VISION DATE: 29/1/24 PAGE NO.

TUTORIAL-3

Q.1] Write a lex program to find out total no. of vowels, and consonants from given input string.

A.1]

```
%{
    int total_vow = 0;
    int total_cons = 0;
%}

%%
[aeiouAEIOU] {total_vow++;}
[a-z A-Z] {total_cons++;}

%%
int yywrap() {return 1;}

int main()
{
   printf("Enter string of vowels and consonants: ");
   yylex();
   printf("The total no. of vowels are: %d\n", total_vow);
   printf("The total no. of consonants are: %d\n", total_cons);
   return 0;
}
```

Q.2] Write a lex program to convert lowercase to uppercase string.

A2]

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

```
%%
[a-z] printf("%c", yytext[0] - ('a' - 'A'));
. {return 0;}
%%

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

Q.3 Write a lex program to remove comments from a C program.

A.3

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

start \/\*
end \*\/

%%
\/\/(.*);
{start}.*{end};
%%

int main(int k, char **argcv)
{
```

```
    yyin = fopen(argv[1], "r");
    yyout = fopen("output.c", "w");
    yylex();
    return 0;
}
```

Q.4] Program to count no. of :-

a) +ve & -ve integers

→

```
%{
    int pos_no = 0, neg_no = 0;
%}

%%
^[-][0-9]+ {neg_no++;
printf("Negative number = %s\n", yytext); }

^[0-9]+ {pos_no++;
printf("Positive number = %s\n", yytext); }

%%
int yywrap() {return 1;}

int main() {
    yylex();
    printf("No. of positive nos. = %d\n",
           No. of negative nos. = %d\n",
           pos_no, neg_no);

    return 0;
}
```

b) +ve & -ve fractions

→
```
%{
    int pos_frac = 0;
    int neg_frac = 0;
%}

DIGIT [0-9]
%%
\+?{DIGIT}*\.{DIGIT}+   pos_frac++;

-{DIGIT}*\.{DIGIT}+   neg_frac++;
.;

%%

int main(){

    yylex();
    printf("No. of positive fractions : %d", pos_frac);
    printf("No. of negative fractions : %d", neg_frac);
    return 0;
}
```

Q.5] Lex program to check valid / invalid.

a) Mobile no. (considering 10 digit mob. no. followed by +91)

→ P.T.O

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

%%

(\+91)?[1-9][0-9]{10} { printf("Valid Mobile Number\n"); }
{ printf("Invalid Mobile Number\n"); exit(0); }
%%

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

b) Email Address -

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

%%
[a-z.0-9]+@[a-z]+".com"|".in" { flag=1; }
%%
int main() {
    yylex();
    if(flag==1)
    printf("Valid email\n");
```

```
else printf ("Invalid email \n");
return 0; }
```

Q.6 Lex program to implement a simple calculator.

A.6

```
%{
    float p, flag, ans;
    char cc;
%}
digit [0-9]+
op "+"|"-"|"*"|"/"
%%

{digit} {
        p = atof (yytext);
        if (flg == 0)
        {
        ans = p;
        flag = 1;
        }
        else
        { switch (cc) {
            case '+' : ans = ans + p;
            case '-' : ans = ans - p;
            case '*' : ans = ans * p;
            case '/' : ans = ans / p;
        }
        }
    } {op} {
```

```
if (strcmp(yytext, "+") == 0)
    cc = '+';
if (strcmp(yytext, '-') == 0)
    cc = '-';
if (strcmp(yytext, '*') == 0)
    cc = '*';
if (strcmp(yytext, '/') == 0)
    cc = '/';
}
{ printf("final ans = %f ", ans); exit(0);
%%
int main(){
    flag = ans = 0;
    yylex();
    return 0;
}
```

Q.7 Given sentence simple or compound.

A.7

```
%{
    int flag = 0;
%}
%%
("  "[aA][nN][dD]"  ") | ("  "[oO][rR]"  ")
("  "[bB][uU][tT]"  ")  { flag = 1; }
%%
```

```
int main()
{
printf("Enter sentence\n");
yylex();
if (flag == 1)
printf("Compound Sentence\n");
else
printf("Simple sentence\n");
return 0;
}
```

Q.8 Lex program recognising and print date formats in input text.

A.8

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

DIGIT [0-9]
%%

{DIGIT}{2}\/{DIGIT}{2}\/{DIGIT}{4} {
    printf("Date found: %s\n", yytext);
}

.|\n    // Ignore other characters, if present
%%

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