Exercises for the C++ Refresher

Exercise 1 Editing and compiling a program

First, you should start a fresh terminal and switch to the working directory of this exercise:

```
[user@dune-vm ~]$ cd iwr-course-2021
[user@dune-vm dune-course]$ cd dune
[user@dune-vm dune]$ cd dune-pdelab-tutorials
[user@dune-vm dune-pdelab-tutorials]$ cd c++
[user@dune-vm gridinterface]$ cd exercise
[user@dune-vm exercise]$ cd task
```

This is the source directory, so you cannot use `make` to build the programs in this exercise. Instead, you will invoke the compiler manually to understand how it works.

In order to compile an exercise program (in this case `task1.cc`), run the following command (note the special flag to switch the compiler to C++14 mode):

```
[user@dunevm task]$ g++ -Wall -std=c++14 -o task1 task1.cc
```

This command will attempt to compile the source code in `task1.cc` and create an executable called `task1` in the current directory.

However, the compiler does output an error message and fails to create the program. Try to understand the compiler error, open the source file in an editor and fix the problem. Afterwards, try to compile and run the program again.

In this case, the error is due to the fact that the compiler sees two functions called `min()`. How can you fix this problem? If you get stuck, you can look at the solution in the `solution` directory next to the `task` directory. You can get there by entering `cd ../solution`.

Exercise 2 References

The second program, `task2.cc`, calculates the cubes of numbers stored in a vector, but it does not work correctly, either.

Look at the program to understand its intended mode of operation. Why do the values in the vector not get updated with the cubed values? Fix the error! (Hint: Take a look at the slides about references).

Exercise 3 Templates

In the third program, we also want to calculate the cubes of a vector of floating point values. However, our current `cubed()` function only works with integers.

Run `task3.cc` and check the calculated cubes. Are they correct?

In order to fix this problem, rewrite the function `cubed()` as a template function (see the lecture for a simple example).

Exercise 4 Lambda Functions

In the final exercise, you will convert normal functions and functors into lambda function. The program creates a vector of floating-point numbers and calculates the squares of those numbers. Afterwards, the numbers are scaled by a fixed factor. These operations are encoded in a template function and a functor. Writing these adds additional boilerplate and requires you to switch between different location in your source. Replace both the function `squared()` and the functor `Scaled` with lambda functions!