CSL411	COMPILER LAB	CATEGORY	L	T	P	CREDIT
		PCC	0	0	3	2

Preamble: This course aims to offer students hands-on experience on compiler design concepts. Students will be able to familiarize with tools such as LEX and YACC and automate different phases of a compiler. This course helps the learners to enhance the capability to design and implement a compiler.

Prerequisite: A sound knowledge in C programming, Data Structures, Formal languages and Automata Theory and Compiler design.

Course Outcomes: After the completion of the course the student will be able to

CO 1	Implement lexical analyzer using the tool LEX. **(Cognitive Knowledge Level: Apply)**
CO 2	Implement Syntax analyzer using the tool YACC. **(Cognitive Knowledge Level: Apply)**
CO 3	Design NFA and DFA for a problem and write programs to perform operations on it. **(Cognitive Knowledge Level: Apply)**
CO 4	Design and Implement Top-Down parsers. **(Cognitive Knowledge Level: Apply)**
CO 5	Design and Implement Bottom-Up parsers. **(Cognitive Knowledge Level: Apply)**
CO 6	Implement intermediate code for expressions. **(Cognitive Knowledge Level: Apply)**

Mapping of course outcomes with program outcomes

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12
CO 1	✓	✓	✓	✓	✓			✓		✓		✓
CO 2	✓	✓	✓	✓	✓			✓		✓		✓
CO 3	✓	✓	✓	✓				✓		✓		✓
CO 4	✓	✓	✓	✓				✓		✓		✓
CO 5	✓	✓	✓	✓				✓		✓		✓
CO 6	✓	✓	✓	✓				✓		✓		✓

Assessment Pattern

Bloom's Category	Continuous Assessment Test %	End Semester Examination %
Remember	20	20
Understand	20	20
Apply	60	60
Analyze		
Evaluate		
Create		

Mark distribution

Total Marks	CIE	ESE	ESE Duration
150	75	75	3 hours

Continuous Internal Evaluation Pattern:

Attendance : 15 marks

Continuous Evaluation in Lab : 30 marks

Continuous Assessment Test : 15 marks

Viva-voce : 15 marks

Internal Examination Pattern: The marks will be distributed as Algorithm 30 marks, Program 20 marks, Output 20 marks and Viva 30 marks. Total 100 marks which will be converted out of 15 while calculating Internal Evaluation marks.

End Semester Examination Pattern: The marks will be distributed as Algorithm 30 marks, Program 20 marks, Output 20 marks and Viva 30 marks. Total 100 marks will be converted out of 75 for End Semester Examination.

Operating System to Use in Lab **:** Linux

Compiler/Software to Use in Lab **:** gcc, lex, yacc

Programming Language to Use in Lab **:** Ansi C

Fair Lab Record:

All Students attending the Compiler Lab should have a Fair Record. The fair record should be produced in the University Lab Examination. Every experiment conducted in the lab should be noted in the fair record. For every experiment in the fair record the right hand page should contain Experiment Heading, Experiment Number, Date of Experiment, Aim of Experiment, Details of Experiment including algorithm and Result of Experiment. The left hand page should contain a print out of the code used for the experiment and sample output obtained for a set of input.

SYLLABUS

1. Implementation of lexical analyzer using the tool LEX.
2. Implementation of Syntax analyzer using the tool YACC.
3. Application problems using NFA and DFA.
4. Implement Top-Down Parser.
5. Implement Bottom-up parser.
6. Simulation of code optimization Techniques.
7. Implement Intermediate code generation for simple expressions.
8. Implement the back end of the compiler.

PRACTICE QUESTIONS

List of Exercises/Experiments:

1. Design and implement a lexical analyzer using C language to recognize all valid tokens in the input program. The lexical analyzer should ignore redundant spaces, tabs and newlines. It should also ignore comments.
2. Implement a Lexical Analyzer for a given program using Lex Tool.
3. Write a lex program to display the number of lines, words and characters in an input text.
4. Write a LEX Program to convert the substring *abc* to *ABC* from the given input string.
5. Write a lex program to find out total number of vowels and consonants from the given input sting.
6. Generate a YACC specification to recognize a valid arithmetic expression that uses operators +, – , *,/ and parenthesis.

7. Generate a YACC specification to recognize a valid identifier which starts with a letter followed by any number of letters or digits.
8. Implementation of Calculator using LEX and YACC
9. Convert the BNF rules into YACC form and write code to generate abstract syntax tree.
10. Write a program to find ε – closure of all states of any given NFA with ε transition.
11. Write a program to convert NFA with ε transition to NFA without ε transition.
12. Write a program to convert NFA to DFA.
13. Write a program to minimize any given DFA.
14. Write a program to find First and Follow of any given grammar.
15. Design and implement a recursive descent parser for a given grammar.
16. Construct a Shift Reduce Parser for a given language.
17. Write a program to perform constant propagation.
18. Implement Intermediate code generation for simple expressions.
19. Implement the back end of the compiler which takes the three address code and produces the 8086 assembly language instructions that can be assembled and run using an 8086 assembler. The target assembly instructions can be simple move, add, sub, jump etc.