Compiler Design Lab

Assignment-2

Implementation of Lexical Analyzer and Symbol table for the patterns using Lex

Name : Vikraman S
Reg No. : 185001195

Code:

```
/*lex program for lexical analyzer*/
%{
#include<stdio.h>
#include<string.h>

typedef struct
{
    char name[10];
    char type[10];
    char value[10];
}symbol_table;

symbol_table t[10];
int ind=0;
char type[10];

int notin(char *a)
{
    int i;
    for(i=0;i<=ind;i++)
    {
        if(strcmp(a,t[i].name)==0)
            return 0;
    }
    return 1;
}
%}

keyword
int|float|char|double|while|for|do|if|break|continue|void|switch|case|long|
struct|const|typedef|return|else|goto|printf|scanf
fun [ ][a-zA-Z].*[(].*[)]
numconst [0-9]+|[0-9]+[.][0-9]+
strconst \".*\"
charconst \'.\'
preprocessor #.+
identifier [a-zA-Z][a-zA-Z0-9]*
specialchar [(){},;]
comment [/*].*[*/]|[//].*
relop "<"|"<="|">"|">="|"=="|"!="
ariop "+"|"-"|"*"|"/"
assign "="

%%

{keyword} {
    printf("KW ");
```

```
    if(strcmp(yytext,"int")==0)
    {
        strcpy(type,yytext);
    }
    else if(strcmp(yytext,"float")==0)
    {
        strcpy(type,yytext);
    }
    else if(strcmp(yytext,"char")==0)
    {
        strcpy(type,yytext);
    }
    else if(strcmp(yytext,"double")==0)
    {
        strcpy(type,yytext);
    }
}
{fun} {printf("FUN ");}
{numconst} {
    printf("NUMCONST ");
    if(strcmp(t[ind].value,"null")==0)
    {
        strcpy(t[ind].value,yytext);
        ind++;
    }
}
{charconst} {
    printf("CHARCONST ");
    if(strcmp(t[ind].value,"null")==0)
    {
        strcpy(t[ind].value,yytext);
        ind++;
    }
}
{strconst} {printf("STRCONST ");}
{preprocessor} {printf("PRE ");}
{identifier} {
    printf("ID ");
    if(notin(yytext))
    {
        strcpy(t[ind].type,type);
        strcpy(t[ind].name,yytext);
        strcpy(t[ind].value,"null");
        ind++;
    }
}
{specialchar} {printf("SP ");}
{comment} {printf("COMMENT ");}
{relop} {printf("RELOP ");}
{ariop} {printf("ARIOP ");}
{assign} {
    printf("ASSIGN ");
    ind--;
    }

%%

int yywrap(void){}

void main()
{
```

```
    FILE *fp;
    char str[100];
    fp=fopen("sample.txt","r");
    printf("\nInput :\n\n");
    while(!feof(fp))
    {
        if(!feof(fp))
        {
            fscanf(fp," %[^\n]",str);
            printf("%s\n",str);
        }
    }
    fclose(fp);

    printf("\n\nOutput :\n\n");
    yyin=fopen("sample.txt","r");
    yylex();
    int i;
    printf("\n\nType\tName\tValue\n");
    for(i=0;i<=ind;i++)
    {
        printf("%s\t%s\t%s\n",t[i].type,t[i].name,t[i].value);
    }
}
```

Output:

```
Input :

void main()
{
int a=12,b;
char c='a',g='2';
float d=1.1,e;
int i=0;
if(a>b)
printf ("a is greater");
else
printf ("b is greater");
while(i<10)
{
printf("Hi");
i++;
}
//bye
//bye


Output :

KW FUN
SP
KW  ID ASSIGN NUMCONST SP ID SP
KW  ID ASSIGN CHARCONST SP ID ASSIGN CHARCONST SP
KW  ID ASSIGN NUMCONST SP ID SP
KW  ID ASSIGN NUMCONST SP
KW SP ID RELOP ID SP
KW  SP STRCONST SP SP
KW
KW  SP STRCONST SP SP
KW SP ID RELOP NUMCONST SP
SP
KW SP STRCONST SP SP
ID ARIOP ARIOP SP
SP
COMMENT



Type    Name    Value
int     a       12
int     b       null
char    c       'a'
char    g       '2'
float   d       1.1
float   e       null
int     i       0
```