RAM LAL ANAND COLLEGE

UNIVERSITY OF DELHI

DEPARTMENT OF COMPUTER 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

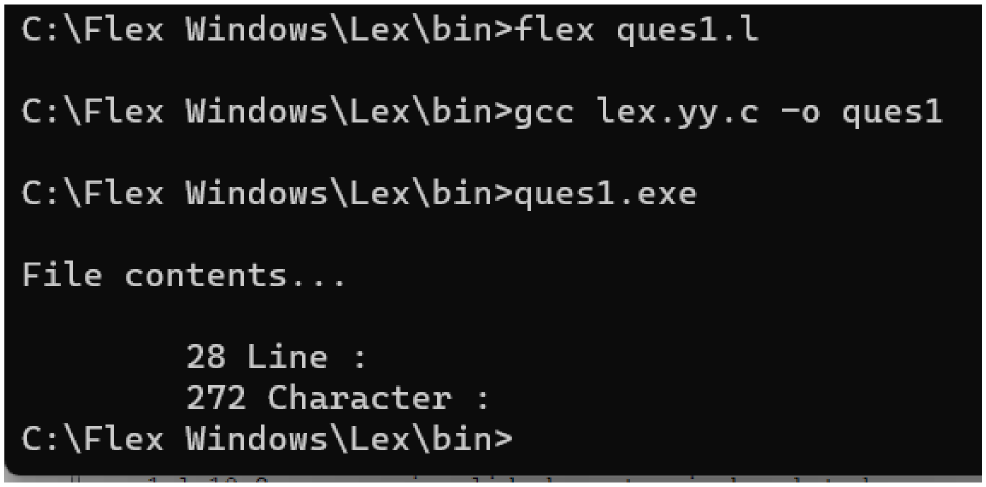
Ques 1

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:

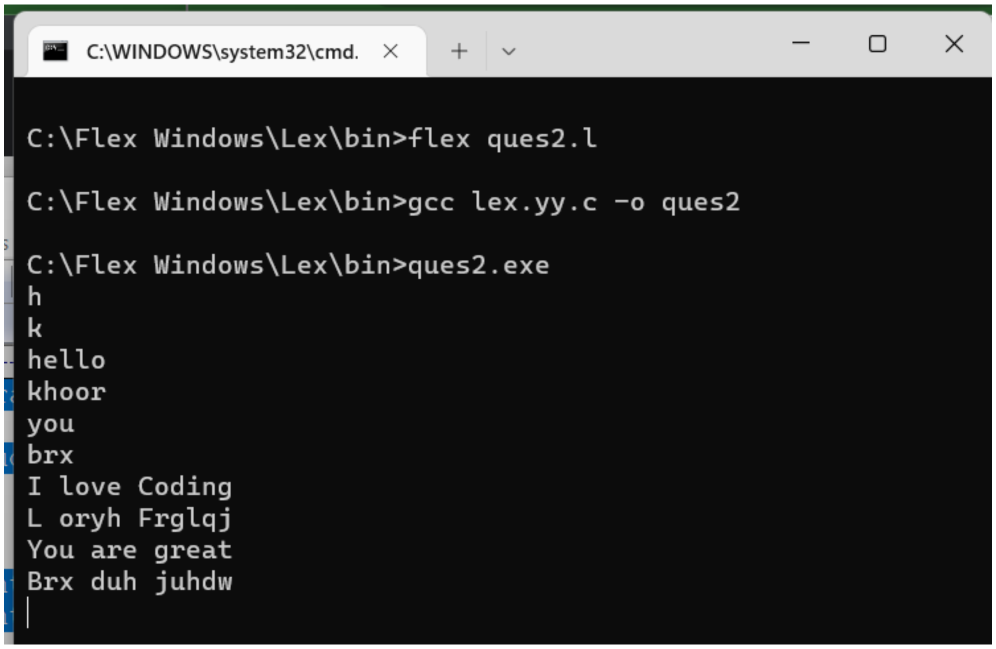


```
C:\Flex Windows\Lex\bin>flex ques1.l

C:\Flex Windows\Lex\bin>gcc lex.yy.c -o ques1

C:\Flex Windows\Lex\bin>ques1.exe

File contents...

        28 Line :
        272 Character :
C:\Flex Windows\Lex\bin>
```

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;
}
```



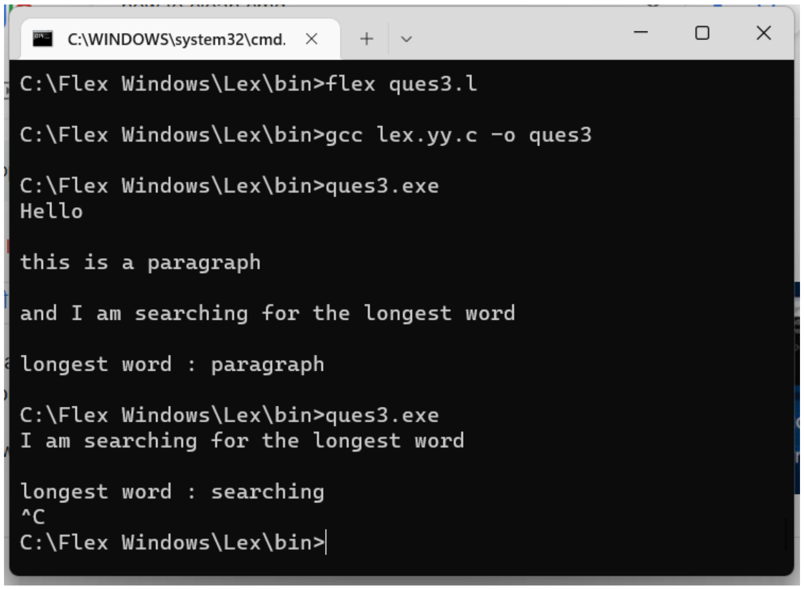
```
C:\Flex Windows\Lex\bin>flex ques2.l

C:\Flex Windows\Lex\bin>gcc lex.yy.c -o ques2

C:\Flex Windows\Lex\bin>ques2.exe
h
k
hello
khoor
you
brx
I love Coding
L oryh Frglqj
You are great
Brx duh juhdw
```

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;
}
```



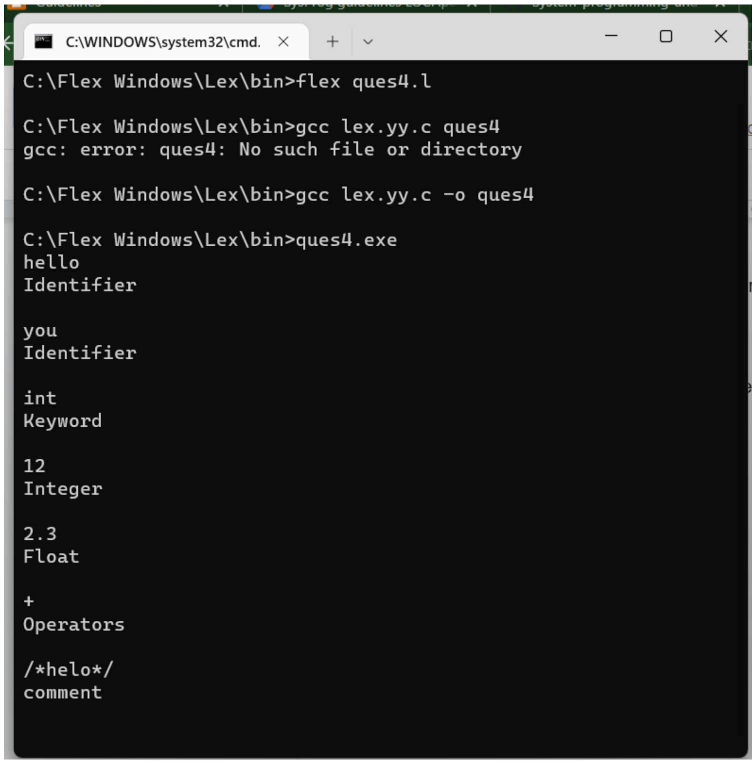
```
C:\Flex Windows\Lex\bin>flex ques3.l

C:\Flex Windows\Lex\bin>gcc lex.yy.c -o ques3

C:\Flex Windows\Lex\bin>ques3.exe
Hello

this is a paragraph

and I am searching for the longest word

longest word : paragraph

C:\Flex Windows\Lex\bin>ques3.exe
I am searching for the longest word

longest word : searching
^C
C:\Flex Windows\Lex\bin>
```

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;
}
```



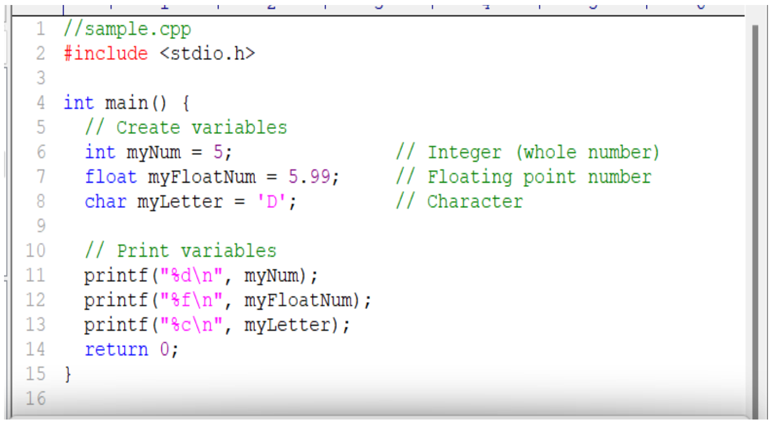
```
C:\Flex Windows\Lex\bin>flex ques4.l

C:\Flex Windows\Lex\bin>gcc lex.yy.c ques4
gcc: error: ques4: No such file or directory

C:\Flex Windows\Lex\bin>gcc lex.yy.c -o ques4

C:\Flex Windows\Lex\bin>ques4.exe
hello
Identifier

you
Identifier

int
Keyword

12
Integer

2.3
Float

+
Operators

/*helo*/
comment
```

Ques 5

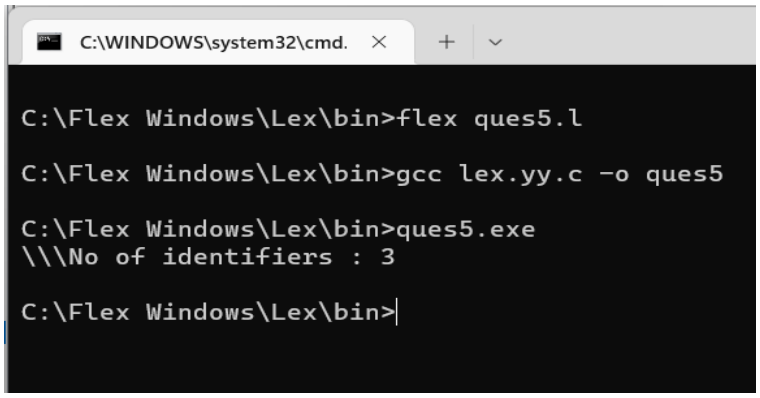
5. Write a Lex program to count the number of identifiers in a C file.

```
%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;
}
```



```
//sample.cpp
#include <stdio.h>

int main() {
  // Create variables
  int myNum = 5;               // Integer (whole number)
  float myFloatNum = 5.99;     // Floating point number
  char myLetter = 'D';         // Character

  // Print variables
  printf("%d\n", myNum);
  printf("%f\n", myFloatNum);
  printf("%c\n", myLetter);
  return 0;
}

```



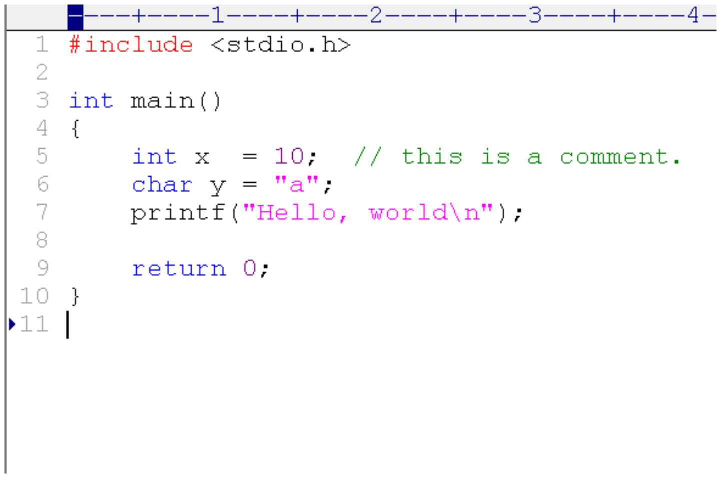
```
C:\Flex Windows\Lex\bin>flex ques5.l

C:\Flex Windows\Lex\bin>gcc lex.yy.c -o ques5

C:\Flex Windows\Lex\bin>ques5.exe
\\\No of identifiers : 3

C:\Flex Windows\Lex\bin>
```

Ques 6

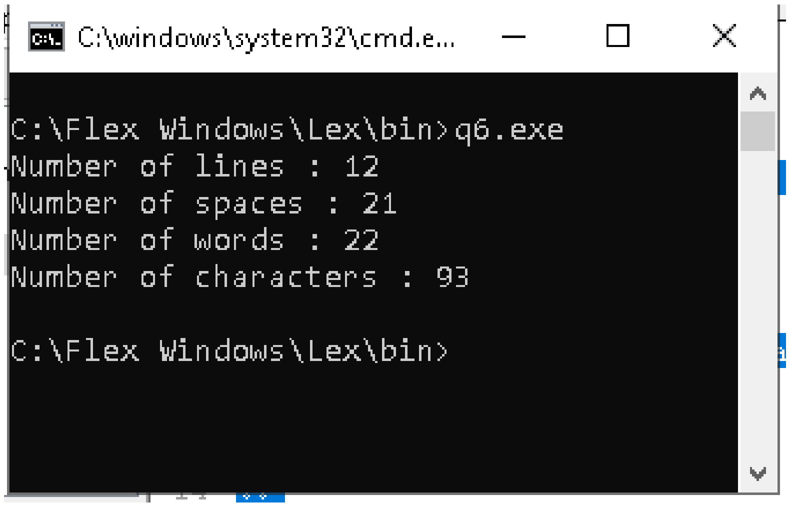
6. Write a Lex program to count the number of words, characters, blank spaces, and lines in a C

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;
}
```



```
#include <stdio.h>

int main()
{
    int x  = 10;  // this is a comment.
    char y = "a";
    printf("Hello, world\n");

    return 0;
}

```



```
C:\Flex Windows\Lex\bin>q6.exe
Number of lines : 12
Number of spaces : 21
Number of words : 22
Number of characters : 93

C:\Flex Windows\Lex\bin>
```

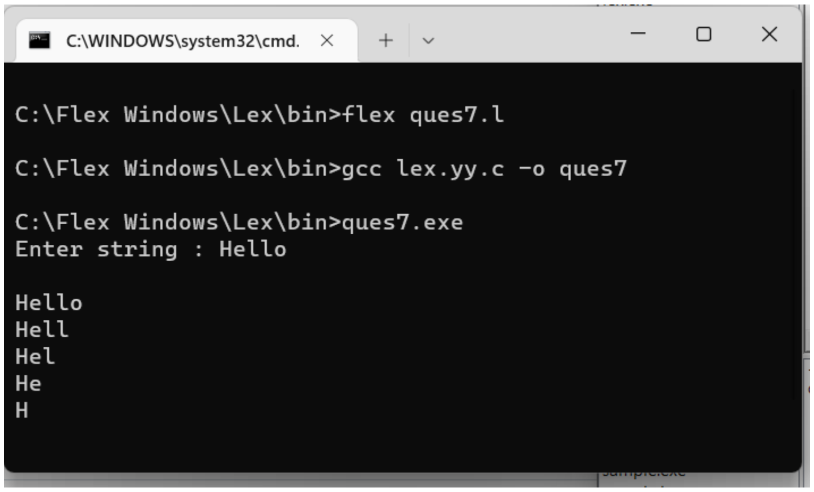
Ques 7

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;
}
```

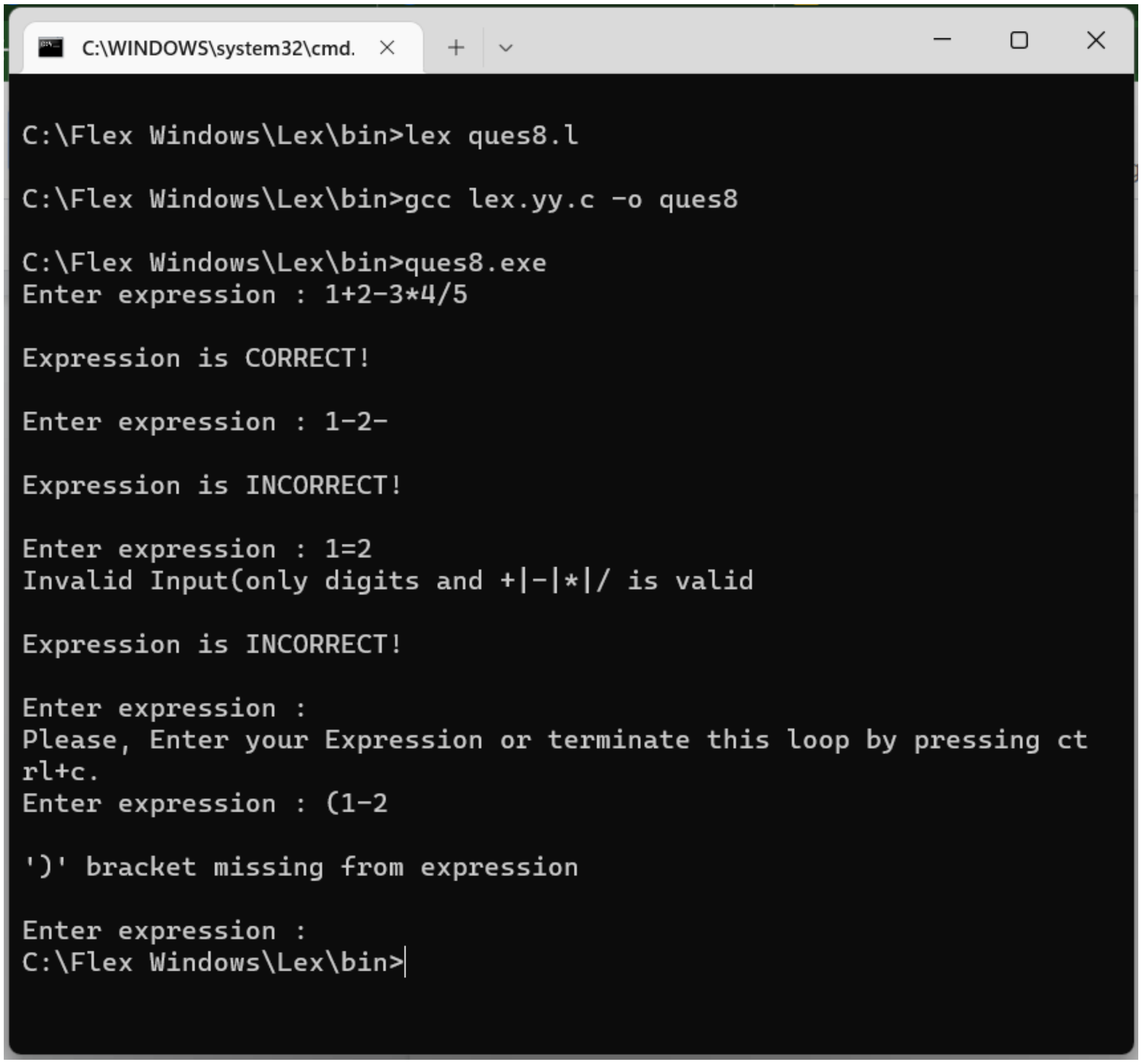


```
C:\Flex Windows\Lex\bin>flex ques7.l

C:\Flex Windows\Lex\bin>gcc lex.yy.c -o ques7

C:\Flex Windows\Lex\bin>ques7.exe
Enter string : Hello

Hello
Hell
Hel
He
H
```

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;
}
```



```
C:\Flex Windows\Lex\bin>lex ques8.l

C:\Flex Windows\Lex\bin>gcc lex.yy.c -o ques8

C:\Flex Windows\Lex\bin>ques8.exe
Enter expression : 1+2-3*4/5

Expression is CORRECT!

Enter expression : 1-2-

Expression is INCORRECT!

Enter expression : 1=2
Invalid Input(only digits and +|-|*|/ is valid

Expression is INCORRECT!

Enter expression :
Please, Enter your Expression or terminate this loop by pressing ct
rl+c.
Enter expression : (1-2

')' bracket missing from expression

Enter expression :
C:\Flex Windows\Lex\bin>
```

Ques 9

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

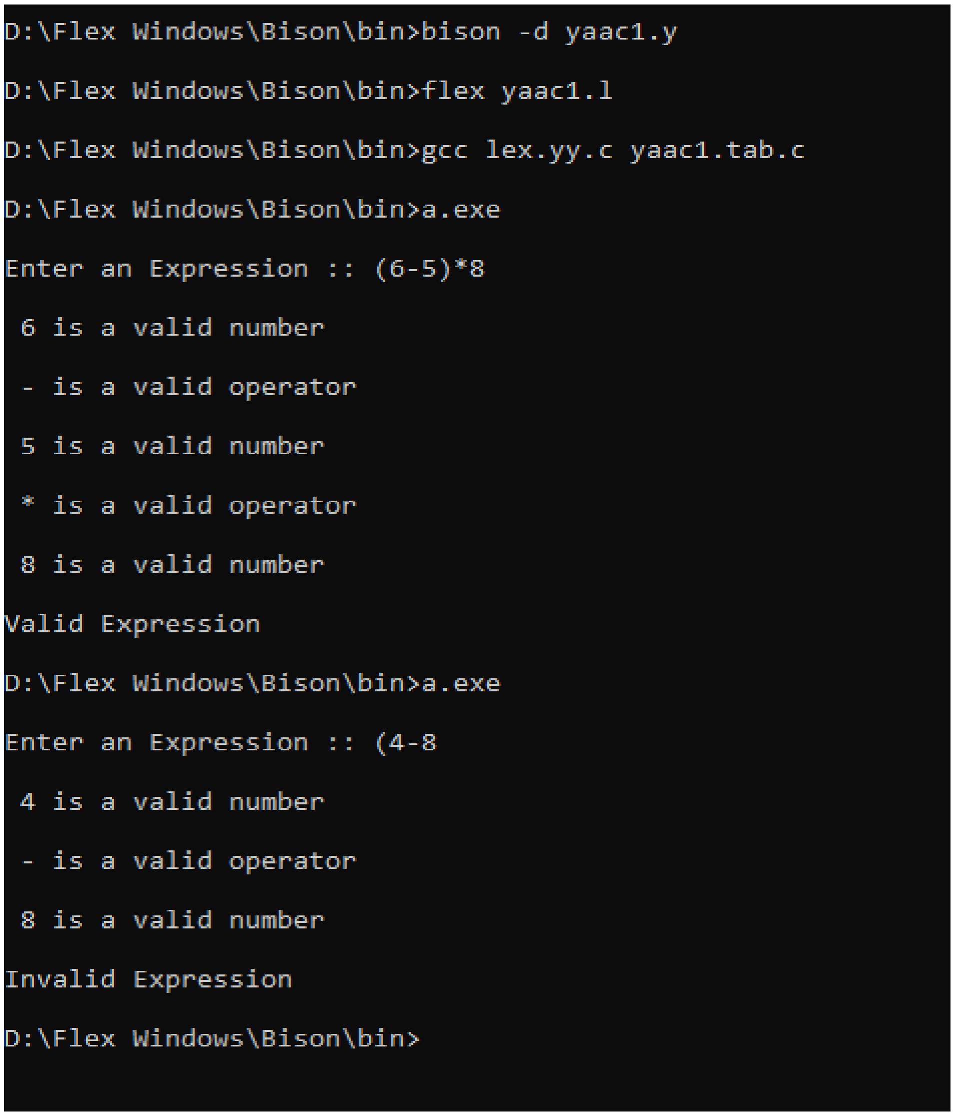
yacc1.l

```
%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);
}
```



```
D:\Flex Windows\Bison\bin>bison -d yaac1.y

D:\Flex Windows\Bison\bin>flex yaac1.l

D:\Flex Windows\Bison\bin>gcc lex.yy.c yaac1.tab.c

D:\Flex Windows\Bison\bin>a.exe

Enter an Expression :: (6-5)*8

 6 is a valid number

 - is a valid operator

 5 is a valid number

 * is a valid operator

 8 is a valid number

Valid Expression

D:\Flex Windows\Bison\bin>a.exe

Enter an Expression :: (4-8

 4 is a valid number

 - is a valid operator

 8 is a valid number

Invalid Expression

D:\Flex Windows\Bison\bin>
```

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.

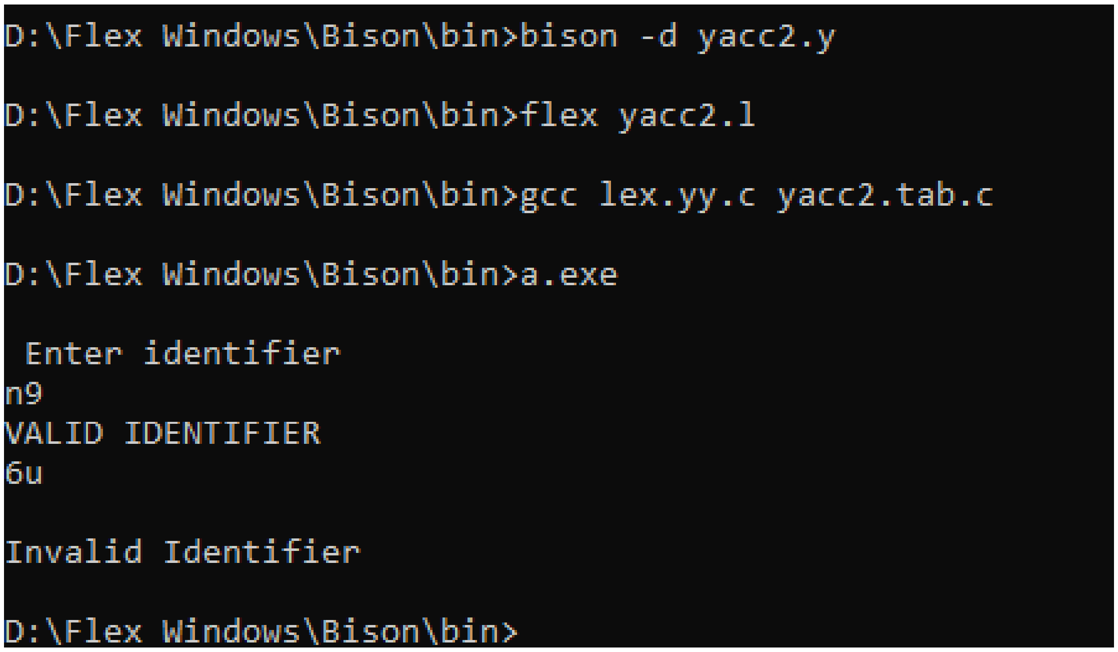
yacc2.l

```
%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);
}
```



```
D:\Flex Windows\Bison\bin>bison -d yacc2.y

D:\Flex Windows\Bison\bin>flex yacc2.l

D:\Flex Windows\Bison\bin>gcc lex.yy.c yacc2.tab.c

D:\Flex Windows\Bison\bin>a.exe

 Enter identifier
n9
VALID IDENTIFIER
6u

Invalid Identifier

D:\Flex Windows\Bison\bin>
```

Ques 11

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

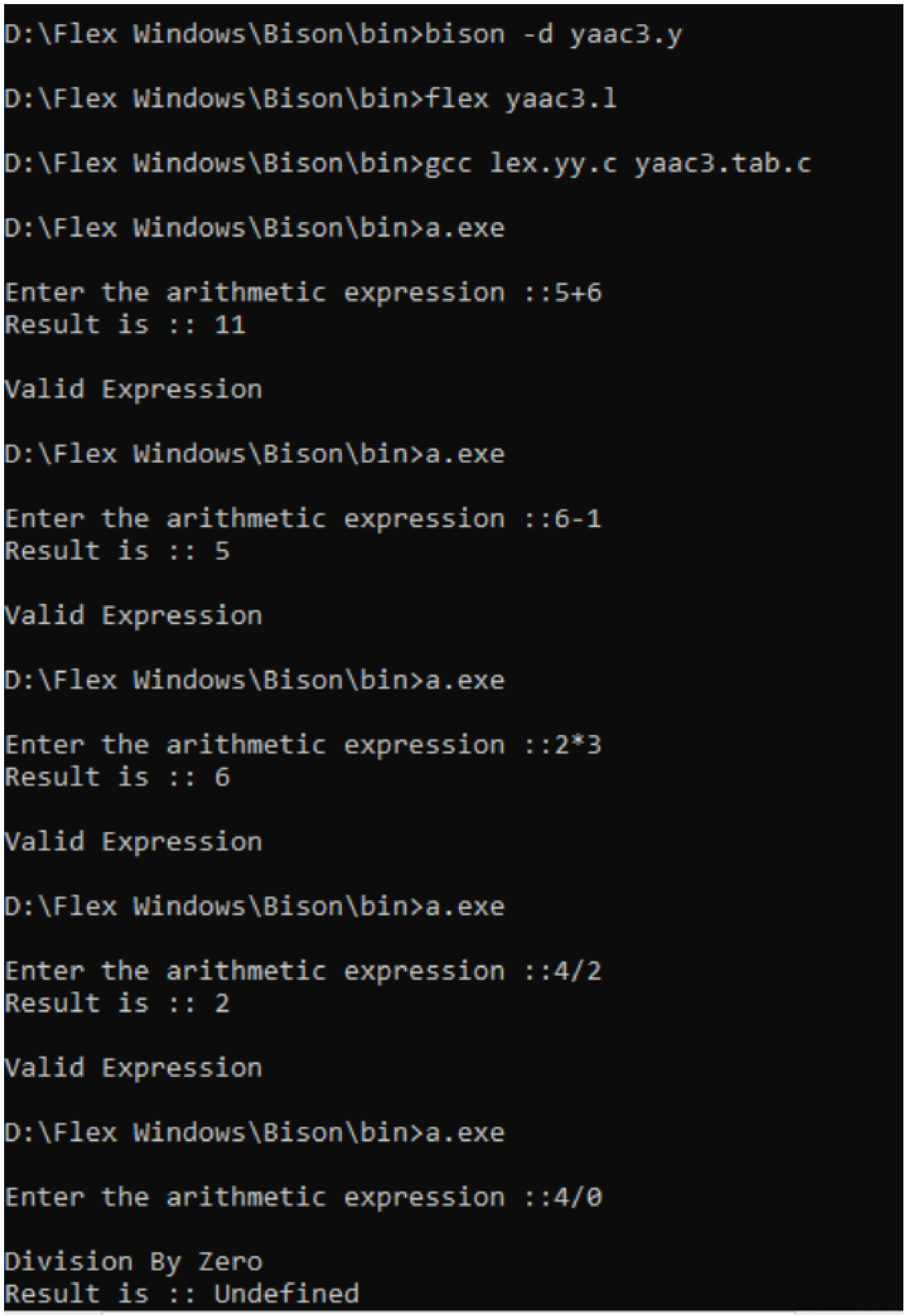
yaac3.l file

```
 %{
        #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);}
```



```
D:\Flex Windows\Bison\bin>bison -d yaac3.y

D:\Flex Windows\Bison\bin>flex yaac3.l

D:\Flex Windows\Bison\bin>gcc lex.yy.c yaac3.tab.c

D:\Flex Windows\Bison\bin>a.exe

Enter the arithmetic expression ::5+6
Result is :: 11

Valid Expression

D:\Flex Windows\Bison\bin>a.exe

Enter the arithmetic expression ::6-1
Result is :: 5

Valid Expression

D:\Flex Windows\Bison\bin>a.exe

Enter the arithmetic expression ::2*3
Result is :: 6

Valid Expression

D:\Flex Windows\Bison\bin>a.exe

Enter the arithmetic expression ::4/2
Result is :: 2

Valid Expression

D:\Flex Windows\Bison\bin>a.exe

Enter the arithmetic expression ::4/0

Division By Zero
Result is :: Undefined
```

Ques 12

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

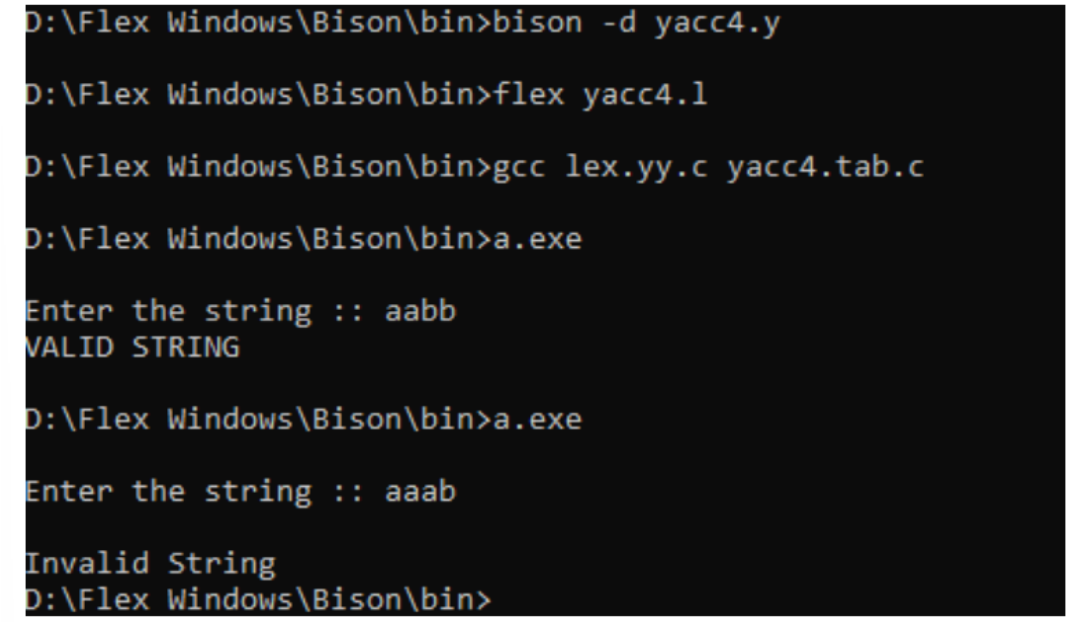
yacc4.l file

```
%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")



```
D:\Flex Windows\Bison\bin>bison -d yacc4.y

D:\Flex Windows\Bison\bin>flex yacc4.l

D:\Flex Windows\Bison\bin>gcc lex.yy.c yacc4.tab.c

D:\Flex Windows\Bison\bin>a.exe

Enter the string :: aabb
VALID STRING

D:\Flex Windows\Bison\bin>a.exe

Enter the string :: aaab

Invalid String
D:\Flex Windows\Bison\bin>
```

Ques 13

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

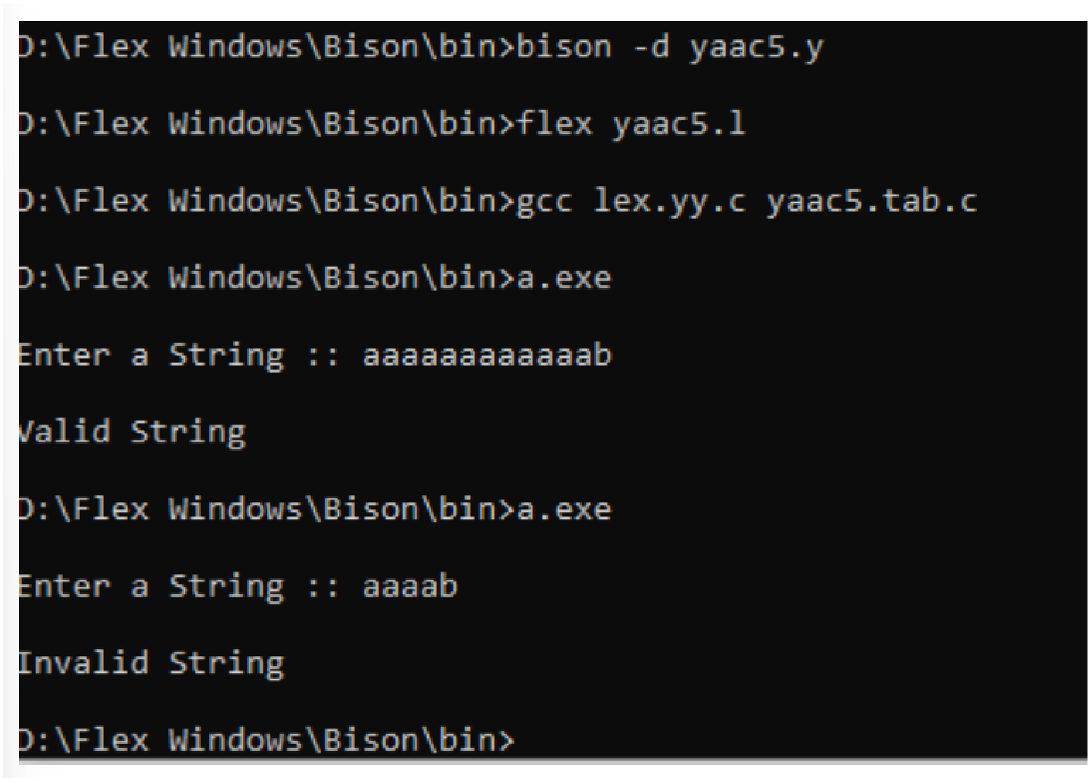
yaac5.l file

```
%{
        #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;}
```



```
D:\Flex Windows\Bison\bin>bison -d yaac5.y

D:\Flex Windows\Bison\bin>flex yaac5.l

D:\Flex Windows\Bison\bin>gcc lex.yy.c yaac5.tab.c

D:\Flex Windows\Bison\bin>a.exe

Enter a String :: aaaaaaaaaaaab

Valid String

D:\Flex Windows\Bison\bin>a.exe

Enter a String :: aaaab

Invalid String

D:\Flex Windows\Bison\bin>
```