Detailed Course Description - Course Plan Development and Updating Procedures/ Department of Computer Science	QF01/0408-3.0E

Faculty	Science & I.T.	Department	Computer Science
Course number	0112214	Course title	Computational Theory
Number of credit hours	3	Pre-requisite/co-requisite	Discrete Mathematics

Brief course description

This course introduces the concepts of computation theory through the study of formal languages and automata. The topics covered include language generators such as grammars and regular expressions and language recognizers such as the different types of automata. It also introduces some basic compiler design principles, and it provides insights into algorithm analysis.

	Course goals and learning outcomes
Goal 1	Ability to use the principles of computer science in understanding, implantation and analysis of mathematical problems and finding their solutions.
Learning outcomes	1.1 Student should understand and analyze mathematical problems. 1.2 Student should be able to use mathematical concepts in algorithm analysis.
Goal 2	Ability to relate formal languages to automata theory
Learning outcomes	2.1 Student should be able to classify languages, grammars, and automata according to the Chomsky Hierarchy. 2.2 Student should relate each formal language to its corresponding grammar and automaton. 2.3 Student should be able transform language generators to language recognizers and vice versa.
Goal 3	Ability to use formal languages and automata theory in compiler design
Learning outcomes	3.1 Student should be able to design finite automata for recognizing strings. 3.2 Student should be able to use regular expressions as language generators 3.3 Student should be able to remove useless productions from context-free grammars. 3.4 Student should be able to normalize context-free grammars. 3.5 Student should be able to design a simple parser.
Goal 4	Ability to relate computation and automata theories to algorithm analysis
Learning outcomes	4.1 Student should understand the Church-Turing Thesis. 4.2 Student should relate different types of Turing Machines to algorithm complexity classes. 4.3 Student should understand the concept of decidability with the Halting Problem as an example.
Textbook	Michael Sipser, *Introduction to the Theory of Computation*, 3rd Edition, 2014.
Supplementary references	1. T.P. Shekhar (Author), K. Srinivas and B. Kavitha Rani, *Formal languages & Automata Theory: A learner's handbook*, 2016. 2. Peter Linz, *An Introduction to Formal Languages and Automata*, 6th Ed., 2016. 3. Gordon J. Pace, *Mathematics of Discrete Structures for Computer Science*, 2012.

Course timeline				
Week	**Number of hours**	**Course topics**	**Pages (textbook)**	**Notes**
01	**1** **1** **1**	- Models of Computation (Automaton, Different Types of Automata) - Mathematical Preliminaries and Notation (Sets, Functions, Relations, Graphs and Trees)	1–10	
02	**1** **1** **1**	- Basic Concepts of Automata Theory (Alphabet, String, String Operations, Languages, Operations on Languages)	10–16	
03	**1** **1** **1**	- Introduction to Finite Automaton - Definition of Deterministic Finite Automaton (DFA) - How a DFA processes strings - Simpler Notation for DFA: Transition diagram, Transition Table - The language of a DFA	31–46	
04	**1** **1** **1**	- The Nondeterministic Finite Automaton (NFA): Definition of NFA, Processing the string by NFA, Extending the transition function and language of an NFA - Finite Automaton with λ-Transitions (Note: $\lambda \cong \varepsilon$). - Equivalence of DFA and NFA	47–62	
05	**1** **1** **1**	- Regular Expressions (REs) and RE Operators - Properties of REs - Identities and Annihilators - Some applications of REs - Converting between Finite Automata and REs	63–76	
06	**1** **1** **1**	Review of Previous Chapters **First Exam (20%)**		
07	**1** **1** **1**	- Properties of Regular Languages (RLs): Union, Concatenation, Closure, Reversal, Complement and Intersection - Relationship between RE and RL - Pumping Lemma	63–81	
08	**1** **1** **1**	- Regular Grammars - The equivalence between regular grammar and RL - Context-Free Grammar (CFG) and Context-free Language (CFL) - Derivations Using a Grammar	101–106	
09	**1** **1** **1**	- The language of a CFG - The relationship between Automata and Grammars - Parse Tree (Construction and Yield) - The relationship between RL and CFL	101–106	
10	**1** **1** **1**	- Removing Ambiguity from Grammars and Languages - Simplification of CFG - Methods for transforming grammars	107–153	

11	**1** **1** **1**	- Removing Useless Productions: Nullable Variables and Unit Productions - λ -Free Language - Chomsky Normal Form - Greibach Normal Form	107–124	
12	**1** **1** **1**	Review of Previous Chapters **Second Exam (20%)**		
13	**1** **1** **1**	- The Pumping Lemma for CFG - CFGs relationship with Programming Languages - Push Down Automaton (PDA) - The Language Accepted by a PDA	125–153	
14	**1** **1** **1**	- Conversion between CFG and PDA - Deterministic PDAs - Parsing - Properties of CFL - Intersection of CFL and RL	125–153	
15	**1** **1** **1**	- The Chomsky Hierarchy - Turing Machines (TMs): Formal Definition, Acceptance of Languages - Computing Functions with TMs - The Church-Turing Thesis - Using TMs in defining complexity classes	165–210, 273–303	
16	**1** **1**	**Final Exam**		

Theoretical course evaluation methods and weight	Participation = 10% First exam 20% Second exam 20% Final exam 50%	**Practical (clinical) course evaluation methods**	Semester students' work = 50% (Reports, research, quizzes, etc.) Final exam = 50%

Approved by head of department		**Date of approval**	

Extra information (to be updated every semester by corresponding faculty member)

Name of teacher		Office Number	
Phone number (extension)		Email	
Office hours			