CSE11111	Formal Language and Automata Theory	L	T	P	C
Version 1.0	**Contact Hours – 45**	**3**	**0**	**0**	**3**
Pre-requisite/Exposure	**NIL**				
Co-requisite	**NIL**				

Course Objectives:

1. Introduce concepts in automata theory and theory of computation
2. Identify different formal language classes and their relationships
3. Design grammars and recognizers for different formal languages
4. Prove or disprove theorems in automata theory using its properties
5. Determine the decidability and intract
6. ability of computational problems

Course Outcomes:

On the completion of this course the student will be able to

CO1: Define the basic concepts in formal language theory, grammars, automata theory, computabilitytheory, and complexity theory.
CO2: Demonstrate abstract models of computing, including deterministic (DFA), non-deterministic (NFA),Push Down Automata (PDA) and Turing (TM) machine models and their power to recognize the languages
CO3: Prove and disprove theorems establishing key properties of formal languages and automata.
CO4: Acquire a fundamental understanding of core concepts relating to the theory of computation and computational models including (but not limited to) decidability and Intractability.
CO5: Solve fundamental problems related to Computational Model.

Course Description:

This course will provide a foundation to the “Theory of Computation”. The student will realize that the sometimes chaotic technology oriented world of computers has a very elegant mathematical basis to it. This basis is deeply rooted in mathematics developed before the days of modern computers. Our study will lead to some interesting implications concerning the theoretical limits of computing. On the practical side, this course is a background for a course on compilers. Topics covered in this course include: mathematical prerequisites, finite state machines (automata), concept of a language and grammars, deterministic and non-deterministic accepters, regularexpressions and languages, context-free languages, normal/canonical forms, pushdown automata, Turing machines, context sensitive languages, recursive and recursively enumerable languages. Each of the language classes has twopoints of view: a class of automata defining the language, and a class of grammars defining the language. Thisdual approach to defining languages, will finally lead to the Chomsky hierarchy of languages. We shall observe that the Turing Machine not only serves to define a language class, but also a mathematical model for computation itself and defines the theoretical limits of computation.

Course Content:

Unit-I	**4 Lecture Hours**
Mathematical Preliminaries: Set Theory, Describing a Set, Empty Set, Identity and Cardinality, Subset, Power Sets, Operations on Sets: Union, Intersection, Set Theoretic Equalities, Sequence versus Set, Ordered Pairs, Cartesian Product,Relations, Binary Relation, Domain and Range of Relation, Operations on Relations, Properties of Relations, Functions, Types of Functions, Alphabet, String and Language, Operations on Language, Grammars, Types of Grammars Chomsky Hierarchy, Graphs and Trees, Directed Graph, Undirected Graph, Trees, Lemma, Theorem Proving, Proof by Induction Proof by Contradiction, Proof by Example.	
Unit-II	**16 Lecture Hours**

Finite Automata:
Finite-state Machine, Finite-Automaton Model, Properties of Transition Function ‘c’, Transition Diagram, Transition Table, Language Acceptance, Two Types of Finite Automata, Deterministic Finite Automata (DFA) Non-deterministic Finite Automaton, Acceptance of NFA, Equivalence of DFAs and NFAs, Converting NFA to DFA, Subset Construction, NFA with Epsilon- -closure), Eliminating e-Transitions, Converting NFA with ε-Transition to NFA, - -Transition ~~to~~ to DFA, Comparison Method for Testing, Equivalence of Two FA, Reduction of Number of States in FA, Indistinguishable States, Equivalent Classes, Minimization of DFA, Minimization of DFA Using Myhill Nerode Theorem, Finite Automata with Output, Moore Machine, Mealy Machine, Equivalence Between Moore and Mealy Machines, Interconversions Between Machines, Applications of Finite Automata with Output, The Full-adder, The String Sequence Detector.
Regular Languages and Regular Grammar:
Regular language, Regular expressions, Deterministic finite automata (DFA) and equivalence with regular expressions, NFA and equivalence with DFA, Regular grammars and equivalence with finite automata, Properties of regular languages, Pumping lemma for regular languages, Problem solving using pumping lemma.
(ε) Transitions, Epsilon Closure (ε
without ε Transition, Converting NFA with ε

Unit-III	**15 Lecture Hours**

Pushdown Automata & Context Free Languages:
Graphical Representation of PDA, Instantaneous Description of PDA, Language Acceptance by PDA, Equivalence of Acceptance of Final State and Empty Stack, Types of PDAs, Deterministic PDA, Closure Properties of DCFL, Decision Properties of DCFLs, DPDA and Regular Languages, DPDA and Ambiguous Grammar, Equivalence of PDA’s and CFG’s, Nondeterministic pushdown automata (NPDA), NPDA and equivalence with CFG, Constructing PDA for Given CFG, Constructing CFG for the Given PDA, Two-stack PDA, Applications of PDA, PDA as a Parser, Top-down Parser Using the PDA, Pumping lemma for context-free languages.
Context Free Grammar:
Context-free grammars (CFG), Leftmost and Rightmost Derivations, Derivation Tree, Equivalence of Parse Trees and Derivations, Ambiguous Grammar, Removing Ambiguity, Inherent Ambiguity, Simplification of Grammars, Elimination of Useless Symbols, Elimination of e-Productions, Eliminating Unit Productions, Chomsky normal forms, Greibach normal forms

Unit-IV	**10 Lecture Hours**

Context Sensitive Grammar and Languages:
Context-sensitive grammars (CSG), Context-sensitive Languages, Linear bounded automata, Linear bounded automata and equivalence with CSG, Properties of Context-sensitive grammars (CSG) and Languages, Properties of Linear bounded automata.
Turing Machine :
Turing Assumptions, Instantaneous Description, Turing Machine as Language Accepter, Turing Machine as a Computational Machine, Techniques for Turing Machine Construction, Storage in Finite Control, Multi-track Tape, Checking off Symbols, Subroutines, Shifting Over, Types of Turing Machines, Non-deterministic Turing Machines, Turing Machines with Two-dimensional, Tapes, Turing Machines with Multiple Tapes, Turing Machines with Multiple Heads, Turing Machines with Infinite Tape, Church’s Thesis, Turing Machines as Enumerators, Universal Turing Machine, Counter Machine, Recursive and Recursively Enumerable Languages
Unrestricted grammars and equivalence with Turing machines, Church-Turing thesis, universal Turing machine, Rice's theorem, undecidable problems about languages.

Unit-V	**10 Lecture Hours**

Decidability:
Decidable Languages, Decidable problems concerning regular languages, Decidable problems concerning context-free languages, Undecidability, The diagonalization method, An undecidable language
A Turing-unrecognizable language
Reducibility:
Undecidable Problems from Language Theory, Reductions via computation histories, Simple Undecidable Problem, Mapping Reducibility , Computable functions, Formal definition of mapping reducibility
Time Complexity:
Measuring Complexity, Big-O and small-o notation, Analyzing algorithms, Complexity relationships among models, The Class P, Polynomial time, Examples of problems in P, The Class NP, Examples of problems in NP,
P versus NP, NP-completeness, Polynomial time reducibility, Definition of NP-completeness, The Cook Levin Theorem
Space Complexity:
Savitch’s Theorem, The Class PSPACE, PSPACE-completeness

Text Books:

1. Introduction to Automata Theory, Languages, and Computation, 3rd Edition, John E. Hopcroft Rajeev Motwani and Jeffrey D. Ullman, Pearson Education.
2. Michael Sipser, Introduction to the Theory of Computation, PWS Publishing
3. An Introduction To Formal Languages And Automata, Peter Linz

Reference Books:

1. Harry R. Lewis and Christos H. Papadimitriou, Elements of the Theory of Computation,Pearson Education Asia.
2. Dexter C. Kozen, Automata and Computability, Undergraduate Texts in Computer Science, Springer.

Modes of Evaluation: Quiz/Assignment/Presentation/Extempore/ Written Examination

Examination Scheme:

Components	Mid Term	Class Assessment	End Term
Weightage (%)	20	30	50

Relationship between the Course Outcomes (COs) and Program Outcomes (POs)

Mapping between COs and POs		
Course Outcomes (COs)		Mapped Program Outcomes
CO1	Define the basic concepts in formal language theory, grammars, automata theory, computability theory, and complexity theory.	PO1, PO, PO3, PO4, PSO1, PSO2
CO2	Demonstrate abstract models of computing, including deterministic (DFA), non-deterministic (NFA), Push Down Automata (PDA) and Turing (TM) machine models and their power to recognize the languages	PO1, PO, PO3, PO4, PSO1, PSO2
CO3	Prove and disprove theorems establishing key properties of formal languages and automata.	PO1, PO, PO3, PO4, PSO1, PSO2
CO4	Acquire a fundamental understanding of core concepts relating to the theory of computation and computational models including (but not limited to) decidability and Intractability.	PO1, PO, PO3, PO4, PSO1, PSO2
CO5	Solve fundamental problems related to Computational Model.	PO1, PO, PO3, PO4, PSO1, PSO2

		Engineering knowledge	Problem analysis	Design/development of solutions	Conduct investigations of complex problems	Modern tool usage	The engineer and society	Environment and sustainability	Ethics	Individual and team work	Communication	Project management and finance	Life-long learning	Adequate strong skills in learning new programming environments, analyse and design algorithms for	The ability to understand the evolutionary changes in computing, apply standard practices and strategies in	Ability to analyse the impact of Computer Science and Engineering solutions in the societal and human
Course Code	Course Title	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2	PSO 3

CSE11111	Formal Language and Automata Theory	3	3	3	3	-	-	-	-	-	-	-	-	3	3	-