On-line course materials

MATH49111 - Scientific Computing

Year: 4 - **Semester:** 1 - **Credit Rating:** 15

Aims

To develop the basic knowledge required to translate some common mathematical concepts used in scientific problem solving into an object-oriented programming language (in this case C++). Students will use a combination of writing their own codes, together with the use of scientific libraries (such as NAG).

Brief Description

The course will teach the syntax and logical structure of C++ programming and object-oriented development with no assumed prior knowledge. The emphasis is placed on the implementation of common mathematical tasks/algorithms in C++. The students must select two miniprojects from a list of available topics in applied maths. The projects will be assessed by a written report and a demonstration/ oral description of the code.

Only a limited number of places are available on this course.

Learning Outcomes

On successful completion of this module students will be able to

- understand the basic structure, content and syntax of a C++ program,
- appreciate the generic concepts underlying the object oriented programming model,
- develop a C++ program that solves a problem in physical applied mathematics,
- debug a C++ program and validate results in the context of a mathematical problem.

Syllabus

- Introduction to C++ programming language.
- Compiling/running/debugging programs.
- Data types - initialisation, scope, precision, input/output.
- Using functions and control structures to reproduce simple algorithms.
- Intrinsic and user-defined functions; call by value, call by reference, the const qualifier.
- Discussion of exception and error handling.
- Using the standard libraries in C++; vectors, lists, sets, maps, iterators, sorts, search, transforms.
- How to link external libraries such as the NAG routines.
- Discretisation of ODE/PDEs
- Object Orientated programming:

- constructors, destructors, methods, member data, public/private qualifiers

- public/private qualifiers, operator overloading

- inheritance; the protected qualifier, virtual methods and run-time polymorphism.

Some possible projects:

- ODE problems
- PDE problems
- Financial problems
- Biological problems
- Fluid Dynamics problems

Students on the Theoretical and Applied Fluid Mechanics MSc are required to choose their projects from the first 4 options on differential equations and continuation methods.

Teaching & Learning Process (Hours Allocated To)

Lectures	Tutorials/ Example Classes	Practical Work/ Laboratory	Private Study	Total
11	0	33	106	150

Assessment and Feedback

- Weekly courseworks: 10%
- Mini-Project 1: 40%
- Mini-Project 2: 50%

Further Reading

- G.D. Smith, Numerical Solution of Partial Differenctial Equations, Clarendon Press, Oxford, 1978.
- D. Alcock, Illustrating C, Cambridge University Press, 1992.
- D. Yang, C++ and object-oriented numeric computing for scientists and engineers, Springer, 2000.
- S. Meyers, Effective C++: 55 specific ways to improve your programs and designs, Addison-Wesley, 2005.
- T.J. Chung, Computational Fluid Dynamics, CUP, 2002
- Y. Saad, Iterative methods for sparse linear systems, 1996, PWS series in computer science.

Data source is EPS system

Back To Top