

# CD Assignment 1

Name - Rlthvik Rajesh Matta

SRN - PES2UG23CS485

Class - 6H

```

CD > assignments > lex.l
1  %{
2  | #include "y.tab.h"
3  | #include <stdio.h>
4  | #include <stdlib.h>
5
6  | int line_no = 1;
7  | %}
8
9  | %%
10
11 | int          { return INT; }
12 | float        { return FLOAT; }
13 | char         { return CHAR; }
14 | double       { return DOUBLE; }
15
16 | if           { return IF; }
17 | else         { return ELSE; }
18 | do          { return DO; }
19 | while        { return WHILE; }
20 | for          { return FOR; }
21 | switch       { return SWITCH; }
22 | case         { return CASE; }
23 | default      { return DEFAULT; }
24 | break        { return BREAK; }
25
26 | "=="         { return EQ; }
27 | "!="         { return NE; }
28 | ">="         { return GE; }
29 | "<="         { return LE; }
30 | ">"          { return GT; }
31 | "<"          { return LT; }
32 | "&&"         { return AND; }
33 | "++"         { return INC; }
34
35 | "+"          { return '+'; }
36 | "-"          { return '-'; }
37 | "*"          { return '*'; }
38 | "/"          { return '/'; }
39 | "="          { return '='; }
40
41 | ";"          { return ';'; }
42 | ","          { return ','; }
43 | ":"          { return ':'; }
44 | "("          { return '('; }
45 | ")"          { return ')'; }
46 | "{"          { return '{'; }
47 | "}"          { return '}'; }
48 | "["          { return '['; }
49 | "]"          { return ']'; }
50
51 | [0-9]+        { return NUM; }
52 | [a-zA-Z_][a-zA-Z0-9]* { return ID; }
53
54 | [ \t\r]+      ;
55 | \n            { line_no++; }
56
57 | .             { return yytext[0]; }
58
59 | %%
60
61 | int yywrap(void)
62 | |
63 | |     return 1;
64 | |

```


Parser.y

```

CD / assignments > 5 parse.y
1  %{
2  #include <stdio.h>
3  #include <stdlib.h>
4
5  extern int yylex();
6  extern int line_no;
7  extern char *yytext;
8
9  void yyerror(const char *s);
10 %}
11
12 /* Tokens */
13 %token INT FLOAT CHAR DOUBLE
14 %token IF ELSE DO WHILE FOR
15 %token SWITCH CASE DEFAULT BREAK
16 %token ID NUM
17 %token EQ NE GE LE GT LT
18 %token AND INC
19
20 /* Operator precedence */
21 %left AND
22 %left GT LT GE LE EQ NE
23 %left '+' '-'
24 %left '*' '/'
25 %right '='
26 %right INC
27
28 /* Fix dangling else */
29 %nonassoc LOWER_THAN_ELSE
30 %nonassoc ELSE
31
32 %%
33
34 program:
35     | program statement
36     | /* empty */
37     ;
38
39 statement:
40     | declaration
41     | expression_stmt
42     | selection_stmt
43     | iteration_stmt
44     | compound_stmt
45     | BREAK ';'
46     ;
47
48 /* ----- DECLARATIONS ----- */
49
50 declaration:
51     | type declarator_list ';'
52     ;
53
54 type:
55     | INT
56     | FLOAT
57     | CHAR
58     | DOUBLE
59     ;
60
61 declarator_list:
62     | declarator
63     | declarator_list ',' declarator
64     ;
65
66 declarator:
67     | ID
68     | ID '=' expression
69     | ID array_dims
70     | ID array_dims '=' expression
71     ;
72
73 array_dims:
74     | '[' NUM '['
75     | array_dims '[' NUM '['
76     ;
77
78 /* ----- EXPRESSIONS ----- */
79
80 expression_stmt:
81     | expression ';'
82     ;
83
84 expression_list:
85     | expression
86     | expression_list ',' expression
87     ;
88
89 /* Array access for expressions */
90 array_access:
91     | ID '[' expression '['
92     | array_access '[' expression '['
93     ;
94
95 expression:
96     | ID '=' expression
97     | array_access '=' expression
98     | expression '+' expression
99     | expression '-' expression
100    | expression '*' expression
101    | expression '/' expression
102    | expression GT expression

```

```

CD > assignments >  parser.y
72
73 ▾ array_dims:
74     '[' NUM ']'
75     | array_dims '[' NUM ']'
76     ;
77
78 /* ----- EXPRESSIONS ----- */
79
80 ▾ expression stmt:
81     expression ';'
82     ;
83
84 ▾ expression_list:
85     expression
86     | expression_list ',' expression
87     ;
88
89 /* Array access for expressions */
90 ▾ array_access:
91     ID '[' expression ']'
92     | array_access '[' expression ']'
93     ;
94
95 ▾ expression:
96     ID '=' expression
97     | array_access '=' expression
98     | expression '+' expression
99     | expression '-' expression
100    | expression '*' expression
101    | expression '/' expression
102    | expression GT expression
103    | expression LT expression
104    | expression GE expression
105    | expression LE expression
106    | expression EQ expression
107    | expression NE expression
108    | expression AND expression
109    | ID INC
110    | array_access
111    | '(' expression ')'
112    | ID
113    | NUM
114    ;
115
116 /* ----- IF / SWITCH ----- */
117
118 ▾ selection_stmt:
119     IF '(' expression ')' statement %prec LOWER_THAN_ELSE
120     | IF '(' expression ')' statement ELSE statement
121     | SWITCH '(' expression ')' '{' case_list '}'
122     ;
123
124 ▾ case_list:
125     case_list case_stmt
126     | case_stmt
127     ;
128
129 ▾ case_stmt:
130     CASE NUM ':' program BREAK ';'
131     | DEFAULT ':' program
132     ;
133
134 /* ----- LOOPS ----- */
135
136 ▾ iteration_stmt:
137     DO statement WHILE '(' expression ')' ';'
138     | WHILE '(' expression ')' statement
139     | FOR '(' for_init ';' expression ';' for_update ')' statement
140     ;
141
142 ▾ for_init:
143     expression_list
144     | /* empty */
145     ;
146
147 ▾ for_update:
148     expression_list
149     | /* empty */
150     ;
151
152 /* ----- BLOCK ----- */
153
154 ▾ compound_stmt:
155     '{' program '}'
156     ;
157
158 %%
159
160 void yyerror(const char *s)
161 {
162     printf("Syntax error at line %d, token '%s': %s\n", line_no, yytext, s);
163 }
164
165 int main()
166 {
167     if (yyparse() == 0)
168         printf("Syntax valid.\n");
169     return 0;
170 }

```

## Variable declaration

```
CD > assignments > C test1.c
1  int a=5, b, c=10;
2  float x;
3  char ch;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$ ./parser < test1.c  
Syntax valid.
- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$

## Array

```
CD > assignments > C test1.c
1  int a[10];
2  int b[5][5];
3  int c[1][2][3][4];
4  int d[4][4], e[5];
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$ ./parser < test1.c  
Syntax valid.
- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$

## For loop

```
CD > assignments > C test1.c
1  int i;
2  for(i=0; i<10; i++)
3  |    i = i + 1;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$ ./parser < test1.c  
Syntax valid.
- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$ █

## Complex for loop

```
CD > assignments > C test1.c
1  int i,j,p,q;
2
3  for(i=0,j=0; i<p && j<q; i++,j++)
4  {
5  |    i = i + 1;
6  }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$ ./parser < test1.c  
Syntax valid.
- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$ █

## While

```
CD > assignments > C test1.c
1  int a=0;
2
3  while(a<10)
4  {
5      a++;
6  }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$ ./parser < test1.c  
Syntax valid.
- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$

## Switch

```
CD > assignments > C test1.c
1  int a,b;
2
3  switch(a)
4  {
5      case 1:
6          b = 10;
7          break;
8      case 2:
9          b = 20;
10         break;
11     default:
12         b = 0;
13 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$ ./parser < test1.c  
Syntax valid.
- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$

## Combined



```
CD > assignments > C test1.c
1  int a=5, b[10], c[5][5];
2
3  for(a=0; a<10; a++)
4  {
5      switch(a)
6      {
7          case 1:
8              b[a] = 10;
9              break;
10         default:
11             b[a] = 0;
12         }
13     }
14
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
● pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments$ ./parser < test1.c
  Syntax valid.
○ pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments$
```

Invalid

Var

```
CD > assignments > C test1.c
1  int a=5
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$ ./parser < test1.c  
Syntax error at line 1, token '=': syntax error
- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$

Wrong syntax

```
CD > assignments > C test1.c
1  int a[];
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$ ./parser < test1.c  
Syntax error at line 1, token ']': syntax error
- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$

Incorrect for loop

```
CD > assignments > C test1.c
1  for(i=0 i<10; i++)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$ ./parser < test1.c  
Syntax error at line 1, token 'i': syntax error
- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$

Missing break

```
CD > assignments > C test1.c
1  switch(a)
2  {
3      case 1:
4          b = 10;
5          break
6  }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$ ./parser < test1.c  
Syntax error at line 6, token '}': syntax error
- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$

Invalid initialization

```
CD > assignments > C test1.c
```

```
1  int a = ;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$ ./parser < test1.c  
Syntax error at line 1, token ';': syntax error
- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$

Invalid while

```
CD > assignments > C test1.c
```

```
1  while a<10)  
2  |    a++;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$ ./parser < test1.c  
Syntax error at line 1, token 'a': syntax error
- pes2ug23cs485@pes2ug23cs485:~/sem6/CD/assignments\$