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	**Computational Logic**						
2.2 Course coordinator	**Lecturer Ph.D. Lupea Mihaiela**						
2.3 Seminar coordinator	**Lecturer Ph.D. Lupea Mihaiela**						
2.4. Year of study	**1**	2.5 Semester	**1**	2.6. Type of evaluation	**exam**	2.7 Type of discipline	**compulsory**

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

3.1 Hours per week	4	Of which: 3.2 course	2	3.3 seminar/laboratory	2
3.4 Total hours in the curriculum	56	Of which: 3.5 course	28	3.6 seminar/laboratory	28
Time allotment:					hours
Learning using manual, course support, bibliography, course notes					20
Additional documentation (in libraries, on electronic platforms, field documentation)					10
Preparation for seminars/labs, homework, papers, portfolios and essays					26
Tutorship					8
Evaluations					30
Other activities:					

3.7 Total individual study hours	94
3.8 Total hours per semester	150
3.9 Number of ECTS credits	6

4. Prerequisites (if necessary)

4.1. curriculum	
4.2. competencies	

5. Conditions (if necessary)

5.1. for the course	
5.2. for the seminar /lab activities	

6. Specific competencies acquired

Professional competencies	• Knowlege of some basic domains in Computer Science: - classical logics (propositionl and first-order) from a theoretical and computational perspective (algorithms and procedures for automated proving) - Boolean functions and logical circuits from a theoretical and computational perspective (simplification methods for Boolean functions) - internal representations of integer and real numbers - simple logical circuits used in the hard component of the computers
Transversal competencies	• Knowledge of the internal representations of integer and real numbers – useful for Computer architecture course • Knowledge of predicate logic - useful in logic programming (Prolog). • The proof methods in classical logics offer a theoretical base for the applicative direction of building automated proof systems useful in mathematics, software engineering, intelligent agents, robotics, natural language. • Knowledge of Boolean functions and logical circuits – useful in electronics.

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

7.1 General objective of the discipline	• To provide the **logical foundations of computer science**: propositional calculus and predicate calculus, theorem proving methods, Boolean algebras and Boolean functions. The connection with logic programming and logical circuits is presented. • To introduce internal **representations** of integer and real numbers.
7.2 Specific objective of the discipline	• Understand how numbers (integer and real) are represented and manipulated internaly by a computer. • Understand simple logical circuits from the hard component of a computer. • Unterstand how to model the human and mathematical reasoning using propositional and predicate logics.

8. Content

8.1 Course	Teaching methods	Remarks
Course 1. Numeration systems: 1. Definitions, representation and operations (algorithms for comparison, addition, subtraction, multiplication division to a digit) of numbers in a base **b.** 2. Conversions between bases using an intermediate base for integer and rational numbers. 3. Rapid conversions (bases 2,4,8,16).	Exposure: description, explanation, examples, discussion of case studies	
Course 2. Internal numbers' representation 1. Representation for unsigned integers, operations. 2. Representation for signed integers: direct code, inverse code, complementary code, operations. 3. Floating-point representation for real numbers.	Exposure: description, explanation, examples, discussion of case studies	

Course 3. Propositional logic – syntax and semantics 1. Syntax: connectives, formulas. 2. Semantics: interpretation, model, consistent formula, inconsistent formula, tautology, logical consequence, truth table for a formula. 3. Laws (logical equivalences): DeMorgan, absorption, commutativity, associativity, distributivity, idempotency. 4. Clauses and normal forms: conjunctive normal form (CNF) and disjunctive normal form (DNF), algorithm for transformation of a formula into DNF and CNF.	Exposure: description, explanation, examples, discussion of case studies, debate, dialog	
Course 4. Propositional logic –formal system 1. Formal (axiomatic) system associated to propositional logic, deduction, theorem. 2. Theorem of deduction and its consequences. 3. Theorem of soundness and theorem of completeness for propositional logic; noncontradiction, coherence and decidability of propositional logic.	Exposure: description, explanation, examples, discussion of case studies, proofs, dialog	
Course 5. Semantic tableaux method – a refutation proof method for propositionl logic. 1. Classes of formulas, decomposition rules, branch (open, closed), construction of a semantic tableau. 2. Theorem of soundness and completeness for the method.	Exposure: description, explanation, examples, discussion of case studies, dialog	
Course 6. Resolution – a refutation proof method for propositional logic 1. Resolution as a formal system. 2. Strategies of resolution: level saturation strategy, set-of–support strategy, deletion strategy.	Exposure: description, explanation, examples, discussion of case studies, proofs	
Course 7. Refinements of propositional resolution 1. Lock resolution, linear resolution. 2.The soundness and completeness properties of general resolution and its refinements.	Exposure: description, explanation, examples, discussion of case studies	
Course 8. Predicate (first-order) logic 1. Syntax: connectives, quantifiers, terms, atoms, formula, clause, literal, closed formula, free formula, the formal (axiomatic) system. 2. Semantics of predicate logic: interpretation, model, valid formula, consistent formula, inconsistent formula, logical consequence. 3. Prenex normal form, Skolem theorem, Skolemization algorithm, clausal normal form. 4. Theorem of soundness and theorem of completeness for predicate logic**;**	Exposure: description, explanation, examples, discussion of case studies, proofs	

noncontradiction, coherence and semi-decidability of predicate logic.		
Course 9. 1. Semantic tableaux method for predicate logic – rules for quantifiers. 2. Substitutions and unifications- theory, algorithm for obtaining the most general unifier o two atoms.	Exposure: description, explanation, examples, discussion of case studies	
Course 10. Resolution in predicate logic 1. Resolution method in predicate logic. Refinements of resolution. The theorem of soundness and completeness. 2. Modeling the common-sense reasoning and mathematical reasoning in predicate logic.	Exposure: description, explanation, examples, discussion of case studies, dialog, debate	
Course 11. Program verification using predicate logic	Exposure: description, explanation, examples	
Course 12. Boolean algebras. Boolean functions . 1. Boolean algebras: definitions, properties, principle of duality, examples; 2. Boolean functions: definitions, maxterms, minterms, the canonic disjunctive form and the canonic conjunctive form, transformation. 3. Definitions: maximal monoms, central monoms, factorization;	Exposure: description, explanation, examples, discussion of case studies	
Course 13. Simplification of Boolean functions 1. Veitch-Karnaugh diagrams method for functions with 2-3-4 variables. 2. Quine's method	Exposure: description, explanation, examples, discussion of case studies	
Course 14. Logical circuits 1. Definitions, representations for basic gates and derived gates. 2. Examples of simple logical circuits: "decoder", "binary codification" circuit, "comparison circuit", "addition" circuit;	Exposure: description, explanation, examples, discussion of case studies	

Bibliography

1. M. Ben-Ari: Mathematical Logic for Computer Science, Ed. Springer, 2001.
2. F.Boian, Bazele Matematice ale Calculatoarelor, Editura Presa Universitara Clujeana, 2002 – library.
3. C.L.Chang, R.C.T.Lee: Symbolic Logic and Mechanical Theorem Proving, Academic Press.
4. M. Cocan, B. Pop: Bazele matematice ale sistemelor de calcul, Editura Albastra, Cluj-Napoca, 2001 – UBB library.
5. M.Fitting: First-order logic and Automated Theorem Proving, Ed.Springer Verlag, 1990.
6. M. Lupea, A. Mihis: Logici clasice şi circuite logice. Teorie şi exemple, ediția 3, Editura Albastra, Cluj-Napoca, 2011 – UBB library.
7. Mihaela Malita, Mircea Malita, Bazele Inteligentei Artificiale, Vol. I, Logici propozitionale, Ed. Tehnica, Bucuresti, 1987 – UBB library.
8. L.C. Paulson: Logic and Proof, Univ. Cambridge, 2000, on-line course.
9. M. Possega: Deduction Systems, Inst. of Informatics, 2002, on-line course.
10. D.Tatar: Bazele matematice ale calculatoarelor, ediția 1999 - UBB library.

8.2 Seminar / laboratory	Teaching methods	Remarks
Seminar 1. Exercises 1. Operations (addition, subtraction, division, multiplication) in different numeration bases. Particular bases: 2,4,8,16. 2. Rapid conversions.	Dialogue, case studies, examples	Seminars' presence is mandatory for at least 70%.
Seminar 2: Exercises 1. Conversions between bases using the methods (calculus in the source base, calculus in the destination base, using an intermediate base) for integer and rational numbers. 2. Representation for unsigned integers, operations.	Dialogue, case studies, examples	
Seminar 3. Exercises **1.** Representation for signed integers: direct code, inverse code, complementary code, operations. **2.** Representations for real numbers: floating-point representation (subunitary normalized mantisa and **mantisa >1).**	Dialogue, case studies, examples	
Seminar 4. Exercises: 1. Using the truth table, decide if a formula is consistent/tautology/inconsistent and write all the models for a consistent formula. 2. Transform a formula into DNF and CNF equivalent forms and using these forms decide if a formula is inconsistent/tautology.	Dialogue, debate, case studies, examples, students presentations	
Seminar 5 **1. One hour - written paper with subjects from courses 1-2 and seminars 1-3.** **2. Exercises** - Apply the theorem of deduction to prove the syllogism rule, separations of premises rule, reunion of premises rule. - Using the axiomatic system prove that a propositional formula is a theorem.	Dialogue, debate, case studies, examples, proofs, students presentations	The presence at the written paper is mandatory.
Seminar 6. Exercises: 1. Build the semantic tableau of a propositional formula, write all its models and anti-models. 2. Check if a propositional formula is a tautology / logical consequence of a set of formulas.	Dialogue, debate, case studies, examples, students presentations	
Seminar 7. Exercises – resolution I: 1. Using resolution check if a set of clauses is inconsistent or not. 2. Check if a propositional formula is a theorem/ deductible from a set of formulas using resolution or one of its strategies.	Dialogue, debate, case studies, examples, students presentations	

Seminar 8. Exercises – resolution II: 1. Apply the refinements of resolution and combinations of strategies and refinements to solve the decisions problems in propositional logic. 2. Details regarding the implementation of lock resolution and linear resolution.	Dialogue, debate, case studies, examples, students presentations	
Seminar 9. Exercises - predicate logic: 1. Transform natural language sentences into predicate formulas. 2. Build models and anti-models for a predicate formula. 3. Build the prenex, Skolem and clausal normal forms for a predicate formula.	Dialogue, debate, case studies, examples, students presentations	
Seminar 10. Exercises: 1. Using the semantic tableaux method solve the decision problems in predicate logic. 2. From a semantic tableau of a predicate formula build the models of that formula. 3. Compute the most general unifier of two or more atoms.	Dialogue, debate, case studies, examples, students presentations	
Seminar 11. Exercises: 1. Check if a predicate formula is a theorem / is deductible from a set of formulas using resolution procedure and its refinements. 2. Model common-sense reasoning on a knowledge base using predicate resolution.	Dialogue, debate, case studies, examples, students presentations	
Seminar 12. Exercises: 1. Build the canonical forms for a Boolean function. 2. Apply Veitch-Karnaugh diagrams method to simplify functions with 2-3-4 variables.	Dialogue, debate, case studies, examples, students presentations	
Seminar 13. Exercises: 1. Apply Quine's method to simplify Boolean functions. 2. Given a Boolean function represented using a tableau containing the values of the function: simplification, implementation of the corresponding logical circuit.	Dialogue, debate, case studies, examples, students presentations	
Seminar 14. Exercises: • Given a Boolean function (with "and", "or", "not", "nor", "nand" operations): simplification with one of the three methods, implementation of the corresponding logical circuit. • Given a logical circuit (with basic and derived gates): write the corresponding boolean function, simplification of this function.	Dialogue, debate, case studies, examples, students presentations	

Bibliography
1. W.Bibel: Automated theorem proving, View Verlag, 1987.
2. Cl.BENZAKEN: Systeme formels. Introduction a la logique, ed.Masson, 1991.
3. J.P.DELAHAYE: Outils logiques pour l'intelligence artificielle, ed.Eyrolls, 1986.
4. D.Tatar: Inteligenta artificiala: demonstrare automata de teoreme si NLP, Ed. Microinformatica, 2001.
5. (ed) A.Thayse: From standard logic to Logic Programming, Ed. J.Wiley, vol1(1989), vol2,vol3(1990).

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 all major universities in Romania and abroad;
- The content of the course offers a theoretical base for the applicative direction of building automated proof systems useful in mathematics, software engineering, intelligent agents, robotics, natural language.

10. Evaluation

Type of activity	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Share in the grade (%)
10.4 Course	- know the basic principles of the domain; - apply the course concepts, methods and algorithms in problem solving	Written paper (regular session) with subjects from courses 3-13.	60%
	- know to perform operations and conversions in different numeration bases - know to represent integer and real numbers	Written paper (seminar 5 -one hour) with subjects from courses 1-2.	15%
10.5 Seminar/lab activities	- solve at home and present at the seminars exercises from an existing benchmark of problems	Seminar activity: responses and individual students presentations of solved exercises.	20%
	- apply theoretical concepts, methods, algorithms to solve difficult exercises at home **or** - implementation of algorithms for operations and conversions in different numeration bases	Optional homework (can increase the final mark)	10%
10.6 Minimum performance standards			
➢ At least grade 5 (from a scale of 1 to 10) at written papers and seminar activity.			

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