**DESCRIPTION OF A STUDY COURSE – SYLLABUS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Title of a course** | **Programming II** | | | | |
| **Study programme** | **Professional undergraduate study Telematics** | | | | |
| **Status of a course** | Obligatory | | | | |
| **Year of study** | 2 | **Semester** | W | **ECTS credits** | 6 |
| **Goals of a course** | | | | | |
| Acquire competencies for designing software systems using UML and implementing them in an object-oriented programming language. | | | | | |
| **Conditions for enrolling course** | | | | | |
| No conditions | | | | | |
| **Learning outcomes on a level of a study programme which includes course** | | | | | |
| Outcome 4: Use computer principles and methods related to the architecture and organization of computers and computer networks.  Outcome 5: Use computer principles and methods related to programming languages, databases, and operating systems.  Outcome 6: Design and implement desktop, web and mobile computer applications and computer programs for microcomputers and microcontrollers, with or without a database.  Outcome 10: Analyse and implement an information system in the field of telematics.  Outcome 15: Participate in teamwork and independently present professional content in written and spoken form in Croatian and English. | | | | | |
| **Expected learning outcomes on a level of a course** | | | | | |
| 1. Describe basic concepts related to object-oriented design 2. Describe basic concepts related to object-oriented programming 3. Create the following UML diagrams: usage diagram, activity diagram, class diagram and sequence diagram 4. Implement classes in Java programming language using MVC architecture according to UML diagrams 5. Compile project documentation | | | | | |
| **Content of a course** | | | | | |
| Introduction into object-oriented languages (C++, C#, Java ) and division into procedural and functional languages; Objects and classes (structure, syntax, encapsulation, principle of privacy); Object-oriented analysis (OOA); Object-oriented design (OOD); Object-oriented modelling with UML (aspects and diagrams, class diagram, diagram of the application case, component diagram, chart of division, diagram of state, sequence and collaboration, application of UML tools, GO and Poseidon); Attributes (constants, variables, parameters, accessibility, right of access, class and object attributes); Operations (call – by – value, call – by- reference, class methods in objects etc.); Object associations (aggregation, composition, cardinality, aspects of association implementation); Control structures (arrays, choice, multiple choice, iteration, structural programming); applications and applets; documentation and guidelines for making a source code, source code documentation. | | | | | |