Exp1 : LEXICAL ANALYZER USING C

```
1. Start
2. while ( ch != EOF ) do
    1. if ( ch = # ) then
        1. while ( ch != > ) read the string into str array
        2. print " header file "
    2. else if ( ch = + | - | * | / ) then
        1. print " operator "
    3. else if ( ch = % | : | ( | ) | { | } | [ | ] | ; | , | ? | @ | " ) then
        1. print " special symbol "
        2. if ( ch = " ) then
            1. while ( ch != " ) read the string into str array
            2. print " string "
    4. else if ( isdigit( ch ) ) then
        1. print " digit "
    5. else if ( isalpha( ch ) ) then
        1. while ( isalnum( ch ) read the string into str array
        2. flag = 0
        3. for i=0 to 32 do
            1. if ( str = keyword[i] ) then
                1. flag = 1
                2. break
        4. if ( flag = 1 ) then print " keyword "
        5. else print " identifier "
    6. end while
3. Stop
```

Lexical Analyzer code (C)

```
#include <stdio.h>
#include <ctype.h>
#include <string.h>

int main(){
   FILE *input = fopen("input.c", "r");
   FILE *output = fopen("output.txt", "w");
   char ch, str[20];
   int i, flag = 0;
   char keyword[32][32] = {"auto", "break", "case", "char", "const", "continue", "default", "do",
"double", "else", "enum", "extern", "float", "for", "goto", "if", "int", "long", "register", "return",
"short", "signed", "sizeof", "static", "struct", "switch", "typedef", "union", "unsigned", "void",
"volatile", "while"};
   fprintf(output, "token\tlexeme\n--------------\n");

   while((ch = fgetc(input)) != EOF){
      if(ch == '#'){
         i = 0;
         str[i] = ch;
         ch = fgetc(input);

         while(ch != '>'){
            i++;
```

```
                str[i] = ch;
                ch = fgetc(input);
            }
            str[i+1] = '>';
            fprintf(output, "%s\theader file\n", str);
            str[i+2] = '\0';
        }else if(ch == '+' || ch == '-' || ch == '*' || ch == '/' || ch == '=' || ch == '<' || ch == '>'){
            fprintf(output, "%c\toperator\n", ch);
        }else if (ch == '%' || ch == ':' || ch == '(' || ch == ')' || ch == '[' || ch == ']' || ch == '{' || ch == '}' ||
    ch == ';' || ch == ',' || ch == '?' || ch == '@' || ch == '"'){
            fprintf(output, "%c\tspecial symbol\n", ch);

            if(ch == '"'){
                i = 0;
                ch = fgetc(input);

                while(ch != '"'){
                    str[i] = ch;
                    i++;
                    ch = fgetc(input);
                }
                str[i] = '\0';
                fprintf(output, "%s\tstring\n", str);
            }
        }else if(isdigit(ch)){
            fprintf(output, "%c\tdigit\n", ch);
        }else if(isalpha(ch)){
            i = 0;
            str[i] = ch;
            ch = fgetc(input);

            while(isalnum(ch) && i < 19){
                i++;
                str[i] = ch;
                ch = fgetc(input);
            }
            str[i + 1] = '\0';
            flag = 0;

            for(i = 0; i < 32; i++){
                if(strcmp(str, keyword[i]) == 0){
                    flag = 1;
                    break;
                }
            }

            if(flag == 1){
                fprintf(output, "%s\tkeyword\n", str);
            }else{
                fprintf(output, "%s\tidentifier\n", str);
            }
        }
    }

            printf("output file created successfully\n");
    fclose(input);
    fclose(output);

    return 0;
}
```

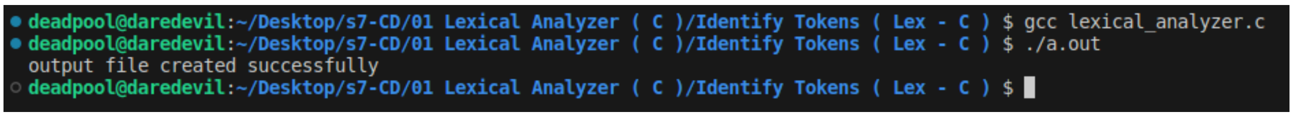
<u>input.c</u>

```
#include<stdio.h>
#include<stdlib.h>
int main(){
    int num1=5,num2=10;
    int sum=num1+num2;
    printf("sum = %d\n",sum);
    return 0;
}
```

<u>output.txt</u>

```
token   lexeme
--------------
#include<stdio.h>      header file
#include<stdlib.h>     header file
int     keyword
main    identifier
)       special symbol
{       special symbol
int     keyword
num1    identifier
5       digit
,       special symbol
num2    identifier
1       digit
0       digit
;       special symbol
int     keyword
sum     identifier
num1    identifier
num2    identifier
printf  identifier
"       special symbol
sum = %d\n    string
,       special symbol
sum     identifier
;       special symbol
return  keyword
0       digit
;       special symbol
}       special symbol
```

output



```
● deadpool@daredevil:~/Desktop/s7-CD/01 Lexical Analyzer ( C )/Identify Tokens ( Lex - C ) $ gcc lexical_analyzer.c
● deadpool@daredevil:~/Desktop/s7-CD/01 Lexical Analyzer ( C )/Identify Tokens ( Lex - C ) $ ./a.out
  output file created successfully
○ deadpool@daredevil:~/Desktop/s7-CD/01 Lexical Analyzer ( C )/Identify Tokens ( Lex - C ) $ 
```