**Practical 4 : Lex**

* Lex is used for regular expression matching for tokenizing the contents of file.
* Lex file has 3 components:-
  + - * + %{ …%} : In these section, all the include statements and other variable declarations are done.
        + %%.....%% : In these , all the regular expressions are listed, and the code that is to be executed on occurrence of match of that particular regular expression is written in front of regular expression between {…..}
        + main() : In this section, generally all the rest of the common logic, eg: printing the outputs etc. are done.

Example code (1):-

d [0-9]

l [A-Za-z]

%{

#include<stdio.h>

int id=0,num=0;

%}

%%

(\_|{l})({l}|{d})\* {printf("Its identifier number %d",++id);}

({d}+)|(.({d})+)? {printf("Its numeral number %d",++num);}

%%

int main()

{

yylex();

printf("%d %d",id,num);

return 0;

}

This lex code will match all the valid identifiers as well as all the valid numbers and show their count.

===========================================

Example code(2):-

d [0-9]

l [A-Za-z\_]

%{

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

int i=0,flag=0,n=0,j=0,\*dt;

char \*\*sym;

%}

%%

({l})({l}|{d})\* {

flag=0;

for(i=0;i<n;i++)

{

if(!strcmp(yytext,sym[i]))

{

printf("id%d",i+1);

flag=1;

break;

}

}

if(flag==0)

{

n++;

printf("id%d",n);

sym=(char \*\*)realloc(sym,n\*sizeof(char \*));

dt=(int \*)realloc(dt,n\*sizeof(int));

sym[n-1]=(char \*)malloc(256\*sizeof(char));

strcpy(sym[n-1],yytext);

dt[n-1]=1;

}

}

("int")(" ")({l})({l}|{d})\*(";") {

j=0;

char \*str=(char \*)malloc(256\*sizeof(char));

while(j<(yyleng-4))

{

str[j]=yytext[j+4];

j++;

}

printf("%s",str);

flag=0;

for(i=0;i<n;i++)

{

if(!strcmp(str,sym[i]))

{

printf("int id%d",i+1);

flag=1;

break;

}

}

if(flag==0)

{

n++;

printf("int id%d",n);

sym=(char \*\*)realloc(sym,n\*sizeof(char \*));

dt=(int \*)realloc(dt,n\*sizeof(int));

sym[n-1]=(char \*)malloc(256\*sizeof(char));

strcpy(sym[n-1],str);

dt[n-1]=1;

}

}

("float")(" ")({l})({l}|{d})\*(";") {

j=0;

char \*str=(char \*)malloc(256\*sizeof(char));

while(j<(yyleng-4))

{

str[j]=yytext[j+4];

j++;

}

printf("%s",str);

flag=0;

for(i=0;i<n;i++)

{

if(!strcmp(str,sym[i]))

{

printf("float id%d",i+1);

flag=1;

break;

}

}

if(flag==0)

{

n++;

printf("float id%d",n);

sym=(char \*\*)realloc(sym,n\*sizeof(char \*));

dt=(int \*)realloc(dt,n\*sizeof(int));

sym[n-1]=(char \*)malloc(256\*sizeof(char));

strcpy(sym[n-1],str);

dt[n-1]=2;

}

}

%%

int main()

{

yylex();

return 0;

}

This code will make a symbol table ,and will give a unique id number to every unique symbol. Hence, this code will substitute their respective id numbers for all symbols from symbol table.

Eg: int a 🡺 int id1

a=b+c 🡺 id1=id2+id3