

# Department of CSE

## SSN College of Engineering

Vishakan Subramanian - 18 5001 196 - Semester VI

7 March 2021

---

### UCS 1602 - Compiler Design

---

#### Exercise 5: Implementation of Desk Calculator Using Yacc Tool

##### **Aim:**

Write a Lex program to recognize relevant tokens required for the **Yacc** parser to implement desk calculator. Write the Grammar for the expression involving the operators namely,  $+$ ,  $-$ ,  $*$ ,  $/$ ,  $^$ ,  $(, )$ . Precedence and associativity has to be preserved. Yacc is available as a command in Linux. The grammar should have non terminals E, OP and a terminal id.

Verify your calculator with the following inputs:

1.  $3 + 9$
2.  $3 + 9 * 6$
3.  $(3 + 4) * 7$
4.  $(3 - 4) + (7 * 6)$
5.  $5/7 + 2$
6.  $(4^2)^1$
7.  $(2^3)^2$

## Code - Yacc Parser File:

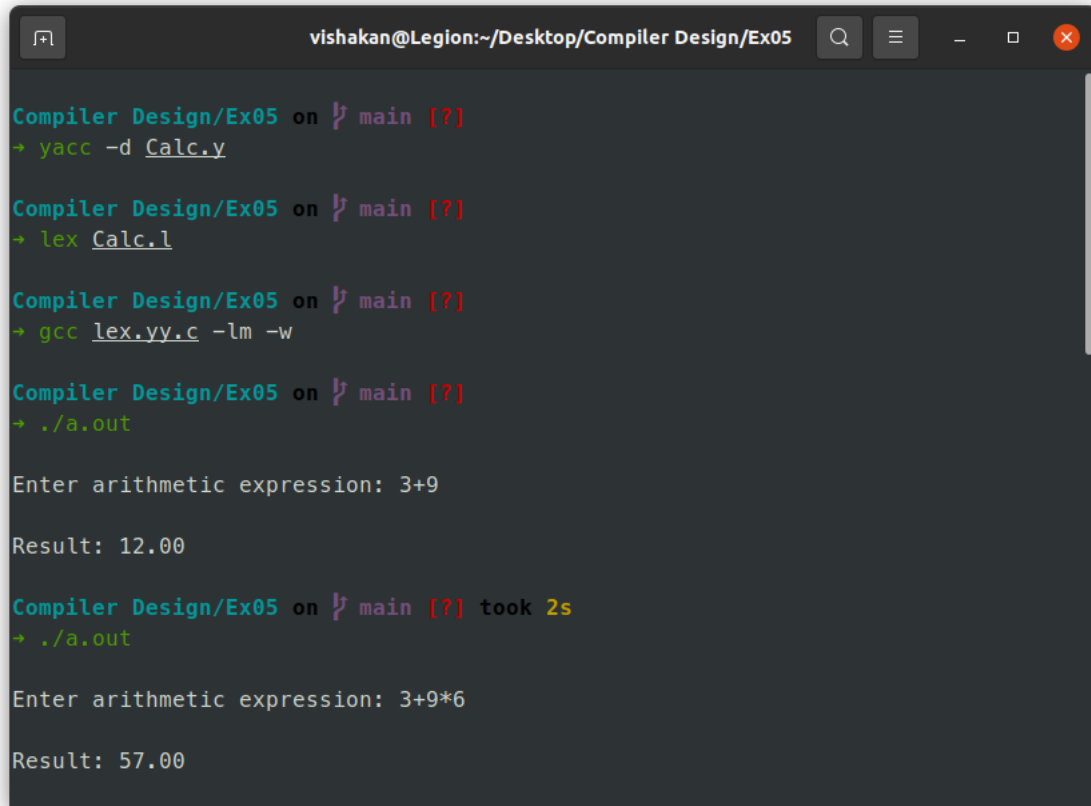
```
1 %{
2     #include <stdio.h>
3     #include <math.h>
4     #define YYSTYPE double
5     int flag = 0;
6 %}
7
8 %token NUM
9 /*Defining the precedence*/
10 %left '+', '-'
11 %left '/', '*'
12 %right '^'
13 %left '(', ')'
14
15 %%
16 Line : Expr {printf("\nResult: %.2f\n", $$);}
17 Expr : Expr '+' Expr {$$ = $1 + $3;}
18      | Expr '-' Expr {$$ = $1 - $3;}
19      | Expr '*' Expr {$$ = $1 * $3;}
20      | Expr '/' Expr {$$ = $1 / $3;}
21      | Expr '^' Expr {$$ = pow($1, $3);}
22      | '(' Expr ')' {$$ = $2;}
23      | NUM {$$ = $1;}
24 %%
25
26 int yyerror(){
27     flag = 1;
28     return 1;
29 }
30
31 int main(void){
32     printf("\nEnter arithmetic expression: ");
33     yyparse();
34
35     if(flag){
36         printf("\nEntered Unexpected Tokens.\n");
37     }
38
39     return 0;
40 }
41
42 /* Usage:
43     Run yacc -d Calc.y
44     Run lex Calc.l
45     Run gcc lex.yy.c -lm -w
46     Run ./a.out
47 */
```

## Code - Lex Grammar File:

```
1 %{
2     #include <stdio.h>
3     #include "y.tab.c"
4     extern YYSTYPE yylval;
5 %}
6
7 %%
8
9 [0-9]+ {yylval = atoi(yytext); return NUM;}
10 [\t]   ;
11 [\n]   return 0;
12 .      return yytext[0];
13
14 %%
15
16 int yywrap(){
17     return 1;
18 }
```

## Output 1:

Figure 1: Console Output - 1.



```
vishakan@Legion:~/Desktop/Compiler Design/Ex05
Compiler Design/Ex05 on 1 main [?]
→ yacc -d Calc.y

Compiler Design/Ex05 on 1 main [?]
→ lex Calc.l

Compiler Design/Ex05 on 1 main [?]
→ gcc lex.yy.c -lm -w

Compiler Design/Ex05 on 1 main [?]
→ ./a.out

Enter arithmetic expression: 3+9

Result: 12.00

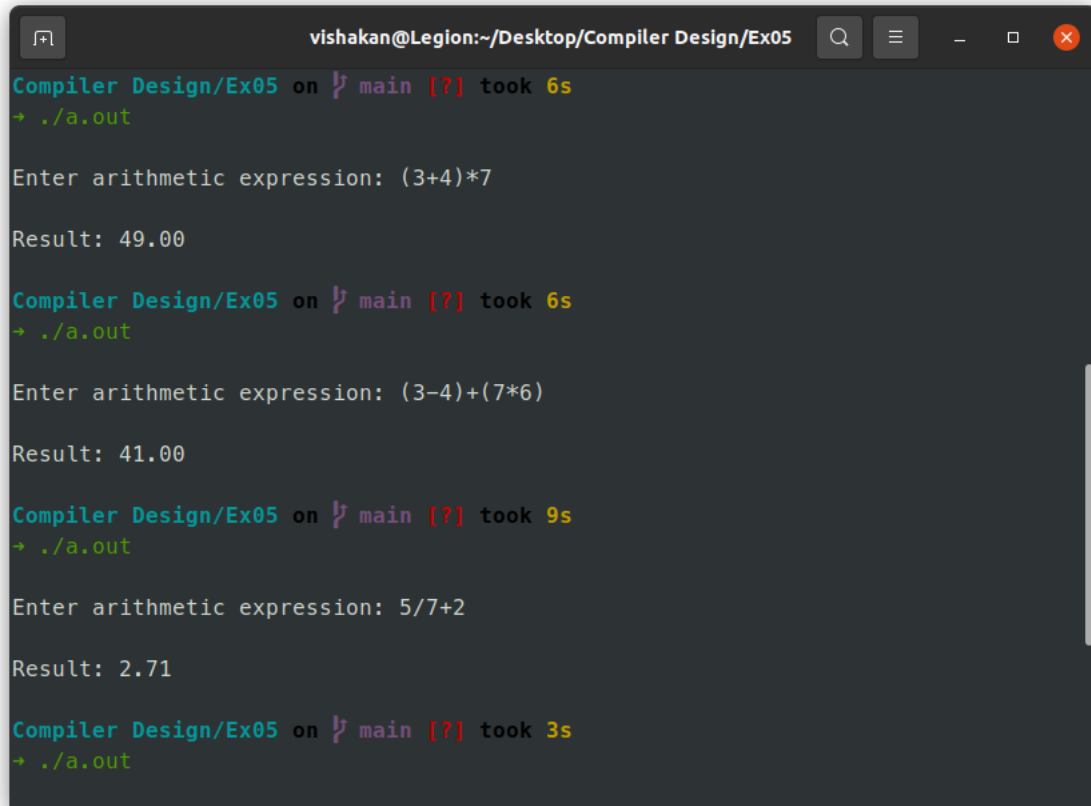
Compiler Design/Ex05 on 1 main [?] took 2s
→ ./a.out

Enter arithmetic expression: 3+9*6

Result: 57.00
```

## Output 2:

Figure 2: Console Output - 2.



```
vishakan@Legion:~/Desktop/Compiler Design/Ex05
Compiler Design/Ex05 on  main [?] took 6s
→ ./a.out

Enter arithmetic expression: (3+4)*7

Result: 49.00

Compiler Design/Ex05 on  main [?] took 6s
→ ./a.out

Enter arithmetic expression: (3-4)+(7*6)

Result: 41.00

Compiler Design/Ex05 on  main [?] took 9s
→ ./a.out

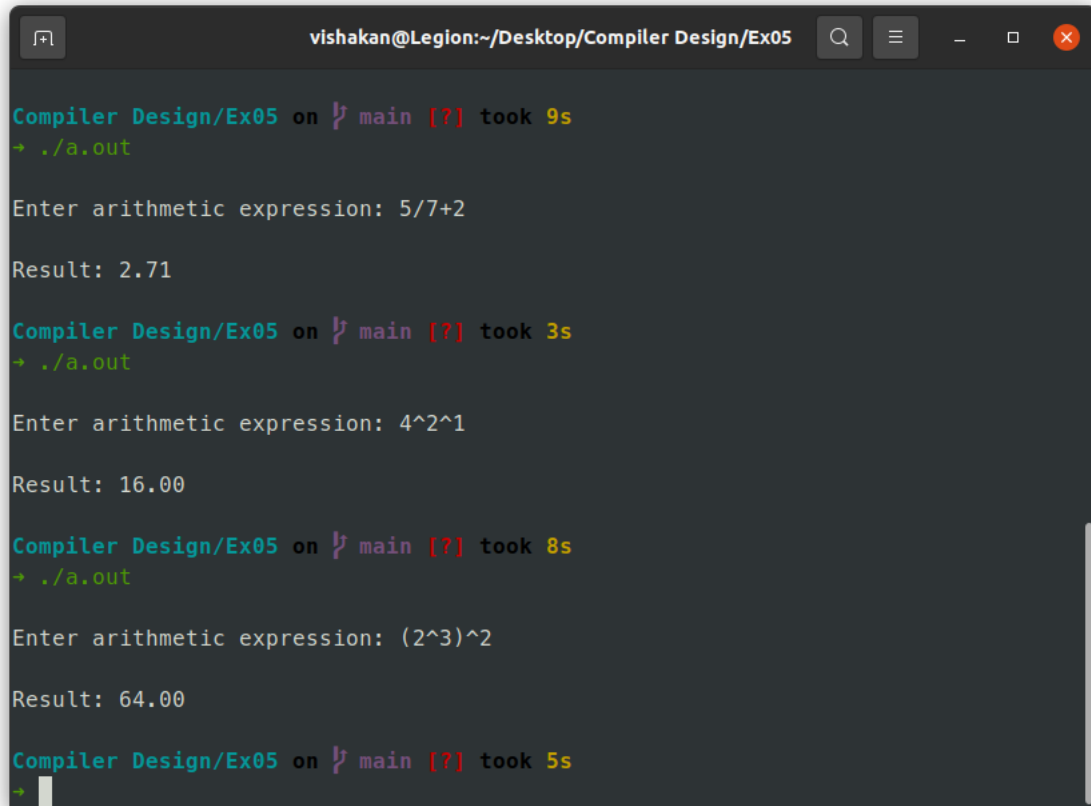
Enter arithmetic expression: 5/7+2

Result: 2.71

Compiler Design/Ex05 on  main [?] took 3s
→ ./a.out
```

## Output 3:

Figure 3: Console Output - 3.



```
vishakan@Legion:~/Desktop/Compiler Design/Ex05

Compiler Design/Ex05 on  main [?] took 9s
→ ./a.out

Enter arithmetic expression: 5/7+2

Result: 2.71

Compiler Design/Ex05 on  main [?] took 3s
→ ./a.out

Enter arithmetic expression: 4^2^1

Result: 16.00

Compiler Design/Ex05 on  main [?] took 8s
→ ./a.out

Enter arithmetic expression: (2^3)^2

Result: 64.00

Compiler Design/Ex05 on  main [?] took 5s
→
```

## Learning Outcome:

- I learnt the basic theory behind **Yacc Parser Generator**.
- I learnt that Yacc stands for Yet Another Compiler-Compiler.
- I understood that Yacc is a LALR(1) parser.
- I understood Yacc's basic syntax and programming logic.
- I learnt that Yacc needs a Lex file along with it to work as intended, to detect and give the tokens to the Yacc parser.
- I was able to visualize how the parser works with the scanner.
- I learnt how to define a simple grammar in Yacc's syntax.
- I was able to implement a parser with Yacc to mimic the features of a desk calculator with precedence logic.
- I understood how to compile and run the Yacc and Lex file together.