Bern University
of Applied Sciences

394661-FS2018-0 - C++ Programming I

EXERCISE-01

TABLE OF CONTENTS

1 Introduction

The first laboratory exercise of 394661-FS2018-0 will focus on installing and using the Qt Creator IDE for C++ development on your platform. Moreover, you get introduced to CMake - a common Makefile-generator.

You will learn the following topics when completing this exercise:

- Installing and setting up Qt-Creator on your specific platform
- Integration and first use of CMake
- Build and debug simple programs

2 Prerequisites

Install and configure Qt Creator according your needs as required by your platform.

2.1 Tasks

Complete the following list of tasks to set up your environment:

1. Install Qt Creator as described in the slides *lecture_01-GettingStarted_slides.pdf*
2. Install CMake
3. Create a plain C++ project with CMake as build system. It should print 'Hello C++ ' onto the console
   - File ⇒ New File or Project ⇒ Non-Qt Project ⇒ Plain C++ Application ⇒ ..
     ⇒ Build System: CMake
   - Run CMake: This will generate a Makefile
   - Build and run ⇒ `Hello World!`
4. Modify the output accordingly

3 Exercise

Once, you have a working programming environment complete following exercise. Create a new CMake-Project to calculate the Fibonacci-Series and the golden ratio.

3.1 Fibonacci-Series

The sequence F_n of Fibonacci numbers is defined by the recurrence relation:

$$F_n = F_{n-1} + F_{n-2} \qquad f_n \in \mathbb{N} \tag{1}$$

with seed value $F_0 = 0$ and $F_1 = 1$.

The ratio of the consecutive Fibonacci numbers converges to the golden ratio ϕ:

$$\phi = \lim_{n\to\infty} \frac{F_{n+1}}{F_n} = \frac{1+\sqrt(5)}{2} \approx 1.61803399 \tag{2}$$

3.2 Fibonacci-Function

Create a CMake project with a main file and two other files containing the declaration and definition of the Fibonacci function

1. `main.cpp`: The entry point of the program with main shown in the code snippet below
2. `fibonacci.h`: A header file with the declaration of the Fibonacci-Function
3. `fibonacci.cpp`: A source file with the definition of Fibonacci-Function

The declaration and definition of Fibonacci is missing, that means you have to implement Eq (1). The function `fibonacci` takes two `int` values to compute the next Fibonacci value. The declaration and definition of the Fibonacci-Function are put into separate files. Your code is the called from the main file as shown below.

```cpp
#include <iostream> // provides output to stdout with cout
#include "fibonacci.h"  // for function fibonacci (...)

int main ()
{
    int f = 1;
    int fprev = 0;

    std::cout << fprev << std::endl;

    do {
        std::cout << f <<  std::endl;
        int tmp =  fibonacci(f, fprev);

        fprev = f;
        f = tmp;

    } while (true);
    std::cout << std::endl;
}
```

Once your code compiles, verify that the output is similar to: `1 2 3 5 8 13 21 34 55 ...`

3.3 Overflow Detection

The type `int` has an upper limit. When the Fibonacci number exceeds this limit the `int` will silently overflow!

- At which loop iteration number is this the case? Use the debugger! To change the build type to Debug add '`set(CMAKE_BUILD_TYPE "Debug")`' to the CMakeLists.txt.
- What is the upper limit of `int` on your system? Use `std::numeric_limits<int>::max()`!
- Can you modify the loop to stop computing Fibonacci numbers beyond the range of the underlying type?
- What could you do to compute larger Fibonacci numbers?

3.4 Golden Ratio (optional)

How close in % can you calculate the golden ratio as defined in Eq. (2). Calculate the deviation for each iteration. Use `<cmath>` and `<iomanip>` includes for the calculation and manipulation (`std::setprecision(17)`, `std::fixed`) of the output format, respectively. You should get a print out similar to:

```
Ratio:  1.00000000000000000 - Dev[%]:  38.19660175201556029
Ratio:  2.00000000000000000 - Dev[%]:  -23.60679649596888652
Ratio:  1.50000000000000000 - Dev[%]:  7.29490262802333689
Ratio:  1.66666666666666674 - Dev[%]:  -3.00566374664075209
Ratio:  1.60000000000000009 - Dev[%]:  1.11456280322488333
...
..
.
```

4 Submission

Submit your source code (as a zip-file) to Ilias Ex-01 **before the deadline** specified in Ilias.