COURSE UNIT DESCRIPTION

Course unit title	Course unit code
Object-Oriented Programming	**ITOP**

Lecturer	Department where the course unit is delivered
Coordinator: lector dr. Agnė Brilingaitė	Department of Computer Science II Faculty of Mathematics and Informatics Vilnius University

Cycle	Type of the course unit
First	Compulsory

Mode of delivery	Semester or period when the course unit is delivered	Languages of instruction
Face-to-face	2nd semester	Lithuanian and English

Prerequisites
Basic knowledge about IT, abilities to program, use text editors and command lines.

Number of ECTS credits allocated	Student's workload	Contact hours	Individual work
5	141	66	75

Purpose of the course unit: programme competences to be developed

Generic competences to be developed

- Ability for abstract thinking , processing and analysing information *(BK3)*
- Ability to use information and communications technologies *(BK5)*

Subject-specific competences to be developed

- Ability to apply general methods of the program design, make and analyse software requirements *(DK1)*
- Ability to analyse the algorithmic process of the task based on the general properties of the algorithm *(DK2)*
- Ability to develop the software project (or IT service) and to write its specification *(DK3)*

Learning outcomes of the course unit	Teaching and learning methods	Assessment methods
Ability to distinguish and explain the principles (inheritance, encapsulation, polymorphism) of object-oriented programming paradigm and find the application of the principle in the programming code.	Studies of literature and analysis of examples.	Exam (various types of questions).
Ability to write the medium size programs in Java programming language for a given task.	Analysis of examples during lectures and practical sessions; implementation of tasks, and project work, consultations.	Programming tests, defence of the project.
Ability to recognize Java programming language in the given code and to understand the program flow and structure; ability to execute the program or modify it.	Executing examples, adapting to new tasks, extension of examples.	Exam (practical exercises), programming tests, review of other students' work (checking, execution), defence of the project.
Ability to implement the specific task or algorithm.	Implementation of the	Programming tests,

	algorithm, consultations.	exam(programming on the paper).
Ability to write program specification by emphasizing the structure and functionality.	Preparation of the project report, consultations.	Project report.
Ability to evaluate the correspondence of the program report to the requirements and the programming part.	Peer review, other students' program testing and analysis of reports.	Review.
Ability to use program development environments, various editors on different operating systems, and the command line.	Example testing and practical tasks.	Project, programming tests.
Ability to use the documentation (*Java API*), to search for packages, classes, methods, examples, and select the necessary information.	Information search in the documentation based on the given task.	Programming tests (find the method based on the task).

Course content: breakdown of the topics	**Individual work: time and assignments**							
	Lectures	Consultations	Seminars	Laboratory work(LW)	*Consultation during LW*	**Contact hours**	**Individual work**	**Assignments**
1. Object-oriented programming paradigm	1					**1**	**1**	
2. Basic of Java syntax. Data types and control structures. Structure of the class. Variables and methods	6			4	*3*	**10**	**8**	Programming test, individual tasks, individual studies of literature and lecture slides
3. Objects. Dynamic and static methods. Method overloading. Constructors. Garbage collection	5			6		**11**	**8**	
4. *Java API*. Usage of packages	2			2	*2*	**4**	**4**	I project part, class work in groups, homework, programming test, studies of literature and lecture slides
5. Information hiding. Encapsulation. Inheritance. Abstract classes and interfaces. Polymorphism. Inner classes	6			8		**14**	**14**	
6. Class diagram. Basics of *UML*	2			2	*4*	**4**	**4**	III project part
7. Development of the graphical user interface	2			2		**4**	**4**	Individual work, II project part, studies of literature and lecture slides
8. Exceptions. Input and output streams	4			4		**8**	**4**	
9. Collections Framework	2			2		**4**	**2**	
Small project		2		1		**3**	**16**	
Review of the project				1		**1**	**4**	
Preparation for the exam						**2**	**6**	
Total	**30**	**2**		**32**		**66**	**75**	

Assessment strategy	**Weight %**	**Deadline**	**Assessment criteria**
Programming tests (virtual learning environment)	20	Week 5, Week 15	Two programming tests; each is worth 1 point. Each test lasts two academic hours. The first test is to check knowledge about syntax basics and ability to apply knowledge in practical situations. The second test is to check the application of knowledge about the object-oriented paradigm. Assessment criteria: 0 – the test is not done or there is no logical solution; 0,3 – methods and classes are declared correctly; 0,6 – methods and classes are declared correctly, and some logic is implemented; 0,9 – there are some small mistakes or some cases are not implemented; 1 – the program corresponds to the task requirements.

Small project and review (virtual learning environment)	30	During the second part of the semester in parts	Project is carried out in small groups (2-3 students) with tasks being distributed. The project has several parts: I part – the implemented framework of the program, chosen structure, but without the main functionality (0,5 point); II part – the program with full functionality based on the requirements and object-oriented principles (0,5 point); III part – the written report on the program structure and functionality (0,5 point); IV part – individual review of other student group's work (report and program). Students must give the insights about the application of object-oriented principles, interpretation of the programming code, system functionality and correspondence of the programming part to the report (0,5 point); V – defence of the project (1 point). Each student is evaluated individually based on the project task and ability to defend the particular project part. During the defence students must demonstrate the ability to argument, modify programming code, find specific parts of the code that are implementation of some functionality.
Exam	50	June (exam session)	Exam consists of a set of questions of different types: 1) question (terminology) that requires a precise answer (0 or 0,5 point); 2) Java coding by applying knowledge about Java programming language (0 – not written or there is no logic, 0,5 – there are big mistakes or missing elements, 1 – the code corresponds to the task with minimal mistakes); 3) code interpretation, checking how the student is able to read the programming code and iterate through the algorithm or the full programming code (0,5 – half of the answer is correct, 1 – the answer is correct); 4) open questions to check the ability to link practical and theoretical knowledge, find and apply the necessary concept, explain the given constructs/diagrams, draw diagrams, find mistakes, etc. (1,5 point is given for 3-7 different open tasks).
			The student must collect at least 1 point during the semester in order to get the permission to go for the examination. The final result is the average of the exam and semester results.

Author	Publishing year	Title	Issue No or volume	Publishing house or Internet site
Required reading				
Oracle and/or its affiliates		The Java^{TM} Tutorials		http://docs.oracle.com/javase/tutorial/index.html
C. Larman	2005	Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative development		Prentice-Hall
A. Brilingaitė	2012	Object-Oriented Programming. Study Guide		
Optional reading				
A. Riškus	2002, 2003, 2005, 2006, 2008	Programavimas Java		Technologija
B. Eckel	2006	Thinking in Java		Prentice-Hall http://www.mindview.net/Books/TIJ/

W. Sawitch	2001	Java, An Introduction to Computer Science & Programming		Prentice-Hall
D. Skrien	2009	Object-Oriented Design Using Java		McGraw-Hill