**PRACTICAL-1**

**AIM:**

Implement a lexical analyzer for a subset of C using LEX Implementation should support Error handling.

**IMPLEMENTATION:**

* lex <filename with .l extension>
* gcc <newly created .c file> -o <file name for exe file>
* <filename of exe file>

In this case, create an extra text file named abc.txt which will contain some C code to work as input for lexical analysis.

**PROGRAM CODE:**

%%

"#" {printf("\n %s \t Preprocessor",yytext);}

"main"|"printf"|"scanf" {printf("\n%s\tfunction",yytext);}

"if"|"else"|"int"|"unsigned"|"long"|"char"|"switch"|"case"|"struct"|"do"|"while"|"void"|"for"|"float"|"continue"|"break"|"include" { printf("\n%s\tKeyword",yytext); }

[\_a-zA-Z][\_a-zA-Z0-9]\* {printf("\n%s\tIdenifier",yytext);}

"+"|"/"|"\*"|"-" {printf("\n%s\tOperator",yytext);}

"="|"<"|">"|"!="|"=="|"<="|">=" {printf("\n%s\tRelational Operator",yytext);}

"%d"|"%s"|"%c"|"%f" {printf("\n%s\tTokenizer",yytext);}

"stdio.h"|"conio.h"|"math.h"|"string.h"|"graphics.h"|"dos.h" {printf("\n%s\tHeader File",yytext);}

";"|"," {printf("\n%s\tDelimiter",yytext);}

"("|")" {if(strcmp(yytext,"(")==0)

     {

        printf("\n%c\tOpening Parenthesis",yytext[0]);

     }

     else

     {

        printf("\n%c\tClosing Parenthesis",yytext[0]);

     }

    ;}

"{" {printf("\n%s\tStart Of Function/Loop",yytext);}

"}" {printf("\n%s\tEnd of Function",yytext);}

%%

int yywrap(void)

{

  return 1;

}

int main()

{

int i;

FILE \*fp;

fp=fopen("abc.txt","r");

    if(fp==NULL)

{

    printf("Unable To Open File");

}

else

{

    yyin=fp;

}

yylex();

printf("\n\nPARTH PATEL\n19DCS098");

return 0;

}

**TEXT FILE:**

#include<stdio.h>

void main()

{

printf("hii");

printf("19DCS098");

}

**OUTPUT:**



