Learning Unit	
Subject	Programação
Title	Aprender C++ com o Robô Botn´Roll
Authors	Nuno Barbosa
School	FORAVE – ASSOCIAÇÃO PARA A EDUCAÇÃO TECNOLÓGICA DO VALE DO AVE
Description of the unit	The aim of this unit is to learn the basics of programming in C++ using robots.
Contents	Programming in C++ language: - Arduino development board - algorithm - using the Arduino IDE - infrared sensors - line follower sensor
Learning Outcomes / Skills	Students should be able to: ●Develop critical thinking and the ability to work in groups; - Develop problem-solving skills; - Develop persistence, autonomy and a willingness to deal with situations involving programming in their school career and life in society; - Develop an interest in programming and appreciate its role in the development of other sciences and areas of human and social activity.
Target students/class	Secondary school (15 – 17 anos)
Prerequisites	Students should be able to: ● Make flowcharts in order to structure the resolution of a problem; ● Create pseudocode in order to structure the resolution of a problem; ● Use compilers/interpreters; ● Identify C++ commands; ● Know the C++ commands needed to control a Botn'Roll robot. ● Identify the libraries needed to control a Botn'Roll robot.
Time expected	4 hours
Interdisciplinary links	ICT

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Learning Unit	
Methodology	Explanation of contents, solving exercises and problems, resolution of worksheets and pair work.
Human Resources (internal and/or external)	Technical Studies Teacher
Resources	● Worksheets; ● Botn'Roll robots; ● Laptops.
Lesson Plan	**<u>1st Lesson:</u>** **Summary**: Algorithms. Flowcharts. Pseudocode. The teacher introduces the theoretical concepts related to flowcharts and pseudocode. After introducing the concepts and analysing the solved example, the teacher proposes solving exercise 1 of the worksheet in pairs. Clarification of doubts. **<u>2nd Lesson</u>:** **Summary**: How does a robot work? The teacher introduces the theoretical concepts related to the hardware and software needed to control a robot. The students will have to upload an example program for the Botn'Roll robot and analyse its behaviour. **<u>3rd Lesson</u>:** **Summary**: Arduino IDE. Btnroll library The teacher explains the concepts needed to understand the Arduino IDE and the Btnroll library and carries out a short example exercise. After introducing the concepts, the teacher suggests solving exercises 2, 3 and 4 of the worksheet in pairs. Clarification of doubts.

Co-funded by the Erasmus+ Programme of the European Union

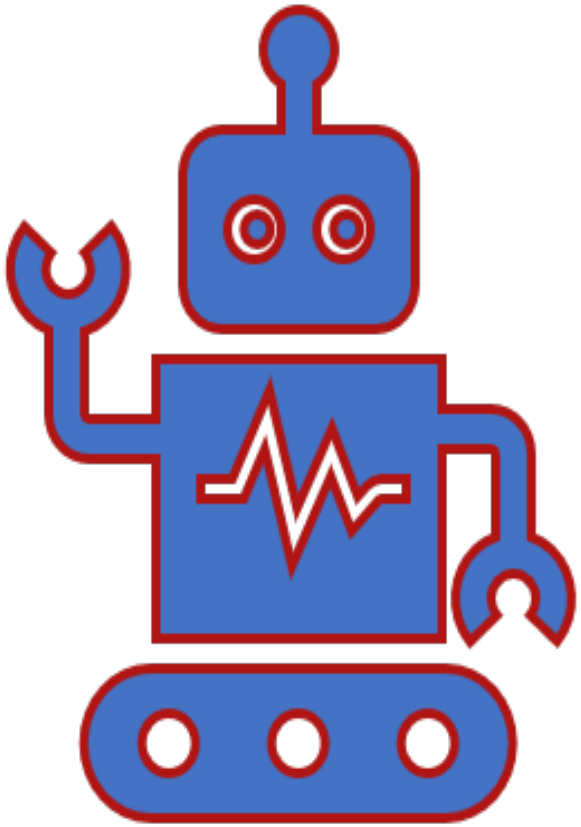
This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Learning Unit	
	4th Lesson: **Summary**: Btnroll Library. The teacher explains the concepts needed to understand the Btn´roll library. After introducing the concepts, the teacher suggests solving exercise 5 of the worksheet in pairs. Correction of the exercise by a student.
Assessment	**Formative assessment:** ● Attendance; ● Punctuality; ● Behaviour: ● Attention and participation in class; ● Observation of the student's performance in solving the proposed exercises; ● Completion of worksheets (direct observation grids).
21st Century Skills	Critical thinking: students will be able to analyse data during practical experiments and communicate their conclusions. Collaboration: students will be able to collaborate within their groups and with other groups, helping each other to understand the content and experimental activities. Communication: students should be able to share conclusions and doubts with their classmates and teachers. Technological literacy: students will be able to use different technological tools to carry out tasks.
Remarks	--

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

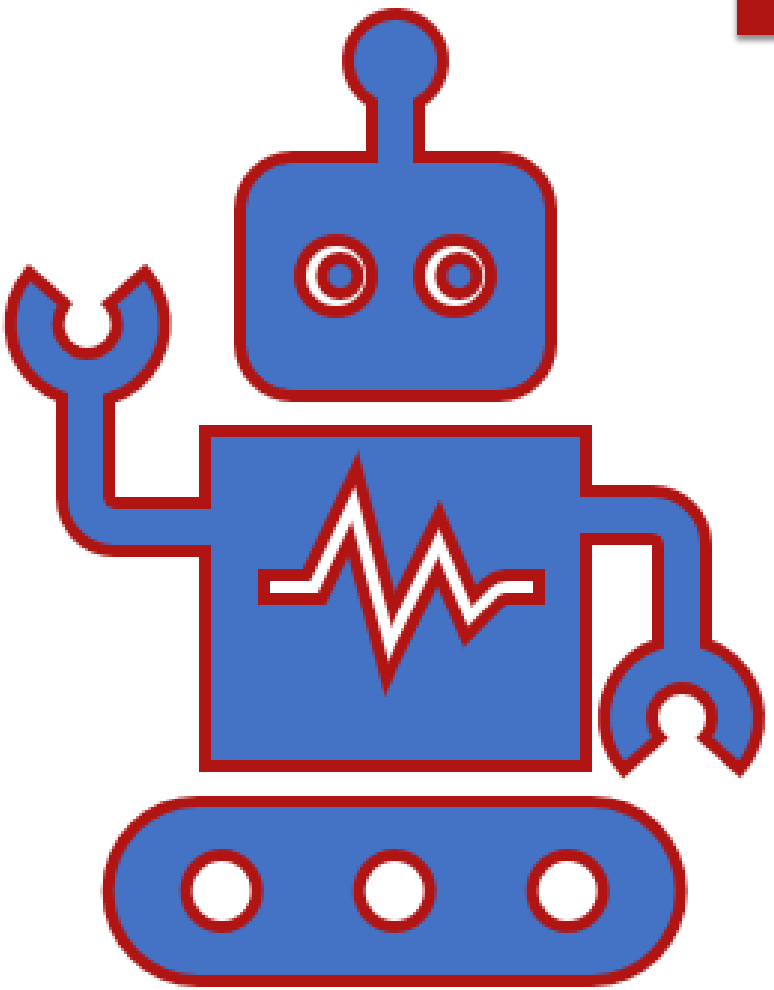
Programming



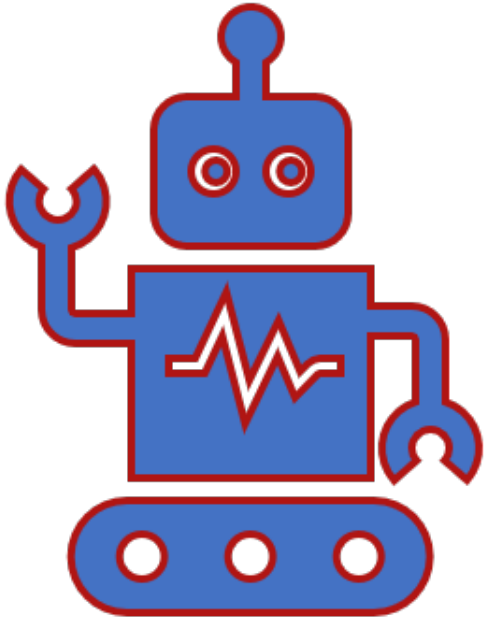
Co-funded by the
Erasmus+ Programme
of the European Union

Contents

- Robot morphology
- Actuators and sensors for robotics
- Robotic platforms based on Arduino microcontrollers
- Troubleshooting using Arduino microcontrollers



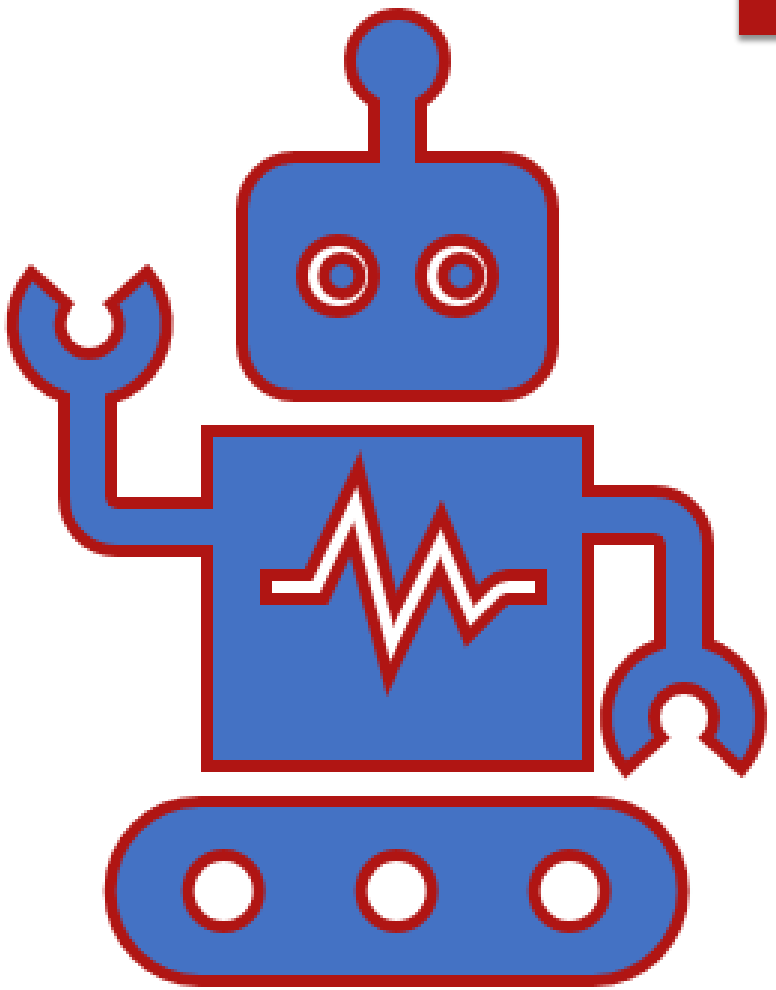
What´s a Robot?



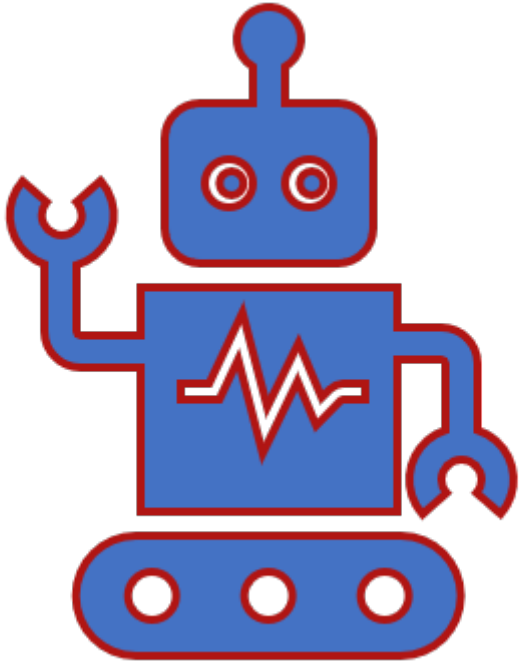
- An industrial robot is officially defined by ISO as an ‘automatically controlled, reprogrammable multipurpose manipulator, programmable in three or more axes’.
- Typical applications for industrial robots include casting, painting, welding, assembly, load handling, product inspection and testing, all carried out with relatively high precision, speed and robustness.

Robots´ operation and programming features

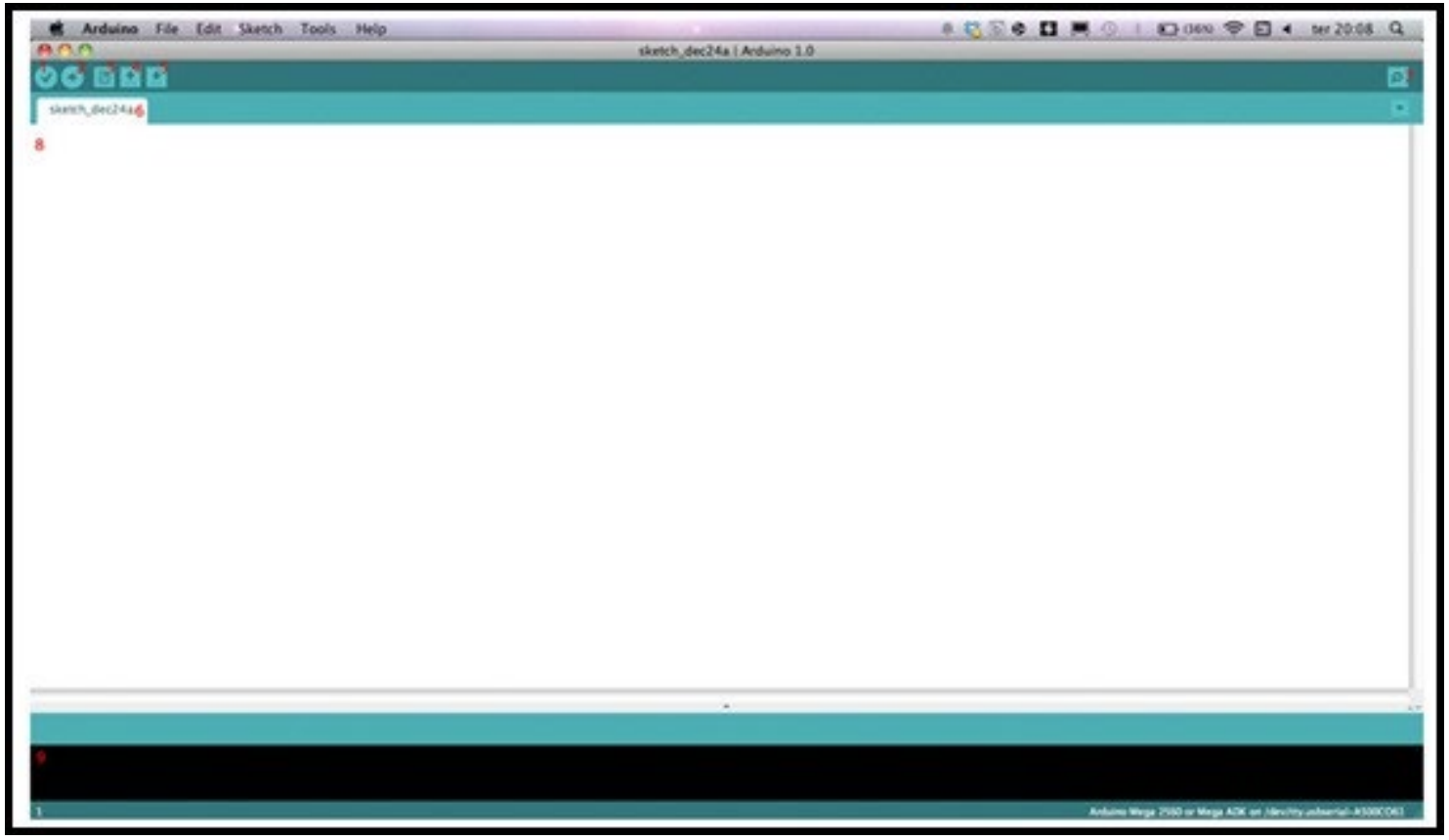
- They define actions by reading values presented by sensors.
- They are reprogrammable and therefore easy to reuse and reintegrate into production lines.
- Their actions are validated automatically without requiring human interaction to perform their function.
- Robots' morphology and kinematics;
- Robots´ navigation methods;
- Robot programming can be based on the Arduino platform, Raspeberry Pi and use of the ROS (Robotic Operating System) operating system.



Programming robots using the Arduino platform

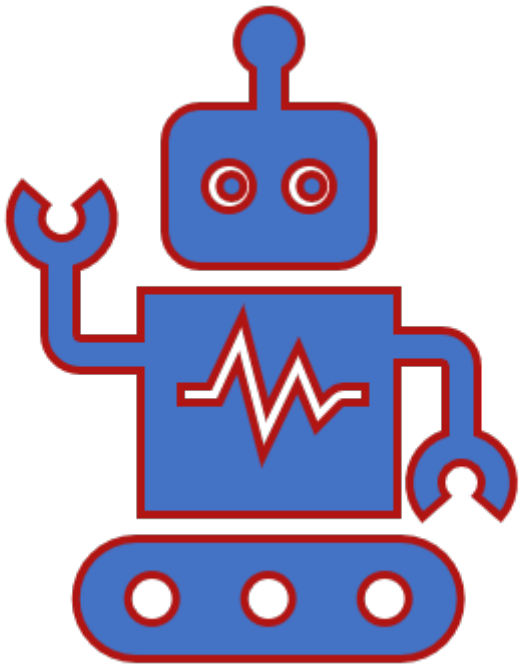


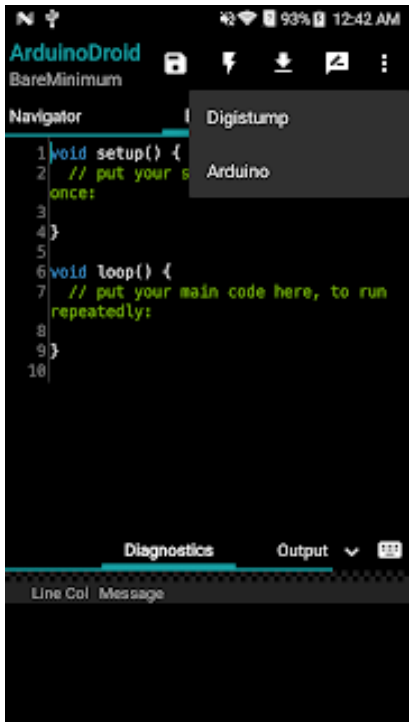
- IDE

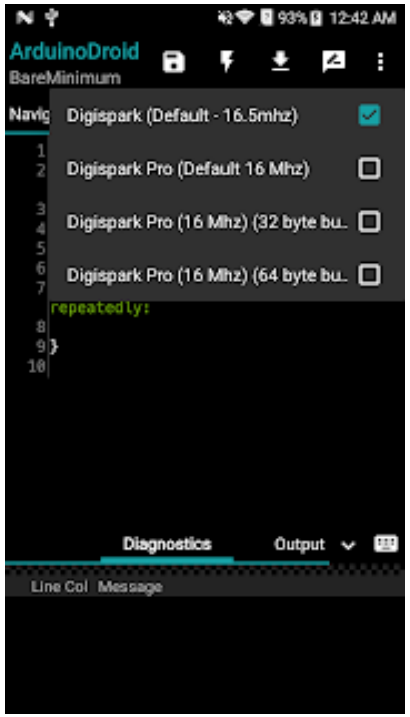


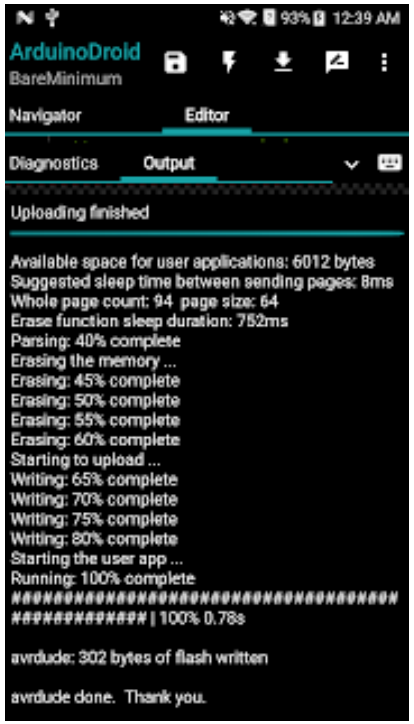
Programming robots using the Arduino platform

- ArduinoDroid

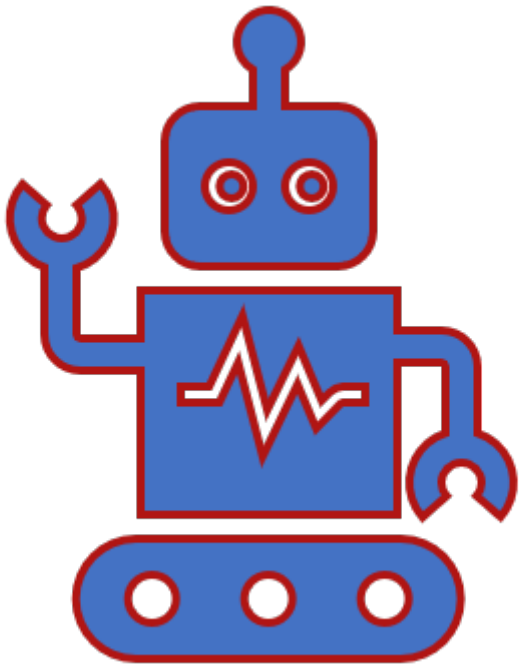




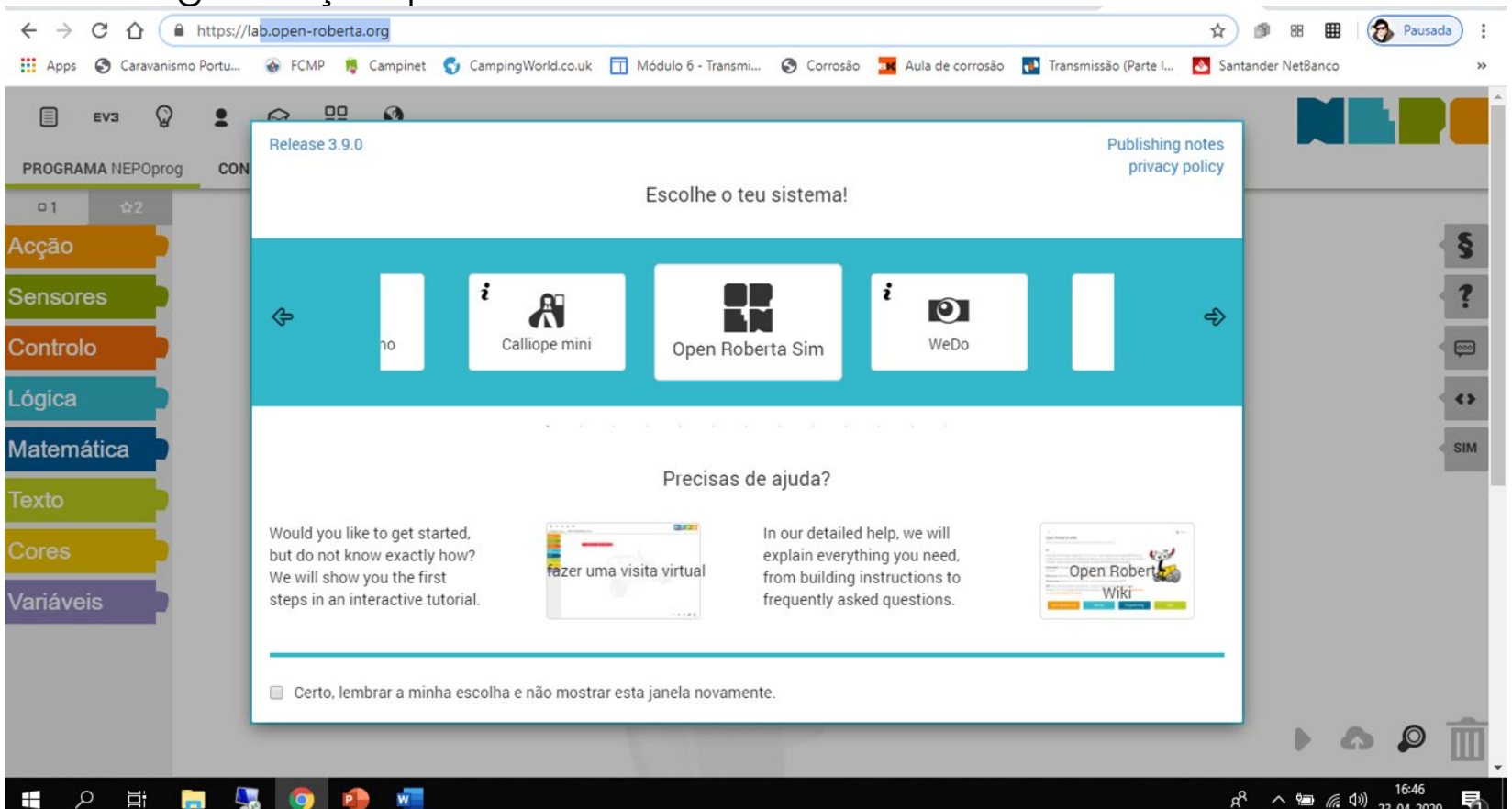




Programming robots using the Arduino platform



- Programação por Blocos



Sensors

Reflective optical sensor

Infrared Heart Rate Sensor

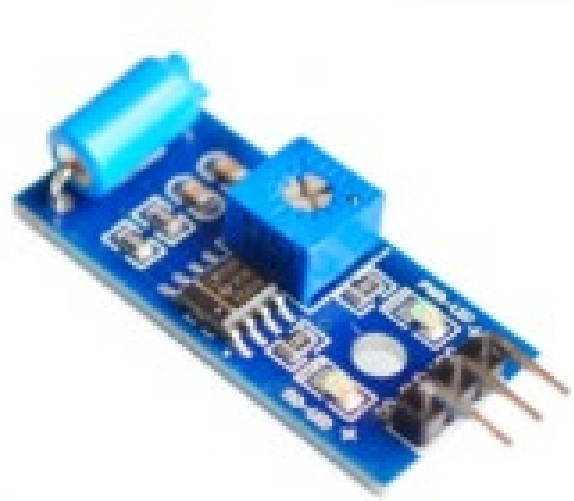
Water Level Sensor Module

Analogue distance sensor module

Analogue distance sensor module

Analogue distance sensor module

Motion and Vibration Sensor Module



Temperature and Humidity Sensor Module

Gas Sensor Module

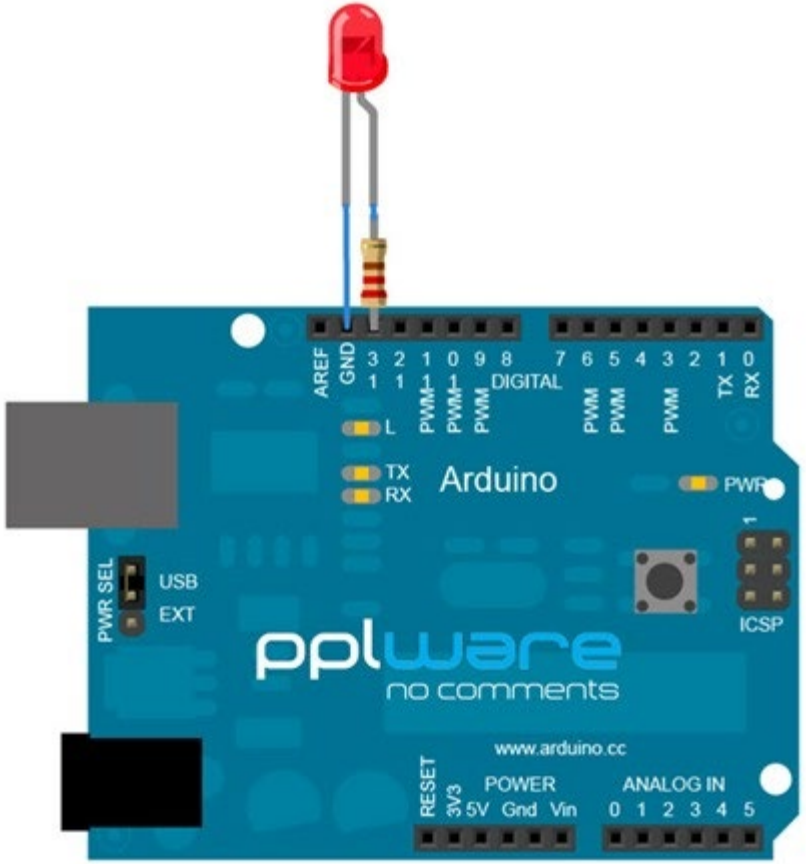
Besides all these options, there is also the possibility of adapting all conventional sensors to our needs...

Programming

- First challenge using the Arduino program…. Make a LED flash

Let´s use the platform

tinkercad.com

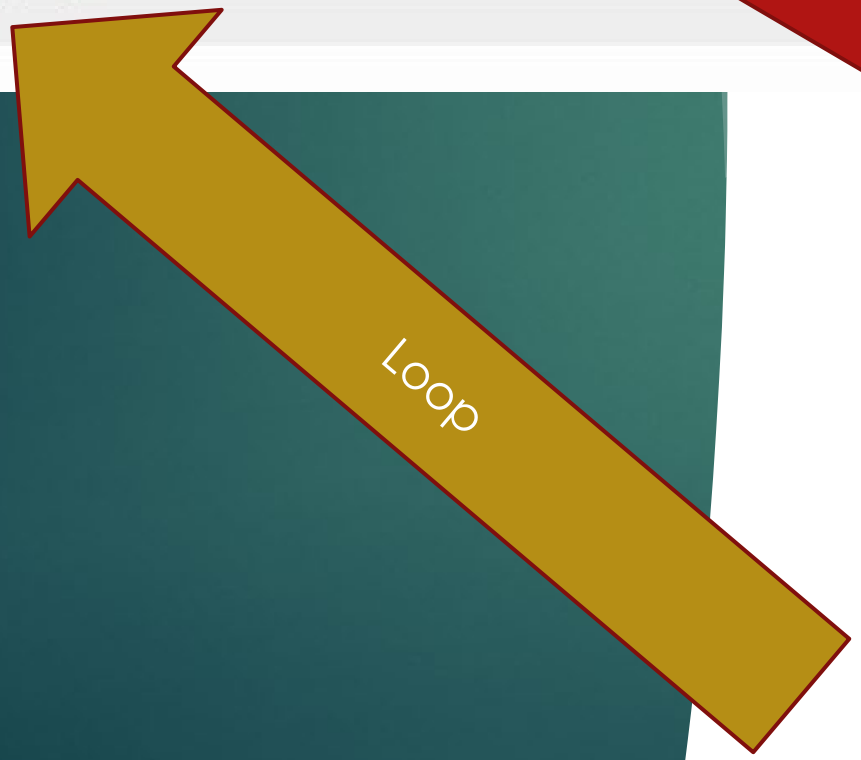


```
void setup() {
pinMode(13, OUTPUT);

}

void loop() {

}
```



```
void setup() {

pinMode(13, OUTPUT);

}

void loop() {

digitalWrite(13, HIGH);

}
```

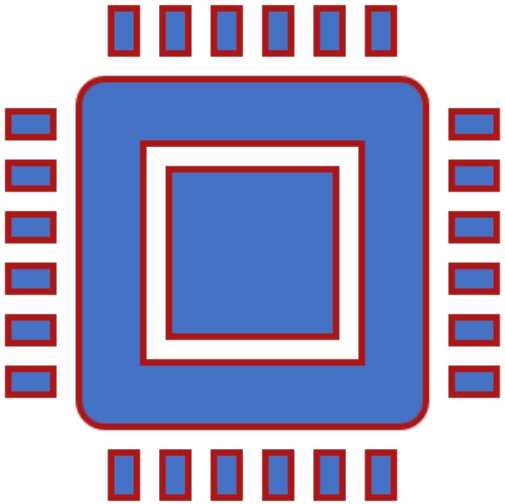
```
void setup() {
pinMode(13, OUTPUT);
}
void loop() {
digitalWrite(13, HIGH);
delay(1000);
digitalWrite(13, LOW);
delay(1000);
}
```

Usual commands

FUNCTIONS

DIGITAL INLETS AND OUTLETS

digitalRead()



Read the value of a specified digital pin, which can be HIGH or LOW.

Sintaxe

digitalRead(pino)

Parameters pin: the number of the Arduino digital pin you want to check

Return
HIGH or LOW

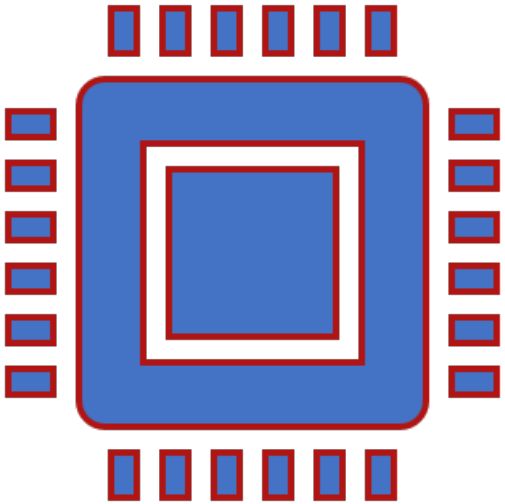
Example Code
Start pin 13 to the same value as pin 7, declared as input.

```
int ledPin = 13;  // LED conectado ao pino digital 13
int inPin = 7;    // botão conectado ao pino digital 7
int val = 0;      // variável para guardar o valor lido

void setup() {
  pinMode(ledPin, OUTPUT);    // configura o pino digital 13 como saída
  pinMode(inPin, INPUT);      // configura o pino digital 7 como entrada
}

void loop() {
  val = digitalRead(inPin);   // lê o pino de entrada
  digitalWrite(ledPin, val);  // aciona o LED com o valor lido do botão
}
```

digitalWrite()



Start a value HIGH or LOW in a digital pin

Sintaxe

digitalWrite(pino, valor)

Parameters pin: the number of the Arduino digital pin

value: HIGH or LOW

Example Code

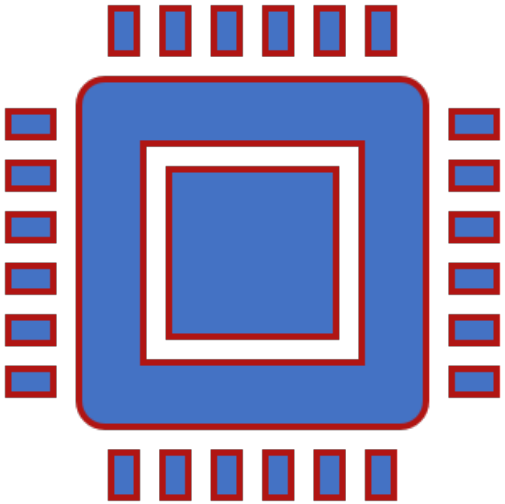
0 code configures digital pin 13 as OUTPUT and switches its state between HIGH and LOW

```
void setup() {
  pinMode(13, OUTPUT);    // configura o pino digital 13 como saída
}

void loop() {
  digitalWrite(13, HIGH); // ativa o pino digital 13
  delay(1000);            // espera por um segundo
  digitalWrite(13, LOW);  // desativa o pino digital 13
  delay(1000);            // espera por um segundo
}
```

ANALOGUE INPUTS AND OUTPUTS

analogRead()



Read the value of a specified analogic pin.

Sintaxe

analogRead(pino)

Parameters pin: the name of the inlet analogue pin you want to read

Example code

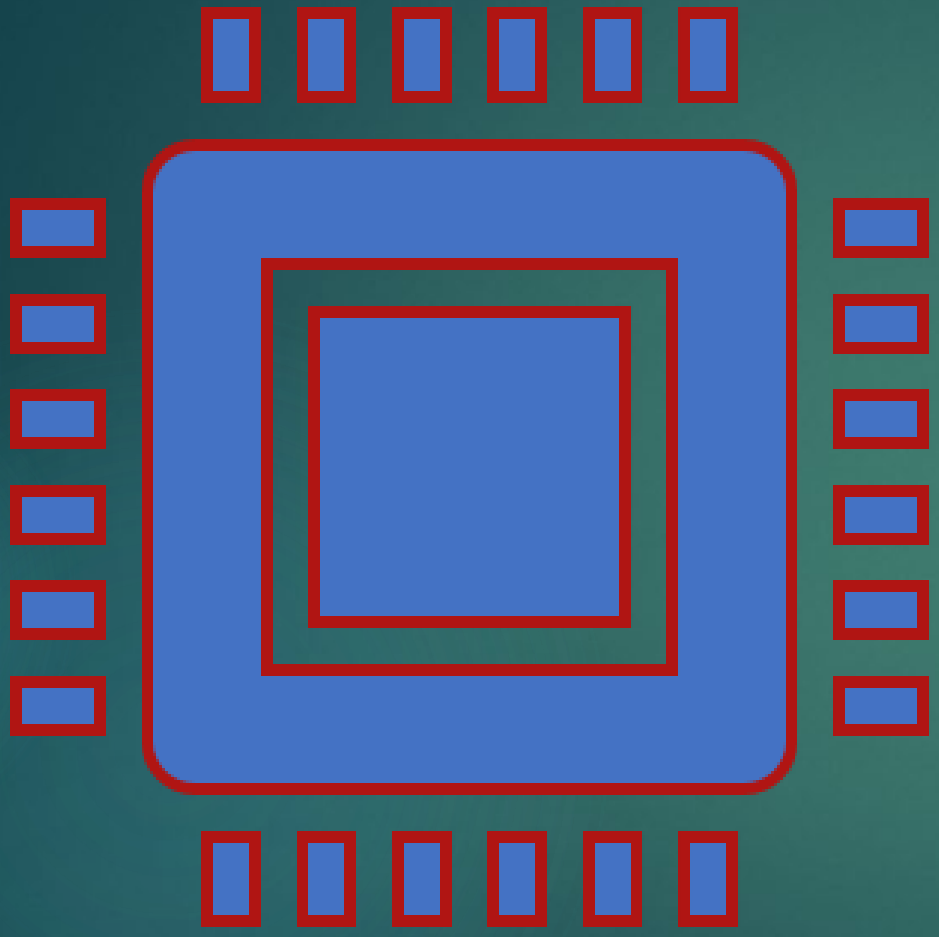
The code below reads the value of an analogue input pin and displays its value on the serial inlet.

```
int analogPin = A3; // terminal do meio de um potênciometro conectado ao pino analógico 3
                    // terminais mais externos são conectados um no ground e o outro em +5V
int val = 0;        // variável para guardar o valor lido

void setup() {
  Serial.begin(9600);           // configura a porta serial
}

void loop() {
  val = analogRead(analogPin);  // lê o pino de entrada
  Serial.println(val);          // imprime o valor na porta serial
}
```

MATHEMATICAL FUNCTIONS



abs()

constrain()

map()

max()

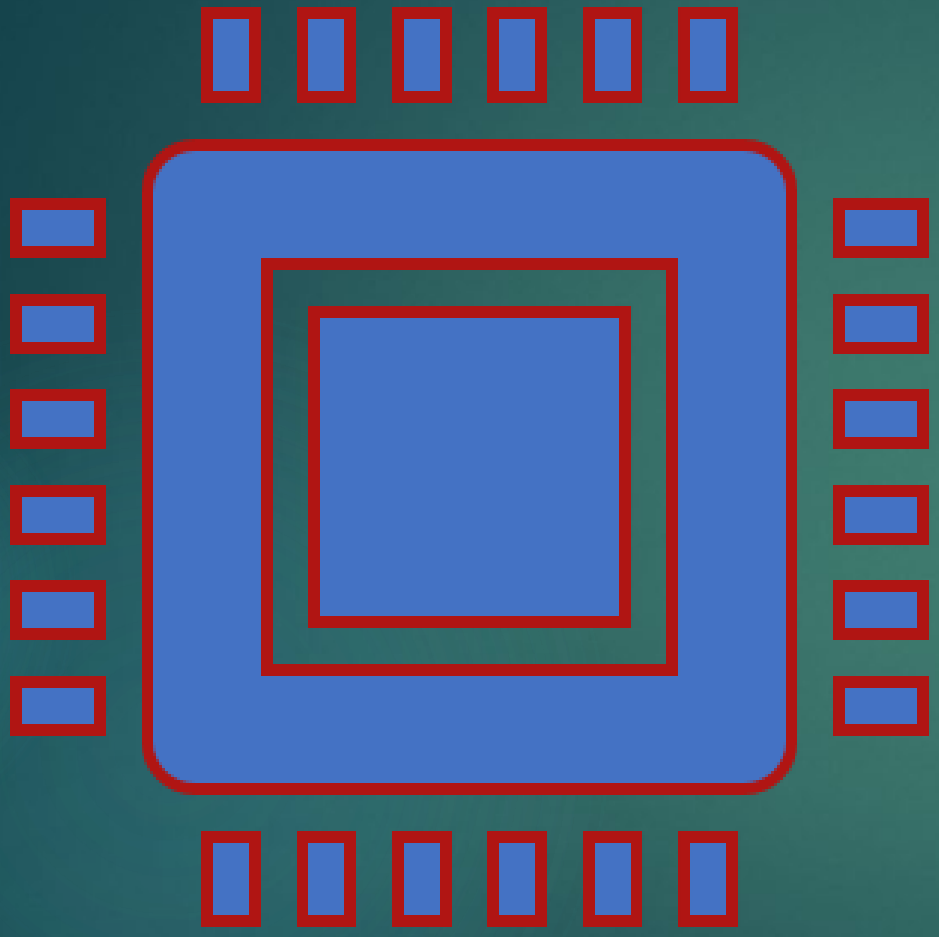
min()

pow()

sq()

sqrt()

TIME FUNCTIONS



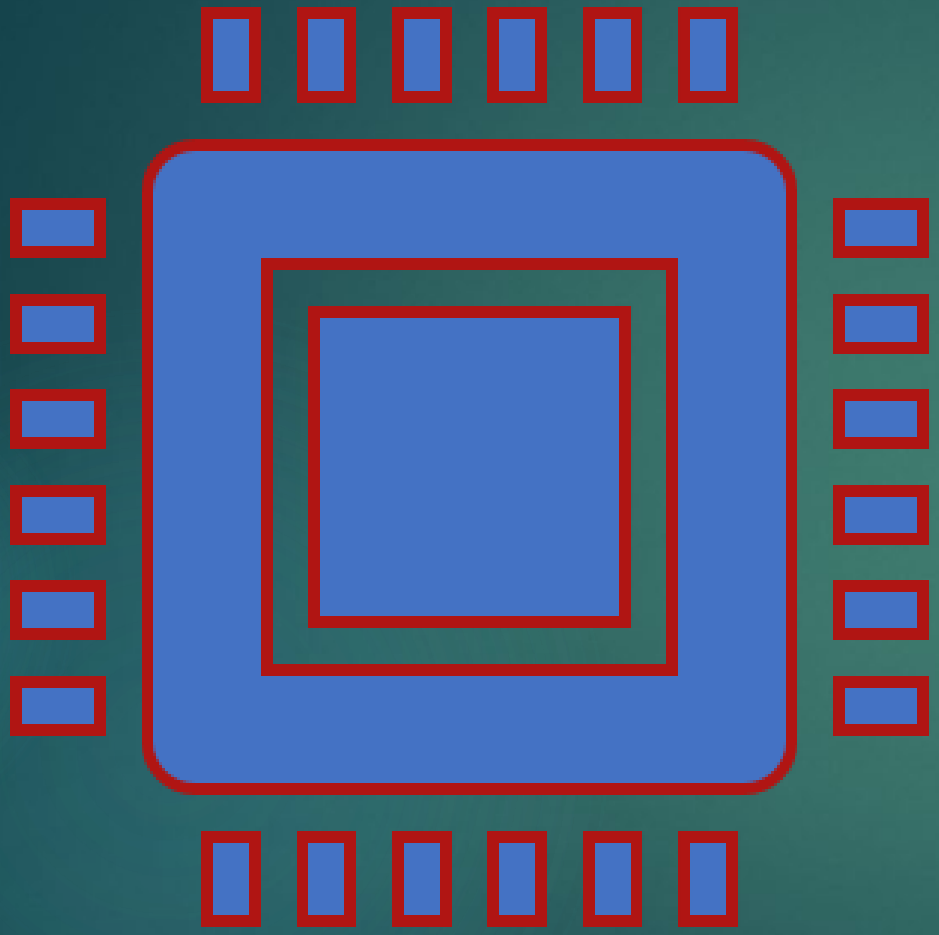
delay()

delayMicroseconds()

micros()

millis()

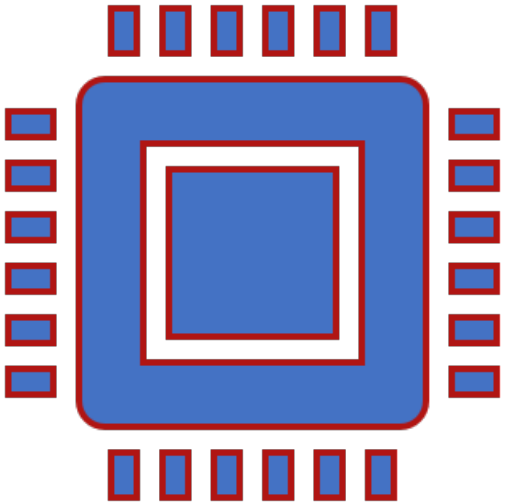
COMMUNICATION



Serial

Stream

IF



The IF command checks a condition and executes the following command or a block of commands delimited by curly braces if the condition is true.

Sintaxe

```
if (condiition) {
  //comando(s)
}
```

Example Codes

Braces can be omitted after an IF statement. If this is done, the next line (defined by the semicolon) is interpreted as the only conditional command. For more than one command, use curly braces to delimit the command block.

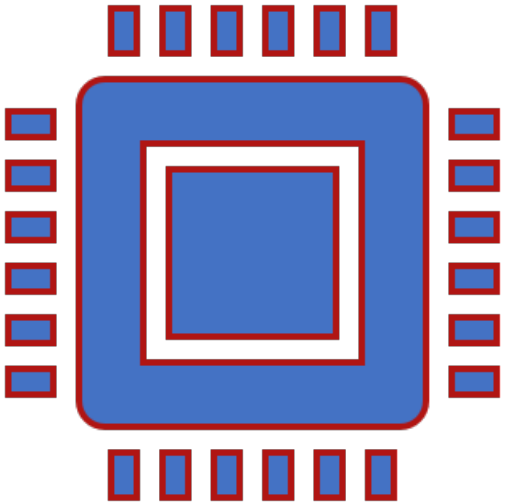
```
if (x > 120) {
  digitalWrite(pinoLED, HIGH);
}

if (x > 120) {
  digitalWrite(pinoLED, HIGH);
}

if (x > 120) {
  digitalWrite(pinoLED, HIGH);
}

if (x > 120) {
  digitalWrite(pinoLED1, HIGH);
  digitalWrite(pinoLED2, HIGH);
} // todas as formas acima estão corretas
```

ELSE



The IF...ELSE combination allows greater control over the code flow than the more basic if command, as it allows multiple tests to be grouped together. An else clause (if present) will be executed if the condition of the if statement results in false. The else can proceed another if test, so multiple, mutually exclusive tests can be run at the same time.

Sintaxe

```
if (condition1) {
  // action A
}
else if (condition 2) {
  // action B
}
else {
  // action C
}
```

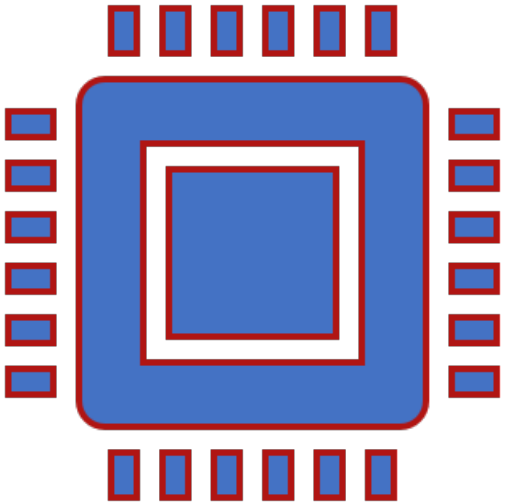
Example Code

Below there is a code extract from a temperature control system

```
if (temperatura >= 70) {
  //Perigo! Desligar o sistema
}
else if (temperatura >= 60 && temperatura < 70) {
  //Cuidado! Requerida a atenção do usuário
}
else {
  //Seguro! Continue as tarefas usuais...
}
```

LOGICAL OPERATORS

E - &&

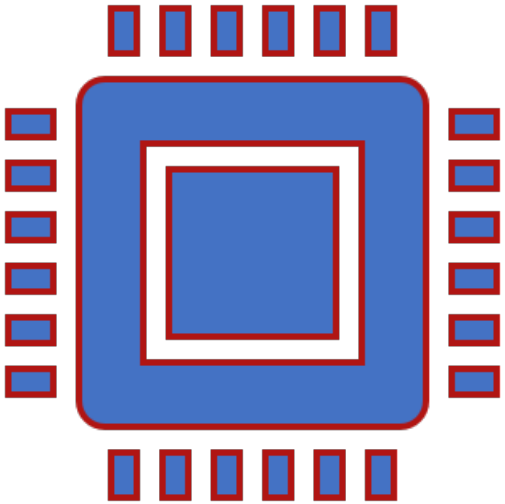


Logical E leads to true only if both operands are true.

Example Code

```
if (digitalRead(2) == HIGH  && digitalRead(3) == HIGH) {  // se AMBOS os botões estão em HIGH
  // código a ser executado caso as duas condições sejam verdadeiras
}
```

Ou - | |

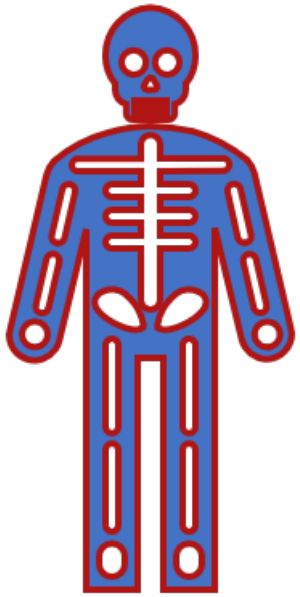


Logical OR results in true if at least one of the operands is true

Example code

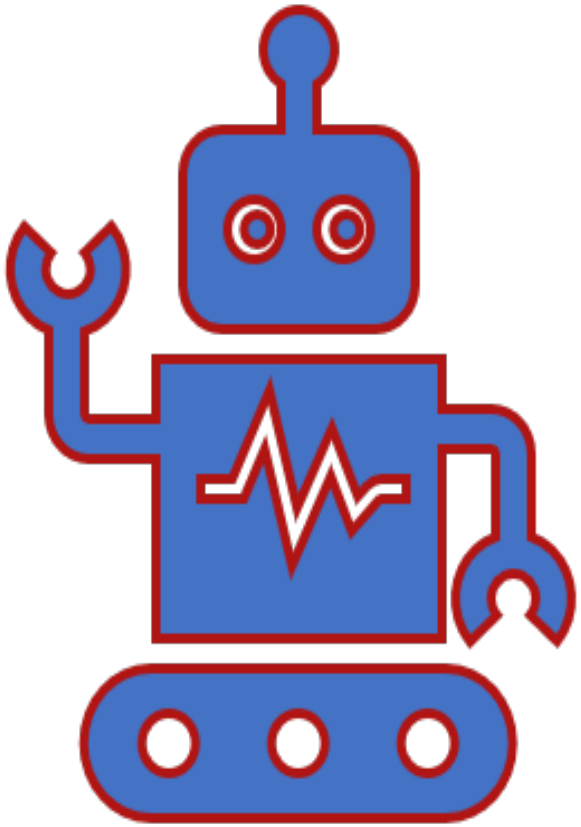
```
if (x > 0 || y > 0) { // se x ou y é maior que zero
  // código a ser executado
}
```

CHALLENGE



Using TINKERCAD, build a robot that commands two DC motors to run when it is not heating up and does not encounter any obstacles in its path.You should place a motion sensor on the right and left side of the machine.You should place a motor on the left and right side of the machine. The Left motor should stop rolling when it detects the presence of an object on the right side of the machine. The Right motor should stop rolling when it detects the presence of an object on the left side of the machine.

Programming



Co-funded by the
Erasmus+ Programme
of the European Union

Worksheet No. 1

Learning Unit	**Aprender C++ com robôs**
Authors	Nuno Barbosa
School	FORAVE – ASSOCIAÇÃO PARA A EDUCAÇÃO TECNOLÓGICA DO VALE DO AVE
Date	

1 - Write a C++ script to get the battery status of the Botn'Roll.

2 - Write a C++ script so that Botn'Roll moves forwards in a straight line for 5 seconds.

3 - Write a C++ script so that Botn'Roll moves forwards in a straight line for 5 seconds and then moves backwards for another 5 seconds.

4 - Write a C++ script so that Botn'Roll dodges obstacles using the infrared sensors.

5 - Write a C++ script so that Botn'Roll follows a black line on a white background using the line follower sensor.

Co-funded by the Erasmus+ Programme of the European Union