

Aayush Shah

19BCE245

1 October 2022

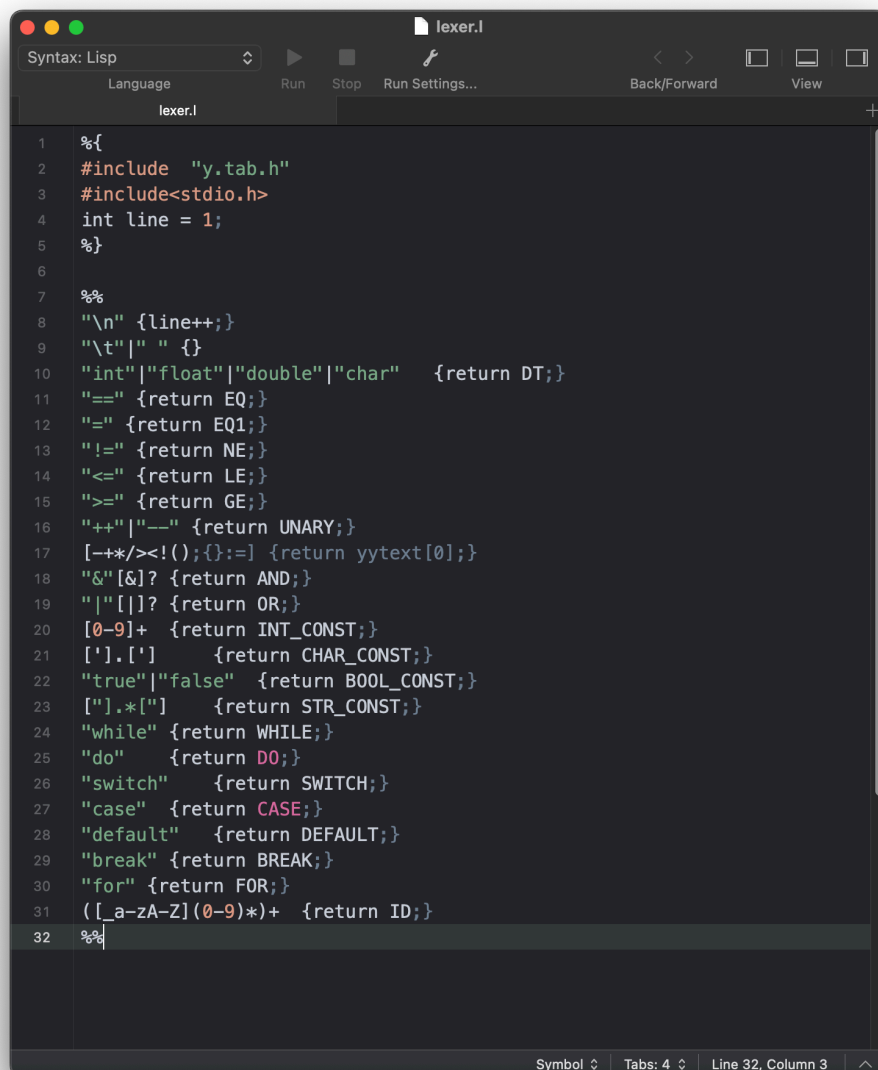
# Compiler Construction

## Practical 7

**To implement lexical analyse to recognize all distinct token classes**

• **Code :**

**lexer.l**



```
1 %{
2 #include "y.tab.h"
3 #include<stdio.h>
4 int line = 1;
5 %}
6
7 %%
8 "\n" {line++;}
9 "\t"|" " {}
10 "int"|"float"|"double"|"char" {return DT;}
11 "==" {return EQ;}
12 "=" {return EQ1;}
13 "!=" {return NE;}
14 "<=" {return LE;}
15 ">=" {return GE;}
16 "++"|"--" {return UNARY;}
17 "[-*>/<!(~);{}:=]" {return yytext[0];}
18 "&"[&]? {return AND;}
19 "|"[]? {return OR;}
20 "[0-9]+" {return INT_CONST;}
21 "[']"[']' {return CHAR_CONST;}
22 "true"|"false" {return BOOL_CONST;}
23 "["].*["] {return STR_CONST;}
24 "while" {return WHILE;}
25 "do" {return DO;}
26 "switch" {return SWITCH;}
27 "case" {return CASE;}
28 "default" {return DEFAULT;}
29 "break" {return BREAK;}
30 "for" {return FOR;}
31 "[_a-zA-Z](0-9)*" {return ID;}
32 %%
```

parser.l

```

1  %{
2  #include<stdio.h>
3  int yylex();
4  void yyerror(char *msg);
5  void yywrap();
6  int
7      count_for=0,count_if=0,count_while=0,count_switch=0,count_do_while
8      =0;
9  extern int line;
10 %}
11 %token WHILE DO SWITCH CASE DEFAULT BREAK DT ID INT_CONST BOOL_CONST
12     CHAR_CONST STR_CONST LE GE EQ EQ1 NE AND OR FOR EQSN IDSET
13
14 %right '='
15 %left AND OR
16 %left '<' '>' LE GE EQ NE
17 %left '+' '-'
18 %left '*' '/'
19 %right UNARY
20 %left '!'
21
22 %%
23
24 PROG : DT ID '(' ')' BLK;
25
26 BLK :  '{' BS '}' ;
27
28 BS : SS
29     |
30     ;
31
32 SS :  S SS
33     |  S
34     ;
35
36 S :  DO_WHILE_STMT

```

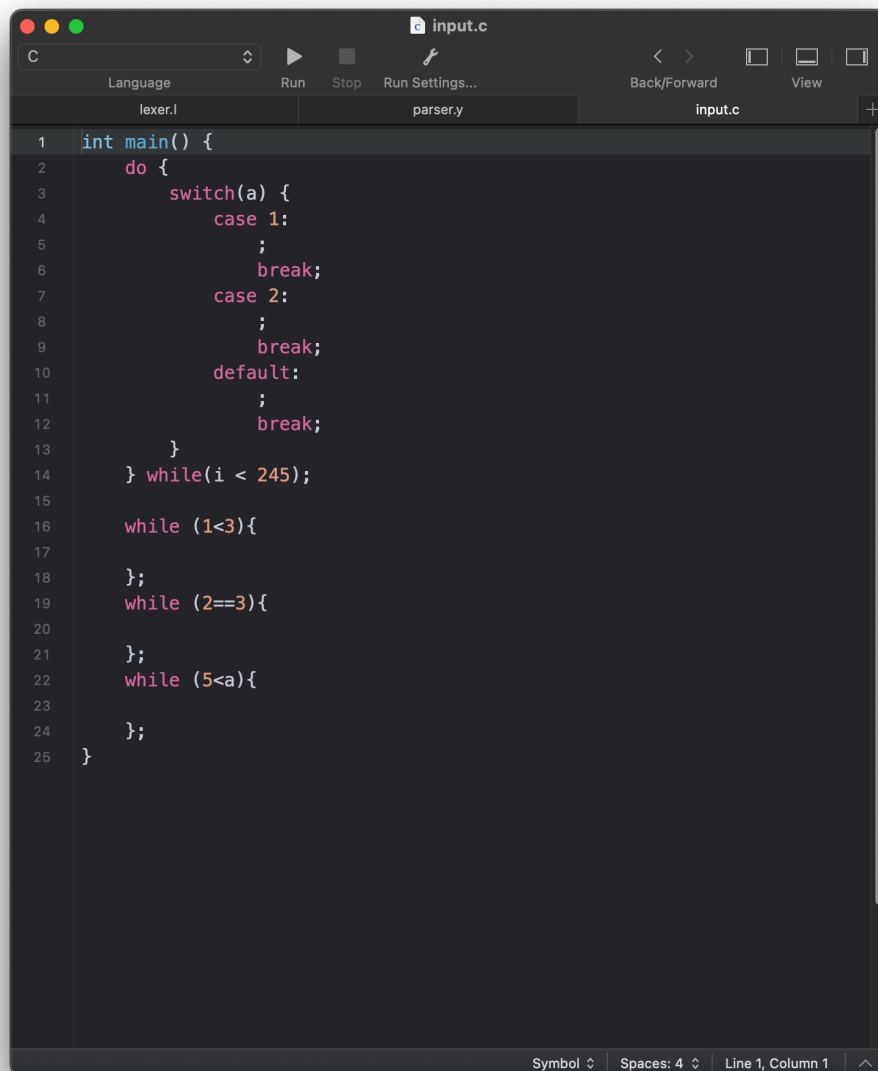
```
parser.y
Syntax: Bison
Run Stop Run Settings... Back/Forward View
lexer.l parser.y
34 | WHILE_STMT
35 | SWITCH_STMT
36 | BLK
37 | E ';'
38 | ';'
39 ;
40
41 SWSS: SS BRK
42 | BRK
43 ;
44
45 BRK : BREAK ';'
46 |
47 ;
48
49
50 ;
51
52 WHILE_STMT : WHILE '(' E ')' BLK {count_while++;};
53
54 DO_WHILE_STMT : DO BLK WHILE '(' E ')' ';' {count_do_while++;};
55
56 SWITCH_STMT : SWITCH '(' E ')' '{' CASE_STMTS DEFAULT_STMT
57              '}' {count_switch++;}
58              ;
59
60 CASE_STMTS : CASE_STMT CASE_STMTS
61            | CASE_STMT
62            ;
63
64 CASE_STMT : CASE CONST ':' SWSS
65           ;
66
67 DEFAULT_STMT : DEFAULT ':' SWSS
68              |
69              ;
```

```

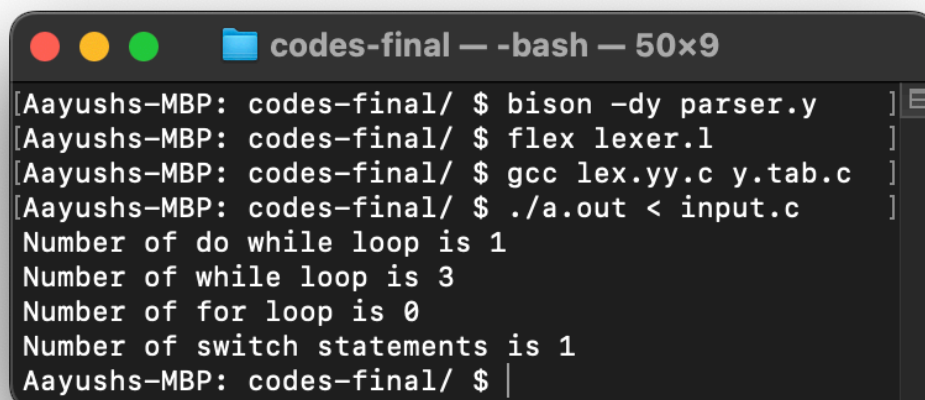
68      ;
69
70  E    :  E '+' E
71        |  E '-' E
72        |  E '*' E
73        |  E '/' E
74        |  E '<' E
75        |  E '>' E
76        |  E EQ E
77        |  E NE E
78        |  E GE E
79        |  E LE E
80        |  ID
81        |  CONST
82      ;
83
84  CONST :  INT_CONST
85         |  CHAR_CONST
86         |  BOOL_CONST
87         |  STR_CONST
88      ;
89
90  %%
91
92  int main() {
93      yyparse();
94      return 0;
95  }
96
97  void yyerror(char *msg) {
98      printf("Error Message :- %s at line %d\n", msg, line);
99  }
100
101  void yywrap() {
102      printf("Number of do while loop is %d\n", count_do_while);
103      printf("Number of while loop is %d\n", count_while);
104      printf("Number of for loop is %d\n", count_for);
105      printf("Number of switch statements is %d\n", count_switch);
106  }
107

```

Symbol ⚙ Spaces: 4 ⚙ Line 85, Column 23 ^

input.c

```
1 int main() {
2     do {
3         switch(a) {
4             case 1:
5                 ;
6                 break;
7             case 2:
8                 ;
9                 break;
10            default:
11                ;
12                break;
13        }
14    } while(i < 245);
15
16    while (1<3){
17
18    };
19    while (2==3){
20
21    };
22    while (5<a){
23
24    };
25 }
```

output

```
codes-final — -bash — 50x9
Aayushs-MBP: codes-final/ $ bison -dy parser.y
Aayushs-MBP: codes-final/ $ flex lexer.l
Aayushs-MBP: codes-final/ $ gcc lex.yy.c y.tab.c
Aayushs-MBP: codes-final/ $ ./a.out < input.c
Number of do while loop is 1
Number of while loop is 3
Number of for loop is 0
Number of switch statements is 1
Aayushs-MBP: codes-final/ $
```