# Procédure de réglage d'un capteur analogique : Nom Prénom TP Support

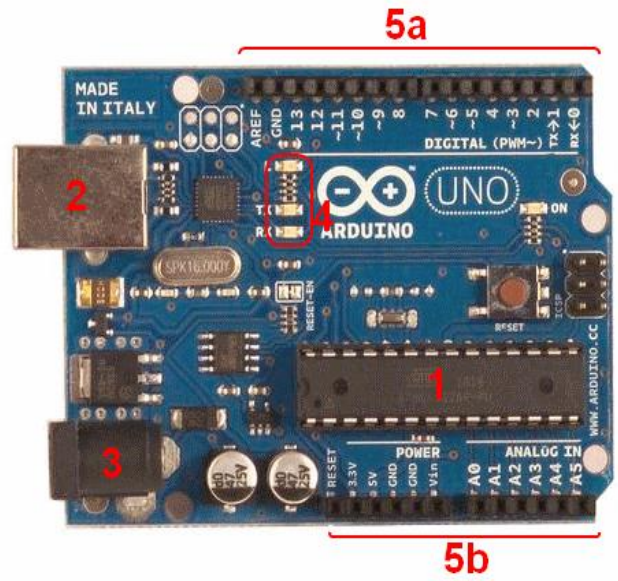
## Programme à analyser

***Commenter au crayon, directement dans le programme***

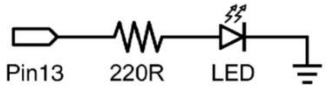
```
/*  Calibration
 Demonstrates one technique for calibrating sensor input.  The
 sensor readings during the first ten seconds of the sketch
 execution define the minimum and maximum of expected values
 attached to the sensor pin.
 The sensor minimum and maximum initial values may seem backwards.
 Initially, you set the minimum high and listen for anything
 lower, saving it as the new minimum. Likewise, you set the
 maximum low and listen for anything higher as the new maximum.

 The circuit:
 * Analog sensor (potentiometer will do) attached to analog input 0
 * LED attached from digital pin 13 to ground

 created 29 Oct 2008 By David A Mellis modified 30 Aug 2011
 By Tom Igoe http://arduino.cc/en/Tutorial/Calibration
 This example code is in the public domain.

 */         // These constants won't change:
const int sensorPin = A0;    // pin that the sensor is attached to
const int ledPin = 13;       // pin that the LED is attached to

// variables:
int sensorValue = 0;         // the sensor value
int sensorMin = 1023;        // minimum sensor value
int sensorMax = 0;           // maximum sensor value

void setup() {
  // turn on LED to signal the start of the calibration period:
  pinMode(13, OUTPUT);
  digitalWrite(13, HIGH);
  // initialize serial communication at 9600 bits per second:
 Serial.begin(9600);

  // calibrate during the first five seconds
  while (millis() < 5000) {
    sensorValue = analogRead(sensorPin);

    // record the maximum sensor value
    if (sensorValue > sensorMax) {
      sensorMax = sensorValue;
    }
    // record the minimum sensor value
    if (sensorValue < sensorMin) {
      sensorMin = sensorValue;

    // affichage de la valeur num lors de la calibration
Serial.println(sensorValue);    }
  }
  // signal the end of the calibration period
  digitalWrite(13, LOW);
}
void loop() {
  // read the sensor:
  sensorValue = analogRead(sensorPin);
  // apply the calibration to the sensor reading
  sensorValue = map(sensorValue, sensorMin, sensorMax, 0, 255);

Serial.println(sensorValue);
  // in case the sensor value is outside the range seen during calibration
  sensorValue = constrain(sensorValue, 0, 255);
  // fade the LED using the calibrated value:
  analogWrite(ledPin, sensorValue);
}
```

**Procédure de réglage d'un capteur analogique : Nom Prénom TP Support**

## La carte UNO : Schéma à compléter (potentiomètre + LED) en respectant les numéros de connexion et les polarités




- **Attention**, on ne branche pas n'importe quoi n'importe comment.




- **Pas** de moteur/lampe, de composants nécessitant de la puissance directement connecté à un port d'entrée/sortie qui est **en mode sortie** !

**1) Le microcontrôleur,** cerveau de notre carte, reçoit le programme créé, le stocke en mémoire puis l'exécute.

**2) et 3) Alimentation :**

Le microcontrôleur fonctionnant sous 5V, la carte peut être alimentée en 5V par le port USB (en **2**) ou bien par une alimentation externe (en **3**) qui est comprise entre 7V et 12V et peut par exemple être fournie par une pile 9V. Un régulateur se charge ensuite de réduire la tension à 5V pour le bon fonctionnement de la carte.

**4) Visualisation :**
Les trois "points blancs" entourés en rouge sont des LED dont la taille est de l'ordre du millimètre.

- La LED tout en haut du cadre est connectée à une broche du microcontrôleur et sert au test matériel. Quand on branche la carte au PC, elle clignote quelques secondes.
- Les deux LED du bas du cadre servent à visualiser l'activité sur la voie série (une pour l'émission et l'autre pour la réception). Le téléchargement du programme dans le microcontrôleur se faisant par cette voie, on peut les voir clignoter lors du chargement.

**5) Connexion des entrées / sorties :**
La carte Arduino ne possède pas de composants pouvant être utilisés pour un programme, mis à part la LED connectée à la broche 13 du microcontrôleur. On ajoute ses composants à l'extérieur ;

⇒ **5.a) Entrée/sorties numériques ou logiques**
⇒ **5.b) Reset / Alimentation / Entrées analogiques**

**6) Extensions :**
Avec sa connectique la carte est "extensible", car l'on peut y brancher tous types de montages et modules.
La carte Arduino Uno peut être étendue avec des shields, comme le « Shield de puissance » qui comporte 2 hacheurs en pont H permettant le pilotage de 2 MCC ou d'un moteur pas à pas à 2 enroulements.