SYLLABUS

1. Information regarding the programme

1.1 Higher education institution	**Babes-Bolyai University, Cluj-Napoca**
1.2 Faculty	**Faculty of Mathematics and Computer Science**
1.3 Department	**Department of Computer Science**
1.4 Field of study	**Computer Science**
1.5 Study cycle	**Bachelor**
1.6 Study programme / Qualification	**Computer Science**

2. Information regarding the discipline

2.1 Name of the discipline	**Applications of logics**						
2.2 Course coordinator	**Lecturer Ph.D. Lupea Mihaiela**						
2.3 Seminar coordinator	**Lecturer Ph.D. Lupea Mihaiela**						
2.4. Year of study	**2**	2.5 Semester	**2**	2.6. Type of evaluation	**C**	2.7 Type of discipline	**optional**

3. Total estimated time (hours/semester of didactic activities)

3.1 Hours per week	3	Of which: 3.2 course	2	3.3 seminar/laboratory	1
3.4 Total hours in the curriculum	42	Of which: 3.5 course	28	3.6 seminar/laboratory	14

Time allotment:	hours
Learning using manual, course support, bibliography, course notes	12
Additional documentation (in libraries, on electronic platforms, field documentation)	6
Preparation for seminars/labs, homework, papers, portfolios and essays	10
Tutorship	6
Evaluations	12
Other activities: individual and collective project	12

3.7 Total individual study hours	58
3.8 Total hours per semester	100
3.9 Number of ECTS credits	4

4. Prerequisites (if necessary)

4.1. curriculum	• Computational logic, Data structures and algorithms
4.2. competencies	• Average programming skills in a high level programming language

5. Conditions (if necessary)

5.1. for the course	
5.2. for the seminar /lab activities	• Laboratory with computers; high level programming language environment (.NET or any Java environment a.s.o.)

6. Specific competencies acquired

Professional competencies	• Knowledge of some basic domains in Computer Science: - classical logics (propositional, first-order), temporal, modal and non-monotonic logics from a theoretical perspective - theorem proving for classical logics – methods and techniques (strategies, heuristics) for efficient implementation - formalization of human and mathematical reasoning using logics - programs' verification using logics
Transversal competencies	• Apply classical logics and description logics to solve different tasks in Natural Language Processing (transformation of natural language sentences into predicate formulas, textual entailment, summarization).

7. Objectives of the discipline (outcome of the acquired competencies)

7.1 General objective of the discipline	• Knowledge, understanding and use of basic concepts of theoretical Computer Science • Ability to work independently and/or in a team in order to solve problems in defined professional contexts. • Good programming skills in high-level languages
7.2 Specific objective of the discipline	• Present theoretical concepts of classical logics, modal, temporal and nonmonotonic logics. • Use logics for modeling common-sense reasoning, mathematical reasoning and programs' verification. • Implement ATP systems as educational tools for theorem proving in mathematics and programs'verification. • Understand the applications of logics in solving different tasks of Natural Language domain.

8. Content

8.1 **Course**	Teaching methods	Remarks
1.Classical logics and their extensions (temporal, modal, non-monotonic). Applications of logics in different domains.	Exposure: description, explanation, examples, discussion of case studies	
2. Automated theorem proving (ATP) systems: architecture, examples.	Exposure: description, explanation, examples, discussion of case studies	
3.Data structures used to represent and manipulate logical formulas.	Exposure: description, explanation, examples, discussion of case studies	
4.Binary decision diagrams in propositional logic.	Exposure: description, explanation, examples, discussion of case studies	
5.Semantic tableaux method – a new approach - Considerations for implementing an ATP system, based on this method.	Exposure: description, explanation, examples, discussion of case studies	

6. Sequent and anti-sequent calculi – two complementary direct proof systems. Considerations for the implementation of an ATP system based on these methods.	Exposure: description, explanation, examples, discussion of case studies	
7. Resolution method – refinements (lock, linear, input, unit, ordered); Considerations for implementation.	Exposure: description, explanation, examples, discussion of case studies	
8.Semantic resolution (hyper-resolution, the set-of-support strategy, ordered). Heuristics and tree-searching techniques used in implementation.	Exposure: description, explanation, examples, discussion of case studies	
9. Formalization of common-sense reasoning (knowledge bases).	Exposure: description, explanation, examples, discussion of case studies	
10. Formalization of mathematical reasoning (algebra, geometry).	Exposure: description, explanation, examples, discussion of case studies	
11. Using logics in programs' verification	Exposure: description, explanation, examples, discussion of case studies	
12. Using classical logics in Natural Language Processing.	Exposure: description, explanation, examples, discussion of case studies	
13. Description logics and their applications in Natural Language Processing.	Exposure: description, explanation, examples, discussion of case studies	
14.Written paper		

Bibliography

1. M. Ben-Ari: *Mathematical Logic for Computer Science*, Ed. Springer, 2001.
2. C.L.Chang, R.C.T.Lee: *Symbolic Logic and Mechanical Theorem Proving*, Academic Press, 1973.
3. M.Fitting: *First-order Logic and Automated Theorem Proving,* Texts and Monographs in Computer Science, Springer Verlag, 1990, Second Edition 1996.
4. M.R. Genesereth, N.J. Nilsson: *Logical Foundations of Artificial Intelligence*, Morgan Kaufman, 1992.
5. D.A. Duffy: *Principles of automated theorem proving*, John Willey & Sons, 1991.
6. M. Lupea, A. Mihis: Logici clasice şi circuite logice. Teorie şi exemple, ediția 3, Editura Albastra, Cluj-Napoca, 2011.
7. L.C. Paulson: *Logic and Proof*, Univ. Cambridge, 2000, on-line course.
8. S.Reeves, M.Clarke: *Logic for computer science*, Addison Wesley Publisher Ltd, 1990.
9. R.M.Smullyan: *First-order logic*, Revised Edition, Dover Press, New York, 1996.
10. D.Tatar: *Inteligenta artificiala: demonstrarea automata si NLP*, Editura Microinformatica, Cluj-Napoca, 2001.

8.2 **Laboratory**	Teaching methods	Remarks
1. Working with some existing theorem provers **3TAP, ft, Gandalf, LeanTAP, METEOR, Otter, Prover9, SATURATE, SETHEO , Vampire, PCProve, Jape, etc**.	Explanation, dialogue, case studies	The laboratory is structured as 2 hours classes every second week

2. Students' individual presentations of a dedicated theorem prover.	Dialog, debate	
3. Data structures for logical formulas – implementation.	Explanation, dialogue, case studies	Teams of 2 students have to implement an ATP system based on one of the studied proof methods. A collective project will incorporate all the teams' projects with an appropriate interface.
4. Choose a proof method to implement – specification and implementation.	Explanation, dialogue, case studies	
5. Build a benchmark of knowledge bases used for common-sense and mathematical reasoning.	Explanation, dialogue, case studies	Each student individually.
6. Build a benchmark of examples of simple programs (transformed in program clauses) used in programs' verification.	Explanation, dialogue, case studies	Each student individually.
7. Students' presentation of the collective project.	Dialog, debate, evaluation	

Bibliography

1. W.Bibel: Automated theorem proving, View Verlag, 1987.
2. M. Lupea: Theorem proving in classical logics, electronic format, 2009.
3. M. Possega: Deduction Systems, Institute of Informatics, 2002, on-line course.
4. (ed) A.Thayse: From standard logic to Logic Programming, Ed. J.Wiley, vol1(1989), vol2(1989), vol3(1990).
5. http://www.cs.otago.ac.nz/staffpriv/hans/logiccourseware.html
6. http://www-formal.stanford.edu/clt/ARS/systems.html

9. Corroborating the content of the discipline with the expectations of the epistemic community, professional associations and representative employers within the field of the program

- The course respects the IEEE and ACM Curricula Recommendations for Computer Science studies;
- The course exists in the studying program of some major universities in Romania and abroad;
- The collective project can be used as an educational tool for theorem proving in mathematics and programs'verification.

10. Evaluation

Type of activity	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Share in the grade (%)
10.4 Course	- know the theoretical concepts of the domain; - apply the course concepts in problem solving	Written paper	40%
10.5 Seminar/lab activities	- be able to implement course concepts and algorithms - apply techniques for different classes of programming languages	Software project – implementation of an ATP system	30%

	-be able to model human and mathematical reasoning	Build a benchmark of examples used for testing the ATP system	20%
	-be able to work with a prover and to present the theoretical aspects of the implemented method	Presentation of a dedicated theorem prover	10%
10.6 Minimum performance standards			
➢ At least grade 5 (from a scale of 1 to 10) at both written paper and laboratory work.			

Date	Signature of course coordinator	Signature of seminar coordinator
10.05.2013	Lecturer Ph.D. Lupea Mihaiela	Lecturer Ph.D. Lupea Mihaiela

Date of approval	Signature of the head of department
....	Prof. PhD Pârv Bazil