Master's Department of Engineering Sciences

Name of the unit: C++ Object Programming

Lecturer: Alice Cohen-Hadria (alice.cohenhadria@ircam.fr)
Course code: UM4RBI22
Student workload: 8h of lectures, 4h of tutorial sessions, 12h of lab sessions@
Credits: 3 ECTS

Specialization tracks:	- **E3A** :	☐ CIMES	☐ Syscom	☐ IPS
	- **AR** :	☐ ROBOT	☐ SI	☐ MeDH
	- **MECA** :	☐ MS2	☐ MF2A	☐ CompMech
		☐ ACOU	☐ EE	☐ EE APP

Semester offered: ☐ S1 ☒ S2 ☐ S3 ☐ S4
Language of instruction: ☐ French ☐ English
Targeted audience: ☐ Engineering Sciences students ☐ Other (specify):
Localization : ☒ PMC Campus ☐ Other (specify):

Course Overview

This course trains students in advanced object-oriented programming in C++. The fundamentals of inheritance, encapsulation, and object polymorphism are covered first. Next, various design patterns are discussed.
Keywords: Object-oriented programming, inheritance, polymorphism, encapsulation, template, design pattern.

Prerequisites

Students should have previously acquired the following prerequisites to follow this course:

- Write a simple program using variables, control structures within classes, and methods.
- Test its functions and classes with different test scenarios described.
- Converting from a simple UML diagram (single class) to the corresponding program, and vice versa.

Intended Learning Outcomes

By the end of this course, students will be able to:

1. Create a test program with a description of the tests performed and their results.
2. Create and compile a makefile.
3. Knowing how to use the shell (navigating, creating, copying, listing, deleting, searching, user rights).
4. Write the documentation for a function (/** */) and generate the corresponding Doxygen file.
5. Using precompiled code with documentation.
6. Moving from a statement describing a situation to a UML class diagram, then to the corresponding code (and vice versa).
7. Factoring code (creating parent classes from existing classes).
8. Knowing how to implement a design pattern from a statement.
9. Knowing how to implement inheritance and associated polymorphism mechanisms.
10. Knowing how to use templates.

Indicative Teaching Sequence and Methods

Week	C/TD/TP*	Content	Preparation	Learn. outc.
S1	C1 /TP1 (2h)	Introduction to C++ and general information: compilation, makefile, shell, documentation		AAV1, 2, 3, 4
S2	C2	Inheritance, Polymorphism, Encapsulation		AAV9, 6, 7, 1, 4
S3	TD1 /TP2 (2h)	on C2		
S4	C3	Template, libraries		AAV 10, 5, 1, 4
S5	TD2/TP3 (2h)	on C3		
S6	TP Solo 4 (2h)	on TP1 and 2: 30-minute quiz, 30-minute solo TP		
S7	C4	Design pattern		AAV8, 1, 4
S8	TP5 (4h)	On C4 + project preparation		
S12	TP6	Project defense		
S15	Exam	Final written exam covering the entire EU		

* C/TD/TP respectively corresponds to lectures, tutorials and lab sessions.

Sequence of the unit:
The first practical assignment follows directly after lecture C1 and focuses on learning how to use makefiles and shells.
Then, after each lecture, the tutorial allows students to put the concepts covered in class into practice with simple exercises, while the practical assignments allow students to code independently.

Indicative Assessment of Intended Learning Outcomes (1st session)

Week	Individ./group	In-person/remote	Type of exam	Evaluated outcomes	Scale %
S6	Individual	In-person	Labs	1,2,4,6,7,9	20 %
S12	Collective	In-person	Other	8,9,10,1,2	30 %
S15	Individual	In-person	Written	1,2,3,4,5,6,7,8,9,10	50 %

2nde session of exam

Session	Individ./group	In-person/remote	Type of exam	Evaluated outcomes	Scale %
2	Individual	In-person	Written		50 %
1	Individual	In-person	Labs	1,2,4,6,7,9	20 %
1	Individual	In-person	Other	8,9,10,1,2	30 %

Date of generation of this unit description: January 14, 2026