(*faculty stamp*)

COURSE DESCRIPTION

1. Course title: AUTOMATIC INFERENCE SYSTEMS	**2. Course code**
3. Validity of course description: 2014/2015	
4. Level of studies: BA, BSc programme	
5. Mode of studies: intramural studies	
6. Field of study: AUTOMATYKA I ROBOTYKA, AEII	(FACULTY SYMBOL)
7. Profile of studies: general academic	
8. Programme:	
9. Semester: 4	
10. Faculty teaching the course: Institute of Automatic Control, RAu1	
11. Course instructor: dr inż. Teresa Główka	
12. Course classification: common courses	
13. Course status: compulsory	
14. Language of instruction: English	
15. Pre-requisite qualifications: Courses (in Polish): Programowanie obliczeń komputerowych, Programowanie obiektowe. It is assumed that before learning of this course, the student has a background in procedural programming languages as well as he knows and distinguishes between basic concepts of mathematical logic.	
16. Course objectives: The aim of the lecture is to provide students with basic knowledge on logic programming languages, their characteristics and applications, based on the example of Prolog language. The aim of the laboratory is the acquisition by the students the skills of programming in Prolog language.	
17. Description of learning outcomes:	

Nr	Learning outcomes description	Method of assessment	Teaching methods	Learning outcomes reference code
1.	(W1) Student knows the tasks of logic programming languages, knows what is the structure of the program, its components, knows basic standard predicates.	SP	WM	K_W5/3; W1/2;W16/1
2.	(W2) Student understands the mechanisms of inference in Prolog, in particular backtracking, recursion, and operations on lists.	SP	WM	K_W5/3; W1/2;W16/1
3.	(U1) Student can predict the result of the program, and also write his own procedures in accordance with the principles of programming in Prolog.	SP, CL	L	K_U11/3; U7/2; U20/1
4.	(U2) Student is able to apply this knowledge to simple problems of combinatorial programming, in particular for problems of other kind than in the given examples.	SP, CL	L	K_U11/3; U7/2; U20/1
5.	(K1) Student shows a creative approach to solving combinatorial problems using known logic programming languages, but he is aware of his own skills and imperfections and he has the need for self-education in this field.	CL	L	K_K01/2
6.	(K2) Student can interact in a small group to work out the best way to solve a given problem.	CL	L	K_K03/3
7.				
8.				

18. Teaching modes and hours

Lecture / Laboratory

Sem. 4 - 30 h. / Sem. 4 - 15 h.

19. Syllabus description:

Semester 4 :

Lecture:

Introduction to the course: basic concepts of logic. Declarative programming languages vs. procedural languages - differences and intended uses, examples. History and implementations of declarative programming languages. Prolog as an example of a declarative programming language. Prolog and relational databases.
The structure of Prolog programs. Prolog and logic of predicates. Variables and constants in the program. Section "predicates", complex structures. Section "domains", defining the types of variables. Section "clauses", definitions of the rules and facts. Section "goal", an external and internal goal of the program.
Elementary database-like programs and their interpretation. Closed-world assumption. Deterministic and non-deterministic predicates. Negation in Prolog.
Mechanisms of logical inference in Prolog. Unification and backtracking. Forcing backtracking using the standard predicate "fail". Cutting, the use of standard predicate "!" (cut) to control backtracking. The use of cut-off to change the predicate behavior from non-deterministic to deterministic. Unification and backtracking in programs containing complex rules.
Recursion: the construction of recursive rules, applicability to the loops forming. Tail and not-tail recursion. The role of the stack for the evaluation of recursive rules. Recursive loop with the accumulator. Various examples of the use of recursive rules.
Lists: structure and meaning, typical predicates using lists. Lists based predicates with accumulator. Operations on lists: inversion, adding, dividing the lists components, etc. Creating lists with the help of the predicate “findall”.
Sorting in the Prolog - different ways of putting data into order and their implementations, together with a discussion of the pros and cons of examples.
Combinatorics - the use of Prolog for creating permutations, combinations and variations of the lists. Examples of combinatorial problems. Methodologies and proceedings in combinatorial problem solving. Solving combinatorial problems by brute-force search (generate and test method). Solving combinatorial problems using backtracking. Comparison of the effectiveness of different methods.
Solving combinatorial problems in a procedural language - example of Java language with JaCoP library.

Laboratory:

1. The structure of the program. Getting known with the basic structural elements of the program in Prolog language. Declaration of predicates, domains, and clauses. External and internal goal in the program. Locality of variables inside a single rule. Elementary logical operations in Prolog.
2. Backtracking and cut. The use of standard predicate "fail" and "!" (cut) to enforce or prevent backtracking. Searching for all the solutions in programs with the of internal goal. Limiting the number of repetitions for the same solution.
3. Recursion. Types of recursions and examples. The use of recursion. The problem of choosing the right type of recursion, comparing different solutions.
4. Lists. The head and tail of the list, the application of lists in different problems. Sorting items in the list.
5. Combinatorial problems (1). Simple combinatorial task and its implementation in Prolog. The use of permutations, combinations or variations of a list in searching for a solution. Brute-force method vs. backtracking method.
6. Combinatorial problems (2). More advanced combinatorial problems. Searching for all the possible solutions.

20. Examination: no

21. Primary sources:

Niederliński, A.: A Gentle Guide to Constraint Logic Programming via ECLiPSe, http://anclp.pl/
Barták, R.: Guide to Prolog Programming, http://kti.mff.cuni.cz/~bartak/prolog/
Visual Prolog, http://www.visual-prolog.com/

22. Secondary sources:

Gatnar, E. K., Stąpor, K.: Prolog, Wydawnictwo PLJ, Warszawa, 1992
Kowalski, R.: Logika w rozwiązywaniu zadań, WNT, Warszawa, 1989
Learn Prolog Now! http://www.learnprolognow.org/
Nillson, U., Maluszynski, J.: Logic, Programming and Prolog (2nd ed.) http://www.ida.liu.se/~ulfni/lpp/

23. Total workload required to achieve learning outcomes

Lp.	Teaching mode :	Contact hours / Student workload hours
1	Lecture	30 / 5
2	Classes	0 / 0
3	Laboratory	15 / 30
4	Project	0 / 0
5	BA/ MA Seminar	0 / 0
6	Other	5 / 5
	Total number of hours	50 / 40

24. Total hours: 90

25. Number of ECTS credits: 3

26. Number of ECTS credits allocated for contact hours: 1.67

27. Number of ECTS credits allocated for in-practice hours (laboratory classes, projects): 1.5

26. Comments:

Approved:

…….
(date, Instructor's signature)

……
(*date , the Director of the Faculty Unit signature)*