Subject:	Automata, Language Theories and Compilers	**Code**	72.39
Credits:	6		
Department	**Digital Systems and Data**	**Version**	2021

Course: Exchange, Computer Science Engineering

Curriculum: IN23, S10 A - Rev18, S10-Rev23, S10 - Rev18

Objectives:

No.	Description
1	1. Implement the passage between formalisms that denote a language 2. Designing automata and grammars for a language, 3. Analyze, evaluate and implement a parser and compiler for a language, 4. Design grammars with attributes.

Contents:

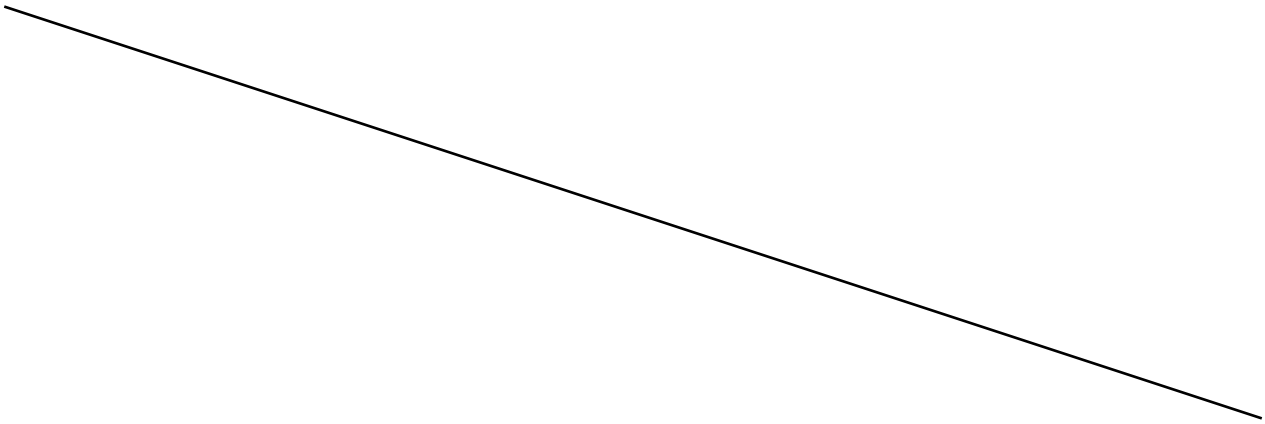
Formal languages. Chomsky's classification of formal languages. Deterministic and non-deterministic finite automaton. Equivalence between both. Sequential machines. Languages and regular expressions. Equivalence between regular expressions and finite automata. Pumping Lemma and applications. Context-free grammars and languages. Context-sensitive grammars and languages. Turing machine: its language and programming. Basic concept of a compiler. Phases in the construction of a compiler. Lexical analysis. Syntactic analysis. Table of symbols. Grammars with attributes. Generation of intermediate code. Use of Lex & Yacc-type practical tools.

Required bibliography:

No.	Description
1	No bibliography has been uploaded.
2	No bibliography has been uploaded.
3	No bibliography has been uploaded.

Optional bibliography:

No.	Description
1	No additional bibliography has been uploaded.

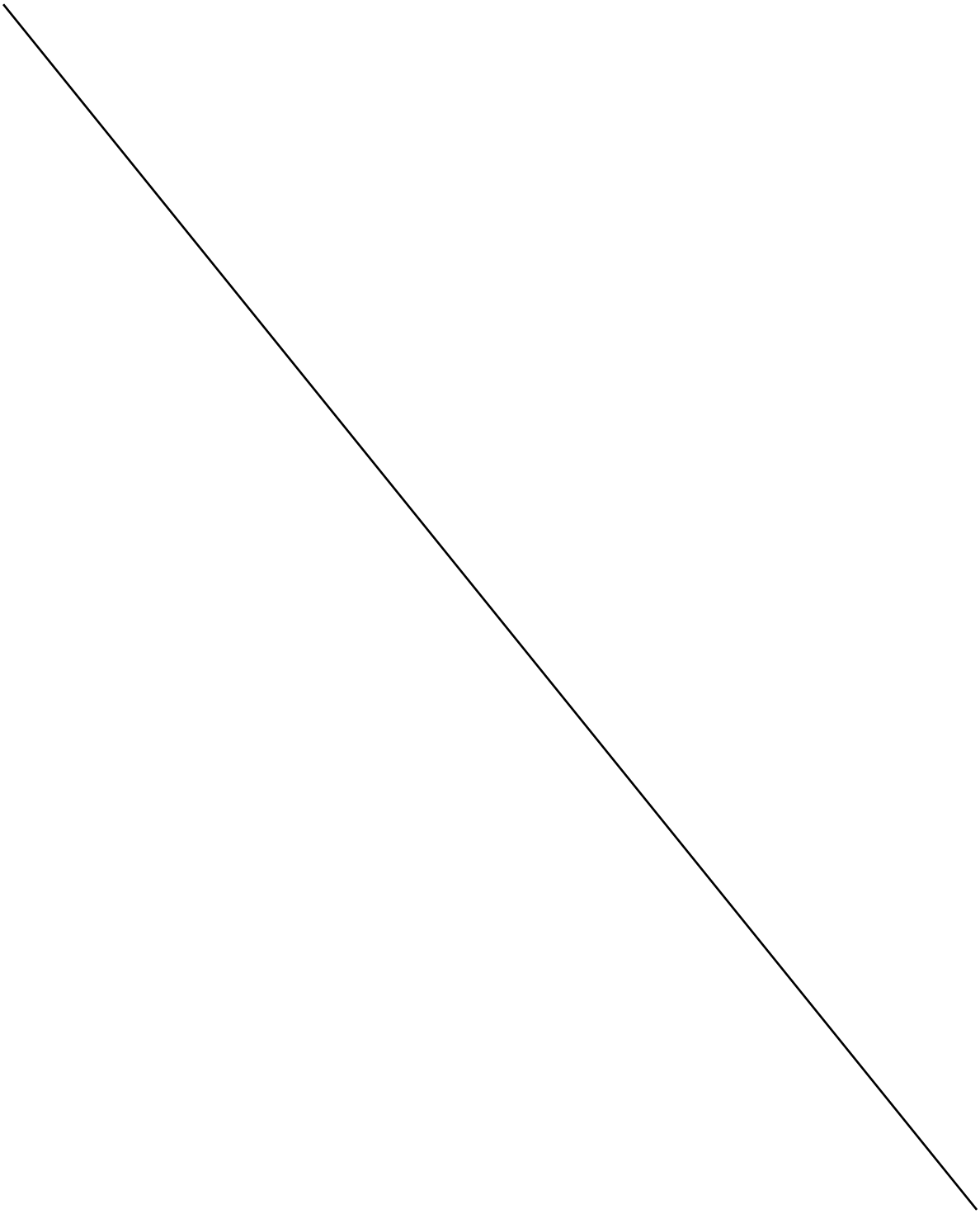


Subject:	Automata, Language Theories and Compilers	**Code**	72.39
Credits:	6		
Department	Digital Systems and Data	**Version**	2021

Course transcript:

No.	Description
1	**I. Introduction** Relation. Relations properties. Reflexivity, Symmetry, Transitivity. Equivalence relations. Composition relation. Identity relation. Power relation. Transitive closure. Reflexive transitive closure. Representation of relations. Union matrix. Intersection matrix. Composition matrix. Wharshall's algorithm. Structural induction.
2	**II. Preliminaries.** Definition of alphabet. Definition of string. String concatenation. Associativity. String Operations: Appending. Head. Body. Reversing. Abuse of notation.
3	**III. Languages and Grammars.** Language. Grammar. Language generated by a grammar. Chomsky classification. Type-0 grammars. Type-1 grammars. Type-2 grammars. Type-3 grammars. Inclusion of languages generated from the grammars. Type i strictly typed languages. Operations on languages. Languages product or concatenation. language exponentiation. Closure of a Language. Kleene closure of a language. Relationship between A* and A+.
4	**IV. Finite automata.** Transition diagrams. Instantaneous configuration. Relation between configurations. Language accepted by an automaton. Deterministic finite automaton. Determinism of a deterministic finite automaton. Non-deterministic finite automaton. Non-deterministic finite automaton with lambda transitions.
5	**Equivalences between Finite Automata** Conversion from a DFA to an NFA Conversion from an NFA to an NFA-lambda. Conversion from an NFA-lambda to a DFA. Closure of a set of states. Algorithm. Transition between sets of states. Algorithm. Minimization of an DFA. Elimination of inaccessible states. Indistinguishable states Properties of indistinguishability. Indistinguishability of order k. Properties of order k indistinguishability. Algorithm for construction of the quotient set. Minimal character of the DFA of minimal states.
6	**VI. Regular expressions. VII. Equivalences between regular language representations.** Passage from finite automaton to regular expression.

Passage from regular expression to finite automaton.
Passage from regular grammar to NDAF-lambda
Conversion from a DFA to a Regular Grammar. Algorithm.
Passage from regular grammar to NDAF Algorithm.
Pumping Lemma



Subject:	Automata, Language Theories and Compilers	**Code**	72.39
Credits:	6		
Department	Digital Systems and Data	**Version**	2021

No.	Description
7	**VIII. Properties of regular languages.** Union. Intersection. Complement. Union and intersection over more than two languages. Decidable problems in regular languages.
8	**IX. Pushdown Automaton** Instant configuration, initial configuration and configuration change. Language recognized by a pushdown automaton. Passing from an end-state PA to an empty-stack PA, and vice versa. Passing from a GLC to a PA accepting by empty stack Deterministic Pushdown Automaton.
9	**X. Turing machine.** Tape-bounded Turing Machine Turing machine for type 1 languages.
10	**XI. Parsing methods** Top-down parsing methods. Introduction. Top-down parser with backtracking. First set. Follow set. Guide symbols. LL(1) grammars. Predictive top-down parser. Predictive top-down parser with table. Augmented grammar. Non-recursiveness of LL(1) grammars. Elimination of left recursion. Factorization. LL(k) grammars. Non-decidable and decidable problems with LL grammars. Bottom-up parsing methods. Introduction. Sentential form. Right sentential form. Left sentential form. Pivot. General algorithm of shift-reduce parsers. Precedence Relations. Precedence grammar. Simple precedence grammar. Simple precedence parsing algorithm. LR grammars. Variable prefix. Item. Complete item. Valid item. Non-deterministic lambda finite automaton for the feasible prefixes of a language. SLR parser. Construction algorithm for the DFA of an LR parser. Closure function. goto function. Construction of the action and shift matrix for an LR(0) parser. Algorithm for an LR(0) parser. Shift-reduction and reduction-reduction conflicts. SLR parser. Construction of the action and shift matrix for an SLR parser. LR(1) parser. Item LR(1). Valid item LR(1). Closure function LR(1). goto function LR(1). LALR(1) parser. Core of a set of LR(1) items. Construction of the DFA for an LALR parser. Persistence of conflicts in an LALR parser.

Subject:	Automata, Language Theories and Compilers	**Code**	72.39
Credits:	6		
Department	Digital Systems and Data	**Version**	2021

No.	Description
11	**XII. Attribute Grammar.** Attribute. Valuation rule. Synthesized attribute. Inherited attribute. Syntax-directed definition (SDD) Attribute Grammar. L-attributed grammar. S-attributed grammar. Dependency graph. Topological ordering. Translation scheme.
12	**XIII. Compilation.** Schematic of a compiler. Type checking with DDS. Generation of intermediate code with DDS.

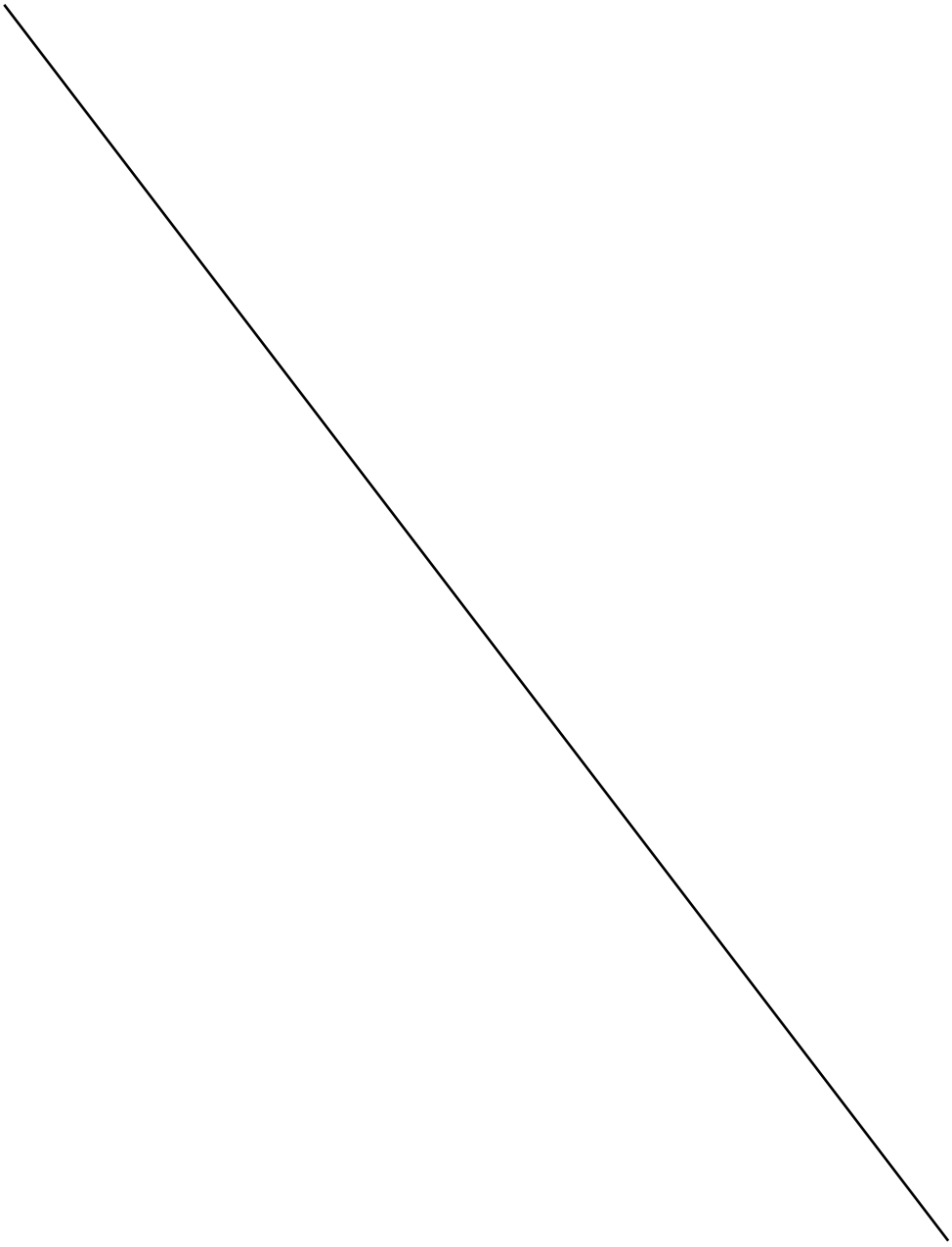
Practical assignments:

No.	Description
1	**Practical assignment 1. Introduction** Preliminaries. Proofs by Induction.
2	**Practical assignment 2. Grammars.** Chomsky hierarchy.
3	**Practical assignment 3. Regular languages 1.** Finite automata.
4	**Practical assignment 4. Regular languages 2.** Regular expressions.
5	**Practical assignment 5. Regular languages 3.** Regular grammars.
6	**Practical assignment 6. Lex.** Lexical analyzer. Instructions for the Special Assignment.
7	**Practical assignment 7. Type 2 languages.** Context Free Grammars and Pushdown Automata.
8	**Practical assignment 8. Turing machines.** Turing machines.
9	**Practical assignment 9. Yacc** Implementation of LALR bottom-up parsers. One-pass compiler. Symbols table, type handling, real-time

execution module.

10 **Practical assignment 10. Top-Down Parsing**

Top-Down parsers. Pseudocode implementations.



Subject:	Automata, Language Theories and Compilers	**Code**	72.39
Credits:	6		
Department	Digital Systems and Data	**Version**	2021

No.	Description
11	**Practical assignment 11. Bottom-up Parsing** LR, SLR, CLR, CLR, LALR bottom-up parsers.
12	**Practical assignment 12. Attribute Grammar** Attribute Grammar. Inherited and synthesized attributes. Dependency Trees.

Laboratory assignments:

No.	Description
1	**Special Practical Assignment: Language and Compiler** Creation of a new language grammar and development of the compiler using Lex and Yacc.

Professor in charge:	Santos, Juan Miguel
Head of Department:	Bolo, Mario Enrique