Part I Courses

Module 1.1:	**Scientific Programming Environment**
Coordinator	Alberto Sartori (SISSA) Nicolas Salles (University of Nova Gorica)
Module Description	This course will introduce Unix-like operating systems, show how to setup the scientific programming environment in such operating systems. It will present the modern software tools required to provide such an environment and discuss important points like documentation and testing.
Main Topics	• Introduction to Unix-like operating systems (kernel vs. userspace, processes/threads, file system semantics) • Shell scripting (bourne shell) • Basics of python programming • Review of basic concepts of C and Fortran programming language, and mixing thereof • Compiler architecture (preprocessor, compiler, assembler, linker) and use of libraries • Build automation tools (make and CMake) • Source code management (git) • Unit and regression testing • Software documentation
Objectives	On successful completion of this module students should have their own software environment and tools prepared and configured for the rest of the activities of the Master Program. Students learn the workflows of software development and working collaboratively. These concepts are the basis on which to develop efficient scientific code for high-performance computing applications.

Module 1.2:	**Foundation of HPC**
Coordinator	Stefano Cozzini (CNR-IOM) Luca Tornatore (DSSC)
Module Description	Introduction to key topics in computer architecture needed in HPC environment including a detailed overview on parallel architectures.
Main Topics	Foundation of HPC • Computer architecture • Memory hierarchy • Modern multi-core CPU systems • Overview on massively parallel processors • Parallel architectures for HPC and future trends • Benchmarking and energy efficiency in HPC • Profiling of a serial applications • False sharing and memory efficiency on Multi-core systems • Practical introduction to software optimization
Objectives	On successful completion of this module students should be able to understand basic concepts of modern computer architecture, and know the different kinds of parallel architectures commonly applied to scientific research.

Module 1.3:	**Introduction to Parallel Computing**
Coordinator	Ivan Girotto (ICTP)
Module Description	Introduction to key topics in parallel computing. Main parallel programming paradigms: message passing (MPI) and multi-threading (OpenMP).
Main Topics	● Introduction to parallel computing ● Overview of Fundamental Approaches to Parallel Programming ● Message-Passing Parallelization (MPI) ● Shared Memory Parallelization (OpenMP) ● Analysis of Scalability and Parallel Performance Metrics ● Introduction to CUDA Programming and Accelerated Computing
Objectives	On successful completion of this module students should be able to write parallel programs and know fundamental techniques of parallel programming to develop parallel applications along with methods of parallel performance analysis

Module 1.4:	**Advanced Programming**
Coordinator	Alberto Sartori (SISSA) Jacopo Rigosa (SISSA)
Module Description	Provide advanced knowledge of both theoretical and practical programming in C++ and Python, with particular regard to the principles of object oriented programming and best practices of software development (advanced use of version control systems, continuous integration, unit testing).
Main Topics	● Scientific programming environment (*nix) ● Introduction to C++: the logic, the basics, the built-in data types and the best practices ● Object-Oriented programming in C++ ● Advanced features of C++11/14/17 ● basics of Python ● Object-Oriented programming in Python
Objectives	On successful completion of this module students should be able to understand modern software engineering and design principles and should develop fundamental programming skills in the context of a language that supports the object-oriented paradigm.

Module 1.5:	**Introduction to Numerical Analysis**
Coordinator	Luca Heltai (SISSA) Gianluigi Rozza (SISSA) Francesco Ballarin (SISSA) Nicola Giuliani (SISSA)
Module Description	Introduction to numerical analysis, with focus on linear algebra, polynomial approximation, numerical integration and numerical solution of ODEs
Main Topics	• The foundations of numerical analysis • Polynomial Interpolation • Numerical Integration • Resolution of non-linear systems • Resolution of large linear systems • Eigenvalues approximation • Numerical solution of ODEs • Numerical solution of PDEs
Objectives	On successful completion of this module students should be able to understand and implement in computer program numerical integration, numerical derivation, functional interpolation and basic linear algebra operations.

Module 1.6:	**Advanced linear algebra libraries and accelerators**
Coordinator	Piotr Luszczek (University of Tennessee, USA)
Module Description	Large supercomputing installations and scientific clusters will be discussed with the emphasis on their architectural features that are essential for good performance and scalability. Although majority of the presented code will be explained, familiarity with programming is welcome and will be helpful in following along.
Main Topics	● The programming portion of the course will cover open community standards including OpenMP, OpenACC, and OpenCL as well as CUDA programming syntax for the modern GPUs. These language extensions will be used to implement components of numerical libraries for linear algebra. ● The specific computing scenarios that will be covered during the course will include dense, sparse, and batch variants of the BLAS and LAPACK routines. ● Additionally, the distributed memory implementation will be introduced for multicore clusters with multiple GPU accelerators per node. This will require the use of some of the more advanced features of MPI but they will be described with sufficient detail during the lectures to underscore their implementation in modern MPI libraries. ● Finally, the benchmarking aspect of numerical libraries will be presented and analyzed based on a number of HPC benchmarks: HPL, HPCC, and HPCG.
Objectives	One of the main objectives of this course is to introduce design, implementation, and optimization of High Performance Computing libraries for numerical linear algebra. The broader hardware landscape will be presented to give an appropriate context for the programmer of these libraries.

Module 1.7:	**Best Practice in Scientific Computing**
Coordinator	Axel Kohlmeyer (Temple U. & ICTP)
Module Description	A module where students are introduced to best practices in scientific computing from different perspective: software development with modern software engineering techniques, optimal exploitation of different HPC platforms, usage and maintenance of large scientific software packages.
Main Topics	● Debug versus optimized mode ● Multilanguage programming ● Strategies for developing scientific codes ● Collaborative ways of developing scientific and technical packages. ● Tools for developing large software packages
Objectives	On successful completion of this module students should have a clear idea about the most successful practices that can be adopted in developing technical and scientific software and make them run efficiently on modern HPC platform.