1. **Design a LEX Code to count the number of lines, space, tab-meta character and rest of characters in a given Input pattern.**

**Code:-**

%{

#include<stdio.h>

int n=0,m=0,t=0,c=0;

%}

%%

\n {n++;}

\t {m++;}

[ ] {t++;}

. {c++;}

%%

int yywrap(){return 1;}

int main()

{

yylex();

printf("\nTotal number of lines=%d",n);

printf("\nTotal number of tabs=%d ",m);

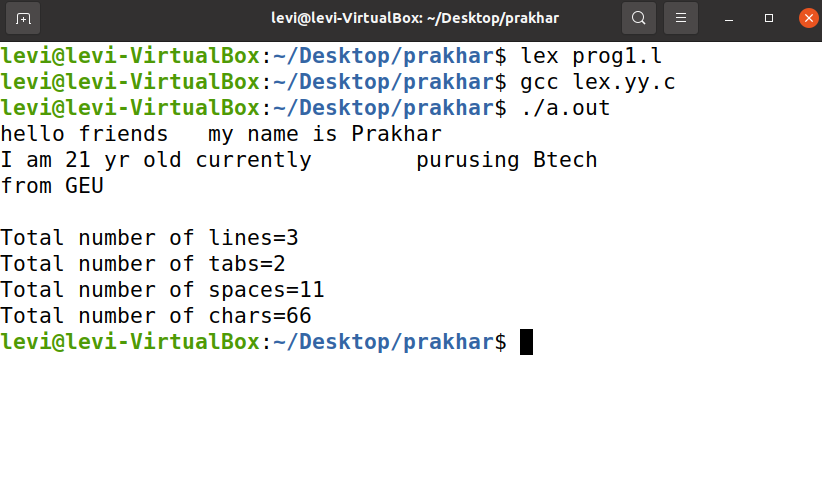
printf("\nTotal number of spaces=%d",t);

printf("\nTotal number of chars=%d\n",c);

return 0;

}

**Output:-**



1. **Design a LEX Code to identify and print valid Identifier of C/C++ in given Input pattern.**

**Code:-**

%{

#include<stdio.h>

int c=0;

%}

%%

^[a-zA-Z\_$][a-zA-Z0-9\_$]\* {c++; printf("\t Valid Identifier = %s\n",yytext);}

.\* {printf("\t Invalid Identifier = %s\n",yytext);};

%%

int yywrap(){return 1;}

int main()

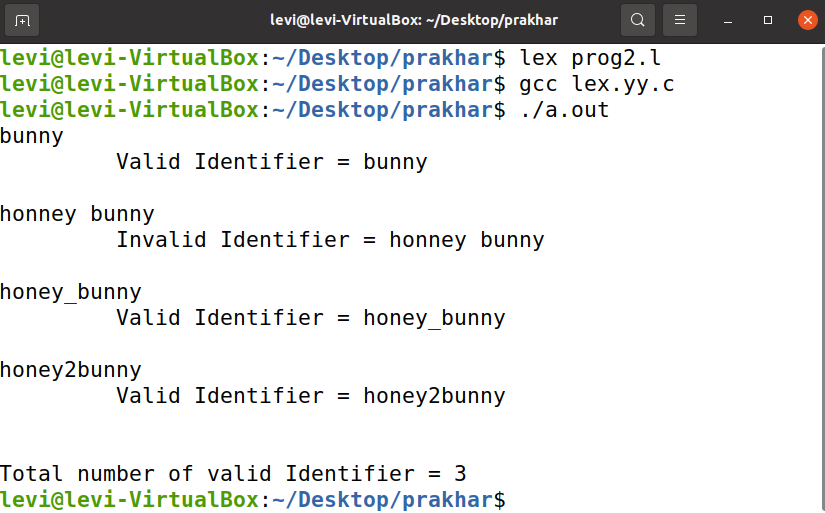
{

yylex();

printf("\nTotal number of valid Identifier = %d \n",c);

}

**Output:-**



1. **Design a LEX Code to identify and print integer and float value in given Input pattern.**

**Code:-**

%{

#include<stdio.h>

int m=0,n=0;

%}

%%

[0-9]+ {m++; printf("\t Integer = %s\n",yytext);}

[0-9]\*"."[0-9]+ {n++; printf("\t Float = %s\n",yytext);}

. ;

%%

int yywrap(){return 1;}

int main()

{

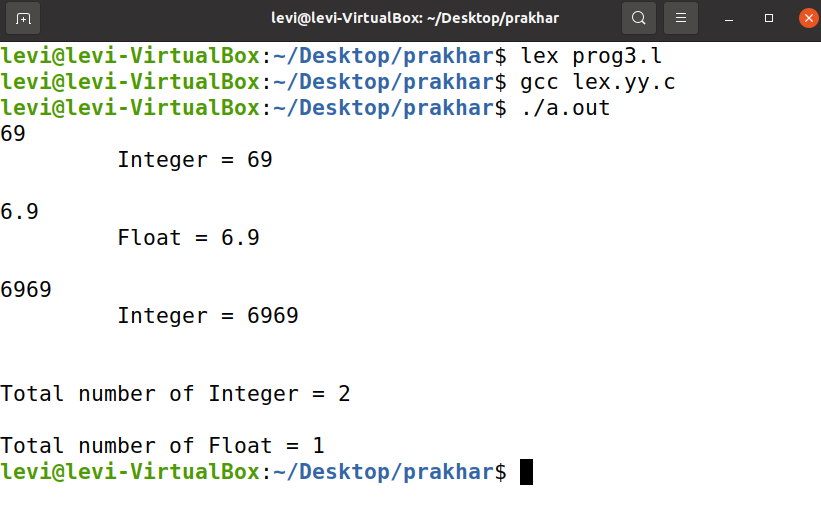
yylex();

printf("\nTotal number of Integer = %d \n",m);

printf("\nTotal number of Float = %d \n",n);

}

**Output:-**



1. **Design a LEX Code for Tokenizing (Identify and print OPERATORS, SEPERATORS, KEYWORDS, IDENTIFERS) the following C-fragment:**

**int p=1,d=0,r=4;**

**float m=0.0, n=200.0;**

**while (p <= 3)**

**{ if(d==0)**

**{ m= m+n\*r+4.5; d++; }**

**else**

**{ r++; m=m+r+1000.0; }**

**p++; }**

**Code:-**

%{

#include<stdio.h>

int n=0;

%}

%%

"while"|"if"|"else"|"int"|"float" {n++; printf("\n\t Keywords: %s",yytext);}

[a-zA-Z\_][a-zA-Z0-9\_]\* {n++; printf("\n\t Identifier: %s",yytext);}

"<="|"=="|"="|"++"|"-"|"\*"|"+" {n++; printf("\n\t Operator: %s",yytext);}

"("|")"|"{"|"}"|","|";" {n++; printf("\n\t Seperators: %s",yytext);}

[0-9]\*"."[0-9]+ {n++; printf("\n\t Float: %s",yytext);}

[0-9]+ {n++; printf("\n\t Integer: %s",yytext);}

. ;

%%

int yywrap(){return 1;}

int main()

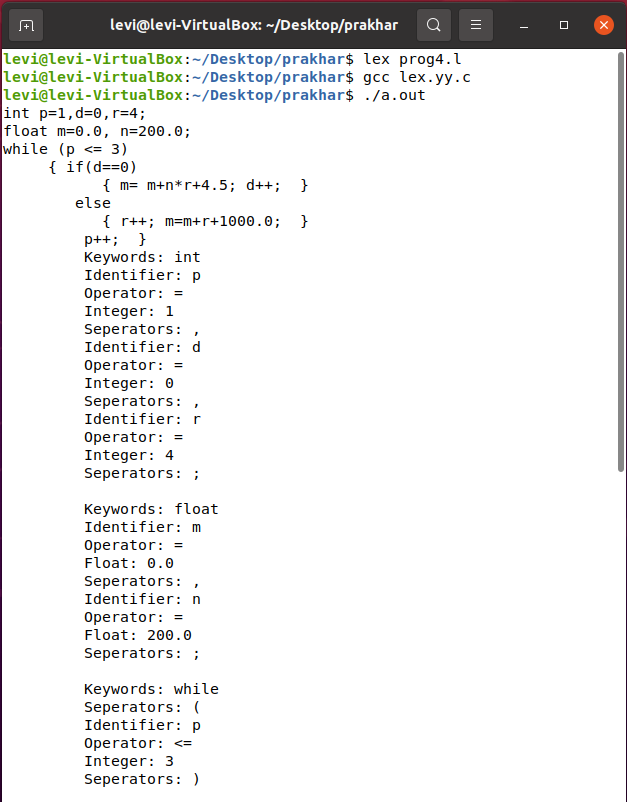
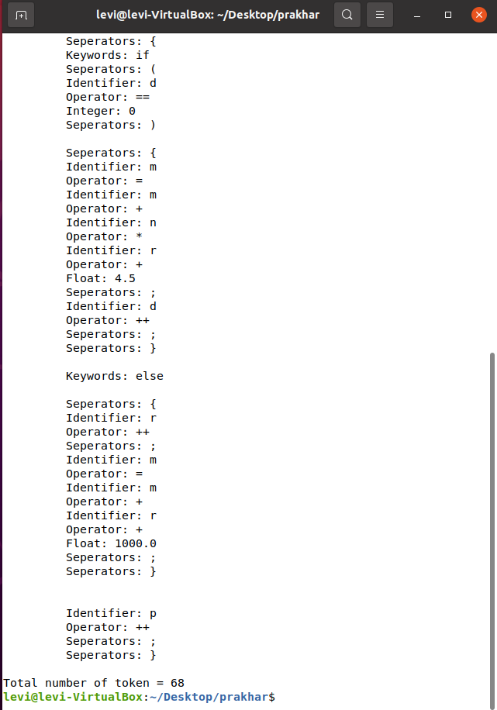
{

yylex();

printf("\nTotal number of token = %d \n",n);

}

Output:-

1. **Design a LEX Code to count and print the number of total characters, words, white spaces in given ‘Input.txt’ file.**

**Code:-**

%{

#include<stdio.h>

int n=0,w=0,c=0;

%}

%%

\n {n++;}

[^ \n\t]+ {w++;c=c+yyleng;}

. c++;

%%

int yywrap(){return 1;}

int main()

{

extern FILE \*yyin;

yyin=fopen("input.txt","r");

yylex();

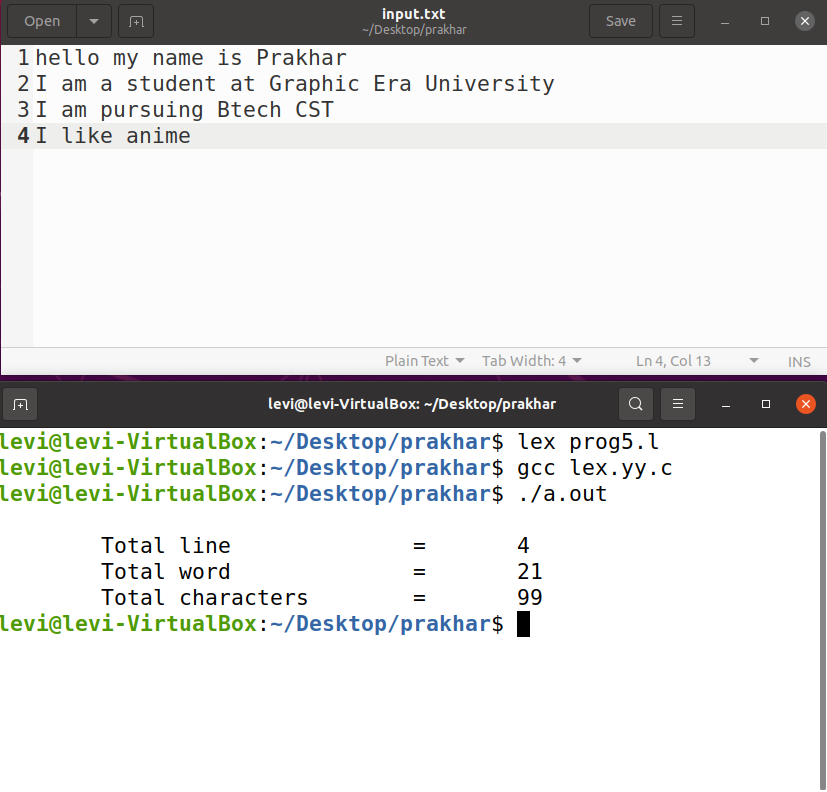
printf("\n\tTotal line\t\t=\t%d",n);

printf("\n\tTotal word\t\t=\t%d",w);

printf("\n\tTotal characters\t=\t%d\n",c);

}

**Output:-**



1. **Design a LEX Code to replace white spaces of ‘Input.txt’ file by a single blank character into ‘Output.txt’ file.**

**Code:-**

%{

#include<stdio.h>

%}

%%

[ \n\t]+ fprintf(yyout," ");

. fprintf(yyout,"%s",yytext);

%%

int yywrap(){return 1;}

int main()

{

extern FILE \*yyin,\*yyout;

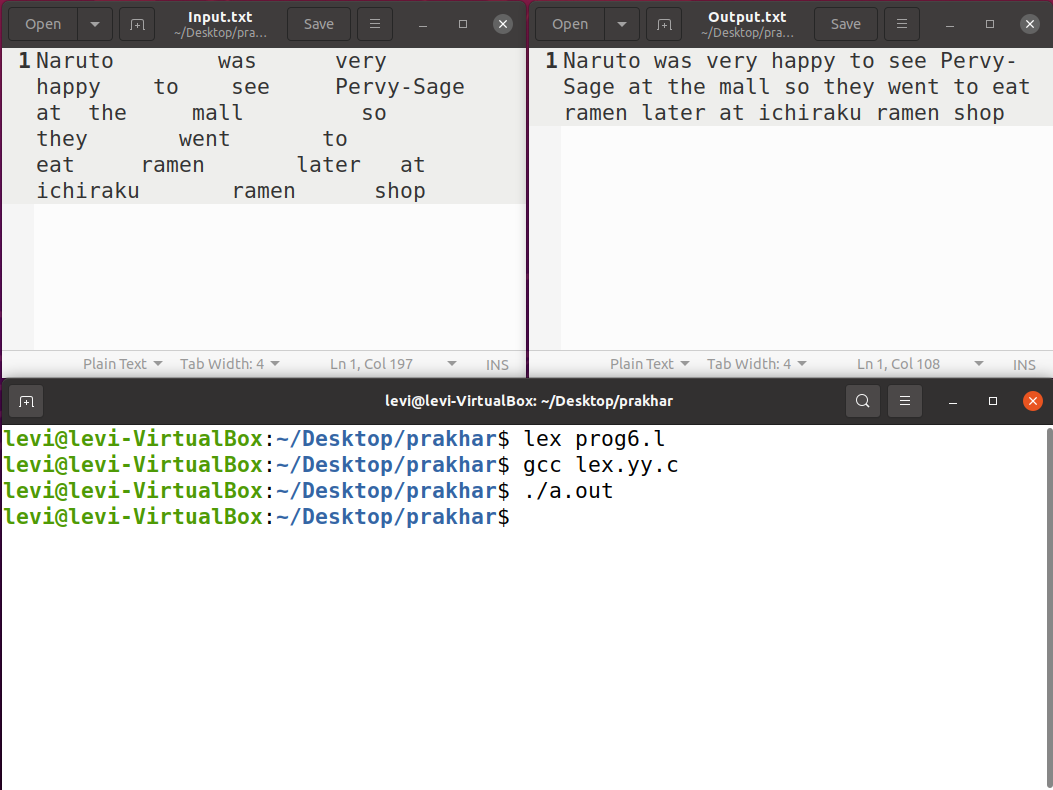
yyin=fopen("Input.txt","r");

yyout=fopen("Output.txt","w");

yylex();

}

**Output:-**



1. **Design a LEX Code to remove the comments from any C-Program given at run-time and store into ‘out.c’ file.**

**Code:-**

%{

#include<stdio.h>

%}

%%

\/\/(.\*) {};

\/\\*(.\*\n)\*.\*\\*\/ {};

%%

int yywrap(){return 1;}

int main ()

{

yyin=fopen("in.c","r");

yyout=fopen("out.c","w");

yylex();

}

**Output:-**



1. **Design a LEX Code to extract all html tags in the given HTML file at run time and store into Text file given at run time.**

**Code:-**

%{

#include<stdio.h>

%}

%%

\<[^>]\*\> {fprintf(yyout,"%s\n",yytext);}

.|\n ;

%%

int yywrap(){return 1;}

int main()

{

yyin=fopen("index.html","r");

yyout=fopen("extract.txt","w");

yylex();

return 0;

}

**Output:-**

