Logic Programming

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.Ii50O.624614f2f0150.22

Lecture languages
english

Mandatoriness
Elective

Block
General Modules

Course related to scientific research
Yes

Course coordinator	Weronika T. Adrian, Antoni Ligęza
Lecturer	Weronika T. Adrian, Antoni Ligęza, Mateusz Ślażyński

Periods Semester 5, Semester 7	**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	To teach students about Logic Programming at level sufficient to apply it later in their career.
C2	To show students the numerous advantages and beauty of the Logic Programming paradigm.

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 has basic knowledge about various programming paradigms (especially declarative) and knows when to apply them	ISI1A_W05, ISI1A_W06	Presentation
W2	Student knows how to apply formal logic in programming tasks	ISI1A_W05, ISI1A_W06	Presentation
W3	Student knows how to use Prolog to solve various programming tasks	ISI1A_W05, ISI1A_W06	Execution of laboratory classes, Test results, Presentation, Completion of laboratory classes
Skills – Student can:			
U1	Student can recognize applicability of different programming paradigms and tools.	ISI1A_U03, ISI1A_U06, ISI1A_U07	Execution of laboratory classes, Test results, Presentation, Completion of laboratory classes
U2	Student can write and understand Prolog programs.	ISI1A_U06, ISI1A_U07	Execution of laboratory classes, Test results, Completion of laboratory classes

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

The course will cover Logic Programming — a declarative programming paradigm aiming at logical and clean view of computer. Student will use Prolog as a dominant Logic Programming representative.

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 the declarative and logic programming — an overview of the domain. Applicability in the Artificial Intelligence problems.	W1, U1	Lectures
2.	Formal foundations of logic programming: syntax, proof strategies.	W1, W2, W3	Lectures, Laboratory classes
3.	Prolog — a logic programming language.	W3, U1, U2	Lectures, Laboratory classes
4.	Research directions and applications of the logic programming paradigm.	W1, U1	Lectures

Extended information/Additional elements

Teaching methods and techniques:

Lectures, E-learning, Group work

Activities	Methods of verification	Credit conditions
Lectures	Presentation, Completion of laboratory classes	According to the University Study Regulations
Lab. classes	Execution of laboratory classes, Test results	According to the University Study Regulations

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, one has to achieve at least 50% points from the following tasks:
— lab assignments; — test.
In case student failed to pass the course in the basic term, they can try again in the resit session.

Method of determining the final grade

The final grade will be a weighted average of the following components:
- lab assignments: 50% of the grade. - test: 50% of the grade.
To pass the course, one has also to get at least 50% from every each component separately. The grade may be further improved with extra-activity during the classes/lectures.

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

In case of an absence, one should contact the teacher to get additional assignemnts/materials.

Prerequisites and additional requirements

Basic acquaintance with formal logic.
Student will be using git version control system.
Generally:
Math at the undergraduate level.
Enthusiasm and creativity :)

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

Lecture: Attendance is not obligatory but strongly recommended. Students are expected ask their questions and discuss their doubts with the lecturer. Recording lectures requires an explicit agreement from the lecturer.
Laboratories: Attendance is obligatory. Students perform programming tasks according to the given instructions. There will be also a test, verifying the acquired knolwedge.
To pass the course, student has to: pass all the lab assignments and the test.

Literature

Obligatory

1. Dave Stuart Robertson, Quick Prolog (http://www.dai.ed.ac.uk/groups/ssp/bookpages/quickprolog/quickprolog.html)
2. GEIST Lab z Prologu, http://ai.ia.agh.edu.pl/wiki/pl:prolog:prolog_lab
3. Jan Wielemaker et. al., SWI-Prolog
4. Patrick Blackburn, Johan Bos, Kristina Striegnitz, Learn Prolog Now! (http://www.learnprolognow.org/)
5. I. Bratko, Prolog Programming for Artificial Intelligence, 4th ed, Addison-Wesley, 2011.
6. Michael Covington et. al., Prolog Programming in Depth.
7. Dennis Merritt, Adventure In Prolog, Amzi!, 2010.
8. Dennis Merritt, Building Expert Systems in Prolog, Amzi!, 2010.
9. Ulf Nilsson and Jan Małuszyński, Logic, Programming and Prolog (2ed)., Wiley, 2006
10. Michael Covington et. al., Prolog Programming in Depth., 1995.
11. Attila Csenki: Applications of Prolog. 2009 Attila Csenki & Ventus Publishing ApS

Optional

1. MICHAEL A. COVINGTON, ROBERTO BAGNARA, RICHARD A. O'KEEFE, JAN WIELEMAKER, SIMON PRICE, Coding guidelines for Prolog, Cambridge University Press, 2011.
2. Implementacja Prologu i dokumentacja: http://www.swi-prolog.org/
3. Materiały do Prologu: http://ai.ia.agh.edu.pl/wiki/pl:dydaktyka:pp:start http://home.agh.edu.pl/~ligeza/wiki/prolog:course

Scientific research and publications

Research

1. Knowledge Representation and Reasoning, Artificial Intelligence, Constraint Programming

Publications

1. Antoni Ligęza: Logical Foundations for Rule-Based Systems. Springer, 2006.
2. Improving efficiency in constraint logic programming through constraint modeling with rules and hypergraphs / Antoni LIGĘZA // W: FedCSIS [Dokument elektroniczny] : proceedings of the Federated Conference on Computer Science and Information Systems 2012 : September 9–12, 2012 Wrocław, Poland / eds. M. Ganzha, L. Maciaszek, M. Paprzycki. — Dane tekstowe. — Warsaw : Polskie Towarzystwo Informatyczne ; Los Alamitos : IEEE Computer Society Press, 2012. — Dane na dysku Flash. — W bazie Web of Science ISBN 978-83-60810-48-4. — ISBN: 978-83-60810-51-4. — S. 101–107.
3. A study of methodological issues in design and development of rule-based systems: proposal of a new approach / Antoni LIGĘZA, Grzegorz J. NALEPA // Data Mining and Knowledge Discovery ; ISSN 1384-5810. — 2011 vol. 1 iss. 2, s. 117–137. — Bibliogr. s. 135–137
4. Models and tools for improving efficiency in constraint logic programming / Antoni LIGĘZA // Decision Making in Manufacturing and Services ; ISSN 1896-8325. — 2011 vol. 5 no. 1–2, s. 69–78. — Bibliogr. s. 78, Abstr.. — tekst: http://journals.bg.agh.edu.pl/DECISION/2011-01-02/DM_2011_1_2_06.pdf
5. The HeKatE methodology : hybrid engineering of intelligent systems / Grzegorz J. NALEPA, Antoni LIGĘZA // International Journal of Applied Mathematics and Computer Science ; ISSN 1641-876X. — 2010 vol. 20 no. 1: Computational intelligence in modern control systems, s. 35–53. — Bibliogr. s. 51–53

Learning outcomes prescribed to a field of study

Code	Content
ISI1A_U03	Potrafi korzystać z literatury fachowej, zdobywać potrzebne informacje w sieci Internet, dokonywać interpretacji zdobytych informacji oraz wyciągać wnioski i formułować opinie.
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.
ISI1A_W06	Ma uporządkowaną, podbudowaną teoretycznie wiedzę z inżynierii oprogramowania, modelowania oprogramowania, zarządzania projektem informatycznym, wdrażania i komercjalizacja rozwiązań informatycznych.