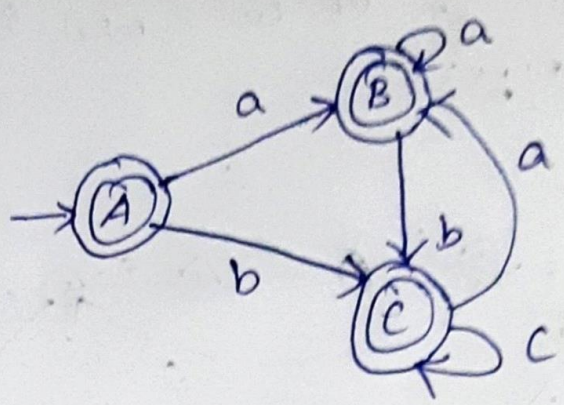
Transition Table.

	a	b
→*A	B	C
*B	B	C
*C	B	C

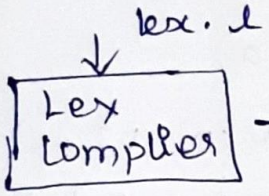


Lex Tool:-

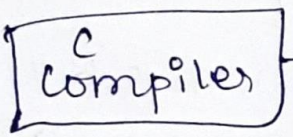
→ Its main Job is to beat I/p an I/p stream into the tokens.

→ Lex is a tool for automatically generating a lexer.

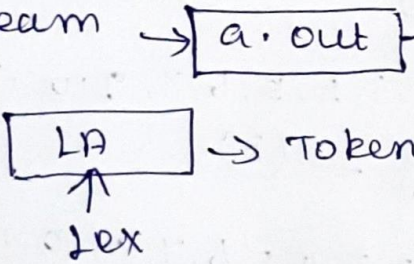
Step1:



Step2:



Step3:

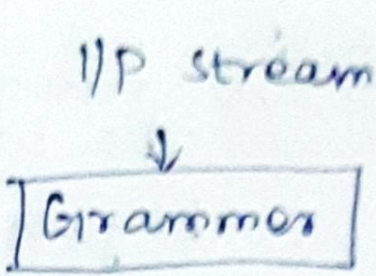


Structure of lex program:-

```
{ declaration  → Declare the Variable.
}
%. %.
{ translation rules → have the patter Action {}
} %. %.
{ auxilary functions → Functions can be Complied
}                      Separately.
```

Ex: lex Program:

⇒ Program to count the no. of vowles & constants in a given grammar.




```
%{
#include<stdio.h>
int vowels=0;
int cons=0;
%}
%%
[a e i o u A E I O U]  {vowels++;}
[a-z A-Z]             {const ++;}
%%
int yywrap()
{ return 1;
}
main() {
Pf("Enter the string at end press ^d \n");
yylex(); → lex tool
Pf("no. of vowels = %d \n
    no. of constants = %d \n", vowels, cons); }.
```