

Experiment - 7 Finding follow of a grammar

Name: _____ **Rollno** : _____
Class : TE.CO **Batch** : _____

#Source Code

```
__author__ = 'Shadab Shaikh'
__title__ = 'Finding follow of a set from a grammar'
__date__ = '01-03-2019'
__version__ = '1.0'

print('Author      : ' + __author__)
print('Title       : ' + __title__)
print('Date        : ' + __date__)
print('Version     : ' + __version__)

grammararr=[]          #stores the grammar maintaining the index
flw=[]                 #stores the final result of follow set
inputs=""              #taking input from user
inputm=""              #continuity of production
s1=""                  #acting as a pointer to compare and find the left most variable

flw2=[]                #Stores the variable production which has to be
replaced with following epsilon
flw3=[]                #Stores the variable production which has to be
replaced with following epsilon after end iteration
index1=[]              #used to remove unwanted production for flw2
index2=[]              #used to remove unwanted production for flw3
index3=[]              #used to remove unwanted production for final iteration
flwcopy=[]             #storing the left out variable
fincap=[]              #getting the left out follow
fincap1=[]             #storing the left out follow
j1=""

while(inputs!='no'):
    grammar = input("\nEnter the grammar left should be variable following with ->
format eg: S->a\n")
    grammar=grammar.replace(" ", "")    #replacing whitespaces with none
    if(grammar[0].islower()):
        grammar[0].upper()             #making left most as variable
    grammararr.append(grammar)          #storing into list
    inputs = input("\nPress no to stop writing productions or write anything to continue")
    #asking for the continuity of grammar
```

```

def searchepsi(c1,grammararr,v,w):
    """function for epsilon production condition."""
    if(grammararr[v][w]==c1):
        try:
            flw3.append("follow{" +grammararr[v][w-1]+"}="+grammararr[v]
            [w+1]) #if epsilon is found, moving the next element and storing in flw
            #list
        except:
            flw3.append("follow{" +grammararr[v][w-1]+"}="+grammararr[v][0])
            #if element is not present leftmost become the follower and storing in
            #flw list
    else:
        w+=1
        if(w<len(grammararr[v])):
            searchepsi(c1,grammararr,v,w)
            #recursively calling determining each element
        else:
            w=0
            v+=1
            if(v<len(grammararr)):
                searchepsi(c1,grammararr,v,w)

```

```

def searchprodcap(grammararr,n1,s1,k,n):
    """function to check if the element is variable the finding first of it."""
    if(grammararr[k][0]==n1):
        if(grammararr[k][3].isupper()):
            #checking if the 3rd index
            #element is uppercase
            n1=grammararr[k][3]
            #if yes then reassigning n1
            searchprodcap(grammararr,n1,s1,k,n)#recursively calling with updated
            #n1
        if(grammararr[k][3]=='#'):
            #if epsilon is found calling searchepsi function
            c1=grammararr[k][0] #updating c1 for comparison to searchepsi
            searchepsi(c1,grammararr,0,3)
        else:
            flw.append("follow{" +s1+"}="+grammararr[k][3])
            #else appending it to final follow list
    k+=1
    #incrementing k by 1
    if(k<len(grammararr)):
        #until k is less than grammar list
        searchprodcap(grammararr,n1,s1,k,n)

```

```

def searchprodrigh(flw,x,y):
    """function to replace variable with corresponding follow element."""
    if(flw[y][10].isupper()):
        #Checking if flw list has variables
        if(flw[x][7]==flw[y][10]):
            #finding variables follow
            flw2.append("follow{" +flw[y][7]+"}="+flw[x][10])#updating flw2 list
            x+=1

```

```

        if(x<len(flw)):
            searchprodrightright(flw,x,y)
    else:
        x+=1
        if(x<len(flw)):
            searchprodrightright(flw,x,y)
    x=0
    y+=1
    if(y<len(flw)):
        searchprodrightright(flw,x,y)                #recursively checking for each index

def searchprodrightright2(flw,x,y):
    """function to replace variable with corresponding follow element after each iteration
    is finished."""
    if(flw[y][10].isupper()):                #Checking if flw list has variables
        if(flw[x][7]==flw[y][10]):          #finding variables follow
            flw3.append("follow{"+flw[y][7]+"}="+flw[x][10])#updating flw3 list
            x+=1
            if(x<len(flw)):
                searchprodrightright2(flw,x,y)
    else:
        x+=1
        if(x<len(flw)):
            searchprodrightright2(flw,x,y)
    x=0
    y+=1
    if(y<len(flw)):
        searchprodrightright2(flw,x,y)                #recursively checking for each index

def searchprod(grammararr,s1,i,k,n):
    """function to find the follow element to be taken into consideration."""
    if(grammararr[k][i]==s1):                #checking if element is present in production
        try:
            if(grammararr[k][i+1].isupper()):    #if found then assigning the right
                                                    #adjacent element, if its
                                                    #uppercase
                n1=grammararr[k][i+1]            #assigning element value to n1
                searchprodcap(grammararr,n1,s1,k,n)#calling
                                                    #searchprodcap function
        else:
            flw.append("follow{"+s1+"}="+grammararr[k][i+1])
                                                    #if element is terminal then updating flw list
    except:
        #searchright(grammararr,previdx)

```

```

        flw.append("follow{" + s1 + "}=" + grammararr[k][0])
        #if element is not present in adjacent right the storing leftmost
        #into flw list
        flwcopy.append("follow{" + s1 + "}=" + grammararr[k][0])
        #will be used for final iteration
        searchprodrightright(flw, 0, 0)
        #calling searchprodrightright function
    else:
        i += 1
        if (i < len(grammararr[k])):
            searchprod(grammararr, s1, i, k, n)
        else:
            k += 1
            i = 3
            if (k < len(grammararr)):
                searchprod(grammararr, s1, i, k, n)
                #recursively calling this func until each element is parsed

def findfollow(grammararr, k):
    """function to find follow of a production variable."""
    s1 = grammararr[k][0] #assigning start variable to s1 initially
    searchprod(grammararr, s1, 3, 0, 1) #calling the searchprod function
    k += 1 #incrementing k by 1
    if (k < len(grammararr)): #until k is less than grammar list
        findfollow(grammararr, k) #recursively calling findfollow function

flw.append("follow{" + grammararr[0][0] + "}=" + "$") #updating flw list by $ for
starting production
findfollow(grammararr, 0) #calling findfollow function

#sorting the list
flw = list(set(flw)) #getting all the unique results
flw.sort()
flw2 = list(set(flw2)) #getting all the unique results
flw2.sort()
flw3 = list(set(flw3)) #getting all the unique results
flw3.sort()

for i in range(len(flw2)):
    flw.append(flw2[i]) #appending the result of flw with flw2list

for r in range(len(flw)):
    if (flw[r][10].isupper()):
        index1.append(flw[r]) #checking if there is any variable updating
index1 list

```

```

for d in range(len(index1)):
    flw.remove(index1[d])                #removing corresponding variable list

for i in range(len(flw3)):
    flw.append(flw3[i])                  #appending the result of flw3 with flwlist

searchprodrigh2(flw,0,0)                 #calling searchprodrigh2 function

flw3=list(set(flw3))
flw3.sort()                             #getting all the unique results

for i in range(len(flw3)):
    flw.append(flw3[i])                  #appending the result of flw3 with flwlist

searchprodrigh2(flw,0,0)                 #again calling searchprodrigh2 function

for i in range(len(flw3)):
    flw.append(flw3[i])                  #again appending the result of flw3 with flwlist

for r in range(len(flw)):
    if(flw[r][10].isupper()):
        index2.append(flw[r])            #checking if there is any variable updating
index1 list

for d in range(len(index2)):
    flw.remove(index2[d])                #removing corresponding variable list

for i in range(len(flwcopy)):
    flw.append(flwcopy[i])               #appending the final stored iteration element

for c in range(len(flw)):
    if(flw[c][10].isupper()):
        fincap.append(flw[c][7]+flw[c][10]) #checking if any variable in production
                                                #of flwlist, appending fincap list

if(fincap!=None):
    for i in range(len(flw)):
        for j in range(len(fincap)):
            if(flw[i][7]==fincap[j][1]):
                fincap1.append("follow{"+fincap[j][0]+"}="+flw[i][10])
            #finding the corresponding variable follow

```

```

if(fincap1!=None):
    fincap1=list(set(fincap1))                #getting all the unique results
    fincap1.sort()

    for i in range(len(fincap1)):
        flw.append(fincap1[i])                #appending result of fincap1 to flw list

    for r in range(len(flw)):
        if(flw[r][10].isupper()):
            index3.append(flw[r])              #finding variable from flwlist

    for d in range(len(index3)):
        flw.remove(index3[d])                  #removing unwanted element from flwlist

flw=list(set(flw))                            #getting all the unique results
flw.sort()
print(*flw, sep = "\n")                       #printing the final result

```

#Sample Input

1. E->TA
 A->+TA/€
 T->FB
 B->*FB/€
 F->id/(E) #(Case of non-splitting of terminal)
2. S->(S)/ €
3. S->aABb
 A->C/€
 B->d/€

#Output

1. Sample 1st input #(Case of non-splitting of terminal)

```
aiktc@aiktc1: /media/aiktc/D2C5-103D/Academics/[Study Material]/Degree/CBGS REV 2016 2017-2020 MUMBAI UNIVERSITY/TE/Sem 6/Labs/CSL602 System Software Lab 01/Experiment 7 - Follow - + x
File Edit View Search Terminal Help
Author : Shadab Shaikh
Title : Finding follow of a set from a grammar
Date : 01-03-2019
Version : 1.0
Enter the grammar left should be variable following with -> format eg: S->a
E->TA
Press no to stop writing productions or write anything to continueyes
Enter the grammar left should be variable following with -> format eg: S->a
A->TA
Press no to stop writing productions or write anything to continueyes
Enter the grammar left should be variable following with -> format eg: S->a
A->#
Press no to stop writing productions or write anything to continueyes
Enter the grammar left should be variable following with -> format eg: S->a
T->FB
Press no to stop writing productions or write anything to continueyes
Enter the grammar left should be variable following with -> format eg: S->a
B->*FB
Press no to stop writing productions or write anything to continueyes
Enter the grammar left should be variable following with -> format eg: S->a
B->#
Press no to stop writing productions or write anything to continueyes
Enter the grammar left should be variable following with -> format eg: S->a
F->id
Press no to stop writing productions or write anything to continueyes
Enter the grammar left should be variable following with -> format eg: S->a
F->(E)
Press no to stop writing productions or write anything to continueyes
follow(A)=S
follow(A)=S
follow(B)=S
follow(B)=S
follow(B)=+
follow(E)=S
follow(E)=S
follow(F)=S
follow(F)=S
follow(F)=+
follow(F)=+
follow(T)=S
follow(T)=S
2 items, Free space: 2.5 GB
```

2. Sample 2nd input

```
aiktc@aiktc1: /media/aiktc/D2C5-103D/Academics/[Study Material]/Degree/CBGS REV 2016 2017-2020 MUMBAI UNIVERSITY/TE/Sem 6/Labs/CSL602 System Software Lab 01/Experiment 7 - Follow - + x
File Edit View Search Terminal Help
aiktc@aiktc1:/media/aiktc/D2C5-103D/Academics/[Study Material]/Degree/CBGS REV 2016 2017-2020 MUMBAI UNIVERSITY/TE/Sem 6/Labs/CSL602 System Software Lab 01/Experiment 7
- Follow$ python3 Follow.py
Author : Shadab Shaikh
Title : Finding follow of a set from a grammar
Date : 01-03-2019
Version : 1.0
Enter the grammar left should be variable following with -> format eg: S->a
S->(S)
Press no to stop writing productions or write anything to continueyes
Enter the grammar left should be variable following with -> format eg: S->a
S->#
Press no to stop writing productions or write anything to continueyes
follow(S)=S
follow(S)=S
aiktc@aiktc1:/media/aiktc/D2C5-103D/Academics/[Study Material]/Degree/CBGS REV 2016 2017-2020 MUMBAI UNIVERSITY/TE/Sem 6/Labs/CSL602 System Software Lab 01/Experiment 7
- Follow$ █
2 items, Free space: 2.5 GB
Network
Gigaset Network
"output1.png" selected (139.4 KB), Free space: 2.5 GB
```

3. Sample 3rd input

```
aiktc@aiktc1: /media/aiktc/D2C5-103D/Academics/[Study Material]/Degree/CBGS REV 2016 2017-2020 MUMBAI UNIVERSITY/TE/Sem 6/Labs/CSL602 System Software Lab 01/Experiment 7 - Follow - + X
File Edit View Search Terminal Help
aiktc@aiktc1: /media/aiktc/D2C5-103D/Academics/[Study Material]/Degree/CBGS REV 2016 2017-2020 MUMBAI UNIVERSITY/TE/Sem 6/Labs/CSL602 System Software Lab 01/Experiment 7
- Follow$
aiktc@aiktc1: /media/aiktc/D2C5-103D/Academics/[Study Material]/Degree/CBGS REV 2016 2017-2020 MUMBAI UNIVERSITY/TE/Sem 6/Labs/CSL602 System Software Lab 01/Experiment 7
- Follow$ python3 Follow.py
Author      : Shadab Shaikh
Title       : Finding follow of a set from a grammar
Date        : 01-03-2019
Version     : 1.0

Enter the grammar left should be variable following with -> format eg: S->a
S->aABb

Press no to stop writing productions or write anything to continueyes

Enter the grammar left should be variable following with -> format eg: S->a
A->C

Press no to stop writing productions or write anything to continueyes

Enter the grammar left should be variable following with -> format eg: S->a
A->#

Press no to stop writing productions or write anything to continueyes

Enter the grammar left should be variable following with -> format eg: S->a
B->d

Press no to stop writing productions or write anything to continueyes

Enter the grammar left should be variable following with -> format eg: S->a
B->#

Press no to stop writing productions or write anything to continueno
follow{A}=b
follow{A}=d
follow{B}=b
follow{S}=S
aiktc@aiktc1: /media/aiktc/D2C5-103D/Academics/[Study Material]/Degree/CBGS REV 2016 2017-2020 MUMBAI UNIVERSITY/TE/Sem 6/Labs/CSL602 System Software Lab 01/Experiment 7
- Follow$
```