|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sl. No** | **Syllabus** | **Curriculum** | **Deployment Strategy and**  **Tool** | **Cross-cutting issues**  **integrated** | **PO, PSO and CO** | **Attainment Verification** |
| 1. | SYSTEM SOFTWARE AND COMPILER | 1. Compiler design principles provide an in-depth view of translation and optimization process. Compiler design covers basic translation mechanism and error detection & recovery. It includes lexical, syntax, and semantic analysis as front end, and code generation and optimization as back-end.  2.Application of Compilers are:   * Compiler design helps full implementation Of High-Level Programming Languages. * Support optimization for Computer Architecture Parallelism. * Design of New Memory Hierarchies of Machines. * Widely used for Translating Programs. * Used with other Software Productivity Tools. | 1. Chalk and   Talk method   1. PPT | * Business   Ethics   * Human   values | PO1:Engineering Knowledge  PO2:Problem Analysis  PO3:Design/Development Of Solutions  PO4:Conduct Investigations Of Complex Problems |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  | PSO1:Professional Skills  PSO2:Problem Solving Skill |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  | **CO1:Design** and **Apply** SIC assembler translation process of given source program to Machine language program, and **Analyse** the architectural features of SIC standard and SIC/XE systems, **Understand** Design Options of SIC Assembler. |
|  |  | **CO2:Understand** the Structure of the Compiler and **Design** the process of Lexical Analyzer. |
|  |  | **CO3:Analyze** and **Design** different types of parsers and **Apply** the Parser process for a given source string and respective grammar |
|  |  | **CO4:Design** and **Demonstrate** some programs using LEX and YACC programming languages. |
|  |  | **CO5:Understand** SDD and SDT, **Apply** the operations of intermediate code generation phase, code generation phase, and code optimization phase to a given example source code. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

