SYLLABUS

1. Data about the program of study

1.1	Institution	The Technical University of Cluj-Napoca
1.2	Faculty	Automation and Computer Science
1.3	Department	Computer Science
1.4	Field of study	Computer Science and Information Technology
1.5	Cycle of study	Bachelor of Science
1.6	Program of study/Qualification	Computer Science/ Engineer
1.7	Form of education	Full time
1.8	Subject code	30.

2. Data about the subject

2.1	Subject name					Design with Microprocessors					
2.2	Subject area					Computer Science and Information Technology					
2.3	Course responsible/lecturer					Conf. dr. eng. Tiberiu Marita – tiberiu.marita@cs.utcluj.ro					
2.4	Teachers in charge of applications					As. dr. eng. Mihai Negru – mihai.negru@cs.utcluj.ro					
2.5	Year of study	III	2.6	Semester	5	2.7	Assessment	exam	2.8	Subject category	DID/OB

3. Estimated total time

Sem.	Subject name	Lecture	Applications			Lecture	Applications			Individual study	TOTAL	Credit
		[hours / week.]				[hours / semester]						
			S	L	P		S	L	P			
5	**Design with Microprocessors**	**2**	**-**	**1**	**1**	**28**	**-**	**14**	**14**	**74**	**130**	**5**

3.1	Number of hours per week	4	3.2	of which, course	2	3.3	applications	2
3.4	Total hours in the teaching plan	56	3.5	of which, course	28	3.6	applications	28

Individual study	Hours
Manual, lecture material and notes, bibliography	28
Supplementary study in the library, online and in the field	14
Preparation for seminars/laboratory works, homework, reports, portfolios, essays	28
Tutoring	0
Exams and tests	4
Other activities	0

3.7	Total hours of individual study	74
3.8	Total hours per semester	130
3.9	Number of credit points	5

4. Pre-requisites (where appropriate)

4.1	Curriculum	Logic design, Computer architecture
4.2	Competence	VHDL design, assembly language and C programming

5. Requirements (where appropriate)

5.1	For the course	Black-board/ White-board, projector, computer
5.2	For the applications	Computer, AVR Studio, Cerebot development boards and Digilent Pmods

6. Specific competences

Professional competences	**C2** – Designing hardware, software and communication components (2 credits) **C2.1** – Describing the structure and functioning of computational, communication and software components and systems **C2.2** – Explaining the role, interaction and functioning of hardware, software and communication components **C2.5** - Implementing hardware, software and communication systems **C5** - Designing, managing the lifetime cycle, integrating and ensuring the integrity of hardware, software and

	communication systems (3 credits) **C5.2** - Using interdisciplinary knowledge for adapting the computing system to the specific requirements of the application field **C5.5** - Creating a project including the problem's identification and analysis, its design and development, also proving an understanding of the basic quality requirements
Cross competences	N/A

7. Discipline objectives (as results from the *key competences gained*)

7.1	General objective	Knowledge, understanding and use of concepts like microprocessor/ microcontroller, bus, memory system, data transfer methods, interface circuits and peripheral devices interfacing, analysis and design of microprocessor systems.
7.2	Specific objectives	To achieve the main objective, specific objectives are pursued: • Knowledge of microprocessors and microcontrollers features and capabilities: hardware capabilities, instruction set architecture, assembly language, and programming solutions. • Knowledge of hardware components used with the microprocessors: electrical and logical characteristics, connection modes. • Development of skills to find solutions based on microprocessors or microcontrollers for real problems with average complexity. • Acquaintance with microcontroller development boards and their software programming tools.

8. Contents

8.1. Lecture (syllabus)		Teaching methods	Notes
1	Lecture Overview. Introduction to MP based systems (AVR MCU family).	Oral, blackboard and multimedia, interactive teaching style, consultations, involvement of students in research / design.	N/A
2	AVR registers and instructions.		
3	AVR I/O ports and interrupts.		
4	AVR timer-counters. Generating signals using timers.		
5	Serial communication interfaces: SPI, UART, PS2.		
6	Analogue signal processing using A/D converters.		
7	Microcontroller based applications.		
8	Architecture and instruction set of the Intel 8086 microprocessor.		
9	Interfacing the I/O devices to 8086. I/O data transfer.		
10	The interrupt system of 8086, handling the interrupts.		
11	DMA transfer.		
12	Interfacing memories to 8086.		
13	Simple I/O interfaces. Parallel interfaces		
14	Multiprocessor systems.		

Bibliography

1. B. B. Brey, "INTEL Microprocessors 8086/8088, 80186/80188, 80286, 80386, 80486, Pentium, Prentium ProProcessor, Pentium II, III, 4", ed. 7, Prentice Hall, 2005
2. S. Nedevschi, "Microprocesoare", Editura UTCN, 1994.
3. M.A. Mazidi,S. Naimi, S. Naimi, AVR Microcontroller and Embedded Systems: Using Assembly and C, Prentice Hall, 2010, ISBN 9780138003319.

Online:

4. http://users.utcluj.ro/~tmarita/PMP/PMPcurs.htm

8.2. Applications (Laboratory, Projects)		Teaching methods	Notes
Laboratory			
1	Introduction of the AVR Studio development tool, and of the Cerebot development boards.	Presentation on the blackboard, experiments on	N/A
2	Interfacing the peripheral devices. Using the seven segment display.		

3	Using the timers. Signal generation using timers.	microcontroller development boards (Cerebot) use of specialized IDE design tools (AVR Studio), involvement of students in research / design.	
4	Serial interfaces. Communication between Cerebot and the PC.		
5	The SPI interface. Communication between Cerebot boards. Communication to SPI-capable peripheral devices.		
6	Interfacing the D/A, A/D converters.		
7	Applications with sensors and actuators.		
Projects			
1	Logic design with 8086, EPROM, SRAM		
2	Logic design with 8086, DRAM, flash		
3	Microcontroller application design: audio signal processing.		
4	Microcontroller application design: sensor monitoring, storing and analyzing sensor data.		
5	Microcontroller application design: environment perception, motor control.		
6	Microcontroller application design: communication system.		
7	Project assessment.		

Bibliography

1. Atmel – ATmega64L - 8 bit AVR Microcontroller datasheet, http://www.atmel.com/Images/Atmel-2490-8-bit-AVR-Microcontroller-ATmega64-L_datasheet.pdf
2. Abdul Maalik Khan, AVR Project Book, http://www.digisoft.com.pk/products/avr-project-book
3. Digilent Inc., Rerebot-II, Reference manual, http://digilentinc.com/Data/Products/CEREBOT-II/Cerebot_II_rm_RevB.pdf.

Online:

4. http://users.utcluj.ro/~tmarita/PMP/PMPcurs.htm

9. Bridging course contents with the expectations of the representatives of the community, professional associations and employers in the field

The course is in the Computer and Information Technology field. Its contents combine fundamentals with specific aspects of the used hardware and software tools, accustoming students with the design principles for microprocessor based systems. The course content was discussed with other universities in the country and abroad, and in conjunction with products /development tools offered by companies in Romania, Europe and the USA (e.g. Digilent, Atmel) and is rated by the Romanian government agencies (CNEAA and ARACIS).

10. Evaluation

Activity type	10.1	Assessment criteria	10.2	Assessment methods	10.3	Weight in the final grade
Course		Testing theoretical knowledge and problem solving skills		Written exam		50 %
Applications		Practical skills for problem solving and implementation of specific problems for applications design. Attendance and activity		Colloquium, project evaluation		50 %
10.4 Minimum standard of performance						
Modeling and implementation of typical engineering problems using the theoretical models and applicative tools specific to the domain.						

Course responsible
Conf.dr.eng. Tiberiu Marita

Head of department
Prof.dr.eng. Rodica Potolea

SYLLABUS

1. Data about the program of study

1.1	Institution	The Technical University of Cluj-Napoca
1.2	Faculty	Automation and Computer Science
1.3	Department	Computer Science
1.4	Field of study	Computer Science and Information Technology
1.5	Cycle of study	Bachelor of Science
1.6	Program of study/Qualification	Computer Science/ Engineer
1.7	Form of education	Full time
1.8	Subject code	32.

2. Data about the subject

2.1	Subject name					Functional Programming					
2.2	Subject area					Computer Science and Information Technology					
2.3	Course responsible/lecturer					Assoc. prof. eng. Adrian Petru Groza – Adrian.Groza@cs.utcluj.ro					
2.4	Teachers in charge of applications					Lect. dr. eng. Radu Slavescu – Radu.Razvan.Slavescu@cs.utcluj.ro Lect. dr. eng. Anca Mărginean – anca.marginean@cs.utcluj.ro					
2.5	Year of study	III	2.6	Semester	5	2.7	Assessment	exam	2.8	Subject category	DID/OB

3. Estimated total time

Sem.	Subject name	Lecture	Applications			Lecture	Applications			Individual study	TOTAL	Credit
		[hours / week.]				[hours / semester]						
			S	L	P		S	L	P			
5	**Functional Programming**	**2**	**-**	**2**	**-**	**28**	**-**	**28**	**-**	**74**	**130**	**5**

3.1	Number of hours per week	4	3.2	of which, course	2	3.3	applications	2
3.4	Total hours in the teaching plan	56	3.5	of which, course	28	3.6	applications	28

Individual study	Hours
Manual, lecture material and notes, bibliography	28
Supplementary study in the library, online and in the field	14
Preparation for seminars/laboratory works, homework, reports, portfolios, essays	28
Tutoring	4
Exams and tests	
Other activities	

3.7	Total hours of individual study	74
3.8	Total hours per semester	130
3.9	Number of credit points	5

4. Pre-requisites (where appropriate)

4.1	Curriculum	
4.2	Competence	

5. Requirements (where appropriate)

5.1	For the course	Basic notions of programming
5.2	For the applications	Linux

6. Specific competences

Professional competences	**C2** Designing hardware, software and communication components (4 credits) **C2.1** Describing the structure and functioning of computational, communication and software components and systems **C2.2** Explaining the role, interaction and functioning of hardware, software and communication components **C2.3** Building the hardware and software components of some computing systems using algorithms, design methods, protocols, languages, data structures, and technologies **C2.4** Evaluating the functional and non-functional characteristics of the computing systems using specific metrics **C4** Improving the performances of the hardware, software and communication systems (1 credit) **C4.1** Identifying and describing the defining elements of the performances of the hardware, software and communication systems **C4.3** Applying the fundamental methods and principles for increasing the performances of the hardware and software
Cross competences	N/A

7. Discipline objectives (as results from the *key competences gained*)

7.1	General objective	Increasing the ability to develop more correct and concise code
7.2	Specific objectives	Writing better code with the concepts introduced by functional programming: high order functions, lazy evaluation, lambda calculus, infinite structure.

8. Contents

8.1. Lecture (syllabus)		Teaching methods	Notes
1	Introduction. Programming Paradigms	Slides, Various student engagement techniques New examples Quick individual work (1 minute) Homework after each class dicusssed at the beginning of the next class.	
2	Basic concepts of programming in Hugs, ML, Lisp: functions, constants, primitive data types, recursion, tuples, infix operators, evaluation.		
3	Basic concepts: local declarations, polymorphism.		
4	Lists: list construction, basic operations on lists.		
5	Lists: polymorphic equality.		
6	Lists: list operators (generators, filters, list expressions).		
7	Trees: alternative data, pattern matching, exceptions, binary trees (list-tree conversions).		
8	Trees: binary trees (binary search trees, AVL balanced trees, examples (operations on sets)).		
9	Trees: binary trees (examples (Huffman codes)), propositional reasoner (example).		
10	Higher-order functions: anonymous functions, partial application, functions as data, data as functions, combinator functions, functionals for lists (list operator style, style without lists).		
11	Infinite data: lazy evaluation, unbounded objects, circular structures.		
12	Transformation and reasoning: structural induction, equivalence of functions, structural induction on trees, induction on number of nodes, general principle of induction.		
13	Lambda calculus: Lambda notation, conversions, combinators.		
14	Para-functional programming: basic language, mapped expressions, eager expressions.		

Bibliography

1. Haskell - A Purely Functional Language, http://www.haskell.org/
2. I.A. Leția, Programare funcțională, Ed. UTPres, UTCN, 1996.
3. H. Conrad Cunningham, Notes on Functional Programming with Haskell, 2007
4. Raul Rojas, A Tutorial Introduction to the Lambda Calculus, FU Berlin, WS-97/98

8.2. Applications (Laboratory)		Teaching methods	Notes
1	Lisp objects, form evaluation, primitive Lisp functions.	New examples	

2	Internal representation, control of evaluation, function definition. Recursion and iteration.	Tracing algorithms Midterm assessment Miniprojects	
3	Scope of variables, iterative forms. LAMBDA-expressions, higher-order functions, mapping.		
4	Association lists, properties, arrays and structures. Macrodefinitions, functions as data, surgery.		
5	Trees in Lisp. Graphs and backtracking.		
6	Pattern matching. Symbolic processing.		
7	Lisp microinterpreter. Review of programming in Lisp, in preparation for the lab test.		
8	Lab test (Programming in Lisp).		
9	ML Lists, Recursion,.		
10	ML type checking		
11	ML Trees		
12	Haskell – High order functions		
13	Haskell -Lazy evaluation, circular lists, infinite lists.		
14	Lab test (Programming in ML and Haskell).		

Bibliography

1. I.A. Leţia, E.Şt. Chifu, C. Cenan, Programare funcţională. Îndrumător de laborator, Ed. Casa cărţii de ştiinţă, 1999.
2. David S. Touretzky, Common Lisp: A Gentle Introduction to Symbolic Computation, The Benjamin/Cummings Publishing Company,Inc, 1989
3. Andrew Cumming, A gentle introduction to ML, Napier University, Edinburgh, 2013

9. Bridging course contents with the expectations of the representatives of the community, professional associations and employers in the field

The content of the class is similar to the contents taught at other international universities. The students are encouraged to indentify elements of functional programming in the current practice of IT companies running at the local level.

10. Evaluation

Activity type	10.1	Assessment criteria	10.2	Assessment methods	10.3	Weight in the final grade
Course		Understanding functional programming elements, Class participation, Homework		Midterm assessment, Writing exam		60
Applications		Quantity and quality of code in Lisp, Haskell and ML		Midterm assessment, Practical exam		40
10.4 Minimum standard of performance						
Understanding and code writing for the following concepts; Recursion, High Order Functions, Pattern Matching						

Course responsible
Assoc. prof. eng. Adrian Petru Groza

Head of department
Prof.dr.eng. Rodica Potolea

SYLLABUS

1. Data about the program of study

1.1	Institution	The Technical University of Cluj-Napoca
1.2	Faculty	Automation and Computer Science
1.3	Department	Computer Science
1.4	Field of study	Computer Science and Information Technology
1.5	Cycle of study	Bachelor of Science
1.6	Program of study/Qualification	Computer Science/ Engineer
1.7	Form of education	Full time
1.8	Subject code	33.

2. Data about the subject

2.1	Subject name					Software Engineering					
2.2	Subject area					Computer Science and Information Technology					
2.3	Course responsible/lecturer					Prof. dr eng. Eneia Nicolae Todoran – Eneia.Todoran@cs.utcluj.ro					
2.4	Teachers in charge of applications					Assoc.prof. dr. Mitrea Paulina – Paulina.Mitrea@cs.utcluj.ro, Sl. dr. eng. Mitrea Delia Delia.Mitrea@cs.utcluj.ro					
2.5	Year of study	III	2.6	Semester	5	2.7	Assessment	exam	2.8	Subject category	DID/OB

3. Estimated total time

Sem.	Subject name	Lecture	Applications			Lecture	Applications			Individual study	TOTAL	Credit
		[hours / week.]				[hours / semester]						
			S	L	P		S	L	P			
5	**Software Engineering**	**2**	**-**	**1**	**1**	**28**	**-**	**14**	**14**	**74**	**130**	**5**

3.1	Number of hours per week	4	3.2	of which, course	2	3.3	applications	2
3.4	Total hours in the teaching plan	56	3.5	of which, course	28	3.6	applications	28

Individual study	Hours
Manual, lecture material and notes, bibliography	25
Supplementary study in the library, online and in the field	17
Preparation for seminars/laboratory works, homework, reports, portfolios, essays	17
Tutoring	5
Exams and tests	10
Other activities	0

3.7	Total hours of individual study	74
3.8	Total hours per semester	130
3.9	Number of credit points	5

4. Pre-requisites (where appropriate)

4.1	Curriculum	Object Oriented Programming, Programming Techniques
4.2	Competence	Competences acquired in the above disciplines

5. Requirements (where appropriate)

5.1	For the course	Blackboard, projector, computer
5.2	For the applications	Computers, specific software

6. Specific competences

Professional competences	**C3 -** Problems solving using specific Computer Science and Computer Engineering tools (2 credits) **C3.1 -** Identifying classes of problems and solving methods that are specific to computing systems **C3.2 -** Using interdisciplinary knowledge, solution patterns and tools, making experiments and interpreting their results **C3.3 -** Applying solution patterns using specific engineering tools and mehods **C3.5 -** Developing and implementing informatic solutions for concrete problems **C4 -** Improving the performances of the hardware, software and communication systems (1 credit)

	C4.2 - Explaining the interaction of the factors that determine the performances of the hardware, software and communication systems **C4.4 -** Choosing the criteria and evaluation methods of the performances of the hardware, software and communication systems **C5 -** Designing, managing the lifetime cycle, integrating and ensuring the integrity of hardware, software and communication systems (2 credits) **C5.1 -** Specifying the relevant criteria regarding the lifetime cycle, quality, security and the computing system's interaction with the environment and the human operator **C5.5 -** Creating a project including the problem's identification and analysis, its design and development, also proving an understanding of the basic quality requirements
Cross competences	N/A

7. Discipline objectives (as results from the *key competences gained*)

7.1	General objective	The overall objective of discipline consists in the study and application of systematic, disciplined and quantifiable approaches in software systems development
7.2	Specific objectives	• Study and application of software development processes • Understanding the specific activities of software engineering • Knowledge of software engineering models • Knowledge of specific tools that can assist software engineers in the specification, design and validation process • Knowledge of methods for software modeling and performance analysis • Application of processes, methods and tools in small to medium-sized software projects

8. Contents

8.1. Lecture (syllabus)		Teaching methods	Notes
1	Introduction and overview of the course		
2	Software development paradigms: basic paradigms ('waterfall', prototyping, reusable components, formal methods), evolutionary paradigms (incremental development, spiral model, concurrent engineering)		
3	Modern processes: the unified process, agile methods and extreme programming		
4	Basic activities (specification, development, validation, evolution): concepts, principles, processes		
5	Conventional methods: introduction to structured analysis and design		
6	Developing requirements: domain analysis, types of requirements, techniques for gathering requirements, capturing requirements as use cases		
7	Modeling with classes: UML class diagrams, the process of developing class diagrams, the semantics of UML class diagrams, implementing class diagrams in Java		
8	Modeling interactions and behavior: UML interaction (sequence and communication), state and activity diagrams. Software performance modeling and analysis.		
9	Architecting and designing software: design principles (increase cohesion, reduce coupling, etc.), architectural patterns (Multi-Layer, Pipe-and-Filter, etc.)		
10	Testing and inspecting to ensure high quality: testing techniques (equivalence partitioning, path testing, etc.) and integration strategies (top-down, bottom-up, scenario-based), inspections		
11	Use case driven development: use case specifications, analysis, design and implementation to realize the use cases, testing the use cases		
12	Program specifications: pre and post assertions, well-founded induction,		

	declarative prototyping		
13	Software engineering based on formal methods: basic concepts, formal specification languages, formal verification		
14	Model-based specification using Z: notation, schema calculus, methodology		

Bibliography

1. I. Sommerville. *Software Engineering* (6th, 7th, 8th, 9th editions). Addison Wesley (2001, 2004, 2006, 2010).
2. T. Lethbridge, R. Laganiere. *Object-Oriented Software Engineering: Practical Software Development using UML and Java* (2nd edition). McGraw-Hill, 2005. http://www.lloseng.com.
3. E. Currie. *The essence of Z*. Prentice Hall, 1999.
4. PRISM manual, 2014. http://www.prismmodelchecker.org/manual/
5. E.N. Todoran. *Inginerie software: studii in prototipizare si specificare formala*. Mediamira, Cluj-Napoca, 2006.

8.2. Applications (Laboratory)		Teaching methods	Notes
1	OCSF – an object client-server framework for reuse oriented development		
2	Simple Chat - an instant messaging system based on OCSF (1)		
3	Simple Chat - an instant messaging system based on OCSF (2)		
4	Using software modeling CASE tools (1): UML use case, class and interaction diagrams		
5	Using software modeling CASE tools (2): UML state, component and deployment diagrams		
6	Using CASE tools for performance software modeling and analysis: PRISM, PEPA		
7	Test cases design with JUnit		
The **project class** attempts to simulate various aspects of the real world of software engineering. The students define the problem to be solved and the scope of the project under the supervision of the teaching assistant. Working alone is permitted, but they are encouraged to work in teams. The students must employ the paradigms and the software development methods that are presented in the taught course. They are expected to deliver three iterations of the project with predefined deadlines. For a traditional 'waterfall' project the deadlines correspond to requirements specification, design, and the final deliverable.			

Bibliography

1. T. Lethbridge, R. Laganiere. *Object-Oriented Software Engineering: Practical Software Development using UML and Java* (2nd edition). McGraw-Hill, 2005. http://www.lloseng.com.

9. Bridging course contents with the expectations of the representatives of the community, professional associations and employers in the field

Software Engineering is a well-established discipline in Computer Science and Information Technology. In this course, students acquire basic knowledge related to software development (paradigms, methods and tools) and learn to apply systematic and quantifiable approaches in the development of software systems. Course content has been developed based on interaction with specialists in Software Engineering from Romania, Europe and Canada and has been rated by Romanian government agencies (CNEAA and ARACIS).

10. Evaluation

Activity type	10.1	Assessment criteria	10.2	Assessment methods	10.3	Weight in the final grade
Course		Problem solving skills		Written examination		75%
Applications		Software design and validation skills		Laboratory colloquium, Project assessment		25%
10.4 Minimum standard of performance						
Development of a medium size software project using the skills taught in the Software Engineering course						

Course responsible
Prof. dr. eng. Eneia Todoran

Head of department
Prof.dr.eng. Rodica Potolea

SYLLABUS

1. Data about the program of study

1.1	Institution	The Technical University of Cluj-Napoca
1.2	Faculty	Automation and Computer Science
1.3	Department	Computer Science
1.4	Field of study	Computer Science and Information Technology
1.5	Cycle of study	Bachelor of Science
1.6	Program of study/Qualification	Computer Science/ Engineer
1.7	Form of education	Full time
1.8	Subject code	34.

2. Data about the subject

2.1	Subject name					Introduction to Artificial Intelligence					
2.2	Subject area					Computer Science and Information Technology					
2.3	Course responsible/lecturer					Prof. dr. eng. Leția Ioan Alfred – Ioan.Alfred.Letia@cs.utcluj.ro					
2.4	Teachers in charge of applications					Assoc. prof. dr. eng. Groza Adrian – Adrian.Groza@cs.utcluj.ro Lect. dr. eng. Marginean Anca – Anca.Marginean@cs.utcluj.ro					
2.5	Year of study	III	2.6	Semester	5	2.7	Assessment	exam	2.8	Subject category	DID/OB

3. Estimated total time

Sem.	Subject name	Lecture	Applications			Lecture	Applications			Individual study	TOTAL	Credit
		[hours / week.]				[hours / semester]						
			S	L	P		S	L	P			
5	**Introduction to Artificial Intelligence**	**2**	**-**	**2**	**-**	**28**	**-**	**28**	**-**	**48**	**104**	**4**

3.1	Number of hours per week	4	3.2	of which, course	2	3.3	applications	2
3.4	Total hours in the teaching plan	56	3.5	of which, course	28	3.6	applications	28

Individual study	Hours
Manual, lecture material and notes, bibliography	18
Supplementary study in the library, online and in the field	5
Preparation for seminars/laboratory works, homework, reports, portfolios, essays	10
Tutoring	6
Exams and tests	9
Other activities	0

3.7	Total hours of individual study	48
3.8	Total hours per semester	104
3.9	Number of credit points	4

4. Pre-requisites (where appropriate)

4.1	Curriculum	Logic Programming, Functional Programming
4.2	Competence	Elementary fundamentals of programming

5. Requirements (where appropriate)

5.1	For the course	Projector, Computer
5.2	For the applications	Computers with Linux, Specific Software

6. Specific competences

Professional competences	**C3 –** Problems solving using specific Computer Science and Computer Engineering tools (1 credit) **C3.1 –** Identification of classes of problems and the methods to solve them characteristic of information systems **C3.2 –** Usage of interdisciplinary knowledge, patterns of solutions and tools, experimentation and interpretation of their results **C3.3 –** Aplication of solution patterns using engineering tools and methods **C3.4 –** Comparative evaluation, including experiments, of alternative solutions, to optimize performance **C3.5 –** Development and implementation of computational solutions for concrete problems **C5 –** Designing, managing the lifetime cycle, integrating and ensuring the integrity of hardware, software and communication systems (1 credit) **C5.1 –** Stating the criteria relevant to quality, security and system interaction with the environment and human operator **C5.2 –** Usage of interdisciplinary knowledge for the adaptation of the informatic system to the requirements of the application domain **C5.3** – Using fundamental principles and methods for ensuring the security, the safety and ease of exploitation of the computing systems **C6 –** Designing intelligent systems (2 credits) **C6.1 –** Describing the intelligent systems' components **C6.3 –** Applying the main methods and principles for specifying solutions for typical problems using intelligent systems **C6.5 –** Developing and implementing professional projects for intelligent systems
Cross competences	N/A

7. Discipline objectives (as results from the *key competences gained*)

7.1	General objective	Knowledge of representation and reasoning of fundamental problems of artificial intelligence
7.2	Specific objectives	Fundamental search methods, Usage of first-order logic and description logics, Basic planning representation and solving methods

8. Contents

8.1. Lecture (syllabus)		Teaching methods	Notes
1	Introduction.	Slides, Algorithms, Quality of solutions, Exceptions, Limits in the representation of the real world,	
2	Intelligent Agents: behavior, environments, structure.		
3	Solving Problems by Searching: uninformed, searching with partial information.		
4	Informed Search Methods and Exploration: heuristics, local search algorithms and optimization problems, local search in continuous spaces.		
5	Constraint Satisfaction Problems: backtracking, local search.		
6	Adversarial Search: alpha-beta pruning, imperfect, real-time decisions, games that include an element of chance.		
7	Logical Agents: knowledge-based agents, propositional logic, effective propositional inference.		
8	First-Order Logic.		
9	Inference in First-Order Logic: forward, backward chaining, resolution.		
10	Knowledge Representation.		
11	Description logics: description languages, terminologies, world description, inferences, reasoning algorithms, language extensions		
12	Planning: partial-order planning, planning graphs.		
13	Planning and Acting in the Real World: schedules and resources, hierarchical network planning, conditional planning, execution monitoring and re-planning, continuous planning.		
14	Course Overview.		
Bibliography			

1. Artificial Intelligence: A Modern Approach: Russell, Norvig, Prentice Hall, 2002 2. Basic Description Logics: Baader, Nutt, CUP, 2003			
8.2. Applications (Laboratory)		Teaching methods	Notes
1	Introduction to the documentation for the assignment	Platform, Documentation, Testing, Examples, New examples	
2	Studying the documentation for the assignment		
3	Studying the design of the tool		
4	Practicing the exercises provided in the archive		
5	Understanding the main parts of the software		
6	Running the system by tracing at high level		
7	Mastering the running of the system and the examples provided		
8	Conceptual design of new examples		
9	Code for the new examples		
10	Testing and debugging the new cases		
11	Measuring the performance of the system		
12	Documenting the new scenarios		
13	Comparison of the differences between the cases developed and those provided		
14	Final evaluation of the exercises developed		
Bibliography 1. Various Artificial Intelligence Tools from the WWW			

9. Bridging course contents with the expectations of the representatives of the community, professional associations and employers in the field

The textbook is one of the most known and used one in the world of the best universities, continuously assessed by the university and research community in the world regarding its influence and use in the software oriented companies.

10. Evaluation

Activity type	10.1	Assessment criteria	10.2	Assessment methods	10.3	Weight in the final grade
Course		Problems and theoretical concepts		Written exam		80%
Applications		Usage of specific tools on the examples developed and tested by the students		Evaluation in the laboratory		20%
10.4 Minimum standard of performance						
Representation of knowledge and its use in solving specific problems using specific tools						

Course responsible
Prof. dr. eng. Ioan Alfred Letia

Head of department
Prof.dr.eng. Rodica Potolea

SYLLABUS

1. Data about the program of study

1.1	Institution	The Technical University of Cluj-Napoca
1.2	Faculty	Automation and Computer Science
1.3	Department	Computer Science
1.4	Field of study	Computer Science and Information Technology
1.5	Cycle of study	Bachelor of Science
1.6	Program of study/Qualification	Computer Science/ Engineer
1.7	Form of education	Full time
1.8	Subject code	35.

2. Data about the subject

2.1	Subject name					Economic legislation					
2.2	Subject area					Computer Science and Information Technology					
2.3	Course responsible/lecturer					Lect. dr. jur. Roxana Cordos					
2.4	Teachers in charge of applications					-					
2.5	Year of study	III	2.6	Semester	5	2.7	Assessment	Colloquium	2.8	Subject category	DC/OB

3. Estimated total time

Sem.	Subject name	Lecture	Applications			Lecture	Applications			Individual study	TOTAL	Credit
		[hours / week.]				[hours / semester]						
			S	L	P		S	L	P			
5	**Economic legislation**	**2**	**-**	**-**	**-**	**28**	**-**	**-**	**-**	**24**	**52**	**2**

3.1	Number of hours per week	2	3.2	of which, course	2	3.3	applications	-
3.4	Total hours in the teaching plan	28	3.5	of which, course	28	3.6	applications	-

Individual study	Hours
Manual, lecture material and notes, bibliography	18
Supplementary study in the library, online and in the field	2
Preparation for seminars/laboratory works, homework, reports, portfolios, essays	
Tutoring	2
Exams and tests	2
Other activities	

3.7	Total hours of individual study	24
3.8	Total hours per semester	52
3.9	Number of credit points	2

4. Pre-requisites (where appropriate)

4.1	Curriculum	Not the case
4.2	Competence	Not the case

5. Requirements (where appropriate)

5.1	For the course	Not the case
5.2	For the applications	Not the case

6. Specific competences

Professional competences	N/A

Cross competences	**CT3** – Demonstrating the spirit of initiative and action for updating professional, economical and organizational culture knowledge (2 credits)

7. Discipline objectives (as results from the *key competences gained*)

7.1	General objective	Applying the general and specific knowledge of technical culture in solving the business issues in this field
7.2	Specific objectives	Knowing the basic legislation in the field and finding solution for different types of problems.

8. Contents

8.1. Lecture (syllabus)		Teaching methods	Notes
1	General notions of economic legislation.	Modern teaching methods	
2	The merchants		
3	Commerce acts		
4	Commercial contracts –general notions		
5	Classification of contracts		
6	The contract of sale		
7	The contract of transportation		
8	The contract of storage, mandate, renting		
9	The contract of leasing		
10	General rules applied to commercial societies		
11	The constitutive act of a firm		
12	Society on shares		
13	The society with limited responsibility		
14	The society in collective name, with simple sleeping partners and on shares sleeping partners.		

Bibliography

1. S.Angheni, M.Volonciu, C.Stoica, M.Lostun, Drept comercial, Ed. Oscar Print, Bucuresti, 2000
2. I.L.Georgescu, I.Bacanu, Drept comercial român, vol.II, Ed.Lumina Lex, Bucuresti, 2000
3. B. Stefanescu, O.Capatâna, Dictionar juridic de comert exterior, Bucuresti, 1986.
4. S.Carpenaru, Drept comercial, Ed.All, Bucuresti, 2007
5. Bodu S., Drept commercial completat cu notiuni fundamentale de drept civil- curs universitar, 2005 (biblioteca UTCN)

8.2. Applications (Seminars, Laboratory, Projects)		Teaching methods	Notes
1	-		

9. Bridging course contents with the expectations of the representatives of the community, professional associations and employers in the field

The students will have the possibility to learn how to put into practice a business idea in the studied domain.

10. Evaluation

Activity type	10.1	Assessment criteria	10.2	Assessment methods	10.3	Weight in the final grade
Course		Colloquium		Written test		100%
Applications						
10.4 Minimum standard of performance						
Grade 5						

Course responsible
Lect. dr. jur. Roxana Cordos

Head of department
Prof.dr.eng. Rodica Potolea

SYLLABUS

6. Data about the program of study

1.1	Institution	The Technical University of Cluj-Napoca
1.2	Faculty	Automation and Computer Science
1.3	Department	Computer Science
1.4	Field of study	Computer Science and Information Technology
1.5	Cycle of study	Bachelor of Science
1.6	Program of study/Qualification	Computer Science/ Engineer
1.7	Form of education	Full time
1.8	Subject code	36.

7. Data about the subject

2.1	Subject name					Graphical Processing Systems					
2.2	Subject area					Computer Science and Information Technology					
2.3	Course responsible/lecturer					Prof. dr. eng. Gorgan Dorian – dorian.gorgan@cs.utcluj.ro					
2.4	Teachers in charge of applications					As. drd. eng. Melenti Cornelia, S.l. dr. eng. Bacu Victor, {cornelia.melenti, victor.bacu}@cs.utcluj.ro					
2.5	Year of study	III	2.6	Semester	5	2.7	Assessment	exam	2.8	Subject category	DID/OB

8. Estimated total time

Sem.	Subject name	Lecture	Applications			Lecture	Applications			Individual study	TOTAL	Credit
		[hours / week.]				[hours / semester]						
			S	L	P		S	L	P			
5	**Graphical Processing Systems**	**2**	**-**	**2**	**-**	**28**	**-**	**28**	**-**	**48**	**104**	**4**

3.1	Number of hours per week	4	3.2	of which, course	2	3.3	applications	2
3.4	Total hours in the teaching plan	56	3.5	of which, course	28	3.6	applications	28

Individual study	Hours
Manual, lecture material and notes, bibliography	20
Supplementary study in the library, online and in the field	6
Preparation for seminars/laboratory works, homework, reports, portfolios, essays	10
Tutoring	3
Exams and tests	9
Other activities	0

3.7	Total hours of individual study	48
3.8	Total hours per semester	104
3.9	Number of credit points	4

9. Pre-requisites (where appropriate)

4.1	Curriculum	Computer programming (C language) Elements of Computer Assisted Graphics
4.2	Competence	Applications development in C programming language, Graphical systems architecture, The graphical processing pipeline

10. Requirements (where appropriate)

5.1	For the course	Projector, computer
5.2	For the applications	Laboratory attendance is mandatory Study of laboratory materials from the server

6. Specific competences

Professional competences	**C4 –** Improving the performances of the hardware, software and communication systems (4 credits) **C4.1 –** Identifying and describing the defining elements of the performances of the hardware, software and communication systems **C4.2 –** Explaining the interaction of the factors that determine the performances of the hardware, software and communication systems **C4.3 –** Applying the fundamental methods and principles for increasing the performances of the hardware, software and communication systems **C4.4 –** Choosing the criteria and evaluation methods of the performances of the hardware, software and communication systems **C4.5 –** Developing professional solutions for hardware, software and communication systems based on performance optimization
Cross competences	N/A

7. Discipline objectives (as results from the *key competences gained*)

7.1	General objective	Study and experiment with the 3D photorealistic algorithms. Development of 2D and 3D graphics applications.
7.2	Specific objectives	1. Creation of the graphic model of a 3D scene of objects 2. Implementation and usage of the fundamental 3D graphics algorithms that can be found in the core of a graphic system 3. Development of graphic applications in a high-level programming language (C, C++) based on graphics libraries (ex. OpenGL) 4. Implementation of the main phases of the graphics transformation pipeline, in order to transform a 3D scene into an image.

8. Contents

8.1. Lecture (syllabus)		Teaching methods	Notes
1	Computational graphics	New multimedia teaching approaches will be used in classes. The course is interactive and includes demonstrations that exemplify graphical methods and algorithms.	During the semester and before each exam there are a few preparation hours planned.
2	Hidden line and surface removal algorithms. Part 1		
3	Hidden line and surface removal algorithms. Part 2		
4	3D objects modeling		
5	Particles based models		
6	Polygonal objects rendering. Part 1		
7	Polygonal objects rendering. Part 2		
8	Illumination models. Local reflection model. Phong model		
9	Shadow computation		
10	Texture mapping. Part1		
11	Texture mapping. Part2		
12	Global reflection models. Ray-tracing algorithm		
13	Global reflection models. Radiosity algorithm		
14	Graphical animation		

Bibliography

4. Watt A., "*3D Computer Graphics*". Addison-Wesley, 1998.
5. Watt A., Policarpo F.:"*3D Games. Real-time Rendering and Software Technology*". Addison-Wesley, 2001.
6. Akenine-Moller T., Haines E., "*Real-Time Rendering"*. A.K. Peters 2nd edition, 2002.
7. Foley J.D., van Dam, A., Feiner, S.K., Hughes, J.F., "*Computer Graphics. Principles and Practice*". Addison-Wesley Pblishing Comp., 1992.
8. Gorgan D., Rusu, D., "*Elemente de Grafică pe Calculator*". Cluj-Napoca, 1996.

In virtual library

1. Curse and practical works, http://cgis.utcluj.ro
2. Course resources, http://cgis.utcluj.ro/didactic

8.2. Applications (Laboratory)		Teaching methods	Notes
1	Introduction. Administrative	Documentation and examples will	Each student will have to
2	OpenGL application framework		

3	Graphics primitives in OpenGL	be available to the students, prior to the laboratory classes, on a dedicated server. The students will work independently but will also be assisted by the teacher.	develop a specific project based on the knowledge acquired at the laboratory hours.
4	Graphics transformations in OpenGL		
5	Data model and file formats		
6	Projections and clipping planes in OpenGL		
7	Lighting model in OpenGL		
8	Texture mappings in OpenGL		
9	Shadow computation in OpenGL applications		
10	Graphical User Interface in OpenGL Applications - Part 1		
11	Graphical User Interface in OpenGL Applications - Part 2		
12	Ray tracing algorithm		
13	Bump mapping		
14	Assessment		

Bibliography

In virtual library

1. Curse and practical works, http://cgis.utcluj.ro

9. Bridging course contents with the expectations of the representatives of the community, professional associations and employers in the field

This discipline is integrated into the Computers and Information Technology domain. The content is classic, yet modern, and introduces to students the fundamentals of 3D graphic systems and algorithms. The content of this discipline has been aligned with the information presented in similar disciplines from other major universities and companies from Romania, Europe and USA and has been evaluated by the authorized Romanian governmental agencies (CNEAA and ARACIS).

10. Evaluation

Activity type	10.1	Assessment criteria	10.2	Assessment methods	10.3	Weight in the final grade
Course		The written exam tests the understanding of the information presented in classes and the ability to apply this knowledge. The activity in class evaluates the active involvement of the students in the teaching process and their participation to the discussions, debates and other class activities during the entire semester.		Evaluation is performed through written exam (E).		60% (E)
Applications		Laboratory assessment evaluates the practical abilities obtained by the students. Through homework assignments the students have the opportunity to develop their skill in applying the notions, concepts and methods presented in class.		Evaluation is performed through written exam and homework assessment.		40%
10.4 Minimum standard of performance						
Condiţie de promovare: N≥5						

Course responsible
Prof.dr.eng. Dorian Gorgan

Head of department
Prof.dr.eng. Rodica Potolea