

CD Project – Report 1

Team members:

15CO239 – R. RuhiTaj

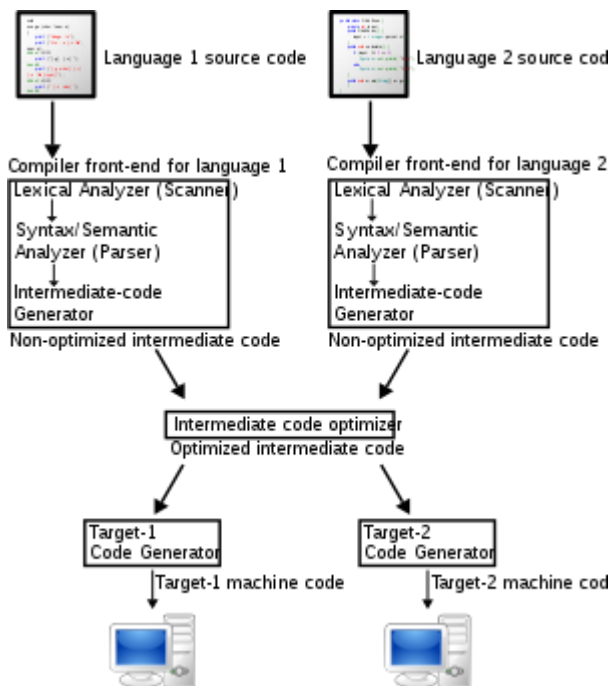
15CO253—Y. M. Greeshma

Language used: C language.

-What is a COMPILER?

A compiler is a **COMPUTER SOFTWARE PROGRAM**.

They are a type of TRANSLATOR that supports digital devices, primarily computers.



A diagram of the operation of a typical multi language, multi target compiler

-What are COMPILERS used for?

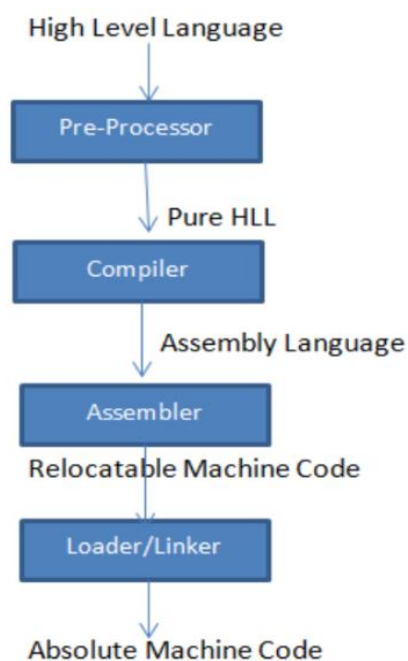
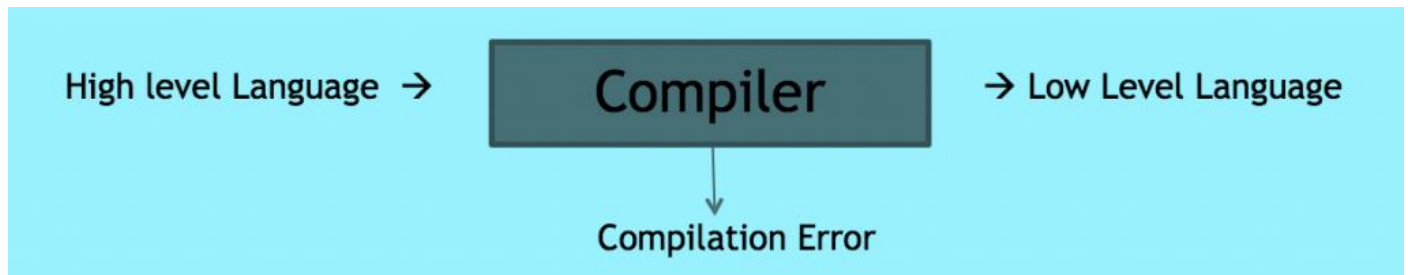
A Compiler transforms code written in one programming language (**Source Code**) to another language (**Target Code**).

The term - Compiler is primarily used for programs that translate source code from a high-level programming language to a lower level language (e.g., assembly language, object code, or machine code) to create an executable program.

-What does a COMPILER do?

A compiler performs the following operations: pre-processing, lexical analysis , parsing, semantic analysis (syntax-directed translation) and conversion of input programs to an intermediate representation, code optimization and code generation.

-General Description of a COMPILER:



We know a computer is a logical assembly of Software and Hardware. The hardware knows a language that is hard for us to grasp, consequently we tend to write programs in high-level language that is much less complicated for us to comprehend and maintain in thoughts. Now these programs go through a series of transformation so that they can readily be used machines. This is where language procedure systems come handy.

Here are three kinds of compilers you should learn to identify:

- Source language = Host language ⇒ **Self compiler**
- Target language = Host language ⇒ **Self-resident compiler**
- Target language ≠ Host language ⇒ **Cross compiler**

Cross Compiler that runs on a machine 'A' and produces a code for another machine 'B'. It is capable of creating code for a platform other than the one on which the compiler is running. It is characterized by three languages: the source language(S), the target language(T) and the implementation language(I).

- This is represented by a T-diagram as:



- In textual form this can be represented as

$S_I T$

Source-to-source Compiler or Transcompiler or transpiler is a compiler that translates source code written in one programming language into source code of another programming language.

-Phases of a COMPILER:

Lexical Analysis:

The first phase of scanner works as a text scanner. This phase scans the source code as a stream of characters and converts it into meaningful lexemes.

It converts the input program into a sequence of **Tokens**.

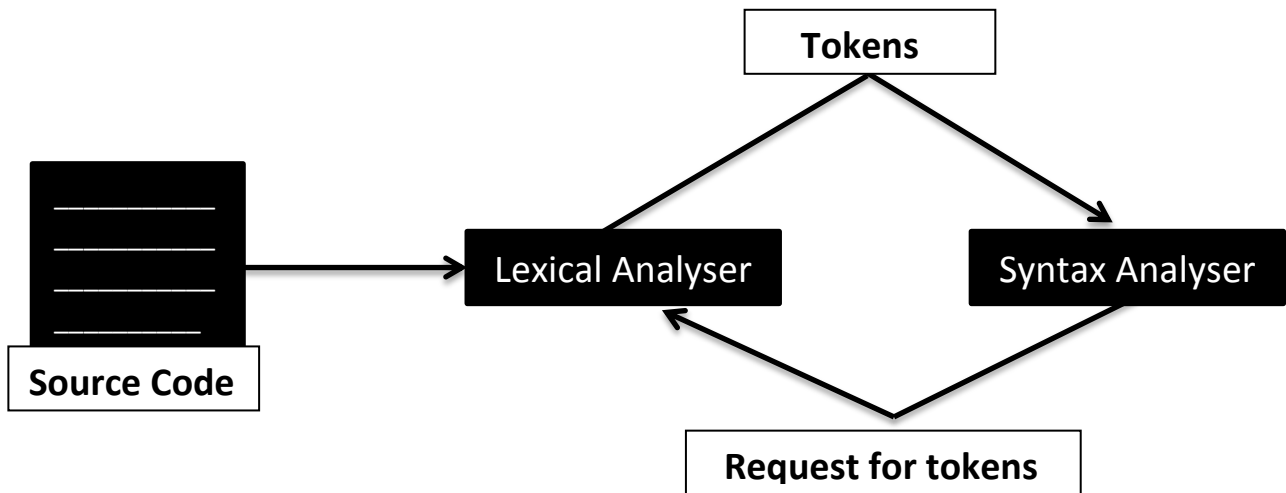
Example of tokens:

- Type token (id, number, real, . . .)
- Punctuation tokens (IF, void, return, . . .)
- Alphabetic tokens (keywords)

Function of Lexical Analyser:

1. Dividing the program into tokens (*Tokenization*).
2. Remove white space characters.
3. Remove comments.
4. Provide error message by providing row number and column number.

The lexical analyser identifies the error with the help of automation machine and the grammar of the given language on which it is based like C, C++.



Need of Lexical Analyser:

1) *Simplicity of design of compiler*

The removal of white spaces and comments enables the syntax analyzer for efficient syntactic constructs.

2) *Compiler efficiency is improved*

Specialized buffering techniques for reading characters speed up the compiler process.

3) *Compiler portability is enhanced.*

Issues in Lexical Analysis:

Lexical analysis is the process of producing tokens from the source program. It has the following issues:

- Look ahead
- Ambiguities

Syntax Analysis:

Syntax Analyser takes the token produced by lexical analysis as input and generates a parse tree.

Semantic Analysis:

Semantic analysis checks whether the parse tree constructed follows the rules of language.

Intermediate Code Generator:

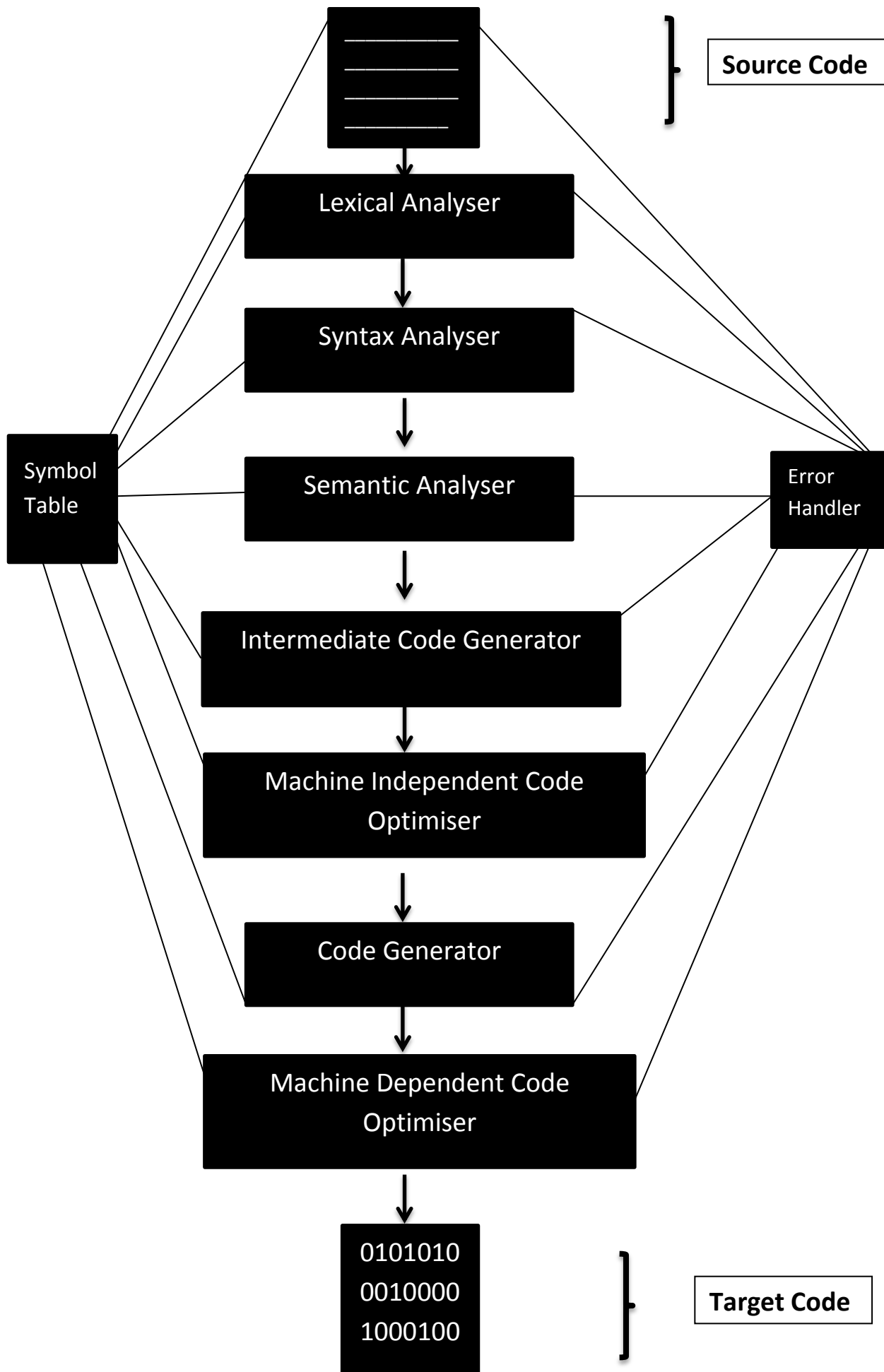
The compiler generates an intermediate code of the source code for the target machine.

Machine Independent Code Optimiser:

Removal of unnecessary code lines, and arranges the sequence of statements in order to speed up.

Code Generator:

Code generator takes the optimized representation of the intermediate code and maps it to the target machine language.



Lex Program:

```
%{
    # include <stdio.h>
    # include <stdlib.h>
    # include "help.h"
    int headFiles=0;
    int keyWords=0;
    int line=0;
}%

onlyDigits[0-9]
dot[.]
onlyAlphabets[a-zA-Z]
qoutes["]
space[ ]
underScore[_]
openArrow[<]
closeArrow[>]
headerNames[stdio]
openBraces[(]
closeBraces[)]
percentage[%]
comma[,]
colon[:]
end[;]
symbol[/]
other[\\]
sym[*]
flowerOpen[{]
flowerClose[}]
subtract[-]
addi[+]

%%
{onlyDigits}+({underScore}|{onlyAlphabets})+ {printf("\n\nImproper Identifier: %s\n\n", yytext);}

({addi}|{subtract})?({onlyDigits}+|{onlyDigits}*{dot}{onlyDigits}+) {
the_medium_to_display(yytext, 1); }

{qoutes}({onlyDigits}|{onlyAlphabets}|{underScore})+{qoutes} {the_medium_to_display(yytext, 10);}

(0x{onlyDigits}*{onlyAlphabets}*|{onlyDigits}*{onlyAlphabets}*{subtract}({onlyAlphabets}|{onlyDigits})*|{onlyDigits}*{onlyAlphabets}*{dot}{onlyDigits}+{onlyAlphabets})
the_medium_to_display(yytext, 2);

({flowerClose}|{flowerOpen}|{end}|{comma}) {the_medium_to_display(yytext,3);}
```

```

"#{space}*include"{space}*{openArrow}{onlyAlphabets}+{dot}h{closeArrow}
{the_medium_to_display(yytext, 4); }

"#{space}*include"{space}*({openArrow}|{onlyAlphabets}|{dot})|h|{closeArrow})* {
printf("\nImproper Header File: %s\n", yytext);}

("int"|"void")?{space}*main{space}*{openBraces}({space}?|"void"|")"{closeBraces} { printf("\nThis
is the main function: %s\n", yytext); }

main {printf("Improper main, please correct it!\n");}

"printf"{openBraces}{qoutes}({space}|{onlyAlphabets}|{onlyDigits}|{percentage}|{colon}|\\)*{qoute
s}({space}*{comma}{space}*({onlyDigits}*{onlyAlphabets}*){space}*)?{closeBraces}{end}
{the_medium_to_display(yytext, 5);}

auto|double|int|struct|break|else|long|switch|case|enum|register|typedef|char|extern|return|union
|const|float|short|unsigned|continue|for|signed|void|default|goto|sizeof|volatile|do|if|static|wh
ile {the_medium_to_display(yytext, 6);}

({onlyAlphabets}|{underScore})({onlyAlphabets}|{underScore}|{onlyDigits})* {
the_medium_to_display(yytext, 7);}

"+"|"-"|"%"|"="|"!="|"!"|"<"|">"|"^"|"~" { the_medium_to_display(yytext, 8);}

"%s"|" %c"|" %d"|" %e"|" %E"|" %f"|" %g"|" %G"|" %hi"|" %hu"|" %i"|" %l"|" %ld"|" %lld"|" %lf"|" %lF"|" %lu"|" %ll
i"|" %o"|" %p"|" %s"|" %u"|" %x"|" %X"|" %n" { the_medium_to_display(yytext, 9);}

"//"{({onlyAlphabets}|{onlyDigits}|{space}|{underScore}|{end})* {printf("It is a Single Line
Comment.\n");}

"/*"("\n"|{space}|{onlyAlphabets}|{onlyDigits}|{underScore}|{end})|"/"*)**"/" {printf("It is a
Multi Line Comment.\n");}

"printf"({onlyAlphabets}|{onlyDigits}|{openBraces}|{space}|{qoutes}|{percentage}|{colon}|{comma}|
{closeBraces}|{end})|\\)* {printf("\n\nThis statement is a Lexical Error: %s\n\n",yytext);}

("*/") {printf("\n\nThis is a lexical error, with improper commenting\n\n");}

"/" {printf("\n\nIllegal stray found in the program - /\n\n");}
{qoutes}({onlyDigits}|{onlyAlphabets}|{underScore})* {printf("\n\nThis is a Lexical Error-
Improper qoutes: %s\n\n", yytext);}

%%

int yywrap(){

    return 1;
}

void displayNames(int no){
}

```

```

int main(){

    yyin=fopen("testCase.c","r");
    yylex();
    int k=1;
    fclose(yyin);
    while(k<10)
    {displayNames(k);
      k=k+1;
    }

    printf("\n\n\n\n\nThe total no of real numbers: %d\n", count[1]);
    printf("\n\nThe total no of string literals: %d\n", count[2]);
    printf("\n\nThe total no of Separators: %d\n", count[3]);
    printf("\n\nThe total no of Header Files: %d\n", count[4]);
    printf("\n\nThe total no of printf: %d\n", count[5]);
    printf("\n\nThe total no of key words: %d\n", count[6]);
    printf("\n\nThe total no of identifiers: %d\n", count[7]);
    printf("\n\nThe total no of format specifiers: %d\n\n\n", count[8]);
    // printf("\n\nTotal no of Lines in program is : %d\n\n\n\n", line);

    theTable();
    return 0;
}

```

Help.h

```

char*
everything[11][50];

    int i,count[10] = {0};
    void namethe(int no){
    switch(no){
    case 1: printf("\n|\n|  Real_Number      ");
            break;
    case 2: printf("\n|\n|  String_Literal  ");
            break;
    case 3: printf("\n|\n|  Separator      ");
            break;
    case 4: printf("\n|\n|  HeaderFile     ");
            break;
    case 5: printf("\n|\n|  printf        ");
            break;
    case 6: printf("\n|\n|  Key_Word       ");
            break;
    case 7: printf("\n|\n|  Identifier    ");
            break;
    case 8: printf("\n|\n|  Operator      ");
            break;

```



```

case 9: printf("\n|\n|  Format_Specifier  ");
        break;
case 10: printf("\n|\n|  Constants          ");
        break;
case 11: printf("\n|\n|  Comment_Lines      ");
        break;
default: break;
}
}
void the_medium_to_display(char* word, int no){
if(no==1){
the_medium_to_display(word,10);
}
int y = strlen(word);

everything[no][count[no]] = (char*)malloc(y*sizeof(char));
int g;
    for(g=0;g<y;g++){

        everything[no][count[no]][g] = word[g];
    }
count[no] = count[no]+1;
//namethe(no);
//printf(":\t%s\n", everything[no][count[no]-1]);
}
void theTable(){
    int m,n;
    FILE * fp;
    int h;
    /* open the file for writing*/
    fp = fopen ("/home/ubuntu/Desktop/correctTestCase/test3/output.txt","w");
    for(h=0;h<20;h++){
printf("___");
    }
    for(m=1;m<11;m++){
if(count[m]>0){

namethe(m);
for(n=0;n<count[m];n++)
    {
        int p = strlen(everything[m][n]);
        if(n==0 && n==count[m]-1){
printf("  | %s\n",everything[m][n]);
        }
        else if(n==0)
        {printf("  | %s\n",everything[m][n]);}
        else if(n== count[m]-1)
        {printf("|\t\t\t\t| %s\n|",everything[m][n]);
        }
        else{
printf("|\t\t\t\t| %s\n",everything[m][n]);

```

```

        }
    }

    if(count[m]==1){
        printf("|");
    }
    for(h=0;h<20;h++){
        printf("___");
    }
    }
    }

printf("\n\n\n");
}

// 1 Real Numbers
//2 String Literal
// 3 Separator
//4 Header Files
// 5 printf
// 6 keyWord
// 7 identifier
// 8 operator
// 9 format specifier
//     struct node{
//     char* text;
//     int type;
//     struct node* link;
//     struct node* down;
//     };
//     struct node* root = (struct node*) malloc(sizeof(struct node));
//     root->link = NULL;
//     root->down = NULL;
// void the_medium_to_display(char* the_text, int category, struct node* head){
//     struct node* temp = (struct node*) malloc(sizeof(struct node));

//     temp->text = the_text;
//     temp->type = category;
//     temp->link=NULL;
//     temp->down=NULL;
// if(head->down == NULL){
//     head->down = temp;
// }
// else{
//     struct node* tmp;
//     tmp = head;
//     int note=0;
//     while(tmp->down!=NULL){
//         tmp = tmp->down;
//         if(tmp->type==category){
//             note=1;
//             struct node* tempr;
//             tempr = tmp;
//             while(tempr->link!=NULL){

```

```

//                tempr= tempr->link;
//                }
//                tempr->link = temp;
//            }
//        }
//        if(note==0){
//            tmp->down = temp;
//        }
//    }
//    printf("%d : %s\n", category, the_text);
// }

```

Test Cases:

Test Case 1:

```

# include
<stdio.h>

# include <std
int main(){
}

```

Outputs:

```

ubuntu@ubuntu: ~/Desktop/correctTestCase/test1
bash: /usr/share/valgrindwrapper/valgrindwrapper_lazy.sh: No such file or directory
ubuntu@ubuntu:~/Desktop/correctTestCase/test1$ ./test1
Improper Header File: # include <std
This is the main function: int main()
The total no of real numbers: 0
The total no of string literals: 0
The total no of Separators: 2
The total no of Header Files: 1
The total no of printf: 0
The total no of key words: 0
The total no of identifiers: 0
The total no of format specifiers: 0
-----
| Separator | {

```

```
ubuntu@ubuntu: ~/Desktop/correctTestCase/test1
This is the main function: int main()

The total no of real numbers: 0

The total no of string literals: 0

The total no of Separators: 2

The total no of Header Files: 1

The total no of printf: 0

The total no of key words: 0

The total no of identifiers: 0

The total no of format specifiers: 0

Separator      | {
                | }

HeaderFile     | # include <stdio.h>

ubuntu@ubuntu:~/Desktop/correctTestCase/test1$
```

Test Case 2:

```
#
#include<stdio.h>

int main()
printf("1");
printf(";
}
```

Output:

```
ubuntu@ubuntu:~/Desktop/correctTestCase/test2
bash: /usr/share/virtualenvwrapper/virtualenvwrapper_lazy.sh: No such file or directory
ubuntu@ubuntu:~/Desktop/correctTestCase/test2$ ./test

This is the main function: int main()

This statement is a Lexical Error: printf(";

The total no of real numbers: 0

The total no of string literals: 0

The total no of Separators: 1

The total no of Header Files: 1

The total no of printf: 1

The total no of key words: 0

The total no of identifiers: 0

The total no of format specifiers: 0
```

```
ubuntu@ubuntu: ~/Desktop/correctTestCase/test2
The total no of real numbers: 0
The total no of string literals: 0
The total no of Separators: 1
The total no of Header Files: 1
The total no of printf: 1
The total no of key words: 0
The total no of identifiers: 0
The total no of format specifiers: 0

Separator      | }
HeaderFile     | # include <stdio.h>
printf         | printf("1");

ubuntu@ubuntu:~/Desktop/correctTestCase/test2$
```

Test Case 3:

```
#include<stdio.h>

int main(){

    98y-98
    0x97yih
    6.78
    -6
    8r.4f
    +.980

}
```

```
ubuntu@ubuntu: ~/Desktop/correctTestCase/test3
ubuntu@ubuntu:~/Desktop/correctTestCase/test3$ lex test.l
ubuntu@ubuntu:~/Desktop/correctTestCase/test3$ gcc -o test lex.yy.c
ubuntu@ubuntu:~/Desktop/correctTestCase/test3$ ./test

This is the main function: int main()

The total no of real numbers: 3
The total no of string literals: 3
The total no of Separators: 1
The total no of Header Files: 1
The total no of printf: 0
The total no of key words: 0
The total no of identifiers: 0
```

```
ubuntu@ubuntu: ~/Desktop/correctTestCase/test3
The total no of Separators: 1
The total no of Header Files: 1
The total no of printf: 0
The total no of key words: 0
The total no of identifiers: 0
The total no of format specifiers: 0

-----
Real_Number      | 6.78
                  | -6
                  | +.980
-----
String_Literal   | 98y-98
                  | 0x97yih
                  | 8r.4f
-----
Separator        | }
-----
HeaderFile       | # include <stdio.h>
-----
Constants        | 6.78
                  | -6
                  | +.980
-----
ubuntu@ubuntu:~/Desktop/correctTestCase/test3$
```

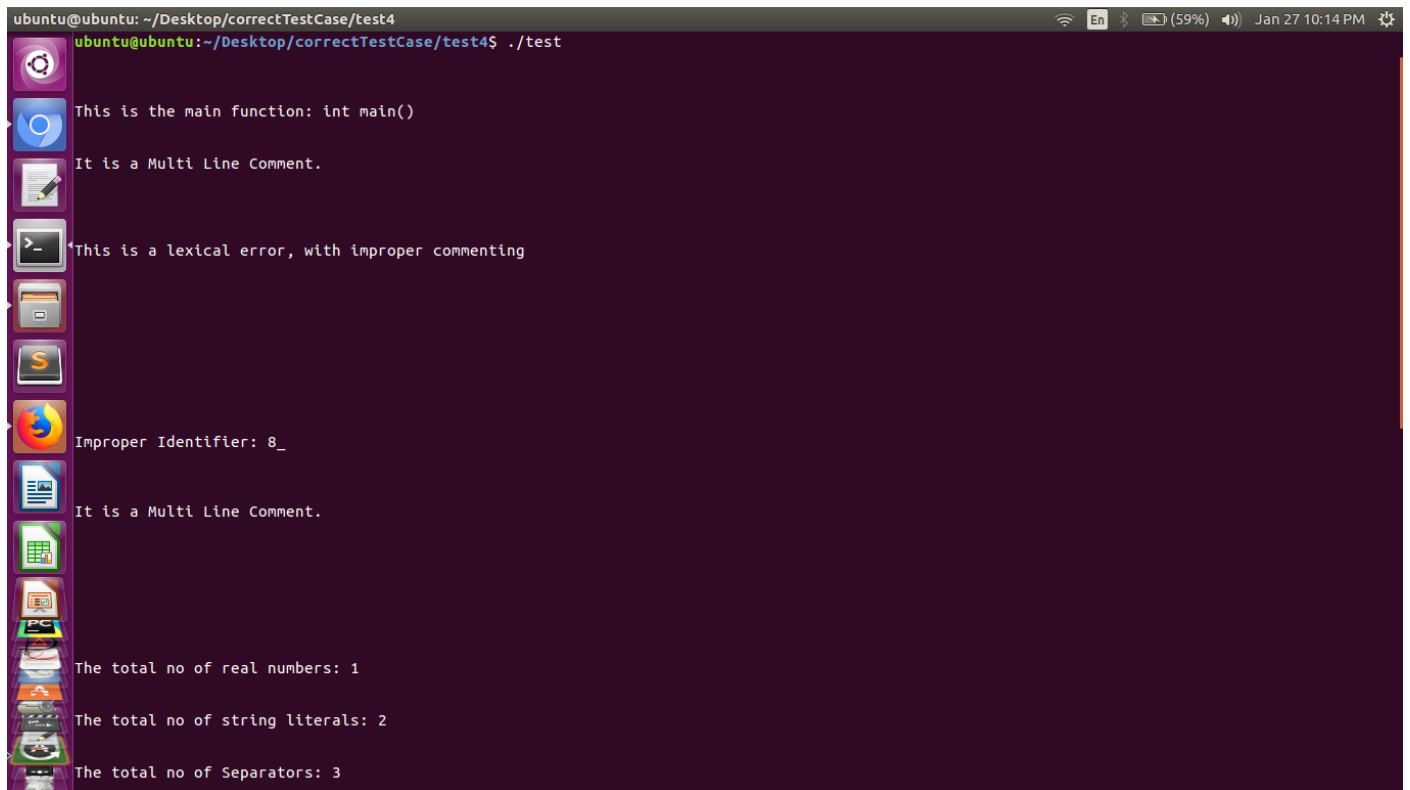
Test Case 4 :

```
# include
<stdio.h>

int main()
/*
*/
Lkj
```

```
*/  
89e-1;  
89a-1  
8_67;  
/*  
oij78;s  
*/  
}
```

Output:



```
ubuntu@ubuntu: ~/Desktop/correctTestCase/test4  
ubuntu@ubuntu:~/Desktop/correctTestCase/test4$ ./test  
This is the main function: int main()  
It is a Multi Line Comment.  
This is a lexical error, with improper commenting  
Improper Identifier: 8_  
It is a Multi Line Comment.  
The total no of real numbers: 1  
The total no of string literals: 2  
The total no of Separators: 3
```

```
ubuntu@ubuntu: ~/Desktop/correctTestCase/test4
The total no of Separators: 3

The total no of Header Files: 1

The total no of printf: 0

The total no of key words: 0

The total no of identifiers: 1

The total no of format specifiers: 0

-----
Real_Number      | 67
-----
String_Literal   | 89e-1
                  | 89a-1
-----
Separator        | ;
                  | ;
                  | }
-----
HeaderFile       | # include <stdio.h>
-----
Identifier       | lkj
-----
Constants        | 67
-----

ubuntu@ubuntu:~/Desktop/correctTestCase/test4$
```

Test Case 5:

```
# include
<stdio.h>

int main()
char str = "ruhi";
char tsr = "taj ;
int y = -9;
}
```

```
ubuntu@ubuntu: ~/Desktop/correctTestCase/test5
ubuntu@ubuntu:~/Desktop/correctTestCase/test5$ gcc -o test lex.yy.c
ubuntu@ubuntu:~/Desktop/correctTestCase/test5$ ./test

This is the main function: int main()

This is a Lexical Error- Improper quotes: "taj

The total no of real numbers: 1

The total no of string literals: 0

The total no of Separators: 4

The total no of Header Files: 1

The total no of printf: 0

The total no of key words: 3

The total no of identifiers: 3
```



```
ubuntu@ubuntu: ~/Desktop/correctTestCase/test5
The total no of key words: 3
The total no of identifiers: 3
The total no of format specifiers: 3
Real_Number      | -9
Separator         | ;
                  | ;
                  | ;
                  | }
HeaderFile        | # include <stdio.h>
Key_Word          | char
                  | char
                  | int
Identifier        | str
                  | tsr
                  | y
Operator          | =
                  | =
                  | =
Constants         | "ruhl"
                  | -9
```

Test Case 6

```
# include
<stdio.h>

int main()
/
float a, _yu;
}
```

```
ubuntu@ubuntu: ~/Desktop/correctTestCase/test6
bash: /usr/share/virtualenvwrapper/virtualenvwrapper_lazy.sh: No such file or directory
ubuntu@ubuntu:~/Desktop/correctTestCase/test6$ lex test.l
ubuntu@ubuntu:~/Desktop/correctTestCase/test6$ gcc -o test lex.yy.c
ubuntu@ubuntu:~/Desktop/correctTestCase/test6$ ./test

This is the main function: int main()

Illegal stray found in the program - /

The total no of real numbers: 0
The total no of string literals: 0
The total no of Separators: 3
The total no of Header Files: 1
The total no of printf: 0
The total no of key words: 1
The total no of identifiers: 2
```

```
ubuntu@ubuntu: ~/Desktop/correctTestCase/test6
The total no of real numbers: 0
The total no of string literals: 0
The total no of Separators: 3
The total no of Header Files: 1
The total no of printf: 0
The total no of key words: 1
The total no of identifiers: 2
The total no of format specifiers: 0

Separator      | ,
                | ;
                | }

HeaderFile     | # include <stdio.h>

Key_Word       | float

Identifier     | a
                | _yu

ubuntu@ubuntu:~/Desktop/correctTestCase/test6$
```

Test Case 7:

```
# include
<stdio.h>

int main()
{
    long long int a = -8 ;
    printf("%lld\n",a);
}
```

ubuntu@ubuntu: ~/Desktop/correctTestCase/test7
ubuntu@ubuntu:~/Desktop/correctTestCase/test7\$./test

This is the main function: int main()

The total no of real numbers: 1

The total no of string literals: 0

The total no of Separators: 3

The total no of Header Files: 1

The total no of printf: 1

The total no of key words: 3

The total no of identifiers: 1

The total no of format specifiers: 1

ubuntu@ubuntu: ~/Desktop/correctTestCase/test7

The total no of key words: 3

The total no of identifiers: 1

The total no of format specifiers: 1

Real_Number	-8
Separator	{ ; }
HeaderFile	# include <stdio.h>
printf	printf("%lld\n",a);
Key_Word	long long int
Identifier	a
Operator	=
Constants	-8

ubuntu@ubuntu:~/Desktop/correctTestCase/test7\$

Test Case 8:

```
# include
<stdio.h>

int main(
{
}
```

```
ubuntu@ubuntu: ~/Desktop/correctTestCase/test8
bash: /usr/share/virtualenvwrapper/virtualenvwrapper_lazy.sh: No such file or directory
ubuntu@ubuntu:~/Desktop/correctTestCase/test8$ lex test.l
ubuntu@ubuntu:~/Desktop/correctTestCase/test8$ gcc -o test lex.yy.c
ubuntu@ubuntu:~/Desktop/correctTestCase/test8$ ./test

Improper main, please correct it!
(

The total no of real numbers: 0
The total no of string literals: 0
The total no of Separators: 2
The total no of Header Files: 1
The total no of printf: 0
The total no of key words: 1
The total no of identifiers: 0
The total no of format specifiers: 0

-----
| Separator          | {
|
```

```
ubuntu@ubuntu: ~/Desktop/correctTestCase/test8

The total no of real numbers: 0
The total no of string literals: 0
The total no of Separators: 2
The total no of Header Files: 1
The total no of printf: 0
The total no of key words: 1
The total no of identifiers: 0
The total no of format specifiers: 0

-----
| Separator          | {
|                    | }
|
| HeaderFile         | # include <stdio.h>
|
| Key_Word           | int
|
-----

ubuntu@ubuntu:~/Desktop/correctTestCase/test8$
```

Test Case 9:

```
# include
<stdio.h>

int main()
int ab, 8_iuh;
}
```

```
ubuntu@ubuntu: ~/Desktop/correctTestCase/test9
bash: /usr/share/virtualenvwrapper/virtualenvwrapper_lazy.sh: No such file or directory
ubuntu@ubuntu:~/Desktop/correctTestCase/test9$ lex test.l
ubuntu@ubuntu:~/Desktop/correctTestCase/test9$ gcc -o test lex.yy.c
ubuntu@ubuntu:~/Desktop/correctTestCase/test9$ ./test

This is the main function: int main()

Improper Identifier: 8_iuh

The total no of real numbers: 0
The total no of string literals: 0
The total no of Separators: 3
The total no of Header Files: 1
The total no of printf: 0
The total no of key words: 1
The total no of identifiers: 1
```

```
ubuntu@ubuntu: ~/Desktop/correctTestCase/test9

The total no of real numbers: 0
The total no of string literals: 0
The total no of Separators: 3
The total no of Header Files: 1
The total no of printf: 0
The total no of key words: 1
The total no of identifiers: 1
The total no of format specifiers: 0

-----
Separator          | ,
                  | ;
                  | }
-----
HeaderFile         | # include <stdio.h>
-----
Key_Word           | int
-----
Identifier         | ab
-----

ubuntu@ubuntu:~/Desktop/correctTestCase/test9$
```