UCLouvain

lsinc1402

2023

Computer Science 2

5.00 credits	30.0 h + 30.0 h	Q1

This learning unit is not open to incoming exchange students!

Teacher(s)	Jodogne Sébastien ;Sadre Ramin ;
Language :	French
Place of the course	Charleroi
Prerequisites	*The prerequisite(s) for this Teaching Unit (Unité d'enseignement – UE) for the programmes/courses that offer this Teaching Unit are specified at the end of this sheet.* *The prerequisite(s) for this Teaching Unit (Unité d'enseignement – UE) for the programmes/courses that offer this Teaching Unit are specified at the end of this sheet.*
Main themes	- Introduction to Java: compilation, byte-code, virtual machine, primitive type, strings, tables - Abstract data types; - Linear and tree structures, and their applications; - recursive solution formulation and recursive algorithms; - reasoning technique: preconditions, postconditions, invariants - Notions of computational complexity and analysis of the temporal and spatial complexity of an algorithm; - Functional programming and higher order programming - Object-oriented modeling (inheritance, composition, reuse, polymorphism, class invariant); - Introduction to design patterns; - Program testing and validation methods.
Learning outcomes	**At the end of this learning unit, the student is able to :** AA 1.2, 1.5-AA 2.4 - AA 4.3, 4.4, 4.5 - AA 5.1, 5.3 With regard to the AA reference of the program "Bachelor in Engineering Sciences, orientation civil engineer", this course contributes to the development, acquisition and evaluation of learning outcomes. following: - AA 1.2, 1.5 - AA 2.4 - AA 4.3, 4.4, 4.5 - AA 5.1, 5.3 <u>More specifically, at the end of the course, the student will be able to:</u> - make a justified choice between several representations of information and several algorithms to process them, - design (fragments of) programs in a functional style, - reasoning about (fragments of) programs: complexity of algorithms and efficiency of programs implementing them, recursive reasoning, - apply object-oriented modeling principles, - design and apply methods for testing a program, - design a simple parallel program Students will have developed methodological and operational skills. In particular, they will have developed their ability to: - Analyze a medium-sized problem, propose a computer solution to solve it and implement it in the Java language.
Evaluation methods	An exam on INGInious takes place at the end of the term and aims to not only verify knowledge of the material but also the ability to apply the acquired knowledge to write programs. An optional individual quiz is held halfway through the term, counting for a maximum of two points in the final course grade, and only if it benefits the student. The use of generative AI tools and any collaboration is strictly prohibited during this quiz. The final grade is calculated using the formula: final_grade_out_of_20 = max(quiz_out_of_2 + exam_out_of_18, exam_out_of_20). The points earned from the quiz will be carried forward for the August session. In case of suspected plagiarism (for example, detected using a plagiarism tool), the course instructors reserve the right to ask the student to take an oral examination on the quiz or the exam.

Teaching methods	Instructors will introduce the content in a lecture setting during a traditional class, and it will be outlined in the syllabus titled "Programming Concepts in Java". For each topic covered, exercises are continuously available on the INGInious platform, allowing the application of theoretical concepts. During the practical workshop sessions, assistants or tutors will be present to support and guide the students. Each student is fully responsible for their own learning. To succeed in the computer exam, it is essential for the student to practice regularly in Java using the IntelliJ tool.
Content	This teaching unit focuses on: - Introduction to Java: compilation, byte-code, virtual machine, primitive type, strings, tables - Abstract data types; - Linear and tree structures, and their applications; - recursive solution formulation and recursive algorithms; - reasoning technique: preconditions, postconditions, invariants - Notions of computational complexity and analysis of the temporal and spatial complexity of an algorithm; - Functional programming and higher order programming - Object-oriented modeling (inheritance, composition, reuse, polymorphism, class invariant); - Introduction to design patterns; - Program testing and validation methods; - Introduction to parallelization: notion of thread and synchronization mechanisms. Students who have successfully completed this course will be able to • to design Java programs • analyze programs according to their performance • to prove correctness of programs using invariants • apply the principles of object-oriented programming such as genericity, abstraction, composition and reuse • design and implement variants of the algorithms studied in high quality Java programs. • design and manipulate simple linear and tree and recursive structures • design tests for programs • design functional programming approaches to solve small algorithmic problems • use design patterns • design simple parallel programs with synchronization mechanisms
Inline resources	https://lepl1402.readthedocs.io/ (links to the slides and other course contents) https://moodle.uclouvain.be/course/view.php?id=3831 Moodle et/ou Teams (communication with the students) https://inginious.info.ucl.ac.be/course/LEPL1402 for the exercises
Other infos	Background: LSINC1101 or a similar course.
Faculty or entity in charge	SINC

Programmes containing this learning unit (UE)

Program title	Acronym	Credits	Prerequisite	Learning outcomes
Bachelor in Computer Science	SINC1BA	5	LSINC1101	🔍