Introduction to Programming Languages

Course description sheet

Basic information

Field of study Computer Science and Intelligent Systems **Major** - **Organisational unit** Faculty of Electrical Engineering, Automatics, Computer Science and Biomedical Engineering **Study level** First-cycle (engineer) programme **Form of study** Full-time studies **Profile** General academic	**Didactic cycle** 2022/2023 **Course code** EISIS.li10O.7ee3ddd63be2cc6d2a34f51cf264e288.22 **Lecture languages** english **Mandatoriness** Elective **Block** General Modules **Course related to scientific research** No

Course coordinator	Weronika T. Adrian, Krystian Jobczyk
Lecturer	Mateusz Ślażyński, Krystian Jobczyk

Period Semester 5	**Method of verification of the learning outcomes** Completing the classes **Activities and hours** Lectures: 14 Laboratory classes: 14	**Number of ECTS credits** 3

Goals

C1	Acquainting students with the formal aspects of programming languages.
C2	Show students the fundamental connection between logic and programming.

Course's learning outcomes

Code	Outcomes in terms of	Learning outcomes prescribed to a field of study	Methods of verification
Knowledge – Student knows and understands:			
W1	Student understands the role of lambda calculus in computer science.	ISI1A_W05	Activity during classes, Execution of laboratory classes, Completion of laboratory classes
W2	Student knows the difference between syntactic and semantic features of a programming language.	ISI1A_W05	Activity during classes, Execution of laboratory classes, Completion of laboratory classes
Skills – Student can:			
U1	Student is able to define recursive computations in the lambda calculus.	ISI1A_U06	Execution of laboratory classes, Completion of laboratory classes
U2	Student can define the formal semantics of a strongly typed programming language.	ISI1A_U01	Execution of laboratory classes, Completion of laboratory classes
U3	Student can implement an interpreter given a formal description of a programming language	ISI1A_U07	Execution of laboratory classes, Completion of laboratory classes
Social competences – Student is ready to:			
K1	Student is ready to use formal definitions to communicate within research and development teams.	ISI1A_K05	Activity during classes, Execution of laboratory classes, Completion of laboratory classes

Program content ensuring the achievement of the learning outcomes prescribed to the module

The course will cover the operational semantics approach to defining programming languages. First, the lambda calculus will be introduced as a basic example of a Turing-complete language. Next, it will be extended with types and various programming concepts. The lectures will be accompanied by corresponding programming exercises.

Student workload

Activity form	Average amount of hours* needed to complete each activity form
Lectures	14
Laboratory classes	14
Realization of independently performed tasks	62
Student workload	**Hours** 90

Workload involving teacher	Hours 28

* hour means 45 minutes

Program content

No.	Program content	Course's learning outcomes	Activities
1.	Introduction to type theory. The historical overview, motivation, and mathematical preliminaries.	W1, W2, K1	Lectures
2.	Lambda calculus — introduction, formalization and implementation.	W1, U1, U3	Lectures, Laboratory classes
3.	Types; the simply-typed lambda-calculus.	U2, U3, K1	Lectures, Laboratory classes
4.	Extending the lambda calculus with basic programming features: ADT, macro system, local variables	U2, U3, K1	Lectures, Laboratory classes
5.	Side effects: modeling memory and exceptions.	U2, U3, K1	Lectures, Laboratory classes
6.	Subtyping and other relations with the type system.	U2, U3, K1	Lectures, Laboratory classes

Extended information/Additional elements

Teaching methods and techniques:

Lectures, E-learning, Project based learning

Activities	Methods of verification	Credit conditions
Lectures	Activity during classes, Completion of laboratory classes	A positive grade from the laboratory classes
Lab. classes	Execution of laboratory classes	At least 50% of points from each laboratory assignment.

Additional info

The laboratory class grade will be evaluated based on the results of laboratory class activities and the submitted assignments. Every class will be accompanied by a programming task that has to be submitted before the next meeting. The submissions will be graded automatically.

Conditions and the manner of completing each form of classes, including the rules of making retakes, as well as the conditions for admission to the exam

To pass the course, student is expected to gather 50% of points for every each laboratory assignment. In case they did not manage to achieve this goal during the semester, there will be a chance to retake the assignments in the resit session.

Method of determining the final grade

The final evaluation will be issued on the basis of the results of the laboratory class results. Moreover, participation in the discussions will be taken into account to raise the final grade.

Manner and mode of making up for the backlog caused by a student justified absence from classes

Absences in laboratory classes will require solving additional homework assigned by the tutor.

Prerequisites and additional requirements

The basic knowledge og the Python programming language is required.
Otherwise, to fully comprehend the course contents, student is recommend to have a basic knowledge about the following domains:

1. formal logic
2. formal languages and automata
3. compilers

Rules of participation in given classes, indicating whether student presence at the lecture is obligatory

Participation in laboratory and seminar classes is mandatory. A physical presence at the lecture is not obligatory, but absent students are expected to self-study the missed topic. Also, active involvement in the lectures may have a positive impact on the final grade.

Literature

Obligatory

1. Types and Programming Languages (The MIT Press); Benjamin C. Pierce; The MIT Press, 2002

Optional

1. Advanced Topics in Types and Programming Languages (The MIT Press); Benjamin C. Pierce; The MIT Press, 2004
2. The Little Typer (The MIT Press); Daniel P. Friedman, David Thrane Christiansen, et al.; The MIT Press, 2018
3. The Little Prover (The MIT Press); Friedman, Daniel P., Eastlund, Carl; The MIT Press, 2015

Learning outcomes prescribed to a field of study

Code	Content
ISI1A_K05	Dostrzega i rozumie konieczność nieustannego doskonalenia swojej wiedzy, umiejętności i kompetencji społecznych.
ISI1A_U01	Potrafi wykorzystać nabytą wiedzę matematyczną do opisu procesów, tworzenia modeli, analizy algorytmów oraz innych działań w obszarze informatyki.
ISI1A_U06	Potrafi algorytmizować wybrane problemy, ocenić ich złożoność obliczeniową, estymować czas wykonania, dobierać właściwe algorytmy do zadanego problemu, stosować metody i techniki Sztucznej Inteligencji.
ISI1A_U07	Potrafi projektować i rozwijać aplikacje z wykorzystaniem poznanych technologii oraz języków programowania. Potrafi doskonalić umiejętności nabyte w trakcie studiów.
ISI1A_W05	Ma uporządkowaną, podbudowaną teoretycznie wiedzę w zakresie języków formalnych, kompilatorów oraz języków programowania.