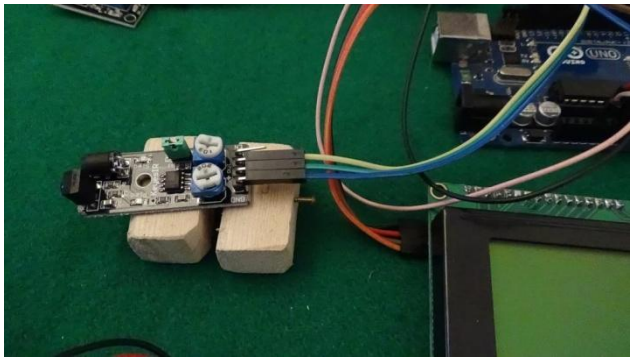
Z0- selettore di peso – weight selector (some notes at end of this section)

Questo impianto verifica il peso di merci che transitano su di un nastro trasportatore e se un pacco e' fuori dai limiti previsti, aziona un braccio meccanico che lo fa uscire dal nastro. E' derivato dal progetto "erogatore a tempo e peso" la cui scheda tecnica e' presente in questa raccolta.

Il sistema e' anche dotato di un modulo avoidance che conta i pezzi che passano sul nastro. In questa versione il sistema conta tutti i pezzi, sia quelli accettati che quelli scartati, ma non sarebbe un problema limitare la conta ai soli pezzi accettati. Il sistema e' in grado di verificare il peso di circa 60 pacchi al minuto. La velocita' e' fortemente influenzata dalla bilancia, che ha un suo tempo di latenza che non puo' essere compresso.

Sotto l'aspetto operativo Arduino, al momento dell'avvio, chiede il peso minimo ed il peso massimo accettabile, dopodiche' si mette in attesa di un pezzo da pesare e nel frattempo mantiene attivo il modulo avoidance per il conteggio dei pezzi che transitano sul nastro trasportatore.
Durante il funzionamento, premendo "A" e' possibile cambiare il peso minimo e massimo mentre la pressione del tasto "C" provoca l'azzeramento del contatore di pezzi



Per quanto riguarda gli aspetti costruttivi, alcune indicazioni riguardanti la costruzione della bilancia e l'uso della tastiera e del display a cristalli liquidi, sono presenti nelle schede:

- 41 bilancia digitale: http://giocarduino.altervista.org/e41-sensore-di-peso-bilancia-digitale.pdf
- 42 tastiera e display: http://giocarduino.altervista.org/e42-tastiera-numerica-e-display-2004.pdf

In particolare, Nella scheda 41 sono presenti sia il programma che le indicazioni per il calcolo del valore di scala, specifico di ogni sensore di peso. Questo valore, una volta calcolato, deve essere poi inserito nel programma prima della sua compilazione (vedi piu' sotto le note in rosso, nella sezione dedicata al programma). Una volta calcolato il valore di scala e' normalmente necessario procedere ad un lavoro di calibratura fine, sperimentando diversi valori (di poco maggiori o inferiori al valore calcolato) fino a quando la bilancia non arriva ad esporre, con precisione, il peso di un campione di riferimento.

Prima di procedere alla compilazione del programma e' necessario, se non gia' fatto, installare le seguenti librerie esterne:

- HX711, liberamente scaricabile da qui: https://github.com/bogde/HX711
- newliquidcrystal, scaricabile da qui: https://bitbucket.org/fmalpartida/new-liquidcrystal/downloads

Per installare una libreria esterna:

- scaricare la libreria

- lanciare l'IDE e andare in sketch->include library->add.zip library
- selezionare la libreria appena scaricata (dovrebbe essere c:/utenti/nome del tuo account/download) e premere "apri"
- verificare che la libreria sia presente nel repository delle librerie (sketch->include library) e quindi chiudere e riaprire l'IDE per rendere operativa la nuova libreria (l'apertura e la chiusura dell'IDE sono necessarie solo per chi sta utilizzando una vecchia IDE).

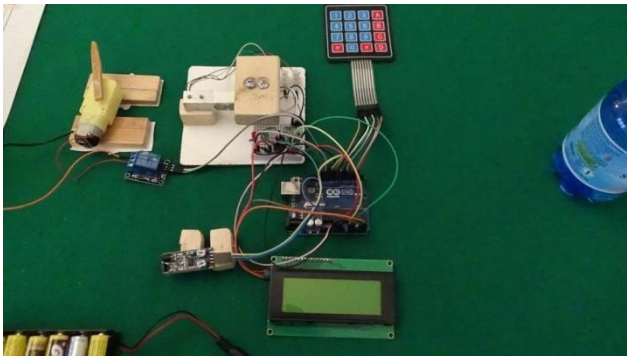
Sempre prima di procedere alla compilazione bisogna anche installare la libreria "keypad", presente ma non installata, nel library manager di Arduino (ide->sketch->include library->manage library->keypad).

Nota: Questo esercizio e questa nota sono parte di una serie che vede protagonisti Arduino ed alcuni dei componenti ad esso collegabili. Per la maggior parte degli esercizi e' anche disponibile un filmato su youtube.

- Esercizi facenti parte della raccolta
- Filmati presenti su youtube
- Informazioni su arduino e sui componenti collegabili (PDF scaricato nell'area di download)
- Breve manuale di programmazione (PDF scaricato nell'area di download)

Per eventuali chiarimenti o suggerimenti sul contenuto di questa scheda scrivere a giocarduino@libero.it

Here some notes about this project, translated by google translator



This system checks the weight of parcels and, if a parcel is outside the defined limits, drives a mechanical arm that makes it get out from tapis roulant. It is derived from the project "water dispenser" whose technical specifications are present in this collection. The system is also equipped with device (an avoidance device) for counts pieces on tapis roulant. In this version the system counts all pieces, both those accepted and those rejected, but is not a problem counting only to accepted pieces. The system can check about 60 parcels per minute. The speed is strongly influenced by the balance, that has a latency time that can not be compressed.

Under the operational aspect, at boot time, Arduino asks the minimum and maximum weight. Then waits for a piece to be weighed and meanwhile keeps the "avoidance" module active, for counting pieces on tapis roulant .
During operation, pressing button "A" you can change the minimum and maximum weight while pressing the button "C" causes the counter to zero pieces.

As for the construction aspects, some indications (in english too) concerning the digital scale construction and use of keyboard and LCD display, can be found in:

- 41 digital scale
- 42 keyboard and display

In project 41, you can find both the program and the signs for the scale value calculation, specific to each sensor. This value, once determined, must be inserted in program prior to its compilation (see under the red notes, in the program section). Once calculated the scale value, is normally necessary a fine calibration work, experimenting different values (slightly more or less than the calculated value) up to when the scale does not exhibit the weight of the reference sample

Before proceeding to program compilation must be installed, if not already done, the libraries:

- HX711, found here
- LiquidCrystal_I2C.h found here

For library installation, see process shown in previous projects, and summarized in:

- library download in compressed form;
- Installation via IDE-> sketch-> includes Library-> add .zip library
- After installation please verify the library. It must be present in IDE-> sketch-> includes Library-> Contributed library

Before compilation must also installed library "keypad", present but not installed, in the Arduino library manager (ide-> sketch-> includes Library-> manage Library-> keypad).

Note: This project and this note is part of a series that sees, as main characters, Arduino and some of connectable components. For most projects there is also a video on youtube.

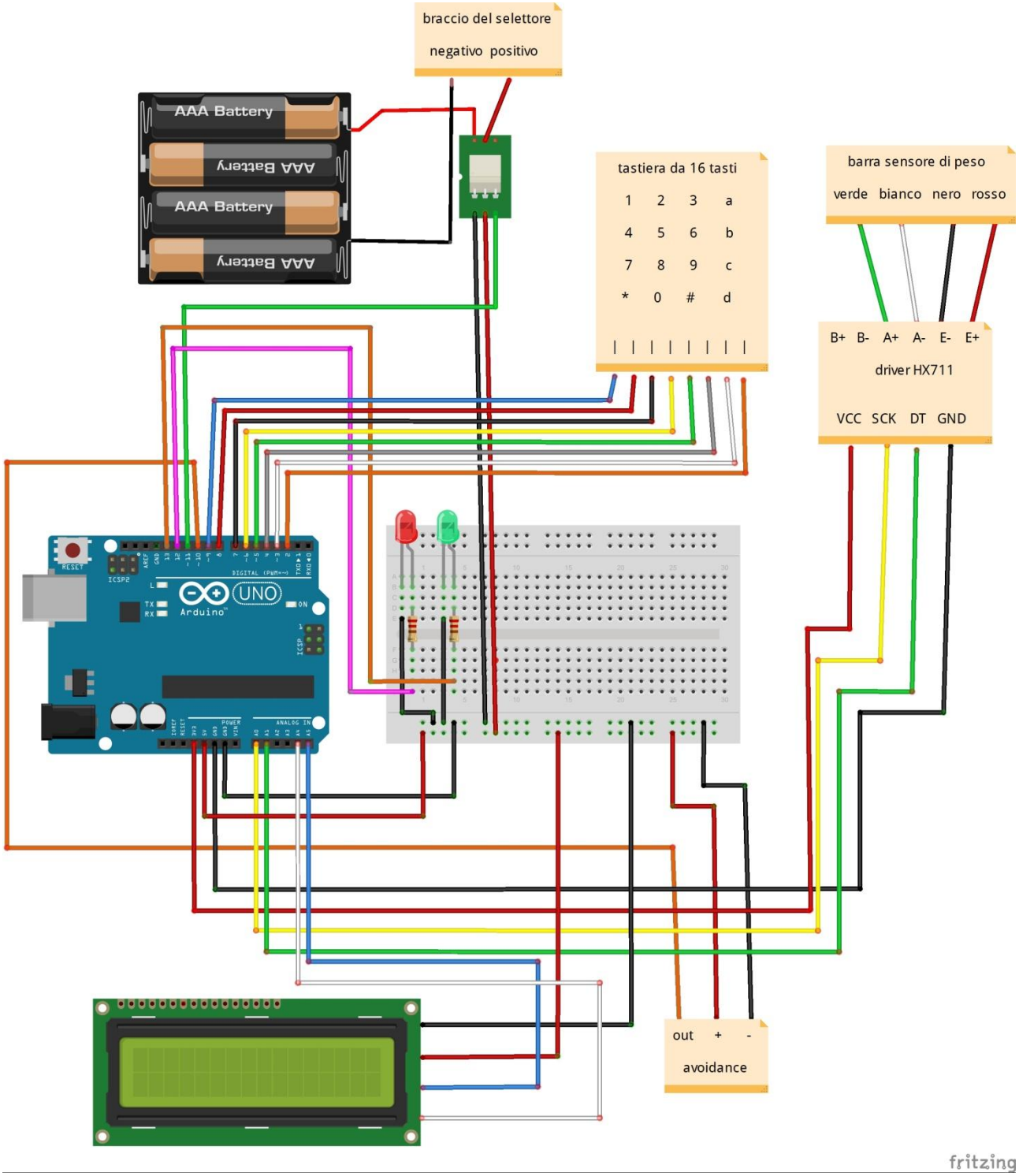
- Projects collection
- Movies on youtube
- About Arduino and components (italian; pdf will be downloaded in your download area
- Quick programming guide (almost english; pdf will be downloaded in your download area)

For any questions or suggestions about this note (and on its english translation), please write to giocarduino@libero.it (simple words and short sentences, please)

Materiali

- Una scheda Arduino Uno R3
- Un sensore di peso e relativo driver HX711
- Una tastiera da 16 tasti
- Un display 2004 (20 caratteri su quattro righe) con driver I2C
- Un led verde ed uno rosso
- Due resistenze da 220 ohm, da associare ai led
- Un rele' da 5 volt
- Un modulo avoidance, per il conteggio dei pezzi sul tapis roulant
- Un pacco batterie da 6 volt per alimentare la pompa
- Un pacco batterie da 9 volt per alimentare Arduino

Schema


braccio del selettore
negativo positivo
AAA Battery
AAA Battery
AAA Battery
AAA Battery
tastiera da 16 tasti
1 2 3 a
4 5 6 b
7 8 9 c
* 0 # d
barra sensore di peso
verde bianco nero rosso
B+ B- A+ A- E- E+
driver HX711
VCC SCK DT GND
UNO
Arduino
out + -
avoidance
fritzing

Programma

```
/* Attenzione: facendo il copia/incolla dal PDF all'IDE si perde la formattazione del testo. Per
* rendere piu' facilmente leggibile il programma e' opportuno formattarlo subito dopo il
* trasferimento nell'IDE, premendo CTRL+T.
*
* Utilizzo di un sensore di peso, di un tastiera, di un display 2004, di un rele' e di un
* modulo avoidance per un impianto di pesatura e conteggio
*
* Prima di compilare il programma scaricare ed installare, se non gia' fatto, le seguenti librerie:
* gestione del driver XH711: https://github.com/bogde/HX711
* driver I2C del display: https://bitbucket.org/fmalpartida/new-liquidcrystal/downloads
* libreria keypad.h, presente nel "library manager" dell'IDE (da installare tramite il library
* manager)
*
* connessioni del modulo H711:
*  Hx711.DO   - porta #A1
*  Hx711.SCK  - porta #A0
*
*  connessioni del sensore di peso:
*   rosso  = E+
*   nero   = E-
*   bianco = A-
*   verde  = A+
*
*  connessioni della tastiera:
*  porte da 2 a 9, in linea con le porte del connettore:
*  tenendo i pulsanti rivolti verso l'alto, collegare il primo connettore di destra
*  alla porta 2, il secondo alla 3 e cosi' via, fino all'ottavo (primo da sinistra)
*  da collegare alla porta 9
*
*  connessioni del display lcd
*  SCL = porta A5
*  SDA = porta A4
*
*  connessioni altri componenti:
*  led verde  = porta 13
*  led rosso  = porta 12
*  rele'      = porta 11
*  modulo avoidance = porta 10
*
* ---------------------------------------------------------------------------------------------------
* Warning: cut&paste from PDF to IDE loses formatting. to restore it press CTRL + T. This program
* uses a weight sensor, a keyboard, a display 2004 and a relay connected to a mechanical arm to
* check parcels weight and deviate the out of limits parcels
*
* Before program compiling must be downloaded and installed, if you have not done, the following
* libraries:
* - XH711 driver : https://github.com/bogde/HX711
* - I2C driver for lcd display: https://bitbucket.org/fmalpartida/new-liquidcrystal/downloads
* - Keypad.h library, present in the "library manager" of IDE (to be installed by the library
*   manager)
*
* Hx711connection:
*  Hx711.DO   - pin #A1
*  Hx711.SCK  - pin #A0
*
* load cell connection:
*   rosso  = E+
*   nero   = E-
*   bianco = A-
*   verde  = A+
*
* keypad connection:
* pin from 2 to 9, in line with the connector pins: holding the buttons facing up, connect the
* first right connector to pin 2, the second to 3 and so on, until the eighth (first from the left)
* to be connected to pin 9
*
* connection lcd display
*  SCL = pin A5
*  SDA = pin A4
*
* Other components:
*  green led  = pin 13
*  red led    = pin 12
*  rele'      = pin 11
*  avoidance  = pin 10
```

Arduino: selettore di peso – weight selector

```
*-----------------------------------------------------------------------------------------------------
*/
#include "HX711.h"
#include <Keypad.h>              // libreria di gestione della tastiera
#include <Wire.h>                // libreria wire presente, di default, nell'IDE
#include <LiquidCrystal_I2C.h>   // libreria di gestione del display lcd
#define DOUT  A1
#define CLK  A0
#define ledverde 13
#define ledrosso 12
#define rele 11
#define avoidance 10
HX711 bilancia(DOUT, CLK);
// nelle prossime due righe vengono attribuiti i parametri per la gestione del display
// . . . . . .  addr, en,rw,rs,d4,d5,d6,d7,bl,blpol
LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);
char tastobattuto;
const byte righe = 4;     // numero di righe della tastiera (quattro)
const byte colonne = 4;   // numero di colonne della tastiera (quattro)
char chiave[righe][colonne] =
{
  /* tabella di trascodifica dei tasti battuti.
   la tabella riproduce, su quattro righe di quattro colonne, la posizione dei tasti
   e assegna ad ogni tasto il valore che gli compete. E'possibile assegnare valori
   diversi  semplicemente modificando i singoli valori in tabella
   transcoding table of keystrokes. Table reproduces, on four rows of four columns, the location of
   keys and assigns to each a value.  You can assign different values simply by changing the
   individual values in table */
  {'1', '2', '3', 'a'},
  {'4', '5', '6', 'b'},
  {'7', '8', '9', 'c'},
  {'*', '0', '#', 'd'}
};
byte pinrighe[righe]     = {9, 8, 7, 6}; //porte cui devono essere connessi i pin delle
// righe (i primi quattro pin di sinistra) della tastiera
byte pincolonne[colonne] = {5, 4, 3, 2}; //porte cui devono essere connessi i pin delle
// colonne (gli ultimi quattro pin di destra) della tastiera

//prossima riga: assegnazione dei parametri alla libreria keypad.h
Keypad keypad = Keypad( makeKeymap(chiave), pinrighe, pincolonne, righe, colonne );

long valore = 0;          // valore immesso da tastiera
char tabnum [7];          // tbella di memorizzazione dei caratteri immessi
byte indtab  = 0;         // indice di scorrimento della tabella tabnum
byte semaforo = 0;        //semaforo per la permanenza nella routine di ricezione valori
byte indice = 0;          // indice utilizzato nel calcolo dei valori immessi
long moltiplicatore = 1; // moltiplicatore utilizzato nel calcolo del valore
int pesomin = 0;          // peso di max di funzionamento della pompa
int pesomax = 0;          // peso di riferimento
int peso     = 0;         // zona di memorizzazione del peso corrente
int pesoprec = 0;         // zona di memorizzazione dell'ultimo peso esposto
int contatore = 0;        // contatore di pezzi
int statocontatore = 0;   // zona di acquisizione dello stato del contatore
int semaforocontatore = 0; //semaforo utilizzato nella routine di conteggio
//
// *****************routine di verifica e conteggio dei pezzi passati **********************
// ********************************** count parcels ***********************************
//
void vediavoidance (void)
{
  statocontatore = digitalRead (avoidance);  // rileva lo stato del modulo
  if (statocontatore == HIGH)  // se non ci sono ostacoli
    semaforocontatore = 0;     // spegne il semaforo in modo da predisporsi alla conta del prossimo
// ostacolo
  else if (semaforocontatore == 0) // se c'e' un ostacolo non ancora conteggiato
  {
    contatore = contatore + 1;  // conteggia l'ostacolo
    lcd.setCursor (11, 2);
    lcd.print (contatore);
    semaforocontatore = 1;   // considera conteggiato l'ostacolo
  }
}
//
// ************************ routine di ripristino del display *********************
// ************************************** display refresh ****************************
//
void ripristinadisplay (void)
{
```

```
  lcd.clear ();
  lcd.print ("peso min: ");
  lcd.setCursor (11, 0);
  lcd.print (pesomin);
  lcd.setCursor (0, 1);
  lcd.print ("peso max: ");
  lcd.setCursor (11, 1);
  lcd.print (pesomax);
  lcd.setCursor (0, 2);
  lcd.print ("contatore: ");
  lcd.print (contatore);
  esponisituazione ();
}
//
// ******************* routine di acquisizione e verifica comandi **********************
// ********************* Acquire and ckeck commands from keypad *************************
//
void veditastiera (void)
{
  tastobattuto = keypad.getKey(); // acquisisce il valore del tasto battuto (gia' trascodificato)
  if (tastobattuto == 'a')        // modifica peso minimo minimo e massimo
  {
    lcd.clear();
    delay (300);                        // attende mezzo secondo)
    pesominmax ();                      // acquisisce i nuovi valori di peso minimo e massimo
  }
  if (tastobattuto == 'c')
  {
    lcd.clear ();
    delay (300);
    lcd.print ("azzera contatore?");
    lcd.setCursor (0, 1);
    lcd.print ("# = SI     * = NO");
    semaforo = 0;
    while (semaforo == 0)
    {
      tastobattuto = keypad.getKey(); // acquisisce il valore del tasto battuto (trascodificato)
      if (tastobattuto)
      {
        if (tastobattuto == '#')
        {
          contatore = 0;                    // azzera il contatore
        }
        semaforo = 1;                       // esce dalla routine di acquisizione comando
      }
    }
    ripristinadisplay ();                 // lancia la routine di ripristino del display
  }
}
//
// ******** routine di acquisizione del peso minimo e massimo*********
// ***************acquire minimum ad maximum weight *****************
//
void pesominmax (void)
{
  lcd.clear();
  lcd.print ("peso min:  ");
  ricevivalore ();
  pesomin = valore;
  lcd.setCursor (0, 1);
  lcd.print ("peso max:  ");
  ricevivalore ();
  pesomax = valore;
  ripristinadisplay ();
}
//
//*****routine di ricezione dei caratteri da tastiera e loro trasformazione in valori **
// ****** aquire numbers from keypad and calculate the total value *********************
//
void ricevivalore (void)
{
  valore = 0;
  semaforo = 0;
  for (indtab = 0; indtab <= 6; indtab++)
    tabnum[indtab] = 0;
  indtab = 0;
  while (semaforo == 0)     // loop di ricezione valori
  {
```

```
    tastobattuto = keypad.getKey(); // acquisisce il valore del tasto battuto (gia' trascodificato)
    if (tastobattuto)               // se  e' stato battuto un tasto
    {
      lcd.print (tastobattuto);    // visualizza il tasto battuto sul display lcd
      if ((tastobattuto >= '0') && (tastobattuto <= '9'))
      {
        tabnum [indtab] = tastobattuto - 48; // memorizza il valore battuto, trasformato in un
// numero da 0 a 9
        indtab++;
      }
      if ((tastobattuto == '#') || (indtab == 5))
      {
        // trasforma in un unico numero i singoli numeri digitati
        semaforo = 1;
        moltiplicatore = 1;
        for  (indice = indtab; indice > 1; indice--)
          moltiplicatore = moltiplicatore * 10;
        for (indice = 0; indice <= indtab; indice++)
        {
          valore = valore + tabnum[indice] * moltiplicatore;
          moltiplicatore = moltiplicatore / 10;
        }
      }
    }
  }
}
//
//  **** routine di esposizione della situazione corrente
//  **** show current status (weight and counter) *********
//
void esponisituazione (void)
{
  lcd.setCursor (0, 3);
  lcd.print ("                    "); // Attenzione: inserire 20 spazi tra gli apici  - warning:
// insert 20 space between " "
  lcd.setCursor (0, 3);
  lcd.print ("peso: ");
  lcd.print(peso);
  if ((peso >= pesomin) && (peso <= pesomax))  // se il peso e' interno ai limiti
  {
    digitalWrite (ledverde, HIGH);
    digitalWrite (ledrosso, LOW);
    digitalWrite (rele , LOW);
  }
  else
  {
    digitalWrite (ledverde, LOW);
    digitalWrite (ledrosso, HIGH);
    digitalWrite (rele , HIGH);
  }
}
//
//
void setup()
{
  Serial.begin (9600); // inizializza la comunicazione seriale
  lcd.begin(20, 4);    // inizializza il display (20 caratteri per 4 righe)
  lcd.backlight();     // illumina lo sfondo del display
  lcd.setCursor(0, 0); // posiziona il cursore all'inizio della prima riga (carattere 0 e riga 0)
  pinMode (ledverde, OUTPUT);
  pinMode (ledrosso, OUTPUT);
  pinMode (rele, OUTPUT);
  pinMode (avoidance, INPUT);
  lcd.print("    buongiorno");
  lcd.setCursor (0, 1);
  lcd.print("calcolo della tara");
  lcd.setCursor (0, 2);
  lcd.print("non porre alcunche' ");
  lcd.setCursor (0, 3);
  lcd.print("sulla bilancia.....");
  bilancia.set_scale(2045); // VALORE DI SCALA: inserire al posto di 2045 il valore calcolato
  // seguendo la procedura indicata nella scheda del progetto 41 -  SCALE VALUE: enter the
  // calculated value (using steps in project 41) in place of 2045
  bilancia.tare(20);        // il peso attuale e' considerato tara
  delay (1000);
  pesominmax();
}
//
```

Arduino: selettore di peso – weight selector

```
//
void loop()
{
  veditastiera();                //verifica se e' in arrivo un input da tastiera
  vediavoidance ();              // verifica se il contatore ha individuato un oggetto
  peso = bilancia.get_units(2), 0; // rileva il peso sul piatto
  esponisituazione ();  // espone la situazione corrente
}
```