

Subject card

Subject name and code	Object-functional programming, PG_00143991						
Field of study	Informatics						
Date of commencement of studies	October 2025		Academic year of realisation of subject		2026/2027		
Education level	Bachelor's studies		Subject group		Obligatory subject group in the field of study Subject group related to practical vocational preparation		
Mode of study	full-time studies		Mode of delivery		at the university		
Year of study	2		Language of instruction		Polish		
Semester of study	4		ECTS credits		3.0		
Learning profile	practical		Assessment form		credit		
Conducting unit							
Name and surname of lecturer (lecturers)	Subject supervisor		dr Tomasz Borzyszkowski				
	Teachers						
Lesson types	Lesson type	Lecture	Tutorial	Laboratory	Project	Seminar	SUM
	Number of study hours	15.0	0.0	30.0	0.0	0.0	45
	E-learning hours included: 0.0						
Learning activity and number of study hours	Learning activity	Participation in didactic classes included in study plan		Participation in consultation hours		Self-study	SUM
	Number of study hours	45		0.0		30.0	75
Subject objectives	The primary aim is to familiarise students with the most important concepts of object-oriented and functional approaches to programming. This broadens and deepens the knowledge already possessed by the students and facilitates their selection of appropriate programming techniques depending on the nature of the task. Projects carried out during laboratory classes serve to consolidate the concepts learnt during the lecture in a practical way. Translated with DeepL.com (free version)						
Learning outcomes	Course outcome		Subject outcome		Method of verification		
	[INFL3_K02] can precisely formulate questions to deepen his/her own understanding of a given topic or to find missing elements of reasoning		is able to formulate precise questions in order to deepen his/her own understanding of a given topic or to find missing pieces of reasoning		[SK1] oral statement/conversation/discussion [SK8] observation of student's independent or team work		
	[INFL3_W04] has ordered, theoretically founded knowledge in programming, algorithms and complexity, programming languages and paradigms		has a structured, theoretically grounded knowledge of programming, algorithms and complexity, programming languages and paradigms		[SW1] oral statement/conversation/discussion [SW2] presentation/project/paper/report		
	[INFL3_U08] assesses the usefulness of various paradigms and related programming tools to solve various types of problems		evaluates the suitability of different paradigms and programming tools to solve problems of different types		[SU2] presentation/project/paper/report [SU8] observation of student's independent or team work		
	[INFL3_U04] can create, run and test programs using dedicated tools and design patterns		is able to create, run and test programs using dedicated tools and design patterns		[SU2] presentation/project/paper/report [SU8] observation of student's independent or team work		
	[INFL3_U09] is able to - in accordance with the given specification - design and implement IT system		is able to design and implement an information system according to a given specification information system		[SU2] presentation/project/paper/report [SU8] observation of student's independent or team work		

Subject contents	1. Introduction of basic concepts: class, object, field, method; fields and methods: static, public and private in Java; initialisation and deletion of an object and the so-called rubbish collector mechanism; overview of control statements.2. Class packages: hiding implementation: package structure; importing packages; setting access rights to package components; building interfaces and their implementation.3. Inheritance and polymorphism: inheritance: syntax and preserving access rights to inherited fields and methods; from abstract to concrete: abstract and final classes; comparison of properties of final and static fields; comparison of properties of overloaded and polymorphic methods; examples of polymorphic function calls.4. An overview of classes implementing common data structures: the concept of static and dynamic data structures; an overview of the properties and operations provided; classes implementing enumerative types and iterators; polymorphic methods to enable sorting of elements stored in collections.5. The concept of lambda-functions and their use in stream processing.6. Programming with exceptions: overview of predefined exceptions; rules for creating new exceptions; reporting and catching exceptions.7. programming using threads: the concept of threads, shared resources and the critical section; an overview of resource sharing methods: resource locking and the thread jamming problem and priority queues and the fairness problem.8 Overview of selected design patterns.		
Prerequisites and co-requisites	No prerequisites		
Assessment methods and criteria	Subject passing criteria	Passing threshold	Percentage of the final grade
	activity in class	0.0%	10.0%
	project presentation, observation of student work	51.0%	90.0%
Recommended reading	Basic literature	1. Eckel B., Thinking in Java - Polish edition. Wydawnictwo HELION, Warszawa, 2010.2 Horstmann C. S., Cornell G., Core Java 2 - Fundamentals. Helion, 2010.3. Horstmann C. S., Cornell G., Core Java 2 - Advanced techniques. Helion, 2010.	
	Supplementary literature	None	
	eResources addresses		
Example issues/ example questions/ tasks being completed	1. Implement a class that calculates arithmetic expressions written in Reverse Polish Notation.2. Implement the Stack class implementing the idea of stacks of writes with the following public methods: push, pop and peek.		
Work placement	Not applicable		

Document generated electronically. Does not require a seal or signature.