SYLLABUS

1. Data about the program of study

1.1 Institution	The Technical University of Cluj-Napoca
1.2 Faculty	Faculty of 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	48.20

2. Data about the subject

2.1 Subject name	***Translator design***				
2.2 Course responsible / lecturer	Assoc. prof. dr. eng. Emil Şefan Chifu - emil.chifu@cs.utcluj.ro				
2.3 Teachers in charge of seminars / laboratory / project	Assoc. prof. dr. eng. Emil Ştefan Chifu - emil.chifu@cs.utcluj.ro				
2.4 Year of study	IV	2.5 Semester	1	2.6 Type of assessment (E - exam, C - colloquium, V - verification)	E
2.7 Subject category	*DF – fundamentală, DD – în domeniu, DS – de specialitate, DC – complementară*				DS
	DI – Impusă, DOp – opțională, DFac – facultativă				DOp

3. Estimated total time

3.1 Number of hours per week	5	of which:	Course	2	Seminars		Laboratory	2	Project	1
3.2 Number of hours per semester	70	of which:	Course	28	Seminars		Laboratory	28	Project	14
3.3 Individual study:										
(a) Manual, lecture material and notes, bibliography										25
(b) Supplementary study in the library, online and in the field										15
(c) Preparation for seminars/laboratory works, homework, reports, portfolios, essays										27
(d) Tutoring										10
(e) Exams and tests										3
(f) Other activities:										0

3.4 Total hours of individual study (suma (3.3(a)...3.3(f)))	80
3.5 Total hours per semester (3.2+3.4)	150
3.6 Number of credit points	6

4. Pre-requisites (where appropriate)

4.1 Curriculum	Formal Languages and Translators, Computer Programming, Data Structures and Algorithms
4.2 Competence	- Basic knowledge of programming and data structures (preferably in the C and Java languages) - Concepts of generative grammars and formal languages - To know the basic principles in the design of interpretors and translators for languages artificial - Basic knowledge of relational databases and web applications

5. Requirements (where appropriate)

5.1. For the course	N/A
5.2. For the applications	Computers, specific software

6. Specific competence

6.1 Professional competences	**C4 -** Improving the performances of the hardware, software and communication systems (2 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 **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 - **C5.2** - Using interdisciplinary knowledge for adapting the computing system to the specific requirements of the application field - **C5.3** - Using fundamental principles and methods for ensuring the security, the safety and ease of exploitation of the computing systems - **C5.4** - Proper utilization of the quality, safety and security standards in the field of information processing - **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 **C6 -** Designing intelligent systems (1 credit) - **C6.1** - Describing the components of intelligent systems - **C6.2** - Using domain-specific tools for explaining and understanding the functioning of intelligent systems - **C6.3** - Applying the fundamental methods and principles - for specifying solutions for typical problems using intelligent systems - **C6.4** - Choosing the criteria and evaluation methods for the quality, performances and limitations of intelligent systems - **C6.5** - Developing and implementing professional projects for intelligent systems
6.2 Cross competences	N/A

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

7.1 General objective	- To know the phases of programming language translators: lexical analysis, syntactic analysis, and code generation. - To master the phases of Natural Language Processing and the BERT language models.
7.2 Specific objectives	- To know the classes of languages for which efficient translators and interpreters can be implemented. - To know the rules for processing typical statements for interpreters. - By using the Prolog language, to build DCG parsers for natural language. - To implement in the NLTK toolkit different phases of natural language processing. - To define, train and test natural language text classifiers, by using the pretrained BERT language models.

8. Contents

8.1 Lectures	Hours	Teaching methods	Notes
Descriptive tools: extended Backus-Naur form.	2	- The main ideas with multimedia techniques - Details and examples at the blackboard, in interaction with the students - There are consultation hours - Students are invited to collaborate in research projects	
Regular grammars and finite automata: finite automata, state diagrams and regular expressions.	2		
Context-free grammars and pushdown auromata: examples.	2		
Lexical analysis: modules and interfaces (decomposition of the grammar, lexical analyzer interface), construction of the lexical analyzer (state diagrams, reserved words method).	2		
LL parsers: the LL(1) parsing algorithm for extended BNF grammars.	2		
LL parsers: computation of FIRST and FOLLOW sets.	2		
LL parsers: examples of recursive-descent applications.	2		
Theoretical results concerning the LL(*k*) and LR(*k*) grammars.	2		
LR parsers: LR(0) states, SLR(1) grammars.	2		
LR parsers: LALR(1) grammars.	2		
LR parsers: the LALR(1) algorithm.	2		
LR parsers: shift-reduce transitions, chain production elimination.	2		
LR parsers: LR table compression.	2		
Basic concepts of attribute grammars.	2		

Bibliography

1. W.M. Waite and G. Goos, Compiler Construction, Springer-Verlag, 1984.
2. I.A. Leţia and E.Şt. Chifu, Limbaje formale şi translatoare, Ed. Casa cărţii de ştiinţă, 1998.
3. A.V. Aho, R. Sethi, and J.D. Ullman, Compilers: Principles, Techniques and Tools, Addison-Wesley, 1986.

8.2 Applications – Seminars/Laboratory/Project	Hours	Teaching methods	Notes
Laboratory			
Building recursive-descent parsers from extended BNF grammars.	2	Brief presentation at the blackboard (the teacher), implementing and testing examples and exercises on the computer (the students)	
Recursive-descent (RD) applications: building abstract syntax trees (AST) for regular expressions.	2		
DR applications: code generator for an imperative language, using AST as intermediate code. Lowering the arithmetic expressions.	2		
Code generator for an imperative language, using AST as intermediate code. Lowering the loops and the conditional statements.	2		
Definite clause grammars (DCGs) for parsing natural language.	2		
DCG: building parse trees and checking agreement.	2		
DCG: dealing with natural language ambiguity. Checking agreement in the Romanian language.	2		
DCG: machine translation.	2		
The NLTK toolkit: semantic analysis of natural language with Lambda calculus.	2		
NLTK: subcategorization frames.	2		
NLTK: using the FrameNet lexical resourse, semantic role labeling (SRL).	2		

NLTK: discourse representation structures (DRS), anaphora resolution.	2		
NLTK: Dependency grammars and dependency parsers.	2		
NLTK: the phases of a natural language processing pipeline: lemmatizing, part of speech tagging, named entity recognition, using the WordNet lexical thesaurus.	2		
Project			
Numerical encoding of natural language text: bag of words (BoW), TF-IDF, and bag of n-grams.	2	Brief presentation at the blackboard (the teacher), implementing and testing examples and exercises on the computer (the students)	
Sentiment analyzer (classifier) using Logistic Regression (LR).	2		
Document categorization by using Logistic Regression: training and testing.	2		
Text encoding by using Word to Vec (word2vec): document categorization.	2		
Using the pretrained BERT language model: sentinent analyzer using Logistic Regression.	2		
Using BERT: fine-tuning the pre-trained BERT vectors.	2		
Using BERT: transfer learning for different downstream tasks: summarization, machine translation, semantic role labelling (SRL).	2		
Bibliography 1. https://www.cs.utexas.edu/users/novak/lexpaper.htm 2. Online lab manual 3. Hugging Face https://huggingface.co/			

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

It is a specialty course in Computer Science, its syllabus being both classical and modern. It teaches the students with the principles of efficient design and implementation of interpreters and translators for artificial languages. The syllabus of the course has been discussed with other important universities and companies from Romania, Europe, and USA. This syllabus has been evaluated by Romanian governmental agencies (CNEAA and ARAIS).

10. Evaluation

Activity type	Assessment criteria	Assessment methods	Weight in the final grade
Lectures	- Problem-solving skills - Attendance, Activity	- Gradual evaluation during the lectures, based on a dialog with the students and their activity at the blackboard during the lectures - There are consultation hours before the exam, during which bonuses for the final exam are granted - The final exam is a written exam	44%
Laboratory	- Problem-solving skills	Lab works:	35%

Project	- Attendance, Activity	- Gradual evaluation of the activity of students, at each lab meeting - Bonuses for the final exam are granted Project lab meetings: - Gradual evaluation of the activity of students, at each project lab meeting	21%

Minimum standard of performance:

Modelling typical engineering problems using the domain specific formal apparatus. Grade calculus: 35% lab + 21% project + 44% final exam

Conditions for participating in the final exam: Lab ≥ 5

Conditions for promotion: grade ≥ 5

Date of filling in:	**Teachers**	**Title First name Last name**	**Signature**
	Course	Assoc. prof. dr. eng. Emil Ştefan Chifu	
	Applications	Assoc. prof. dr. eng. Emil Ştefan Chifu	

Date of approval in the department	Head of department, Prof.dr.eng. Rodica Potolea
Date of approval in the Faculty Council	Dean, Prof.dr.eng. Mihaela Dînșoreanu