**NATIONAL INSTITUTE OF TECHNOLOGY PUDUCHERRY**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**CS306 – PCD LAB**

**B. TECH / CSE - VI Semester**

**EXERCISE 7 Duration : 3 Hours**

**Marks : 5 Marks**

1. **AIM:**To generate the intermediate code for the given for expression.

**CODE**:

**Lex Program:**

alpha [A-Za-z]

digit [0-9]

%%

[ ] ;

for return FOR;

{digit}+ return NUM;

{alpha}({alpha}|{digit})\* return ID;

"<=" return LE;

">=" return GE;

"==" return EQ;

"!=" return NE;

"||" return OR;

"&&" return AND;

. return yytext[0];

%%

int yyerror(char \* mes)

{

puts(mes);

}

**Yacc Program:**

%{

#include <stdio.h>

#include <stdlib.h>

%}

%token ID NUM FOR LE GE EQ NE OR AND

%right "="

%left OR AND

%left '>' '<' LE GE EQ NE

%left '+' '-'

%left '\*' '/'

%right UMINUS

%left '!'

%%

S : FOR '(' E ';'{lab1();} E {lab2();}';' E { lab3();}')' E ';'{lab4(); exit(0);}

E : V '=' {push();} E{codegen\_assign();}

| V '<' {push();} E{codegen\_cond();}

| E '+' {push();} E{codegen();}

| E '-' {push();} E{codegen();}

| E '\*' {push();} E{codegen();}

| E '/' {push();} E{codegen();}

| '(' E ')'

| '-' {push();} E{codegen\_umin();} %prec UMINUS

| V

| NUM{push();}

;

V : ID {push();}

;

%%

#include "lex.yy.c"

#include<string.h>

#include<ctype.h>

char st[100][10];

int label[20];

int top=0;

char i\_[2]="0";

char temp[2]="t";

int lno=0,ltop=0;

int start;

int main()

{

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

yyparse();

}

int push()

{

strcpy(st[++top],yytext);

}

int 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]++;

}

int codegen\_cond()

{}

int codegen\_umin()

{

strcpy(temp,"t");

strcat(temp,i\_);

printf("%s = -%s\n",temp,st[top]);

top--;

strcpy(st[top],temp);

i\_[0]++;

}

int codegen\_assign()

{

printf("%s = %s\n",st[top-2],st[top]);

top-=2;

}

int lab1()

{

printf("L%d: \n",lno++);

}

int lab2()

{

printf("if not %s<%s goto L%d\n",st[top-2],st[top],lno);

top=top-2;

i\_[0]++;

label[++ltop]=lno;

lno++;

printf("goto L%d\n",lno);

label[++ltop]=lno;

printf("L%d: \n",++lno);

}

int lab3()

{

int x;

x=label[ltop--];

printf("goto L%d \n",start);

printf("L%d: \n",x);

}

int lab4()

{

int x;

x=label[ltop--];

printf("goto L%d \n",lno);

printf("L%d: \n",x);

}

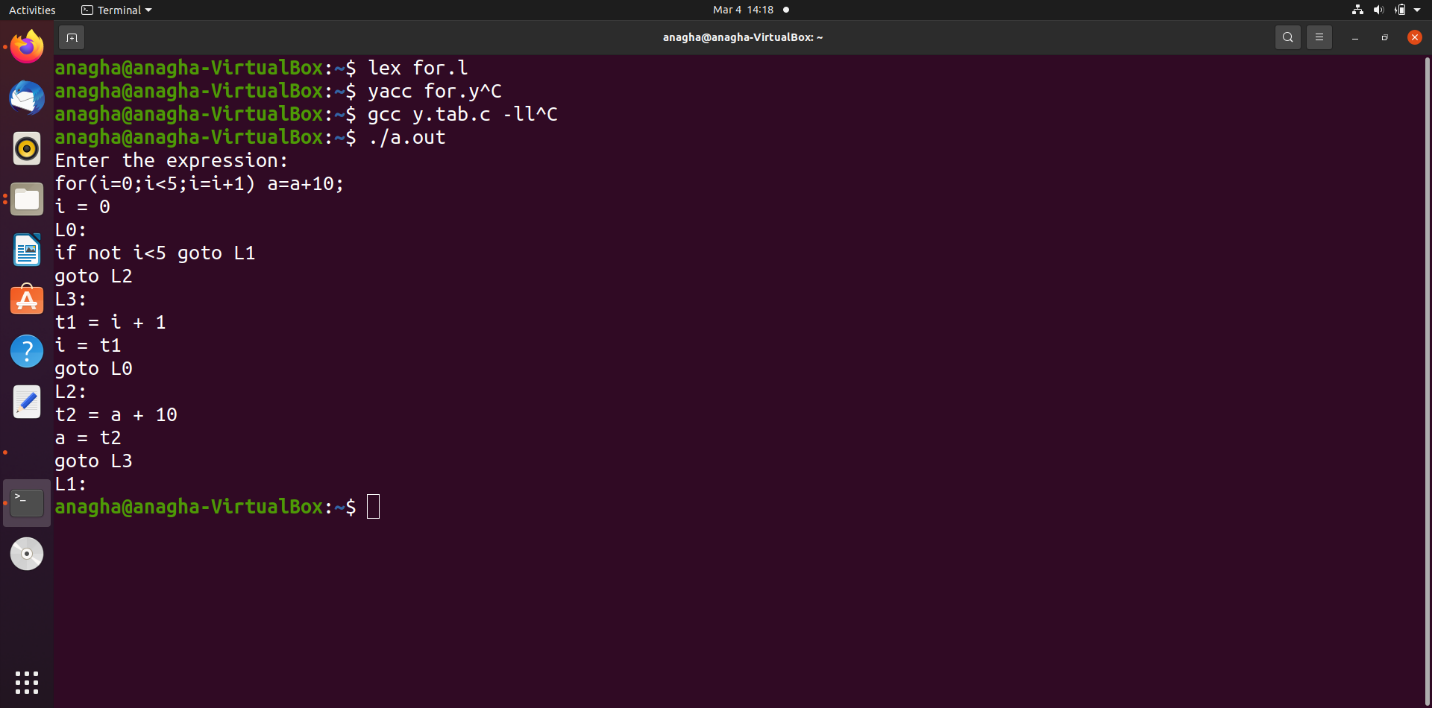
int yywrap()

{

return 1;

}

**OUTPUT**:



1. **AIM:** To generate intermediate code for the given infix expression

**CODE**:

**Lex Program:**

%{

%}

ALPHA [A-Za-z]

DIGIT [0-9]

%%

{ALPHA}({ALPHA}|{DIGIT})\* return ID;

{DIGIT}+ {yylval=atoi(yytext); return NUM;}

[\n\t] yyterminate();

. return yytext[0];

%%

void yyerror(char \* mes)

{

puts(mes);

}

int yywrap()

{

return 1;

}

**Yacc Program:**

%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>

#include<string.h>

char st[100][25];

int top=0,ptr=0;

int tint=0; int tintar[200];

int main()

{

printf("Enter the expression : ");

yyparse();

}

push()

{

strcpy(st[++top],yytext);

ptr++;

}

codegen()

{

printf("t%d = %s",tint,st[top-2]);

printnum(2);

printf(" %s %s",st[top-1],st[top]);

printnum(0);

printf("\n");

top-=2;ptr-=2;

strcpy(st[top],"t");

tintar[ptr]=tint;

tint++;

}

codegen\_umin()

{

printf("t%d = -%s\n",tint,st[top]);

printnum(0);

top--;

ptr--;

strcpy(st[top],"t");

tintar[ptr]=tint;

tint++;

}

codegen\_assign()

{

printf("%s = ",st[top-2]);

printnum(2);

printf("%s",st[top]);

printnum(0);

printf("\n");

top-=2;ptr-=2;

}

printnum(int n)

{

if( strcmp(st[top-n],"t")==0)

{

printf("%d",tintar[ptr-n]);

}

}

**OUTPUT**:

