College Of Engineering Trivandrum

Compiler Design Lab

CS431

Submitted By:
Neethu S
S6 CSE Roll No:42

TVE18CS043

Department of Computer Science

December 18, 2021

Contents

Cycle 3 Experiment 2

1 LEX Programs - 2

1.1 Aim

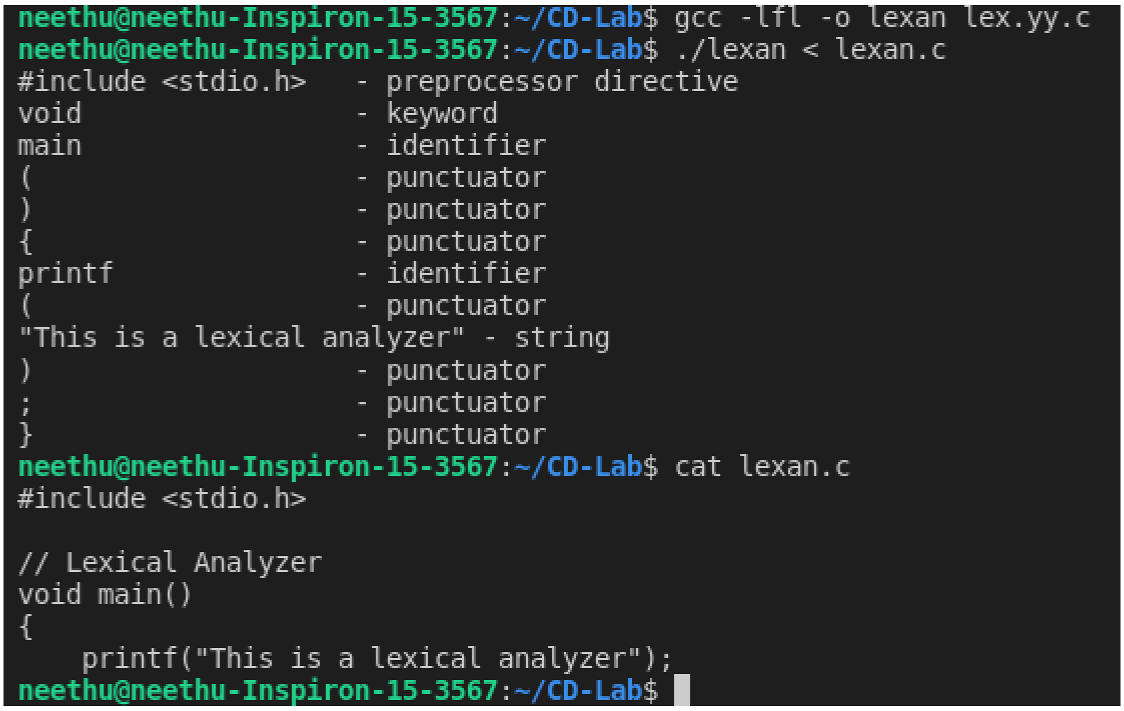
To write a LEX program for:

1. Write a LEX program to implement a lexical analyzer
2. To write a YACC program to recognize valid identifiers, operators and keywords in a given C program

1.2 Code

LEX program to implement a lexical analyzer

```
%{
    #include <stdio.h>
    #include <stdbool.h>

    bool is_single_comment = false, is_comment = false;
%}
identifier [a-zA-Z][a-zA-Z0-9]*
%%
#.* {
printf("%-20s - %s\n", yytext, "preprocessor directive");
}
int |
float |
char |
double |
while |
for |
struct |
typedef |
do |
if |
break |
continue |
void |
switch |
return |
else |
goto {
printf("%-20s - %s\n", yytext, "keyword");
}
[\/\/] {
is_comment = is_single_comment = true;
}
"/*" {
is_comment = true;
}
"*/" {
if (is_comment)
is_comment = false;
}
[\n] {
if (is_single_comment)
is_comment = is_single_comment = false;
}
[ \t\r]+ {
; // white space
}
[\{\}\,\;\:\[\]\(\)] {
if (!is_comment)
printf("%-20s - %s\n", yytext, "punctuator");
}
{identifier}(\[[0-9]*\])? {
if (!is_comment)
printf("%-20s - %s\n", yytext, "identifier");
}
\".*\" {
if (!is_comment)
printf("%-20s - %s\n", yytext, "string");
}
[0-9]*\.[0-9]+ {
if (!is_comment)
printf("%-20s - %s\n", yytext, "float");
}
[0-9]+ {
if (!is_comment)
printf("%-20s - %s\n", yytext, "integer");
}
= {
if (!is_comment)
printf("%-20s - %s\n", yytext, "assignment operator");
}
[\+\-\*\/] {
if (!is_comment)
printf("%-20s - %s\n", yytext, "arithmetic operator");
}
\! |
\&\& |
\|\| {
if (!is_comment)
printf("%-20s - %s\n", yytext, "logical operator");
}
\<= |
\>= |
\< |
== |
\> {
if (!is_comment)
printf("%-20s - %s\n", yytext, "relational operator");
}
%%

void main() {
yylex();
}

int yywrap() {
return 1;
}
```



```
neethu@neethu-Inspiron-15-3567:~/CD-Lab$ gcc -lfl -o lexan lex.yy.c
neethu@neethu-Inspiron-15-3567:~/CD-Lab$ ./lexan < lexan.c
#include <stdio.h>   - preprocessor directive
void                 - keyword
main                 - identifier
(                    - punctuator
)                    - punctuator
{                    - punctuator
printf               - identifier
(                    - punctuator
"This is a lexical analyzer" - string
)                    - punctuator
;                    - punctuator
}                    - punctuator
neethu@neethu-Inspiron-15-3567:~/CD-Lab$ cat lexan.c
#include <stdio.h>

// Lexical Analyzer
void main()
{
    printf("This is a lexical analyzer");
neethu@neethu-Inspiron-15-3567:~/CD-Lab$
```

YACC program to recognize valid identifiers, operators and keywords in a given C program

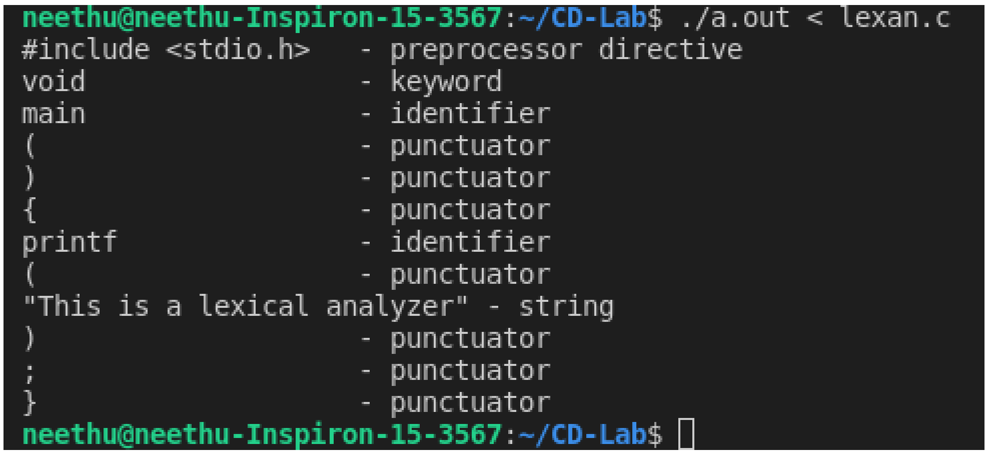
lex file

```
%{
#include <stdio.h>
#include "identify.h"

extern int yylval;
%}

%%
[ \t];
[+|-|*|/|=|<|>] {
printf("%-20s - %s\n", yytext, "operator");
return OP;
}
```

```
[0-9]+ {
printf("%-20s - %s\n", yytext, "number");
return DIGIT;
}
int|char|bool|float|void|for|do|while|if|else|return|void {
printf("%-20s - %s\n", yytext, "keyword");
return KEY;
}
[a-zA-Z0-9]+ {
printf("%-20s - %s\n", yytext, "identifier");
return ID;
}
. ;
%%
```

yacc file

```
%{
#include <stdio.h>
#include <stdlib.h>

int id = 0, dig = 0, key = 0, op = 0;
%}
%token DIGIT ID KEY OP

%%
input:
DIGIT input { dig++; }
| ID input { id++; }
| KEY input { key++; }
| OP input {op++;}
| DIGIT { dig++; }
| ID { id++; }
| KEY { key++; }
| OP { op++;}
;
%%

#include <stdio.h>
extern int yylex();
extern int yyparse();
extern FILE *yyin;

void main()  {
yyin = stdin;
do {
yyparse();
} while (!feof(yyin));
printf("%-20s: %d\n", "identifiers", id);
printf("%-20s: %d\n", "numbers", dig);
printf("%-20s: %d\n", "keywords", key);
printf("%-20s: %d\n", "operators", op);
}

int yyerror() {
printf("Error\n");
}
```



```
neethu@neethu-Inspiron-15-3567:~/CD-Lab$ ./a.out < lexan.c
#include <stdio.h>    - preprocessor directive
void                  - keyword
main                  - identifier
(                     - punctuator
)                     - punctuator
{                     - punctuator
printf                - identifier
(                     - punctuator
"This is a lexical analyzer" - string
)                     - punctuator
;                     - punctuator
}                     - punctuator
neethu@neethu-Inspiron-15-3567:~/CD-Lab$ 
```

1.3 Result

Implemented the lex programs in Ubuntu 20.04 with kernel and the above outputs were obtained.