



Subject card

Subject name and code	Programming Languages, PG_00047657						
Field of study	Informatics						
Date of commencement of studies	October 2023	Academic year of realisation of subject			2024/2025		
Education level	first-cycle studies	Subject group			Obligatory subject group in the field of study Subject group related to scientific research in the field of study		
Mode of study	Full-time studies	Mode of delivery			at the university		
Year of study	2	Language of instruction			Polish		
Semester of study	3	ECTS credits			3.0		
Learning profile	general academic profile	Assessment form			assessment		
Conducting unit	Department of Algorithms and Systems Modelling -> Faculty of Electronics, Telecommunications and Informatics						
Name and surname of lecturer (lecturers)	Subject supervisor	dr inż. Piotr Mironowicz					
	Teachers	dr inż. Piotr Mironowicz					
Lesson types and methods of instruction	Lesson type	Lecture	Tutorial	Laboratory	Project	Seminar	SUM
	Number of study hours	15.0	0.0	0.0	15.0	0.0	30
	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	30		3.0		42.0	75
Subject objectives	Student learned popular programming paradigms and learned to use them.						

Learning outcomes	Course outcome	Subject outcome	Method of verification
	[K6_W05] Knows and understands, to an advanced extent, methods of supporting processes and functions, specific to the field of study	The student knows conceptually and historically relevant programming languages such as Modula, Ada, Smalltalk, Haskell, and Prolog. He is aware of what problems underlied particular solutions.	[SW3] Assessment of knowledge contained in written work and projects
	[K6_U41] can produce, test or evaluate software using modern programming platforms, tools, languages and paradigms of different levels, as well as use software packages supporting scientific and research processes as well as business decision-making processes and teamwork	He can program in the languages of procedural and object paradigm (Modula, Smalltalk), functional (Haskell) and in logic (Prolog). Knows environments and packages supporting programming.	[SU4] Assessment of ability to use methods and tools
	[K6_U03] can design, according to required specifications, and make a simple device, facility, system or carry out a process, specific to the field of study, using suitable methods, techniques, tools and materials, following engineering standards and norms, applying technologies specific to the field of study and experience gained in the professional engineering environment	The student is able to code a solution to a specified problem using the properties of the selected programming paradigm. He recognizes which of the modeling approaches will be most appropriate for the given problem.	[SU1] Assessment of task fulfilment
[K6_W04] Knows and understands, to an advanced extent, the principles, methods and techniques of programming and the principles of computer software development or programming devices or controllers using microprocessors or programmable elements or systems specific to the field of study, and organisation of systems using computers or such devices	The student is familiar with all important programming paradigms and their importance for the principles of software development. He understands the relationship between high-level languages and the specificity of microprocessors.	[SW1] Assessment of factual knowledge	
Subject contents	<ol style="list-style-type: none"> 1. Procedural programming. 2. Linear syntax. FORTRAN 3. Activation records and subroutines 4. Recursive procedure call. 5. Block syntax. Control flow abstraction. 6. Binding of the names with objects. Range bonds. 7. The parameters of the procedure call. 8. Activation records for languages with recursion. 9. Static and dynamic calls. ALGOL. PASCAL. 10. Restrictions of block languages. 11. Abstraction of data and access protection. 12. Modularization. Modula-2. ADA83, ADA95 13. Exceptions. Exception handling models. 14. Concurrent procedures. Rendezvous. 15. Object-oriented programming. Objects, classes, hierarchies. 16. Dynamic types. Polymorphism. Smalltalk. C + +. 17. Recursive interpreted commands. 18. Symbolic transformation. Tail recursion. 19. LISP. Atoms and lists. 20. Functional programming. Haskell, XSL. 21. Reduction, filtering and casting. 22. Lambda calculus. 23. Memory management in LISP systems 24. Programming in logic. Prolog. 		
Prerequisites and co-requisites			
Assessment methods and criteria	Subject passing criteria	Passing threshold	Percentage of the final grade
	Tests	50.0%	40.0%
	Project	50.0%	60.0%

Recommended reading	Basic literature	<p>S. Mangano: XSLT receptury, wyd.2, Helion 2007 Cincom Smalltalk Downloads, http://www.cincomsmalltalk.com/ SAXON - The XSLT and XQuery Processor, http://saxon.sourceforge.net/ W.F. Clocksin, W.F., Mellish, C.S.: Prolog Programowanie. Helion 2003 Ada Programming, http://en.wikibooks.org/wiki/Ada SWI-Prolog downloads, www.swi-prolog.org/download.html ADA Core, the GNAT Pro Company, http://www.adacore.com/home, https://libre.adacore.com/ D. S. Touretzky: Common Lisp: A Gentle Introduction to Symbolic Computation, http://www.cs.cmu.edu/~dst/LispBook/ Z. Huzar, Z. Fryźlewicz, I. Dubielewicz, B. Hnatk: Ada 95, Helion 1998 Polski serwis języka Smalltalk, http://www.objectspace.net/</p>
	Supplementary literature	http://en.wikipedia.org/wiki/Programming_paradigm
	eResources addresses	Adresy na platformie eNauczanie:
Example issues/ example questions/ tasks being completed		
Work placement	Not applicable	