Datenstrukturen und Algorithmen

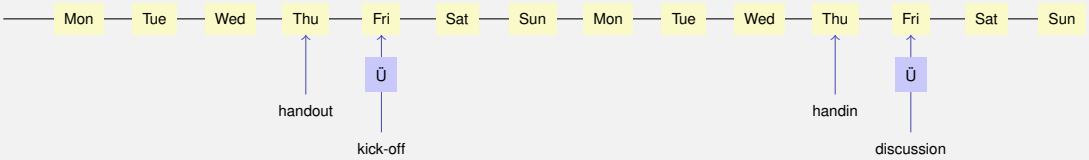
Exercise 1

FS 2018

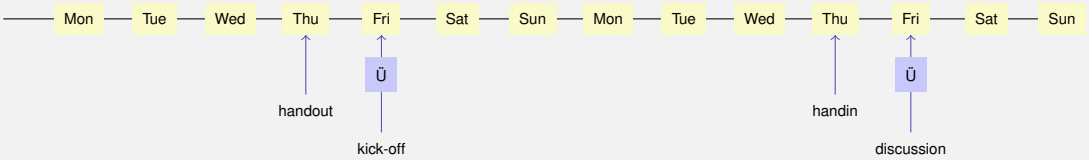
Schedule for today

1. Exercise Process
2. Repetition Theory
3. Programming Exercise

Process for the exercises



Process for the exercises



- Thursday:
  - Handout of new exercise sheet (online per Code Expert).
  - Submission of old exercise sheet (online per Code Expert).
- Friday during exercise class:
  - Kick-off presentation of new exercise sheet.
  - Discussion of old exercise sheet.
  - Opportunity to ask questions about lecture and exercises.

Offer

- Doing the weekly exercise series $\rightarrow$ bonus of maximally 0.25 of a grade points for the exam.
- The bonus is proportional to the achieved points of **specially marked bonus-task**. The full number of points corresponds to a bonus of 0.25 of a grade point.
- For the **admission** to bonus task 1 you need to gain 201 points on the first three exercise tasks.
- Rationale: You should have had a serious look at the exercise before doing the bonus task.
- The bonus task is unlocked as soon as you have the required 201 points.

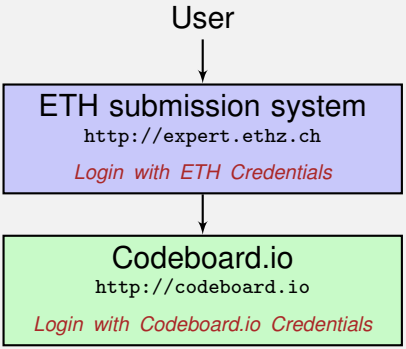
Submission using Code Expert

- Create account
- Sign in
- Solve exercises and submit them

Code Expert @ETH

Code Expert consists of two independend systems that interact:

- **The ETH submission system:** Allows us to grade your exercises
- **The online IDE:** The development environment



Codeboard

Codeboard.io registration

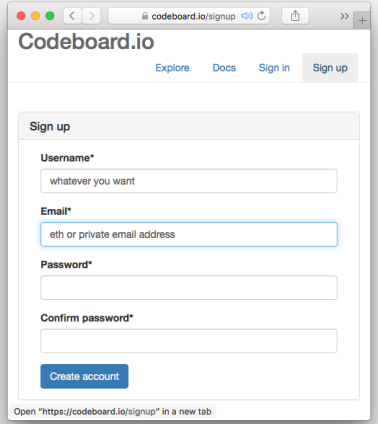
Go to `https://codeboard.io` and create an account; preferably stay logged in.

Enrolling to exercise groups

Go to `https://expert.ethz.ch/da2018` and register to an exercise group.

Codeboard.io Registration

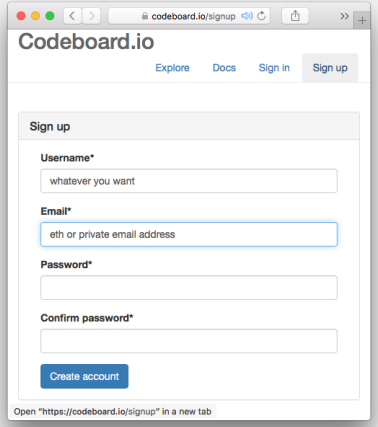
If you don’t have a **Codeboard.io** account ...



- We use the online IDE **Codeboard.io**

Codeboard.io Registration

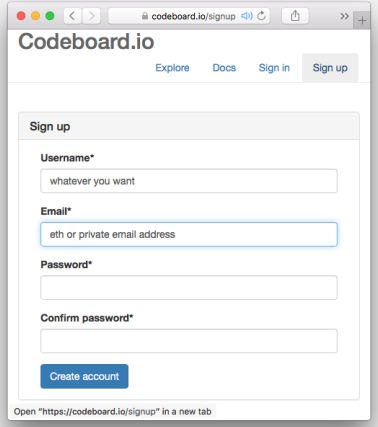
If you don’t have a **Codeboard.io** account ...



- We use the online IDE **Codeboard.io**
- Create an account there to save your progress and to later look at submissions

Codeboard.io Registration

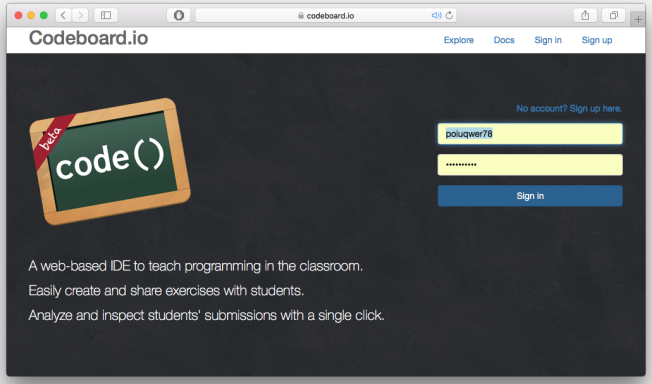
If you don’t have a **Codeboard.io** account ...



- We use the online IDE **Codeboard.io**
- Create an account there to save your progress and to later look at submissions
- Login credentials can be chosen as you like. *Don’t use your ETH password!*

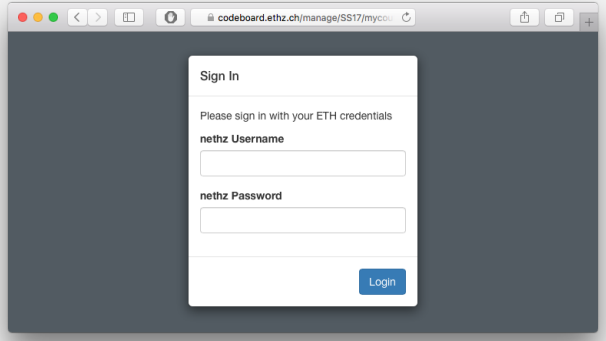
Codeboard.io Login

If you already have an account, sign in:



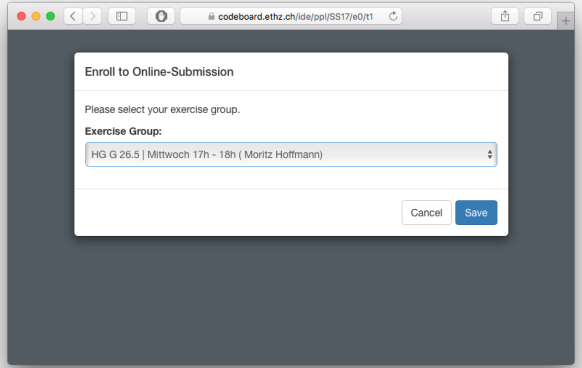
Enrolling to exercise groups - I

- Go to `https://expert.ethz.ch/da2018`
- Sign in using your ethz account.



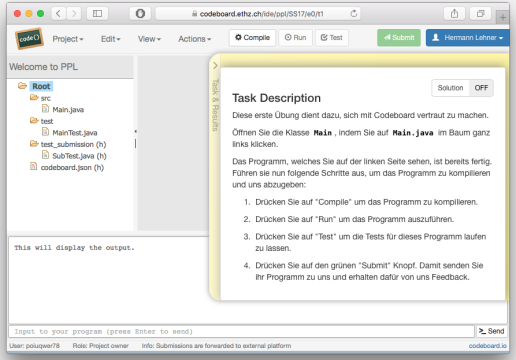
Enrolling to exercise groups - II

Use the dialogue to enroll to an exercise group.



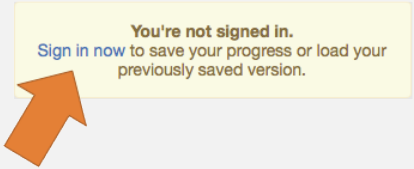
The first exercise

As you are now signed in and the first exercise is loaded. Follow the instructions in the yellow box.



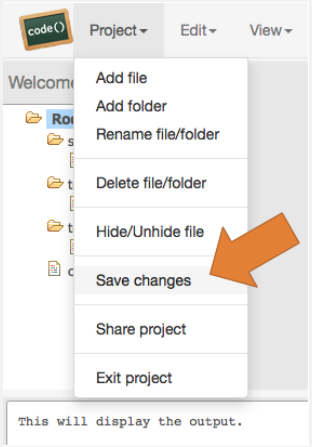
The first exercise: Codeboard.io login

Attention! If you see this message, click on Sign in now and sign in there with your **Codeboard.io** account.



The first exercise: Save your progress

Attention! Save your progress regularly, so you can continue working anywhere.



2. Repetition Theory

Warm-up

- What is a problem?

Warm-up

- What is a problem?
- What is an algorithm?

Warm-up

- What is a problem?
- What is an algorithm?
  - ➜ well-defined computing procedure to compute output data from input data.

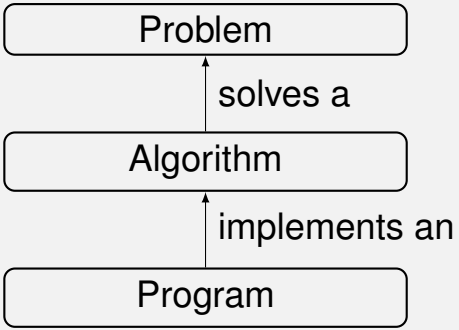
Warm-up

- What is a problem?
- What is an algorithm?
  - ➜ well-defined computing procedure to compute output data from input data.
- What is a program?

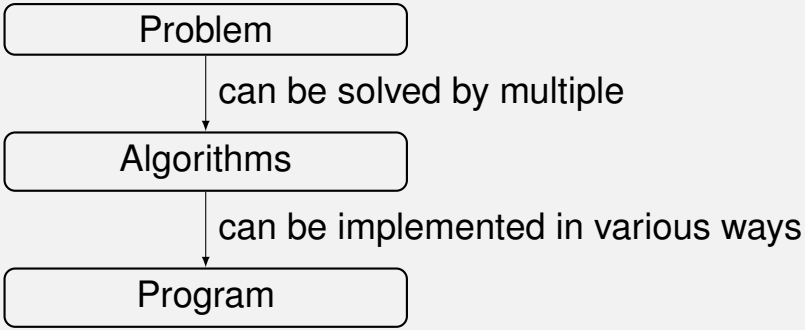
Warm-up

- What is a problem?
- What is an algorithm?
  - ➜ well-defined computing procedure to compute output data from input data.
- What is a program?
  - ➜ Concrete implementation of an algorithm

Warm-up



Warm-up


Problem
can be solved by multiple
Algorithms
can be implemented in various ways
Program

Efficiency

Problem	Complexity	Minimal (asymptotic) cost over all algorithms that solve the problem.
Algorithm	Cost	Number of elementary operations
Program	Computing time	Measurable value on an actual machine.

Efficiency

Problem	Complexity	Minimal (asymptotic) cost over all algorithms that solve the problem.
Algorithm	Cost	Number of elementary operations
Program	Computing time	Measurable value on an actual machine.

➜ Estimation of cost or computing time depending on the input size, denoted by n.

Asymptotic behavior

- What are $\Omega(g(n))$, $\Theta(g(n))$, $\mathcal{O}(g(n))$?

Asymptotic behavior

- What are $\Omega(g(n))$, $\Theta(g(n))$, $\mathcal{O}(g(n))$?

➜ Sets of functions!

Asymptotic behavior

- What are $\Omega(g(n))$, $\Theta(g(n))$, $\mathcal{O}(g(n))$?

➜ Sets of functions!

Repetition, sets A, B:	
subset	$A \subseteq B$
proper subset	$A \subsetneq B$
intersection	$A \cap B$

Asymptotic behavior

Given: function $f : \mathbb{N} \to \mathbb{R}$.

Definition:

$$\mathcal{O}(g) = \{f : \mathbb{N} \to \mathbb{R} | \exists c > 0, n_0 \in \mathbb{N} : 0 \leq f(n) \leq c \cdot g(n) \; \forall n \geq n_0\}$$
$$\Omega(g) = \{f : \mathbb{N} \to \mathbb{R} | \exists c > 0, n_0 \in \mathbb{N} : 0 \leq c \cdot g(n) \leq f(n) \; \forall n \geq n_0\}$$
$$\Theta(g) = \mathcal{O}(g) \cap \Omega(g)$$

Useful information for the exercise

Theorem

1. $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0 \Rightarrow f \in \mathcal{O}(g), \mathcal{O}(f) \subsetneq \mathcal{O}(g)$.
2. $\lim_{n\to\infty} \frac{f(n)}{g(n)} = C > 0$ *(C constant)* $\Rightarrow f \in \Theta(g)$.
3. $\frac{f(n)}{g(n)} \underset{n\to\infty}{\to} \infty \Rightarrow g \in \mathcal{O}(f), \mathcal{O}(g) \subsetneq \mathcal{O}(f)$.

Useful information for the exercise

Theorem

1. $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0 \Rightarrow f \in \mathcal{O}(g), \mathcal{O}(f) \subsetneq \mathcal{O}(g).$
2. $\lim_{n\to\infty} \frac{f(n)}{g(n)} = C > 0$ *(C constant)* $\Rightarrow f \in \Theta(g).$
3. $\frac{f(n)}{g(n)} \underset{n\to\infty}{\to} \infty \Rightarrow g \in \mathcal{O}(f), \mathcal{O}(g) \subsetneq \mathcal{O}(f).$

Beispiel

1. $\lim_{n\to\infty} \frac{n}{n^2} = 0 \Rightarrow n \in \mathcal{O}(n^2), \mathcal{O}(n) \subsetneq \mathcal{O}(n^2).$
2. $\lim_{n\to\infty} \frac{2n}{n} = 2 > 0 \Rightarrow 2n \in \Theta(n).$
3. $\frac{n^2}{n} \underset{n\to\infty}{\to} \infty \Rightarrow n \in \mathcal{O}(n^2), \mathcal{O}(n) \subsetneq \mathcal{O}(n^2).$

Quiz

$1 \in \mathcal{O}(15)$?

Quiz

$1 \in \mathcal{O}(15)$? ✓ better $1 \in \mathcal{O}(1)$

Quiz

$1 \in \mathcal{O}(15)$?

✓ better $1 \in \mathcal{O}(1)$

$2n + 1 \in \Theta(n)$?

Quiz

$1 \in \mathcal{O}(15)$?	✓ better $1 \in \mathcal{O}(1)$
$2n + 1 \in \Theta(n)$?	✓

Quiz

$1 \in \mathcal{O}(15)$?	✓ better $1 \in \mathcal{O}(1)$
$2n + 1 \in \Theta(n)$?	✓
$\sqrt{n} \in \mathcal{O}(n)$?	

Quiz

$1 \in \mathcal{O}(15)$?	✓ better $1 \in \mathcal{O}(1)$
$2n + 1 \in \Theta(n)$?	✓
$\sqrt{n} \in \mathcal{O}(n)$?	✓

Quiz

$1 \in \mathcal{O}(15)$?	✓ better $1 \in \mathcal{O}(1)$
$2n + 1 \in \Theta(n)$?	✓
$\sqrt{n} \in \mathcal{O}(n)$?	✓
$\sqrt{n} \in \Omega(n)$?	

Quiz

$1 \in \mathcal{O}(15)$?	✓ better $1 \in \mathcal{O}(1)$
$2n + 1 \in \Theta(n)$?	✓
$\sqrt{n} \in \mathcal{O}(n)$?	✓
$\sqrt{n} \in \Omega(n)$?	✗

Quiz

$1 \in \mathcal{O}(15)$?	✓ better $1 \in \mathcal{O}(1)$
$2n + 1 \in \Theta(n)$?	✓
$\sqrt{n} \in \mathcal{O}(n)$?	✓
$\sqrt{n} \in \Omega(n)$?	✗
$n \in \Omega(\sqrt{n})$?	

Quiz

$1 \in \mathcal{O}(15)$?	✓ better $1 \in \mathcal{O}(1)$
$2n + 1 \in \Theta(n)$?	✓
$\sqrt{n} \in \mathcal{O}(n)$?	✓
$\sqrt{n} \in \Omega(n)$?	✗
$n \in \Omega(\sqrt{n})$?	✓

Quiz

$1 \in \mathcal{O}(15)$?	✓ better $1 \in \mathcal{O}(1)$
$2n + 1 \in \Theta(n)$?	✓
$\sqrt{n} \in \mathcal{O}(n)$?	✓
$\sqrt{n} \in \Omega(n)$?	✗
$n \in \Omega(\sqrt{n})$?	✓
$\sqrt{n} \notin \Theta(n)$?	

Quiz

$1 \in \mathcal{O}(15)$?	✓ better $1 \in \mathcal{O}(1)$
$2n + 1 \in \Theta(n)$?	✓
$\sqrt{n} \in \mathcal{O}(n)$?	✓
$\sqrt{n} \in \Omega(n)$?	✗
$n \in \Omega(\sqrt{n})$?	✓
$\sqrt{n} \notin \Theta(n)$?	✓

Quiz

$1 \in \mathcal{O}(15)$?	✓ better $1 \in \mathcal{O}(1)$
$2n + 1 \in \Theta(n)$?	✓
$\sqrt{n} \in \mathcal{O}(n)$?	✓
$\sqrt{n} \in \Omega(n)$?	✗
$n \in \Omega(\sqrt{n})$?	✓
$\sqrt{n} \notin \Theta(n)$?	✓
$\mathcal{O}(\sqrt{n}) \subset \mathcal{O}(n)$?	

Quiz

$1 \in \mathcal{O}(15)$?	✓ better $1 \in \mathcal{O}(1)$
$2n + 1 \in \Theta(n)$?	✓
$\sqrt{n} \in \mathcal{O}(n)$?	✓
$\sqrt{n} \in \Omega(n)$?	✗
$n \in \Omega(\sqrt{n})$?	✓
$\sqrt{n} \notin \Theta(n)$?	✓
$\mathcal{O}(\sqrt{n}) \subset \mathcal{O}(n)$?	✓

Quiz

$1 \in \mathcal{O}(15)$?	✓ better $1 \in \mathcal{O}(1)$
$2n + 1 \in \Theta(n)$?	✓
$\sqrt{n} \in \mathcal{O}(n)$?	✓
$\sqrt{n} \in \Omega(n)$?	✗
$n \in \Omega(\sqrt{n})$?	✓
$\sqrt{n} \notin \Theta(n)$?	✓
$\mathcal{O}(\sqrt{n}) \subset \mathcal{O}(n)$?	✓
$2^n \notin \mathcal{O}(\exp(n))$?	

Quiz

$1 \in \mathcal{O}(15)$?	✓ better $1 \in \mathcal{O}(1)$
$2n + 1 \in \Theta(n)$?	✓
$\sqrt{n} \in \mathcal{O}(n)$?	✓
$\sqrt{n} \in \Omega(n)$?	✗
$n \in \Omega(\sqrt{n})$?	✓
$\sqrt{n} \notin \Theta(n)$?	✓
$\mathcal{O}(\sqrt{n}) \subset \mathcal{O}(n)$?	✓
$2^n \notin \mathcal{O}(\exp(n))$?	✗

Quiz

$1 \in \mathcal{O}(15)$?	✓ better $1 \in \mathcal{O}(1)$
$2n + 1 \in \Theta(n)$?	✓
$\sqrt{n} \in \mathcal{O}(n)$?	✓
$\sqrt{n} \in \Omega(n)$?	✗
$n \in \Omega(\sqrt{n})$?	✓
$\sqrt{n} \notin \Theta(n)$?	✓
$\mathcal{O}(\sqrt{n}) \subset \mathcal{O}(n)$?	✓
$2^n \notin \mathcal{O}(\exp(n))$?	✗

3. Programming Exercise

Sum in sub-interval (naive algorithm)

Input : A sequence of n numbers $(a_0, a_1, \ldots, a_{n-1})$ and a sub-interval $I = [x_0, x_1]$

Output : $\sum_{i=x_0}^{x_1} a_i$.

$\mathcal{S} \leftarrow 0$
for $i \in \{x_0, \ldots, x_1\}$ **do**
  $\mathcal{S} \leftarrow \mathcal{S} + a_i$
return $\mathcal{S}$

Sum in sub-interval (naive algorithm)

Input : A sequence of n numbers $(a_0, a_1, \ldots, a_{n-1})$ and a sub-interval $I = [x_0, x_1]$

Output : $\sum_{i=x_0}^{x_1} a_i$.

$\mathcal{S} \leftarrow 0$
for $i \in \{x_0, \ldots, x_1\}$ **do**
  $\mathcal{S} \leftarrow \mathcal{S} + a_i$
return $\mathcal{S}$

Idea

- Use the prefix sum to compute the sum of arbitrary sub-intervals with constant complexity
- Generalization to two dimensions.

Multidimensional vectors

Definition

```
std::vector< std::vector<int> > my_vec( n_rows,
std::vector<int>(n_cols,init_value) );
```

Indexing

```
my_vec[row][col]
```

Classes

```
class Insurance { // Definition
public: // public section
        Insurance(double rate) {rate_ = rate;} // Construktor
        double get_rate() {return rate_;} // member function
private: // private section
        double rate_; // data member
};

int main() {
        Insurance insurance(2.);
        std::cout << insurance.get_rate();
        return 0;
}
```

Questions?

Questions?

Let’s get to work.