SAVEETHA SCHOOL OF ENGINEERING

SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES

INSTITUTE OF AI & ML

DEPARTMENT OF APPLIED MACHINE LEARNING

CSA14 - COMPILER DESIGN

LIST OF EXPERIMENTS

1. The lexical analyzer should ignore redundant spaces, tabs, and new lines. It should also ignore comments. Although the syntax specification states that identifiers can be arbitrarily long, you may restrict the length to some reasonable value. Develop a lexical Analyzer to identify identifiers, constants, and operators using the C program.
2. Extend the lexical Analyzer to Check comments, defined as follows in C:
   a) A comment begins with // and includes all characters until the end of that line.
   b) A comment begins with /* and includes all characters through the next occurrence of the character sequence */Develop a lexical Analyzer to identify whether a given line is a comment or not.
3. Design a lexical Analyzer for a given language should ignore the redundant spaces, tabs, and new lines and ignore comments.
4. Design a lexical Analyzer to validate operators to recognize the operators +,-,*,/ using regular Arithmetic operators.
5. Design a lexical Analyzer to find the number of whitespaces and newline characters.
6. Develop a lexical Analyzer to test whether a given identifier is valid or not.
7. Write a C program to find FIRST( ) - predictive parser for the given grammar

   S → AaAb / BbBa
   A → ∈
   B → ∈

8. Write a C program to find FOLLOW( ) - predictive parser for the given grammar

   S → AaAb / BbBa
   A → ∈
   B → ∈

9. Implement a C program to eliminate left recursion from a given CFG.

   S → (L) / a
   L → L , S / S

10. Implement a C program to eliminate left factoring from a given CFG.

    S → iEtS / iEtSeS / a
    E → b

11. Implement a C program to perform symbol table operations.
12. Write a C program to construct recursive descent parsing for the given grammar

    E → TE'
    E' → +TE' / ∈
    T → FT'
    T' → *FT' / ∈
    F → (E) / id

13. All languages have Grammar. When people frame a sentence we usually say whether the sentence is framed as per the rules of the Grammar or Not. Similarly use the same ideology, implement either Top Down parsing technique or Bottom Up Parsing technique to check whether the given input string is satisfying the grammar or not.
14. The main function of the Intermediate code generation is producing three address code statements for a given input expression. The three address codes help in determining the sequence in which operations are actioned by the compiler. The key work of Intermediate code generators is to simplify the process of Code generators. Write a C Program to Generate the Three address code representation for the given input statement.
15. Write a C program for implementing a Lexical Analyzer to Scan and Count the number of characters, words, and lines in a file
16. Write a C program to implement the back end of the compiler.
17. Write a C program to compute LEADING( ) – operator precedence parser for the given grammar

    E → E + T | T
    T → T * F | F
    F → (E) | id

18. Write a C program to compute TRAILING( ) – operator precedence parser for the given grammar

    E → E + T | T
    T → T * F | F
    F → (E) | id

19. The lexical analyzer should ignore redundant spaces, tabs and new lines. It should also ignore comments. Although the syntax specification states that identifiers can be arbitrarily long, you may restrict the length to some reasonable value. Write a LEX specification file to take input C program from a .c file and count the number of characters, number of lines & number of words.

    Input Source Program: (sample.c)

```
#include <stdio.h>
int main()
{
int number1, number2, sum;
printf("Enter two integers: ");
scanf("%d %d", &number1, &number2);
sum = number1 + number2;
printf("%d + %d = %d", number1, number2, sum);
 return 0;
}
```

20. Write a LEX program to print all the constants in the given C source program file.

    Input Source Program: (sample.c)

```
#define PI 3.14
#include<stdio.h>
#include<conio.h>
  void main()
  {

  int a,b,c = 30;
  printf("hello");
```

}Write a LEX program to count the number of Macros defined and header filesincluded in the C program.

Input Source Program: (sample.c)

```
#define PI 3.14
#include<stdio.h>
#include<conio.h>
void main()
{

int a,b,c = 30;
printf("hello");
}
```

21. In a class, an English teacher was teaching the vowels and consonants to the students. She says "Vowel sounds allow the air to flow freely, causing the chin to drop noticeably, whilst consonant sounds are produced by restricting the air flow". As a class activity the students are asked to identify the vowels and consonants in the given word/sentence and count the number of elements in each. Write an algorithm to help the student to count the number of vowels in the given sentence.
22. Write a LEX program to print all HTML tags in the input file.

    Input Source Program: (sample.html)

    ```
    <html>
    <body>
    <h1>My First Heading</h1>
    <p>My first paragraph.</p>
    </body>
    </html>
    ```
23. Write a LEX program which adds line numbers to the given C program file and display the same in the standard output.

    Input Source Program: (sample.c)

    ```
    #define PI 3.14
    #include<stdio.h>
    #include<conio.h>
    void main()
    {

    int a,b,c = 30;
    printf("hello");
    }
    ```
24. Write a LEX program to count the number of comment lines in a given C program and eliminate them and write into another file.

    Input Source File: (input.c)

    ```
    #include<stdio.h>
    int main()
    {

    int a,b,c; /*varible declaration*/
    printf("enter two numbers");
    scanf("%d %d",&a,&b);
    ```

```
c=a+b;//adding two numbers
printf("sum is %d",c);
return 0;
}
```

25. Write a LEX program to identify the capital words from the given input.
26. Write a LEX Program to check the email address is valid or not.
27. Write a LEX Program to convert the substring abc to ABC from the given input string.
28. The Company ABC runs with employees with several departments. The Organization manager had all the mobile numbers of employees. Assume that you are the manager and need to verify the valid mobile numbers because there may be some invalid numbers present. Implement a LEX program to check whether the mobile number is valid or not.
29. Implement Lexical Analyzer using LEX or FLEX (Fast Lexical Analyzer). The program should separate the tokens in the given C program and display them with the appropriate caption.

    Input Source Program: (sample.c)

```
#include<stdio.h>

void main()
{
int a,b,c = 30;

printf("hello");
}
```

30. In a class, an English teacher was teaching the vowels and consonants to the students. She says "Vowel sounds allow the air to flow freely, causing the chin to drop noticeably, whilst consonant sounds are produced by restricting the airflow". As a class activity, the students are asked to identify the vowels and consonants in the given word/sentence and count the number of elements in each. Write an algorithm to help the student to count the number of consonants in the given sentence.
31. Keywords are predefined, reserved words used in programming that have special meanings to the compiler. Keywords are part of the syntax and they cannot be used as an identifier. In general, there are 32 keywords. The prime function of Lexical Analyser is token Generation. Among the 6 types of tokens, differentiating Keywords and identifiers is a challenging issue. Thus write a LEX program to separate keywords and identifiers.
32. Write a LEX program to identify and count positive and negative numbers.
33. A networking company wants to validate the URL for its clients. Write a LEX program to implement the same.
34. School management wants to validate the DOB of all students. Write a LEX program to implement it.
35. Write a LEX program to check whether the given input is a digit or not.
36. A School student was asked to do basic mathematical operations. Implement a LEX program to implement the same.
37. Write a Lex program to count the frequency of the given word in a given sentence.
38. Write a Lex program to find the length of the longest word.
39. Write a Lex code to replace a word with another word in a file.
40. Write a FLEX (Fast Lexical Analyzer) program that should perform the token separation for the given C program and display with appropriate caption.