**Name: Niranjan Vinod Patil.**

**Batch: B31.**

**Roll No: SCETTYB305.**

**Course: Compiler Design.**

**----------------------------------------------------------------------------------------------------**

**Assignment No: 05**

**Aim:**

Design sample syntax analyzer and implement same for c++ language using YAAC tool. Syntax analyzer should recognize syntax r=errors like missing parenthesis, missing semicolons etc

**Program**

cd\_ass.l  
  
%{  
    #include "cd\_ass5.tab.h"  
%}  
  
%%  
  
"for" {return FOR;}  
"if" {return IF;}  
"else" {return ELSE;}  
"while" {return WHILE;}  
"do" {return DO;}  
"switch" {return SWITCH;}  
"case" {return CASE;}  
"break" {return BREAK;}  
"default" {return DEFAULT;}  
  
[a-zA-Z]+ {return alpha;}  
[0-9]+ {return num;}  
[ \n\t] {;}  
"==" {return EQUAL;}  
"++" {return INC;}  
"--" {return DEC;}  
">=" {return GE;}  
"<=" {return LE;}  
">" {return GT;}  
"<" {return LT;}  
"!=" {return NE;}  
"&&" {return AND;}  
"||" {return OR;}  
"$" {return END;}    
.  {return yytext[0];}    
%%  

cd\_ass5.y  
  
%{  
#include<stdio.h>  
#include<stdlib.h>  
%}  
  
%token FOR EQUAL INC DEC alpha num GE LE GT LT NE AND OR END WHILE DO SWITCH CASE BREAK DEFAULT  
%left '+''-'  
%left '\*''/'  
%right '='  
%left '^'  
%nonassoc UMINUS  
%nonassoc IF  
%nonassoc ELSE  
//%nonassoc WHILE  
//%nonassoc DO  
//%nonassoc SWITCH  
//%nonassoc CASE  
//%nonassoc BREAK  
%left LT GT LE GE EQUAL NE  
%left AND OR  
  
%%  
  
S:    ST END {printf("\nAccepted\n"); exit(0);}  
;  
ST:    IF'('F')''{'ST'}'%prec IF  
    |IF'('F')''{'ST'}' ELSE'{'ST'}'  
    |FOR '('E';'F';'E')''{'ST'}'  
    |WHILE'('F')''{'ST'}'  
    |DO'{'ST'}'WHILE'('F')'';'  
    |SWITCH'('E')''{'CS'}'  
    |E';'  
    |E';'ST  
;  
  
CS:    CASE E':'ST BREAK';'  
    |CASE E':'ST BREAK';'CS  
    |CS DEFAULT':'ST  
;  
  
F:    C LO C  
    |C  
;  
  
C:    E RELOP E  
    |E  
;  
  
LO:    AND  
    |OR  
;  
  
RELOP:    LT|  
    GT|  
    LE|  
    GE|  
    EQUAL|  
    NE  
;  
  
E:    alpha '=' E      
    |E '+' E  
    |E '-' E  
    |E '\*' E  
    |E '/' E  
    |E '^' E  
    |'('E')'  
    |'-'E %prec UMINUS  
    |alpha  
    |num  
    |alpha INC  
    |alpha DEC  
;  
%%  
  
int main(){  
    yyparse();  
    return 0;  
}  
  
int yywrap(){  
    return 0;  
}  
  
int yyerror(char\*s){  
    printf("Error");  
}

**Output**

[student@localhost ~]$ bison -d cd\_ass5.y  
cd\_ass5.y: warning: 1 shift/reduce conflict [-Wconflicts-sr]  
[student@localhost ~]$ flex cd\_ass5.l  
[student@localhost ~]$ gcc -o cd\_ass5 cd\_ass5.tab.c lex.yy.c  
cd\_ass5.tab.c: In function ‘yyparse’:  
cd\_ass5.tab.c:1202:16: warning: implicit declaration of function ‘yylex’ [-Wimplicit-function-declaration]  
       yychar = yylex ();  
                ^~~~~  
cd\_ass5.tab.c:1337:7: warning: implicit declaration of function ‘yyerror’; did you mean ‘yyerrok’? [-Wimplicit-function-declaration]  
       yyerror (YY\_("syntax error"));  
       ^~~~~~~  
       yyerrok  
[student@localhost ~]$ ./cd\_ass5  
if(a<b){c=a+b;}else{c=a-b;}$  
  
Accepted  
[student@localhost ~]$

