Compiler Design Laboratory
Assignment 8

Narasimhan Kovalai-20CS01075

Q1)

The CFG of the basic calculator grammar

G -> E

E -> E + T | E - T | T

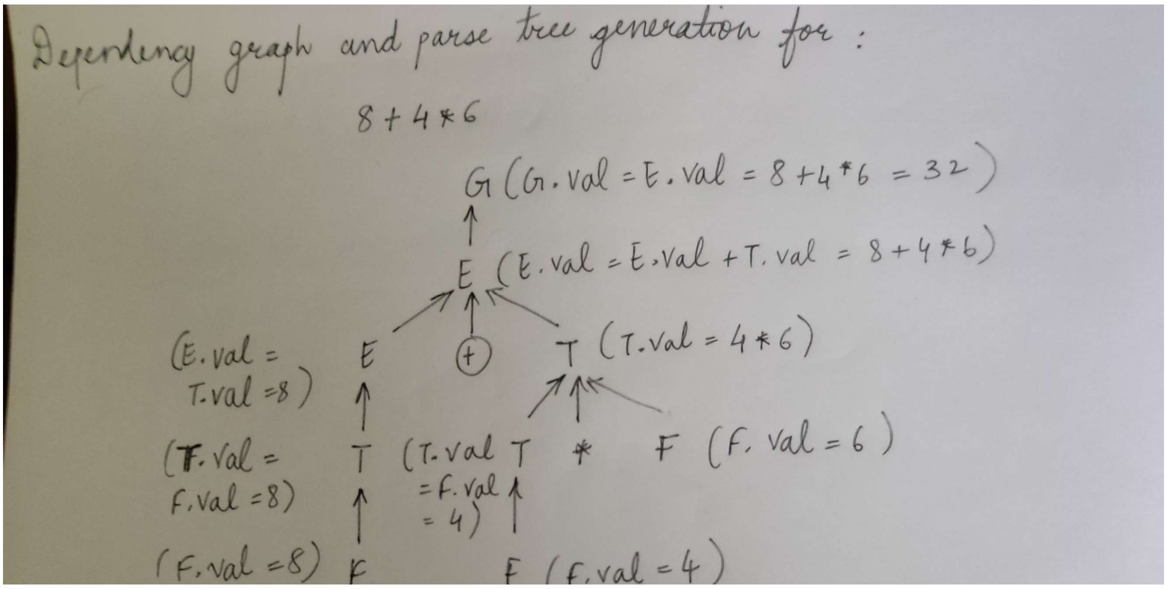
T -> T * F | T / F | F

F -> (E) | num

Here, G:Goal,E:expression,T:term,num:[0-9]+

Yes, it is suitable for LR(1) parsing .

The dependency graph and parse table for the above is :



Attribute Grammar (where AS stands for Synthesised Attribute)

AS(G) = AS(E) = AS(T) = AS(F) = { val }

G -> E {G.val = E.val}

E -> E + T {E1.val = E2.val + T.val}

E -> E - T {E1.val = E2.val - E'.val}

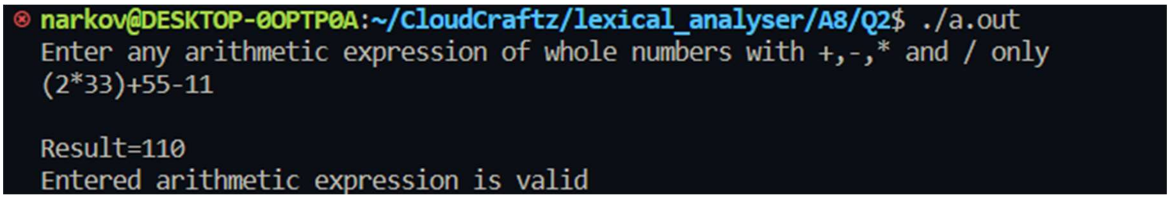
E -> T {E.val = T.val}

T -> T * F {T1.val = T2.val * F.val}

T -> T / F {T1.val = T2.val / F.val}

T -> F {T.val = F.val}

F -> (num) {F.val = num.lexval}



```
⊗ narkov@DESKTOP-0OPTP0A:~/CloudCraftz/lexical_analyser/A8/Q2$ ./a.out
  Enter any arithmetic expression of whole numbers with +,-,* and / only
  (2*33)+55-11

  Result=110
  Entered arithmetic expression is valid
```

Steps for Running Q1:

1. yacc calc.y -d
2. lex calc.l
3. gcc y.tab.c
4. ./a.out

Steps for Running Q2:

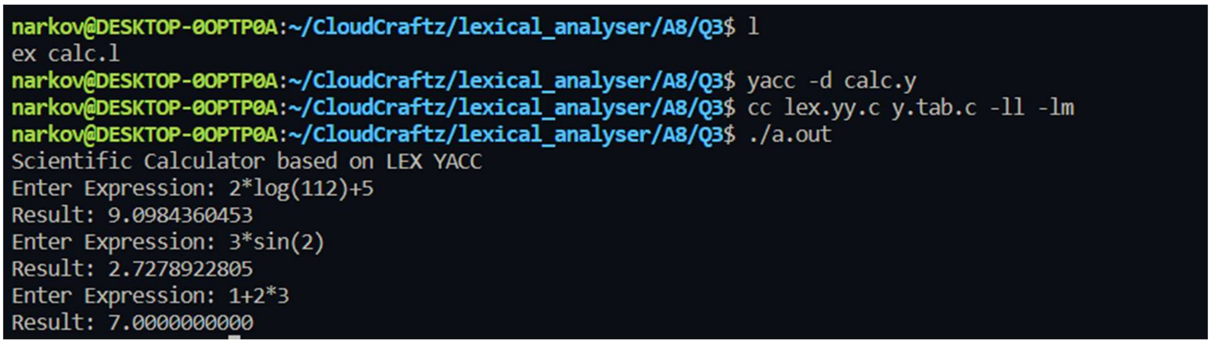
To run the program:-

STEP 1) lex calc.l

STEP 2) yacc -d calc.y

STEP 3) cc lex.yy.c y.tab.c -ll -lm

STEP 4) ./a.out



```
narkov@DESKTOP-0OPTP0A:~/CloudCraftz/lexical_analyser/A8/Q3$ l
ex calc.l
narkov@DESKTOP-0OPTP0A:~/CloudCraftz/lexical_analyser/A8/Q3$ yacc -d calc.y
narkov@DESKTOP-0OPTP0A:~/CloudCraftz/lexical_analyser/A8/Q3$ cc lex.yy.c y.tab.c -ll -lm
narkov@DESKTOP-0OPTP0A:~/CloudCraftz/lexical_analyser/A8/Q3$ ./a.out
Scientific Calculator based on LEX YACC
Enter Expression: 2*log(112)+5
Result: 9.0984360453
Enter Expression: 3*sin(2)
Result: 2.7278922805
Enter Expression: 1+2*3
Result: 7.0000000000
```