Arduino Programming
Part 6: LCD Panel Output

EAS 199B, Winter 2010

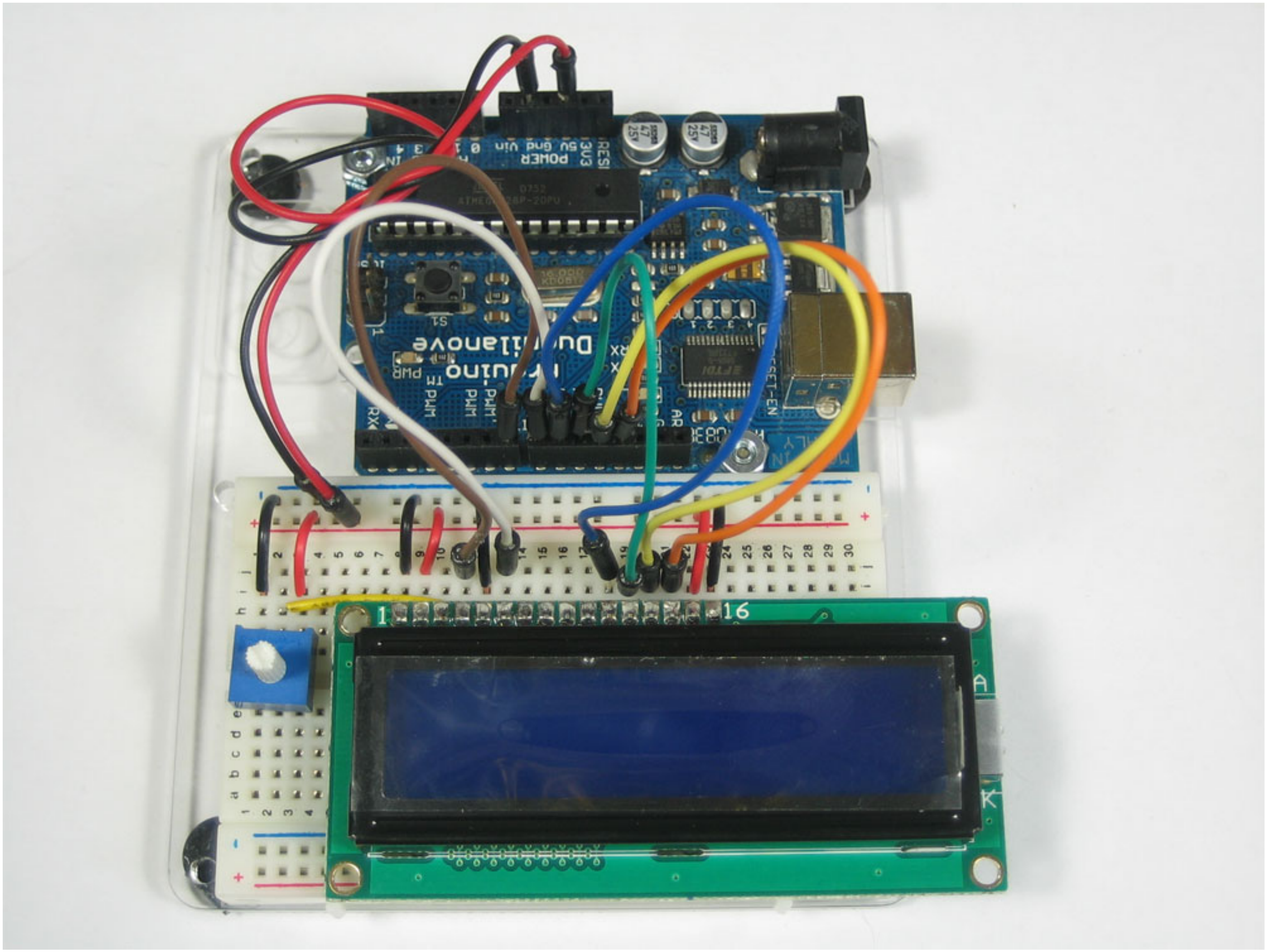
Gerald Recktenwald
Portland State University
gerry@me.pdx.edu

Goals

Use the 20x4 character LCD display for output

- Overview of assembly — detailed instructions on the web
  - http://web.cecs.pdx.edu/~gerry/class/EAS199B/howto/LCDwiring/
  - http://www.ladyada.net/learn/lcd/charlcd.html
- Introduction to the LCD library
  - http://www.arduino.cc/en/Tutorial/LiquidCrystal
- Simple demonstration
- Map the 20x4 character display for fish tank data

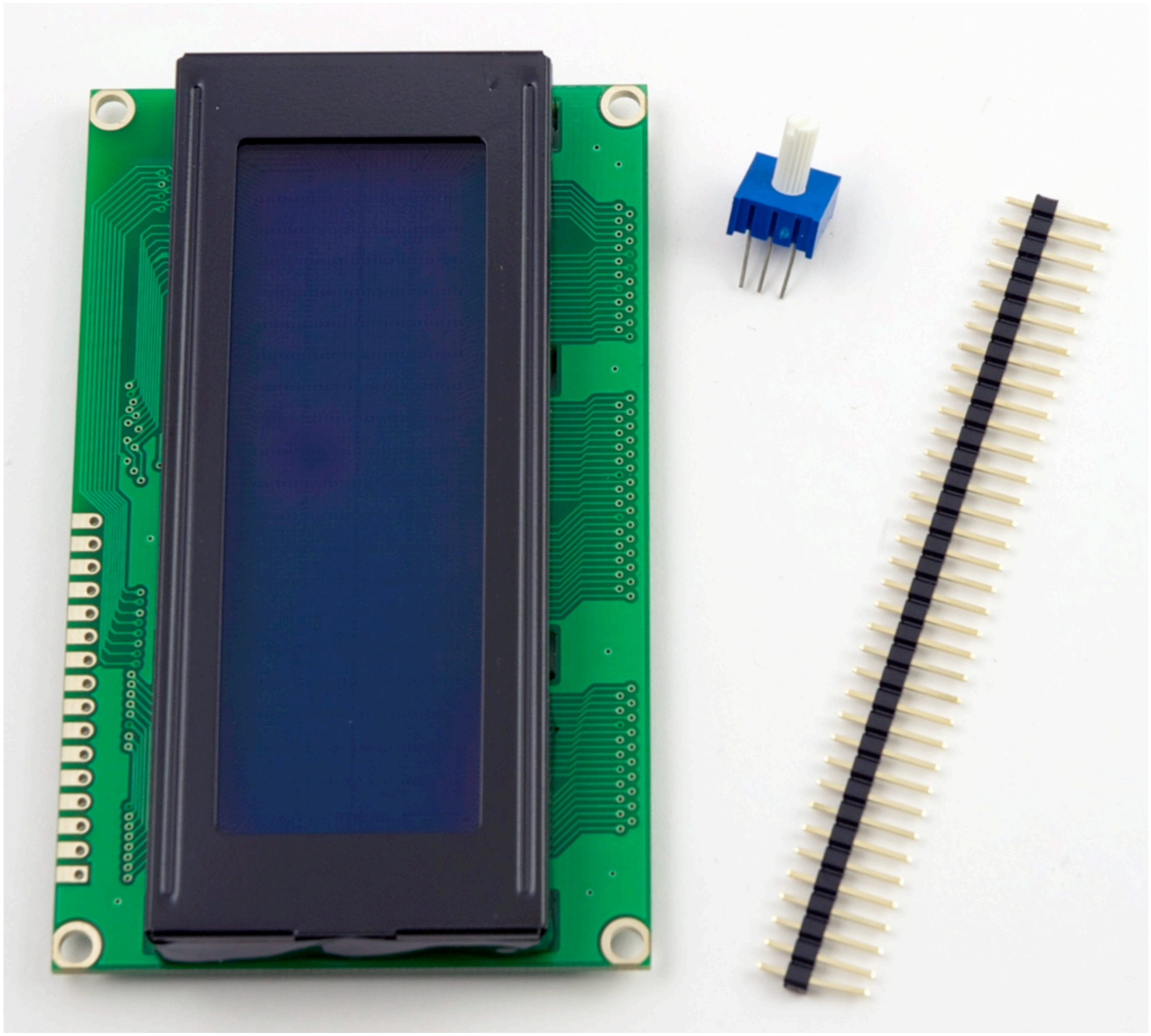
Breadboard connection via Adafruit Tutorial



http://www.ladyada.net/learn/lcd/charlcd.html

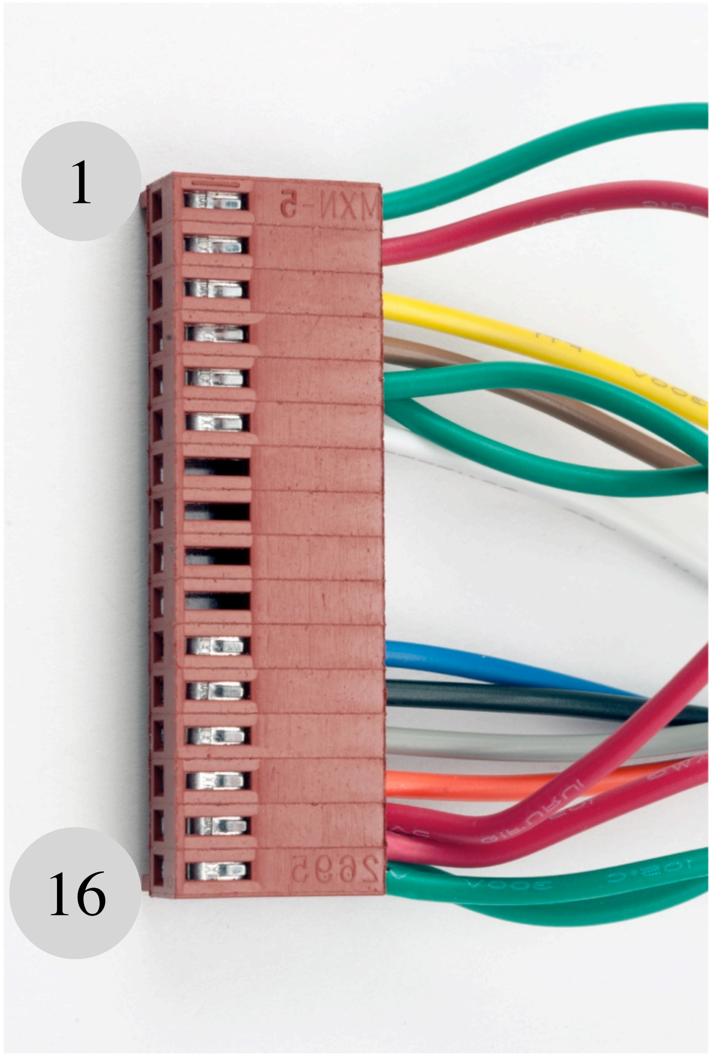
The Adafruit kit

1. Header for electrical connections
2. Potentiometer for contrast adjustment
3. Panel on PCB

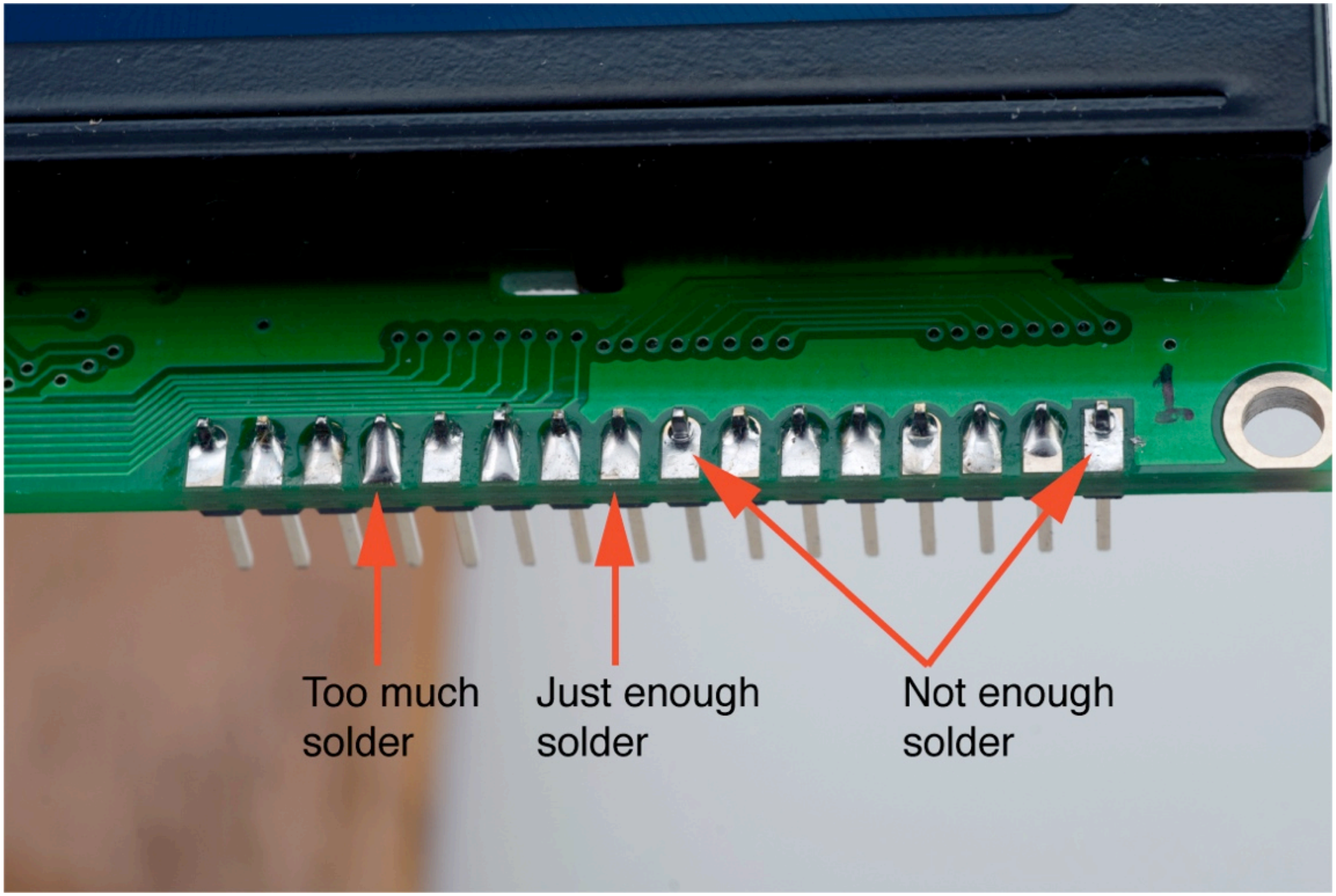


Wiring diagram

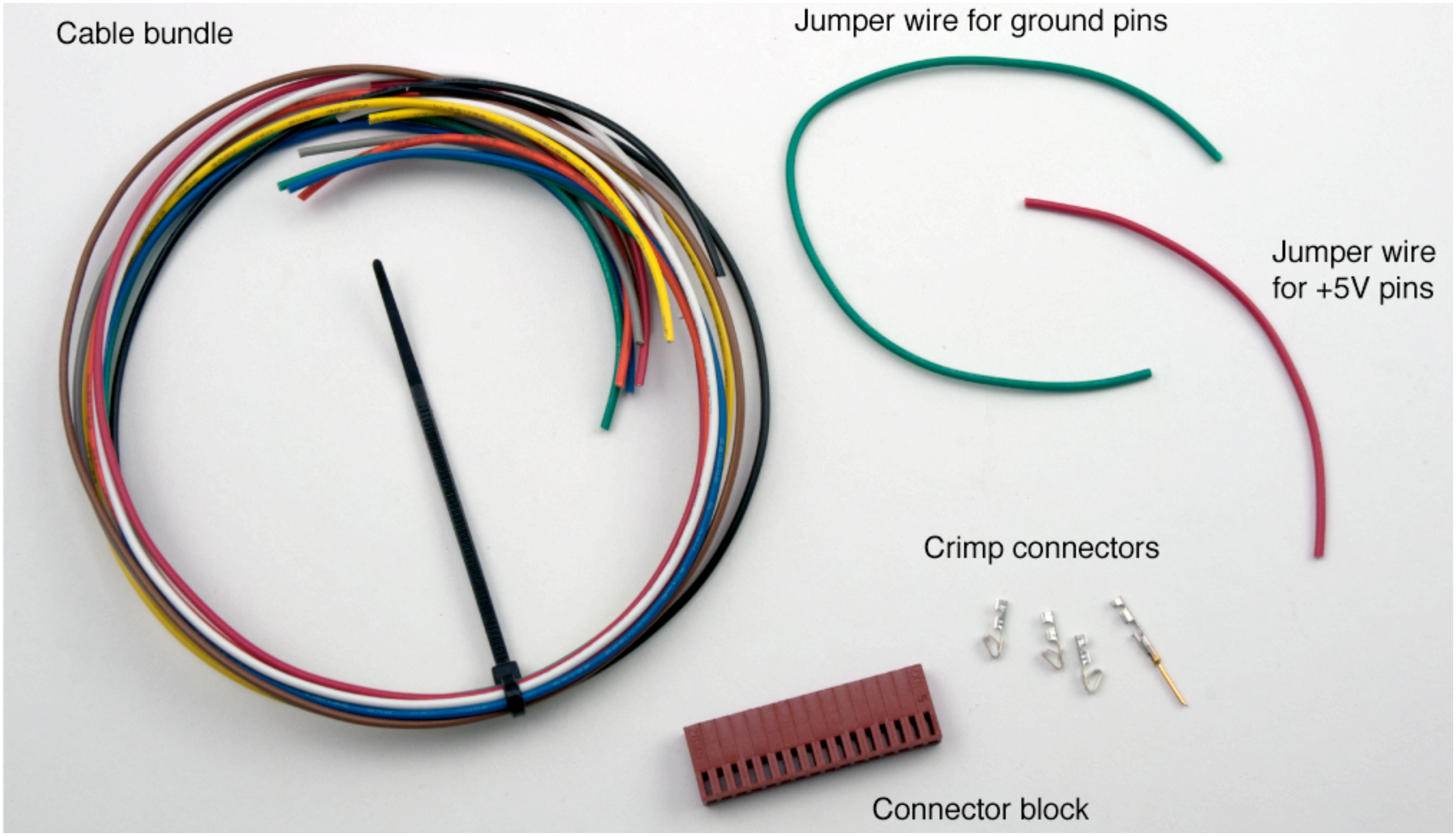
Connector			Arduino
1	Green	🟩	Ground
2	Red	🟥	+5V
3	Yellow	🟨	Wiper of 10k potentiometer
4	Brown	🟫	pin 8
5	Green	🟩	Ground
6	White	⬜	pin 9
7			
8			
9			
10			
11	Blue	🟦	pin 10
12	Black	⬛	pin 11
13	Gray	▨	pin 12
14	Orange	🟧	pin 13
15	Red	🟥	+5V
16	Green	🟩	Ground



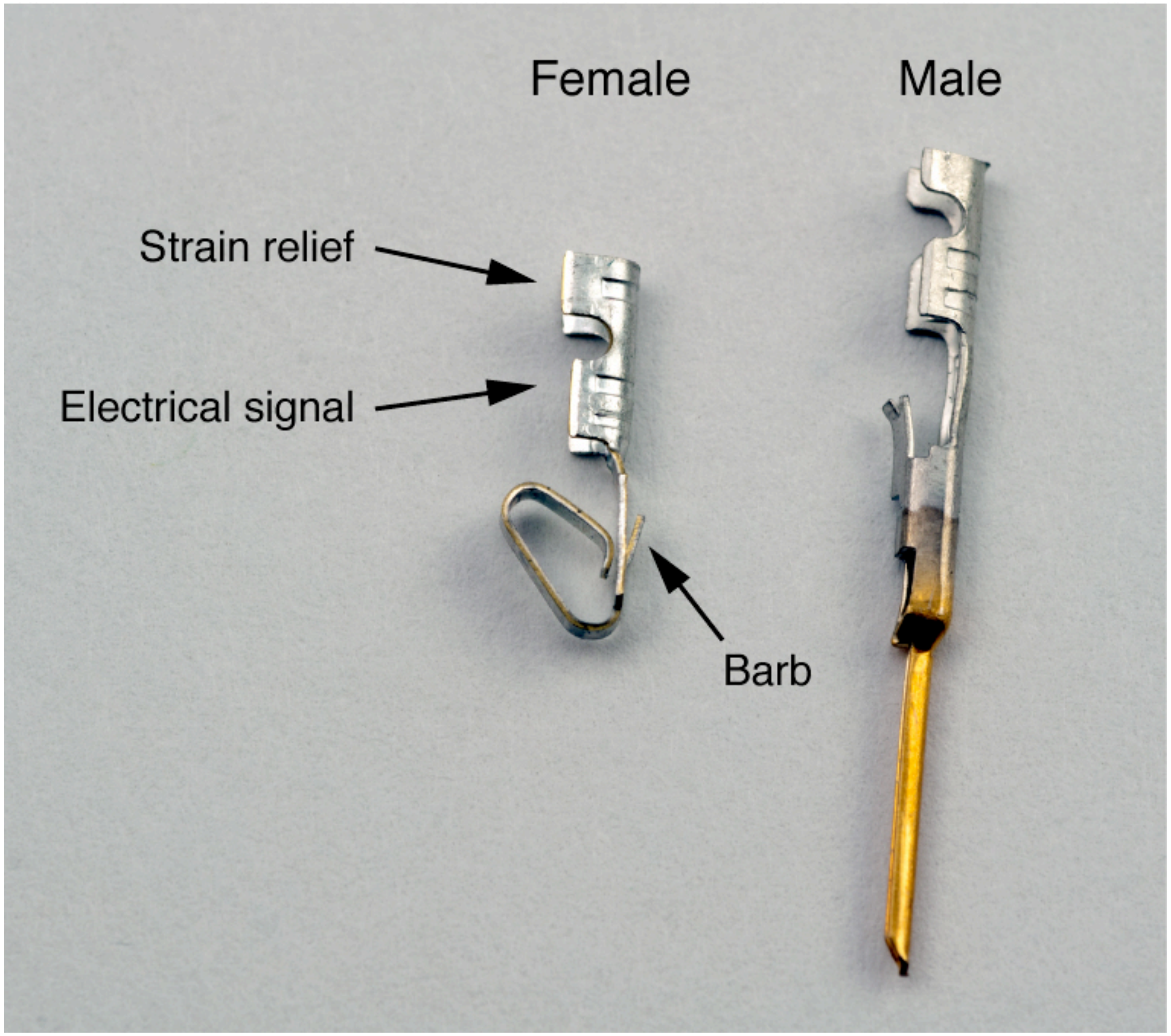
Step 1: Solder the header



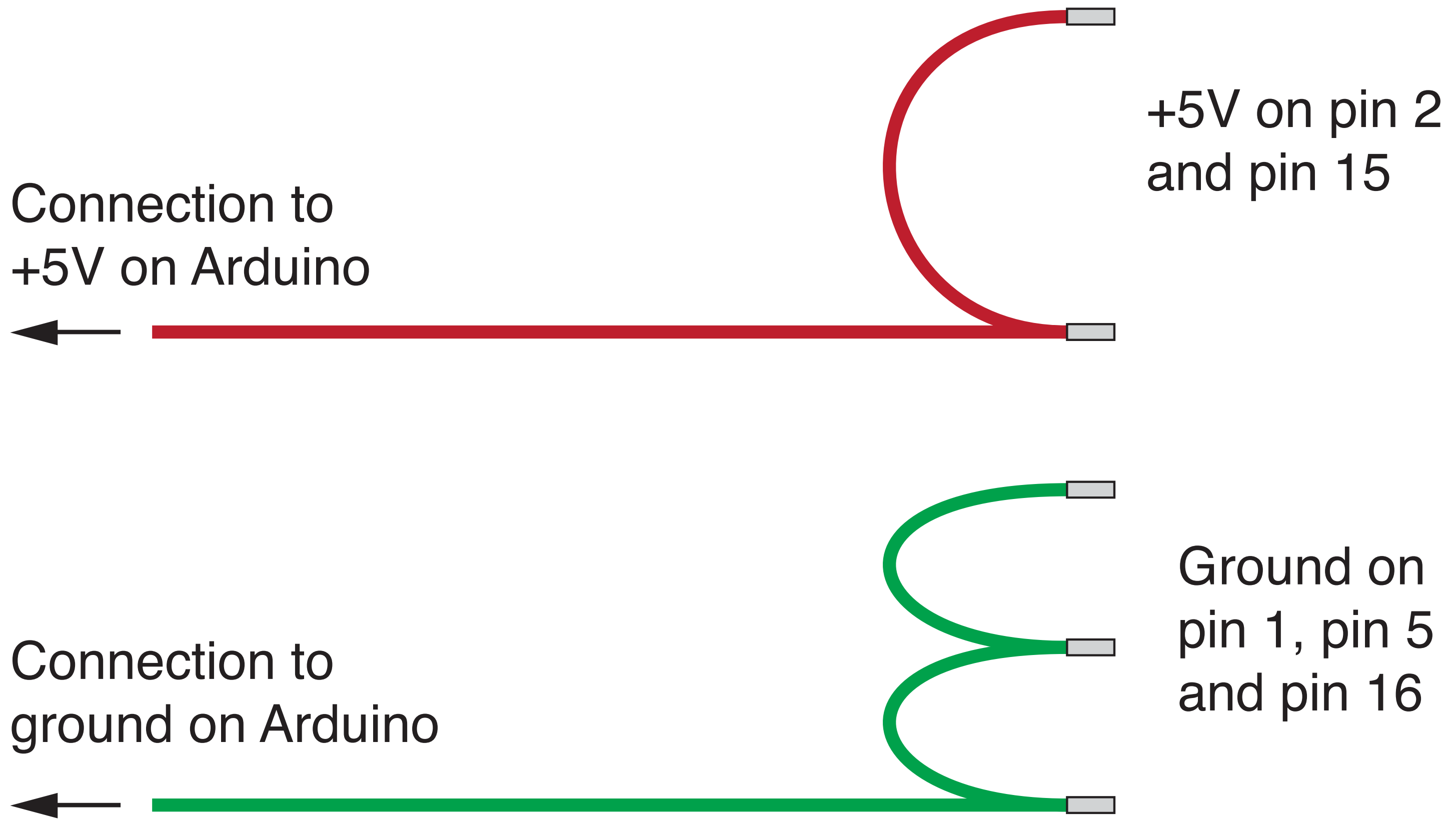
Step 2: Assemble the wiring harness



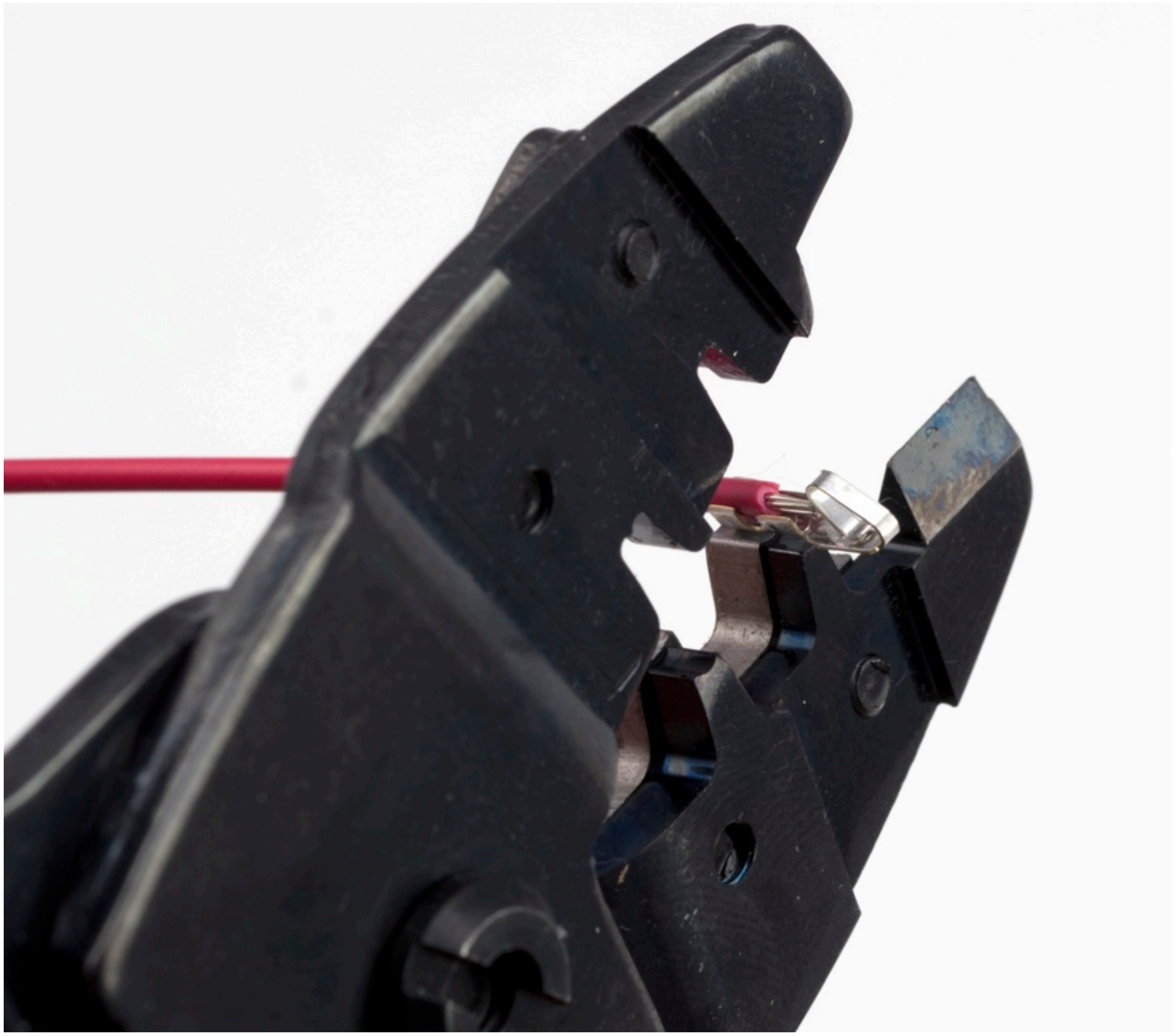
Crimp Connectors: they are small!



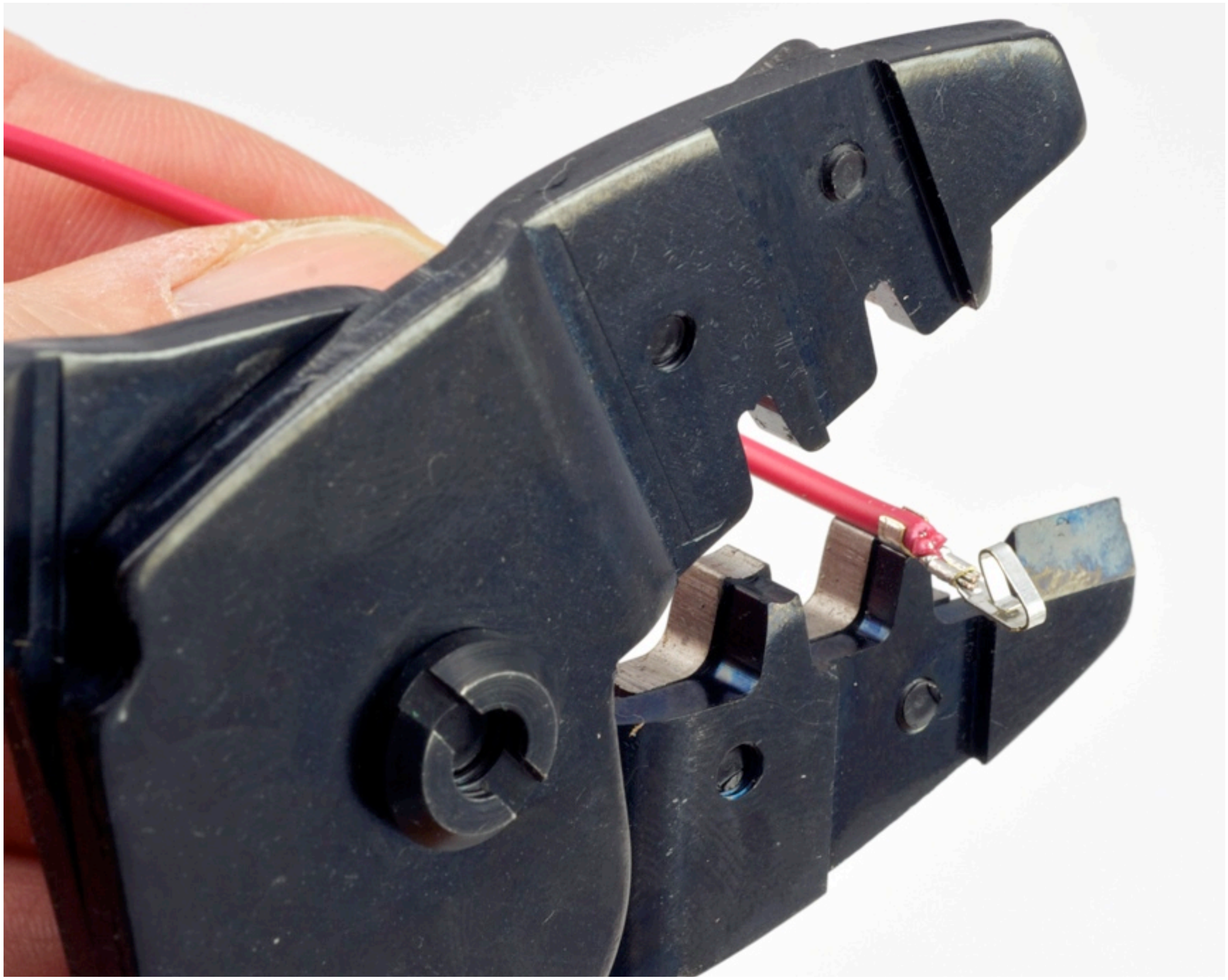
Use jumpers to avoid unnecessary wire


+5V on pin 2 and pin 15
Connection to +5V on Arduino
Ground on pin 1, pin 5 and pin 16
Connection to ground on Arduino

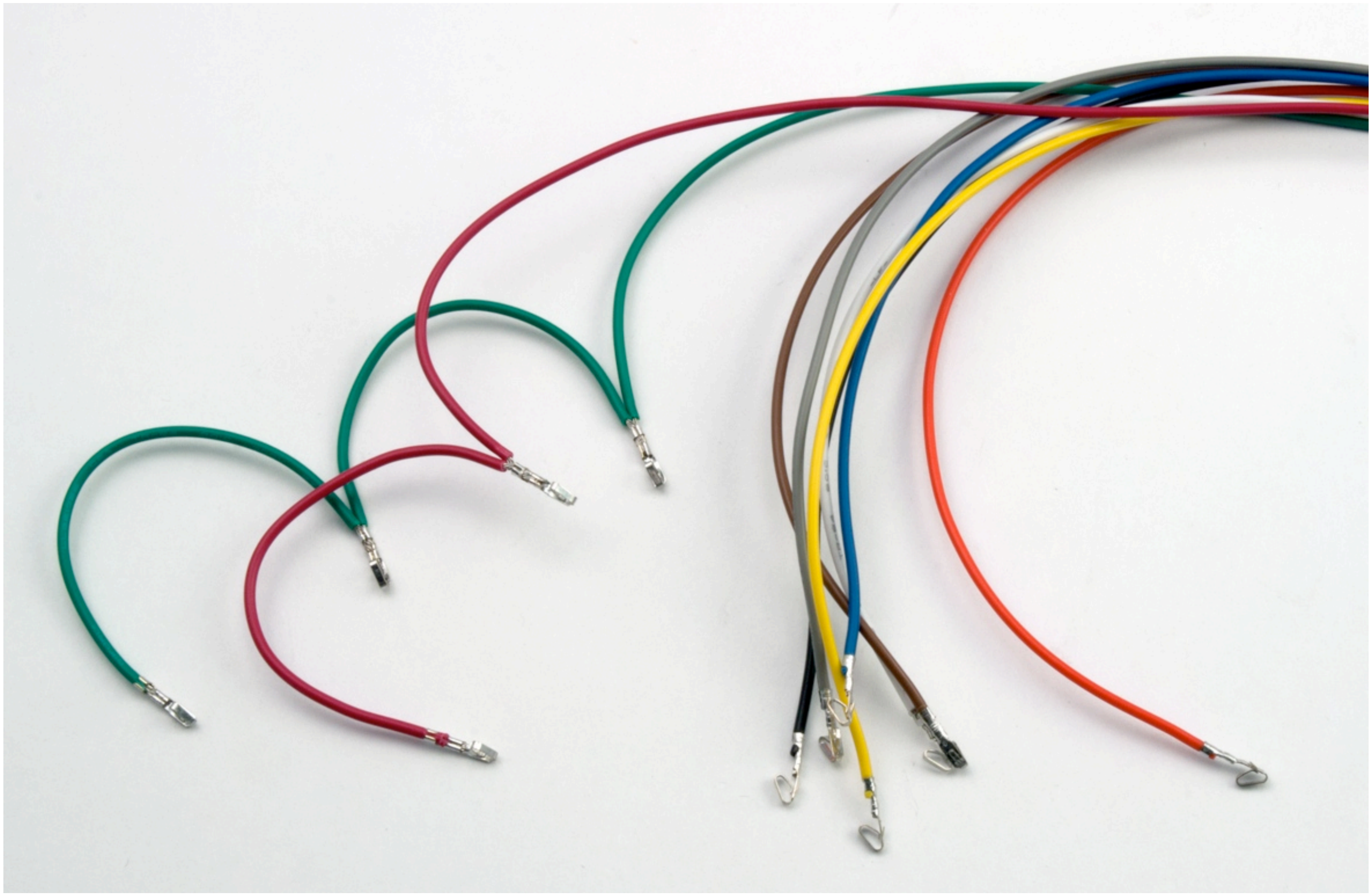
Locate the crimp connector in the tool



Crimp the strain relief

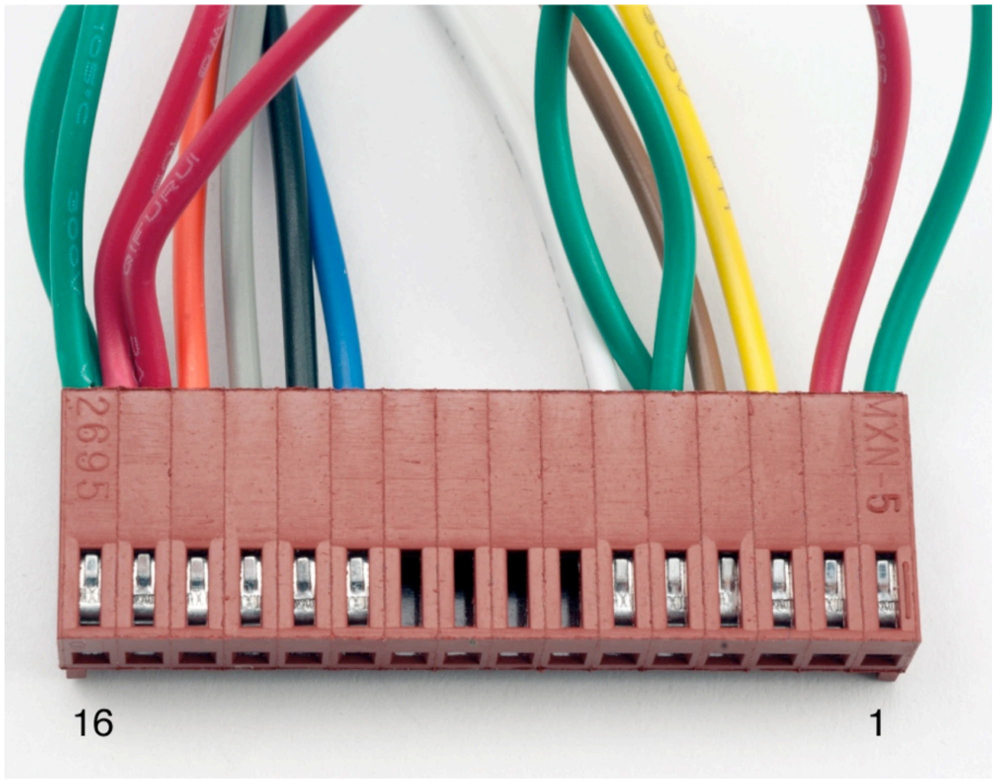


Finished crimping for the female connector

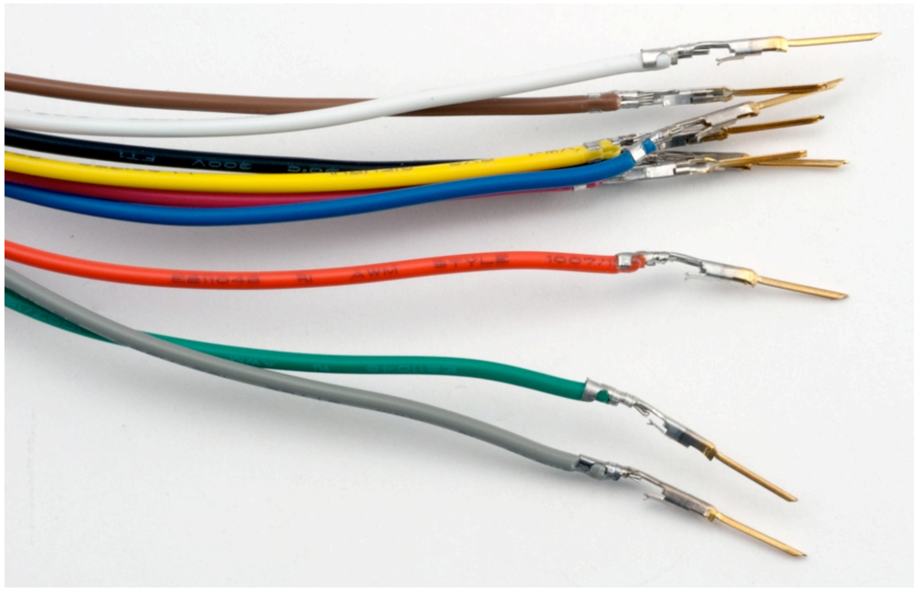


Finished female and male connectors

Female connector for LCD end



Male pins for Arduino end



Note: These male pins still need heat shrink to insulate pins from each other when they are inserted into a breadboard.

Programming Arduino for LCD Display

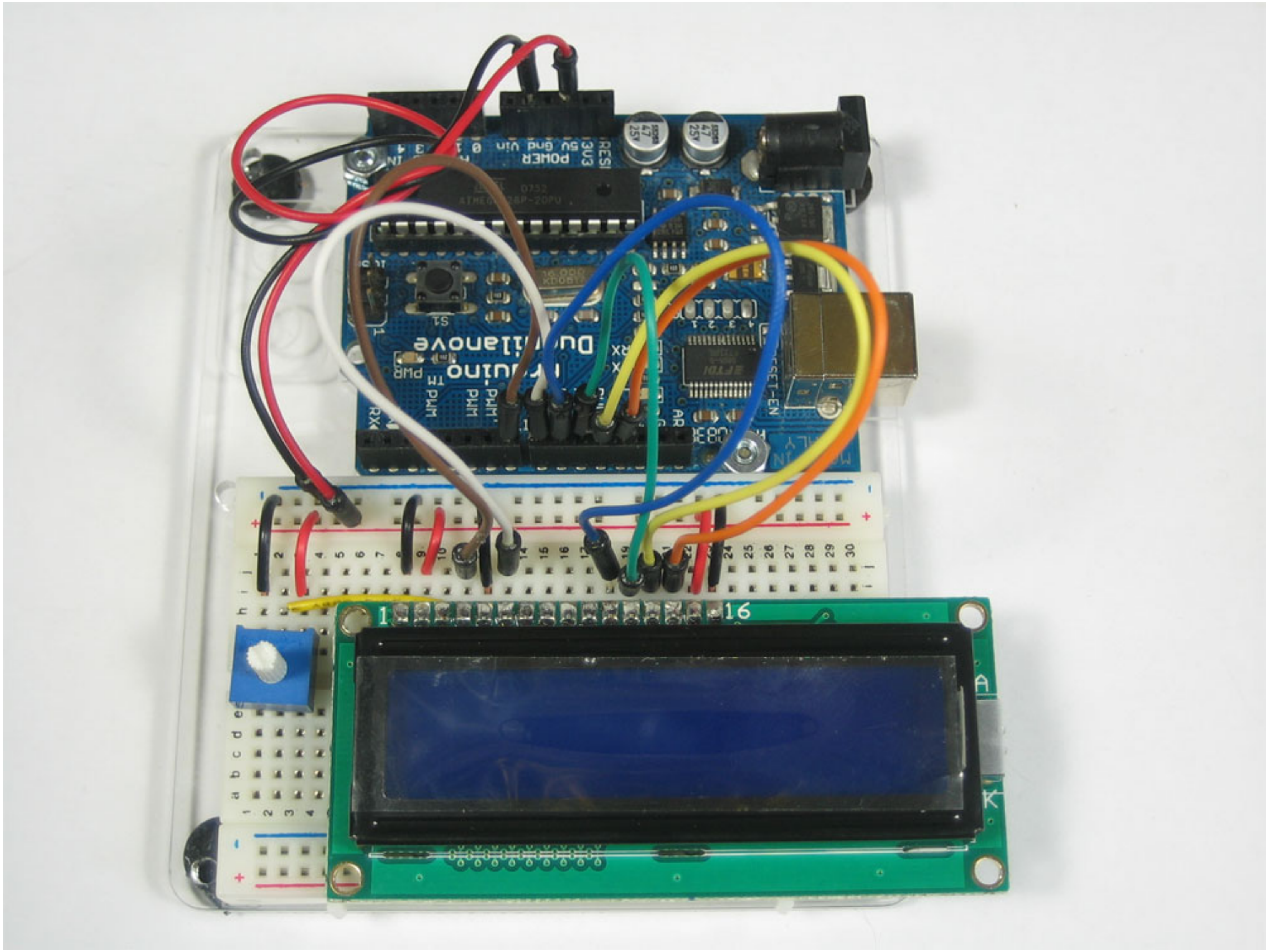
Refer to Adafruit tutorial

- http://www.ladyada.net/learn/lcd/charlcd.html

and Arduino documentation

- http://www.arduino.cc/en/Tutorial/LiquidCrystal

Breadboard connection via Adafruit Tutorial



http://www.ladyada.net/learn/lcd/charlcd.html

Test the display

File ⟹ Examples ⟹ LiquidCrystal ⟹ HelloWorld

```
// include the library code:
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.print("hello, world!");
}

void loop() {
  // set the cursor to column 0, line 1
  // Line 1 is the second row, because counting begins with 0
  lcd.setCursor(0, 1);
  // print the number of seconds since reset:
  lcd.print(millis()/1000);
}
```

Test the display

Change pin assignments to match wiring harness: (8,9,10,11,12,13)

```
// include the library code:
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.print("hello, world!");
}

void loop() {
  // set the cursor to column 0, line 1
  // Line 1 is the second row, because counting begins with 0
  lcd.setCursor(0, 1);
  // print the number of seconds since reset:
  lcd.print(millis()/1000);
}
```

Change to (20,4)

File ⟹ Examples ⟹ LiquidCrystal ⟹ HelloWorld

Test the display

```
// include the library code:
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.print("hello, world!");
}

void loop() {
  // set the cursor to column 0, line 1
  // Line 1 is the second row, because counting begins with 0
  lcd.setCursor(0, 1);
  // print the number of seconds since reset:
  lcd.print(millis()/1000);
}
```

lcd is a LiquidCrystal object

File ⟹ Examples ⟹ LiquidCrystal ⟹ HelloWorld

Arduino code to write to the LCD panel

Include the LCD library

In the header:
(outside and before setup)

```
#include <LiquidCrystal.h>
```

Initialize the display by creating a LiquidCrystal object

Before using the display:

```
LiquidCrystal lcd(p1,p2,p3,p4,p5,p6);
lcd.begin(20,4);
```

Send characters in a two-step process

Move the cursor:

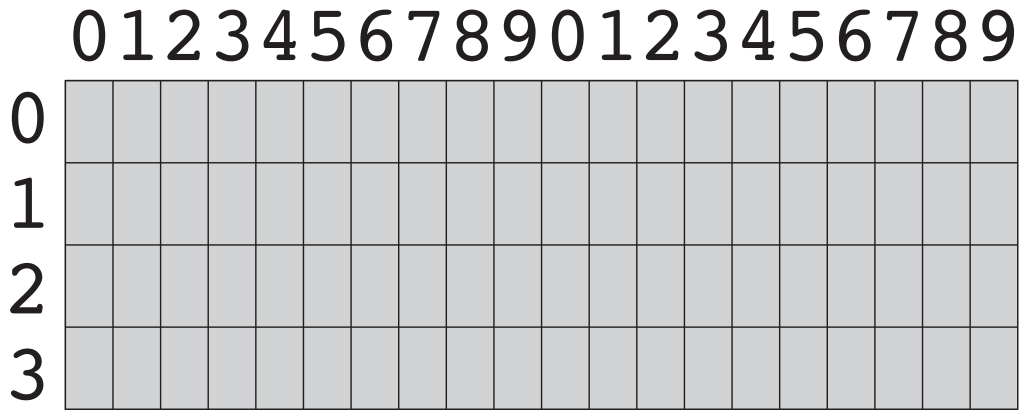
```
lcd.setCursor(column,row)
```

Display the message:

```
lcd. print(“message”)
```

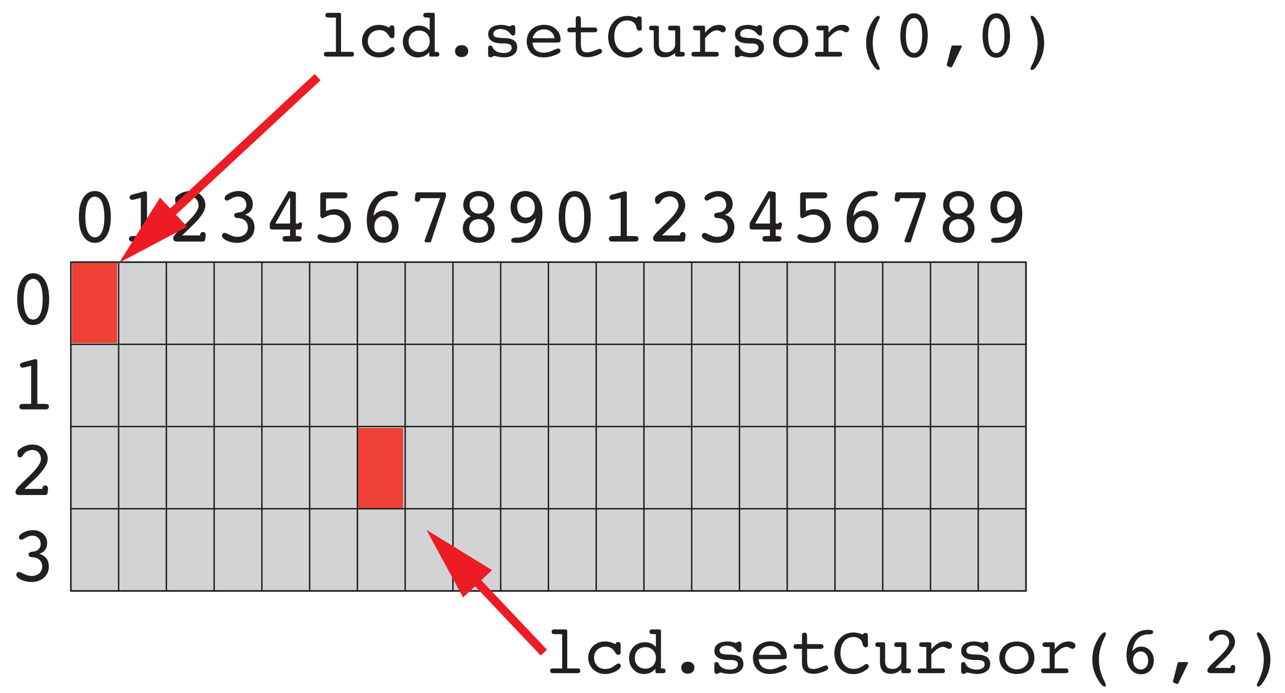
Character matrix on a 4 X 20 display

Row and column indices begin with zero



Character matrix on a 4 X 20 display

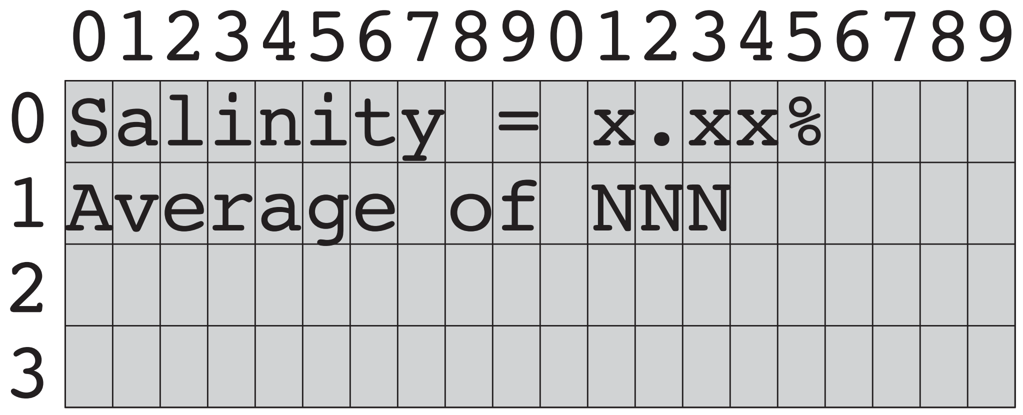
Row and column indices begin with zero



Display fish tank salinity

Modify the HelloWorld code to display the salinity

- “Salinity = ” and “Average of ” can be displayed once at the start
- x.xx and NNN values change, and are updated on the display.



Programming Paradigms

To think about styles of programming, we can organize programming languages into paradigms

Paradigm	**Representative Languages**
Procedural or Sequential	Fortran, C, Basic
Object-oriented	C++, smalltalk
Parallel /Concurrent	occam, erlang
Dataflow	LabVIEW
Functional	Haskel, Lisp
Scripting	perl, python

Note that many modern program languages have features of more than one paradigm

Object-Oriented Programming (OOP)

As you might expect, *Objects* are central to OOP

- Objects have data
- Objects have methods (like functions)
- Objects can be assembled into other objects.

Arduino Programming

- Uses the object-oriented language C++
- Don't get carried away with the OOP on Arduino
  - Keep your Arduino programs from becoming too complex
  - Basic structure of code, with setup() and loop() is sequential
- Libraries for the Serial Monitor and LCD output use OOP
  - Know enough OOP to use existing libraries
  - OOP can be handy when programming with new types of sensors

OOP in the LCD library code

Create a new LiquidCrystal object:

```
LiquidCrystal lcd(p1,p2,p3,p4,p5,p6);
```

Type of object

Name of the new object

Data passed to the object constructor

When a new object is created, the data passed to the constructor is *stored in* the object. Thus, whenever we use the variable `lcd` again in the program, the `lcd` object "knows" that it is connected to `p1, p2, ..., p6`.

OOP in the LCD library code

Tell the lcd object about the size of the display

```
lcd.begin(20,4)
```

Run the "begin" method

Pass the values 20 and 4 to the "begin" method

Objects have data and methods

- Data are values associated with a particular "instance" of an object
- Some data may be "public". Programmers can view or change public data.
- Some data may be "private", and therefore unavailable to programmers.
- Methods are functions that an object knows how to perform
    - Methods can return values
    - Methods can change public data
    - Methods can perform computations and interact with the environment (sensors)

OOP in the LCD library code

Change the current cursor position:

```
lcd.setCursor(12,1)
```

Run the “setCursor” method

Pass 12 and 1 to the “setCursor” method

The setCursor methods prepares lcd for its next action

```
lcd.print(“Hello”)
```

Run the “print” method

Use “Hello” as data for the print method

lcd.print(...) works because the lcd object “knows” about its current position (from setCursor), the size of the display (from begin), and from the pin assignments from the constructor. When the lcd.print() method runs, it unleashes action that is constrained by data stored in the object.