Output:

Test case 1:

Enter any identifier : var1

var1 is a valid identifier.

Test case 2:

Enter any identifier : 1_var

1_var is not a valid identifier

Test case 3:

Enter any identifier : _var1

_var1 is a valid identifier.

Expt.No.

Page No. 1
Date.

ASSIGNMENT - 1

1) Write a C program to identify the valid identifier variables.

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
int isValidIdentifier(char str[])
{
int i;
if ((str[0]>='a' && str[0]<='z') || (str[0]>='A'
&& str[0]<='Z') || (str[0]=='_'))
{
for(int i=0; i<=strlen(str); i++)
if ((str[i]>='a' && str[i]<='z') || (str[i]>='A'
&& str[i]<='Z') && (str[i]>='0' &&
str[i]<='9') && (str[i]=='_'))
{
return 1;
}
}
else {
return 0;
}
```

OXFORD

Teacher's Signature .....

Output

Test case 1

Enter any comment in C: //comment
It is a valid Comment.

Test case 2

Enter any comment in C: /* comment line */
It is multiline Comment.

Test case 3

Enter any comment in C: #comment.
It is not a valid comment in C.

Expt.No.

Page No. 2
Date.

```
int main() {
char str[1000];
printf("Enter any identifier: ");
scanf("%[^\n]s", &str);
if (isValidIdentifier(str)==1) {
printf("%s is a valid identifier", str);
}
else {
printf("%s is not a valid identifier", str);
}
return 0;
}
```

2) Write a C program to identify one valid comments in C.

```
#include <stdio.h>
#include <string.h>
#include <conio.h>
int isValidComment (char str[])
{
if(str[0] == '/' && str[1] == '/' && str[2] != '/')
{
return 1;
}
```

OXFORD

Teacher's Signature .....

```
else if (str[0] == '/' && str[1] == '*' && str[strlen(str)-1] == '/' && str[strlen(str)-2] == '*')
    {
    return 2;
    }
else {
    return 0;
    }
}
int main()
{
char str[1000];
printf("Enter any comment in C:");
scanf("%[^\n]s", &str);
if(isValidComment == 1) {
printf("It is a single line comment");
}
else if (isValidComment == 2) {
printf("It is a multi-line comment");
}
else {
printf("It is not a valid comment in C");
    }
return 0;
}
```

Output

Enter a file : demofile.c

Successfully removed comments from the file.

Expt.No.

Page No. 4
Date.

Assignment -2

WAP in C to remove comments from a file.

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

int remove_comments_from_file(char *file_name)
{
FILE *fp = fopen(filename, "r");
if (fp == NULL)
{
printf("Error opening file.\n");
return 0;
}
FILE *fp_new = fopen("temp.c", "w");
if (fp_new == NULL)
{
printf("Error opening file.\n");
return 0;
}
char line[100];
while (fgets(line, sizeof(line), fp)
{
```

OXFORD

Teacher's Signature .....

Expt.No.

Page No. 5
Date.

```
if(line[0] == '/' && line[1] == '/')
{ continue;
}
if(line[0] == '/' && (line[1] == '*')
continue;
fprintf(fp_new, "%.s", line);
{
fclose(fp_new);
remove(file_name);
rename("temp.c", file_name);
return 1;
}

int main(void)
{
char filename[100];
printf("Enter a filename");
scanf("%s", filename);
if (remove_components_from_file(filename))
{
printf("Successfully removed comments from the
file.\n");
}else
{
printf("Failed to remove comments from the file\n");
return 0;
}
```

OXFORD®

Teacher's Signature .....

Output

Test Case 1:

Input: 4+5*6/2-1+(2^2)

Output:

Enter an Infix Expression: 4+5*6/2-1+(2^2)

The Postfix Expression will be: 456*2/+1-22^+

The Result of the given expression is 22.

Test Case 2:

Input: 5+6+6*8/2-3

Output:

Enter an Infix Expression: 5+6+6*8/2-3

The postfix Expression will be: 56+68*2/+3-

The Result of the given expression is 32.

Test Case 3:

Input: 2+6+5+7+5+-8-9-2

Output

Enter an Infix Expression: 2+6+5+7+5+-8-9-2

The Post Expression will be: 26+5+7+5+8-9-2-

The Result of the given expression is 6.

Expt.No.

Page No. 6
Date.

Assignment-3

Write a C program to evaluate an arithmetic expression which is given as a string. Consider the input has no parentheses and contains the following operators only: +, -, *, /.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>

int stack2[30], temp, length=0, index=0, pos=0, top=-1;
char symbol, infix[20], postfix[20], stack[30];

void push(char);
void push2(int);
char pop();
int pop2();
int precedence(char);
void infix_to_postfix(char[]);
void eval_postfix(char[]);

void push(char symbol){
    top = top + 1;
    stack[top] = symbol;
}
```

OXFORD

Teacher's Signature .....

Test Case-4:

Input: 2^3+4

Output:

Enter an Infix Expression: 2^3+4

The Postfix expression will be: 23^4+

The Result of the given expression is 12

Test Case-5:

Input: (5+6)*(5+6)

Output:

Enter an Infix expression: (5+6)*(5+6)

The Postfix Expression will be: 56+56+*

The Result of the given expression is 121.

Expt.No.

Page No. 7
Date.

```
char pop(){
  temp = stack[top];
  top = top - 1;
  return temp;
}

int precedence(char symbol){
  int priority;
  switch(symbol){
    case '#': {
        priority = 0;
        break;
    }
    case '(':
    case ')': {
        priority = 1;
        break;
    }
    case '+':
    case '-': {
        priority = 2;
        break;
    }
    case '*':
    case '/': {
        priority = 3;
        break;
    }
```

OXFORD

Teacher's Signature .....

Expt.No.

Page No. 8
Date.

```
case '^': {
        priority=4;
        break;
        }
}
return priority;
}
void infix_to_postfix(char infix[]) {
length= strlen(infix);
push('#');
while(indx<length){
symbol = infix[indx++];
switch(symbol){

case '(': {
        push(symbol);
        break;
        }
case ')': {
        temp=pop();
while(temp!='(') {
        postfix[pos++]=temp;
        temp=pop();
        }
        break;
}
```

OXFORD

Teacher's Signature .....

Expt.No.

Page No. 9
Date.

```
case '-':
case '+':
case '*':
case '/':
case '^': {
    while (precedance(stack[top]) >= precedance(symbol))
    {
        temp = pop();
        postfix[pos++] = temp;
    }
    push(symbol);
    break;
}
default: {
    postfix[pos++] = symbol;
    break;
}
}
}
while (top > 0) {
    temp = pop();
    postfix[pos++] = temp;
    postfix[pos] = '\0';
}
}
```

Teacher's Signature .....

OXFORD

Expt.No.

Page No. 10
Date.

```
void push2 (int n){
  stack2[++top] = n;
}
int pop2 (){
  return stack2[top--];
}
void eval_postfix(char postfix[]) {
  char *temp;
  int n1, n2, n3, num;
    temp = postfix;
  while (*temp != '\0') {
      if (isdigit (*temp)) {
        num = *temp - 48;
        push2 (num);
      }
      else {
        n1 = pop2();
        n2 = pop2();
        switch (*temp) {
        case '+';
          { n3 = n1 + n2;
            break;
          }
```

OXFORD

Teacher's Signature .....

```
case '-':
{   n3 = n2 - n1;
    break;
}
case '*': {
    n3 = n1 * n2;
    break;
}
case '/':
{   n3 = n2 / n1;
    break;
}
case '^':
{   n3 = pow(n2, n1);
    break;
}
}
push2(n3);
}
temp++;
}
printf("\n The Result of the given expression is
        %d", pop2());
}
```

Teacher's Signature .....

Expt.No.

Page No. 12
Date.

```
// Main function
int main()
{
    printf("Enter an Infix Expression:");
    gets(infix);
    infix_to_postfix(infix);
    printf("\n The Postfix Expression will be:");
    puts(postfix);
    eval_postfix(postfix);
    return 0;
}
```

OXFORD®

Teacher's Signature .....

Input

Input : University

Output :
Enter the string of vowels & consonants : University

Number of vowels are : 4
Number of consonants are : 6

Expt.No.

Page No. 13
Date.

Assignment-4

Write a lex program to count the number of vowels and consonants, in a string.

```
%{
    int vowel_count=0,
    int const_count=0,
%}

%%
[aeiouAEIOU] {vow_count++;}
[a-zA-Z] {const_count++;}
%%
int yywrap() {}
int main()
{
    printf ("Enter the string of vowels and
            consonants :");
    yylex();
    printf ("Number of vowels are: %d\n", vow_count);
    printf ("Number of consonants are: %d\n", const_
                                                count);
    return 0;
}
```

OXFORD

Teacher's Signature .....

Output

Input: Hello

Output:
Number of words present in the string = 1

Expt.No.

Page No. 14
Date. /

Assignment -5

Write a lex program to count the number of words in a string.

```
%{
int count=0;
%}

%%
[a-zA-Z0-9]* {count++;}
%%

int yywrap() {}
int main()
{
printf("Enter a string: ");
yylex();
printf("Number of words present in the string is = %d\n", count);
}
```

OXFORD

Teacher's Signature .....

Expt.No.

Page No. 15
Date.

Write a lex program which will count the number of words, characters, digits, lines, and special characters present in a file.

```
%{
int word=0;
int digits=0;
int spl=0;
int line=0;
int temp=0;
int count=0;
%}

%%
[a-z A-Z]* {word++; count=count+yleng;}
[0-9] {digits++;}
\n {line++; word++;}
. {spl++;}
%%

int yywrap() {}

int main()
{
yyin=fopen("dummy.txt","r");
```

Output

Input: dummy.txt
This is a dummy file.
Call me @ 1234567890

Output: No. of words = 9
No. of digits = 9
No. of special characters = 10
No. of lines = 2
No. of characters = 41

OXFORD

Teacher's Signature .....

Expt.No.

Page No. 16
Date.

```
yylex();
temp = count + spl + digits;
printf("Number of words = %d\n Number of digits = %d\n
Number of special characters = %d\n Number of lines =
%d\n Number of characters = %d\n", word, digits,
spl, line, temp);
return 0;
}
```

OXFORD

Teacher's Signature .....

Test cases

```
$ touch program.c
$ touch new.c
$ program
$ cat program.c
# include <stdio.h>
  int main()
   {int a,b;
    printf("Enter 1st number: ");
    scanf("%d",&a);
    printf("Enter 2nd number: ");
     scanf("%d",&b);
     int sum1;
      sum1=a+b;
      printf("the sum of %d & %d", a,b, sum1);
      return 0;
     }

$ vi countAndExchange.l
$ lex countAndExchange.l
$ cc lex.yy.c -lfl
$ ./a.out
Number of scanfs = 2
Number of printfs = 3
$ cat new.c
#include<stdio.h>
 int main()
  {int a,b;
  printf("Enter 1st number");
   scanf("%d",&a);
   printf("Enter 2nd number");
    scanf("%d",&b);
```

Expt.No.

Page No. 17
Date.

Assignment-6

1) write a lex program which will count the number of scanf and printf statements in a C program, replace them with readf and writef statements respectively.

```
%{
#include<stdio.h>
int sf=0, pf=0;
%}

%%
"scanf" {sf++; fprintf(yyout, "readf");}
"printf" {pf++; fprintf(yyout, "writef");}
%%

int yywrap()
{
return 1;
}
int main()
{
yyin = fopen("program.c", "r");
yyout = fopen("new.c", "w");
```

OXFORD

Teacher's Signature .....

```
int sum1;
sum1=a+b;
printf("The sum of %d and %d is %d", a,b, sum1);
return 0;
}
```

Test case 1:

Input: Amigt
Output: 6

Input: Hey
Output: 4

Expt.No.

Page No. 18
Date.

```
yylex();
printf("Number of words = %d\n Number of prints=
%d\n", ss, pf);
return 0;
}
```

8) write a lex program to count the number of words.

```
%{
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
%}
%%
([a-z]|[A-z])* {printf("%d", strlen(yytext));}
%%

int main()
{
yylex();
return 0;
}
int yywrap()
{
}
```

OXFORD

Teacher's Signature .....

Test case 1:

Input : // Hello

Output :

— Single Line comment Hence, removed! —

Test case-2 :

Input : UEM kolkata

output : UEM kolkata

Expt.No.

Page No. 19
Date.

Q) write a Lex program to identify the single line comment and remove it.

```
%{
#include <stdio.h>
%}

%%
"//".*\n|[a-z]|[0-9]|[a-z]|" ")*;;
%%

int main()
{
yylex();
return 0;
}

int yywrap()
{
}
```

OXFORD

Teacher's Signature .....

Output

while (j<=0);

keywords : while separators : (identifiers : j
operators : <= constant value : 0 separators :)
separators : ;

Total number of Tokens : 7

Expt.No.

Page No. 20
Date. 08/04/2022

Assignment - 7

Q) Write a Lex program which will count the number of tokens and also identify them.

```
%{
int n=0;
%}

%%
"while"|"if"|"else"|"auto"|"break"|"case"|"char"|
"continue"|"do"|"default"|"const"|"double"|"enum"|"extern"|
"for"|"goto"|"float"|"int"|"long"|"register"|"return"|
"signed"|"static"|"sizeof"|"short"|"struct"|"switch"|
"typedef"|"union"|"void"|"volatile"|"unsigned"
{n++;
printf("\t keywords : %s", yytext);}

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

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

OXFORD

Teacher's Signature .....

2) Output file

~~Output file~~

```
1) #include<stdio.h>
2) int main()
3) {
4)   int a, b;
5)   printf("Enter 1st number:");
6)   scanf("%d", &a);
7)   printf("Enter 2nd no:");
8)   scanf("%d", &b);
9)   int sum1;
10)  sum1 = a+b;
11)  printf("the sum of %d and %d is %d", a,b,sum1);
12)  return 0;
13) }
```

output

Expt.No.

Page No. 21
Date.

```
%{
int yywrap()
{
return 1;
}
int main()
{
yylex();
printf("\n Total number of tokens :%d\n",n);
}
```

3) Write a lex program which will add line numbers to a file.

```
%{
int line_no = 1;
%}
line .*\n
%%
{line} { printf("%10d > %s", line_no++, yytext); }
%%
int yywrap()
{ return 1;
}
```

OXFORD

Teacher's Signature .....

Input file:

I Live at Kolkata.
I am 21 years old.

Output file:

I Live at Kolkata I am 21 years old.

Expt.No.

Page No. 22
Date.

```
int main()
{
yyin = fopen ("program.c", "r");
yylex();
return 0;
}
```

Write a Lex program which will remove multiple spaces, tabs and lines from a user given file.

```
%.{
%.}
%.%
[\n\t] + {fprintf (yyout, " ");}
.{ fprintf (yyout, "%s", yytext);}
%.%
int yywrap()
{
return 1;
}
int main()
{
yyin = fopen ("input.txt", "r");
yyout = fopen ("output.txt", "w");
```

OXFORD

Teacher's Signature .....

Expt.No.

Page No. 23
Date.

```
system();
return 0;
}
```

OXFORD®

Teacher's Signature .....

Output Test Case-1

Type the expression & press enter key:
1 + 5 * 6 / 7
Valid Expression!

Test case 2

Type the expression & press enter key:
a + b + c + d
Invalid expression :(

Expt.No.

Page No. 24
Date. 22/04/2022

Assignment-8

Write a YACC program to check the validity of an arithmetic expression that uses the operators such as +, -, * and /

```
%{
#include "ytab.h"
%}
%%
[0-9]+(\.[0-9]+)?
{
return NUM; }
[a-zA-Z_][a-zA-Z0-9]*
{
return ID; }
[\t] ;
\n      return 0;
.       return yytext[0];
%%
```

OXFORD

Teacher's Signature .....

Expt.No.

Page No. 25
Date.

→

```
#include <stdio.h>
#include <stdlib.h>
int yylex();
int yyerror();
%}
%token NUM ID
%left '+' '-'
%left '*' '/'
%%
e: e'+'e
 | e'-'e
 | e'*'e
 | e'/'e
 |'('e')'
 | NUM
 | ID ;
%%
int main()
{
printf ("Type the expression & press enter key.\n");
yyparse ();
printf ("Valid expression\n");
return 0;
}
```

OXFORD®

Teacher's Signature .....

Output

Test case-1

Enter any Arithmetic expression:
5+6
Result = 11
Entered arithmetic expression is Valid

Test case-2

Enter any Arithmetic expression:
(5+6+8) * 0
Result = 0
Entered Arithmetic expression is Valid.

Expt.No. 12

Page No. 26
Date.

int yyerror()
{
printf("Invalid expression : (\n");
exit(0);
}

write a YACC program to implement a basic calculator which can perform basic arithmetic operations.

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

%%
[0-9]+ {
yylval = atoi(yytext);
return NUMBER;
}
[\t];
[\n] return 0;

OXFORD

Teacher's Signature .....

Expt.No.

Page No. 27
Date.

```
return yytext[0];
%%

int yywrap()
{
return 1;
}

%{
#include<stdio.h>
int flag=0;
%}

%token NUMBER

%left '+' '-'
%left '*' '/' '%'
%left '(' ')'
%%

ArithmeticExpression: E{
    printf("\nResult=%d\n", $$);
    return 0;
};
```

Teacher's Signature .....

Expt.No.

Page No. 28
Date.

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

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

| E '*' E { $$ = $1 * $3; }

| E '/' E { $$ = $1 / $3; }

| E '%' E { $$ = $1 % $3; }

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

| NUMBER { $$ = $1; }

;

%%

// driver code

void main()

{

printf ("\nEnter any Arithmetic expression :\n");

yyparse();

if(flag == 0)

printf ("\nEntered arithmetic expression is Valid \n\n");

}

Teacher's Signature .....

OXFORD

Expt.No.

Page No. 29
Date.

```
void yyerror()
{
printf("\nEntered arithmetic expression is Invalid\n");
flag = 1;
}
```

OXFORD

Teacher's Signature .....