Tutorial 01 - 22.10.2019

Group 06 - Moritz Makowski

Introduction

Today's Agenda

- What is Computer Science and Programming?
- Installing `gcc`
- Navigating inside a Terminal with `pwd`, `cd` and `ls`
- Writing our first program `hello_world.c`
- Compiling our first program
- Why should we use an IDE?
- Showcase: CLion
- Homework for IN8011

What is Computer Science?

> Informatics is the study of the storage, transformation, and communication of information.

> Informatik ist die Wissenschaft der systematischen, automatisierten Verarbeitung von Informationen.

Source: Department of Informatics @ TUM

What is Programming?

> Computer programming is the process of designing and building an executable computer program for accomplishing a specific computing task.

> Programmierung bezeichnet das Konzipieren und Erstellen ausführbarer Computerprogramme, die einen bestimmten Zweck erfüllen sollen.

Source: Computer programming @ Wikipedia

Important: Programming is only a part of Computer Science!

Let's just cut to the chase ... What do we do here?

Programmers write computer programs in a **programming language** of their choice.

Each language has its own benefits and trade offs.

We will use a programming language called **C**.

Example Script `hello_world.c`:

```c
#include <stdio.h>

int main() {
    int a = 2;
    int b = 7;
    int c = a + b;
    printf("Our number is: %d.", c);
}
```

And what can we do with it?

Computers can't read these scripts directly. You have to **translate** it first so that a computer can understand it. This translating is called **compiling**.

C Script ⇝ compiles to ⇝ **Executable Program**

Compilation progress:

```
$ gcc hello_world.c -o program
$ ./program
Our number is: ?.
```

Install gcc on Windows - #1

First, let’s check if **gcc** is already installed on your computer.

Open the terminal by typing `cmd` into your search bar and open it.

Then type **"gcc"** and press `enter`. If it says **`gcc: fatal error: no input files`** **gcc** is already installed, and you can lean back.

If it doesn’t recognize the command, follow the following instructions.

Install gcc on Windows - #2

1. Go to google, type **"MinGW getting started"**, open first result
2. Click on **"mingw-get-setup.exe"**
3. Open downloaded file
4. Choose **"C:\MinGW"** as path
5. Download the installation manager and open it
6. Mark **"mingw32-gcc-g++"** for installation
7. Click on **installation -> apply changes -> apply**

Install gcc on Windows - #3

You just installed gcc. But to be able to simply use it in the terminal, you first have to tell your system where to find the new program.

1. Close the Installation-Manager, open the explorer and go to the folder you just created (if you followed this instruction it's **"C:\MinGW"**)
2. Open **"bin"**
3. Click onto the bar with the path in it and copy the path, close explorer

Install gcc on Windows - #4

4. Type **"Erweiterte Systemeinstellungen"/"Advanced System Settings"** into the search bar and open it
5. Click on **"Systemumgebungsvariablen bearbeiten"** and then on the button **"Umgebungsvariablen"**
6. Search for the Variable **"Path"** (IMPORTANT! Path not PATH!) in the lower part of the window (System-Variables not User Variables)
7. Double-click **"Path"** and click on **"Neu"**
8. Paste the path you just copied and click on **"OK"** to close all the windows you just opened

Install gcc on Windows - #5

You did it! You can now test if your compiler works. Therefore type **"cmd"** into your searchbar and open it.

*(**IMPORTANT:** If the terminal is still open from the last time you used it, close and reopen it, otherwise the changes we just did won't be recognized.)*

Type **"gcc"** and press `enter`.

If it now says `gcc: fatal error: no input files` you are finished and have successfully installed gcc!

Install gcc on Mac - #1

It is really easy to install gcc on Mac!

1. Open the program called **terminal**
2. Type in **"gcc"** and hit `enter`

If it says **`clang: error: no input files`** then gcc is install correctly.

3. If you haven't installed gcc already a window pops up saying something like **"Install command line tools now?"**
4. Hit confirm and wait for the installation to finish (Can take a few minutes)

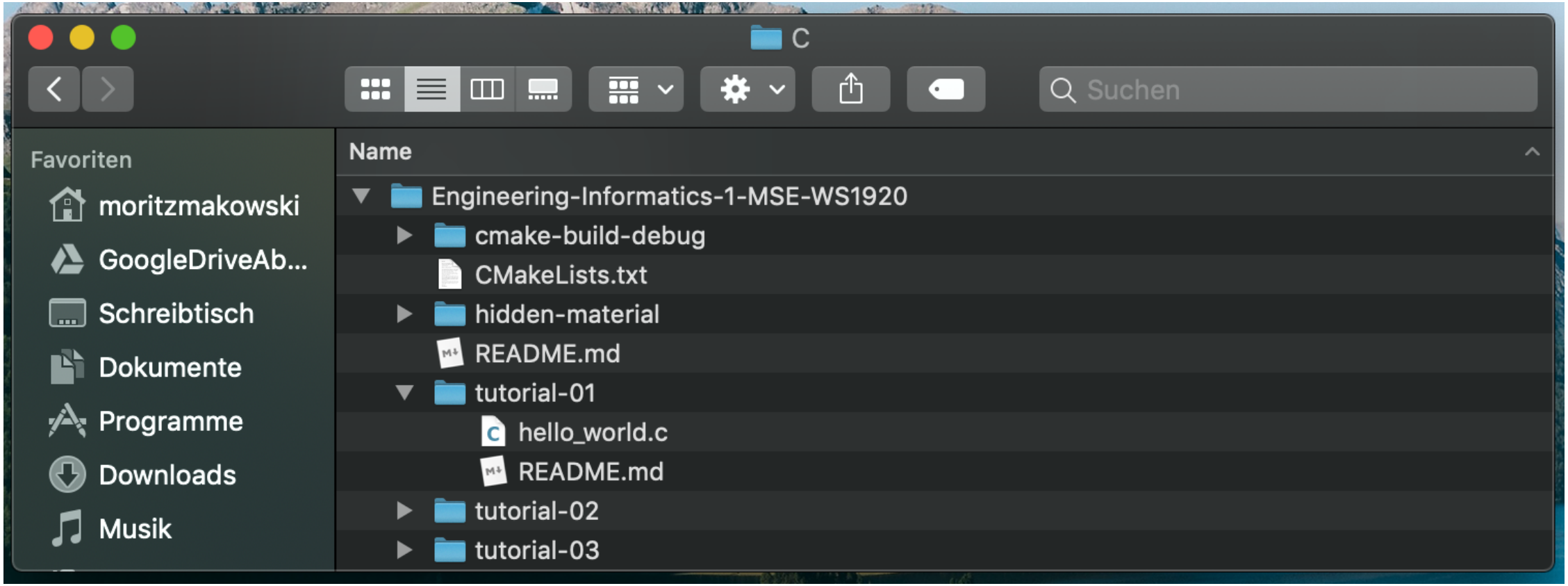
Install gcc on Mac - #2

Alternative installation via Hombrew:

1. Install Homebrew: https://brew.sh/
2. Run **"brew install gcc"** (Can take a few minutes)

Text based navigation through your file system - #1

The type of navigation you already know:



Text based navigation through your file system - #2

In the **console/terminal** we use three commands to navigate **without a real GUI** (Graphical User Interface).

1. `pwd` ("print working directory") prints out the path of the directory we are currently in. In windows you can use `echo %cd%`.
2. `cd <directory_name>` ("change directory") switches to another directory. `cd ..` goes to the parent directory of the current directory.
3. `ls` ("list directory") prints out all files inside the current directory. `ls -a` also prints hidden files (filename begins with `.`). In windows you can use `dir`.

Text based navigation through your file system - #3

`pwd`, `cd` and `ls` in action:

```
$ pwd
/Users/moritzmakowski/ComputerScience/C

$ ls
Big Projects                            Exercises
Cheatsheet Variables.pages              Small Projects
Engineering-Informatics-1-MSE-WS1920

$ cd Engineering-Informatics-1-MSE-WS1920

$ ls
CMakeLists.txt          tutorial-01             tutorial-04
README.md               tutorial-02             tutorial-05
cmake-build-debug       tutorial-03             tutorial-06

$ pwd
/Users/moritzmakowski/ComputerScience/C/Engineering-Informatics-1-MSE-WS1920
```

Our first C-Program - #1

We are inside the directory `tutorial-01`.

Inside the file `hello_world.c` we write:

```c
#include <stdio.h>

int main() {
    int a = 2;
    int b = 7;
    int c = a + b;
    printf("Our number is: %d.\n", c);
}
```

Our first C-Program - #2

Now we can compile it:

```
$ ls
hello_world.c

$ gcc hello_world.c -o program

$ ls
hello_world.c   program
```

We can execute the program with:

```
$ ./program
Our number is: 9.
```

Will we code with TextEdit forever?

No! There are special programs which are exclusively made for writing code:

- Editors
- IDE's (Integrated Development Environments)

Editors are made to write code. The program highlights the text in different colors so you can read the code more easily. This is called **syntax highlighting**.

Example:

```c
#include <stdio.h>

int main() {
    int a = 2;
    int b = 7;
    int c = a + b;
    printf("Our number is: %d.\n", c);
    return 0;
}
```

IDE's (Integrated Development Environments) are not only doing syntax highlighting but a lot more, such as:

- Code analysis: The program can find errors in your code and show specific error messages - sometimes even with a proposed solution
- Refactoring: Restructure your code, e.g. change variable names, file names, etc.
- Generate files from blueprints - e.g. a `hello_world.c` file when you create a new project
- ...

My Recommentdation: CLion by JetBrains

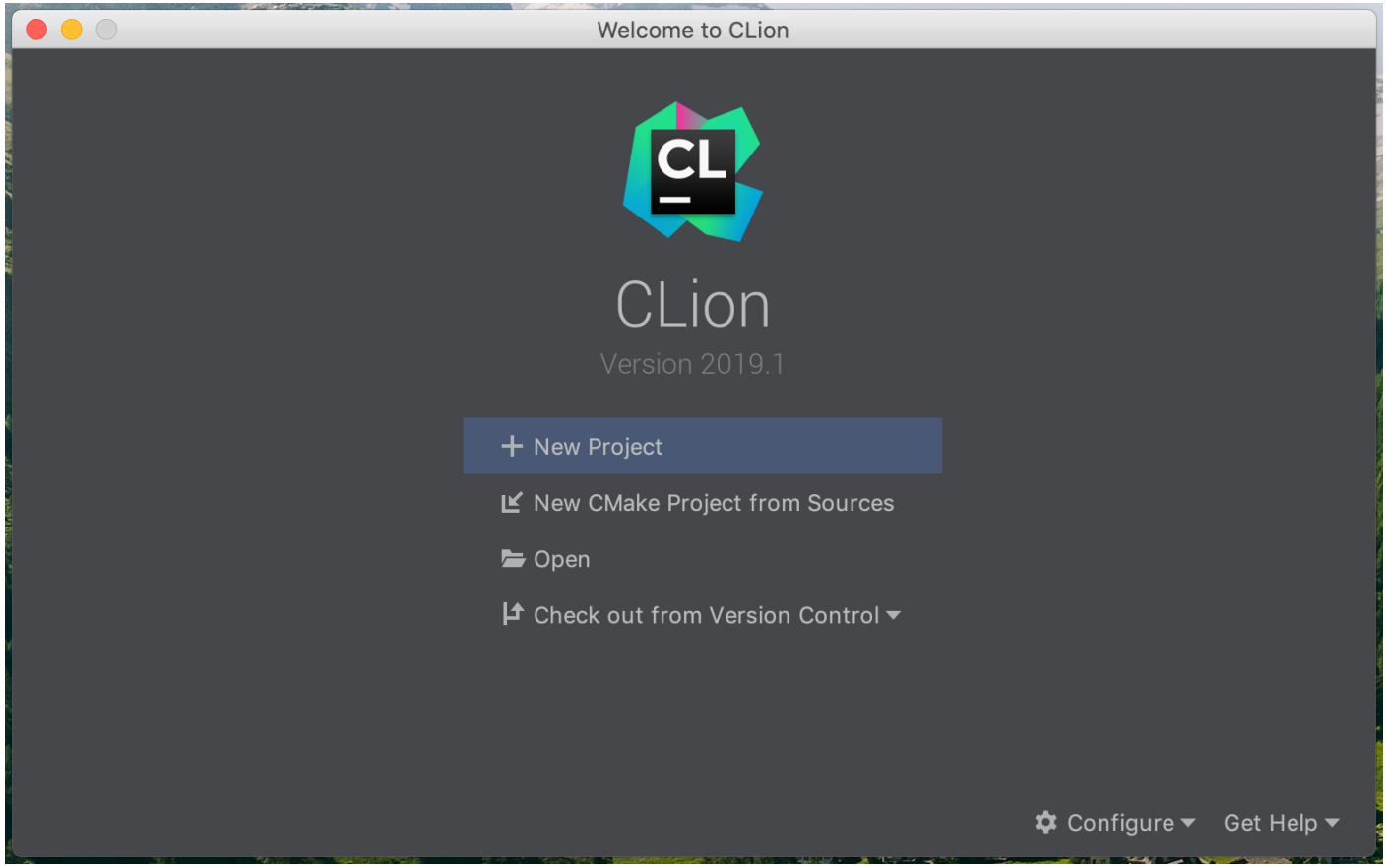
- Great Code Analysis and Refactoring
- JetBrains has a free student license as long as you are enrolled as a student. After that the price is definitely justified.
- Only moderate file size ~ 1GB

Get it here.

However you are free to use any other Editor or IDE. You can also continue to use TextEdit.

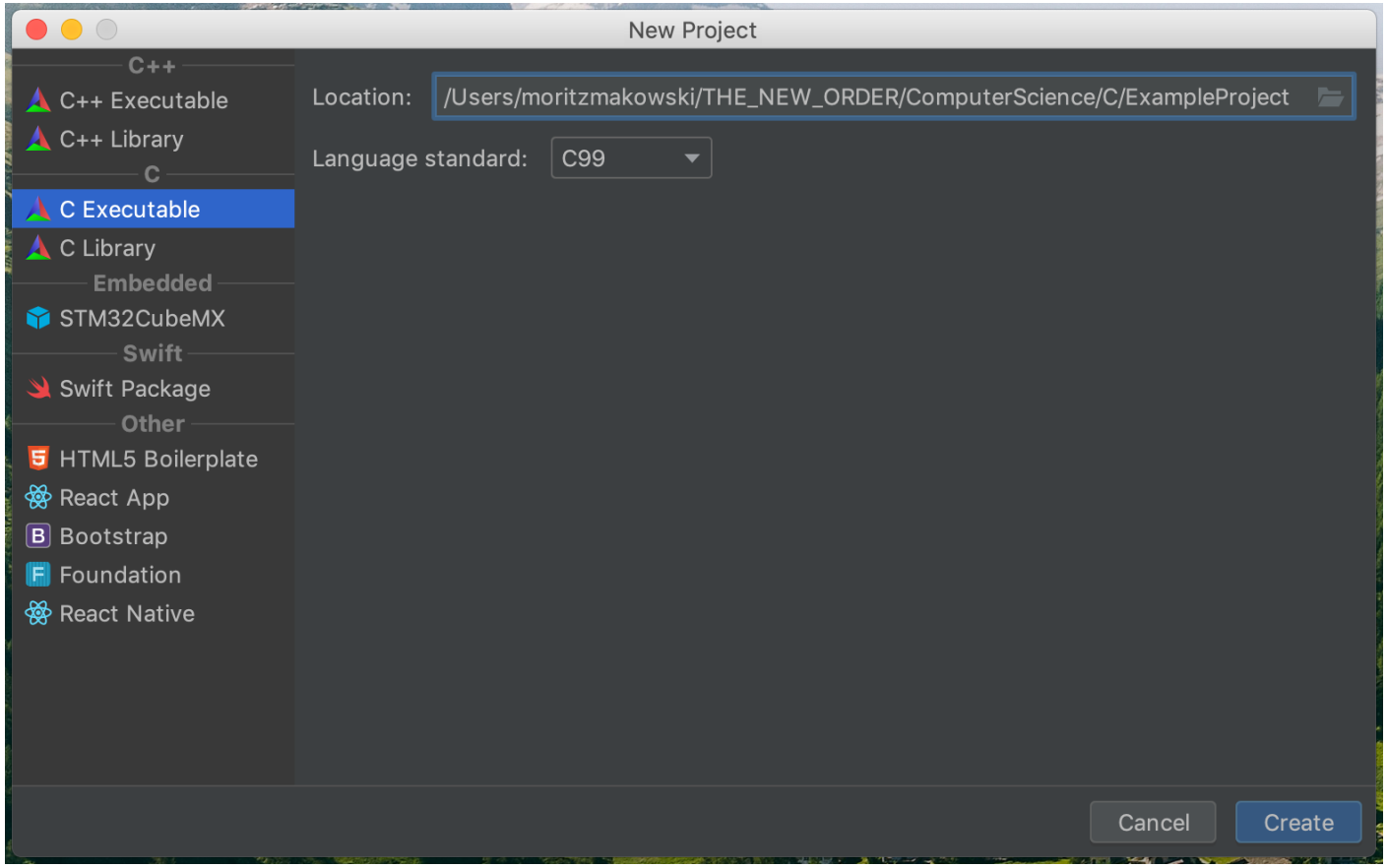
Short Intro to CLion - #1

The first time you open CLion you have to **create a project**.



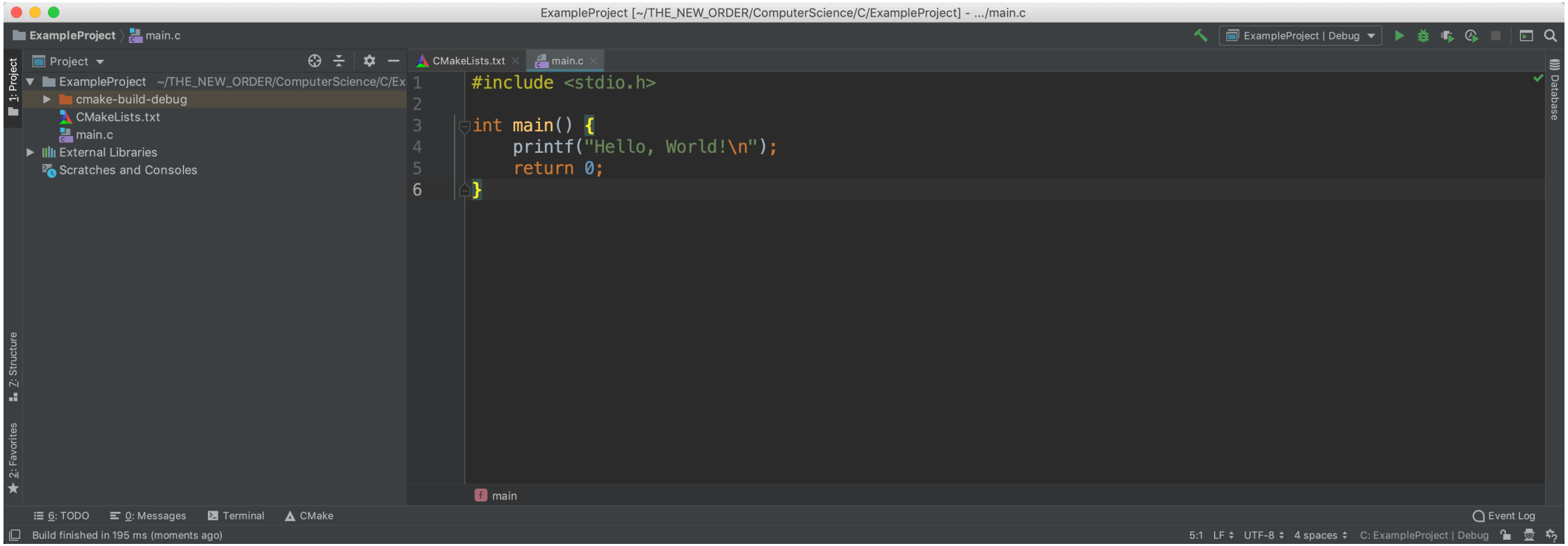
Short Intro to CLion - #2

You now have to specify the **project type** (left column) and a **project directory**.



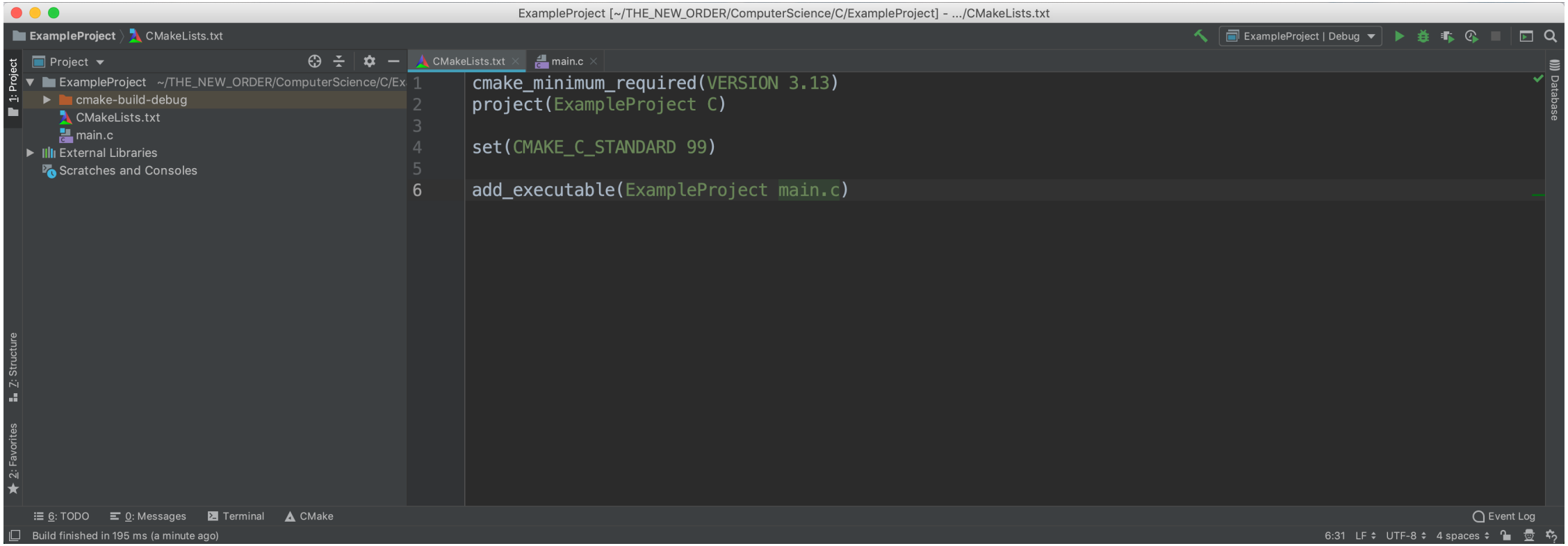
Short Intro to CLion - #3

CLion automatically generates two files called `main.c` and `CMakeLists.txt`.



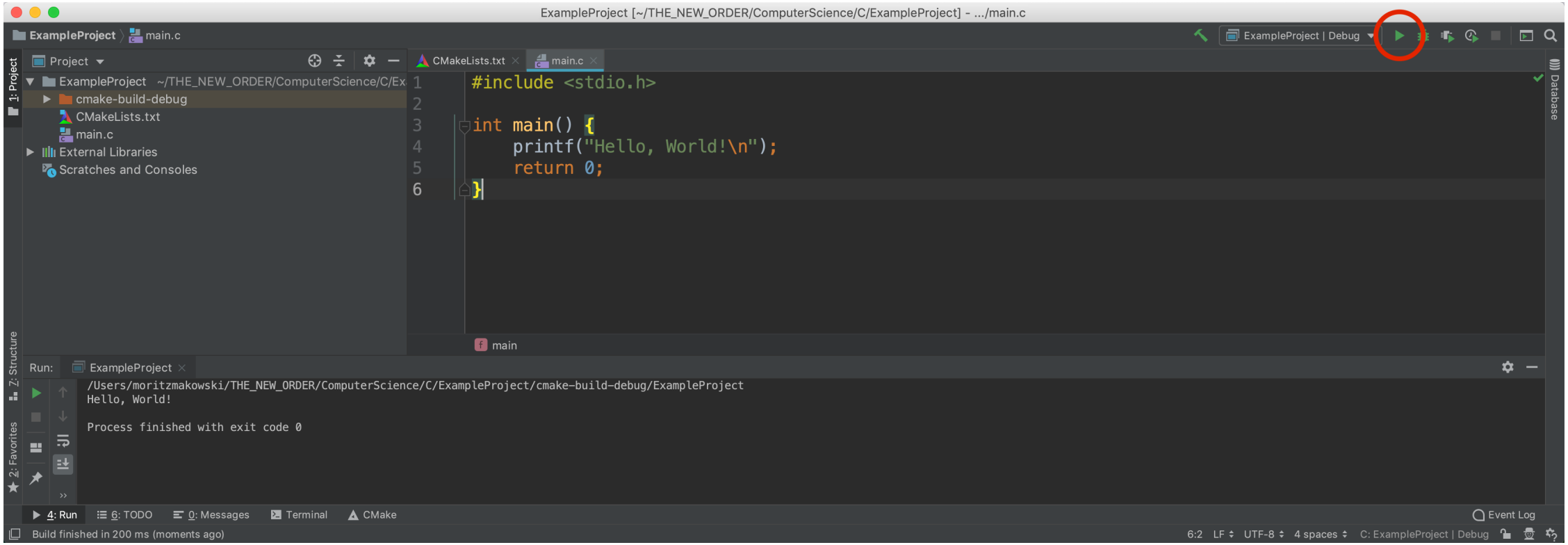
Short Intro to CLion - #4

Inside `CMakeLists.txt` you can specify the files to be compiled.



Short Intro to CLion - #5

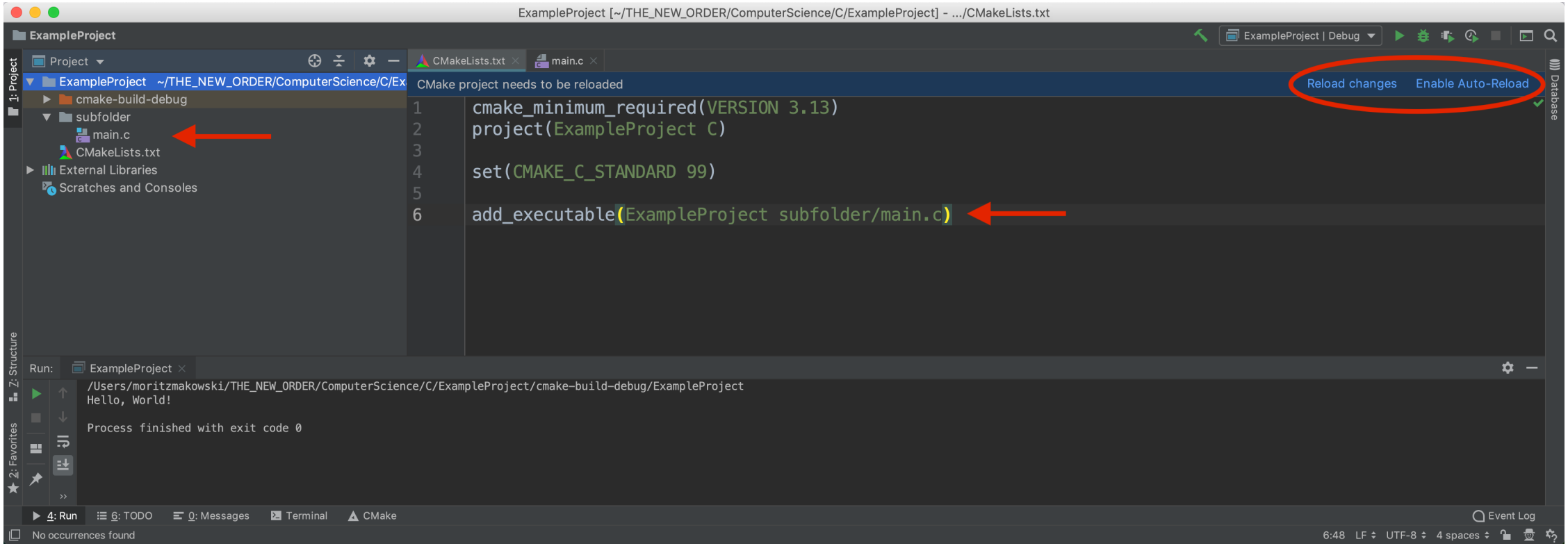
On the top right is a `"Run"-Button` which compiles and executes your program.



However, there will not be a compiled executable file next to you C file.

Short Intro to CLion - #6

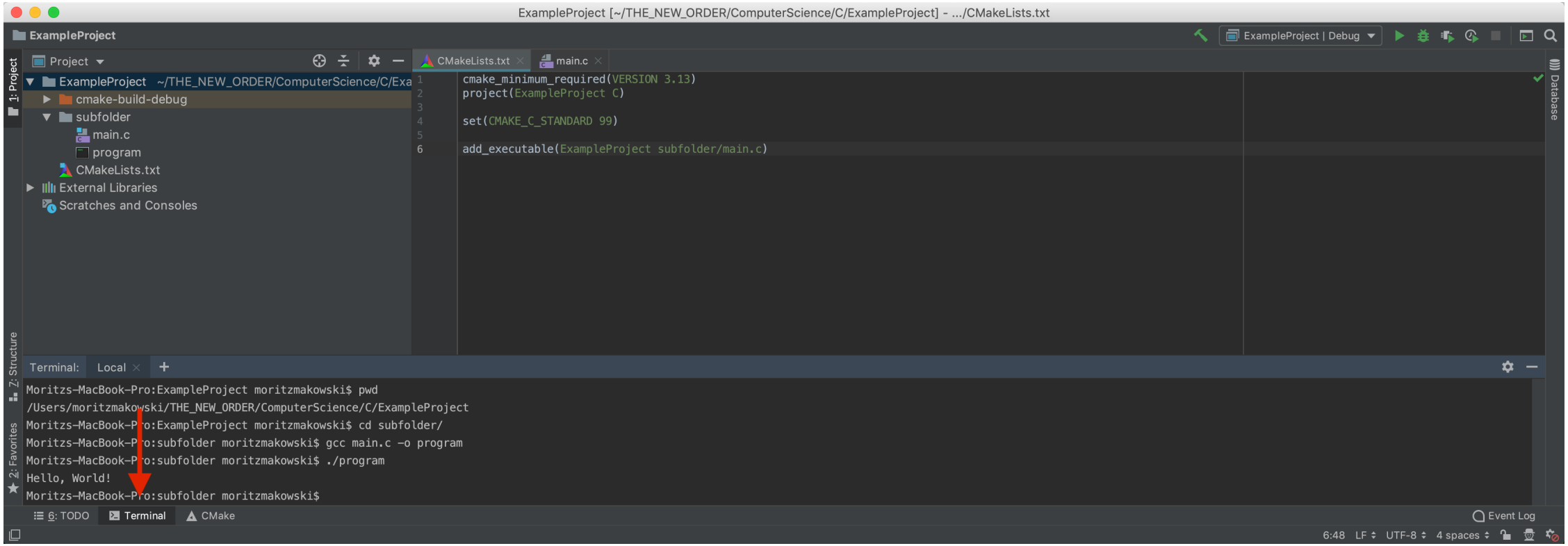
When we change the location of our `main.c` we also have to modify `CMakeLists.txt`.



After changing `CMakeLists.txt` you have to click **Reload Changes** in order to use the mentioned "Run"-Button again.

Short Intro to CLion - #7

There is also a **terminal** inside the IDE which you can use to compile your program.



There will be homework?

Yes! But listen ... Learning theoretically about programming is like learning swimming in theory.

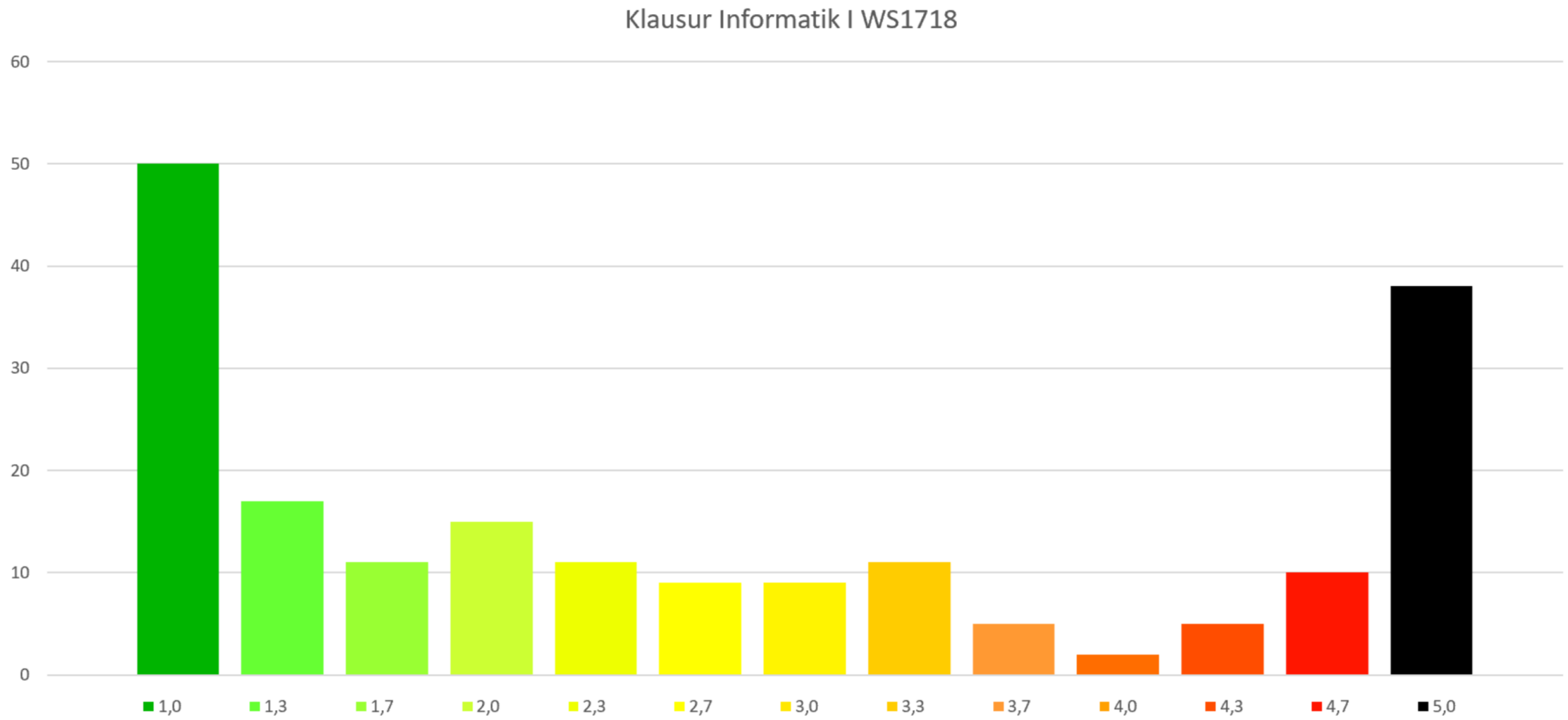
You can only learn how to program by doing it, and to get you started these homeworks are some example exercises so that you have something to actually practice on.

How much homework?

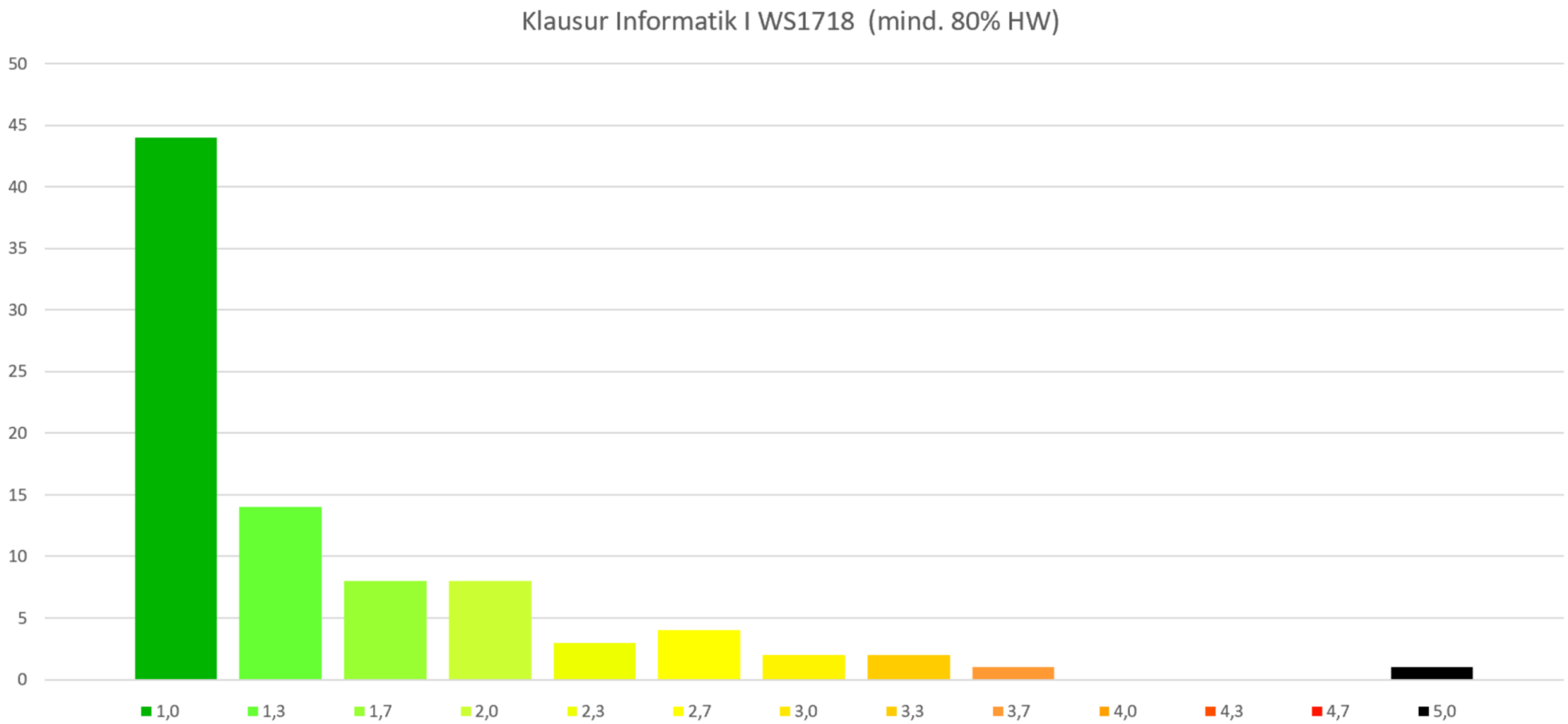
There will be **10 homework sheets**.

You don't have to do these exercises, but we strongly recommend it. If you complete at least 8/10 assignments, you will get a 0.3 bonus on your overall exam grade. However you can not surpass 1.0 and have to pass the exam without the bonus of 0.3.

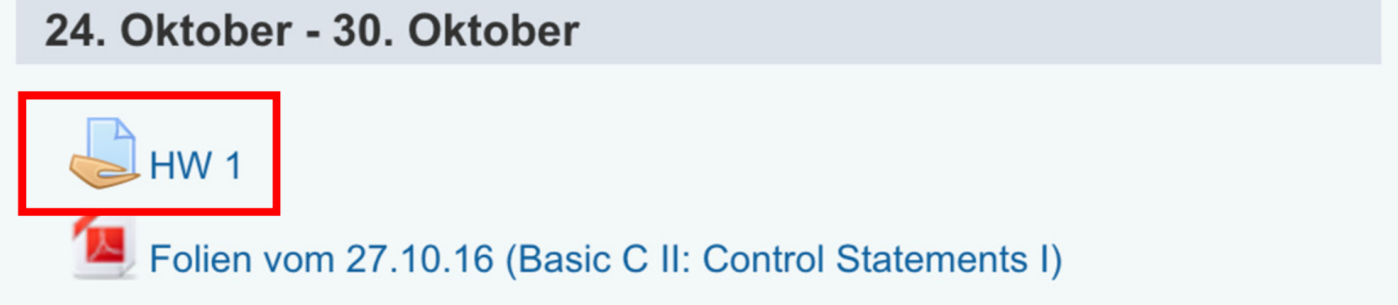
You wan't to know, why we strongly recommend, doing homework assignments?



That's why ...


Klausur Informatik I WS1718 (mind. 80% HW)
50
45
40
35
30
25
20
15
10
5
0
1,0
1,3
1,7
2,0
2,3
2,7
3,0
3,3
3,7
4,0
4,3
4,7
5,0

Once there are homework assignments due, you can upload them on Moodle:



We will correct them as fast as possible and give you a short feedback on the submit page on Moodle. You can ask us in directly for a more detailed feedback. The tutor which is presenting your tutorial will also correct your homework.

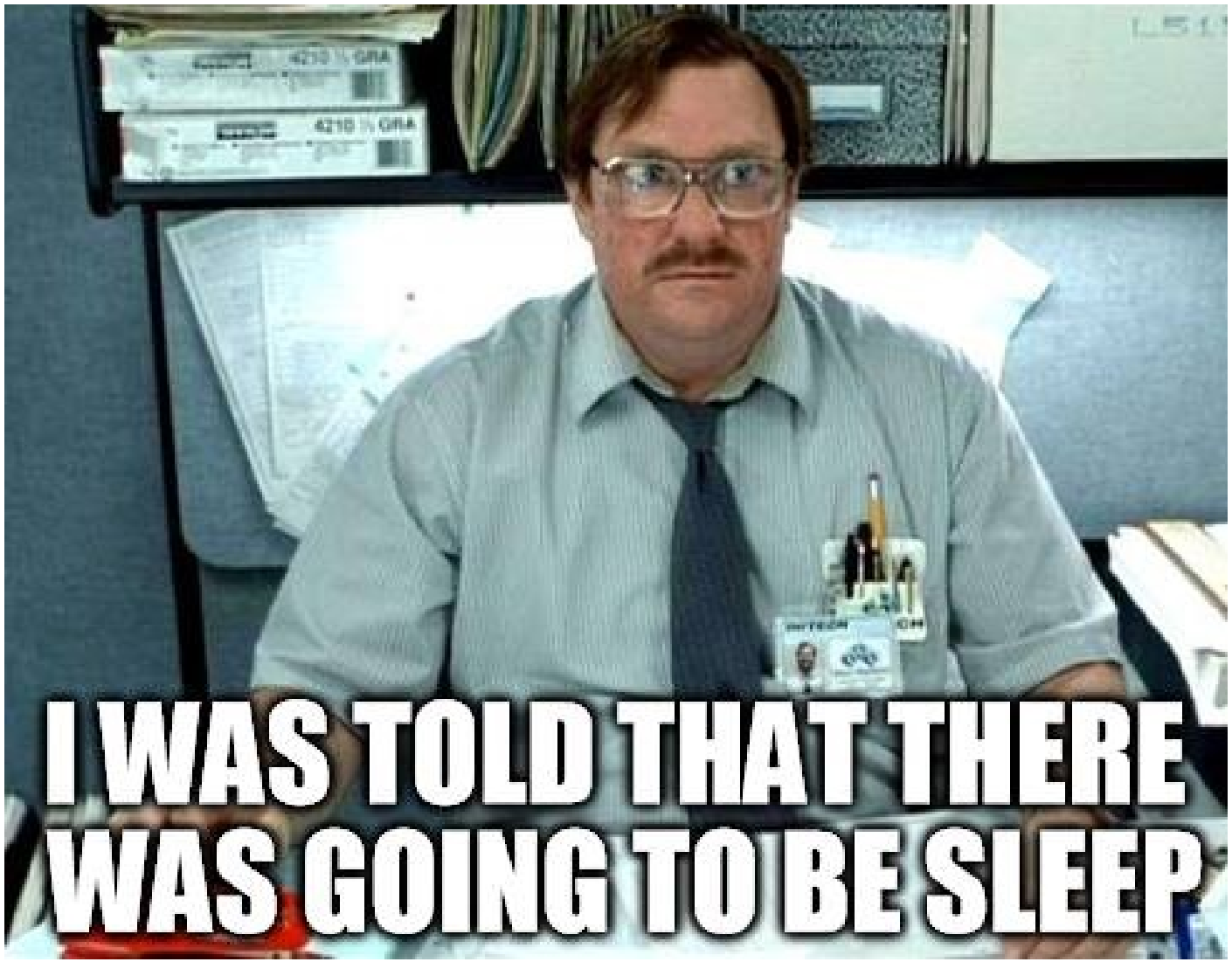
A more detailed instruction on how to submit your homework will follow next week.

Additional Resources

You can find all files related to my tutorial session on my GitHub Respository.

I will upload all my tutorial **presentations** as well as all **code examples** I am using during the session.

Don't worry, you don't have to know how Git and GitHub work to see the files!


I WAS TOLD THAT THERE
WAS GOING TO BE SLEEP