Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Informatik I am D-BAUG
Autumn Semester 2017

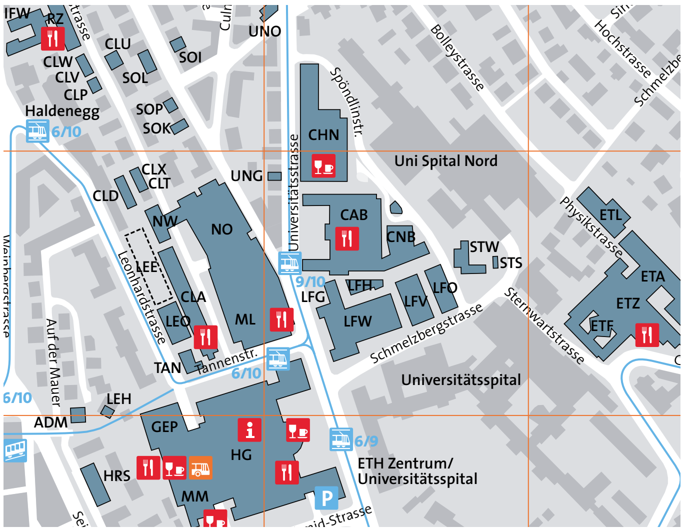
Exercise Organisation and Submission Environment

Exercise Subscription via mystudies / echo.ethz.ch

Once you have registered for the course in mystudies, you can subscribe to an exercise group. Go to the website `http://echo.ethz.ch` and log in with your ETH account and then choose your group. This procedure allows a synchronization between the different courses (Analysis, Linear Algebra, Informatik). For the explanations following in the remainder of this document it is assumed that you have successfully registered to a group using echo.ethz.ch.

Exercise Times and Places, Site Map

Donnerstag 13:15 - 15:00	
Philipp Schimmelfennig	CHN G 22
Kai Sandbrink	ETZ J 91
Cimen Gökcen	*HG G 26.1
Patrick Gruntz	IFW B 42
Irfan Bunjaku	IFW C 31
Simon Guldimann	IFW C 33
Clemens Bachmann	LFW C 5
Donnerstag 15:15 - 17:00	
Staal Sander	ETZ J 91
Frederic Vogel	CHN G 22
Rafael Wampfler	HG G 26.1
Malte Schwerhoff	LFW C 1
Isik Nihat	LFW C 11



Exercise Organisation

- Exercises will be made available online via our course homepage `http://lec.inf.ethz.ch/baug/informatik1`.
- You can work on an exercise for about one week after you have looked at it in the exercise groups (the *submission period*).
- You submit all your solutions via an online submission system.
- Your programming solutions are checked and automatically graded by the submission system. You will get an immediate feedback indicating the correctness of your solution. Your assistant has access to your submissions, will check your solution and give feedback. Solutions to theory exercises will manually be graded by your assistant within the submission system.
- The assistant will look at the final submission only. A submission is final if it was the last submission within the submission period of an exercise or if you explicitly handed it in.
- The assistant has the right to override the points provided by the auto-grader, for example if the auto-grader failed to detect that you did not follow the exercise instructions.
- Within the submission period you can always resubmit unless you have explicitly handed in your solution. Should you want to modify a handed-in solution within the submission period, please inform your assistant.

- In order to submit a program, it has to compile without errors.
- Programs should only contain language constructs that have been discussed in the course or exercise sessions. This may sometimes be harder if you already have some programming experience – it is nevertheless mandatory (and beneficial for you) to respect this rule.
- Program texts are written by and for humans, even if they can be understood by machines. Therefore you must pay attention to the visual appearance of your programs. In particular, this requires commenting as well as indentation and use of spaces, e.g. before and after operators and keywords. If your coding style is unacceptable in this sense, we reserve the right to deduct points.
- The assistants are asked to correct your submitted exercises within a week.

Exam and Bonus Points

The exam for this lecture will take place in Winter 2018 or in Summer 2018, in form of a one-hour written exam (no auxiliary material allowed).

During the semester you can collect bonus points by regularly submitting solutions to exercises. You can achieve points for each exercise, and the points collected during the semester will be summed up. The achieved grade bonus is proportional to the achieved points over all exercises. Achieving all points corresponds to a bonus of 1/4 of a grade.

Bonus Points Rules

1. **No cheating allowed.** While we encourage *discussing* exercises in groups, each submission must represent the work of a single student. If we discover, that two or more students submitted the same solution, or that you submitted a solution from a previous year, you will lose the complete bonus grade: your assistant will set the achieved points to zero for all exercises of the complete course.
2. **Follow the rules**. Certain exercises have specific rules, e.g., exercise 1.2 does not allow to use multiplications, loops, or recursion. Submitted solutions that do not follow these rules do not receive any points (set to zero manually by your assistant). This includes the rule that your solution can only contain language constructs that have been discussed during the course or the exercise sessions.
3. **No hard coded results**. Solutions that are not solving the given task but only the given test cases count as failed. Your assistant will reduce the score for such hard-coded solutions.

The Programming and Submission Environment

We use a web-based integrated development environment (IDE) for the exercises. You can work at any computer providing internet access. Note that ETH features a lot of computing rooms with public computers that you can use to execute the exercises.

Of course, you can also use your own computer, if available. If you intend to buy a (new) notebook you may want to check Neptun, a platform for ETH members providing good conditions (`http://www.neptun.ethz.ch/`).

If you intend to work with your own laptop, then bring it along to the first exercise hour. Goal of the first exercise session is that everyone has access to and knows how to use the web-based IDE and the submission system.

For this lecture, we use `[code]expert` and `codeboard.io`. This is a pair of independent online services that provide an exercise submission system and a web-based integrated development environment (IDE). The two systems communicate with each other.

You log into the submission system using the following URL: `https://expert.ethz.ch`. To login, use your regular ETH credentials (nethz username and password).

Note: The submission system is still young. If you encounter issues with the system, please report them. We are in direct contact with the developers. Depending on the issue we may be able to fix it even before the course ends, so you can still benefit. Please report issues using the email contact address on the lecture homepage.

We have a daily maintenance window from 08:00 - 09:00 (morning). This window allows us a timely installation of bugfixes. During this time the system may not be always available, but these down times are usually short. If the system is not available during the maintenance window, simply try again after a few minutes.

Performing a programming task

For the first login, use the link `https://expert.ethz.ch/baugi1_2017e00t01`. This link allows you to enroll in the online exercise submission. It will also open the programming environment, and take you directly to your first programming task.

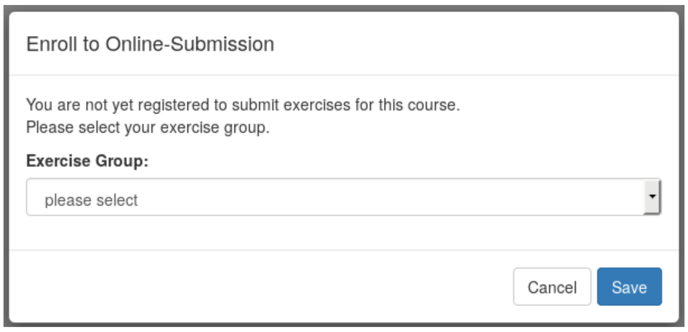
Performing a programming task is generally done in the following steps:

1. Open the programming environment (via exercise link or overview)
2. Write the program as requested in the exercise.
3. Test the program manually using your own inputs (compile & run).
4. Use the automated tests (using `RunTests`).
5. Submit. Remember, you can submit multiple times.

Of course, you can go back to a previous step if you are not satisfied with the result. The following sections explain the above steps in more detail.

Some of the tasks will not require any coding. In this case, instead of a coding environment you will see a simple text editor to write your answer. Within this editor you can also upload attachments. For example you can upload screenshot images if asked in the task description.

Enrolling in the Online Submission


Enroll to Online-Submission

You are not yet registered to submit exercises for this course.
Please select your exercise group.

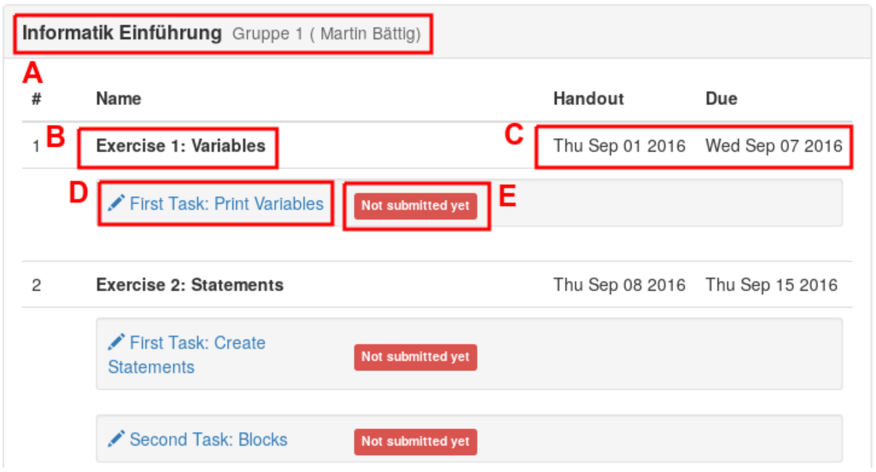
Exercise Group:

please select

The first time you log into the system, and select a programming task of this course you are automatically enrolled in the online exercise submission of this course. Before you can start working on the task, you have to find and select your exercise group.

Important: This selection does not replace exercise group assignment. It is your task to select the exact same exercise group as the one that you are enrolled in via the subscription link that your received via email at the begin of the semester.

The Overview Page



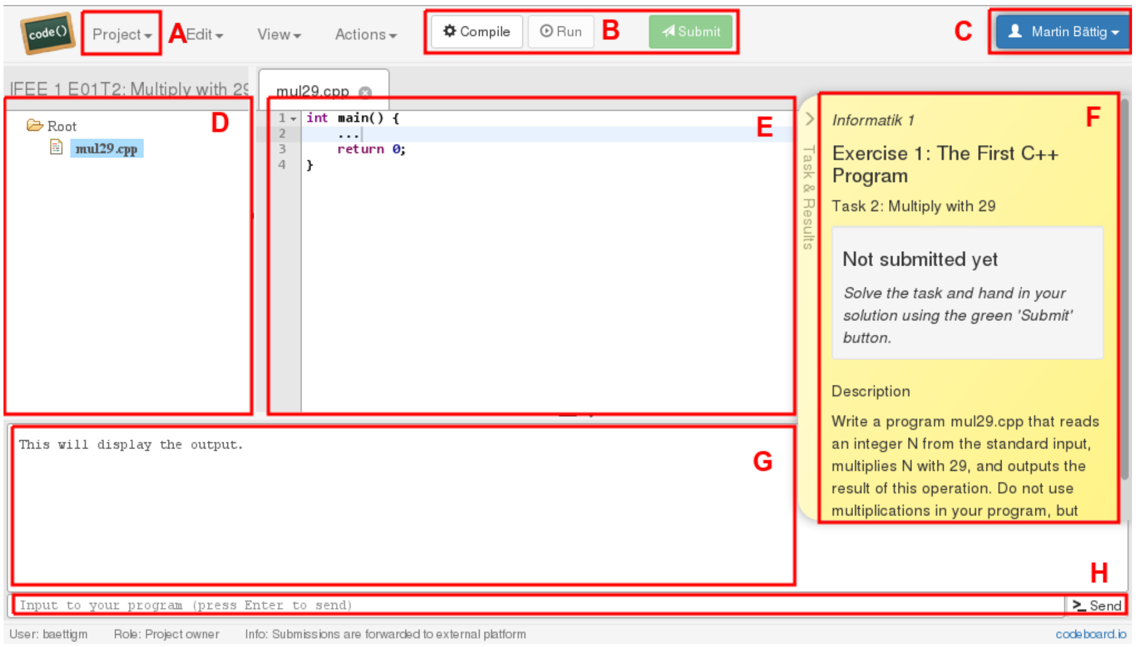
Informatik Einführung Gruppe 1 (Martin Bättig)

#	Name	Handout	Due
1	**Exercise 1: Variables**	Thu Sep 01 2016	Wed Sep 07 2016
	First Task: Print Variables — Not submitted yet		
2	**Exercise 2: Statements**	Thu Sep 08 2016	Thu Sep 15 2016
	First Task: Create Statements — Not submitted yet		
	Second Task: Blocks — Not submitted yet		

The overview page shows the available exercises and your progress on completion of the programming tasks. You reach the overview page by selecting overview in the top right menu or if you login to the system via `https://expert.ethz.ch`. In the overview, you can see the course that you are enrolled in and your group assignment (A). For each exercise sheet (B) you can see the handout date and the due date (C). The due date is not strictly enforced by the system to allow exceptions (e.g., late hand-ins due to sickness). Per exercise sheet you see the links to the programming tasks (D). Clicking on a programming task link opens the programming environment. These are the same links that you find on the exercises sheets themselves. For each programming task, the overview page shows its submission status (E). The section *Submission* contains an overview of the different submission statuses.

Programming

By clicking on a programming task link in the overview page, or by using a link on the exercise sheet, you will open the specific task in the programming environment. The screen looks like this:



```
int main() {
    ...
    return 0;
}
```

Informatik 1

Exercise 1: The First C++ Program

Task 2: Multiply with 29

Not submitted yet

Solve the task and hand in your solution using the green 'Submit' button.

Description

Write a program mul29.cpp that reads an integer N from the standard input, multiplies N with 29, and outputs the result of this operation. Do not use multiplications in your program, but

```
This will display the output.
```

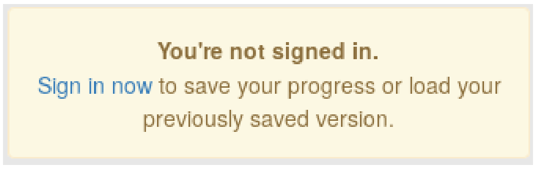
```
Input to your program (press Enter to send)
```

User: baettigm Role: Project owner Info: Submissions are forwarded to external platform

The file overview (D) lets you select a file to edit. The file is then opened in the editor. The name of the file that is important for the programming task is usually mentioned in the exercise task description. In the editor (E), you will write the actual code. If you have multiple files open, you can switch between them by using the tabs above the editor. After you wrote your program, you have to compile it (B). Compile time errors are listed in the output area (G). If your program compiles without errors, you can execute it by clicking on the run button (B). During program execution, the output area (E) shows the output of your program. In the input area (H), you can enter the input for the program. Use return or the send button to send the input to your program. Further, there is the task area (F). The task area shows the programming task you are currently working on, the result of your previous submissions, and a short description of the task.

Saving a Program

To save your program, use the project menu (A), and chose option *save changes*. The first time you do this, you need to create a user account on `codeboard.io` to be able to save your progress:


You're not signed in.
Sign in now to save your progress or load your previously saved version.

This user account is different from the nethz account. To create a new user, click on *Sign in now*, and then on *Sign up* (top right), and follow the directions on the page. Note: This account is not under control of ETH, thus make sure to use a password different than the one of your nethz account.[1]

Useful keyboard shortcuts

The following table lists some useful keyboard shortcuts for the text editor:

Combination	Action
Ctrl-/	Toggle comment
Tab	Indent
Shift-Tab	Outdent
Ctrl-Z	Undo
Ctrl-Shift-Z, Ctrl-Y	Redo
Ctrl-F	Find
Ctrl-H	Replace
Ctrl-K	Find next
Ctrl-Shift-K	Find previous
Ctrl-L	Go to line

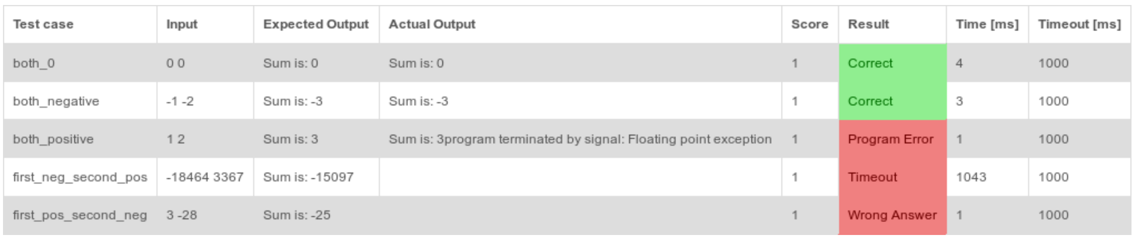
Testing

To test your solution, you can run it against a set of predefined test inputs. By default these tests are disabled, i.e., the required line in the source code is commented. To enable the tests, remove the comment sign (the two forward slashes), so that the line becomes:

```
#include "tests.h"
```

Then recompile your program by clicking the compile button again. Now you can click on the run button and your program is tested automatically. It will produce a report like the following:

[1]The reason for this separate account is that `[code]expert` uses the Codeboard service to provide the programming environment. `https://www.codeboard.io`. Your program is stored on this Codeboard service.



Test case	Input	Expected Output	Actual Output	Score	Result	Time [ms]	Timeout [ms]
both_0	0 0	Sum is: 0	Sum is: 0	1	Correct	4	1000
both_negative	-1 -2	Sum is: -3	Sum is: -3	1	Correct	3	1000
both_positive	1 2	Sum is: 3	Sum is: 3program terminated by signal: Floating point exception	1	Program Error	1	1000
first_neg_second_pos	-18464 3367	Sum is: -15097		1	Timeout	1043	1000
first_pos_second_neg	3 -28	Sum is: -25		1	Wrong Answer	1	1000

The system tests your program using multiple test cases. Each test case consists of an input and an expected output. Your program passes a test if it produces an actual output for an input that matches the expected output of the test case. The comparison is quite tolerant. It is performed line by line and ignores redundant lines such as "Please enter a number:". Further, the comparison ignores white space such as spaces and newlines and the casing of words.

The result of a test case can be:

- **Correct.** Your program returns the correct answer, i.e., the actual output matches the expected output.
- **Wrong Answer:** Your program does not return the correct answer, i.e., the actual output does not match the expected output.
- **Program Error:** The program either crashed (e.g. division by zero), or returned an exit code unequal to zero.
- **Timeout:** The duration of the test case exceeded the maximal allowed duration. The timeout settings are usually generous and are primarily there to catch endless loops.

Submitting

To submit your programming task, use the submit button in the programming environment. The system automatically enables the tests and uses them to rate your solution according to the score assigned to each test case. Note that your assistant is only required to look at the last submitted version of each programming task, so make sure that your last submitted version is the one you want to be corrected.

A programming task can have the following submission statuses:

- **Not submitted.** This task is not submitted: Not submitted yet.
- **Submitted.** This task is submitted, but not yet reviewed. This is indicated with a percentage showing your score in this task. Example: 40%.
- **Reviewed with comment**. This task is submitted and reviewed. Your teaching assistant left a comment. This is indicated by a small mail icon on the right hand side of the score. Example: 80% ✉.
- **Reviewed without comment**. This task is submitted and reviewed. Your teaching assistant did not leave a comment. This is indicated by a small eye icon on the right hand side of the score. Example: 100% 👁.