LSINF1151

2010-2011

Problem solving using computers

7.0 credits	0 h + 60.0 h	2q

Teacher(s) :	Mens Kim ;
Language :	Français
Place of the course	Louvain-la-Neuve
Inline resources:	> http://www.icampus.ucl.ac.be/claroline/course/index.php?cid=LINF1151
Prerequisites :	introduction to programming in a high-level language (e.g. SINF1160, SINF1161)
Main themes :	-- first experience of building object-based software allowing them to develop a simple application using an IDE (Integrated Development Environment) -- programming in a high-level language -- unit tests -- application quality: correct, modular, readable, documented -- object-oriented programming: classes, interfaces, inheritance, polymorphism, ... -- collaboration between developers
Aims :	Students completing successfully this course will be able to -- perform a fairly small application, that id correct, modular, readable, and well documented; -- put into practice the concepts of object-oriented programming such as data encapsulation, separation of implementation and behavior, classes, subclasses, interfaces and inheritance, polymorphism, class hierarchy , the collection classes, iteration protocols, etc. -- understand and extend an existing application that will be provided as a starting point for new developments; -- implement unit tests to verify program correctness; -- structure an application in a modular fashion into classes and methods; -- analyze the needs for a simple extension to this application. Students will have developed skills and operational methodology. In particular, they have developed their ability to: -- program effectively in a high-level language (Java); -- use a tool such as JUnit for unit testing; -- use a programming environment like Eclipse which includes integrated programming tools (smart editor, compiler, debugger), and tools for file management, testing, documentation, re-engineering; -- argue the design choices regarding the features of the implemented solution and comparing them with alternative solutions; -- collaborate with other (more or less experienced) developers through a discussion forum. *The contribution of this Teaching Unit to the development and command of the skills and learning outcomes of the programme(s) can be accessed at the end of this sheet, in the section entitled “Programmes/courses offering this Teaching Unit”.*
Evaluation methods :	Different grades contribute to final grade: -- Two interim evaluations during the semester, to estimate the progress of the programming project, and based on reports written by students and the code quality -- A final evaluation at the end of the project, including a summary report and an oral defense in which each pair defends its programming project and demonstrates the proper functioning of the application. As this is a programming laboratory, the absence at an interim evaluation or non submission of required documents, involves a final score corresponding to an absence for the course. In case of no success in the first session, the student must individually complete and improve the same programming project, including extensions requested and add an additional non-trivial extension to the project. The final score for the September session will be based on (and only on) the quality of the entire code and its documentation and will be evaluated on the basis of a report and a defense of the project during the session September.
Teaching methods :	This course is designed as a programming project to be implemented in three phases. The first two are guided and allow students to discover the initial application provided to them, by implementing some rather easy extensions. For the last phase, the student has the opportunity to go further in implementing the software system by proposing and realizing his own extensions. In each phase the students work on a different subset of problem, predefined in the project statement. Students work in pairs to have the opportunity to exchange views. Collaboration between pairs is organized through the discussion forum of the course where everyone can ask questions. Other students and supervisors provide possible solutions.
Content :	-- Analyse the problem; -- Design the application to be implemented; -- Implement the application (in Java); -- Document the application (preconditions, postconditions, invariants, alternative solutions, algorithms used, manual); -- Test the application; -- Write a report;

	-- Defend and give a demonstration of the final application.
Bibliography :	The course notes and reference books used for the courses SINF1160 and SINF1161 remain valid for this course, as well as any other introductory books or other reference material on programming in Java.
Cycle and year of study :	> Bachelor in Computer Science
Faculty or entity in charge:	INFO