MANOJ RL 220701161

EXP NO: 03 DATE:

DEVELOP A LEXICAL ANALYSER TO RECOGNIZE A FEW PATTERNS IN C. (EX.IDENTIFIERS, CONSTANTS, COMMENTS, AND OPERATORS, ETC.) USING LEX TOOL.

AIM:

To develop a Lexical Analyzer using the LEX tool that recognizes different tokens in a given C program snippet, including Identifier, Constants, Comments, Operators, Keywords, Special Symbols.

ALGORITHM:

- Start
- Define token patterns in LEX for:

- Keywords (e.g., int, float, if, else)□
- Identifiers (variable/function names)□
- Constants (integer and floating-point numbers)□
- Operators (+, -, =, ==, !=, *, /)□
- Comments (// single-line, /* multi-line */)□
- Special Symbols ({, }, (,), ;, ,)□

- Read input source code.
- Match the code tokens using LEX rules.
- Print each recognized token with its type.
- End

PROGRAM:□

□

```
%{
   #include <stdio.h>
   #include <stdlib.h>
   #include <string.h>
   #include <stddef.h>
%}

%%
"int"|"float"|"if"|"else"    { printf("KEYWORD: %s\n", yytext); }
```

```
[a-zA-Z_][a-zA-Z0-9_]*       { printf("IDENTIFIER: %s\n", yytext); }
[0-9]+                       { printf("INTEGER CONSTANT: %s\n", yytext); }
[0-9]*\.[0-9]+               { printf("FLOAT CONSTANT: %s\n", yytext); }
\/\/.*                       { printf("SINGLE-LINE COMMENT\n"); }
\/\*([^*]|\*+[^/*])*\*\/     { printf("MULTI-LINE COMMENT\n"); }
\+|\-|\*|\/|\%|=|==|!=       { printf("OPERATOR: %s\n", yytext); }
[\{\}\(\)\;\,]               { printf("SPECIAL SYMBOL: %s\n", yytext); }
[ \t\n]                      {  }

%%
int yywrap() {
   return 1;
}

int main() {
   yylex();
return 0;
}
```

MANOJ RL 220701161

OUTPUT:

```
lex lexer.l  cc lex.yy.c -o lexer
./a.out  Sample Input  int main()
{  int a = 10;  float b = 20.5;  /*
This is a multi-line comment */
if (a > b) {  a = a + b;  }  return
0;  }
```

```
Keyword: int
Identifier: main
Special Symbol: (
Special Symbol: )
Special Symbol: {
Keyword: int
Identifier: a
Operator: =
Constant: 10
Special Symbol: ;
Keyword: float
Identifier: b
Operator: =
Constant: 20.5
Special Symbol: ;
Multi-line Comment: /* This is a multi-line comment */
Keyword: if
Special Symbol: (
Identifier: a
Operator: >
Identifier: b
Special Symbol: )
Special Symbol: {
Identifier: a
```

MANOJ RL 220701161

MANOJ RL 220701161

Implementation	
Output/Signature	

RESULT:

Thus the above program reads a C code snippet, tokenizes it using LEX rules, recognizes and categorizes keywords, identifiers, constants, operators, comments, and special symbols, and then displays each token along with its type.