Name of the Course: Theory of Computation

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** The Theory of Computation explores the fundamental principles that define what problems can be solved using computers and how efficiently they can be solved. It lays the groundwork for understanding computation through abstract machines, formal languages, and logical reasoning, making it a cornerstone of theoretical computer science. **Relevance:** This course is highly relevant as it forms the theoretical basis for designing algorithms, compilers, and programming languages. It deepens the student's understanding of the limits of computation, which is critical for solving complex computational problems systematically and effectively. **Usefulness:** The subject helps students learn how to model computational problems using mathematical tools and abstract machines like automata and Turing machines. These skills are vital when optimizing algorithms, designing new computing systems, or developing efficient parsing tools in programming languages. **Application:** Theory of Computation finds applications in compiler design, artificial intelligence, natural language processing, software verification, and cryptography. It also plays a key role in understanding whether problems can be solved algorithmically and what resources are required for their solutions. **Interest:** Students often find this subject intellectually stimulating because it challenges their logical and mathematical thinking. It involves elegant problem-solving techniques and creative ways to model and classify computational problems, sparking curiosity about the power and limits of machines. **Connection with Other Courses:** The course connects closely with subjects like Compiler Design, Artificial Intelligence, Algorithms and Data

		Structures, and Programming Language Theory. It provides the mathematical and logical foundation required for developing more advanced computer science concepts and tools. **Demand in the Industry:** Industries focused on algorithm development, formal verification, secure systems, and artificial intelligence highly value the principles taught in this course. Companies working in automation, language processing, and software correctness increasingly seek professionals with strong theoretical backgrounds. **Job Prospects:** Career opportunities include roles such as compiler engineer, language designer, research scientist, software developer, algorithm designer, and systems programmer. It also serves as a gateway for advanced studies or research in computer science, especially in theoretical and mathematical computing domains.
2	**Vertical:**	Major
3	**Type:**	Theory
4	**Credits:**	2 credits (1 credit = 15 Hours for Theory)
5	**Hours Allotted:**	30 Hours
6	**Marks Allotted:**	50 Marks
7	**Course Objectives (CO):** **CO 1.** Introduce the fundamental concepts of theoretical computer science, including automata theory, formal languages, and computational models. **CO 2.** Understand the mathematical foundations necessary to describe and analyze computation. **CO 3.** Explore different models of computation, including finite automata, pushdown automata, linear bound automata, and Turing machines. **CO 4.** Analyze the decidability and complexity of computational problems. **CO 5.** Introduce the theory of computability and the limitations of algorithmic solutions. **CO 6.** Familiarize students with the classification of problems based on complexity classes such as P, NP, and NP-Complete.	
8	**Course Outcomes (OC):** After successful completion of this course, students would be able to - **OC 1.** Describe the fundamental concepts and significance of Theory of Computation in Computer Science. **OC 2.** Design deterministic and non-deterministic finite automata for regular languages and prove their equivalence.	

	OC 3. Apply regular expressions and grammars to define and generate formal languages. **OC 4.** Construct context-free grammars and analyse their ambiguity, simplification, and normal forms. **OC 5.** Differentiate between complexity classes such as P, NP, NP-Complete, and NP-Hard problems. **OC 6.** Identify and analyze undecidable problems.
9	**Modules** **Module 1 (15 hours):**
	Introduction to Theory of Computation: Basics of Computation, Importance of Theory of Computation in Computer Science, Mathematical Foundations (Sets, Relations, Functions, Proof Techniques) **Automata Theory:** Defining Automaton, Finite Automaton, Transitions and Its properties, Acceptability by Finite Automaton, Nondeterministic Finite State Machines, DFA and NDFA equivalence, Mealy and Moore Machines, Minimizing Automata. **Formal Languages:** Defining Grammar, Derivations, Languges generated by Grammar, Chomsky Classification of Grammar and Languages, Recursive Enumerable Sets, Operations on Languages, Languages and Automata **Regular Languages:** Regular Grammar, Regular Expressions, Finite automata and Regular Expressions, Pumping Lemma and its Applications, Closure Properties, Regular Sets and Regular Grammar Context Free Languages: Context-free Languages, Derivation Tree, Ambiguity of Grammar, CFG simplification, Normal Forms, Pumping Lemma for CFG
	Module 2 (15 hours):
	Pushdown Automata: Definitions, Acceptance by PDA, PDA and CFG **Linear Bound Automata:** The Linear Bound Automata Model, Linear Bound Automata and Languages. **Turing Machines:** Turing Machine Definition, Representations, Acceptability by Turing Machines, Designing and Description of Turing Machines, Turing Machine Construction, Variants of Turing Machine, Decidability and Undecidability, The Church-Turing thesis, Universal Turing Machine, Halting Problem, Introduction to Unsolvable Problems **Computability and Complexity:** Time Complexity and Space Complexity, Big-O Notation, Class P and Class NP, NP-Complete and NP-Hard Problems, Polynomial Reductions, Introduction to Complexity Hierarchies
10	**Text Books** 1. Theory of Computer Science, K. L. P Mishra, Chandrasekharan, PHI,3rd Edition 2. Introduction to Computer Theory, Daniel Cohen, Wiley,2nd Edition 3. Introductory Theory of Computer Science, E.V. Krishnamurthy,Affiliated East-West Press.

11	**Reference Books** 1. Theory of Computation, Kavi Mahesh, Wiley India 2. Elements of The Theory of Computation, Lewis, Papadimitriou, PHI 3. Introduction to Languages and the Theory of Computation, John E Martin, McGraw-Hill Education 4. Introduction to Theory of Computation, Michel Sipser, Thomson	
12	**Internal Continuous Assessment: 40%**	**Semester End Examination: 60%**