Experiment no-10

Write the Concept to implement yaac in lex program

Theory:- Lex and Yaac work together to process structured input

- Lex (Lexical Analyzer): Converts input into tokens
- YAAC (Yet Another Compiler Compiler): Parses tokens using grammar rules and processes them

Steps to implement

1. Lex Scans input and Sends tokens to YAAC.
2. YAAC parses the token and performs operation
3. Compile & run using yaac, flex and gcc.

Example:- Simple Calculator

Lex file (lexer.l)

Code:-

```
%{
  #include "y.tab.h"
%}
%%
[0-9]+   { yylval = atoi(yytext); return Number; }
[+\-*/()] { return yytext[0]; }
\n       { return 0; } // End of input
[\t]     { /* ignore spaces */ }
%%
int yywrap() { return 1; }
```

Teacher's Signature:

EXPT. NO | NAME | Page No. 23 | Date | YOUVA

YACC File (Parser.y)

Code:

```
%{
    #include <stdio.h>
    #include <stdlib.h>
%}
%token NUMBER
%%
expr: expr '+' expr { printf("Result: %d\n",
                      $1 + $3); }
    | expr '-' expr { printf("Result: %d\n",
                      $1 - $3); }
    | expr '*' expr { printf("Result: %d\n",
                      $1 * $3); }
    | expr '/' expr { printf("Result: %d\n",
                      $1 / $3); }
    | '(' expr ')' { $$ = $2; }
    | NUMBER { $$ = $1; }
    ;
%%
int main() {
    printf("Enter an expression:\n");
    yyparse();
    return 0;
}
int yyerror(char *msg) {
    printf("Syntax Error: %s\n", msg);
```

Teacher's Signature:

Example Execution

Enter an expression:

3+5*2

Result :13

EXPT NO:

NAME:

M T W T

Page No.: 24

Date:

```
    return 0;
}
```

Experiment No:- 12

(1) Aim: Write a lex program to count the number of characters and number of lines in the input

Code:-

```
/* Declaring two Counters one for number
   of lines other for number of characters */
%{
    int no_of_lines=0;
    int no_of_chars=0;
%}

/*** rule 1 Counts the number of lines, rule 2
   Counts the number of characters and rule 3
   Specifies when to stop taking input ***/
%%
\n      ++no_of_lines;
.       ++no_of_chars;
end     return 0;
%%
  /*** User Code Section ***/
  int yywrap (){
   int main (int argc, char **argv)
   {
      YYlex();
      print ("number of lines = %d, number of chars = %d\n",
              no_of_lines, no_of_chars);
      return 0;
   }
```

Teacher's Signature:

For Compile & Execution

```
flex char.l
gcc lex.yy.c -o char.exe
.\char.exe
```

Output

```
22
Yard
bit
number of lines = 3
number of chars = 10
```

(iii) Write a lex program to count number of Comments in the program

Code:

```
%{
#include<stdio.h>
int nc=0;
%}
%%
"/*"[a-zA-Z0-9\n\t]*"*/"  {nc++;}
"//"[a-zA-Z0-9\t]*"\n"  {nc++;}
%%
int main(int argc, char* argv[])
  {
    if(argc==2)
     {
       yyin=fopen(argv[1],"r");
     }
    else
      {
        printf("Enter the input\n");
        yyin=stdin;
      }
    yyout=fopen("Output.c","w");
    yylex();
    printf("The number of comment lines=%d\n",
                                    nc);
```

Teacher's Signature.

/* this is the program to write the input */

#include<stdio.h>

use ctrl+z button

The number of Comment line = 1

```
    fclose (yyin);
    fclose (yyout);
}

int yywrap()
{
    return 1;
}
```

Experiment No:-13

Aim:- Write a lex program to count the number of vowels using text file.

Code:-

```
%{
  #include <stdio.h>
  int vowel_count = 0;
%}
%%
[aeiouAEIOU] { vowel_count++; }
.|\n ; // Ignore all other characters
%%
int main(){
    printf("Enter the text file content :\n");
    yylex(); // call the lexer
    printf("\n Total number of vowels : %d\n",
                vowel_count);
    return 0;
}
int yywrap(){
    return 1;
}
```

Input

My name is Amrita and I am
Student at BIT Patna.

Output

Total number of digits :- 15

Experiment No - 14

Aim: Write a lex program to count the number of consonants from text file

Code:-

```
%{
    #include <stdio.h>
    int consonant_count=0;
%}
%%
[a-zA-Z] { if (!(yytext[0]=='a' || yytext[0]
                =='e' || yytext[0]=='i' ||
            yytext[0]=='o' || yytext[0]=='u' ||
            yytext[0]=='A' || yytext[0]=='E' ||
            yytext[0]=='I' || yytext[0]=='O' ||
            yytext[0]=='U'))
          consonant_count++; }
.|\n    ; // Ignore other characters
%%
int main(int argc, char *argv[]) {
    if (argc != 2) {
        printf("Usage: %s <input_file>\n",
               argv[0]);
        return 1;
    }
    FILE *file = fopen(argv[1], "r");
    if (!file) {
        printf("Error opening file: %s\n",
               argv[1]);
```

Teacher's Signature:

Input

Hello World ! This is a Lex program.

Output

Number of Consonants : 15

```
    return 1;
}
yyin = file;
yylex();
fclose(file);

printf("Number of Consonants: %d\n", Consonant_count);
return 0;
}
int yywrap(){
    return 1;
}
```

EXPT NO. NAME:

Page No. 31 YOUVA

Date

M T W T F S S

Experiment No: 15

Aim:- Write a lex program to generate as well as test even or odd numbers

Code:-

```
%{
  #include <stdio.h>
%}
%%
[0-9]+ {
    int num = atoi(yytext);
    if (num % 2 == 0)
        printf("%d is Even\n", num);
    else
        printf("%d is odd\n", num);
}
.|\n : // ignore other characters
%%
int main(int argc, char *argv[]){
    if (argc != 2){
        printf("Usage: %s <input_file>\n",
                argv[0]);
        return 1;
    }
    FILE *file = fopen(argv[1], "r");
    if (!file){
        printf("Error opening file: %s\n",
                argv[1]);
```

Teacher's Signature:

```
flex even-odd.l
gcc lex.yy.c -o even-odd -ll
```

Input

```
45
78
102
67
88
```

Output:-

```
45 is odd
78 is even
182 is even
67 is odd
68 is even.
```

```
    return 1;
}
FILE *file = fopen(argv[1], "r");
if (!file) {
    printf("Error opening file: %s\n", argv[1]);
    return 1;
}
yyin = file;
yylex();
fclose(file);
return 0;
}
int yywrap(){
    return 1;
}
```

```
return 1;
}
FILE *file = fopen(argv[1], "r");
if (!file) {
    printf("Error opening file: %s\n", argv[
    return 1;
}
yyin = file;
yylex();
fclose(file);
return 0;
}
int yywrap(){
    return 1;
}
```