**(Lex Program : intar.l)**  
  
ALPHA [A-Za-z]  
DIGIT [0-9]  
%%  
  
{ALPHA}({ALPHA}|{DIGIT})\* return ID;  
{DIGIT}+ {yylval=atoi(yytext); return NUM;}  
[\n\t] yyterminate();  
. return yytext[0];  
%%  
  
**(Yacc Program : intar.y)**  
  
%token ID NUM  
%right '='  
%left '+' '-'  
%left '\*' '/'  
%left UMINUS  
%%  
  
S : ID{push();} '='{push();} E{codegen\_assign();}  
   ;  
E : E '+'{push();} T{codegen();}  
   | E '-'{push();} T{codegen();}  
   | T  
   ;  
T : T '\*'{push();} F{codegen();}  
   | T '/'{push();} F{codegen();}  
   | F  
   ;  
F : '(' E ')'  
   | '-'{push();} F{codegen\_umin();} %prec UMINUS  
   | ID{push();}  
   | NUM{push();}  
   ;  
%%  
  
#include "lex.yy.c"  
#include<ctype.h>  
char st[100][10];  
int top=0;  
char i\_[2]="0";  
char temp[2]="t";  
  
main()  
 {  
 printf("Enter the expression : ");  
 yyparse();  
 }  
 push()  
{  
  strcpy(st[++top],yytext);  
 }  
 codegen()  
 {  
 strcpy(temp,"t");  
 strcat(temp,i\_);  
  printf("%s = %s %s %s\n",temp,st[top-2],st[top-1],st[top]);  
  top-=2;  
 strcpy(st[top],temp);  
 i\_[0]++;  
 }  
  
 codegen\_umin()  
 {  
 strcpy(temp,"t");  
 strcat(temp,i\_);  
 printf("%s = -%s\n",temp,st[top]);  
 top--;  
 strcpy(st[top],temp);  
 i\_[0]++;  
 }  
 codegen\_assign()  
 {  
 printf("%s = %s\n",st[top-2],st[top]);  
 top-=2;  
 }  
**Output**:  
nn@linuxmint ~ $ lex intar.l  
nn@linuxmint ~ $ yacc intar.y  
nn@linuxmint ~ $ gcc y.tab.c -ll -ly  
nn@linuxmint ~ $ ./a.out  
Enter the expression : a=(k+8)\*(c-s)  
t0 = k + 8  
t1 = c - s  
t2 = t0 \* t1  
a = t2