MANDSAUR UNIVERSITY - Faculty of Engineering and Technology

Department of Computer Science & Engineering

SYLLABUS FOR 5 Semester B.Tech PROGRAMME

Compiler Design (CSE340TR1)

Type of Course: B.Tech

Prerequisite:

Rationale: -

Teaching and Examination Scheme:

Teaching Scheme			Credit	Examination Scheme					Total
Lecture Hrs/ Week	Tutorial Hrs/ Week	Lab Hrs/ Week		External		Internal			
				TH-EST	PR-EST	Th - MST	CET	Pr - MST	
2	1	0	3	60	-	30	10	-	100

SEE - Semester End Examination, **Th - MST** - Mid Semester Test, **Pr - MST** - Mid Semester Practical

Contents:

Sr.	Topic	Weightage	Teaching Hrs.
1	**Introduction of Compiler**: Introduction of Compiler, Major data Structure in compiler, BOOT Strapping & Porting, Compiler structure: analysis-synthesis model of compilation, various phases of a compiler, Lexical analysis: Input buffering, Specification & Recognition of Tokens, LEX.	20%	9
2	**Syntax analysis**: Syntax analysis: CFGs, Top down parsing, Brute force approach, recursive descent parsing, Transformation on the grammars, predictive parsing, bottom up parsing, operator precedence parsing, LR parsers (SLR,LALR, LR),Parser generation. Syntax directed definitions: Construction of Syntax trees, Bottom up evaluation of S-attributed definition, L-attribute definition, Top down translation, Bottom Up evaluation of inherited attributes Recursive Evaluation, Analysis of Syntax directed definition	20%	9
3	**Type checking**: Type checking: type system, specification of simple type checker, equivalence of expression, types, type conversion, overloading of functions and operations, polymorphic functions. Run time Environment: storage organization, Storage allocation strategies, parameter passing, Dynamic storage allocation, Symbol table	20%	9
4	**Intermediate code generation**: Intermediate code generation: Declarations, Assignment statements, Boolean expressions, Case statements, Back patching, Procedure calls Code Generation: Issues in the design of code generator, Basic block and flow graphs, Register allocation and assignment, DAG representation of basic blocks, peephole optimization, generating code from DAG.	20%	9
5	**Introduction to Code optimization**: Introduction to Code optimization: sources of optimization of basic blocks, loops in flow graphs, dead code elimination, loop optimization, Introduction to global data flow analysis, Code improving transformations ,Data flow analysis of structure flow graph Symbolic debugging of optimized code	20%	9

***Continuous Evaluation:**

It consists of Assignments/Seminars/Presentations/Quizzes/Surprise Tests (Summative/MCQ) etc.

Reference Books:

1. Compiler Construction: Principles and Practice
   By Willey Pub.Louden
2. Compiler Design in C
   By A. C. Holub
3. Compiler Design
   By B.S Raghavan

List of Practical:

1. Write a program in C/C++to check whether a string belongs to the grammar or not.
2. Write a program in C/C++to identify whether a given string is an identifier or not.
3. Write a program in C/C++ to check whether a given string is a keyword or not.
4. Write a program in C/C++ to implement the token separation operation.
5. Write a program in C/C++ to compute FIRST of non-terminals.
6. Write a program in C/C++ to compute FOLLOW(A).
7. Write a program to implement the Lexical analysis using C.
8. Write a program in C/C++ to implement recursive descendent parsing.
9. Write a program in C/C++ to calculate LEADING of non terminals.
10. Write a program in C/C++ to implement the Symbol table operation.
11. Write a program in C/C++ to implement operator precedence parsing.