**ICG.l**

%{  
 #include "y.tab.h"  
%}  
%%  
[a-z] {yylval = \*yytext; return VARIABLE;}  
[0-9]+ {yylval = atoi(yytext); return NUMBER;}  
[ \t\n] ;  
"+" return(PLUS);  
"-" return(MINUS);  
"\*" return(TIMES);  
"/" return(DIVIDE);  
"^" return(POWER);  
"(" return(LEFT\_PARENTHESIS);  
")" return(RIGHT\_PARENTHESIS);  
";" return(END);  
"=" return(EQUAL);  
%%  
int yywrap(void)  
{  
 return 1;  
}

**ICG.y**

%{

#include <stdio.h>

#include <stdlib.h>

int var = 0;

%}

%start lines

%token NUMBER

%token VARIABLE

%token EQUAL

%token PLUS MINUS TIMES DIVIDE POWER

%token LEFT\_PARENTHESIS RIGHT\_PARENTHESIS

%token END

%left PLUS MINUS

%left TIMES DIVIDE

%right POWER

%%

lines: /\* empty \*/

| lines line /\* do nothing \*/

line: VARIABLE EQUAL exp END { printf("%c = t%d",$1,($3-127));}

;

exp : term { $$=$1; }

| exp PLUS term {

$$=printequation($1,$3,43);

}

| exp MINUS term {

$$=printequation($1,$3,45);

}

;

term : factor { $$=$1; }

| term TIMES factor {

$$=printequation($1,$3,42);

}

| term DIVIDE factor {

$$=printequation($1,$3,47);

}

;

factor : NUMBER { $$=yylval; }

| VARIABLE { $$=yylval; }

| LEFT\_PARENTHESIS exp RIGHT\_PARENTHESIS

{ printf("found PARENS exp\n"); }

;

%%

int main (void) {

return yyparse ( );

}

int yyerror (char \*s) {

fprintf (stderr, "%s\n", s);

}

int printequation(int input1, int input2, int sign){

int output;

printf("t%d = ",var);

if(input1 < 127 && input2 < 127)

{

printf("%c %c %c\n",input1,sign,input2);output=127+var;var++;

}

else if( input1 >= 127 && input2 >= 127){

printf("t%d %c t%d\n",(input1-127),sign,(input2-127));output=127+var;var++;

}

else if( input1 >= 127){

printf("t%d %c %c\n",(input1-127),sign,input2);output=127+var;var++;

}

else if( input2 >= 127){

printf("%c %c t%d\n",input1,sign,(input2-127));output=127+var;var++;

}

return output;

}

**Output**

F:\SEPROJECT\Lex>icg.exe

a=b+c-d/e\*f;

t0 = b + c

t1 = d / e

t2 = t1 \* f

t3 = t0 - t2

a = t3