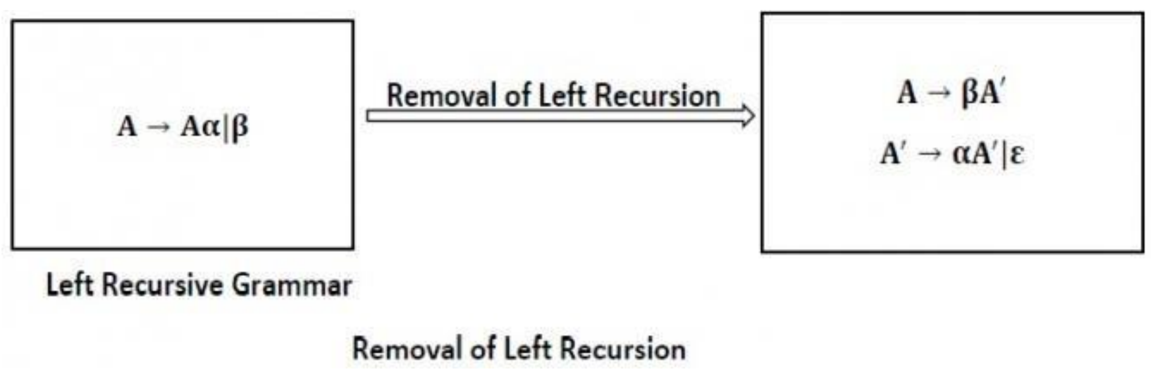
19BCE248

Dhruvil Shah

Compiler construction

Practical 4

Aim: To eliminate direct and indirect left recursion from LL1 grammer



Example:

$E \rightarrow E + T|T$

$T \rightarrow T * F|F$

$F \rightarrow (E)|id$

Ans :

$E \rightarrow TE'$

$E' \rightarrow +TE'| \in$

$T \rightarrow FT'$

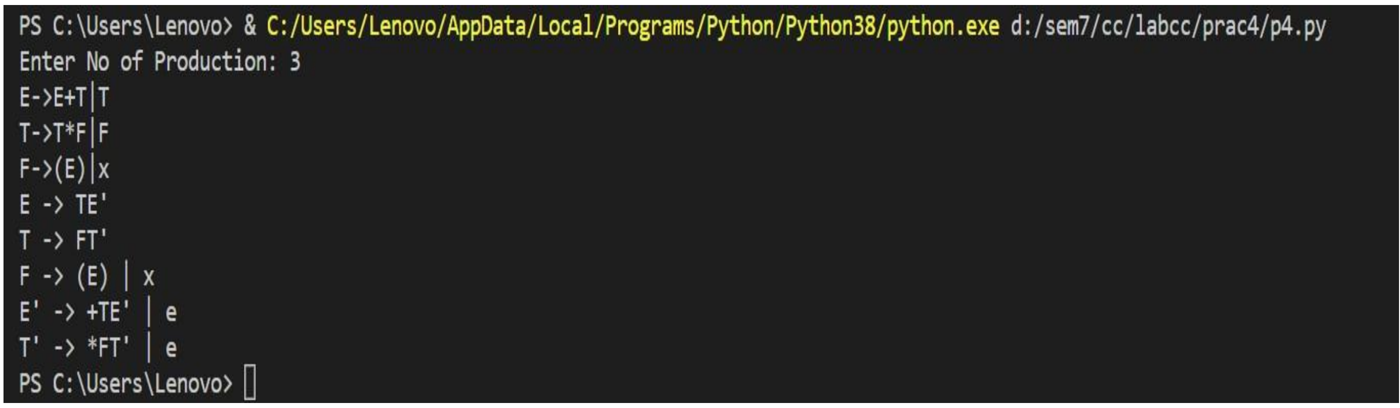
$T' \rightarrow * FT'| \in$

$F \rightarrow (E)|id$

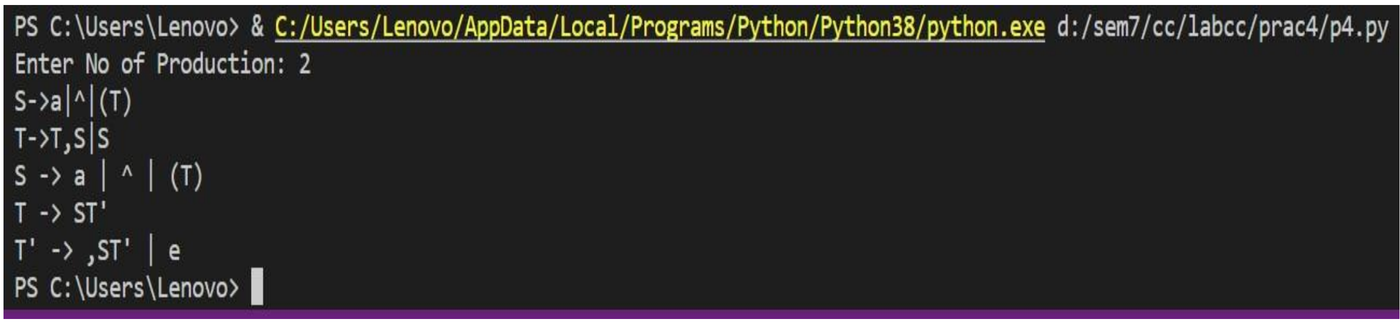
Code:

```python
##Note : This code is direct and indirect left recursion

#  'e' means ϵ (epsalion)



gram = {}

def add(str):                                       #to rules together
    str = str.replace(" ", "").replace("    ", "").replace("\n", "")
    x = str.split("->")
    y = x[1]
    x.pop()
    z = y.split("|")
    x.append(z)
    gram[x[0]]=x[1]

def removeDirectLR(gramA, A):
    """gramA is dictonary"""
    temp = gramA[A]
    tempCr = []
    tempInCr = []
    for i in temp:
        if i[0] == A:
            #tempInCr.append(i[1:])
            tempInCr.append(i[1:]+[A+"'"])
        else:
            #tempCr.append(i)
            tempCr.append(i+[A+"'"])
    tempInCr.append(["e"])
    gramA[A] = tempCr
    gramA[A+"'"] = tempInCr
    return gramA


def checkForIndirect(gramA, a, ai):
    if ai not in gramA:
        return False
    if a == ai:
        return True
    for i in gramA[ai]:
        if i[0] == ai:
            return False
        if i[0] in gramA:
            return checkForIndirect(gramA, a, i[0])
    return False

def rep(gramA, A):
    temp = gramA[A]
    newTemp = []
    for i in temp:
        if checkForIndirect(gramA, A, i[0]):
            t = []
            for k in gramA[i[0]]:
                t=[]
                t+=k
                t+=i[1:]
                newTemp.append(t)

        else:
            newTemp.append(i)
    gramA[A] = newTemp
    return gramA

def rem(gram):
    c = 1
    conv = {}
    gramA = {}
    revconv = {}
    for j in gram:
        conv[j] = "A"+str(c)
        gramA["A"+str(c)] = []
        c+=1

    for i in gram:
        for j in gram[i]:
            temp  =  []
            for k in j:
                if k in conv:
                    temp.append(conv[k])
                else:
                    temp.append(k)
            gramA[conv[i]].append(temp)


    #print(gramA)
    for i in range(c-1,0,-1):
        ai = "A"+str(i)
        for j in range(0,i):
            aj = gramA[ai][0][0]
            if ai!=aj :
                if aj in gramA and checkForIndirect(gramA,ai,aj):
                    gramA = rep(gramA, ai)
    for i in range(1,c):
        ai = "A"+str(i)
        for j in gramA[ai]:
            if ai==j[0]:
                gramA = removeDirectLR(gramA, ai)
                break

    op = {}
    for i in gramA:
        a = str(i)
        for j in conv:
            a = a.replace(conv[j],j)
        revconv[i] = a

    for i in gramA:
        l = []
        for j in gramA[i]:
            k = []
            for m in j:
                if m in revconv:
                    k.append(m.replace(m,revconv[m]))
                else:
                    k.append(m)
            l.append(k)
        op[revconv[i]] = l

    return op

n = int(input("Enter No of Production: "))
for i in range(n):
    txt=input()
    add(txt)

result = rem(gram)

for x,y in result.items():
    print(f'{x} ->  ',  end="")
    for index, i in enumerate(y):
        #print(i)
        for j in i:
            print(j, end="")
        if (index != len(y) - 1):
            print(" | ", end="")
    print()
```

Output:



```
PS C:\Users\Lenovo> & C:/Users/Lenovo/AppData/Local/Programs/Python/Python38/python.exe d:/sem7/cc/labcc/prac4/p4.py
Enter No of Production: 3
E->E+T|T
T->T*F|F
F->(E)|x
E -> TE'
T -> FT'
F -> (E) | x
E' -> +TE' | e
T' -> *FT' | e
PS C:\Users\Lenovo>
```



```
PS C:\Users\Lenovo> & C:/Users/Lenovo/AppData/Local/Programs/Python/Python38/python.exe d:/sem7/cc/labcc/prac4/p4.py
Enter No of Production: 2
S->a|^|(T)
T->T,S|S
S -> a | ^ | (T)
T -> ST'
T' -> ,ST' | e
PS C:\Users\Lenovo>
```