Course Code CSE2004	Theory of Computation and Compiler Design	Course Type LT	Credits 4

Course Objectives:

- To introduce Formal Languages, Automata Theory and Abstract models of Computation and Computability.
- To gain knowledge in computational theory.
- To realize the theoretical concepts and techniques involved in the software system development
- To provide an insight into compiler design concepts.

Course Outcomes:

Students will be able to

- Apply the theoretical concepts and techniques in designing the software systems.
- Identify, analyze, design and formulate problems using computational theory.
- Develop scanning and parsing techniques for languages.
- Relate built-in tools in the construction of compiler.

Student Outcomes (SO): a, b, e, l

a. An ability to apply the knowledge of mathematics, science and computing appropriate to the discipline

b. An ability to analyze a problem, identify and define the computing requirements appropriate to its solution.

e. An ability to identify, formulate and solve engineering problems.

l. An ability to apply mathematical foundations, algorithmic principles and computer science theory in the modeling and design of computer-based systems.

Unit No	Unit Content	No. of hours	SOs
1	Basic concepts – Theorem proving – Finite automata: NFA, DFA, € - NFA, Regular expressions - Equivalence between FA and RE – Minimization – Decision properties – Pumping lemma for Regular Languages. Problems: Design of FA – Inter-conversion between RE and FA – Proving languages to be not regular	9+3	a, b, e, l
2	Specification of tokens – FA and RE to represent token formats – LEX. Context Free Grammar – Derivations – Parse trees – Ambiguity – Pushdown Automata – DPDA & NPDA – Decision properties – Pumping lemma for CFL. Problems: Design of CFG – Design of PDA – Inter-conversion between PDA & CFG – Proving languages to be not context-free	9+3	a, b, e, l
3	Chomsky Normal Form – Griebach Normal Form – Parsing – Top-down Parsing – Predictive Parsing - Bottom up parsing – SLR, CLR and LALR Parsing – YACC. Problems: Conversion from CFG to CNF, GNF – Parsing	9+3	a, b, e, l

4	Turing machines – TM as a computation model – TM as a recognizer – TM with multiple tapes – Other models of TM – Linear Bounded Automata. Three Address Codes – Code optimization techniques. Problems: Design of TM – Design of LBA – Conversion from parse tree to TAC – optimization techniques	**9+3**	**a, b, e, l**
5	Chomsky Hierarchy of languages – Undecidability – Recursive and non – recursive languages – Examples - Code generation. Problems: Identification of Undecidability – Code generation	**8+2**	**a, b, e, l**
6	Guest Lecture on Contemporary Topics	**2**	
	Total Hrs.:	**60**	

Mode of Teaching and Learning: Flipped Class Room, Activity Based Teaching/Learning, Digital/Computer based models, wherever possible to augment lecture for practice/tutorial and minimum 2 hours lectures by industry experts on contemporary topics

Mode of Evaluation and assessment:

The assessment and evaluation components may consist of unannounced open book examinations, quizzes, student's portfolio generation and assessment, and any other innovative assessment practices followed by faculty, in addition to the Continuous Assessment Tests and Term End Examinations.

Text Books:

1. John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman, "Introduction to Automata Theory, Languages and Computation", 3rd Edition, Pearson Education, 2014.
2. Alfred V. Aho, Monica S Lam, Ravi Sethi, Jeffery D Ullman, " Compilers: Principles, Techniques, and Tools", 2nd Edition, Pearson Education, 2015.

Reference Books:

1. Michael Sipser, "Introduction to the Theory of Computation", 2nd Edition, Wadsworth Publishing Co Inc, 3rd Edition, 2012.

Recommendation by the Board of Studies on	
Approval by Academic council on	
Compiled by	