**Lex File :-**

%{

#include "y.tab.h"

#include<math.h>

extern YYSTYPE yylval;

%}

%%

[-+\*/();=] {return yytext[0];}

[a-zA-Z\_][a-zA-Z\_0-9]\* {strcpy(yylval.name, yytext); return id;}

"$" {return 0;}

%%

**Yacc File :-**

%{

#include<stdio.h>

#include<string.h>

#include<math.h>

struct quad

{

char op[20];

char arg1[20];

char arg2[20];

char res[20];

};

struct quad Q[10];

int tindex=1,qindex=0,i=0;

%}

%union

{

char name[20];

}

%token <name> id

%left '+' '-'

%left '\*' '/'

%nonassoc UM

%type <name> E

%%

slist : slist S

| S

;

S:id '=' E ';' {addquad("=", $3, " ", $1);}

|E ';'

;

E:E '+' E {sprintf($$,"t%d",tindex); tindex++; addquad("+",$1,$3,$$);}

|E '-' E {sprintf($$,"t%d",tindex); tindex++; addquad("-",$1,$3,$$);}

|'-' E %prec UM {sprintf($$,"t%d",tindex); tindex++; addquad("um",$2,"",$$);}

|E '\*' E {sprintf($$,"t%d",tindex); tindex++; addquad("\*",$1,$3,$$);}

|E '/' E {sprintf($$,"t%d",tindex); tindex++; addquad("/",$1,$3,$$);}

|'('E')' {strcpy($$,$2);}

|id {strcpy($$,$1);}

;

%%

int addquad(char \*op,char \*arg1,char \*arg2,char \*res)

{

strcpy(Q[qindex].op, op);

strcpy(Q[qindex].arg1, arg1);

strcpy(Q[qindex].arg2, arg2);

strcpy(Q[qindex].res, res);

qindex++;

}

void display()

{

printf("Op\tArg1\tArg2\tResult");

for(i=0; i<qindex; i++)

{

printf("\n%s\t%s\t%s\t%s", Q[i].op, Q[i].arg1, Q[i].arg2, Q[i].res);

}

printf("\n\n");

}

main()

{

yyparse();

display();

}

int yyerror(char \*s)

{

printf("%s (Invalid Input)\n",s);

}

**Output**

pccoe@212A-07:~/BE121/as4$ yacc -d quad.y

pccoe@212A-07:~/BE121/as4$ lex q.l

pccoe@212A-07:~/BE121/as4$ gcc lex.yy.c y.tab.c -ll

pccoe@212A-07:~/BE121/as4$ ./a.out

x=(a+b)\*(c-d)/g;$

Op Arg1 Arg2 Result

+ a b t1

- c d t2

\* t1 t2 t3

/ t3 g t4

= t4 x

pccoe@212A-07:~/BE121/as4$