Summary
The aim of this course is to provide the background in scientific computing. The class includes a brief introduction to basic programming in C++, it then focus on object oriented programming and C++ specific programming techniques.

Content
• Flow control, I/O
• Pointers
• Blocks, functions, variables
• Classes, derivation and inheritance
• Templates
• Linear algebra
• Basics of parallel programming

Learning Prerequisites
Required courses
Analysis I and II
Linear Algebra
Numerical Analysis
The course Numerical Analysis and Computational Mathematics has to be followed in parallel to the course if its contents are not yet mastered.

Recommended courses
A programming language (C, C++, Fortran, Java, ...)
Introduction to the Finite Element Method.

Learning Outcomes
By the end of the course, the student must be able to:
• Interpret algorithms in C++
• Modify algorithms in C++
• Implement algorithms in C++

Transversal skills
• Assess progress against the plan, and adapt the plan as appropriate.
• Set objectives and design an action plan to reach those objectives.
• Use both general and domain specific IT resources and tools
• Give feedback (critique) in an appropriate fashion.

Teaching methods
Interactive lecture and projects in classroom

Expected student activities
Before each class the student is required to prepare with assigned reading. Programming assignements during the project hours and at home.

Assessment methods
The students will be evaluated with two quiz (QCM) during the semester and then with oral evaluations based on programming at the end of the semester.

Resources
Bibliography
Other references:
Parallel scientific computing in C++ and MPI, Karniadakis, G. and Kirby, R.M., Cambridge University Press, 2003

Ressources en bibliothèque
• C++ and Object Oriented Numeric Computing for Scientists and Engineers / Yang
• Guide to Scientific Computing in C++ / Pitt-Francis
• Parallel scientific computing in C++ and MPI /Karniadakis

Notes/Handbook