COMPILER DESIGN (CST -309)

LEXICAL ANALYSER

Submitted to – Dr. Dinesh Goplani

Submitted by

- Ashutosh Soni (2018ucp1505)
- Shubham Yadav (2018ucp1421)

Aim: Implement Lexical Analyser using LEX program and analyse the output.

Introduction:

Lexical analyser is also known as ‘Scanner’. Its main job is to convert given input source file into stream of tokens.

Programmatically, lex is a tool for automatically generating a scanner starting from lex specification.

Lexical Program Consists of three parts

1) Declarations
2) Transition Rules
3) Auxiliary procedures

Below is the code for its implementation

Implementation of lexical Analyser using LEX program.

```
/* Program for Lexical Analyser */
/* Declaration Part */

%{

        int comment = 0;
        int count_of_comment = 0;

%}
identifier [a-zA-Z_][a-zA-Z0-9]*
/* Transition Rules */

%%
#.*  { printf("\n%s is a PREPROCESSOR DIRECTIVE \n",yytext); }

auto |
break |
case |
char |
continue |
do |
default |
const |
double |
else |
enum |
extern |
for |
if |
goto |
float |
int |
long |
register |
return |
signed |
static |
sizeof |
short |
struct |
typedef |
union |
void |
while |
volatile |
unsigned { printf(" %s is a KEYWORD \n",yytext); }
```

```
"/*" { comment = 1; }
"*/" { comment = 0;
count_of_comment++; }

{identifier}\( {
        if(!comment)
                printf("\nFUNCTION   %s \n",yytext);
        }
        \{ {
                if(!comment)
                        printf("\t BLOCK BEGINS \n");
        }
        \} {
                if(!comment)
                        printf("\t BLOCK ENDS \n");
        }
        {identifier}(\[[0-9]*\])? {
                if(!comment)
                        printf("\t %s is a IDENTIFIER \n",yytext);
        }
        \".*\" {
                if(!comment)
                printf("\t %s is a STRING \n",yytext);
        }
        [0-9]+ {
                if(!comment)
                        printf("\t %s is a NUMBER \n",yytext);
        }
\)(\;)? {
        if(!comment)
                printf("\n");
        ECHO;
}
\(
        ECHO;
        = {
                if(!comment)
                        printf("\t %s is a ASSIGNMENT OPERATOR \n",yytext);
        }
        \ <= |
        \ >= |
        \ <  |
        \ == |
        \ > {
                if(!comment)
                        printf("\t %s is a RELATIONAL OPERATOR \n",yytext);
        }
```

```
%%

int main(int argc,char *argv[]){
        if(argc!=2){
                printf("Please give input file \n");
                printf("terminating...\n");
                exit(0);
        }

        FILE *file = NULL ;
        file = fopen(argv[1],"r");
        if(file == NULL){
                printf("Error in opening file \n");
                printf("Try again ... \n terminating ....\n");
                exit(0);
        }

        yyin = file;

        yylex();

        printf(" \n Total number of comments in this file is %d \n",count_of_comment);
        return 0;

}

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

Procedure to run Lex Program:

Step 1: Install Flex in Ubuntu

```
sudo apt-get update
sudo apt-get install flex
```

Step 2: Run command on lexical complier

```
lex filename.l
```

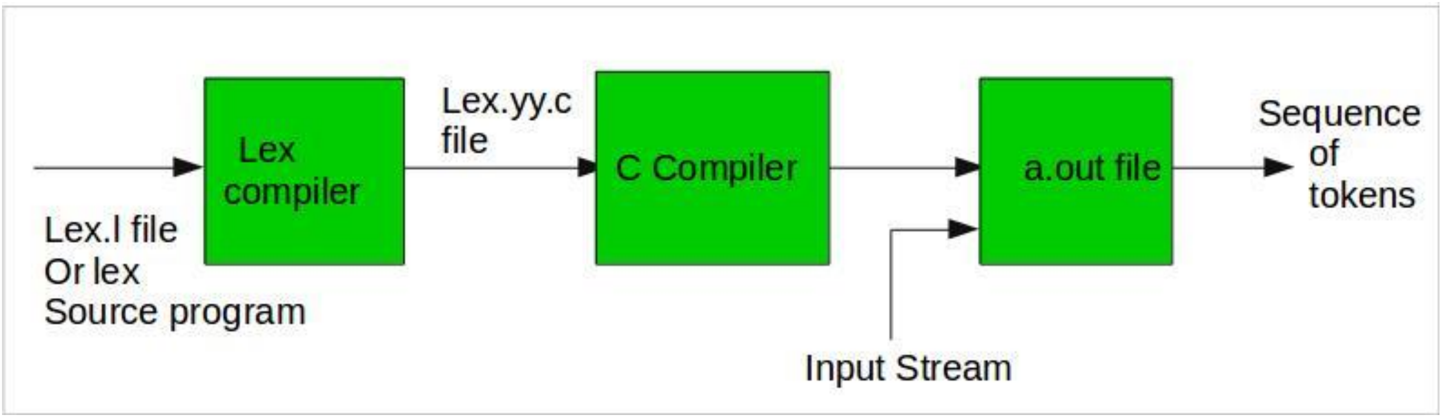
This will change the .l file to .yy.c file.

Step 3: Run command on C compiler

```
gcc lex.yy.c
```

As output of this we get a.out file

Step 4: ./a.out inputfile.cpp

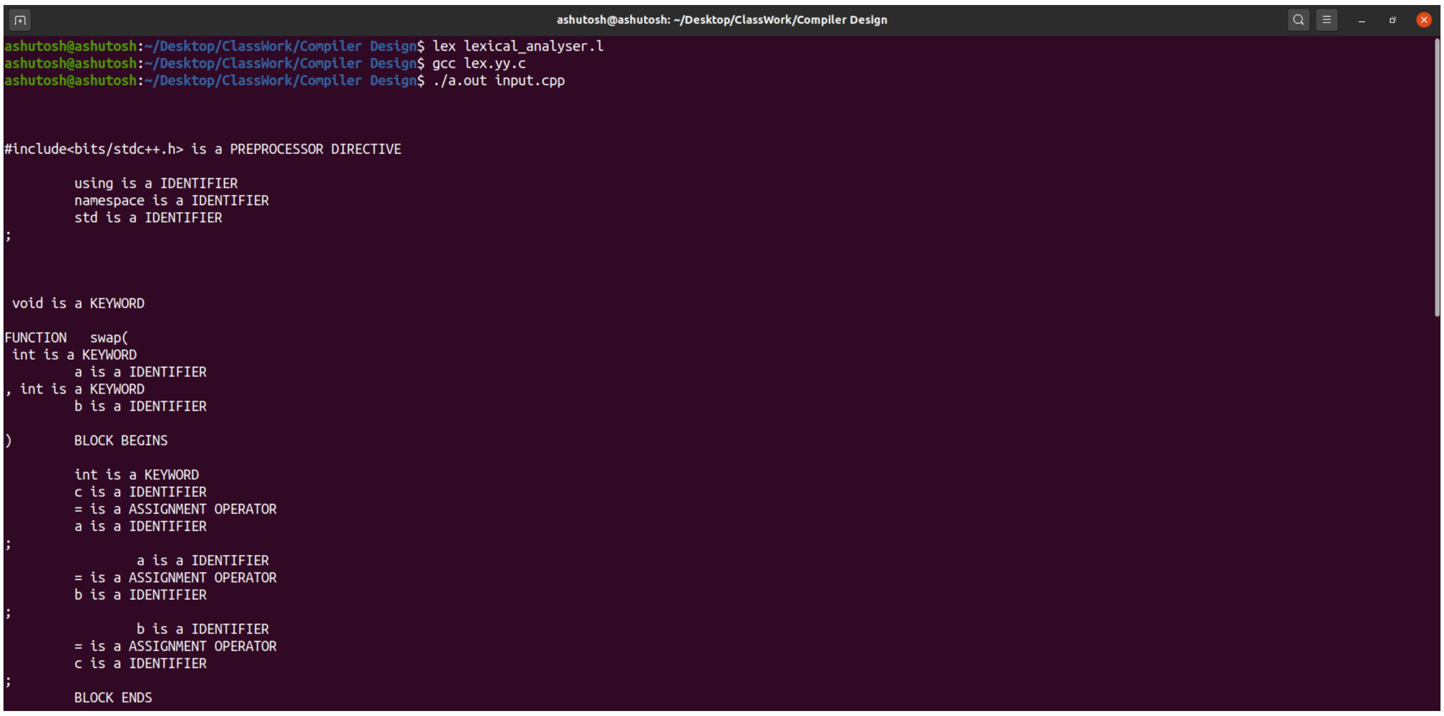


For our Analysis we use this input file:

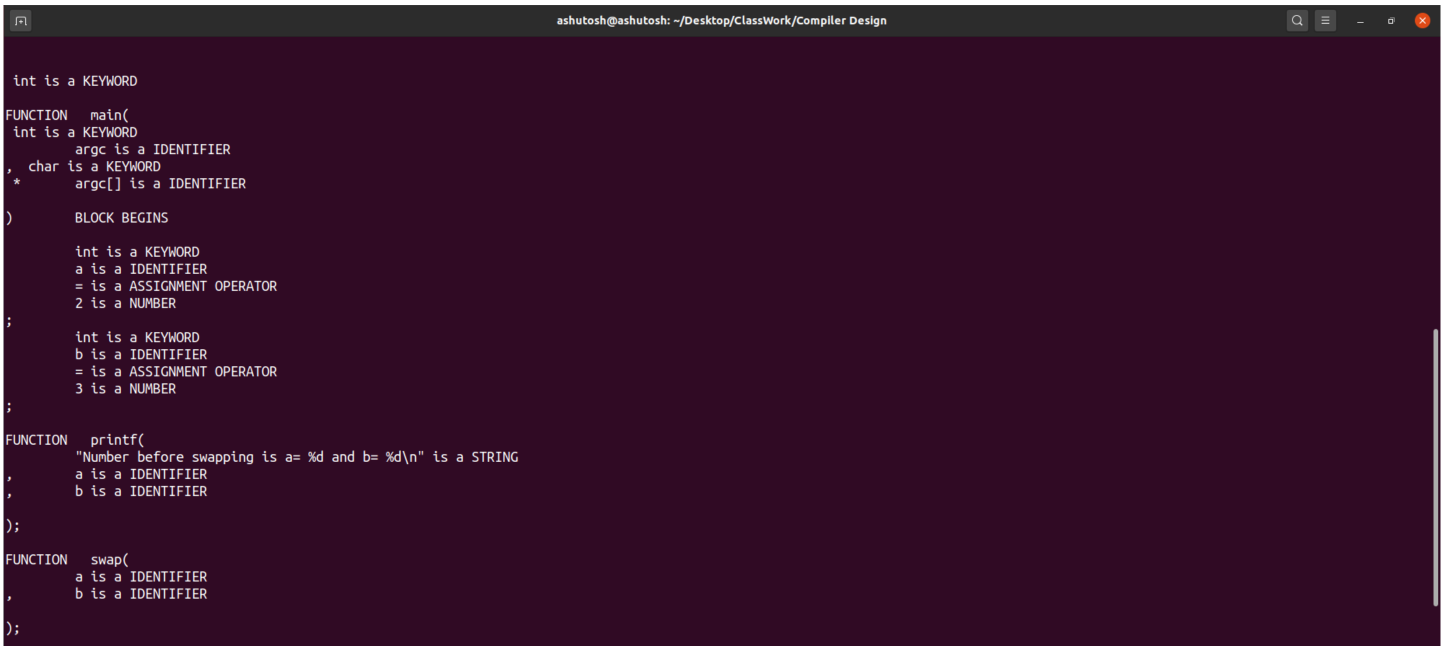
Input File 1:

```
/* Including Header file */
#include<bits/stdc++.h>
using namespace std;
/* Swapping numbers */
void swap(int a,int b){
        int c=a;
        a=b;
        b=c;
}
/* Main Function */
int main(int argc, char *argc[]){
        int a=2;
        int b=3;
        printf("Number before swapping is a= %d and b= %d\n",a,b);
        swap(a,b);
        printf("Number after swapping is a= %d and b= %d\n",a,b );
        printf("Sum of these number is %d\n",a+b );
        return 0;
}
```

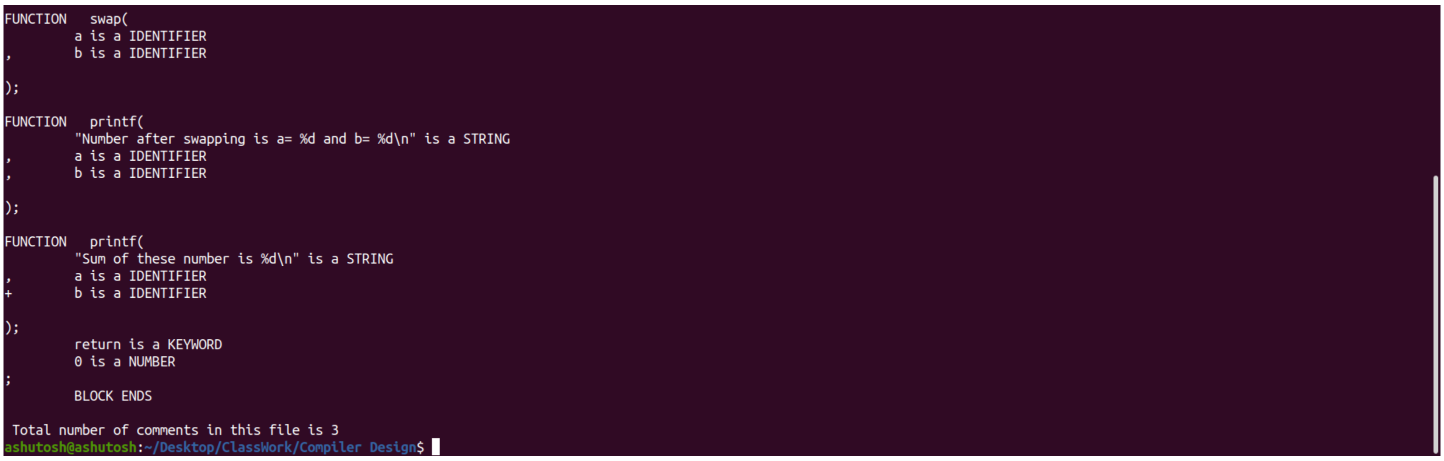
Result on Terminal:



```
ashutosh@ashutosh:~/Desktop/ClassWork/Compiler Design$ lex lexical_analyser.l
ashutosh@ashutosh:~/Desktop/ClassWork/Compiler Design$ gcc lex.yy.c
ashutosh@ashutosh:~/Desktop/ClassWork/Compiler Design$ ./a.out input.cpp



#include<bits/stdc++.h> is a PREPROCESSOR DIRECTIVE

         using is a IDENTIFIER
         namespace is a IDENTIFIER
         std is a IDENTIFIER
;



 void is a KEYWORD

FUNCTION   swap(
 int is a KEYWORD
         a is a IDENTIFIER
, int is a KEYWORD
         b is a IDENTIFIER

)        BLOCK BEGINS

         int is a KEYWORD
         c is a IDENTIFIER
         = is a ASSIGNMENT OPERATOR
         a is a IDENTIFIER
;
                 a is a IDENTIFIER
         = is a ASSIGNMENT OPERATOR
         b is a IDENTIFIER
;
                 b is a IDENTIFIER
         = is a ASSIGNMENT OPERATOR
         c is a IDENTIFIER
;
         BLOCK ENDS
```



```

 int is a KEYWORD

FUNCTION   main(
 int is a KEYWORD
         argc is a IDENTIFIER
,  char is a KEYWORD
 *       argc[] is a IDENTIFIER

)        BLOCK BEGINS

         int is a KEYWORD
         a is a IDENTIFIER
         = is a ASSIGNMENT OPERATOR
         2 is a NUMBER
;
         int is a KEYWORD
         b is a IDENTIFIER
         = is a ASSIGNMENT OPERATOR
         3 is a NUMBER
;

FUNCTION   printf(
         "Number before swapping is a= %d and b= %d\n" is a STRING
,        a is a IDENTIFIER
,        b is a IDENTIFIER

);

FUNCTION   swap(
         a is a IDENTIFIER
,        b is a IDENTIFIER

);
```

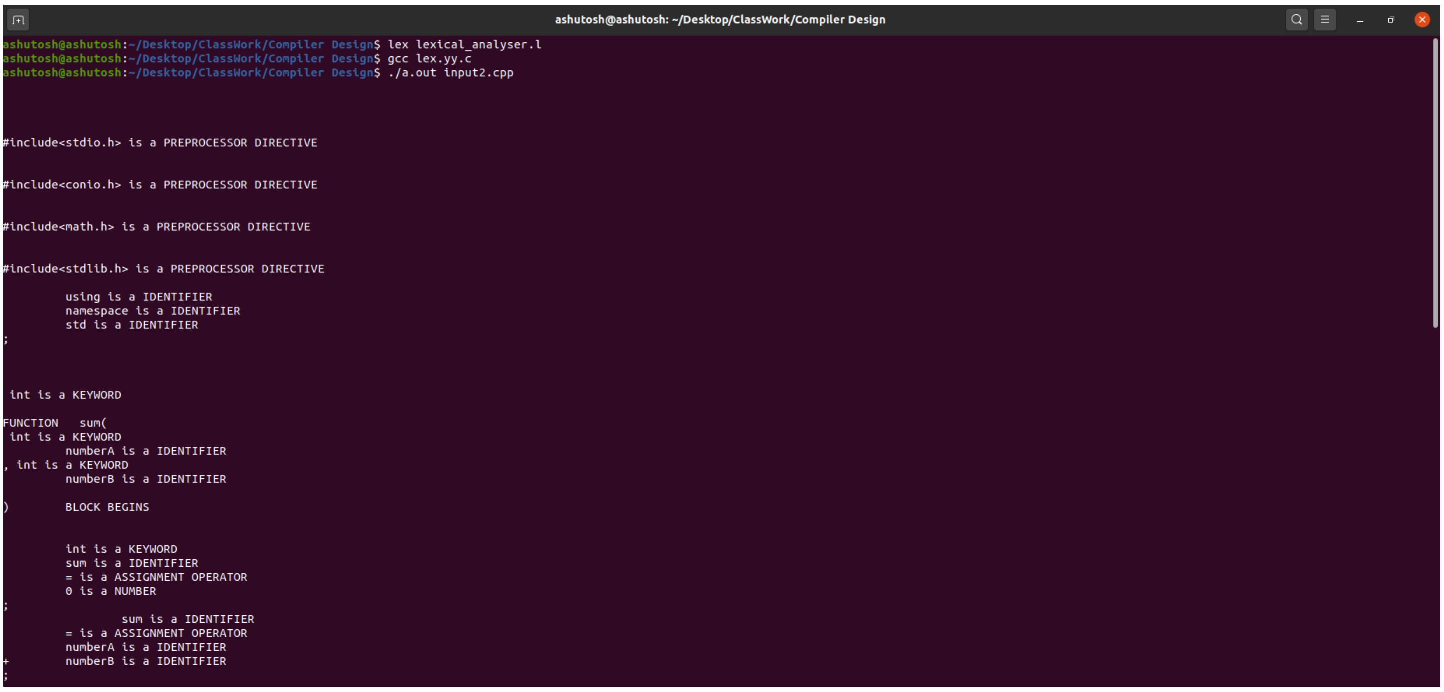


```
FUNCTION   swap(
         a is a IDENTIFIER
,        b is a IDENTIFIER

);

FUNCTION   printf(
         "Number after swapping is a= %d and b= %d\n" is a STRING
,        a is a IDENTIFIER
,        b is a IDENTIFIER

);

FUNCTION   printf(
         "Sum of these number is %d\n" is a STRING
,        a is a IDENTIFIER
+        b is a IDENTIFIER

);
         return is a KEYWORD
         0 is a NUMBER
;
         BLOCK ENDS

 Total number of comments in this file is 3
ashutosh@ashutosh:~/Desktop/ClassWork/Compiler Design$
```

Input File 2:

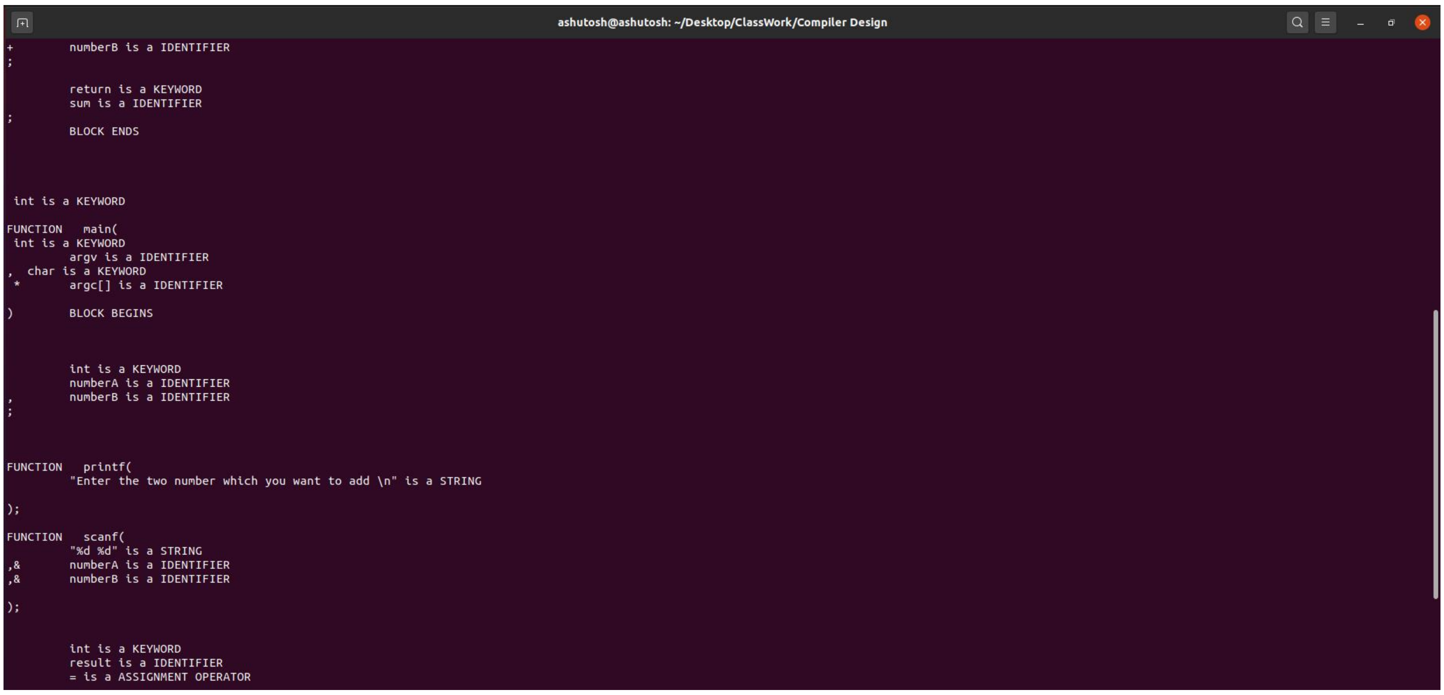
```
/* Including Header File */

#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<stdlib.h>
using namespace std;

/* Sum Function*/

int sum(int numberA,int numberB){
        /* Declaration of sum function */
        int sum=0;
        sum=numberA+numberB;
        /* Returning the value */
        return sum;
}

/* Main Function*/

int main(int argv, char *argc[]){
        /* Declaration on two numbers */

        int numberA,numberB;

        /* Taking input of two numbers */
        printf("Enter the two number which you want to add \n");
        scanf("%d %d",&numberA,&numberB);

        /* Calling the function */
        int result = sum(numberA,numberB);

        /* Output the result */
        printf("Sum of these two number is %d\n",result );

        return 0;
}
```

Result on Terminal:



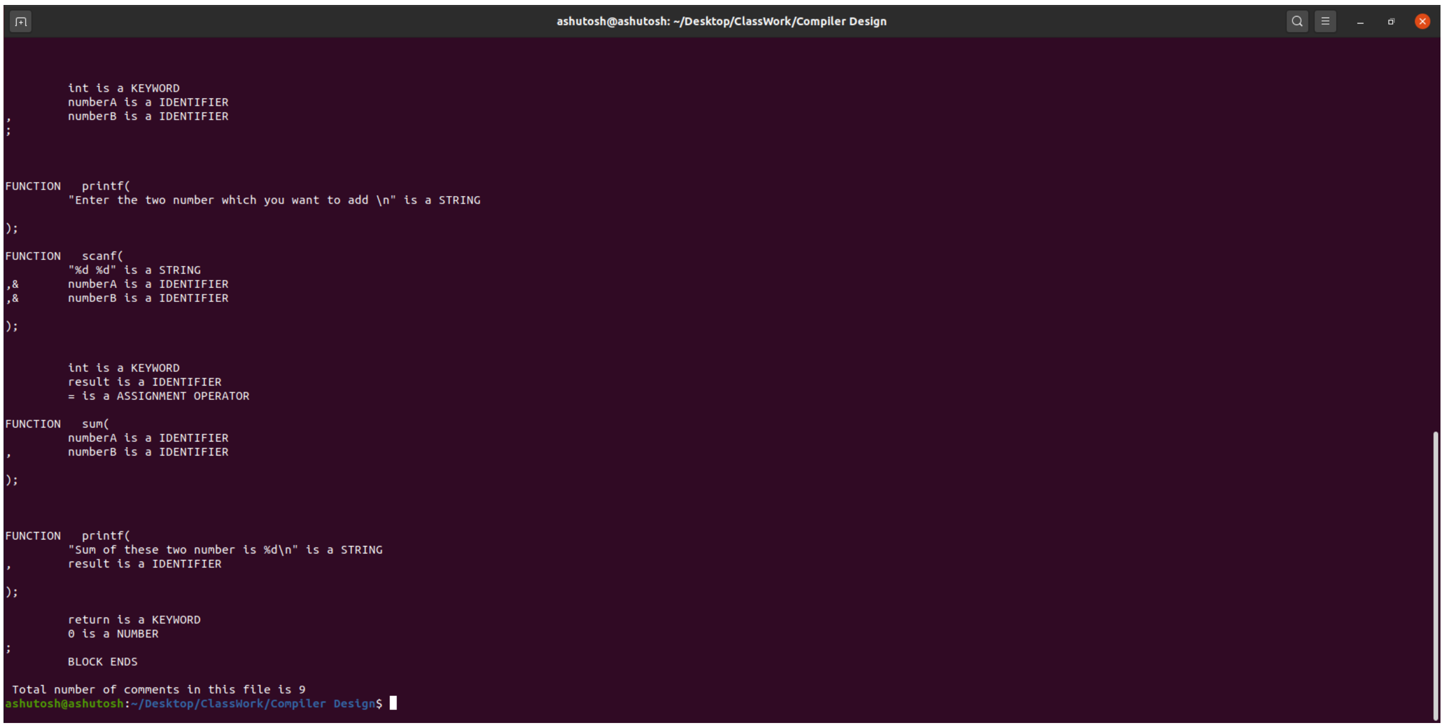
ashutosh@ashutosh: ~/Desktop/ClassWork/Compiler Design

```
ashutosh@ashutosh:~/Desktop/ClassWork/Compiler Design$ lex lexical_analyser.l
ashutosh@ashutosh:~/Desktop/ClassWork/Compiler Design$ gcc lex.yy.c
ashutosh@ashutosh:~/Desktop/ClassWork/Compiler Design$ ./a.out input2.cpp




#include<stdio.h> is a PREPROCESSOR DIRECTIVE


#include<conio.h> is a PREPROCESSOR DIRECTIVE


#include<math.h> is a PREPROCESSOR DIRECTIVE


#include<stdlib.h> is a PREPROCESSOR DIRECTIVE

         using is a IDENTIFIER
         namespace is a IDENTIFIER
         std is a IDENTIFIER
;



 int is a KEYWORD

FUNCTION   sum(
 int is a KEYWORD
         numberA is a IDENTIFIER
, int is a KEYWORD
         numberB is a IDENTIFIER

)        BLOCK BEGINS


         int is a KEYWORD
         sum is a IDENTIFIER
         = is a ASSIGNMENT OPERATOR
         0 is a NUMBER
;
                 sum is a IDENTIFIER
         = is a ASSIGNMENT OPERATOR
         numberA is a IDENTIFIER
+        numberB is a IDENTIFIER
;
```



ashutosh@ashutosh: ~/Desktop/ClassWork/Compiler Design

```
+        numberB is a IDENTIFIER
;


         return is a KEYWORD
         sum is a IDENTIFIER
;
         BLOCK ENDS




 int is a KEYWORD

FUNCTION   main(
 int is a KEYWORD
         argv is a IDENTIFIER
,  char is a KEYWORD
 *       argc[] is a IDENTIFIER

)        BLOCK BEGINS



         int is a KEYWORD
         numberA is a IDENTIFIER
,        numberB is a IDENTIFIER
;




FUNCTION   printf(
         "Enter the two number which you want to add \n" is a STRING

);

FUNCTION   scanf(
         "%d %d" is a STRING
,&       numberA is a IDENTIFIER
,&       numberB is a IDENTIFIER

);


         int is a KEYWORD
         result is a IDENTIFIER
         = is a ASSIGNMENT OPERATOR
```



```
         int is a KEYWORD
         numberA is a IDENTIFIER
,        numberB is a IDENTIFIER
;



FUNCTION   printf(
         "Enter the two number which you want to add \n" is a STRING

);

FUNCTION   scanf(
         "%d %d" is a STRING
,&       numberA is a IDENTIFIER
,&       numberB is a IDENTIFIER

);


         int is a KEYWORD
         result is a IDENTIFIER
         = is a ASSIGNMENT OPERATOR

FUNCTION   sum(
         numberA is a IDENTIFIER
,        numberB is a IDENTIFIER

);



FUNCTION   printf(
         "Sum of these two number is %d\n" is a STRING
,        result is a IDENTIFIER

);

         return is a KEYWORD
         0 is a NUMBER
;
         BLOCK ENDS

 Total number of comments in this file is 9
ashutosh@ashutosh:~/Desktop/ClassWork/Compiler Design$
```