Shri Ramdeobaba College of Engineering and Management, Nagpur
Department of Computer Science and Engineering
Session: 2021-2022 [EVEN SEM]

Compiler Design Lab

PRACTICAL No. 3

Name : Shantanu Mane

Roll No. : E63

Topic: Parser Construction

Platform: Windows or Linux

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

Aim:

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

For a generalized grammar: $A \rightarrow \alpha XY$

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)

Input: Grammar rules from a file or from console entered by user.
Following inputs can be used:

Batch A1:
$A \rightarrow SB \mid B$
$S \rightarrow a \mid Bc \mid \varepsilon$
$B \rightarrow b \mid d$

Batch A2:
$S \rightarrow A \mid BC$
$A \rightarrow a \mid b$
$B \rightarrow p \mid \varepsilon$
$C \rightarrow c$

Batch A3:
$S \rightarrow AB \mid C$
$A \rightarrow a \mid b \mid \varepsilon$
$B \rightarrow p \mid \varepsilon$
$C \rightarrow c$

Batch A4:
$S \rightarrow ABC \mid C$
$A \rightarrow a \mid bB \mid \varepsilon$
$B \rightarrow p \mid \varepsilon$
$C \rightarrow c$

Implementation: FIRST rules
Output: FIRST information for each non-terminal

(B) Calculate Follow for the given grammar manually, input the follow information and Construct the LL (1) parsing table using the FIRST and FOLLOW values computed above.

Submission Format: Pdf should contain- Aim, scanned copy of hand solved numerical (batch specific), code, and execution screen shot.

Code :

```
def first(string):
    first_ = set()
    if string in non_terminals:
        alt = productions_dict[string]

        for altel in alt:
            first_2 = first(altel)
            first_ = first_ | first_2

    elif string in terminals:
        first_ = {string}

    elif string == '' or string == '@':
        first_ = {'@'}

    else:
        first_2 = first(string[0])
        if '@' in first_2:
            i = 1
            while '@' in first_2:
                # print("inside while")

                first_ = first_ | (first_2 - {'@'})
                # print('string[i:]=', string[i:])
                if string[i:] in terminals:
                    first_ = first_ | {string[i:]}
                    break
                elif string[i:] == '':
                    first_ = first_ | {'@'}
                    break
                first_2 = first(string[i:])
                first_ = first_ | first_2 - {'@'}
                i += 1
        else:
            first_ = first_ | first_2

    return first_


no_of_terminals = int(input("Enter no. of terminals: "))

terminals = []

print("Enter the terminals :")
for _ in range(no_of_terminals):
    terminals.append(input())

no_of_non_terminals = int(input("Enter no. of non terminals: "))

non_terminals = []

print("Enter the non terminals :")
for _ in range(no_of_non_terminals):
    non_terminals.append(input())
```

```python
starting_symbol = input("Enter the starting symbol: ")

no_of_productions = int(input("Enter no of productions: "))

productions = []

print("Enter the productions:")
for _ in range(no_of_productions):
    productions.append(input())

productions_dict = {}

for nT in non_terminals:
    productions_dict[nT] = []

for production in productions:
    nonterm_to_prod = production.split("->")
    alternatives = nonterm_to_prod[1].split("/")
    for alternative in alternatives:
        productions_dict[nonterm_to_prod[0]].append(alternative)


FIRST = {}

for non_terminal in non_terminals:
    FIRST[non_terminal] = set()

for non_terminal in non_terminals:
    FIRST[non_terminal] = FIRST[non_terminal] | first(non_terminal)

print("{: ^20}{: ^20}".format('Non Terminals', 'First'))
for non_terminal in non_terminals:
    print("{: ^20}{: ^20}".format(non_terminal, str(FIRST[non_terminal])))
```

Output :

```
SHANTANU ▶ py .\CalculateFirst.py

Enter no. of terminals: 4

Enter the terminals :

a

b

p

c

Enter no. of non terminals: 4

Enter the non terminals :

S

A
```

```
B

C

Enter the starting symbol: S

Enter no of productions: 4

Enter the productions:

S->ABC/C

A->a/bB/@

B->p/@

C->c

   Non Terminals           First

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

         A            {'a', '@', 'b'}

         B               {'p', '@'}

         C                 {'c'}
```

Shantanu Mane

E-63

Grammar :

$S \longrightarrow ABC \mid c$

$A \longrightarrow a \mid bB \mid \varepsilon$

$B \longrightarrow p \mid \varepsilon$

$C \longrightarrow c$

First (C) = $\{c\}$

Fi (B) = $\{p, \varepsilon\}$

Fi (A) = $\{a, b, \varepsilon\}$

Fi (S) = $\{a, b, p, c\}$

Fo (S) = $\{\$\}$

Fo (A) = $\{p, c\}$

Fo (B) = $\{p, c, \$\}$

Fo (C) = $\{\$\}$

	a	b	c	p	$
S	$S \to ABC$	$S \to ABC$	$S \to ABC$	$S \to ABC$	-
A	$A \to a$	$A \to bB$	$A \to \varepsilon$	$A \to \varepsilon$	-
B	-	-	$B \to \varepsilon$	$B \to \Delta$ $B \to \varepsilon$	-
C	-	-	$C \to c$	-	-