1. a) Write a LEX program to recognize valid ***arithmetic expression***. Identifiers in the expression could be only integers and operators could be + and *. Count the identifiers & operators present and print them separately.

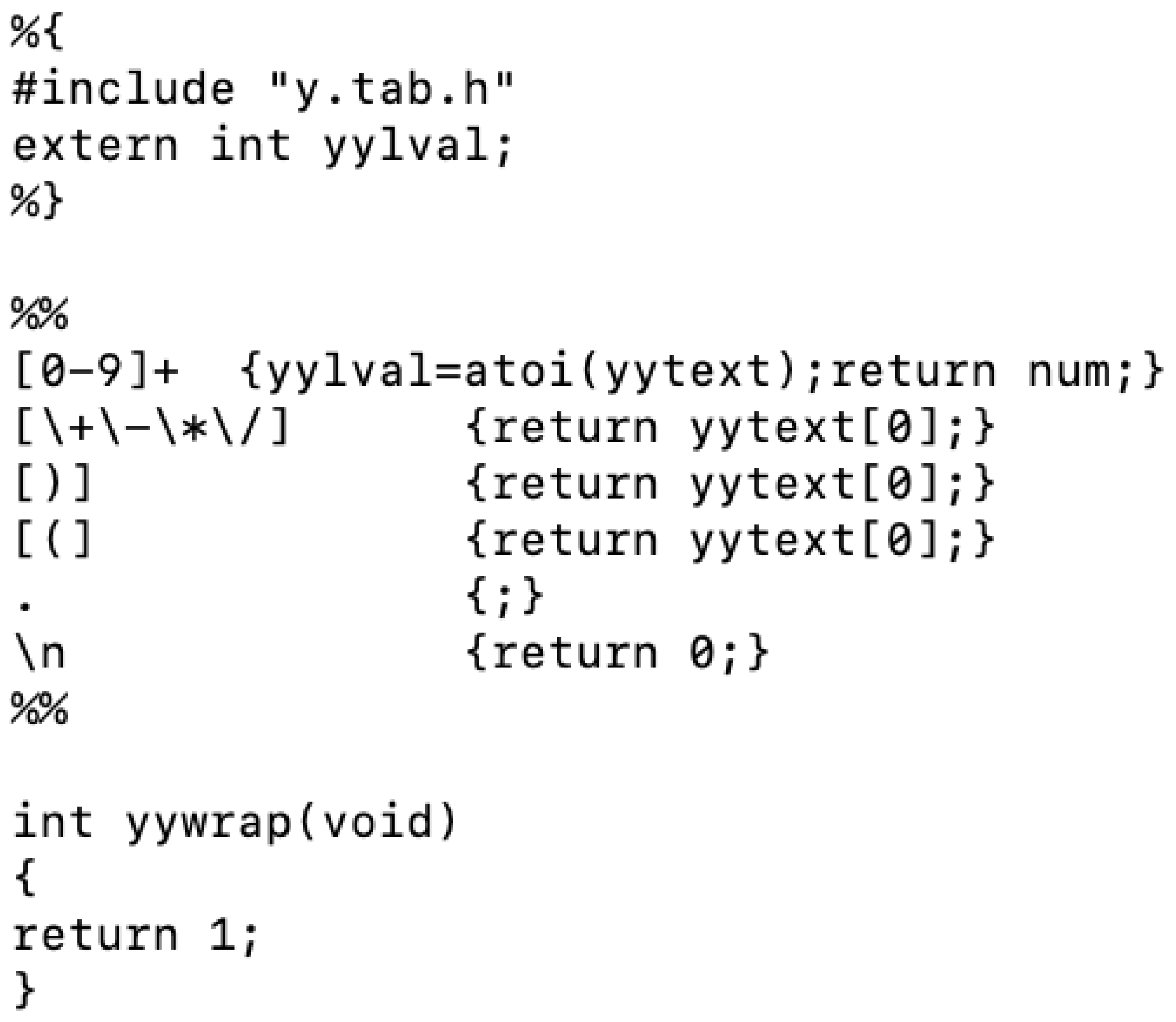
```
%{
#include<stdio.h>
int v=0,op=0,id=0,flag=0;
%}

%%
[a-zA-Z][a-zA-Z0-9]*  {id++; printf("\n identifier\t");ECHO;}
[\+\-\*\/\=]            {op++; printf("\n operator\t");ECHO;}
"("                     {v++;}
 ")"            {if(v!=0) v--; else flag=1;}
";"                     {flag=1;}
.|\n                    {;}
%%

int yywrap(void)
{
return 1;
}

int main()
{
printf("enter an expression");
yylex();
if((op+1)==id && v==0 &&flag==0)
{
printf("\ngiven expression is valid");
printf("\ntotal no of identifier is %d\n",id);
printf("\ntotal no of operators is %d\n",op);
}
else
{
printf("\n invalid operator");
}
return 0;
}
```

b) Write YACC program to evaluate ***arithmetic expression*** involving operators: +, -, *, and /.



```
%{
#include "y.tab.h"
extern int yylval;
%}

%%
[0-9]+  {yylval=atoi(yytext);return num;}
[\+\-\*\/]      {return yytext[0];}
[)]             {return yytext[0];}
[(]             {return yytext[0];}
.               {;}
\n              {return 0;}
%%

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

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

%token num
%left '*' '/'
%left '+' '-'

%%

input:exp       { printf("%d\n",$$);exit(0);}

exp:exp'+'exp   {$$=$1+$3;}
|exp'-'exp      {$$=$1-$3;}
|exp'*'exp      {$$=$1*$3;}
|exp'/'exp      { if($3==0) { printf("Divide by Zero\n");exit(0);}
                                else
                                $$=$1/$3;}
|'('exp')'              {$$=$2;}
|num                    {$$=$1;}
;
%%

int yyerror() {
printf("error");
exit(0);
}

int main()
{
printf("Enter an expression:\n");
yyparse();
}
```