LESSON PLAN

	DPG Institute of Technology and Management **Sector 34, Gurugram HR 122004**
	Lesson Plan
	Course Name: Automata Theory & Compiler Design
	Faculty Name: Ms. Bhawna kumari

No. of Lecture Hours/Week	**3**	**Exam Hours**	**3**
Total No. of Lecture Hours	**36**	**Exam Marks**	**75**
Course Code	**PCC-DS-305G**		

Objectives of the course:

The objective of this course is to develop methods by which computer scientists can describe and analyze the dynamic behavior of discrete systems, in which signals are sampled periodically.

1. To provide an understanding of automata, grammars and language translators.
2. To know the various techniques used in compiler construction.
3. To be aware of the process of semantic analysis.
4. To analyze the code optimization & code generation techniques.
5. To understand the concept of storage administration for different programming environment.

Detailed Lesson Plan

Lecture No.	Topics to be ~~covered~~	Teaching Methodology/ Pedagogy	Class Activity	Remark/CO
UNIT - 1				
1	Regular Expressions, Rules for regular expressions	Interactive Lecture	Group exercise: Write regular expressions for given patterns	CO1
2	Finite Automata- DFA, NFA	Lecture + Diagrammatic Explanation	Build DFA/NFA for simple languages in groups	CO1
3	Conversion of NFA to DFA, Conversion of regular expressions to NFA	Lecture + Problem Solving Step-by-Step Tutorial	Practice conversion problems	CO1
4	Compiler and phases of compiler	Lecture + Discussion	Create a flowchart of compiler phases	CO1
UNIT - 2				
5	Chomsky hierarchy of languages, CNF & GNF	Discussion + Whiteboard Teaching	Identify language types using Chomsky hierarchy	CO2
6	Parsing and its types	Interactive Lecture, Problem-Based Learning	Parse simple expressions using parser	CO2

7	Recursive Descent Parsers	Interactive Lecture, Problem-Based Learning	Practice recursive descent parsing with examples	CO2
8	Shift Reduce Parsers	Interactive Lecture, Problem-Based Learning	Practice shift-reduce parsing with examples	CO2
9	Push Down Automata, Deterministic PDA & NPDA	Interactive Q&A + Example-Based	Design PDA for a given language	CO2
10	PDA Questions practice	Interactive Lecture, Problem-Based Learning	Design PDA for a given language	CO2
11	Turing Machine basics (model, design, diagram)	Interactive Lecture, Interactive Q&A + Example-Based	Identify Turing machine design challenge	CO2
12	Turing Machine Questions practice , halting problem	Interactive Lecture, Problem-Based Learning	Practice Turing based problems	CO2
13	Turing Machine Questions practice	Interactive Lecture, Problem-Based Learning	Practice Turing based problems	CO2
14	Syntax Directed Translation basics	Lecture + Visual Diagrams	Build syntax trees for simple expressions	CO2
15	Intermediate Code Generation	Flipped Classroom	Code a basic intermediate code for assignment statements	CO2
16	Translation of Assignments Statements and Boolean Expressions	Interactive Lecture, Problem-Based Learning	Practice topic based problems	CO2
17	Revision for sessional 1	Interactive Lecture, Problem-Based Learning	Assignment and Previous year question paper discussion	CO2
UNIT - 3				
18	Type Checking Introduction, Specifications of simple type checker	Case Study + Group Discussion	Design a simple type checker	CO3
19	Type conversions, Equivalence of Type Expressions	Interactive Lecture, Problem-Based	Practice topic based problems	CO3
20	Storage Organization, Static Storage Allocation Techniques	Interactive Lecture + Discussion	Role-play: Simulate memory allocation strategies	CO3
21	Parameter passing, Symbol Table	Interactive Lecture + Discussion, Problem-Based Learning	Build a symbol table for a sample program	CO3
22	Dynamics Storage Allocation Techniques	Interactive Lecture + Flowcharts	Students are asked to draw flowchart of Storage Allocation Techniques	CO3
UNIT - 4				
23	Code Optimization (Principal Sources and Basics)	Interactive Lecture + Discussion	Optimize given code snippets in groups	CO4
24	Loops in flowgraphs	Interactive Lecture + Discussion, Flowcharts	Practice topic based problems	CO4
25	Global Data Flow Analysis	Interactive Lecture + Discussion	Practice topic based problems	CO4

26	Peephole Optimization	Interactive Lecture + Diagrammatic Explanation	Students have been asked to prepare note making of the given topic.	CO4
27	Issues in design of Code Generator	Interactive Lecture + Discussion	Identify issues in design of Code Generator	CO4
28	Register Allocation and Assignment	Interactive Lecture using chalk and talk	Hands-on: Create DAGs for basic blocks	CO4
29	DAG representation of basic block	Interactive Lecture + Diagrammatic Teaching	Hands-on: Create DAGs for basic blocks	CO4
30	Generating code from DAGs	Interactive Lecture + Discussion	Team activity: Generate code from given DAGs	CO4
31	Revision	Interactive Lecture + Discussion	Practice topic based problems	CO4
32	Revision	Interactive Lecture + Discussion	Peer-to-peer teaching for key concepts	CO4
33	Last Year Paper Discussion	Interactive Lecture + Discussion	Practice topic based problems	CO4
34	Last Year Paper Discussion	Interactive Lecture + Discussion	Practice topic based problems	CO4
35	Last Year Paper Discussion	Interactive Lecture + Discussion	Peer-to-peer teaching for key concepts	CO4
36	Doubt Session	Interactive Lecture + Discussion	Practice topic based problems	CO4

Course Faculty: Ms. Bhawna Kumari **Signature:**	**HOD: Dr. Sarika Chaudhary** **Signature:**

Suggested books:

1. Compilers Principle, Techniques & Tools – Alfread V. AHO, Ravi Sethi & J.D. Ullman; 1998 Addison Wesley.
2. Introduction to Automata Theory Languages & Computation, 3rd Edition, Hopcroft, Ullman, PEA

Suggested reference books :

1. Theory and practice of compiler writing, Tremblay & Sorenson, 1985, Mc. Graw Hill.
2. System software by Dhamdhere, 1986, MGH.
3. Principles of compiler Design, Narosa Publication
4. Elements compiler Design, Dr. M. Joseph, University Science Press

At the end of the course, the student will be able to :

CO1	Read and write finite automata and grammars for programming language constructs.
CO2	Understand the functionality of parsing mechanisms
CO3	Understand the concepts of storage administration for different programming environments.
CO4	Understand the concepts of optimization and generate the machine code.