










Teacher(s)	Mens Kim ;Nijssen Siegfried ;Pecheur Charles ;
Language :	French
Place of the course	Louvain-la-Neuve
Main themes	<ul style="list-style-type: none"> <li>• Fundamental concepts of object-oriented programming;</li> <li>• Python programming language;</li> <li>• Analysis of a computer problem, design, specification and implementation of a solution;</li> <li>• Linear data structures.</li> </ul>
Learning outcomes	<p><b>At the end of this learning unit, the student is able to :</b></p> <p>Given the learning outcomes of the "Bachelor in Computer science" program, this course contributes to the development, acquisition and evaluation of the following learning outcomes:</p> <ul style="list-style-type: none"> <li>• S1.I2</li> <li>• S2.2, S2.4</li> </ul> <p>Students who have successfully completed this course will be able to :</p> <ul style="list-style-type: none"> <li>• Apply the concepts, laws, reasonings to a disciplinary problem of squared complexity.</li> <li>• Describe adequate modeling and calculation tools to solve a disciplined disciplinary problem.</li> <li>1 • Model a problem and design one or more technical solutions that meet the specifications</li> <li>• Implement and test a solution in the form of a model, a prototype and / or a digital model.</li> <li>• Commit collectively to a work plan, a timetable (and roles to keep).</li> <li>• Communicate in graphical and schematic form interpret a diagram, present the results of a work, structure information.</li> <li>• Read, analyze and exploit technical documents (standards, plans, specifications, specifications, ...).</li> <li>• Write written summary documents taking into account the requirements of the missions (projects and problems).</li> <li>• Demonstrate a good understanding of the concepts and methodology of object-oriented programming.</li> <li>• Make good use of the elements of an object-oriented language such as Python.</li> </ul>
Evaluation methods	<p>A programming assignment is due each week.</p> <p>A mid-term evaluation will be organised halfway throughout the quadrimester.</p> <p>The end-of-term exam aims to assess both the understanding of the course material and the capacity to apply it to write simple but correct Python programs.</p> <p>The final course mark takes into account the mid-term evaluation and assignments during the quadrimester, in addition to the mark of the end-term exam.</p> <p>The assignments and mid-term evaluation cannot be retaken for the June or September sessions.</p> <p>If the mark for the mid-term evaluation is higher than that for the end-term exam, it will count for 1/3 and the mark of the end-term exam for 2/3.</p> <p>If the mark for the mid-term evaluation is lower than that for the end-term exam, only the mark for the exam will be used to calculate the final course mark.</p> <p>A bonus of 1 point will be granted to students who have participated in and regularly submitted their programming assignments during the quadrimester.</p> <p>In case of plagiarism detection confirmed by a plagiarism detection tool the course teachers reserve the right to invite the student to pass an oral interrogation.</p>
Teaching methods	<p>The chosen teaching method relies on active student participation, through a mixture of :</p> <ul style="list-style-type: none"> <li>• course lectures,</li> <li>• partical exercice sessions with tutors,</li> <li>• programming exercices on the INGIous platform?</li> </ul> <p>Even though preference will be given to face-to-face teaching sessions, depending on the health situation and the number of students enrolled, other forms of teaching and evaluation (online, co-modal or hybrid) may be considered.</p>
Content	<ul style="list-style-type: none"> <li>• Programs, source code and program execution</li> <li>• Identifiers, variables, values, types, assignment</li> <li>• Expressions, statements</li> </ul>

	<ul style="list-style-type: none"> <li>• Conditional structures and loops</li> <li>• Functions, parameters, calls, results, execution, variable scoping</li> <li>• Specifications and tests</li> <li>• Modules</li> <li>• Data structures, lists, strings and their operations</li> <li>• References and nested data structures</li> <li>• Nested lists, tuples, matrices, dictionaries</li> <li>• Dichotomic search algorithms</li> <li>• File handling, input/output</li> <li>• Exception handling</li> <li>• Object-oriented programming and garbage collection</li> <li>• Classes, objects, constructors, methods</li> <li>• References to an object, self-references and self-calls</li> <li>• Class, instance and local variables, scope and visibility</li> <li>• Class composition, inheritance and encapsulation</li> <li>• Polymorphism, super calls and dynamic binding</li> <li>• Object equality</li> <li>• Linked data structures</li> </ul>
<p>Inline resources</p>	<p>All course material will be made available online: slides, syllabus, exercices, ...</p>
<p>Faculty or entity in charge</p>	<p>INFO</p>

Programmes containing this learning unit (UE)				
Program title	Acronym	Credits	Prerequisite	Learning outcomes
Minor in numerical technologies and society	<a href="#">MINSTIC</a>	5		
Master [120] in Data Science : Statistic	<a href="#">DATS2M</a>	5		
Master [120] in Linguistics	<a href="#">LING2M</a>	5		
Additionnal module in Geography	<a href="#">APPGEOG</a>	5		
Bachelor in Mathematics	<a href="#">MATH1BA</a>	6		
Bachelor in Computer Science	<a href="#">SINF1BA</a>	5		
Approfondissement en statistique et sciences des données	<a href="#">APPSTAT</a>	5		
Minor in Computer Sciences	<a href="#">MINSINF</a>	5		
Minor in Statistics, Actuarial Sciences and Data Sciences	<a href="#">MINSTAT</a>	5		
Certificat d'université : Statistique et science des données (15/30 crédits)	<a href="#">STAT2FC</a>	5		