<table>
<tr><th><b>NAME OF THE COURSE</b></th><th colspan="7"><b>Programming paradigms</b></th></tr>
<tr><td>Code</td><td>PMID45</td><td>Year of study</td><td colspan="5">GU-1</td></tr>
<tr><td>Course teacher</td><td>izv. prof.dr. sc. Saša Mladenović</td><td>Credits (ECTS)</td><td colspan="5">5,0</td></tr>
<tr><td rowspan="2">Associate teachers</td><td rowspan="2"></td><td rowspan="2">Type of instruction (number of hours)</td><td>L</td><td>S</td><td>E</td><td colspan="2">F</td></tr>
<tr><td>30</td><td></td><td>30</td><td colspan="2"></td></tr>
<tr><td>Status of the course</td><td>mandatory</td><td>Percentage of application of e-learning</td><td colspan="5">25%</td></tr>
<tr><td colspan="8"><b>COURSE DESCRIPTION</b></td></tr>
<tr><td>Course objectives</td><td colspan="7">Adopt the basic knowledge of programming paradigms.</td></tr>
<tr><td>Course enrolment requirements and entry competences required for the course</td><td colspan="7">Data structures and algorithms
Object-oriented programming</td></tr>
<tr><td>Learning outcomes expected at the level of the course (4 to 10 learning outcomes)</td><td colspan="7">To argument the advantages and disadvantages of a single programming paradigm.
To develop a simple program using different programming paradigms and languages.
To choose the appropriate programming paradigm in different usage contexts.
To argument advantages and disadvantages in using functional and imperative paradigms in concurrent program execution.</td></tr>
<tr><td>Course content broken down in detail by weekly class schedule (syllabus)</td><td colspan="7">Overview of programming paradigms with examples in an appropriate programming language.
Common properties of programming languages.
Imperative programming
Object-oriented programming - class based
Object-oriented programming - prototype based
Functional programming
Concurrent and imperative programming
Concurrent and functional programming
Logic programming
Best practice cases
Comparison of solutions in different programming paradigms on known problems</td></tr>
<tr><td>Format of instruction</td><td colspan="2">☒ lectures
☐ seminars and workshops
☐ exercises
☐ on line in entirety
☒ partial e-learning
☐ field work</td><td colspan="5">☒ independent assignments
☐ multimedia
☒ laboratory
☐ work with mentor
☒ homework assignments</td></tr>
<tr><td>Student responsibilities</td><td colspan="7">Lecture and laboratory attendance, active participation in course activities, homework and project realization, final exam.</td></tr>
</table>

	Name	Ects	Name	Ects	Name	Ects
Screening student work *(name the proportion of ECTS credits for each activity so that the total number of ECTS credits is equal to the ECTS value of the course)*	Class attendance	2	Research		Experimental work	
	Oral exam	0.5	Report		Homework assignments	0.5
	Seminar essay		Essay			
	Tests		Practical training			
	Written exam	0.5	Project	1		

Grading and evaluating student work in class and at the final exam	Attendance/Participation (20%) Project (40%) Final/Oral Exam (40%)

	Title	**Number of copies in the library**	**Availability via other media**
Required literature (available in the library and via other media)	Robert W Sebesta, Concepts of Programming Languages, 10th Edition, Addison-Wesley, 2013	0	

Optional literature (at the time of submission of study programme proposal)	Bruce A. Tate, Seven Languages in Seven Weeks: A Pragmatic Guide to Learning Programming Languages, The Pragmatic Programmers, 2010
Quality assurance methods that ensure the acquisition of exit competences	Student discussion, anonymous student evaluation questionnaire, student success rate, self-assessment
Other (as the proposer wishes to add)	