General information		
Course leader	**Danijel Kučak, Senior lecturer.**	
Course title	**Advanced Programming Paradigms**	
Study programme		
Course status	Elective	
Year	Year 1, semester 2	
Number of credits and mode of teaching delivery	ECTS student workload coefficient	5
	Number of hours (L+E+S)	60 (30 P + 30 V + 0 S)

COURSE DESCRIPTION
1.1. Course objectives
Introducing students to the use of unit testing and use of test driven and behavior driven development. Students will implement aspect oriented concepts to increase modularity by allowing the separation of cross-cutting concerns. Students will learn how to use advanced features of version control systems. Students will be introduced with concepts of functional programming and visualizing code metrics.
1.2. Conditions for enrolment in the course
No formal conditions. Student should be able to write programs comfortably in any object-oriented programming language.
1.3. Expected learning outcomes of the course
• **LO1 – Describe the basic concepts of software solution testing and compare types of software solution testing.**   • **LO2 - Design cases for unit testing of software solution and use software tools for unit testing of software solutions.**   • **LO3 - Detect logical errors in a given software solution using a software tool (debugging).**   • **LO4 - Analyse the time spent and allocated memory for executing a given software solution using a software tool (profiling). Plan testing of software systems acceptability and compliance with relevant standards**   • **LO5 - Analyse the software solution for the purpose of detecting shared code**   • **LO6 - Propose and implement an aspect-oriented approach to the organization of shared code**   • **LO7 - Implement a source code management strategy**   • **LO8 - Analyse and refactor the software solution according to the principles of pure code**   • **LO9 - Implement metrics over software solution**
1.4. Course content
Unit testing

Test driven development **Behaviour driven development** **Aspect oriented programming** **- AspectJ + PostSharp** **Code versioning** **Clean code approach** **Metrics in programming** **- Visualisation of metrics** **Functional programming**

1.5. Teaching delivery modes:	☒ lectures ☐ seminars and workshops ☒ exercises ☐ remote learning ☐ field work	☒ independent work ☐ multimedia and network ☒ laboratory ☒ mentoring ☐ other ________
1.6. Comments		

1.7. Student obligations
STUDENT ATTENDANCE **Class attendance is mandatory in the percentage prescribed by the Studies and examination regulations.** **PASSING THE EXAM** **The course has defined learning outcomes. In order for a student to pass the course, he/she must achieve a minimum of 50% of the points available for each learning outcome and collect a minimum of 50.01 points out of a possible 100 points per course.**
1.8. Monitoring[1] student work

Class attendance		Activity during class		Seminar paper		Experimental work	
Written exam		Oral exam		Essay		Research	
Project	100%	Continuous assessment of knowledge		Student report		Practical work	
Portfolio		Homework					

1.9. Assessment and evaluation of student work during classes and the final exam
A grading system based is on a credit accumulation model combined with a defined sub-model, providing a model of the grading method and checking the satisfaction of learning outcomes used in this course.

1 IMPORTANT NOTES: Next to each method of monitoring student work it is necessary to insert an adequate share of each activity in ECTS credits, so the total number of ECTS credits corresponds to the credit value of the course. You can use empty fields for additional activities.

CONCRETE REVIEW OF EVALUATION METHODS

The maximum number of points that a student can earn in a course is 100. Grades are calculated according to the following criteria table within which the distribution of passing grades in terms of the number of points is applied.

Points	**Grade**
0,00 - 50,00	(1) unsatisfactory
50,01 - 58,00	(2) sufficient
58,01 - 75,00	(3) good
75,01 - 92,00	(4) very good
92,01 - 100,00	(5) excellent

The method of accumulating points is determined in this course in accordance with the elements of scoring as follows:

Criterion	**Maximum points**
Project	100
TOTAL	100

The way of taking the colloquiums, the learning outcomes it covers, as well as the implementation of exams and remedial exams are defined by the "Instructions for attending and taking the course".

1.10. Required reading (at the moment of submitting the joint study programme report)

- **McLauglin, Pollice, West: Head First Object-Oriented Analysis and Design**
- **Martin: Clean Code: A Handbook of Agile Software Craftsmanship**

1.11. Additional reading (at the moment of submitting the joint study programme report)

1.12. Number of copies of required reading in relation to the number of students who currently attend a course

Title	*Number of copies*	*Number of students*

1.13. Methods of quality monitoring that ensure the acquisition of knowledge, skills and competencies.

Monitoring the fulfilment of the desired learning outcomes is an important element of

assessment because learning outcomes are the "guarantees" that the school gives to students, but also to employers and the wider community. Learning outcomes represent the minimum threshold that each student must achieve in order to pass the course. For a passing grade, the student must satisfy all the learning outcomes with the demonstrated knowledge, which corresponds to 50% of the points achieved for each learning outcome. The method of scoring based on learning outcomes is presented in the document "Instructions for attending and taking the course".