

Output:

Test case 1:

Enter any identifier : var1
var1 is a valid identifier.

Test case 2:

Enter any identifier : -var
-var is not a valid identifier

Test Case 3:

Enter any identifier : var1
var1 is a valid identifier.

Expt.No.

Page No. /
Date. /

ASSIGNMENT -1

→ write a C program to identify the valid identifier
variable's .

```
#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;
}
```

Teacher's Signature

OXFORD®

OutputTest 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[100];
    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;
}
```

Q) Write a C program to identify the valid comments in C.

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

Teacher's Signature

OXFORD®

```
else if (start) == ' ' && 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 *filename)
```

```
{ 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 (ffgetline (line, sizeof (line), fp)
```

Teacher's Signature

OXFORD®

```
if (line[0] == '/' && line[1] == '/')  
    { continue;  
    }  
if (line[0] == '/' && line[1] == '*')  
    { continue;  
    }  
printf (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;  
}
```

Teacher's Signature

OutputTest 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: $2+6+5+7+5+-8-9-2$ OutputEnter an Infix Expression: $2+6+5+7+5+-8-9-2$ The Postfix 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 <conio.h>

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

void push(char);

void push(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;

}

Teacher's Signature

OXFORD®

Test Case-4:

Input: $2^3 \cdot 4$

Output:

Enter an Infix Expression: $2^3 \cdot 4$

The Postfix expression will be: $23^4\cdot$

The result of the given expression is 12.

Test Case-5:

Input: $(5+6) \cdot (5+6)$

Output:

Enter an Infix expression: $(5+6) \cdot (5+6)$

The Postfix expression will be: $56+56+\cdot$

The result of the given expression is 12.

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;
```

Teacher's Signature

OXFORD®

```
case '^': {  
    priority = 4;  
    break;  
}  
}  
action priority;  
}  
void infix_to_postfix(char infix[]){  
    length = strlen(infix);  
    push('#');  
    while (mdn < length){  
        symbol = infix[mdn++];  
        switch (symbol){  
            case '(': {  
                push(symbol);  
                break;  
            }  
            case ')': {  
                temp = pop();  
                while (temp != '(') {  
                    postfix[pos++] = temp;  
                    temp = pop();  
                }  
                break;  
            }  
        }  
    }  
}
```

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

```
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;
            }
        }
    }
}
```

```
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 ("The result of the given expression is  
%.2f", pop2());
```

// Main function

int main()

{

 printf ("Enter an Infix Expression : ");

 gets (infix);

 infix_to_postfix (infix);

 printf ("In The Postfix Expression will be : ");

 puts (postfix);

 eval_postfix (postfix);

 return 0;

}

Ques

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

1.9

```
int vowel-count=0,  
int const-count=0,
```

1.9 }

1.9.

```
[aeiouAEIOU] {vowel-count++;  
[a-zA-Z] {const-count++;
```

1.9.

```
int getch() {  
int main()
```

1.

```
printf ("Enter the string of vowels and  
consonants : ");
```

scanf () ;

```
printf ("Number of vowels are : %d", vowel-count);  
printf ("Number of consonants are : %d", const-  
count);  
return 0;
```

1.

Teacher's Signature

OXFORD'

Output

Input: hello

Output

Number of words present in the string = 1

Expt.No.

Page No. / 14
Date /Assignment -5

7. Write a C program to count the numbers & words in a string;

1.9

int count=0,

1.7

1.7

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

1.7

1.7 if (isspace(c))

int main()

{ cout << "Enter a string:";

scanf();

printf ("Number of words present in the string
is = %d\n", count);

}

oxford'

Teacher's Signature

Output

Input: dummy.txt
This is a dummy file.
Call me @ 123456789

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

Expt.No.

Page No. / 15

Date. /

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

```
1.9  
int word=0;  
int digits=0;  
int spcl=0;  
int line=0;  
int temp=0;  
int count=0;  
1.9
```

```
1.9.  
[a-zA-Z]* {word++; count=count+1; long; }  
[0-9] {digits++; }  
In {line++; word++; }  
.spcl++;  
1.9.
```

```
int yywrap()  
{
```

```
int main()  
{
```

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

Teacher's Signature

OXFORD

yytext();

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,

}

Test cases

```

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

$ vi countAndExchange.c
$ vim countAndExchange.c
$ cc longgg.c -lfl
$ ./longgg
Number of scanf = 2
Number of printf = 3
$ rm new.c
$ include <stdio.h>
int main()
{
    float a, b;
    scanf ("Enter 1st number");
    scanf ("%f", &a);
    printf ("Enter 2nd number");
    scanf ("%f", &b);
}

```

Expt.No.

Page No./	12
Date./	

Assignment-6

- 7) Write a C program which will count the number of `scanf` and `printf` statements in a C program, replace them with `readf` and `writf` statements respectively.

```

7.
#include <stdio.h>
int sf=0, pf=0;
7.

```

7.1.

```

"scanf" {sf++; fprintf (yyout, "readf");}
"printf" {pf++; fprintf (yyout, "writf");}
7.1.

```

int main ()

{

return;

}

int main ()

{

yyin = fopen ("program.c", "r"),
yyout = fopen ("new.c", "w");

Teacher's Signature

```
int sum();
sum(a+b,
    cout << "The sum of " << a << " and " << b << " is " << a+b, sum);
return 0;
}
```

Test Case 1:

Input: 4
Output: 6

Input: very
Output: 4

Expt.No.

Page No. / 18
Date. / /

```
yylevel();
parmf ("Number of words = % In Number of printfs");
i. d (n", cf, pf);
return 0;
}
```

⇒ write a C++ program to count the number of words.

```
#include <iostream.h>
#include <stdlib.h>
#include <string.h>
{.}
```

```
(fa = z)(a-z)* {parmf ("%.d", strlen(str));}
{.}
```

int main()

```
{  
yylevel();  
return 0;  
}  
int yywrap()  
{.}
```

Teacher's Signature

OXFORD®

Test case 1:

Input: //Hello

Output:

-- Single line comment! Hence, removed! --

Test case-2:

Input: /*Hello

Output: /*Hello

Expt.No.

Page No. / 19
Date. /

Q) Write a C++ program to identify the single line comment and remove it.

Y- [
include <stdio.h>

Y-]

Y- .
"// ([a-z][a-zA-Z][a-zA-Z] | " ") * ;
Y- .

int main()

{
 system();
 return 0;

}

int system ()

{
}

Teacher's Signature

Output

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

Total number of Tokens: 7

Expt.No.

Page No./ 20
Date/ 08/04/2022

Assignment-7

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

7.9
int n=0;
7.9

7.9.
"while" | "if" | "else" | "auto" | "break" | "case" | "char"
"continue" | "do" | "default" | "const" | "double" | "enum" | "extern"
"for" | "goto" | "float" | "int" | "long" | "register" | "return"
"signed" | "static" | "sleep" | "short" | "struct" | "switch"
"typedef" | "union" | "void" | "volatile" | "unsigned"
{ n++;
printf ("it keywords : %s", yytext); }
[a-zA-Z][a-zA-Z0-9]*
n++;
printf ("it operators : %s", yytext); }
[0-9]+ n++; printf ("it constant values: %s", yytext); }

Teacher's Signature

2) Output file

Input file

```
#include <stdio.h>
int main()
{
    int a, b;
    printf("Enter two numbers:");
    scanf ("%d", &a);
    scanf ("Enter 2nd no.:");
    scanf ("%d", &b);
    int sum;
    sum = a+b;
    printf ("The sum of %d and %d is %d", a,b,sum);
    return 0;
}
```

Output

Expt.No.

Page No. / 21
Date. /

1.1.

int yywrap()

{
 return 1;

} int main()

{
 yylex();

printf ("In Total number of tokens %d\n", n);

}

2) write a C program which will add line numbers to a file.

1.2

int line_no = 1;

1.3

line *ln

{.}

{line} { printf ("%1.10d > %s, line-no++, yytext); }

1.4

int yywrap()

{
 return 1;

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()

```
{  
    FILE *fin = fopen ("program.c", "r");  
    FILE *fout = fopen ("output.out", "w");  
    int c;  
    while ((c = fgetc (fin)) != EOF)  
        fputc (c, fout);  
}
```

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

1.1

1.1

```
[in 1] + fprintf (yyout, ""); ]  
[ fprintf (yyout, "%s", yytext), ]
```

1.1

int yywrap()

{

return 1;

} int main()

```
{  
    FILE *fin = fopen ("input.out", "r");  
    FILE *fout = fopen ("output.out", "w");  
    int c;  
    while ((c = fgetc (fin)) != EOF)  
        fputc (c, fout);  
}
```

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 /

Y. S

#include "y.tab.h"

Y. J

Y. I

[0-9]+([.][0-9]+)?

Y.

return NUM;

[a-zA-Z][a-zA-Z0-9]*

Y.

return ID;

[*];

Y. N

return 0;

Y. D

return yyyent[0];

Y. Y

OXFORD

Teacher's Signature

```
#include <stdio.h>
#include <stdlib.h>
int yyfem();
int yyerror();
y.
y.
y. taken NUM ID
y. left '+' '-'
y. left '*' '/'
y.-
e: e '+' e
l e '*' e
l e '/' e
l '(' e ')'
INUM
120 ;
y.
int main()
{
    printf ("Type the expression & press enter key . h");
    yyparse ();
    printf ("Valid Expression/ In");
    return 0;
}
```

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.

Page No. / 26
Date. /

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

Q) Write a YACC program to implement a basic calculator which can perform basic arithmetic operations.

```
1. q
    %include <math.h>
    #include "y.tab.h"
    extern int yyval;
    y.
}
{0-9} + q
yyval = atoi(yytext);
action NUMBER;
}
[4];
[1-9] return 0;
```

Teacher's Signature

OXFORD®

return yytext[0];

y.y.

int yywrap()

{
 return 1;
}



y.y
#include <stdio.h>

int flag=0;

y.y

y.y token NUMBER

y.y left '+' '-'

y.y left '*' '/' '**'

y.y left '(' ')'

y.y

Arithmetic Expression : E {

printf ("In result = %d\n", \$);

return 0;

} ;

Expt.No.

Page No./ 28

Date. /

$$E : E' + ' E \{ \$\$ = \$14\$3; \}$$

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

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

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

$$E \sim ' E \{ \$\$ = \$1 \sim \$3; \}$$

~~$$E \sim (\sim E) \{ \$\$ = \$2; \}$$~~

NUMBER { \$\\$ = \$1; }

;

/*

// driver code

void main()

{ printf ("Enter any Arithmetic expression :\n");

scanf ();

if (flag == 0)

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

}

OXFORD®

Teacher's Signature

```
void yyerror()
{
    printf ("Incorrect arithmetic expression is %s\n");
    flag = 1;
}
```