A group of soccer players in white jerseys with red accents are celebrating on a field. They are holding a large gold trophy in the center. Many players have gold medals around their necks and their arms are raised in the air. The background is filled with falling gold confetti.

Programming for Engineers

Winter 2015

Andreas Zeller, Saarland University

A Computer

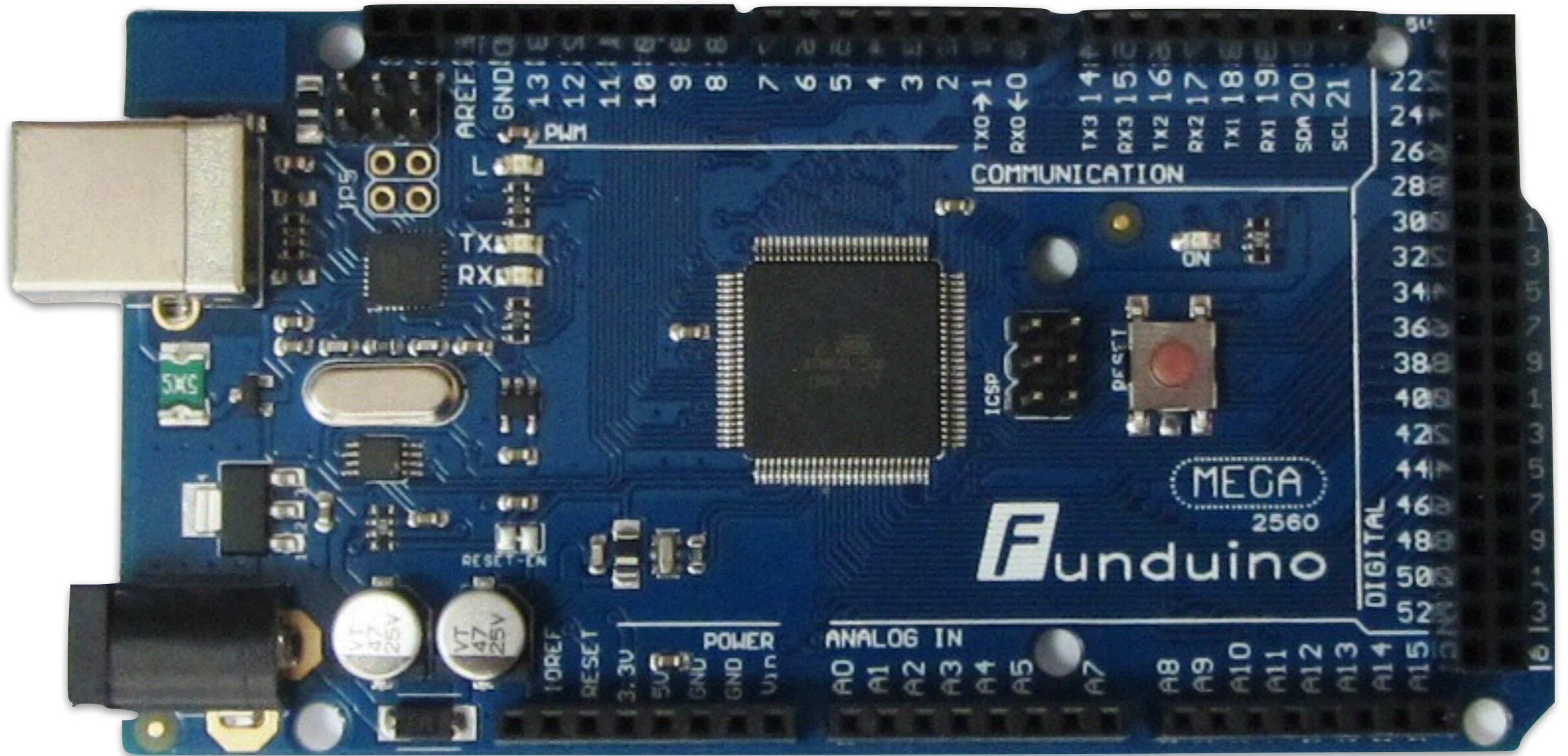


- Device that *processes data* according to an *algorithm*.

Computers are everywhere



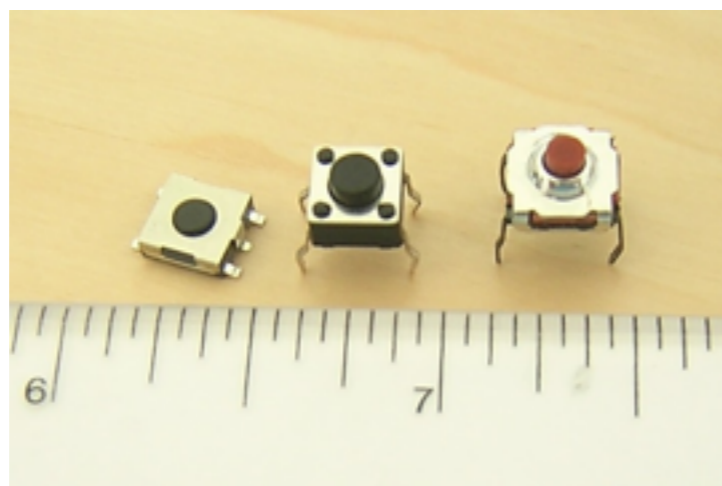
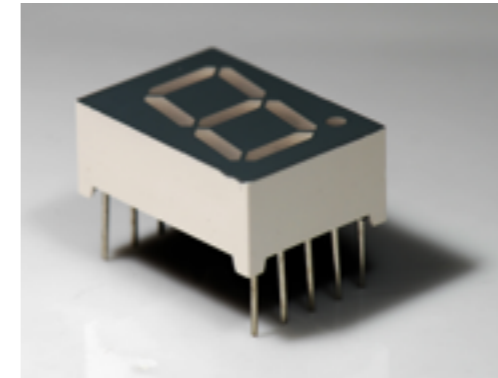
Your Computer



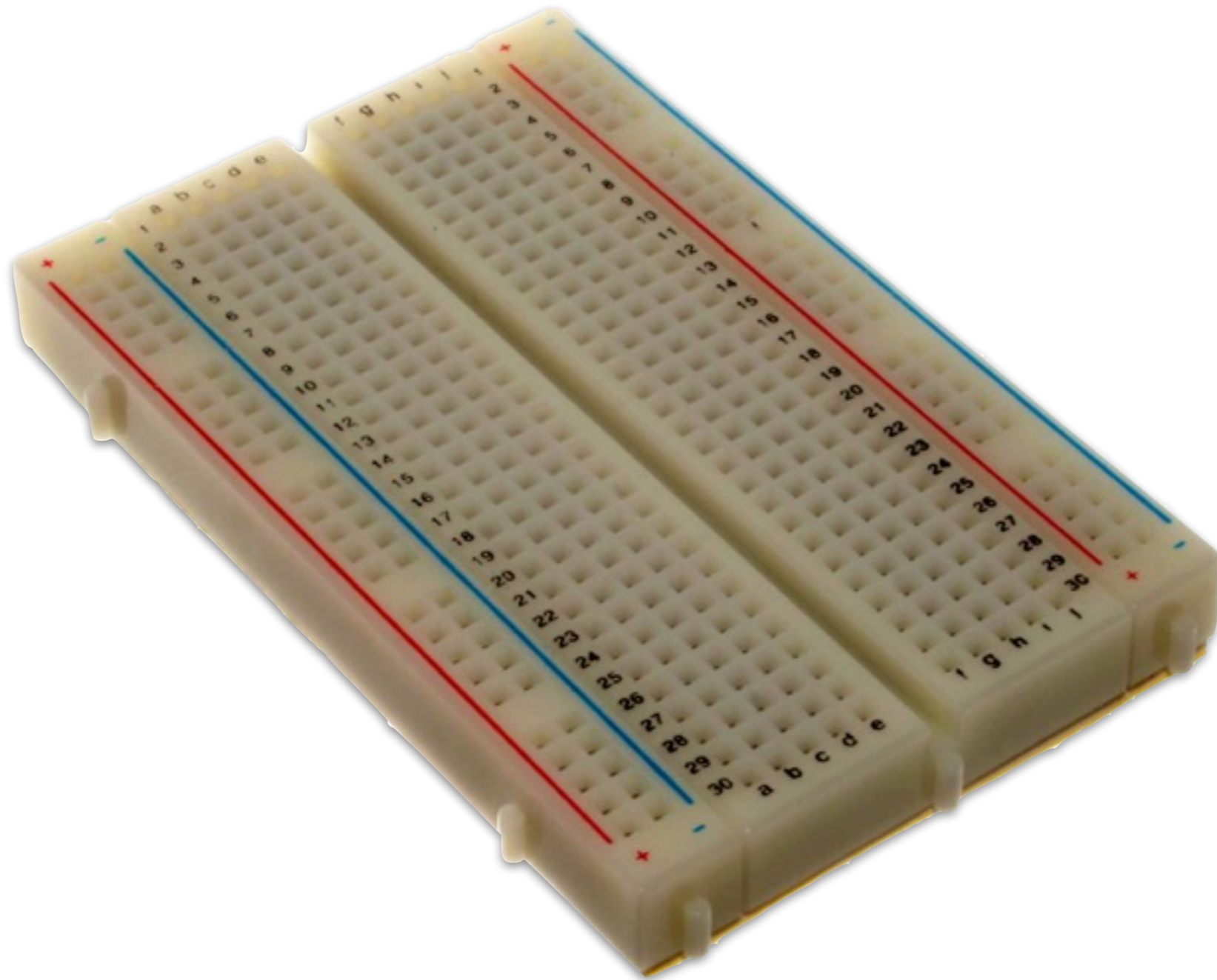
Arduino

- *Physical-Computing-Platform* for creating interactive, physical systems that connect hardware and software
- *Microcontroller* (processor) with *analog* and *digital* inputs and outputs
- *Development environment* on PC

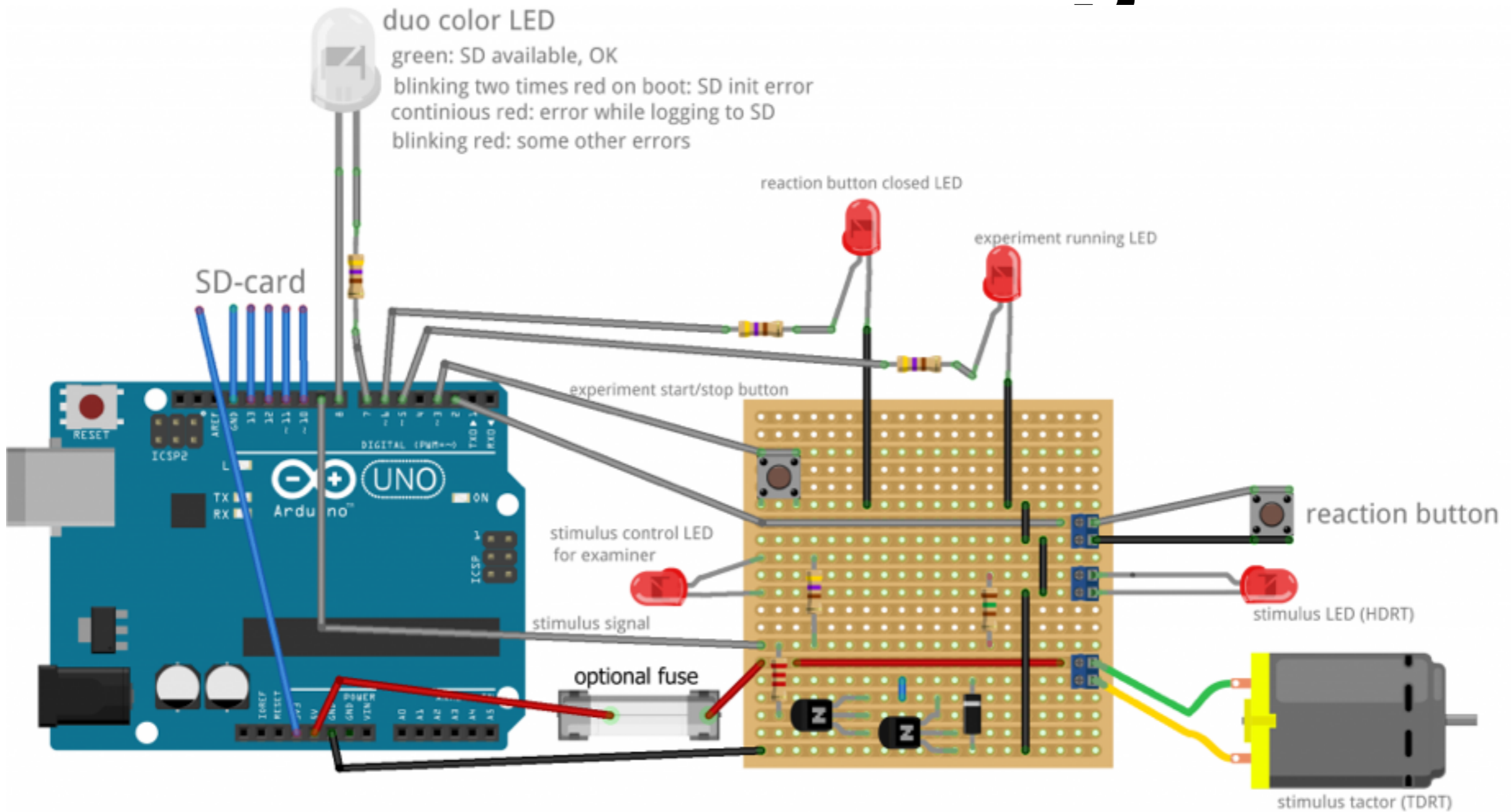
Input/Output



Breadboard



Plugging and Connecting



Programming

```
Blink | Arduino 1.5.3
Blink
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.
  */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);

  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);             // wait for a second
  digitalWrite(led, LOW);  // turn the LED off by making the voltage LOW
  delay(1000);             // wait for a second
}
```

Speichern abgeschlossen.

Transfer complete

#

#

Goals

- Fundamentals of programming
- Handling of input/output devices
- Programming own controllers

Structure

- *Lecture on programming*
- *Exercise on Arduino-Board*
- Free project
- Exam at the end of the course

Lecture

– topics on programming –

- Arduino overview
- Basics
(commands, control, main loop)
- Functions with parameters
- Control structures
- Arrays

Lecture

- Characters
- Input/Output
- Algorithms
- Graphs
- Data structures
- Testing + Debugging

Exercises

– Projects –

- Blinkenlights
- Morse-Code
- Traffic light
- Nim the game
- Tic-Tac-Toe

Exercises

- More Sensors
- Process measurements
- Navigation
- Webserver
- Internet of Things

Work



Team



Success in a Team



Exercises

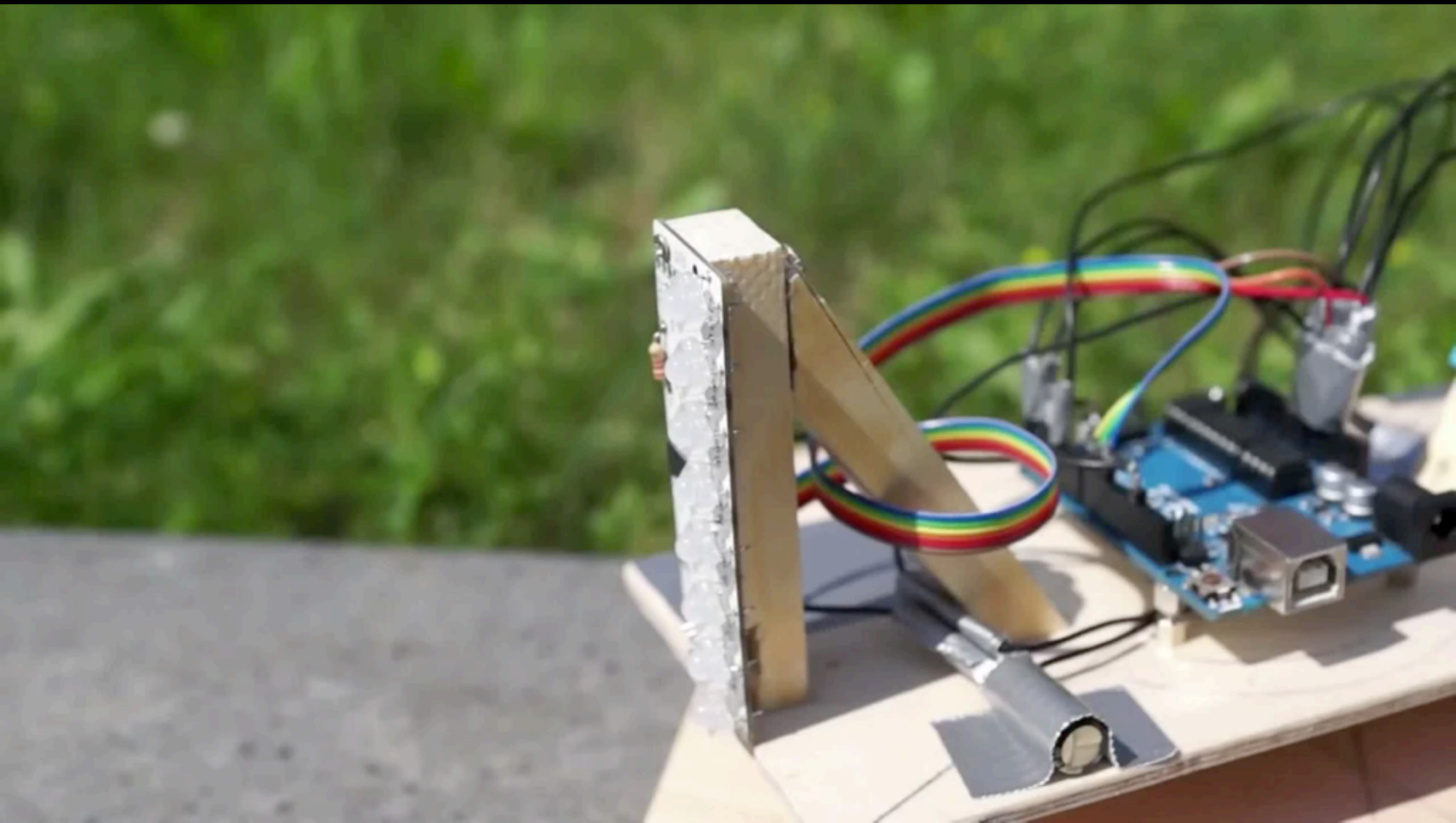
1. Programming individually
(to be submitted to tutor)
2. Executing and testing on own board
3. Demonstration and individual
explanation
to tutor

Free Project

- You come up with an Arduino project (with hardware and software) in a group
- You implement the project
- Grading according to
 - ★ Originality
 - ★ Complexity
 - ★ Functionality

Light Clock

Murat Güner, Maximilian Junk, Pierre Kehl and Thomas Kreis



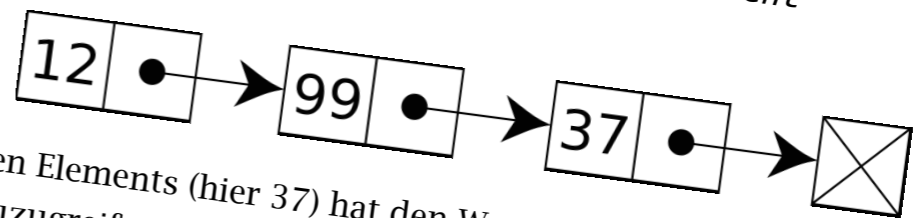
Exam

- Checks your programming skills
- At the end of the course
- In English

3 Datenstrukturen [15 Punkte]

Eine *verkettete Liste* ist eine dynamische Datenstruktur, die eine Speicherung von miteinander Beziehung stehenden Objekten erlaubt. Die Anzahl der Objekte ist im Vorhinein nicht bestimmt. Die Liste wird durch *Zeiger* auf das jeweils folgende Element realisiert. Das folgende Bild zeigt eine Liste, bestehend aus drei Elementen. Jedes Element ist definiert als

```
struct Elem {  
    int value; // Der Wert  
    struct Elem *next; // Zeiger auf das nächste Element  
};
```



Der Zeiger des letzten Elements (hier 37) hat den Wert NULL. Um auf eine Liste zuzugreifen, fängt man beim ersten Element (hier 12) an, und folgt dann den Zeigern auf das jeweils nächste Element. Die folgende Funktion prüft, ob ein Element mit dem Wert x in der Liste e enthalten ist. Wenn ja, gibt sie einen Zeiger auf das Element zurück; wenn nicht, gibt sie NULL zurück.

```
// Erstes Element der Liste LIST mit Wert X zurückgeben  
// (oder NULL, wenn nicht gefunden)  
struct Elem *search(struct Elem *list, int x) {  
    struct Elem *e = list; // Erstes Element  
    while (e != NULL && e->value != x)  
        e = e->next;  
    return e;  
}
```

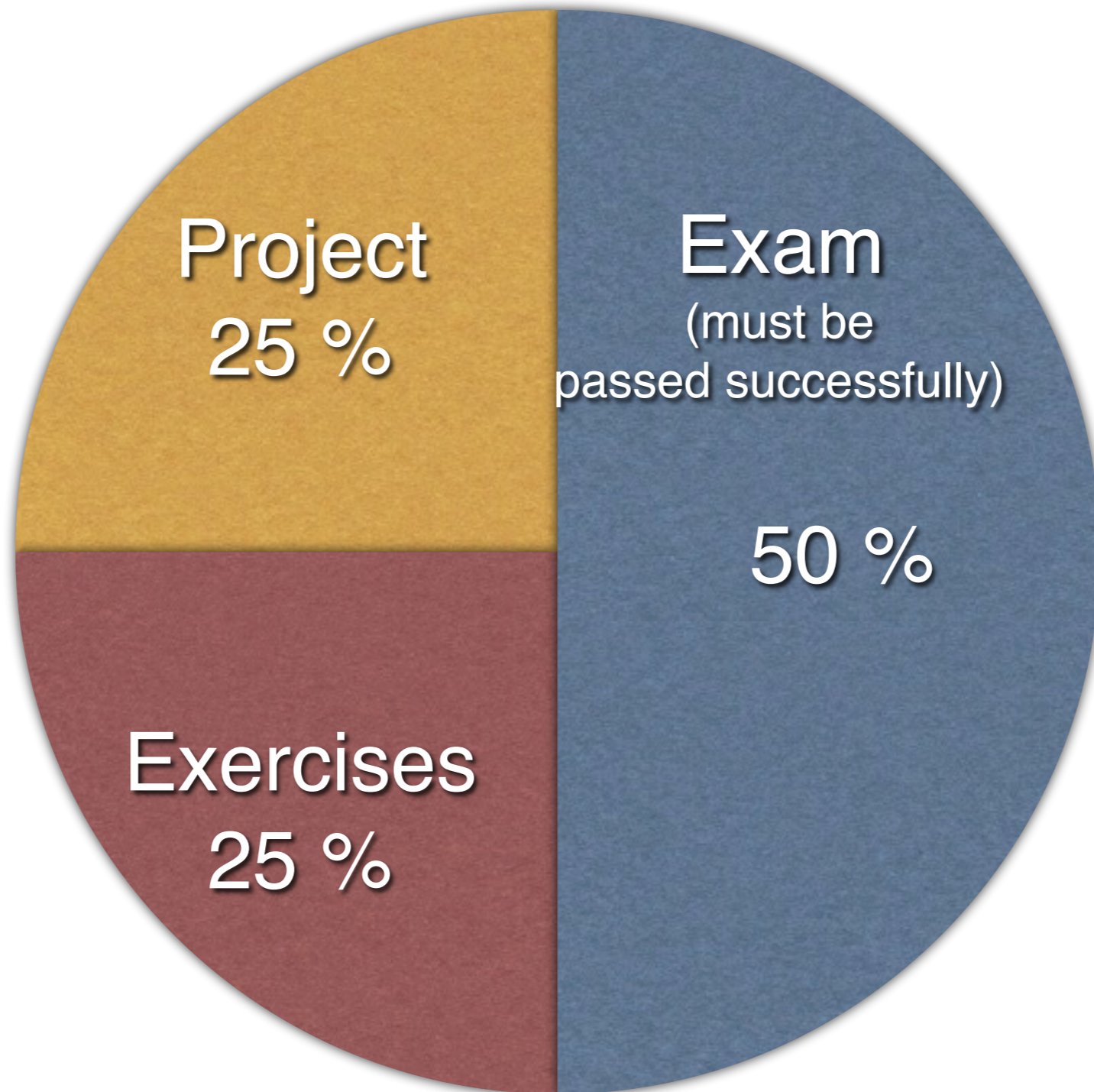
Ihre Aufgabe ist es, eine Funktion zu schreiben, die ein gegebenes Element e als letztes Element an eine nicht-leere Liste $list$ anhängt.

- [3 Punkte] Nehmen wir an, Sie möchten ein Element mit Wert 44 an die Liste anhängen. Zeichnen Sie (ähnlich zu obigem Diagramm), wie die Liste nach dem Anhängen aussieht.
- [8 Punkte] Um ein Element anzuhängen, müssen Sie zunächst das letzte Element finden. Schreiben Sie eine Funktion `last()`, die (ähnlich wie `search()` oben) durch die Liste geht, und das letzte Element liefert.

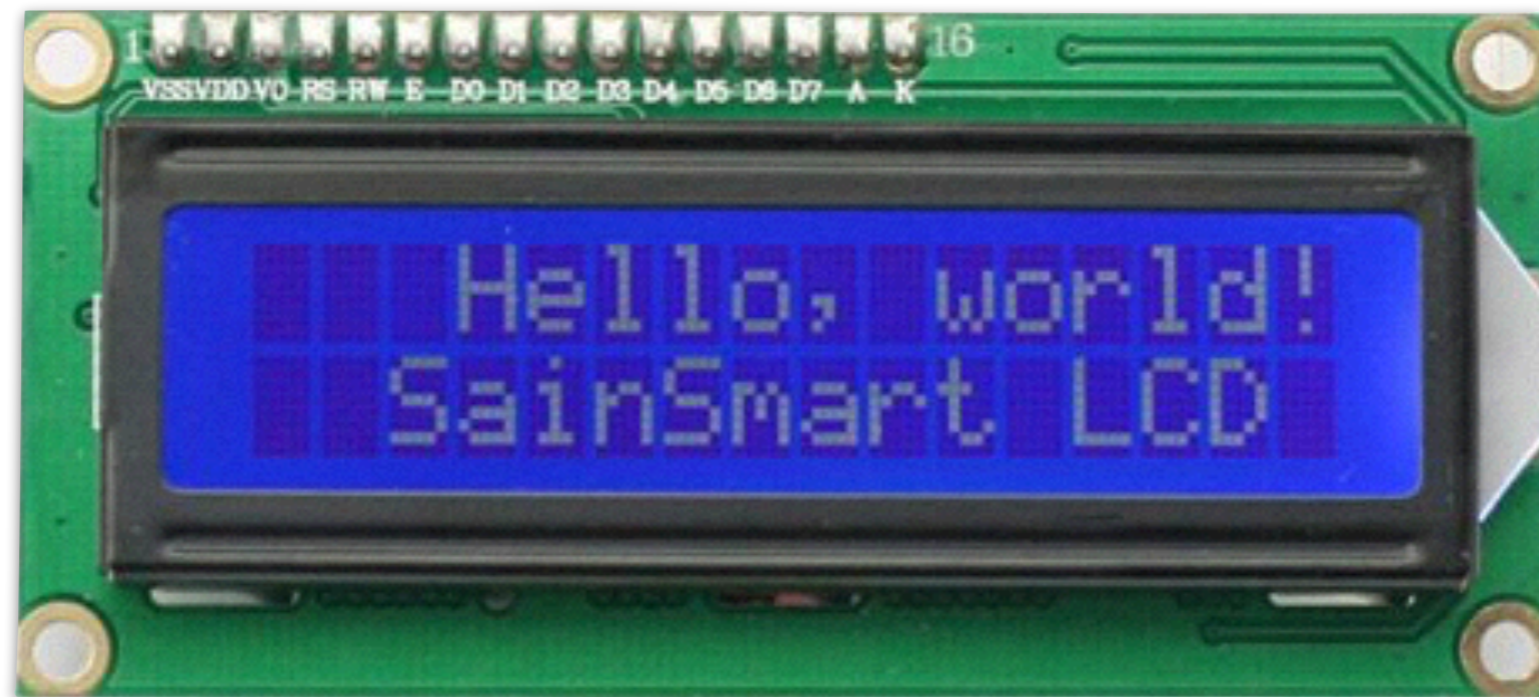
```
// Letztes Element der Liste LIST zurückgeben  
struct Elem *last(struct Elem *list) {  
    // Ihr Code hier  
}
```

- [4 Punkte] Gegeben sei nun ein existierendes Element E . Nutzen Sie `append()` um an das letzte Listenelement das Element E anzuhängen. Schreiben Sie eine Funktion `append()` entsprechend, die das Element E an die Liste anhängt.

Overall Grading

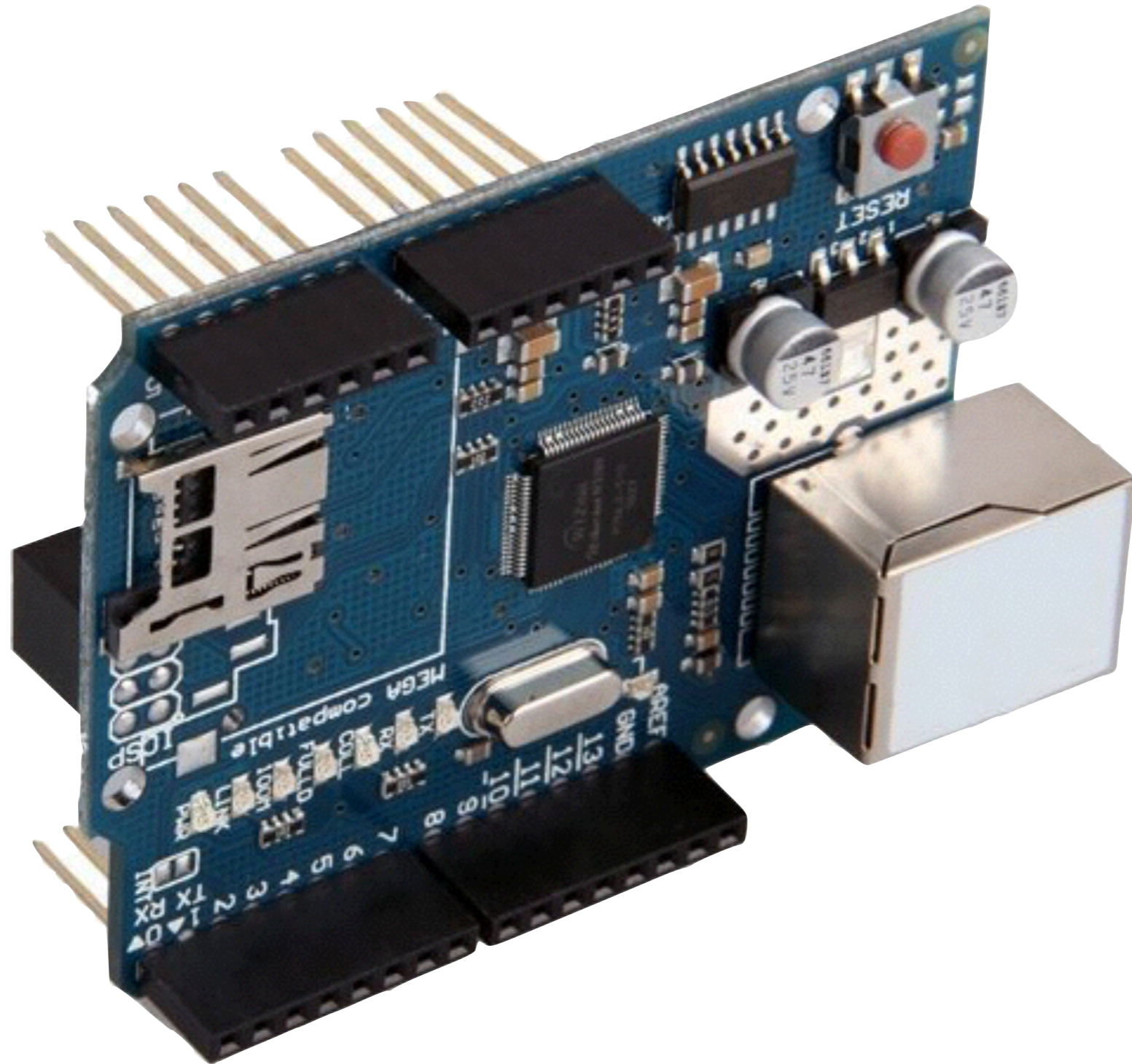


What You Need



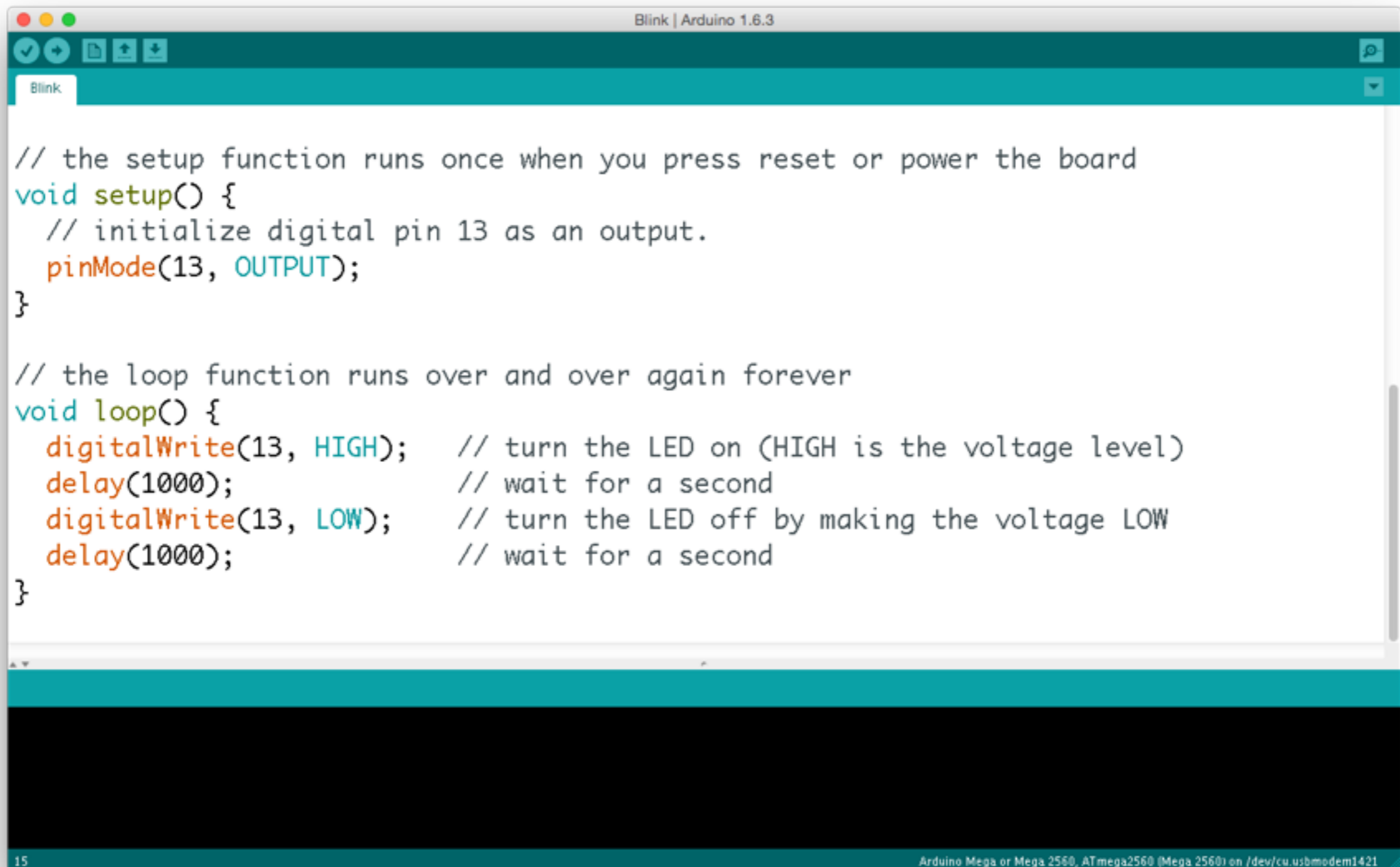
LCD-Module I2C/TWI 1602 Serial

What You Need



W5100 Ethernet Shield for Arduino Mega

What You Need

A screenshot of the Arduino IDE interface. The window title is "Blink | Arduino 1.6.3". The code editor shows the following C++ code for a Blink sketch:

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(13, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);           // wait for a second
  digitalWrite(13, LOW); // turn the LED off by making the voltage LOW
  delay(1000);           // wait for a second
}
```

The IDE interface includes a toolbar with icons for saving, running, and uploading, and a status bar at the bottom showing "15" and "Arduino Mega or Mega 2560, ATmega2560 (Mega 2560) on /dev/cu.usbmodem1421".

Development environment (Windows, Mac OS X, Linux)

What You Need



Programmieren für Ingenieure... x

https://www.st.cs.uni-saarland.de/edu/pfe/ws15/

Programming for Engineers

Background

The program [International Engineering](#) is designed to offer people with refugee background the possibility to become quickly integrated in engineering studies. In combination with German language courses the lecture "Programming for Engineers" allows people entitled to political asylum to be prepared for a follow-up study in engineering, mechatronics or likewise.

General Information

What is programming? What makes it exciting? In this lecture you will be introduced to the world of programming and will solve challenging tasks on embedded micro controllers such as the [Arduino Mega Boards](#). We will mainly focus on learning basic programming concepts and gaining practical experience.

| | |
|-----------------|--|
| Type | 8 CP (for students of mechatronics and participants of International Engineering) |
| Time | Tuesday 16 - 18 |
| Location | E 1.3. HS 002 |
| Course language | english |
| Contact | pfe15@lists.st.cs.uni-saarland.de |

Questions regarding the lecture, exercises or likewise may asked on the [mailing list](#).

Regulatories

Prerequisites

This course is especially offered to those who are entitled to political asylum or are recognised refugees and do not possess their certificates anymore, but have previous education, acquired in their native countries, enabling them to study at a university.



People with this background need to

- pass the entrance test that assesses your subject-specific qualification enabling you to study a subject of the "MINT" range. Have a look at the [official regulations](#)
- have sufficient english language skills. Lecture, exam and excecises are held in english.

The course is open for students of mechatronics, as well as micro technology and nano structures (BCP). Furhermore, all students can enroll for this lecture and can achieve 5 CP.

Registration

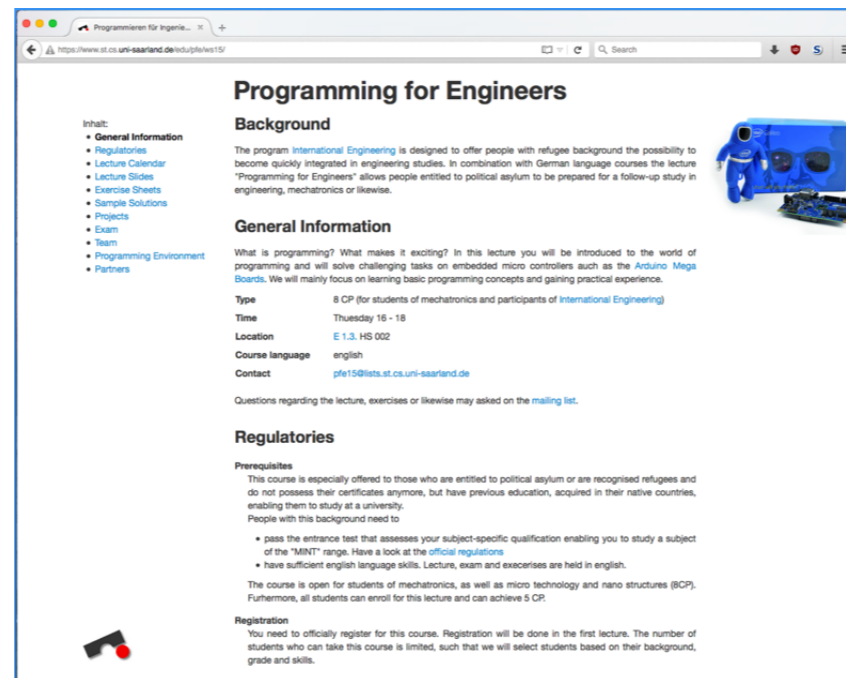
You need to officially register for this course. Registration will be done in the first lecture. The number of students who can take this course is limited, such that we will select students based on their background, grade and skills.



<https://www.st.cs.uni-saarland.de/edu/pfe/ws15/>

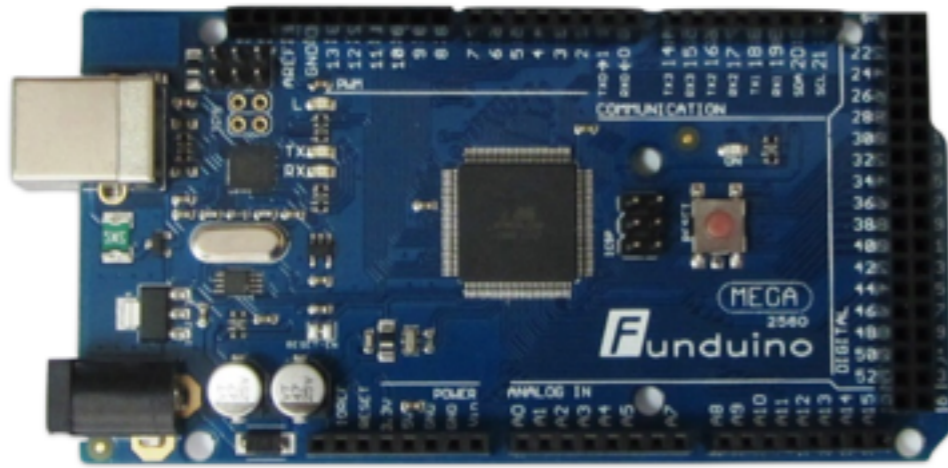
Next Steps

- Registration
- Assignment to groups
- Obtaining of boards and parts



<https://www.st.cs.uni-saarland.de/edu/pfe/ws15/>

Your Computer

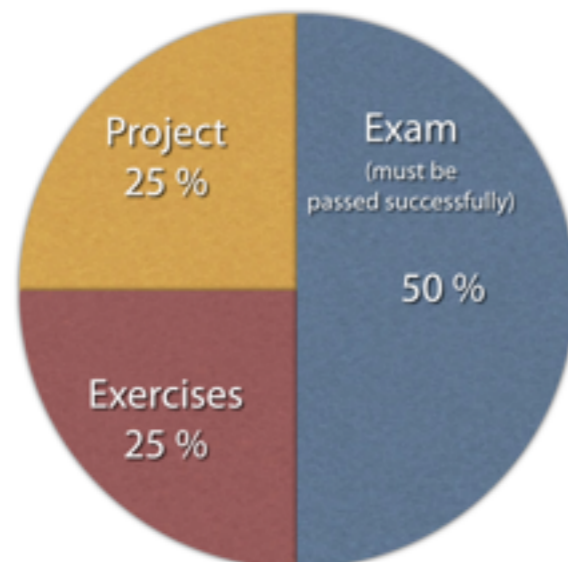


Structure

- Lecture on programming
- Exercise on Arduino-Board
- Free project
- Exam at the end of the course

<https://www.st.cs.uni-saarland.de/edu/pfe/ws15/>

Overall Grading



Next Steps

- Registration
- Assignment to groups
- Obtaining of boards and parts



<https://www.st.cs.uni-saarland.de/edu/pfe/ws15/>