***Code 1***

%{

#include<stdio.h>

int ch=0, bl=0, ln=0, tb=0;

%}

%%

[\n] {ln++;}

[\t] {tb++;}

[" "] {bl++;}

. {ch++;}

%%

int yywrap(){return 1;}

int main()

{

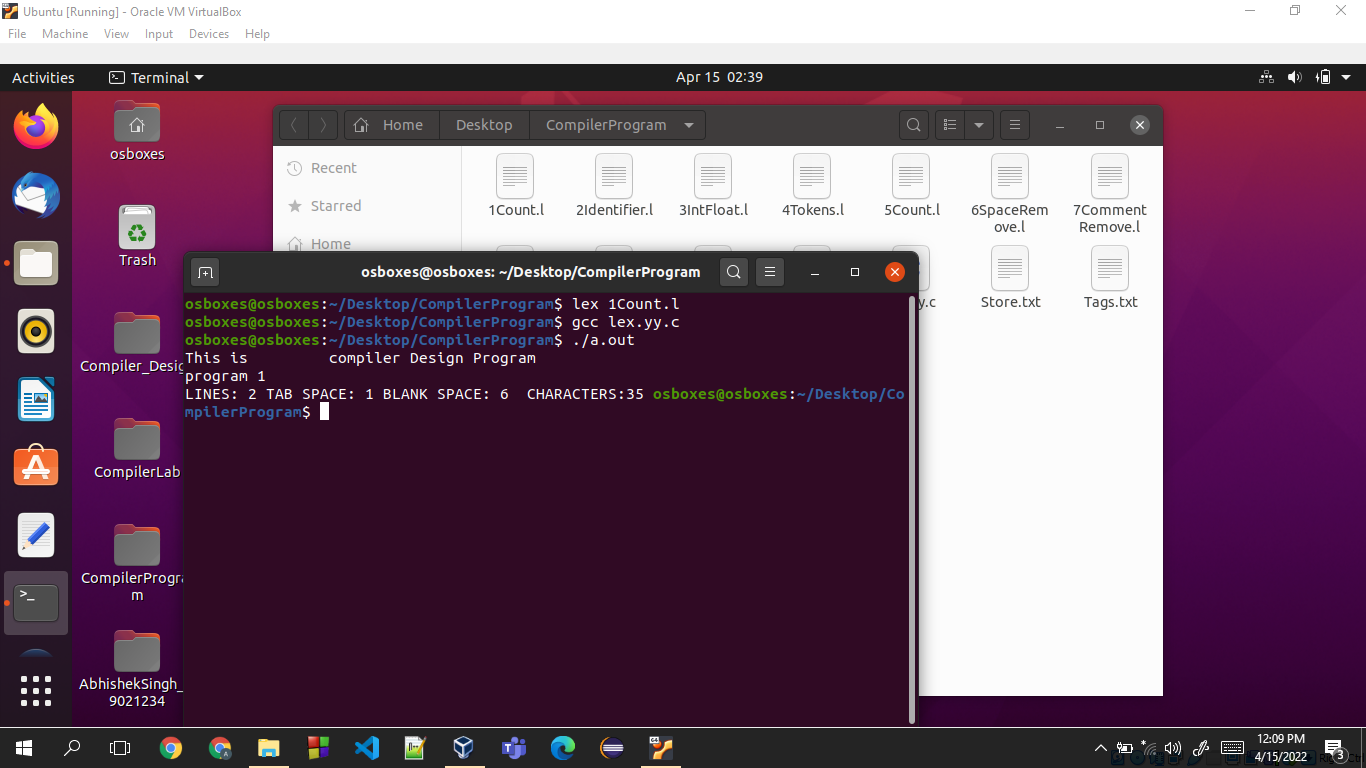
yylex();

printf("LINES: %d TAB SPACE: %d BLANK SPACE: %d CHARACTERS:%d ",ln,tb,bl,ch);

return 0;

}

**OUTPUT**



***Code 2***

%{

int count=0;

%}

op [+-\*/]

letter [a-zA-Z]

digitt [0-9]

id {letter}\*|({letter}{digitt})+

notid ({digitt}|{letter})+

%%

[\t\n]+

("int")|("float")|("char")|("case")|("default")|("if")|("for")|("printf")|("scanf") {printf("%s is a keyword\n", yytext);}

{id} {printf("%s is an identifier\n", yytext); count++;}

{notid} {printf("\n%s is not an identifier\n", yytext);}

%%

int yywrap(){

return 1;

}

int main(int argc, char \*argv[]) {

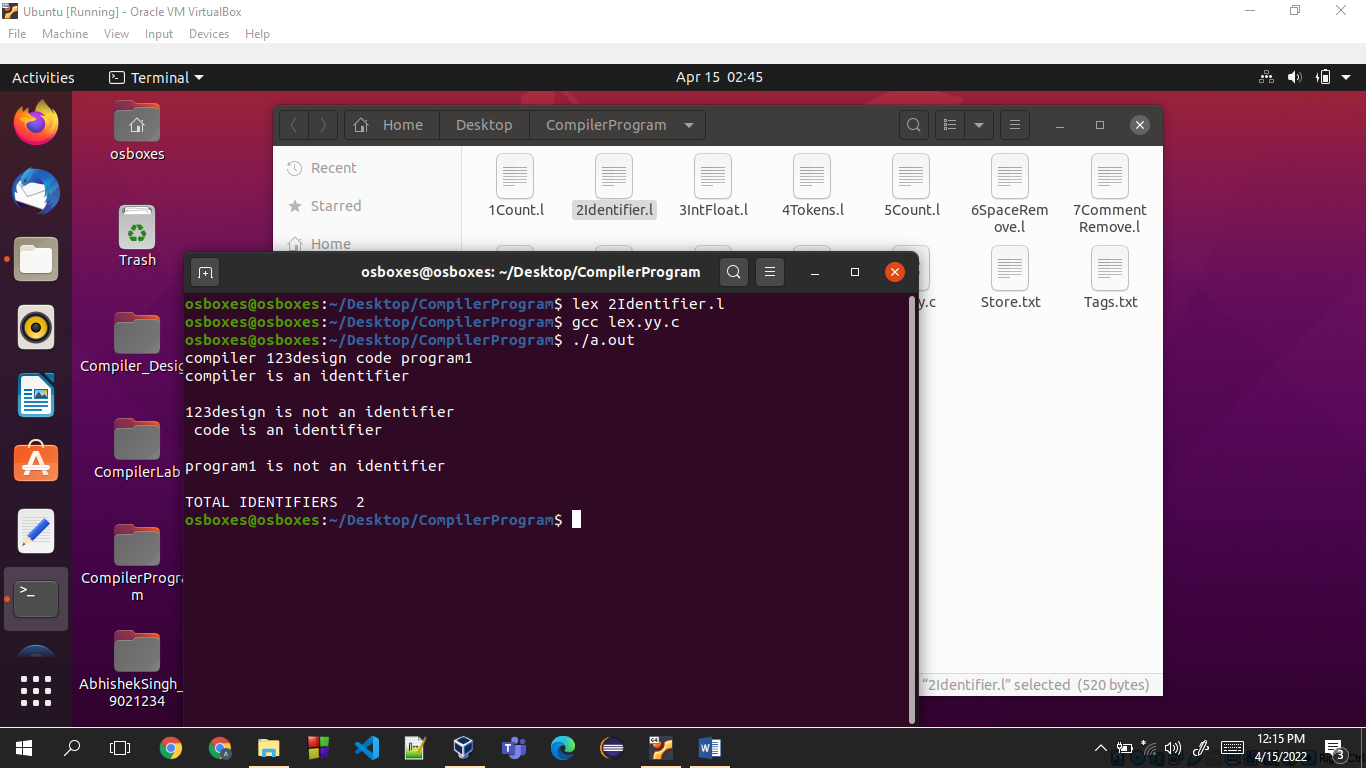
yylex();

printf("\nTOTAL IDENTIFIERS %d\n",count);

return 0;

}

**OUTPUT**



***Code 3***

%{

%}

DIGIT [0-9]

%%

{DIGIT}\* {printf("is a Integer\n");}

{DIGIT}\*?\.{DIGIT}\* {printf("is a Float\n");}

%%

int yywrap(){}

int main(int argc, char \*argv[])

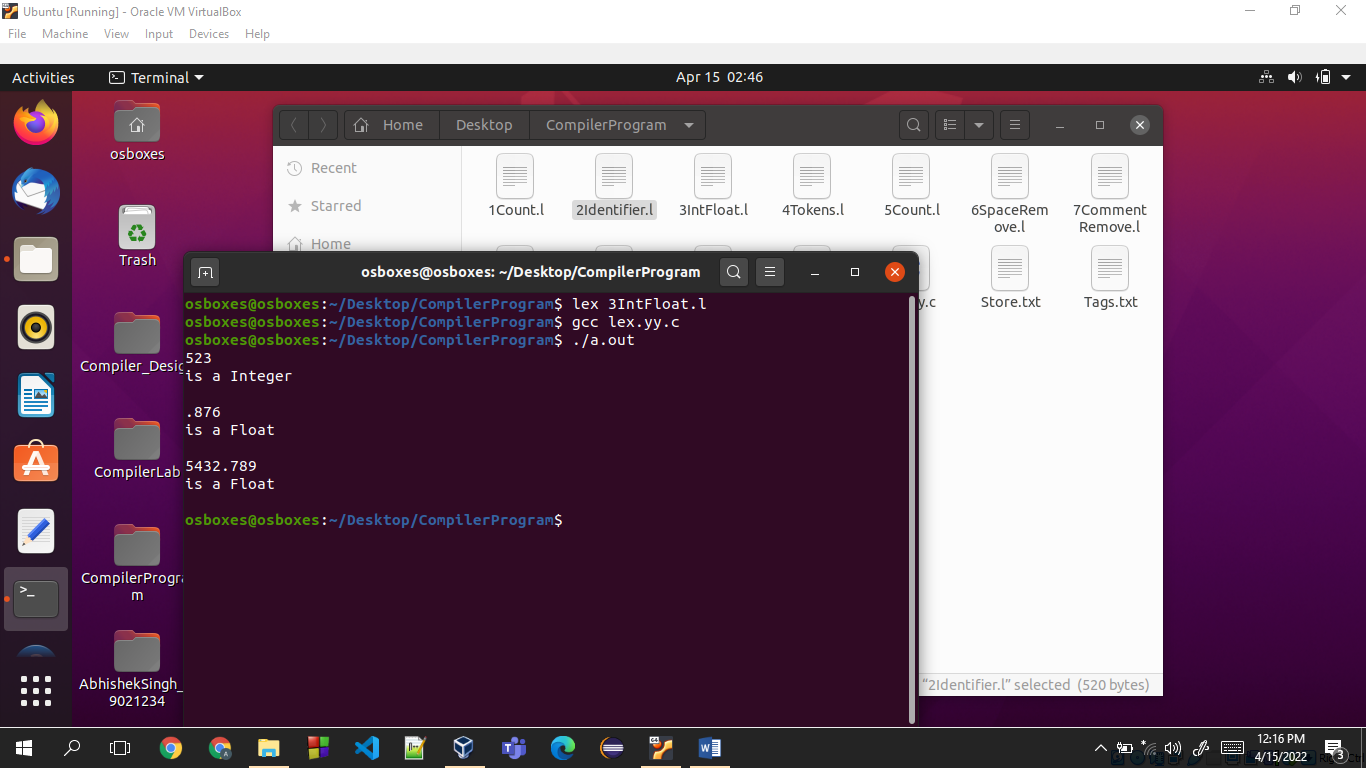
{

yylex();

return 0;

}

**OUTPUT**



***Code 4***

%{

#include <stdio.h>

%}

DIGIT [0-9]

NUMBER {DIGIT}+

REAL {DIGIT}\*"."{DIGIT}+

TEXT [a-zA-Z ]+

TEXT\_NUMBERS [a-zA-Z0-9]

CONDITIONALS "if"|"else"|"else if"|"switch"|"case"

KEYWORD "break"|"continue"|"goto"|"printf"|"scanf"|"sprintf"|"sscanf"|"fopen"|"fwrite"|"fread"|"fclose"|" write"|"read"|"open"|"close"|"return"|"int"|"float"|"char"|"unsigned"|"signed"|"short"|"long"|"d ouble"

ITERATORS "for"|"while"|"do"

PREPROCESSOR "#"|"#line"|"#undef"|"#error"|"#elif"|"#else"|"#endif"|"#if"|"#define"|"#include"|"#pragma"|"#i fndef"|"#ifdef"

DELIMITER [; :\t\n()"]

IDENTIFIER [a-zA-Z]{TEXT\_NUMBERS}\*|[a-zAZ]{TEXT\_NUMBERS}\*[[{NUMBER}+]]

FORMAT\_SPECIFIER "%"{TEXT\_NUMBERS}+

FILE "<"{IDENTIFIER}.h">"

COMMENT "/\*"[a-zA-Z0-9 \t\n;.~!@#$%^&\*()\_+=<>?:"{}]\*"\*/"

AOPERATOR "+"|"-"|"\*"|"/"|"="

BLOCK\_BEGINS "{"

BLOCK\_ENDS "}"

UNARY "++"|"--"

LOPERATOR "&"|"|"|"&&"|"~"|"||"|">"|"<"|">="|"<="|"=="

FUNCTION {IDENTIFIER}+"("{DELIMITER}\*{TEXT}{TEXT\_NUMBERS}\*{DELIMITER}\*")"

%%

{CONDITIONALS} { printf("%s is a conditional\n", yytext); }

{ITERATORS} { printf("%s is an iterator\n", yytext); }

{DIGIT} { printf("%s is a digit\n", yytext); }

{NUMBER} { printf("%s is a number\n", yytext); }

{REAL} { printf("%s is a real number\n", yytext); }

{PREPROCESSOR} { printf("%s is a preprocessor directive\n", yytext); }

{DELIMITER} { printf("%s is a delimiter\n", yytext); }

{KEYWORD} { printf("%s is a keyword\n", yytext); }

{IDENTIFIER} { printf("%s is an identifier\n", yytext); }

{COMMENT} { printf("%s is a comment\n", yytext); }

{AOPERATOR} { printf("%s is a mathematical operator\n", yytext); }

{LOPERATOR} { printf("%s is a logical operator\n", yytext); }

{BLOCK\_BEGINS} { printf("Block begins\n", yytext); }

{BLOCK\_ENDS} { printf("Block ends\n", yytext); }

{FILE} { printf("%s is a file\n", yytext); }

{UNARY} { printf("%s is a unary operator\n", yytext); }

{FUNCTION} { printf("%s is a function\n", yytext); }

{FORMAT\_SPECIFIER} {printf("%s is a format specifier\n", yytext); }

%%

int yywrap(){

return 1;

}

int main(int argc, char \*argv[]) {

extern FILE \*yyin;

yyin = fopen(argv[1],"r");

yylex();

return 0;

}

**Input**

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++; }

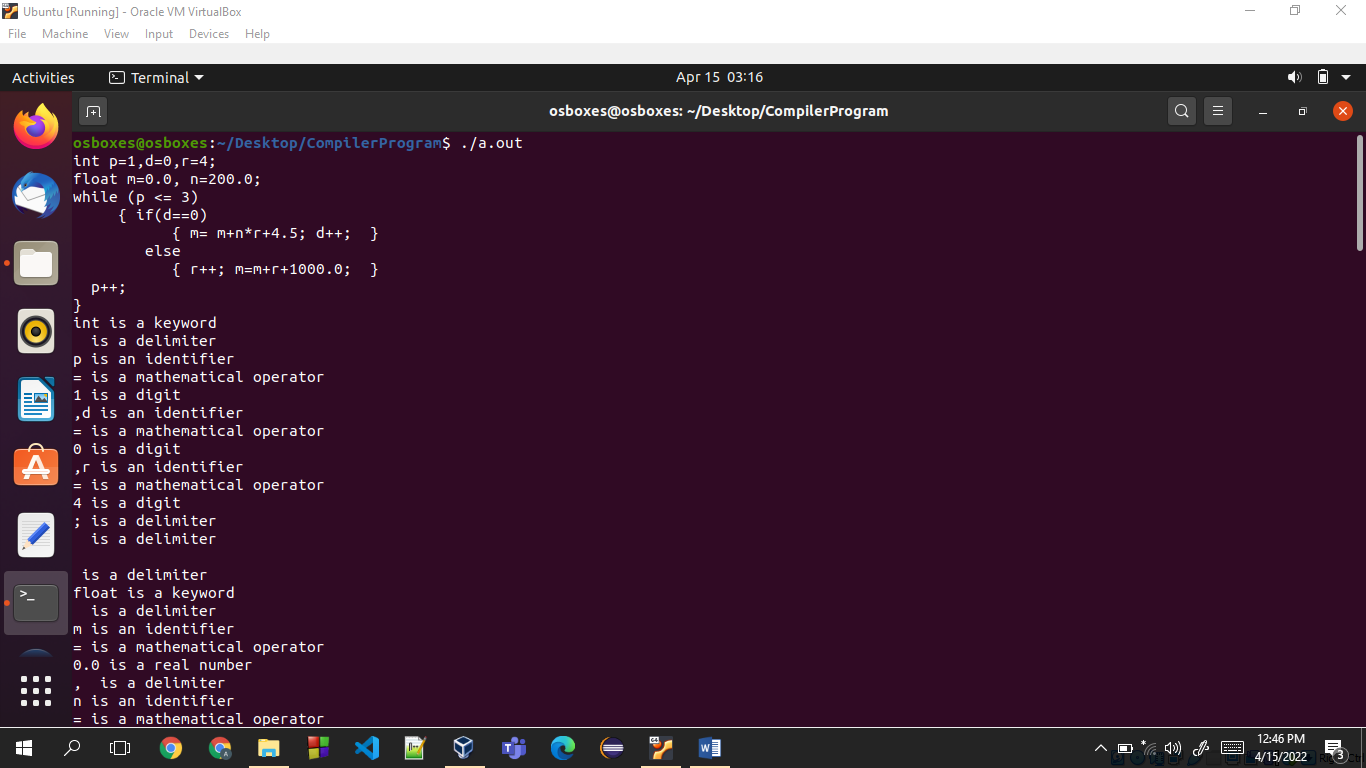
else

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

p++;

}

**OUTPUT**



***Code 5***

%{

#include<stdio.h>

int words=0,spaces=0,tchar=0,line=0;

%}

%%

\n line++;

" " {spaces++;words++;}

[\t\n] {words++;}

. {tchar++;}

%%

int yywrap(){

return 1;

}

int main(int argc, char \*argv[])

{

extern FILE \*yyin;

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

yylex();

printf("\nLines : %d Characters : %d WORDS : %d SPACES %d \n",line,tchar,words,spaces);

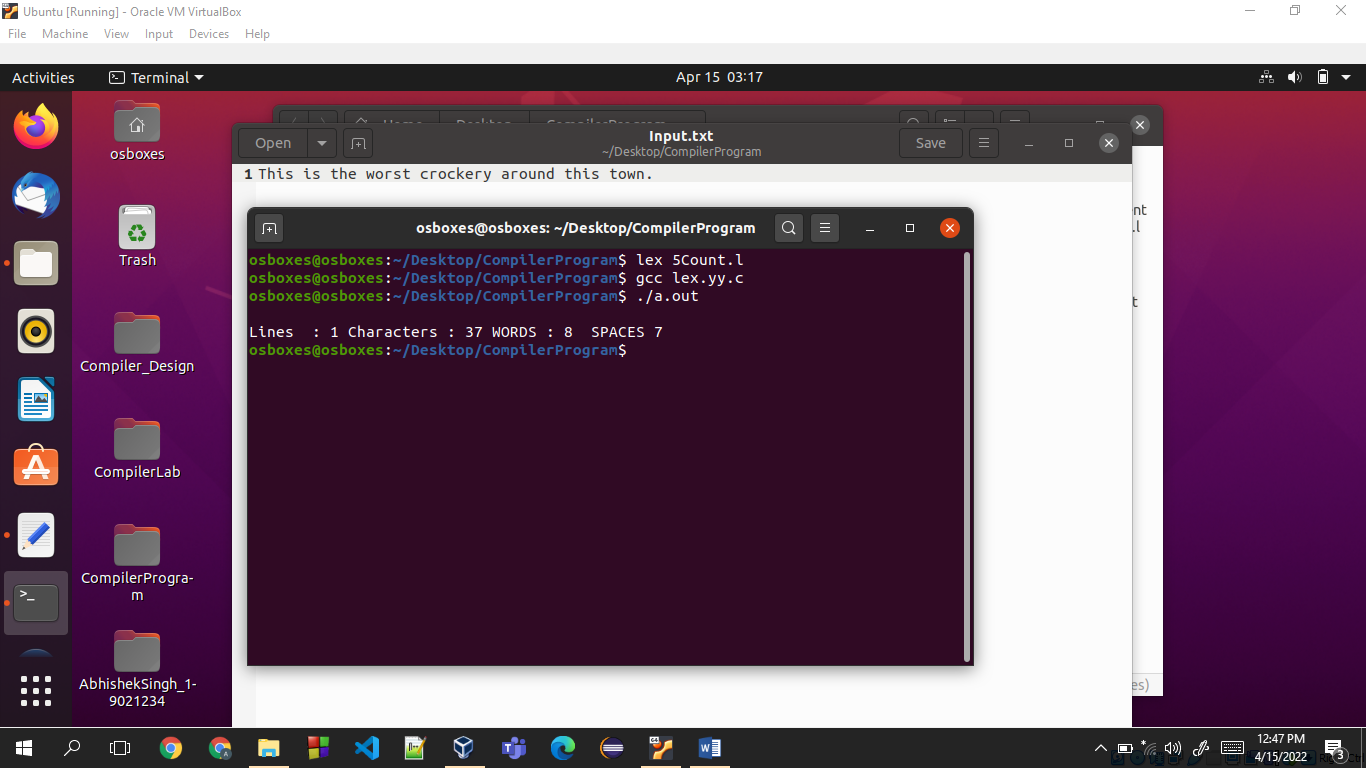
return 0;

}

**Input File -> 5\_Input.txt**

This is the worst crockery around this town.

**OUTPUT**



***Code 6***

%{

%}

space [ \t]

emptyline \n

%%

{space}+ fprintf(yyout," ");

{emptyline}+ fprintf(yyout,"\n");

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

%%

int yywrap(){

return 1;

}

int main(int argc, char \*argv[])

{

extern FILE \*yyout;

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

yylex();

return 0;

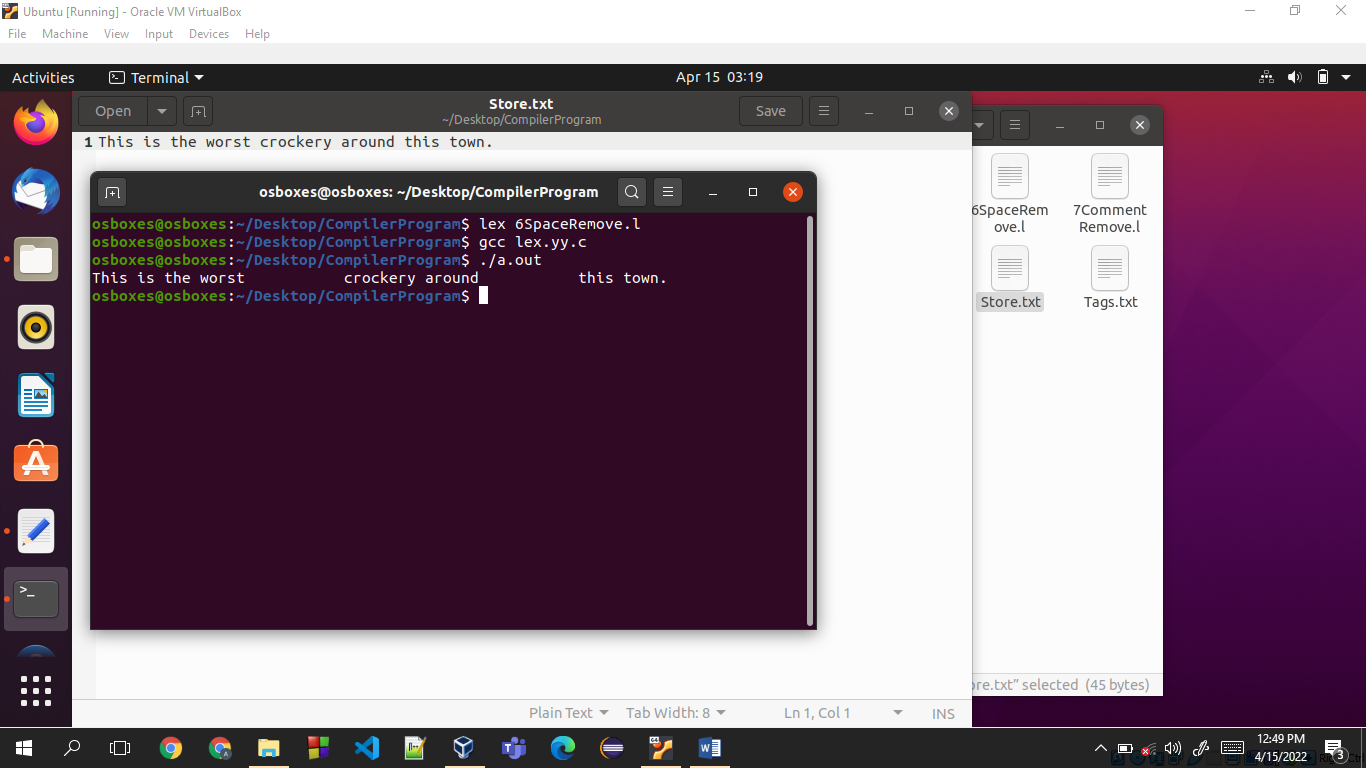
}

**INPUT**

This is the worst crockery around this town.

**OUTPUT File -> Store.txt**

This is the worst crockery around this town.



***Code 7***

%{

%}

%%

\/\/.\* ;

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

%%

int yywrap(){

return 1;

}

int main(int argc, char \*argv[])

{

extern FILE \*yyout;

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

yylex();

return 0;

}

**INPUT File -> comment.txt**

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

float m=0.0, n=200.0; // hello

while (p <= 3)

{ if(d==0) //this is wrong

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

else

{ r++; m=m+r+1000.0; } // haha

p++; }

**OUTPUT**

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

float m=0.0, n=200.0;

while (p <= 3)

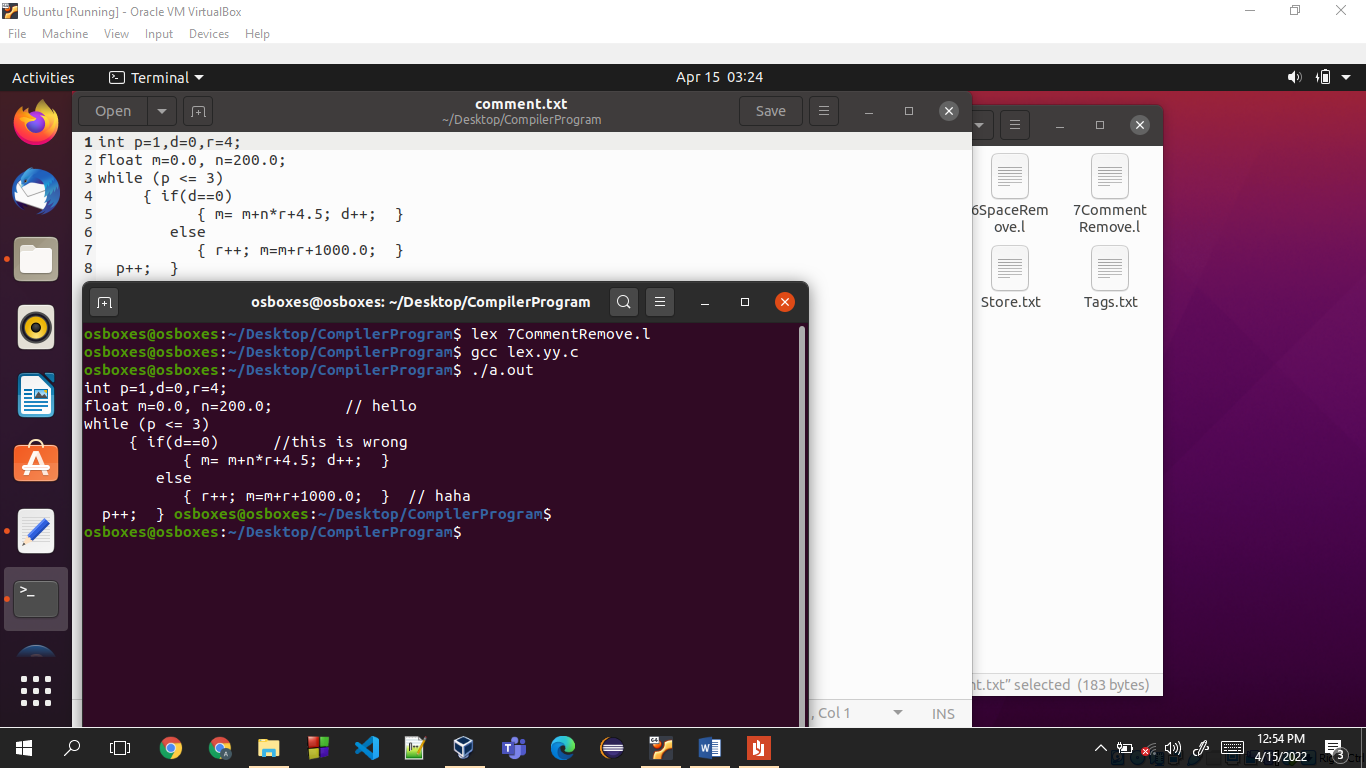
{ if(d==0)

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

else

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

p++; }



***Code 8***

%{

%}

%%

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

. ;

%%

int yywrap(){

return 1;

}

int main(int argc, char \*argv[])

{

extern FILE \*yyin;

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

yylex();

return 0;

}

**INPUT File -> Tags.txt**

<html> heloo </html>

<html> whatever </html>

ZXXZ

zxZX

zxzX

**OUTPUT**

