**RAM LAL ANAND COLLEGE**

**UNIVERSITY OF DELHI**

**DEPARTMENT OF COM****PUTER SCIENCE**

**SESSION: June-December 2022**

**PRACTICAL FILE**

Submitted to: Manisha Wadhwa Arora

**Program Name: B.Sc(H) Computer Science**

**Semester: V**

**Title of the paper: System Programming**

**Unique Paper code: 32347501**

**Name of the Student: Md Ghulam Hussain**

**Examination Roll No.: 20058570021**

**Class roll no.: 4021**

# Ques 1

1. Write a Lex program to count the number of lines and characters in the input file.

Write a Lex program to count the number of lines and characters in the input file.

/\*ques1.l\*/

%{

#include<stdio.h>

int nlines=0, nchar=0;

%}

%%

[\n] {nlines++;}

. {nchar++;}

%%

int main()

{

yyin = fopen("ques1.l", "r");

yylex();

printf("\nFile contents...\n");

printf("\n\t%d Line : ", nlines);

printf("\n\t%d Character : ", nchar);

return 0;

}

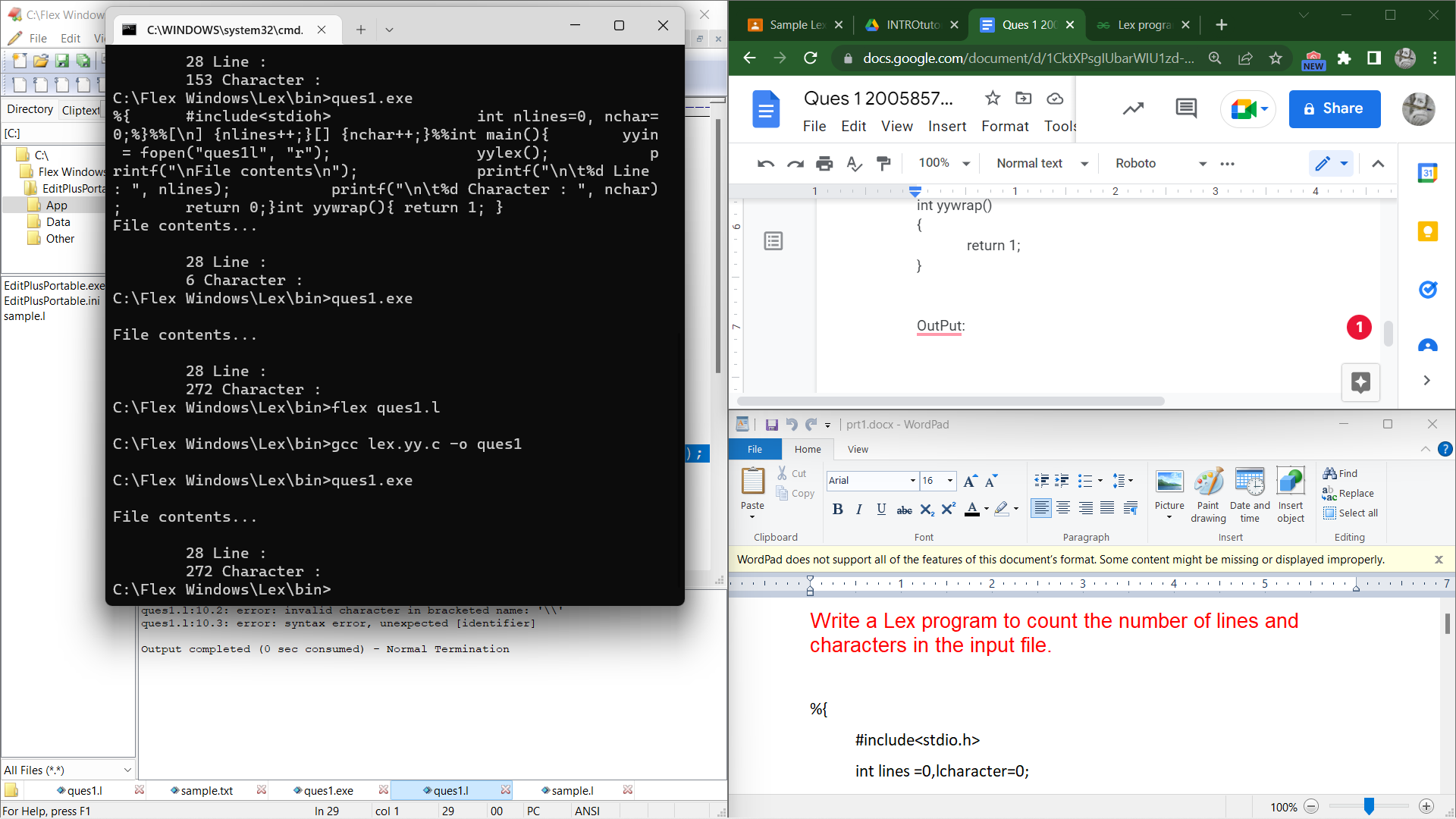
int yywrap()

{

return 1;

}

OutPut:



# Ques 2

2. Write a Lex program that implements the Caesar cipher: it replaces every letter with the one

three letters after in alphabetical order, wrapping around at Z. e.g. a is replaced by d, b by e,

and so on z by c.

%option noyywrap

%{

#include<stdio.h>

%}

%%

[A-Wa-w] {printf("%c",yytext[0]+3);}

[X-Zx-z] {printf("%c",yytext[0]-23);}

%%

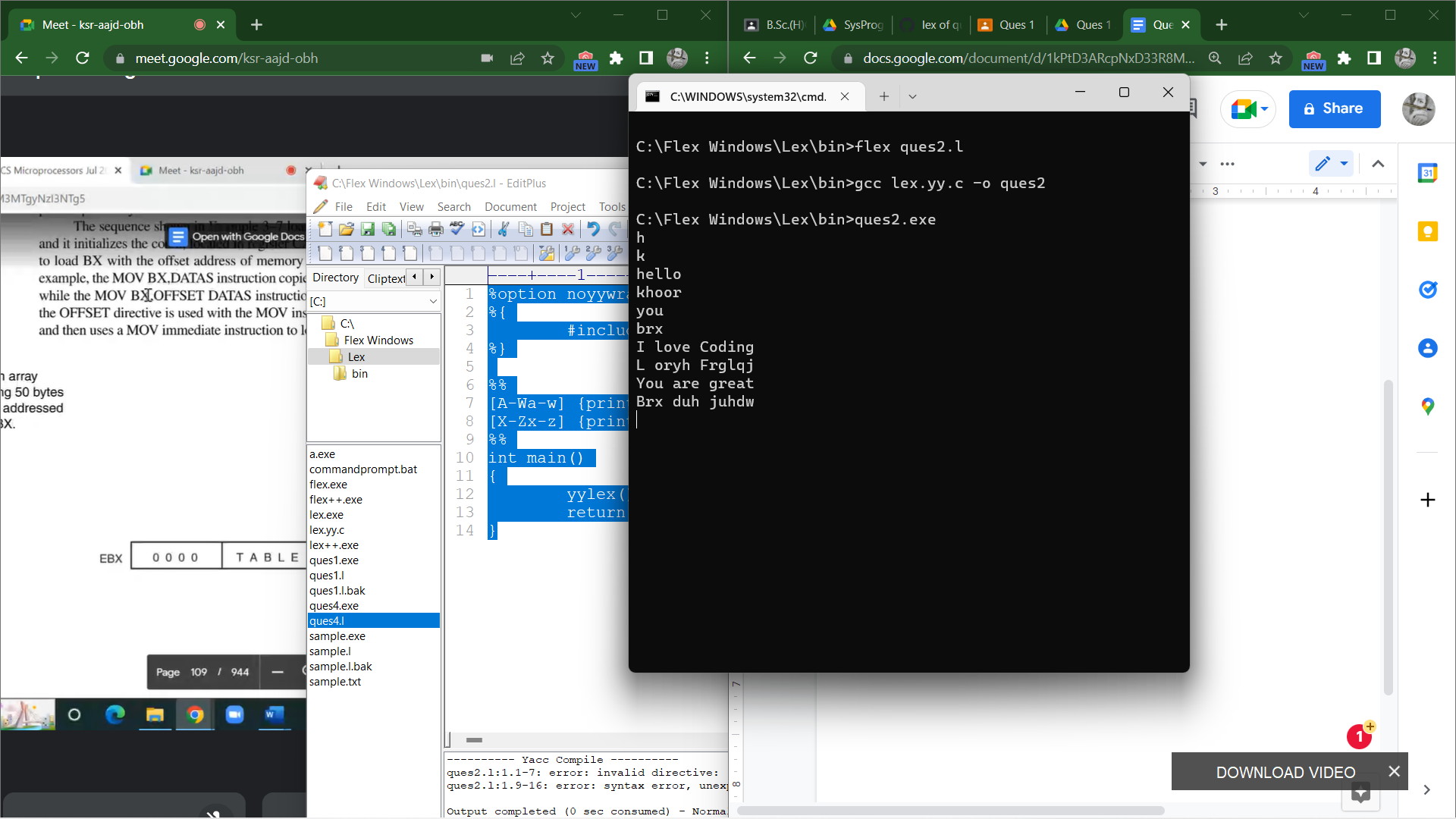
int main()

{

yylex();

return 1;

}



# Ques 3

3. Write a Lex program that finds the longest word (defined as a contiguous string of upper-

and lower-case letters) in the input.

%option noyywrap

%{

#include<stdio.h>

#include<strings.h>

int count=0;

char longest[50];

%}

%%

[A-Za-z0-9]+ { if (yyleng > count) {

count=yyleng;

strcpy(longest,yytext);

}

}

%%

int main()

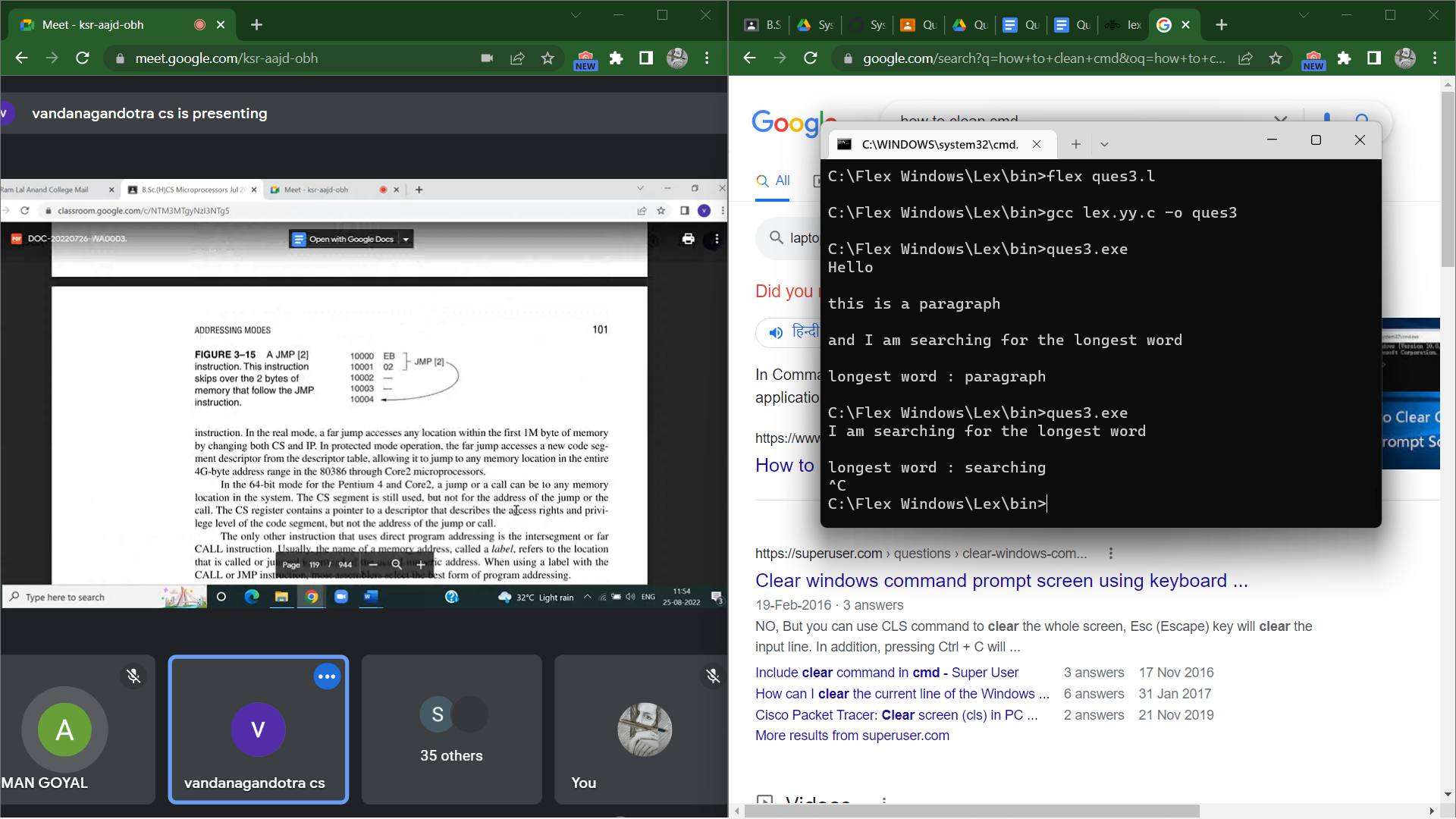
{

yylex();

printf("longest word : %s\n",longest);

return 1;

}



# Ques 4

Write a Lex program that distinguishes keywords, integers, floats, identifiers, operators, and

comments in any simple programming language.

%option noyywrap

%{

#include<stdio.h>

%}

%%

[0-9]\* {printf("Integer\n");}

[0-9]+\.[0-9]+ {printf("Float\n"); }

int|float|if|else|printf|main|exit|switch {printf("Keyword\n");}

[+|\*|/|%|&] {printf("Operators\n");}

"-" {printf("Operators\n");}

"/\*".\*"\*/" {printf("comment\n");}

[\_a-zA-Z][\_a-zA-Z0-9]{0,30} {printf("Identifier\n");}

. {printf("Invalid\n");}

%%

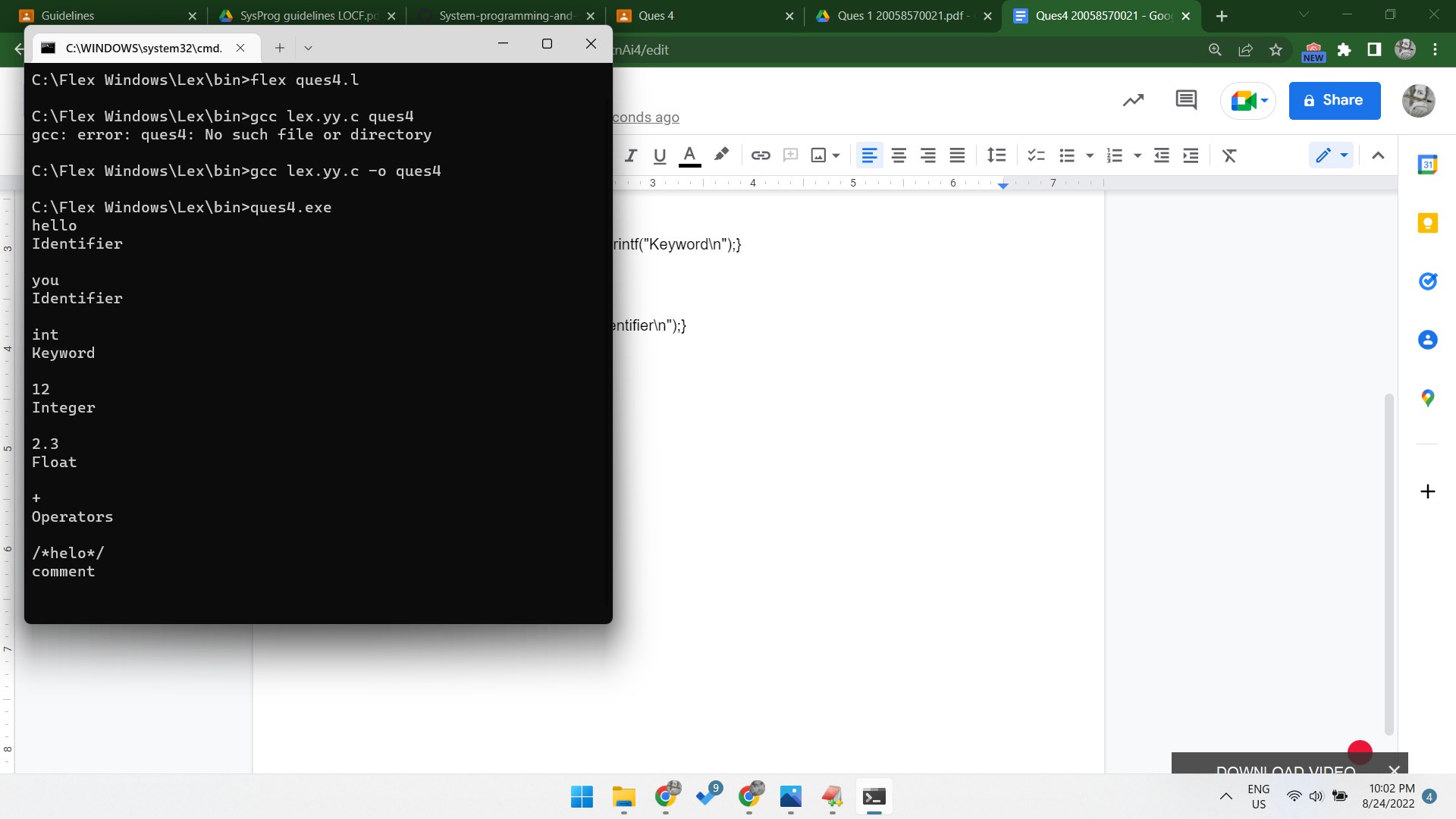
int main()

{

yylex();

return 1;

}



# Ques 5

5. Write a Lex program to count the number of identifiers in a C file.

%option noyywrap

%{

int count=0;

int spch =0;

%}

digit [0-9]

letter [A-Za-z\_]

specialChar [,|<|>|.|\_|(|)|;|$|:|%|#|?|'|&|{|}|"|^|!|\*|/|-|\|~|+|=|]

%%

([ ])int|float|char|enum|long|struct|double|void([ ]) {count++;}

{specialChar}|{digit}|{letter}|([ ])|\n {spch++;}

%%

int main()

{

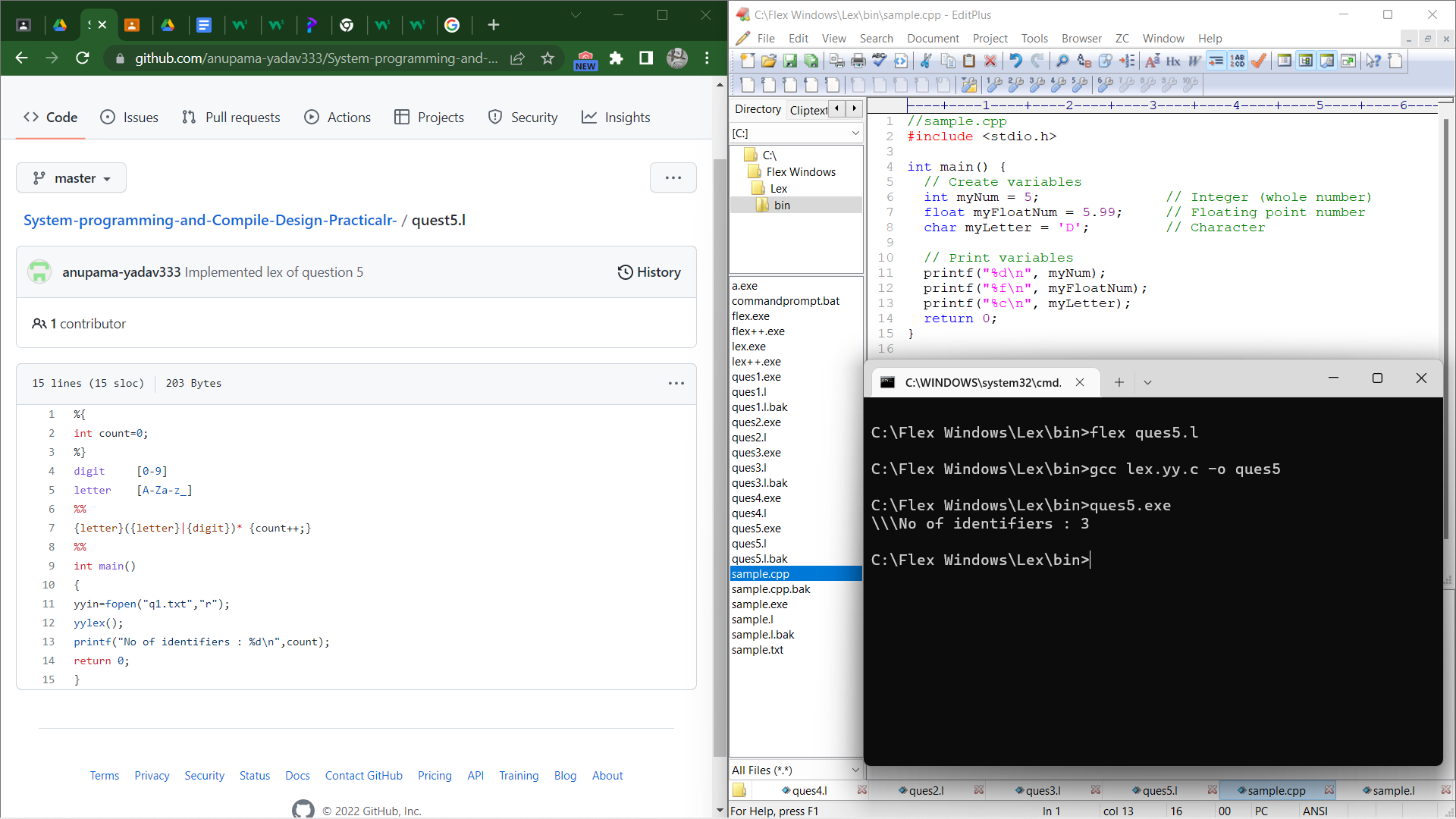
yyin=fopen("sample.cpp","r");

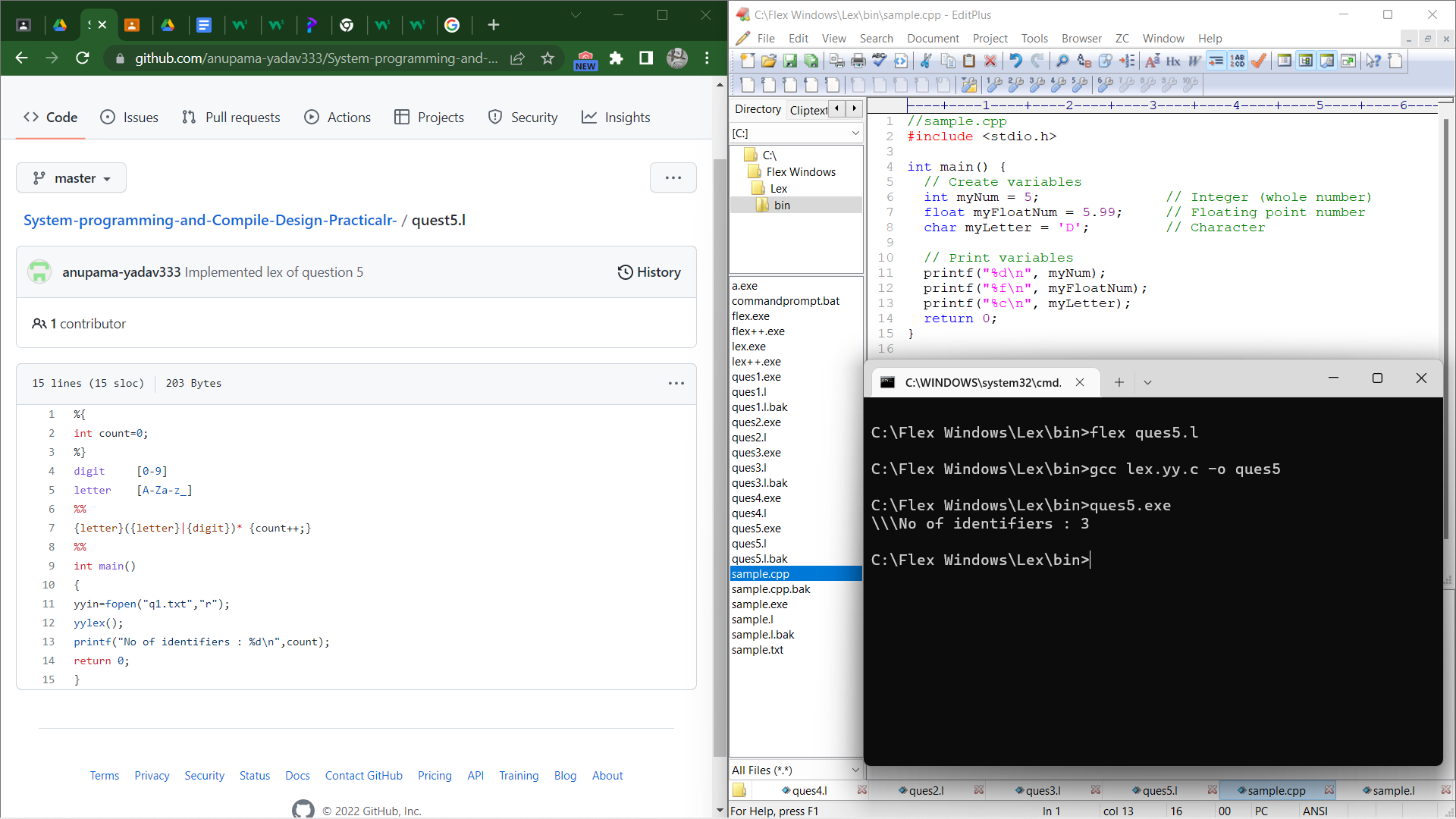
yylex();

printf("No of identifiers : %d\n",count);

return 1;

}





# Ques 6

6. Write a Lex program to count the number of words, characters, blank spaces, and lines in a C

File.

%option noyywrap

%{

#include<stdio.h>

#include<string.h>

int lines = 0, nchar = 0, nspc = 0, nwrd = 0;

%}

%%

[\n]|[.] {lines++; }

[A-Za-z|0-9]+ {nwrd++;nchar = nchar+strlen(yytext);}

([ ])|[\t|\r]+ {nspc++;}

. {nchar++;}

%%

int main()

{

yyin=fopen("sample.cpp", "r");

yylex();

printf("Number of lines : %d\n", lines);

printf("Number of spaces : %d\n", nwrd);

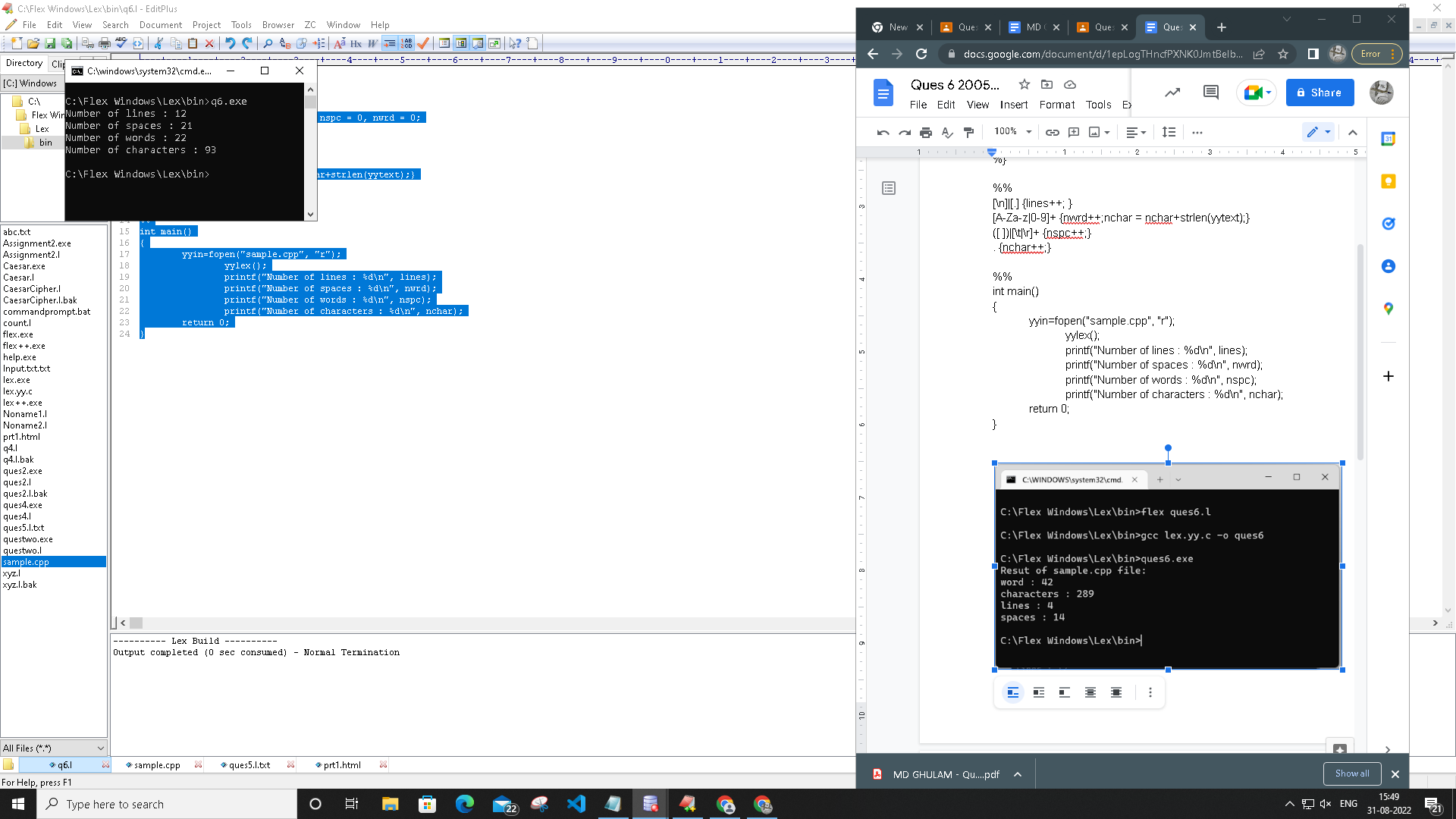
printf("Number of words : %d\n", nspc);

printf("Number of characters : %d\n", nchar);

return 0;

}





# Ques 7

Write a Lex specification program that generates a C program which takes a string “abcd”

and prints the following output.

abcd

abc

ab

a

%option noyywrap

%{

#include<stdio.h>

%}

%%

[A-Za-z]+ {int len=yyleng;

int i=len;

printf("\n");

while(i>=0)

{

int j=0;

while(j<i)

{

printf("%c",yytext[j]);

j++;

}

printf("\n");

i--;

}

}

%%

int main()

{

printf("Enter string : ");

yylex();

return 0;

}



# Ques 8

8. A program in Lex to recognize a valid arithmetic expression.

%option noyywrap

%{

#include<strings.h>

int opcount=0,intcount=0,check=1,top=0, prnt=0;;

%}

%%

['('] {check=0;}

[')'] {check=1;}

[+|\*|/|-] {opcount++; prnt=1;}

[0-9]+ {intcount++; prnt=1;}

. {printf("Invalid Input(only digits and +|-|\*|/ is valid\n");}

[\n] {

if(prnt==1)

{

if(intcount==opcount+1)

{

if(check==1)

{

printf("\nExpression is CORRECT!\n");

}

else{

printf("\n')' bracket missing from expression\n");

}

}

else

{

printf("\nExpression is INCORRECT!\n");

}

prnt=0;

opcount=0;

intcount=0;

check=1;

printf("\nEnter expression : ");

}

else

{

printf("Please, Enter your Expression or terminate this loop by pressing ctrl+c. ");

printf("\nEnter expression : ");

}

}

%%

int main()

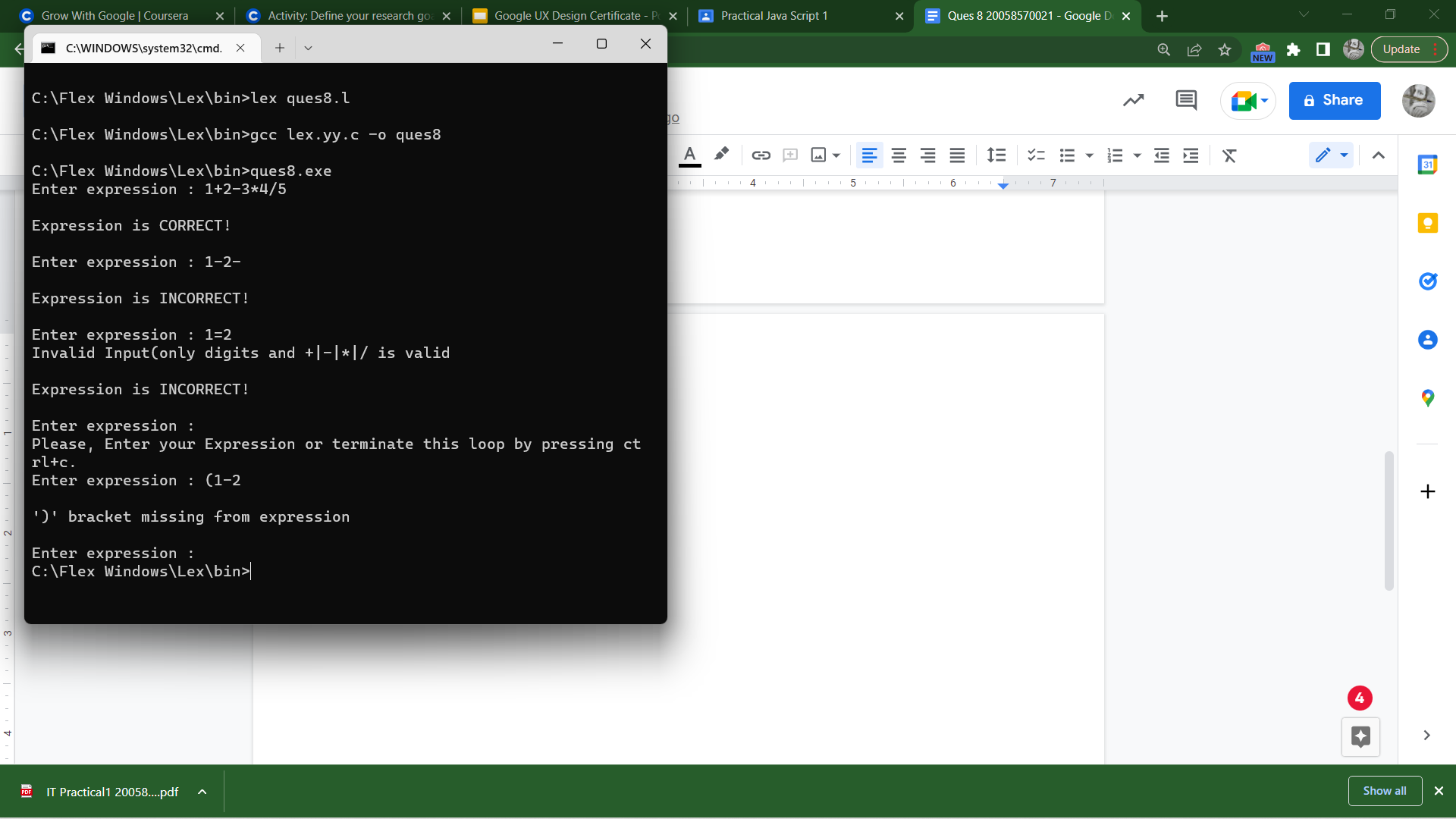
{

printf("Enter expression : ");

yylex();

return 0;

}



# Ques 9

9. Write a YACC program to find the validity of a given expression (for operators + - \* and /)

yacc1.l

%option noyywrap

%{

#include<stdio.h>

#include<stdlib.h>

#include “yacc1.tab.h"

%}

%%

[\t]+;

[0-9]+ { printf("\n %s is a valid number \n", yytext);

return NUM;}

[a-z\_]+[a-z\_0-9]\* {printf("\n%s is a valid variable\n", yytext);

return VAR;}

[+] {printf("\n %s is a valid operator\n",yytext);

return "+";}

[-] {printf("\n %s is a valid operator\n",yytext);

return "-";}

[/] {printf("\n %s is a valid operator\n",yytext);

return "/";}

[\*] {printf("\n %s is a valid operator\n",yytext);

return "\*";}

\n {return NL;}

. {return yytext[0];}

%%

yacc1.y

%{

#include "yacc1.tab.h"

%}

%token NUM VAR NL

%%

#include<stdio.h>

#include<stdlib.h>

%left '+' '-' '\*' '/';

S: S1 NL {print("\nValid Expression\n");return 0;}

S1: S1 '+' S1|S1 '-' S1|S1 '/' S1|S1 '\*' S1| '(' S1 ')'| VAR | NUM |;

%%

int main(){

printf("\nEnter an Expression: ");

yyparse();

return 0;

}

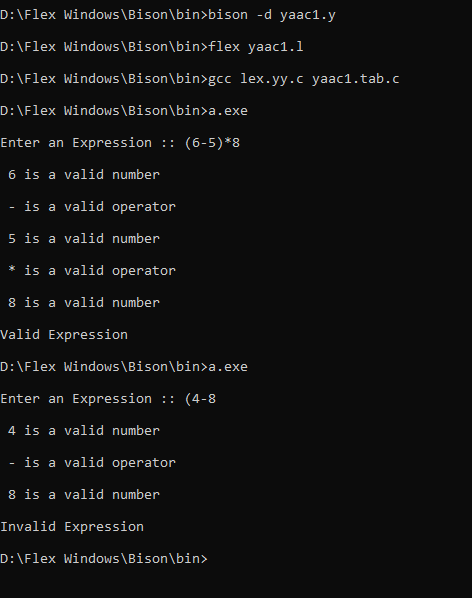
int yywrap(){}

int yyerror(){

printf("\nInvalid Expression\n");

exit(1);

}



# Ques 10

10. A Program in YACC which recognizes a valid variable which starts with letter followed by a digit. The letter should be in lowercase only.

yacc2.l

%option noyywrap

%{

#include <stdio.h>

#include <stdlib.h>

#include "yacc2.tab.h"

%}

%%

[a-z] { return L; }

[0-9] { return D; }

[ \t\n]+ { ; }

.{ return yytext[0]; }

%%

yacc2.y

%{

#include <stdio.h>

#include <stdlib.h>

#include "yacc2.tab.h"

%}

%token D L

%%

S : L D { printf("VALID IDENTIFIER\n"); }

;

%%

int main()

{

printf("\n Enter identifier\n");

yyparse();

return 0;

}

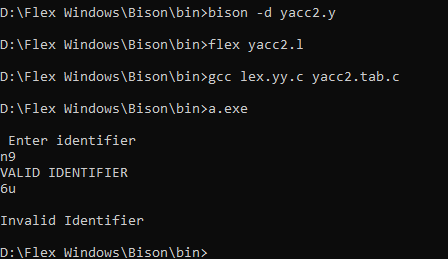
int yywrap(){}

int yyerror(){

tf("\nInvalid Identifier\n");

exit(1);

}



# Ques 11

11. A Program in YACC to evaluate an expression (simple calculator program for addition and subtraction, multiplication, division).

**yaac3.l file**

%{

#include <stdio.h>

#include <stdlib.h>

#include "yaac3.tab.h"

int yylval;

%}

%%

[0-9]+ { yylval = atoi(yytext); return NUM;}

[\t]+ ;

\n {return 0;}

. {return yytext[0];}

%%

**yaac3.y file**

%{

#include <stdio.h>

#include <stdlib.h>

#include "yaac3.tab.h"

%}

%token NUM

%left '+' '-'

%left '/' '\*'

%left '(' ')'

%%

expr: e { printf("Result is :: %d\n",$$); return 0;}

e: e '+' e{$$ = $1+$3;}

|e '-' e{$$ = $1-$3;}

|e '\*' e{$$ = $1\*$3;}

|e '/' e {

if($3==0){

printf("\nDivision By Zero\n");

printf("Result is :: Undefined");

return 0; }

else {$$ = $1/$3;}

}

|'(' e ')'{$$ = $2;}

|NUM {$$ = $1;}

%%

int main(){

printf("\nEnter the arithmetic expression ::");

yyparse();

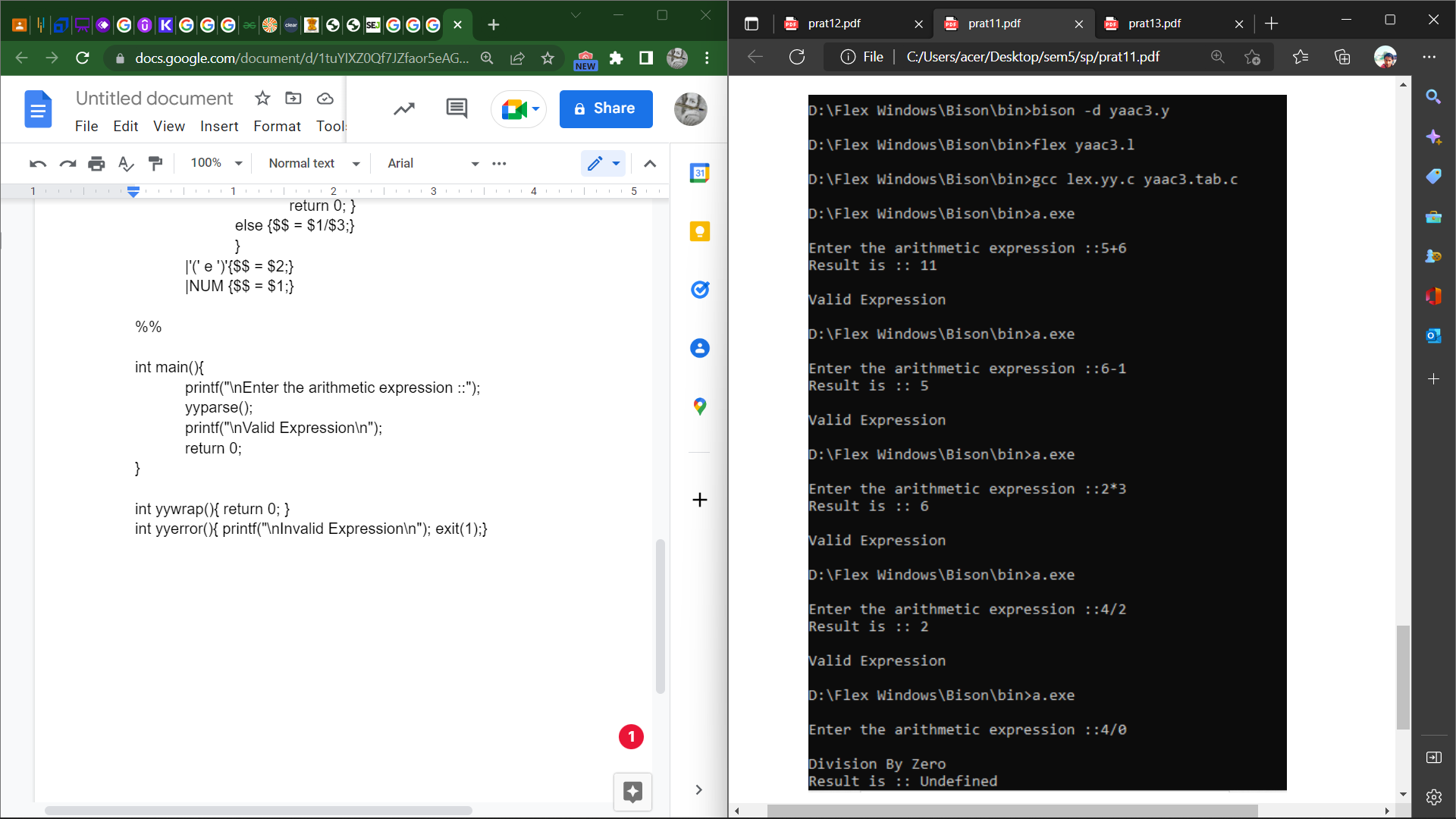
printf("\nValid Expression\n");

return 0;

}

int yywrap(){ return 0; }

int yyerror(){ printf("\nInvalid Expression\n"); exit(1);}



# Ques 12

12. Program in YACC to recognize the strings “ab”, “aabb”, ”aaabbb”,… of the language (anbn, n>=1).

**yacc4.l file**

%option noyywrap

%{

#include <stdio.h>

#include <stdlib.h>

#include "yacc4.tab.h"

%}

%%

[a] { return A; }

[b] { return B; }

[ |\n|\t] { return yytext[0]; }

. { return yytext[0]; }

%%

**yacc4.y file**

%{

#include <stdio.h>

#include <stdlib.h>

#include "yacc4.tab.h"

%}

%token A B

%%

S : E '\n' { printf("VALID STRING\n"); exit(0); };

E: A E B

| A B ;

%%

int main(){

printf("\nEnter the string :: ");

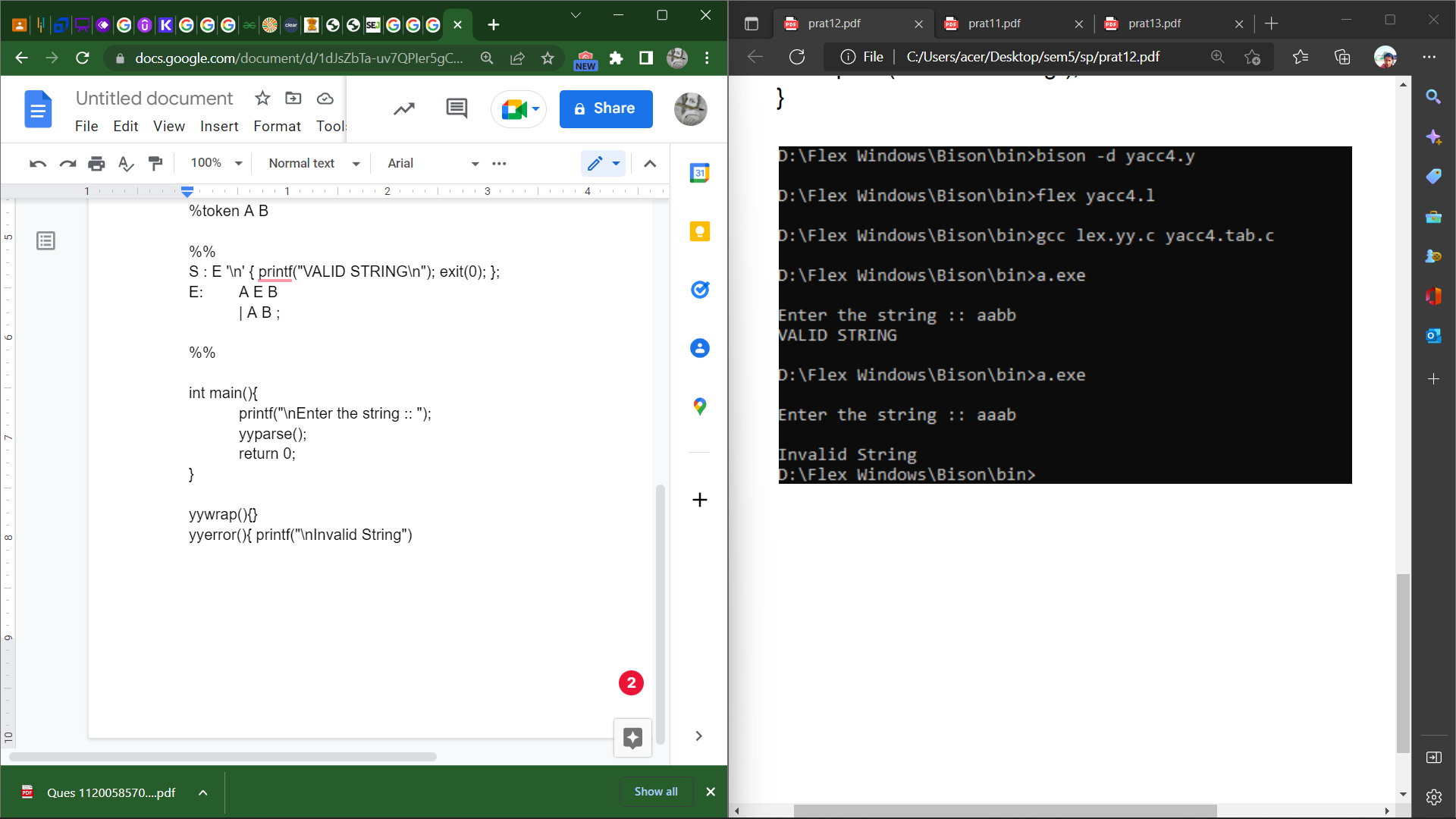
yyparse();

return 0;

}

yywrap(){}

yyerror(){ printf("\nInvalid String")



# Ques 13

13. Program in YACC to recognize the language (anb, n>=10). (Output to say input is valid or not)

yaac5.l file

%{

#include <stdio.h>

#include <stdlib.h>

#include "yaac5.tab.h"

%}

%%

[a] {return A;}

[b] {return B;}

\n {return NL;}

. {return yytext[0];}

%%

**yaac5.y file**

%{

#include <stdio.h>

#include <stdlib.h>

#include "yaac5.tab.h"

%}

%token A B NL

%%

S : A A A A A A A A A A S1 B NL

{ printf("\nValid String \n"); return 0;}

S1 : A S1

|;

%%

int main(){

printf("\nEnter a String :: ");

yyparse();

}

yywrap(){}

yyerror(){ printf("\nInvalid String\n"); return 0;}

