Name: Atharva Paliwal
Roll No: 40

Practical No. 3

Topic: Parser Construction

Platform: Windows or Linux

Language to be used: Python or Java (based on the companies targeted for placement)

CO Mapped: CO3- Implement different types of Parsing techniques

Aim:

A. Write a program to find FIRST for any grammar. All the following rules of FIRST must be implemented.

For a generalized grammar: A → αXY
FIRST (A) = FIRST (αXY)

= α if α is the terminal symbol (Rule-1)

= FIRST (α) if α is a non-terminal and FIRST (α) does not contain ε (Rule-2)

= FIRST (α) - ε ∪ FIRST (XY) if a is a non-terminal and FIRST (α) contains ε (Rule-3)

(B) Compute Follow information and computer the LL(1) parsing table using the FIRST values computed above.

CODE-

```
def find_first(Non_Terminal,Terminal,Rules,First,NT):
    li=[]
    for i in Rules[NT]:
        li1=[]
        if i[0] in Terminal:
            li=li+[i[0]]
        elif i[0] in Non_Terminal:
            li1=li1+(find_first(Non_Terminal,Terminal,Rules,First,i[0]))
            k=1
            while ('0' in li1) and k<len(i):
                li1.remove('0')
                if i[k] in Terminal:
```

```
                    li1=li1+[i[k]]
                    break
                elif i[k] in Non_Terminal:
                    li1=li1+find_first(Non_Terminal,Terminal,Rules,First,i[k])
                k=k+1
        else :
            li1=li1+['0']
        li=li+li1
    return list(set(li))


def find_follow(Follow,First,Terminal,Non_Terminal,Rules):
    for i in Non_Terminal:
        for j in Rules[i]:
            l=len(j)
            for k in range(l):
                if j[k] in Non_Terminal:
                    if l==1:
                        Follow[j[k]]=Follow[j[k]]+Follow[i]
                    elif k+1==l:
                        Follow[j[k]]=Follow[j[k]]+Follow[i]
                    elif j[k+1] in Terminal:
                        Follow[j[k]]=list(set(Follow[j[k]]+[j[k+1]]))
                    elif j[k+1] in Non_Terminal:
                        m=k+2
                        li=First[j[k+1]]
                        while '0' in li and m<l:
                            if j[m] in Terminal :
                                li.remove('0')
                                li=li+[j[m]]
                                break
                            elif j[m] in Non_Terminal:
                                li=li+Follow[l[m]]
                                m=m+1
                        Follow[j[k]]=Follow[j[k]]+list(set(li))
    return Follow

#Driver Code
Non_Terminal,Terminal,Rules,First,Follow,parse_table=list(),list(),dict(),dict(),
dict(),dict()

# Input Non Terminal
print('Enter Non Terminals : ')
Non_Terminal=list(str(i) for i in input().split())
for x in Non_Terminal:
```

```
    Rules[x]=[]
    First[x]=[]
    Follow[x]=[]

# Input Terminal
print('Enter Terminals:')
Terminal=list(str(i) for i in input().split())

# Input Production Rules
print('Enter rules in format A : BC|D and 0 for null : ')
for _ in range(len(Non_Terminal)):
    rule=input().split(':')
    for i in rule[1].split('|'):
        Rules[rule[0].strip()].append(i.strip())

# Finding First
for NT,rule in Rules.items():
    First[NT]=find_first(Non_Terminal,Terminal,Rules,First,NT)

print('\n--------First Sets--------\n')
for k,v in First.items():
    print(k,' : ',set(v),'\n')

# Follow
Follow[Non_Terminal[0]]=['$']
Follow=find_follow(Follow,First,Terminal,Non_Terminal,Rules)
print('\n--------Follow Sets--------\n')
for k,v in Follow.items():
    print(k,' : ',set(v),'\n')

# Parse Table
for i in Non_Terminal:
    for j in Rules[i]:
        e=j[0]
        entry=[]
        if e in Terminal:
            entry=entry+[e]
        elif e in Non_Terminal:
            entry=First[e]
            k=1
            while '0' in entry and k <len(j):
                e=j[k]
                if e in Terminal:
                    entry.remove('0')
                    entry=entry+[e]
```

```
                    break
                elif e in Non_Terminal:
                    entry.remove('0')
                    entry=entry+First[e]
                k=k+1
            if '0' in entry:
                entry=Follow[i]
        else:
            entry=Follow[i]
        for l in entry:
            parse_table[(i,l)]=i+' - > '+j

print('\n--------Parsing Table--------\n')
for k,v in parse_table.items():
    print(k,' : ',v,'\n')
```

OUTPUT:

```
Enter Non Terminals :

S A B C

Enter Terminals:

a b c p

Enter rules in format A : BC|D and 0 for null :

S:A | BC

A: a | b

B: p | 0

C: c


--------First Sets--------


S  :  {'a', 'c', 'b', 'p'}
```

A : {'a', 'b'}

B : {'0', 'p'}

C : {'c'}

--------Follow Sets--------

S : {'$'}

A : {'$'}

B : {'c'}

C : {'$'}

--------Parsing Table--------

('S', 'a') : S - > A

('S', 'b') : S - > A

('S', 'p') : S - > BC

```
('S', 'c')  :  S - > BC

('A', 'a')  :  A - > a

('A', 'b')  :  A - > b

('B', 'p')  :  B - > p

('B', 'c')  :  B - > 0

('C', 'c')  :  C - > c
```