uses 2410 bytes (7%) of program storage space. Maximum is 32256 bytes.
Global variables use 24 bytes (0%) of dynamic memory, leaving 2536 bytes for local variables. Maximum is 2560 bytes.
```

The status bar at the bottom of the IDE indicates: "Teensy 2.0, Serial, 16 MHz, Portuguese on COM5".

Arduino IDE



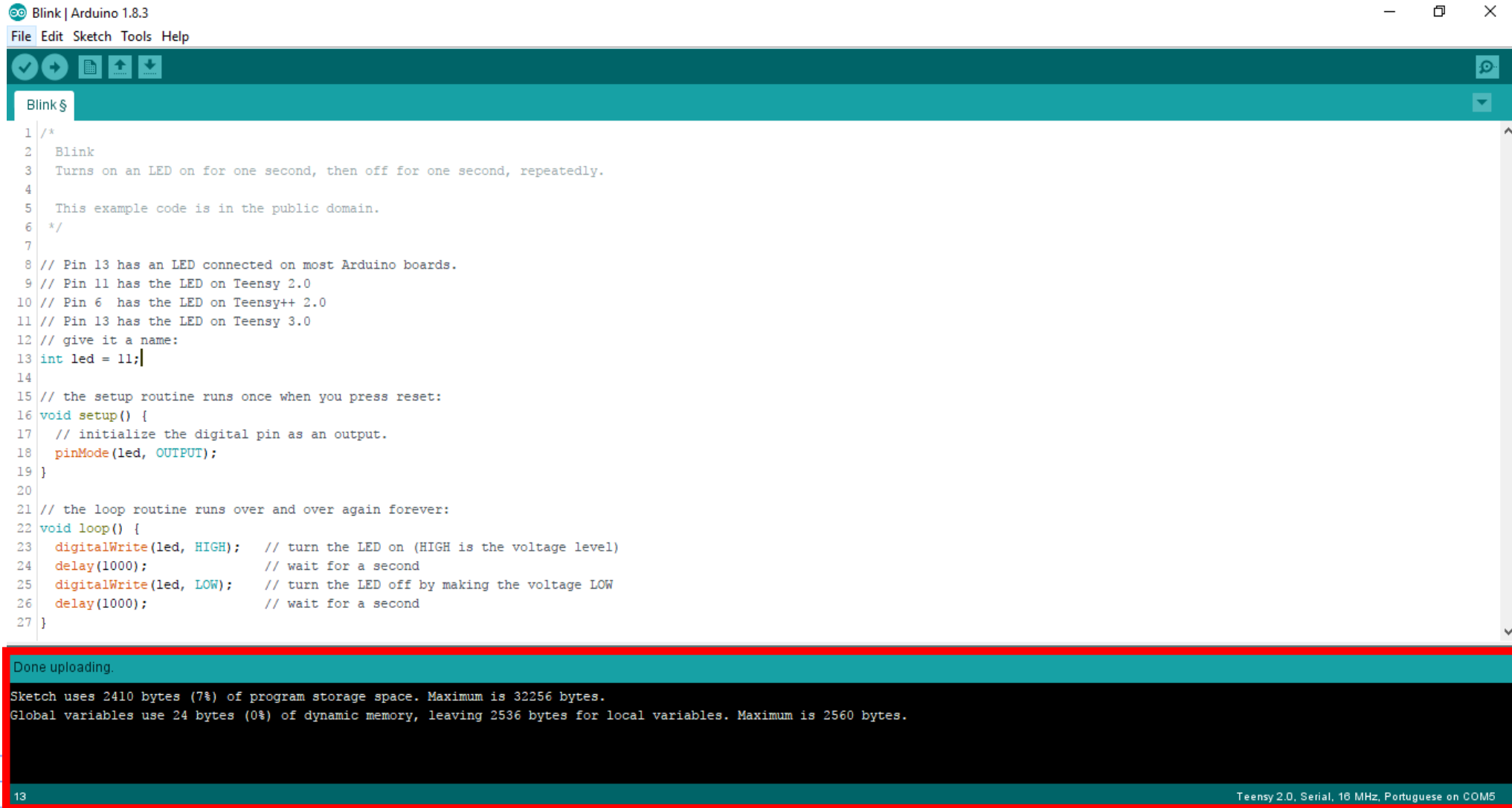
The screenshot shows the Arduino IDE interface. The title bar reads "Blink | Arduino 1.8.3". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". The toolbar contains several icons, with the upload icon (a right-pointing arrow) highlighted by a red arrow. The main text area contains the following code:

```
1 /*
2  Blink
3  Turns on an LED on for one second, then off for one second, repeatedly.
4
5  This example code is in the public domain.
6  */
7
8  // Pin 13 has an LED connected on most Arduino boards.
9  // Pin 11 has the LED on Teensy 2.0
10 // Pin 6 has the LED on Teensy++ 2.0
11 // Pin 13 has the LED on Teensy 3.0
12 // give it a name:
13 int led = 13;
14
15 // the setup routine runs once when you press reset:
16 void setup() {
17   // initialize the digital pin as an output.
18   pinMode(led, OUTPUT);
19 }
20
21 // the loop routine runs over and over again forever:
22 void loop() {
23   digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
24   delay(1000);             // wait for a second
25   digitalWrite(led, LOW);  // turn the LED off by making the voltage LOW
26   delay(1000);             // wait for a second
27 }
```

At the bottom of the IDE, a status bar displays "13" on the left and "Teensy 2.0, Serial, 16 MHz, Portuguese on COM5" on the right.

Click on
upload

Arduino IDE



The image shows the Arduino IDE interface. The main window displays a sketch named "Blink" with the following code:

```
1 /*
2  Blink
3  Turns on an LED on for one second, then off for one second, repeatedly.
4
5  This example code is in the public domain.
6  */
7
8  // Pin 13 has an LED connected on most Arduino boards.
9  // Pin 11 has the LED on Teensy 2.0
10 // Pin 6 has the LED on Teensy++ 2.0
11 // Pin 13 has the LED on Teensy 3.0
12 // give it a name:
13 int led = 11;
14
15 // the setup routine runs once when you press reset:
16 void setup() {
17   // initialize the digital pin as an output.
18   pinMode(led, OUTPUT);
19 }
20
21 // the loop routine runs over and over again forever:
22 void loop() {
23   digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
24   delay(1000);             // wait for a second
25   digitalWrite(led, LOW);  // turn the LED off by making the voltage LOW
26   delay(1000);             // wait for a second
27 }
```

Below the code editor, a status bar indicates the upload process is complete:

```
Done uploading.
Sketch uses 2410 bytes (7%) of program storage space. Maximum is 32256 bytes.
Global variables use 24 bytes (0%) of dynamic memory, leaving 2536 bytes for local variables. Maximum is 2560 bytes.
```

The status bar at the bottom of the IDE shows the board and port configuration: "Teensy 2.0, Serial, 16 MHz, Portuguese on COM5".

Challenge n°3



MAGIC

Arduino Code

The code is always written and read from the top to bottom, and left to right

A command as always to end with a ";" except when it's a function or a condition

```
digitalWrite(led, HIGH);  
delay(1000);
```

It's good practice to comment our code, to do that just type "//" and the rest of the text after is just comments

Arduino Code

Variables are used to store information to be used in a computer program

They also provide a way of labeling data with a descriptive name

Variables are containers that hold information

Arduino Code

Types of variables



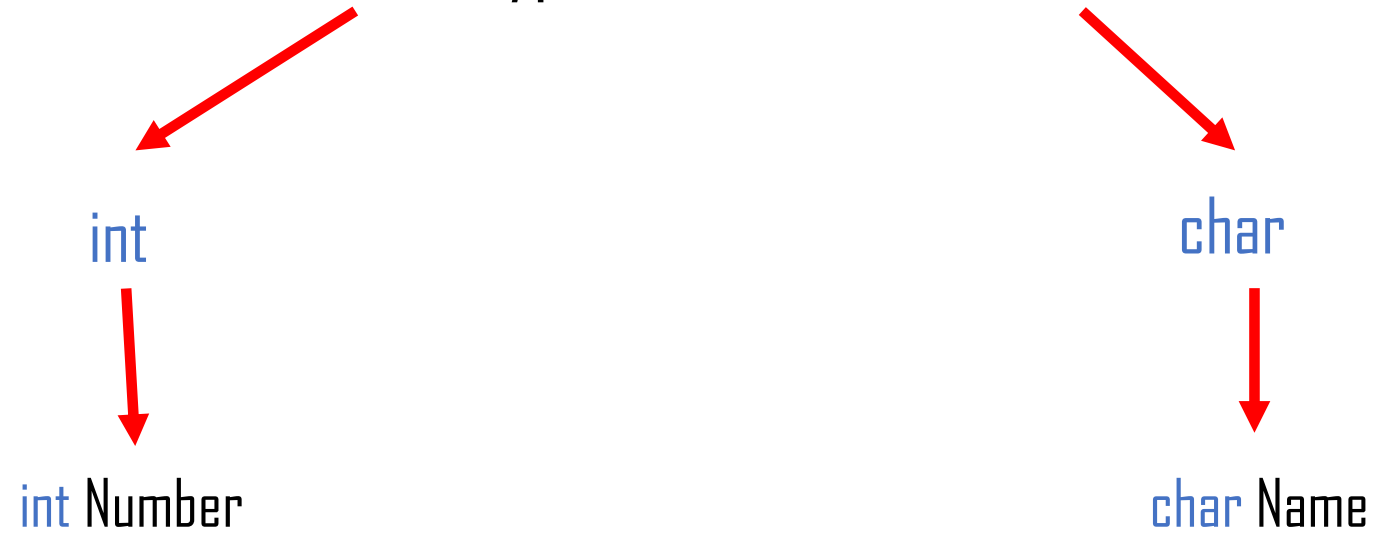
int



char

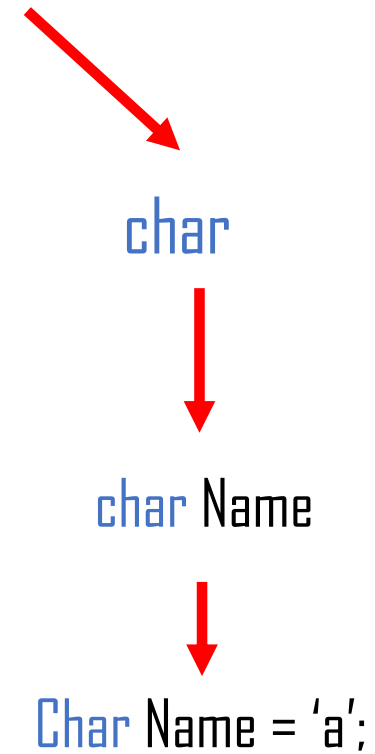
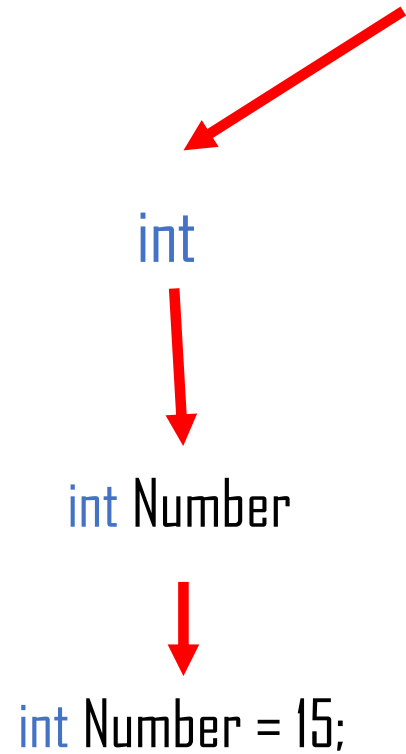
Arduino Code

Types of variables



Arduino Code

Types of variables



Arduino Code

Functions allows a programmer to create modular pieces of code

```
void name_of_the_function  
{  
  
}
```

Arduino Code

Functions allows a programmer to create modular pieces of code

```
void name_of_the_function
```

Name of our function

```
{
```

Starts with open curly bracket

```
}
```

Ends with close curly bracket

Arduino Code

Functions allows a programmer to create modular pieces of code

```
void name_of_the_function
```

Name of our function

```
{  
  
}  
  

```

Starts with open curly bracket

Ends with close curly bracket

Everything between the curly brackets is what happens in our function

Arduino Code

Functions allows a programmer to create modular pieces of code

```
void setup()  
{  
  
}  
}
```



- Used to state what we will be using in our project
- Number of LED, sensors, etc...

Arduino Code

Functions allows a programmer to create modular pieces of code

```
void setup()
```

```
{
```

```
}
```

```
void loop()
```

```
{
```

```
}
```



➤ Used to state what we will be using in our project

➤ Number of LED, sensors, etc...



➤ Where the action begins

➤ Happens in a infinit loop

Arduino Code

Arduino has built-in functions that can be use for different situations

Arduino Code

Arduino has built-in functions that can be use for different situations

When we want to use a LED:

```
pinMode ( 13 , OUTPUT ) ;
```

Function to configures the
specified pin to behave either
as an input or an output

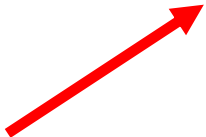


Arduino Code

Arduino has built-in functions that can be use for different situations

When we want to use a LED:

```
pinMode ( 13 , OUTPUT ) ;
```



Function to configures the specified pin to behave either as an input or an output



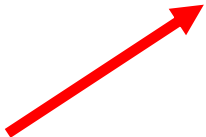
Number of the pin

Arduino Code

Arduino has built-in functions that can be use for different situations

When we want to use a LED:


```
pinMode ( 13 , OUTPUT ) ;
```



Function to configures the specified pin to behave either as an input or an output



Number of the pin



Direction of the signal

Arduino Code

Arduino has built-in functions that can be use for different situations

When we want to use a LED:

```
digitalWrite ( 13 , HIGH ) ;
```


Function to send the signal



Number of the pin



Value of the signal



Arduino Code

```
int led = 11 ;

// the setup routine runs once when you press reset:
void setup()
{
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop()
{
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);             // wait for a second
  digitalWrite(led, LOW); // turn the LED off by making the voltage LOW
  delay(1000);             // wait for a second
}
```

Arduino Code

```
// the setup routine runs once when you press reset:
void setup()
{
  int led = 11;
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop()
{
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);             // wait for a second
  digitalWrite(led, LOW); // turn the LED off by making the voltage LOW
  delay(1000);             // wait for a second
}
```

Arduino Code

'led' was not declared in this scope

```
Blink: In function 'void loop()':
```

```
Blink:13: error: 'led' was not declared in this scope
```

```
    digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
```

Arduino Code

Global variable

When a variable is declared outside of the functions it can be used by any of them

```
int led = 13;

// the setup routine runs once
void setup()
{
  pinMode(led, OUTPUT);
}
```

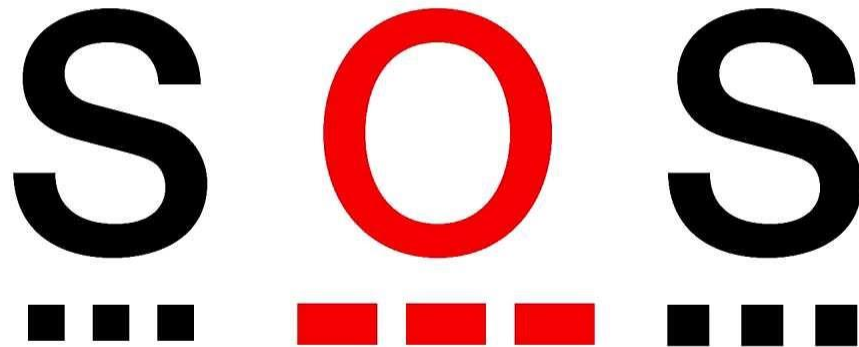
Local variable

When a variable is defined in a function, it can only be used inside the function

```
void setup()
{
  int led = 13;
  pinMode(led, OUTPUT);
}
```

Challenge nº 4

Send a S.O.S. signal from arduino with the built-in LED



You have 15 minutes!!

Using teensy to read a signal

Now you learn how to use the teensy to read a signal

To do that, we will be using a push botton to send the signal to the teensy

Depending on the state off the pin that we will read, the LED should be on or off

Using teensy

Use the PIN 0 of teensy to read a signal

```
pinMode ( 0 , INPUT );
```

Function to configures the specified pin to behave either as an input or an output

Number of the pin

Direction of the signal

```
digitalRead (0)
```

Function that returns the state of the pin

PIN to read

Using teensy

If condition is used to compare values and take action according to the result

```
if (digitalRead(0) == HIGH)
{
    digitalWrite(led, HIGH);
} else
{
    digitalWrite(led, LOW);
}
```

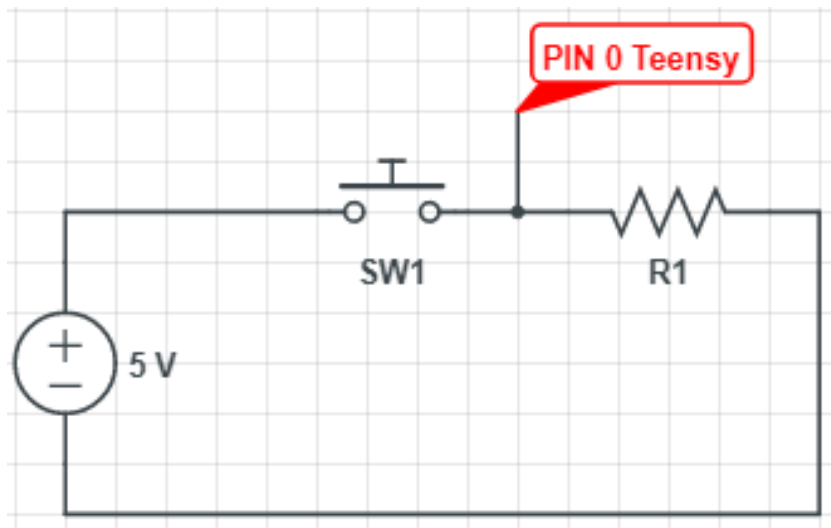
Our condition

True

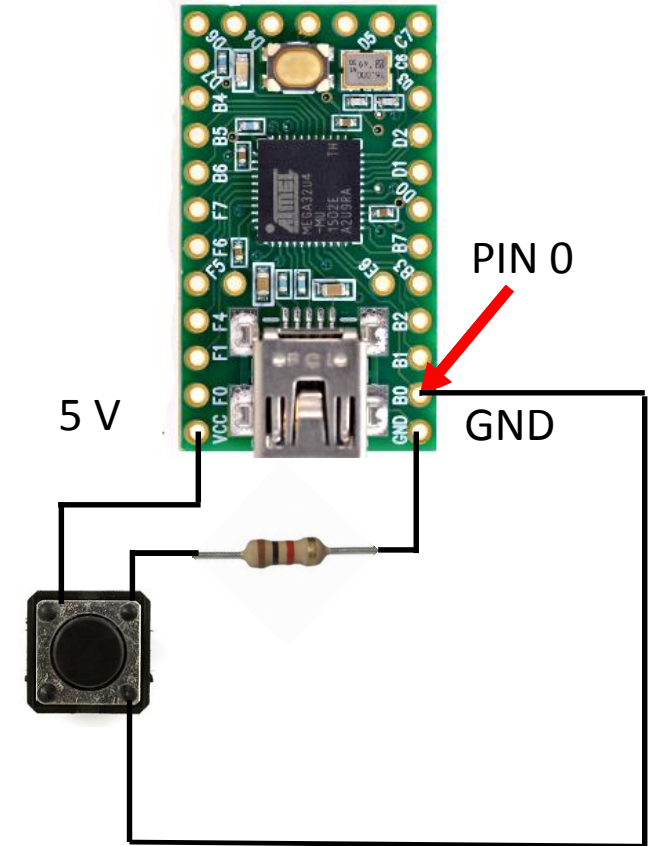
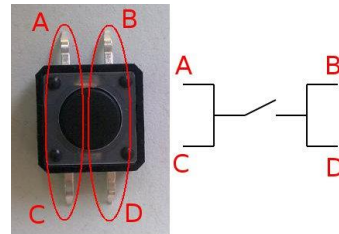
False

= to assign values
== to compare values

Challenge n° 5



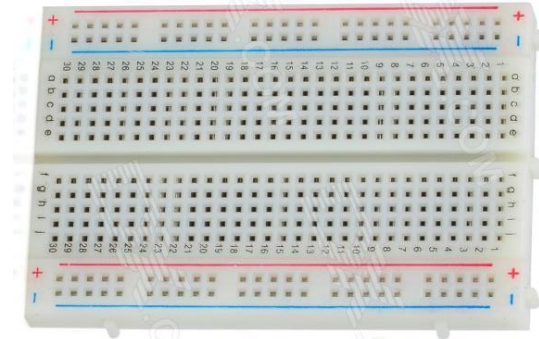
Send the S.O.S. only when you press the push button



You have 10 minutes!!

The Big Challenge

Create a game of your own using:



You have 90 minutes!!