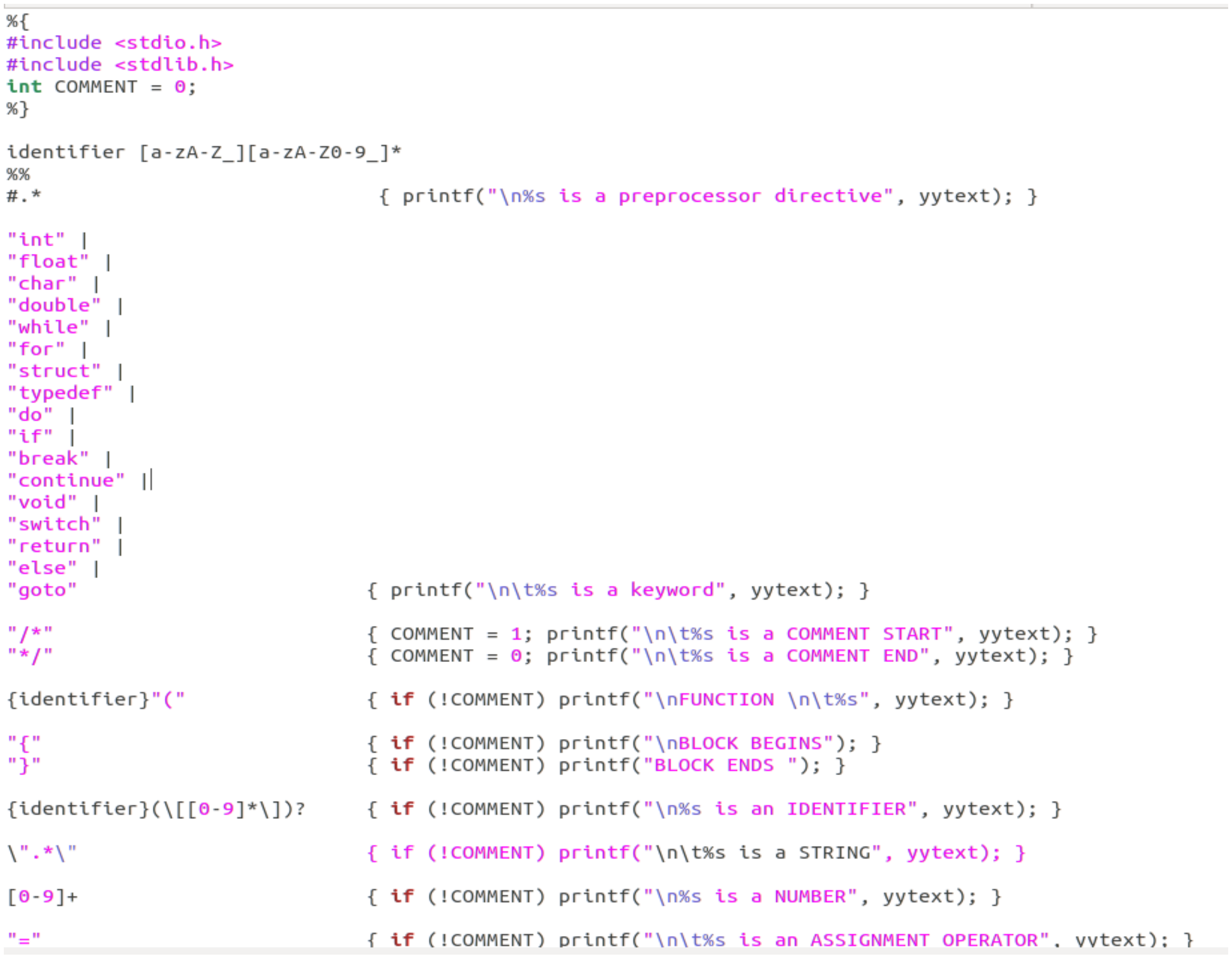
Date: 21/07/2025

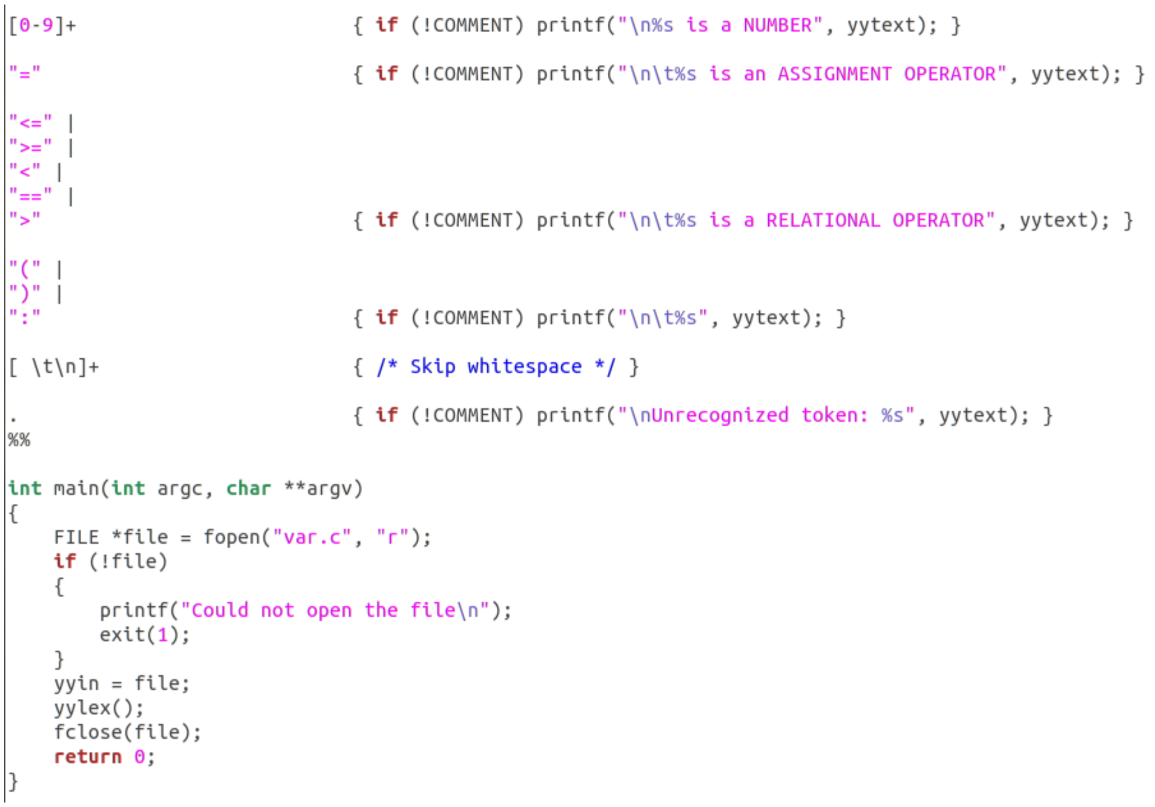
Experiment No: 01

Aim: To implement Lexical Analyzer Using Lex Tool

Code:

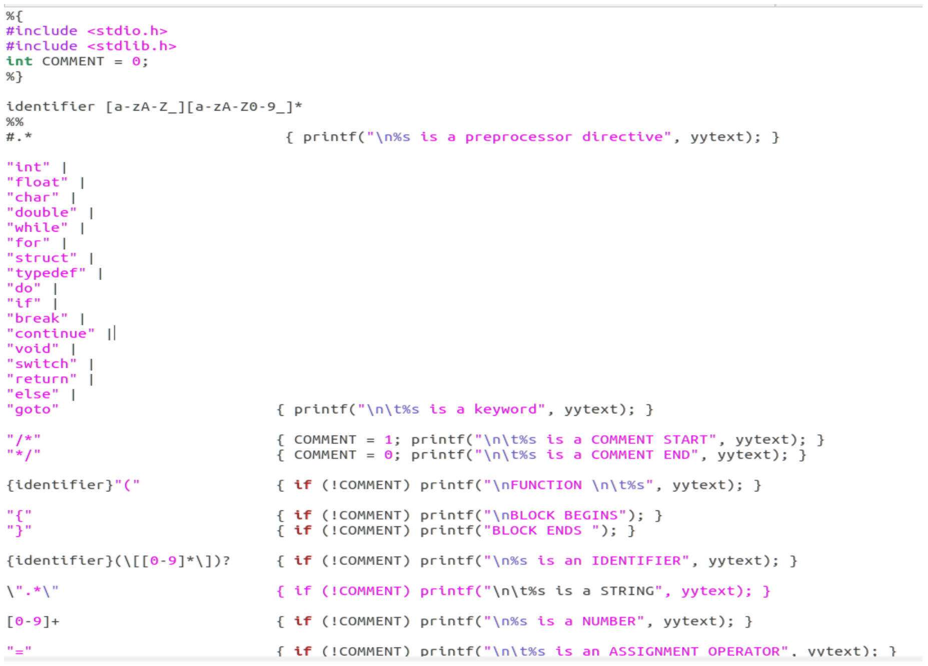


```
%{
#include <stdio.h>
#include <stdlib.h>
int COMMENT = 0;
%}

identifier [a-zA-Z_][a-zA-Z0-9_]*
%%
#.*                             { printf("\n%s is a preprocessor directive", yytext); }

"int" |
"float" |
"char" |
"double" |
"while" |
"for" |
"struct" |
"typedef" |
"do" |
"if" |
"break" |
"continue" |
"void" |
"switch" |
"return" |
"else" |
"goto"                         { printf("\n\t%s is a keyword", yytext); }

"/*"                           { COMMENT = 1; printf("\n\t%s is a COMMENT START", yytext); }
"*/"                           { COMMENT = 0; printf("\n\t%s is a COMMENT END", yytext); }

{identifier}"("                { if (!COMMENT) printf("\nFUNCTION \n\t%s", yytext); }

"{"                            { if (!COMMENT) printf("\nBLOCK BEGINS"); }
"}"                            { if (!COMMENT) printf("BLOCK ENDS "); }

{identifier}(\[[0-9]*\])?      { if (!COMMENT) printf("\n%s is an IDENTIFIER", yytext); }

\".*\"                         { if (!COMMENT) printf("\n\t%s is a STRING", yytext); }

[0-9]+                         { if (!COMMENT) printf("\n%s is a NUMBER", yytext); }

"="                            { if (!COMMENT) printf("\n\t%s is an ASSIGNMENT OPERATOR", yytext); }
```



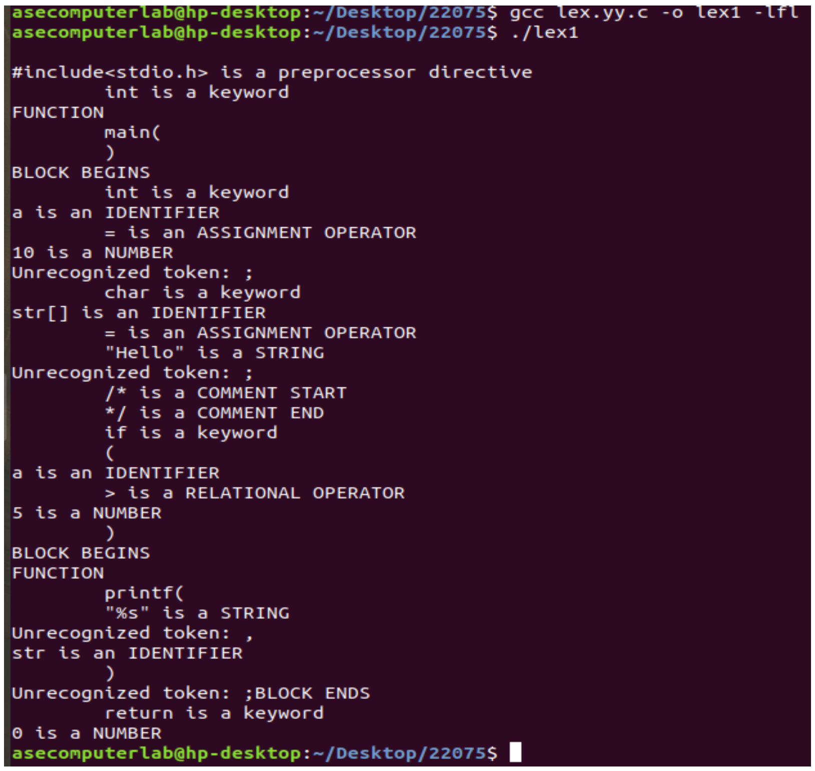
```
[0-9]+                         { if (!COMMENT) printf("\n%s is a NUMBER", yytext); }

"="                            { if (!COMMENT) printf("\n\t%s is an ASSIGNMENT OPERATOR", yytext); }

"<=" |
">=" |
"<" |
"==" |
">"                            { if (!COMMENT) printf("\n\t%s is a RELATIONAL OPERATOR", yytext); }

"(" |
")" |
":"                            { if (!COMMENT) printf("\n\t%s", yytext); }

[ \t\n]+                       { /* Skip whitespace */ }

.                              { if (!COMMENT) printf("\nUnrecognized token: %s", yytext); }
%%

int main(int argc, char **argv)
{
    FILE *file = fopen("var.c", "r");
    if (!file)
    {
        printf("Could not open the file\n");
        exit(1);
    }
    yyin = file;
    yylex();
    fclose(file);
    return 0;
}
```

var.c



```
%{
#include <stdio.h>
#include <stdlib.h>
int COMMENT = 0;
%}

identifier [a-zA-Z_][a-zA-Z0-9_]*
%%
#.*                              { printf("\n%s is a preprocessor directive", yytext); }

"int" |
"float" |
"char" |
"double" |
"while" |
"for" |
"struct" |
"typedef" |
"do" |
"if" |
"break" |
"continue" ||
"void" |
"switch" |
"return" |
"else" |
"goto"                          { printf("\n\t%s is a keyword", yytext); }

"/*"                            { COMMENT = 1; printf("\n\t%s is a COMMENT START", yytext); }
"*/"                            { COMMENT = 0; printf("\n\t%s is a COMMENT END", yytext); }

{identifier}"("                 { if (!COMMENT) printf("\nFUNCTION \n\t%s", yytext); }

"{"                             { if (!COMMENT) printf("\nBLOCK BEGINS"); }
"}"                             { if (!COMMENT) printf("BLOCK ENDS "); }

{identifier}(\[[0-9]*\])?       { if (!COMMENT) printf("\n%s is an IDENTIFIER", yytext); }

\".*\"                          { if (!COMMENT) printf("\n\t%s is a STRING", yytext); }

[0-9]+                          { if (!COMMENT) printf("\n%s is a NUMBER", yytext); }

"="                             { if (!COMMENT) printf("\n\t%s is an ASSIGNMENT OPERATOR", yytext); }
```

Output:



```
asecomputerlab@hp-desktop:~/Desktop/22075$ gcc lex.yy.c -o lex1 -lfl
asecomputerlab@hp-desktop:~/Desktop/22075$ ./lex1

#include<stdio.h> is a preprocessor directive
        int is a keyword
FUNCTION
        main(
        )
BLOCK BEGINS
        int is a keyword
a is an IDENTIFIER
        = is an ASSIGNMENT OPERATOR
10 is a NUMBER
Unrecognized token: ;
        char is a keyword
str[] is an IDENTIFIER
        = is an ASSIGNMENT OPERATOR
        "Hello" is a STRING
Unrecognized token: ;
        /* is a COMMENT START
        */ is a COMMENT END
        if is a keyword
        (
a is an IDENTIFIER
        > is a RELATIONAL OPERATOR
5 is a NUMBER
        )
BLOCK BEGINS
FUNCTION
        printf(
        "%s" is a STRING
Unrecognized token: ,
str is an IDENTIFIER
        )
Unrecognized token: ;BLOCK ENDS
        return is a keyword
0 is a NUMBER
asecomputerlab@hp-desktop:~/Desktop/22075$
```