2EL5030 – Programming efficiently in C++

Instructors: Frederic Pennerath
Department: CAMPUS DE METZ
Language of instruction: ANGLAIS
Campus: CAMPUS DE METZ
Workload (HEE): 60
On-site hours (HPE): 35,00
Elective Category : Fundamental Sciences
Advanced level : No

Description

Knowing how to code an algorithm effectively in a given programming language requires a prior understanding of the associated calculation model and how the instructions in that language are translated into machine instructions. Too many students still approach programming in a superficial and risky way, lacking the basic knowledge necessary to write elegant and effective code.
The unique strength of the C++ language is to allow the production of compiled codes close to the optimal machine code while offering different high-level programming approaches such as strong typing, object programming, functional programming and meta-programming (automatic code generation at compilation). For this reason, C++ has become the essential language for developing optimized algorithms. Its only disadvantage is its richness, which has continued to grow in its most recent versions (C++11/14/17/20) and which makes it difficult to understand the language in its entirety without adequate training.
This course is intended for students, including beginners, who want to master the different aspects of C++ programming in order to be able to write code that combines performance and elegance. The course adopts a bottom-up approach starting from the mechanisms of elementary program execution and gradually moving towards the most advanced language functionalities.
The objective is to transmit to the students a real know-how of programming, on the one hand by illustrating the concepts through relevant examples of use, and on the other hand by devoting a significant part of the hourly volume to laboratory work.

Quarter number

SG8

Prerequisites (in terms of CS courses)
No specific requirements are expected, but a basic experience of programming.

Syllabus

Total course length: 15h of lectures, 18h of labeworks and 2h of exam

Lectures (15h):

- Memory management and variable life cycle (1h30)
- Struct and smart pointers (1h30)
- Types (1h30)
- Functional programming: lambda functions, callable types, exceptions,... (1h30)
- Class and inheritance (1h30)
- Basic mechanisms of templates (1h30)
- Generic programming based on templates (1h30)
- Generic versus object-oriented programming (1h30)
- Standard library: content and concepts (1h30)
- System programming: threads and synchronisation mechanisms (1h30)

Labworks (18h):

- TP on memory manipulation (3h)
- TP on object-oriented programming and inheritance (3h)
- TP on functional programming (3h)
- TP on generic programming (3h + 1h30)
- TP on the standard library (1h30)
- TP on system programming (3h)

Practical exam (2h)

Class components (lecture, labs, etc.)

- Courses based on code examples
- Significant part (50%) devoted to practical programming work

Grading
The assessment will be based on a single examination scheduled at the end of the course.
This exam will last two hours and will consist of short programming exercises. Each student will work alone on his or her own computer without access to the Internet, but will be provided with all the necessary

documentation to pass the exam (course materials, technical documentation).
The modalities of remedial examination will be identical to the ones of the initial examination.

Course support, bibliography

- Website provided by teachers
- Effective Modern C++, Scott Meyers, 2014
- Professional C++, Marc Gredoire, 2014
- A Tour of C++, Bjarne Stroustrup, 2013

Resources

- Two teachers: Hervé Frezza-Buet and Frédéric Pennerath
- No tutorials but only practical work on the machine
- One workstation per student
- One or two labwork groups of 15 students maximum each.
- The labwork will be carried out in the Linux environment and will be based exclusively on free software (g++, CMAKE, etc.).

Learning outcomes covered on the course

- To know how to write a program in C++ using different programming paradigms such as object programming, functional programming and generic programming.
- To know certain aspects of the C++ language that have a decisive influence on the performance of programs during their execution.
- To know the functionalities offered by the most recent specifications of the C++ language (C++11, C++14, C++17 and C++20).
- To know how to use a C++ compilation and debugging environment

Description of the skills acquired at the end of the course

- C2: Develop a deep competence in an engineering field and in a family of professions.
- C6: Be operational, responsible and innovative in the digital world.