24-Feb-22

**Experiment 5 - FIRST and FOLLOW Computation**

Dhawal Patil

RA1911003010575

CSE A2

**Experiment 5**

Aim:

To write a program to perform first and follow using Python language.

Algorithm:

**For computing the first:**

1. If X is a terminal then FIRST(X) = {X}

Example: F -> I | id

We can write it as FIRST(F) -> { ( , id )

1. If X is a non-terminal like E -> T then to get FIRSTI substitute T with other productions until you get a terminal as the first symbol
2. If X -> ε then add ε to FIRST(X).

**For computing the follow:**

1. Always check the right side of the productions for a non-terminal, whose FOLLOW set is being found. (never see the left side).
2. (a) If that non-terminal (S,A,B…) is followed by any terminal (a,b…,\*,+,(,)…) , then add that terminal into the FOLLOW set.   
   (b) If that non-terminal is followed by any other non-terminal then add FIRST of other nonterminal into the FOLLOW set.

Code:

gram = {

"E":["E+T","T"],

"T":["T\*F","F"],

"F":["(E)","id"]

}

def removeDirectLR(gramA, A):

temp = gramA[A]

tempCr = []

tempInCr = []

for i in temp:

if i[0] == A:

tempInCr.append(i[1:]+[A+"'"])

else:

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)

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

result = rem(gram)

def first(gram, term):

a = []

if term not in gram:

return [term]

for i in gram[term]:

if i[0] not in gram:

a.append(i[0])

elif i[0] in gram:

a += first(gram, i[0])

return a

firsts = {}

for i in result:

firsts[i] = first(result,i)

print(f'First({i}):',firsts[i])

def follow(gram, term):

a = []

for rule in gram:

for i in gram[rule]:

if term in i:

temp = i

indx = i.index(term)

if indx+1!=len(i):

if i[-1] in firsts:

a+=firsts[i[-1]]

else:

a+=[i[-1]]

else:

a+=["e"]

if rule != term and "e" in a:

a+= follow(gram,rule)

return a

follows = {}

for i in result:

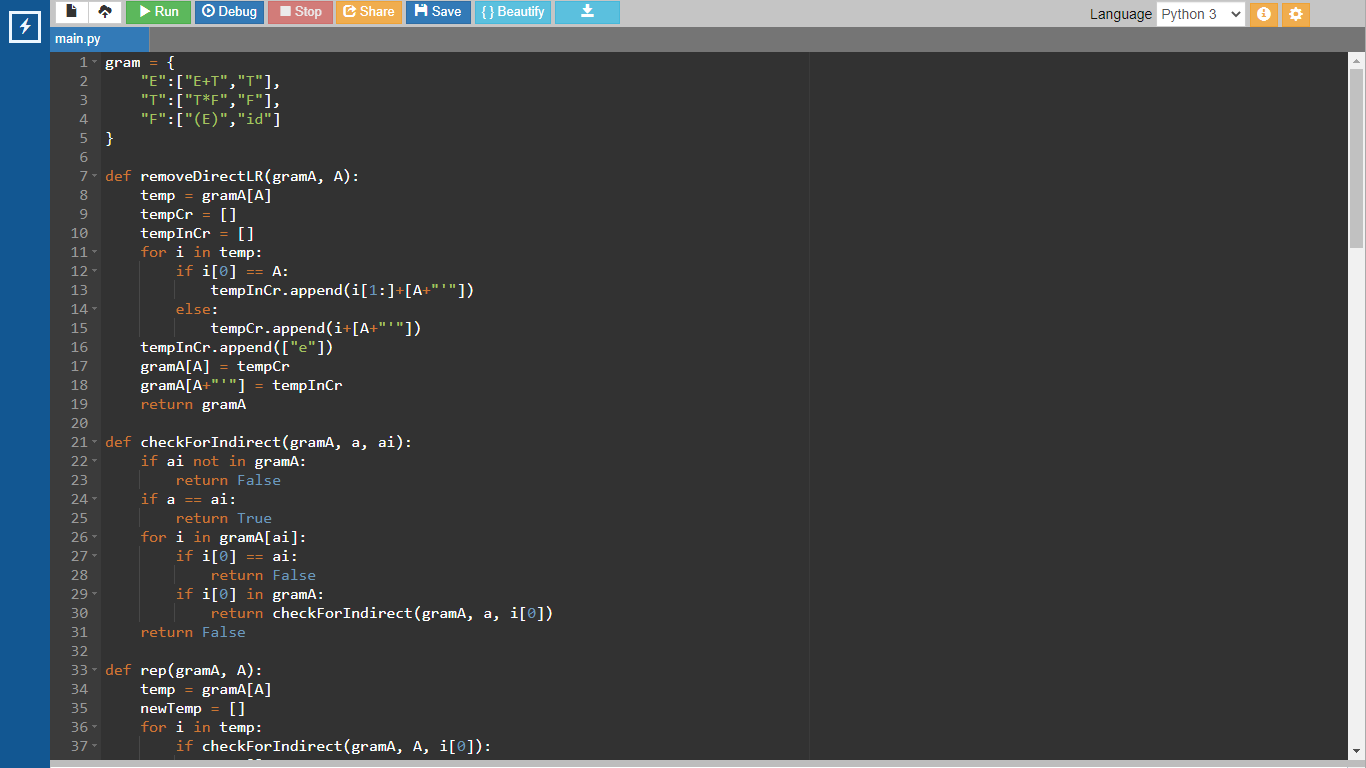
follows[i] = list(set(follow(result,i)))

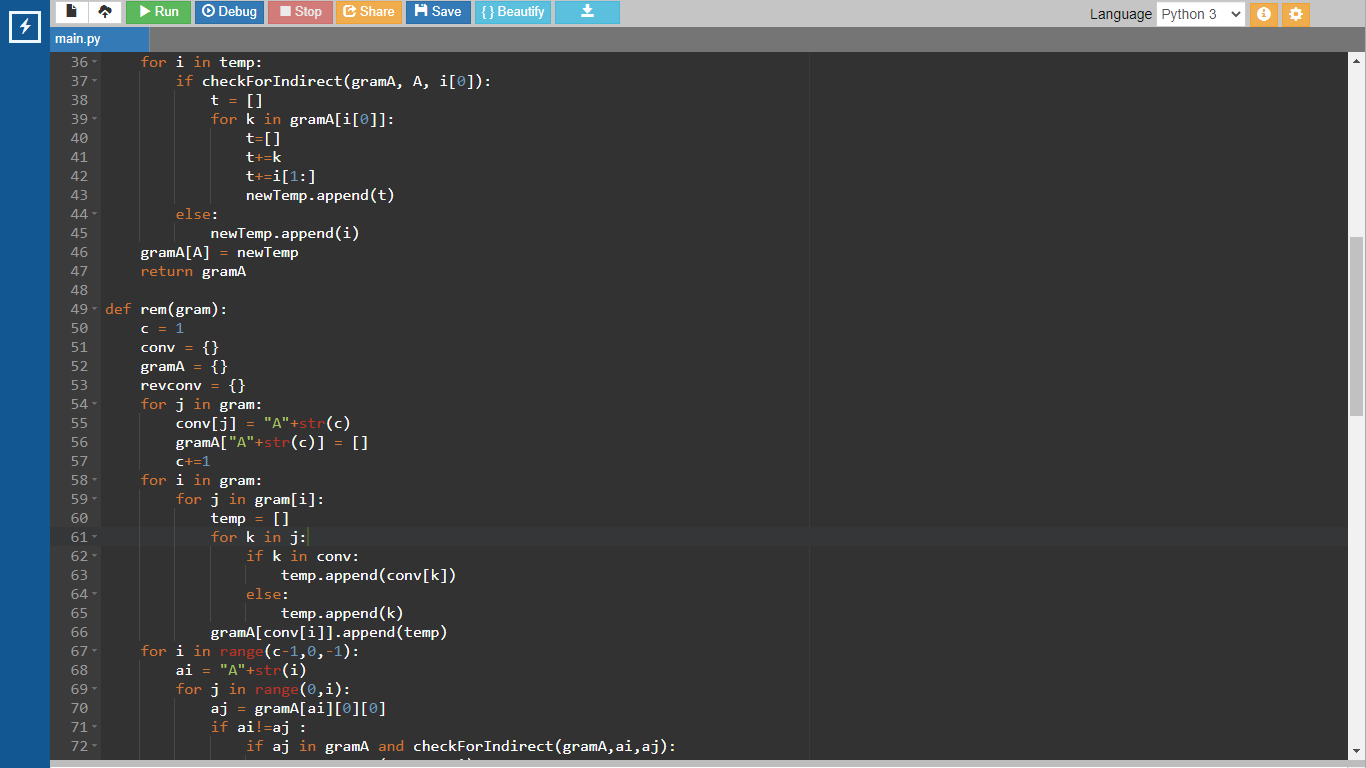
if "e" in follows[i]:

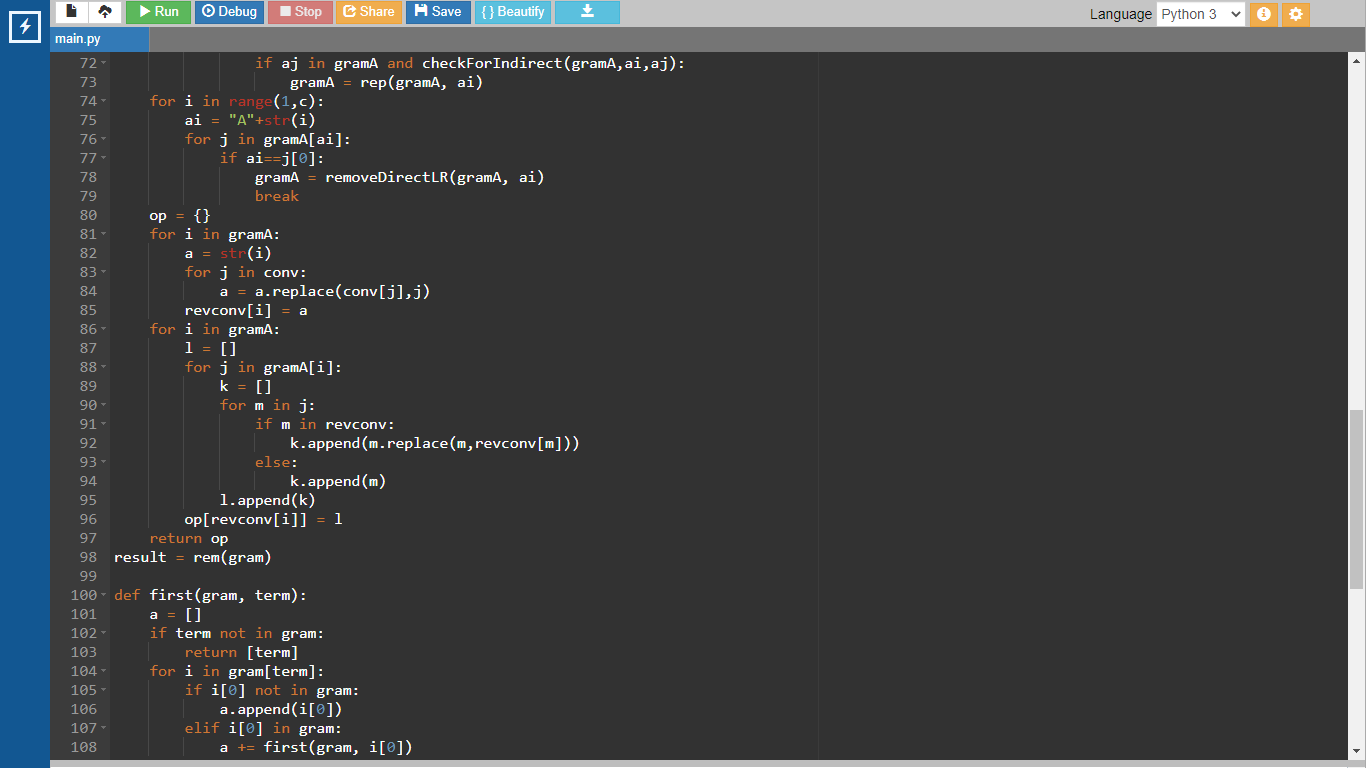
follows[i].pop(follows[i].index("e"))

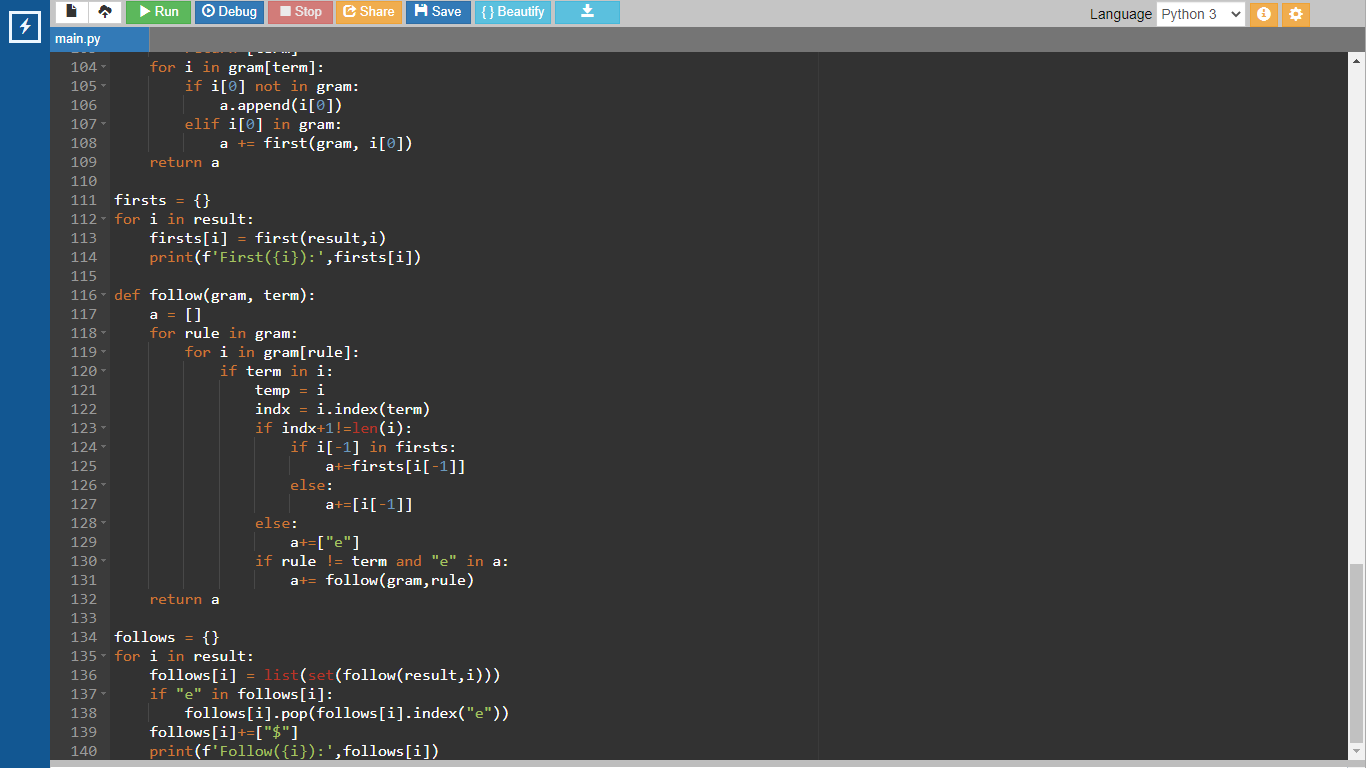
follows[i]+=["$"]

print(f'Follow({i}):',follows[i])

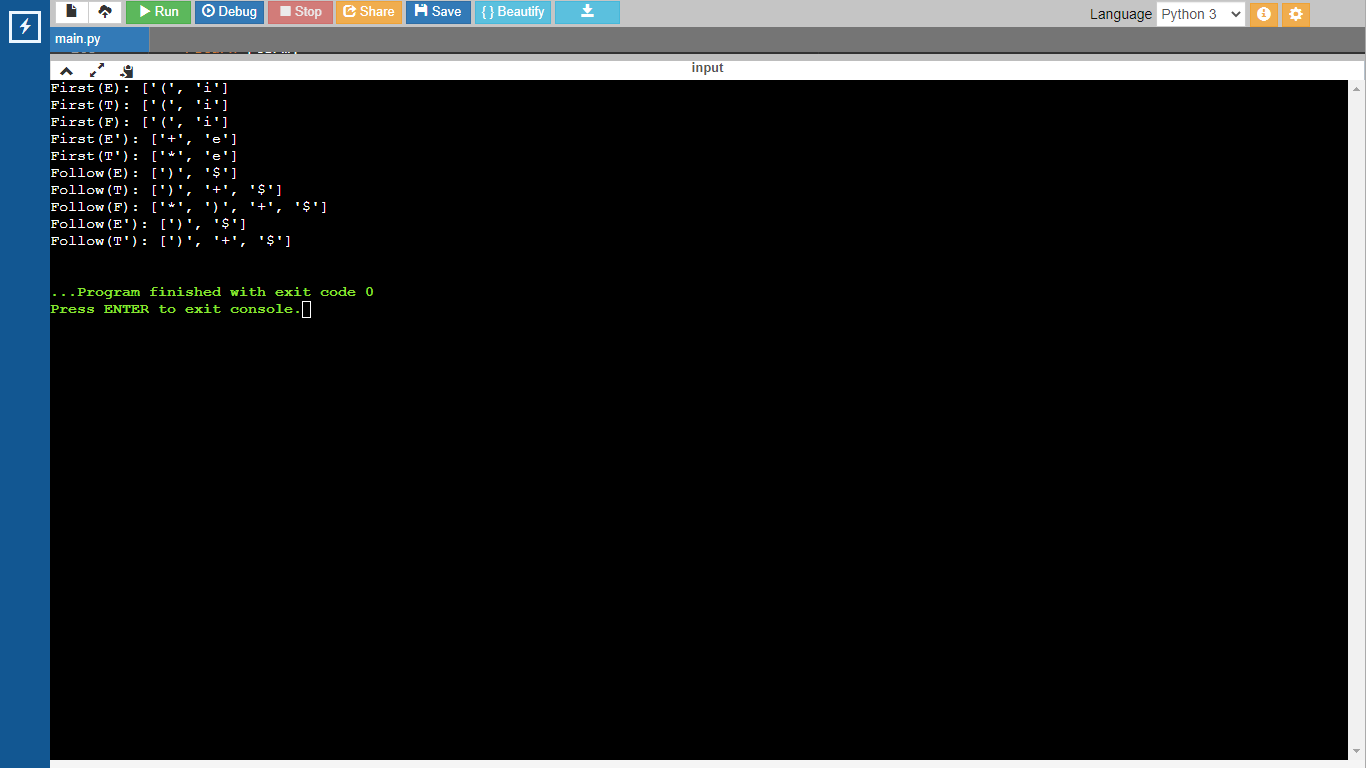








Output:



Result:

A program for FIRST and FOLLOW computation was run successfully.