॥वसुधैव कुटुम्बकम्॥

Symbiosis Institute of Technology, Pune

Department of Computer Science and Engineering

Academic Year 2025-26

Compiler Construction Lab

Batch 2022-26 – Sem VII

Lab Assignment No: - 7	
Name of Student	Soham Phadke
PRN No.	22070122214
Batch	2022-26
Class	CSE-C2
Academic Year & Semester	2025-26 Semester-7
Date of Submission	11/09/2025
Title of Assignment	Postfix Expression Evaluation.
Practice Questions	1. YACC program for Postfix Expression Evaluation. 2. YACC program for Conversion of Infix to Postfix expression.

Source Code

1. **YACC program for Postfix Expression Evaluation.** *LEX code:*

```
%{
#include<stdio.h>
#include "y.tab.h"
%}

digit   [0-9] number
{digit}+
operator [+\-*/]

%%
{number}  {yylval.n=atoi(yytext); return oprnd;}
{operator}  {return yytext[0];}
.
%%

int yywrap(){
```

```
return 1;
}
```

YACC code:

```
%{
#include <stdio.h>
extern int yylex(); int
yyerror(const char*);
int pop();
void push(int);

%}



%union {
   int n;
}
%token <n> oprnd

%%

S: | E { printf("\nResult=%d\n", pop()); }
;

E:E E '+' {
        int a, b;
a = pop();         b
= pop();
push(b + a);
   } |E E '-' {
int a, b;         a =
pop();         b =
pop();
push(b - a);
   }
   |E E '*' {
int a, b;         a =
pop();         b =
pop();
push(b * a);
   }
  |E E '/' {
int a, b;         a =
pop();         b =
pop();
push(b / a);
   }
  |oprnd {
push($1);
   }
;
```

	%%

```
void push(int val) {
stack[++top] = val;
} int pop() {     return
stack[top--];
}
int main() {     printf("\nEnter the postfix
expression: ");     yyparse();     return 0;
}

int yyerror(const char *s) {
fprintf(stderr, "\nError: %s\n", s);
return 0;
}
```

2. **YACC program for Conversion of Infix to Postfix expression.**

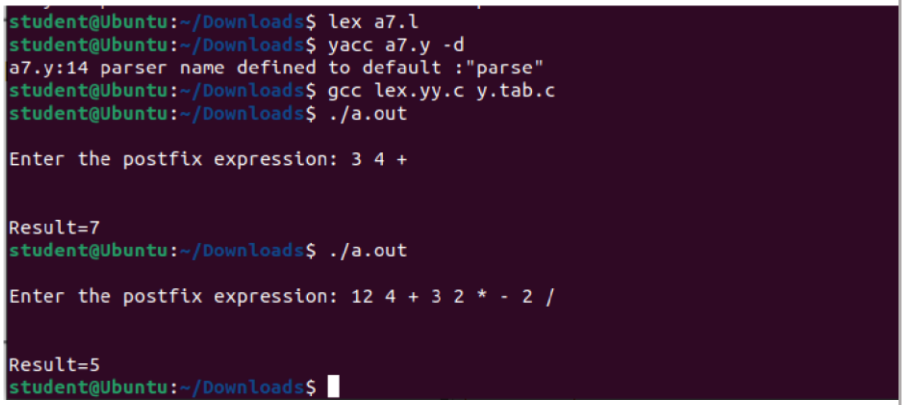
LEX Code:

```
%{
#include "y.tab.h"
%}

%%

[0-9] { yylval = yytext[0]; return DIGIT; }
[+\-*/()] { return yytext[0]; }
[ \t\n]  ;  /* skip whitespace */
.       { printf("Invalid character: %s\n", yytext); }

%%

int yywrap() {
return 1;
}
```

YACC Code:

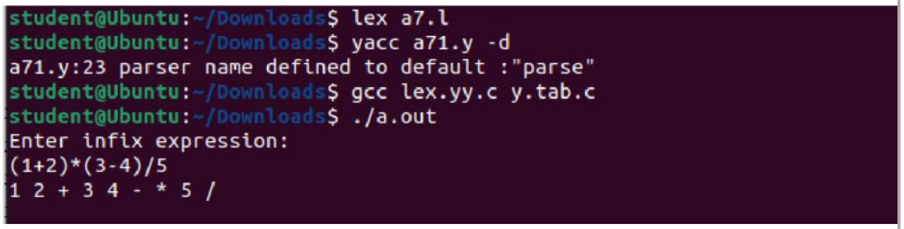
```
%{
#include <stdio.h> int yylex(void); void yyerror(const
char *s) { printf("Error: %s\n", s); }
%}

%token DIGIT
%left '+' '-'
%left '*' '/'

%%

expr:
      expr '+' expr  { printf("+ "); }
| expr '-' expr  { printf("- "); }
   | expr '*' expr  { printf("* "); }
   | expr '/' expr  { printf("/ "); }
   | '(' expr ')'   { /* grouping - no output */ }
  | DIGIT { printf("%c ", $1); }


   ;

%%

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

Output Screenshot



```
student@Ubuntu:~/Downloads$ lex a7.l
student@Ubuntu:~/Downloads$ yacc a7.y -d
a7.y:14 parser name defined to default :"parse"
student@Ubuntu:~/Downloads$ gcc lex.yy.c y.tab.c
student@Ubuntu:~/Downloads$ ./a.out

Enter the postfix expression: 3 4 +


Result=7
student@Ubuntu:~/Downloads$ ./a.out

Enter the postfix expression: 12 4 + 3 2 * - 2 /


Result=5
student@Ubuntu:~/Downloads$
```

Output for Postfix Expression Evaluation.



```
student@Ubuntu:~/Downloads$ lex a7.l
student@Ubuntu:~/Downloads$ yacc a71.y -d
a71.y:23 parser name defined to default :"parse"
student@Ubuntu:~/Downloads$ gcc lex.yy.c y.tab.c
student@Ubuntu:~/Downloads$ ./a.out
Enter infix expression:
(1+2)*(3-4)/5
1 2 + 3 4 - * 5 /
```

Output for for Conversion of Infix to Postfix expression

Post lab questions	YACC program for evaluating postfix expressions containing floating point numbers.
	<u>LEX Code:</u>
	%{
	#include <stdio.h>
	#include <stdlib.h> /* for atof */
	#include "y.tab.h"
	%}
	
	digit [0-9] number {digit}+(\.{digit}+)? /*
	integers or floats */ operator [+\-*/]

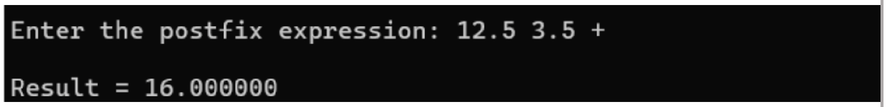
```
%%
{number}   { yylval.f = atof(yytext); return oprnd; }
{operator} { return yytext[0]; }
.          { ; }   /* catch all other characters and ignore */ [
\t\n]    { ; }   /* ignore whitespace */
%%

int yywrap() {     return
1;
}
```

YACC Code:

```
%{
#include <stdio.h>
#include <stdlib.h>

extern int yylex(); int
yyerror(const char*); float
pop(); void push(float);

float stack[100]; int
top = -1;
%}

%union {
   float f;
}
%token <f> oprnd

%%

S:
   | E { printf("\nResult = %f\n", pop()); }
;

E: E E '+' {         float
a = pop();         float b
= pop();         push(b +
a);
    }
  | E E '-' {         float a
= pop();         float b =
pop();         push(b -
a);
    }
  | E E '*' {         float a
= pop();         float b =
pop();
```

```
        push(b * a);
    }
  | E E '/' {         float
a = pop();          float
b = pop();
push(b / a);
    }
  | oprnd {
push($1);
    }
;

%%

void push(float val) {
stack[++top] = val;
} float pop()
{
    return stack[top--];
}

int main() {     printf("\nEnter the postfix
expression: ");     yyparse();     return 0;
}

int yyerror(const char *s) {
fprintf(stderr, "\nError: %s\n", s);
return 0;
}
```

<u>Output:</u>



```
Enter the postfix expression: 12.5 3.5 +

Result = 16.000000
```

Conclusion	The YACC and LEX programs successfully evaluate postfix expressions, including floating-point numbers, and convert infix to postfix form.