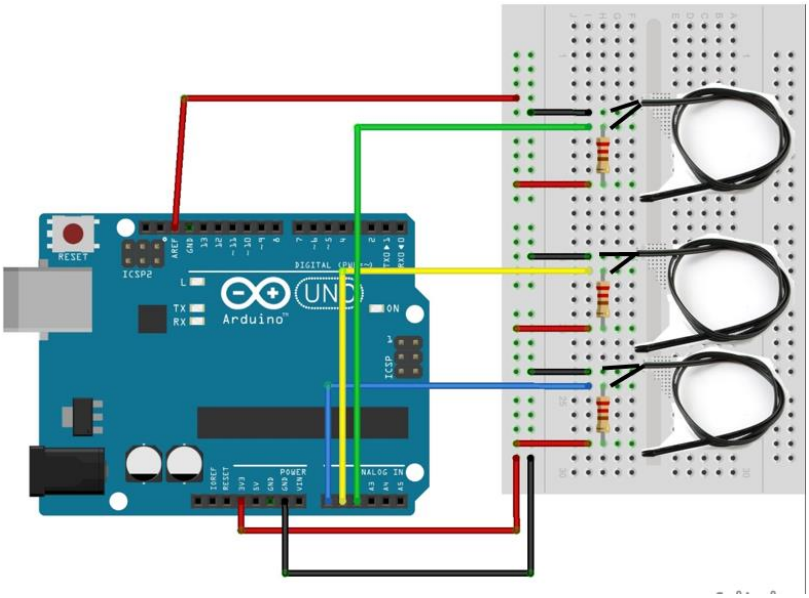
Friday 10th March - Materials Decathlon - Challenge n°10 - 40'

ECO-ALTERNATIVE TO DISPOSABLE PLASTIC PLATES

Everyday in a big city like Milan thousands of students eat in school canteens. By the end of the year the number of plastic disposable sets (dish, knife and fork, glass, etc.) thrown away amounts to approx. 13 millions. 550 tons of plastic sent to the incinerator (since due to food contamination this plastic cannot be recycled!). Multiply it for all Italian schools and … you will swim in a sea of plastic! Quite recently a new biocompatible kind of plate has come into the market and has already been adopted by many municipalities for their school canteens. It looks like a plate made of biscuit. It can contain liquid and semiliquid for approx. one hour before melting. The manufacturer also states that it ***keeps food warmer for a longer time****. Let's test this comparing three different kinds of plates: a) disposable plastic; b) classic ceramic; c) biocompatible biscuit.*

On the desk you have the three plates and some hot "food" [*Actually it's only hot water!! Let's imagine it's some kind of soup!!*]. Once the "food" is poured into the plates we will monitor temperature and plot the three cooling curves. We will use Arduino with three temperature sensors. This will NOT plot a real time graph but data collection will be shown on the serial monitor, then data may be pasted and copied in Excel to make ONE graph with THREE cooling curves.
Warning*: be careful to put the same quantity of "food" into each plate; the initial temperature of the food should be the same as you have token it from the same "pot".*

1. **The circuit** - Each sensor needs a 10kOhm resistor. Connections are
   a. One leg of the temp probe to the Ground (**GND**) (black)
   b. The second leg of the probe connected to the 10KOhm resistor
   c. The second leg of the resistor to **VCC** 3,3 V (red)
   d. The common ends (green, yellow, blue) are the signal and go to **A0**, **A1**, **A2** respectively
   e. **AREF** to **VCC**3,3 V (red cable on top)



2. **The Code** - Use "***3_thermistor_aref_2.ino***". You can modify the acquisition rate (now it is set to 10 seconds)
3. **Data collection** - Insert the temperature probes in the three plates and start data collection. Open the serial monitor to see the readings scrolling with time. Take measure for at least 15'.
4. Copy the data and paste them in a TXT file, then import in Excel and plot the comparative graph.

☞ OUTPUT WANTED: plot of the three cooling curves + short comment on the features of the different plates. ***Paste everything on a word file and save it on the PC at your disposal.***

 Funded by EU under the Erasmus+ KA2 grant N° 2014-1-IT02-KA201-003604_1.

Teachers' Notes

Technical notes:

- See the "sensors" section of the project documents for alternatives, such as use of Vernier probes with and without Arduino. A real time graph surely is more immediate and powerful in conveying the idea of the differences in cooling. However we use Arduino here to insist on the possibilities offered by microcontrollers + low cost sensors to home labs and inquiry based science.

Organizational notes:

- In this Challenge there is NO Answer Sheet: JUST GIVE OVER THE GRAPHS!

Correction grid

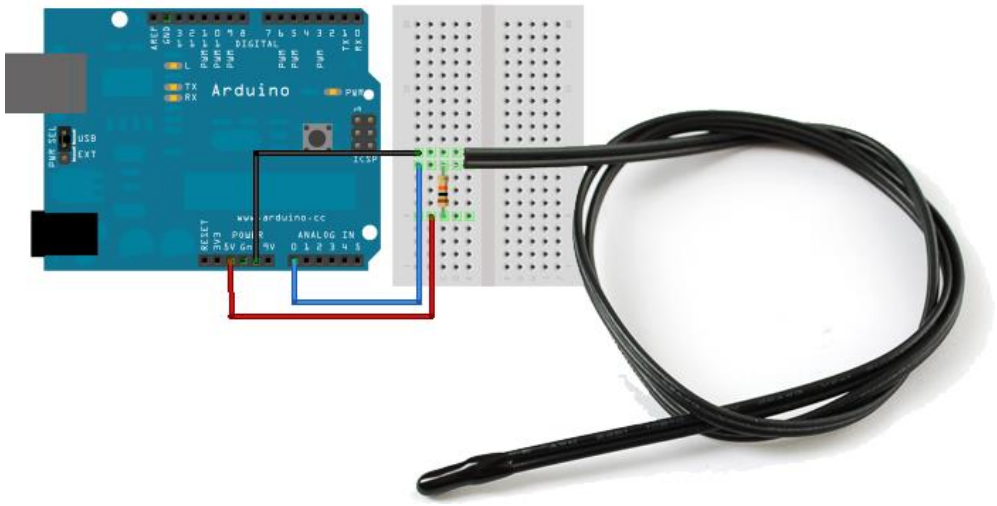
Question or Request	Note	Max. score
Graphs of the cooling curve	Just control that there aren't evident mistakes Evaluate if the graph are clearly readable	10
Comment	Evaluate if well motivated on data collected	10

Key to Answer

The best one in keeping food warm is the classical ceramic plate. Pappami, the eco plate rates well and the plastic plates cool down much quicker than the other two.

Technical Specification on Sensors and Code Used

4. TERMISTORE 10K (807939 ROBOTITALY)
https://learn.adafruit.com/thermistor/using-a-thermistor



Sensore: a Ground (nero) e al resistore
Resistore (10 kOhm) :
Segnale (blu) ad A0
Secondo estremo a 5V (rosso)

 Funded by EU under the Erasmus+ KA2 grant N° 2014-1-IT02-KA201-003604_1.

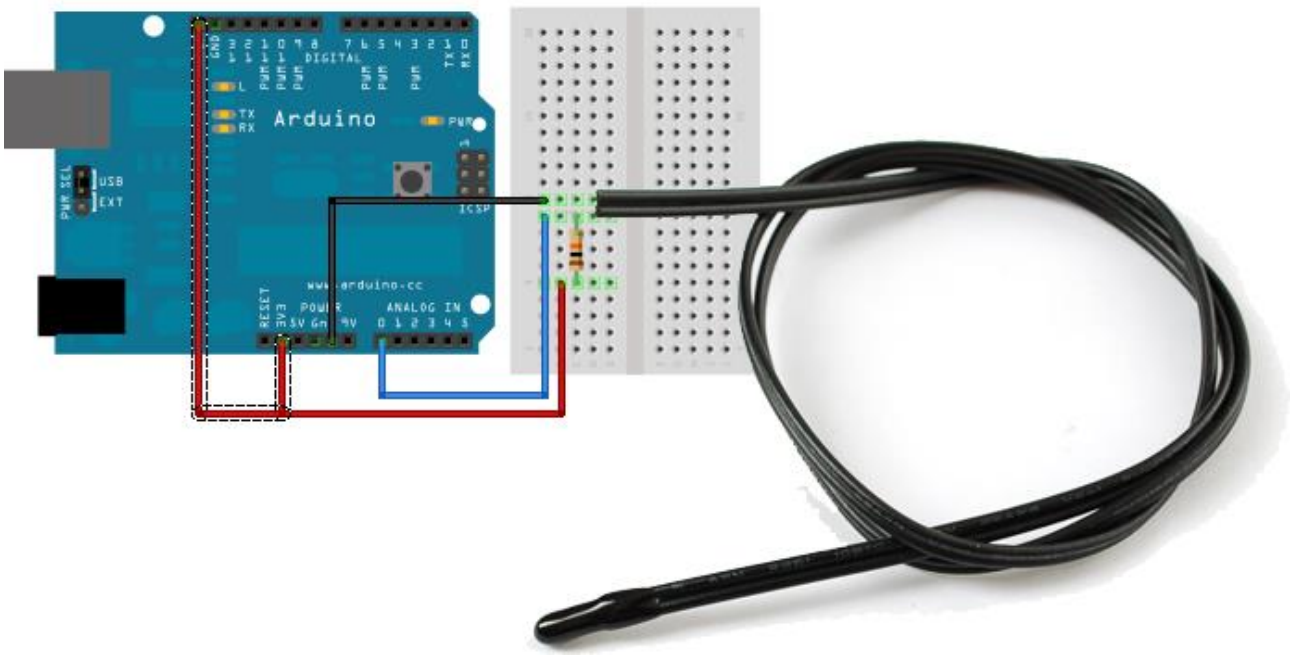
Migliorare la lettura dei valori

When you collect analogic data , particularly with a board like Arduino , which is very "noisy"", you have two ways to improve this aspect.

1. Use the 3.3V pin as *analogreference.* The 5V of power directly come from PC to Arduino board via USB and this line is much more "noisy" then the 3,3 V (before reaching the board power goes through a secondary filter specifically dedicated to stabilization).
   Connect 3,3 V to AREF and use this as power source (VCC). Just do not forget to insert in the new sketch the line of code that sets 3,3V analogReference(EXTERNAL); in voidsetup.

2. Prendete una serie di letture e fatene la media anzichè limitarvi ad una sola lettura questo per limitare le fluttuazioni sul "rumore". Prendete almeno 5 misure su cui fare la media ed ottenere la lettura finale.

This is the new connection scheme



Code for 1 temperature probe (thermistor_aref2.ino**)**

```
// which analog pin to connect
#define THERMISTORPIN A0
// resistance at 25 degrees C
#define THERMISTORNOMINAL 10000
// temp. for nominal resistance (almost always 25 C)
#define TEMPERATURENOMINAL 25
// how many samples to take and average, more takes longer
// but is more 'smooth'
#define NUMSAMPLES 5
// The beta coefficient of the thermistor (usually 3000-4000)
```

 Funded by EU under the Erasmus+ KA2 grant N° 2014-1-IT02-KA201-003604_1.

```
#define BCOEFFICIENT 3950
// the value of the 'other' resistor
#define SERIESRESISTOR 10000

int samples[NUMSAMPLES];

void setup(void) {
Serial.begin(9600);
analogReference(EXTERNAL);
}

void loop(void) {
uint8_ti;
float average;

 // take N samples in a row, with a slight delay
for (i=0; i< NUMSAMPLES; i++) {
samples[i] = analogRead(THERMISTORPIN);
delay(10);
 }

 // average all the samples out
average = 0;

for (i=0; i< NUMSAMPLES; i++) {
average += samples[i];
 }
average /= NUMSAMPLES;

Serial.print("Average analog reading ");
Serial.println(average);

 // convert the value to resistance
average = 1023 / average - 1;
average = SERIESRESISTOR / average;
Serial.print("Thermistor resistance ");
Serial.println(average);

floatsteinhart;
steinhart = average / THERMISTORNOMINAL;     // (R/Ro)
steinhart = log(steinhart);                  // ln(R/Ro)
steinhart /= BCOEFFICIENT;                   // 1/B * ln(R/Ro)
steinhart += 1.0 / (TEMPERATURENOMINAL + 273.15); // + (1/To)
steinhart = 1.0 / steinhart;                 // Invert
```

 Funded by EU under the Erasmus+ KA2 grant N° 2014-1-IT02-KA201-003604_1.

```
steinhart -= 273.15;                    // convert to C

Serial.print("Temperature ");
Serial.print(steinhart);
Serial.println(" *C");

delay(1000);
}
```

Code for 3 temperature probes (_3_thermistor_aref2.ino)

```
// which analog pin to connect
#define THERMISTORPIN_0 A0
#define THERMISTORPIN_1 A1
#define THERMISTORPIN_2 A2
// resistance at 25 degrees C
#define THERMISTORNOMINAL 10000
// temp. for nominal resistance (almost always 25 C)
#define TEMPERATURENOMINAL 25
// how many samples to take and average, more takes longer
// but is more 'smooth'
#define NUMSAMPLES 5
// The beta coefficient of the thermistor (usually 3000-4000)
#define BCOEFFICIENT 3950
// the value of the 'other' resistor
#define SERIESRESISTOR 10000

int samples[NUMSAMPLES];

void setup(void) {
Serial.begin(9600);
analogReference(EXTERNAL);
}

void loop(void) {
uint8_ti;
float average_0;
float average_1;
float average_2;

 //sensore0 take N samples in a row, with a slight delay
for (i=0; i< NUMSAMPLES; i++) {
samples[i] = analogRead(THERMISTORPIN_0);
delay(10);
 }

 // average all the samples out
 average_0 = 0;
```

 Funded by EU under the Erasmus+ KA2 grant N° 2014-1-IT02-KA201-003604_1.

```
for (i=0; i< NUMSAMPLES; i++) {
   average_0 += samples[i];
  }
  average_0 /= NUMSAMPLES;

  //Serial.print("Average analog reading ");
  //Serial.println(average_0);


  // convert the value to resistance
  average_0 = 1023 / average_0 - 1;
  average_0 = SERIESRESISTOR / average_0;
  //Serial.print("Thermistor_0 resistance ");
 // Serial.println(average_0);

float steinhart_0;
  steinhart_0 = average_0 / THERMISTORNOMINAL;     // (R/Ro)
  steinhart_0 = log(steinhart_0);                  // ln(R/Ro)
  steinhart_0 /= BCOEFFICIENT;                     // 1/B * ln(R/Ro)
  steinhart_0 += 1.0 / (TEMPERATURENOMINAL + 273.15); // + (1/To)
  steinhart_0 = 1.0 / steinhart_0;                 // Invert
  steinhart_0 -= 273.15;                         // convert to C

 //Serial.print("Temperature_0 ");
 //Serial.println(steinhart_0);
 // Serial.println(" *C");

    //Sensore 1 take N samples in a row, with a slight delay
for (i=0; i< NUMSAMPLES; i++) {
samples[i] = analogRead(THERMISTORPIN_1);
delay(10);
  }

  // average all the samples out
  average_1 = 0;
for (i=0; i< NUMSAMPLES; i++) {
   average_1 += samples[i];
  }
  average_1 /= NUMSAMPLES;

  //Serial.print("Average analog reading ");
  //Serial.println(average_1);


  // convert the value to resistance
  average_1 = 1023 / average_1 - 1;
  average_1 = SERIESRESISTOR / average_1;
 // Serial.print("Thermistor_1 resistance ");
  //Serial.println(average_1);

float steinhart_1;
```

 Funded by EU under the Erasmus+ KA2 grant N° 2014-1-IT02-KA201-003604_1.

```
 steinhart_1 = average_1 / THERMISTORNOMINAL;     // (R/Ro)
 steinhart_1 = log(steinhart_1);                  // ln(R/Ro)
 steinhart_1 /= BCOEFFICIENT;                     // 1/B * ln(R/Ro)
 steinhart_1 += 1.0 / (TEMPERATURENOMINAL + 273.15); // + (1/To)
 steinhart_1 = 1.0 / steinhart_1;                 // Invert
 steinhart_1 -= 273.15;                         // convert to C

 //Serial.print("Temperature_1 ");
 //Serial.println(steinhart_1);
// Serial.println(" *C");

   //Sensore2 take N samples in a row, with a slight delay
for (i=0; i< NUMSAMPLES; i++) {
samples[i] = analogRead(THERMISTORPIN_2);
delay(10);
 }

 // average all the samples out
 average_2 = 0;
for (i=0; i< NUMSAMPLES; i++) {
   average_2 += samples[i];
 }
 average_2 /= NUMSAMPLES;

 //Serial.print("Average analog reading ");
 //Serial.println(average_2);


 // convert the value to resistance
 average_2 = 1023 / average_2 - 1;
 average_2 = SERIESRESISTOR / average_2;
// Serial.print("Thermistor_2 resistance ");
 //Serial.println(average_2);

float steinhart_2;
 steinhart_2 = average_2 / THERMISTORNOMINAL;     // (R/Ro)
 steinhart_2 = log(steinhart_2);                  // ln(R/Ro)
 steinhart_2 /= BCOEFFICIENT;                     // 1/B * ln(R/Ro)
 steinhart_2 += 1.0 / (TEMPERATURENOMINAL + 273.15); // + (1/To)
 steinhart_2 = 1.0 / steinhart_2;                 // Invert
 steinhart_2 -= 273.15;                         // convert to C

  //Serial.print("Temperature_2 ");
Serial.print(steinhart_0);
Serial.print(",");

Serial.print(steinhart_1);
Serial.print(",");

Serial.println(steinhart_2);
```

 Funded by EU under the Erasmus+ KA2 grant N° 2014-1-IT02-KA201-003604_1.

```
 //Serial.println(" *C");

delay(10000);
}
```

Acknowledgments

Thanks for offering the ecoplates for experiments and for the final dinner of the project mobility week
http://www.pappami.com/it/

Pappami! Il piatto che si mangia!

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

This project has received funding from the European Union's Erasmus + Programme for Education under KA2 grant 2014-1-IT02-KA201-003604. The European Commission support for the production of these didactical materials does not constitute an endorsement of the contents which reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

