



**C. U. SHAH UNIVERSITY**  
**Wadhwan City**

**FACULTY OF:** - Technology and Engineering  
**DEPARTMENT OF:** - Information Technology  
**SEMESTER:** - VII  
**CODE:** - 4TE07CDE1  
**NAME:** – Compiler Design

**Teaching & Evaluation Scheme: -**

Subject Code	Subject Name	Teaching Scheme (Hours)				Credits	Evaluation Scheme							
		Th	Tu	Pr	Total		Theory				Practical			Total
							Session Exam		University Exam		Internal		University	
							Marks	Hours	Marks	Hours	Pr/ Viva	TW	Pr	
4TE07CDE1	Compiler Design	03	00	02	05	04	30	1.5	70	3.0	---	20	30	150

**Objective:** To introduce students to the techniques used in designing and writing compilers.

**Prerequisite:** Basic knowledge of data structure and programming languages.

Sr. No.	Course Content	Total Hours
1.	Introduction to the Translation Process, A Simple Compiler, comparison with other translators, overview and use of linker and loader, types of Compiler, Analysis of the Source Program, the phases of a Compiler, Cousins of the Compiler, grouping of Phases, Front-end and Back-end of compiler, pass structure, Error Handling in the various phases of compiler, Compiler Tools	04
2.	Role of Lexical Analyzer, Input Buffering, Specification and recognition of Tokens, A Language for Specifying Lexical Analyzers, Finite Automata, From a Regular Expression to an NFA, Design of a Lexical Analyzer Generator, Optimization of DFA, Hard Coding and Automatic Generation Lexical Analyzers,	08
3.	Top-Down Parsing Methods, Brute-Force approach, Recursive descent and predictive parsers, Bottom-Up Parsing, shift –reduce parsing, Operator-Precedence Parsing, LR Parsers, Using Ambiguous Grammars, Parser Generators, Error handling in parser, Automatic Generation of Parsers, Syntax-Directed Definitions and translation schemes Construction of Syntax Trees, Bottom-Up Evaluation of S-Attributed Definitions, L-Attributed Definitions	12
4.	Type Checking, type systems, type expressions, type conversion and overloading.	02
5.	Intermediate languages, syntax trees, postfix code, triples and indirect triples, Syntax Directed Translation Mechanisms And Attributed Mechanisms And Attributed Definition.	04
6.	Source Language Issues, Storage Organization, Storage organization and Allocation Strategies, and Access to Non local Names, Parameter Passing, Symbol Tables, Language Facilities for Dynamic Storage Allocation, Dynamic Storage Allocation Techniques.	04

7.	Issues in the Design of a Code Generator, The Target Machine, Run-Time Storage Management, Basic Blocks and Flow Graphs, Next-Use Information, A Simple Code Generator, Register Allocation and Assignment, The DAG Representation of Basic Blocks, Peephole Optimization, Generating Code from DAGs, Dynamic Programming Code-Generation Algorithm	08
8.	Sources of code optimization, optimization of basic blocks, data flow analysis, code improving transformations	06
<b>Total</b>		<b>48</b>

### **Outcomes:**

At the end of the course, students will be able to design and develop compiler for any kind of languages.

### **Books Recommended:**

1. Compiler Tools Techniques - A.V.Aho, Ravi Sethi, J.D.Ullman, Addison Wesley
2. The Theory and Practice Of Compiler Writing - Trembley J.P. And Sorenson P.G. Mcgraw-Hill
3. Modern Compiler Design - Dick Grune, Henri E. Bal, Jacob, Langendoen, WILEY India
4. Compiler Construction - Waite W.N. And Goos G., Springer Verlag
5. Compiler Construction-Principles and Practices - D.M.Dhamdhare, Mcmillian
6. Principles of Compiler Design, V. Raghavan, McGrawHill