**Part B- Compiler Design Lab- Question Bank**

1. Write a C / C++ program to accept a C program and do error detection & correction for the following. **(CO1)**
Check for un-terminated string constant in the input C program. i.e A string constant begins with double quotes and extends for more than one line. Intimate the error line numbers and the corrective actions to user.

q1.c

```
#include <stdio.h>
#include <string.h>
int main()
{
    FILE *fp;
    char line[100];
    fp = fopen("text1.txt", "r");
    if (!fp)
    {
        printf("File cant be opened\n");
        return 0;
    }
    printf("File opened correctly!\n");

    for (int lineno = 1; fgets(line, sizeof(line), fp); lineno++)
    {
        int found = 0, flag = 0;
        for (int i = 0; i < strlen(line); i++)
            if (line[i] == '"')
            {
                flag = !flag;
                found = 1;
            }

        if (found)
        {
            if (flag)
                printf("\n Unterminated string in line %d. String Has
to be closed", lineno);
            else
                printf("\n String usage in line %d is validated!",
lineno);
        }
    }
    return 0;
}
```

Text1.txt

```
#include<stdio.h>
```

```
#include<conio.h>
int s[35]="gh";
 void main(){
int a; char
c[10]="msrit",f[]="lk;
strlen("hjkl); a=a+/*b;
}
```

Output

```
File opened correctly!

 String usage in line 3 is validated!
 Unterminated string in line 6. String Has to be closed
 Unterminated string in line 7. String Has to be closed%
```

2. Write a C / C++ program to accept a C program and do error detection & correction for the following. **(CO1)**

Check for un- terminated multi line comment statement in your C program.

q2.c

```
#include <stdio.h>
#include <string.h>
int main()
{
	FILE *fp;
	char line[100];
	int isOpen = 0, openlineno;

	fp = fopen("text2.txt", "r");
	if (!fp)
	{
		printf("File cant be opened\n");
		return 0;
	}
	printf("File opened correctly!\n");

	for (int lineno = 1; fgets(line, sizeof(line), fp); lineno++)
	{
		if (isOpen)
			printf("\n%s", line);
		if (strstr(line, "/*") && !isOpen)
		{
			isOpen = 1;
			openlineno = lineno;
			printf("\n%s", line);
		}
		if (strstr(line, "*/") && isOpen)
			printf("\nComment is displayed above!\nComment opened in line no %d and closed in line no %d", openlineno, lineno);
	}
	if (isOpen)
```

```
		printf("\nUnterminated comment in begin in line no %d. It Has to be closed", openlineno);
	return 0;
}
```

text2.txt

```
#include<stdio.h>
#include<conio.h>
/*/*dfgdfgddfgdfg*/
int s[35]="gh";
void main(){
	int a;
	/*
		char c[10]="msrit",f[]="lk;
	*/
	strlen("hjkl);
	/*dgdfgdfg*/
	a=a+b; /*
	fsdgdgds
	sdgfsd sdfsdf
}
```

Output-

```
File opened correctly!

/*dgdfgdfg*/

Comment is displayed above!
Comment opened in line no 48 and closed in line no 48
a=a+b; /*

fsdgdgds

sdgfsd sdfsdf

}

Unterminated comment in begin in line no 48. It Has to be closed%
```

3. Write a Lex program to accept a C program and do error detection & correction for the following. **(CO1)**

Check for un-terminated string constant in the input C program. i.e A string constant begins with double quotes and extends for more than one line. Intimate the error line numbers and the corrective actions to user.

Q3.l

```
%{

#include <stdio.h>

int c = 1;
```

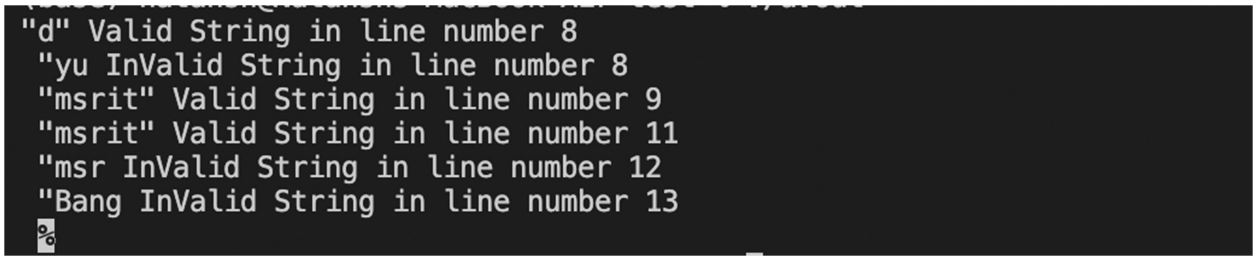
```
%}

%option noyywrap

%%
\n { c++; }
\"[a-zA-Z0-9]*\" {
    ECHO;
    printf(" Valid String in line number %d\n ", c);
}
\"[a-zA-Z0-9]* {
    ECHO;
    printf(" InValid String in line number %d\n ",c);
}
. ;
%%
int main()
{
    yyin = fopen("text3.txt", "r");
    yylex();
    fclose(yyin);
    return 0;
}
```

Text3.txt

```
#include<stdio.h>
```

```
#include<conio.h>

#include<string.h>

void main()

{

int a,b,h;

a=a+b;

 char d[20]="d",h[67]="yu ;

 char c[10]="msrit";

a=a+/b+h;

strlen("msrit");

strlen("msr);

strcpy(c,"Bang

alore);

 b=b+*; }
```

Output-




4. Write a Lex program to accept a C program and do error detection & correction for the following.(CO1) Check for valid arithmetic expressions in the input C program. Report the errors in the statements to user.

Q4.l-

```
%{

#include<stdio.h>
```

```
int c=1;
%}

%option noyywrap
operator [-+*/]
identifier [a-zA-Z_][a-zA-Z0-9_]*
number [0-9]+(\.[0-9])?[0-9]*
expression ({identifier}|{number}){operator}({identifier}|{number})

%%
\n { c++; }
^#.+ ;
^(int\s|float\s|char\s).+ ;
(void|int)\smain\(\) ;
{identifier}=({expression}+;) {
   printf("Valid expression in line no : %d\t",c);
   ECHO;
   printf("\n");
}
{identifier}=({number};|{identifier};) {
   printf("Valid expression in line no : %d\t",c);
   ECHO;
   printf("\n");
}
({number}|([0-9]*[a-zA-Z0-9-]+))={expression}+ {
```

```
    printf("Invalid expression in line no : %d; Lvalue should satisfy the identifier rules\t",c);

    ECHO;

    printf("\n");

}

{identifier}=; {

    printf("Invalid expression in line no : %d; R-value required; Expression is needed at right hand side of assignment operation\t",c);

    ECHO;

    printf("\n");

}

{operator}{operator}+ {

    printf("Invalid expression in line no: %d; More than one operator can't be used in expression consequetively",c);

    ECHO;

    printf("\n");

}

. ;

%%


int main() {

    yyin=fopen("text4.txt","r");

    yylex();

    fclose(yyin);

    return 0;

}
```

Text4.txt-

```
#include<stdio.h>

#include<conio.h>

#include<string.h>

int  main()

{

int a=1s,b,h;

a=a+b;

a=a+/b+h;

1a=7+j-;

a=;

b=b+*;

 }
```

Output-

```
Valid expression in line no : 7 a=a+b;
Invalid expression in line no: 8; More than one operator can't be used in expression consequetively+/
Invalid expression in line no : 9; Lvalue should satisfy the identifier rules   1a=7+j
Invalid expression in line no : 10; R-value required; Expression is needed at right hand side of assign
ment operation  a=;
Invalid expression in line no: 11; More than one operator can't be used in expression consequetively++
```

5. Write a Lex program to accept a C program and do the following error detection & correction.(CO1) Check for the valid usages of numerical constants in the input C program. Intimate the invalid usages to user.

Q5.l

```
%{

#include<stdio.h>

int c=1;

%}


%option noyywrap

number [0-9]+(\.[0-9])?[0-9]*
```

```
invalid [0-9]+(".")[0-9]*((".")[0-9]*)+


%%
\n { c++; }
{number} {
   printf("\nValid number in line number %d : ",c);
   ECHO;
   printf("\n");
}
{number}[a-zA-Z0-9_]+ {
   printf("\nInvalid number in line number %d: Number followed with alphabets is invalid",c);
   ECHO;
   printf("\n");
}
{invalid} {
   printf("\nInvalid number in line number %d: Number with more than one decimal point sis invalid",c);
   ECHO;
   printf("\n");
}
. ;
%%
int main()
{
   yyin = fopen("text5.txt","r");
   yylex();
   fclose(yyin);
```

```
    return 0;
}
```

Text5.txt

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{ int a=56;
a=1b; a=a+5h;
a=a+4.5+5.
6.6;
}
```

Output-

```
Valid number in line number 5 : 56
Invalid number in line number 6: Number followed with alphabets is invalid1b
Invalid number in line number 6: Number followed with alphabets is invalid5h
Valid number in line number 7 : 4.5
Valid number in line number 7 : 5
Valid number in line number 8 : 6.6
```

6 Check for valid declarative statements in your program. (CO1) eg: int a,b;

Q6.l

```
%{
#include<stdio.h>
int c=1;
%}

%option noyywrap
%s DECLARE
identifier [a-zA-Z_][a-zA-Z0-9_]*
number [0-9]+(\.[0-9])?[0-9]*
string \"[a-zA-Z0-9]+\"
```

```
%%
\n {c++;}
"int "|"float " {BEGIN DECLARE;ECHO;}
<DECLARE>{identifier}(={number})?, {ECHO;}
<DECLARE>{identifier}(={number})?; {
   BEGIN 0;
   ECHO;
   printf("\nValid declaration\n");
}
<DECLARE>{identifier}("="{string}) {
   printf("\n Invalid variable declaration in line no %d; string can't be assigned to integer or float variable:",c);
   ECHO;
   printf("\n");
}
<DECLARE>[,]+ {
   printf("\n Invalid usage of more than one comma in declaration in line no %d",c);
   BEGIN DECLARE;
   ECHO;
   printf("\n");
}
. ;
%%

int main()
{
   yyin = fopen("text6.txt","r");
   yylex();
   fclose(yyin);
   return 0;
}
```

Text6.txt

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main() {
int a=9,b=78;
int a;
int g="78",,;
int a=9 b=0
float c=5.6h=5;
sa=5; a=a+b;
printf("\n");
}
```

Output-

```
(base) natansh@Natansh's MacBook Air test % ./a.out
int a=9,b=78;
Valid declaration
int a;
Valid declaration
int
 Invalid variable declaration in line no 7; string can't be assigned to integer or float variable:g="78
"

 Invalid usage of more than one comma in declaration in line no 7,,
int float h=5;
Valid declaration
```

7. Write a Lex program to accept a C program and do the following error detection & correction.(CO1) Check for the valid if statement in the input C program. Report the errors to users.

Q7.I

```
%{
#include<stdio.h>
int c=1, bc=0, fc=0;
%}

%option noyywrap
%s IF B1 B2
%%

\n { c++; }
"if" {
   BEGIN IF;
   ECHO;
}
<IF>\s {ECHO;}
<IF>\( {
   BEGIN B1;
   bc++;
   ECHO;
}
<B1>\( {
   bc++;
   ECHO;
}
<B1>\) {
   bc--;
   ECHO;
}
<B1>\{ {
   ECHO;
   if(bc==0)
   {
```

```
        BEGIN B2;
        fc++;
    }
    else{
        printf("\nInvalid if statement, not all closed\n");
        BEGIN 0;
        bc=0;
        fc=0;
    }
}
<B1>. {ECHO;}
<B2>\} {
    ECHO;
    fc--;
    if(fc==0){
        printf("\nValid\n");
        BEGIN 0;
    }
}
<B2>\{ {
    ECHO;
    fc++;
}
<B2>. {ECHO;}
. ;
%%
int main() {
    yyin=fopen("text7.txt","r");
    yylex();
    fclose(yyin);
    return 0;
}
```

Text7.txt

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
 void main() {
int a,b=78;
if((a<5&&j<9) {
a=a+h; g=6+7;
a=a+b;
 printf("\n");
}
if
(a<n)
```

```
{ h=j+k;
}
 if(a<n))
{ g=h+k;
}
}
```

Output

```
 if((a<5&&j<9) {
 Invalid if statement, not all closed
 if(a<n){ h=j+k;}
 Valid
 if(a<n)){
 Invalid if statement, not all closed
```

8. Check for un- terminated multi line comment statement in your C program.(CO1)

Q8.1

```
%{
#include<stdio.h>
int c=1, flag=0;
%}

%option noyywrap
%s COMMENT

%%
\n {c++;}
"/*" {
   BEGIN COMMENT;
   printf("Comment begins in line no : %d.....\n", c);
   ECHO;
   flag=1;
}
<COMMENT>"*/" {
   BEGIN 0;
   ECHO;
   flag=0;
   printf("\nComment ends in line no : %d.....\n\n", c);
}
<COMMENT>. {ECHO;}
. ;
%%

int main()
{
   yyin=fopen("text8.txt","r");
   yylex();
   fclose(yyin);
   if(flag)
      printf("\nComment is not closed till the end of file!");
```

```
    return 0;
}
```

Text8.txt-

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
 /*dfddf*/ void
main()
{
/*vbhfghfgh
dfhfgh
fghgfhfg
fghfh */ int
a,b=78;
if((a<5&&j
<9) { a=a+h;
g=6+7;
a=a+b;
printf("\n
"); } /*
if(a<n) {
h=j+k; }
if(a<n))
{ g=h+k;
}
}
```

Output-

```
Comment begins in line no : 4.....
/*dfddf*/
Comment ends in line no : 4.....

Comment begins in line no : 7.....
/*vbhfghfghdfhfghfghgfhfgfghfh */
Comment ends in line no : 10.....

Comment begins in line no : 17.....
/*if(a<n) {h=j+k; }if(a<n)){ g=h+k;}}
Comment is not closed till the end of file!%
```

9. Write Yacc program to accept a statement and do the following error detection.(CO2)

a) Check for valid arithmetic expressions in the input C statement. Evaluate the arithmetic expression.

Q9.l

```
%{
#include "y.tab.h"
#include<stdio.h>
#include<ctype.h>
extern int yylval; int val;
%}
%%
```

```
[a-zA-Z][a-zA-Z0-9]* {
   printf("\n enter the value of variable %s:",yytext);
   scanf("%d",&val);
   yylval=val;
   return id;
}
[0-9]+[.]?[0-9]* {
   yylval=atoi(yytext);
   return num;
}
[\t] ;
\n {return 0;}
. {return yytext[0];}

%%

int yywrap()
{ return 1;
}
```