CSL 411 COMPILER DESIGN LAB

CYCLE 1

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. Write a program to find ε – closure of all states of any given NFA with ε transition.

3. Write a program to convert NFA with ε transition to NFA without ε transition.

4. Write a program to convert NFA to DFA.

5. Write a program to minimize any given DFA.

CYCLE 2

1. Implement a Lexical Analyzer for a given program using Lex Tool.

2. Write a lex program to display the number of lines, words and characters in an input text.

3. Write a LEX Program to convert the substring abc to ABC from the given input string.

4. Write a lex program to find out total number of vowels and consonants from the given input sting.

5. Generate a YACC specification to recognize a valid arithmetic expression that uses operators +, – , *,/ and parenthesis.

6. Generate a YACC specification to recognize a valid identifier which starts with a letter followed by any number of letters or digits.

7. Implementation of Calculator using LEX and YACC

CYCLE 3

1. Write a program to find the First and Follow of any given grammar (Eliminate left recursion in the program).

2. Design and implement a recursive descent parser for the given grammar

$E \rightarrow E+T|T$

$T \rightarrow T*F|F$

$F \rightarrow (E)|id$

3. Construct a Shift Reduce Parser for a given language.

4. Write a program to perform constant propagation.

5. Implement Intermediate code generation for simple expressions.

6. 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.