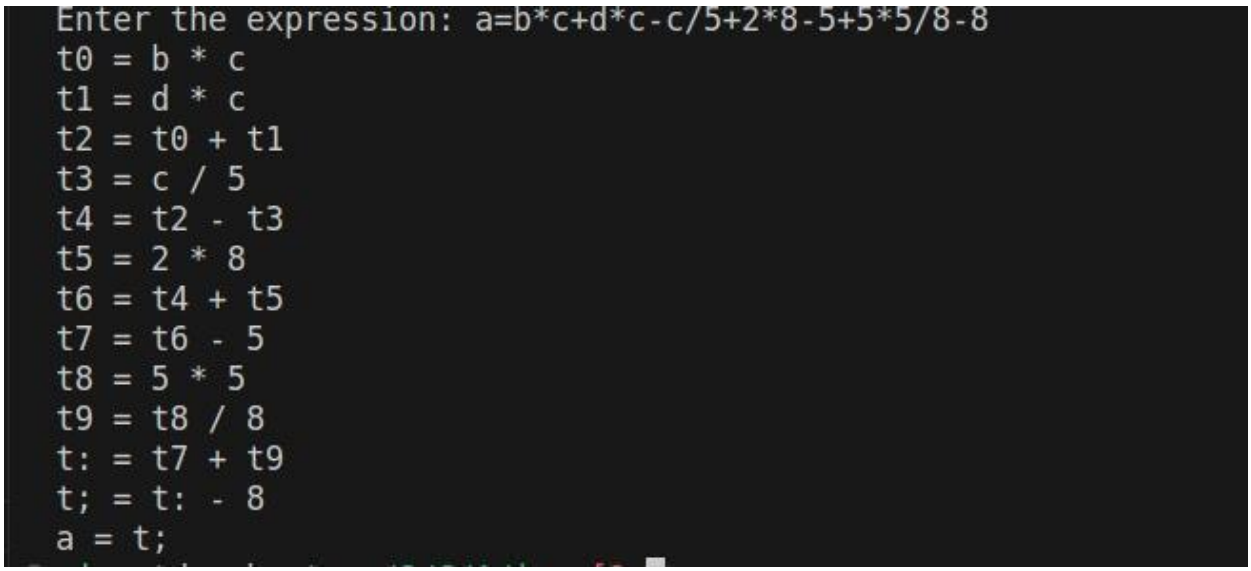
Antriksh Sharma
20070122021
CCL - LEX & YACC

Input & Output



Program

YACC

```
%{
#include <stdio.h>
#include <stdlib.h>
#define YYSTYPE double

int yylex(void);
void yyerror(char const* s);

void push();
%}

%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";

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;
 }

 void yyerror (char const *s) {
   printf("reenter previous line:");
}

int main()
{
      printf("Enter the expression: ");
      yyparse();
      return 0;
}
```

LEX

```
ALPHA [A-Za-z]
DIGIT [0-9]

%%
{ALPHA}({ALPHA}|{DIGIT})* return ID;
{DIGIT}+ {yylval=atoi(yytext); return NUM;}
[\n\t] yyterminate();
. return yytext[0];
%%
```

Writeup

This program accepts a valid expression as input using LEX and converts it to intermediate TAC using YACC