



**Alexandria University CS321**  
**Faculty of Engineering**  
**Compilers**  
**Computer and Systems Engineering Dept.**  
**Third Year**

**Phase1: Lexical Analyzer Generator**  
**Bonus Report**

Names	ID
Peter Atef	19
Radwa Adel	23
Aliaa Othman	41
Mark Philip	49

## **What is Flex ?**

Flex is an **open source** program designed to automatically and quickly generate scanners, also known as tokenizers, which recognize lexical patterns in text. Flex is an acronym that stands for "fast lexical analyzer generator. " It is a free alternative to Lex, the standard lexical analyzer generator in **Unix**-based systems. Flex was originally written in the **C** programming language by Vern Paxson in 1987.

## **Detailed description of the required steps :**

1- **install Flex using the following two commands (ubuntu)**

```
sudo apt-get update  
sudo apt-get install flex
```

2- **for compiling a lex program :**

- write the lex program in a file and save it as file.l (where file is the name of the file).
- open the terminal and navigate to the directory where you have saved the file.l
- type - `lex file.l`
- then type - `cc lex.yy.c -lfl`
- then type - `./a.out`

## **screenshots for using the tool :**

### **1-file.l**

```

1 letter  [a-z]|[A-Z]
2 digit   [0-9]
3 digits  [0-9]+
4 %%
5 boolean
6 int
7 float
8 if
9 else
10 while
11
12 {letter}({letter}|{digit})*
13 {digit}+|{digit}+"."{digits}+("\n"|E{digits}+)
14 ==|!=|>|>=|<|<=
15 =
16 "+"|-
17 "*"|"|"
18 ";"
19 "
20 "("
21 ")"
22 "{"
23 "}"
24 "]"
25 %%
26
27 main()
28 {
29     printf("Enter your input:\n");
30     yylex();
31 }

```

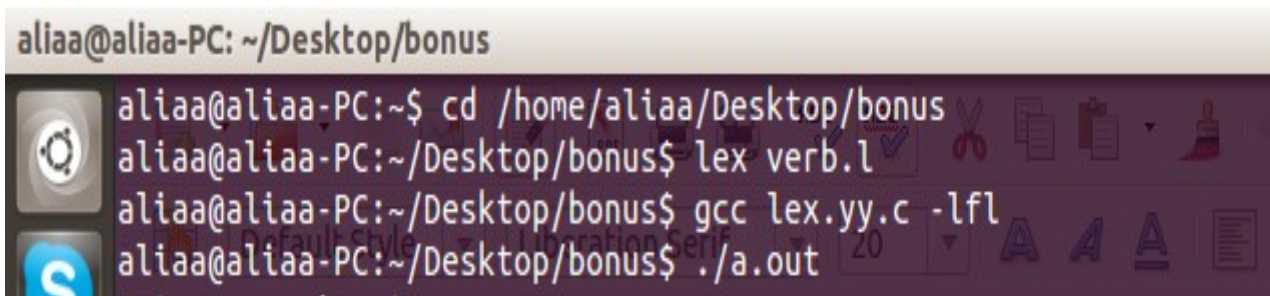
```

printf("boolean\n");
printf("int\n");
printf("float\n");
printf("if\n");
printf("else\n");
printf("while\n");

printf("id\n");
printf("num\n");
printf("relop\n");
printf("assign\n");
printf("addop\n");
printf("mulop\n");
printf(";\n");
printf(",\n");
printf("( \n");
printf(")\n");
printf("{\n");
printf("}\n");
printf("]\n");
printf("]\n");

```

## 2-compiling the file and generating output for test cases:



A terminal window with a dark background and light-colored text. The prompt is 'aliaa@aliaa-PC: ~/Desktop/bonus'. The user enters three commands: 'cd /home/aliaa/Desktop/bonus', 'lex verb.l', and 'gcc lex.yy.c -lfl'. The output shows the directory change, the output of the lex command, and the execution of the gcc command. The terminal also shows a file manager icon on the left and a toolbar on the right.

```

aliaa@aliaa-PC: ~/Desktop/bonus
aliaa@aliaa-PC:~$ cd /home/aliaa/Desktop/bonus
aliaa@aliaa-PC:~/Desktop/bonus$ lex verb.l
aliaa@aliaa-PC:~/Desktop/bonus$ gcc lex.yy.c -lfl
aliaa@aliaa-PC:~/Desktop/bonus$ ./a.out

```

aliaa@aliaa-PC: ~/Desktop/bonus

Enter your input:  
int sum , count , pass ,

mnt; while (pass != 10)

{

pass = pass + 1 ;

}int

id

, Music

id

, Pictures

id

, Videos

id

, Trash

id

LENOVO

;

while

(

aliala

id

Windows8\_OS

id

LRS\_ESP

relop

num Computer

)

Network

{

Browse Network

id

assign

id

addop

num

;

}



a.out



lex.yy.c



verb.l

aliaa@aliaa-PC: ~/Desktop/bonus

aliaa@aliaa-PC:~\$ cd /home/aliaa/Desktop/bonus

aliaa@aliaa-PC:~/Desktop/bonus\$ lex verb.l

aliaa@aliaa-PC:~/Desktop/bonus\$ gcc lex.yy.c -lfl

aliaa@aliaa-PC:~/Desktop/bonus\$ ./a.out

Enter your input:

222.33E55

num

<>aliaa

relop

relop

id

boolean

boolean for int /\*

boolean eos

id

int

mulop

mulop

for(int i =0 ; i<5 ; i++){

(

int

id

assign

num

;

id

relop

num

;

id

addop

addop

)

{

}

a.out

lex.yy.c

verb.l