|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| III- Year I- Semester | Name of the Course | L | T | P | C |
| PC3101 | Automata and Compiler Design | 2 | 0 | 1 | 3 |

**Course Objectives**: To learn fundamentals of Regular and Context Free Grammars and Languages

 To understand the relation between Regular Language and Finite Automata and machines

 To understand the relation between Contexts free Languages, Push Down Automata and Turing Machine

 To study various phases in the design of compiler and understanding the machine independent phases of compiler

 To understand machine dependent phases of compiler

**Syllabus:**

**UNIT-I: Finite Automata 12hrs**

Need for Automata Theory, Alphabet, Strings, Language, Operations, Deterministic Finite Automata, Non-Deterministic Finite Automata, Design of DFA, Design of NFA, Equivalence of NFA, DFA

**Finite Automata Conversions:**

Conversion from NFA to DFA, NFA ε to NFA, Minimization of DFA, Moore and Mealy Machines.

**UNIT-II: Regular Expressions and Grammars 14hrs**

**Regular Expressions**: Regular Sets, Identity Rules, Constructing finite Automata for a given regular expressions, Inter Conversion, Equivalence between FA and RE, Pumping Lemma of Regular Sets, Closure Properties of Regular Sets. Grammars: Grammars, Classification of Grammars, Regular grammars- Right and Left Linear

Regular Grammars, Equivalence between RG and FA, Inter Conversion, Context Free Grammar, Leftmost and Rightmost Derivations, derivation trees.

**Unit-III: Context Free Grammar, Push Down Automata and Turing Machines 12hrs**

**Context Free Grammar:**

Ambiguous Grammars, Simplification of Context Free Grammars, Normal Forms- ChomskyNormal Form, Griebach Normal Form.

**Push Down Automata (PDA):**

Definition, Model, Design of PDA, Deterministic PDA, Non-deterministic PDA, Equivalence of PDA and Context Free Grammars.

**Turing Machine (TM):** Definition, Model, Design of Turing Machine, Deterministic TM, Non-deterministic TM.

**UNIT-IV: Machine Independent Phases 14hs**

**Lexical Analysis**: Logical phases of compiler, Lexical Analysis, Lexemes Tokens and patterns, Lexical Errors. Syntax Analysis: Parsing definition, types of parsers, left recursion, left factoring, Top-down parser, First() and Follow(), LL(1) Grammars, Non- Recursive predictive parsing, Bottom- up Parsers, Shift Reduce Parser, LR parsers. Semantic Analysis: Syntax Directed Translation, L-attributed and S-attributed definitions.

**UNIT-V: Machine Dependent Phases 12hrs**

Intermediate Code Generation: Intermediate code, three address code, quadruples, triples, indirect triples, Directed acyclic graph. Code Optimization: Common sub expression elimination, copy propagation, dead code elimination, constant folding, strength reduction, loop optimization. Code Generation: Basic blocks & flow graphs, Peephole optimization.

**Text Books:**

1. Introduction to Automata Theory, Languages and Computation, J. E. Hopcroft, R. Motwani

and J. D. Ullman, 3rd Edition, Pearson, 2008

2. Theory of Computer Science-Automata, Languages and Computation, K. L. P. Mishra andN. Chandrasekharan, 3rd Edition, PHI, 2007

3. Compilers, Principles Techniques and Tools- Alfred V Aho, Monica S Lam, Ravi Sethi, Jeffrey D. Ullman,2nd ed, Pearson,2007.

**Reference Books**

1. Elements of Theory of Computation, Lewis H.P. & Papadimition C.H., Pearson /PHI

2. Theory of Computation, V. Kulkarni, Oxford University Press, 2013

3. Principles of compiler design, V. Raghavan, 2nd ed, TMH, 2011. 4. Compiler construction, Principles and Practice, Kenneth C Louden, CENGAGE

**Web Resources**

1.https://nptel.ac.in/courses/106/104/106104028/

2.https://nptel.ac.in/courses/106/105/106105190/

University AcademyYoutube Channel for Automata Theory and Compiler Design:

1.https://www.youtube.com/playlist?list=PL-JvKqQx2AtdhlS7j6jFoEnxmUEEsH9KH2.https://www.youtube.com/playlist?list=PL-JvKqQx2Ate5DWhppx-MUOtGNA4S3spTGATE Lectures:

1.https://www.youtube.com/playlist?list=PLEbnTDJUr\_IdM FmDFBJBz0zCsOFxfK2.https://www.youtube.com/playlist?list=PLMzYNEvC0P7FwwnrX

**Course Outcomes**: By the end the of the course, the student will be able to

**CO1**: Classify machines by their power to recognize languages.

**CO2**: Summarize language classes and grammars relationship among them with the help of Chomsky hierarchy.

**CO3**: Employ finite state machines in problem solving and also illustrate deterministic and non-deterministic machines.

**CO4**: Design and implement scanners and parsers.

**CO5**: Perform code optimization to improve performance and apply algorithms to generate code.