

# System Programming and Operating Systems Lab

## ASSIGNMENT 2

**Name: Shrirang Mhalgi**

**Roll No: 322008**

**Batch: B 1**

### **1 Date of Completion:**

8/01/2018

### **2 Aim:**

Write a program using Lex specifications to implement lexical analysis phase of compiler to count no. of words, lines and characters of given input file.

### **3 Objectives:**

To count no. of words, lines and characters of given input file using Lex.

### **4 Theory:**

Lex is officially known as a "Lexical Analyser". Its main job is to break up an input stream into more usable elements. Or in other words, to identify the "interesting bits" in a text file. For example, if you are writing a compiler for the C programming language, the symbols ( ) ; all have significance on their own. The letter a usually appears as part of a keyword or variable name, and is not interesting on its own. Instead, we are interested in the whole word. Spaces and newlines are completely uninteresting, and we want to ignore them completely, unless they appear within quotes "like this". All of these things are handled by the Lexical Analyser.

Lex helps write programs whose control flow is directed by instances of regular expressions in the input stream. It is well suited for editor-script type transformations and for segmenting input in preparation for a parsing routine.

Lex source is a table of regular expressions and corresponding program fragments. The table is translated to a program which reads an input stream, copying it to an output stream and partitioning the input into strings which match the given expressions. As each such string is recognized the corresponding program fragment is executed. The recognition of the expressions is performed by a deterministic finite automaton generated by Lex. The program fragments written by the user are executed in the order in which the corresponding regular expressions occur in the input stream.

The lexical analysis programs written with Lex accept ambiguous specifications and choose the longest match possible at each input point. If necessary, substantial lookahead is performed on the input, but the input stream will be backed up to the end of the current partition, so that

the user has general freedom to manipulate it.

Lex can generate analyzers in either C or Ratfor, a language which can be translated automatically to portable Fortran. It is available on the PDP-11 UNIX, Honeywell GCOS, and IBM OS systems. This manual, however, will only discuss generating analyzers in C on the UNIX system, which is the only supported form of Lex under UNIX Version 7. Lex is designed to simplify interfacing with Yacc, for those with access to this compiler-compiler system.

### **Running Lex:**

```
>lex filename.l  
>gcc lex.yy.c -lfl  
>./a.out
```

To compile a lex program, do the following: Use the lex program to change the specification file into a C language program. The resulting program is in the lex.yy.c file. Use the cc command with the -ll flag to compile and link the program with a library of lex subroutines. The resulting executable program is in the a.out file.

## 5 Algorithm:

Step1: Start

Step2:Read input file name

Step3 :Initialize word count,letter count,line count and character count to zero

Step4 :If input file matches

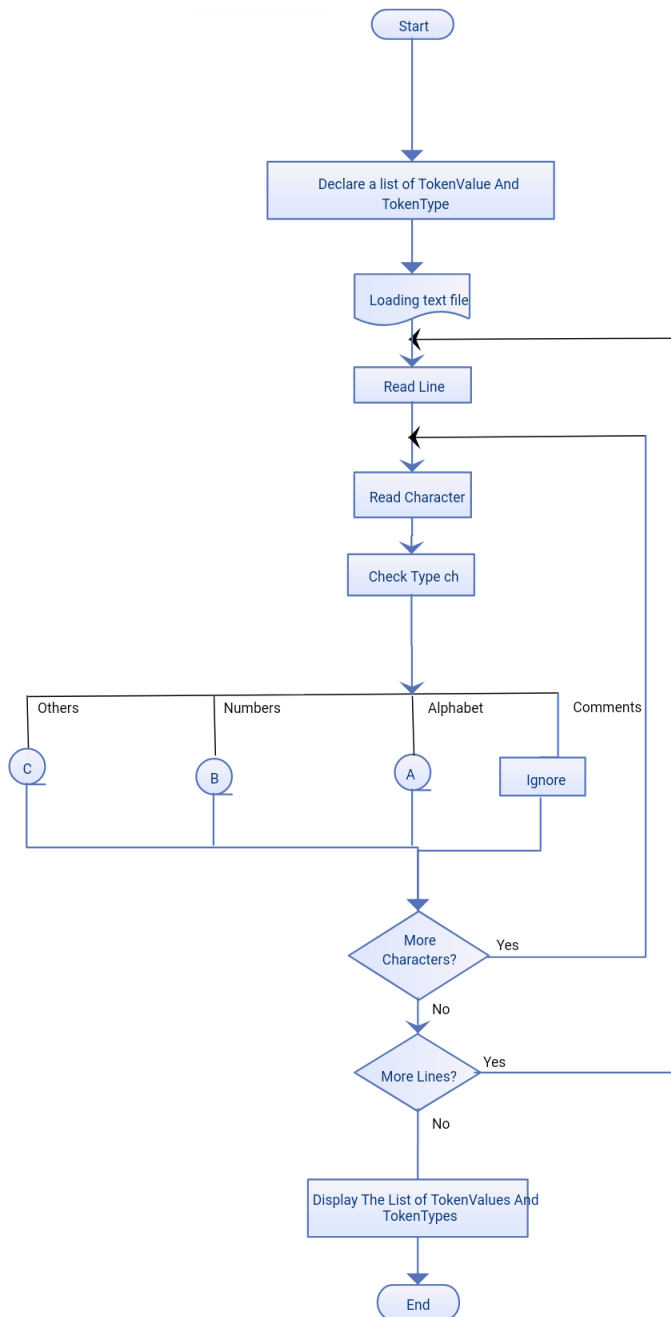
[^~~¥~~t~~¥~~n]+

then increment the word count,line count,character count

Step5 :Print the count of word,line,character

Step6 :Stop

## 6 Flowchart:



## 7 Code:

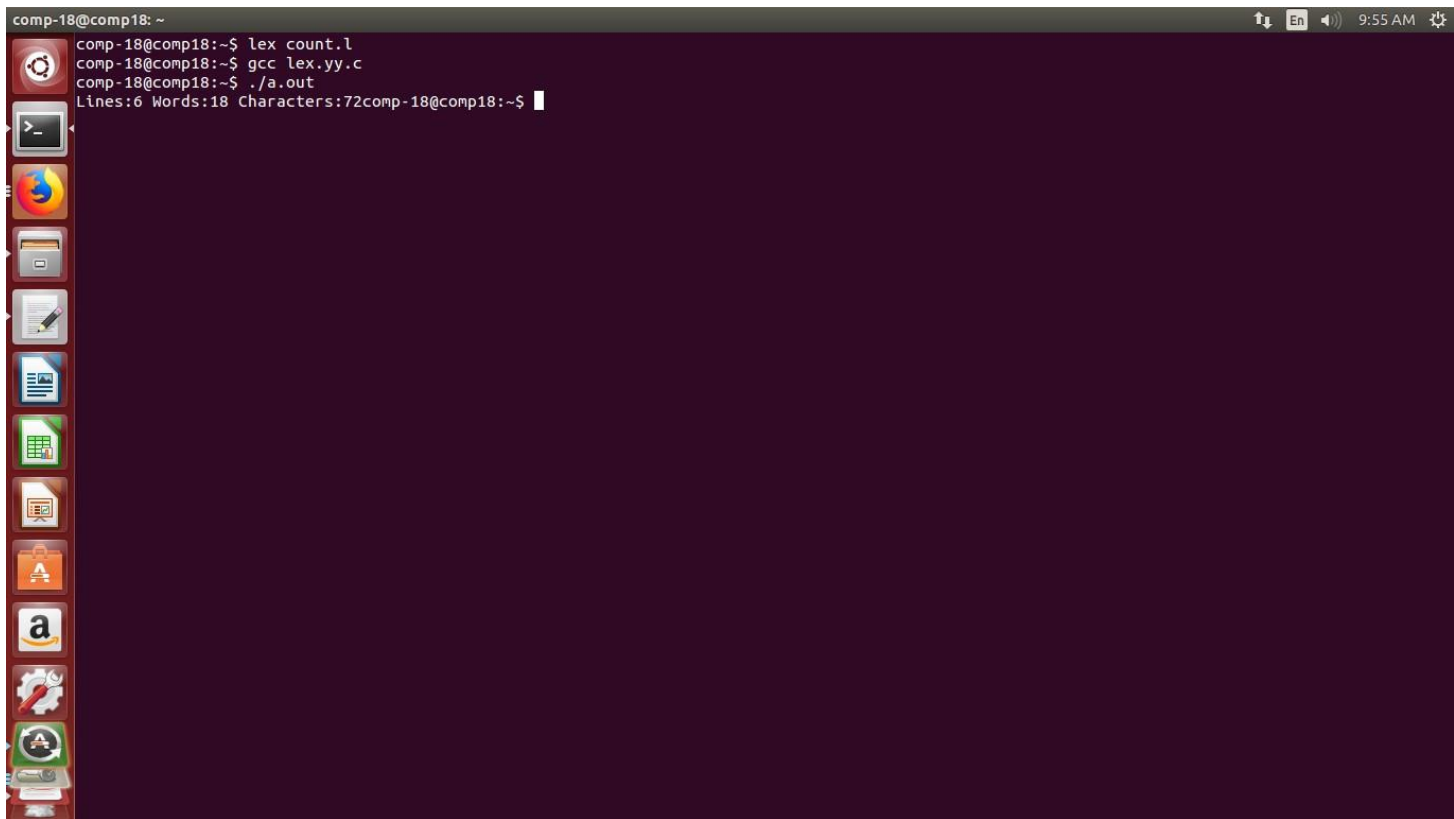
```
%{
    int linecount=0,wordcount=0,charactercount=0;
}%

%%
[^\t\n]+ {wordcount++;charactercount+=yyleng;} /* if not space and
not new line then it is a word*/
[\n] {linecount++;}
[a-zA-Z0-9] {charactercount++;}
%%

int main()
{
    yyin=fopen("count.l","r");
    yylex();
    printf("\n wordcount :%d \n linecount:%d
\n Charactercount :%d \n",wordcount,linecount,charactercount);
}

int yywrap()
{
}
```

## 8 Output:



A terminal window titled 'comp-18@comp18: ~' with a dark purple background. The window shows the following commands and output:

```
comp-18@comp18:~$ lex count.l
comp-18@comp18:~$ gcc lex.yy.c
comp-18@comp18:~$ ./a.out
Lines:6 Words:18 Characters:72comp-18@comp18:~$
```

The terminal window has a sidebar on the left with various application icons, including a terminal, a web browser, a file manager, a text editor, a spreadsheet, a presentation, a shopping cart, an Amazon logo, a gear, and a printer. The top right corner of the window shows system icons for network, language (En), volume, and a clock displaying 9:55 AM.

## 9 Conclusion:

In this assignment we learn in detail the concept of lex compiler and observe the Lex specifications to implement lexical analysis phase of compiler to count no. of words, lines and characters of given input file.