

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CD Practical 10

Aim : Implement YACC Program to evaluate a given arithmetic expression.

Code (p10.l) :

```
%{  
    /* Definition section*/  
    #include "p10.tab.h"  
    extern int yylval;  
}%  
  
%%  
[0-9]+ {  
    yylval = atoi(yytext);  
    return NUMBER;  
}  
  
[a-zA-Z]+ { return ID; }  
[ \t]+      ; /*For skipping whitespaces*/  
  
\n        { return 0; }  
.  
        { return yytext[0]; }  
  
%%  
int yywrap()  
{  
    return 0;  
}
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CD Practical 10

Code (p10.y) :

```
%{  
    #include <stdio.h>  
    int yylex(void);  
    void yyerror(const char *str) {  
        fprintf(stderr, "error: %s\n", str);  
    }  
%}  
  
%token NUMBER ID  
// setting the precedence  
// and associativity of operators  
%left '+' '-'  
%left '*' '/'  
  
/* Rule Section */  
%%  
E : T    {  
        printf("\n\tResult = %d\n\n", $$);  
        return 0;  
    }  
  
T :  
    T '+' T { $$ = $1 + $3; }  
    | T '-' T { $$ = $1 - $3; }  
    | T '*' T { $$ = $1 * $3; }  
    | T '/' T { $$ = $1 / $3; }  
    | '-' NUMBER { $$ = -$2; }
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CD Practical 10

```
| '-' ID { $$ = -$2; }  
| '(' T ')' { $$ = $2; }  
| NUMBER { $$ = $1; }  
| ID { $$ = $1; };
```

%%

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

Output :

The screenshot shows a code editor with two panes. The left pane displays the C code for a calculator, which includes a grammar for expressions and a main function that prompts the user for an expression. The right pane shows the output of the program, where the user has entered the expression '0+2/1-3/4+2' and the program has calculated the result as 13.

```
1 1 p10.y You, 13 seconds ago | 1 author (You)  
2 2  
3 3 #include <stdio.h>  
4 4 int yylex(void);  
5 5 void yyerror(const char *str) {  
6 6     fprintf(stderr, "error: %s\n", str);  
7 7 }  
8 8  
9 9 %token NUMBER ID  
10 10 // setting the precedence  
11 11 // and associativity of operators  
12 12 %left '+' '-'  
13 13 %left '*' '/'  
14 14  
15 15 /* Rule Section */  
16 16 %  
17 17 E : T {  
18 18     printf("\n\tResult = %d\n\n", $$);  
19 19     return 0;  
20 20 }  
21 21  
22 22 T :  
23 23     T '+' T { $$ = $1 + $3; }  
24 24     T '-' T { $$ = $1 - $3; }  
25 25     T '*' T { $$ = $1 * $3; }  
26 26     T '/' T { $$ = $1 / $3; }  
27 27     '-' NUMBER { $$ = -$2; }  
28 28     '-' ID { $$ = -$2; } You, 7 minutes ago • updated  
29 29     '(' T ')' { $$ = $2; }  
30 30     NUMBER { $$ = $1; }  
31 31     ID { $$ = $1; }  
32 32 %  
33 33  
34 34 int main() {  
35 35     printf("\nEnter the expression : ");  
36 36     yyparse();  
37 37     return 0;  
38 38 } <- #34-38 int main()  
39 39
```

0s .. //sem7practicals/CD/p10 • P main
flex p10.l ; bison -d p10.y ; gcc lex.yy.c p10.tab.c ; ./a.out
Enter the expression : 0+2/1-3/4+2
Result = 13
15s .. //sem7practicals/CD/p10 • P main