

19CSE401 - Compiler Design

# **Lexical Analyser Tool: LEX**

Lab sheet-2

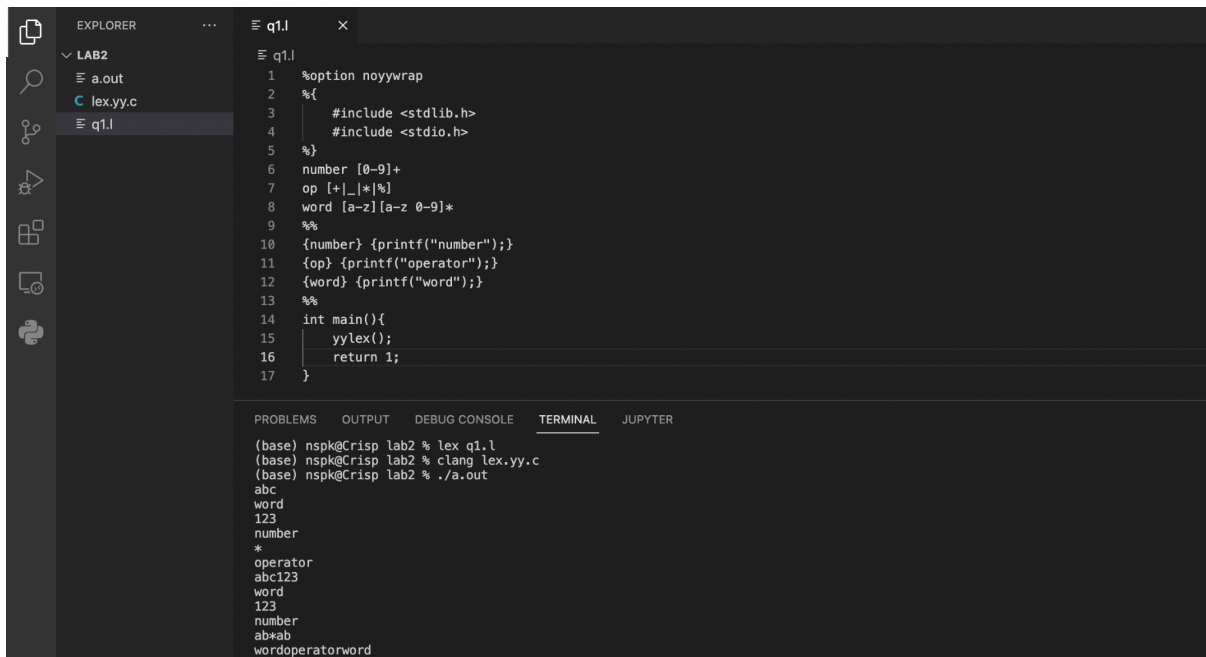
## **Index**

- Sample Program
- Keywords, Operators and Symbols Identification

**Done By:**

N Sai Pavan Krishna  
AM.EN.U4CSE19347

## Q1. Sample Program



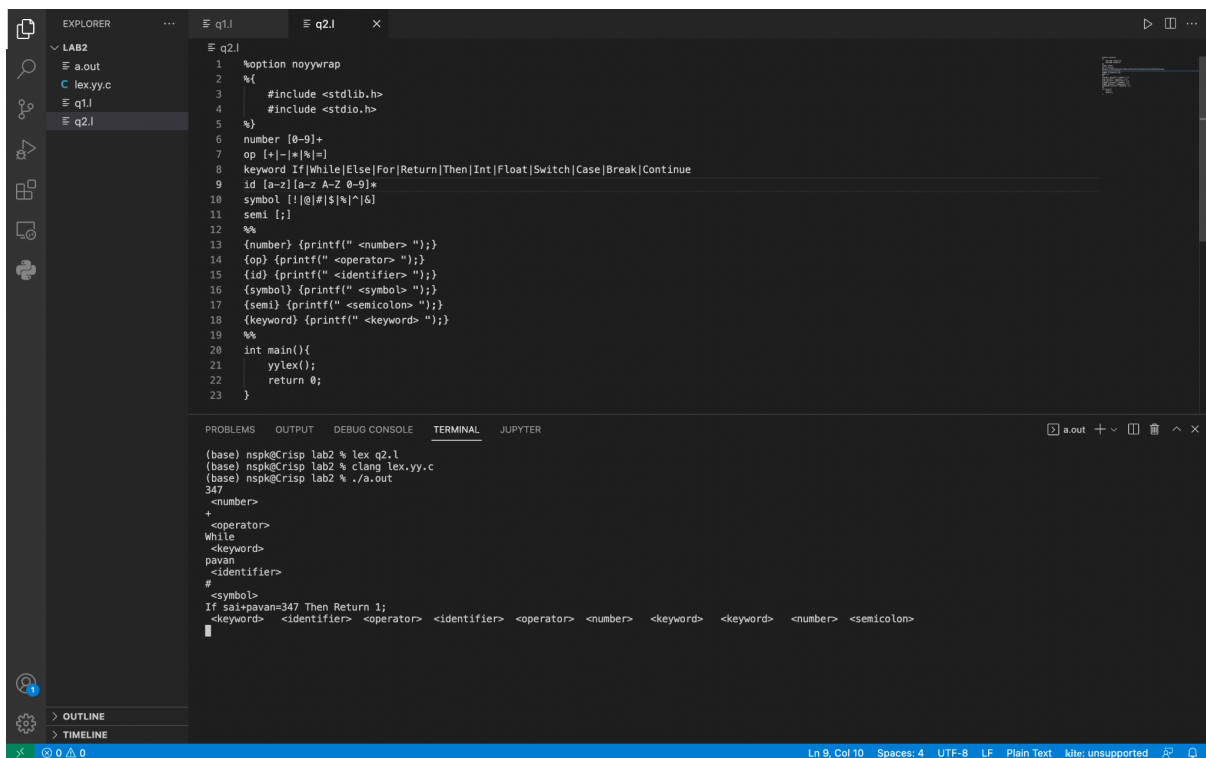
The screenshot shows a Visual Studio Code editor with a file explorer on the left containing 'LAB2', 'a.out', 'lex.yy.c', and 'q1.l'. The main editor displays the content of 'q1.l', a C program using the flex library. The terminal at the bottom shows the compilation and execution process, including the command 'clang lex.yy.c' and the output of the program, which identifies tokens in the input string 'abc123 word number \* operator abc123 word number ab\*ab wordoperatorword'.

```
1 %option noyywrap
2 %{
3     #include <stdlib.h>
4     #include <stdio.h>
5 %}
6 number [0-9]+
7 op [+|_|*|%]
8 word [a-z][a-z 0-9]*
9 %%
10 {number} {printf("number");}
11 {op} {printf("operator");}
12 {word} {printf("word");}
13 %%
14 int main(){
15     yylex();
16     return 1;
17 }
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** JUPYTER

```
(base) nspk@Crisp lab2 % lex q1.l
(base) nspk@Crisp lab2 % clang lex.yy.c
(base) nspk@Crisp lab2 % ./a.out
abc
word
123
number
*
operator
abc123
word
123
number
ab*ab
wordoperatorword
```

## Q2. Do a program to identify the Keywords, operators and Symbols in the C program



The screenshot shows a Visual Studio Code editor with a file explorer on the left containing 'LAB2', 'a.out', 'lex.yy.c', 'q1.l', and 'q2.l'. The main editor displays the content of 'q2.l', a C program using the flex library to identify keywords, operators, and symbols in a C program. The terminal at the bottom shows the compilation and execution process, including the command 'clang lex.yy.c' and the output of the program, which identifies tokens in the input string '347 <number> + <operator> While <keyword> pavan <identifier> # <symbol> If sai+pavan=347 Then Return 1; <keyword> <identifier> <operator> <identifier> <operator> <number> <keyword> <keyword> <number> <semicolon>'.

```
1 %option noyywrap
2 %{
3     #include <stdlib.h>
4     #include <stdio.h>
5 %}
6 number [0-9]+
7 op [+|_|*|%]
8 keyword If|While|Else|For|Return|Then|Int|Float|Switch|Case|Break|Continue
9 id [a-z][a-z A-Z 0-9]*
10 symbol [|@#|_|$|^|&]
11 semi [:]
12 %%
13 {number} {printf(" <number> ");}
14 {op} {printf(" <operator> ");}
15 {id} {printf(" <identifier> ");}
16 {symbol} {printf(" <symbol> ");}
17 {semi} {printf(" <semicolon> ");}
18 {keyword} {printf(" <keyword> ");}
19 %%
20 int main(){
21     yylex();
22     return 0;
23 }
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** JUPYTER

```
(base) nspk@Crisp lab2 % lex q2.l
(base) nspk@Crisp lab2 % clang lex.yy.c
(base) nspk@Crisp lab2 % ./a.out
347
<number>
+
<operator>
While
<keyword>
pavan
<identifier>
#
<symbol>
If sai+pavan=347 Then Return 1;
<keyword> <identifier> <operator> <identifier> <operator> <number> <keyword> <keyword> <number> <semicolon>
```