`

National Computer Education Accreditation Council NCEAC

NCEAC.FORM.001-C

COURSE DESCRIPTION FORM: CS301- Theory of Automata

INSTITUTION National University of Computer & Emerging Sciences, Islamabad

BS-CS –**-Fall 2024**

PROGRAMS TO BE EVALUATED

Course Description

(Fill out the following table for each course in your computer science curriculum. A filled-out form should not be more than 2-3 pages.)

Course Code	CS-3005
Course Title	Theory of Automata
Credit Hours	3
Prerequisites by Course(s) and Topics	Discrete Structures (CS1005)
Grading Policy	Absolute grading
Policy about missed assessment items in the course	Retake of missed assessment items (other than midterm/ final exam) will not be held. For a missed midterm/ final exam, an exam retake/ pretake application along with necessary evidence are required to be submitted to the department secretary. The examination assessment and retake committee decides the exam retake/ pretake cases.
Course Plagiarism Policy	Plagiarism in the project or midterm/ final exam may result in F grade in the course. Plagiarism in an assignment will result in zero marks in the whole assignments category.
Assessment Instruments with Weights (homework, quizzes, midterms, final, programming assignments, lab work, etc.)	100% theory Breakdown of Course Work (Total): Sessional 2 30% Quizzes 5 10% Assignments 5 10% Final 1 50%
Course Coordinator	Labiba Fahad
URL (if any)	
Current Catalog Description	The course introduces some fundamental concepts in automata theory and formal languages including grammar, finite automaton, regular expression, formal language, pushdown automaton, and Turing machine. Not only do they form basic models of

	computation, they are also the foundation of many branches of computer science, e.g. compilers, software engineering, concurrent systems, etc. The properties of these models will be studied and various rigorous techniques for analyzing and comparing them will be discussed, by using both formalism and examples.
Textbook (or **Laboratory Manual** for Laboratory Courses)	**Introduction to Computer Theory**, by Deniel I. A. Cohen John Wiley & Sons, Inc., 2nd Edition.
Reference Material	**Introduction to Automata Theory,** Languages and Computation by Jeffrey Ullman and John Hopcroft, 3rd Edition, Pearson, 2008, ISBN: 978-81-317-2047-9 .
Course Learning Outcomes	The course objective is to prepare students to understand the fundamentals of Computer Science and relate it to the computing as it is practiced.

At the completion of the course, the students shall demonstrate the understanding of the following concepts:

1. The mathematical models that will describe with varying degree of accuracy, parts of computers, types of computing devices and similar machines.
2. The theory of formal languages and the use of various abstract machines which model the computing devices in the most general sense by characterizing their capabilities as recognizers.
3. These recognizers will be used to identify/validate the synthetic characteristics of programming languages.
4. Introduce parsing as the basis of compiler constructions.
5. The ability to apply concepts in proof theory to other aspects of computer science.

B. Program Learning Outcomes

For each attribute below, indicate whether this attribute is covered in this course or not. Leave the cell blank if the enablement is little or non-existent.

1. Computing Knowledge	Apply knowledge of mathematics, natural sciences, computing fundamentals, and a computing specialization to the solution of complex computing problems.	✔
2. Problem Analysis	Identify, formulate, research literature, and analyze complex computing problems, reaching substantiated conclusions using first principles of mathematics, natural sciences, and computing sciences.	✔
3. Design/ Develop Solutions	Design solutions for complex computing problems and design systems, components, and processes that meet specified needs with appropriate consideration for public health and safety, cultural, societal, and environmental considerations.	✔

4. Investigation & Experimentation	Conduct investigation of complex computing problems using research based knowledge and research based methods.	
5. Modern Tool Usage	Create, select, and apply appropriate techniques, resources and modern computing tools, including prediction and modelling for complex computing problems.	✔
6. Society Responsibility	Apply reasoning informed by contextual knowledge to assess societal, health, safety, legal, and cultural issues relevant to context of complex computing problems.	
7. Environment and Sustainability	Understand and evaluate sustainability and impact of professional computing work in the solution of complex computing problems.	
8. Ethics	Apply ethical principles and commit to professional ethics and responsibilities and norms of computing practice.	
9. Individual and Team Work	Function effectively as an individual, and as a member or leader in diverse teams and in multi-disciplinary settings.	
10. Communication	Communicate effectively on complex computing activities with the computing community and with society at large.	
11. Project Management and Finance	Demonstrate knowledge and understanding of management principles and economic decision making and apply these to one's own work as a member or a team.	
12. Life Long Learning	Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological changes.	

C. Mapping of CLOs on PLOs
(CLO: Course Learning Outcome, PLOs: Program Learning Outcom

		PLOs											
		1	2	3	4	5	6	7	8	9	10	11	12
	1	✔	✔	✔									

<table>
<tr><td></td><td>
<table>
<tr><td rowspan="3">CLOs</td><td>2</td><td>✔</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>3</td><td>✔</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>4</td><td>✔</td><td>✔</td><td>✔</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>5</td><td>✔</td><td>✔</td><td>✔</td><td></td><td>✔</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td></tr>
<tr><td>Topics covered in the course with number of lectures on each topic (assume 15 weeks of instruction and 1.5 hour lecture duration)</td><td>
<table>
<tr><td>• Regular expressions/Regular languages</td><td>5 Lectures</td></tr>
<tr><td>• Finite automata (FAs)</td><td>3 Lecture</td></tr>
<tr><td>• Transition graphs (TGs) , NFAs, Kleene’s theorem</td><td>3 Lectures</td></tr>
<tr><td>• Transducers (automata with output), Pumping lemma and non regular language</td><td>4 Lectures</td></tr>
<tr><td>• Context free grammars</td><td>3 Lectures</td></tr>
<tr><td>• Derivations, derivation trees and ambiguity</td><td>3 Lectures</td></tr>
<tr><td>• Normal form grammars</td><td>4 Lectures</td></tr>
<tr><td>• Push-down Automata</td><td>3 Lectures</td></tr>
<tr><td>• Non-context free languages, Decidability, Chomsky’s hierarchy of grammars</td><td>4 Lectures</td></tr>
<tr><td>• Turing Machines</td><td>3 Lectures</td></tr>
<tr><td>• Post Machine, Variations on TM</td><td>3 Lectures</td></tr>
<tr><td>• 2PDA</td><td>2 Lectures</td></tr>
<tr><td>• Computable Functions</td><td>2 Lectures</td></tr>
<tr><td>• Universal Computing Machines</td><td>2 Lectures</td></tr>
</table>
</td></tr>
<tr><td>Laboratory Projects/ Experiments done in the Course</td><td>None</td></tr>
<tr><td>Programming Assignments Done in the Course</td><td></td></tr>
<tr><td>Class Time Spent on (in percentage)</td><td>
<table>
<tr><th>Theory (%)</th><th>Problem Analysis (%)</th><th>Solution Design (%)</th><th>Social and Ethical Issues (%)</th></tr>
<tr><td>75%</td><td>10%</td><td>10%</td><td>5%</td></tr>
</table>
</td></tr>
<tr><td>Oral and Written Communications</td><td>Every student is required to engage in class participation, group discussions, presentation and demonstration.</td></tr>
</table>

A. Tentative course outline and lecture plan

Number of Lectures	Topics	Chapter
5	**Regular expressions/Regular languages**	**3,4**

	• Introduction • Descriptive Definition • Recursive Definition • Regular Expressions(RE)	
3	**Finite automata (FAs)**	**5**
3	**Transition graphs (TGs) , NFAs, kleene's theorem**	**6, 7**
4	**Transducers (automata with output), Pumping lemma and non regular language**	**9**
3	**Context Free Grammars**	**13**
7	**Derivations, derivation trees and ambiguity, Operations preserving CFLs, Normal Forms**	**14, 15, 16**
3	**Push-down Automata**	**18**
5	**Non-context free languages, Decidability, Chomsky's hierarchy of grammars**	**19, 20, 21, 30**
4	**Turing Machines**	**24**
3	**Post Machine, Variations on TM**	**25**
6	**2PDA, Computable functions,**	**26, 27, 28, 31**