

## **Compiler Design Lab (CS 306)**

### **Week-4 Lab**

AP19110010246  
Ghanta Jaiya  
CSE-E

**Implement lexical analyzer using LEX for recognizing the following tokens:**

- **A minimum of 10 keywords of your choice**
- **Identifiers with the regular expression : letter(letter | digit)\***
- **Signed as well as unsigned integers**
- **Signed as well as unsigned Floats in fractional as well as exponential notation.**
- **Relational operators: <, >, <=, >=, ==, !=**
- **Assignment Operator:=**
- **Ignores everything between comments: single line as well as multiline comments as in C**
- **Storing identifiers in the symbol table**
- **Using files for input and output.**

## Code:

### Lex:

```
/*Definition Section*/
%{
#include<stdio.h>
%}
%%

auto|double|int|struct|break|else|long|switch|case|enum|register|typedef|char|extern|return|union|continue|for|signed|void|do|if|static|while|default|goto|sizeof|volatile|const|float|short {ECHO; printf("\tKEYWORD\n");} //rule for keyword

[{};,(,)] {ECHO; printf("\tSEPERATOR\n");} //rule for separator

[+/-/=*%] {ECHO; printf("\tOPERATOR\n");} //rule for operator

[a-zA-Z_][a-zA-Z0-9_]* {ECHO; printf("\tIdentifier\n");} //rule for identifier

[0-9]+ {ECHO;printf("\t Digit\n", yytext);} //rule for digit

"<" | ">" | "<=" | ">=" | "==" | "!=" {ECHO; printf("\tRELATIONAL OPERATOR\n");} //rule for relational operator

[/][^]*[^(^*/)[^]*]+([/*/][^]*[^(^*/)[^]*]+)*[/] {} //rule for skipping multi line comments

. {ECHO;printf("\t Other\n");} //rule for skipping other characters (not above mentioned)
%%

/*call the yywrap function*/
int yywrap(){
int main(void)
{
/*call the yylex function.*/
yyin=fopen("fibonacci.c","r");
yylex();
return 0;}
}
```

## Output:

```
D:\compiler_design_lab_codes\lex_programs\lab_assignment1>problem2.exe
#      Other
include Identifier
      Other
stdio  Identifier
.      OPERATOR
h      Identifier
>      Other

int    KEYWORD
      Other
main   Identifier
(      SEPERATOR
)      SEPERATOR

{      SEPERATOR

      Other
      Other
      Other
      Other

      Other
      Other
      Other
      Other

int    KEYWORD
      Other
n1     Identifier
      Other
=      OPERATOR
      Other
0      Digit
,      SEPERATOR
      Other
n2     Identifier
      Other
=      OPERATOR
      Other
1      Digit
,      SEPERATOR
      Other
n3     Identifier
,      SEPERATOR
      Other
i      Identifier
,      SEPERATOR
      Other
number Identifier
```

```

number Identifier
; SEPERATOR

    Other
    Other
    Other
    Other
printf Identifier
( SEPERATOR
" Other
Enter Identifier
    Other
the Identifier
    Other
number Identifier
    Other
of Identifier
    Other
elements Identifier
: Other
" Other
) SEPERATOR
; SEPERATOR

    Other
    Other
    Other
    Other
scanf Identifier
( SEPERATOR
" Other
% OPERATOR
d Identifier
" Other
, SEPERATOR
    Other
& Other
number Identifier
) SEPERATOR
; SEPERATOR

    Other
    Other
    Other
    Other
printf Identifier
( SEPERATOR
" Other
\ Other

```

n	Identifier
%	OPERATOR
d	Identifier
	Other
%	OPERATOR
d	Identifier
"	Other
,	SEPERATOR
	Other
n1	Identifier
,	SEPERATOR
	Other
n2	Identifier
)	SEPERATOR
;	SEPERATOR
	Other
	Other
	Other
/	OPERATOR
/	OPERATOR
printing	Identifier
	Other
0	Digit
	Other
and	Identifier
	Other
1	Digit
	Other
	Other
	Other
	Other
for	KEYWORD
	Other
(	SEPERATOR
i	Identifier
	Other
=	OPERATOR
	Other
2	Digit
;	SEPERATOR
	Other
i	Identifier
	Other
	Other
number	Identifier
;	SEPERATOR
	Other

+	OPERATOR
+	OPERATOR
i	Identifier
)	SEPERATOR
	Other
/	OPERATOR
/	OPERATOR
loop	Identifier
	Other
starts	Identifier
	Other
from	Identifier
	Other
2	Digit
	Other
because	Identifier
	Other
0	Digit
	Other
and	Identifier
	Other
1	Digit
	Other
are	Identifier
	Other
already	Identifier
	Other
printed	Identifier
	Other
	Other
	Other
	Other
{	SEPERATOR
	Other
	Other
	Other
	Other
	Other
	Other
	Other
	Other
n3	Identifier
	Other
=	OPERATOR
	Other
n1	Identifier

```

    Other
+    OPERATOR
    Other
n2    Identifier
;    SEPERATOR
```

```

    Other
    Other
    Other
    Other
    Other
    Other
    Other
    Other
printf Identifier
(    SEPERATOR
"    Other
    Other
%    OPERATOR
d    Identifier
"    Other
,    SEPERATOR
    Other
n3    Identifier
)    SEPERATOR
;    SEPERATOR
```

```

    Other
    Other
    Other
    Other
    Other
    Other
    Other
    Other
n1    Identifier
    Other
=    OPERATOR
    Other
n2    Identifier
;    SEPERATOR
```

```

    Other
    Other
    Other
    Other
    Other
    Other
```

```
Other
+ OPERATOR
Other
n2 Identifier
;
```

```
Other
Other
Other
Other
Other
Other
Other
Other
printf Identifier
( SEPERATOR
" Other
Other
% OPERATOR
d Identifier
" Other
, SEPERATOR
Other
n3 Identifier
) SEPERATOR
;
```

```
Other
Other
Other
Other
Other
Other
Other
Other
n1 Identifier
Other
= OPERATOR
Other
n2 Identifier
;
```

```
Other
Other
Other
Other
Other
Other
```



```

    Other
    Other
n2    Identifier
    Other
=    OPERATOR
    Other
n3    Identifier
;    SEPERATOR

    Other
    Other
    Other
    Other
}    SEPERATOR

    Other
    Other
    Other
    Other
return KEYWORD
    Other
0    Digit
;    SEPERATOR

}    SEPERATOR
```