**EX-3:**

**LEX PROGRAM:**

%{

#include "y.tab.h"

%}

%%

[0-9]+ {yylval.sym=(char)yytext[0]; return NUMBER;}

[a-zA-Z]+ {yylval.sym=(char)yytext[0];return LETTER;}

\n {return 0;}

. {return yytext[0];}

%%

yywrap()

{

return 1;

}

**YACC PROGRAM:**

%{

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

void ThreeAddressCode();

void qudraple();

char AddToTable(char ,char, char);

int ind=0;//counter for the number of symbols

char temp = '1';//for t1,t2,t3.....

struct incod

{

char opd1;

char opd2;

char opr;

};

%}

%union

{

char sym;

}

%token <sym> LETTER NUMBER

%type <sym> expr

%left '+'

%left '\*''/'

%left '-'

%%

statement: LETTER '=' expr ';' {AddToTable((char)$1,(char)$3,'=');}

;

expr:

expr '+' expr {$$ = AddToTable((char)$1,(char)$3,'+');}

| expr '-' expr {$$ = AddToTable((char)$1,(char)$3,'-');}

| expr '\*' expr {$$ = AddToTable((char)$1,(char)$3,'\*');}

| expr '/' expr {$$ = AddToTable((char)$1,(char)$3,'/');}

| '(' expr ')' {$$ = (char)$2;}

| NUMBER {$$ = (char)$1;}

| LETTER {$$ = (char)$1;}

|'-' expr {$$ = AddToTable((char)$2,(char)'\t','-');}

;

%%

yyerror(char \*s)

{

printf("%s",s);

exit(0);

}

struct incod code[20];

char AddToTable(char opd1,char opd2,char opr)

{

code[ind].opd1=opd1;

code[ind].opd2=opd2;

code[ind].opr=opr;

ind++;

return temp++;

}

void ThreeAddressCode()

{

int cnt = 0;

char temp = '1';

printf("\n\n\t THREE ADDRESS CODE\n\n");

while(cnt<ind)

{

if(code[cnt].opr != '=')

printf("t%c : = \t",temp++);

if(isalpha(code[cnt].opd1))

printf(" %c\t",code[cnt].opd1);

else if(code[cnt].opd1 >='1' && code[cnt].opd1 <='9')

printf("t%c\t",code[cnt].opd1);

printf(" %c\t",code[cnt].opr);

if(isalpha(code[cnt].opd2))

printf(" %c\n",code[cnt].opd2);

else if(code[cnt].opd2 >='1' && code[cnt].opd2 <='9')

printf("t%c\n",code[cnt].opd2);

cnt++;

}

}

void quadraple()

{

int cnt = 0;

char temp = '1';

printf("\n\n\t QUADRAPLE CODE\n\n");

while(cnt<ind)

{

printf(" %c\t",code[cnt].opr);

if(code[cnt].opr == '=')

{

if(isalpha(code[cnt].opd2))

printf(" %c\t \t",code[cnt].opd2);

else if(code[cnt].opd2 >='1' && code[cnt].opd2 <='9')

printf("t%c\t \t",code[cnt].opd2);

printf(" %c\n",code[cnt].opd1);

cnt++;

continue;

}

if(isalpha(code[cnt].opd1))

printf(" %c\t",code[cnt].opd1);

else if(code[cnt].opd1 >='1' && code[cnt].opd1 <='9')

printf("t%c\t",code[cnt].opd1);

if(isalpha(code[cnt].opd2))

printf(" %c\t",code[cnt].opd2);

else if(code[cnt].opd2 >='1' && code[cnt].opd2 <='9')

printf("t%c\t",code[cnt].opd2);

else printf(" %c",code[cnt].opd2);

printf("t%c\n",temp++);

cnt++;

}

}

main()

{

printf("\n Enter the Expression : ");

yyparse();

ThreeAddressCode();

quadraple();

}

**OUTPUT:**

Z:\Computation\programs\EX-3>flex ex3l.l

Z:\Computation\programs\EX-3>bison -dy ex3y.y

Z:\Computation\programs\EX-3>gcc lex.yy.c y.tab.c

Z:\Computation\programs\EX-3>a.exe

Enter the Expression : C=A-(B+C);

THREE ADDRESS CODE

t1 : = B + C

t2 : = A - t1

C = t2

QUADRAPLE CODE

+ B C t1

- A t1 t2

= t2 C

Z:\Computation\programs\EX-3>