

UE20CS353-CD LAB-5

NAME	SRN	CLASS & SECTION
VIJAY J	PES2UG20CS815	6 - J

Lexer.l

```
lexer.l
~/PESU/6th Sem/CD/CD LAB/Week - 5/Lab - 5
Save

1 %{
2 // NAME: Vijay J
3 // SRN: PES2UG20CS815
4 // SECTION: J
5 #define YYSTYPE char*
6 #include <unistd.h>
7 #include "y.tab.h"
8 #include <stdio.h>
9 extern void yyerror(const char *); // declare the error handling function
10 %}
11
12 /* Regular definitions */
13 digit [0-9]
14 letter [a-zA-Z]
15 id {letter}({letter}|{digit})*
16 digits {digit}+
17 opFraction (\.{digits})?
18 opExponent ([Ee][+-]?{digits})?
19 number {digits}{opFraction}{opExponent}
20 %o /home/yoyo/Music
21
22 %%
23 \\/(.*) ; // ignore comments
24 [\t\n] ; // ignore whitespaces
25 "(" {return *yytext;}
26 ")" {return *yytext;}
27 "." {return *yytext;}
28 "," {return *yytext;}
29 "*" {return *yytext;}
30 "+" {return *yytext;}
31 ";" {return *yytext;}
32 "-" {return *yytext;}
33 "/" {return *yytext;}
34 "=" {return *yytext;}
35 ">" /home/yoyo/Music {return *yytext;}
36 "<" {return *yytext;}
37 {number} {
38     yylval = strdup(yytext); //stores the value of the number to be used
39     later for symbol table insertion
40     return T_NUM;
41 }
42 {id} {
43     yylval = strdup(yytext); //stores the identifier to be
44     used later for symbol table insertion
45     return T_ID;
46 }
47 . {} // anything else => ignore
48 %%
```

Parser.y

```
1 %{
2 // NAME: Vijay J
3 // SRN: PES2UG20CS815
4 // SECTION: J
5     #include "quad_generation.c"
6     #include <stdio.h>
7     #include <stdlib.h>
8     #include <string.h>
9
10    #define YYSTYPE char*
11
12    void yyerror(char*
13 s); //
14    error handling function
15    int
16    yylex();
17    declare the function performing lexical analysis
18    extern int
19    yylineno;
20    track the line number
21
22    FILE* icg_quad_file;
23    int temp_no = 1;
24 %}
25
26 %token T_ID T_NUM
27
28 /* specify start symbol */
29 %start START
```

```
parser.y
~/PESU/6th Sem/CD/CD LAB/Week - 5/Lab - 5
Save

27 %%
28 START : ASSGN {
29     printf("Valid syntax\n");
30
31     YYACCEPT; // If
32     program fits the grammar, syntax is valid
33 }
34 /* Grammar for assignment */
35 ASSGN : T_ID '=' E {
36     $$ = strdup($1);
37     char* op = strdup("=");
38     char* op1 = strdup(" ");
39     quad_code_gen($$, $3, op, op1);
40 }
41 ;
42 /* Expression Grammar */
43 E : E '+' T {
44     $$ = new_temp();
45     char* op = strdup("+");
46     quad_code_gen($$, $1, op, $3);
47 }
48 | E '-' T {
49     $$ = new_temp();
50     char* op = strdup("-");
51     quad_code_gen($$, $1, op, $3);
52 }
53 | T
54 ;
55
```

```
Open  parsery  Save  -  x
~/PESU/6th Sem/CD/CD LAB/Week - 5/Lab - 5

57 T : T '*' F {
58     $$ = new_temp();
59     char* op = strdup("*");
60     quad_code_gen($$, $1, op, $3);
61 }
62 | T '/' F {
63     $$ = new_temp();
64     char* op = strdup("/");
65     quad_code_gen($$, $1, op, $3);
66 }
67 | F
68 ;
69
70 F : '(' E ')' {
71     $$ = $2;
72 }
73 | T_ID {
74     $$ = strdup($1);
75 }
76 | T_NUM {
77     $$ = strdup($1);
78 }
79 ;
80
81 %%
82
83
84 /* error handling function */
85 void yyerror(char* s)
86 {
87     printf("Error :%s at %d \n",s,yylineno);
88 }

90 int yywrap() {
91     return 1;
92 }
93
94 /* main function - calls the yyparse() function which will in turn drive yylex() as well */
95 int main(int argc, char* argv[])
96 {
97     icg_quad_file = fopen("icg_quad.txt","w");
98     yyparse();
99     fclose(icg_quad_file);
100     return 0;
101 }
```

Quad_generation.c

```
Open  quad_generation.c  Save  -  x
~/PESU/6th Sem/CD/CD LAB/Week - 5/Lab - 5

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include "quad_generation.h"
5
6 void quad_code_gen(char* a, char* b, char* op, char* c)
7 {
8     fprintf(icg_quad_file, "%s, %s, %s, %s\n", op, b, c, a);
9 }
10
11 char* new_temp()
12 {
13     char* temp = (char*)malloc(sizeof(char)*4);
14     sprintf(temp, "t%d", temp_no);
15     ++temp_no;
16     return temp;
17 }
```

MakeFile

```
makefile
~/PESU/6th Sem/CD/CD LAB/Week - 5/Lab - 5

1 compiler: y.tab.c lex.yy.c
2     gcc y.tab.c lex.yy.c
3
4 y.tab.c: parser.y
5     yacc -d parser.y
6
7 lex.yy.c: lexer.l
8     lex lexer.l
9
10 clean:
11     rm *.yy.c y.tab.h y.tab.c a.out
```

Input and Output

Test_input_1.c

```
test_input_1.c
~/PESU/6th Sem/CD/CD LAB/Week - 5/Lab - 5

1 x = 9/2 + a-b
```

Output

```
~/P/6/C/C/P/Lab - 5

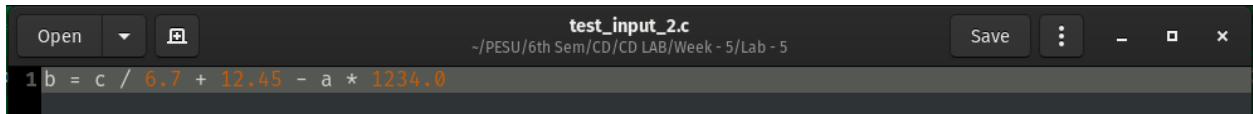
yoyo@zaemon in ~/PESU/6th Sem/CD/CD LAB/PES2UG20CS815/Lab - 5 via C v12.2.1-gcc took 2ms
λ make
yacc -d parser.y
lex lexer.l
gcc y.tab.c lex.yy.c

yoyo@zaemon in ~/PESU/6th Sem/CD/CD LAB/PES2UG20CS815/Lab - 5 via C v12.2.1-gcc took 309ms
λ ./a.out < test_input_1.c
Valid syntax

yoyo@zaemon in ~/PESU/6th Sem/CD/CD LAB/PES2UG20CS815/Lab - 5 via C v12.2.1-gcc took 2ms
λ cat icg_quad.txt
File: icg_quad.txt
/, 9, 2, t1
+, t1, a, t2
-, t2, b, t3
=, t3, , x

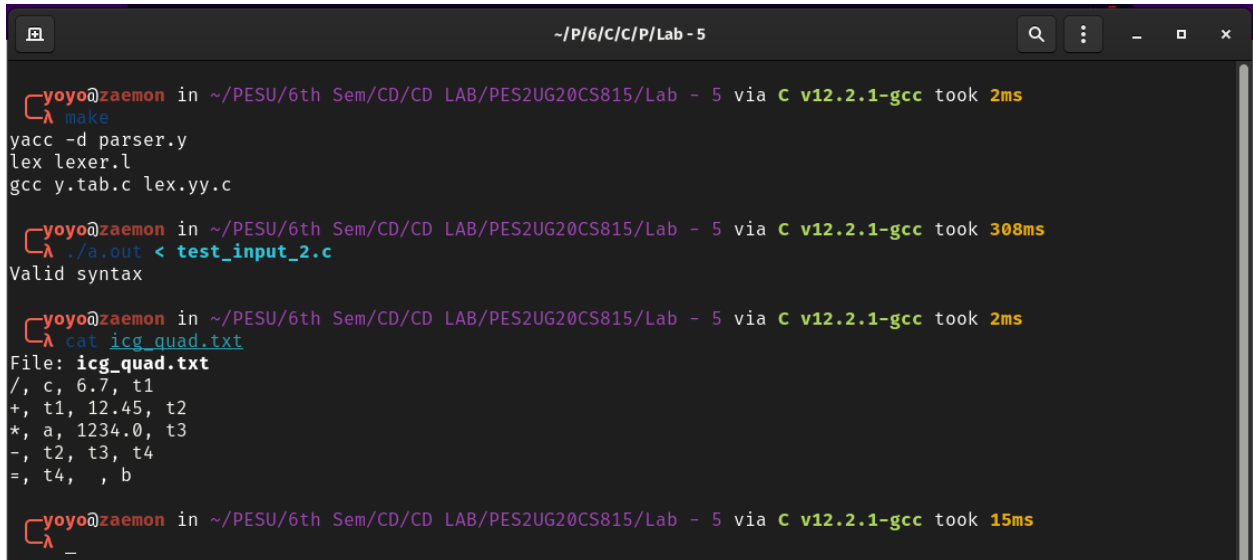
yoyo@zaemon in ~/PESU/6th Sem/CD/CD LAB/PES2UG20CS815/Lab - 5 via C v12.2.1-gcc took 68ms
λ _
```

Test_input_2.c



The screenshot shows a code editor window titled "test_input_2.c" with the file path "~/PESU/6th Sem/CD/CD LAB/Week - 5/Lab - 5". The editor contains a single line of code: `1 b = c / 6.7 + 12.45 - a * 1234.0`.

Output



The screenshot shows a terminal window with the following commands and output:

```
yoyo@zaemon in ~/PESU/6th Sem/CD/CD LAB/PES2UG20CS815/Lab - 5 via C v12.2.1-gcc took 2ms
λ make
yacc -d parser.y
lex lexer.l
gcc y.tab.c lex.yy.c

yoyo@zaemon in ~/PESU/6th Sem/CD/CD LAB/PES2UG20CS815/Lab - 5 via C v12.2.1-gcc took 308ms
λ ./a.out < test_input_2.c
Valid syntax

yoyo@zaemon in ~/PESU/6th Sem/CD/CD LAB/PES2UG20CS815/Lab - 5 via C v12.2.1-gcc took 2ms
λ cat icg_quad.txt
File: icg_quad.txt
/, c, 6.7, t1
+, t1, 12.45, t2
*, a, 1234.0, t3
-, t2, t3, t4
=, t4, , b

yoyo@zaemon in ~/PESU/6th Sem/CD/CD LAB/PES2UG20CS815/Lab - 5 via C v12.2.1-gcc took 15ms
λ _
```