# Experiment No.4

## Title: - First Sets

**Aim:-** Write a program to find first() sets of given Grammar

**Finding First:-**

**First(x)** for all grammar symbols X

Apply following rules:

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

2. If X → ε is a production, then add ε to FIRST(X).

3. If X is a non-terminal, and X→ Y1Y2… Yk is a production, and ε is in all of FIRST(Y1), …, FIRST(Yk), then add ε to FIRST(X).

4. If X is a non-terminal, and X→ Y1Y2… Yk is a production, then add a to FIRST(X) if for some i, a is in FIRST(Yi), and ε is in all of FIRST(Y1), …, FIRST(Yi-1).

Applying rules 1 and 2 is obvious. Applying rules 3 and 4 for FIRST(Y1Y2… Yk) can be done as follows:

Add all the non-ε symbols of FIRST(Y1) to FIRST(Y1 Y2 … Yk). If ε ∈ FIRST(Y1), add all the non-ε symbols of FIRST(Y2). If ε ∈ FIRST(Y1) and ε ∈ FIRST(Y2), add all the non-ε symbols of FIRST(Y3), and so on. Finally, add ε to FIRST(Y1 Y2 … Yk) if ε ∈ FIRST(Yi), for all $1 \leq i \leq k$.

**Example**:

Consider the following grammar.

E → E + T | T

T → T * F | F

F → (E) | id

Grammar after removing left recursion:

E → TX

X → +TX | ε

T → FY

Y → *FY | ε

F → (E) | id

For the above grammar, following the above rules, the FIRST sets could be computed as follows:

FIRST(E) = FIRST(T) = FIRST(F) = {(, id}

FIRST(X) = {+, ε}

FIRST(Y) = {*, ε}

**Exercise:-**

**Q1. The following grammar can be used to describe traveling schemes:**

TS → TS Time Time TS | Station

Station → Identifier

Time → Nat : Nat

Which of the following grammars is equivalent but no longer left-recursive? **(Tick Mark right answer)**

a) TS → TS (Time Time Station) ∗

Station → Identifier

Time → Nat : Nat

b) TS → Station | Station Z

Z → Time Time TS | Time Time TS Z

Station → Identifier

Time → Nat : Nat

c) TS → Z Time Time TS | Station

Z → TS | #

Station → Identifier

d) TS → Station Time Time Z

Z → Station | TS

Station → Identifier

Time → Nat : Nat

Time → Nat : Nat

**Q2. Consider the grammar**

S → SX | SSb | XS | a
X → Xb | Sa | b
Eliminate left recursion and rewrite the grammar.

**Answer:-**

**Q3. Is the resulting grammar of Q2. suitable for top down parsing?**

**Answer:**