**Compiler Design Lab**

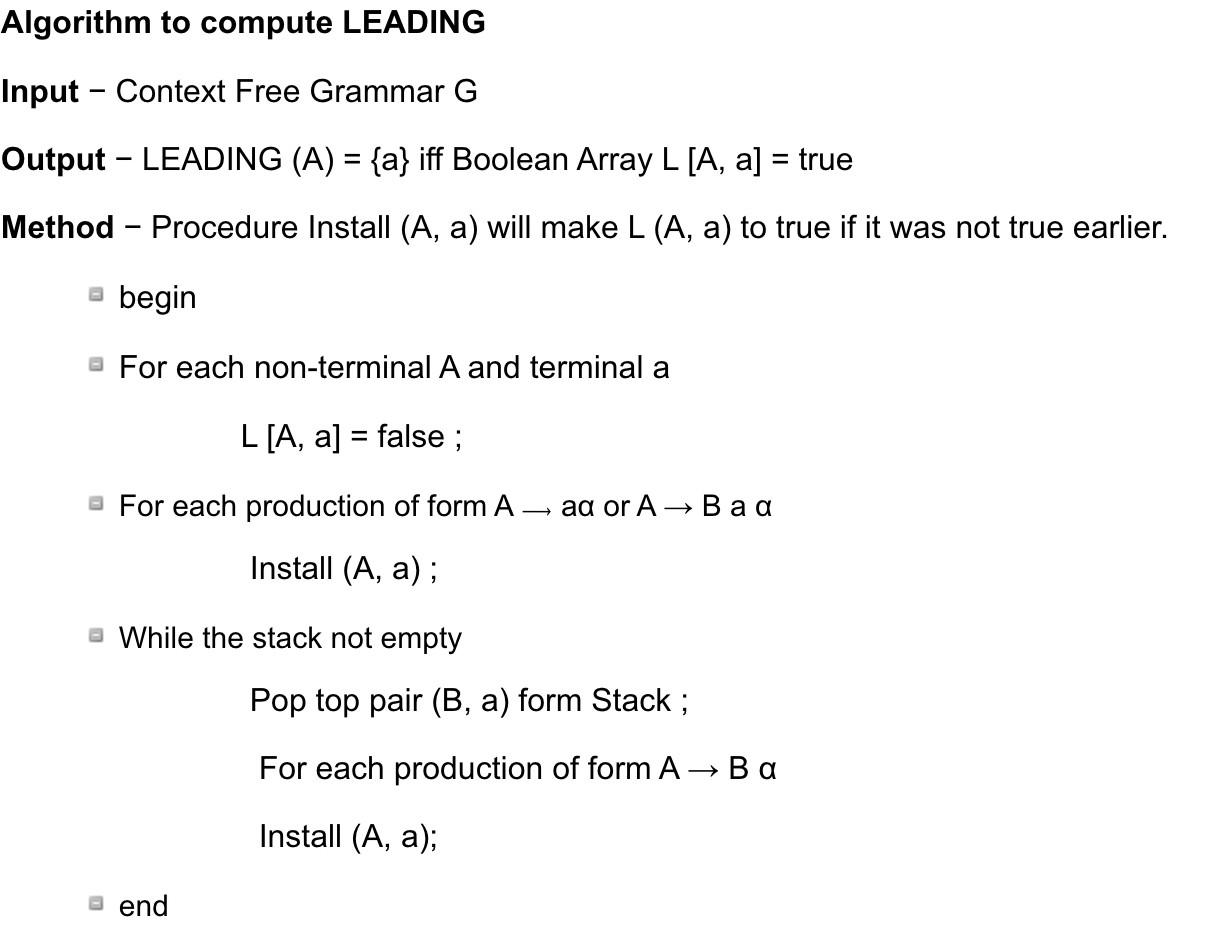
**Name –** N Aditya Sai

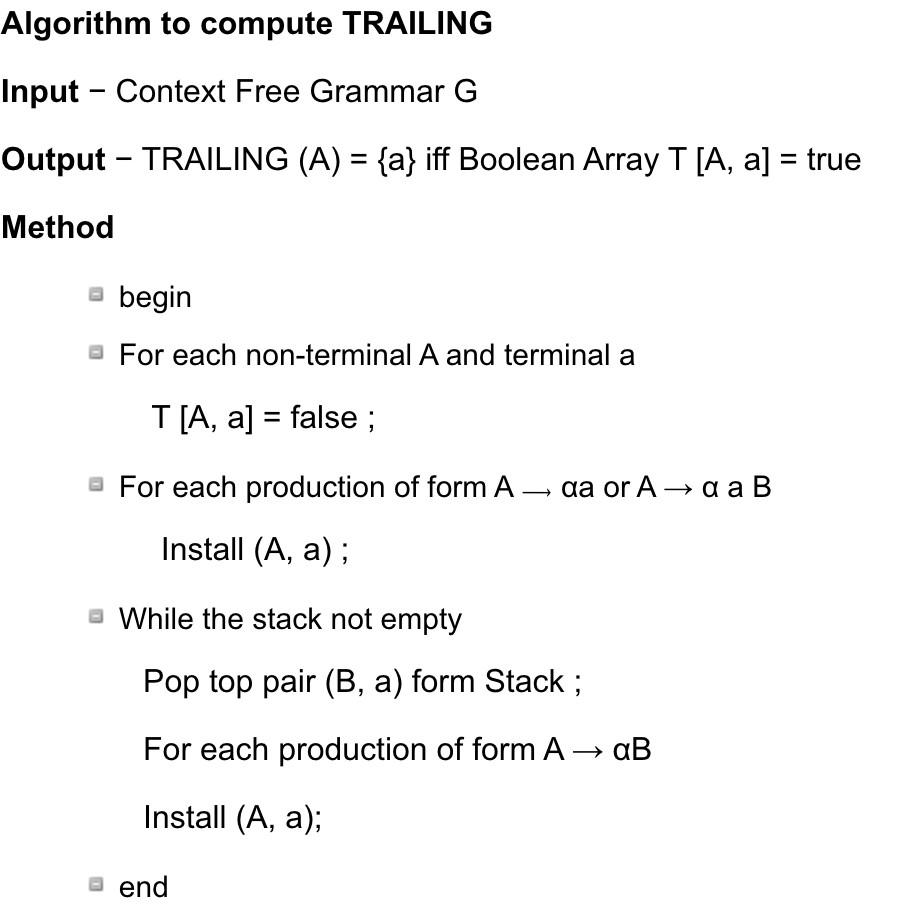
**Register No –** RA1911030010075

**Experiment –** 8

**Aim:** To compute leading and trailing for a given set of productions.

**Algorithm**:





**Code:**

#EXPT 8 Leading Trailing a = ["E=E+T",

"E=T",

"T=T\*F",

"T=F", "F=(E)", "F=i"]

rules = {} # dictionary for storing rules terms = [] #array for terms

for i in a:

temp = i.split("=")

terms.append(temp[0]) try:

rules[temp[0]] += [temp[1]] except:

rules[temp[0]] = [temp[1]] terms = list(set(terms)) print(rules,terms)

def leading(gram, rules, term, start): s = []

if gram[0] not in terms: return gram[0]

elif len(gram) == 1: return [0]

elif gram[1] not in terms and gram[-1] is not start: for i in rules[gram[-1]]:

s+= leading(i, rules, gram[-1], start) s+= [gram[1]]

return s

def trailing(gram, rules, term, start): s = []

if gram[-1] not in terms: return gram[-1]

elif len(gram) == 1: return [0]

elif gram[-2] not in terms and gram[-1] is not start: for i in rules[gram[-1]]:

s+= trailing(i, rules, gram[-1], start) s+= [gram[-2]]

return s leads = {} trails = {}

for i in terms: s = [0]

for j in rules[i]: s+=leading(j,rules,i,i)

s = set(s) s.remove(0) leads[i] = s s = [0]

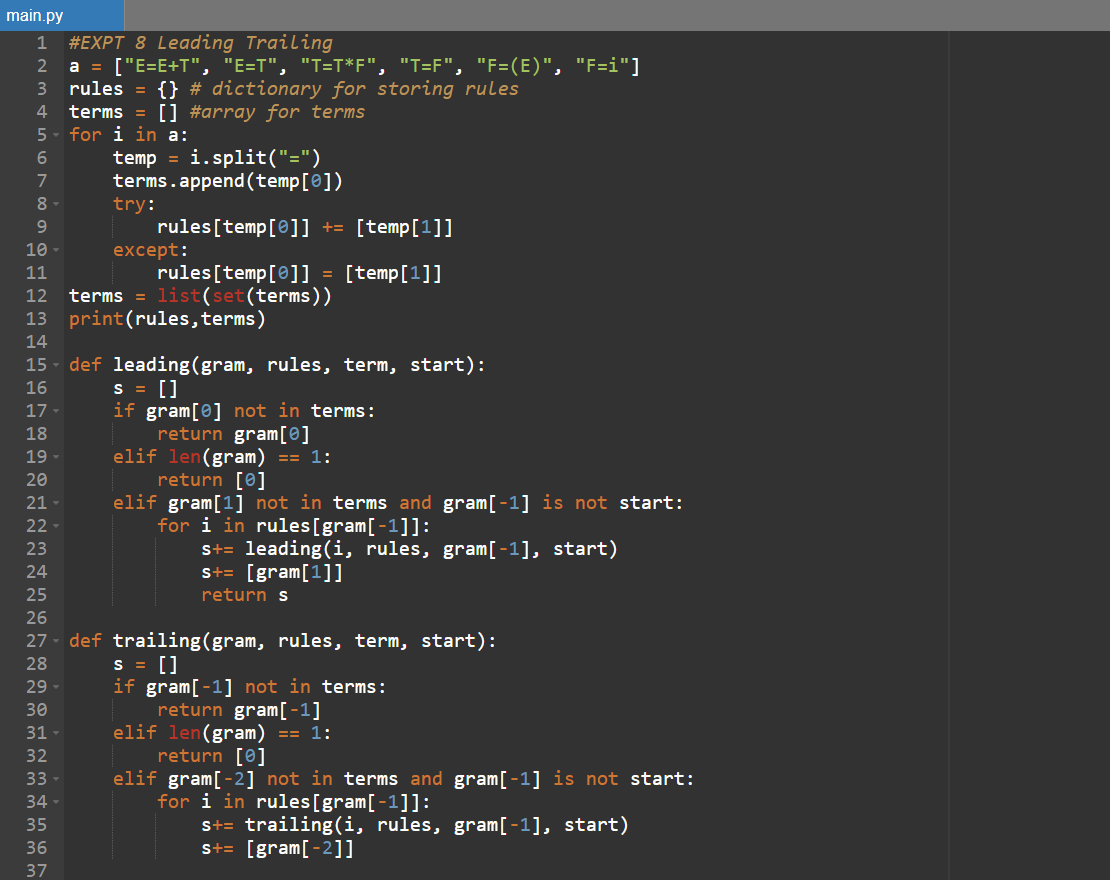
for j in rules[i]: s+=trailing(j,rules,i,i)

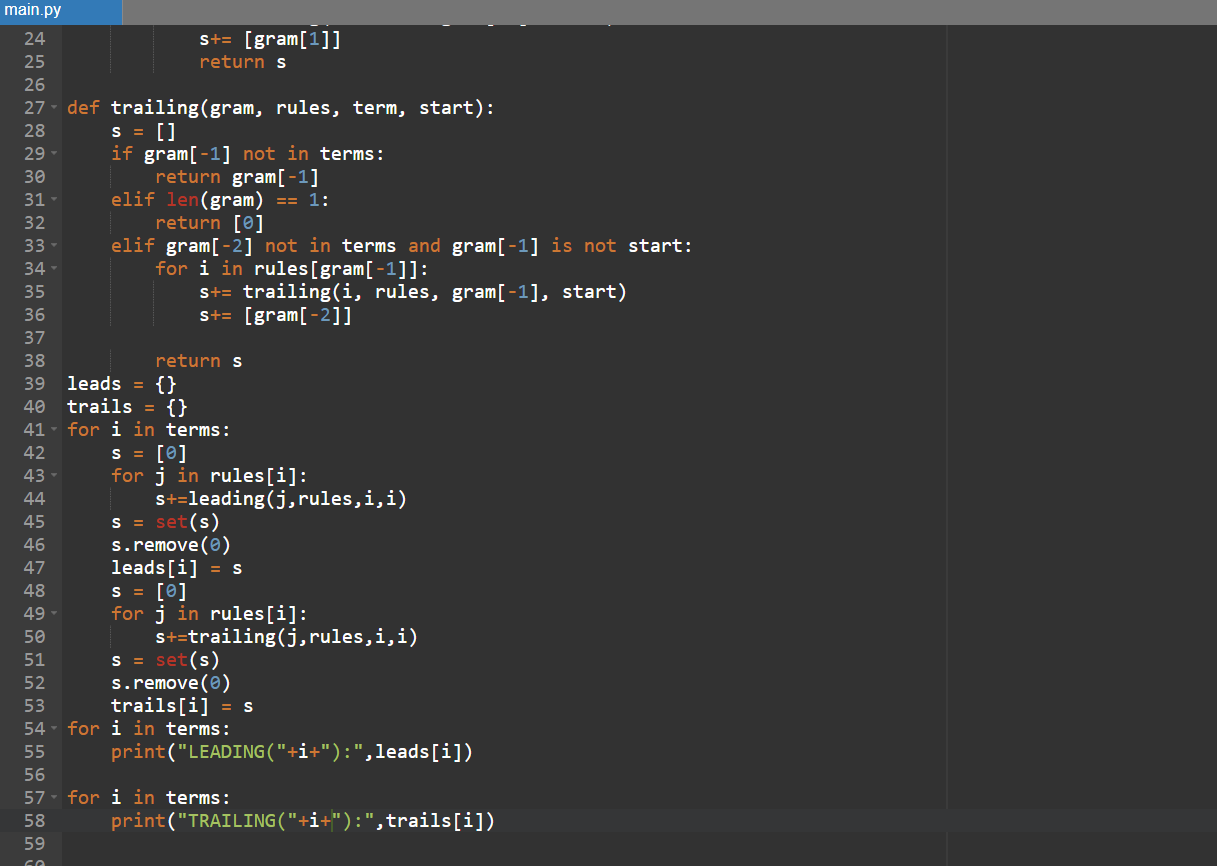
s = set(s) s.remove(0) trails[i] = s

for i in terms: print("LEADING("+i+"):",leads[i])

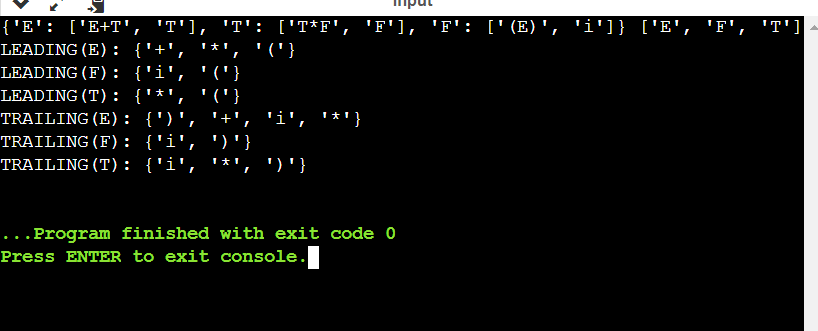
for i in terms: print(“TRAILING("+i+"):",trails[i])

**Screenshots:**

****

****

**Output:**



**Result:** Successfully computed leading trailing values for a given

set of grammar in python.