Experiment No: 19

- 11	
	Per course
).	Min?: Write as programs to eliminate meft
	secursion from the given grammer. (1.10)
	Theory:
-	- A production of the grammar is saidy to have
-	left recursion if the leftmost variable of
	its RHS is some as variable of its LHS.
	Asugrammer Gevitobbis) is left secureire
	Because the left production in the form
•	$A \rightarrow A \propto B $
	The American Comment of the Comment
-	The above grammar is left recursive because
	the left of production is only at first position
	on right side of production.
-	We can replace left recursion by replacing
	a pair bafresproduction withis xs
	a pair of sproduction withinsxs
	A > aA'beach
	s otolumia
-	In left recursive grammar, expansion of A
	will generate Ax, Axx, Axx on each side,
	causing it to enter into an infinite loop.
•	Example:
	Modisinos F F F F F F F F F F F F F F F F F F F
	T-) TAFIF moreous
	(E) id
y	Compare E + E+T T

Plich toamingx) lex source program storing > Lex Compiler story lexity.c (len. f) monorp assign the most noisyusses notes wo arouf lenipylico = compiler left recursion if the leftmest variable of its RHS is same as variable of itigati Sequence of tokens Streams 131 31 Anout left production in the form errored grieruses figel si rommore evodo. the left of production is only at hist polition can achlore lette recordion by replacing rexemptifo not suborformand ripy begin A A A Automation Ax CA somulator In 1eft reconsive grammar, expansion of A A A & AKK , AKK ON POCK Side, to enter into an infinite loop. Transition compiler program ouc tron T 1 T+3 1 3 sabour. alaA + A Ation

4 A' 0 ' 2



	ENGINEERING COLLEGE	
	AST CO A JO LOVERISA SALVA CO A	
	3 (A >A = E ; X = +B = -	
	$A \rightarrow A \times B \rightarrow change to \rightarrow A' \rightarrow A'$	3118
	: A BA' means E -> TE'	
	Compare THT FIF	
	A=T = X=++, B=F	1
	A = BA' means T' -> *FT' E)
	(A)	<u> </u>
	Production F + (E) ! idprovdoes not have any !	6 F +
3	recursion. +331 33	
90	osecursion (x)	-
3	∴ E → TE'	
	E' -> +TE' E	-
	T -> F-7'	(A
	T, -> + Ed, 8	
	F -> (E) id	
•	about left recursion and how to manage	
	remove left recursion.	
) (803 2094	
	1031	
28		

A -> AalB Removal of

| Removal of | A -> BA' | A -> AA' | E change to ton Removabil(3) + 7 recursion 31/+7+ 6 / bil (a) <= 3 Conduction: In this experiment about left secussion and how ave left recursion

Code:

```
import re
print("Enter the grammar")
gm = {"A->ABd|Aa|a", "B->Be|b"}
aplha = []
beta = []
for i in gm:
   # print(re.split(r'->|\|', i))
   exp = re.split(r"->|\|", i)
   nt = exp[0]
   cnt = 0
    for i in exp:
        if cnt == 0:
            cnt += 1
            continue
        else:
            if i[0] == exp[0]:
                aplha.append(i[1:])
            else:
                beta.append(i)
   # print('alpha',aplha)
   # print('beta',beta)
   # use left recursion
   print(exp[0], "->", end="")
    for i in beta:
        # print(i+exp[0]+'\'','|', end='')
       # if last beta dont add | else add |
        if i == beta[-1]:
            print(i + exp[0] + "'", end="")
            print(i + exp[0] + "'", "|", end="")
    print("\n", exp[0] + "'", "->", end="", sep="")
   for i in aplha:
        if i == aplha[-1]:
            print(i + exp[0] + "'", "|", "e", end="")
            print(i + exp[0] + "'", "|", end="")
    alpha = []
    beta = []
    print("\n\n")
```

Output: